Robot Mapping

Least Squares

Gian Diego Tipaldi, Luciano Spinello, Wolfram Burgard

Three Main SLAM Paradigms

Kalman filter

Particle filter

Graph-based

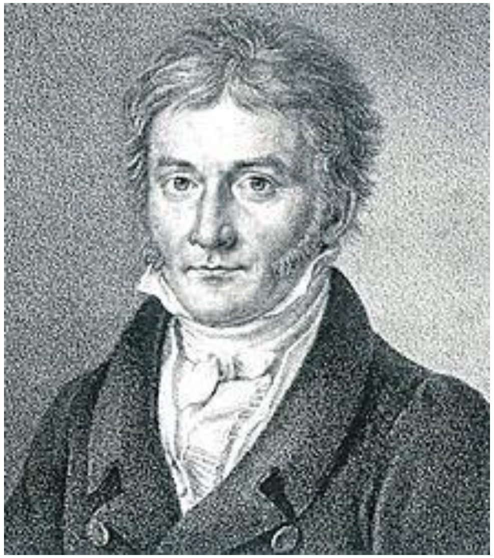
least squares approach to SLAM

Least Squares in General

- Approach for computing a solution for an **overdetermined system**
- “More equations than unknowns”
- Minimizes the **sum of the squared errors** in the equations
- Standard approach to a large set of problems

Least Squares History

- Method developed by Carl Friedrich Gauss in 1795 (he was 18 years old)
- First showcase: predicting the future location of the asteroid Ceres in 1801



Courtesy: Astronomische Nachrichten, 1828

Problem

- Given a system described by a set of n observation functions $\{f_i(\mathbf{x})\}_{i=1:n}$
- Let
  - $\mathbf{x}$ be the state vector
  - $\mathbf{z}_i$ be a measurement of the state $\mathbf{x}$
  - $\widehat{\mathbf{z}}_i = f_i(\mathbf{x})$ be a function which maps $\mathbf{x}$ to a predicted measurement $\widehat{\mathbf{z}}_i$
- Given n noisy measurements $\mathbf{z}_{1:n}$ about the state $\mathbf{x}$

Goal: Estimate the state $\mathbf{x}$ which bests explains the measurements $\mathbf{z}_{1:n}$

Graphical Explanation

$$\mathbf{x}$$

$$f_1(\mathbf{x}) = \hat{\mathbf{z}}_1 \qquad \mathbf{z}_1$$

$$f_2(\mathbf{x}) = \hat{\mathbf{z}}_2 \qquad \mathbf{z}_2$$

$$\cdots$$

$$f_n(\mathbf{x}) = \hat{\mathbf{z}}_n \qquad \mathbf{z}_n$$

state (unknown)

predicted measurements

real measurements

Example

$$
\mathbf{x} \;\rightarrow\;
\begin{array}{lcl}
f_1(\mathbf{x}) = \widehat{\mathbf{z}}_1 & \qquad & \mathbf{z}_1 \\
f_2(\mathbf{x}) = \widehat{\mathbf{z}}_2 & \qquad & \mathbf{z}_2 \\
\qquad \dots & & \\
f_n(\mathbf{x}) = \widehat{\mathbf{z}}_n & \qquad & \mathbf{z}_n
\end{array}
$$

- $\mathbf{x}$ position of 3D features
- $\mathbf{z}_i$ coordinates of the 3D features projected on camera images
- Estimate the most likely 3D position of the features based on the image projections (given the camera poses)

Error Function

- Error $\mathbf{e}_i$ is typically the **difference** between the **predicted and actual** measurement

$$\mathbf{e}_i(\mathbf{x}) \quad = \quad \mathbf{z}_i - f_i(\mathbf{x})$$

- We assume that the error has **zero mean** and is **normally distributed**
- Gaussian error with information matrix $\mathbf{\Omega}_i$
- The squared error of a measurement depends only on the state and is a scalar

$$e_i(\mathbf{x}) \quad = \quad \mathbf{e}_i(\mathbf{x})^T \mathbf{\Omega}_i \mathbf{e}_i(\mathbf{x})$$

