Kalman Filters for Mapping and Localization

G.D. Hager
S. Leonard

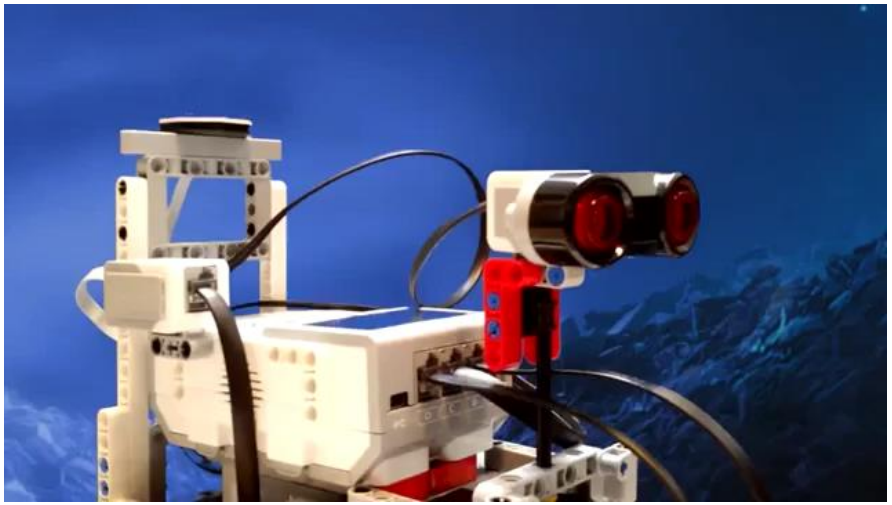
Robot Arms

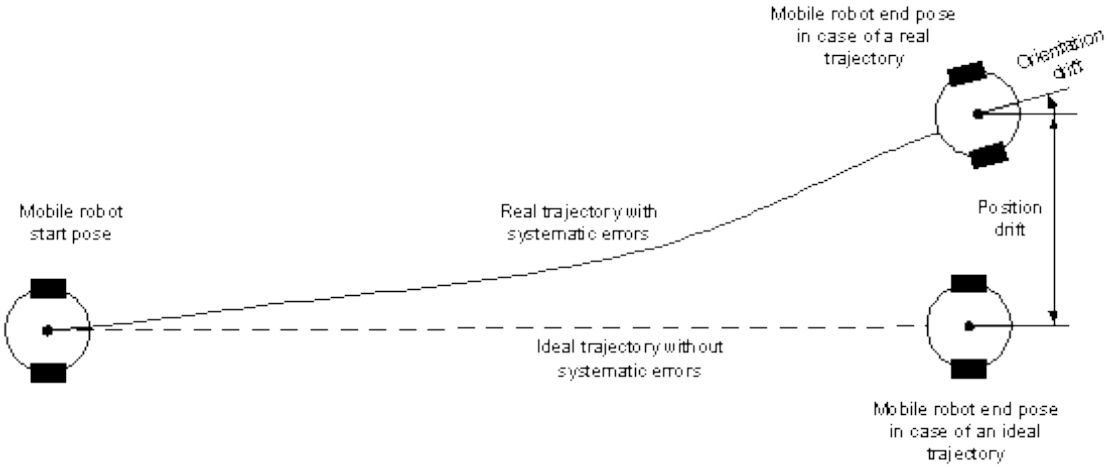
- Very accurate
  - Encoders at each joint (0.04 degree resolution)
  - Industrial grade design and assembled
  - ISO standard for accuracy and repeatability (9283)
  - Some arms are even metrology instruments

Supported robot types	Position accuracy (Typical production data)		
Robot type	Average (mm)	% within 1 mm	Maximum (mm)
IRB 140	0.35	100%	0.80
IRB1400	0.45	100%	0.95
IRB 2400L	0.45	95%	1.15
IRB 2400/10 and 16	0.30	100%	0.70
IRB 4400	0.30	100%	0.70
IRB 6600-175/2.55; 225/2.55; 200/2.75	0.55	90%	1.20
IRB 6600-175/2.80; 125/3.20	0.75	80%	1.45
IRB 6650S-200/3.00; 125/3.50			
IRB 7600-400/2.55; -150/3.50	0.65	85%	1.40
IRB 7600-500/2.30	0.55	90%	1.20
IRB 7600-340/2.80			

Mobile Robots

- Wheels and steering
  - Wheel slip
  - Wheel diameter
  - Wheel alignment
  - Steering alignment
  - Integration of velocity



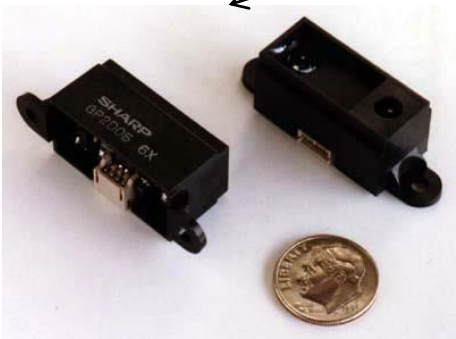


Sensors

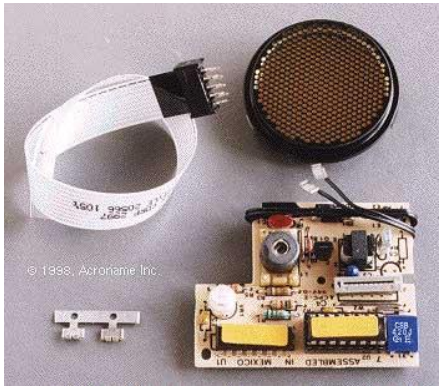
If you can't model the world, then sensors are the robot's link to the external world (obsession with depth)

Laser

Kinect



IR rangefinder



sonar rangefinder

Light-field camera

Sensors

Robots' link to the external world...

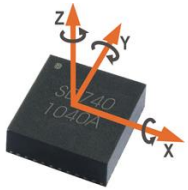
Sensors, sensors, sensors!
and tracking what is sensed: world models

Gyroscope

(orientatiation)

GPS

Force/
Torque



Inertial measurement unit
(gyro + accelerometer)

Compass

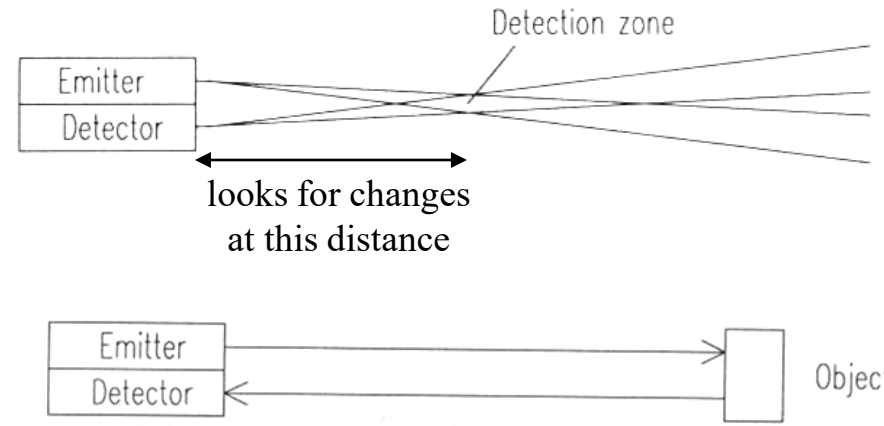
Infrared sensors

"Noncontact bump sensor"

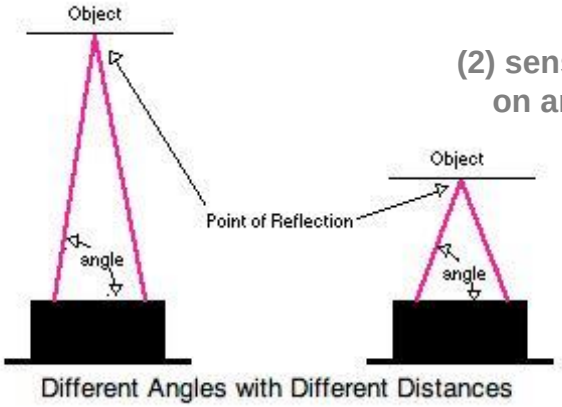
IR emitter/detector pair

(1) sensing is based on light intensity.

"object-sensing" IR



diffuse distance-sensing IR

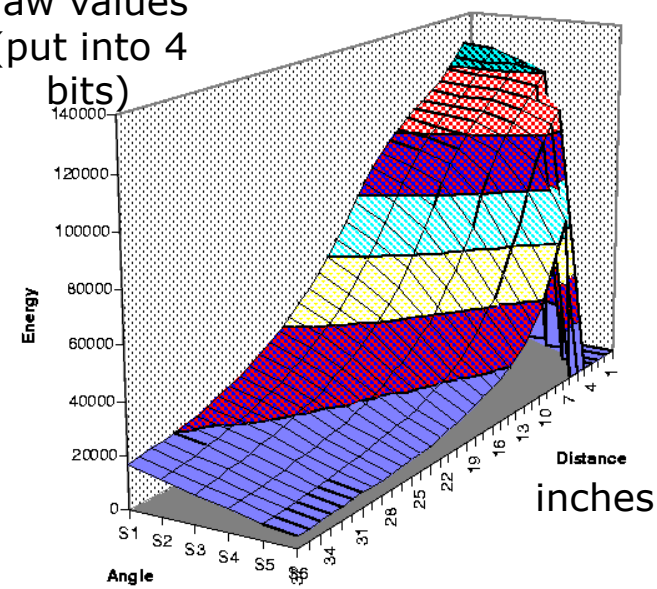


(2) sensing is based on angle received.

Infrared Calibration

The response to white copy paper (a dull, reflective surface)

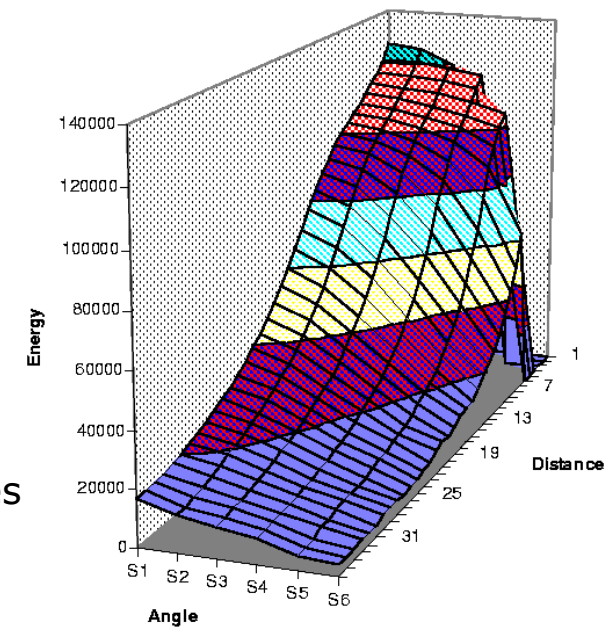
raw values (put into 4 bits)



