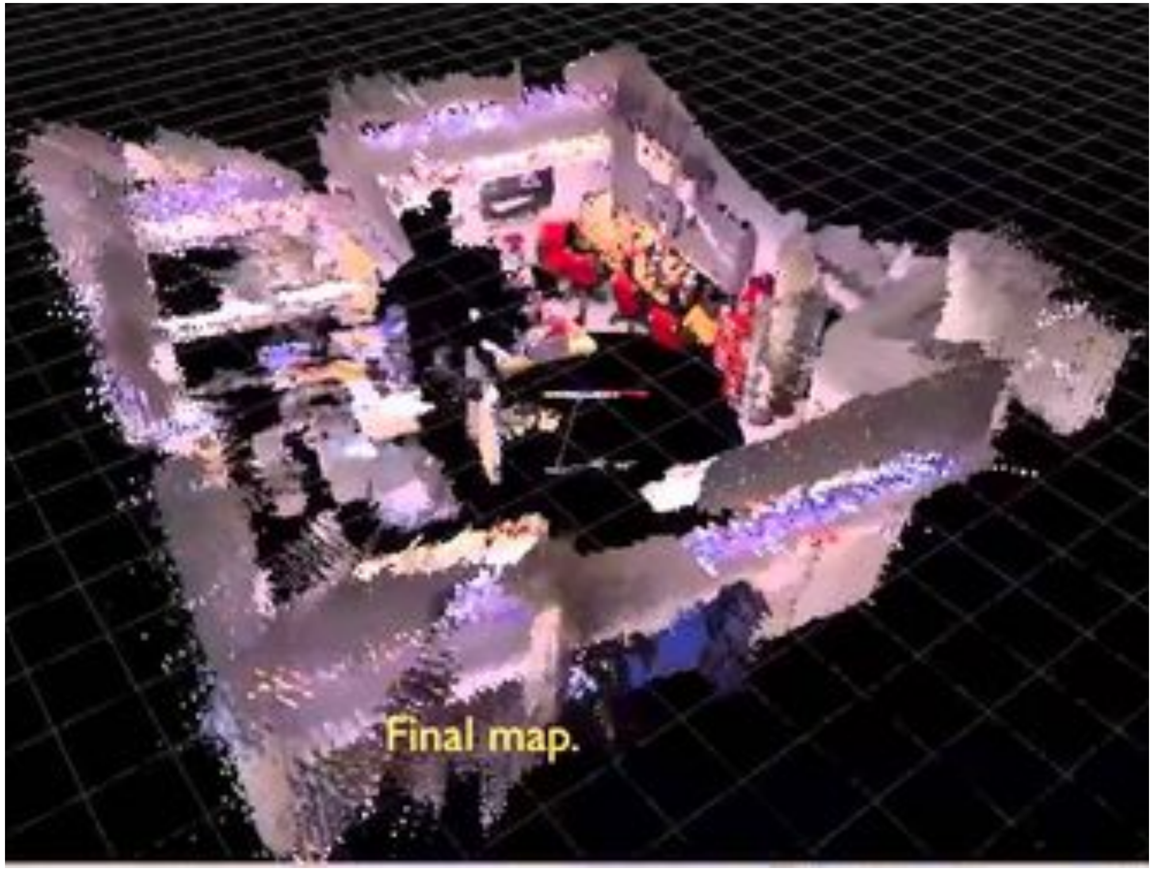
2.160 Identification, Estimation, and Learning

Part 2 Estimation

Lecture 11

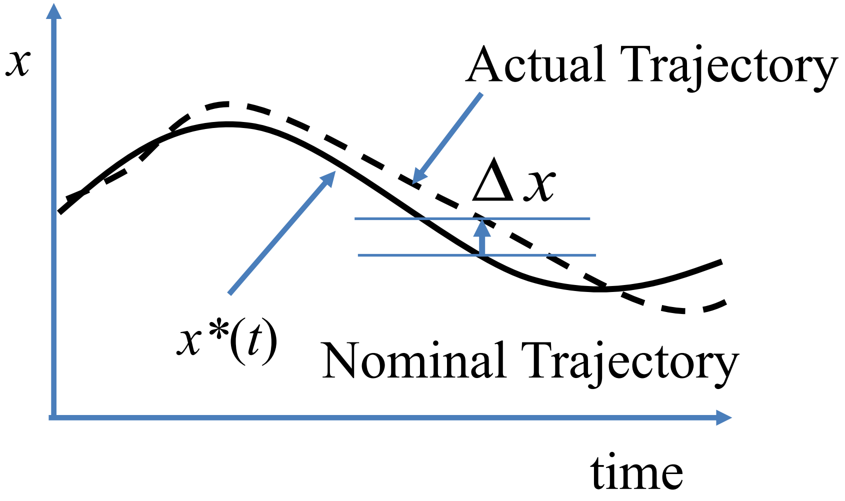
Simultaneous Localization And Mapping (SLAM)



H. Harry Asada
Department of Mechanical Engineering
MIT

Clarification of Linearized KF and Extended KF

- Linearize the nonlinear state equation around a nominal trajectory, which is a prescribed time function. The resultant system is a Linear Time-Varying system to which Kalman Filter is applicable.



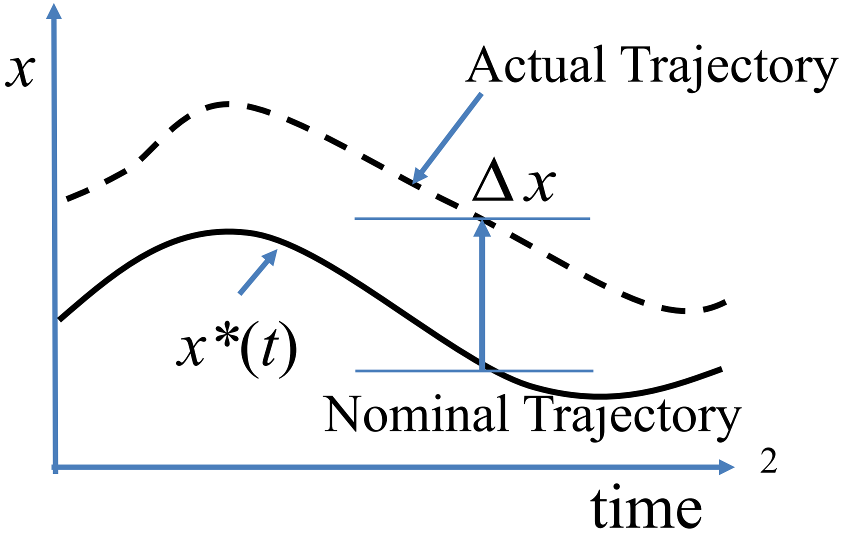
- The nominal trajectory must satisfy the original nonlinear state equation.

$$\dot{x}^* = f(x^*,u,t) \qquad x = x^* + \Delta x$$

- Consider deviation from the nominal trajectory:

$$\Delta\dot{x} \cong \left.\frac{\partial f}{\partial x}\right|_{x^*} \Delta x + w(t)$$

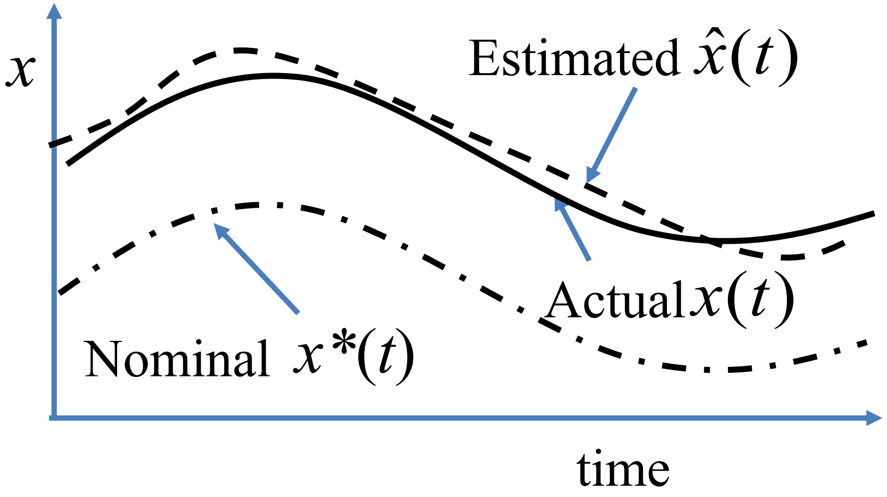
- As the actual trajectory significantly deviates from the nominal trajectory, the linearized model becomes erroneous, leading to possible divergence of estimation.



7.3 Extended Kalman Filter

❑ Extended Kalman Filter is a significant improvement in two major aspects:

1) The Jacobian matrices are evaluated not at nominal state $x^*(t)$ but at an estimated state $\hat{x}(t)$



$$F(t) = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}(t)} \qquad \cancel{F(t) = \left.\frac{\partial f}{\partial x}\right|_{x^*}}$$

$$H(t) = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}(t)} \qquad \cancel{H(t) = \left.\frac{\partial h}{\partial x}\right|_{x^*}}$$

2) State propagation and update use the full nonlinear state equation and measurement equation.

$$\dot{\hat{x}} = f(\hat{x}(t),t) + K(t)[y(t) - h(\hat{x}(t),t)] \qquad \cancel{\dot{\hat{x}} = F(t)\hat{x}(t) + K(t)[y(t) - H(t)\hat{x}(t)]}$$

Original Nonlinear equations

$$K(t) = P(t)H^T(t)R^{-1}(t)$$

❑ Estimated state $\hat{x}(t)$ and predicted output $\hat{y}(t)$ are in the original coordinate system, not the deviation from a reference.

Unscented Kalman Filter

- ❑ No linearization and Jacobians are involved.
- ❑ Propagation and update are all in the original coordinate system.

- ❑ Propagate the Sigma points through the state equation, noting that the process noise is zero mean.

$$\tilde{x}^{i*}_{t|t-1} = f(\tilde{x}^{i}_{t-1}, t-1) + \cancelto{0}{w_{t-1}}, \quad i = 0, \cdots, 2n$$

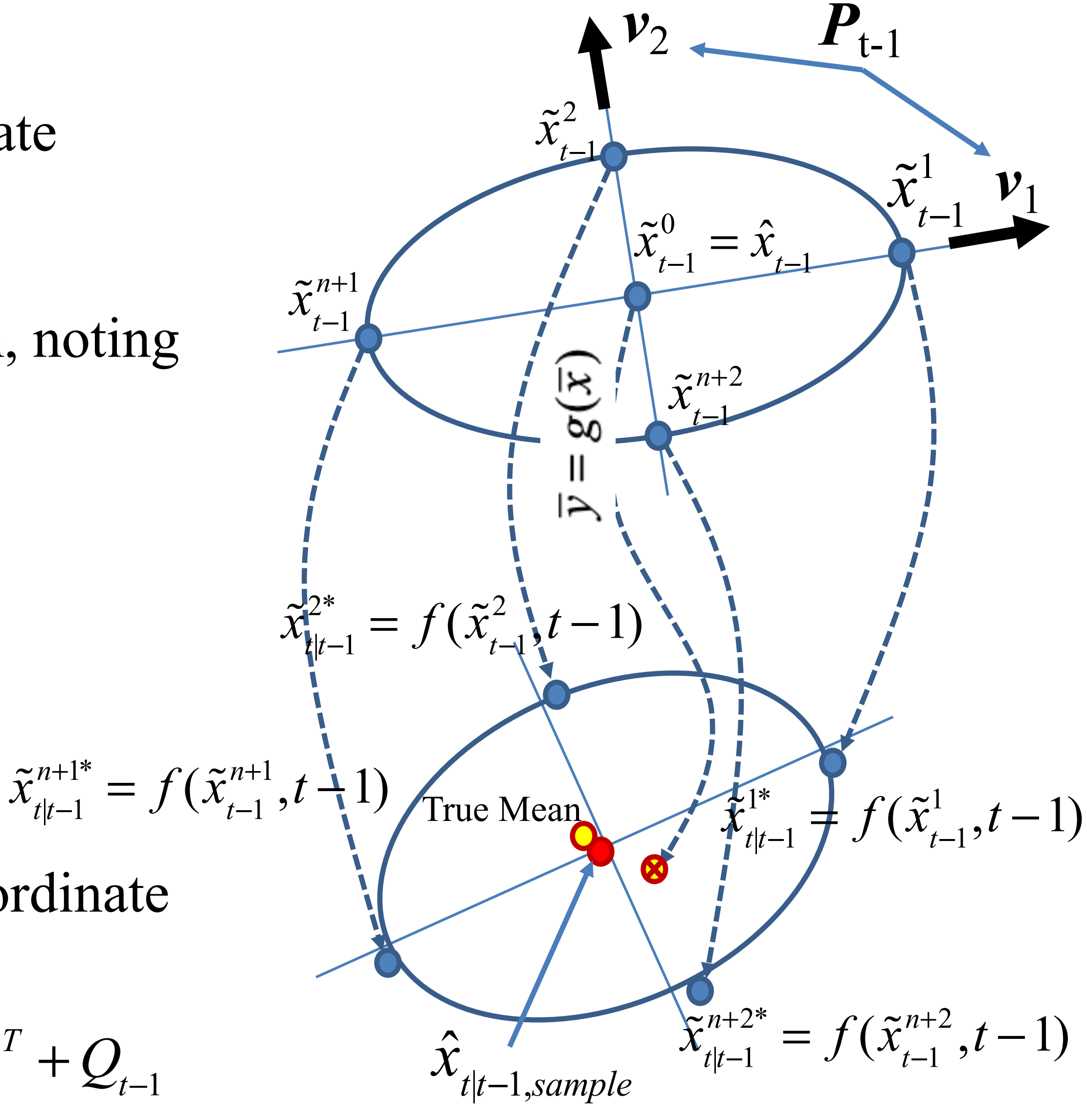
- ❑ For these (2n+1) Sigma points, the weighted mean is computed:

$$\hat{x}_{t|t-1,sample} = \sum_{i=0}^{2n} W_i \hat{x}^{i*}_{t|t-1}$$

- ❑ Covariance and state update, too, are in the global coordinate system.

