Unscented Kalman Filters

Slide credits: Cyrill Stachniss , Sebastian Thrun, Wolfram Burgard, Dieter Fox, and others

Robot Mapping

Unscented Kalman Filter

Cyrill Stachniss

Professor in Germany

- mapping class

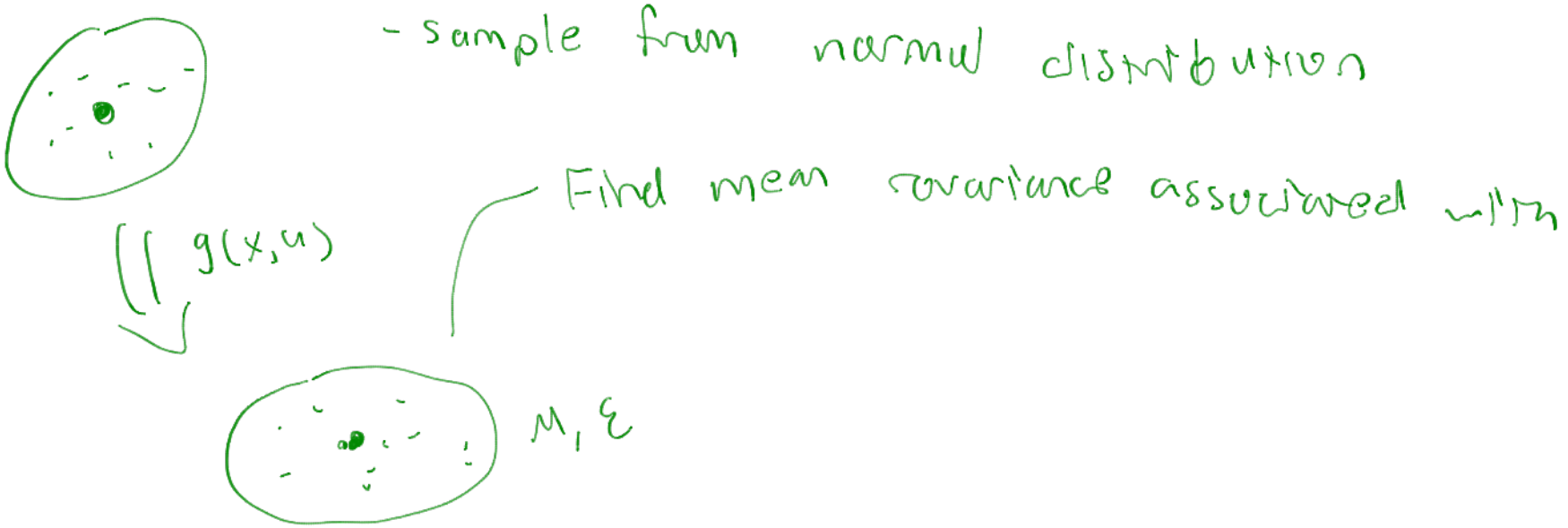
video of him teaching

http://ais.informatik.uni-freiburg.de/teaching/ws13/mapping/

KF, EKF and UKF

- Kalman filter requires linear models
- EKF linearizes via Taylor expansion

Is there a better way to linearize?



KF, EKF and UKF

- Kalman filter requires linear models
- EKF linearizes via Taylor expansion

Is there a better way to linearize?

Unscented Transform

Unscented Kalman Filter (UKF)

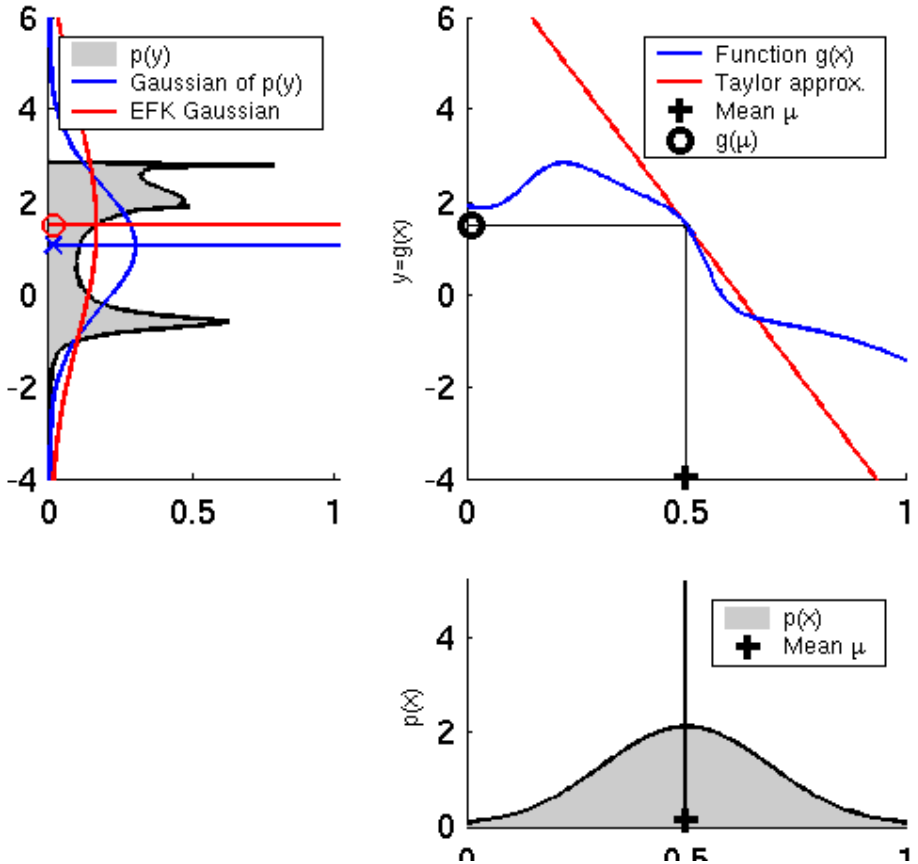
Motivation for UKF

Consider the following intuition: *With a fixed number of parameters it should be easier to approximate a given distribution than it is to approximate an arbitrary nonlinear function/transformation*. Following this intuition, the goal is to find a parameterization that captures the mean and covariance information while at the same time permitting the direct propagation of the information through an arbitrary set of nonlinear equations. This can be accomplished by generating a discrete distribution having the same first and second (and possibly higher) moments, where each point in the discrete approximation can be directly transformed. The mean and covariance of the transformed ensemble can then be computed as the estimate of the nonlinear transformation of the original distribution. More generally, the application of a given nonlinear transformation to a discrete distribution of points, computed so as to capture a set of known statistics of an unknown distribution, is referred to as an *unscented transformation*.

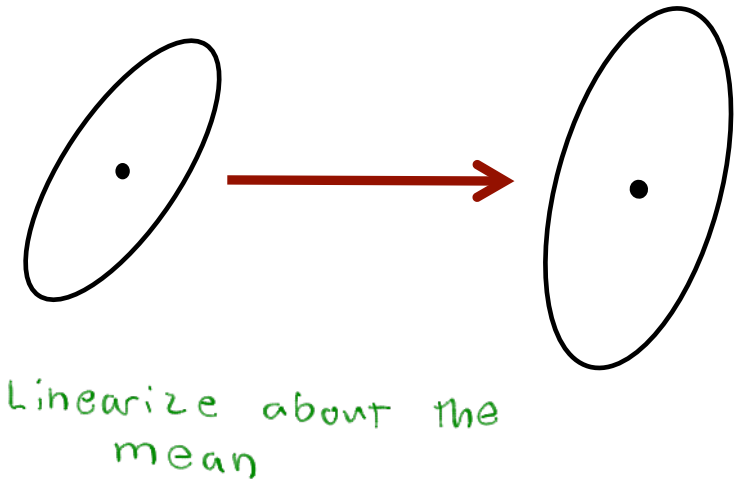
Jeffrey Uhlmann, 1994

Intuition behind UKF

Better to apply exact nonlinear function to an approximating probability distribution (UKF) than to apply an approximating linear function to to partial distribution information (mean and covariance estimate) (EKF)

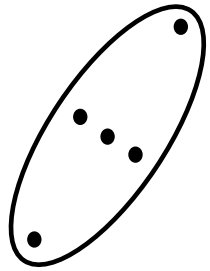


Taylor Approximation (EKF)



Linearization of the non-linear function through Taylor expansion

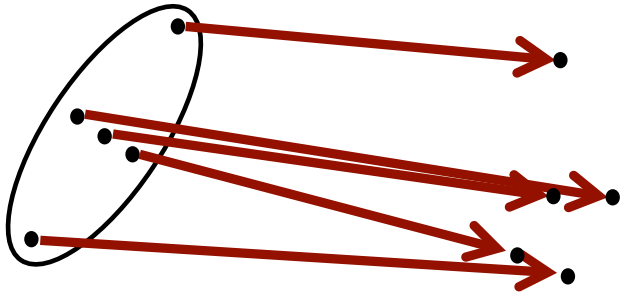
Unscented Transform



Sample the distribution in a certain way. One point is always the mean. 2 sets of points associated with each eigenvector or degree of freedom

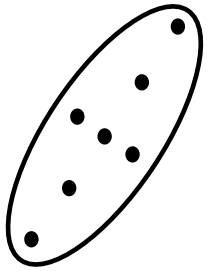
Compute a set of (so-called) sigma points

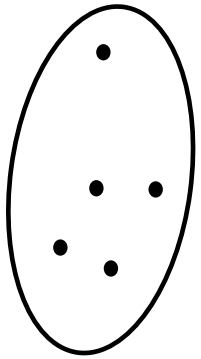
Unscented Transform



Transform each sigma point
through the non-linear function

Unscented Transform





Calculate mean and covariance from new set of points

Compute Gaussian from the transformed and weighted sigma points

Unscented Transform Overview

- Compute a set of sigma points
- Each sigma points has a weight
- Transform the point through the non-linear function
- Compute a Gaussian from weighted points

- Avoids to linearize **around the mean** as Taylor expansion (and EKF) does

Sigma Points

- How to choose the sigma points?
- How to set the weights?

What constraints do we have?

- Sigma points need same mean and standard deviation

Sigma Points Properties

- How to choose the sigma points?
- How to set the weights?
- Select $\mathcal{X}^{[i]}, w^{[i]}$ so that:

sigma points — weights

$$\sum_i w^{[i]} = 1$$

mean

$$\mu = \sum_i w^{[i]} \mathcal{X}^{[i]}$$

covariance

$$\Sigma = \sum_i w^{[i]} (\mathcal{X}^{[i]} - \mu)(\mathcal{X}^{[i]} - \mu)^T$$

- There is no unique solution for $\mathcal{X}^{[i]}, w^{[i]}$

Sigma Points

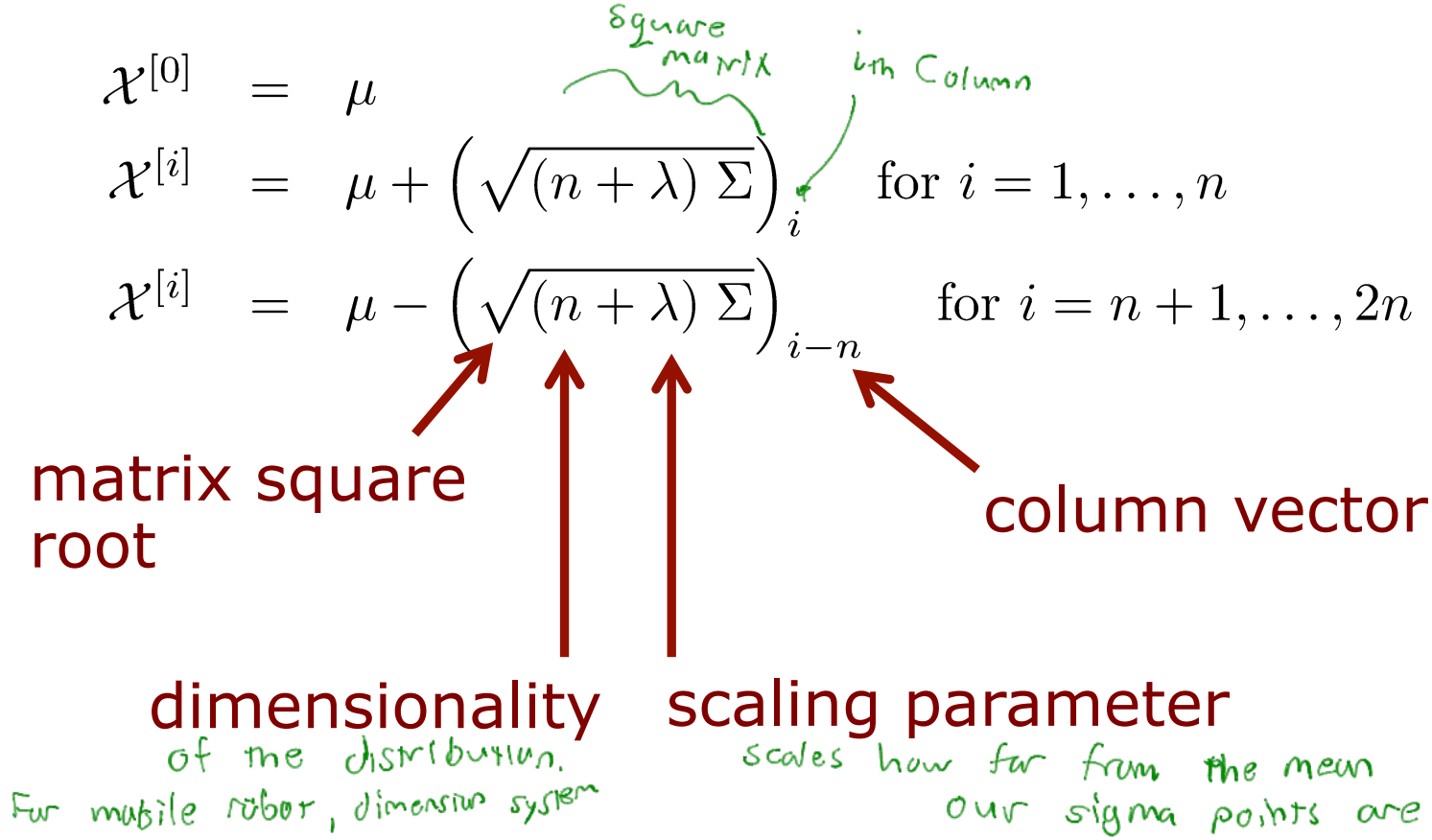
- Choosing the sigma points

$$\mathcal{X}^{[0]} = \mu$$

First sigma point is the mean

Sigma Points

- Choosing the sigma points



Cholesky Matrix Square Root

- Alternative definition of the matrix square root

$$L \text{ with } \Sigma = LL^T$$

- Result of the Cholesky decomposition
- Numerically stable solution
- Often used in UKF implementations
- ~~L and Σ have the same Eigenvectors~~ ?

true if (only) sigma points lie on the principle axes

$L^T = \text{chol}(\Sigma)$

$L = \text{chol}(\Sigma)'$

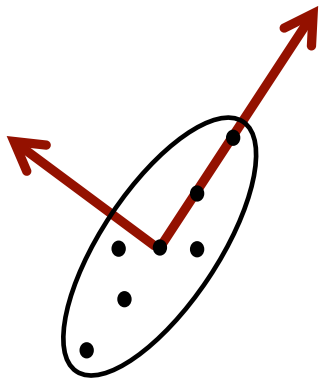
L needs to be Lower triangular factor matlab gives upper

Sigma Points and Eigenvectors

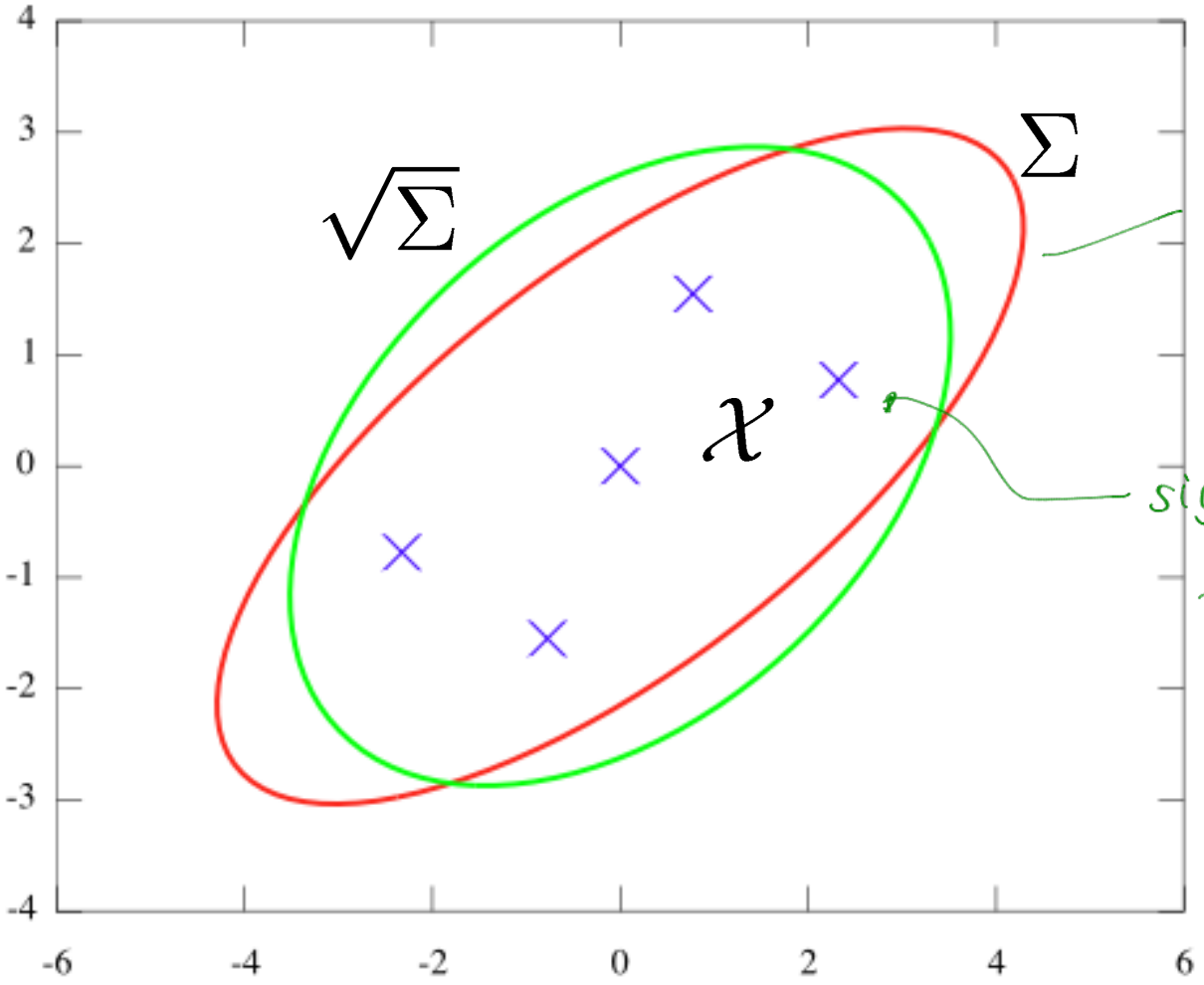
- Sigma point **can** but **do not have to** lie on the main axes of Σ

$$\mathcal{X}^{[i]} = \mu + \left(\sqrt{(n+\lambda)\,\Sigma}\right)_i \quad \text{for } i = 1, \ldots, n$$

$$\mathcal{X}^{[i]} = \mu - \left(\sqrt{(n+\lambda)\,\Sigma}\right)_{i-n} \quad \text{for } i = n+1, \ldots, 2n$$