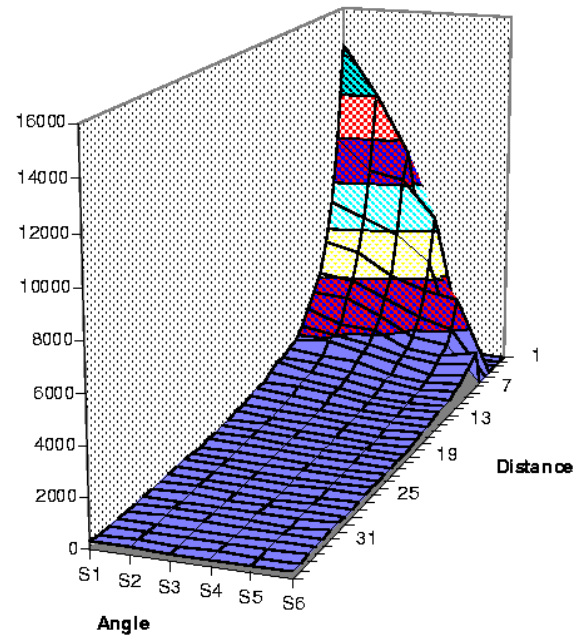
inches

15º increments

in the dark

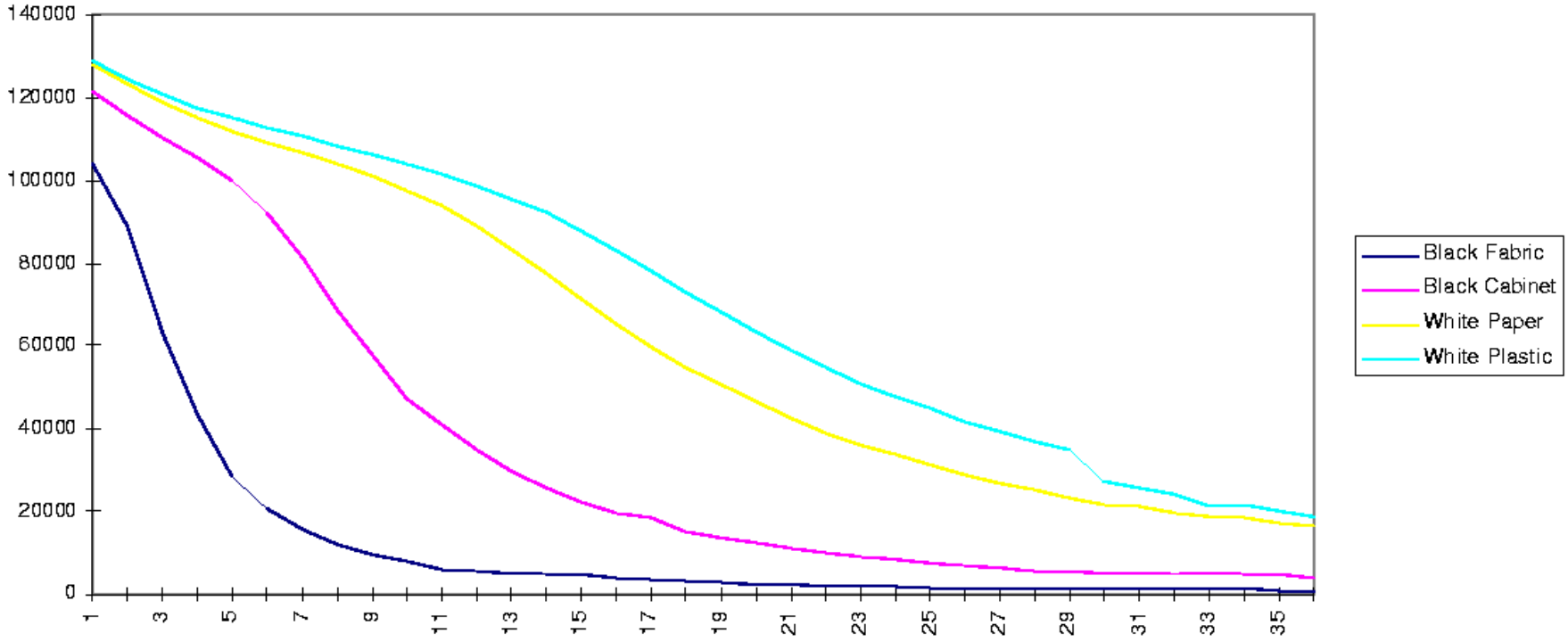


fluorescent light



incandescent light

Infrared Calibration



energy vs. distance for various materials

(the incident angle is 0º, or head-on)

(with no ambient light)

Sonar sensing

single-transducer sonar timeline

0 — 75μs — .5s

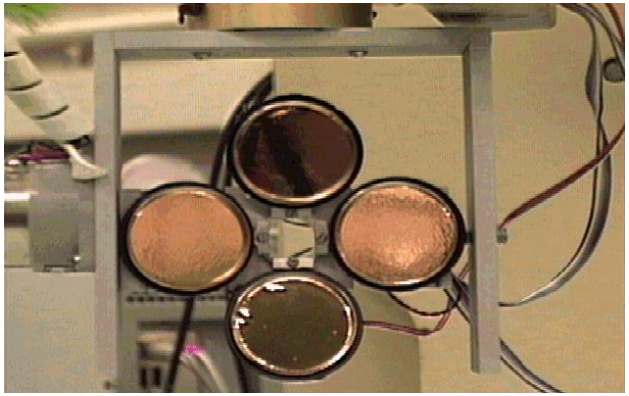
a "chirp" is emitted into the environment

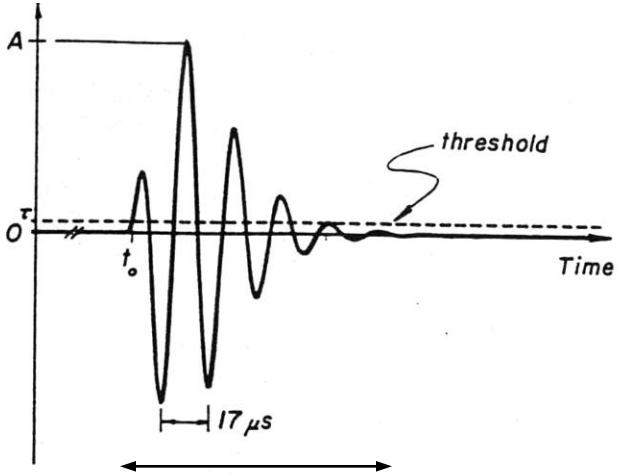
typically when reverberations from the initial chirp have stopped

the transducer goes into "receiving" mode and awaits a signal...

limiting range sensing

after a short time, the signal will be too weak to be detected



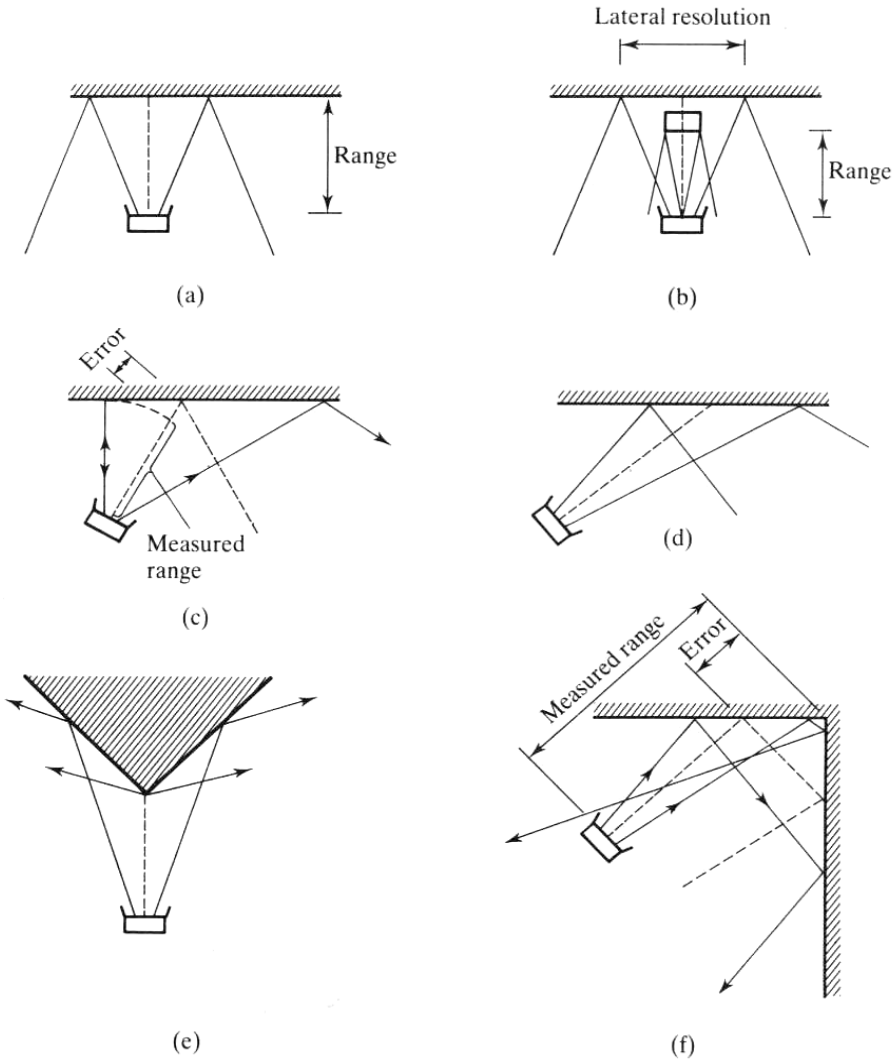


blanking time

Polaroid sonar emitter/receivers

No lower range limit for *paired* sonars...

Sonar effects



(a) Sonar providing an accurate range measurement

(b-c) Lateral resolution is not very precise; the closest object in the beam's cone provides the response

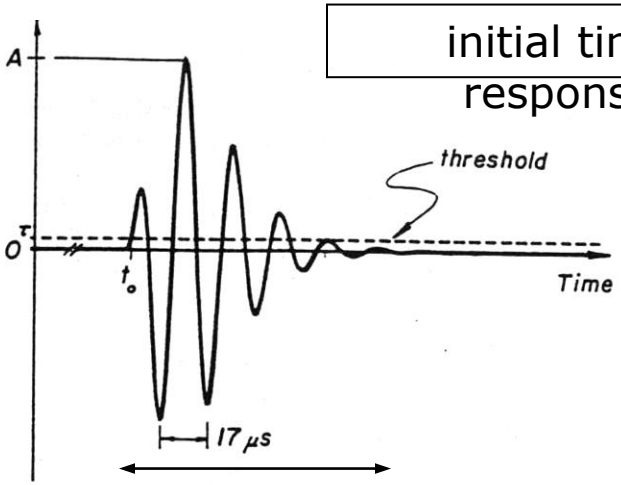
(d) Specular reflections cause walls to disappear

(e) Open corners produce a weak spherical wavefront

(f) Closed corners measure to the corner itself because of multiple reflections --> sonar ray tracing

