

Robot Mapping

Extended Kalman Filter

**Gian Diego Tipaldi, Luciano Spinello,
Wolfram Burgard**

SLAM is a State Estimation Problem

- Estimate the map and robot's pose
- Bayes filter is one tool for state estimation

- **Prediction**

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

- **Correction**

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

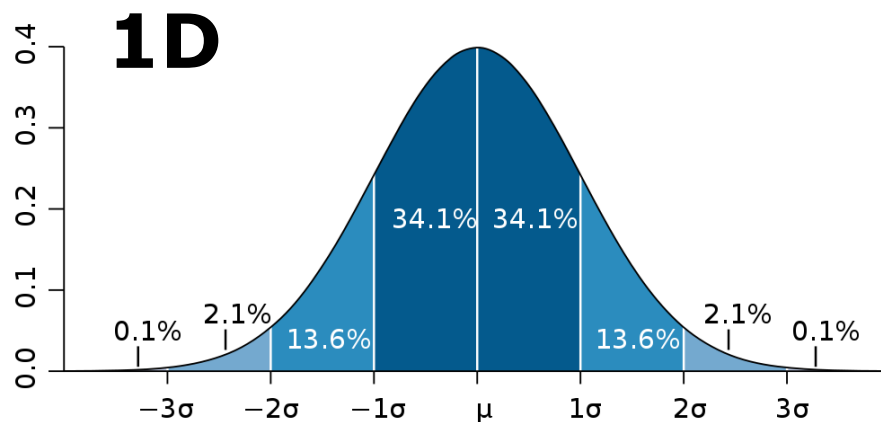
Kalman Filter

- It is a Bayes filter
- Estimator for the linear Gaussian case
- Optimal solution for linear models and Gaussian distributions

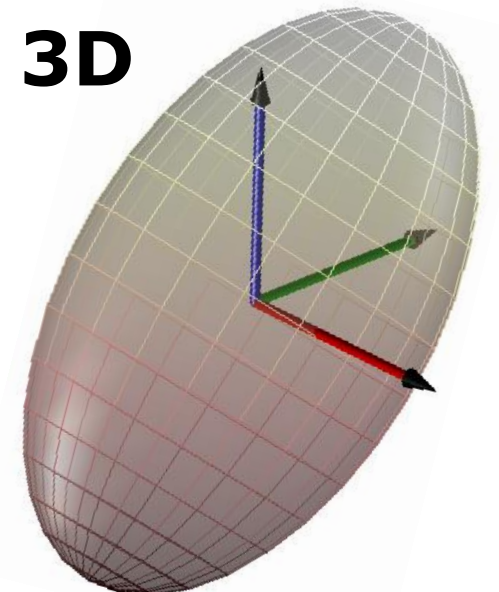
Kalman Filter Distribution

- 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



Properties: Marginalization and Conditioning

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

- The marginals are Gaussians

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

- as well as the conditionals

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

Marginalization

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

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

- The marginal distribution is

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

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

Conditioning

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

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

- The conditional distribution is

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

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

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

Linear Model

- 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 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.
- δ_t

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

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

- R_t describes the noise of the motion

Linear Observation Model

- Measuring under Gaussian noise leads to

$$p(z_t | x_t) = ?$$

Linear Observation Model

- Measuring under Gaussian noise leads to

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

- Q_t describes the measurement noise

Everything stays Gaussian

- Given an initial Gaussian belief, the belief is always Gaussian

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

$$bel(x_t) = \eta \underline{p(z_t | x_t)} \underline{\overline{bel}(x_t)}$$

- Proof is non-trivial
(see Probabilistic Robotics, Sec. 3.2.4)