$$P_{t|t-1,sample} = \sum_{i=0}^{i=2n} W_i (\tilde{x}^{i}_{t|t-1} - \hat{x}_{t|t-1,sample})(\tilde{x}^{i}_{t|t-1} - \hat{x}_{t|t-1,sample})^T + Q_{t-1}$$

$$\hat{x}_t = \hat{x}_{t|t-1,sample} + K_t[y_t - \hat{y}_{t,sample}] \qquad K_t = P_{xy} P_y^{-1}$$



$\boldsymbol{v}_2$
$\boldsymbol{P}_{\text{t-1}}$
$\tilde{x}^2_{t-1}$
$\tilde{x}^1_{t-1}$
$\boldsymbol{v}_1$
$\tilde{x}^0_{t-1} = \hat{x}_{t-1}$
$\tilde{x}^{n+1}_{t-1}$
$\tilde{x}^{n+2}_{t-1}$
$\bar{y} = g(\bar{x})$
$\tilde{x}^{2*}_{t|t-1} = f(\tilde{x}^2_{t-1}, t-1)$
$\tilde{x}^{n+1*}_{t|t-1} = f(\tilde{x}^{n+1}_{t-1}, t-1)$
True Mean
$\tilde{x}^{1*}_{t|t-1} = f(\tilde{x}^1_{t-1}, t-1)$
$\tilde{x}^{n+2*}_{t|t-1} = f(\tilde{x}^{n+2}_{t-1}, t-1)$
$\hat{x}_{t|t-1,sample}$

Weighted mean

Bayes Filter

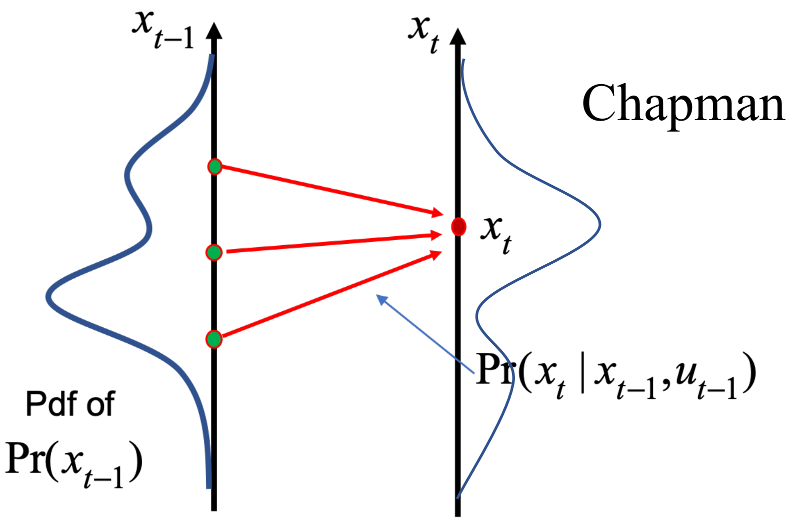
1. Initial Conditions: $g_0(x_0)$ set $t = 1$;
2. Belief Propagation:

$$g_{t|t-1}(x_t) = \int_{-\infty}^{\infty} f_W(x_t - f(x_{t-1},u_{t-1}))\, g_{t-1}(x_{t-1})\, dx_{t-1}$$

3. Assimilate y_t and update the a priori belief

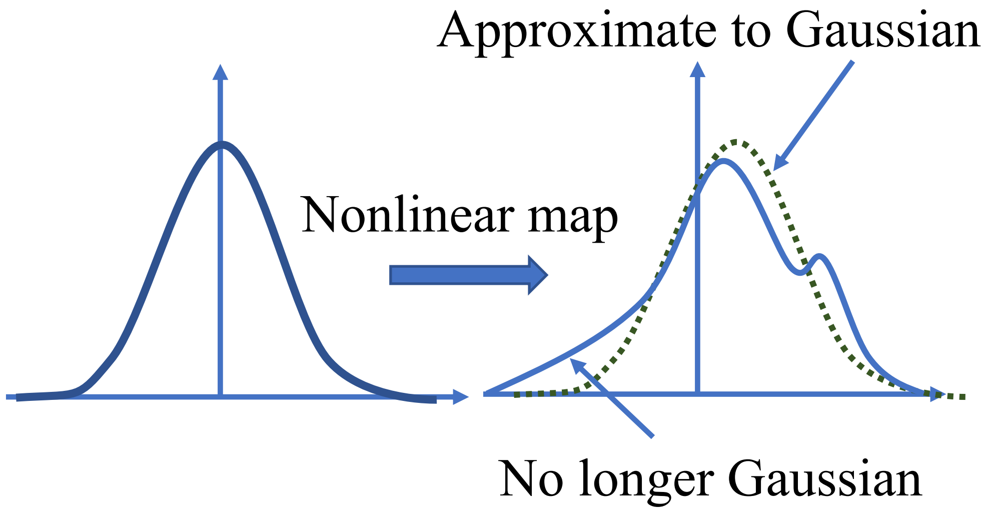
$$g_t(x_t) = \eta f_V(y_t - h(x_t,t)) g_{t|t-1}(x_t)$$

4. Return $g_t(x_t)$. Set $t = t + 1$ and repeat.

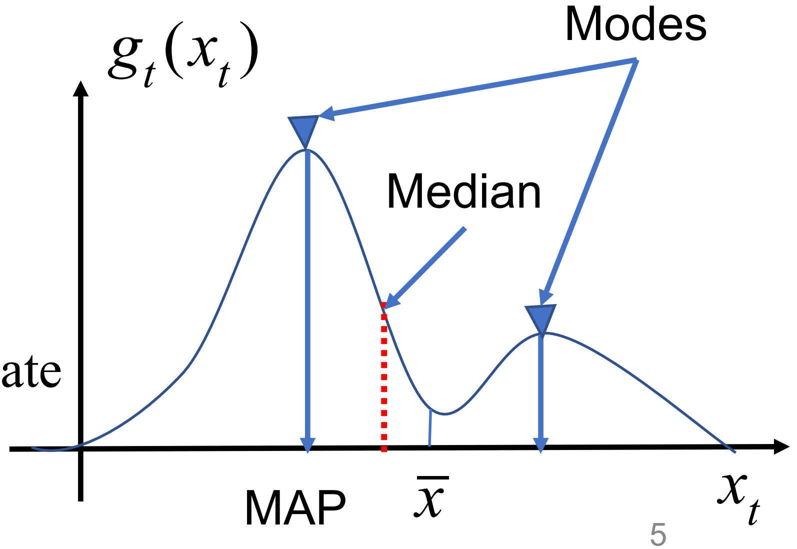


Chapman-Kolmogorov Eq.

Limitation to UKF



Multi-Hypothesis Estimate



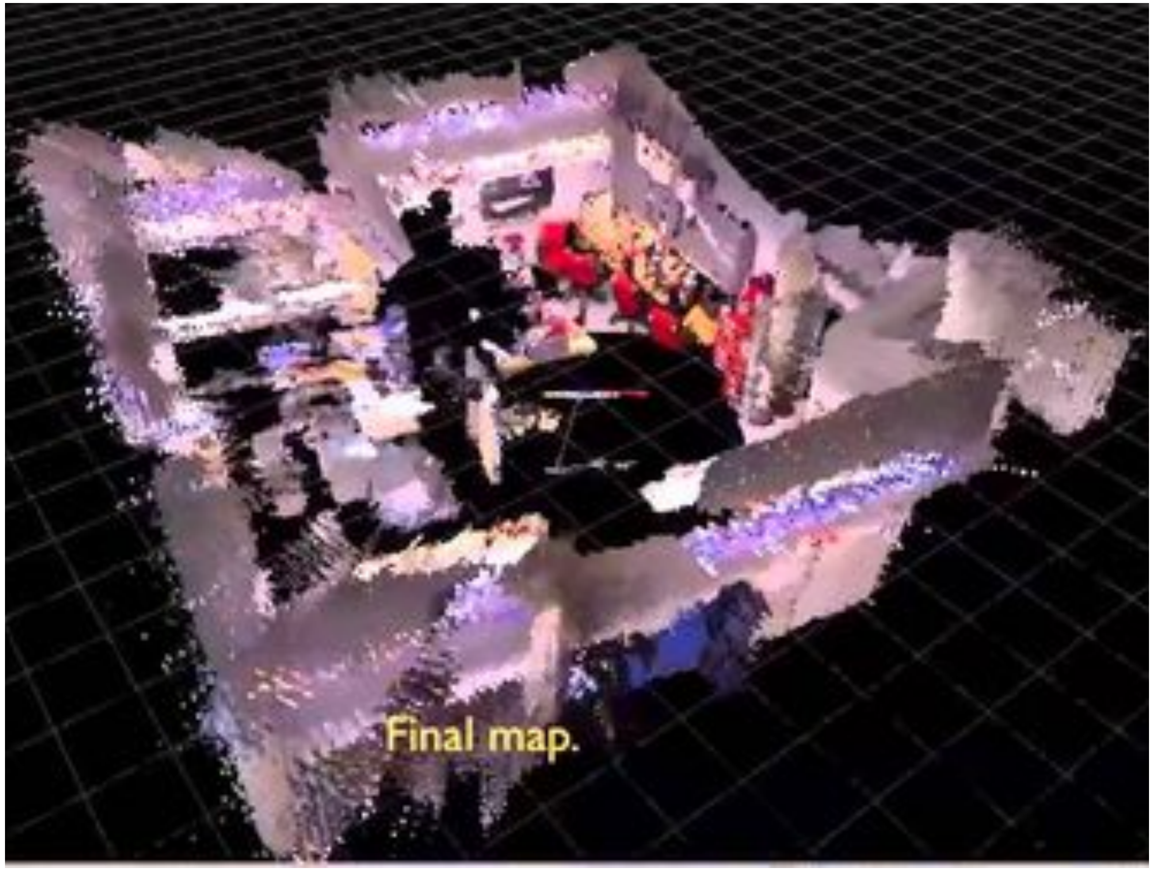
2.160 Identification, Estimation, and Learning

Part 2 Estimation

Lecture 11

Simultaneous Localization And Mapping (SLAM)

H. Harry Asada
Department of Mechanical Engineering
MIT



Self-Driving Cars

Late 1980’s Nav Lab, CMU Robotics Institute, Chuck Thorpe et al.

https://gaia.adage.com/images/bin/image/x-large/241402914.jpg

$6T market

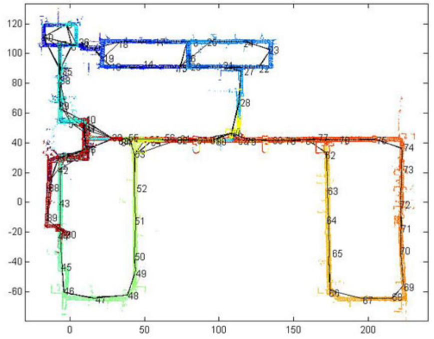
http://moovafrica.com/news/wp-content/uploads/2017/08/self-driving-car-1.jpg

Simultaneous Localization and Mapping (SLAM)

Hugh Durrant-Whyte

John Leonard

Simultaneous state (location) and parameter (involved in the map) estimation

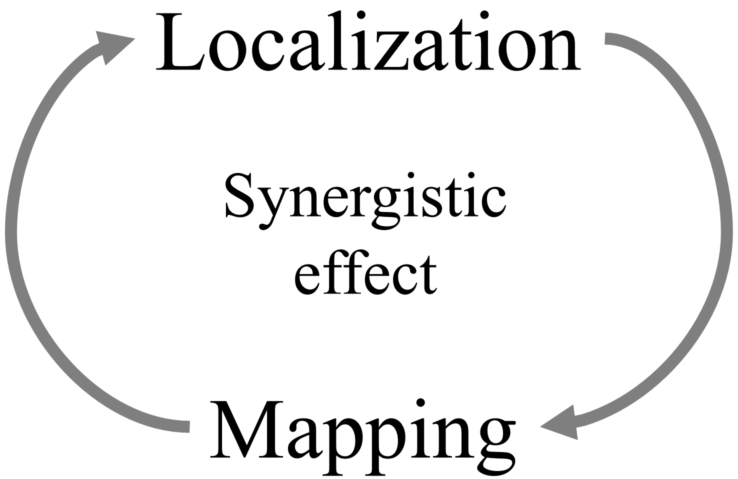


J. J. Leonard and H. F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. IEEE Trans. Robotics and Automation, 7(3):376-382, June 1991.