Sonar modeling

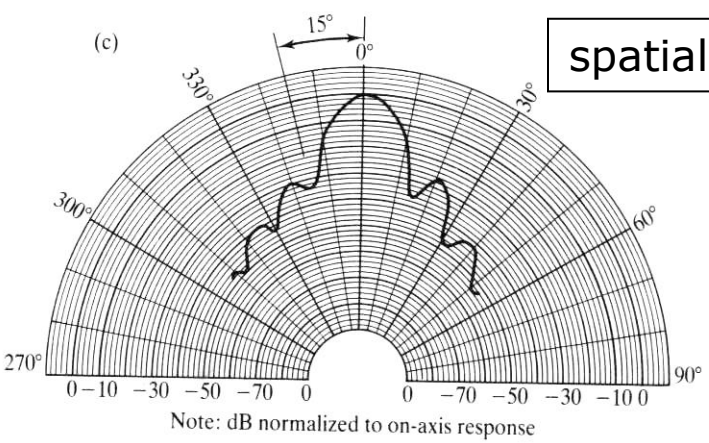
initial time response



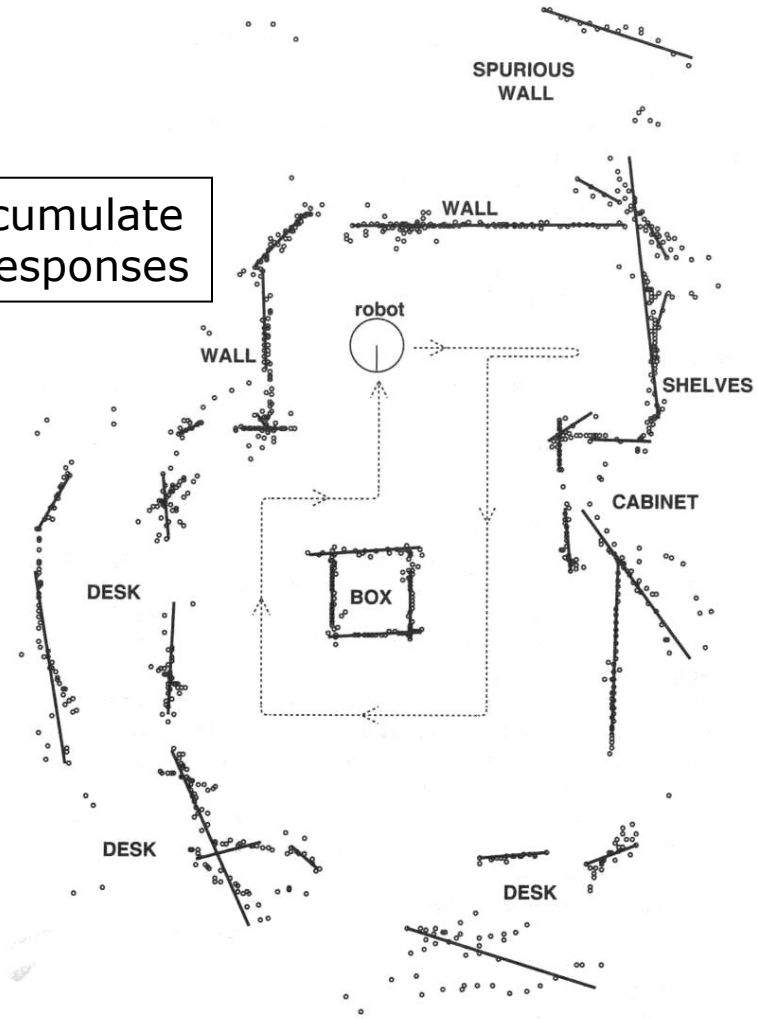
blanking time

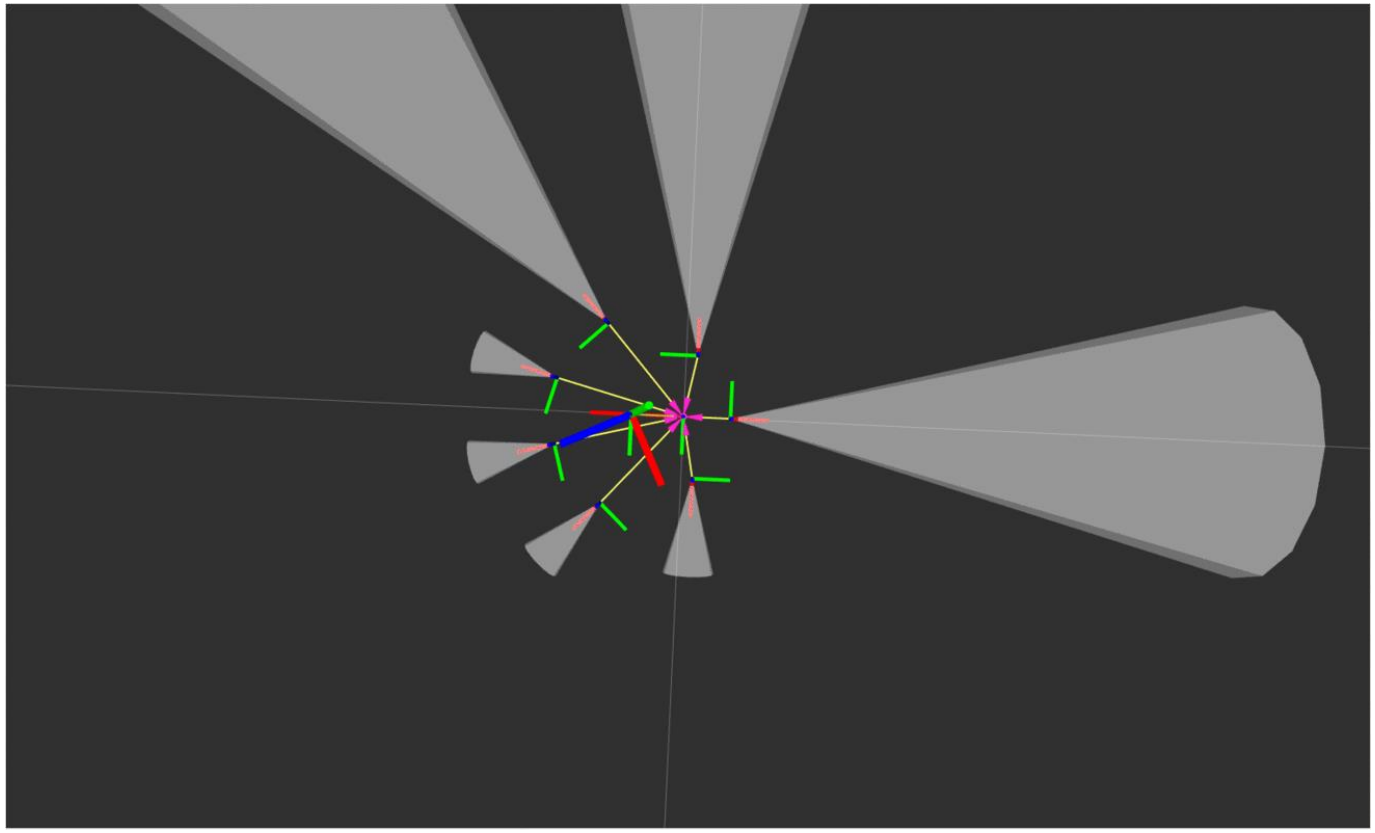
cone width

spatial response



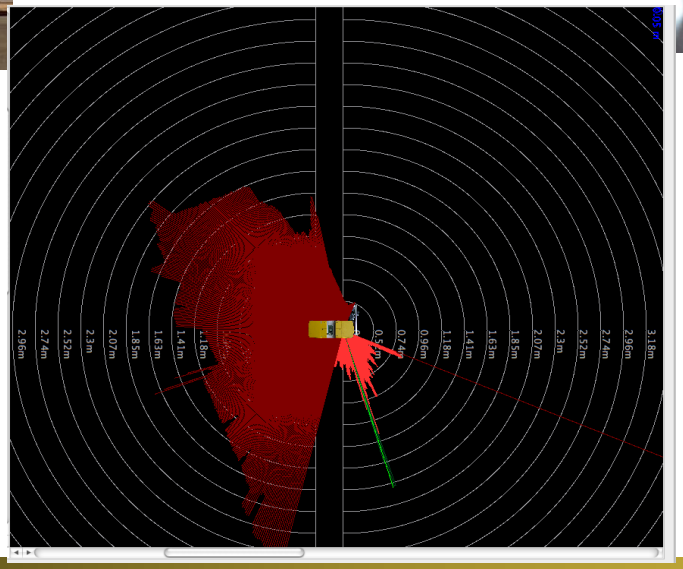
accumulated responses

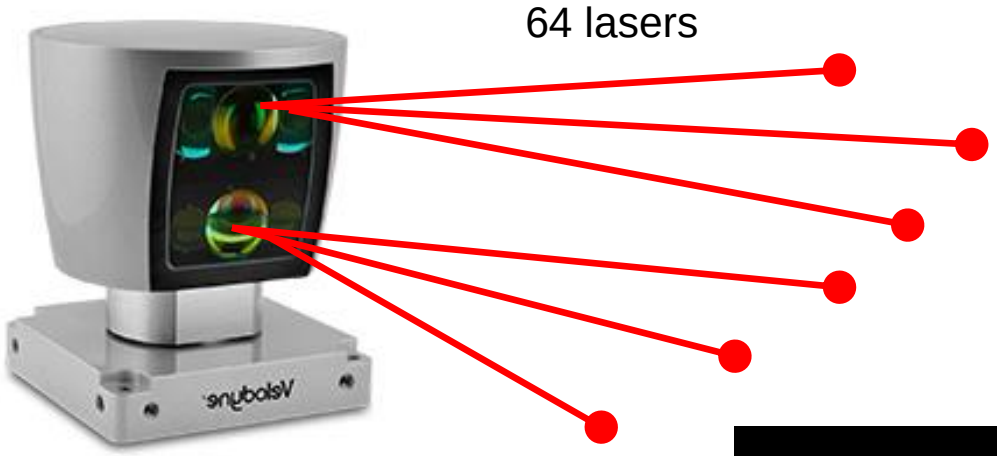


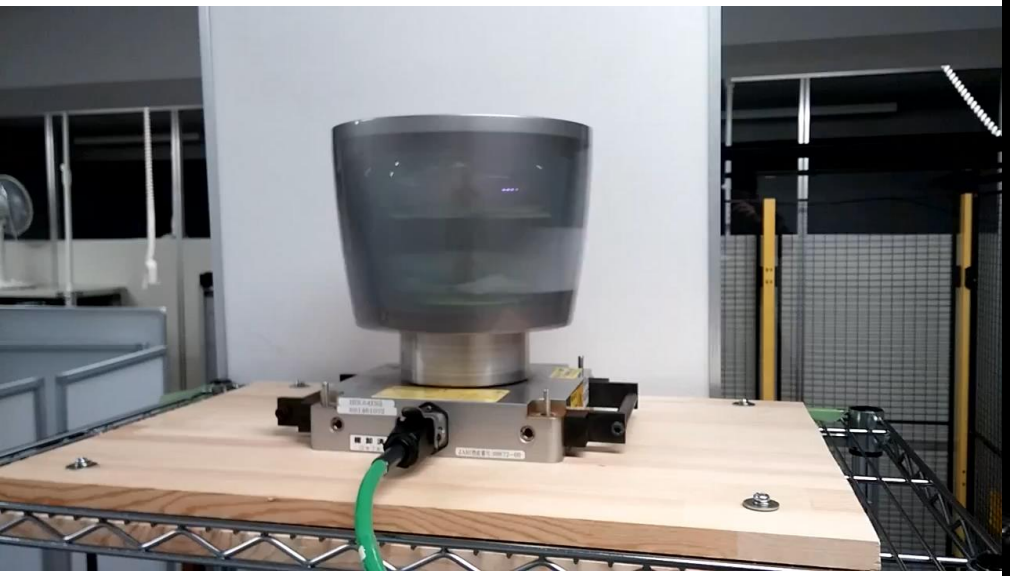


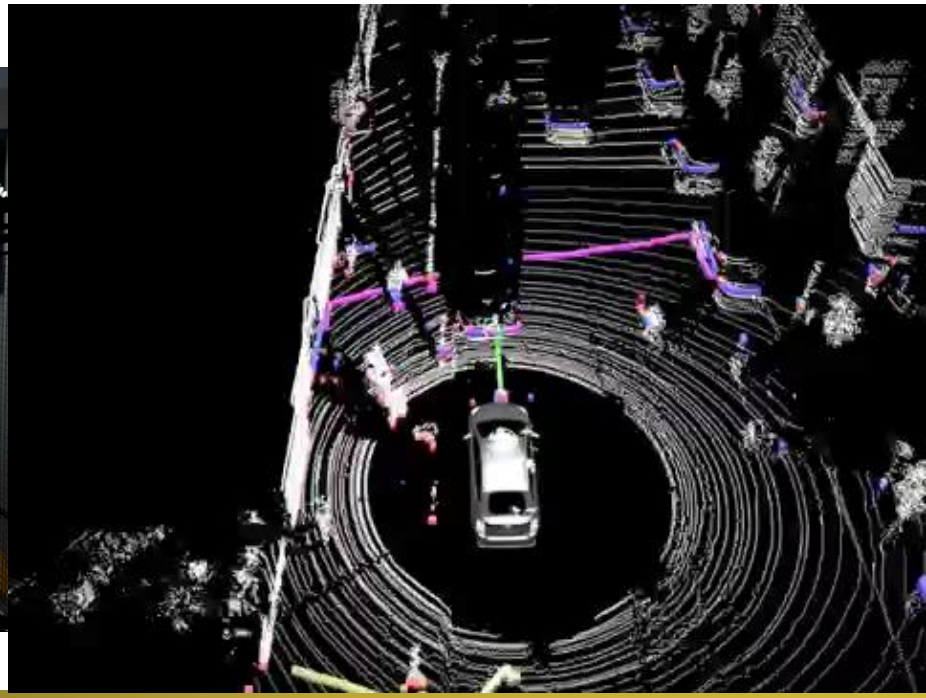
Laser Ranging

LIDAR/Laser range finder

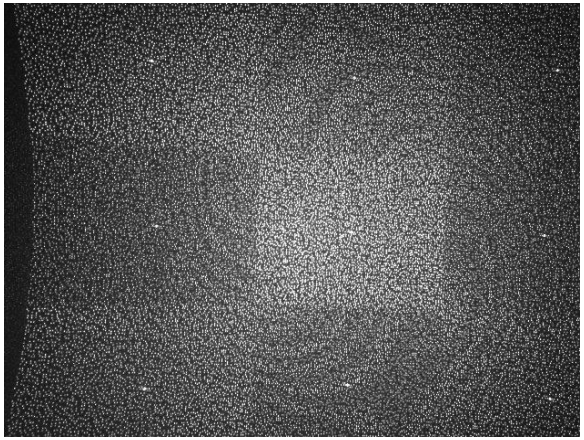



64 lasers





Recent...