Kalman Filter Algorithm

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

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

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

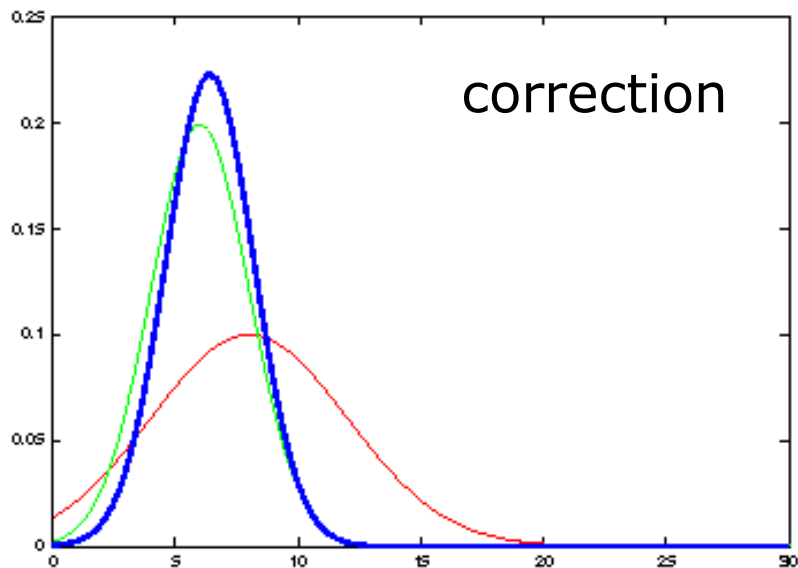
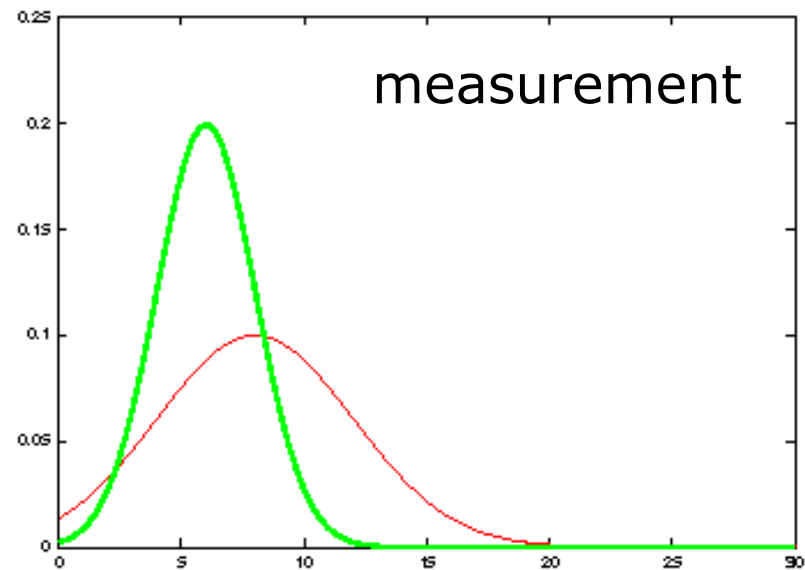
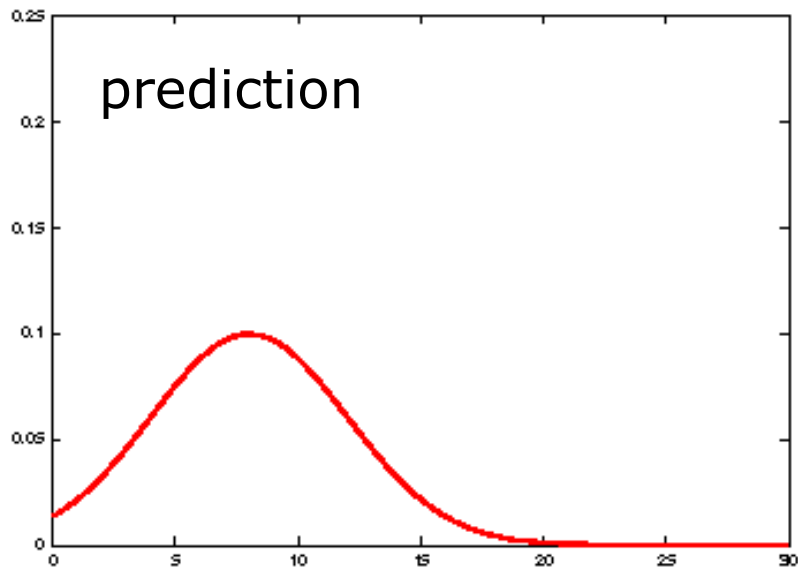
4: $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

