Photogrammetry & Robotics Lab

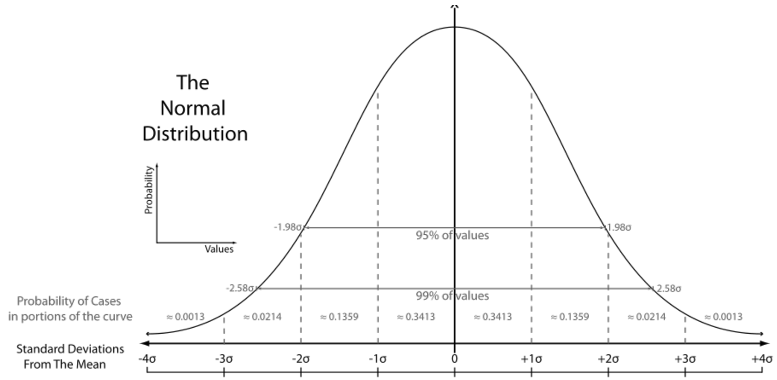
Robust Least Squares for SLAM

Cyrill Stachniss

Partial slide courtesy: Nived Chebrolu, Pratik Agarwal

Least Squares Minimization

- Minimizes **sum of squared errors**
- ML estimation for the Gaussian case



Least Squares Minimization

- Minimizes **sum of squared errors**
- ML estimation for the Gaussian case
- **Key assumption: No outliers!**

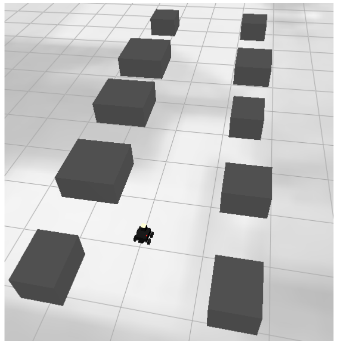
Problems:

- Outliers and ambiguities always occur in the real world
- Optimization is sensitive to outliers
- Gaussian distributions (one mode)

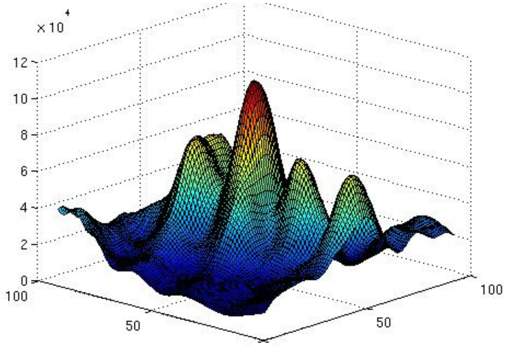
Data Association Is Ambiguous And Not Always Perfect

- Places that look identical
- Similar rooms in the same building
- Cluttered scenes
- GPS multi path (signal reflections)
- …

Example



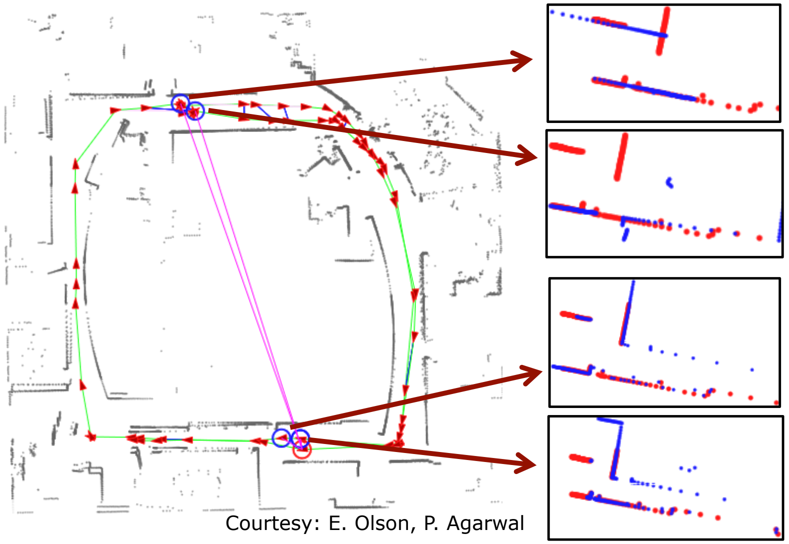
3D world



belief about the robot's pose

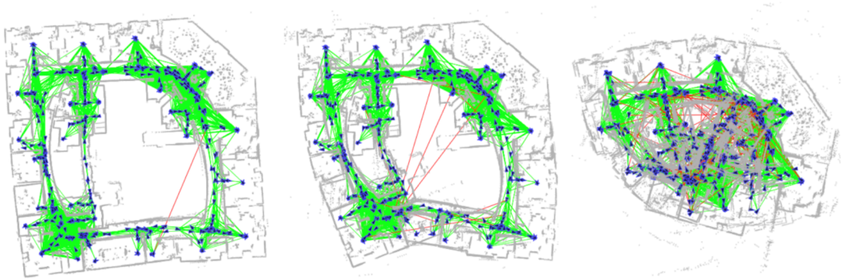
Courtesy: E. Olson

Ambiguities



Courtesy: E. Olson, P. Agarwal

Committing To The Wrong Mode Can Lead to Mapping Failures



Courtesy: E. Olson, P. Agarwal

Data Association Is Ambiguous And Not Always Perfect

- Places that look identical
- Similar rooms in the same building
- Cluttered scenes
- GPS multi path (signal reflections)
- …

How to deal with this problem in graph-based SLAM?