Structured light:
Project a known dot pattern with an IR transmitter (invisible to humans)

Infer depth from deformation to that pattern
depth from focus: Points far away are blurry
depth from stereo: Closer points are shifted

The Problem

- Mapping: What is the world around me (geometry, landmarks)
  - sense from various positions
  - integrate measurements to produce map
  - assumes perfect knowledge of position
- Localization: Where am I in the world (position wrt landmarks)
  - sense
  - relate sensor readings to a world model
  - compute location relative to model
  - assumes a perfect world model
- Together, these are SLAM (Simultaneous Localization and Mapping)
  - How can you localize without a map?
  - How can you map without localization?
- All localization, mapping or SLAM methods are based on updating a state:
  - What makes a state? Localization? Map? Both?
  - How certain is the state?

Representations for Bayesian Robot Localization

Discrete approaches (’95)
- Topological representation (’95)
  - uncertainty handling (POMDPs)
  - occas. global localization, recovery
- Grid-based, metric representation (’96)
  - global localization, recovery

Kalman filters (late-80s?)
- Gaussians
- approximately linear models
- position tracking

Robotics

AI

Particle filters (’99)
- sample-based representation
- global localization, recovery

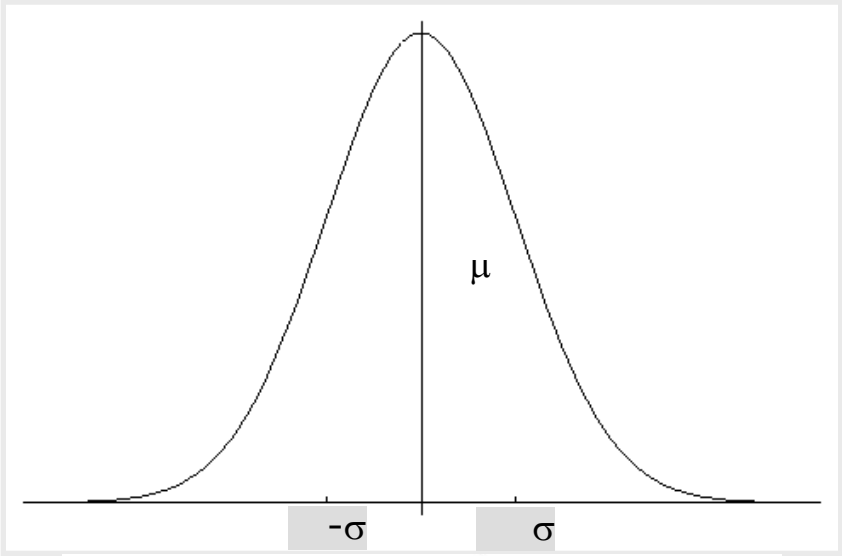
Multi-hypothesis (’00)
- multiple Kalman filters
- global localization, recovery

Gaussian (or Normal) Distribution

Univariate

$$p(x) \sim N(\mu, \sigma^2)$$

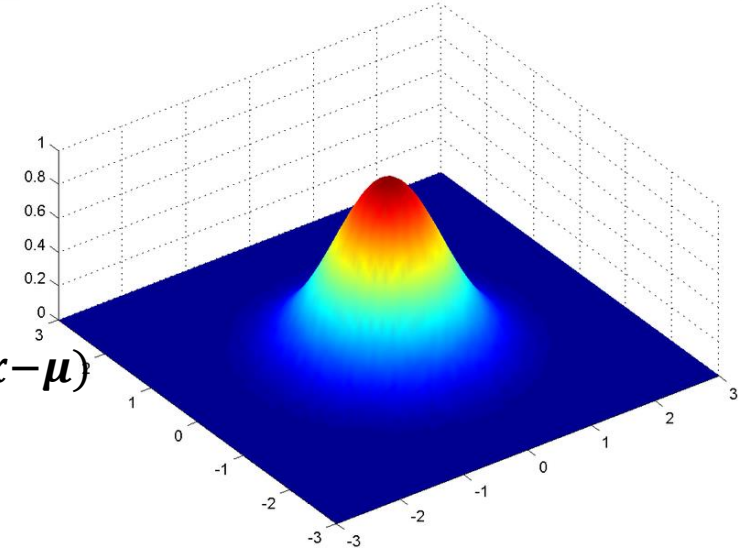
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$



Multivariate

$$p(\boldsymbol{x}) \sim N(\boldsymbol{\mu}, \Sigma)$$

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$



Properties of Gaussians

$$\left.\begin{array}{l} X \sim N(\mu, \sigma^2) \\ Y = aX + b \end{array}\right\} \Rightarrow Y \sim N(a\mu + b, a^2\sigma^2)$$

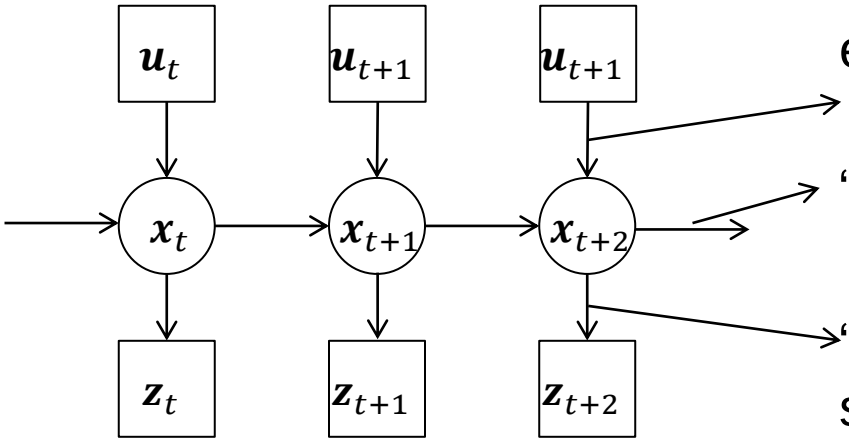
$$\left.\begin{array}{l} X_1 \sim N\left(\mu_1, \sigma_1^{\ 2}\right) \\ X_2 \sim N\left(\mu_2, \sigma_2^{\ 2}\right) \end{array}\right\} \Rightarrow X_1 + X_2 \sim N(\mu_1 + \mu_2, \sigma_1^{\ 2} + \sigma_2^{\ 2})$$

- We stay in the "Gaussian world" as long as we start with Gaussians and perform only linear transformations (closed under linear transformation)

- Same holds for multivariate Gaussians

Dynamical Systems

- "System" that changes
- We will represent a system by its state $\boldsymbol{x}_t$ at time t
- We will also require that a system be observable



external command input

"process model" (state transition)

"observation model" (measurements of the states)

Kalman Filter

- Seminal paper published in 1960
- Great web page at http://www.cs.unc.edu/~welch/kalman/
- Recursive solution for discrete linear filtering problems
  - A state $\boldsymbol{x} \in R^n$
  - A measurement $\boldsymbol{z} \in R^m$
  - Discrete (i.e. for time t = 1, 2, 3, …)
  - Recursive process (i.e. $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1})$)
  - Linear system (i.e $\boldsymbol{x}_t = A\,\boldsymbol{x}_{t-1}$)
- The system is defined by:

1) *linear* process model

$$\boldsymbol{x}_t = A\,\boldsymbol{x}_{t-1}$$

state transition

How a state transitions into another state

2) *linear* measurement model

$$\boldsymbol{z}_t = H\,\boldsymbol{x}_t$$

observation model

How a state relates to a measurement

Kalman Filter

1. *Prior estimate* $\hat{\boldsymbol{x}}_t'$ at step t : Use the process model to predict what will be the next state of the robot

2. *Posterior estimate* $\hat{\boldsymbol{x}}_t$ at step t : Use the observation model to correct the prediction by using sensor measurement

Compute *posterior* estimate as a linear combination of the *prior* estimate and difference between the actual measurement and expected measurement

$$\hat{x}_t = \hat{x}_t' + K_t(z_t - H\hat{x}_t')$$

Where the Kalman gain K_t is a blending factor that adds a measurement innovation.

Kalman Filter

Define ***prior error*** between true state and *prior* estimate

$$\boldsymbol{e}'_t = \boldsymbol{x}_t - \hat{\boldsymbol{x}}'_t$$

and the ***prior covariance*** as

$$\Sigma'_t = E(\boldsymbol{e}'_t \boldsymbol{e}'^T_t)$$

Define ***posterior error*** between true state and *posterior* estimate

$$\boldsymbol{e}_t = \boldsymbol{x}_t - \hat{\boldsymbol{x}}_t$$

and it's ***posterior covariance*** as

$$\Sigma_t = E(\boldsymbol{e}_t \boldsymbol{e}^T_t)$$

The gain K that ***minimizes*** the ***posterior covariance*** is defined by

$$K_t = \Sigma'_t H^T (H \Sigma'_t H^T + R)^{-1}$$

Note that

- If $R \to 0$ then $K_t = H^{-1}$ and $\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}'_t + K_t(\boldsymbol{z}_t - H\hat{\boldsymbol{x}}'_t) =$
- If $\Sigma'_t \to 0$ then $K_t = 0$ and $\hat{\boldsymbol{x}}_t =$

Kalman Filter

- Recipe:
  1. Start with an initial guess

$$\hat{\boldsymbol{x}}_0, \Sigma_0$$

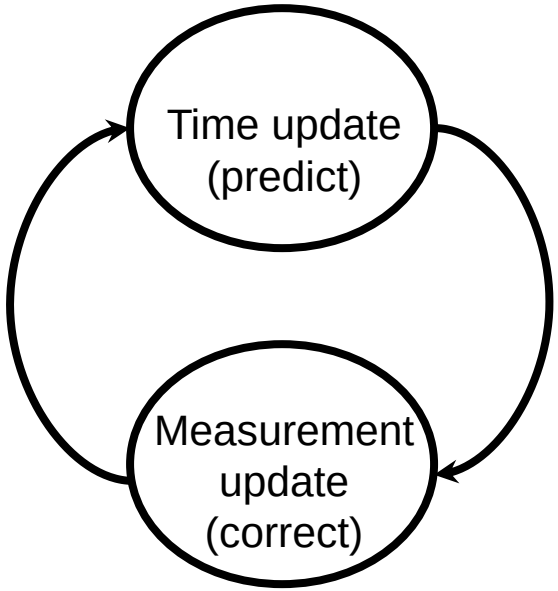
  2. Compute prior (prediction)

$$\hat{\boldsymbol{x}}_t' = A\hat{\boldsymbol{x}}_{t-1} + B\boldsymbol{u}_{t-1}$$
$$\Sigma_t' = A\Sigma_{t-1}A^T + Q$$

  3. Compute posterior (correction)

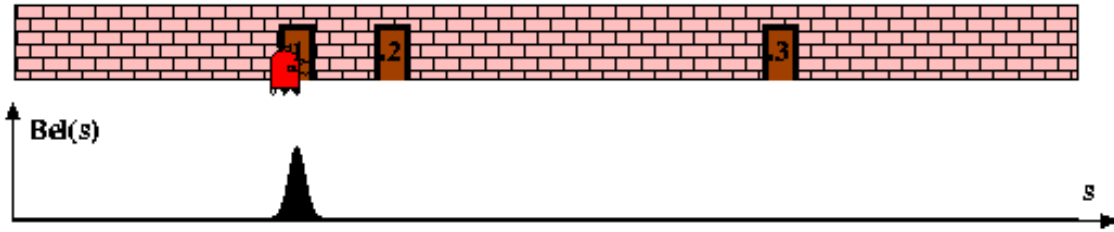
$$K_t = \Sigma_t' H^T (H\Sigma_t' H^T + R)^{-1}$$
$$\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}_t' + K_t(\boldsymbol{z}_t - H\hat{\boldsymbol{x}}_t')$$
$$\Sigma_t = (1 - K_t H)\Sigma_t'$$