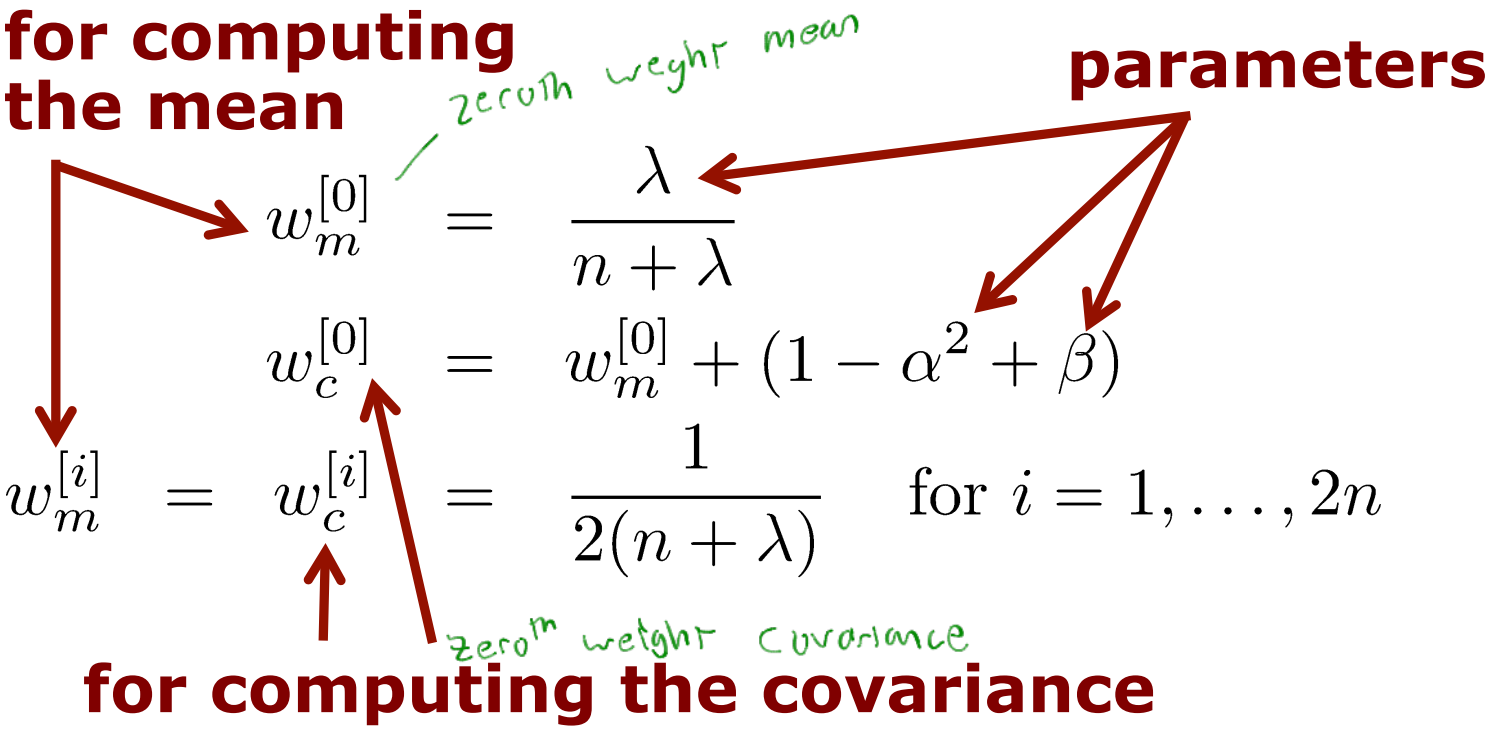


Sigma Points Example



Sigma Point Weights

- Weight sigma points



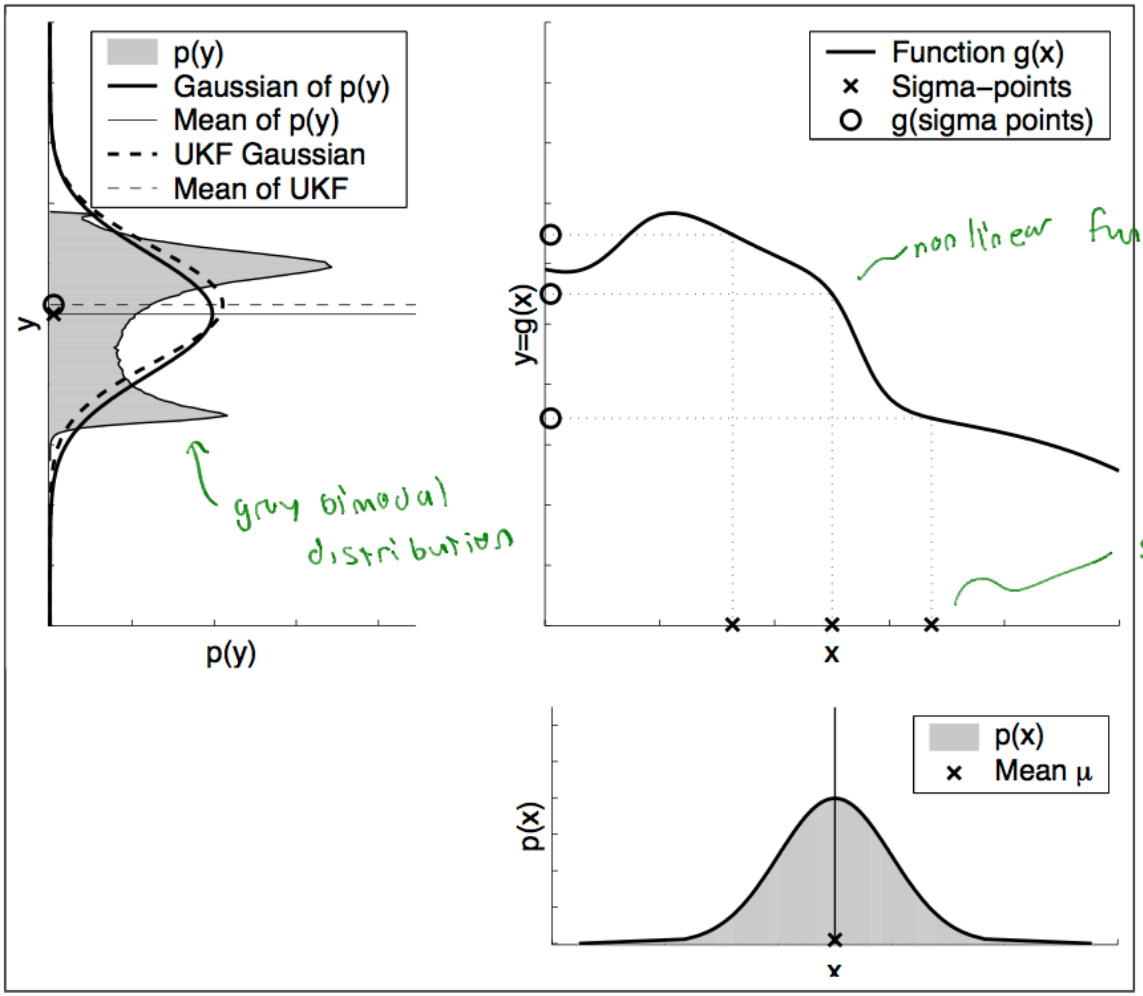
Recover the Gaussian

- Compute Gaussian from weighted and transformed points

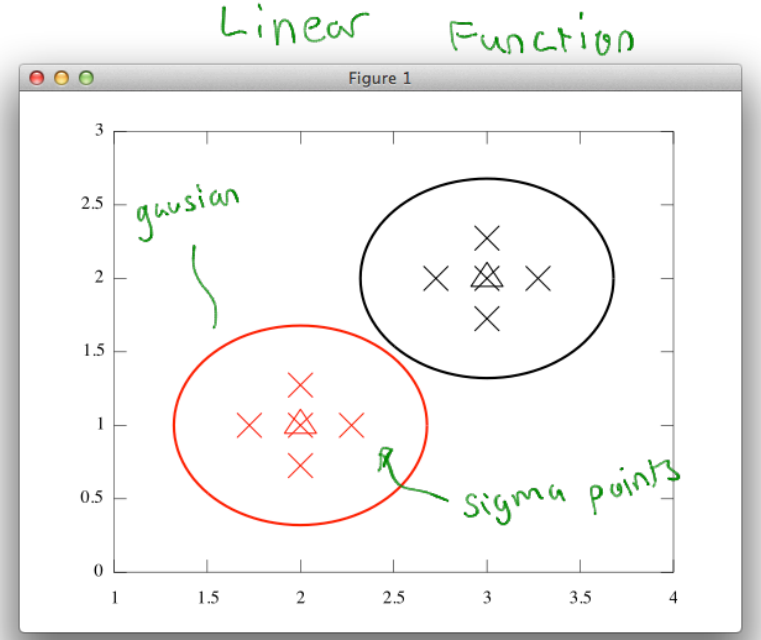
$$\mu' = \sum_{i=0}^{2n} w_m^{[i]} \, g(\mathcal{X}^{[i]})$$

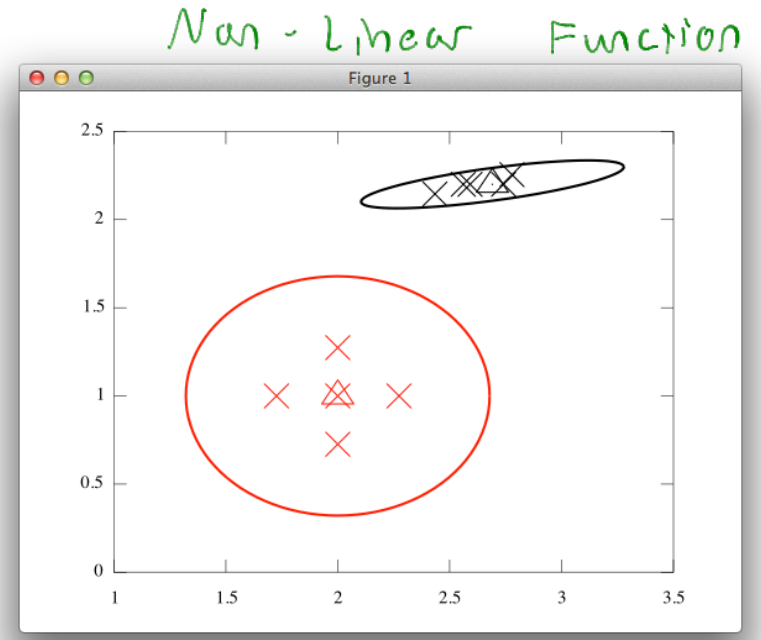
$$\Sigma' = \sum_{i=0}^{2n} w_c^{[i]} \, (g(\mathcal{X}^{[i]}) - \mu')(g(\mathcal{X}^{[i]}) - \mu')^T$$

Example


p(y)
Gaussian of p(y)
Mean of p(y)
UKF Gaussian
Mean of UKF
y
p(y)
gray bimodal distribution
Function g(x)
Sigma–points
g(sigma points)
y=g(x)
x
nonlinear function
sigma points of distribution below
p(x)
Mean μ
p(x)
x

Examples





$$g((x,y)^T) = \begin{pmatrix} x+1 \\ y+1 \end{pmatrix}^T$$

$$g((x,y)^T) = \begin{pmatrix} 1+x+\sin(2x)+\cos(y) \\ 2+0.2y \end{pmatrix}^T$$

Unscented Transform Summary

- Sigma points

$$\begin{aligned}
\mathcal{X}^{[0]} &= \mu \\
\mathcal{X}^{[i]} &= \mu + \left(\sqrt{(n+\lambda)\,\Sigma}\right)_i \qquad \text{for } i = 1, \ldots, n \\
\mathcal{X}^{[i]} &= \mu - \left(\sqrt{(n+\lambda)\,\Sigma}\right)_{i-n} \qquad \text{for } i = n+1, \ldots, 2n
\end{aligned}$$

- Weights

$$\begin{aligned}
w_m^{[0]} &= \frac{\lambda}{n+\lambda} \\
w_c^{[0]} &= w_m^{[0]} + (1 - \alpha^2 + \beta) \\
w_m^{[i]} = w_c^{[i]} &= \frac{1}{2(n+\lambda)} \qquad \text{for } i = 1, \ldots, 2n
\end{aligned}$$

UT Parameters

- Free parameters as there is no unique solution
- Scaled Unscented Transform suggests

read scaled unscented transform Lit to understand

$$\kappa \geq 0$$

$$\alpha \in (0, 1]$$

Influence how far the sigma points are away from the mean

$$\lambda = \alpha^2(n + \kappa) - n$$

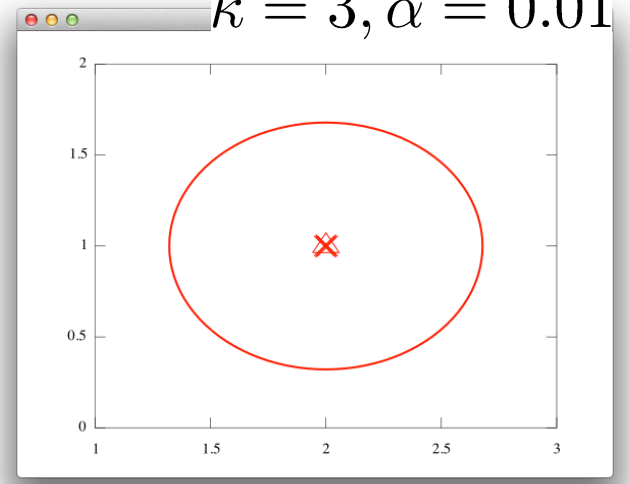
$$\beta = 2$$

Optimal choice for Gaussians

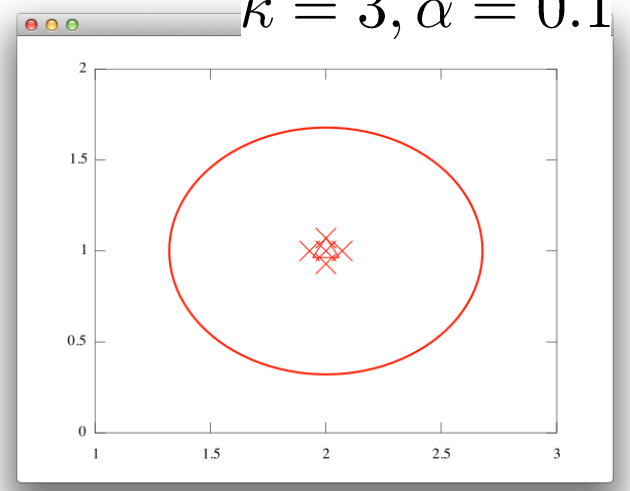
corresponds to 2 moments (mean & std)

