Photogrammetry & Robotics Lab

Kalman Filter and Extended Kalman Filter

Cyrill Stachniss

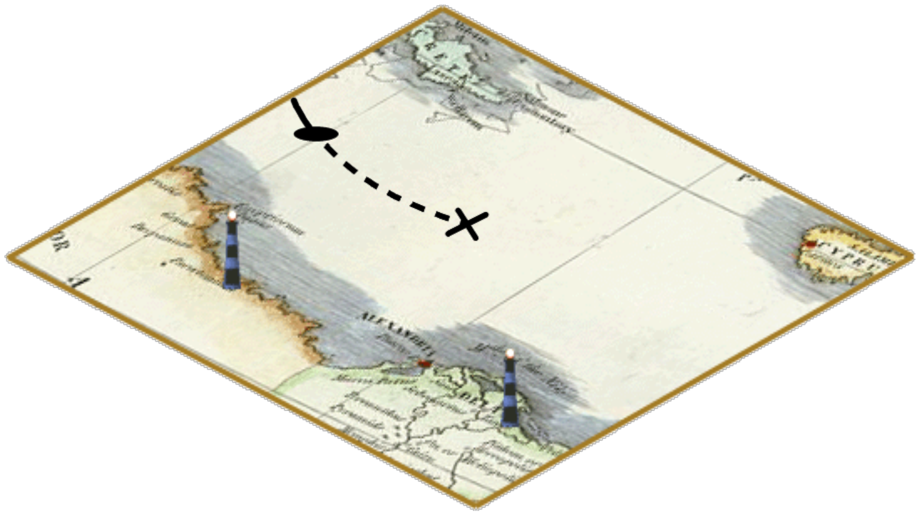
5 Minute Preparation for Today

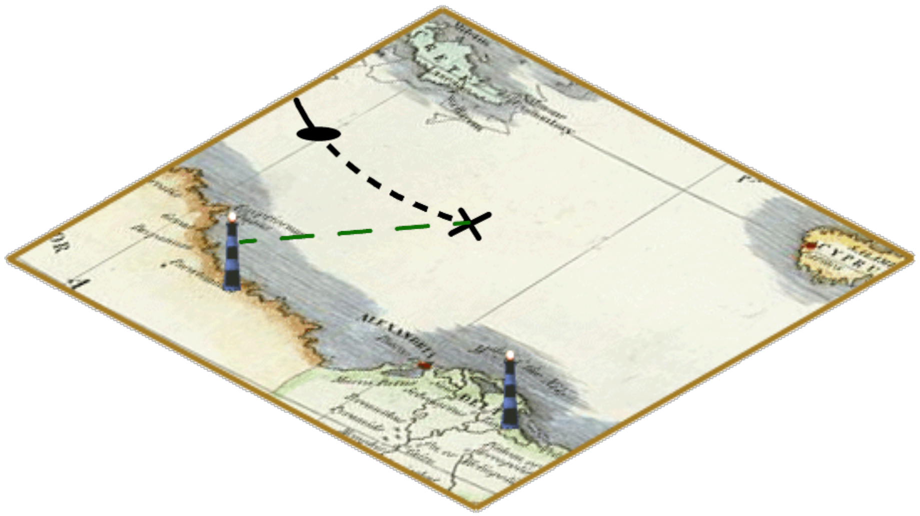
https://www.ipb.uni-bonn.de/5min/

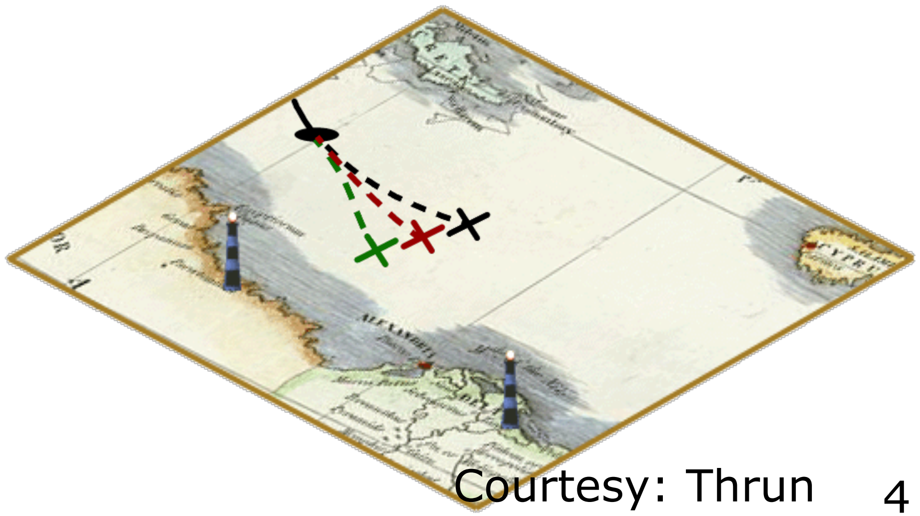
Kalman Filter

- It is a Bayes filter
- Performs recursive state estimation
- **Prediction** step to exploit the **controls**
- **Correction** step to exploit the **observations**

Kalman Filter Example







Courtesy: Thrun

Mapping and Localization are State Estimation Problems

- Bayes filter is one tool for state estimation
- **Prediction**

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\; bel(x_{t-1})\, dx_{t-1}$$

- **Correction**

$$bel(x_t) = \eta\, p(z_t \mid x_t)\; \overline{bel}(x_t)$$

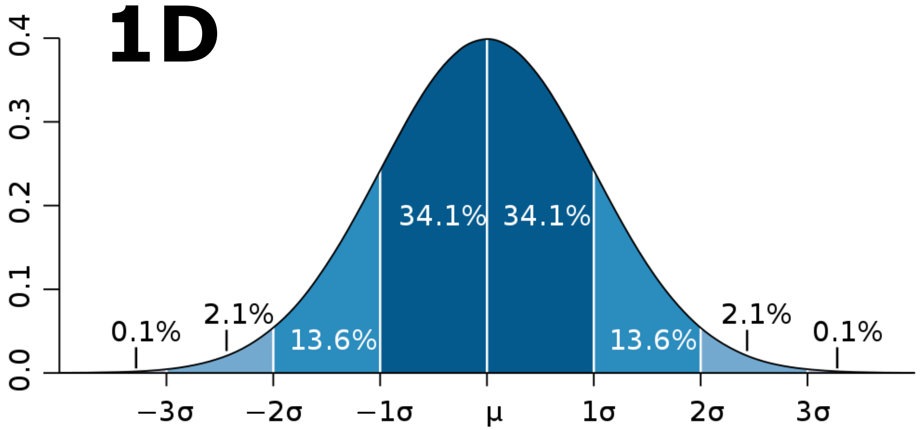
Kalman Filter

- Bayes filter
- Estimator for the **linear Gaussian** case
- **Optimal solution** for linear models and Gaussian distributions
- Result equivalent to least squares solution in a linear Gaussian world

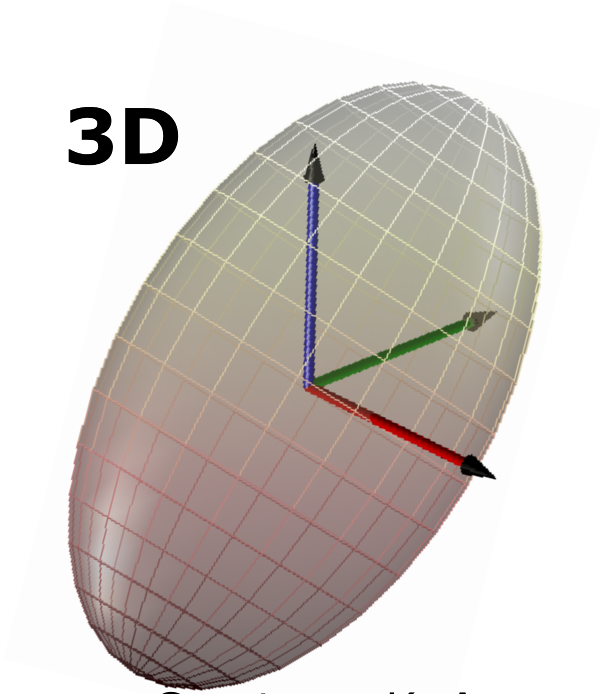
Kalman Filter Distribution

- Everything is Gaussian

$$p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$



3D



Courtesy: K. Arras

How to Update a Gaussian Belief Based on Motions and Observations?