REPEAT



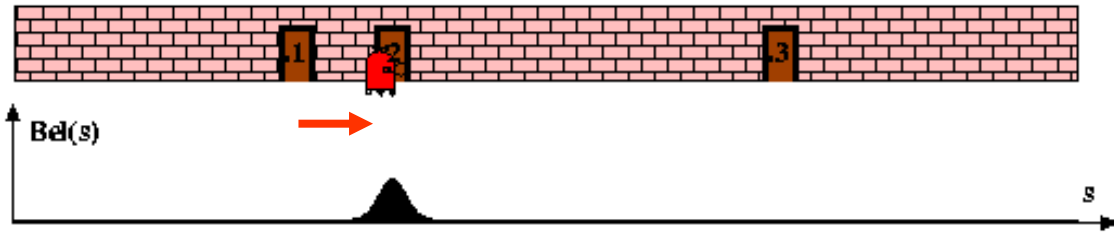
Initial

$$\hat{\boldsymbol{x}}_0, \Sigma_0$$



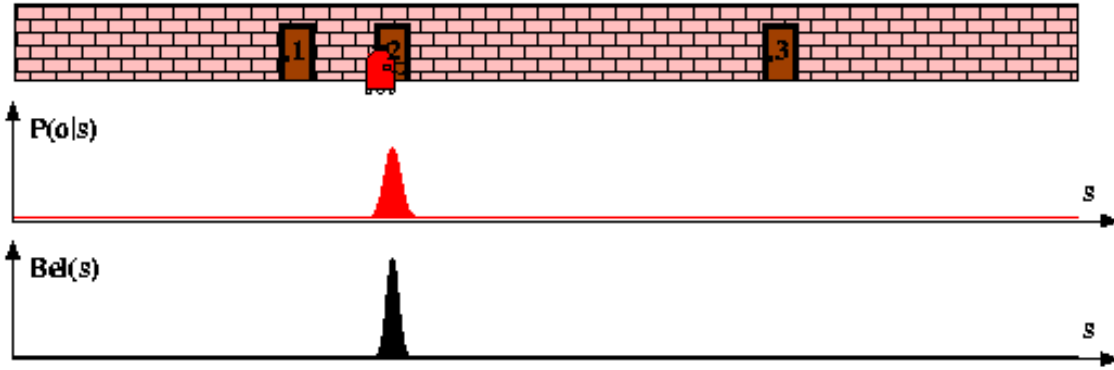
Predict $\hat{\boldsymbol{x}}_1'$ from $\hat{\boldsymbol{x}}_0$ and $\boldsymbol{u}_0$

$$\hat{\boldsymbol{x}}_1' = A\hat{\boldsymbol{x}}_0 + B\boldsymbol{u}_0$$
$$\Sigma_1' = A\Sigma_0 A^T + Q$$



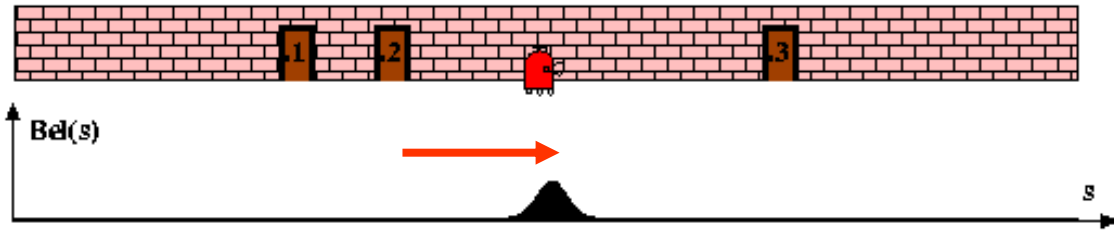
Correction using $\boldsymbol{z}_1$

$$K_1 = \Sigma_1' H^T (H\Sigma_1' H^T + R)^{-1}$$
$$\hat{\boldsymbol{x}}_1 = \hat{\boldsymbol{x}}_1' + K_1(\boldsymbol{z}_1 - H\hat{\boldsymbol{x}}_1')$$
$$\Sigma_1 = (1 - K_1 H)\Sigma_1'$$

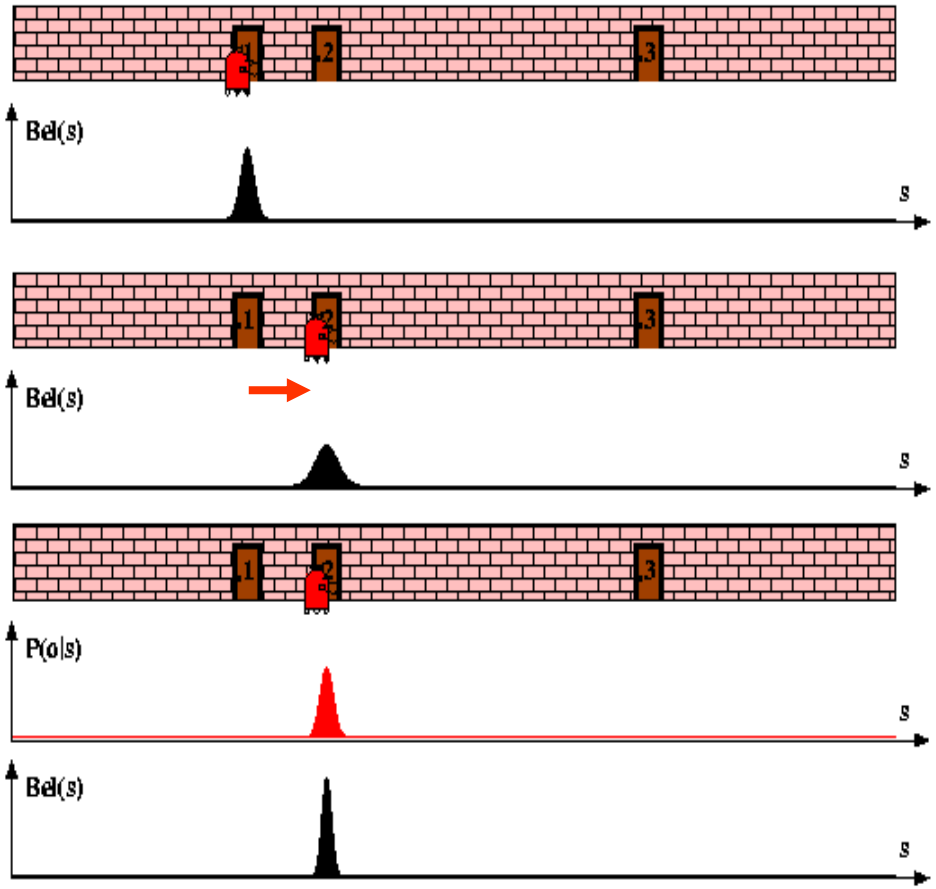


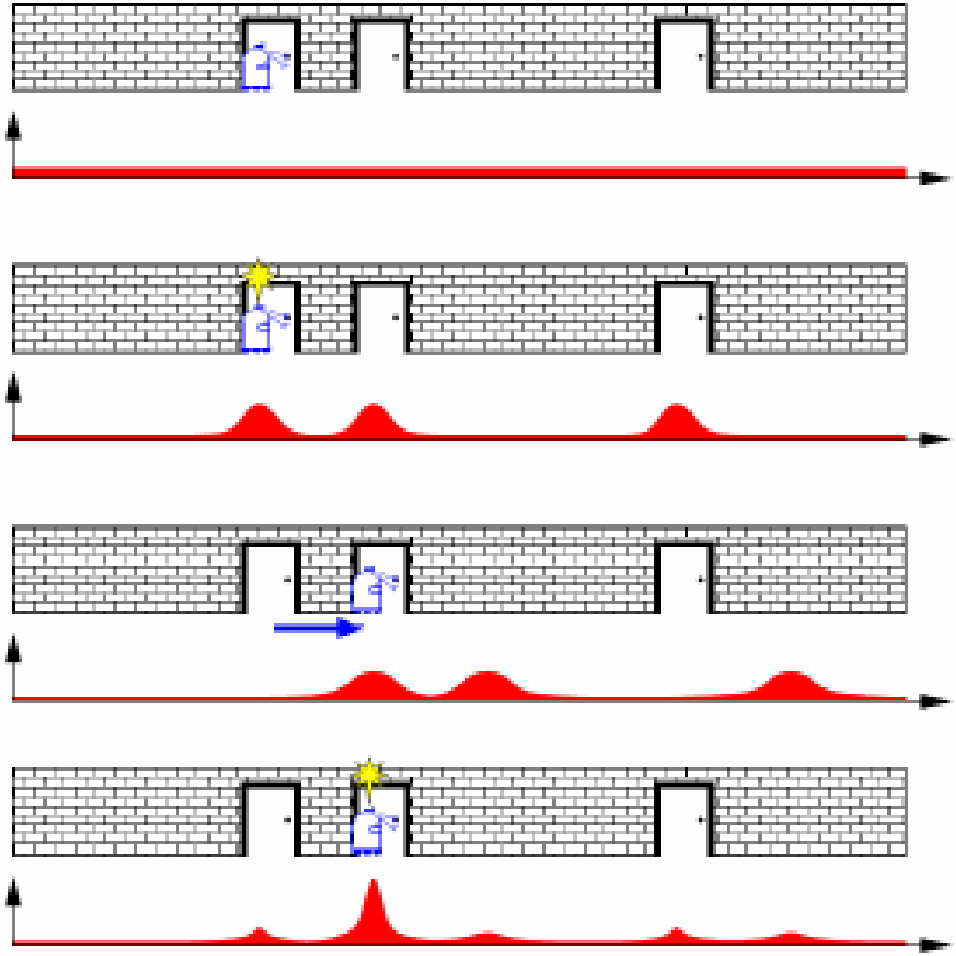
Predict $\hat{\boldsymbol{x}}_2'$ from $\hat{\boldsymbol{x}}_1$ and $\boldsymbol{u}_1$

$$\hat{\boldsymbol{x}}_2' = A\hat{\boldsymbol{x}}_1 + B\boldsymbol{u}_1$$
$$\Sigma_2' = A\Sigma_1 A^T + Q$$



Fully Observable vs Partially Observable





Kalman Filter Limitations

- Assumptions:
  - Linear process model $\boldsymbol{x}_{t+1} = A\boldsymbol{x}_t + B\boldsymbol{u}_t + \boldsymbol{w}_t$
  - Linear observation model $\boldsymbol{z}_t = H\boldsymbol{x}_t + \boldsymbol{v}_t$
  - White Gaussian noise $N(0, \Sigma)$

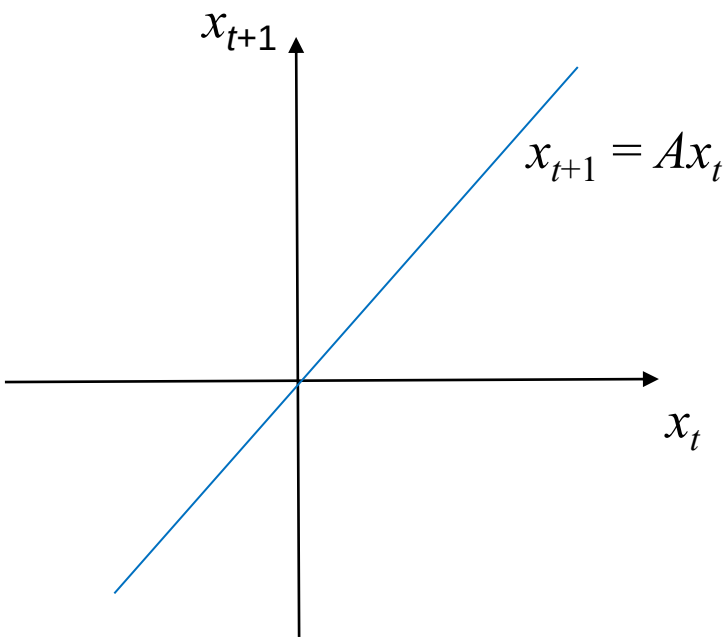
- What can we do if system is not linear?
  - Non-linear state dynamics $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{w}_t)$
  - Non-linear observations $\boldsymbol{z}_t = h(\boldsymbol{x}_t, \boldsymbol{v}_t)$