Examples

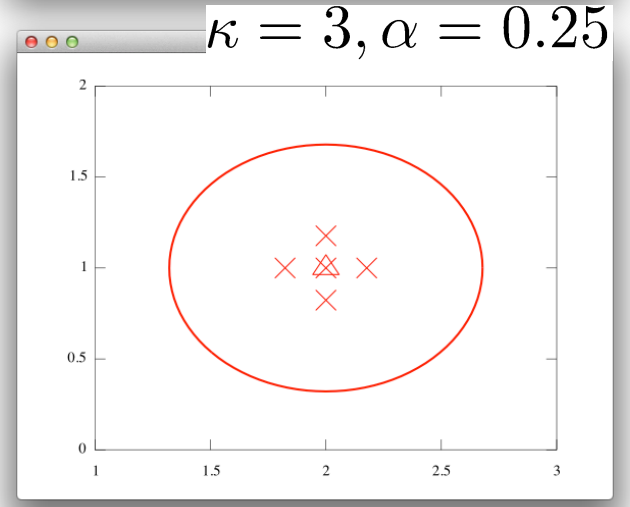
$\kappa = 3, \alpha = 0.01$



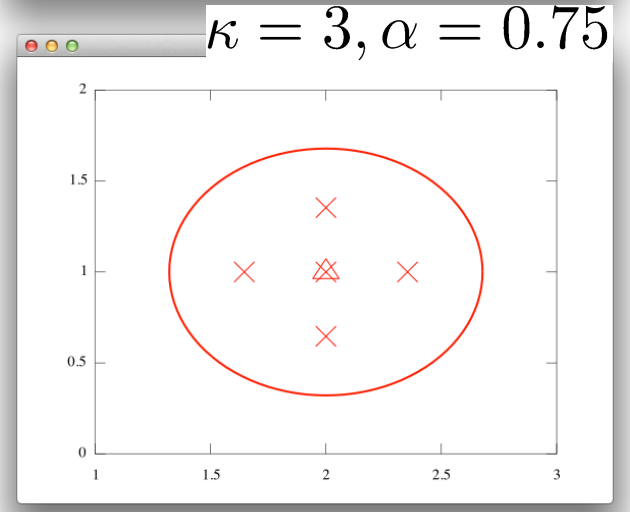
$\kappa = 3, \alpha = 0.1$



$\kappa = 3, \alpha = 0.25$

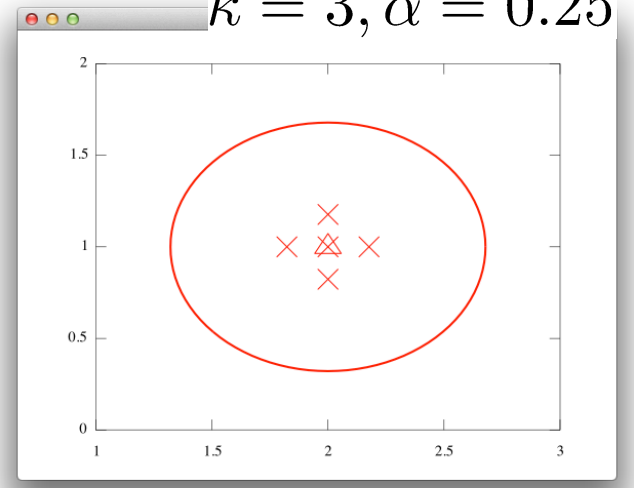


$\kappa = 3, \alpha = 0.75$

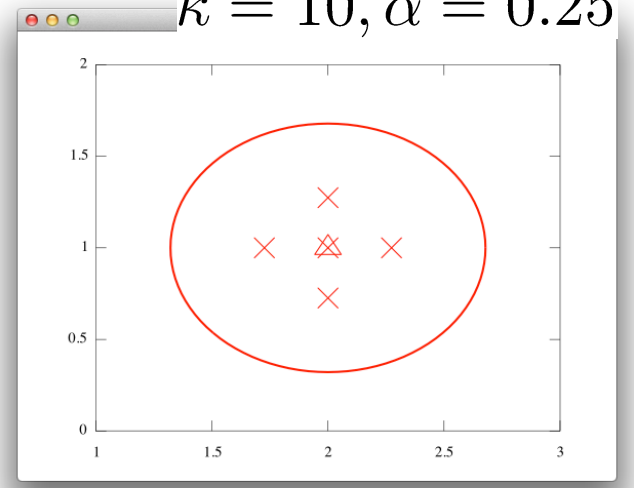


Examples

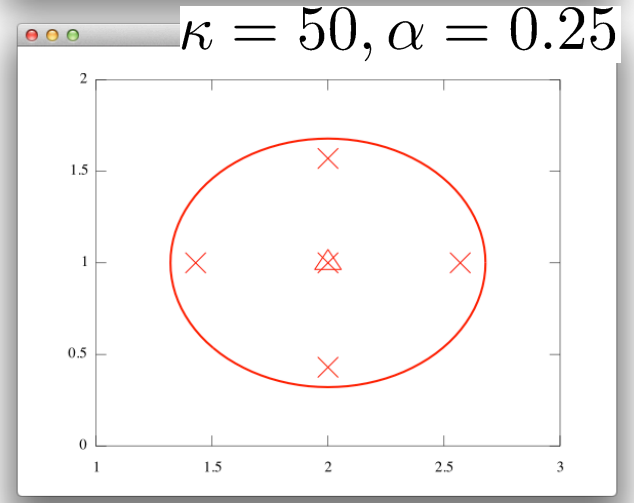
$\kappa = 3, \alpha = 0.25$



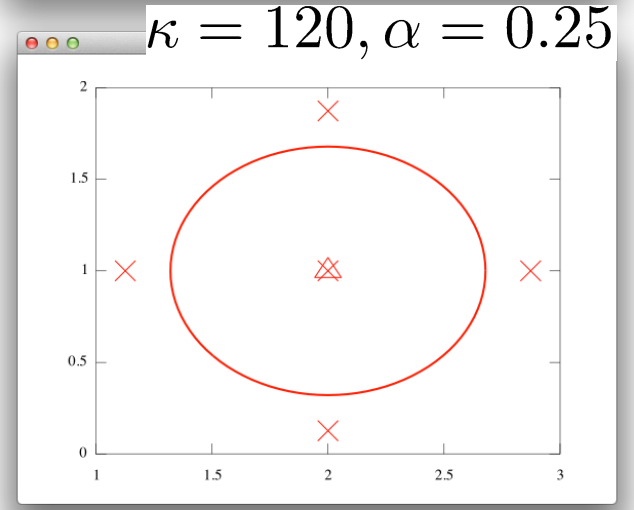
$\kappa = 10, \alpha = 0.25$



$\kappa = 50, \alpha = 0.25$



$\kappa = 120, \alpha = 0.25$



EKF Algorithm

1: **Extended_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\bar{\mu}_t = g(u_t, \mu_{t-1})$

3: $\bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$

4: $K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$

6: $\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

7: *return* μ_t, Σ_t

EKF to UKF – Prediction

Unscented

1: **~~Extended~~_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\bar{\mu}_t =$ replace this by sigma point

3: $\bar{\Sigma}_t =$ propagation of the motion

4: $K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$

6: $\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

7: *return* μ_t, Σ_t

UKF Algorithm – Prediction

1: **Unscented_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\mathcal{X}_{t-1} = (\mu_{t-1} \quad \mu_{t-1} + \sqrt{(n+\lambda)\Sigma_{t-1}} \quad \mu_{t-1} - \sqrt{(n+\lambda)\Sigma_{t-1}})$

3: $\bar{\mathcal{X}}_t^* = g(u_t, \mathcal{X}_{t-1})$

4: $\bar{\mu}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{X}}_t^{*[i]}$

5: $\bar{\Sigma}_t = \left(\sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)(\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)^T \right) + R_t$

EKF to UKF – Correction

Unscented

1: ~~Extended~~ **Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\bar{\mu}_t =$ replace this by sigma point

3: $\bar{\Sigma}_t =$ propagation of the motion

use sigma point propagation for the expected observation and Kalman gain

5: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$

6: $\Sigma_t = \bar{\Sigma}_t - K_t\, S_t\, K_t^T$

7: $return\ \mu_t, \Sigma_t$

UKF Algorithm – Correction (1)

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \quad \bar{\mu}_t + \sqrt{(n+\lambda)\bar{\Sigma}_t} \quad \bar{\mu}_t - \sqrt{(n+\lambda)\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

uncertainty in measurement from uncertainty in propagation step

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

helps fill role of Jacobian. How measurements are influenced by state

11: $K_t = \bar{\Sigma}_t^{x,z} \; S_t^{-1}$

EKF

$z = h(x, n)$

$H = \frac{\partial h}{\partial x}$

UKF Algorithm – Correction (1)

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \quad \bar{\mu}_t + \sqrt{(n+\lambda)\bar{\Sigma}_t} \quad \bar{\mu}_t - \sqrt{(n+\lambda)\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

11: $K_t = \bar{\Sigma}_t^{x,z} \; S_t^{-1}$

$$K_t = \overbrace{\bar{\Sigma}_t \; H_t^T}^{\bar{\Sigma}_t^{x,z}} \overbrace{(H_t \; \bar{\Sigma}_t \; H_t^T + Q_t)^{-1}}^{S_t}$$

(from EKF)

Kalman gain calc from EKF

model process uncertainty maps to measurement uncertainty

UKF Algorithm – Correction (2)

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \quad \bar{\mu}_t + \sqrt{(n+\lambda)\bar{\Sigma}_t} \quad \bar{\mu}_t - \sqrt{(n+\lambda)\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

11: $K_t = \bar{\Sigma}_t^{x,z} \; S_t^{-1}$

12: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$

13: $\Sigma_t = \bar{\Sigma}_t - K_t \; S_t \; K_t^T$

14: *return* μ_t, Σ_t

UKF Algorithm – Correction (2)

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \quad \bar{\mu}_t + \sqrt{(n+\lambda)\bar{\Sigma}_t} \quad \bar{\mu}_t - \sqrt{(n+\lambda)\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

11: $K_t = \bar{\Sigma}_t^{x,z} \; S_t^{-1}$

12: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$

13: $\Sigma_t = \bar{\Sigma}_t - K_t \; S_t \; K_t^T$

14: *return* μ_t, Σ_t

EKF

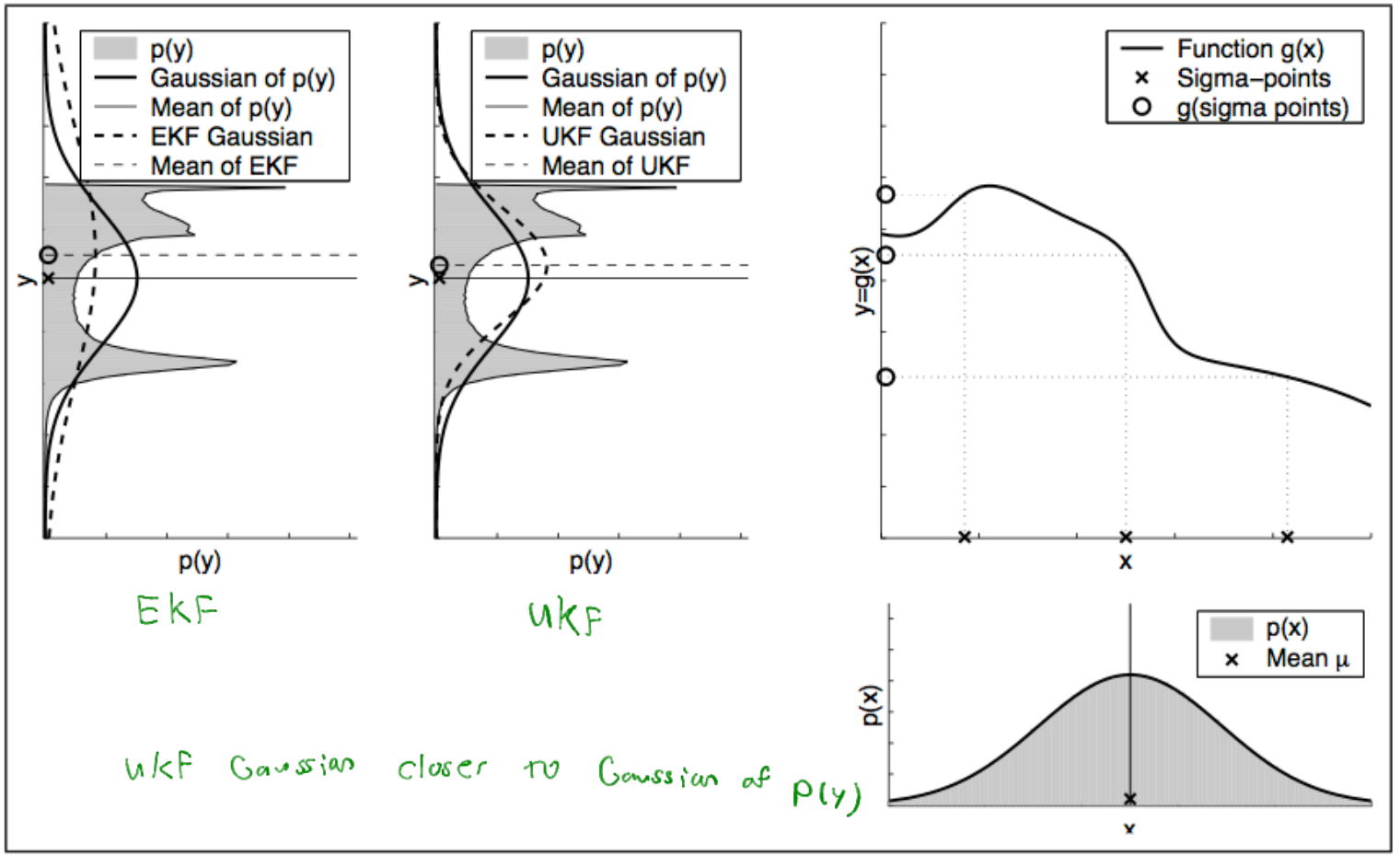
$$
\begin{aligned}
\Sigma_t &= (I - K_t H_t)\bar{\Sigma}_t \\
&= \bar{\Sigma}_t - K_t H_t \bar{\Sigma}_t \\
&= \bar{\Sigma}_t - K_t \left(\Sigma^{x,z}\right)^T \\
&= \bar{\Sigma}_t - K_t \left(\Sigma^{x,z} s_t^{-1} S_t\right)^T \\
&= \bar{\Sigma}_t - K_t \left(K_t S_t\right)^T \\
&= \bar{\Sigma}_t - K_t S_t^T K_t^T \\
&= \bar{\Sigma}_t - K_t S_t K_t^T
\end{aligned}
$$

(see next slide)