Properties: Marginalization and Conditioning

- Given

$$x = \begin{pmatrix} x_a \\ x_b \end{pmatrix} \quad p(x) = \mathcal{N}$$

- The marginals are Gaussians

$$p(x_a) = \mathcal{N} \qquad p(x_b) = \mathcal{N}$$

- as well as the conditionals

$$p(x_a \mid x_b) = \mathcal{N} \qquad p(x_b \mid x_a) = \mathcal{N}$$

Marginalization

- Given $p(x) = p(x_a, x_b) = \mathcal{N}(\mu, \Sigma)$

  with $\mu = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix}$ $\Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$

- The **marginal distribution** is

$$p(x_a) = \int p(x_a, x_b)\, dx_b = \mathcal{N}(\mu, \Sigma)$$

  with $\mu = \mu_a$ $\Sigma = \Sigma_{aa}$

Conditioning

- Given $p(x) = p(x_a, x_b) = \mathcal{N}(\mu, \Sigma)$

  with $\mu = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix} \quad \Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$

- The **conditional distribution** is

$$p(x_a \mid x_b) = \frac{p(x_a, x_b)}{p(x_b)} = \mathcal{N}(\mu, \Sigma)$$

with $\mu = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(b - \mu_b)$

$$\Sigma = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$

Marginalization and Conditioning

$$p\left(\left(\begin{array}{c} x_a \\ x_b \end{array}\right)\right) = \mathcal{N}(\mu, \Sigma) = \mathcal{N}\left(\left(\begin{array}{c} \mu_a \\ \mu_b \end{array}\right), \left(\begin{array}{cc} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{array}\right)\right)$$

marginalization

$$p(x_a) = \mathcal{N}(\mu, \Sigma)$$

$$\mu = \mu_a$$

$$\Sigma = \Sigma_{aa}$$

conditioning

$$p(x_a \mid x_b) = \mathcal{N}(\mu, \Sigma)$$

$$\mu = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(b - \mu_b)$$

$$\Sigma = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$

Linear Model for Motions and Observations

Linear Models

- Both models can be expressed through a linear function

$$f(x) = Ax + b$$

Linear Models

- Both models can be expressed through a linear function

$$f(x) = Ax + b$$

- A **Gaussian** that istransformed trough a linear function **stays Gaussian**

Linear Models

- The Kalman filter assumes a linear transition and observation model
- Zero mean Gaussian noise

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

$$z_t = C_t x_t + \delta_t$$

Components of a Kalman Filter

A_t Matrix $(n \times n)$ that describes how the state evolves from $t-1$ to t without controls or noise.

B_t Matrix $(n \times l)$ that describes how the control u_t changes the state from $t-1$ to t.

C_t Matrix $(k \times n)$ that describes how to map the state x_t to an observation z_t.

ϵ_t

δ_t Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

Linear Motion Model

- Motion under Gaussian noise leads to

$$p(x_t \mid u_t, x_{t-1}) = ?$$

Linear Motion Model

- Motion under Gaussian noise leads to

$$p(x_t \mid u_t, x_{t-1}) = \det(2\pi R_t)^{-\frac{1}{2}}$$
$$\exp\left(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\right)$$

- R_t describes the noise of the motion

Linear Observation Model

- Measuring under Gaussian noise leads to

$$p(z_t \mid x_t) = ?$$

Linear Observation Model

- Measuring under Gaussian noise leads to

$$p(z_t \mid x_t) = \det(2\pi Q_t)^{-\frac{1}{2}}$$
$$\exp\left(-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right)$$

- Q_t describes the measurement noise

Everything stays Gaussian

- Given an initial Gaussian belief, the belief stays Gaussian

$$bel(x_t) = \eta \, \underbrace{p(z_t \mid x_t)}_{\text{Gaussian}} \, \underbrace{\overline{bel}(x_t)}_{?}$$

Everything stays Gaussian

- Given an initial Gaussian belief, the belief stays Gaussian

$$bel(x_t) = \eta \, \underbrace{p(z_t \mid x_t)}_{\text{Gaussian}} \, \underbrace{\overline{bel}(x_t)}_{?}$$

- The product of two Gaussian is again a Gaussian
- We only need to show that $\overline{bel}(x_t)$ is Gaussian so that $bel(x_t)$ is Gaussian

Everything stays Gaussian

- Given an initial Gaussian belief, the belief stays Gaussian

$$\overline{bel}(x_t) = \int \underbrace{p(x_t \mid u_t, x_{t-1})}_{\text{Gaussian}} \; \underbrace{bel(x_{t-1})}_{\text{Gaussian}} \, dx_{t-1}$$

Everything stays Gaussian

- Given an initial Gaussian belief, the belief stays Gaussian

$$\overline{bel}(x_t) = \int \underbrace{p(x_t \mid u_t, x_{t-1})}_{\text{Gaussian}} \; \underbrace{bel(x_{t-1})}_{\text{Gaussian}} \, dx_{t-1}$$

- Is that sufficient so that $\overline{bel}(x_t)$ is Gaussian?

Everything stays Gaussian

- We can write

$$
\begin{aligned}
&\overline{bel}(x_t) \\
&\quad = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1} \\
&\quad = \eta \int \exp\left(-\frac{1}{2}\,(x_t - A_t\, x_{t-1} - B_t\, u_t)^T\, R_t^{-1}\,(x_t - A_t\, x_{t-1} - B_t\, u_t)\right) \\
&\qquad\qquad \exp\left(-\frac{1}{2}\,(x_{t-1} - \mu_{t-1})^T \Sigma_{t-1}^{-1} (x_{t-1} - \mu_{t-1})\right)\, dx_{t-1}
\end{aligned}
$$

Everything stays Gaussian

- We can write

$$
\begin{aligned}
&\overline{bel}(x_t) \\
&\quad = \eta \int \exp\left(-\frac{1}{2}\,(x_t - A_t\,x_{t-1} - B_t\,u_t)^T\,R_t^{-1}\,(x_t - A_t\,x_{t-1} - B_t\,u_t)\right) \\
&\qquad\qquad \exp\left(-\frac{1}{2}\,(x_{t-1} - \mu_{t-1})^T \Sigma_{t-1}^{-1}(x_{t-1} - \mu_{t-1})\right)\,dx_{t-1}
\end{aligned}
$$

- and thus

$$
\begin{aligned}
\overline{bel}(x_t) &= \eta \int \exp\left(-L_t\right)\,dx_{t-1} \\
L_t &= \frac{1}{2}\,(x_t - A_t\,x_{t-1} - B_t\,u_t)^T\,R_t^{-1}\,(x_t - A_t\,x_{t-1} - B_t\,u_t) \\
&\quad + \frac{1}{2}\,(x_{t-1} - \mu_{t-1})^T\,\Sigma_{t-1}^{-1}\,(x_{t-1} - \mu_{t-1})
\end{aligned}
$$

Everything stays Gaussian

- We can split up L_t in a part that depends on x_t and on x_t, x_{t-1}

$$L_t = L_t(x_{t-1}, x_t) + L_t(x_t)$$

- Thus

$$\begin{aligned}\overline{bel}(x_t) &= \eta \int \exp\left(-L_t(x_{t-1}, x_t) - L_t(x_t)\right)\, dx_{t-1} \\ &= \eta \underbrace{\exp\left(-L_t(x_t)\right)}_{\text{Gaussian}} \underbrace{\int \exp\left(-L_t(x_{t-1}, x_t)\right)\, dx_{t-1}}_{\text{Marginalization}}\end{aligned}$$

- Details: Probabilistic Robotics, Ch. 3.2 (p. 46-49)

Everything stays Gaussian

- Given an initial Gaussian belief, the belief stays Gaussian

$$\underbrace{\overline{bel}(x_t)}_{\text{Gaussian}} = \underbrace{\int \underbrace{p(x_t \mid u_t, x_{t-1})}_{\text{Gaussian}} \; \underbrace{bel(x_{t-1})}_{\text{Gaussian}} \, dx_{t-1}}_{\text{Gaussian}}$$

$$\underbrace{bel(x_t)}_{\text{Gaussian}} = \eta \; \underbrace{p(z_t \mid x_t)}_{\text{Gaussian}} \; \underbrace{\overline{bel}(x_t)}_{\text{Gaussian}}$$

Everything is and stays Gaussian!

How Do We Typically Represent Gaussians?

$$\mu \quad \Sigma$$

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\ bel(x_{t-1})\, dx_{t-1}$$

$$bel(x_t) = \eta\, p(z_t \mid x_t)\ \overline{bel}(x_t)$$

$$\Rightarrow \quad \mu_t = ? \qquad \Sigma_t = ?$$

To Derive the Kalman Filter Algorithm, One Exploits...

- Product of two Gaussians is a Gaussian
- Gaussians stays Gaussians under linear transformations
- Marginal and conditional distribution of a Gaussian stays a Gaussian
- Computing mean and covariance of the marginal and conditional of a Gaussian
- Matrix inversion lemma
- ...

This leads us to...