MaxMixtures
or Dealing with Multiple Modes

Mathematical Model

- Can we formulate constraints modeling Gaussian noise differently?

$$p(\mathbf{z} \mid \mathbf{x}) = \eta \exp(-\frac{1}{2}\mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij})$$

Mathematical Model

- We can express a multi-modal belief by a sum of Gaussians

$$p(\mathbf{z} \mid \mathbf{x}) = \eta \exp(-\frac{1}{2}\mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij})$$

$$\Downarrow$$

$$p(\mathbf{z} \mid \mathbf{x}) = \sum_k w_k \eta_k \exp(-\frac{1}{2}\mathbf{e}_{ij_k}^T \mathbf{\Omega}_{ij_k} \mathbf{e}_{ij_k})$$

Sum of Gaussians with k modes

Problem

- During error minimization, we consider the negative log likelihood

$$-\log p(\mathbf{z} \mid \mathbf{x}) = \frac{1}{2}\mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij} - \log \eta$$

$$\Downarrow$$

$$-\log p(\mathbf{z} \mid \mathbf{x}) = -\log \sum_k w_k \eta_k \exp(-\frac{1}{2}\mathbf{e}_{ij_k}^T \mathbf{\Omega}_{ij_k} \mathbf{e}_{ij_k})$$

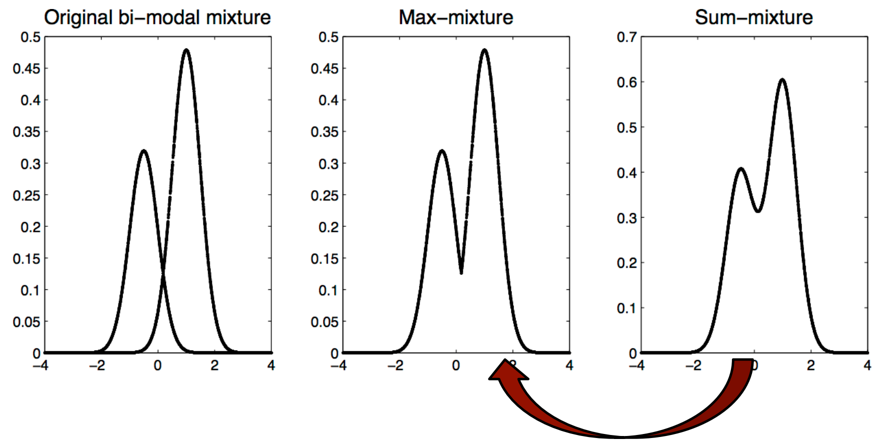
The log cannot be moved inside the sum!

Max-Mixture Approximation

- Instead of computing the sum of Gaussians at $\mathbf{x}$, compute the maximum of the Gaussians

$$
\begin{aligned}
p(\mathbf{z} \mid \mathbf{x}) &= \sum_k w_k \eta_k \exp(-\frac{1}{2}\mathbf{e}_{ij_k}^T \boldsymbol{\Omega}_{ij_k} \mathbf{e}_{ij_k}) \\
&\simeq \max_k w_k \eta_k \exp(-\frac{1}{2}\mathbf{e}_{ij_k}^T \boldsymbol{\Omega}_{ij_k} \mathbf{e}_{ij_k})
\end{aligned}
$$

Max-Mixture Approximation



approximation error

Log Likelihood Of The Max-Mixture Formulation

- The log can be moved inside the max operator

$$
p(\mathbf{z} \mid \mathbf{x}) \simeq \max_k w_k \eta_k \exp(-\frac{1}{2}\mathbf{e}_{ij_k}^T \boldsymbol{\Omega}_{ij_k} \mathbf{e}_{ij_k})
$$

$$
\log p(\mathbf{z} \mid \mathbf{x}) \simeq \max_k -\frac{1}{2}\mathbf{e}_{ij_k}^T \boldsymbol{\Omega}_{ij_k} \mathbf{e}_{ij_k} + \log(w_k \eta_k)
$$

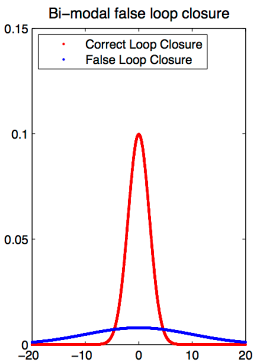
$$
\text{or:} \quad -\log p(\mathbf{z} \mid \mathbf{x}) \simeq \min_k \frac{1}{2}\mathbf{e}_{ij_k}^T \boldsymbol{\Omega}_{ij_k} \mathbf{e}_{ij_k} - \log(w_k \eta_k)
$$

Integration

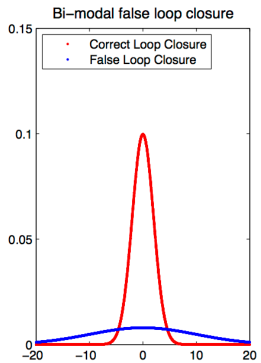
- With the max-mixture formulation, the log likelihood again results in local quadratic forms
- Easy to integrate in the optimizer:

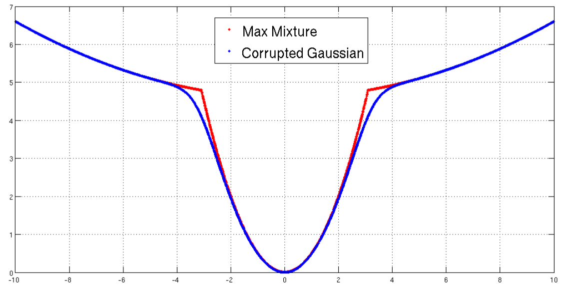
1. Evaluate all k components
2. Select the component with the maximum log likelihood
3. Perform the optimization as before using only the max components (as a single Gaussian)