Simultaneous

Localization : estimating the position and orientation of a vehicle

and Mapping : generating and updating a map



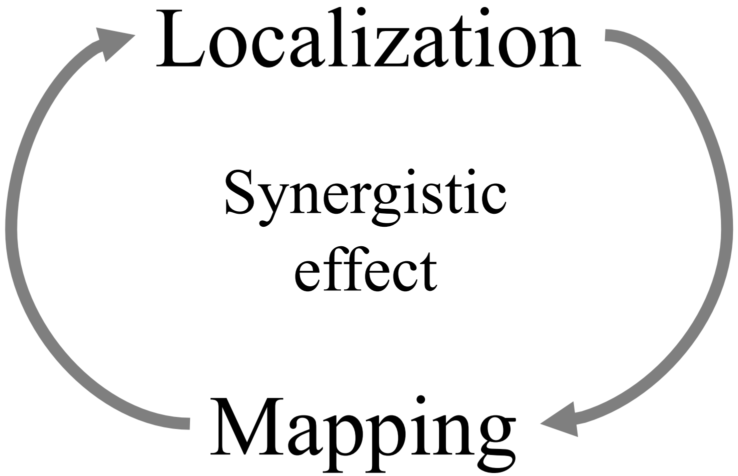
Localization

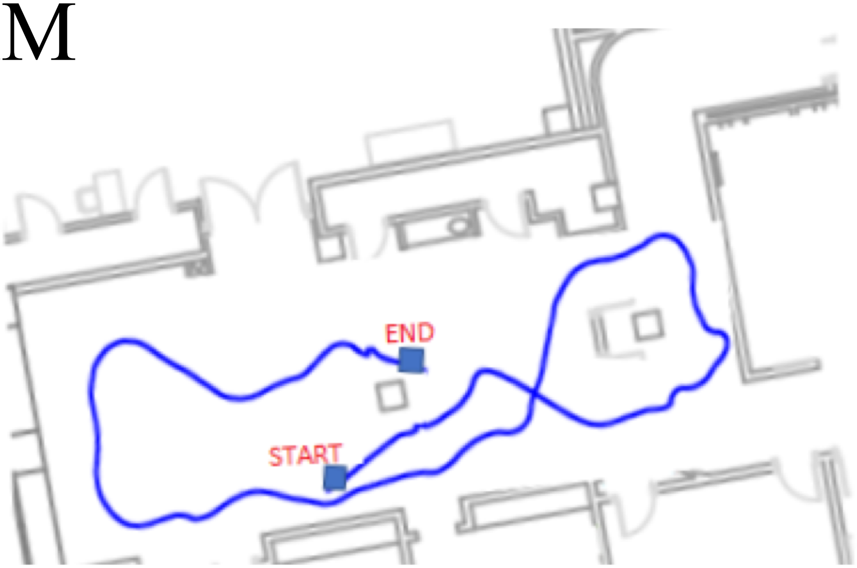
Synergistic effect

Mapping

A more accurate map can localize the robot more accurately.
More accurate localization can generate a more accurate map.

SLAM





Augment the state variables by including feature parameters, e.g. line parameters, to be estimated. There is no fundamental difference between state estimate and parameter estimate.

If n features (line parameters) are in the map, the state variables are:

$$X_t = \left(\underbrace{x_t \quad y_t \quad \theta_t}_{\text{Robot location}} \quad \underbrace{\alpha_1 \quad r_1 \quad \alpha_2 \quad r_2 \quad \cdots \quad \alpha_n \quad r_n}_{\text{Parameters of Map}} \right)^T$$

If the environment is static, these features are constant but unknown.

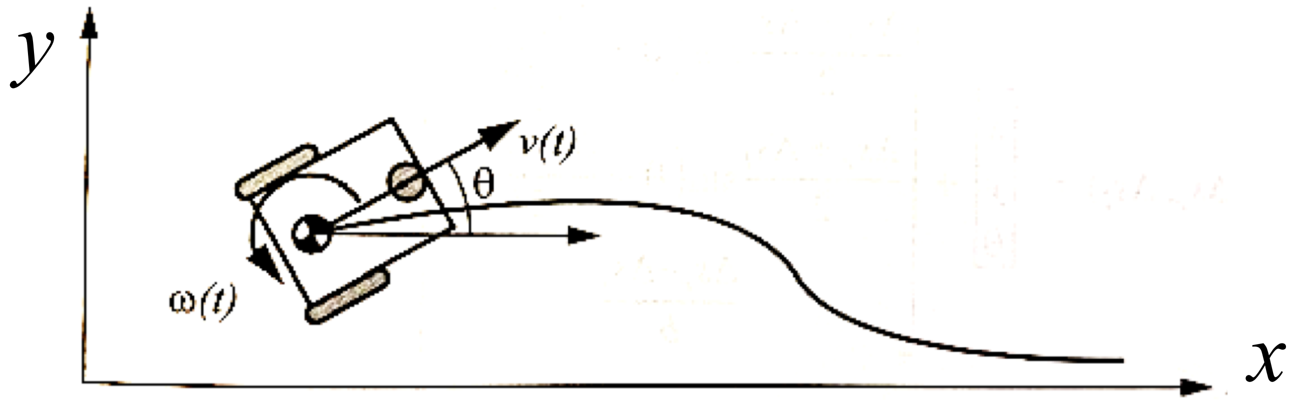
Outline

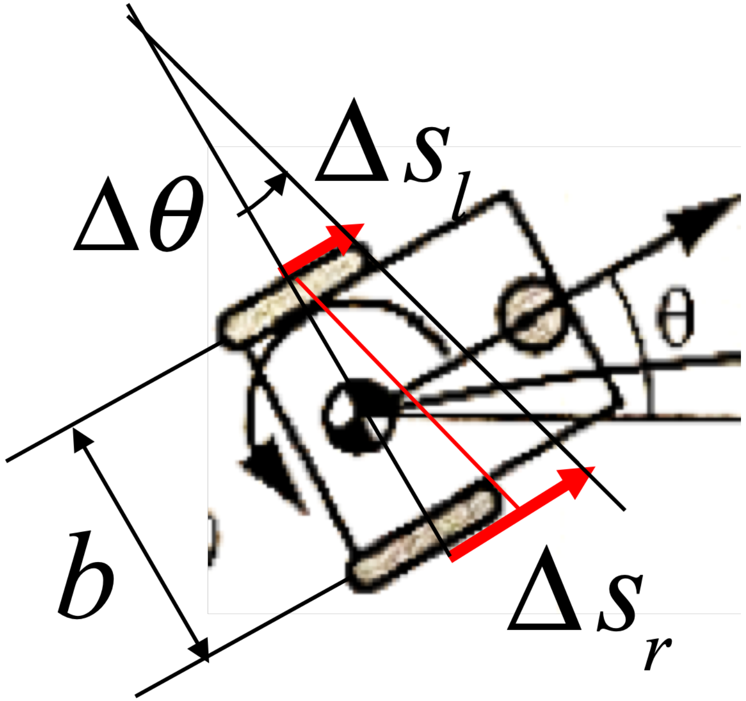
- SLAM formulation
  - Landmark estimation via augmentation of state variables
- Vehicle localization based on Kalman Filter
  - Vehicle kinematics
  - State equation and process noise
  - Map representation (a simple example)
  - Range data processing: line/wall detection
  - Measurement prediction
  - Matching: data association
  - State update
- Introduction to Particle Filter
- Rao-Blackwell Filter

Vehicle kinematics

$$v = \frac{1}{2}(\Delta s_r + \Delta s_l)$$

$$\Delta\theta = \frac{1}{b}(\Delta s_r - \Delta s_l)$$





$$\Delta x = v\cos(\theta + \frac{1}{2}\Delta\theta) \qquad \Delta y = v\sin(\theta + \frac{1}{2}\Delta\theta)$$

Deterministic State Equation

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} v_t \cos(\theta_t + \frac{1}{2}\Delta\theta_t) \\ v_t \sin(\theta_t + \frac{1}{2}\Delta\theta_t) \\ \Delta\theta_t \end{pmatrix} = f(x_t, y_t, \theta_t, \Delta s_r, \Delta s_l)$$

$$\Downarrow$$

$$x_{t+1} = A_t x_t + B_t u_t \cdot$$

$$A_t = \left.\frac{\partial f}{\partial x_t}\right|_t = \begin{pmatrix} 1 & 0 & -v_t \sin(\theta_t + \frac{1}{2}\Delta\theta_t) \\ 0 & 1 & v_t \cos(\theta_t + \frac{1}{2}\Delta\theta_t) \\ 0 & 0 & 1 \end{pmatrix}$$

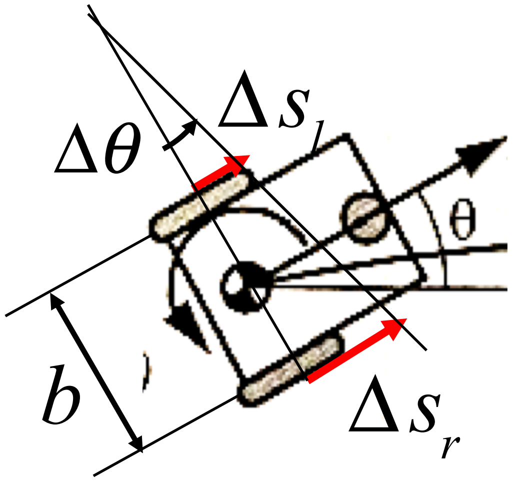
This state transition matrix subsumes inputs:

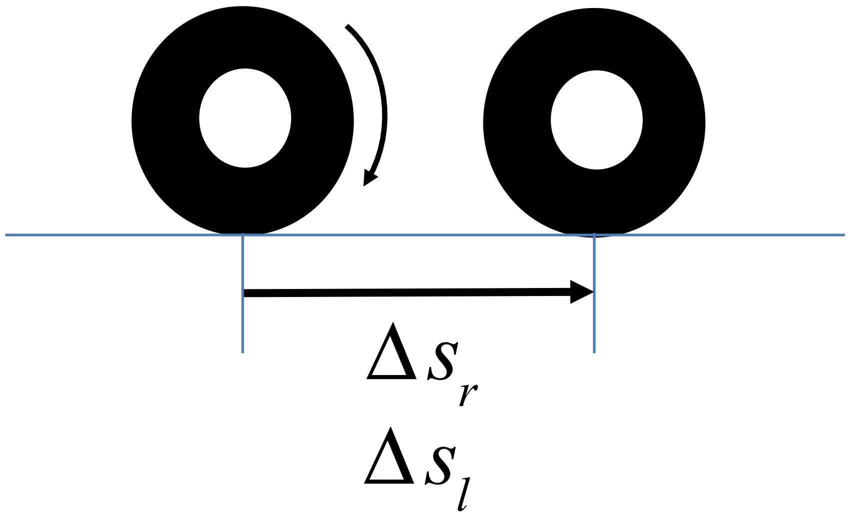
$$v_t = \frac{1}{2}(\Delta s_r + \Delta s_l)$$

$$\Delta\theta = \frac{1}{b}(\Delta s_r - \Delta s_l)$$

$$\boxed{x_{t+1} = A_t(u_t)x_t}$$

Process Noise Dynamics $x_{t+1} = A_t(u_t)x_t + G_t w_t$





❑ The wheel displacement (path length) is uncertain due to slip, roughness of the ground, feedback control error, etc. We model the uncertainty as additive noise with covariance Σ_S.