Kalman Filter Algorithm

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

$$\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$$

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

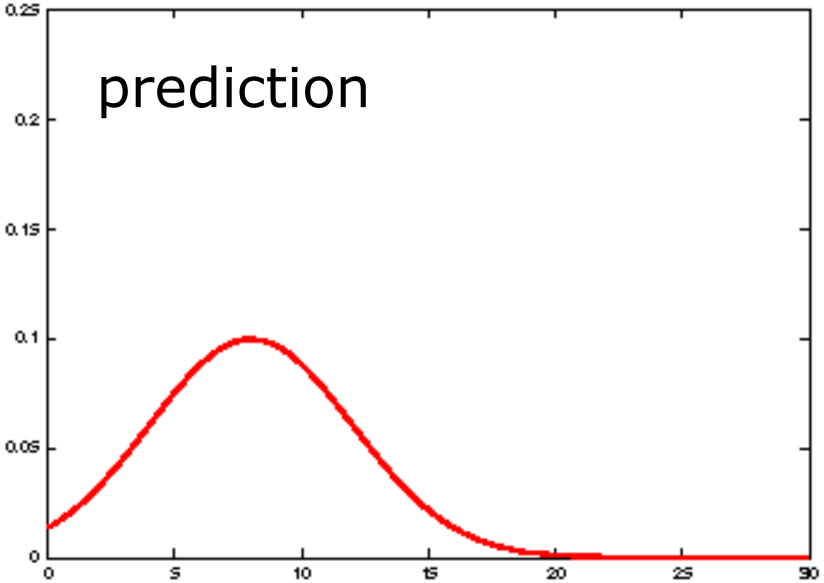
$$K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$$

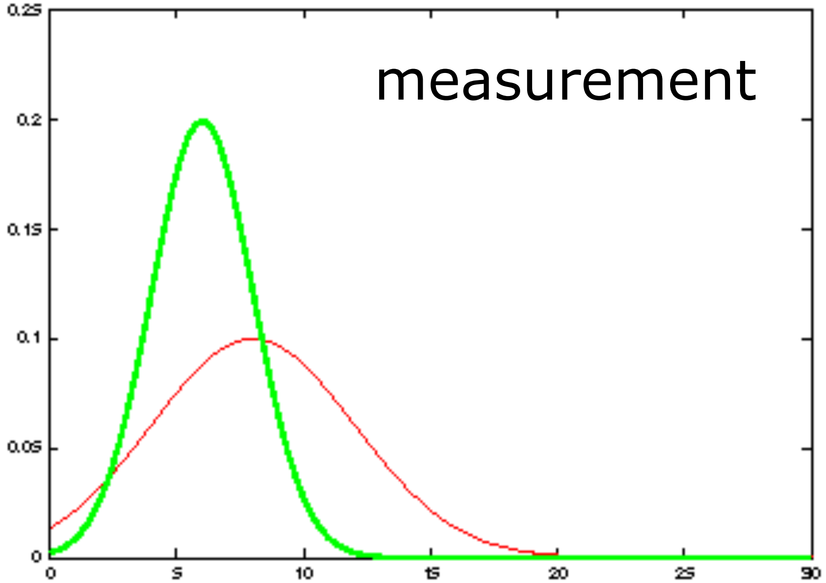
$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t\, \bar{\mu}_t)$$

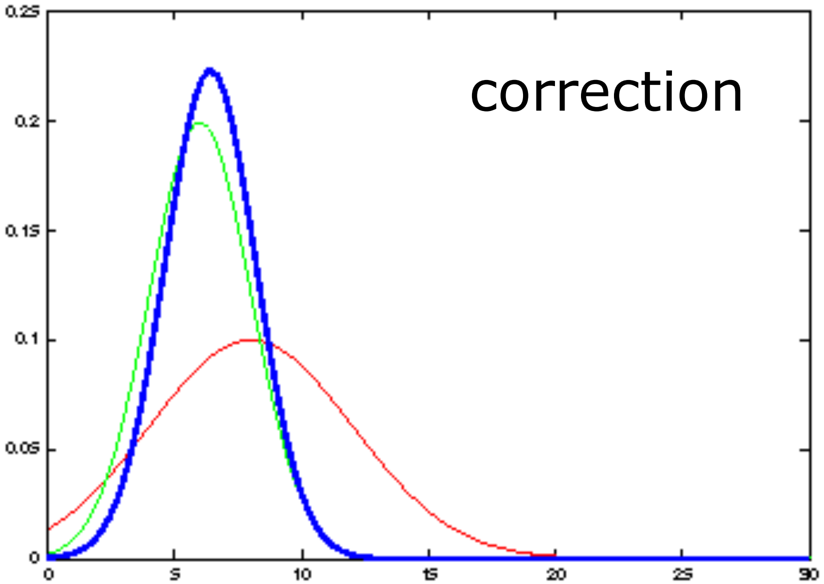
$$\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$$

$return\ \mu_t, \Sigma_t$

1D Kalman Filter Example (1)

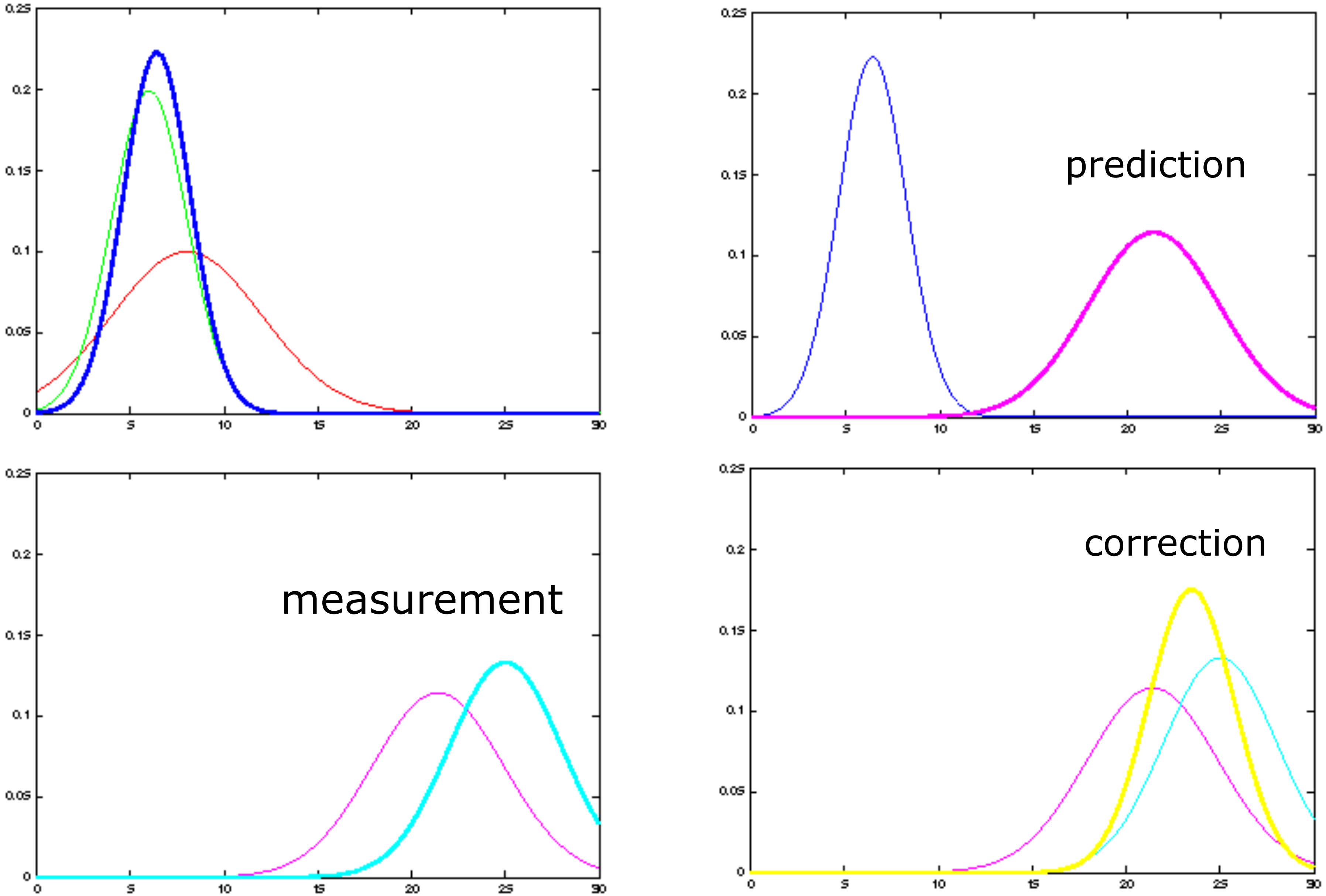






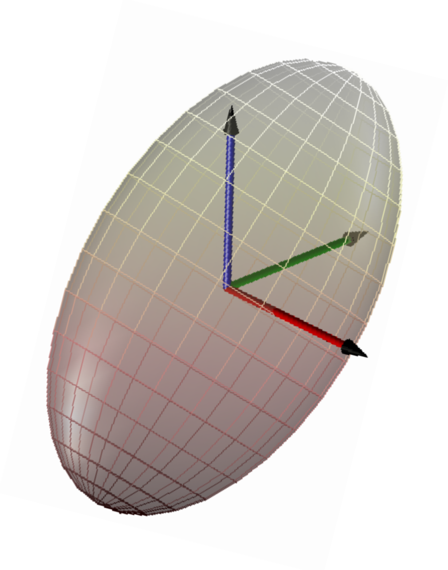
It's a weighted mean!

1D Kalman Filter Example (2)



Kalman Filter Assumptions

- Gaussian distributions and noise
- Linear motion and observation model



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

$$z_t = C_t x_t + \delta_t$$

What if this is not the case?