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

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

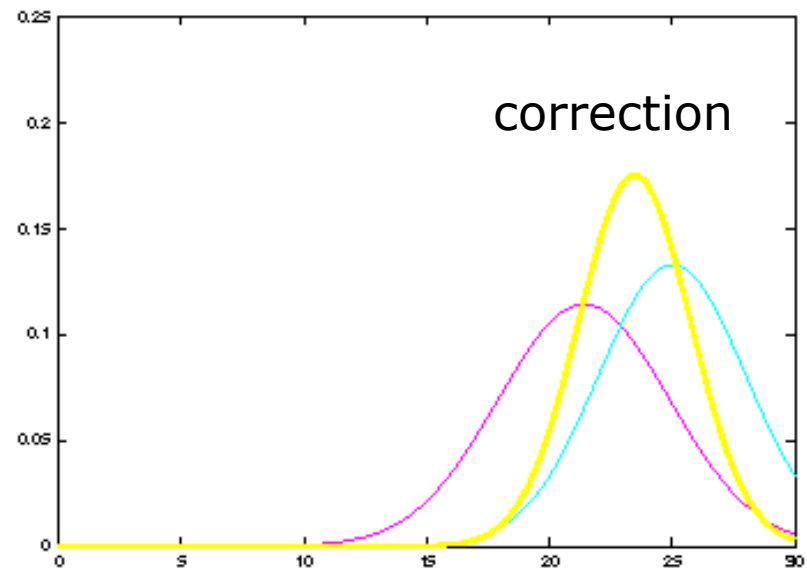
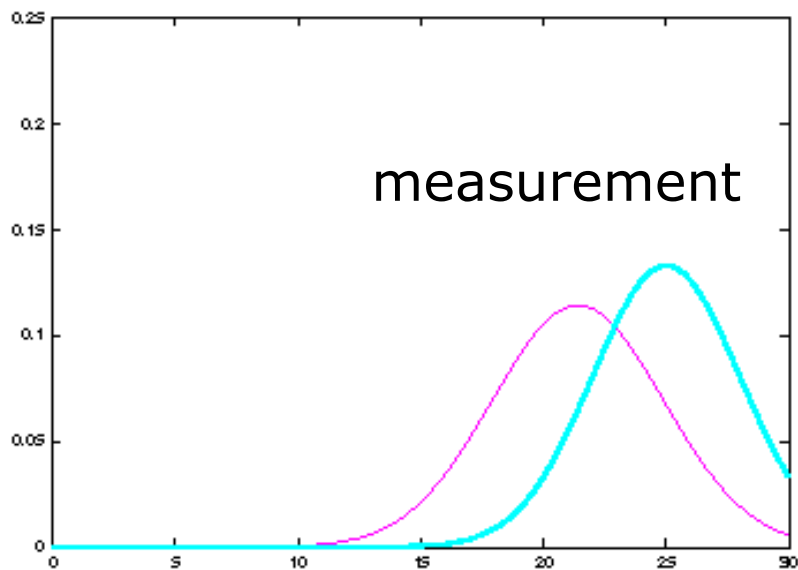
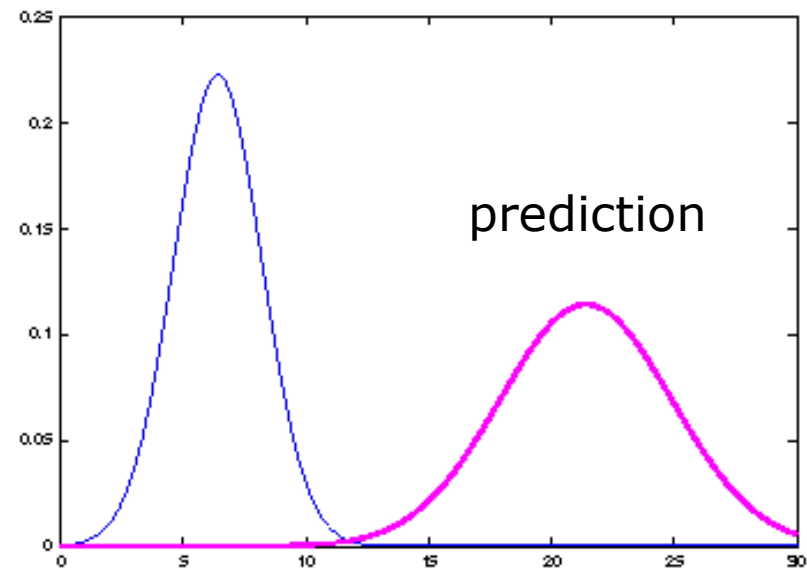
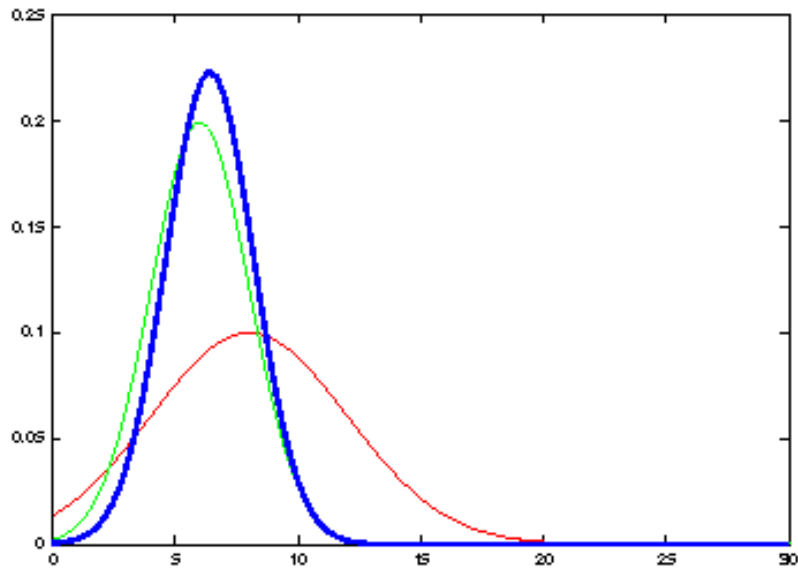
7: *return* μ_t, Σ_t

1D Kalman Filter Example (1)



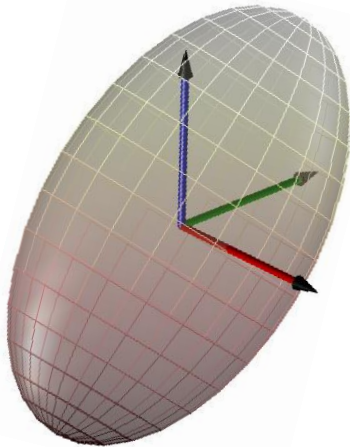
It's a weighted mean!