❑ The variance increases in proportion to the distance traveled.

$$\Delta s_r = \Delta \overline{s}_r + w_r, \quad \Delta s_l = \Delta \overline{s}_l + w_l$$

$$\Sigma_S = \mathrm{cov}(\Delta s_r, \Delta s_l) = \begin{pmatrix} k_r|\Delta s_r| & 0 \\ 0 & k_l|\Delta s_l| \end{pmatrix} \qquad w_t = \begin{pmatrix} w_{r,t} \\ w_{l,t} \end{pmatrix}$$

Process Noise Dynamics

$$w_t = \begin{pmatrix} w_{r,t} \\ w_{l,t} \end{pmatrix}$$

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} v_t \cos(\theta_t + \frac{1}{2}\Delta\theta_t) \\ v_t \sin(\theta_t + \frac{1}{2}\Delta\theta_t) \\ \Delta\theta_t \end{pmatrix} = f(x_t, y_t, \theta_t, \Delta s_r, \Delta s_l)$$

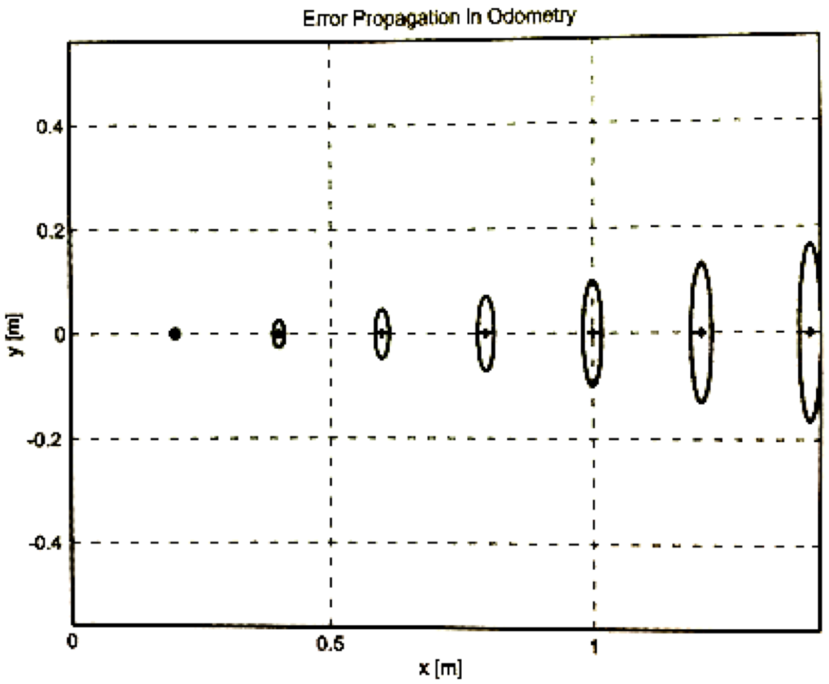
$$\Downarrow$$

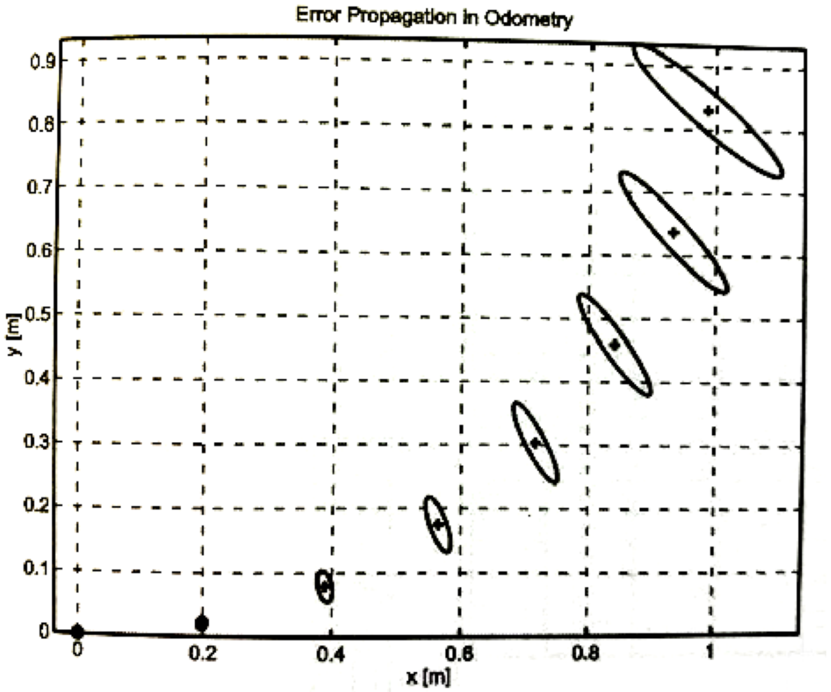
$$x_{t+1} = A_t(u_t)x_t + G_t w_t$$

$$v_t = \frac{1}{2}(\Delta s_r + \Delta s_l) = \frac{\Delta s}{2} \qquad \Delta\theta = \frac{1}{b}(\Delta s_r - \Delta s_l)$$

$$G_t = \left.\frac{\partial f}{\partial(\Delta s_r, \Delta s_l)}\right|_t = \begin{pmatrix} \frac{1}{2}\cos(\theta_t + \frac{1}{2}\Delta\theta_t) - \frac{\Delta s}{2b}\sin(\theta_t + \frac{1}{2}\Delta\theta_t) & \frac{1}{2}\cos(\theta_t + \frac{1}{2}\Delta\theta_t) + \frac{\Delta s}{2b}\sin(\theta_t + \frac{1}{2}\Delta\theta_t) \\ \frac{1}{2}\sin(\theta_t + \frac{1}{2}\Delta\theta_t) + \frac{\Delta s}{2b}\cos(\theta_t + \frac{1}{2}\Delta\theta_t) & \frac{1}{2}\sin(\theta_t + \frac{1}{2}\Delta\theta_t) - \frac{\Delta s}{2b}\cos(\theta_t + \frac{1}{2}\Delta\theta_t) \\ \frac{1}{b} & -\frac{1}{b} \end{pmatrix}$$

State Propagation $P_{t+1|t} = A_t P_t A_t^T + G_t \Sigma_s G_t^T$



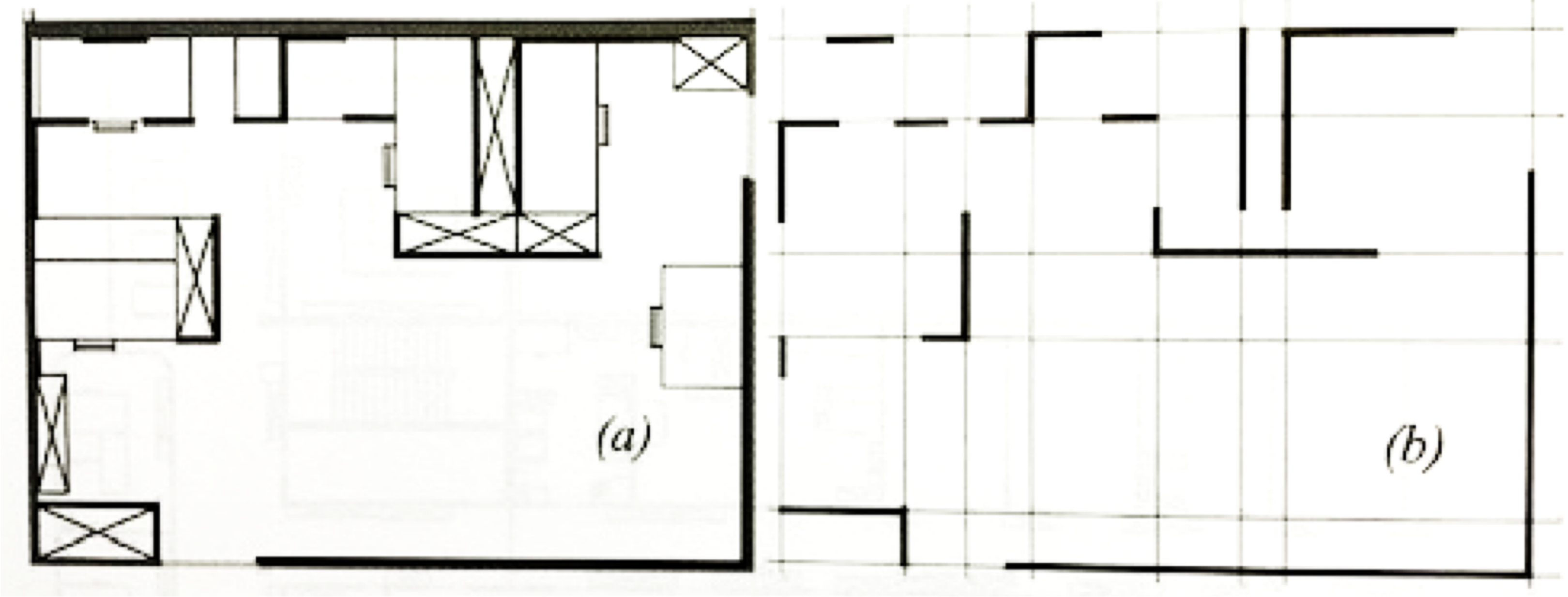


The vehicle location uncertainty increases as the vehicle travels. The uncertainty perpendicular to the movement grows more rapidly than the longitudinal direction.

As it goes around a circle, the principal axis of the uncertainty ellipse is no longer perpendicular to the direction of movement.

Map Representation

A simple example. We consider only Line representation.

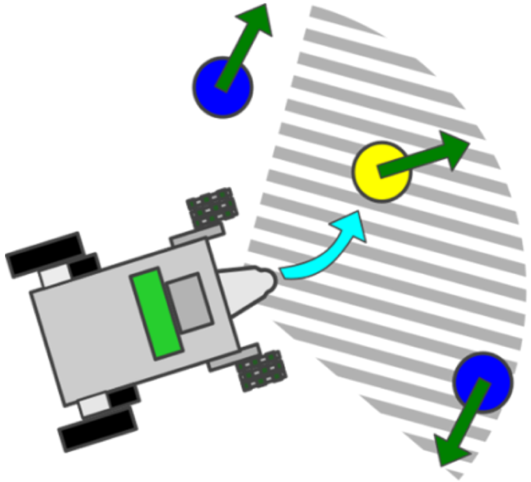


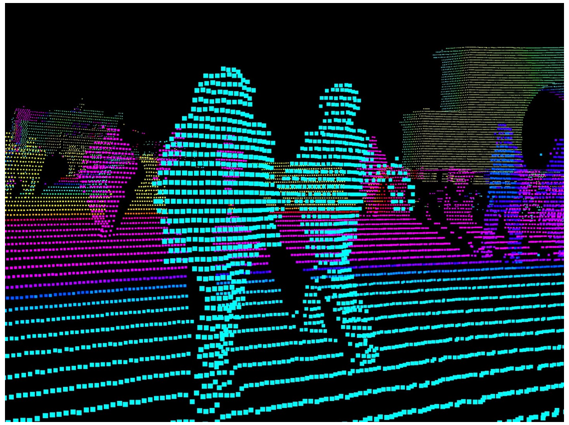
- ❑ The state of the art is 3D map building with locally tunable resolution based on point-cloud data obtained from 3D scanners, e.g. 3D LIDAR, depth cameras.
- ❑ Image processing is heavily involved. Kalman Filter is used for integrating RGB camera image with LIDAR depth information.

Range Sensors
LIght Detection And Ranging (LIDAR)

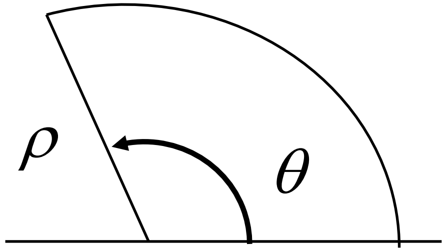
3D LIDAR

2D LIDAR





LIDAR is in Polar coordinate.