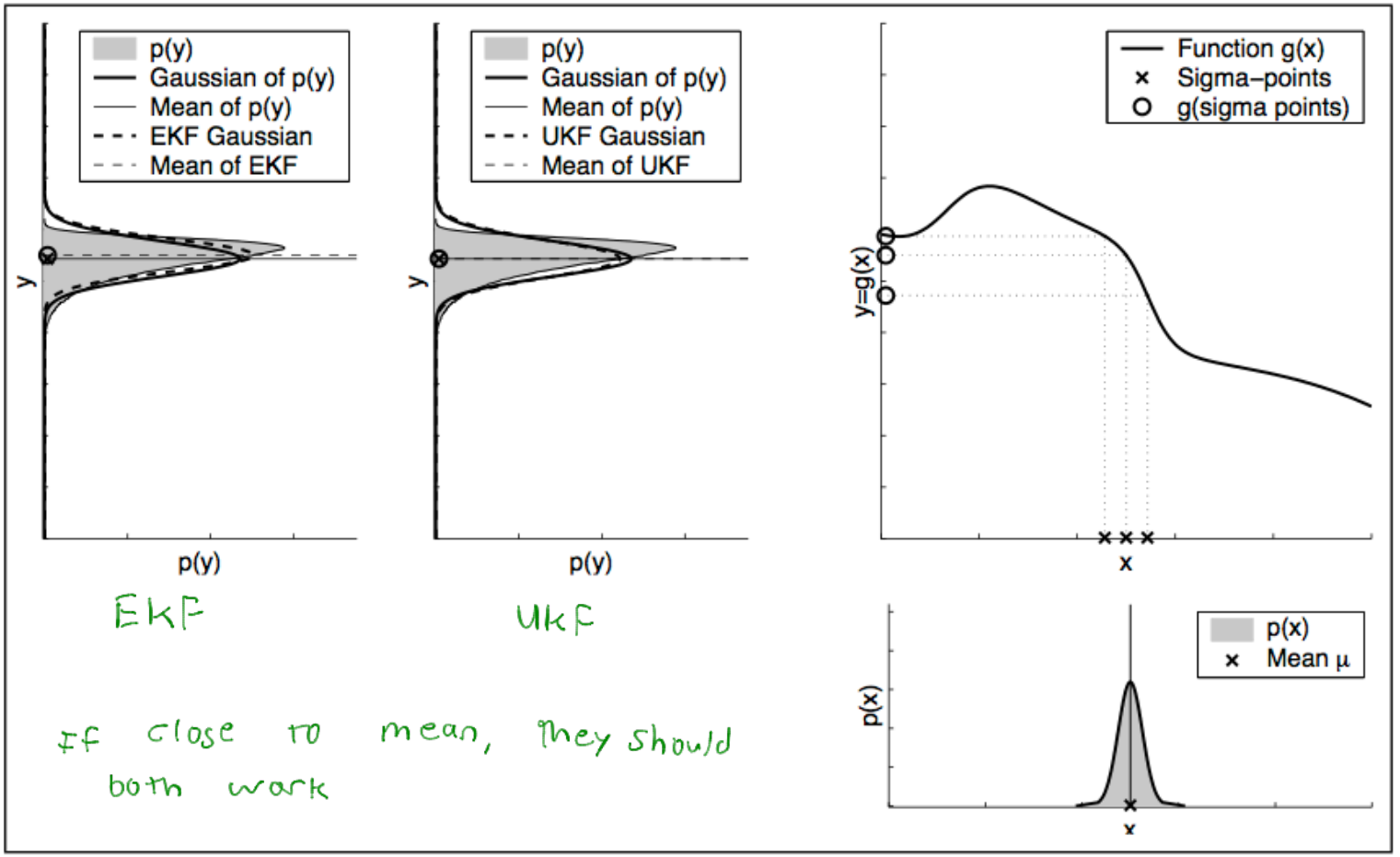
From EKF to UKF – Computing the Covariance

$$
\begin{aligned}
\Sigma_t &= (I - K_t H_t)\bar{\Sigma}_t \leftarrow \text{EKF} \\
&= \bar{\Sigma}_t - K_t H_t \bar{\Sigma}_t \\
&= \bar{\Sigma}_t - K_t \left(\bar{\Sigma}^{x,z}\right)^T \\
&= \bar{\Sigma}_t - K_t \left(\bar{\Sigma}^{x,z} S_t^{-1} S_t\right)^T \\
&= \bar{\Sigma}_t - K_t \left(K_t S_t\right)^T \\
&= \bar{\Sigma}_t - K_t S_t^T K_t^T \quad S \text{ is symmetric} \\
&= \bar{\Sigma}_t - K_t S_t K_t^T \leftarrow \text{UKF}
\end{aligned}
$$

UKF vs. EKF


p(y)
Gaussian of p(y)
Mean of p(y)
EKF Gaussian
Mean of EKF
y
p(y)
EKF
p(y)
Gaussian of p(y)
Mean of p(y)
UKF Gaussian
Mean of UKF
y
p(y)
UKF
Function g(x)
Sigma–points
g(sigma points)
y=g(x)
x
p(x)
Mean μ
p(x)
x
ukf Gaussian closer to Gaussian of p(y)

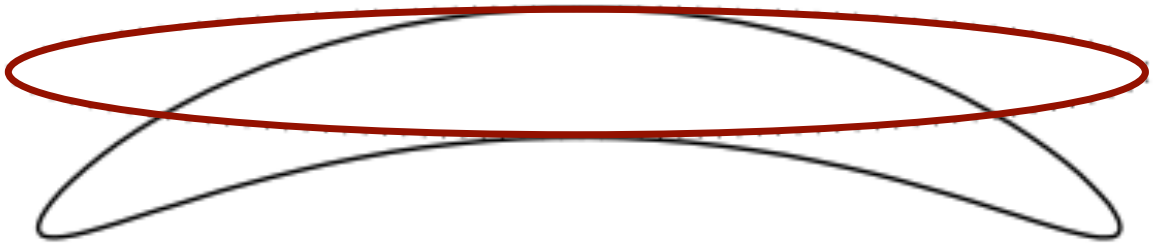
UKF vs. EKF (Small Covariance)



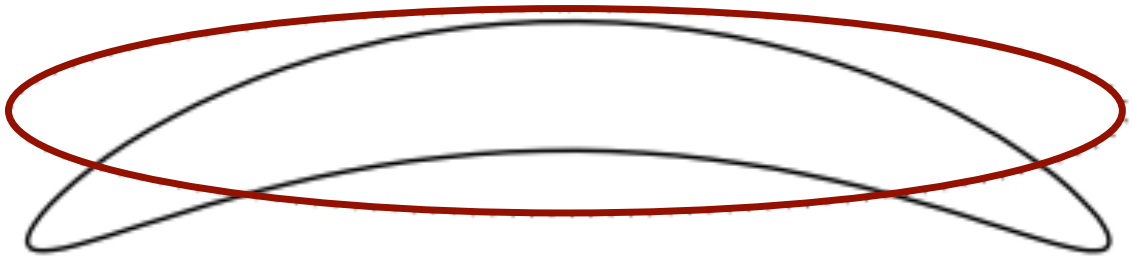
UKF vs. EKF – Banana Shape

Error due to heading

EKF approximation



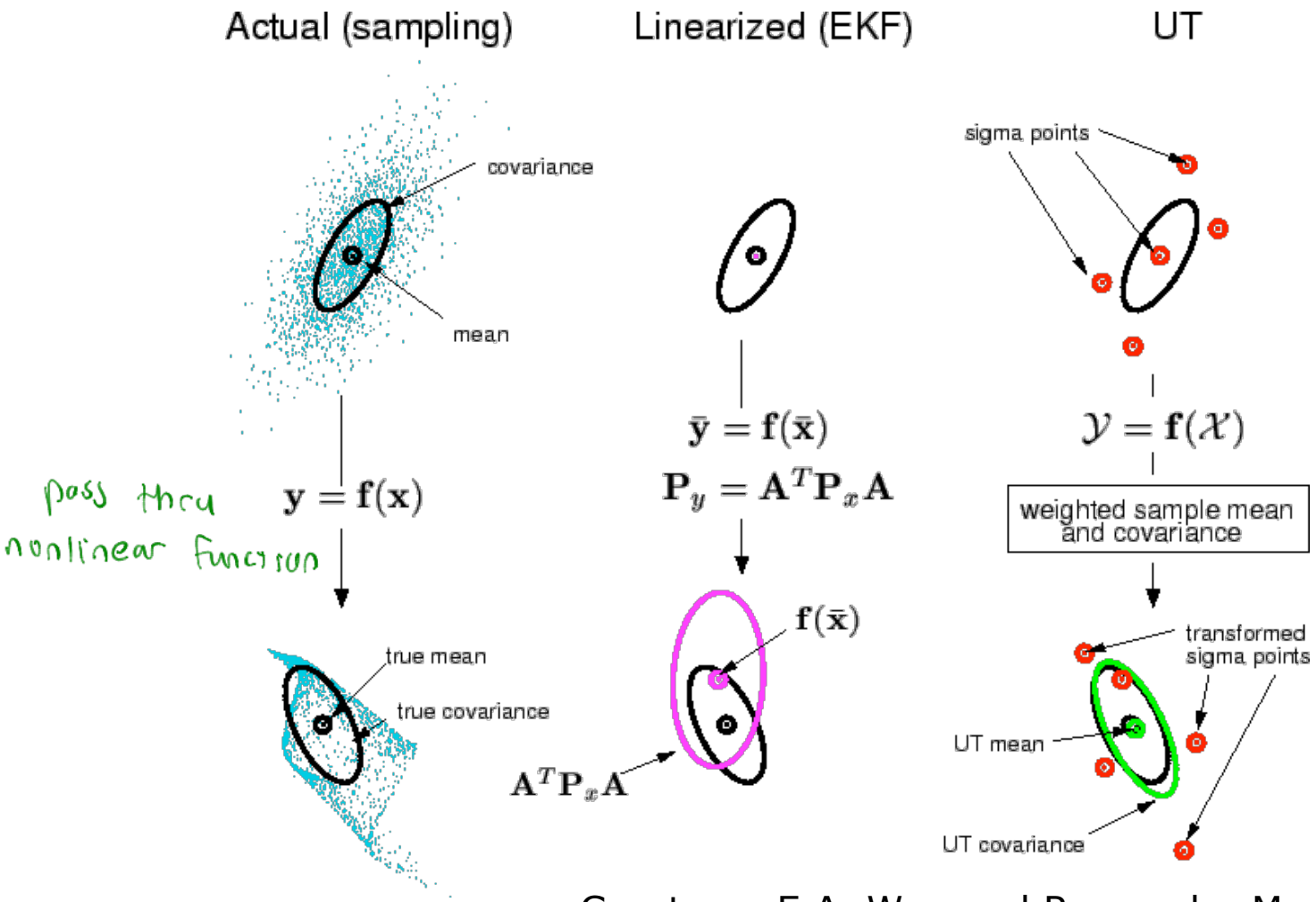
UKF approximation



class project?

N^2 space?

UKF vs. EKF



Courtesy: E.A. Wan and R. van der Merwe

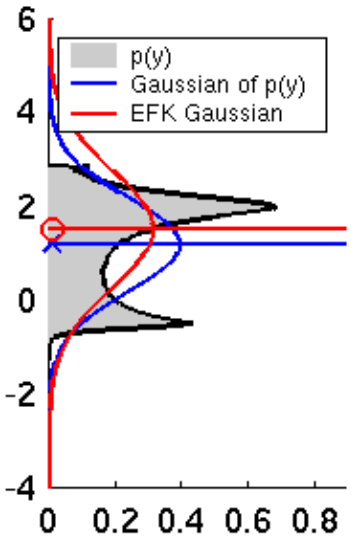
UT/UKF Summary

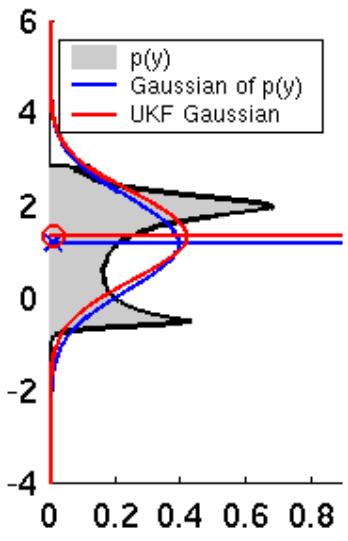
- Unscented transforms as an alternative to linearization
- UT is a better approximation than Taylor expansion
- UT uses sigma point propagation
- Free parameters in UT
- UKF uses the UT in the prediction and correction step