Non-linear Dynamic Systems

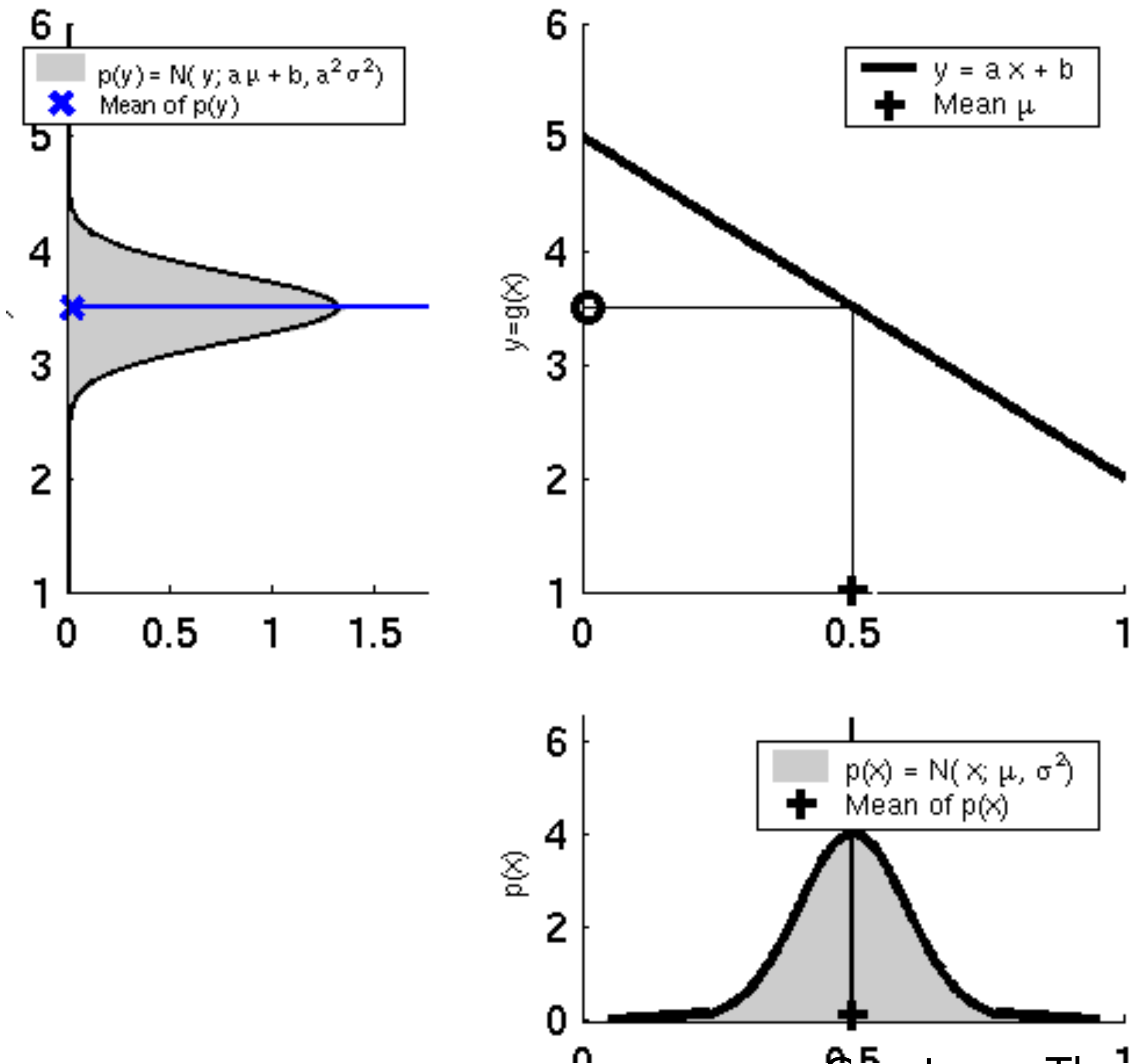
- Most realistic problems (in robotics) involve nonlinear functions

$$\cancel{x_t = A_t x_{t-1} + B_t u_t + \epsilon_t} \qquad \cancel{z_t = C_t x_t + \delta_t}$$

$$\Downarrow \qquad\qquad\qquad \Downarrow$$

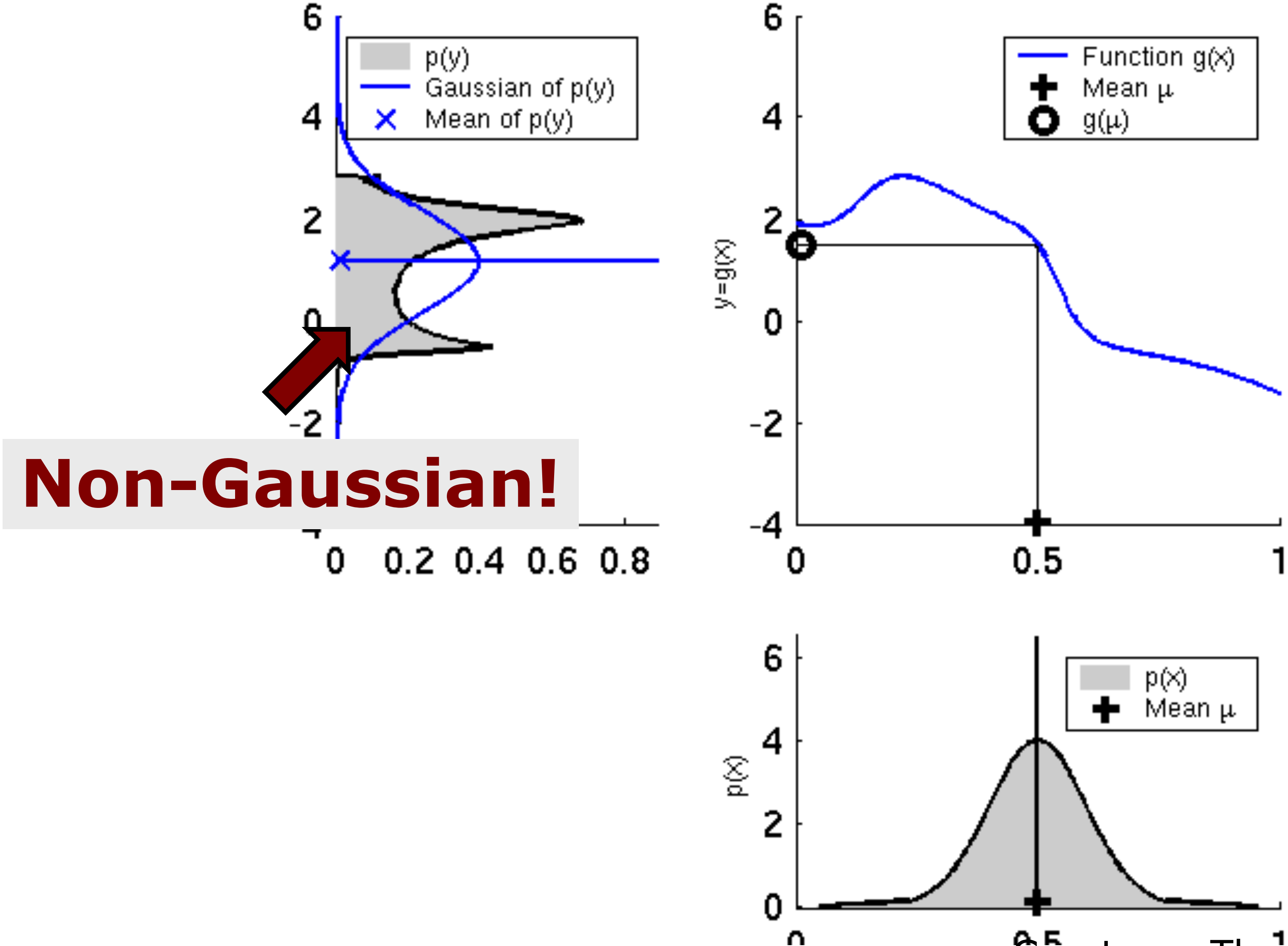
$$x_t = g(u_t, x_{t-1}) + \epsilon_t \qquad z_t = h(x_t) + \delta_t$$

Linearity Assumption Revisited



Courtesy: Thrun, Burgard, Fox

Non-Linear Function



Courtesy: Thrun, Burgard, Fox

Non-Gaussian Distributions

- The non-linear functions lead to non-Gaussian distributions
- Kalman filter is not applicable anymore!

What can be done to resolve this?

Non-Gaussian Distributions

- The non-linear functions lead to non-Gaussian distributions
- Kalman filter is not applicable anymore!

What can be done to resolve this?

Local linearization!