Representing a line in Polar Coordinate

- ❑ A straight line with parameters *r* and *α*.

$$\begin{pmatrix} x & y \end{pmatrix}\begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} = x\cos\alpha + y\sin\alpha = r$$

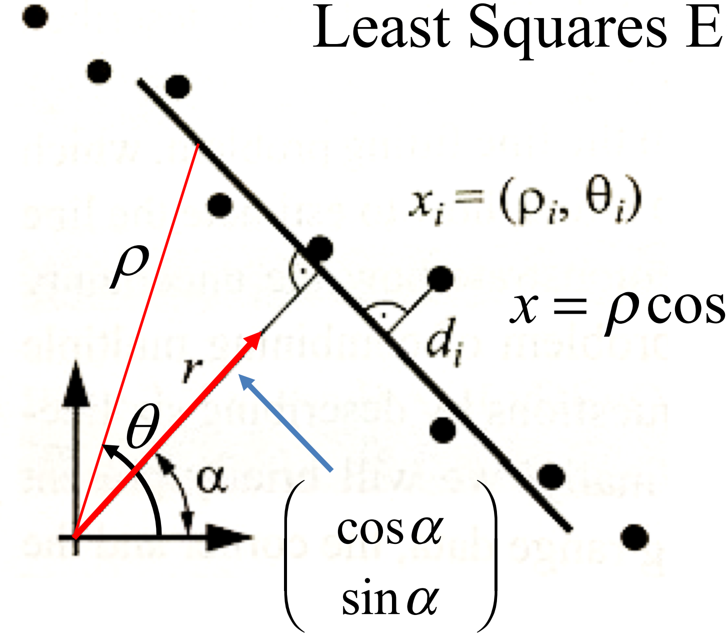
- ❑ If a LIDAR detects a point on the line:

$$\rho\cos(\theta-\alpha)-r=0$$

- ❑ Suppose that the LIDAR detects *N* points:

$$D=\{(\rho_i,\theta_i)\,|\,i=1,\cdots,N\}$$

Least Squares Estimate



$$\begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix}$$

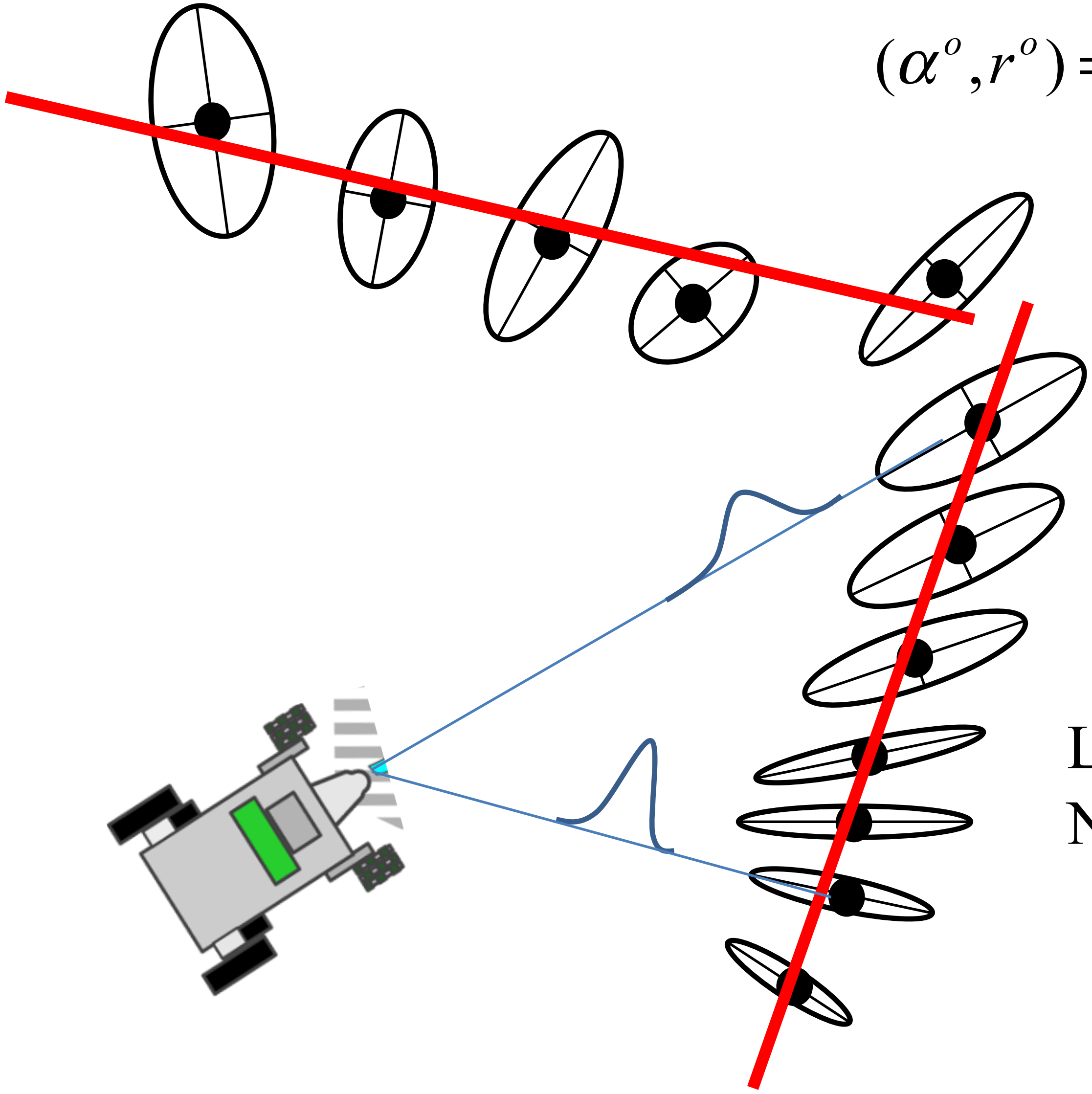
$$x=\rho\cos\theta,\ y=\rho\sin\theta$$

- ❑ The line parameters, *r* and *α*, that minimizes the total squared distances between the sample points and the line are given by

$$(\alpha^o, r^o) = \arg\min_{\alpha,r}\sum_{i=1}^{N} d_i^2 \qquad \text{where} \qquad d_i = \rho_i\cos(\theta_i-\alpha)-r$$

Considering the LIDAR noise covariance
we use the weighted squared distance to minimize.

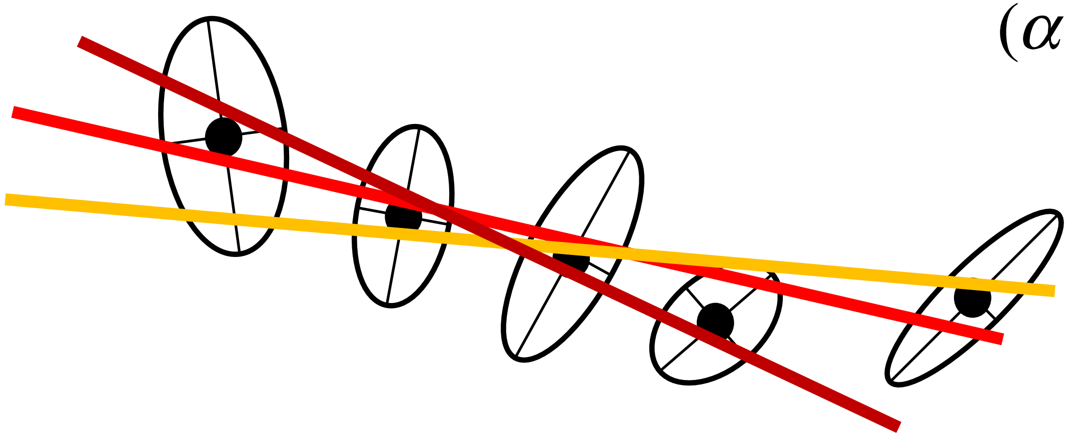
$$(\alpha^o, r^o) = \arg\min_{\alpha,r} \sum_{i=1}^{N} \frac{1}{\sigma^2} d_i^2$$



σ^2 becomes larger as the distance gets longer. Depth measurement is more inaccurate.

LIDAR sensor
Noise covariance

Observation Error Covariance



$$(\alpha^o, r^o) = \arg\min_{\alpha,r} \sum_{i=1}^{N} \frac{1}{\sigma^2} d_i^2$$

- ❑ Due to the LIDAR noise, the estimated line parameters have variability.

- ❑ The uncertainty in estimating the *i*-th line parameters is quantified with error covariance

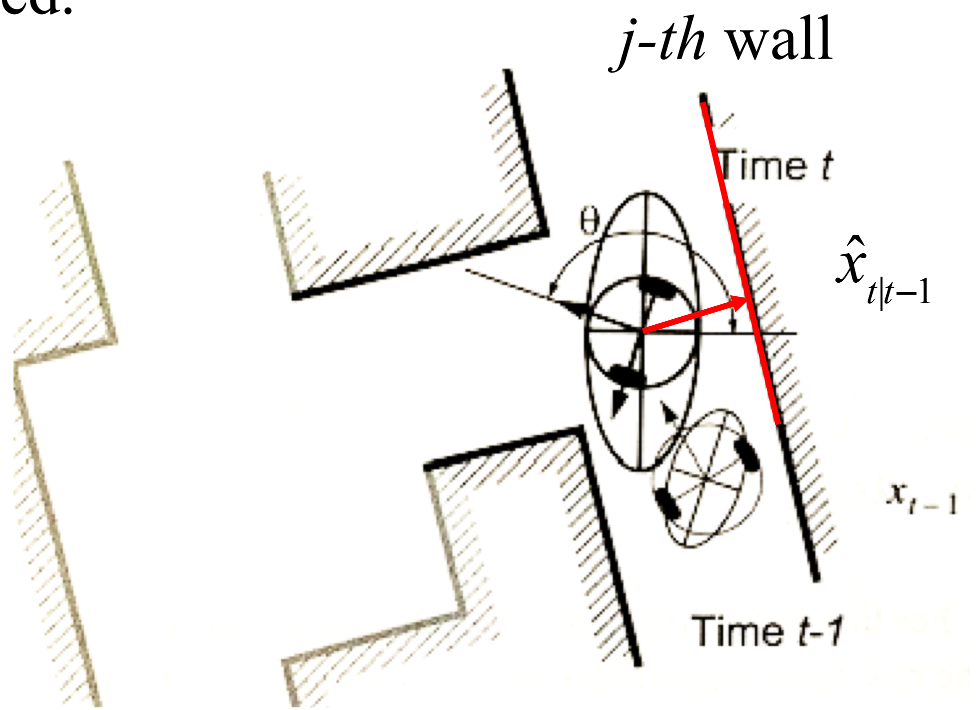
$$\mathrm{Cov}(\alpha, r) = R_t^i = \begin{pmatrix} \sigma_{\alpha\alpha,t}^{\ i} & \sigma_{\alpha r,t}^{\ i} \\ \sigma_{\alpha r,t}^{\ i} & \sigma_{rr,t}^{\ i} \end{pmatrix}$$

- ❑ Here, we treat α and r as measurements, i.e. sensor outputs.

