

Sensor Data Fusion

Lecture 1

Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab



Room: 3.115, Institute of Computer Science
Tel: +49 551 39 172024
E-Mail: marcus.baum@cs.uni-goettingen.de

Since 2019 **Professor of Computer Science**
2015 - 2019 **Juniorprofessor**



2017 - 2019 **Visiting Professor, Chair of Sensor Technology**



2013/14 **Postdoc / Assistant Research Prof.**
Peter Willett, Yaakov Bar-Shalom



2013 **Dr.-Ing., Institute of Anthropomatics and Robotics**
2007 **Dipl.-Inform.**





Room: 3.101, Institute of Computer Science
Tel: +49 551 39 26039
E-Mail: kolja.thormann@cs.uni-goettingen.de

Since 2019	Research Assistant Institute of Computer Science	 GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN
2017 - 2019	Research Assistant Chair of Sensor Technology	 UNIVERSITÄT PASSAU
2017	MSc Applied Computer Science Institute of Computer Science	 GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN
2016	BSc Applied Computer Science Institute of Computer Science	 GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN



Room:
Tel:
E-Mail:

3.101, Institute of Computer Science
+49 551 39 172034
simon.steuernagel@cs.uni-goettingen.de

Since 2022

Research Assistant
Institute of Computer Science



2021

MSc Applied Computer Science
Institute of Computer Science



2019

BSc Applied Computer Science
Institute of Computer Science



Head

- Marcus Baum

Assistance and Administration

- Tina Bockler
- Gunnar Krull

Post-Docs

- Kolja Thormann

PhD Candidates

- Fabian Sigges
- Simon Steuernagel
- Laura Wolf
- Jaya Shradha Fowdur
DLR, Institute of Communications and Navigation
- Nils Kornfeld
DLR, Institute of Transportation Systems
- Claudia Malzer
Max-Planck-Institute for Dynamics and Self-Organization

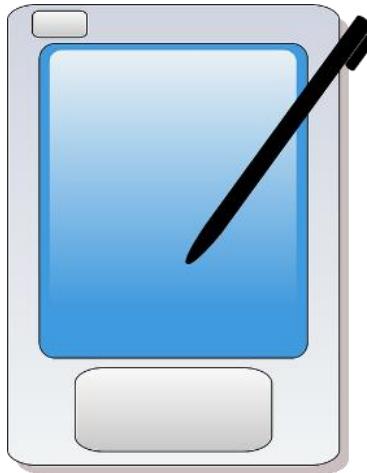
Research area:

Perception and navigation of autonomous systems

- Object detection and tracking
- Localization
- Multi-sensor fusion



- Lecture and exercise take place via face-to-face meetings
- Online material is available at Stud.IP!
- **Password:** SF_ugOe_SoSe23



zum Erwartungswert $\hat{z} = \begin{bmatrix} z \\ 1 \end{bmatrix}$

mit Kovarianz $C_z = \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}$

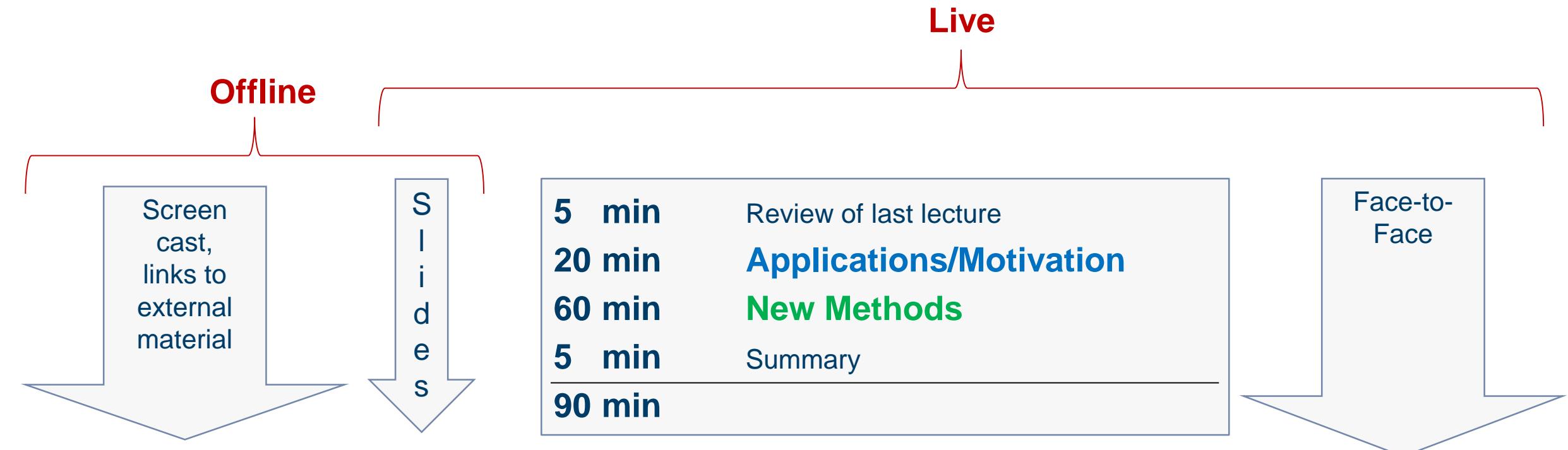
für gegebene Punkte y^* gilt,
 $f(x|y^*) = \sqrt{(x-x^*)(C^{-1})}$

wobei $x^* = \hat{x} + C_{xy} C_{yy}^{-1} (y^* - \hat{y})$

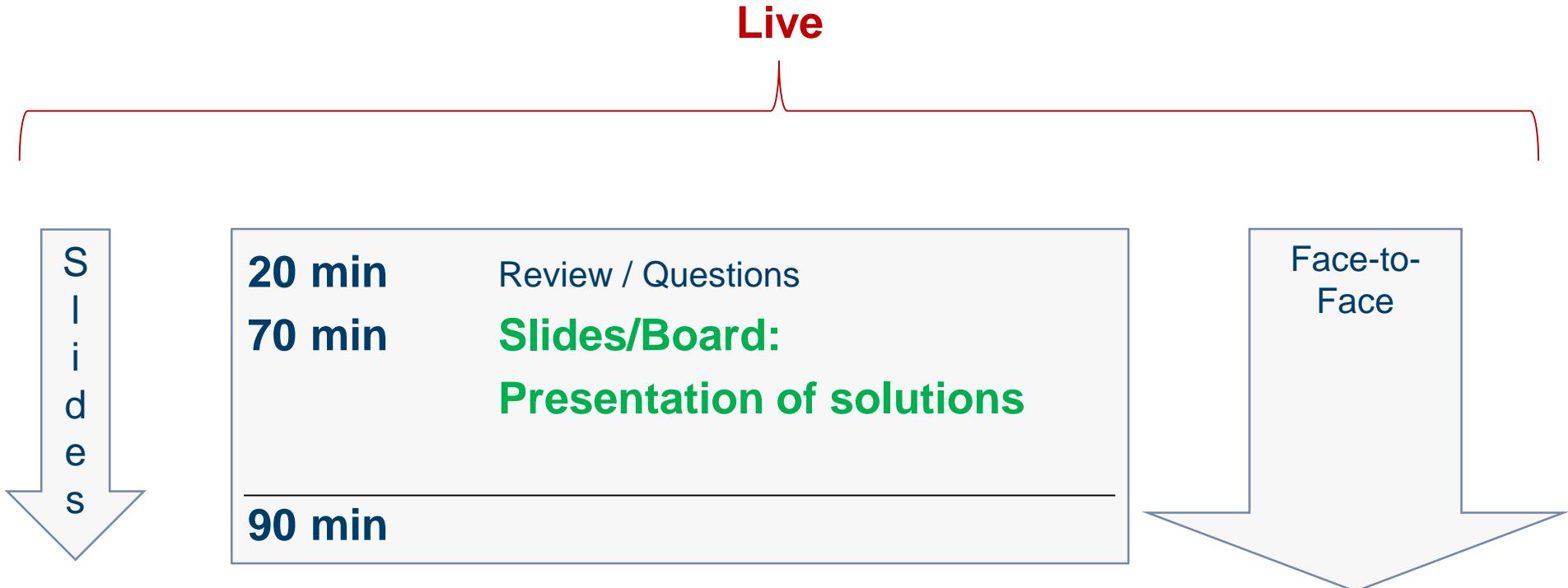
$$C^* = C_{xx} - C_{xy} C_{yy}^{-1} C_{yx}$$

- **Prerequisites:**
Bachelor in Computer Science, Mathematics, Physics, or similar
- **Workload:** 5 ECTS (4 SWS)
 - 2 SWS lecture (Tuesday)
 - 2 SWS exercise (Thursday)
- **Oral Exams**





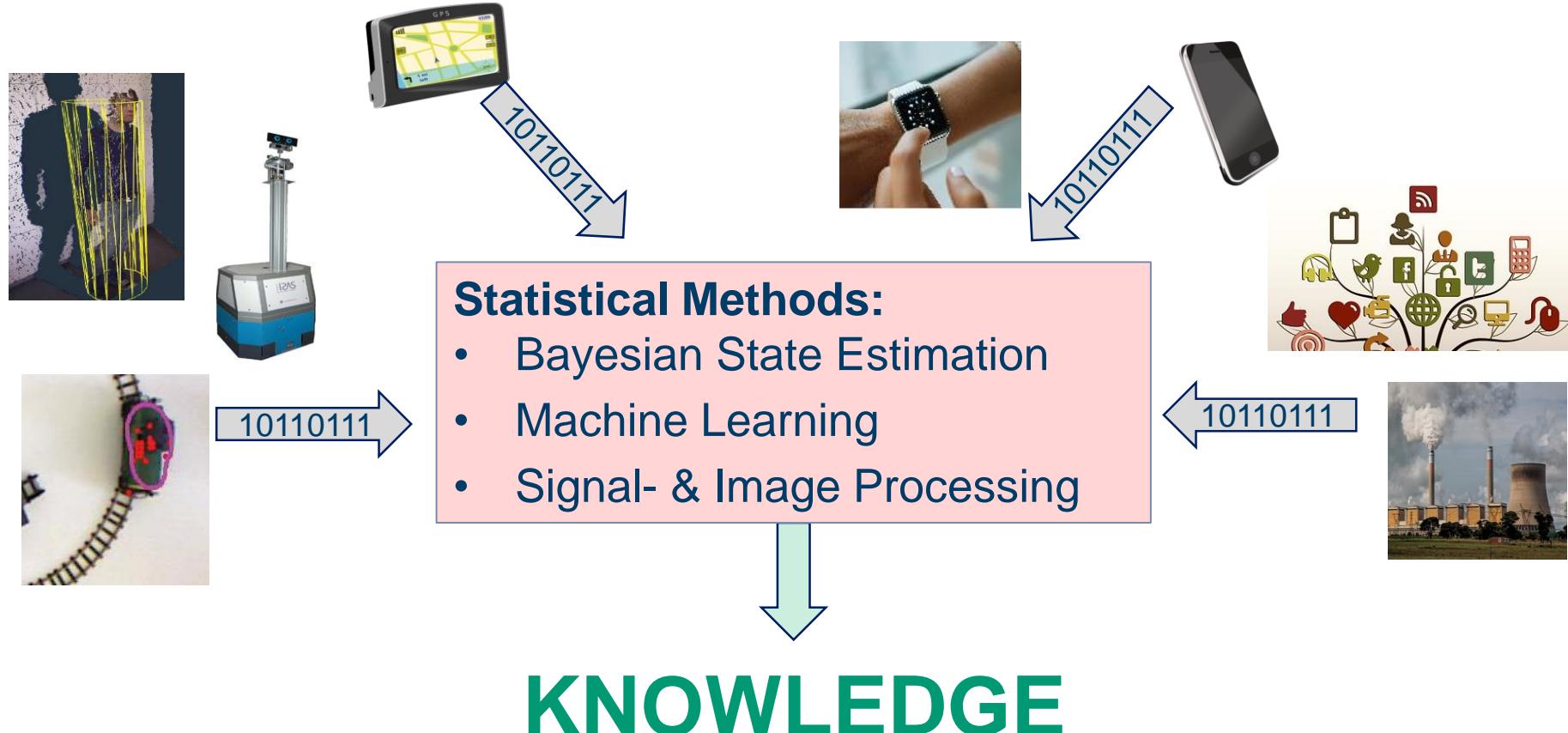
- Questions via Stud.IP, e-mail, or live meetings!
- Slides, screencasts and notes available online for download



- Problem sheet available in advance
- Solutions/notes available after the exercise

First exercise: Thursday, April 20

How to **combine** noisy or incomplete **data** in order to gain **knowledge**?



Data Fusion is Ubiquitous

- Every technical device and creature performs data fusion
- Huge amount of (sensor) data is available nowadays

Application Areas

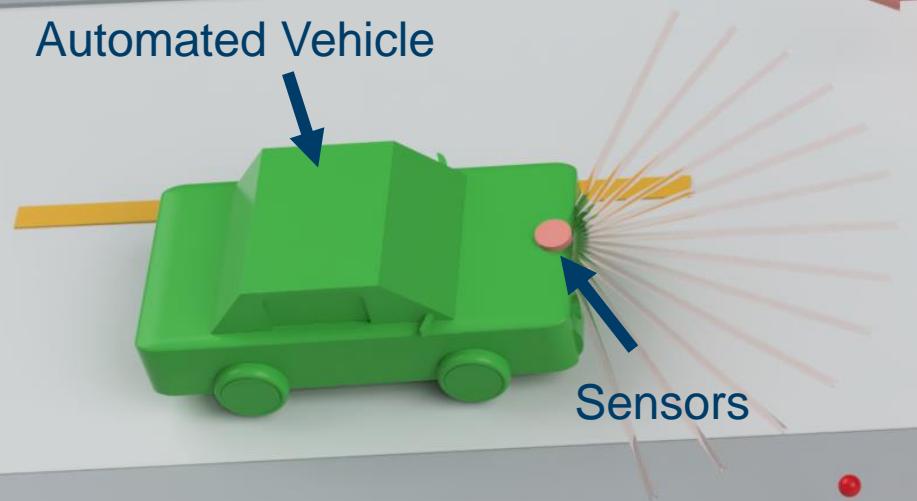
- Navigation and tracking
 - (Mobile) robotics
 - Autonomous driving
 - Cyber-physical systems
 - Internet-of-Things
 - Industry 4.0
 - Smart Cities
- ... and every other (multi-)sensor system



Mobile Robot

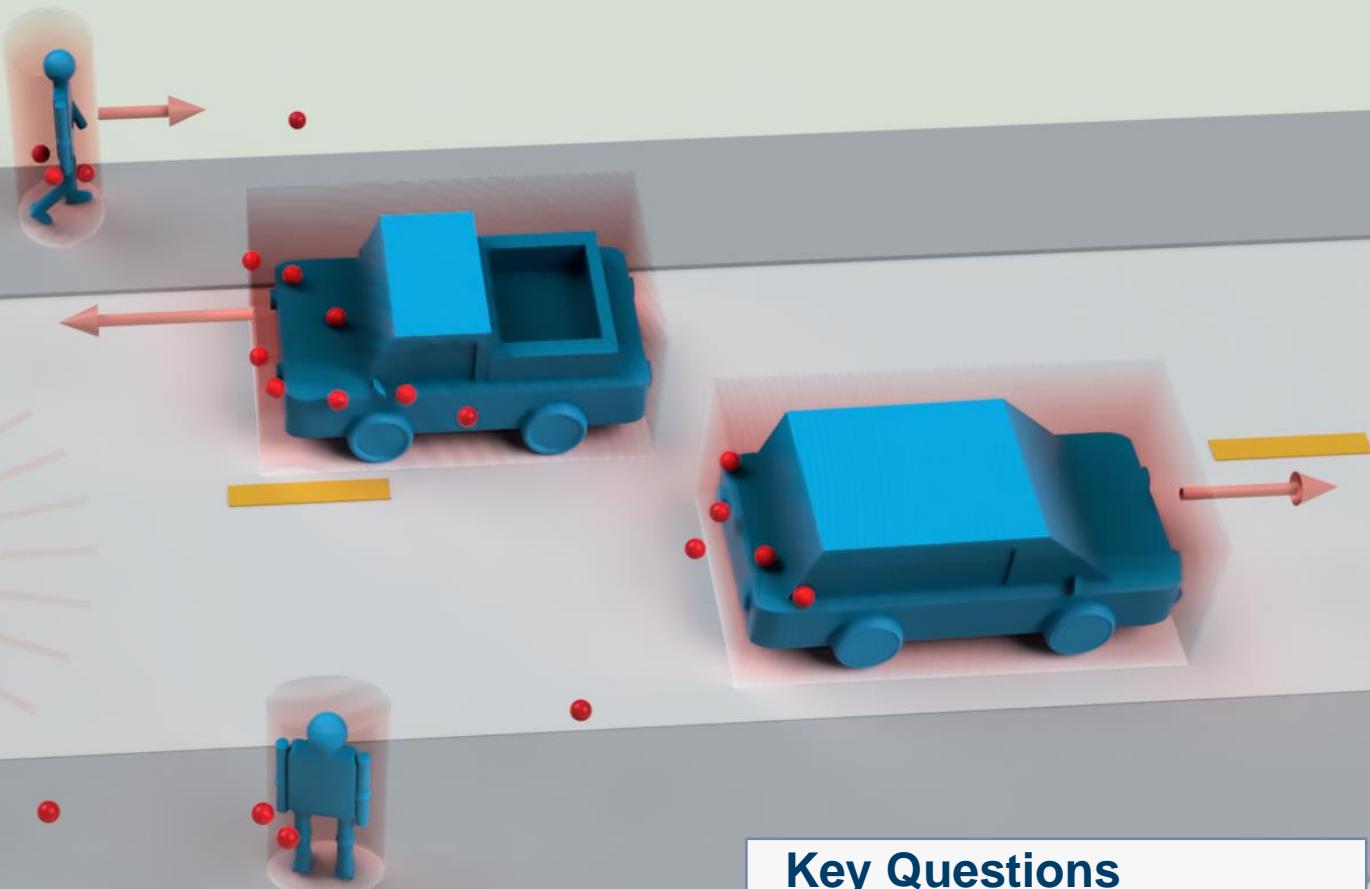
A robot that can move around in the environment

Automated Vehicle



Components:

- Sensors
- Actuators
- Processing unit
- Power system



Key Questions

- Where am I?
- Where are the others?
- Where am I going?
- How do I get there?

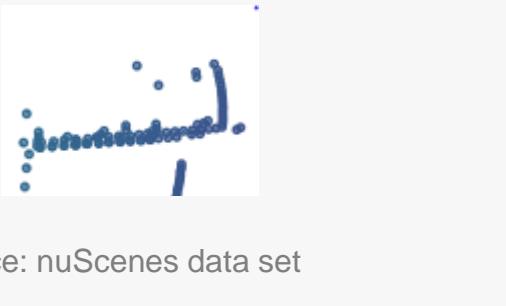
Radar

- Measures velocity
- Weather & light independent
- Low resolution
- No color information



Laser-Scanner

- Light independent
- High resolution and long range
- Weather dependent (snow/rain)
- No color information



Camera

- Color available
- High resolution and long range
- Weather and light dependent
- No velocity



Source: nuScenes data set

Every sensor technology has advantages and disadvantages

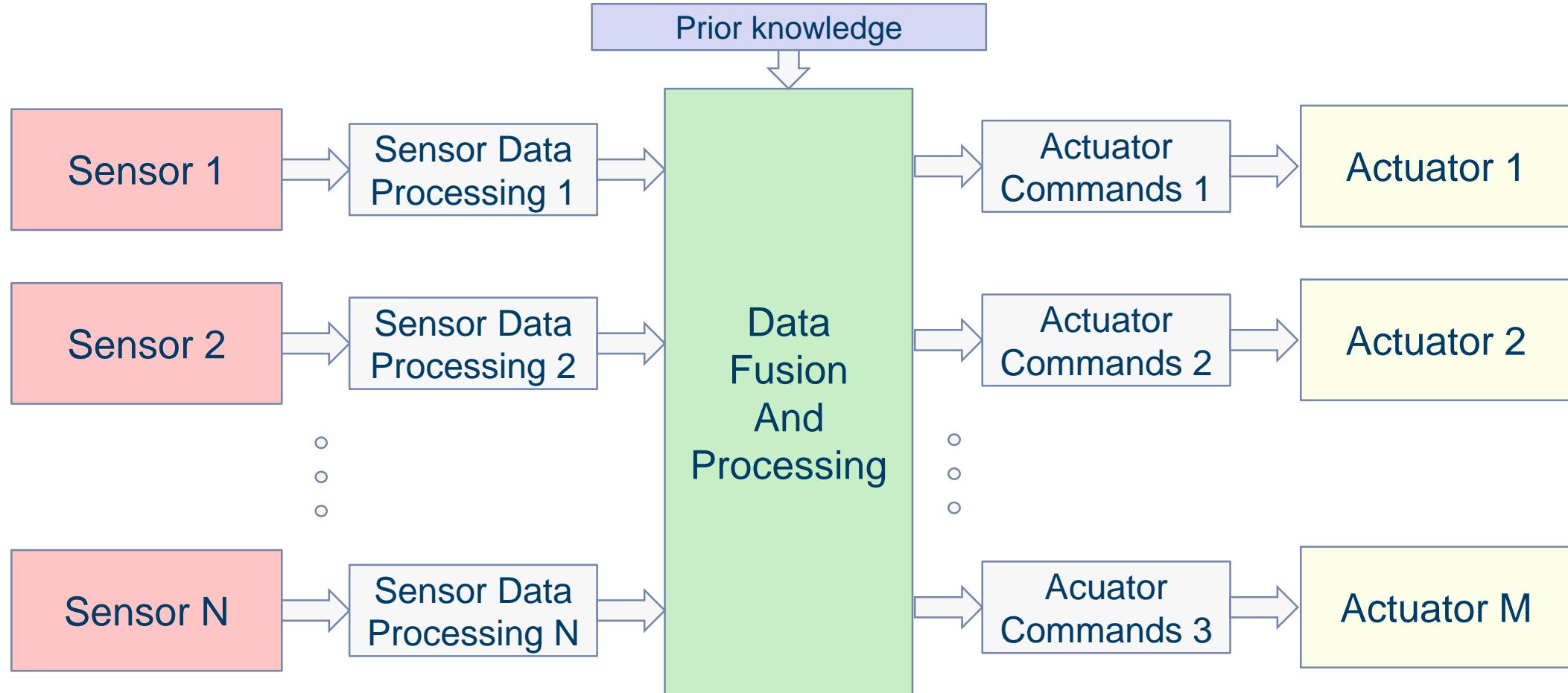


Sensor fusion:
Combine multiple sensors

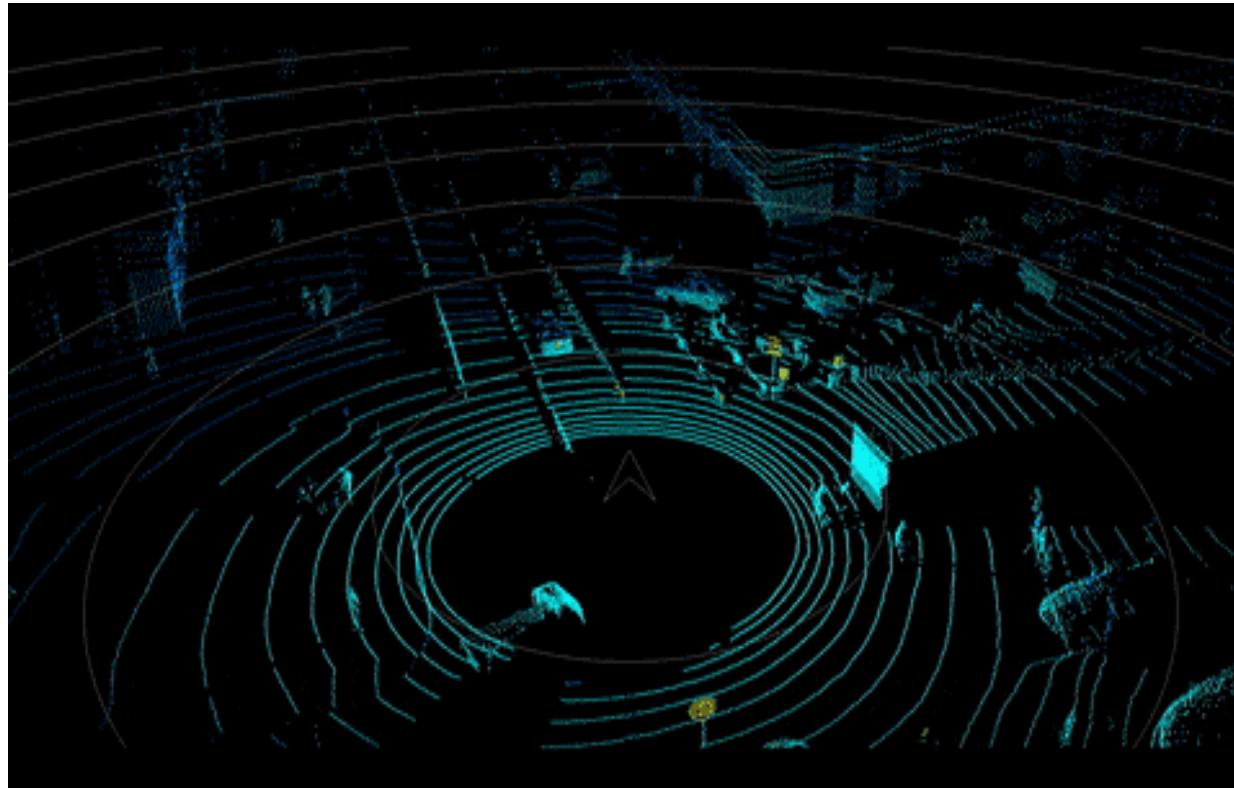


Source: <https://www.youtube.com/watch?v=YtAwOl08dQQ>

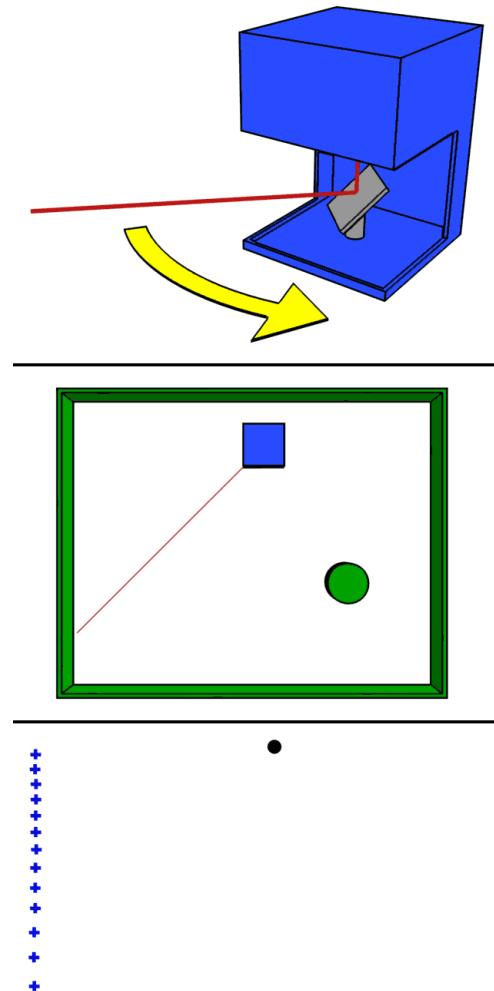
See-Think-Act Cycle: Multi-Sensor System