MM For Outlier Rejection

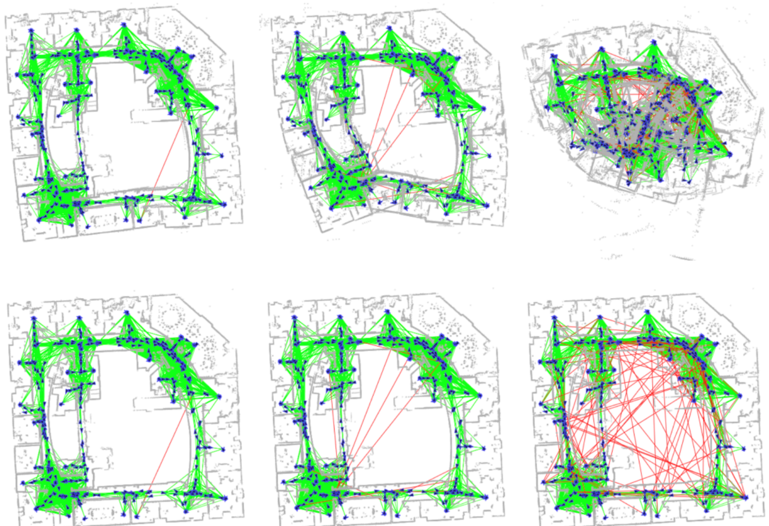


MM For Outlier Rejection

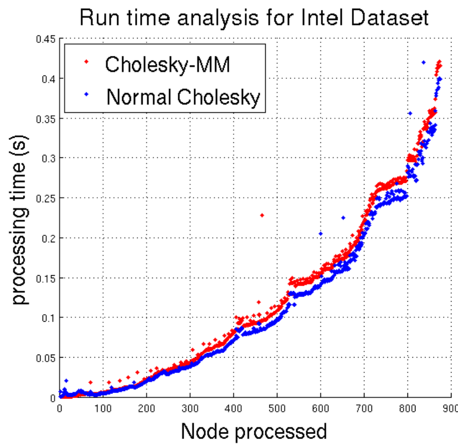




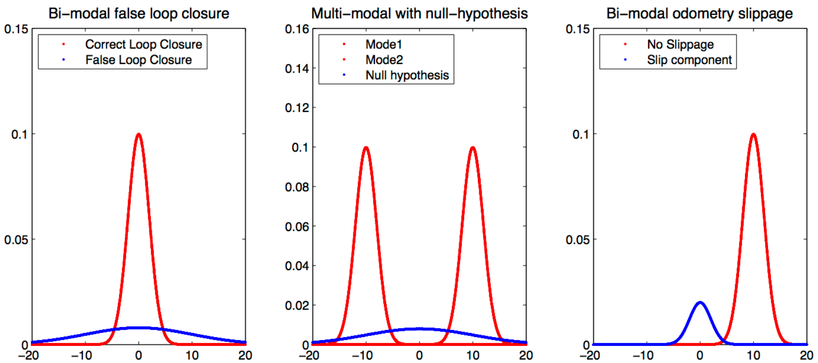
Performance (Gauss vs. MM)



Runtime



MM For Outlier Rejection and Data Association Ambiguities

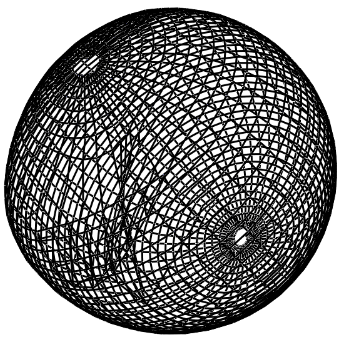


Courtesy: E. Olson, P. Agarwal

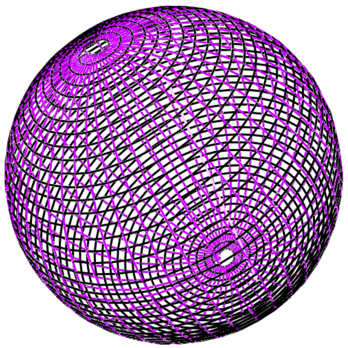
Max-Mixture and Outliers

- MM formulation is useful for multi-model constraints (D.A. ambiguities)
- MM is also a handy tool for dealing with outliers
- Outliers: one mode represents the main constraint and a second model uses a flat Gaussian for the outlier hypothesis

Performance (1 outlier)

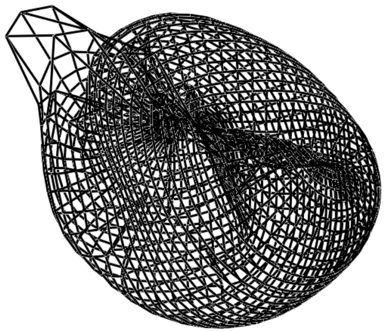


Gauss-Newton

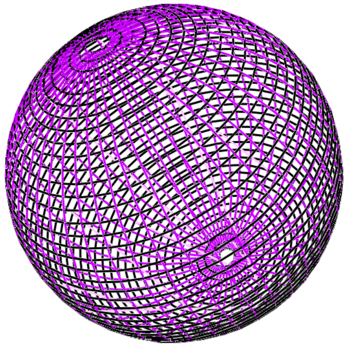


MM Gauss-Newton

Performance (10 outliers)

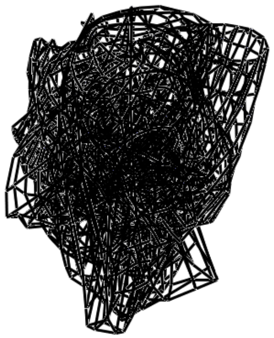


Gauss-Newton

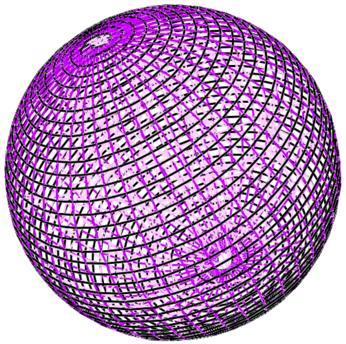


MM Gauss-Newton

Performance (100 outliers)



Gauss-Newton



MM Gauss-Newton

MaxMixtues for Dealing with Outliers

- Supports multi-model constraints
- Approximate the sum of Gaussians using the max operator
- Idea: "Select the best mode of a sum of Gaussians and use it as if it would be a single Gaussian"
- Easy to use, quite effective

Dynamic Covariance Scaling

Standard Least Squares

$$X^* = \operatorname*{argmin}_X \sum_{ij} \underbrace{\mathbf{e}_{ij}(X)^T \Omega_{ij} \mathbf{e}_{ij}(X)}_{\chi^2_{ij}}$$

Dynamic Covariance Scaling

$$X^* = \underset{X}{\operatorname{argmin}} \sum_{ij} \underbrace{\mathbf{e}_{ij}(X)^T \Omega_{ij} \mathbf{e}_{ij}(X)}_{\chi^2_{ij}}$$