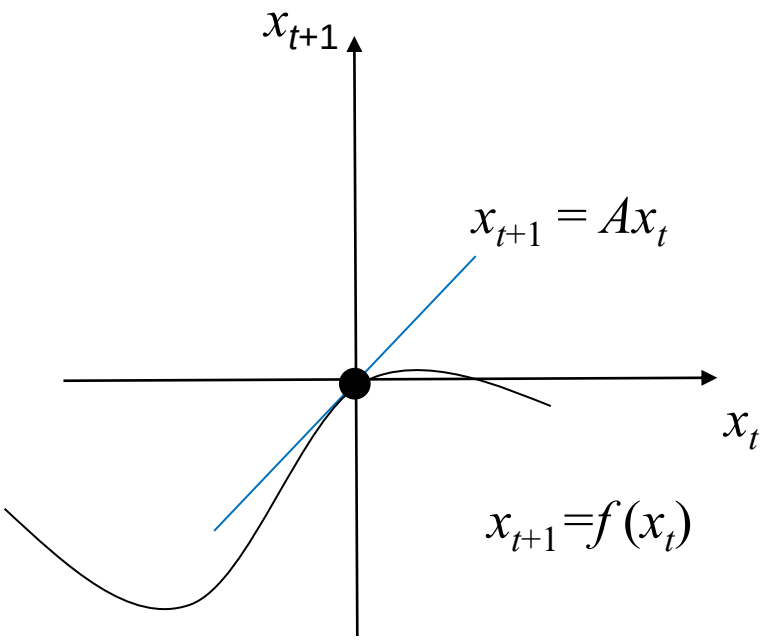
Linearize it!

$$\boldsymbol{x}_{t+1} \approx \tilde{\boldsymbol{x}}_{t+1} + A(\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t) + W\boldsymbol{w}_t \qquad \boldsymbol{z}_t \approx \tilde{\boldsymbol{z}}_t + H(\boldsymbol{x}_t - \tilde{\boldsymbol{x}}_t) + V\boldsymbol{v}_t$$

$$\tilde{\boldsymbol{x}}_{t+1} = f(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t, 0) \qquad \tilde{\boldsymbol{z}}_t = h(\tilde{\boldsymbol{x}}_t, 0)$$

KF vs EKF





Extended Kalman Filter

- Where A, H, W and V are Jacobians defined by

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},w) = \begin{bmatrix} f_1(\boldsymbol{x},\boldsymbol{u},w) \\ \vdots \\ f_M(\boldsymbol{x},\boldsymbol{u},w) \end{bmatrix} \qquad \boldsymbol{h}(\boldsymbol{x},\boldsymbol{v}) = \begin{bmatrix} h_1(\boldsymbol{x},\boldsymbol{v}) \\ \vdots \\ h_N(\boldsymbol{x},\boldsymbol{v}) \end{bmatrix}$$

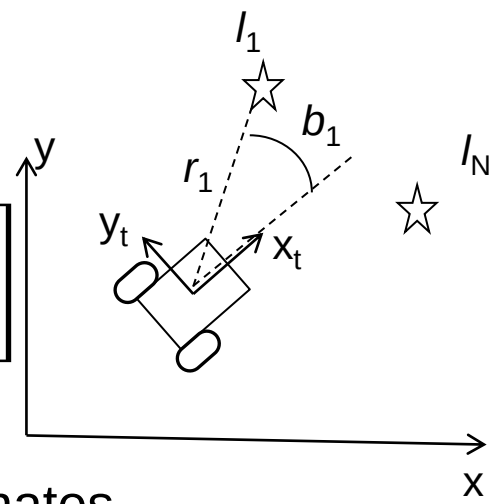
$$A(\boldsymbol{x}_t) = \begin{bmatrix} \frac{\partial f_1(,\boldsymbol{x}_t\boldsymbol{u}_t,0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial f_1(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{x}_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_M(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial f_M(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{x}_M} \end{bmatrix} \quad W(\boldsymbol{x}_t) = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{w}_1} & \cdots & \frac{\partial f_1(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{w}_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_M(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{w}_1} & \cdots & \frac{\partial f_M(\boldsymbol{x}_t,\boldsymbol{u}_t,0)}{\partial \boldsymbol{w}_N} \end{bmatrix}$$

$$H(\boldsymbol{x}_t) = \begin{bmatrix} \frac{\partial h_1(\boldsymbol{x}_t,0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial h_1(\boldsymbol{x}_t,0)}{\partial \boldsymbol{x}_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_N(\boldsymbol{x}_t,0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial h_N(\boldsymbol{x}_t,0)}{\partial \boldsymbol{x}_M} \end{bmatrix} \quad V(\boldsymbol{x}_t) = \begin{bmatrix} \frac{\partial h_1(\boldsymbol{x}_t,0)}{\partial \boldsymbol{v}_1} & \cdots & \frac{\partial h_1(\boldsymbol{x}_t,0)}{\partial \boldsymbol{v}_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_M(\boldsymbol{x}_t,0)}{\partial \boldsymbol{v}_1} & \cdots & \frac{\partial h_M(\boldsymbol{x}_t,0)}{\partial \boldsymbol{v}_N} \end{bmatrix}$$

EKF for Range-Bearing Localization

- State $\boldsymbol{s}_t = [x_t \quad y_t \quad \theta_t]^T$ 2D position and orientation
- Input $\boldsymbol{u}_t = [v_t \quad \omega_t]^T$ linear and angular velocity
- Process model

$$\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t, \boldsymbol{w}_t) = \begin{bmatrix} x_{t-1} + (\Delta t) v_{t-1} \cos(\theta_{t-1}) \\ y_{t-1} + (\Delta t) v_{t-1} \sin(\theta_{t-1}) \\ \theta_{t-1} + (\Delta t)\omega_{t-1} \end{bmatrix} + \begin{bmatrix} w_{x_t} \\ w_y \\ w_{\theta_t} \end{bmatrix}$$



- Given a map, the robot sees N landmarks with coordinates

$$\boldsymbol{l}_1 = [x_{l_1} \quad y_{l_1}]^T, \cdots, \boldsymbol{l}_N = [x_{l_N} \quad y_{l_N}]^T$$

The observation model is

$$\boldsymbol{z}_t = \begin{bmatrix} \boldsymbol{h}_1(\boldsymbol{s}_t, \boldsymbol{v}_1) \\ \vdots \\ \boldsymbol{h}_N(\boldsymbol{s}_t, \boldsymbol{v}_N) \end{bmatrix} \quad \boldsymbol{h}_i(\boldsymbol{s}_t, \boldsymbol{v}_t) = \begin{bmatrix} \sqrt{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2} \\ \operatorname{atan2}(y_{l_i} - y_t, x_{l_i} - x_t) - \theta_t \end{bmatrix} + \begin{bmatrix} v_r \\ v_b \end{bmatrix}$$

Linearize Process Model

$$\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t, \boldsymbol{w}_t) = \begin{bmatrix} x_{t-1} + (\Delta t) v_{t-1} \cos(\theta_{t-1}) \\ y_{t-1} + (\Delta t) v_{t-1} \sin(\theta_{t-1}) \\ \theta_{t-1} + (\Delta t) \omega_{t-1} \end{bmatrix} + \begin{bmatrix} w_{x_t} \\ w_y \\ w_{\theta_t} \end{bmatrix}$$

$$A(\boldsymbol{x}_t) = \begin{bmatrix} \frac{\partial f_1(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t, 0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial f_1(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t, 0)}{\partial \boldsymbol{x}_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_M(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t, 0)}{\partial \boldsymbol{x}_1} & \cdots & \frac{\partial f_M(\hat{\boldsymbol{x}}_t, \boldsymbol{u}_t, 0)}{\partial \boldsymbol{x}_N} \end{bmatrix}$$

$$A(\boldsymbol{s}_{t-1}) = \begin{bmatrix} 1 & 0 & -\Delta t \; v_{t-1} \sin(\theta_{t-1}) \\ 0 & 1 & \Delta t \; v_{t-1} \cos(\theta_{t-1}) \\ 0 & 0 & 1 \end{bmatrix}$$

Linearize Observation Model

$$\boldsymbol{z}_t = \begin{bmatrix} \boldsymbol{h}_1(\boldsymbol{s}_t, \boldsymbol{v}_1) \\ \vdots \\ \boldsymbol{h}_N(\boldsymbol{s}_t, \boldsymbol{v}_N) \end{bmatrix} \qquad \boldsymbol{h}_i(\boldsymbol{s}_t, \boldsymbol{v}_t) = \begin{bmatrix} \sqrt{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2} \\ \operatorname{atan2}(y_{l_i} - y_t, x_{l_i} - x_t) - \theta_t \end{bmatrix} + \begin{bmatrix} v_r \\ v_b \end{bmatrix}$$

$$H(\boldsymbol{s}_t) = \begin{bmatrix} \frac{\partial \boldsymbol{h}_1(\boldsymbol{s}_t, 0)}{\partial x_t} & \frac{\partial \boldsymbol{h}_1(\boldsymbol{s}_t, 0)}{\partial y_t} & \frac{\partial \boldsymbol{h}_1(\boldsymbol{s}_t, 0)}{\partial \theta_t} \\ \vdots & \vdots & \vdots \\ \frac{\partial \boldsymbol{h}_N(\boldsymbol{s}_t, 0)}{\partial x_t} & \frac{\partial \boldsymbol{h}_N(\boldsymbol{s}_t, 0)}{\partial y_t} & \frac{\partial \boldsymbol{h}_N(\boldsymbol{s}_t, 0)}{\partial \theta_t} \end{bmatrix}$$

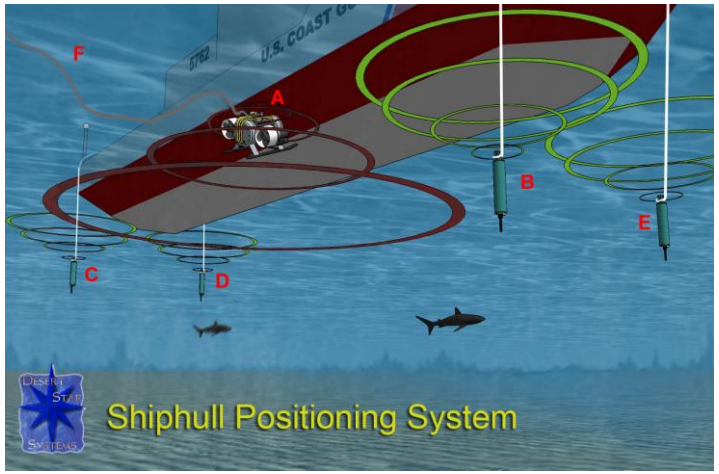
$$H_i(\boldsymbol{s}_t) = \begin{bmatrix} \frac{-x_{li} + x_t}{\sqrt{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2}} & \frac{-y_{li} + y_t}{\sqrt{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2}} & 0 \\ \frac{y_{li} - y_t}{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2} & \frac{-x_{li} + x_t}{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2} & -1 \end{bmatrix}$$