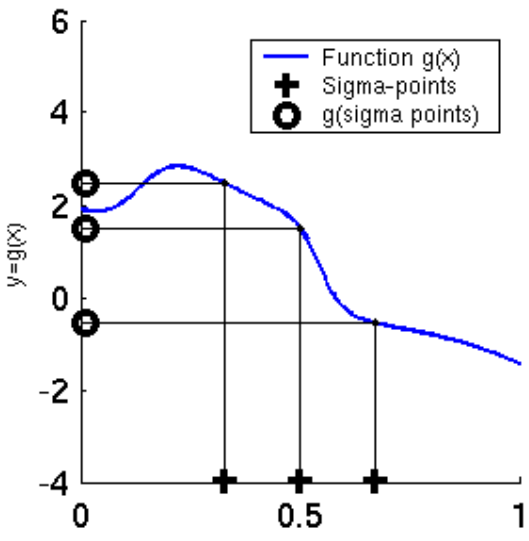
UKF vs. EKF

- Same results as EKF for linear models
- Better approximation than EKF for non-linear models
- Differences often “somewhat small”
- No Jacobians needed for the UKF
- Same complexity class
- Slightly slower than the EKF
- Still restricted to Gaussian distributions

Linearization via Unscented Transform

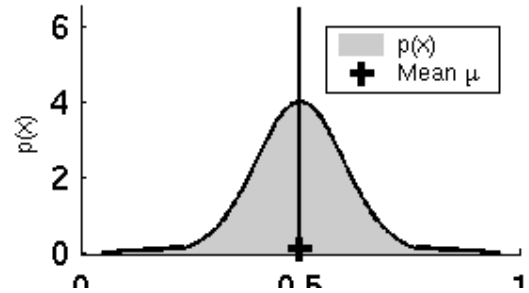




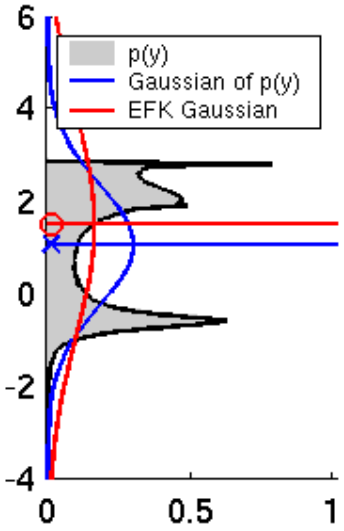


EKF

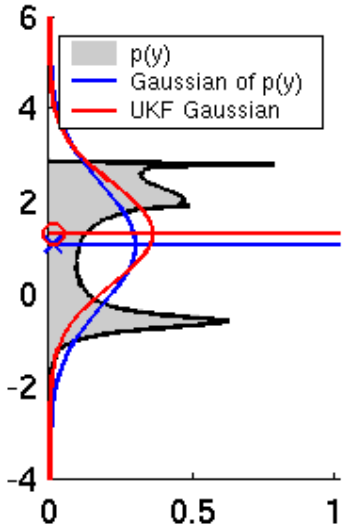
UKF



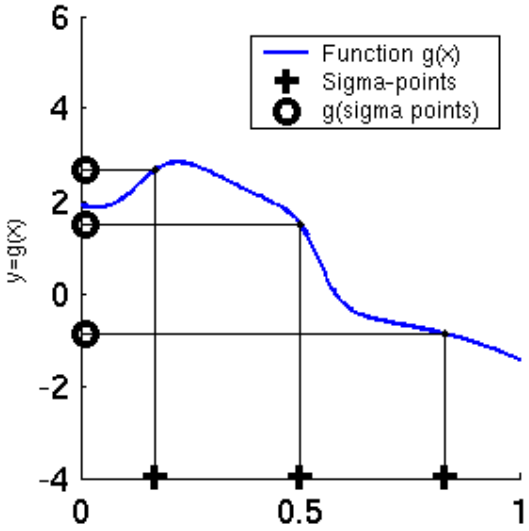
UKF Sigma-Point Estimate (2)

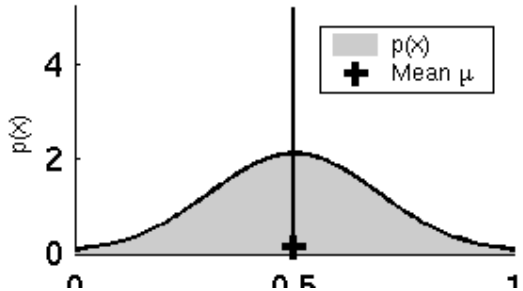


EKF

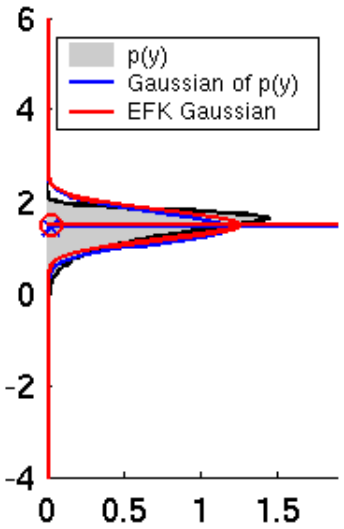


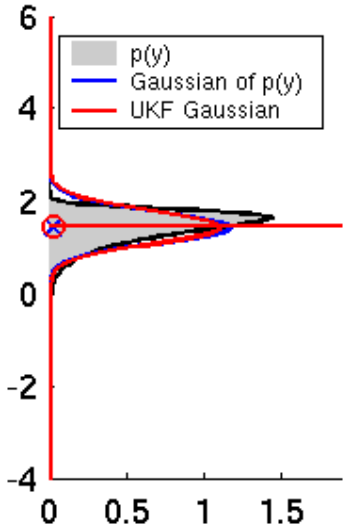
UKF

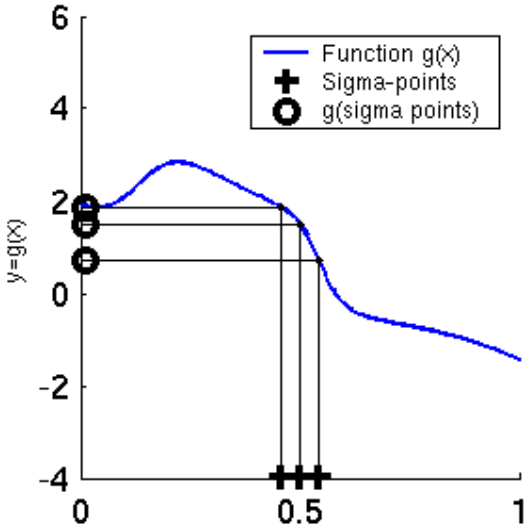




UKF Sigma-Point Estimate (3)

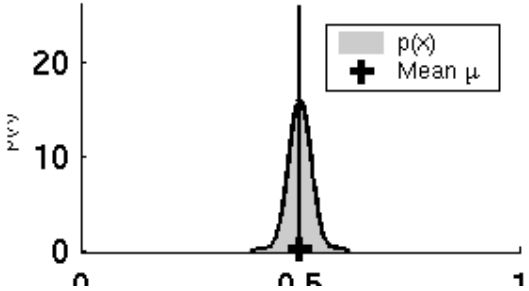






EKF

UKF



Unscented Transform

Sigma points

$$\chi^0 = \mu$$

$$\chi^i = \mu \pm \left(\sqrt{(n+\lambda)\Sigma}\right)_i$$

Weights

$$w_m^0 = \frac{\lambda}{n+\lambda} \qquad w_c^0 = \frac{\lambda}{n+\lambda} + (1-\alpha^2+\beta)$$

$$w_m^i = w_c^i = \frac{1}{2(n+\lambda)} \qquad \text{for } i = 1,\ldots,2n$$

Pass sigma points through nonlinear function

$$\psi^i = g(\chi^i)$$

Recover mean and covariance

$$\mu' = \sum_{i=0}^{2n} w_m^i \psi^i$$

$$\Sigma' = \sum_{i=0}^{2n} w_c^i (\psi^i - \mu)(\psi^i - \mu)^T$$

UKF localization algorithm

1: **Algorithm UKF_localization**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m)$**:**

Generate augmented mean and covariance

2: $M_t = \begin{pmatrix} \alpha_1 v_t^2 + \alpha_2 \omega_t^2 & 0 \\ 0 & \alpha_3 v_t^2 + \alpha_4 \omega_t^2 \end{pmatrix}$ control uncertainty

3: $Q_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\phi^2 \end{pmatrix}$ measurement uncertainty

augmented state estimate

state — mean control noise — mean measurement noise

4: $\mu_{t-1}^a = (\mu_{t-1}^T \quad (0 \;\; 0)^T \quad (0 \;\; 0)^T)^T$ augmented state (7×1), $L = 7$

5: $\Sigma_{t-1}^a = \begin{pmatrix} \Sigma_{t-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & M_t & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & Q_t \end{pmatrix}$ covariance of augmented state estimation error (7×7)

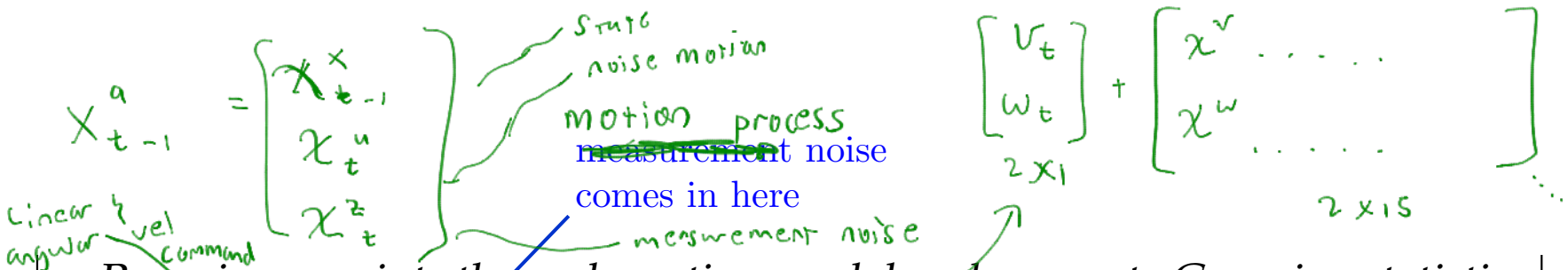
covariance state — covariance control noise — covariance measurement

UKF localization algorithm

states * 2 + 1 = # sigma points

Generate sigma points 15 sigma points – $2L + 1$

6: $\mathcal{X}^a_{t-1} = (\mu^a_{t-1} \quad \mu^a_{t-1} + \gamma\sqrt{\Sigma^a_{t-1}} \quad \mu^a_{t-1} - \gamma\sqrt{\Sigma^a_{t-1}})$ (7×15)



Pass sigma points through motion model and compute Gaussian statistics

7: $\bar{\mathcal{X}}^x_t = g(u_t + \mathcal{X}^u_t, \mathcal{X}^x_{t-1})$ propagate state sigma points from $(t-1)$ to t

8: $\bar{\mu}_t = \sum_{i=0}^{2L} w_i^{(m)} \bar{\mathcal{X}}^x_{i,t}$ weighted mean of state sigma points

9: $\bar{\Sigma}_t = \sum_{i=0}^{2L} w_i^{(c)} (\bar{\mathcal{X}}^x_{i,t} - \bar{\mu}_t)(\bar{\mathcal{X}}^x_{i,t} - \bar{\mu}_t)^T$ weighted covariance of state sigma points

recover mean and covariance of state

UKF localization algorithm

state sigma points

sensor noise comes in here

Predict observations at sigma points and compute Gaussian statistics

10: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t^x) + \mathcal{X}_t^z$ — calculate measurement sigma points

11: $\hat{z}_t = \sum_{i=0}^{2L} w_i^{(m)} \bar{\mathcal{Z}}_{i,t}$ — weighted mean of measurement sigma points

predicted measurement

covariance of predicted measurement

12: $S_t = \sum_{i=0}^{2L} w_i^{(c)} (\bar{\mathcal{Z}}_{i,t} - \hat{z}_t)(\bar{\mathcal{Z}}_{i,t} - \hat{z}_t)^T$

13: $\Sigma_t^{x,z} = \sum_{i=0}^{2L} w_i^{(c)} (\bar{\mathcal{X}}_{i,t}^x - \bar{\mu}_t)(\bar{\mathcal{Z}}_{i,t} - \hat{z}_t)^T$ — cross covariance between state and measurement

captures effect of model uncertainty on measurement uncertainty

describes how much of the uncertainty $\bar{\mathcal{Z}}_t$ comes from uncertainty $\bar{\chi}_t^x$

UKF localization algorithm

Update mean and covariance

14: $K_t = \Sigma_t^{x,z} \; S_t^{-1}$ Kalman gain — Kalman gain

15: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$ innovation — measurement update or correction