EKF Linearization: First Order Taylor Expansion

- Prediction:

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{\frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}}_{=:\, G_t} (x_{t-1} - \mu_{t-1})$$

- Correction:

$$h(x_t) \approx h(\bar{\mu}_t) + \underbrace{\frac{\partial h(\bar{\mu}_t)}{\partial x_t}}_{=:\, H_t} (x_t - \bar{\mu}_t)$$

Jacobian matrices

Reminder: Jacobian Matrix

- It is a **non-square matrix** $m \times n$ in general
- Given a vector-valued function

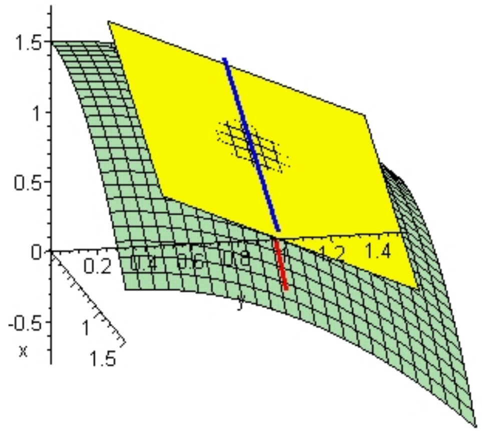
$$g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix}$$

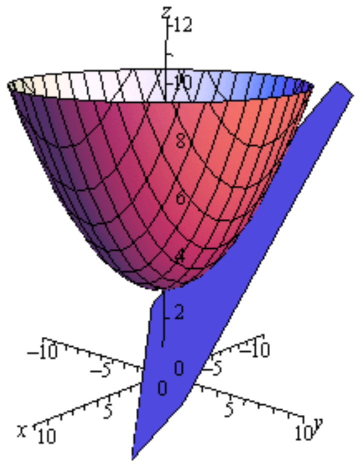
- The **Jacobian matrix** is defined as

$$G_x = \begin{pmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{pmatrix}$$

Reminder: Jacobian Matrix

- It is the orientation of the tangent plane to the vector-valued function at a given point





Courtesy: K. Arras

- Generalizes the gradient of a scalar valued function

EKF Linearization: First Order Taylor Expansion

- Prediction:

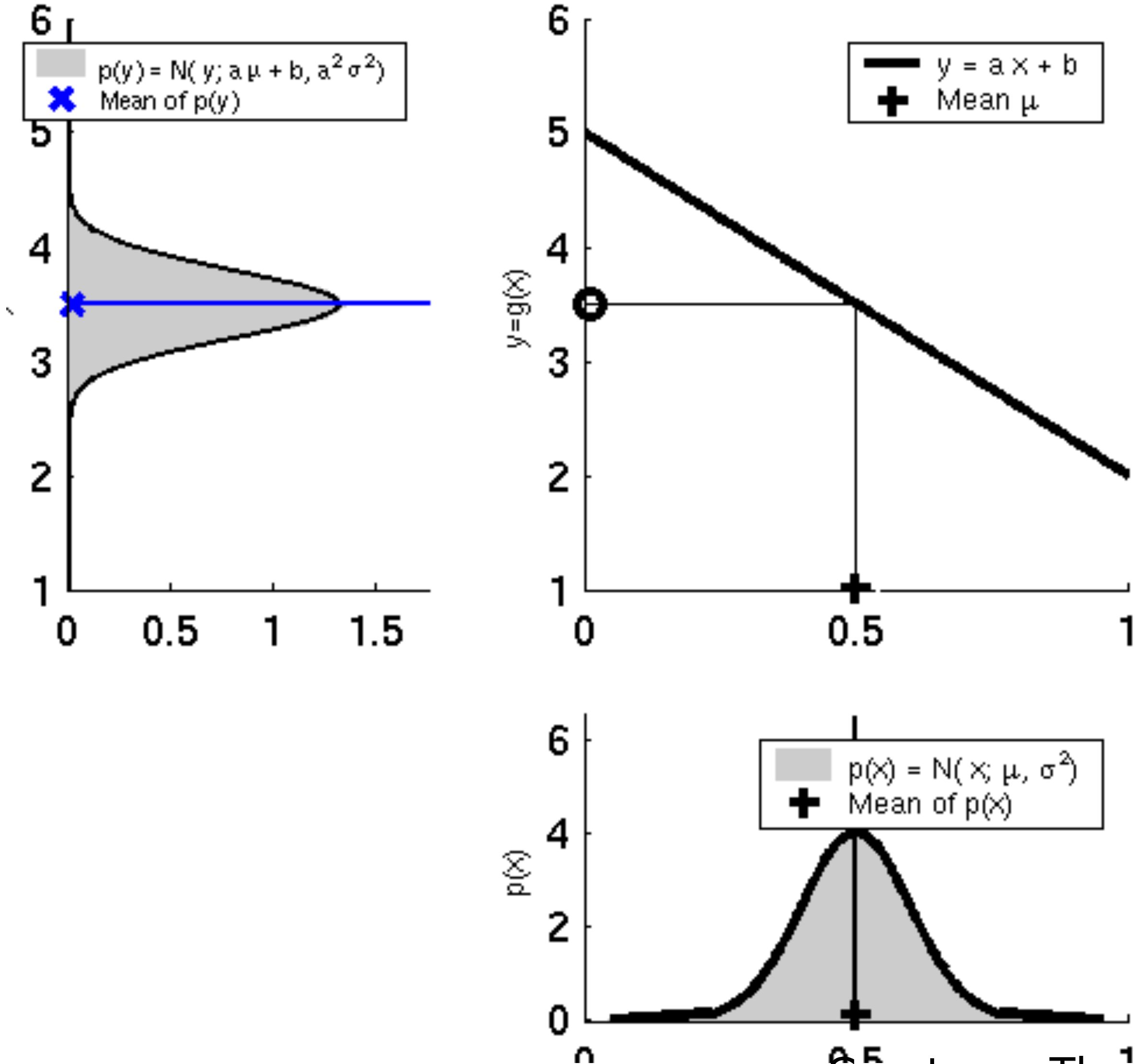
$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{\frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}}_{=:\, G_t} (x_{t-1} - \mu_{t-1})$$

- Correction:

$$h(x_t) \approx h(\bar{\mu}_t) + \underbrace{\frac{\partial h(\bar{\mu}_t)}{\partial x_t}}_{=:\, H_t} (x_t - \bar{\mu}_t)$$

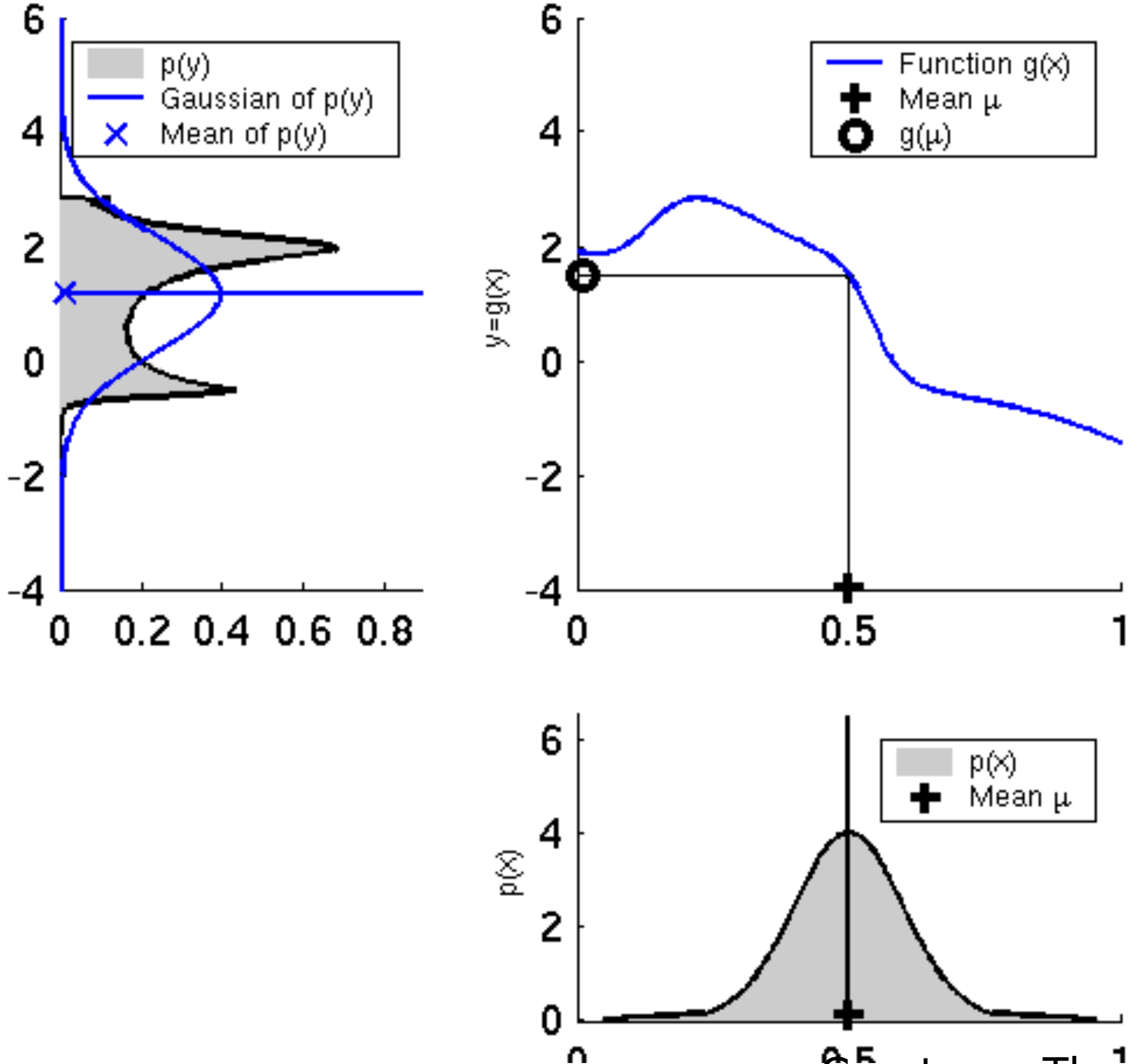
Linear functions!