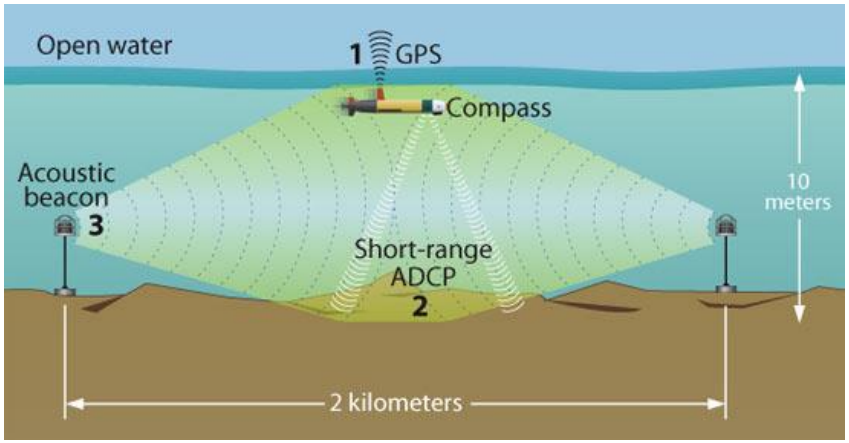
Extended Kalman Filter

- Kalman Filter Recipe:
  - Given
    $\hat{x}_0, \Sigma_0$
  - Prediction
    $$\hat{\boldsymbol{x}}'_t = A\hat{\boldsymbol{x}}_{t-1} + B\boldsymbol{u}_{t-1}$$
    $$\Sigma'_t = A\Sigma_{t-1}A^T + Q$$
  - Measurement correction
    $$K_t = \Sigma'_t H^T (H\Sigma'_t H^T + R)^{-1}$$
    $$\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}'_t + K(\boldsymbol{z}_t - H\hat{\boldsymbol{x}}'_t)$$
    $$\Sigma_t = (I - K_t H)\Sigma'_t$$

- Extended Kalman Filter Recipe:
  - Given
    $\hat{x}_0, \Sigma_0$
  - Prediction
    $$\hat{\boldsymbol{x}}'_t = \boldsymbol{f}(\hat{\boldsymbol{x}}_{t-1}, \boldsymbol{u}_{t-1}, 0)$$
    $$\Sigma'_t = A_t\Sigma_{t-1}A_t^T + W_t M_t W_t^T$$
  - Measurement correction
    $$K_t = \Sigma'_t H_t^T (H_t\Sigma'_t H_t^T + V_t R V_t^T)^{-1}$$
    $$\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}'_t + K_t(\boldsymbol{z}_t - \boldsymbol{h}(\hat{\boldsymbol{x}}'_t, 0))$$
    $$\Sigma_t = (I - K_t H_t)\Sigma'_t$$

Artificial Landmarks

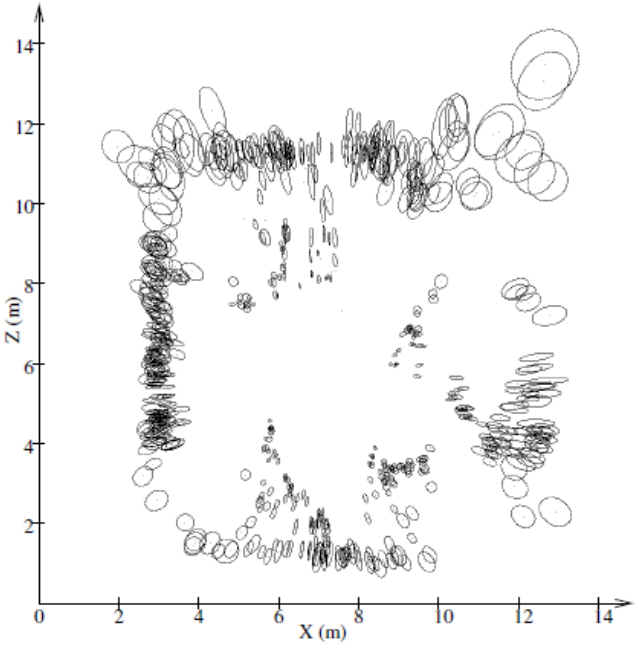




Lippsett

Natural Visual Landmarks

- Visual landmarks: SIFT, SURF, corners,



Se IJRR

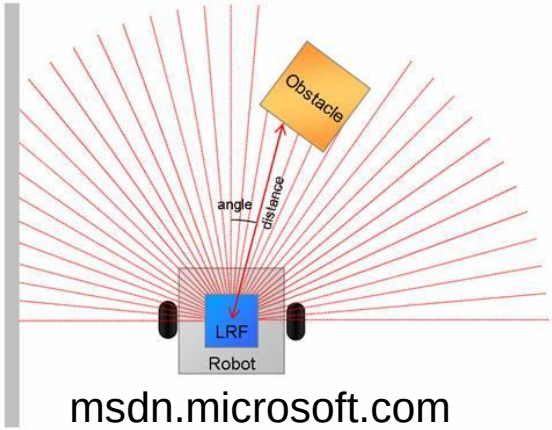
Data Association

- From observation model, we have an expected

$$\boldsymbol{z}_t = \begin{bmatrix} \boldsymbol{h}_1(\boldsymbol{s}_t, \boldsymbol{v}_1) \\ \vdots \\ \boldsymbol{h}_N(\boldsymbol{s}_t, \boldsymbol{v}_N) \end{bmatrix}$$

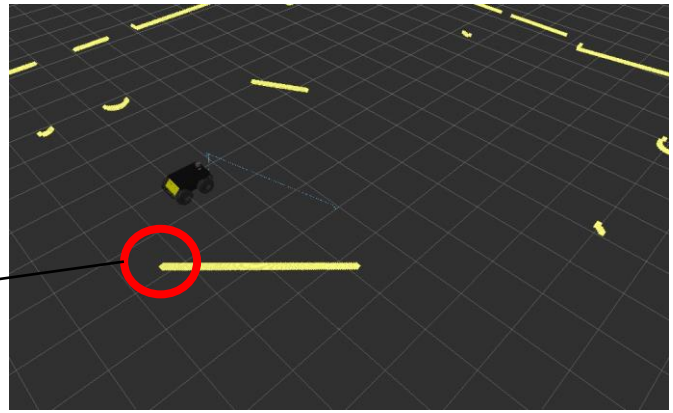
Observation of landmark #1

Observation of landmark #N



msdn.microsoft.com

- So if we have N landmarks $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_N$ and we are given a scan $\boldsymbol{z}_t$, how do associate each landmark to a scan observation?



$$\begin{bmatrix} \vdots \\ \hat{\boldsymbol{l}}_{i_t} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ \hat{\boldsymbol{l}}'_{i_t} \\ \vdots \end{bmatrix} + K_t \left(\begin{bmatrix} \vdots \\ \boldsymbol{z}_{i_t} \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ \boldsymbol{h}_i(\hat{\boldsymbol{x}}'_t, 0) \\ \vdots \end{bmatrix} \right)$$

Observation of landmark #i must be in the ith row!

From Localization to Mapping

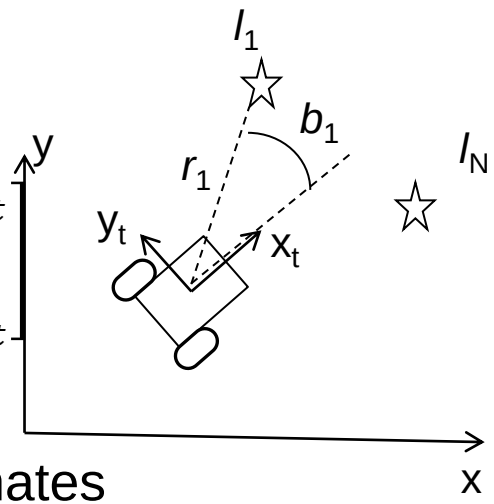
- For us, the landmarks have been a known quantity (we have a map with the coordinates of the landmarks), but landmarks are not part of the state
- Two choices:
  - Make the state the location of the landmarks relative to the robot (I also know exactly where I am …)
    - - No notion of location relative to past history
    - - No fixed reference for landmarks
  - Make the state the robot location now (relative to where we started) plus landmark locations
    - + Landmarks now have fixed location
    - - Knowledge of my location slowly degrades (but this is inevitable …)

iRobot

EKF for Range-Bearing Localization

- State $\boldsymbol{s}_t = [x_t \quad y_t \quad \theta_t]^T$ 2D position and orientation
- Input $\boldsymbol{u}_t = [v_t \quad \omega_t]^T$ linear and angular velocity
- Process model

$$\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t, \boldsymbol{w}_t) = \begin{bmatrix} x_{t-1} + (\Delta t) v_{t-1} \cos(\theta_{t-1}) \\ y_{t-1} + (\Delta t) v_{t-1} \sin(\theta_{t-1}) \\ \theta_{t-1} + (\Delta t) \omega_{t-1} \end{bmatrix} + \begin{bmatrix} w_{x_t} \\ w_y \\ w_{\theta_t} \end{bmatrix}$$



- Given a map, the robot sees N landmarks with coordinates

$$\boldsymbol{l}_1 = [x_{l_1} \quad y_{l_1}]^T, \cdots, \boldsymbol{l}_N = [x_{l_N} \quad y_{l_N}]^T$$

The observation model is

$$\boldsymbol{z}_t = \begin{bmatrix} \boldsymbol{h}_1(\boldsymbol{s}_t, \boldsymbol{v}_1) \\ \vdots \\ \boldsymbol{h}_N(\boldsymbol{s}_t, \boldsymbol{v}_N) \end{bmatrix} \quad \boldsymbol{h}_i(\boldsymbol{s}_t, \boldsymbol{v}_t) = \begin{bmatrix} \sqrt{\left(x_t - x_{l_i}\right)^2 + \left(y_t - y_{l_i}\right)^2} \\ \operatorname{atan2}(y_{l_i} - y_t, x_{l_i} - x_t) - \theta_t \end{bmatrix} + \begin{bmatrix} v_r \\ v_b \end{bmatrix}$$

Kalman Filters and SLAM

- Localization: state is the location of the robot
- Mapping: state is the location of 2D landmarks
- SLAM: state combines both
- If the state is $\boldsymbol{s}_t = \begin{bmatrix} x_t & y_t & \theta_t & l_{1_t}^T & \cdots & l_{N_t}^T \end{bmatrix}^T$
  then we can write a linear observation system

EKF Range Bearing SLAM
Prior State Estimation

- State $\boldsymbol{s}_t = [x_t \quad y_t \quad \theta_t \quad \boldsymbol{l}_1^T \quad \cdots \quad \boldsymbol{l}_N^T]^T$ position/orientation of robot and landmarks coordinates
- Input $\boldsymbol{u}_t = [v_t \quad \omega_t]^T$ forward and angular velocity
- The process model for localization is

$$\boldsymbol{s}_t' = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{bmatrix} + \begin{bmatrix} \Delta t v_{t-1} \cos(\theta_{t-1}) \\ \Delta t v_{t-1} \sin(\theta_{t-1}) \\ \Delta t \omega_{t-1} \end{bmatrix}$$

This model is augmented for 2N+3 dimensions to accommodate landmarks. This results in the process equation

$$\begin{bmatrix} x_t' \\ y_t' \\ \theta_t' \\ \boldsymbol{l}_{1_t}' \\ \vdots \\ \boldsymbol{l}_{N_t}' \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \\ \boldsymbol{l}_{1_{t-1}} \\ \vdots \\ \boldsymbol{l}_{N_{t-1}} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta t v_{t-1} \cos(\theta_{t-1}) \\ \Delta t v_{t-1} \sin(\theta_{t-1}) \\ \Delta t \omega_{t-1} \end{bmatrix}$$

Landmarks don't depend on external input

EKF Range Bearing SLAM
Prior Covariance Update

We assume static landmarks. Therefore, the function **f**(***s***,***u***,***w***) only affects the robot's location and not the landmarks.

$$\Sigma_t' = A_t \Sigma_{t-1} A_t^T + W_t Q W_t^T$$

Jacobian of the robot motion

Jacobian of the process model

$$A = \begin{bmatrix} \frac{\partial x}{\partial x} & \frac{\partial x}{\partial y} & \frac{\partial x}{\partial \theta} & \\ \frac{\partial y}{\partial x} & \frac{\partial y}{\partial y} & \frac{\partial y}{\partial \theta} & 0 \\ \frac{\partial \theta}{\partial x} & \frac{\partial \theta}{\partial y} & \frac{\partial \theta}{\partial \theta} & \\ & 0 & & I \end{bmatrix}$$

The motion of the robot does not affect the coordinates of the landmarks

2Nx2N identity

EKF Range Bearing SLAM Kalman Gain

$$K_t = \Sigma_t' H_t^T (H_t \Sigma_t' H_t^T + V_t R V_t^T)^{-1}$$

$$\boldsymbol{h}_i(\boldsymbol{s}_t, \boldsymbol{v}_t) = \begin{bmatrix} \sqrt{(x_t - x_{l_i})^2 + (y_t - y_{l_i})^2} \\ \operatorname{atan2}(y_{l_i} - y_t, x_{l_i} - x_t) - \theta_t \end{bmatrix} + \begin{bmatrix} v_r \\ v_b \end{bmatrix}$$