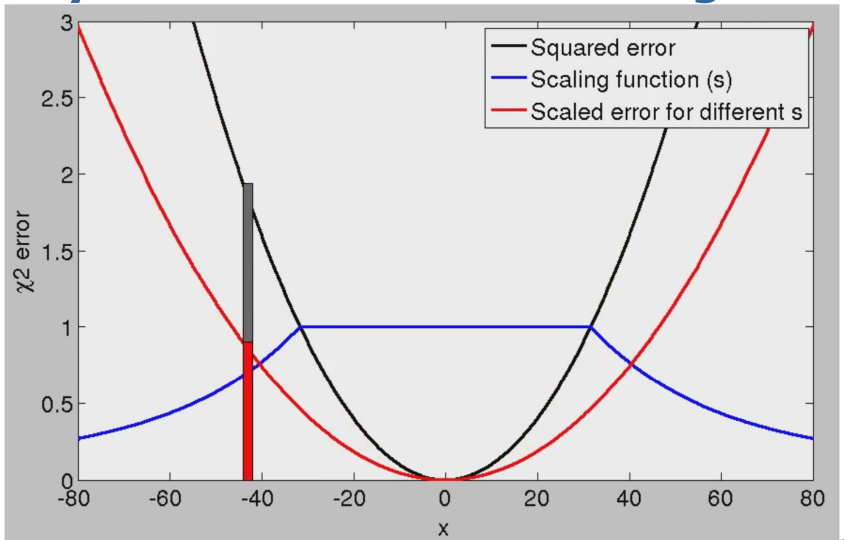
$$X^* = \underset{X}{\operatorname{argmin}} \sum_{ij} \mathbf{e}_{ij}(X)^T \left(s_{ij}^2 \Omega_{ij}\right) \mathbf{e}_{ij}(X)$$

Scaling Parameter

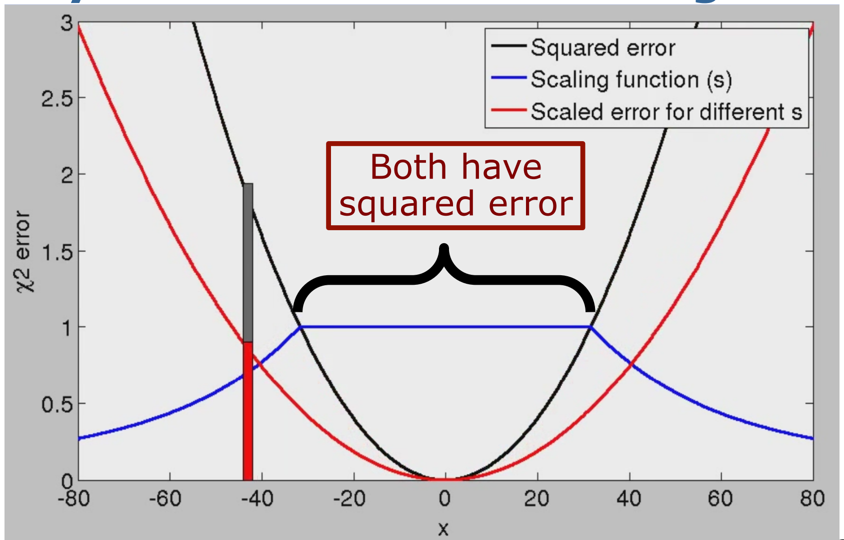
$$X^* = \underset{X}{\operatorname{argmin}} \sum_{ij} \mathbf{e}_{ij}(X)^T \left(s_{ij}^2 \Omega_{ij}\right) \mathbf{e}_{ij}(X)$$