Linearity Assumption Revisited



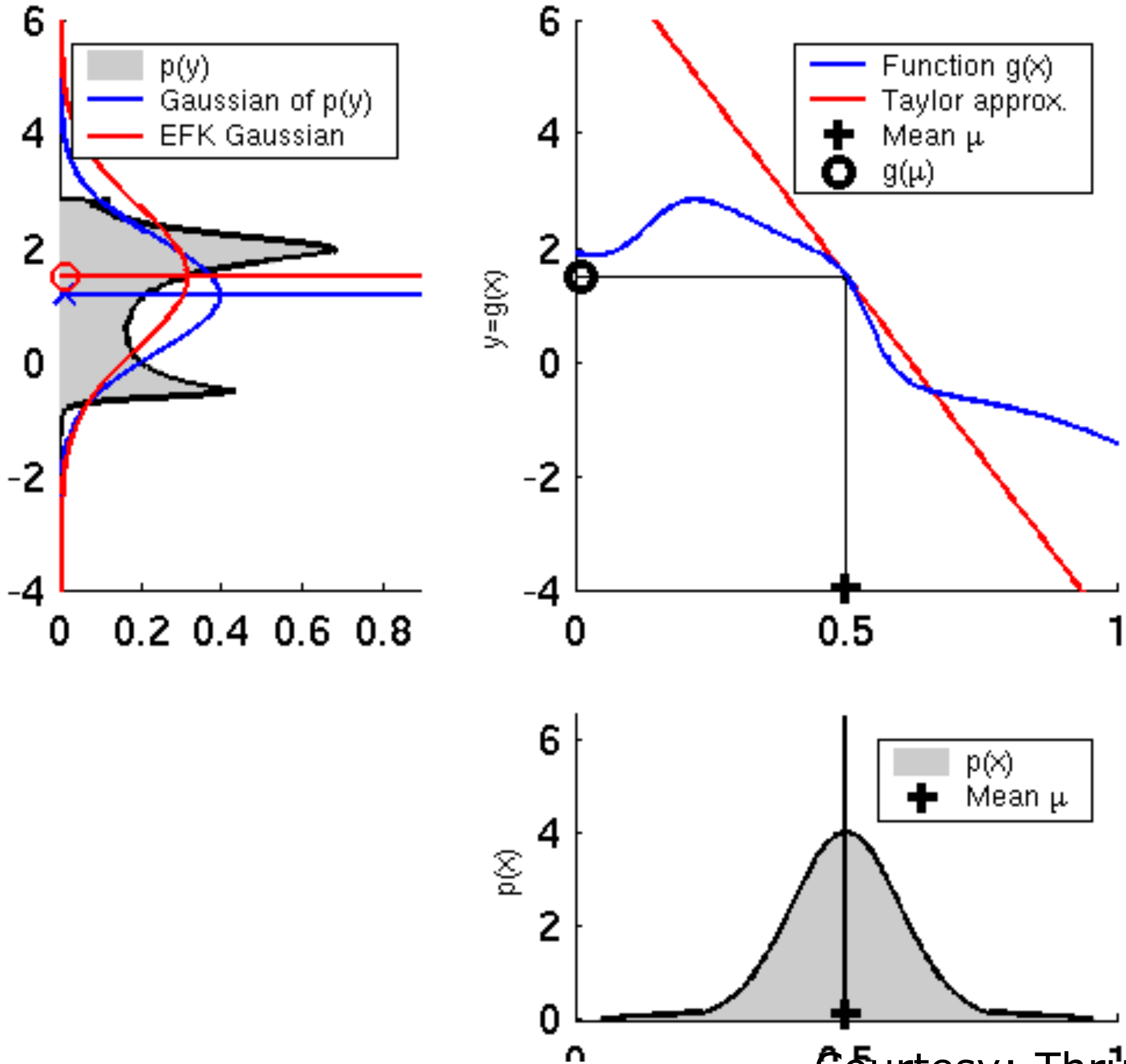
Courtesy: Thrun, Burgard, Fox

Non-Linear Function



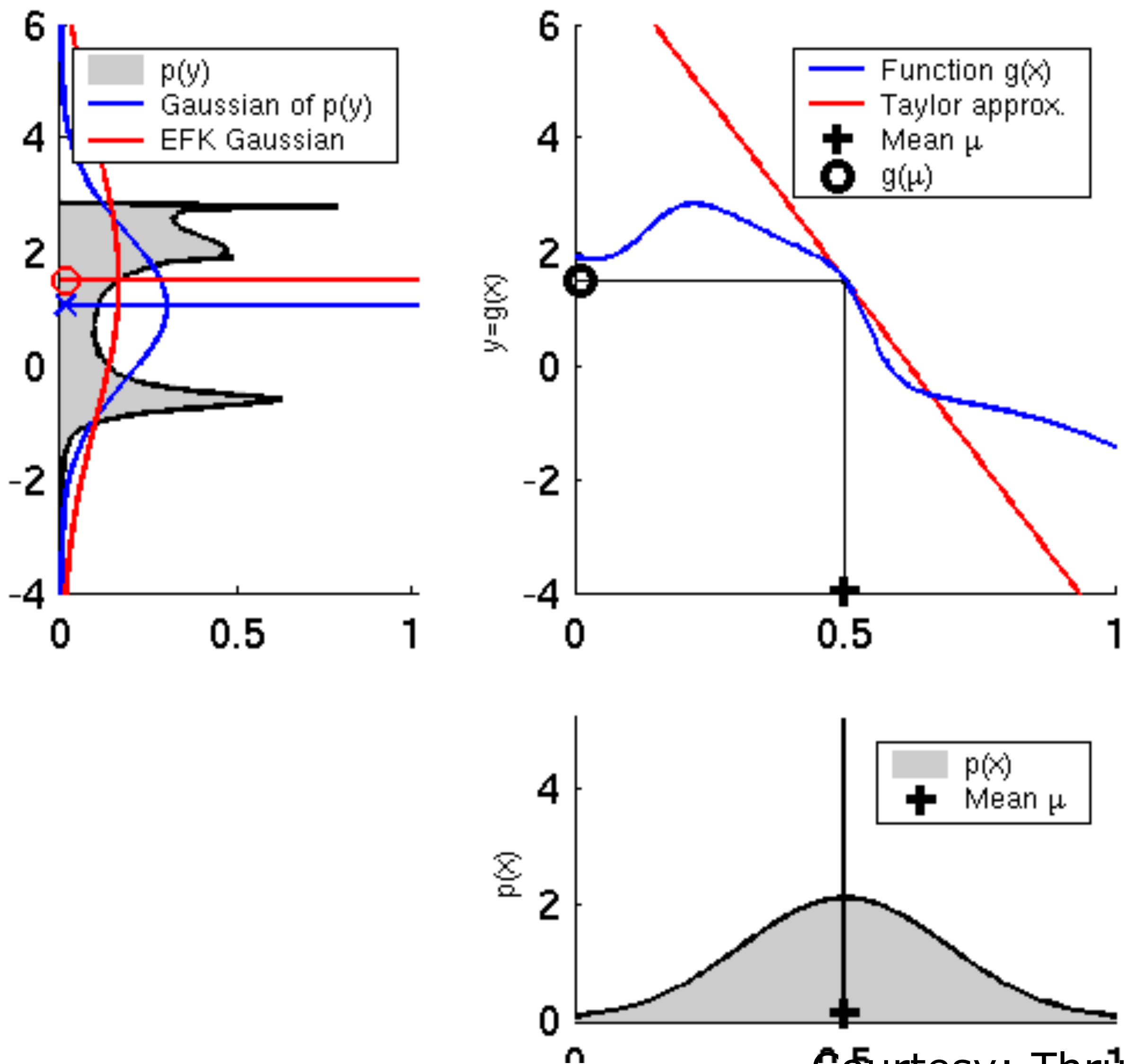
Courtesy: Thrun, Burgard, Fox

EKF Linearization (1)