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

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



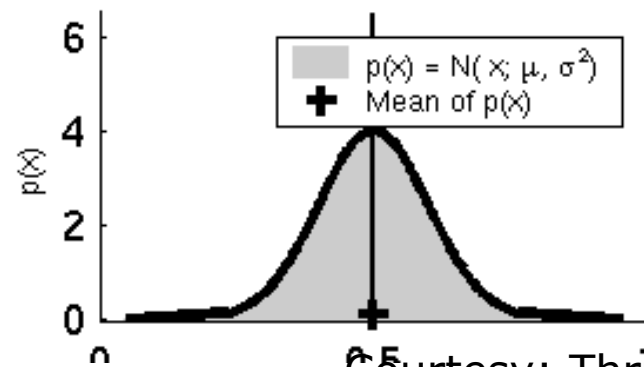
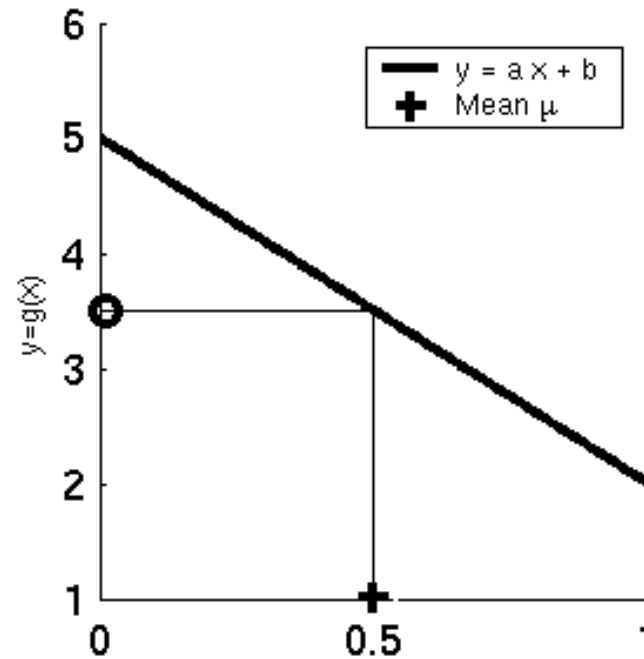
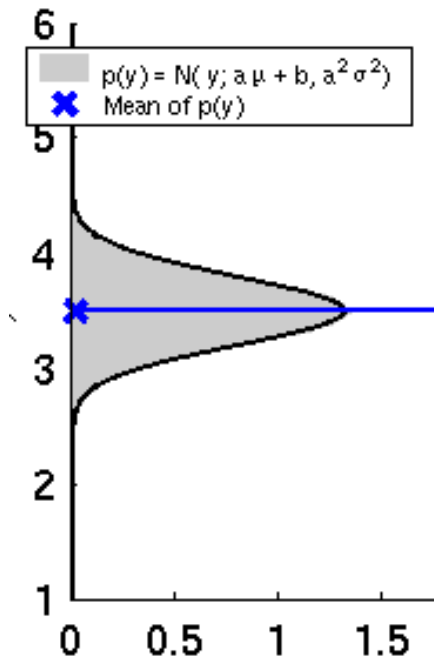
$$x_t = g(u_t, x_{t-1}) + \epsilon_t$$

$$\cancel{z_t = C_t x_t + \delta_t}$$

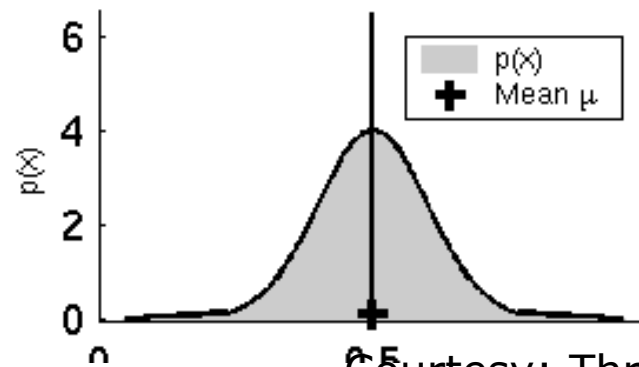
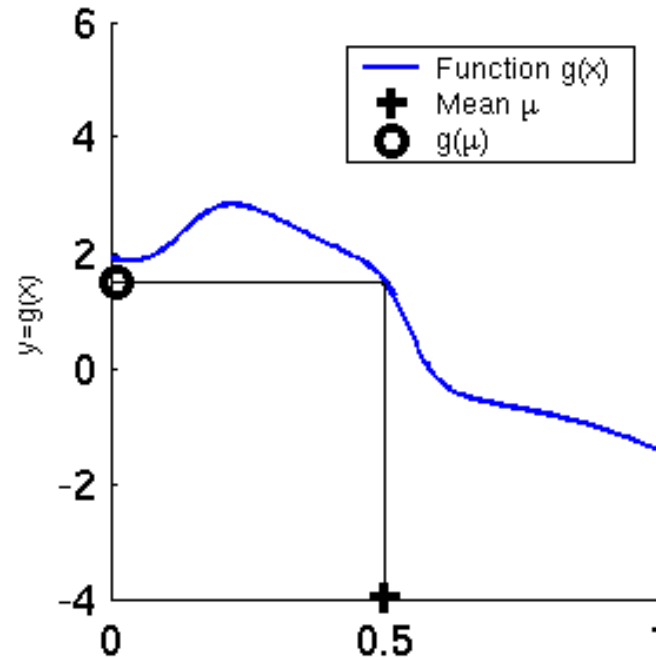
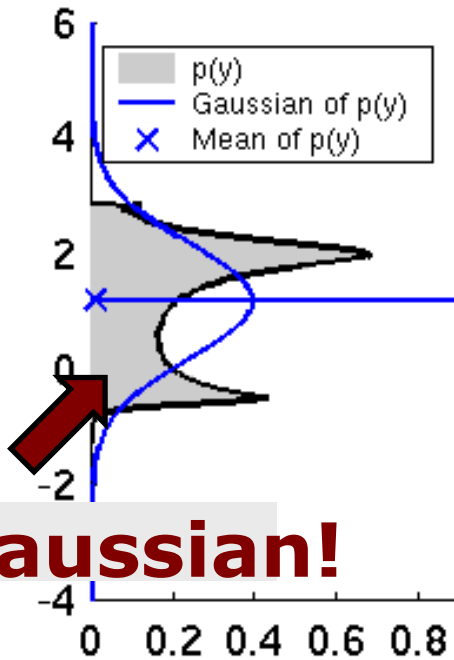