$$s_{ij} = \min\left(1, \frac{2\Phi}{\Phi + \chi^2_{ij}}\right)$$

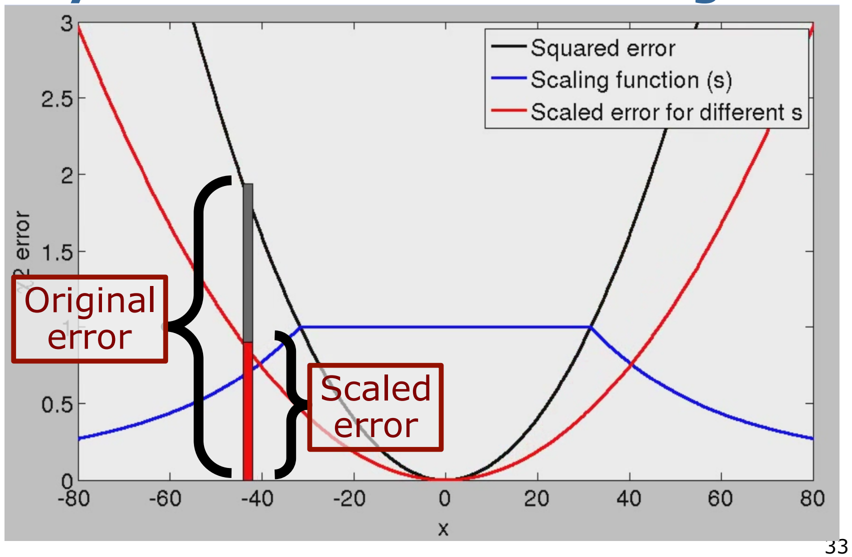
Dynamic Covariance Scaling



Dynamic Covariance Scaling

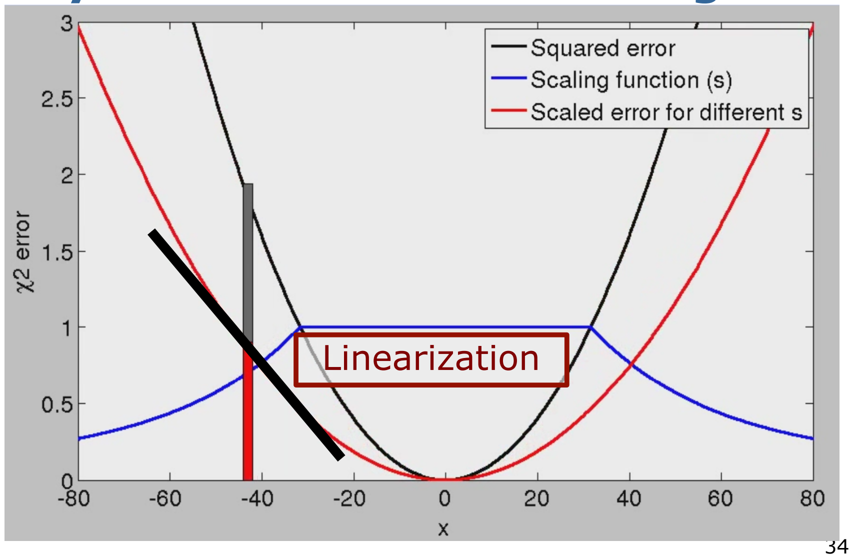


Dynamic Covariance Scaling



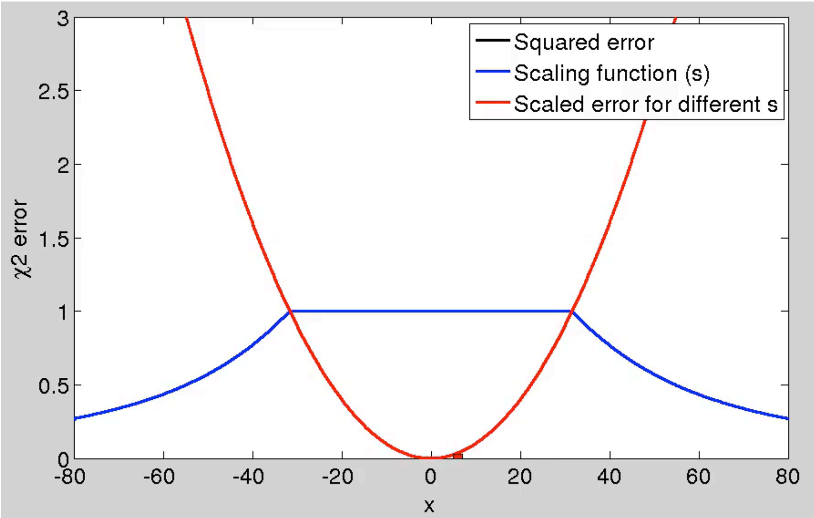
33

Dynamic Covariance Scaling



34

Dynamic Covariance Scaling



35

DCS for Dealing with Outliers

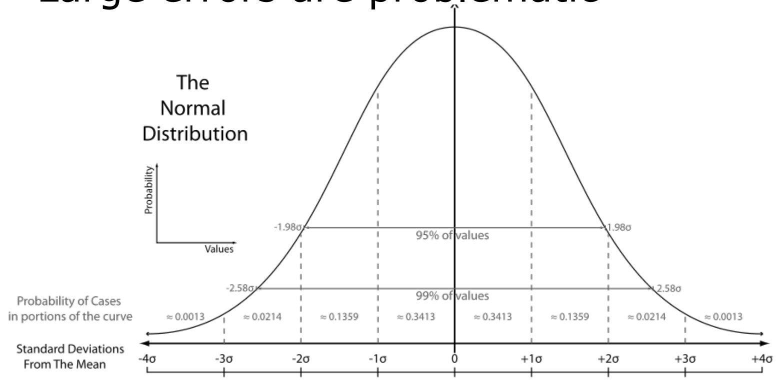
- Add an additional weighting term to the error function
- The weight depends on the error value
- Idea: "Weight down constraints that are far away from the mean estimate"
- A special case of robust least squares estimation (Geman-McClure kernel)

36

Least Squares with Robust Kernels

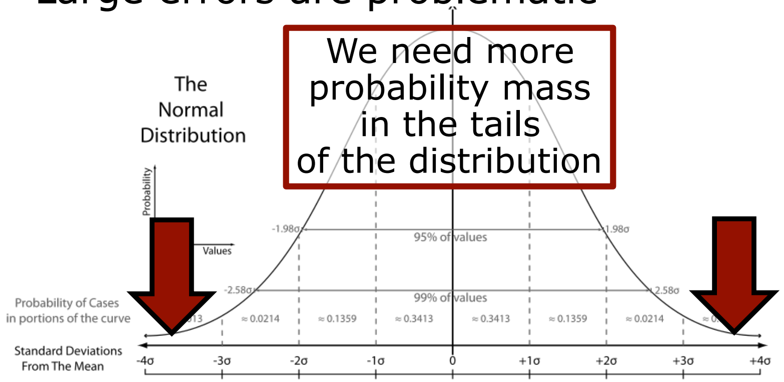
Optimizing With Outliers

- Assuming a Gaussian error in the constraints is not always realistic
- Large errors are problematic



Optimizing With Outliers

- Assuming a Gaussian error in the constraints is not always realistic
- Large errors are problematic



Robust M-Estimators

- Assume non-normally-distributed noise
- Intuitively: PDF with "heavy tails"
- $\rho(e)$ function used to define the PDF

$$p(e) = \exp(-\rho(e))$$

- Minimizing the neg. log likelihood

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i \rho(e_i(\mathbf{x}))$$

Robust M-Estimators: Gaussian Case

- Kernel function $\rho(e)$ used to define the PDF

$$p(e) = \exp(-\rho(e))$$

- For the Gaussian case, we set $\rho(e)$ to be a quadratic function $\rho(e) = e^2$

Different Rho Functions

- Gaussian: $\rho(e) = e^2$
- Absolute values (L1 norm): $\rho(e) = |e|$
- Huber M-estimator

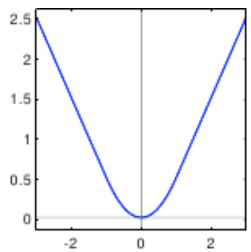
$$\rho(e) = \begin{cases} \frac{e^2}{2} & \text{if } |e| < c \\ c(|e| - \frac{c}{2}) & \text{otherwise} \end{cases}$$

- Several others (Tukey, Cauchy, Blake-Zisserman, Corrupted Gaussian, …)