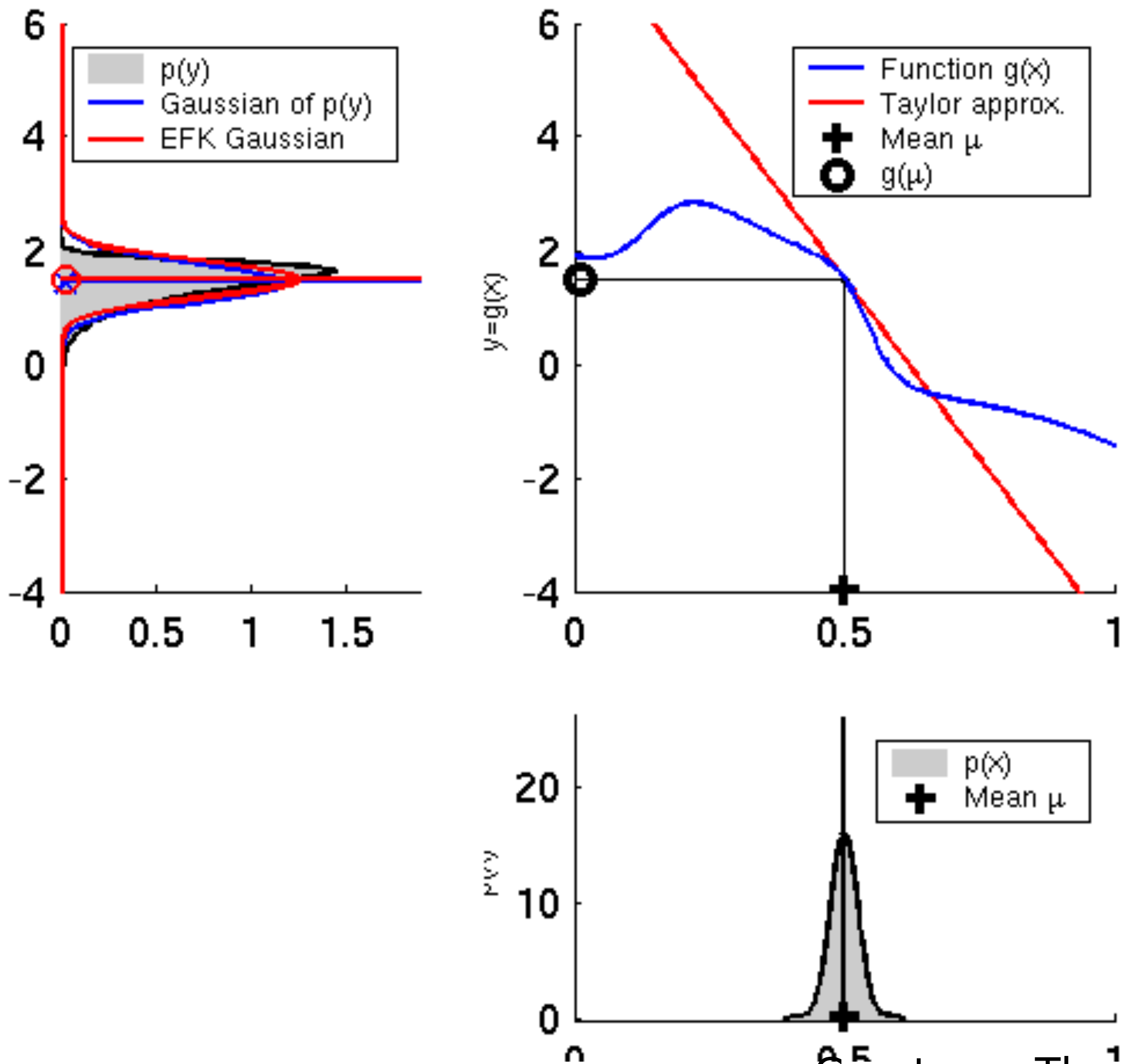
Courtesy: Thrun, Burgard, Fox

EKF Linearization (2)



Courtesy: Thrun, Burgard, Fox

EKF Linearization (3)



Courtesy: Thrun, Burgard, Fox

Linearized Motion Model

- The linearized model leads to

$$p(x_t \mid u_t, x_{t-1}) \approx \det\left(2\pi R_t\right)^{-\frac{1}{2}}$$
$$\exp\left(-\frac{1}{2}\left(x_t - g(u_t, \mu_{t-1}) - G_t\left(x_{t-1} - \mu_{t-1}\right)\right)^T\right.$$
$$\left. R_t^{-1}\left(x_t - \underbrace{g(u_t, \mu_{t-1}) - G_t\left(x_{t-1} - \mu_{t-1}\right)}_{\text{linearized model}}\right)\right)$$

- R_t describes the noise of the motion

Linearized Observation Model

- The linearized model leads to

$$p(z_t \mid x_t) = \det\left(2\pi Q_t\right)^{-\frac{1}{2}}$$
$$\exp\Big(-\frac{1}{2}\left(z_t - h(\bar{\mu}_t) - H_t\left(x_t - \bar{\mu}_t\right)\right)^T$$
$$Q_t^{-1}\left(z_t - \underbrace{h(\bar{\mu}_t) - H_t\left(x_t - \bar{\mu}_t\right)}_{\text{linearized model}}\right)\Big)$$

- Q_t describes the measurement noise

Extended Kalman Filter Algorithm

1: $\textbf{Extended\_Kalman\_filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

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

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

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

7: return μ_t, Σ_t

$$\boxed{\begin{array}{c} A_t \leftrightarrow G_t \\ C_t \leftrightarrow H_t \end{array}}$$

KF vs. EKF

EKF Localization Example

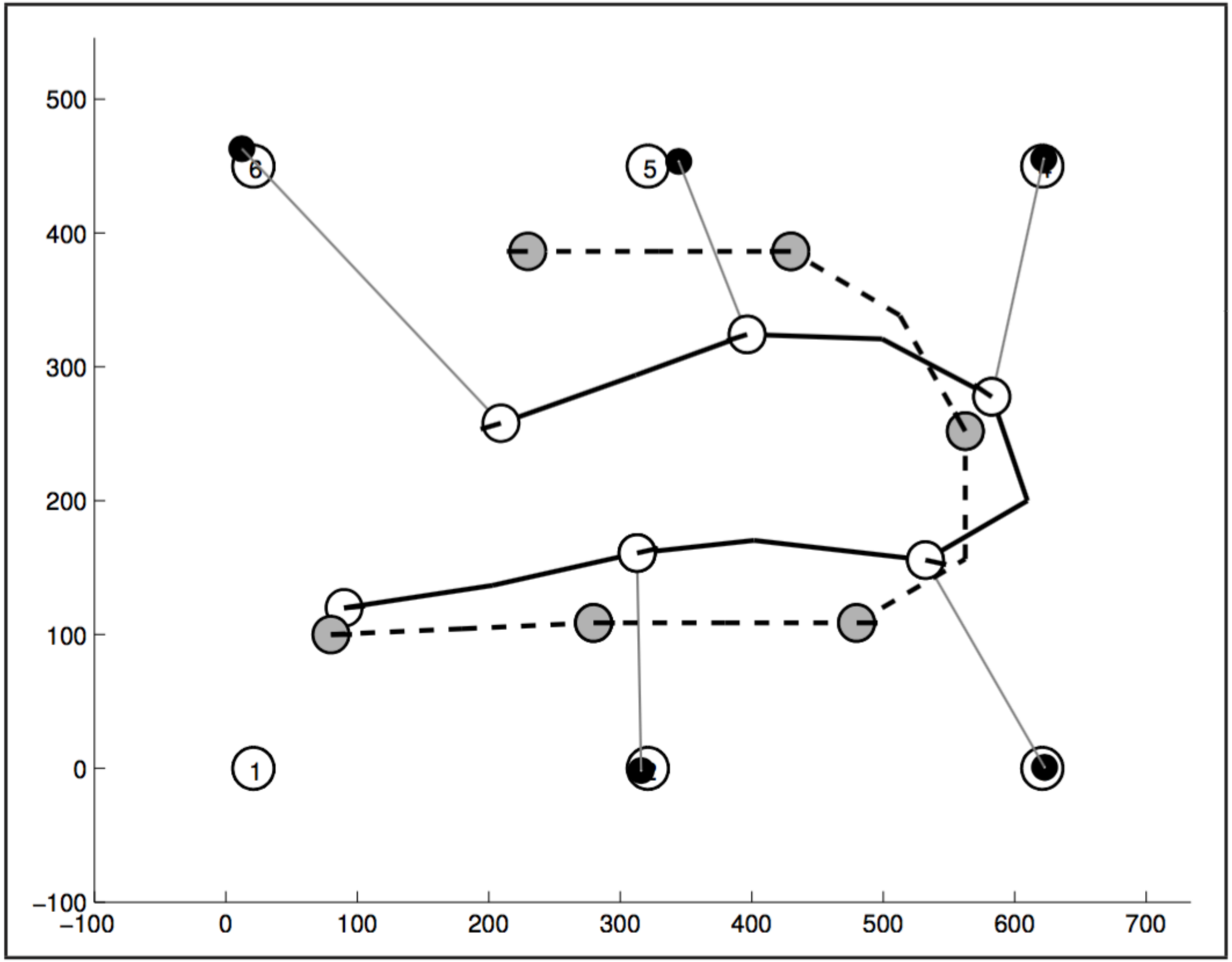
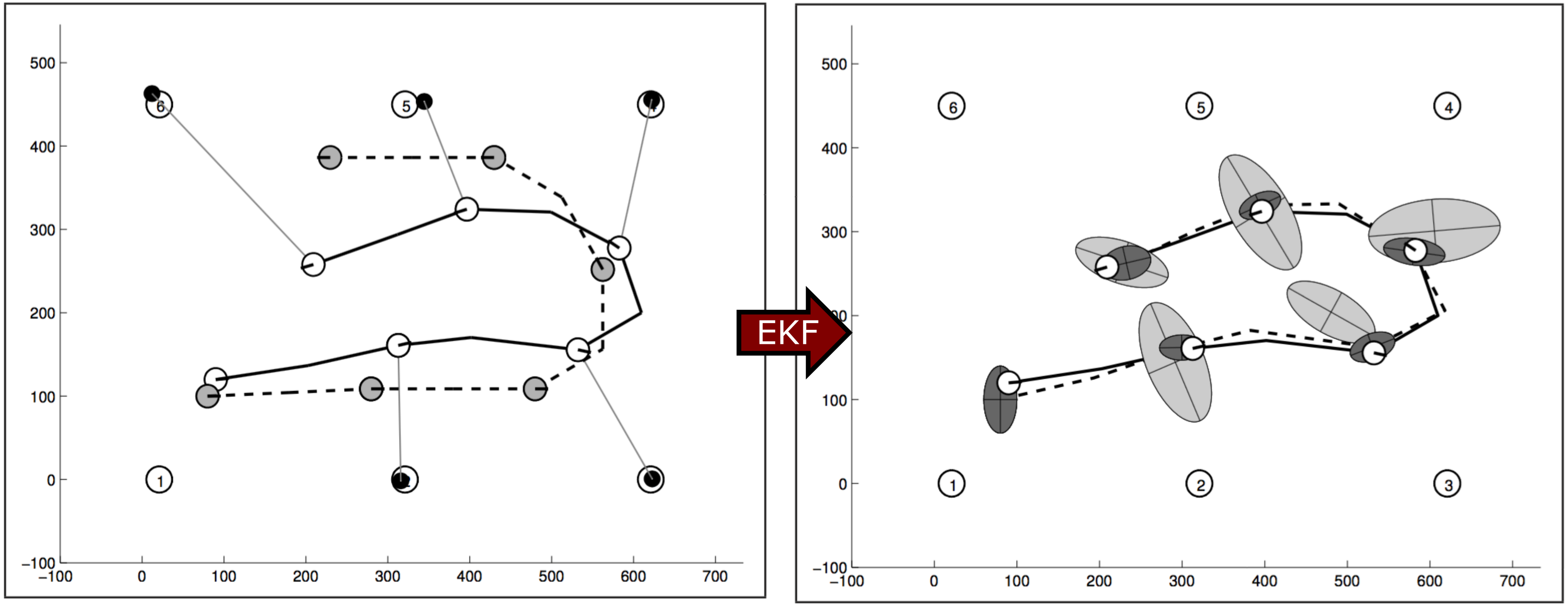