Goal: Find the Minimum

- Find the state $\mathbf{x}^*$ which minimizes the error given all measurements

$$
\begin{aligned}
\mathbf{x}^* &= \underset{\mathbf{x}}{\operatorname{argmin}}\, F(\mathbf{x}) \\
&= \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i e_i(\mathbf{x}) \\
&= \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i \mathbf{e}_i^T(\mathbf{x})\boldsymbol{\Omega}_i\mathbf{e}_i(\mathbf{x})
\end{aligned}
$$

global error (scalar)

squared error terms (scalar)

error terms (vector)

Goal: Find the Minimum

- Find the state $\mathbf{x}^*$ which minimizes the error given all measurements

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \sum_i \mathbf{e}_i^T(\mathbf{x}) \boldsymbol{\Omega}_i \mathbf{e}_i(\mathbf{x})$$

- A general solution is to derive the global error function and find its nulls
- In general complex and no closed form solution

$\Rightarrow$ Numerical approaches

Assumption

- A “good” initial guess is available
- The error functions are “smooth” in the neighborhood of the (hopefully global) minima

- Then, we can solve the problem by iterative local linearizations

Solve Via Iterative Local Linearizations

- Linearize the error terms around the current solution/initial guess
- Compute the first derivative of the squared error function
- Set it to zero and solve linear system
- Obtain the new state (that is hopefully closer to the minimum)
- Iterate

Linearizing the Error Function

- Approximate the error functions around an initial guess $\hat{\mathbf{x}}$ via Taylor expansion

$$\mathbf{e}_i(\hat{\mathbf{x}} + \mathbf{\Delta x}) \quad \simeq \quad \underbrace{\mathbf{e}_i(\hat{\mathbf{x}})}_{\mathbf{e}_i} + \mathbf{J}_i(\hat{\mathbf{x}})\mathbf{\Delta x}$$

- Reminder: Jacobian

$$\mathbf{J}_f(x) \quad = \quad \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \cdots & \frac{\partial f_2(x)}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_m(x)}{\partial x_1} & \frac{\partial f_m(x)}{\partial x_2} & \cdots & \frac{\partial f_m(x)}{\partial x_n} \end{pmatrix}$$

Squared Error

- With the previous linearization, we can fix $\mathbf{x}$ and carry out the minimization in the increments $\boldsymbol{\Delta}\mathbf{x}$
- We replace the Taylor expansion in the squared error terms:

$$e_i(\hat{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}) \;=\; \ldots$$

Squared Error

- With the previous linearization, we can fix $\mathbf{x}$ and carry out the minimization in the increments $\mathbf{\Delta x}$
- We replace the Taylor expansion in the squared error terms:

$$
\begin{aligned}
e_i(\hat{\mathbf{x}} + \mathbf{\Delta x}) &= \mathbf{e}_i^T(\hat{\mathbf{x}} + \mathbf{\Delta x})\mathbf{\Omega}_i \mathbf{e}_i(\hat{\mathbf{x}} + \mathbf{\Delta x}) \\
&\simeq (\mathbf{e}_i + \mathbf{J}_i \mathbf{\Delta x})^T \mathbf{\Omega}_i (\mathbf{e}_i + \mathbf{J}_i \mathbf{\Delta x}) \\
&= \mathbf{e}_i^T \mathbf{\Omega}_i \mathbf{e}_i + \\
&\quad\ \mathbf{e}_i^T \mathbf{\Omega}_i \mathbf{J}_i \mathbf{\Delta x} + \mathbf{\Delta x}^T \mathbf{J}_i^T \mathbf{\Omega}_i \mathbf{e}_i + \\
&\quad\ \mathbf{\Delta x}^T \mathbf{J}_i^T \mathbf{\Omega}_i \mathbf{J}_i \mathbf{\Delta x}
\end{aligned}
$$

Squared Error (cont.)

- All summands are scalar so the transposition has no effect
- By grouping similar terms, we obtain:

$$
\begin{aligned}
& e_i(\hat{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}) \\
& \quad \simeq \quad \mathbf{e}_i^T \boldsymbol{\Omega}_i \mathbf{e}_i + \\
& \qquad\quad \mathbf{e}_i^T \boldsymbol{\Omega}_i \mathbf{J}_i \boldsymbol{\Delta}\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}^T \mathbf{J}_i^T \boldsymbol{\Omega}_i \mathbf{e}_i + \\
& \qquad\quad \boldsymbol{\Delta}\mathbf{x}^T \mathbf{J}_i^T \boldsymbol{\Omega}_i \mathbf{J}_i \boldsymbol{\Delta}\mathbf{x} \\
& \quad = \quad \underbrace{\mathbf{e}_i^T \boldsymbol{\Omega}_i \mathbf{e}_i}_{c_i} + 2 \underbrace{\mathbf{e}_i^T \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{b}_i^T} \boldsymbol{\Delta}\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}^T \underbrace{\mathbf{J}_i^T \boldsymbol{\Omega}_i \mathbf{J}_i}_{\mathbf{H}_i} \boldsymbol{\Delta}\mathbf{x} \\
& \quad = \quad c_i + 2\mathbf{b}_i^T \boldsymbol{\Delta}\mathbf{x} + \boldsymbol{\Delta}\mathbf{x}^T \mathbf{H}_i \boldsymbol{\Delta}\mathbf{x}
\end{aligned}
$$

Global Error

- The global error is the sum of the squared errors terms corresponding to the individual measurements
- Form a new expression which approximates the global error in the neighborhood of the current solution $\mathbf{x}$

$$
\begin{aligned}
F(\hat{\mathbf{x}}+\mathbf{\Delta x}) &\simeq \sum_i \Big(c_i + \mathbf{b}_i^T \mathbf{\Delta x} + \mathbf{\Delta x}^T \mathbf{H}_i \mathbf{\Delta x}\Big) \\
&= \sum_i c_i + 2(\sum_i \mathbf{b}_i^T)\mathbf{\Delta x} + \mathbf{\Delta x}^T (\sum_i \mathbf{H}_i)\mathbf{\Delta x}
\end{aligned}
$$

Global Error (cont.)

$$
\begin{aligned}
F(\hat{\mathbf{x}}+\Delta\mathbf{x}) &\simeq \sum_i \left(c_i + \mathbf{b}_i^T \Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{H}_i \Delta\mathbf{x}\right) \\
&= \underbrace{\sum_i c_i}_{c} + 2\,\underbrace{\left(\sum_i \mathbf{b}_i^T\right)}_{\mathbf{b}^T} \Delta\mathbf{x} + \Delta\mathbf{x}^T \underbrace{\left(\sum_i \mathbf{H}_i\right)}_{\mathbf{H}} \Delta\mathbf{x} \\
&= c + 2\mathbf{b}^T \Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}
\end{aligned}
$$

with

$$
\begin{aligned}
\mathbf{b}^T &= \sum_i \mathbf{e}_i^T \boldsymbol{\Omega}_i \mathbf{J}_i \\
\mathbf{H} &= \sum_i \mathbf{J}_i^T \boldsymbol{\Omega} \mathbf{J}_i
\end{aligned}
$$

Quadratic Form

- We can write the global error terms as a quadratic form in $\Delta\mathbf{x}$

$$F(\hat{\mathbf{x}} + \Delta\mathbf{x}) \;\simeq\; c + 2\mathbf{b}^T\Delta\mathbf{x} + \Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x}$$

- We need to compute the derivative of $F(\hat{\mathbf{x}} + \Delta\mathbf{x})$ w.r.t. $\Delta\mathbf{x}$ (given $\hat{\mathbf{x}}$)

Deriving a Quadratic Form

- Assume a quadratic form

$$f(\mathbf{x}) \;=\; \mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{b}^T\mathbf{x}$$

- The first derivative is

$$\frac{\partial f}{\partial \mathbf{x}} \;=\; (\mathbf{H} + \mathbf{H}^T)\mathbf{x} + \mathbf{b}$$

See: The Matrix Cookbook, Section 2.2.4

Quadratic Form

- We can write the global error terms as a quadratic form in $\Delta \mathbf{x}$

$$F(\hat{\mathbf{x}} + \Delta \mathbf{x}) \;\simeq\; c + 2\mathbf{b}^T \Delta \mathbf{x} + \Delta \mathbf{x}^T \mathbf{H} \Delta \mathbf{x}$$

- The derivative of the approximated $F(\hat{\mathbf{x}} + \Delta \mathbf{x})$ w.r.t. $\Delta \mathbf{x}$ is then:

$$\frac{\partial F(\hat{\mathbf{x}} + \Delta \mathbf{x})}{\partial \Delta \mathbf{x}} \;\simeq\; 2\mathbf{b} + 2\mathbf{H} \Delta \mathbf{x}$$