$$z_t = h(x_t) + \delta_t$$

Linearity Assumption Revisited



Non-Linear Function



Non-Gaussian!

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

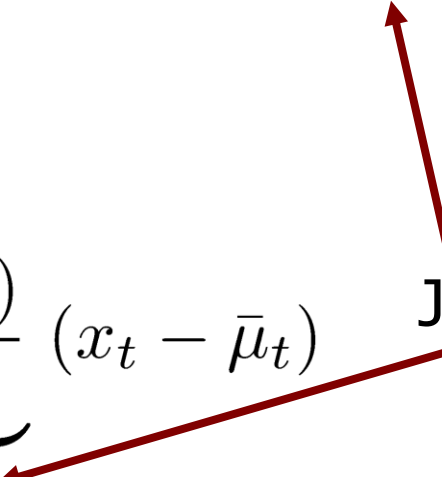
- Prediction:

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

- 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

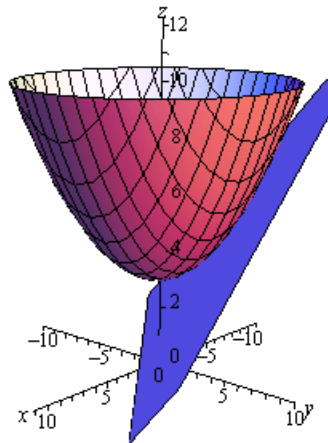
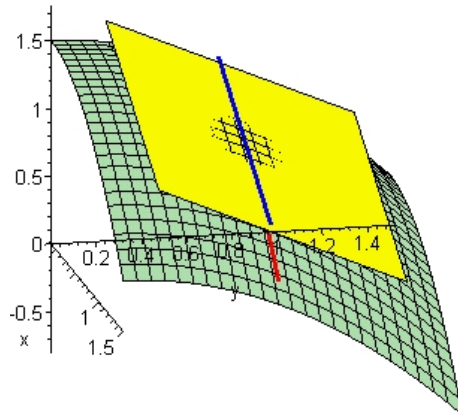
$$g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix}$$

- 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

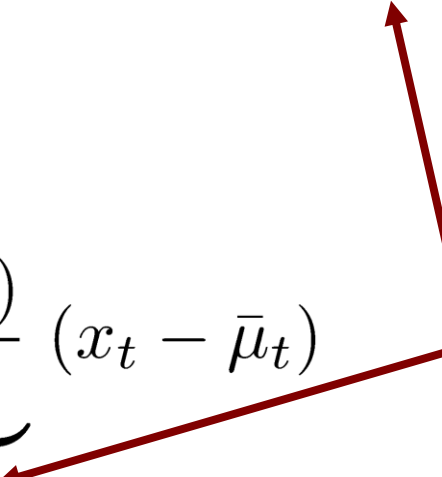
- Prediction:

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

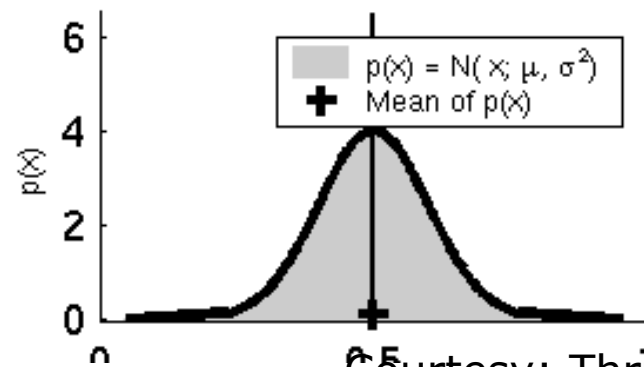
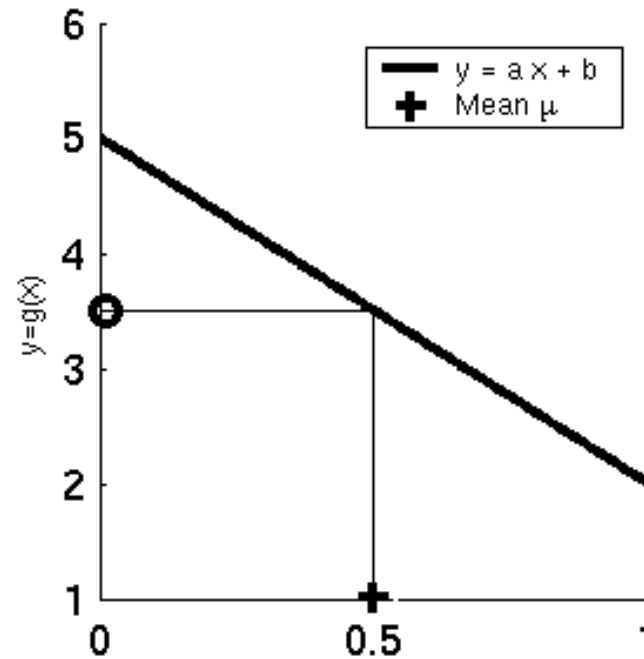
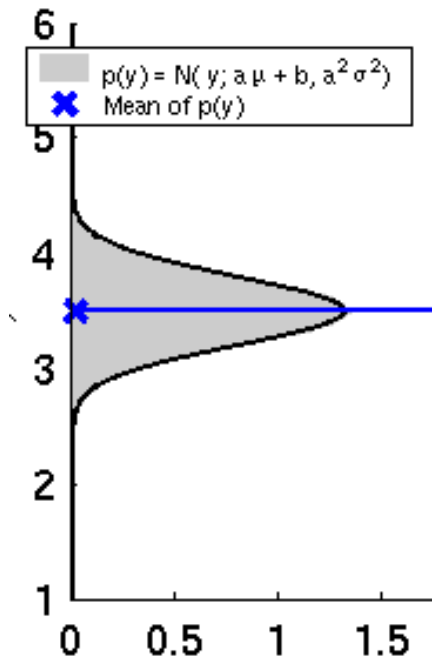
- Correction:

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

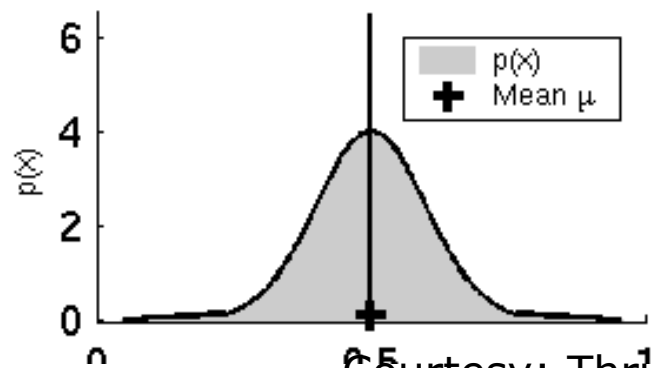
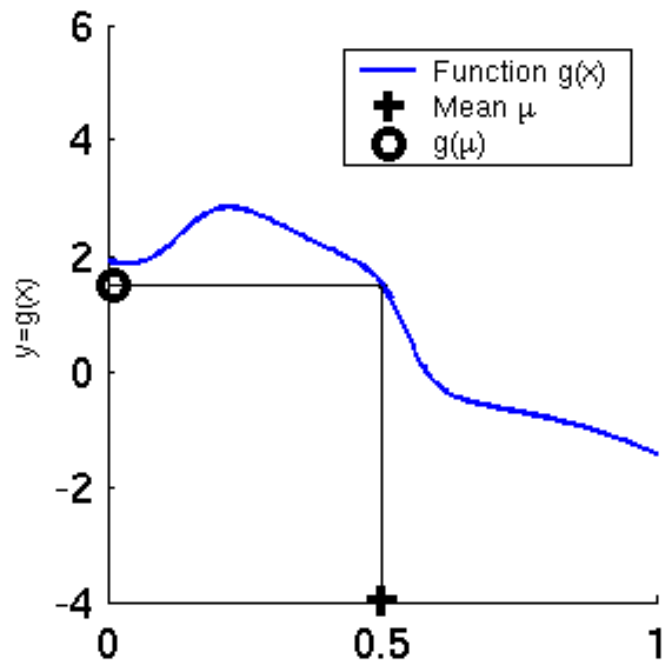
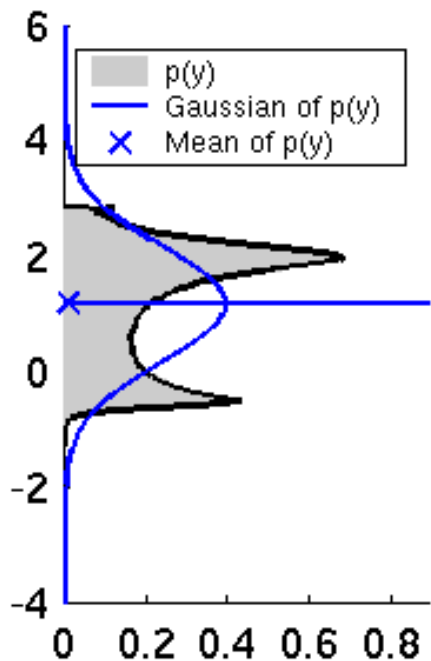
Linear functions!