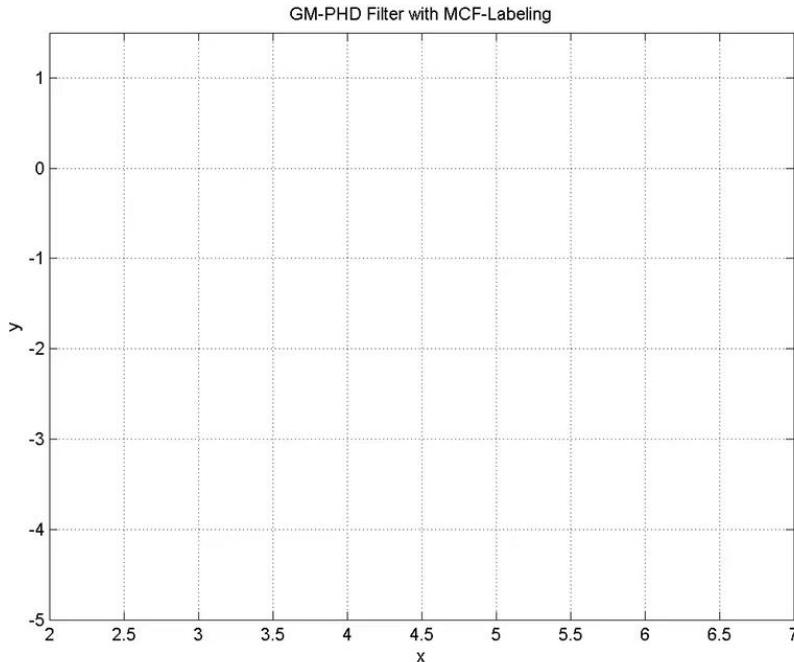
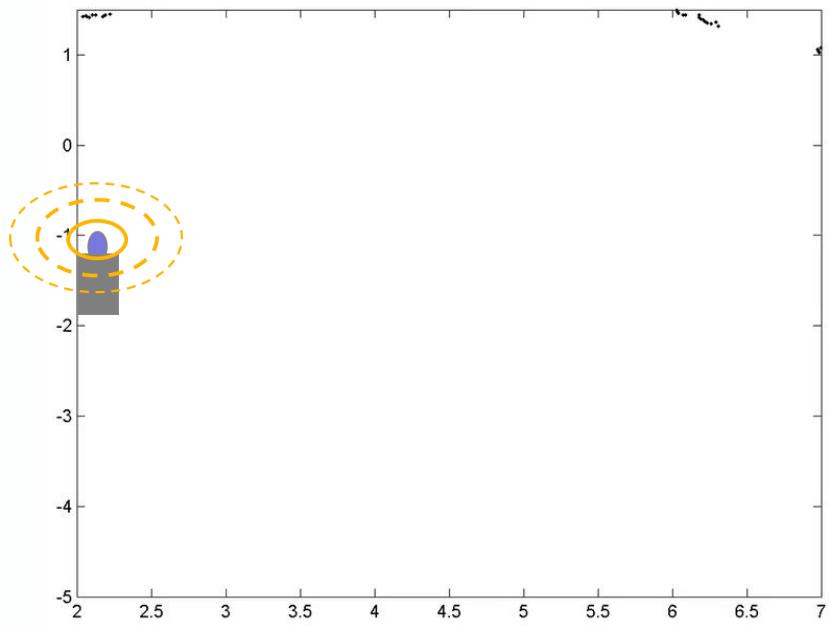
LIDAR (Light Detection And Ranging)



Source: HESAI, http://www.hesaitech.com/en/autonomous_driving.html



<https://de.wikipedia.org/wiki/Lidar>



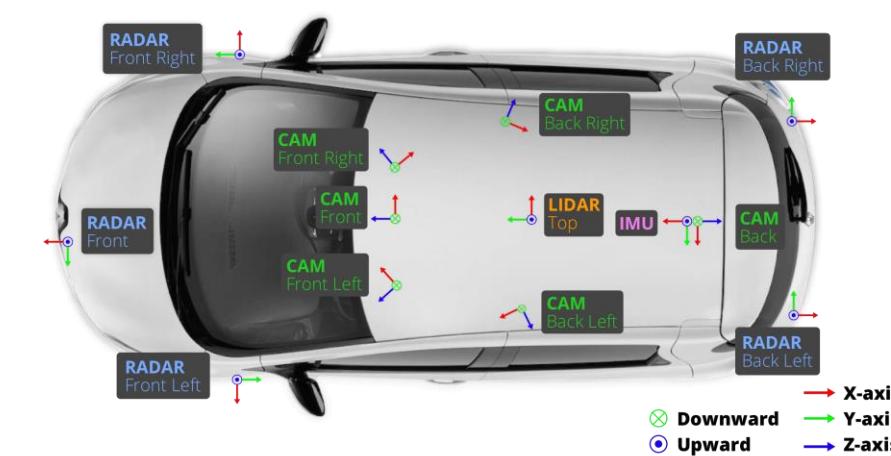
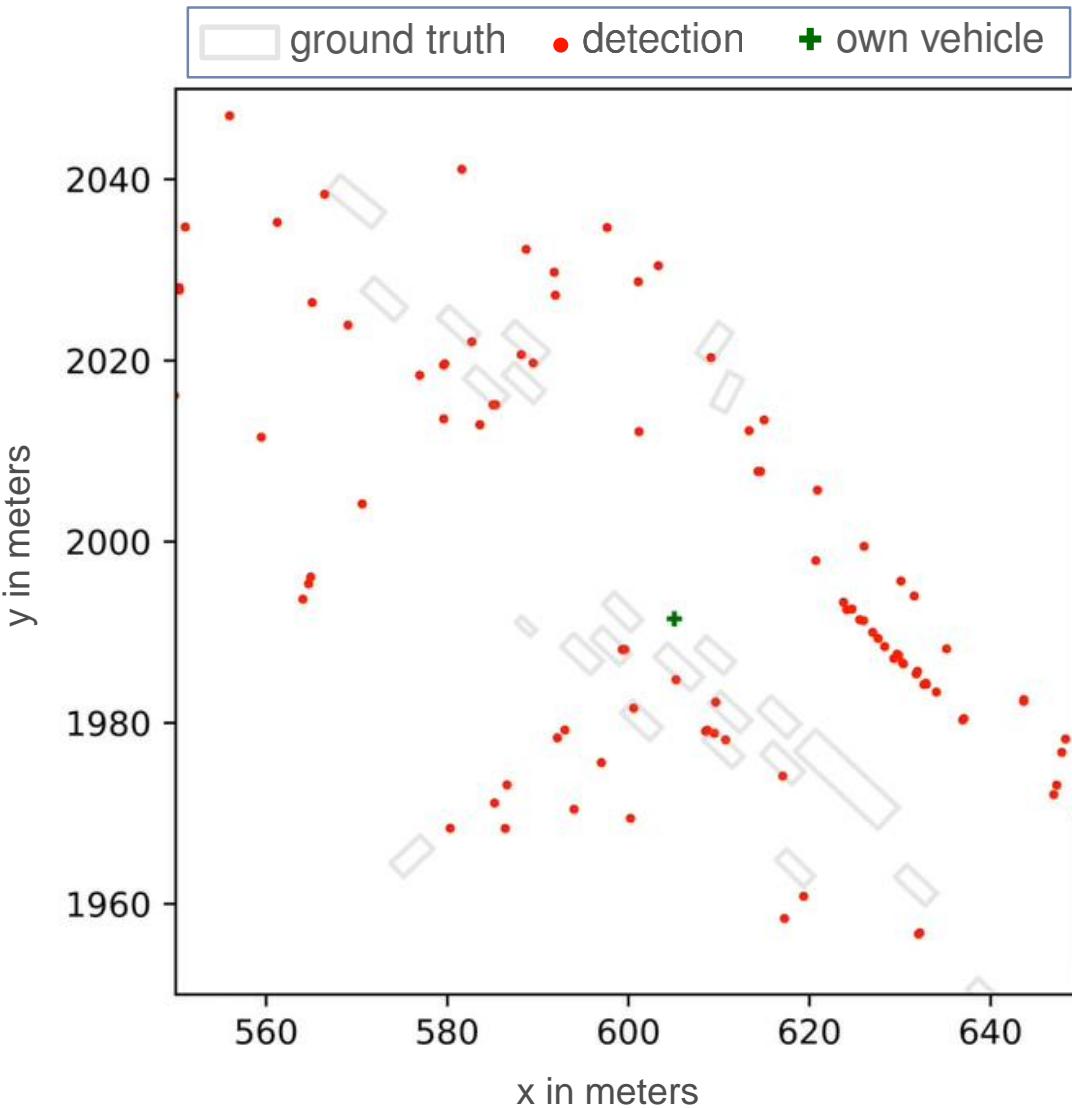
Video by Florian Teich

Radar-Based Object Tracking: NuScenes Dataset

[Caesar et al, 2019]



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

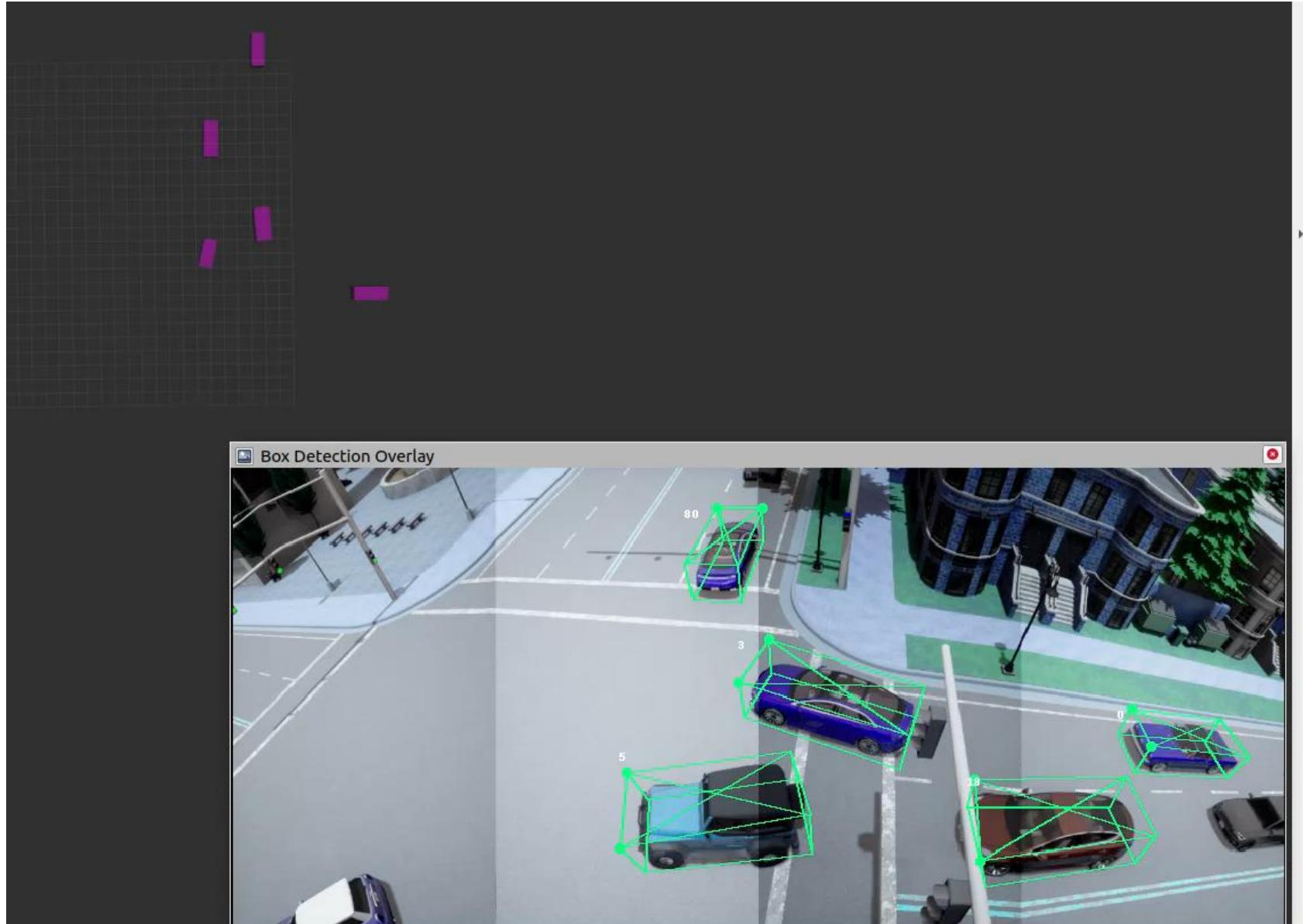


<https://www.nuscenes.org/>

- Surround view camera
- Deep learning based object detection
- Probabilistic multiple object tracker

Video by Simon Steuernagel

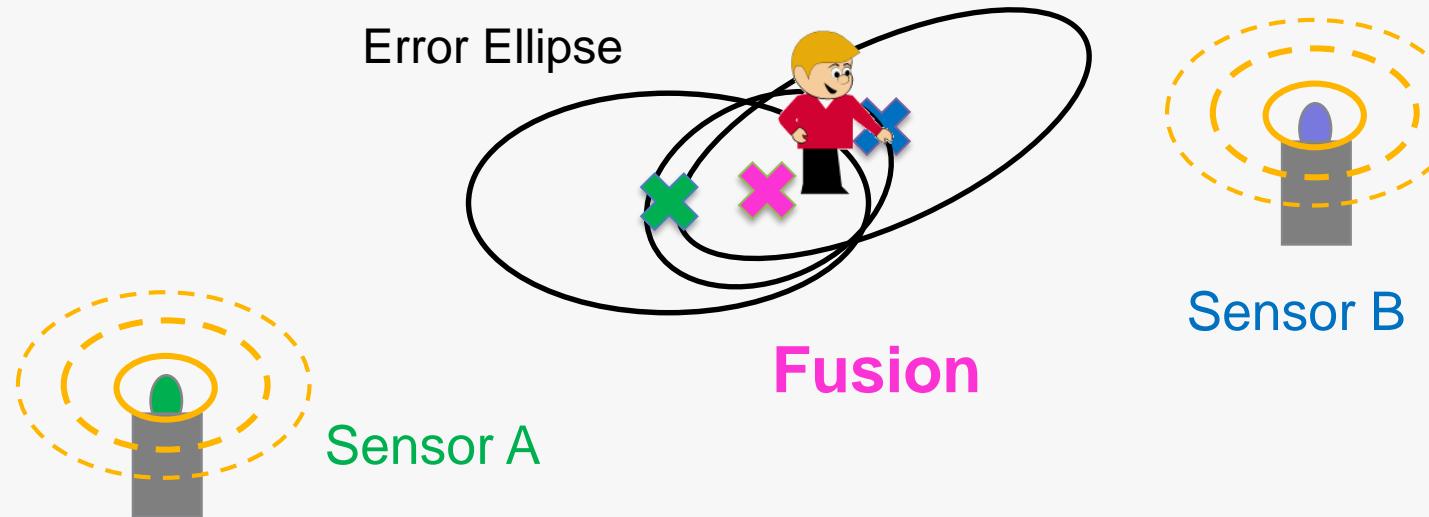
Cooperation with Andreas Leich,
DLR Institute of Transportation Systems



Definition:

Transformation of data from different sources and points in time
in order to increase knowledge about a specific phenomenon

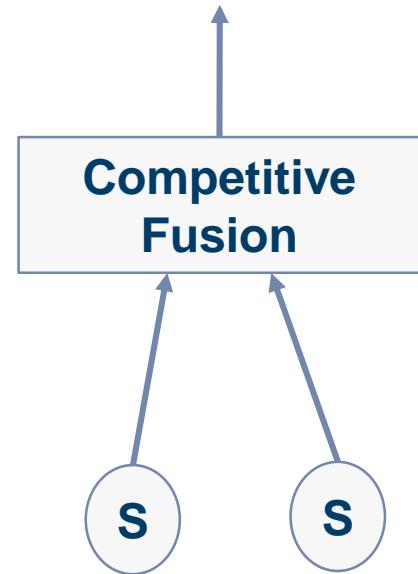
Example: Localization of Person



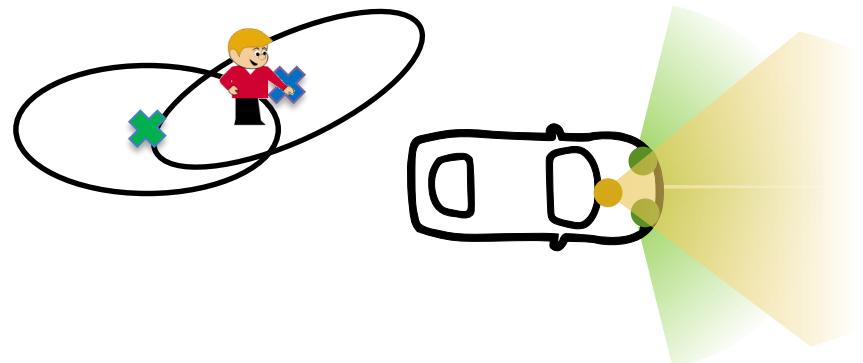
Achievement:

Reliability, Accuracy

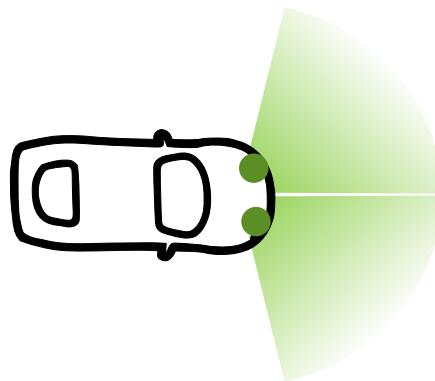
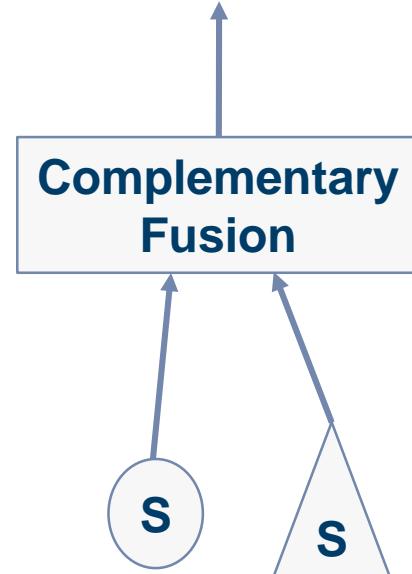
Fusion:



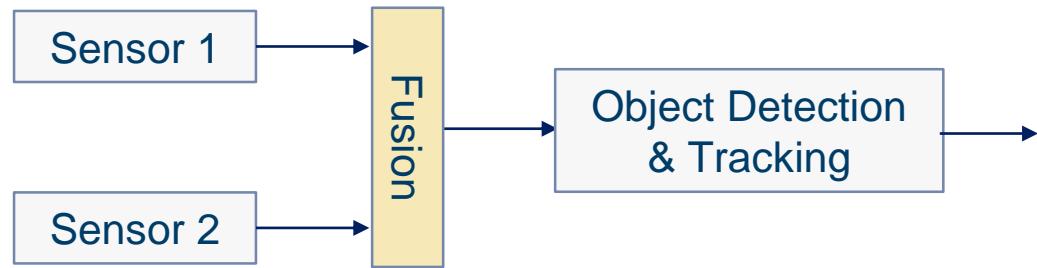
Sensors:



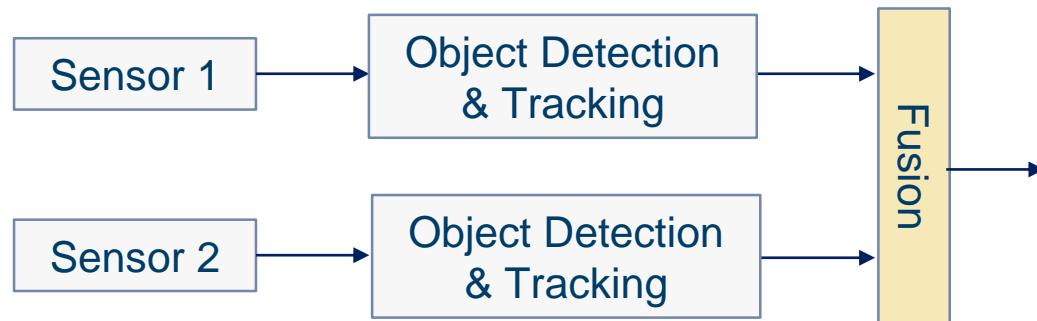
Completeness

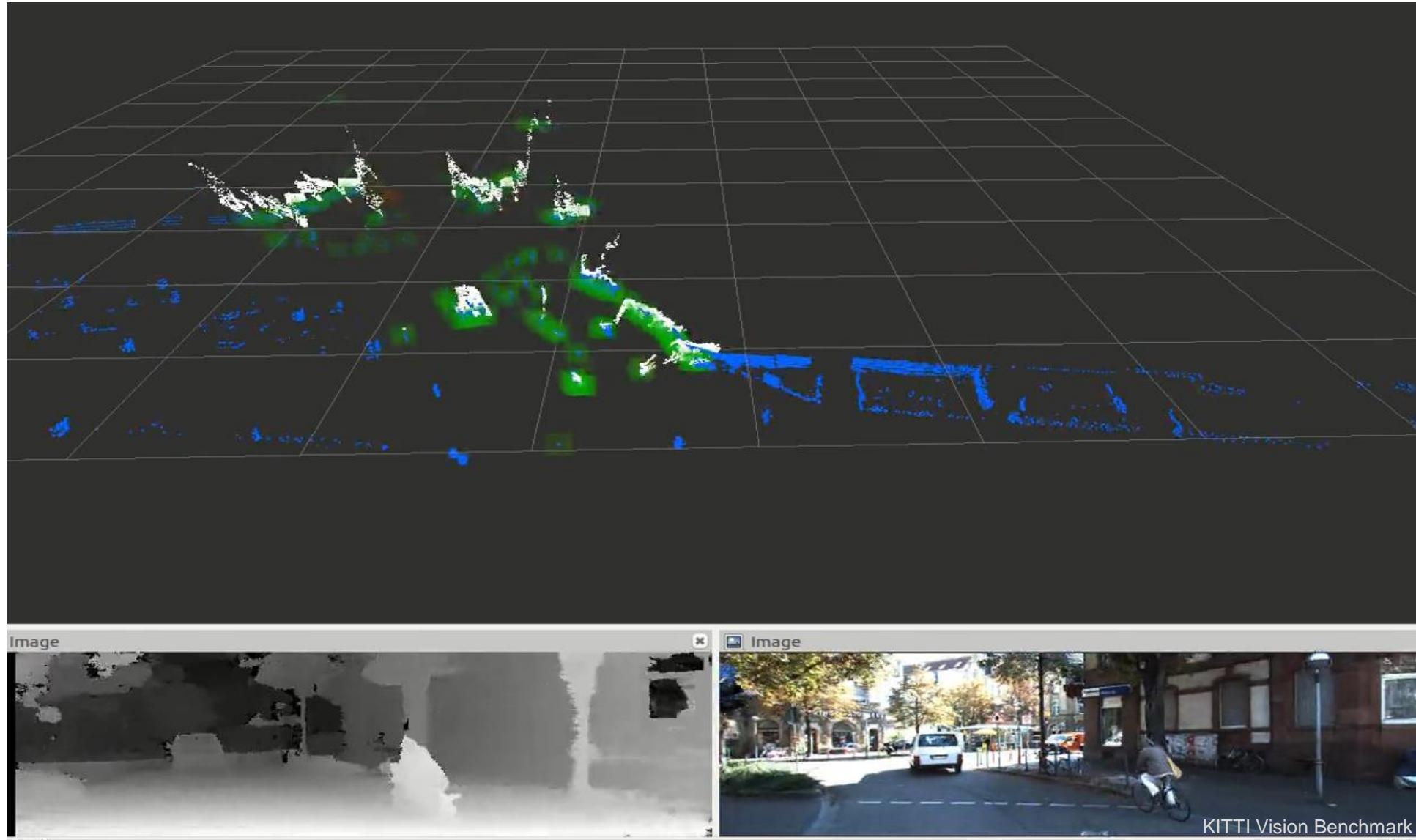


Low-Level-Fusion

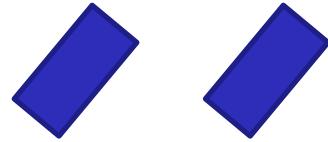


Object-Level-Fusion



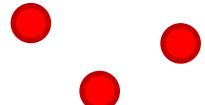


Object Level Fusion: Cooperative Perception



Vehicles

- equipped with sensors
- track objects
- send tracks to RSU



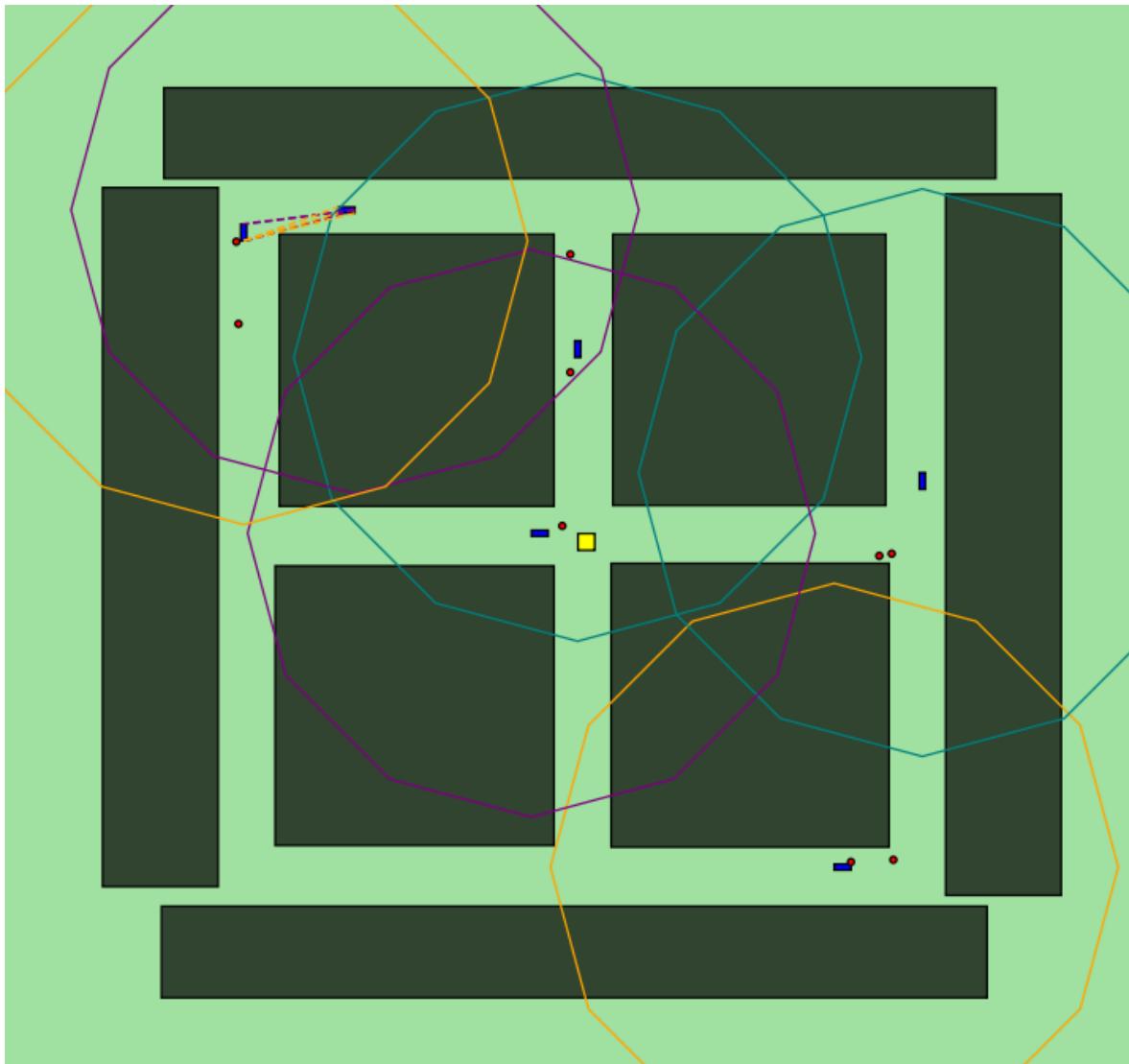
Vulnerable Road Users (VRUs)

- pedestrian etc.
- no sensors



Road Side Unit (RSU)

- association and fusion of tracks
- sends results back to vehicles



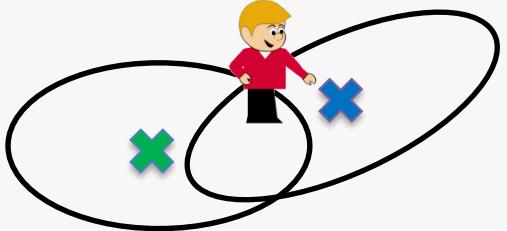
Cooperative Perception

Research question:
How to fuse sensor data by incorporating communication constraints?

Cooperation with
LU Hannover,
Prof. Markus Fidler

Foundations of Data Fusion:

- Formalization as state estimation problem
- Linear and basic nonlinear estimation methods
- Typical sensor and motion models



Data Fusion Applications:

- Traditional and modern applications
- Focus on tracking and navigation
 - Smartphone sensors
 - Kinect sensor



- The course does not cover in detail how sensors work!
- The focus lies on the processing of sensor data
- Abstract models of the sensors are sufficient

Example Model: Gyroscope

- Measures the change of orientation
- Measurement y in rad/s with standard deviation σ in rad/s



Traditional gyroscope:
Conservation of angular momentum

Course Recommendation:
Sensordatenverarbeitung (BA)

<https://en.wikipedia.org/wiki/Gyroscope>

- | | |
|-----------------------------------------------------|-------------------------------------------------------------------|
| 1. Introduction | - what is datafusion?
- how can we model it prob. |
| 2. Measurement Equation and Linear least squares | - what is cost funct. |
| 3. Nonlinear least squares: Analytic approaches | - Normal equation (plot with proj.) |
| 4. Nonlinear least squares: Optimization algorithms | - Measur. equa. for Tril. |
| 5. Stochastic least squares | - Possibilities of pos. |
| 6. Fundamental fusion formulas | - What is / how to solve NonLineaLS |
| 7. Bayesian approach | - adv. / disadv. |
| 8. Bayesian approach: Linear and Gaussian case | - Stochastic LS - Estimator, good/bad
unbiased, BLUE, inves... |
| 9. Kalman filter: Introduction | - diff. Fusion arch. / covariance inters. |
| 10. (Extended) Kalman filter | ↳ what type of error (correlations?) |
| 11. Unscented Transform ^{entfallt} | - correlation bounds (covariance inters. algo.) |
| 12. Dynamic models | - Fisher special case of Bayesian... |
| 13. Uncertainty modeling | - Random walk, nearly constant acceleration |
| 14. Summary and Discussion | - what type of uncertainties do exist? |

Book/Lecture:

Estimation with Applications to Tracking and Navigation

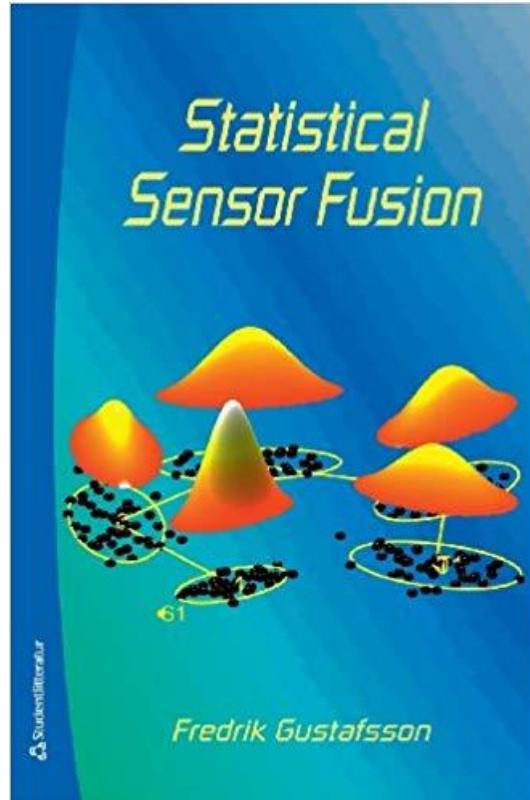
Yaakov Bar-Shalom, X. Rong Li,
T. Kirubarajan

Book/Lecture:

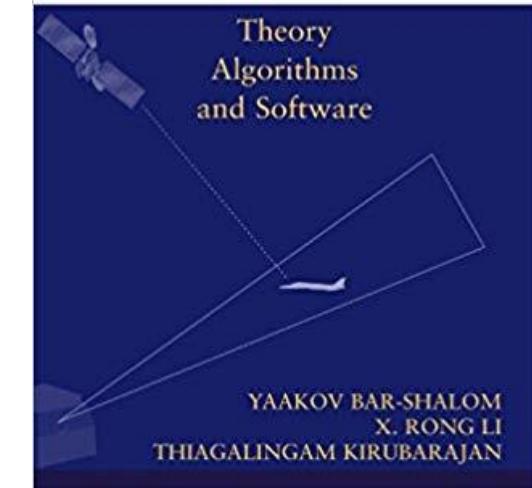
Statistical Sensor Fusion

Fredrik Gustafsson,
Linköping University

WWW: <https://sensorfusion.se/>



**Estimation with
Applications to
Tracking and
Navigation**



Further references for specific topics will be announced during the lecture...

Localization:

Estimation of the pose (position & orientation) of an object

Tracking:

Estimation of the state (e.g., position) of a moving object

Navigation:

- Estimation of one's own state
(the object on which sensors are mounted)
- Planning of the shortest path to goal (not covered in this course)

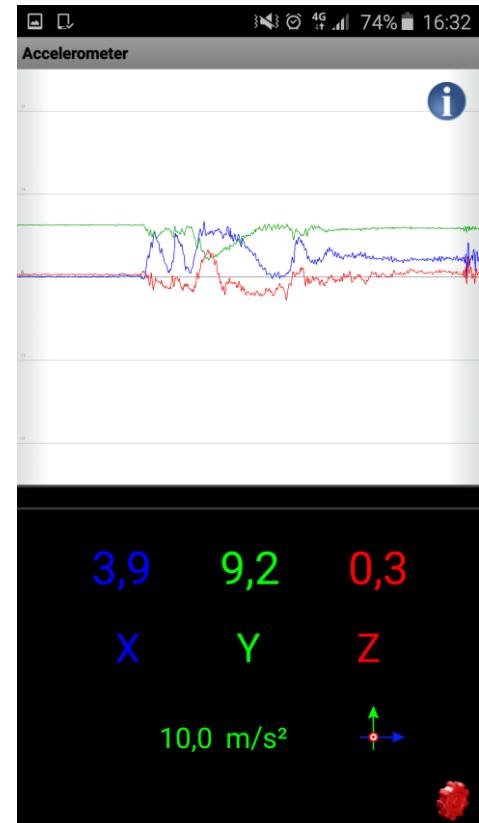
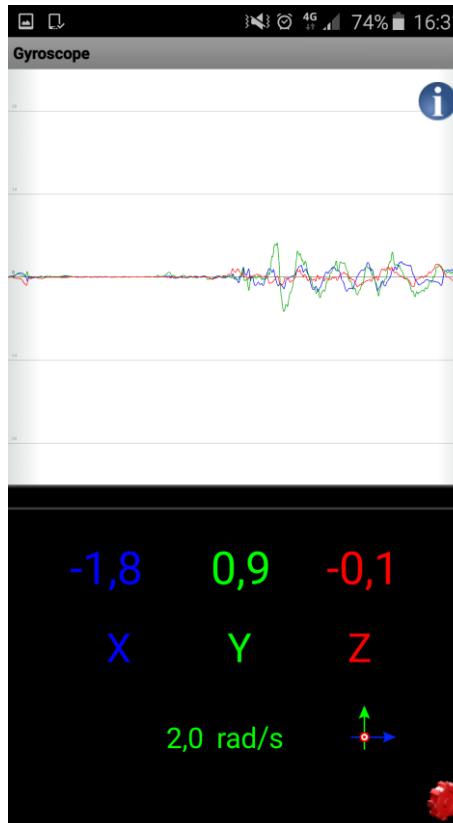
Note: There are also slightly different definitions in literature



Sensor Fusion App

WWW: <http://users.isy.liu.se/en/rt/fredrik/app/>

Google Playstore: play.google.com/store/apps/details?id=com.hiq.sensor



Data Fusion is Ubiquitous:

- Interdisciplinary
- Huge variety of interesting applications
- Will become increasingly important in the future
- All applications have the same theoretical foundations, i.e.,
(Bayesian) state estimation and the Kalman filter
that are extremely important concepts in applied computer science, engineering, etc.
- Very active research area (e.g., in the context of artificial intelligence)
- Huge need of data fusion experts in industry

Summer Term 2023

- Sensor Data Fusion (MA)
- Environment Perception for Automotive Systems (BA/MA)
- Practical Course Data Fusion (BA/MA)

First meeting:
Fr., April 21, noon

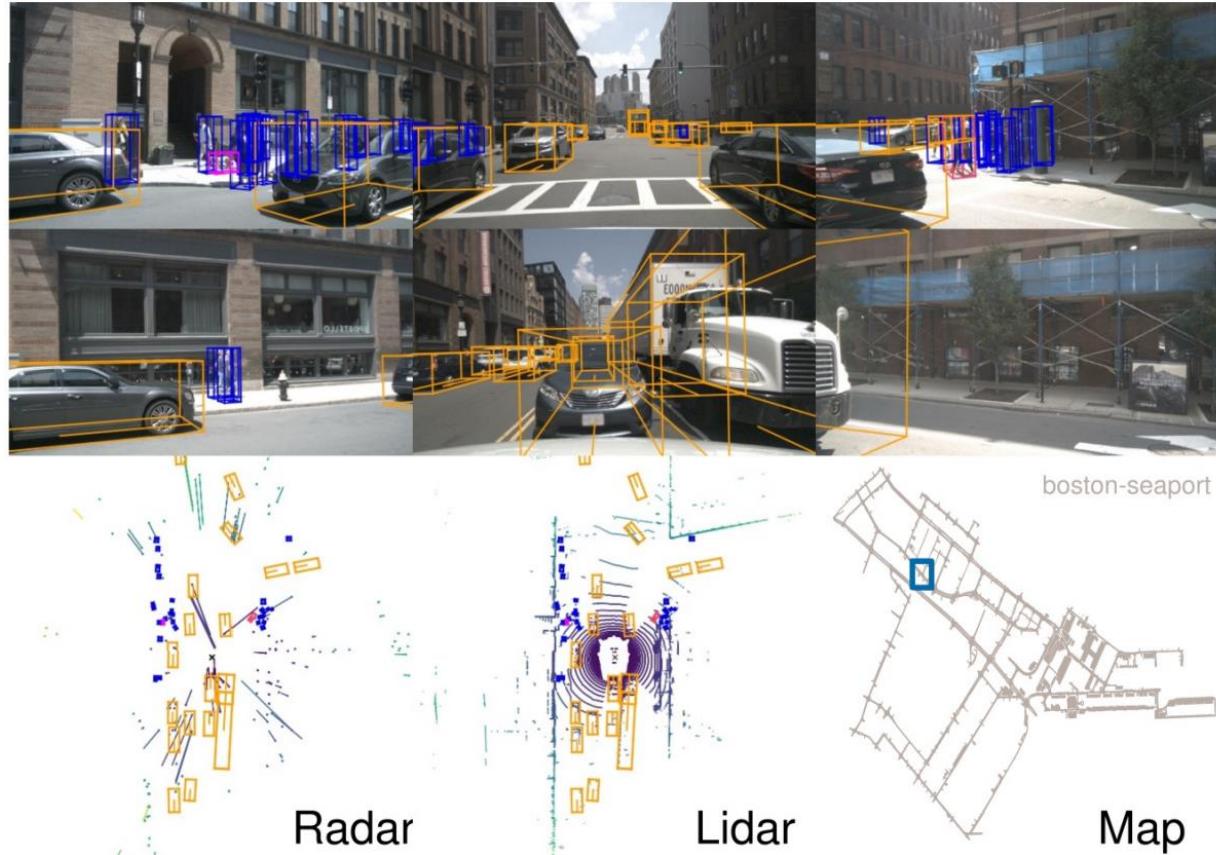
Winter Term 2023/2024

- Mobile Robotics (MA)
- ~~Sensordatenverarbeitung (BA)~~
- Environment Perception for Automotive Systems (BA/MA)
- Practical Course Data Fusion (BA/MA)

Block course in
September

Environment Perception for Automotive Systems (BA/MA)

- Report and Presentation
- Topic areas
 - Detection and tracking (lanes, pedestrians etc.)
 - Localization, GNSS
 - Collective perception
 - Scene understanding
 - Sensors: Radar, lidar, camera



H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving", In arXiv preprint arXiv:1903.11027.

Block Course in August/September

- Application of sensor fusion methods to real-world data
 - Robot perception
 - Localization
 - Tracking
- Team work
- Report and poster presentation

Lecture 2:

Folie 9: Notizen

distance measurement 1
 $d_1 = 23 \text{ m}$ $d_2 = 24 \text{ m}$ $d_3 = 19 \text{ m}$

estimate $\rightarrow \hat{d} = \frac{1}{3} (d_1 + d_2 + d_3) = 23 \text{ m}$

Folie 10:

- low dim. measurements: höhere Dimensionen herunterbrechen

- correlations: Sensordaten können voneinander abhängen
 ↳ Bsp. Frage "wie kalt ist es?" fällt anders aus wenn sich Leute darüber unterhalten haben

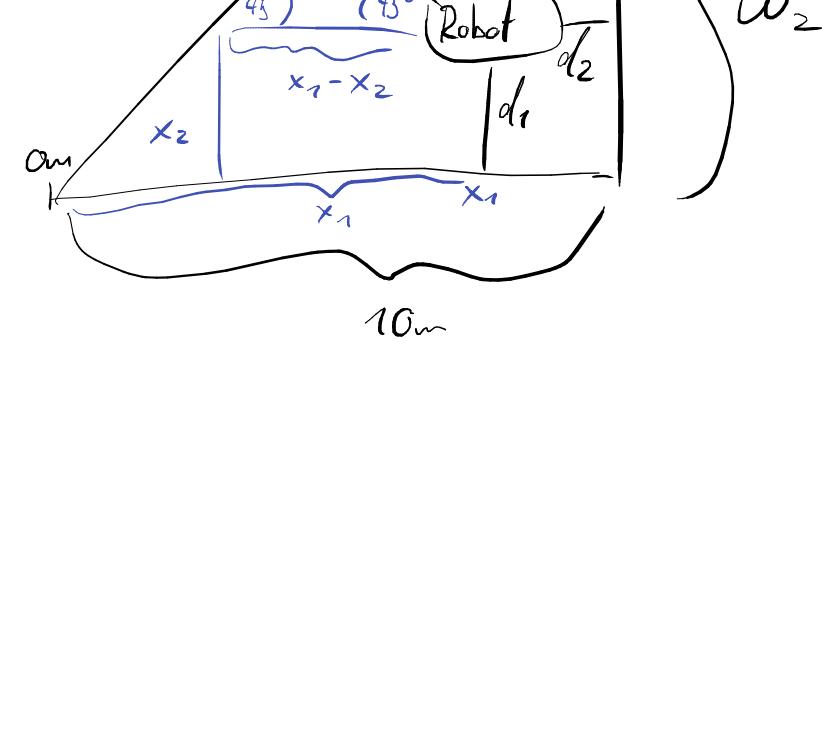
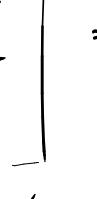
Folie 11:

- Review on Thursday

$x = [x_1, x_2]^T \in \mathbb{R}^2$

Measurement Equation (Measurement / observation model)

$$d_1 - [0 \ 1] \cdot x \\ 10 \text{ m} - d_2 = [1 \ 0] \cdot x \\ d_3 - \sqrt{\frac{1}{2}(x_1 - x_2)^2} = \left[\frac{1}{2} - \frac{1}{\sqrt{2}} \right] \cdot x$$



Stacked Measurement Equation

$$\begin{bmatrix} d_1 \\ 10 \text{ m} - d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \cdot x + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}$$

y H (Measurement Matrix) e (Measurement Error)

Folie 12: Estimator: Fusion Algorithm

Mapping y to x

$$y = Hx + e \quad H \in \mathbb{R}^{m \times n}$$

$$x^s = \underset{x}{\operatorname{argmin}} \|y - Hx\|^2_w$$

$w \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ weighted Matrix
 hier wird d. zweite Vektor doppelt gewichtet

$$= \underset{x}{\operatorname{argmin}} (y - Hx)^T w (y - Hx) \Rightarrow \text{Exercise}$$

Folie 15

$$c = [1 \ 2]^T \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$1x_1 + 2x_2$$

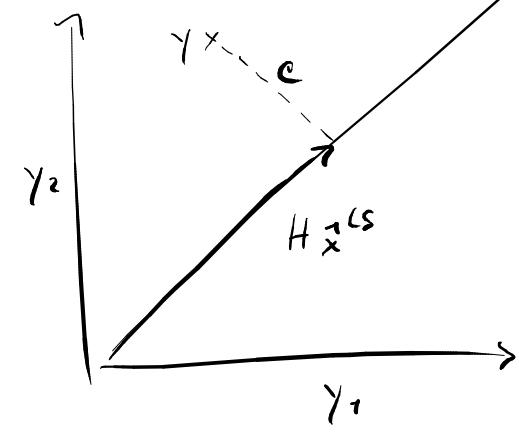
$$c = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 1x_1 + 2x_2$$

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \cdot c = \begin{bmatrix} 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = 1 \cdot 1 + 2 \cdot 2 = 5$$

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \cdot (c - \begin{bmatrix} 1 & 2 \end{bmatrix} \cdot x) = 0$$

$$h^T$$



blue 1 brown: Lineare Algebra Wiederholung

- **Unit Vector:** Vektoren der Länge 1 in jeweiliger Dimension
 $\hookrightarrow \vec{i} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \vec{j} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \rightarrow$ Auch als Basisvektoren bezeichnet

- the "span" of \vec{v} & \vec{w} is the set of all their linear combinations

$$a\vec{v} + b\vec{w}$$

lets say a, b vary over all real numbers

- **Linear Dependant:** wenn ein Vektor durch die Skalierung anderer Vektoren dargestellt werden kann, ist er von diesen anderen Vektoren linear abhängig

Bsp. $\vec{u} = a\vec{v} + b\vec{w}$ $\left| \begin{array}{l} \vec{u} \text{ stellt einfach einen beliebigen anderen Vektor dar} \\ \text{for some values of } a \& b \end{array} \right.$

- **Linear independent:** each Vektor adds another dimension to the span.

$$\vec{u} \neq a\vec{v} + b\vec{w}$$

For all values of $a \& b$

- **Linear Transformation:** Der Raum der Vektoren wird skaliert die Spalten der stielenden Matrix stellen die neuen Positionen der Basisvektoren dar. Beliebige Vektoren werden ansch. mit Matrix nach folgendem Schema verrechnet

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = x \begin{bmatrix} a \\ c \end{bmatrix} + y \begin{bmatrix} b \\ d \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix}$$

- **Linear Composition:** Mehrere Transformationen hintereinander durchführen

$$\underbrace{\begin{bmatrix} a & b \\ c & d \end{bmatrix}}_{M_1} \underbrace{\begin{bmatrix} e & f \\ g & h \end{bmatrix}}_{M_2} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

$$M_1 M_2 \neq M_2 M_1$$

- **Determinant:** Factor um den der Raum bei einer Matrixtransformation gestreckt wird. Bspw. wie sich die Fläche im 2D Raum beim Einheitsquadrat verhält. Wenn der Wert negativ ist, haben sich die Achsenorientierungen verändert. Bei einem Wert von 0 sind die Spaltenvektoren der Matrix nicht linear unabhängig.

$$\det \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$

$$\det \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} = a \det \begin{pmatrix} e & f \\ i & j \end{pmatrix} - b \det \begin{pmatrix} d & f \\ g & i \end{pmatrix}$$

$$+ c \det \begin{pmatrix} d & e \\ g & h \end{pmatrix}$$

- **Rank:** Wenn die Determinante einer Matrix 0 ist gibt es mehrere Szenarien

- Rank 1: alle Vektoren liegen nach der Transformation auf einer Linie (1 Dimension)
- Rank 2: -||- auf einer Ebene (2 Dimensionen)

\Rightarrow "Rank" \leftrightarrow Number of dimensions in the output of a transformation

- **full column/row rank:** when each of the rows/columns of a matrix are linear independent.