Minimizing the Quadratic Form

- Derivative of $F(\hat{\mathbf{x}} + \mathbf{\Delta x})$

$$\frac{\partial F(\hat{\mathbf{x}} + \mathbf{\Delta x})}{\partial \mathbf{\Delta x}} \simeq 2\mathbf{b} + 2\mathbf{H}\mathbf{\Delta x}$$

- Setting it to zero leads to

$$0 = 2\mathbf{b} + 2\mathbf{H}\mathbf{\Delta x}$$

- Which leads to the linear system

$$\mathbf{H}\mathbf{\Delta x} = -\mathbf{b}$$

- The solution for the increment $\mathbf{\Delta x}^*$ is

$$\mathbf{\Delta x}^* = -\mathbf{H}^{-1}\mathbf{b}$$

Gauss-Newton Solution

Iterate the following steps:

- Linearize around $\mathbf{x}$ and compute for each measurement

$$\mathrm{e}_i(\hat{\mathrm{x}} + \mathbf{\Delta}\mathrm{x}) \simeq \mathrm{e}_i(\hat{\mathrm{x}}) + \mathbf{J}_i \mathbf{\Delta}\mathrm{x}$$

- Compute the terms for the linear system $\mathrm{b}^T = \sum_i \mathrm{e}_i^T \mathbf{\Omega}_i \mathbf{J}_i \qquad \mathbf{H} = \sum_i \mathbf{J}_i^T \mathbf{\Omega}_i \mathbf{J}_i$

- Solve the linear system

$$\mathbf{\Delta}\mathrm{x}^* = -\mathbf{H}^{-1}\mathrm{b}$$

- Updating state $\hat{\mathrm{x}} \leftarrow \hat{\mathrm{x}} + \mathbf{\Delta}\mathrm{x}^*$

Example: Odometry Calibration

- Odometry measurements $\mathbf{u}_i$
- Eliminate systematic error through calibration
- Assumption: Ground truth odometry $\mathbf{u}_i^*$ is available
- Ground truth by motion capture, scan-matching, or a SLAM system

Example: Odometry Calibration

- There is a function $f_i(\mathbf{x})$ which, given some bias parameters $\mathbf{x}$, returns a an unbiased (corrected) odometry for the reading $\mathbf{u}'_i$ as follows

$$\mathbf{u}'_i = f_i(\mathbf{x}) = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} \mathbf{u}_i$$

- To obtain the correction function $f(\mathbf{x})$, we need to find the parameters $\mathbf{x}$

Odometry Calibration (cont.)

- The state vector is

$$\mathbf{x} = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{21} & x_{22} & x_{23} & x_{31} & x_{32} & x_{33} \end{pmatrix}^T$$

- The error function is

$$\mathbf{e}_i(\mathbf{x}) = \mathbf{u}_i^* - \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} \mathbf{u}_i$$

- Its derivative is:

$$\mathbf{J}_i = \frac{\partial \mathbf{e}_i(\mathbf{x})}{\partial \mathbf{x}} = - \begin{pmatrix} u_{i,x} & u_{i,y} & u_{i,\theta} & & & & & & \\ & & & u_{i,x} & u_{i,y} & u_{i,\theta} & & & \\ & & & & & & u_{i,x} & u_{i,y} & u_{i,\theta} \end{pmatrix}$$

Does not depend on ***x***, why? What are the consequences?

e is linear, no need to iterate!

Questions

- How do the parameters look like if the odometry is perfect?
- How many measurements (at least) are needed to find a solution for the calibration problem?
- $\mathbf{H}$ is symmetric. Why?
- How does the structure of the measurement function affects the structure of $\mathbf{H}$?

How to Efficiently Solve the Linear System?