Measurement Prediction

Based on the stored map and the a priori estimate of the vehicle location $\hat{x}_{t|t-1}$, the measurement prediction of the expected features are generated.

$$\hat{y}_t^j = \begin{pmatrix} \hat{\alpha}_t^j \\ \hat{r}_t^j \end{pmatrix}$$



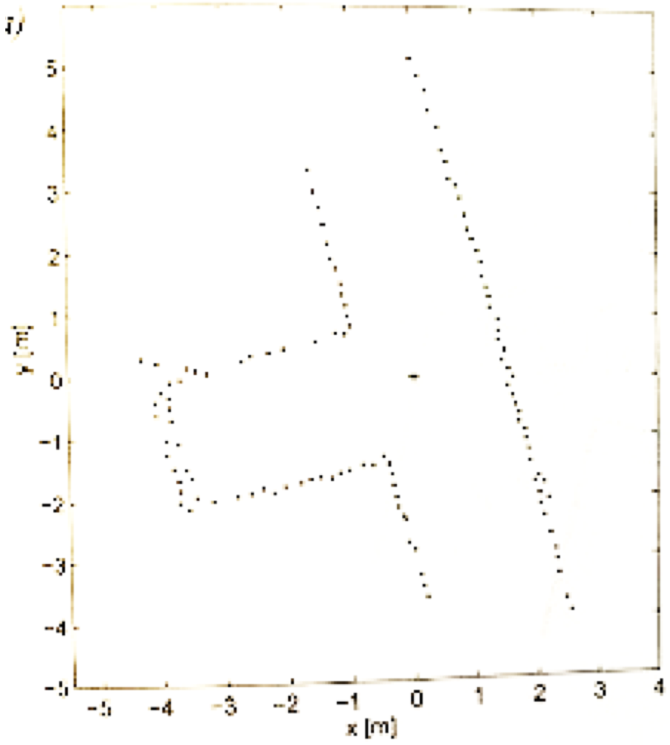
$$\hat{y}_t^j = h(\hat{x}_{t|t-1}, \underline{\alpha_{map}^j, r_{map}^j})$$

Coordinate transformation based on $\hat{x}_{t|t-1}$

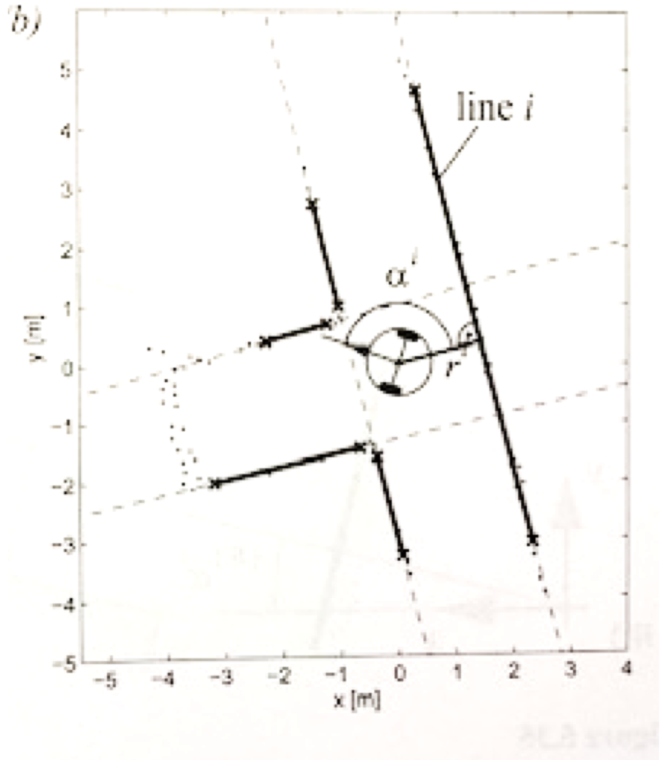
The *j*-th line parameters stored in a map

Flow of Image Data Processing

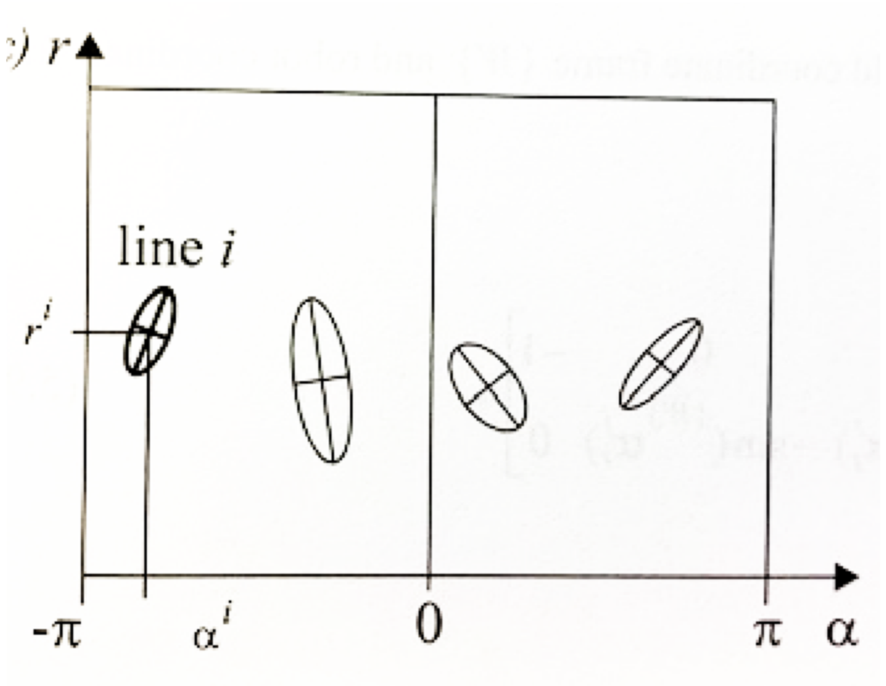
Point Cloud



Identified Lines



$\alpha - r$ space



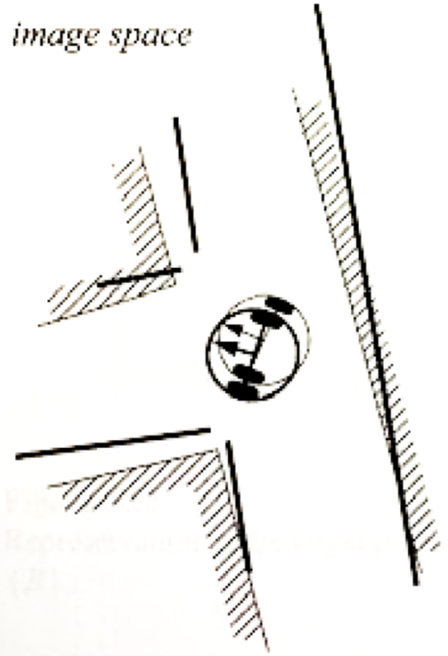
Line parameter distribution and Estimation error covariance

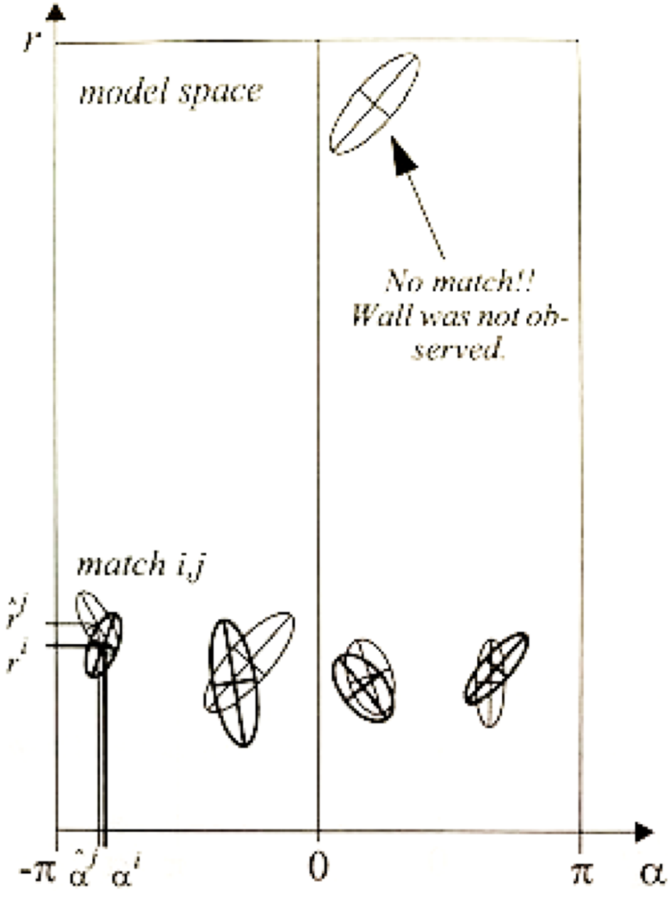
Data Association

❑ Matching between predicted line parameters (features) and the measured line parameters: Which line matches which line.

Measured Predicted

$$\Delta y_t^{ij} = \begin{pmatrix} \alpha_t^i \\ r_t^i \end{pmatrix}^T - \begin{pmatrix} \hat{\alpha}_t^j \\ \hat{r}_t^j \end{pmatrix}^T$$





❑ Mahalanobis Distance

$$\Delta y_t^{ij} (P_{y,t}^{ij})^{-1} (\Delta y_t^{ij})^T \leq g^2$$

Find *i* and *j* that are within the distance of g.

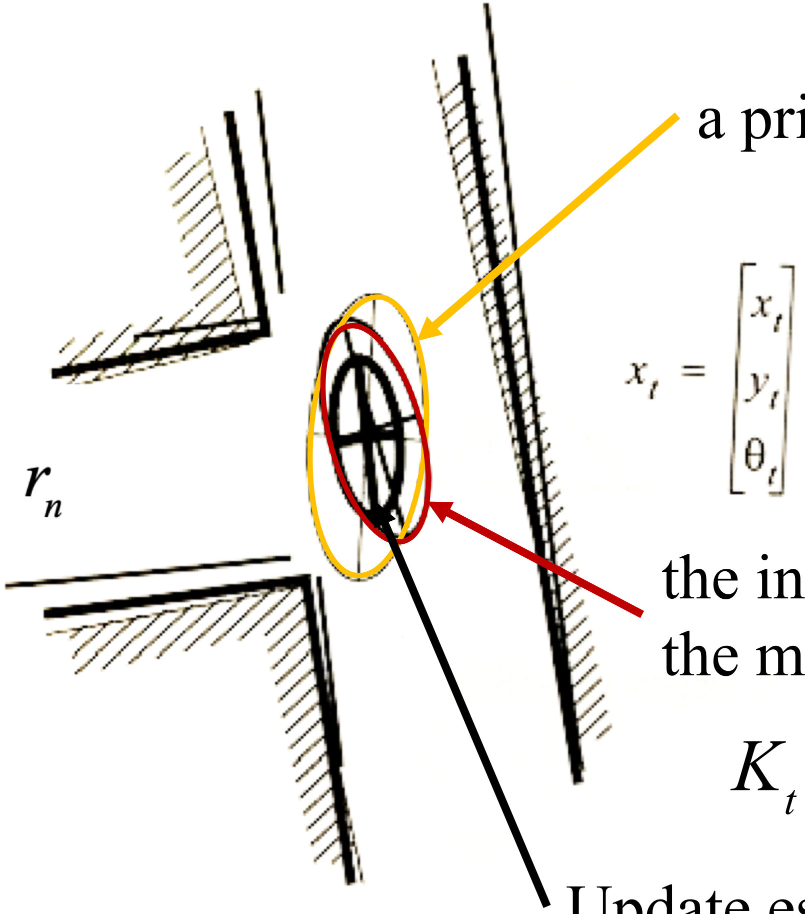
❑ Here the distance is weighted by the inverse of the innovation covariance.

$$P_{y,t}^{ij} = H_t^j P_{t|t-1} (H_t^j)^T + R_t^i$$

State Update: Kalman filter estimate of the new vehicle location

Measurement y_t

$$\alpha_1 \quad r_1 \quad \alpha_2 \quad r_2 \quad \cdots \quad \alpha_n \quad r_n$$



$$x_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

a priori estimate of vehicle location

$$\hat{x}_{t|t-1}, P_{t|t-1}$$

the innovation gained by the measurement

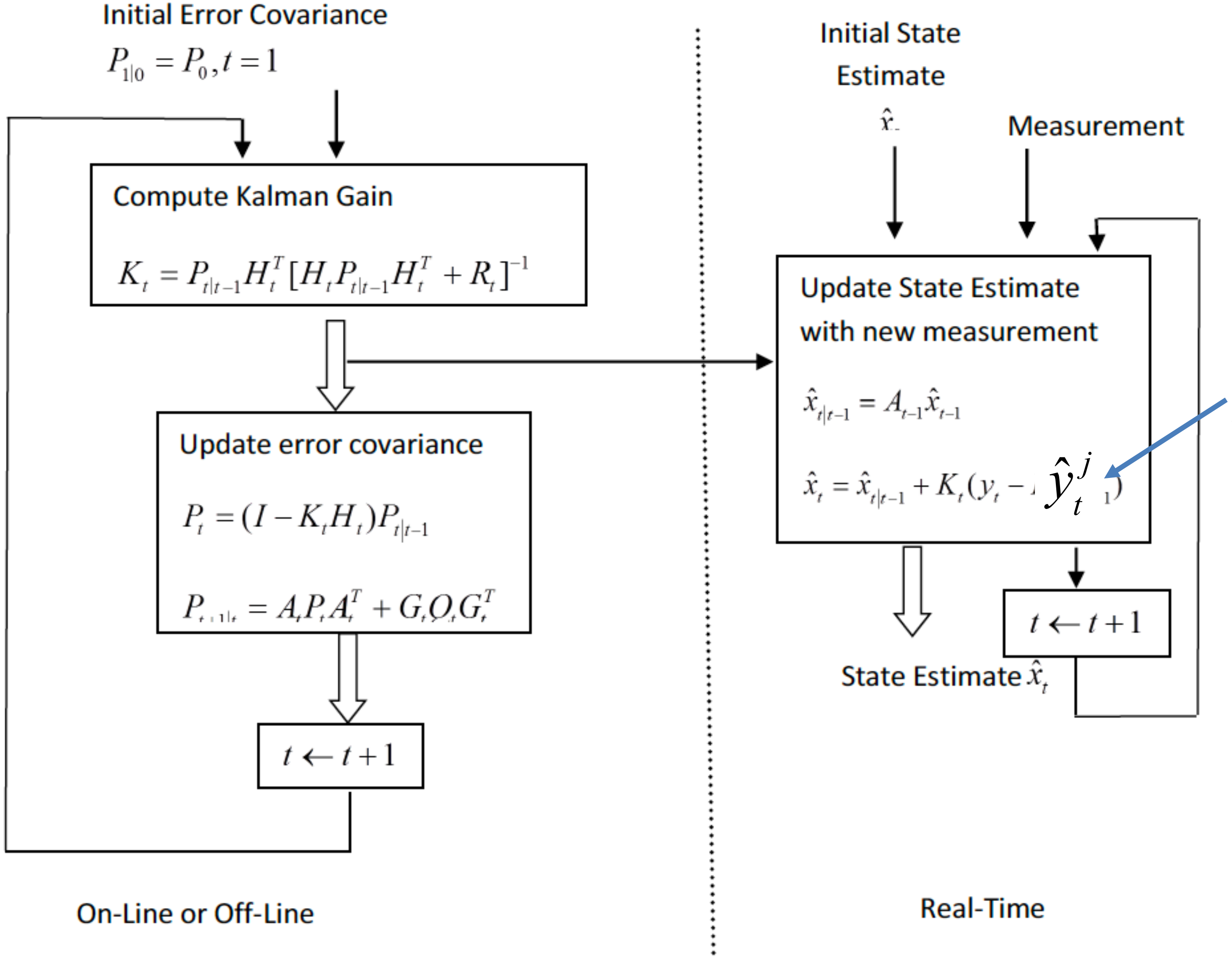
$$K_t(y_t - \hat{y}_t), \;\; P_y = H_t P_{t|t-1} H_t^T + R_t$$