- **full rank:** beides trifft zu, bei einer quadratischen Matrix trifft das nur zu, wenn die Determinante ungleich null ist

- **Eigenvektoren:** diejenigen Vektoren, welche bei einer Matrix-Transformation auf ihrem span bleiben (sprich, sie werden nur durch einen Faktor (Eigenwert) skaliert)

\hookrightarrow Eigenwerte = 1 beschreiben daher Rotationen

$$\begin{aligned} \vec{v} &= \lambda \vec{v} && \xrightarrow{\substack{\text{rechte Seite in Matrix} \\ \text{Transf. überführen}}} \\ &\text{Matrix Vector} && \xrightarrow{\substack{\text{scalar multiplication} \\ \text{Multiplication}}} \\ &\downarrow && \xrightarrow{\substack{\text{Factor out } \vec{v}}} \\ A\vec{v} &= (\lambda I)\vec{v} && A\vec{v} = \vec{0} \end{aligned}$$

$(A - \lambda I)\vec{v} = \vec{0}$ \hookrightarrow zero vector. Der Vektor \vec{v} muss in den Ursprung transformiert werden d.h. die gesuchte Transformation besitzt eine Determinante = 0, der Vektor wird in eine niedrige Dimension transformiert

$\det(A - \lambda I) = 0$

\hookrightarrow um Eigenvektoren herauszufinden

müssen die Eigenwerte in die vorberechnete Matrix $(A - \lambda I)$ eingebaut werden: $(A - \lambda I) \cdot \vec{x} = \vec{0}$

anschließend muss das Gleichungssystem gelöst werden

Sensor Data Fusion

Linear Least Squares

Prof. Dr.-Ing. Marcus Baum

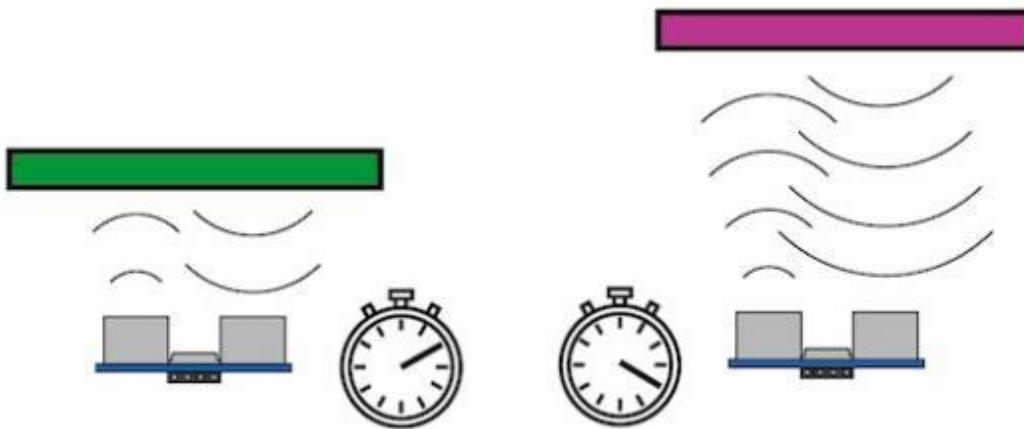
www.fusion.informatik.uni-goettingen.de



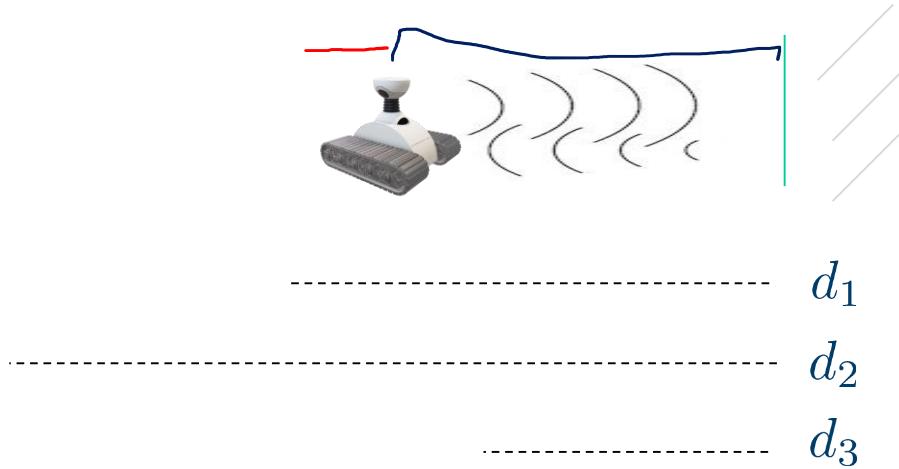
GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Above the frequencies of audible sound, i.e., $> 20 \text{ kHz}$
- Speaker sends out sound wave
- Microphone measures time-of-flight and calculates distance using speed of sound 340m/s.



<http://arcbotics.com/products/sparki/parts/ultrasonic-range-finder/>



- Three distance measurements:

$$d_1 = 23m , \quad d_2 = 24m , \quad d_3 = 19m$$

- Fusion by averaging:

$$\hat{x} = \frac{1}{3}(d_1 + d_2 + d_3) = \underline{\underline{22m}}$$

- Why taking the average?
- Varying quality of measurements?
- Low-dimensional measurements?
- Temporal evolution of state?
- Dependencies, i.e., correlations?
- Confidence in our estimate?
- Do we converge to the true value?

Distance Measurements to Walls

- Desired: Robot position

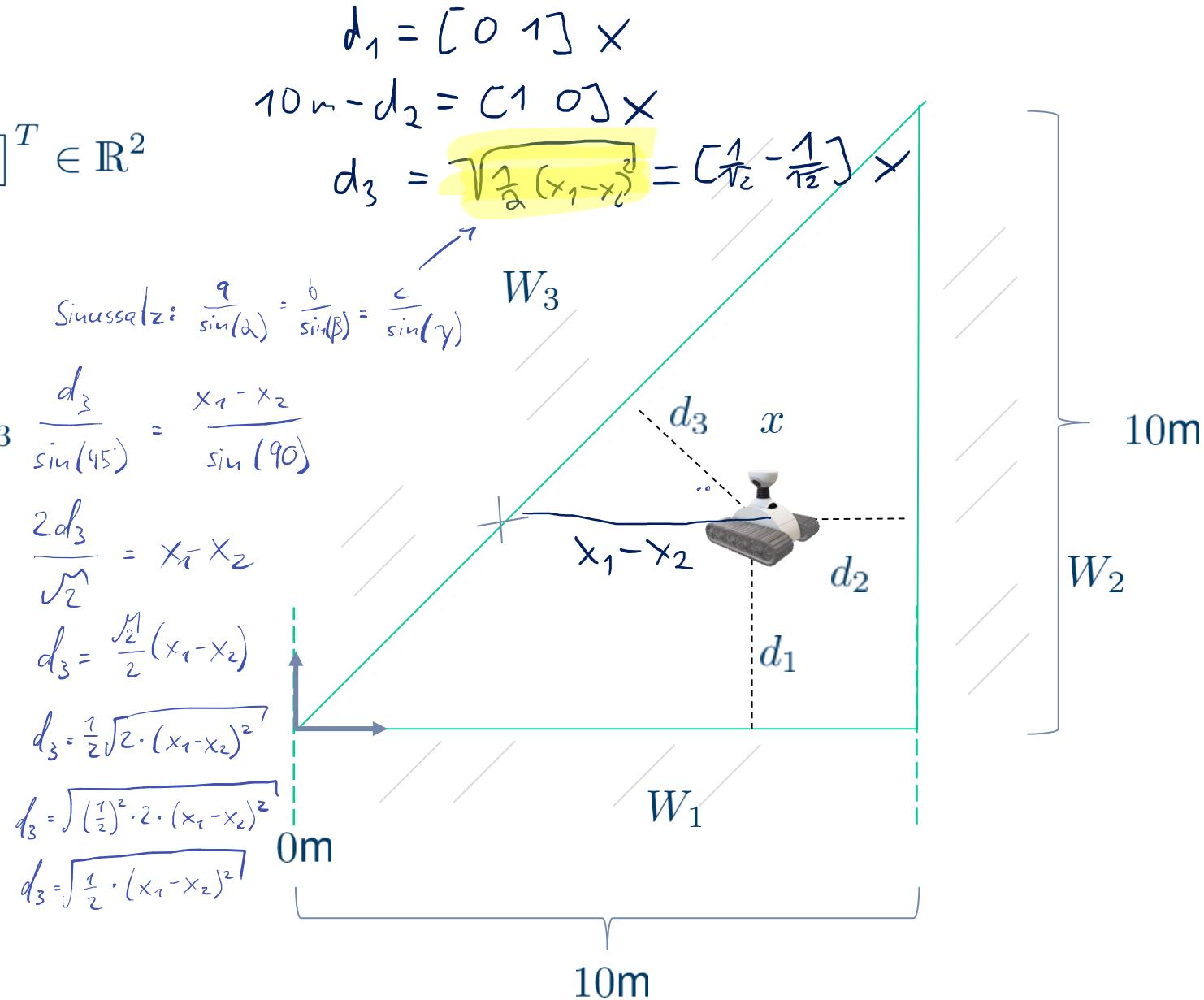
$$x = [x_1, x_2]^T \in \mathbb{R}^2$$

- Given:

- Three walls W_1, W_2, W_3
- Distance measurements d_1, d_2, d_3

- Measurement Equation:

$$\begin{bmatrix} d_1 \\ 10 - d_2 \\ d_3 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 1 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1 \\ e_3 \\ e_3 \end{bmatrix}}_{=:e}$$



State: Quantity of interest (unknown, desired)

$$x \in \mathbb{R}^n$$

Measurement: Observation received from the sensor (given)

$$y \in \mathbb{R}^m$$

Meas. Error: Sensor error due to noise (unknown)

$$e \in \mathbb{R}^m$$

Rechenregeln:

$$\Rightarrow \|A\|^2 = A^T A$$

$$\|y - Hx\|^2_w = (y - Hx)^T (y - Hx)_w$$

$$= (y^T - x^T H^T) (y - Hx)_w$$

$$= y^T \cdot y_w - y^T Hx_w - x^T H^T y_w - x^T H^T Hx_w$$

$$= w \cdot (y^T \cdot y - y^T Hx - x^T H^T y - x^T H^T Hx)$$

$$(y - Hx)^T W \cdot (y - Hx)$$

- Linear measurement equation:

$$y = Hx + e$$

Independent columns
full column rank

- Assumption: $m \geq n$ and $\text{rank}(H) = n$

- Objective: Find an x^{LS} so that

$$x^{LS} = \arg \min_x \|y - Hx\|_W^2$$

with semi-positive definite weighting matrix $W \in \mathbb{R}^{m \times m}$

$$\begin{pmatrix} 1 & \\ & 1_2 \end{pmatrix}$$

$\hat{\gamma}$ in der UC gegebene
 Bsp. weighting Matrix, hier
 wird d. dritte Dimension
 doppelt gewertet

Weighted Least Squares: Solution



$$c, x \in \mathbb{R}^n$$
$$A \in \mathbb{R}^{n \times n}$$

- Cost function:

$$\begin{aligned} G(x) &:= (y - \mathbf{H}x)^T \mathbf{W}(y - \mathbf{H}x) \\ &= \cancel{y^T \mathbf{W} y} - \cancel{y^T \mathbf{W} \mathbf{H} x} - \underbrace{x^T \mathbf{H}^T \mathbf{W} y}_{\not\in C} + \underbrace{x^T \mathbf{H}^T \mathbf{W} \mathbf{H} x}_{\not\in A} \end{aligned}$$

entfällt weil kein x enthalten $\hat{\cong} C^T$

- Derivative shall be zero vector $\mathbf{0}_{n \times 1}$:

$$\frac{\partial}{\partial x} G(x) = -\mathbf{H}^T \mathbf{W} y - \mathbf{H}^T \mathbf{W} y + 2\mathbf{H}^T \mathbf{W} \mathbf{H} x \stackrel{!}{=} \mathbf{0}_{n \times 1}$$

Frage: warum nicht $y^T \mathbf{W} \mathbf{H} x$? (aut ChafGBT zwar das gleiche,
ich verstehe die Umwandlung aber nicht)

⇒ Normal equation:

$$\boxed{\mathbf{H}^T \mathbf{W} y = \mathbf{H}^T \mathbf{W} \mathbf{H} x}$$

auf andere Seite ziehen

- As \mathbf{H} has full column rank:

$$x^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y$$

- $\frac{\partial}{\partial x} c^T x = \frac{\partial}{\partial x} x^T c = c$
(scalar-by-vector = column vector)
- $\frac{\partial}{\partial x} x^T \mathbf{A} x = 2\mathbf{A} x$
(scalar-by-vector = column vector)

weitere Matrix Redenzregeln:

$$(\mathbf{A}\mathbf{B})^T = \mathbf{B}^T \mathbf{A}^T$$



$$G(x) = (x - d_1)^2 + (x - d_2)^2$$

- Measurement equation:

$$\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot x + \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}$$

2 Messungen = 2 dim
1 dim state

State: Quantity of interest (unknown, desired) $x \in \mathbb{R}^n$
Measurement: Observation received from the sensor (given) $y \in \mathbb{R}^m$
Meas. Error: Sensor error due to noise (unknown) $e \in \mathbb{R}^m$

- Least squares solution (for equally weighted measurements): $x^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y$

$$x^{LS} = \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \frac{1}{2}(d_1 + d_2)$$

- One-dimensional state $x \in \mathbb{R}$
- Two dimensional measurement $y \in \mathbb{R}^2$
- Measurement equation:

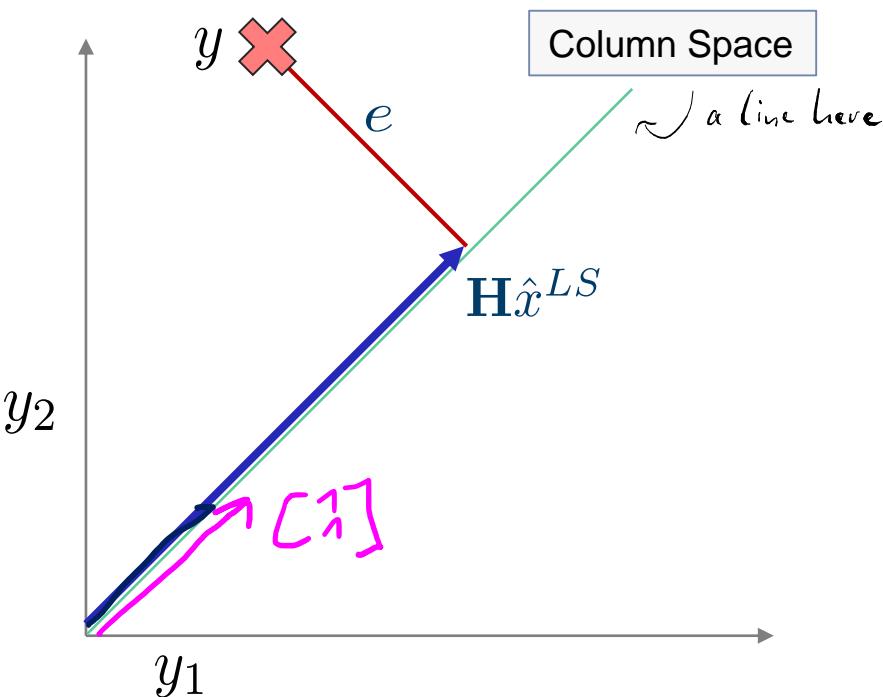
$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1 \\ e_2 \end{bmatrix}}_{=:e}$$

- Normal equation:

$$[1, 1] \cdot \underbrace{\left(y - \begin{bmatrix} 1 \\ 1 \end{bmatrix} x \right)}_e = 0$$

Measurement equation
nach \vec{e} umgestellt

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 3 \\ 5 & 0 \end{bmatrix}$$



- Desired:

Two-dimensional location $x \in \mathbb{R}^n$

$n = 2$

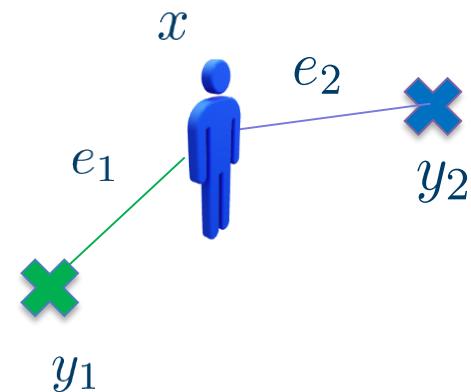
- Given:

- Meas. Sensor A $y_1 \in \mathbb{R}^n$ with \mathbf{W}_1
- Meas. Sensor B $y_2 \in \mathbb{R}^n$ with \mathbf{W}_2

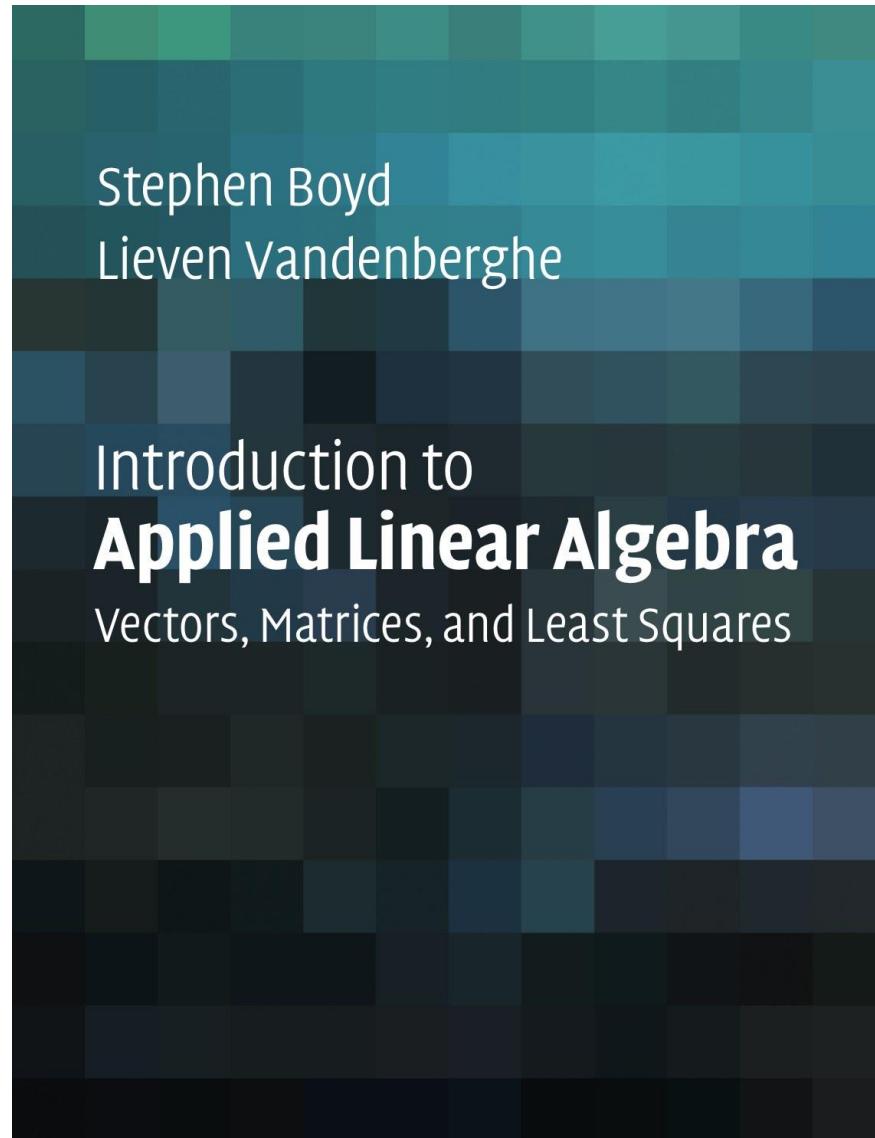
- Stacked measurement equation:

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1 \\ e_2 \end{bmatrix}}_{=:e},$$

$$\begin{aligned} x^{LS} &= \left(\begin{bmatrix} \mathbf{I}_2 \\ \mathbf{I}_2 \end{bmatrix}^T \begin{bmatrix} \mathbf{W}_1 & 0 \\ 0 & \mathbf{W}_2 \end{bmatrix} \begin{bmatrix} \mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix}^T \begin{bmatrix} \mathbf{W}_1 & 0 \\ 0 & \mathbf{W}_2 \end{bmatrix} y \\ &= (\mathbf{W}_1 + \mathbf{W}_2)^{-1} \cdot (\mathbf{W}_1 y_1 + \mathbf{W}_2 y_2) \end{aligned}$$



Wiederholung:
 Weighted least
 square solution
 $x^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y$
 ⚡ H has full column rank



Introduction to Applied Linear Algebra –
Vectors, Matrices, and Least Squares

Stephen Boyd and Lieven Vandenberghe

Cambridge University Press

Online available:
<https://web.stanford.edu/~boyd/vmls/>

Questions?

→ Stud.IP, e-mail, or live session

Sensor Data Fusion

Exercise 1

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

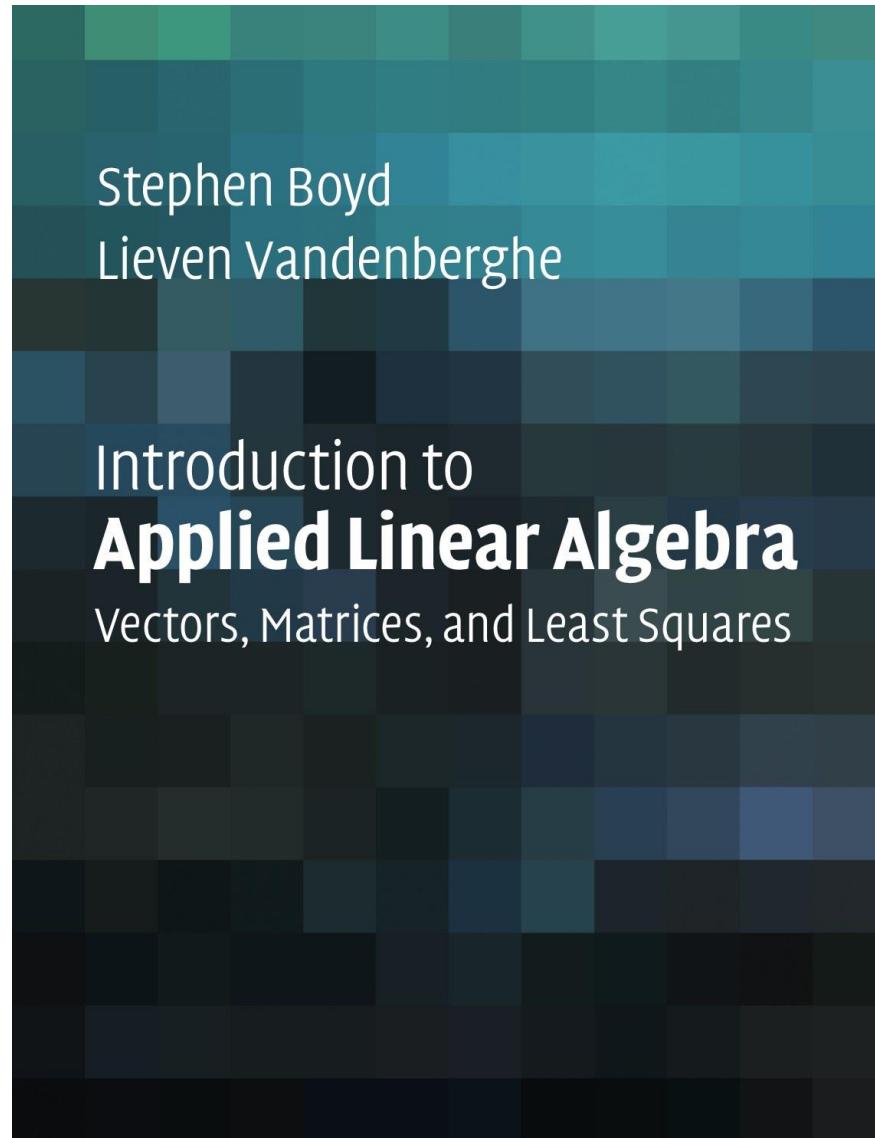
www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Math review
- Lecture review
- Questions
- Problem - Weight matrix visualization
- Homework



Introduction to Applied Linear Algebra –
Vectors, Matrices, and Least Squares

Stephen Boyd and Lieven Vandenberghe

Cambridge University Press

Online available:
<https://web.stanford.edu/~boyd/vmls/>

Petersen, Kaare Brandt, and Michael Syskind Pedersen.
"The matrix cookbook." *Technical University of Denmark*
7.15 (2008): 510.

- A symmetric matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ is positive definite if

$$\underline{\mathbf{z}^T \mathbf{W} \mathbf{z} > 0}$$

for all non-zero $\mathbf{z} \in \mathbb{R}^n$.