- Linear system $\mathbf{H}\Delta\mathbf{x} = -\mathbf{b}$
- Can be solved by matrix inversion (in theory)
- In practice:
  - Cholesky factorization
  - QR decomposition
  - Iterative methods such as conjugate gradients (for large systems)

Cholesky Decomposition for Solving a Linear System

- $\mathbf{A}$ symmetric and positive definite
- System to solve $\mathbf{A}\mathbf{x} = \mathbf{b}$
- Cholesky leads to $\mathbf{A} = \mathbf{L}\mathbf{L}^T$ with $\mathbf{L}$ being a lower triangular matrix
- Solve first

$$\mathbf{L}\mathbf{y} = \mathbf{b}$$

- an then

$$\mathbf{L}^T\mathbf{x} = \mathbf{y}$$

Gauss-Newton Summary

Method to minimize a squared error:

- Start with an initial guess
- Linearize the individual error functions
- This leads to a quadratic form
- One obtains a linear system by settings its derivative to zero
- Solving the linear systems leads to a state update
- Iterate

Relation to Probabilistic State Estimation

- So far, we minimized an error function
- How does this relate to state estimation in the probabilistic sense?

General State Estimation

- Bayes rule, independence and Markov assumptions allow us to write

$$
\begin{aligned}
& p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\
& \quad = \ \eta \, p(x_0) \prod_t \left[p(x_t \mid x_{t-1}, u_t) \, p(z_t \mid x_t) \right]
\end{aligned}
$$

Log Likelihood

- Written as the log likelihood, leads to

$$
\begin{aligned}
&\log p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\
&\quad = \quad \text{const.} + \log p(x_0) \\
&\qquad\quad + \sum_t \left[\log p(x_t \mid x_{t-1}, u_t) + \log p(z_t \mid x_t)\right]
\end{aligned}
$$

Gaussian Assumption

- Assuming Gaussian distributions

$$
\begin{aligned}
\log p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\
= \quad & \text{const.} + \log \underbrace{p(x_0)}_{\mathcal{N}} \\
& + \sum_t \left[\log \underbrace{p(x_t \mid x_{t-1}, u_t)}_{\mathcal{N}} + \log \underbrace{p(z_t \mid x_t)}_{\mathcal{N}} \right]
\end{aligned}
$$

Log of a Gaussian

- Log likelihood of a Gaussian

$$\begin{aligned}\log \mathcal{N}(x, \mu, \Sigma) \\ = \quad \text{const.} - \frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\end{aligned}$$

Error Function as Exponent

- Log likelihood of a Gaussian

$$\begin{aligned}\log \mathcal{N}(x, \mu, \Sigma) \\ = \quad & \text{const.} - \frac{1}{2} \underbrace{\underbrace{(x-\mu)^T}_{\mathbf{e}^T(x)} \underbrace{\Sigma^{-1}}_{\Omega} \underbrace{(x-\mu)}_{\mathbf{e}(x)}}_{e(x)}\end{aligned}$$

- is up to a constant equivalent to the error functions used before

Log Likelihood with Error Terms

- Assuming Gaussian distributions

$$\log p(x_{0:t} \mid z_{1:t}, u_{1:t})$$
$$= \quad \text{const.} - \frac{1}{2} e_p(x) - \frac{1}{2} \sum_t \left[e_{u_t}(x) + e_{z_t}(x) \right]$$

Maximizing the Log Likelihood

- Assuming Gaussian distributions

$$\begin{aligned}\log p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\ = \quad \text{const.} - \frac{1}{2} e_p(x) - \frac{1}{2} \sum_t \left[e_{u_t}(x) + e_{z_t}(x) \right]\end{aligned}$$

- Maximizing the log likelihood leads to

$$\begin{aligned}\operatorname{argmax} \log p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\ = \quad \operatorname{argmin} e_p(x) + \sum_t \left[e_{u_t}(x) + e_{z_t}(x) \right]\end{aligned}$$

Minimizing the Squared Error is Equivalent to Maximizing the Log Likelihood of Independent Gaussian Distributions

with individual error terms for the motions, measurements, and prior:

$$\begin{aligned} & \operatorname{argmax} \log p(x_{0:t} \mid z_{1:t}, u_{1:t}) \\ = \quad & \operatorname{argmin} e_p(x) + \sum_t \left[e_{u_t}(x) + e_{z_t}(x) \right] \end{aligned}$$

Summary

- Technique to minimize squared error functions
- Gauss-Newton is an iterative approach for non-linear problems
- Uses linearization (approximation!)
- Equivalent to maximizing the log likelihood of independent Gaussians
- Popular method in a lot of disciplines

Literature

Least Squares and Gauss-Newton

- Basically every textbook on numeric calculus or optimization
- Wikipedia (for a brief summary)

Relation to Probability Theory

- Thrun et al.: “Probabilistic Robotics”, Chapter 11.4

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