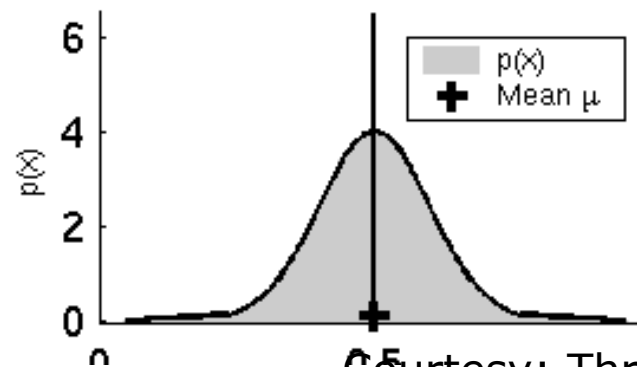
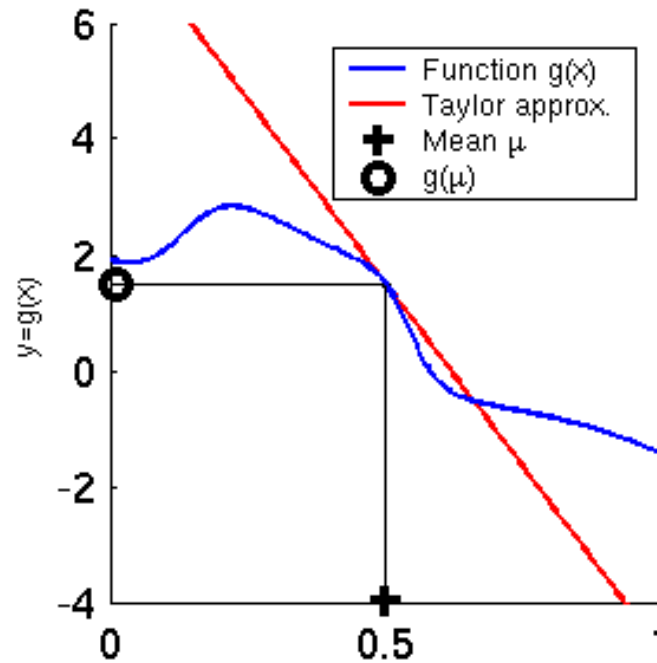
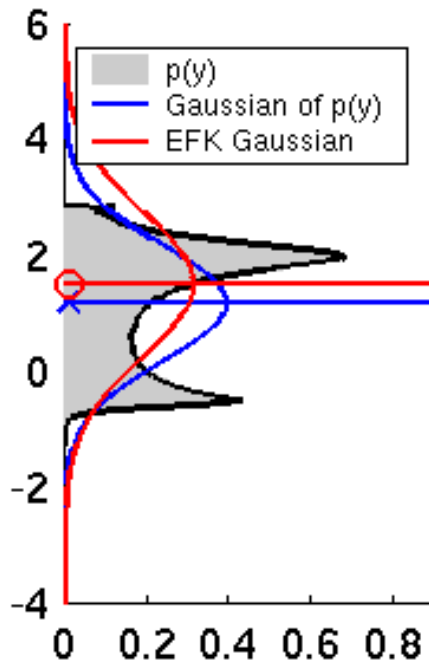
Linearity Assumption Revisited



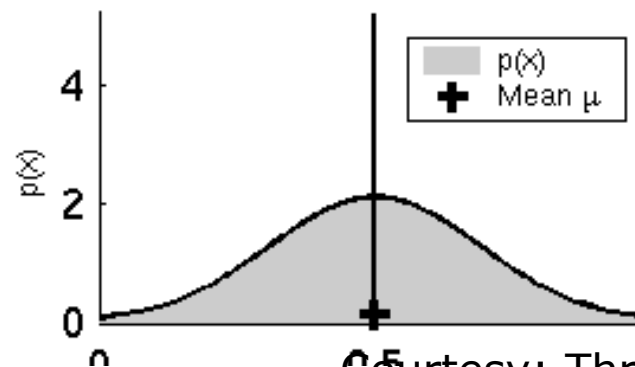
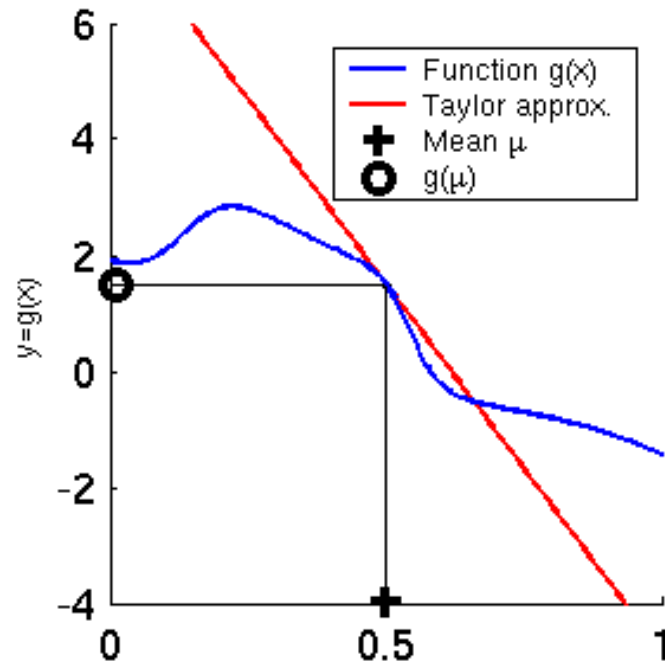
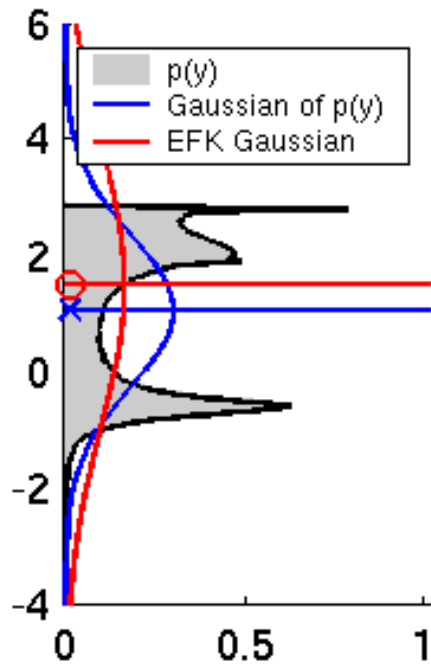
Non-Linear Function