- Equivalent condition: All its eigenvalues are positive
- A positive semi-definite matrix \mathbf{W} can be written as

$$\underline{\mathbf{W} = \mathbf{C} \mathbf{D} \mathbf{C}^T},$$

where $\mathbf{C} \in \mathbb{R}^{n \times n}$ is a rotation matrix (orthogonal matrix), i.e., $\mathbf{C}^T \mathbf{C} = \mathbf{C} \mathbf{C}^T = \mathbf{I}$ and \mathbf{D} is a diagonal matrix with the eigenvalues

- Positive definite matrices are invertible



- For a matrix $\mathbf{H} \in \mathbb{R}^{m \times n}$
 - $\text{rank}(\mathbf{H})$ is defined as the max. number of independent columns (or eq. rows)
 - $\text{rank}(\mathbf{H}) \leq \min\{m, n\}$
- \mathbf{H} has full (column) rank if $\text{rank}(\mathbf{H}) = n$ and $n \leq m$
 - Kernel/Nullspace $\mathcal{N}(\mathbf{H}) = \{\mathbf{0}\}$, i.e., $\mathbf{Hx} = \mathbf{0}$ iff $x = \mathbf{0}$
 - $\mathbf{H}^T \mathbf{H} \in \mathbb{R}^{n \times n}$ is positive definite, Proof: $x^T \mathbf{H}^T \mathbf{H} x = (\mathbf{H}x)^T \mathbf{H} x = \|\mathbf{H}x\|^2 > 0$
 - $\mathbf{H}^T \mathbf{H}$ is invertible (follows from positive definite)



- The derivative of a scalar function w.r.t. a vector is a column vector
- The derivative of a vector function

$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$ w.r.t. a vector $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ is written as

↑ with respect to

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \dots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \dots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \dots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}.$$

(The transpose of the Jacobian)

2,
,

Assume a function $f(x) \in \mathbb{R}^m$ depending on a vector $x \in \mathbb{R}^n$ (f is differentiable at x). The Jacobian \mathbf{J} is defined as the $m \times n$ matrix of all first-order partial derivatives of f by x :

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} = \left(\frac{\partial f}{\partial x} \right)^T$$

Example: Given the function $f([x_1 \quad x_2]^T) = [x_1 + 2x_2 \quad x_1^2 + x_2]^T$, the Jacobian would be

$$\mathbf{J} = \begin{bmatrix} 1 & 2 \\ 2x_1 & 1 \end{bmatrix}$$

(2)
o

Assume a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$. The Hessian \mathbf{H} is defined as the $n \times n$ square matrix of all second-order partial derivatives of f by x :

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Example: Given a function $f([x_1 \quad x_2]^T) = 3x_1^2 + x_1x_2 + x_2^3$, the Hessian would be

$$\mathbf{H} = \begin{bmatrix} 6 & 1 \\ 1 & 6x_2 \end{bmatrix}$$

(2)
9

For $c, x \in \mathbb{R}^n$ and symmetric $\mathbf{A} \in \mathbb{R}^{n \times n}$, we have

1. $\frac{\partial}{\partial x} c^T x = \frac{\partial}{\partial x} x^T c = c$
(scalar-by-vector = column vector)
2. $\frac{\partial}{\partial x} x^T \mathbf{A} x = 2\mathbf{A} x$
(scalar-by-vector = column vector)
3. $\frac{\partial}{\partial x} \mathbf{A} x = \frac{\partial}{\partial x} x^T \mathbf{A} = \mathbf{A}$
(vector-by-vector = matrix)

Examples



1. $\frac{\partial}{\partial x} c^T x = \frac{\partial}{\partial x} x^T c = c$
(scalar-by-vector = column vector)

Set $x = [x_1 \quad x_2]^T$, $c = [1 \quad 2]^T$, and $f(x) = x^T c$. We have

$$f(x) = x_1 + 2x_2$$

$$\frac{\partial f}{\partial x} = \left[\frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2} \right]^T = [1 \quad 2]^T = c$$

2. $\frac{\partial}{\partial x} x^T \mathbf{A} x = 2 \mathbf{A} x$
(scalar-by-vector = column vector)

Set $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}$ and $g(x) = x^T \mathbf{A} x$. We have

$$g(x) = x_1(x_1 + 2x_2) + x_2(2x_1 + 6x_2) = x_1^2 + 4x_1x_2 + 6x_2^2$$

$$\frac{\partial g}{\partial x} = \left[\frac{\partial g}{\partial x_1} \quad \frac{\partial g}{\partial x_2} \right]^T = [2x_1 + 4x_2 \quad 4x_1 + 6x_2]^T = \begin{bmatrix} 2 & 4 \\ 4 & 6 \end{bmatrix} x = 2 \mathbf{A} x$$

Trace

- Given is a positive definite matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$.

- The trace of \mathbf{C} is the sum of its main diagonal, but also the sum of its eigenvalues.

$$\text{tr}\{\mathbf{C}\} = \text{tr}\{\mathbf{R}\mathbf{D}\mathbf{R}^T\} = \text{tr}\{(\mathbf{R}\mathbf{D})\mathbf{R}^T\} = \text{tr}\{\mathbf{R}^T\mathbf{R}\mathbf{D}\} = \text{tr}\{\mathbf{I}\mathbf{D}\} = \text{tr}\{\mathbf{D}\}$$

- Per definition, all eigenvalues of \mathbf{C} are non-negative. So we have $\text{tr}\{\mathbf{C}\} > 0$.
- Given a full rank matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$ with $\text{rank}\{\mathbf{R}\} = m$, $\mathbf{R}\mathbf{C}\mathbf{R}^T$ is still positive definite. So we still have $\text{tr}\{\mathbf{R}\mathbf{C}\mathbf{R}^T\} > 0$.

Assume non-zero $z \in \mathbb{R}^m$.

$$z^T \mathbf{R}\mathbf{C}\mathbf{R}^T z = (\underbrace{\mathbf{R}^T z}_x)^T \mathbf{C} \underbrace{\mathbf{R}^T z}_x > 0 \text{ for all non-zeros } x \in \mathbb{R}^n$$

Full column rank \mathbf{R}^T gives us $x = \mathbf{R}^T z = 0 \iff z = 0$

As we only regard non-zero z , x is also non-zero.

- A symmetric matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ is positive definite if
Wiederholung $z^T \mathbf{W} z > 0$
for all non-zero $z \in \mathbb{R}^n$.
- Equivalent condition: All its eigenvalues are positive

- Semi-positive definite weighting matrix \mathbf{W}

- Weighted Norm:

$$\|e\|_{\mathbf{W}}^2 = e^T \mathbf{W} e$$

- Example: For $m = 2$,

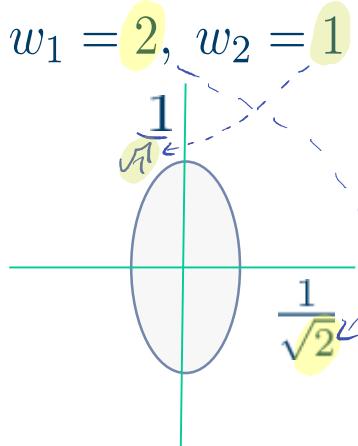
$$\mathbf{W} = \begin{bmatrix} w_1 & 0 \\ 0 & w_2 \end{bmatrix} \text{ and } e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}, \text{ yields}$$

$$\|e\|_{\mathbf{W}}^2 = w_1 e_1^2 + w_2 e_2^2$$

- Higher weight means more effect on estimate

Visualization:

$$w_1 = 2, w_2 = 1$$



$$\|e\|_{\mathbf{W}}^2 = 1$$

- a) Why do we need multiple sensor types?
- b) Name and explain two sensor fusion concepts.

Sensors have advantages and disadvantages. Using multiple different sensors can balance these out.

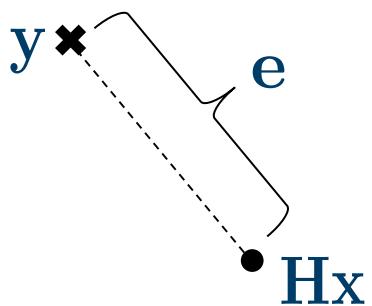
Two concepts are competitive fusion (multiple sensors measure the same space to increase accuracy) and complementary fusion (multiple sensors measure different spaces to increase completeness).

What is a measurement equation? Could you write down the linear measurement equation in a general form and explain the components in it?

The measurement equation projects the state onto the measurement space and describes the measurement as the sum of the projected state and an error.

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

with state $\mathbf{x} \in \mathcal{R}^n$, measurement $\mathbf{y} \in \mathcal{R}^m$, measurement matrix $\mathbf{H} \in \mathcal{R}^{m \times n}$, and error $\mathbf{e} \in \mathcal{R}^m$.



What are the assumptions for least squares?

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

$$\mathbf{H} \in \mathbb{R}^{m \times n}, \quad m \geq n, \quad \text{rank}(\mathbf{H}) = n$$

Assume semi-positive definite weight matrix with eigenvalue decomposition $\mathbf{W} = \mathbf{R}\mathbf{D}\mathbf{R}^T$. We get a rotation matrix with $\mathbf{R}\mathbf{R}^T = \mathbf{R}^T\mathbf{R} = \mathbf{I}$ and a diagonal matrix $\mathbf{D} = \begin{bmatrix} w_1 & 0 \\ 0 & w_2 \end{bmatrix}$.

R^T hier gleichbedeutend mit R⁻¹, weil orthogonale Basis (nur Rotation)

- Now visualize the weight matrix as before using $w_1 = 4$, $w_2 = 0.25$ and $\mathbf{R} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$ with $\alpha = \frac{\pi}{4}$
- Next, take the weight matrix $\mathbf{W}_2 = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix}$. Apply eigenvalue decomposition to obtain the elements of \mathbf{D} and calculate \mathbf{R} depending on α .

Reminder:

Eigenvalue λ with $\det(\mathbf{W} - \lambda\mathbf{I}_2) = 0$; $\sin(\alpha) = \cos(\alpha) \implies \alpha = \frac{\pi}{4}$

- Use \mathbf{D} and \mathbf{R} to visualize \mathbf{W}_2 like before.

- A positive semi-definite matrix \mathbf{W} can be written as

$$\mathbf{W} = \mathbf{CDC}^T,$$

where $\mathbf{C} \in \mathbb{R}^{n \times n}$ is a rotation matrix (orthogonal matrix), i.e., $\mathbf{C}^T\mathbf{C} = \mathbf{CC}^T = \mathbf{I}$ and \mathbf{D} is a diagonal matrix with the eigenvalues

alle Basisvektoren orthogonal zueinander & Länge = 1

Solution 1: Weight Matrix Visualization

$$w_1 = 4, w_2 = 0.25 \text{ and } \mathbf{R} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \text{ with } \alpha = \frac{\pi}{4}$$

$$\|\mathbf{e}\|_{\mathbf{W}}^2 = \mathbf{e}^T \mathbf{W} \mathbf{e} = \mathbf{e}^T \mathbf{R} \mathbf{D} \mathbf{R}^T \mathbf{e} = (\mathbf{R}^T \mathbf{e})^T \mathbf{D} \mathbf{R}^T \mathbf{e}$$

(aut wolfram alpha:

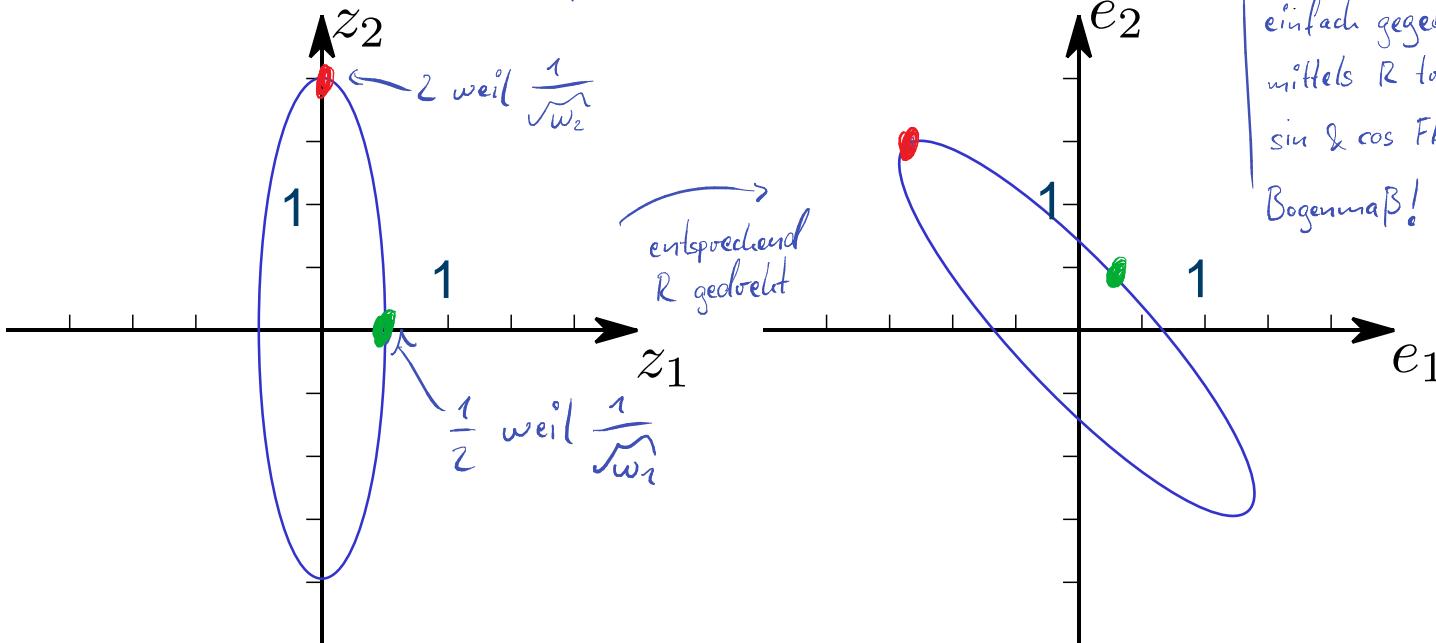
$$\mathbf{R} \mathbf{D} \mathbf{R}^T = \begin{pmatrix} 2.125 & 1.875 \\ 1.875 & 2.125 \end{pmatrix}$$

hier aber irrelevant

$$\mathbf{z} = \mathbf{R}^T \mathbf{e}$$

$$\mathbf{e} = \mathbf{R} \mathbf{z}$$

$$\mathbf{c} = \mathbf{y} - \mathbf{H} \mathbf{x}$$



Exkurs: $\mathbf{M} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$

$$f(x, y) = [x \ y] \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix} = ax^2 + 2bxy + cy^2$$

$$[x \ y] \cdot \begin{bmatrix} ax + by \\ bx + cy \end{bmatrix} = ax^2 + 2bxy + cy^2$$

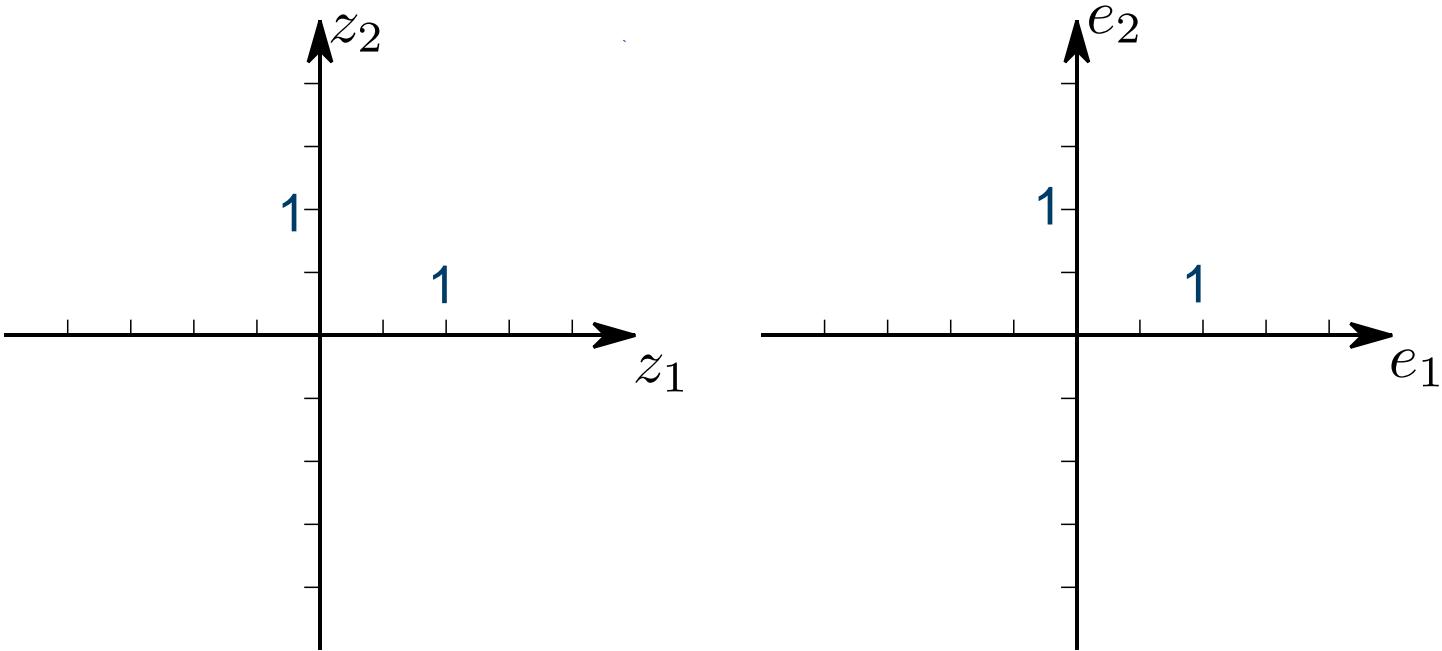
einfach gegebene Punkte nehmen & mittels R transformieren, wichtig: die sin & cos Fkt. rechnen hier mit dem Bogenmaß! Bsp. $R \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1.414 \\ 1.414 \end{pmatrix}$

Solution 1: Weight Matrix Visualization

$$\mathbf{W}_2 = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix}$$

Reminder:

Eigenvalue λ with $\det(\mathbf{W} - \lambda \mathbf{I}_2) = 0$; $\sin(\alpha) = \cos(\alpha) \implies \alpha = \frac{\pi}{4}$



- Next, take the weight matrix $\mathbf{W}_2 = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix}$. Apply eigenvalue decomposition to obtain the elements of \mathbf{D} and calculate \mathbf{R} depending on α .

Reminder:

$$\text{Eigenvalue } \lambda \text{ with } \det(\mathbf{W} - \lambda \mathbf{I}_2) = 0; \sin(\alpha) = \cos(\alpha) \implies \alpha = \frac{\pi}{4}$$

- Use \mathbf{D} and \mathbf{R} to visualize \mathbf{W}_2 like before.

$$\det(\mathbf{W} - \lambda \mathbf{I}) = \det \begin{bmatrix} 2.5 - \lambda & 1.5 \\ 1.5 & 2.5 - \lambda \end{bmatrix} = 0$$

$$(2.5 - \lambda)^2 - 1.5^2 = 0$$

$$2.5^2 - 2 \cdot 2.5 \cdot \lambda + \lambda^2 - 1.5^2 = 0$$

$$\frac{25}{4} - 5\lambda + \lambda^2 - \frac{9}{4} = 0$$

$$\lambda^2 - 5\lambda + 4 = 0$$

$$\lambda_{1,2} = -\frac{(-5)}{2} \pm \sqrt{\left(\frac{-5}{2}\right)^2 - 4}$$

$$\lambda_1 = 1; \lambda_2 = 4 \leftarrow \text{Eigenwerte}$$

Wiederholung: Binomische Formeln

Produktform	Quadratform	Summenform	Bezeichnung
$(a+b) \cdot (a+b)$	$(a+b)^2$	$a^2 + 2ab + b^2$	1. Binom
$(a-b) \cdot (a-b)$	$(a-b)^2$	$a^2 - 2ab + b^2$	2. Binom
$(a+b) \cdot (a-b)$		$a^2 - b^2$	3. Binom

Wiederholung: PQ-Formel

$$\text{Normalform: } x^2 + px + q = 0$$

$$\Rightarrow x_{1,2} = -\frac{p}{2} \pm \sqrt{\left(\frac{p}{2}\right)^2 - q}$$

$$\mathbf{W} = \mathbf{R} \mathbf{D} \mathbf{R}^T$$

$\hookrightarrow \mathbf{W} \mathbf{R}^T = \mathbf{D} \mathbf{R}^T$ (dann Werte einsetzen & lösen)

$$\Rightarrow \mathbf{R} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}^T$$

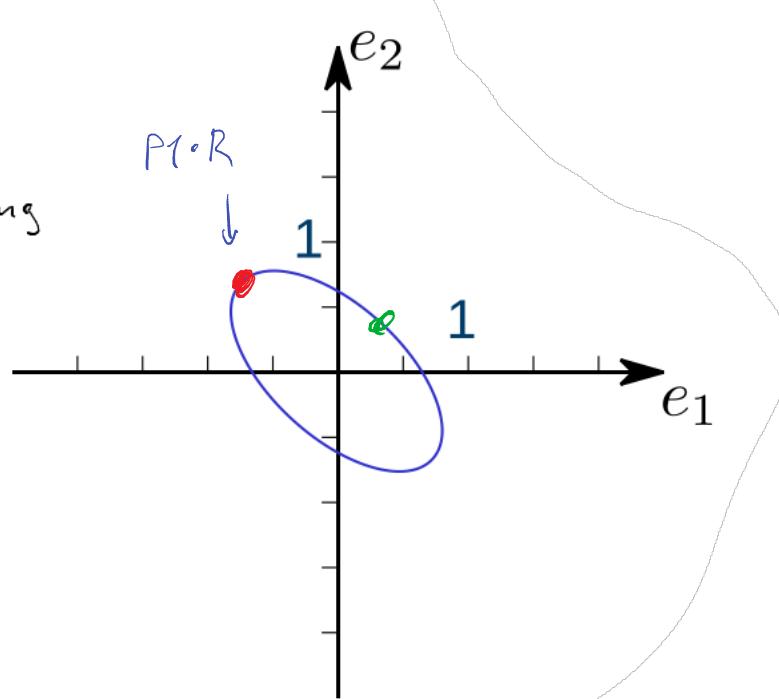
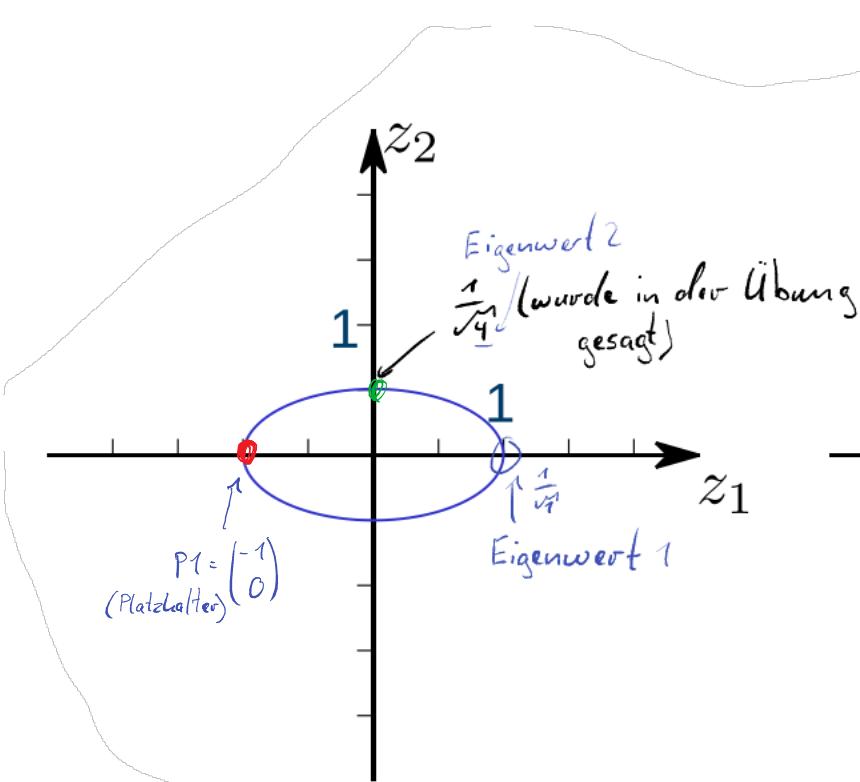
\uparrow gegebene Matrix \uparrow \uparrow
 $\mathbf{D} = \text{berechnete Eigenwerte}$

$$\Rightarrow 2.5 \cos(\alpha) + 1.5 \sin(\alpha) = \cos(\alpha) \quad | -2.5 \cos(\alpha)$$

$$1.5 \sin(\alpha) = -1.5 \cos(\alpha)$$

$$\begin{aligned} \sin(\alpha) &= -\cos(\alpha) \\ \alpha &= -\frac{\pi}{4} \end{aligned} \quad \left. \begin{array}{l} \text{in Aufgabenstellung} \\ \text{gegeben} \end{array} \right\}$$

nicht negativ, weil
Transponiert



Solution 1: Weight Matrix Visualization

$$\mathbf{W}_2 = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 2.5 \end{bmatrix}$$

$$det(\mathbf{W}_2 - \lambda \mathbf{I}) = 0 \quad \mathbf{R} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

$$(2.5 - \lambda)^2 - 1.5^2 = 0$$

$$\lambda^2 - 5\lambda + 6.25 - 2.25 = 0$$

$$\lambda^2 - 5\lambda + 4 = 0$$

$$\lambda_{1/2} = 2.5 \pm \sqrt{6.25 - 4}$$

$$\lambda_{1/2} = 2.5 \pm 1.5$$

$$\lambda_1 = 1$$

$$\lambda_2 = 4$$

$$\omega = \mathbf{R} \mathbf{D} \mathbf{R}^T$$

$\omega \mathbf{R}^T = \mathbf{D} \mathbf{R}^T$ dann Werte einsetzen

$$2.5a + 1.5c = a$$

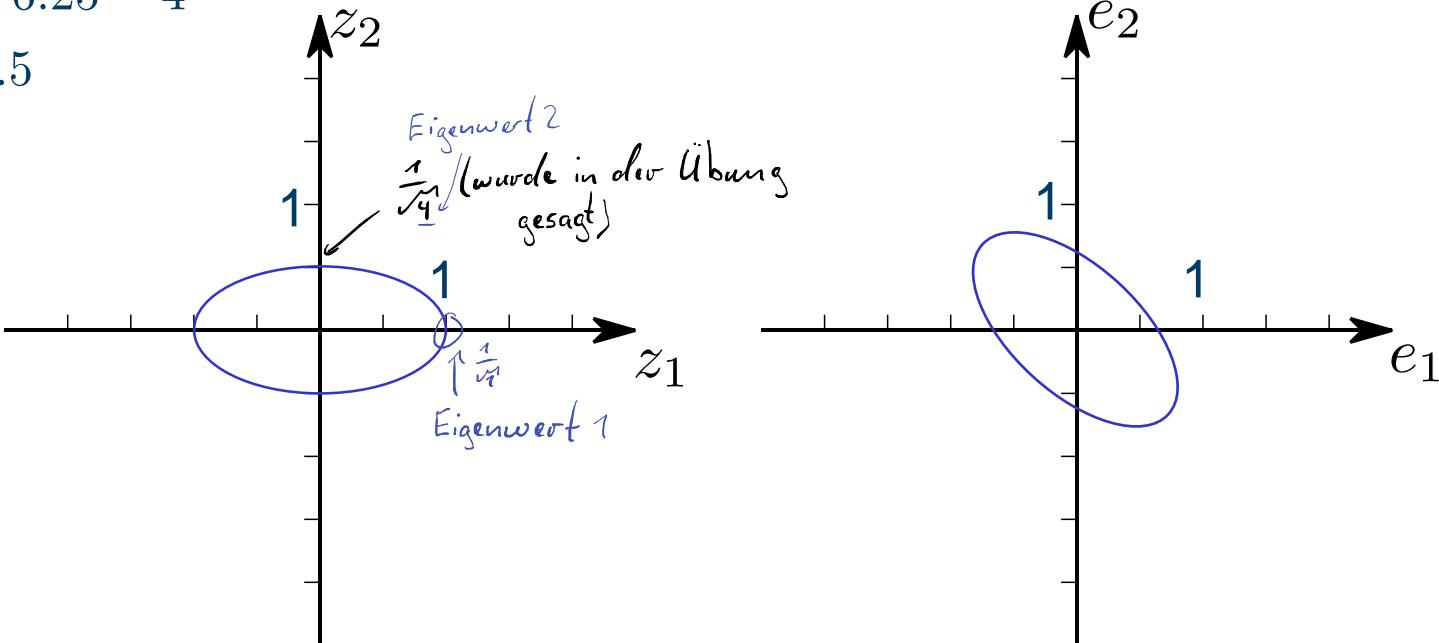
$$1.5a = -1.5c$$

$$a = -c$$

$$\cos(\alpha) = -\sin(\alpha)$$

$$\cos(-\alpha) = \sin(-\alpha)$$

$$\alpha = -\frac{\pi}{4}$$



Example – Distance measurements to walls

We measure $d_1 = 4\text{m}$, $d_2 = 3\text{m}$, and $d_3 = 7\text{m}$. Calculate the Least Squares solution.