Update estimate of the vehicle location

$$\hat{x}_t, P_t$$

$$K_t = P_{t|t-1} H_t^T [H_t P_{t|t-1} H_t^T + R_t]^{-1}$$

$$\hat{x}_t = \hat{x}_{t|t-1} + P_t H_t^T R_t^{-1} \Delta y_t$$



Initial Error Covariance

$P_{1|0} = P_0, t = 1$

Compute Kalman Gain

$K_t = P_{t|t-1} H_t^T [H_t P_{t|t-1} H_t^T + R_t]^{-1}$

Update error covariance

$P_t = (I - K_t H_t) P_{t|t-1}$

$P_{t+1|t} = A_t P_t A_t^T + G_t Q_t G_t^T$

$t \leftarrow t+1$

On-Line or Off-Line

Initial State Estimate

$\hat{x}_{-}$

Measurement

Update State Estimate with new measurement

$\hat{x}_{t|t-1} = A_{t-1} \hat{x}_{t-1}$

$\hat{x}_t = \hat{x}_{t|t-1} + K_t(y_t - \hat{y}_t^j{}_1)$

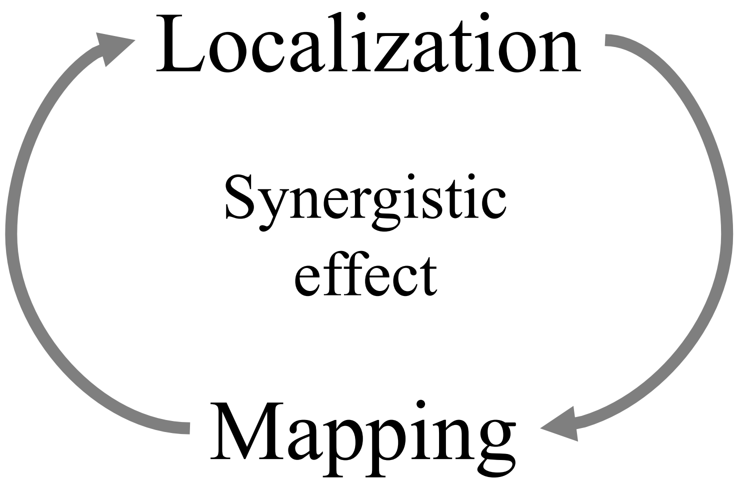
$t \leftarrow t+1$

State Estimate $\hat{x}_t$

Real-Time

$$\hat{y}_t^j = h(\hat{x}_{t|t-1}, \alpha_{map}^j, r_{map}^j)$$

SLAM


Localization

Synergistic effect

Mapping

□ Augment the state variables by including the feature parameters, e.g. line parameters, to be estimated.

□ If n features (line parameters) are in the map, the state variables are:

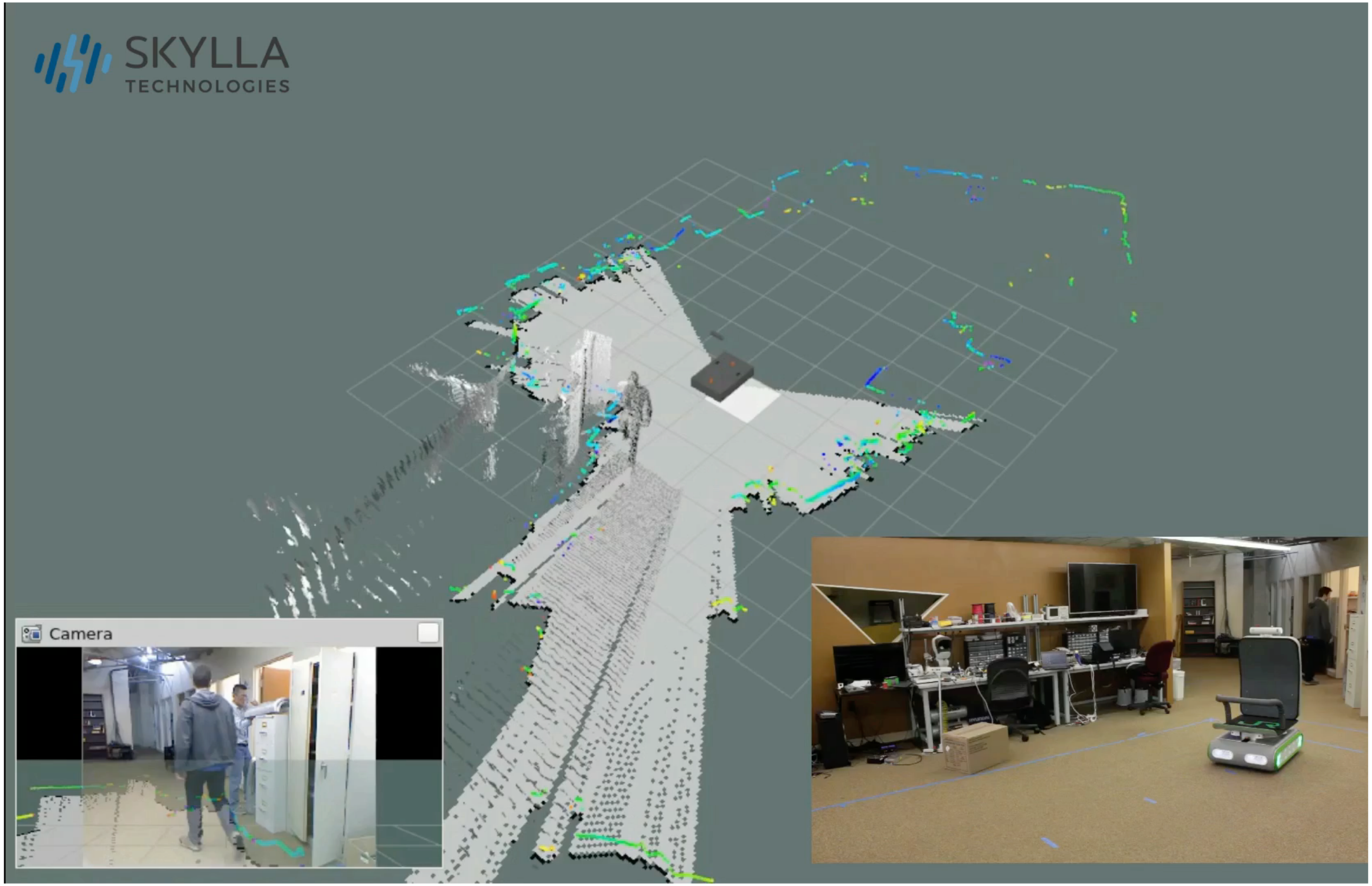
$$X_t = \begin{pmatrix} x_t & y_t & \theta_t & \alpha_1 & r_1 & \alpha_2 & r_2 & \cdots & \alpha_n & r_n \end{pmatrix}^T$$

□ If the environment is static, these features are constant but unknown.

□ As the features are better known, the updated feature values are used for predicting sensor outputs: $\hat{y}_t^j = h(\hat{x}_{t|t-1}, \hat{\alpha}^j, \hat{r}^j)$