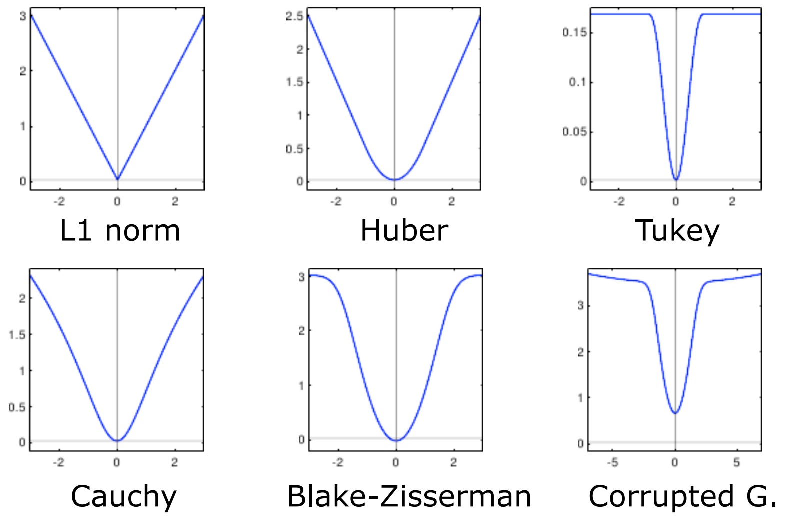
Huber Loss

- Mixture of a quadratic and a linear function
- Quadratic around the solution (noise)
- Linear for outliers (error > threshold)

$$\rho(e) = \begin{cases} \frac{e^2}{2} & \text{if } |e| < c \\ c(|e| - \frac{c}{2}) & \text{otherwise} \end{cases}$$



Different Robust Loss Functions



Robust Estimation as Weighted Least Squares

- Weighted Least Squares

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{N} w_i \|e_i(\mathbf{x})\|^2$$

- Robust Estimation

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{N} \rho(e_i(\mathbf{x}))$$

Robust Estimation as Weighted Least Squares

- Gradient at optimum goes to zero
- For **weighted least squares**:

$$\frac{1}{2} \frac{\partial(w_i e_i^2(\mathbf{x}))}{\partial \mathbf{x}} = w_i e_i(\mathbf{x}) \frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

Robust Estimation as Weighted Least Squares

- Gradient at optimum goes to zero
- For the **robust estimation**:

$$\frac{\partial(\rho(e_i(\mathbf{x})))}{\partial \mathbf{x}} = \rho'(e_i(\mathbf{x})) \frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

Robust Estimation as Weighted Least Squares

- Compare both equations

$$\frac{1}{2} \frac{\partial(w_i e_i^2(\mathbf{x}))}{\partial \mathbf{x}} = w_i e_i(\mathbf{x}) \frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

$$\frac{\partial(\rho(e_i(\mathbf{x})))}{\partial \mathbf{x}} = \rho'(e_i(\mathbf{x})) \frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

Robust Estimation as Weighted Least Squares

- Compare both equations

$$\frac{1}{2}\frac{\partial(w_i e_i^2(\mathbf{x}))}{\partial \mathbf{x}} = w_i e_i(\mathbf{x})\frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

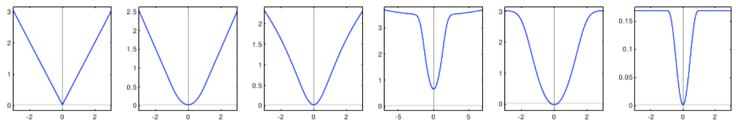
$$\frac{\partial(\rho(e_i(\mathbf{x})))}{\partial \mathbf{x}} = \rho'(e_i(\mathbf{x}))\frac{\partial e_i(\mathbf{x})}{\partial \mathbf{x}} = 0$$

- We can use weighted least squares if we set the weight using the kernel as:

$$\boxed{w_i = \frac{1}{e_i(\mathbf{x})}\rho'(e_i(\mathbf{x}))}$$

Robust Least Squares

- We can use the weighted least squares approach to realize robust L.S.
- The kernel will impact the Jacobians
- The rest stays the same
- The choice of the kernel must align with the outlier distribution



Generalized Robust Kernels

Which Function to Chose?

- Which loss/kernel function to chose?

Which Function to Chose?

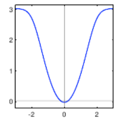
- Which loss/kernel function to chose?
- **It depends on the type of outliers!**

Which Function to Chose?

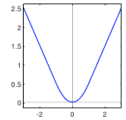
- Which loss/kernel function to chose?
- **It depends on the type of outliers!**

Some approaches combine them:

1. Start with a strong tails for N_1 iterations
2. N_2 iteration with weaker tails
3. Remove all outliers larger c
4. Gaussian/Huber for the rest

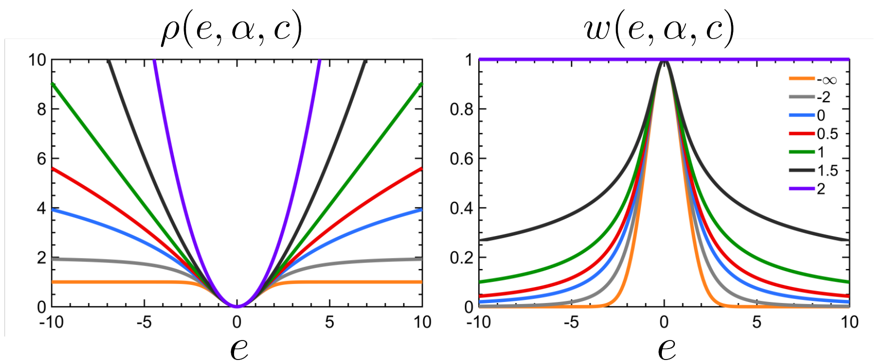


...



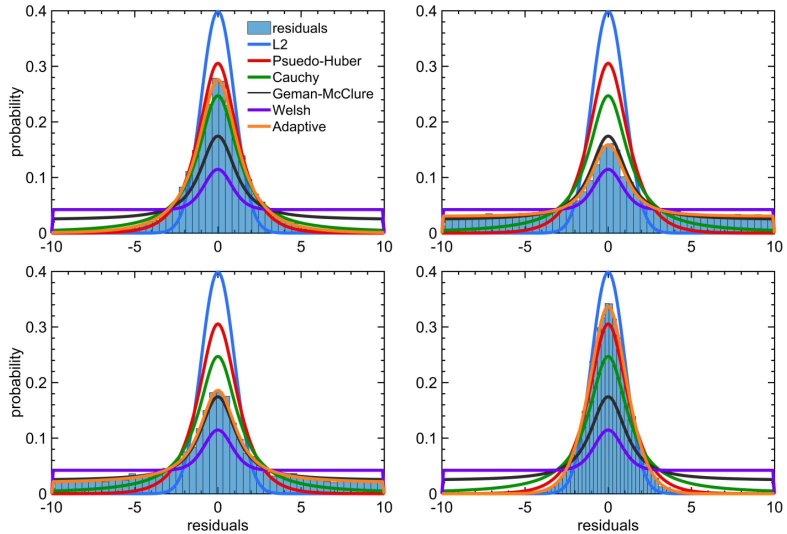
General Robust Loss Function

$$\rho(e, \alpha, c) \quad = \quad \frac{|\alpha - 2|}{\alpha}\left(\left(\frac{(e/c)^2}{|\alpha - 2|} + 1\right)^{\alpha/2} - 1\right)$$