$$70-7 \quad \begin{bmatrix} 4 \\ 3 \\ 3 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}}_{=:H} x^R + \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}}_{=:e}$$

Fehler e bew. W ignoriert

$$x^{\text{LS}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{y}$$

$$= \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 3.5 \\ 3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \\ 3 \end{bmatrix}$$

Wiederholung:

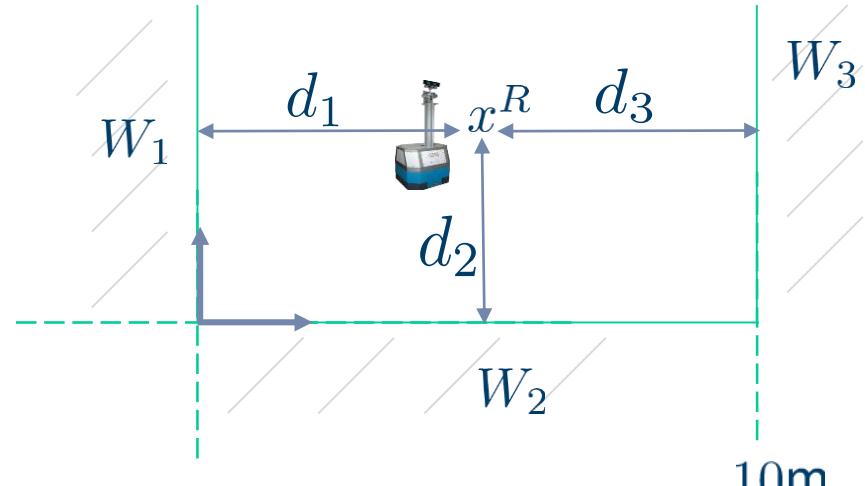
⇒ Normal equation:

- As \mathbf{H} has full column rank:

$$\boxed{\mathbf{H}^T \mathbf{W} \mathbf{y} = \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{x}}$$

auf andere Seite ziehen

$$x^{\text{LS}} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}$$



Exkurs: 2×2 Matrix Inverse

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

$$A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

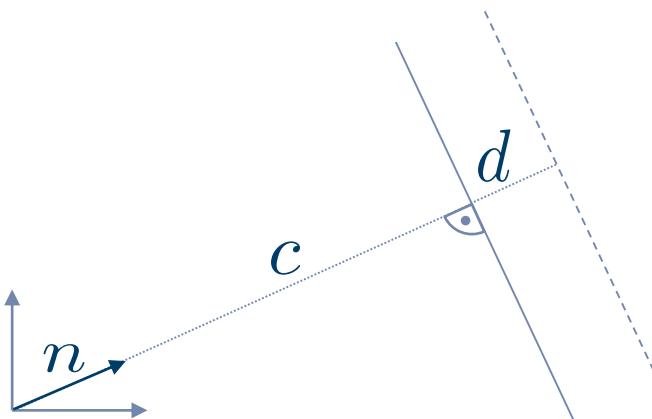
A wall in 2D space can be described using a normal vector $n = [n_1 \ n_2]^T$ (with $\|n\|_2 = 1$) and a scalar c

$$n_1 x_1 + n_2 x_2 = c \quad \text{← } c = \text{distance to the wall}$$

which holds true for all $x = [x_1 \ x_2]^T$ on the wall. Note that c in this case describes the shortest distance from the origin to the wall. So any shift d of c would describe all points within a distance d of the wall (see Figure below). So the formula

$$n_1 x_1 + n_2 x_2 = c + d$$

holds true for all x with distance d to the wall in the direction of n .



If the given vector \hat{n} is not a normal vector, so we have

$$\hat{n}_1 x_1 + \hat{n}_2 x_2 = \hat{c}$$

with $\|\hat{n}\|_2 \neq 1$, we need to normalize it first. This would result in the distance formula being

$$\frac{\hat{n}_1 x_1 + \hat{n}_2 x_2 - \hat{c}}{\|\hat{n}\|_2} = d$$

$$\begin{aligned}\hat{n}_1 x_1 + \hat{n}_2 x_2 - c &= d \cdot \|\hat{n}\| \\ \hat{n}_2 x_2 &= d \cdot \|\hat{n}\| - \hat{n}_1 x_1 + c \\ x_2 &= \frac{d \cdot \|\hat{n}\| - \hat{n}_1 x_1 + c}{\hat{n}_2}\end{aligned}$$

The objective is to estimate the two-dimensional object location $x = [x_1, x_2]^T \in \mathbb{R}^2$ using (noise-corrupted) distance measurements $d^i \in \mathbb{R}$ to N walls. The location of the i -th wall is given in normal form

$$n_1^i \cdot x_1^w + n_2^i \cdot x_2^w = c^i .$$

Assume n^i points to the half space where the object is located. Given are four walls with corresponding measurements:

i	n_1^i	n_2^i	c^i	distance d^i
1	-5	-1	-45	4.7
2	-1	-8	-70	5.2
3	-1	9	5	5.5
4	8	-1	7	4.5

- a) Please write a function which visualizes walls and measurements using different colors.
- b) Formulate a linear measurement equation $y^i = \mathbf{H}^i x + e^i$, which relates the measurement to the i -th wall with x and the error e^i . In the same manner, formulate a measurement equation that relates N walls, i.e., the 1-st to N -th walls, with x and e . Note that the n^i vectors are not normalized.

c) Could you calculate the unique location for the first case in 1b), if not, please explain. If an unique location could be obtained, which requirements are needed?

d) Based on the measurement equation formulated in 1b), write a function which calculates the least squares solutions based on the measurements.

Using the function you implemented, calculate the least squares solutions \hat{x}_{12} , \hat{x}_{34} and \hat{x}_{1234} based on the measurements (y_1, y_2) , (y_3, y_4) as well as (y_1, y_2, y_3, y_4) .

e) Given the true location x , implement a function which calculates the estimation error e using Euclidean norm, e.g.,

$$e_{12} = \|\hat{x}_{12} - x\|, e_{34} = \|\hat{x}_{34} - x\|, e_{1234} = \|\hat{x}_{1234} - x\|$$

Assuming the true location $x = [5, 5]^T$, calculate e_{12} , e_{34} and e_{1234} . What can you observe?

The objective is to estimate the two-dimensional object location $x = [x_1, x_2]^T \in \mathbb{R}^2$ using (noise-corrupted) distance measurements $d^i \in \mathbb{R}$ to N walls. The location of the i -th wall is given in normal form

$$n_1^i \cdot x_1^w + n_2^i \cdot x_2^w = c^i .$$

Assume n^i points to the half space where the object is located. Given are four walls with corresponding measurements:

i	n_1^i	n_2^i	c^i	distance d^i
1	-5	-1	-45	4.7
2	-1	-8	-70	5.2
3	-1	9	5	5.5
4	8	-1	7	4.5

- a) Please write a function which visualizes walls and measurements using different colors.

b) Formulate a linear measurement equation $y^i = \mathbf{H}^i x + e^i$, which relates the measurement to the i -th wall with x and the error e^i . In the same manner, formulate a measurement equation that relates N walls, i.e., the 1-st to N -th walls, with x and e . Note that the n^i vectors are not normalized.

The distance d^i to wall i

$$d^i = \frac{n_1^i x_1 + n_2^i x_2 - c^i}{\sqrt{(n_1^i)^2 + (n_2^i)^2}} + e^i$$

Man kann doch diesen Faktor nicht überziehen, die e -Komponente aber unberücksichtigen?

↳ in Übung meint er, das wird erstmal vernachlässigt
"we can just assume it's a different error"

$$d^i \sqrt{(n_1^i)^2 + (n_2^i)^2} + c^i = [n_1 \quad n_2] \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x + e^i$$

$$\underbrace{d^i \sqrt{(n_1^i)^2 + (n_2^i)^2} + c^i}_{y^i} = \underbrace{\begin{bmatrix} n_1^1 & n_2^1 \\ \vdots & \vdots \\ n_1^i & n_2^i \\ \vdots & \vdots \\ n_1^N & n_2^N \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x + \underbrace{\begin{bmatrix} e^1 \\ \vdots \\ e^i \\ \vdots \\ e^N \end{bmatrix}}_e$$

Distances to N walls

c) Could you calculate the unique location for the first case in 1b), if not, please explain. If an unique location could be obtained, which requirements are needed?

It is not possible to get an exact solution because we have 2 unknowns and only 1 formula. The exact solution can be calculated if $\text{rank}(\mathbf{H}) = 2$.

)

d) Based on the measurement equation formulated in 1b), write a function which calculates the least squares solutions based on the measurements. Using the function you implemented, calculate the least squares solutions \hat{x}_{12} , \hat{x}_{34} and \hat{x}_{1234} based on the measurements (y_1, y_2) , (y_3, y_4) as well as (y_1, y_2, y_3, y_4) .

e) Given the true location x , implement a function which calculates the estimation error e using Euclidean norm, e.g.,

$$e_{12} = \|\hat{x}_{12} - x\|, e_{34} = \|\hat{x}_{34} - x\|, e_{1234} = \|\hat{x}_{1234} - x\|$$

Assuming the true location $x = [5, 5]^T$, calculate e_{12} , e_{34} and e_{1234} . What can you observe?

Sensor Data Fusion

Nonlinear Least Squares: Analytic Approaches

www.fusion.informatik.uni-goettingen.de

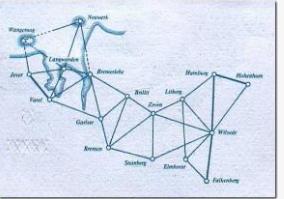
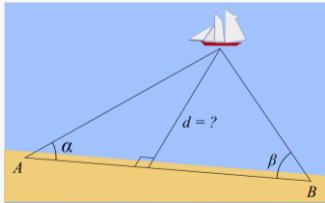


GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Dead reckoning
12. Dynamic models: Tracking (**live**)
13. Advanced Topic
14. Summary and Discussion (**live**)

Triangulation: Angular measurements



Trilateration: Distance measurements



Signal strength



Time-Of-Arrival (TOA)



dienen i.d.R. als Backup für GPS etc.

Multilateration: Distance differences



Time-Difference-
Of-Arrival (TDOA)



Lecture:
SVY1110: Introduction to GPS

Peter Gibbings
University of Southern Queensland (USQ)

<http://www.usq.edu.au/course/material/SVY2106/lectures/Breeze/SVY1110CodeRange/>

Global Positioning System (GPS)

- Worldwide radio-navigation system developed and operated by the U.S.
- Allows localization on Earth



Global Navigation Satellite System (GNSS)

Any satellite navigation system:

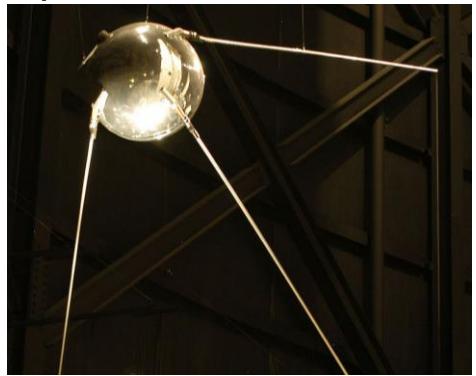
- GPS (U.S., since 1994)
- GLONASS (Russia, since 1995/2010)
- Galileo (European Union, since 2018)
- BeiDou (China, test system since 2000)



U.S. Department of Defences (DOD) sponsored project

- Inspired by ground-based radio-navigation systems from 1940s
- Feasibility studies in 1960s during Cold War (TRANSIT)
- First satellite launched in 1978
- 24 satellites in mid 1990s; currently 28 in orbit
- Assisted GPS in 2004

Sputnik 1, 1957



TRANSIT 1, Tracking Station



- U.S. scientists determined Sputnik's orbit using Doppler shift
- Based on this idea, TRANSIT was developed in the late 1950s

LATITUDE

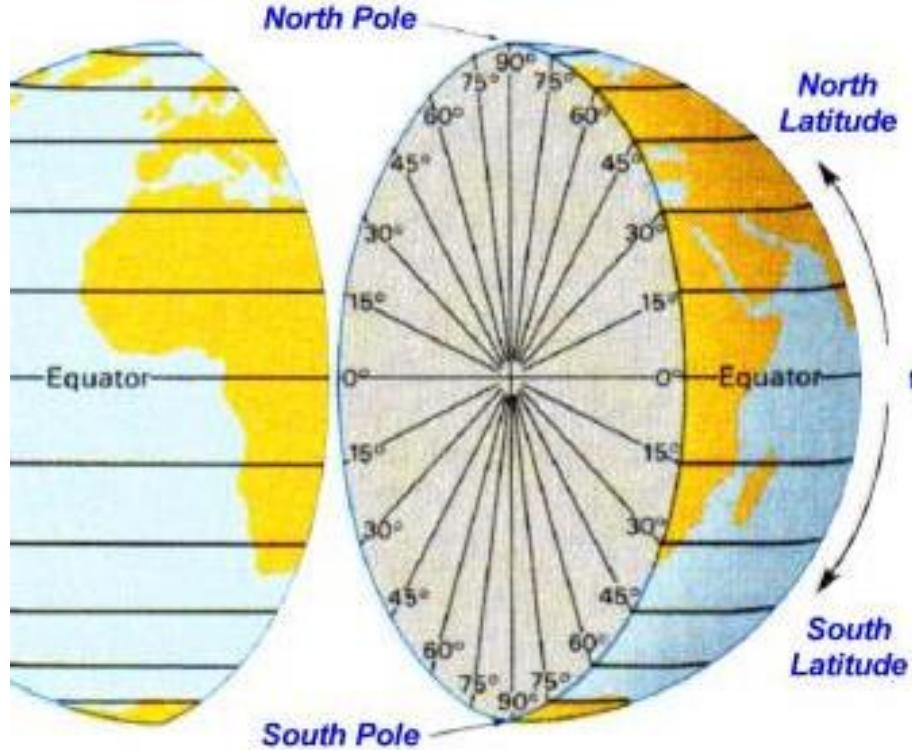
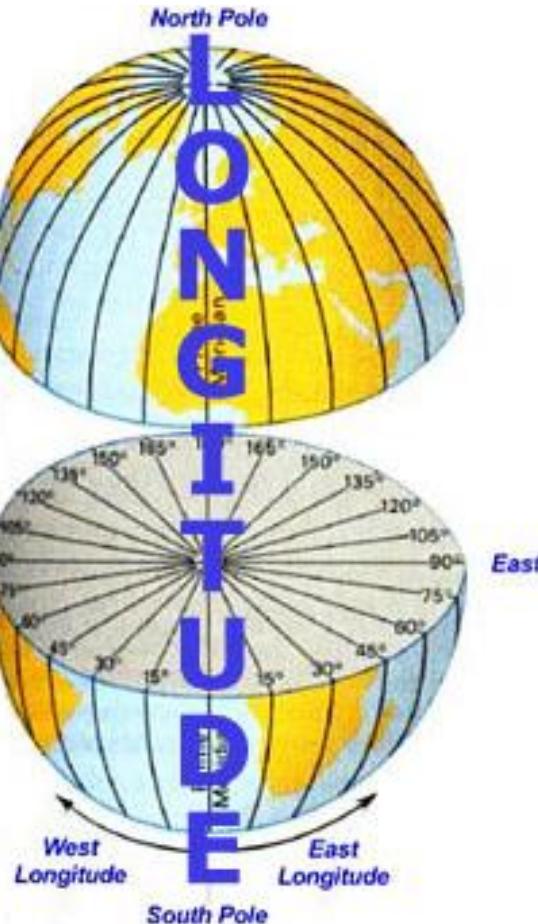


Image globes: geographyworldonline.com



Latitude:

- Parallels from the equator
- North: “+”

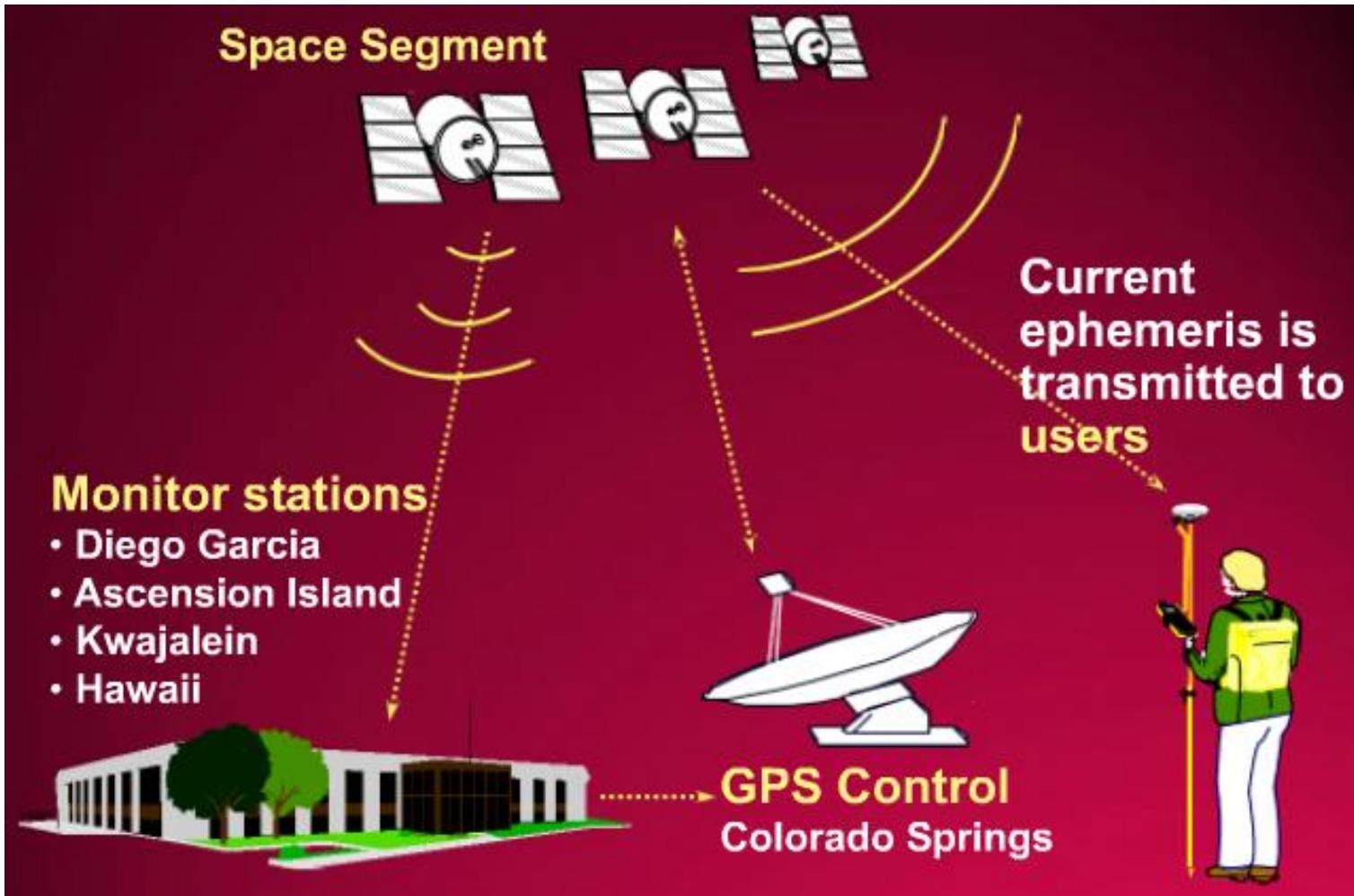
Longitude:

- Meridians from Greenwich
- East: “+”

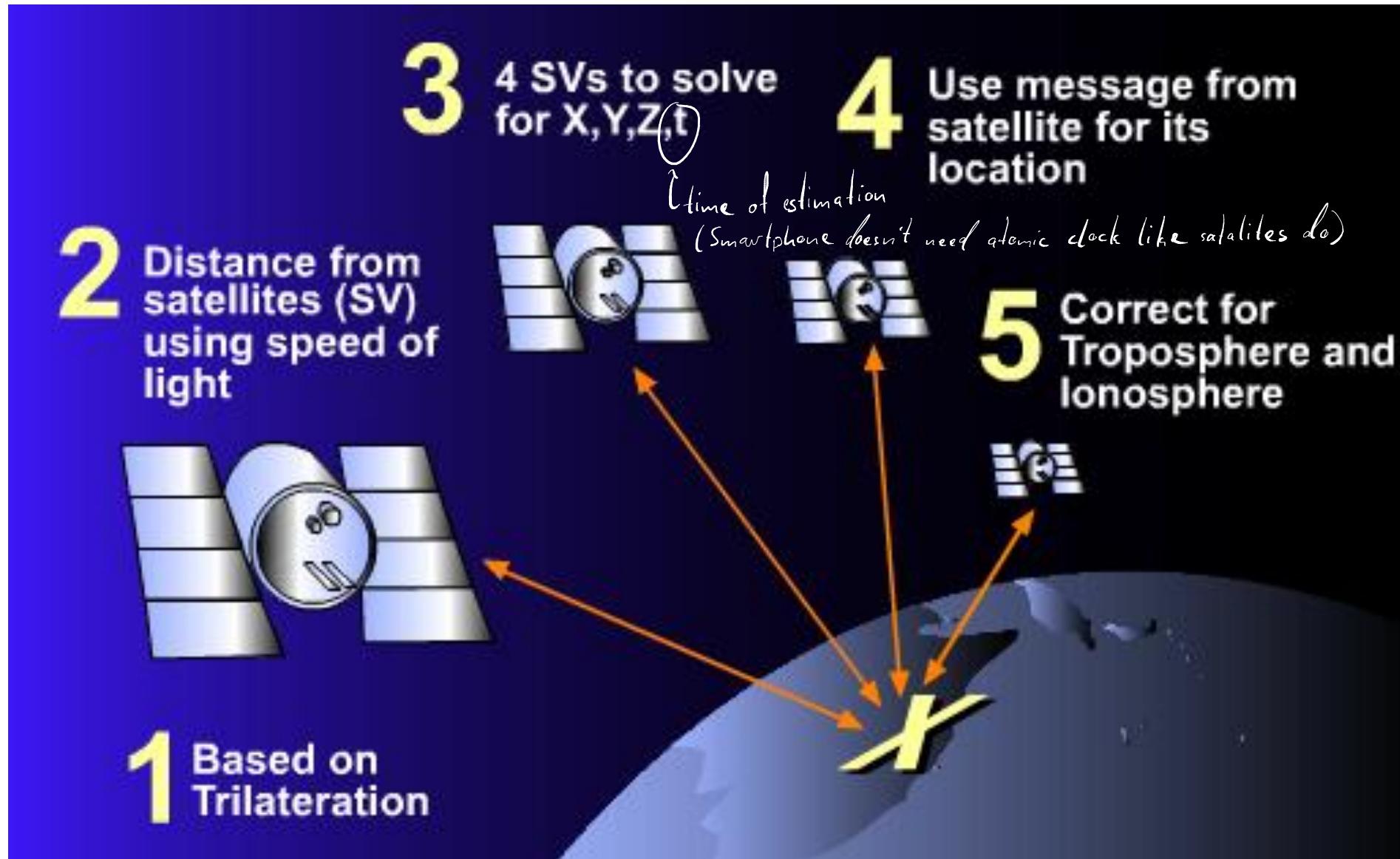
Altitude:

- Above Mean Sea Level (MSL)
- Up: “+”

- **Space-satellites**
 - 24 Satellites
 - 20.000 km above MSL
 - At least 4 SVs visible at anytime, anywhere
- **Ground stations**
 - Track SV orbits
 - Monitor clocks
 - Update ephemeris & clock corrections for each SV
- **Users/GPS receivers**
 - Convert SV signals into position, velocity, and time estimates



Introduction to GPS, Peter Gibbins

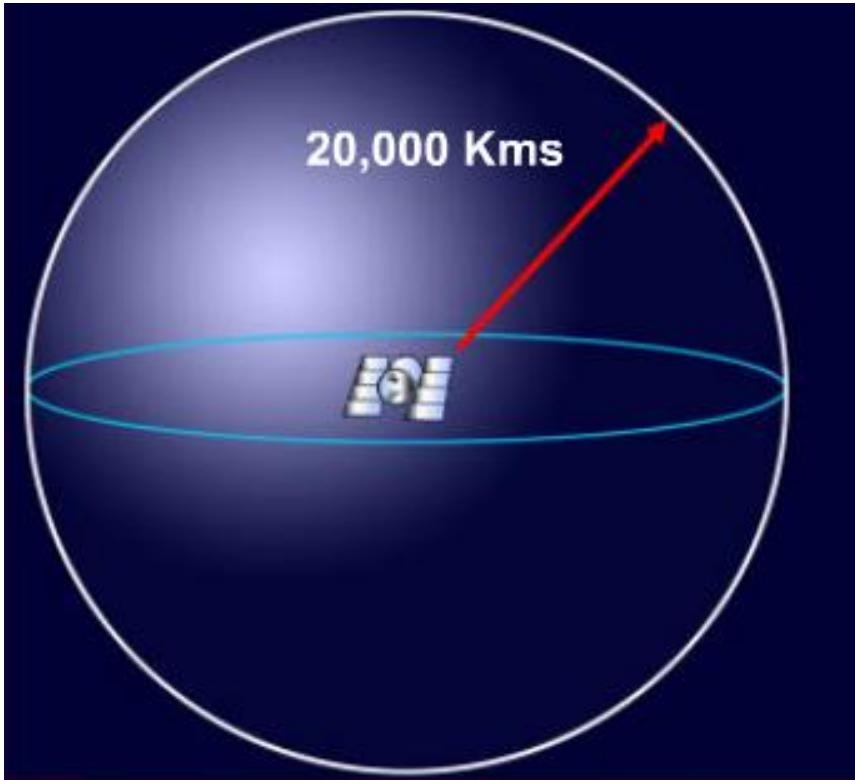


Introduction to GPS, Peter Gibbins

Step 1: Trilateration (One Measurement)

Trilateration is the process of determining locations with distance measurements

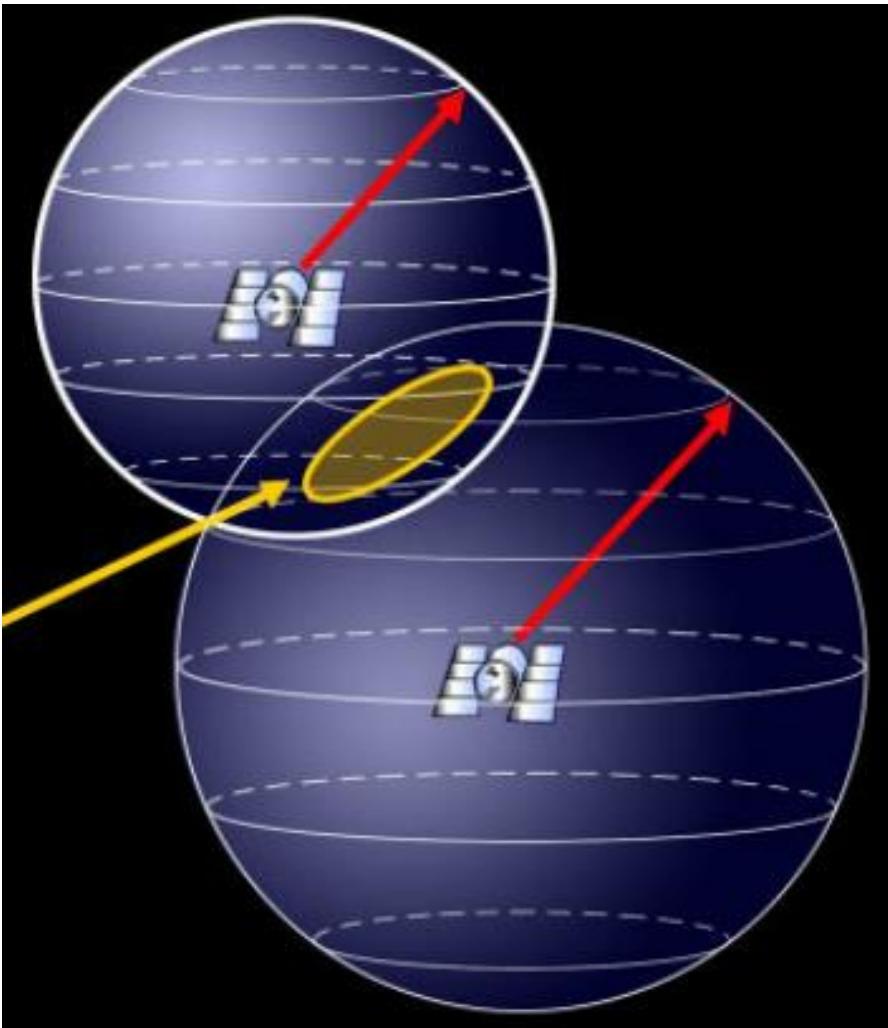
One distance measurement narrows down the location to a sphere



Introduction to GPS, Peter Gibbons

Step 1: Trilateration (Two Measurements)

Two distance measurements
narrow down the location to
a circle

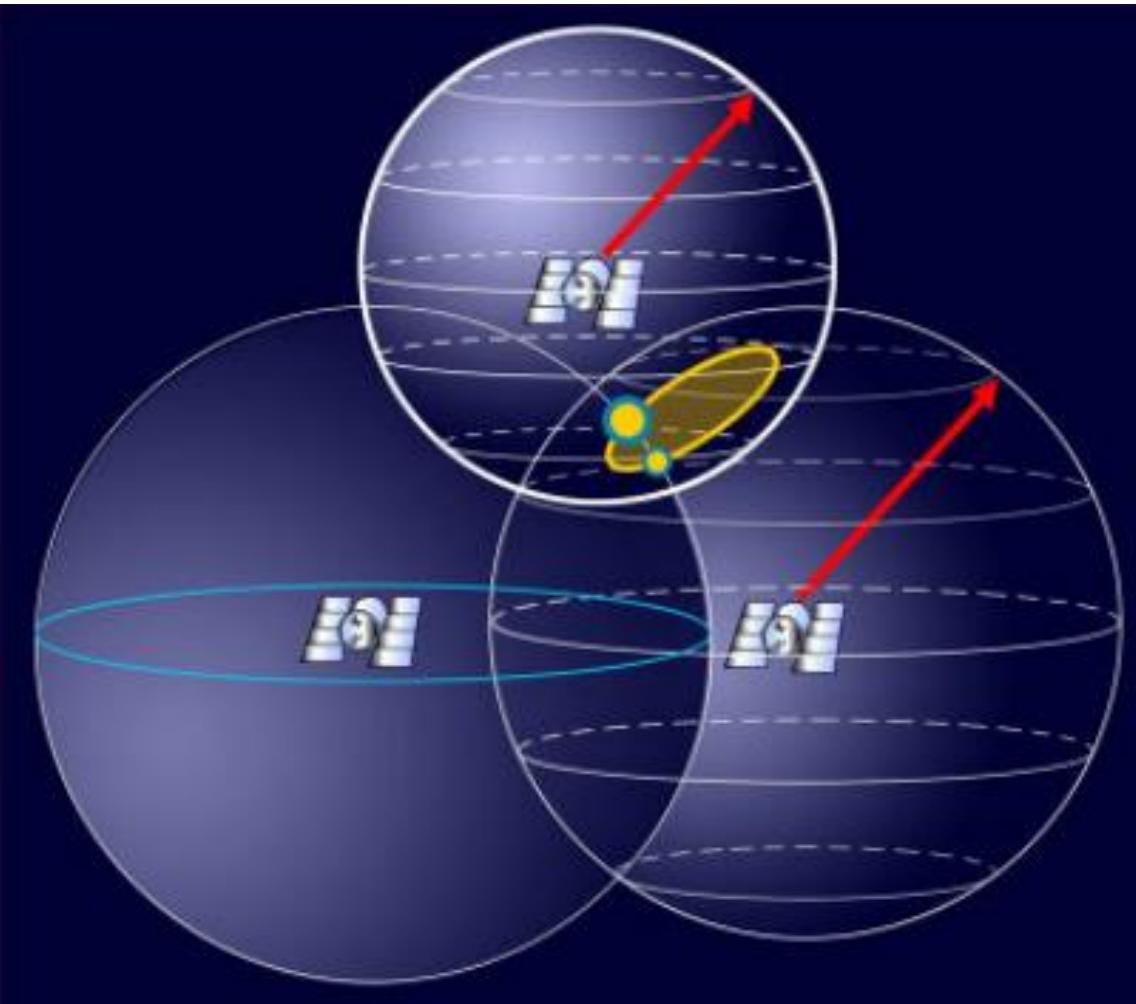


Introduction to GPS, Peter Gibbons

Step 1: Trilateration (Three Measurements)

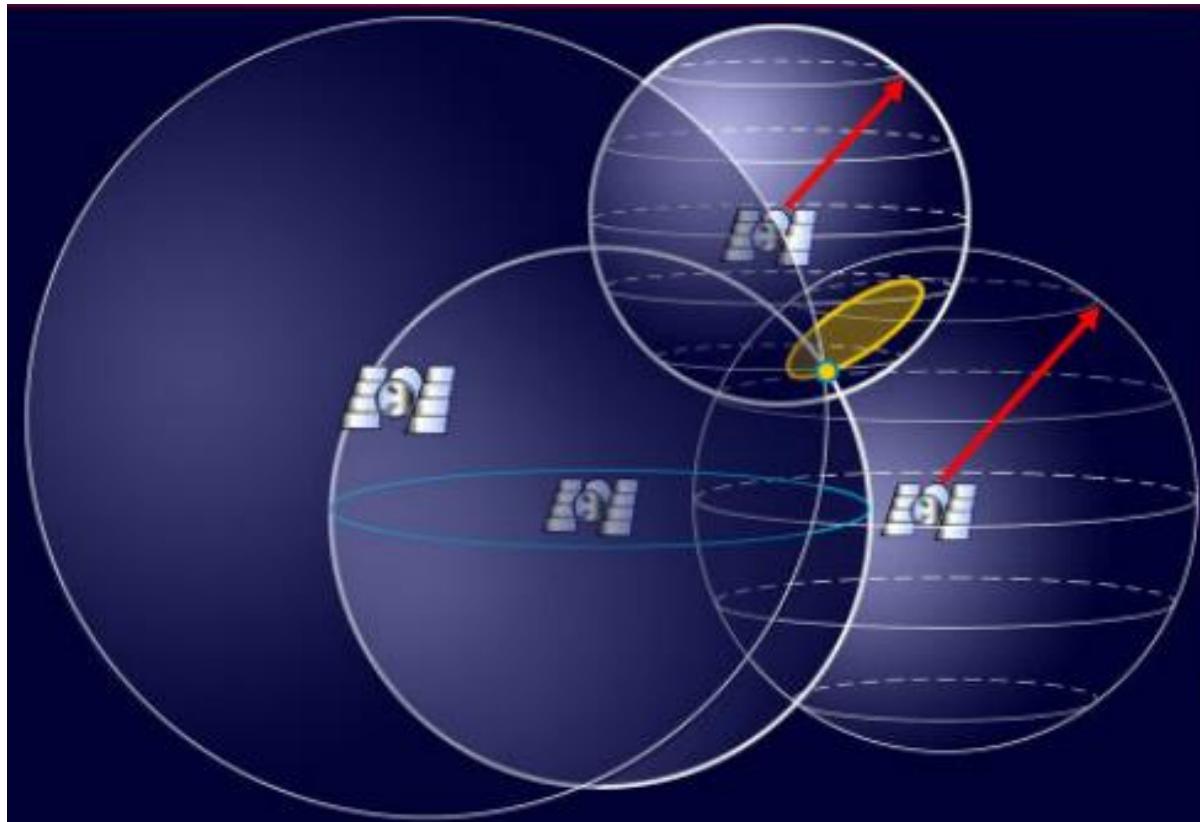
Three distance measurements result in two possible locations

One location is often not on Earth and can be cancelled out



Step 1: Trilateration (Four Measurements)

Four distance
measurements result
in a unique location



Introduction to GPS, Peter Gibbons

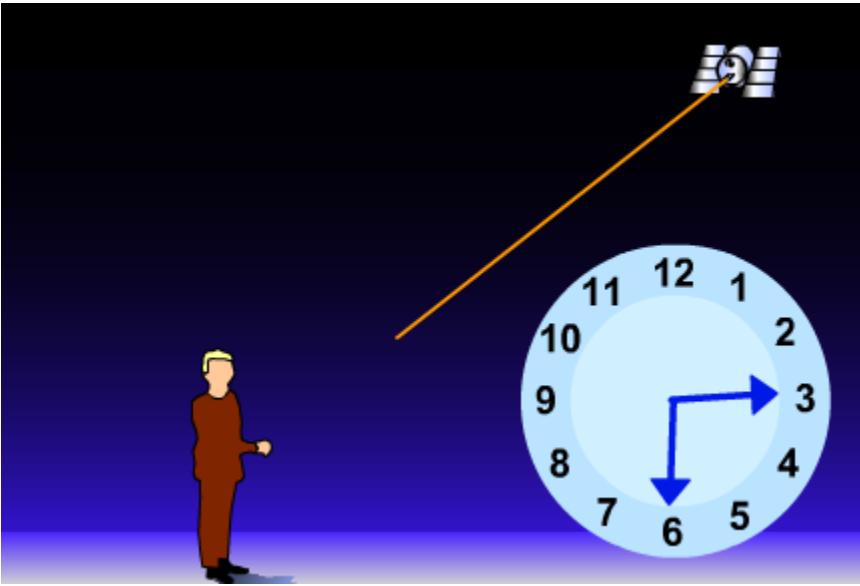
Step 2: Measuring the Distance

- Satellite sends out signal
- Receiver measures how long it takes to arrive
- Distance can be calculated based on speed-of-light

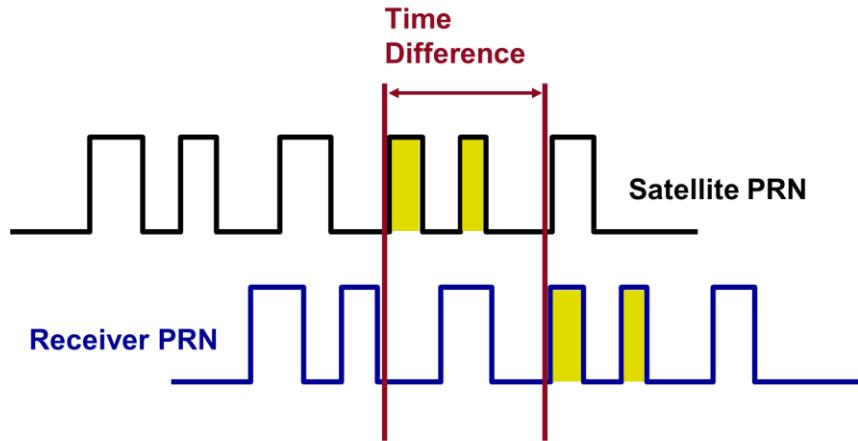
$$\text{Dist (km)} = \Delta T (\text{s}) \times 300.000 \text{ (km/s)}$$

Pseudo Range Code

- Same code on receiver and sender
 - Clocks are synchronized
- Receiver can determine how long ago the signal was generated



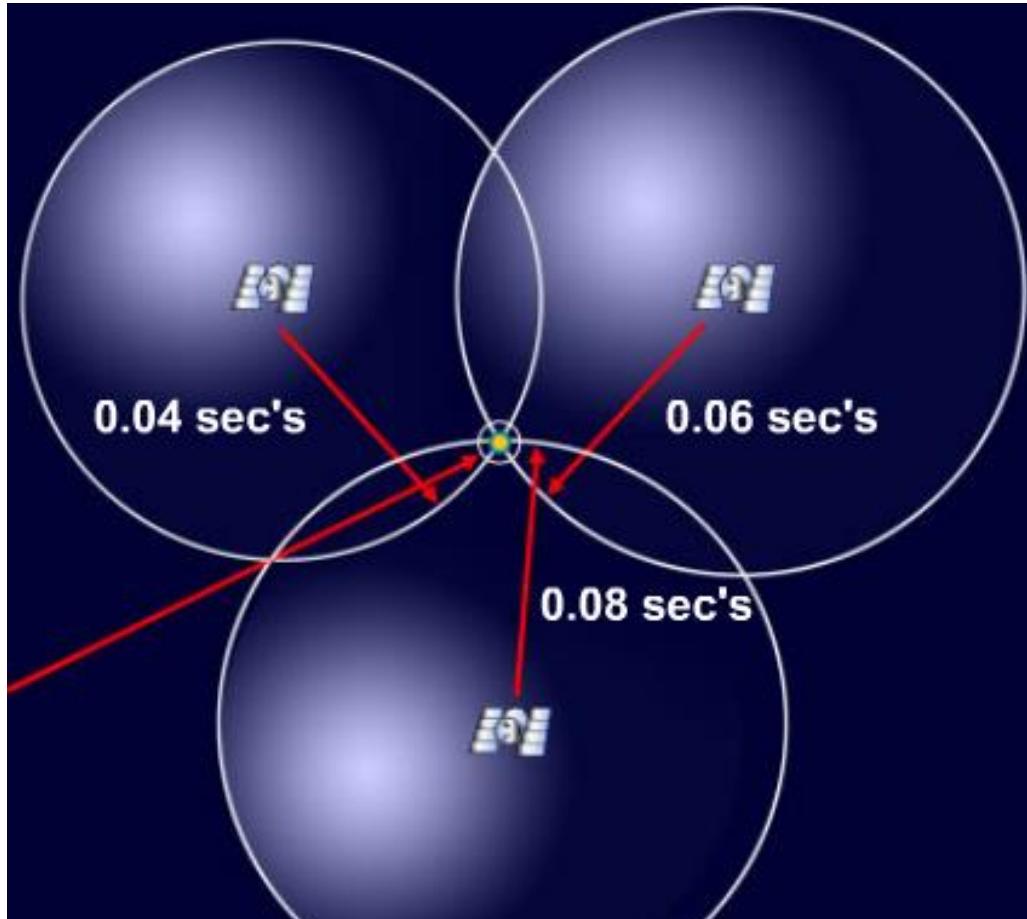
Introduction to GPS, Peter Gibbons



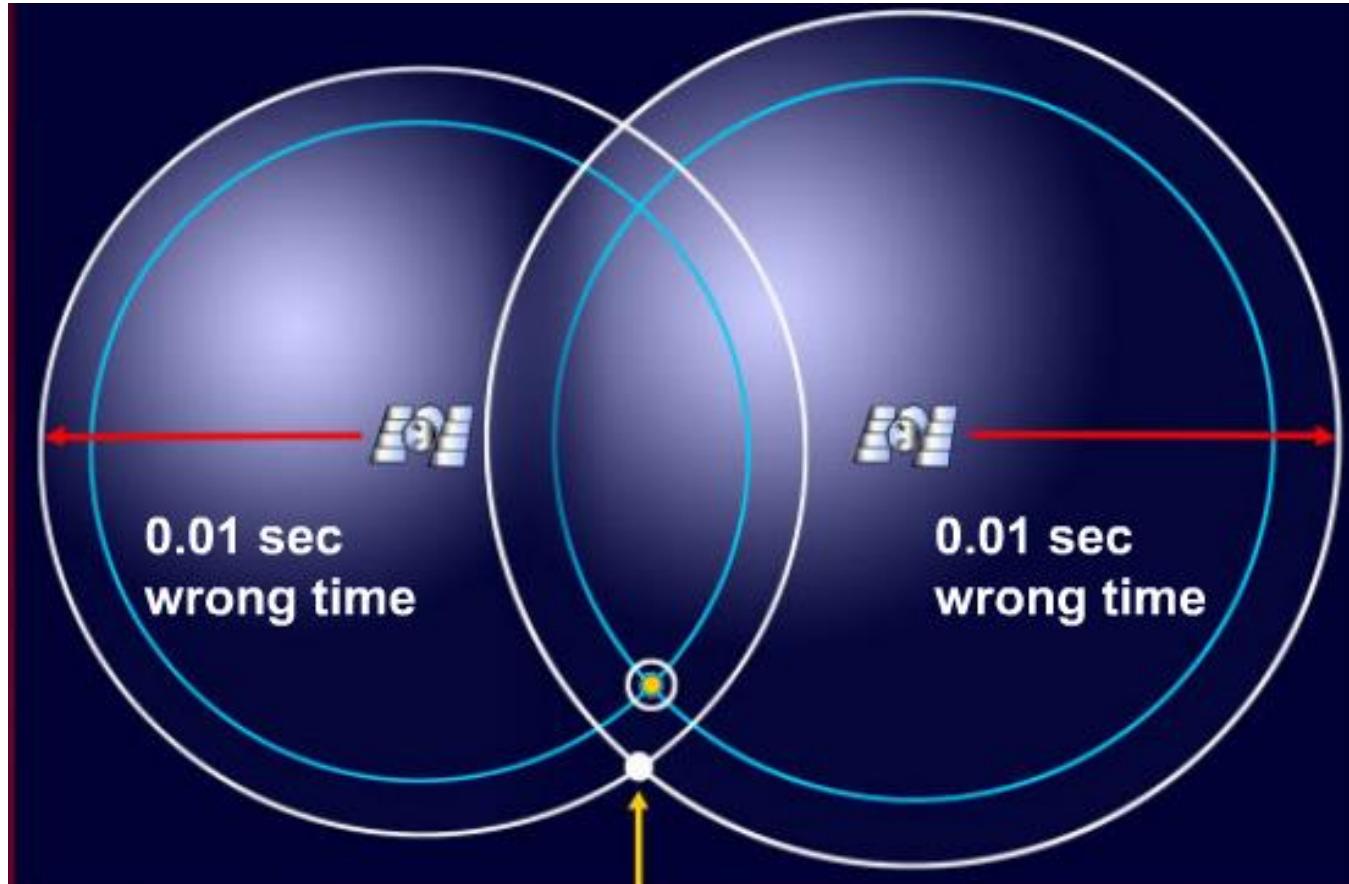
Step 3: Accurate Clocks

- System depends on accurate clocks
- Satellites have atomic clocks
- Ground receivers need synchronized clocks

Three measurements
go through one
position in 2D



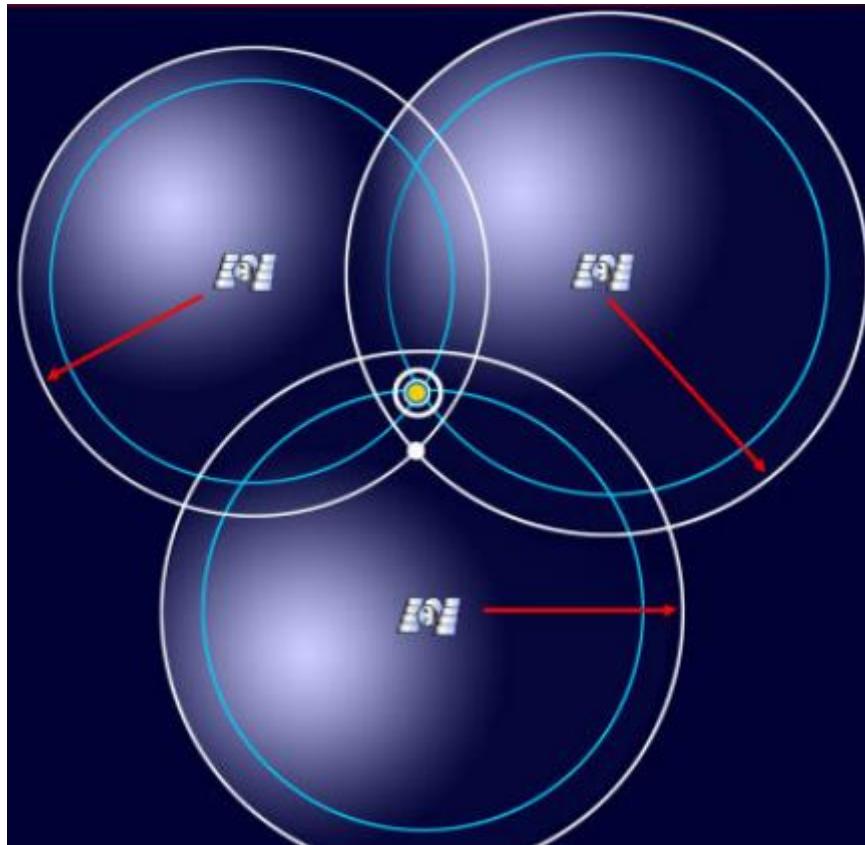
Introduction to GPS, Peter Gibbins



Introduction to GPS, Peter Gibbons

Bad position because clock is off by 0.01 seconds

Step 3: Fast Clock – Three Measurements



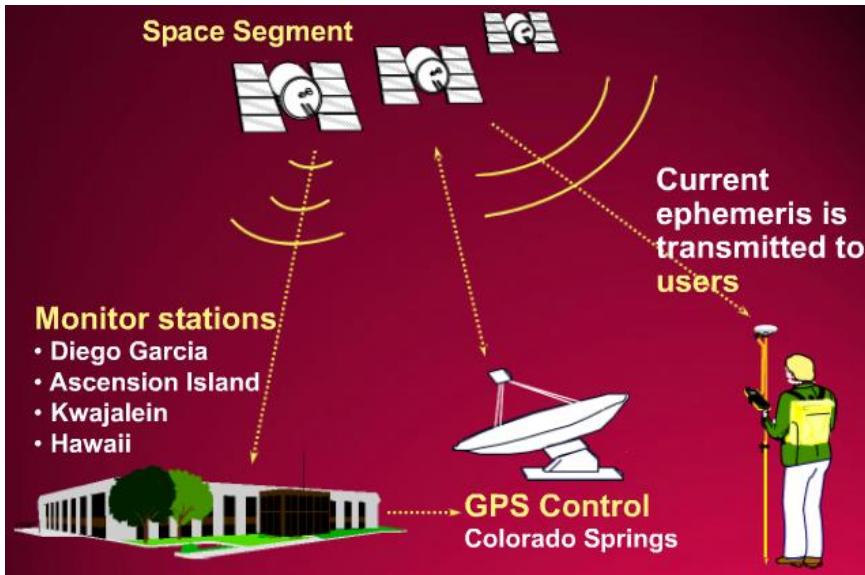
Introduction to GPS, Peter Gibbons

- Third measurement allows to determine the time offset (in 2D)
- Four measurements required in 3D (in 2D werden 3 Sensoren gebraucht)

Step 4: Knowing Where the Satellites Are?