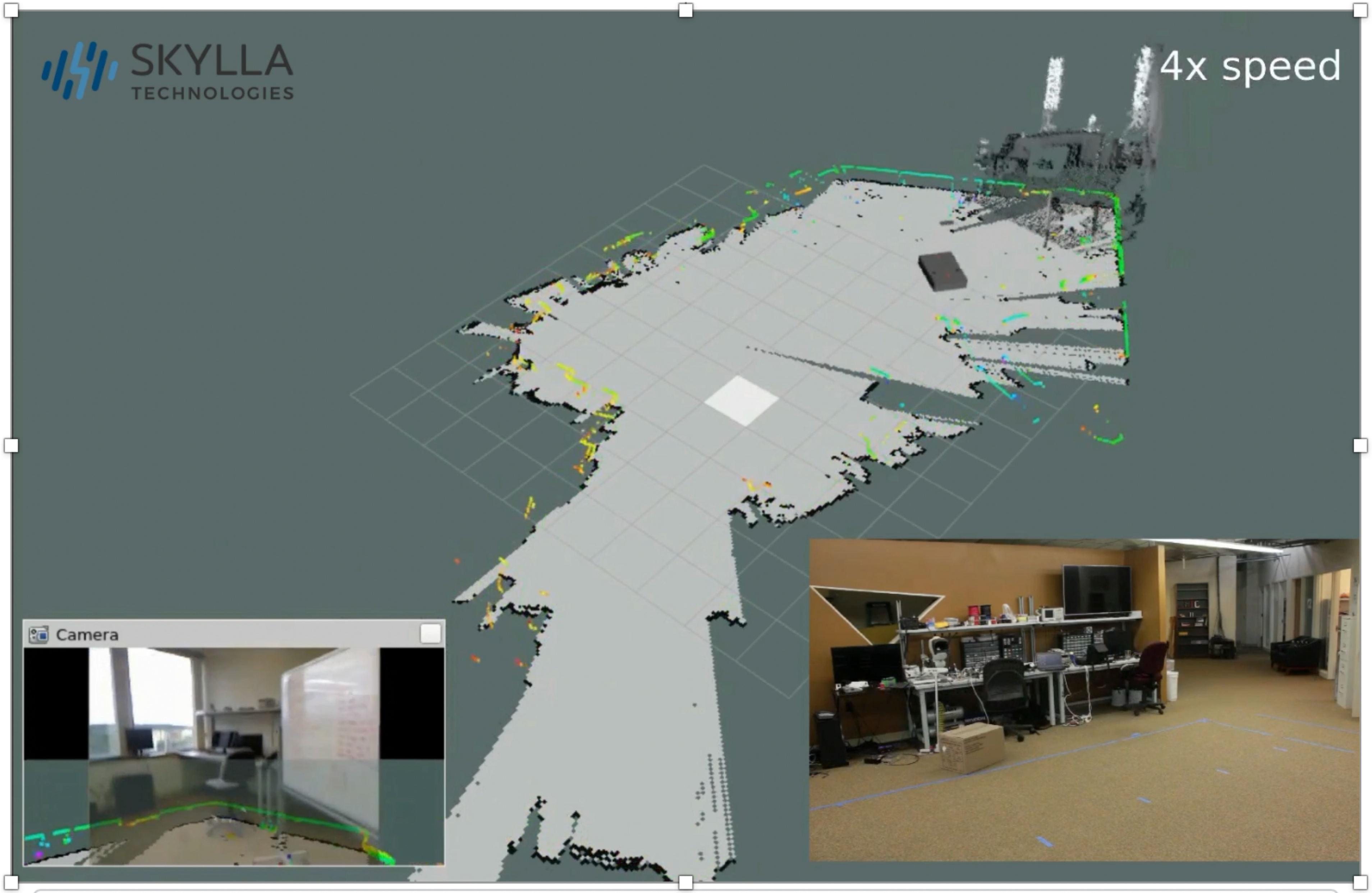
Kalman Filter Based SLAM

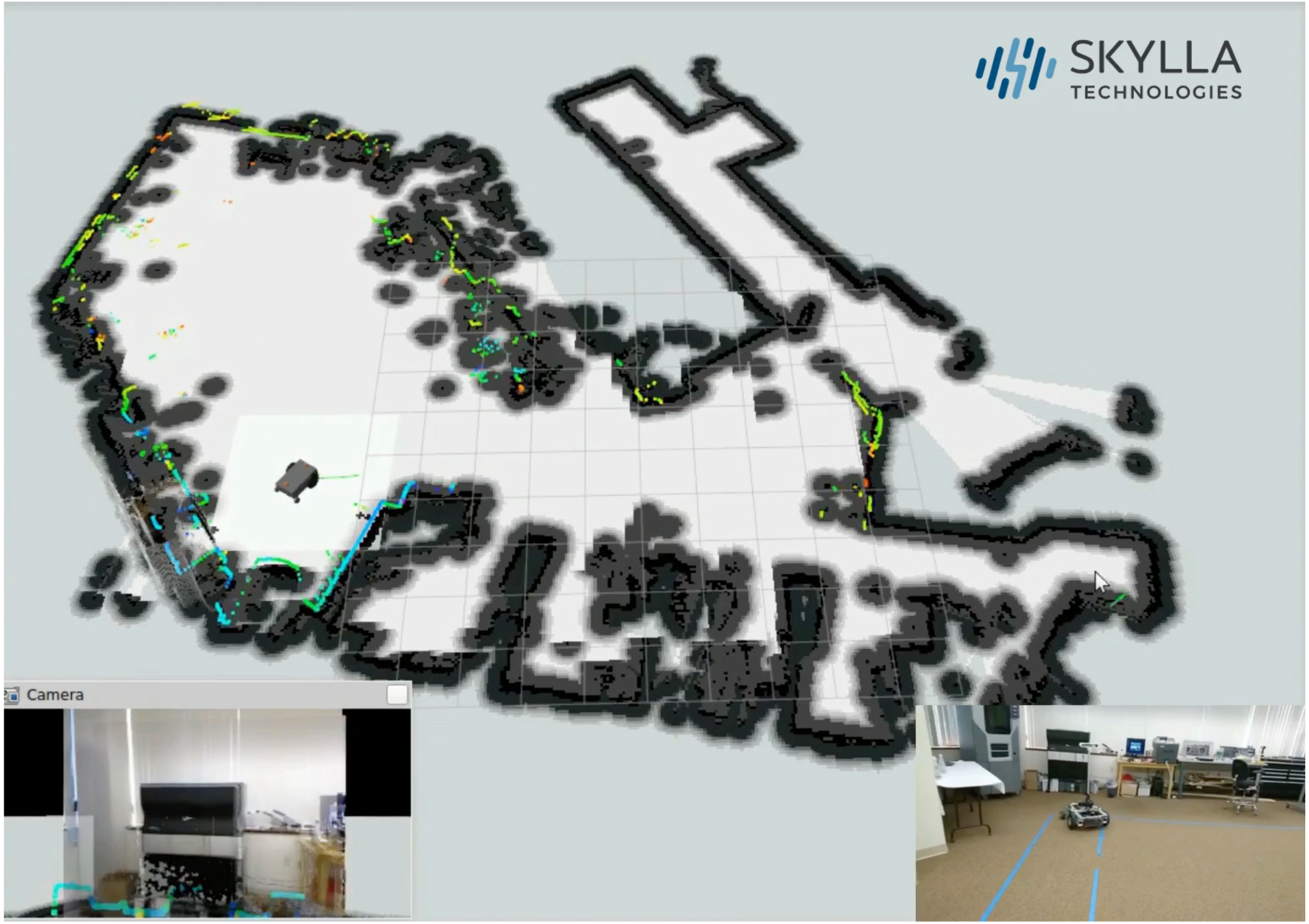
- Generally, fast computation
- Assume Gaussian noise: In reality, measurement noise and process noise are non-Gaussian.
- Extended Kalman Filter is limited in dealing with rapidly changing nonlinearity.
- SLAM is limited to slowly changing environment. It is still a future issue to deal with dynamically changing environment.
- Parametric representation of 3D environment is often a challenge.



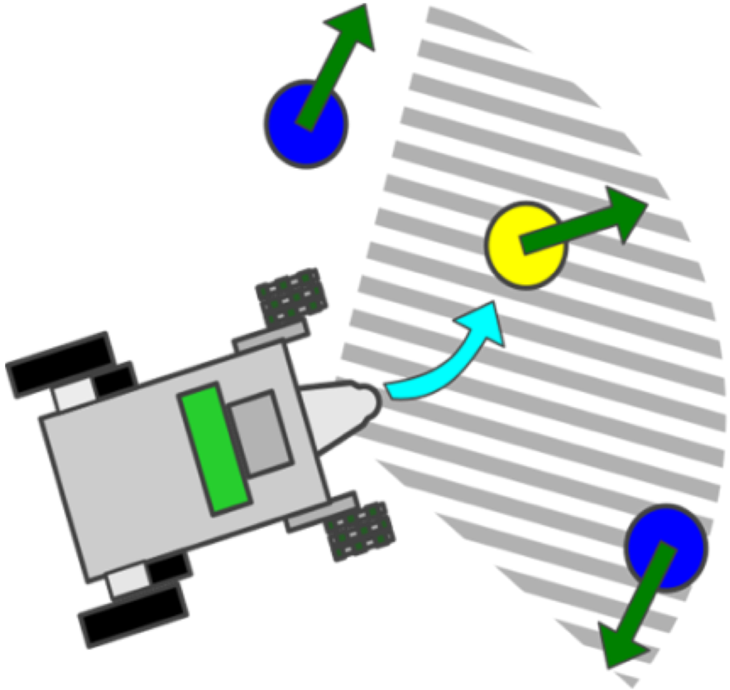
SKYLLA
TECHNOLOGIES
Camera

The map is revised and becomes more accurate, as the robot moves around and more images are obtained.

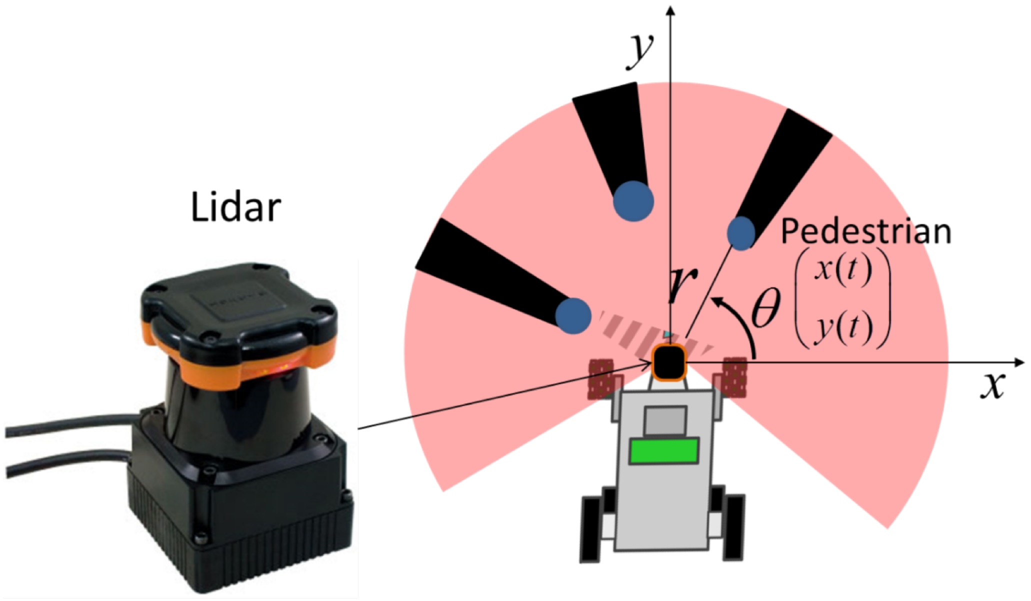



SKYLLA
TECHNOLOGIES
Camera

Context-Oriented Project #2
SLAM



Part 1: Pedestrian Tracking



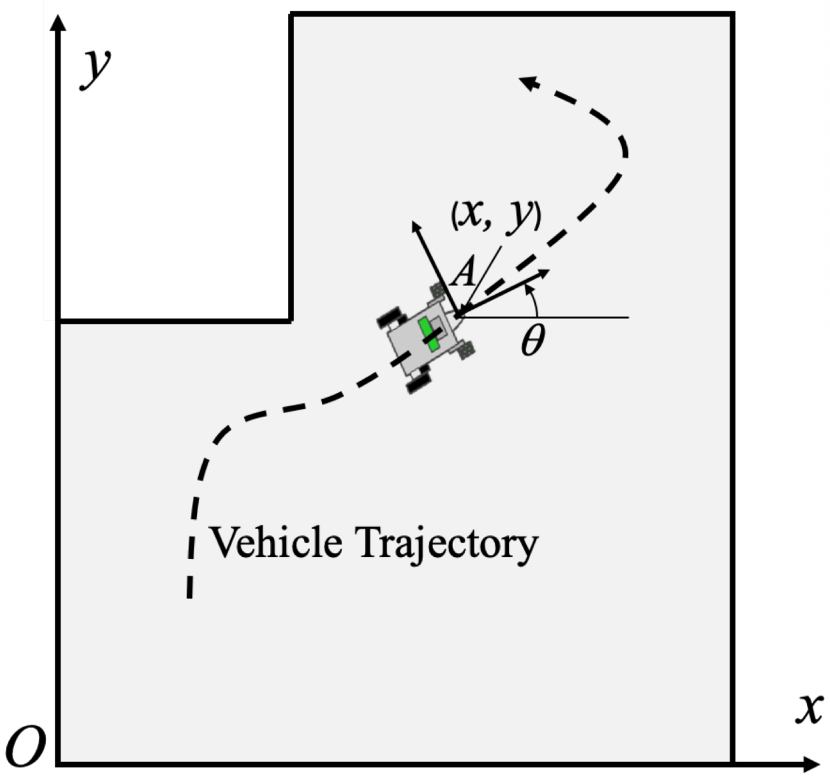
a) Download the first data file, LIDAR-Pedestrian-1. Read the file format instruction. Construct an Extended Kalman Filter to estimate the trajectory of a single pedestrian in the data file.

b) For the same data, implement Unscented Kalman Filter and discuss the difference.

c) Download the second data file, LIDAR-Pedestrian-2, and read the data format instruction. In this file, two pedestrians are walking in different directions. The two pedestrians get close to each other, and the LIDAR cannot detect one pedestrian behind the other for some time. Namely, occlusion occurs in the data. How can you handle this type of situation, where data are not available for several time frames?

Part 2: Robot Localization

$$Q_t = (\Sigma_S =)\operatorname{cov}(\Delta s_r, \Delta s_l) = \begin{pmatrix} k_r\left|\Delta s_r\right| & 0 \\ 0 & k_l\left|\Delta s_l\right| \end{pmatrix}$$



c) Obtain the predicted output measurement for the initial step ($t = 1$):

$$\hat{y}_1^{\,j} = \begin{pmatrix} \hat{\alpha}_1^{\,j} \\ \hat{r}_1^{\,j} \end{pmatrix} = \hat{y}_1^{\,j}(\hat{x}_{1|0}, \alpha_{map}^{j}, r_{map}^{j}),$$

for a set of walls, $j = 1, 2\ldots$, captured by the LIDAR at the initial pose.

d) Segment the LIDAR data into the individual walls by hand, and estimate the wall (feature) parameters, α and r. Obtain the initial measurement y_1^i for all the viewable walls at the initial pose.

e) Associate the predicted output $\hat{y}_1^j$ with measured output y_1^i; which i corresponds to which j. You can manually do this or use a distance measure.

f) Obtain the measurement error covariance R_1 and form the Kalman gain K_1.

g) Update the robot pose and repeat the process several times. Show the trajectory of the robot in the world coordinates.

SLAM beyond the standard Kalman Filter

- Dealing with non-Gaussian noise properties;
- Dealing with nonlinear process dynamics without linearization;
- Bayes filter – Nonlinear dynamics, non-Gaussian noise
- Particle Filter – Monte Carlo implementation of Bayes Filter
- Rao-Blackwell Filter – Combine Kalman Filter and Particle Filter