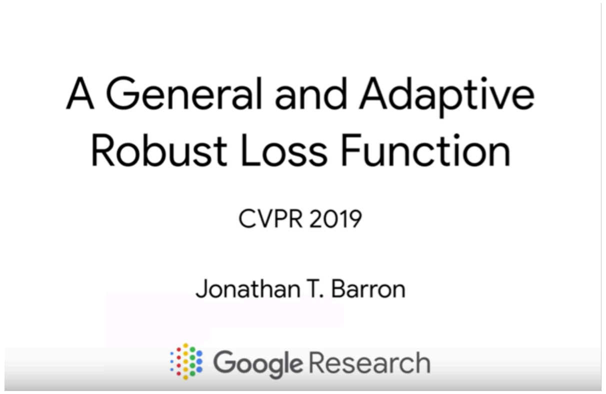


Source: A General and Adaptive Robust Loss Function, Barron [CVPR 2019]

Adaptive Robust Loss Function

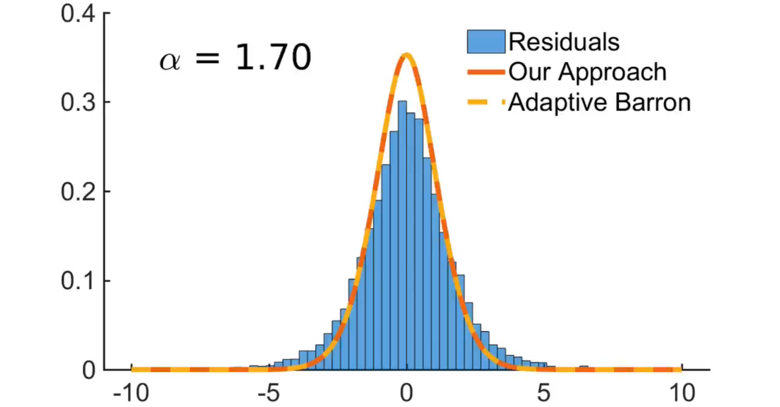


Recommended Video to Watch



https://www.youtube.com/watch?v=BmNKbnF69eY

Adaptive Robust Loss Function



Adaptive Robust Loss Function

- Jointly optimize over $\mathbf{x}$ and α

$$(\mathbf{x}^*, \alpha^*) = \underset{(\mathbf{x},\alpha)}{\operatorname{argmin}} \sum_{i=1}^{N} \rho(e_i(\mathbf{x}), \alpha).$$

- Define a probability distribution for the general loss function

$$P(e, \alpha, c) = \frac{1}{cZ(\alpha)} e^{-\rho(e,\alpha,c)}$$

$$Z(\alpha) = \int_{-\infty}^{\infty} e^{-\rho(e,\alpha,1)}\, de$$

Adaptive Robust Loss Function

- Adaptive loss function defined as

$$\begin{aligned} \rho_a(e, \alpha, c) &= -\log P(e, \alpha, c) \\ &= \rho(e, \alpha, c) + \log cZ(\alpha) \end{aligned}$$

- with

$$P(e, \alpha, c) = \frac{1}{cZ(\alpha)} e^{-\rho(e,\alpha,c)}$$

$$Z(\alpha) = \int_{-\infty}^{\infty} e^{-\rho(e,\alpha,1)}\, de$$

Adaptive Robust Loss Function

- Adaptive loss function defined as

$$\begin{aligned}\rho_a(e,\alpha,c) &= -\log P(e,\alpha,c)\\ &= \rho(e,\alpha,c)+\log cZ(\alpha)\end{aligned}$$

- with

$$\begin{aligned}P(e,\alpha,c) &= \frac{1}{cZ(\alpha)}e^{-\rho(e,\alpha,c)}\\ Z(\alpha) &= \boxed{\int_{-\infty}^{\infty} e^{-\rho(e,\alpha,1)}\,de}\end{aligned}$$

approaches infinity for negative α

Adaptive Robust Loss Function

- Adaptive loss function defined as

$$\begin{aligned}\rho_a(e,\alpha,c) &= -\log P(e,\alpha,c)\\ &= \rho(e,\alpha,c)+\log cZ(\alpha)\end{aligned}$$

- with

$$\begin{aligned}P(e,\alpha,c) &= \frac{1}{cZ(\alpha)}e^{-\rho(e,\alpha,c)}\\ Z(\alpha) &= \int_{\boxed{-\infty}}^{\boxed{\infty}} e^{-\rho(e,\alpha,1)}\,de\end{aligned}$$

We can limit the range of outliers to maintain a pdf

Adaptive Robust Loss Function

- Adaptive loss function defined as

$$\begin{aligned}\rho_a(e,\alpha,c) &= -\log P(e,\alpha,c)\\ &= \rho(e,\alpha,c)+\log cZ(\alpha)\end{aligned}$$

- with

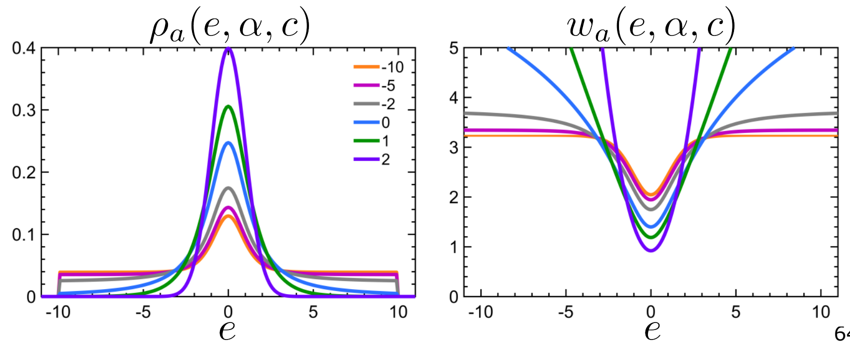
$$\begin{aligned}P(e,\alpha,c) &= \frac{1}{cZ(\alpha)}e^{-\rho(e,\alpha,c)}\\ Z(\alpha) &= \boxed{\int_{-\tau}^{\tau} e^{-\rho(e,\alpha,1)}\,de}\end{aligned}$$