16: $\Sigma_t = \bar{\Sigma}_t - K_t \; S_t \; K_t^T$ prediction covariance — update measurement covariance

Kalman gain is big when measurement uncertainty is low and when correlation between model uncertainty and measurement uncertainty is high

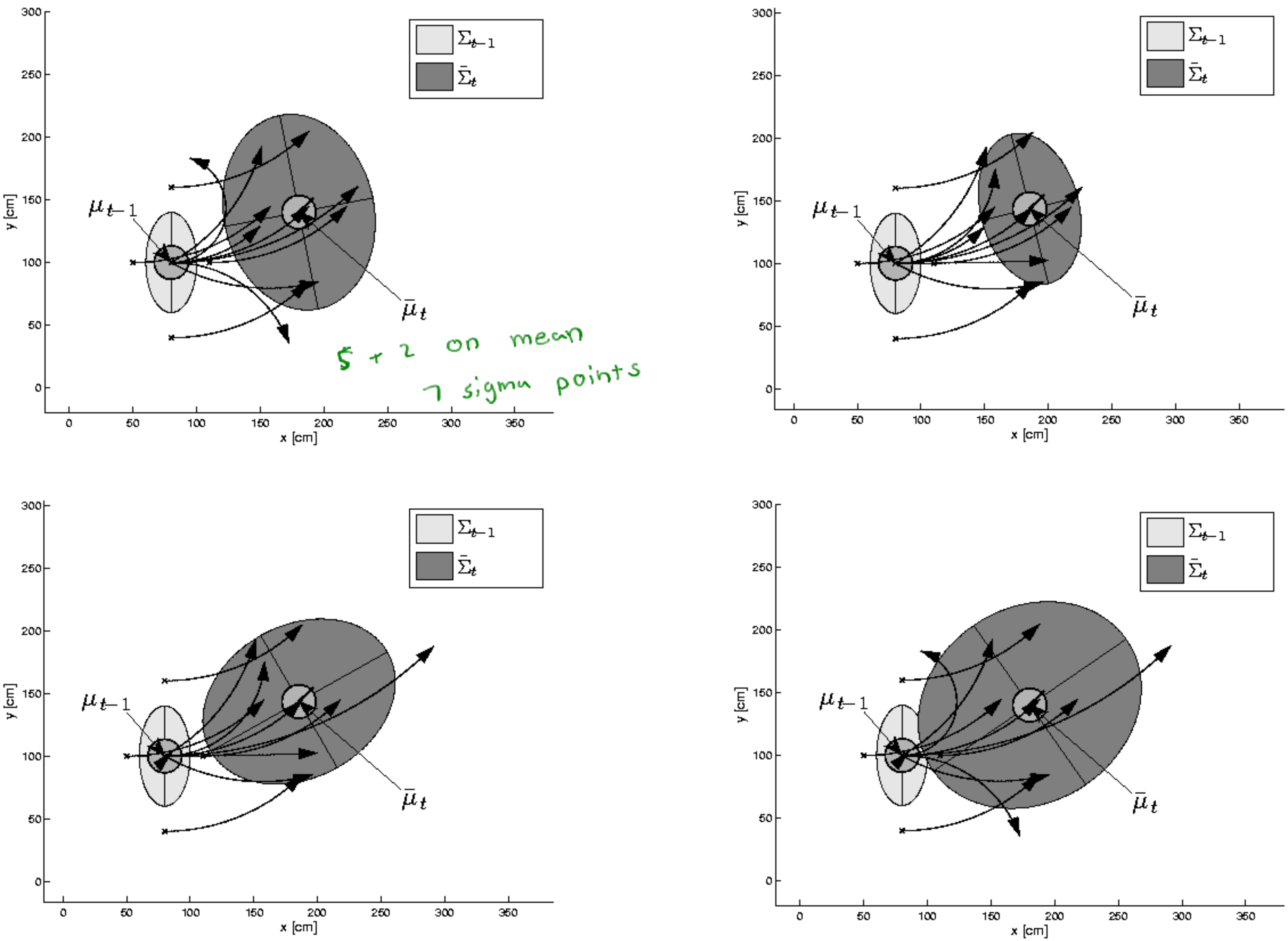
Kalman gain is small when measurement uncertainty is high and when correlation between model uncertainty and measurement uncertainty is low

17: $p_{z_t} = \det(2\pi S_t)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_t - \hat{z}_t)^T S_t^{-1}(z_t - \hat{z}_t)\right\}$

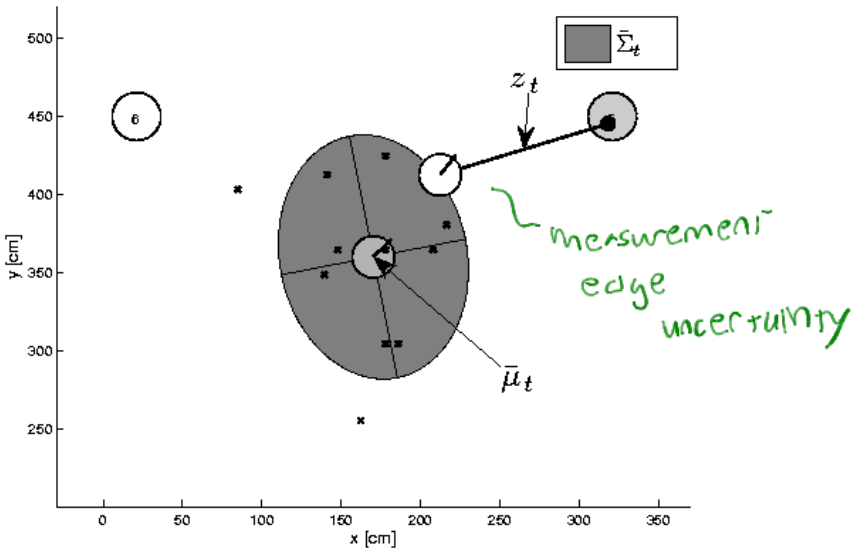
18: *return* μ_t, Σ_t, p_{z_t}

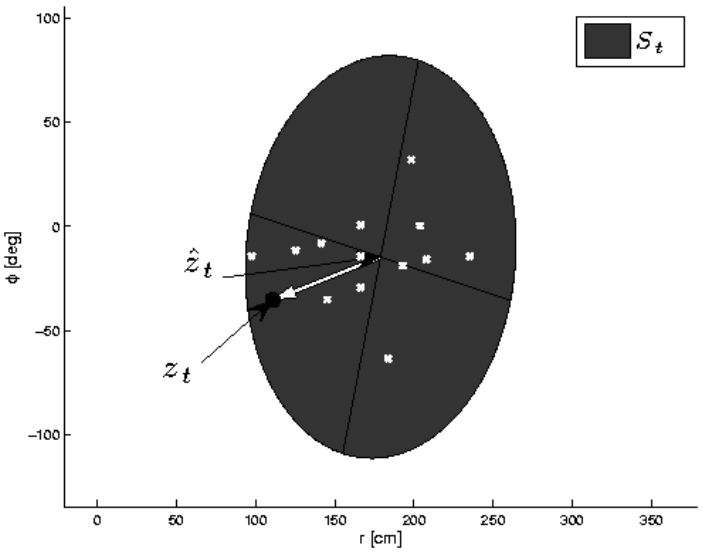
every time update measurement, need to create new sigma points otherwise filter will diverge