Compute the Jacobian H_i of each $\boldsymbol{h}_i$ and then stack them into one big matrix H. Note that $\boldsymbol{h}_i$ only depends on 5 variables: x_t, y_t, θ_t, x_{li}, y_{li}

Need to be in the correct columns of H

$$H_i = \begin{bmatrix} \frac{\partial y_i}{\partial x_r} \\ \frac{\partial y_i}{\partial y_r} \end{bmatrix} =$$

$$\begin{bmatrix} \frac{-x_{\ell i}(k)+x_r(k)}{\rho_i} & \frac{-y_{\ell i}(k)+y_r(k)}{\rho_i} & 0 & \cdots & \frac{x_{\ell i}(k)-x_r(k)}{\rho_i} & \frac{y_{\ell i}(k)-y_r(k)}{\rho_i} & \cdots \\ \frac{y_{\ell i}(k)-y_r(k)}{\rho_i^2} & \frac{-x_{\ell i}(k)+x_r(k)}{\rho_i^2} & -1 & \cdots & \frac{-y_{\ell i}(k)+y_r(k)}{\rho_i^2} & \frac{x_{\ell i}(k)-x_r(k)}{\rho_i^2} & \cdots \end{bmatrix}$$

ρ_i is the range of the landmark

EKF Range Bearing SLAM Measurement

$$\boldsymbol{z}_t = \begin{bmatrix} \boldsymbol{h}_1(\boldsymbol{s}_t, \boldsymbol{v}_1) \\ \vdots \\ \boldsymbol{h}_N(\boldsymbol{s}_t, \boldsymbol{v}_N) \end{bmatrix} \Longleftrightarrow \quad \overbrace{= \begin{bmatrix} \sqrt{(x_t - x_{l_i})^2 + (y_t - y_{l_i})^2} \\ \operatorname{atan2}(y_{l_i} - y_t, x_{l_i} - x_t) - \theta_t \end{bmatrix}}^{\boldsymbol{h}_i(\boldsymbol{s}_t, \boldsymbol{v}_t)} + \begin{bmatrix} v_r \\ v_b \end{bmatrix}$$

- Observe N landmarks $\boldsymbol{z}_{i_t} = [r_{i_t} \quad \phi_{i_t}]^T$
- Must have data association
  - Which measured landmark corresponds to $\boldsymbol{h}_i$?
  - If $\boldsymbol{s}_t$ contains the coordinates of *N* landmarks in the map, $\boldsymbol{h}_i$ predicts the measurement of each landmark
- Must figure which measured landmark corresponds to $\boldsymbol{h}_i$

$$\hat{\boldsymbol{s}}_t = \hat{\boldsymbol{s}}'_t + K_t\big(\boldsymbol{z}_t - \boldsymbol{h}(\hat{\boldsymbol{s}}'_t, 0)\big)$$

Make sure that each landmark observation $\boldsymbol{z}_{i_t}$ appears in the correct rows of $\boldsymbol{z}_t$

$$\begin{bmatrix} \vdots \\ \hat{\boldsymbol{l}}_{i_t} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ \hat{\boldsymbol{l}}'_{i_t} \\ \vdots \end{bmatrix} + K_t\left(\begin{bmatrix} \vdots \\ \boldsymbol{z}_{i_t} \\ \vdots \end{bmatrix} - \begin{bmatrix} \vdots \\ \boldsymbol{h}_i(\hat{\boldsymbol{s}}'_t, 0) \\ \vdots \end{bmatrix}\right)$$

EKF Range Bearing SLAM Posterior Update

- From K_t and H update the posterior state estimate

$$\hat{\boldsymbol{s}}_t = \hat{\boldsymbol{s}}'_t + K_t\big(\boldsymbol{z}_t - \boldsymbol{h}(\hat{\boldsymbol{s}}'_t, 0)\big)$$
$$\Sigma_t = (I - K_t H_t)\Sigma'_t$$

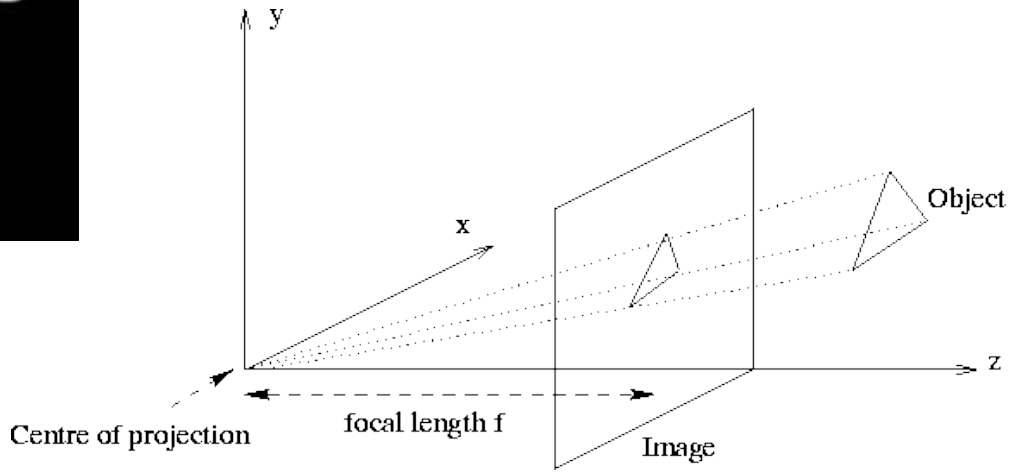
$$\underbrace{\begin{pmatrix}
\sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xm_{1,x}} & \sigma_{xm_{1,y}} & \cdots & \sigma_{xm_{n,x}} & \sigma_{xm_{n,y}} \\
\sigma_{yx} & \sigma_{yy} & \sigma_{y\theta} & \sigma_{ym_{1,x}} & \sigma_{ym_{1,y}} & \cdots & \sigma_{m_{n,x}} & \sigma_{m_{n,y}} \\
\sigma_{\theta x} & \sigma_{\theta y} & \sigma_{\theta\theta} & \sigma_{\theta m_{1,x}} & \sigma_{\theta m_{1,y}} & \cdots & \sigma_{\theta m_{n,x}} & \sigma_{\theta m_{n,y}} \\
\sigma_{m_{1,x}x} & \sigma_{m_{1,x}y} & \sigma_{\theta} & \sigma_{m_{1,x}m_{1,x}} & \sigma_{m_{1,x}m_{1,y}} & \cdots & \sigma_{m_{1,x}m_{n,x}} & \sigma_{m_{1,x}m_{n,y}} \\
\sigma_{m_{1,y}x} & \sigma_{m_{1,y}y} & \sigma_{\theta} & \sigma_{m_{1,y}m_{1,x}} & \sigma_{m_{1,y}m_{1,y}} & \cdots & \sigma_{m_{1,y}m_{n,x}} & \sigma_{m_{1,y}m_{n,y}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\sigma_{m_{n,x}x} & \sigma_{m_{n,x}y} & \sigma_{\theta} & \sigma_{m_{n,x}m_{1,x}} & \sigma_{m_{n,x}m_{1,y}} & \cdots & \sigma_{m_{n,x}m_{n,x}} & \sigma_{m_{n,x}m_{n,y}} \\
\sigma_{m_{n,y}x} & \sigma_{m_{n,y}y} & \sigma_{\theta} & \sigma_{m_{n,y}m_{1,x}} & \sigma_{m_{n,y}m_{1,y}} & \cdots & \sigma_{m_{n,y}m_{n,x}} & \sigma_{m_{n,y}m_{n,y}}
\end{pmatrix}}_{\Sigma}$$

http://ais.informatik.uni-freiburg.de

Mono SLAM

Robot Vision, Imperial College

- A visual landmark with a single camera does not provide range
- Data association is given by tracking or matching visual descriptors/patches



Submaps

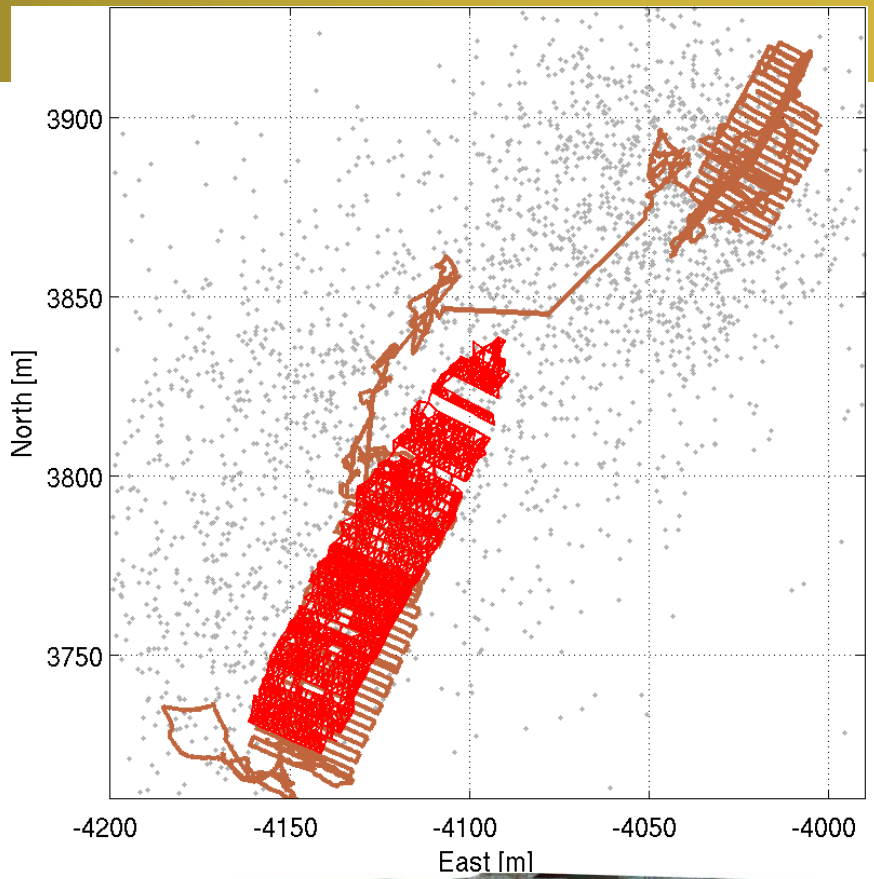
- As landmarks are added, the state vector grows
- For large maps only a few landmarks can be visible at any given time
- Use submaps to reduce the number of landmarks to manageable numbers

Navigation: RMS Titanic

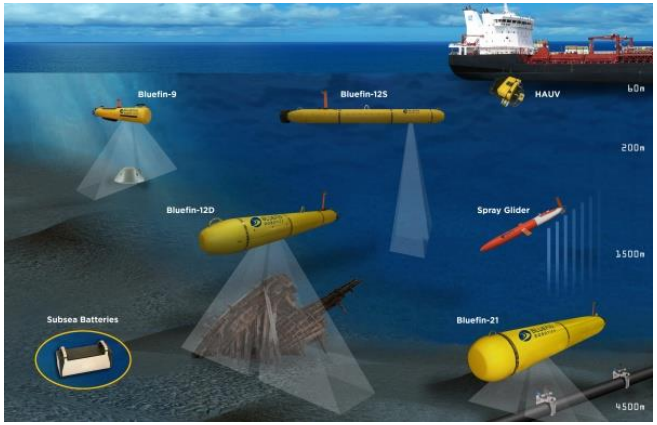
Leonard & Eustice

- EKF-based system
- 866 images
- 3494 camera constraints
- Path length 3.1km 2D / 3.4km 3D
- Convex hull > 3100m^2
- 344 min. data / 39 min. ESDF*

*excludes image registration time



Search of Flight 370



Inertial Navigation System (INS):
A glorified IMU with Kalman filter

Summary

- Basic system modeling ideas
- Kalman filter as an estimation method from a system model
- Linearization as a way of attacking a wider variety of problems
- Mapping localization and mapping into EKF
- Extensions for managing landmark matching and not-well-constrained systems.