We can limit the range of outliers to maintain a pdf

Adaptive Robust Loss Function

- Adaptive loss function defined as

$$\begin{aligned}\rho_a(e,\alpha,c) &= -\log P(e,\alpha,c)\\ &= \rho(e,\alpha,c)+\log cZ(\alpha)\end{aligned}$$



Joint Optimization with the Adaptive Robust Kernel

- In theory, we can now solve a joint optimization problem

$$(\mathbf{x}^*, \alpha^*) \quad = \quad \underset{(\mathbf{x},\alpha)}{\operatorname{argmin}} \sum_{i=1}^{N} \rho(e_i(\mathbf{x}), \alpha).$$

- in the weighted least squares sense using $\rho_a(e, \alpha, c)$ as our robust kernel

Joint Optimization with the Adaptive Robust Kernel

- Joint optimization of $(\mathbf{x},\alpha)$:

$$(\mathbf{x}^*, \alpha^*) \quad = \quad \underset{(\mathbf{x},\alpha)}{\operatorname{argmin}} \sum_{i=1}^{N} \rho(e_i(\mathbf{x}), \alpha).$$

Problems in practice:

- New Jacobians need to be computed
- α can dominate the parameter estimation for complex problems
- Sensitive to initial guess

EM-Based Optimization with the Adaptive Robust Kernel

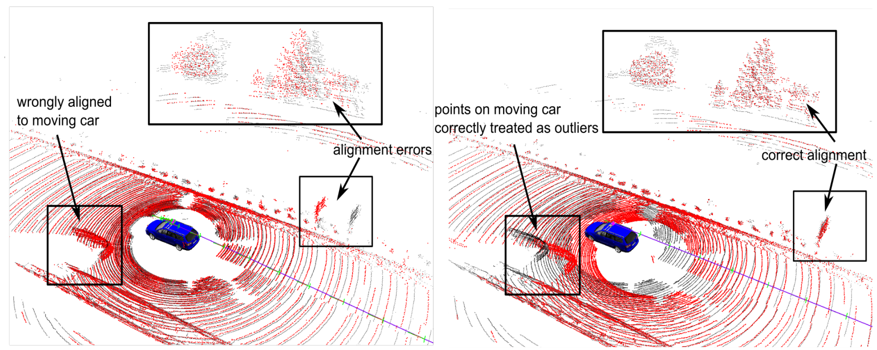
- Solve via Expectation-Maximization
- E-Step: 1D line search problem

$$\alpha^t = \underset{\alpha}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(e_i(\mathbf{x}^{t-1}), \alpha^{t-1}, c)$$

- M-Step: Minimize as weighted least squares

$$\mathbf{x}^t = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{i=1}^{N} \rho_a(e_i(\mathbf{x}), \alpha^t, c)$$

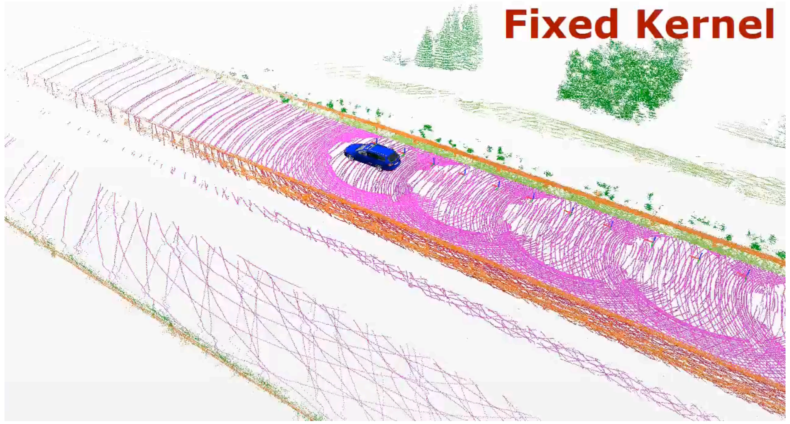
Iterative Closest Point Example



Huber Loss — Adaptive Robust Loss

Outlier rejection in presence of dynamic objects

Iterative Closest Point Example



EM-Based Optimization with the Adaptive Robust Kernel

- Kernel and unknown are estimated in a iterative fashion
- EM-based estimation provides better results than the joint optimization
- Kernel function adapts to the current situation by adapting α
- No need to change the least squares problem (Jacobians stay the same)

Dealing with Outliers

- There are different ways for dealing with outliers during optimization
- Key question: how does the outlier distribution look like?

Dealing with Outliers Summary

- Max-Mixtures supports multi-model constraints
- It approximates the sum of Gaussians using the max operator
- Dynamic Covariance Scaling is a good choice for a fixed kernel in SLAM
- DCS is a form of Geman-McClure
- Choice of the robust loss function depends on the problem at hand

Dealing with Outliers Summary

- Robust least squares using kernels
- Choice of the rho function (kernel) depends on the problem at hand
- Popular: Huber, L1, DCS/Geman McClure
- Better: Don't commit on one kernel
- Adaptive robust kernel is the most flexible way
- We obtained best results with adaptive kernels in an EM-style estimation

Literature

Max-Mixtures:

- Olson, Agarwal: "Inference on Networks of Mixtures for Robust Robot Mapping"

Dynamic Covariance Scaling:

- Agarwal, Tipaldi, Spinello, Stachniss, Burgard: "Robust Map Optimization Using Dynamic Covariance Scaling"

General Robust Loss Function:

- Barron: "A General and Adaptive Robust Loss Function"

EM-based Estimation of Kernel and LS Problem:

- Chebrolu, Läbe, Vysotska, Behley, Stachniss: "Adaptive Robust Kernels for Non-Linear Least Squares Problems"

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 of Edwin Olson, Pratik Agarwal, 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