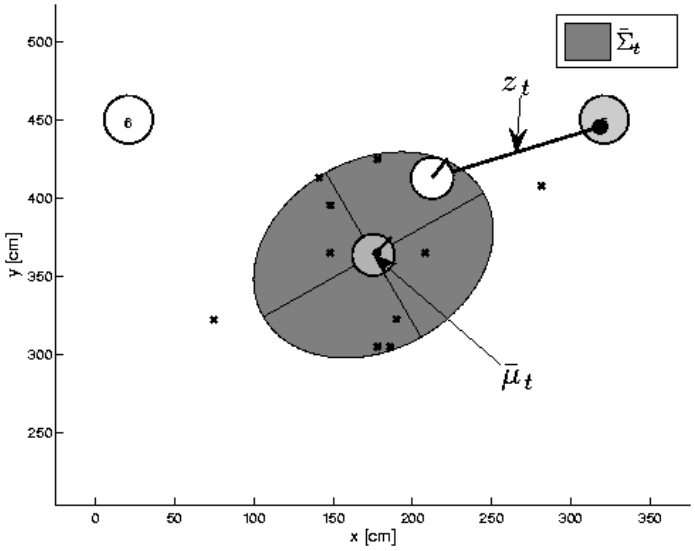
UKF prediction step

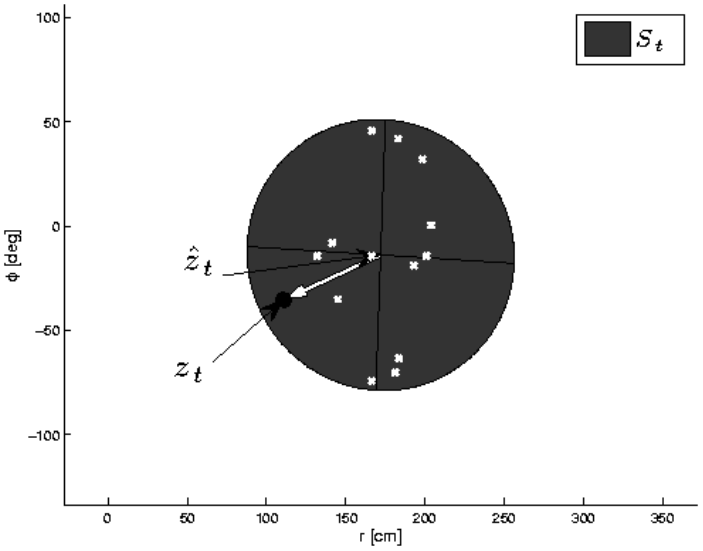


UKF observation prediction step

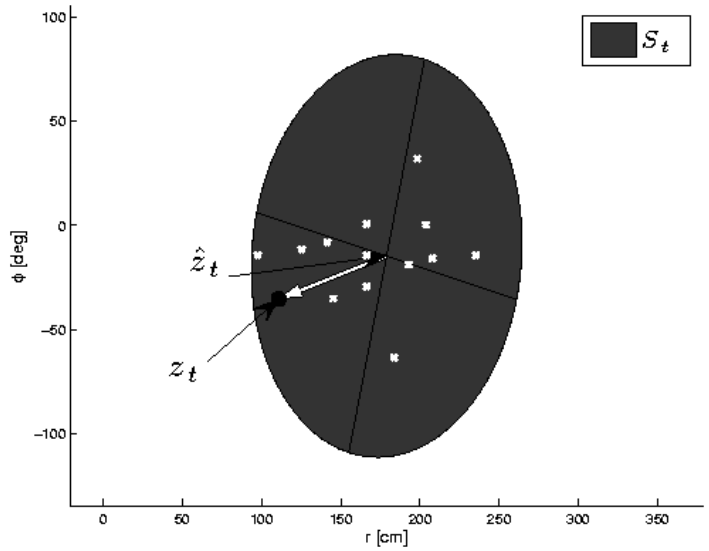


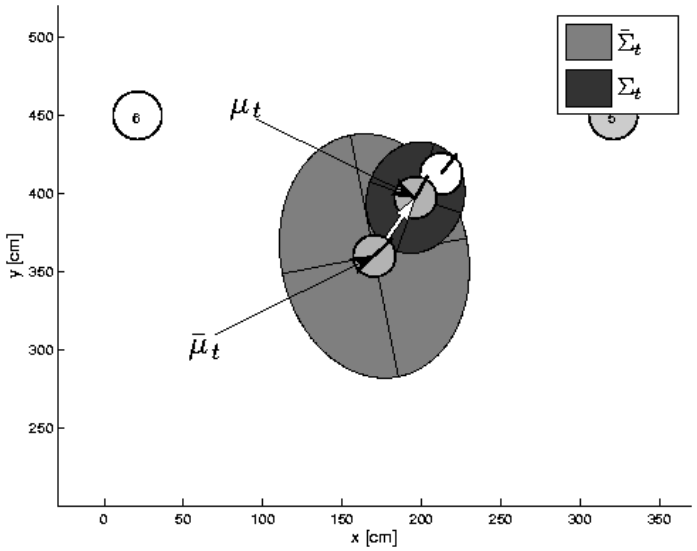


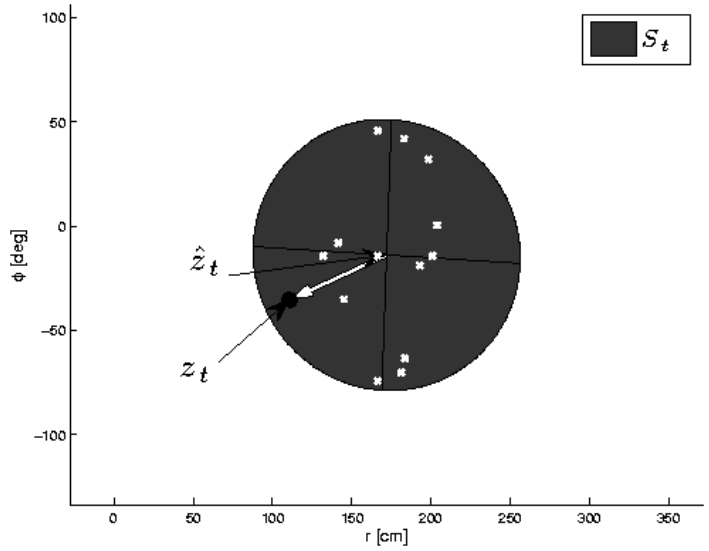


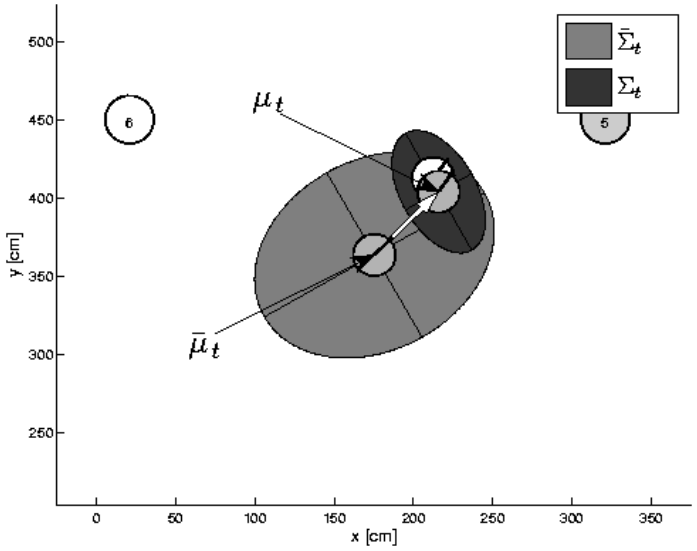


UKF correction step

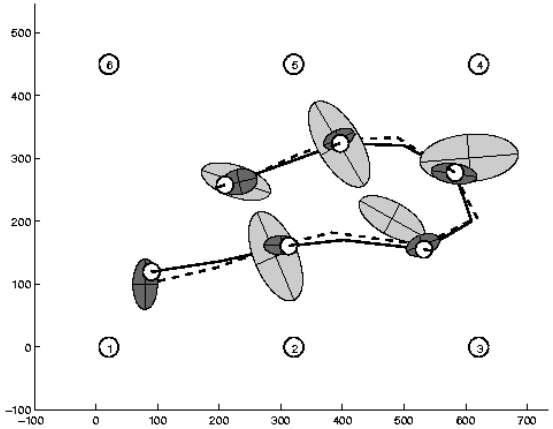




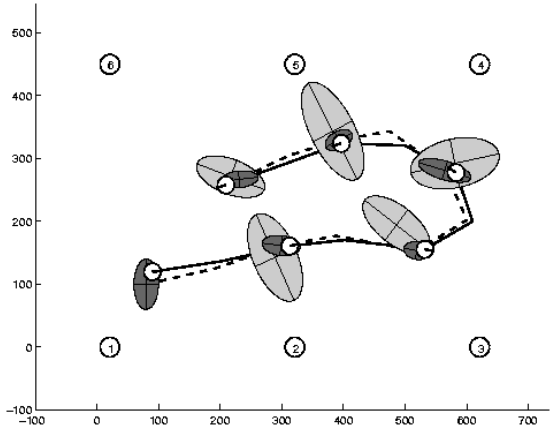




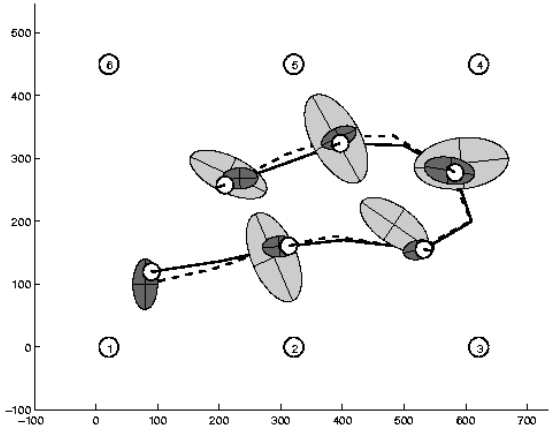
Estimation sequence



EKF

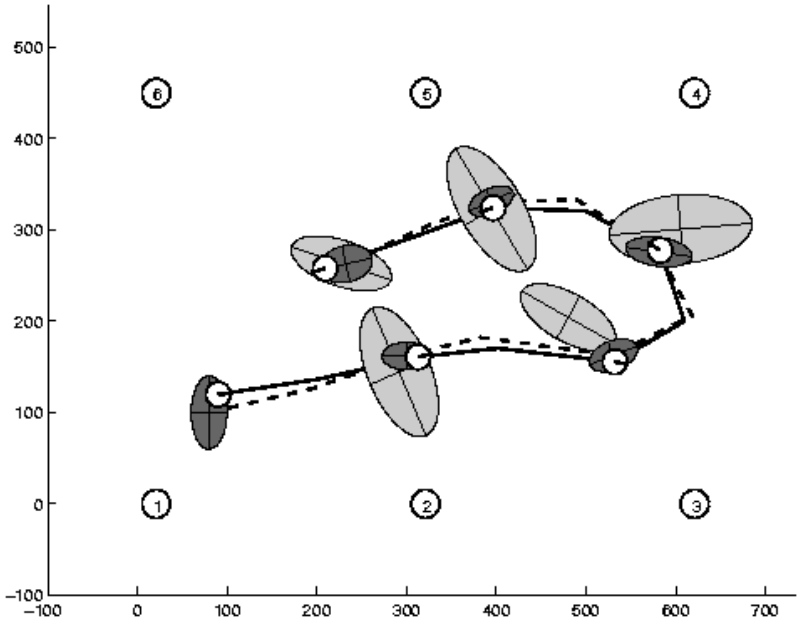


PF

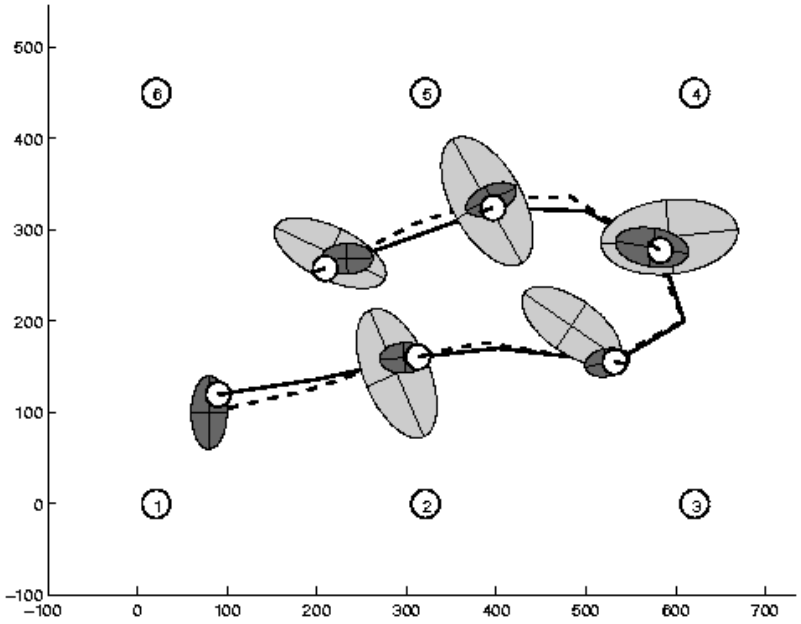


UKF

Estimation sequence

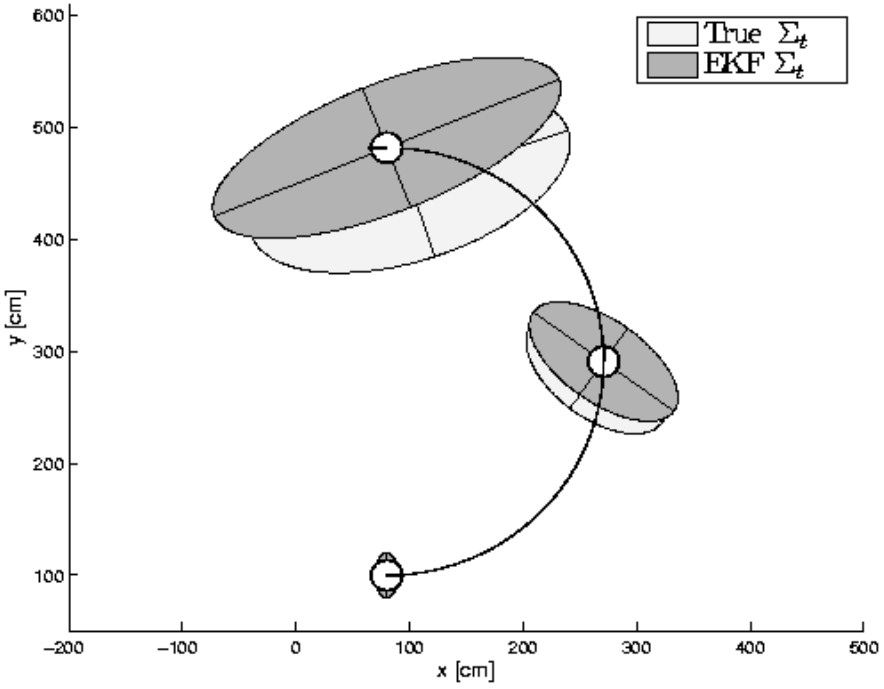


EKF

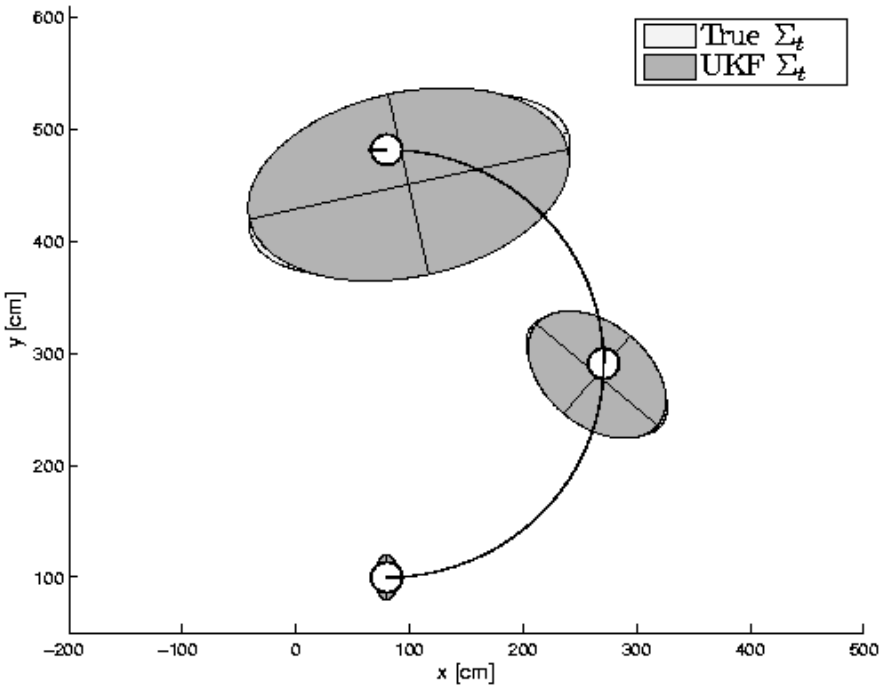


UKF

Prediction quality



EKF



UKF

UKF Summary

- Highly efficient: Same complexity as EKF, with a constant factor slower in typical practical applications
- Better linearization than EKF: Accurate in first two terms of Taylor expansion (EKF only first term)
- Derivative-free: No Jacobians needed
- Still not optimal!

MATLAB SIMULATIONS