Image courtesy: Thrun, Burgard, Fox

EKF Localization Example

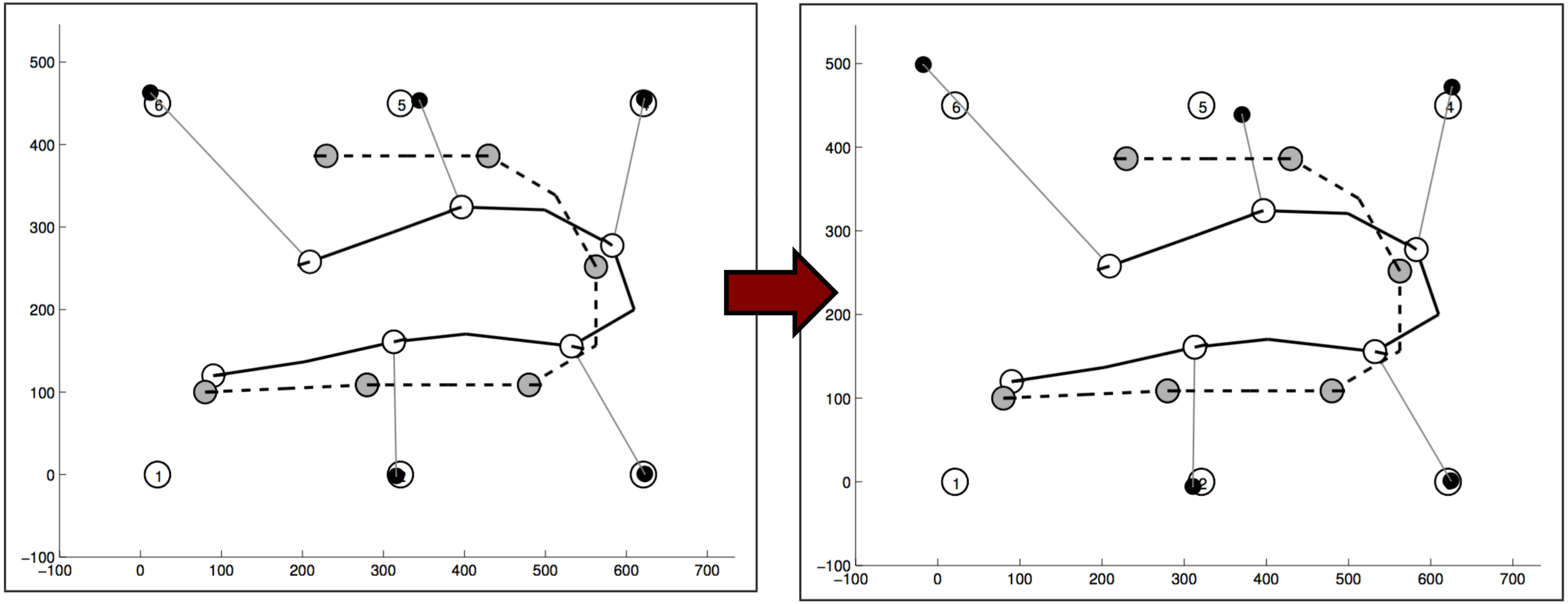


weighted sum of predictions and observations

Image courtesy: Thrun, Burgard, Fox

EKF Localization Example

More noisy sensor…

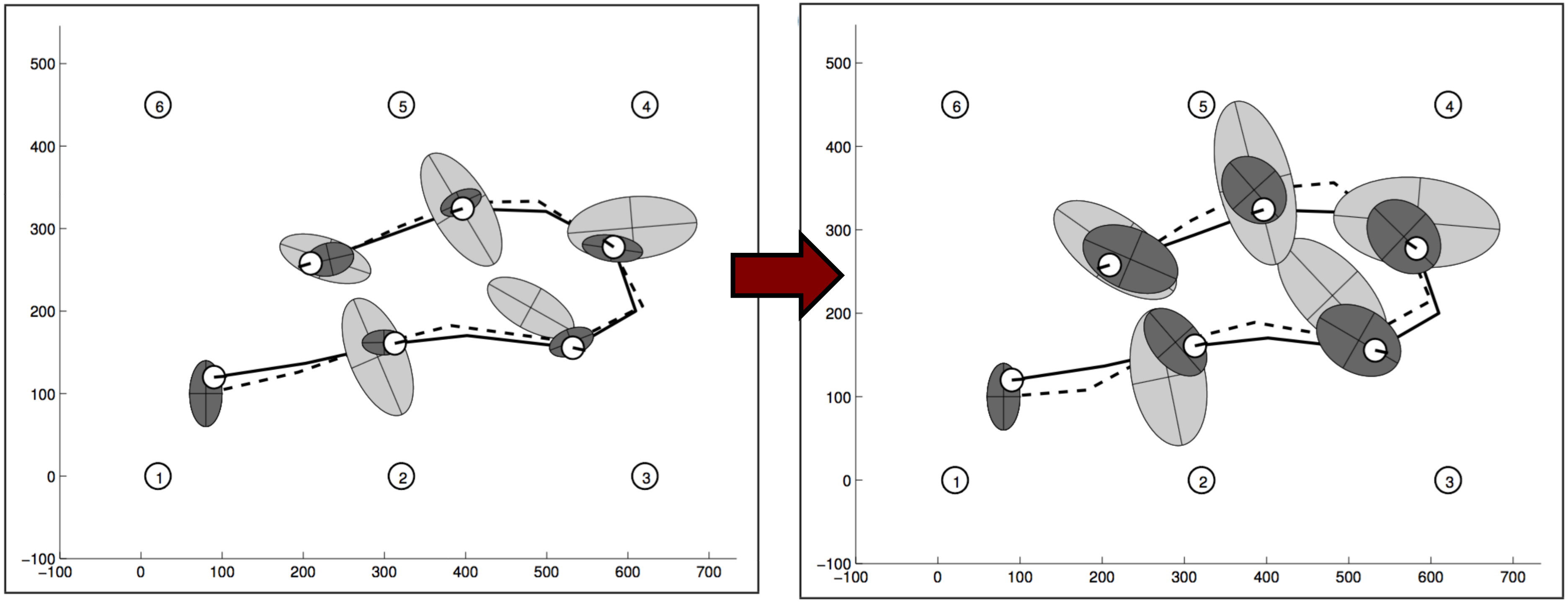


larger covariances for the observations

Image courtesy: Thrun, Burgard, Fox

EKF Localization Example

More noisy sensor…



larger covariances, trusts the prediction more

Image courtesy: Thrun, Burgard, Fox

Extended Kalman Filter Summary

- Extension of the Kalman filter
- One way to handle the non-linearities
- Performs local linearizations
- Works well in practice for moderate non-linearities
- Large uncertainty leads to increased approximation error error

Literature

Kalman Filter and EKF

- Thrun et al.: “Probabilistic Robotics”, Chapter 3
- Schön and Lindsten: “Manipulating the Multivariate Gaussian Density”
- Welch and Bishop: “Kalman Filter Tutorial”