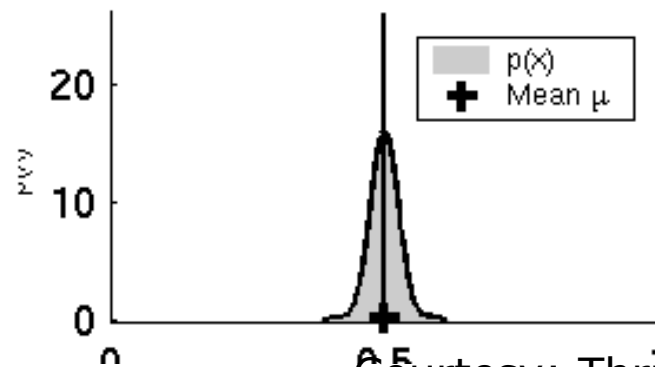
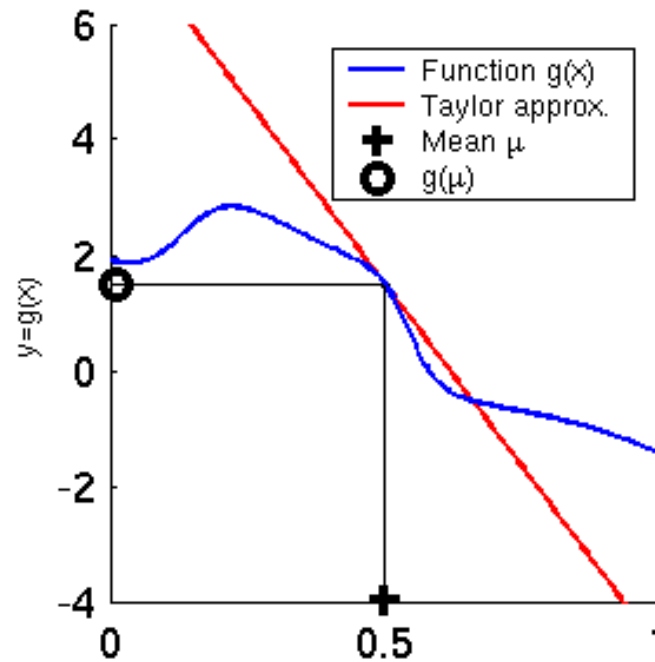
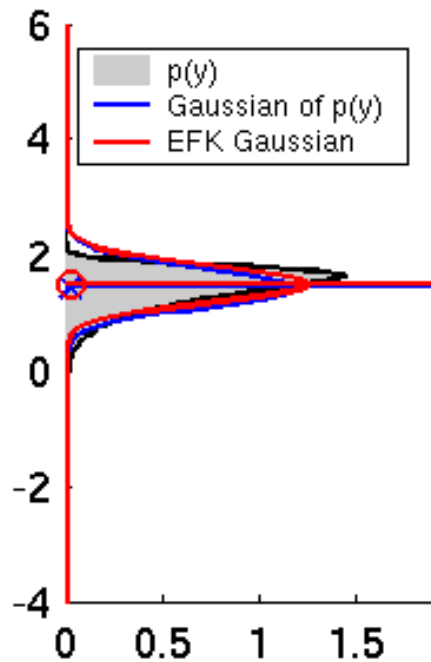
EKF Linearization (1)



EKF Linearization (2)



EKF Linearization (3)



Linearized Motion Model

- The linearized model leads to

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

- R_t describes the noise of the motion

Linearized Observation Model

- The linearized model leads to

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

- Q_t describes the measurement noise

Extended Kalman Filter Algorithm

- 1: **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

$$A_t \leftrightarrow G_t$$

$$C_t \leftrightarrow H_t$$

KF vs. EKF

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”

Slide Information

- These slides have been created by Cyrill Stachniss as part of the robot mapping course taught in 2012/13 and 2013/14. I created this set of slides partially extending existing material from courses of Wolfram Burgard, Dieter Fox, and myself.
- I tried to acknowledge all people that contributed image or video material. In case I missed something, please let me know. If you adapt this course material, please make sure you keep the acknowledgements.
- Feel free to use and change the slides. If you use them, I would appreciate an acknowledgement as well. To satisfy my own curiosity, I appreciate a short email notice in case you use the material in your course.
- My video recordings are available through YouTube:
http://www.youtube.com/playlist?list=PLgnQpQtFTOGQrZ4O5QzbIHgl3b1JHimN_&feature=g-list

Cyrill Stachniss, 2014
cyrill.stachniss@igg.uni-
bonn.de 38