Satellite Location:

- 20,000 km – high orbit
- Very stable orbit
- No atmospheric drag
- Location can be precisely predicted!
- Monitored and corrected by US DoD



Introduction to GPS, Peter Gibbins

Each satellite transmits...

Ephemeris:

- Every 30s
- Contains *own* precise location

Almanac:

- Every 12,5 min
- Contains coarse location information about *all* SVs

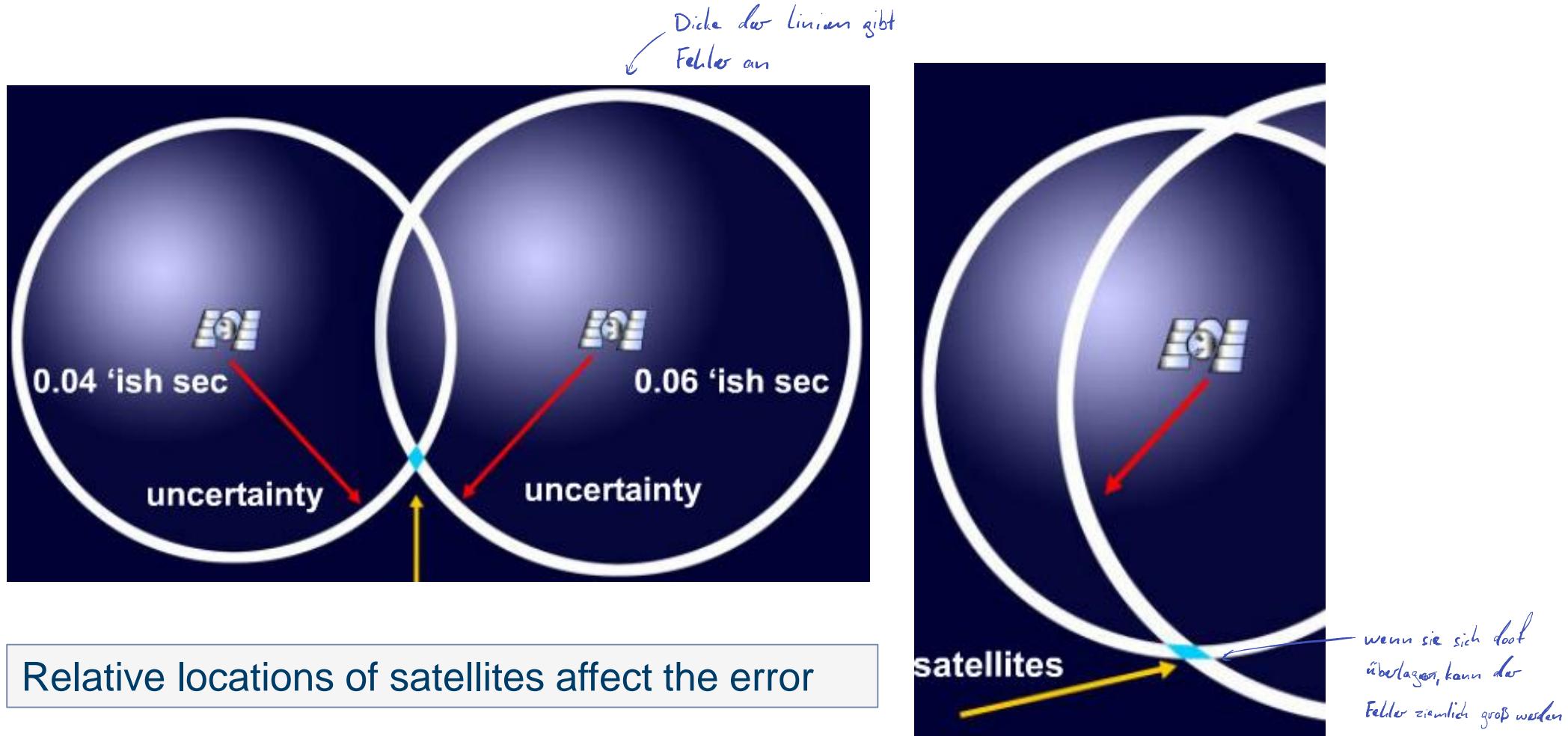
Step 5: Atmospheric Corrections

- Signals are delayed
- Ionosphere
 - 50km to 500km altitude
 - Ionized particles
- Troposphere
 - lower part of atmosphere
 - Full of water vapor
- Receiver factors in the angle of the signal entering
- Models take into account charged particles and varying gaseous
- Satellite constantly transmits updates of ionospheric model



Introduction to GPS, Peter Gibbings

• Satellite clock error	\pm 2 meter
• Receiver noise	\pm 0.5 meters
• Interference in ionosphere/troposphere	\pm 5 meters
• Multipath error	\pm 1.4 meters
• Satellite position (“ephemeris”) error	\pm 2 meters
• Poor geometric dilution of precision	up to 200 meters

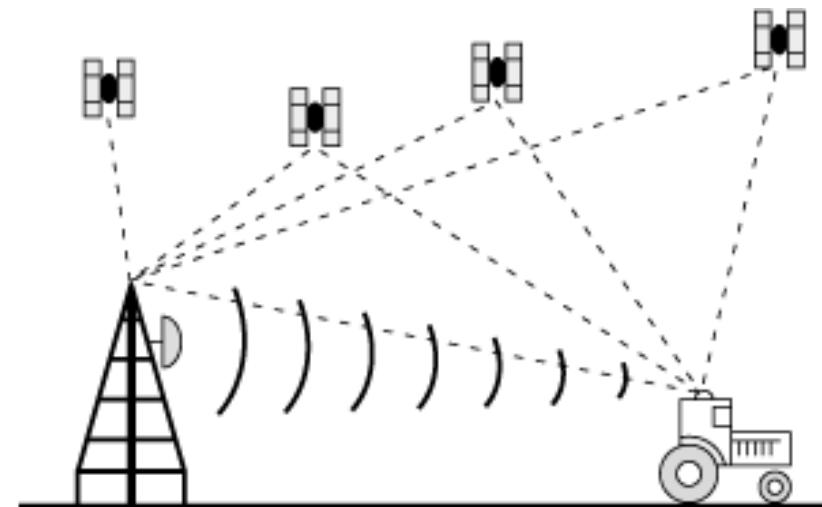


Introduction to GPS, Peter Gibbins

Most errors depend on the location of the receiver. Thus:

- A GPS receiver is located at a known location (base station)
- Base station calculates error by comparing true location with measured location
- Sends corrections to user receiver
→ Accuracy about 10cm under good conditions possible

http://de.wikipedia.org/wiki/Differential_Global_Positioning_System
<http://www2.ca.uky.edu/agc/pubs/pa/pa5/pa5.htm>



Differential GPS System

- **Desired:** Receiver location $x \in \mathbb{R}^2$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

- **Given:** Distances to m landmarks:

- Position $p_i = [p_{i,1}, p_{i,2}]^T \in \mathbb{R}^2$
- Distance $d_i \in \mathbb{R}$ to landmark i

- **Indiv. measurement equation:**

$$d_i = \|x - p_i\| + e_i + b$$

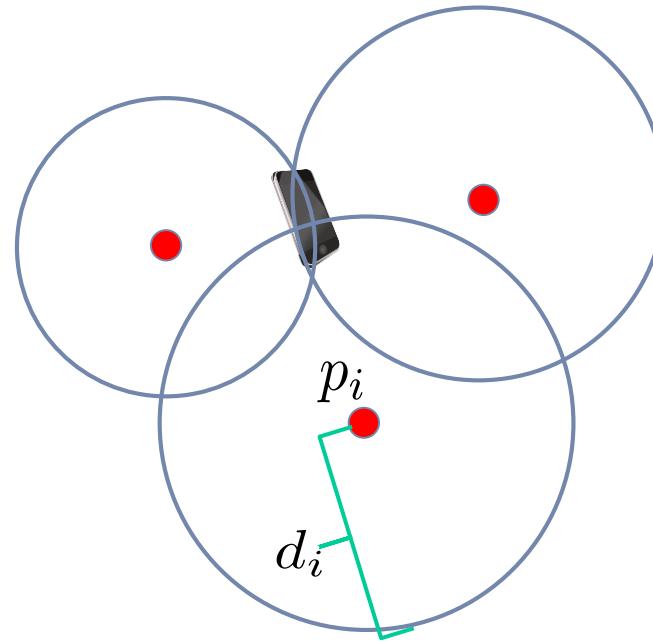
Fehler erstmals ignoriert
b ≡ bias

with measurement error $e_i \in \mathbb{R}$.

- **Stacked measurement equation:**

$$\begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix} = \underbrace{\begin{bmatrix} \|x - p_1\| \\ \vdots \\ \|x - p_m\| \end{bmatrix}}_{=: h(x)} + \begin{bmatrix} e_1 \\ \vdots \\ e_m \end{bmatrix}$$

*nonlinear Mapping
 Es kann nicht als Hx dargestellt werden*



Nonlinear Measurement Equation with Additive Noise:

$$y = \underbrace{h(x)}_{\text{Measurement Function of state } (x)} + e$$

Assumption: Overdetermined, $m > n$

Objective:

$$x^{LS} = \arg \min_x \|y - h(x)\|_{\mathbf{W}}^2$$

more measurements
than the state has

hypothesised measurement

semi positive

Iterative Optimization

(standard way)

↳ Gauß-Newton Algorithm

- Requires initial guess
- Number of iterations not known in advance
- Precise, however, may suffer from local minima

Direct (closed-form) solution

- Computationally efficient
- No initial guess required
- Might not be precise in high noise scenarios
- Does not suffer from local minima

Trilateration: Simple Closed-form Solution

=> Goal: use linear least squares:

- was wir brauchen: $y = Hx + e$
- was wir haben: $y = h(x) + e$

- Squared meas. equations ($i = 1, \dots, m$):

$$\text{erster Schritt: } d_1^2 = \|x - p_1\|^2$$

Quadratieren

$$d_2^2 = \|x - p_2\|^2$$

\vdots in first step ignore error
(quadratiert wird das sonst zu komplex)

$$d_m^2 = \|x - p_m\|^2$$

$$\begin{aligned} &= (x_1 - p_{1,1})^2 + (x_2 - p_{1,2})^2 = -2x_1 p_{1,1} - 2x_2 p_{1,2} + \|p_1\|^2 + \|x\|^2 \\ &= (x_1 - p_{2,1})^2 + (x_2 - p_{2,2})^2 = -2x_1 p_{2,1} - 2x_2 p_{2,2} + \|p_2\|^2 + \|x\|^2 \\ &\vdots \\ &= (x_1 - p_{m,1})^2 + (x_2 - p_{m,2})^2 = -2x_1 p_{m,1} - 2x_2 p_{m,2} + \|p_m\|^2 + \|x\|^2 \end{aligned}$$

Ist zwar quadratisch, ist aber
ohne x & daher konstant

\vdash durch Abziehen der
Zeile, verlieren wir
das $\|x\|^2$. Danach nur
noch lineares Problem

- Subtracting the last equation m from all others results in $m - 1$ linear equations:

$$\begin{aligned} d_i^2 - d_m^2 &= -2x_1 p_{i,1} - 2x_2 p_{i,2} + \|p_i\|^2 + \|x\|^2 + 2x_1 p_{m,1} + 2x_2 p_{m,2} - \|p_m\|^2 - \|x\|^2 \\ &= -2x_1 p_{i,1} + 2x_1 p_{m,1} - 2x_2 p_{i,2} + 2x_2 p_{m,2} + \|p_i\|^2 - \|p_m\|^2 \end{aligned}$$

\hookrightarrow jetzt möglich in Hx umzuwandeln

- Individual measurement equation:

$$d_i^2 - d_m^2 - \|p_i\|^2 + \|p_m\|^2 = -2x_1 p_{i,1} + 2x_1 p_{m,1} - 2x_2 p_{i,2} + 2x_2 p_{m,2}$$

- Stacked measurement equation:

$$y^* = \mathbf{H}^* x + e^*$$

$e^* \neq e$

with

$$y^* = \begin{bmatrix} d_1^2 - d_m^2 - \|p_1\|^2 + \|p_m\|^2 \\ \vdots \\ d_{m-1}^2 - d_m^2 - \|p_{m-1}\|^2 + \|p_m\|^2 \end{bmatrix}, \quad \mathbf{H}^* = 2 \begin{bmatrix} p_{m,1} - p_{1,1} & p_{m,2} - p_{1,2} \\ \vdots & \vdots \\ p_{m,1} - p_{m-1,1} & p_{m,2} - p_{m-1,2} \end{bmatrix}$$

⇒ Linear least squares solution for x

$$\|y - h(x)\|_w^2$$

\neq "only an approximation
 but with linear least squares we have a closed form solution"
 $\|y^* - H^* x\|_w^2$ ← Quadrat bezieht sich auf Betrag
 nicht auf weighting matrix

different weighting matrix
 because it has one less dimension

Trilateration: Intuitive Example

- Given: Measurements to $m = 2$ satellites:

- Distances $d_1, d_2 \in \mathbb{R}$ to two satellite
- Positions

$$p_1 = [0, 0]^T$$

$$p_2 = [r, 0]^T$$

- Measurement equation:

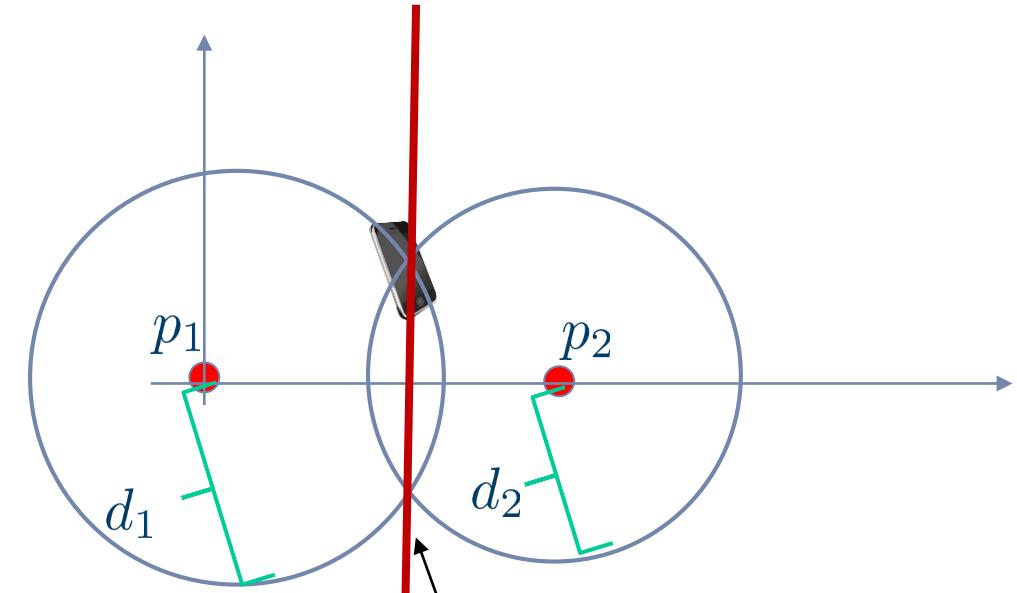
$$d_1^2 = x_1^2 + x_2^2$$

$$d_2^2 = (r - x_1)^2 + x_2^2$$

- Line of Positioning:

$$\begin{aligned} d_1^2 - d_2^2 &= x_1^2 + x_2^2 - (r - x_1)^2 - x_2^2 \\ &= -r^2 + 2rx_1 \end{aligned}$$

nach x_1 umgestellt



$$x_1 = \frac{d_1^2 - d_2^2 + r^2}{2r}$$

$$x_2 = \pm \sqrt{d_1^2 - x_1^2}$$

Bancroft: $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, b$

Intersection of Lines of Positioning (LOP)

- Disadvantage?
- Advantage?

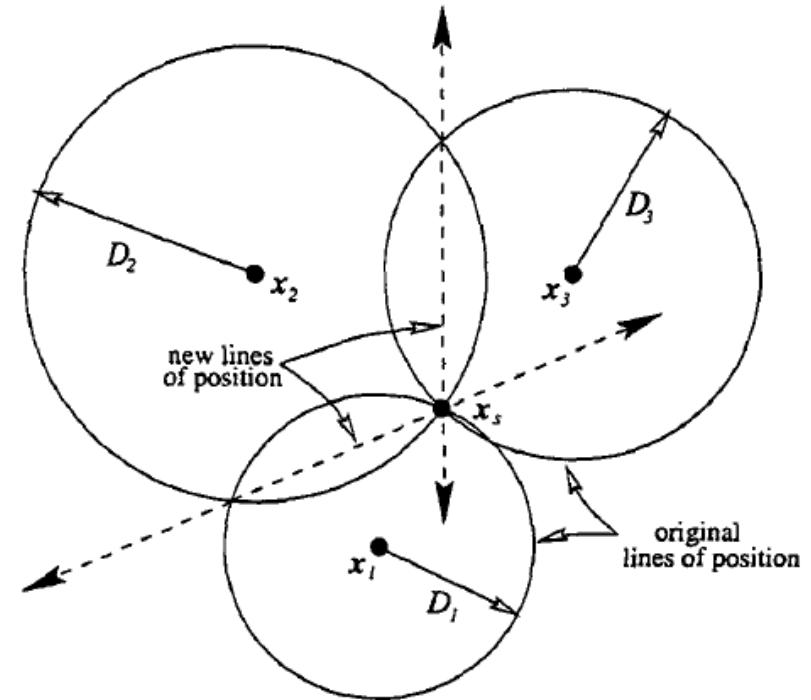


Figure 1. Geometry of TOA-based location showing circular LOPs and linear LOPs.

N. Sirola, "Closed-form algorithms in mobile positioning: Myths and misconceptions," Positioning Navigation and Communication (WPNC), 2010 7th Workshop on, Dresden, 2010, pp. 38-44.

J. J. Caffery, "A new approach to the geometry of TOA location," Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000.

Drone to Phone: Victim Localization using RSS measurements



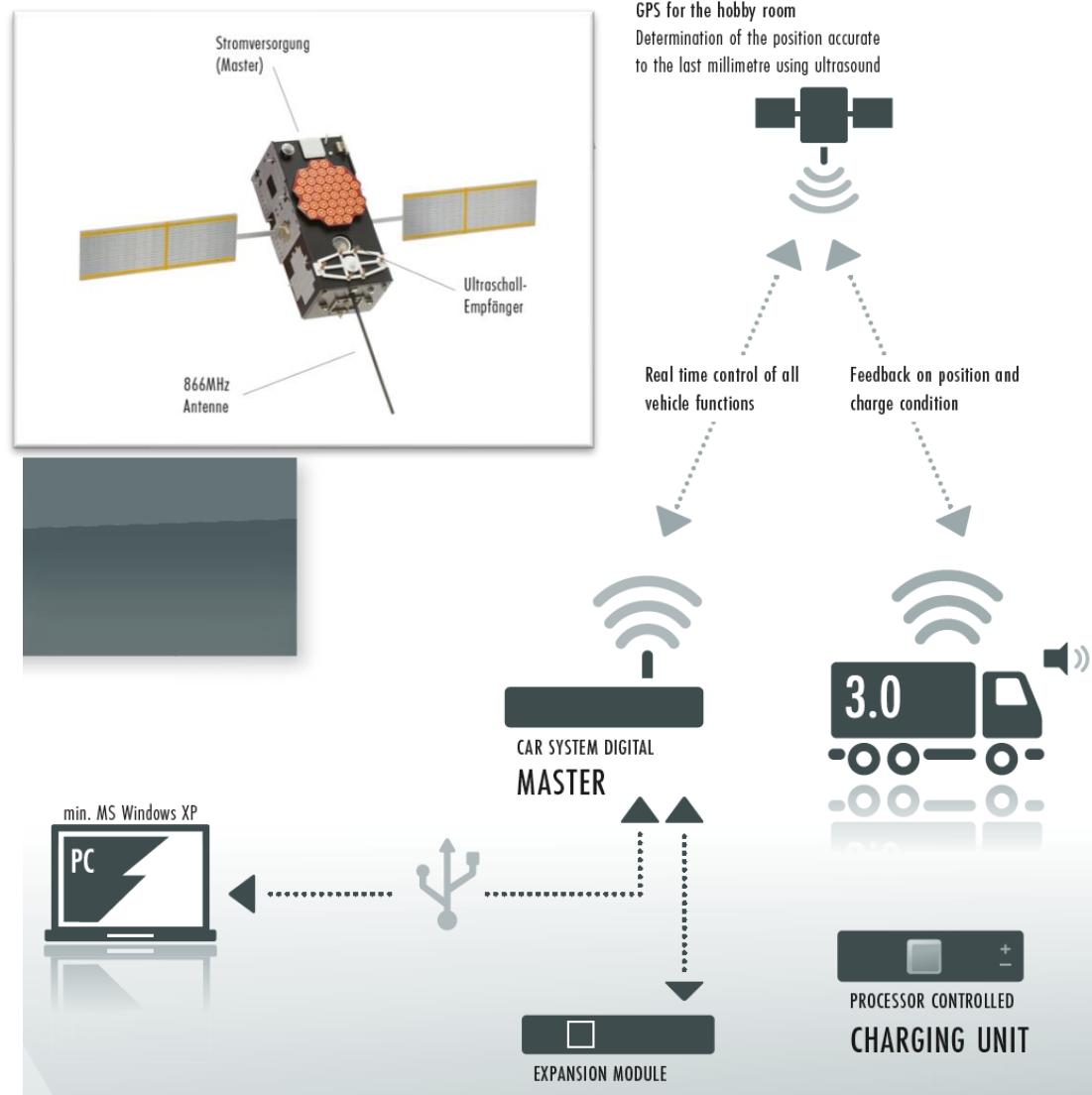
Victim Localization using RF-signals and Multiple Agents in Search & Rescue, by Jonas Ekskog & Jacob Sundqvist at Linköpings University and collaboration with Combitech AB.

Vehicle Localization

- Vehicles send out ultrasonic wave
- Distance to satellites
- 10mm accuracy
- 1mm accuracy with software



<https://www.car-system-digital.de>



Questions?

→ Stud.IP, e-mail or live session

Sensor Data Fusion

Exercise 2

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework - Distance measurements to walls
- Problem - Normal equation
- Example - Camera
- Homework - GPS

What are the assumptions for least squares?

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

$$\mathbf{H} \in \mathbb{R}^{m \times n}, \quad m \geq n, \quad \text{rank}(\mathbf{H}) = n$$

What is the objective/cost function of least squares?

objective: minimize cost function $G(x)$

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

$$\mathbf{x}^{LS} = \underset{\mathbf{x}}{\operatorname{argmin}} \underbrace{\|\mathbf{y} - \mathbf{Hx}\|_{\mathbf{W}}^2}_{G(\mathbf{x})}$$

$$G(\mathbf{x}) = (\mathbf{y} - \mathbf{Hx})^T \mathbf{W} (\mathbf{y} - \mathbf{Hx})$$

$$G'(\mathbf{x}^{LS}) \stackrel{!}{=} 0$$

$$\mathbf{H}^T \mathbf{W} \mathbf{y} = \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{x}^{LS}$$

$$\mathbf{x}^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}$$

(alte Folie...)

What is the objective/cost function of least squares?

$$y = Hx + e \quad \text{weighted squared error (should be minimized)}$$

nach e umgestellt $\rightarrow x^{LS} = \underset{x}{\operatorname{argmin}} \underbrace{\|y - Hx\|_W^2}_{G(x)}$

$$G(x) = (y - Hx)^T W (y - Hx) \quad \text{cost function}$$

$$G'(x^{LS}) \stackrel{!}{=} 0 \quad (\text{quadrierte Fehler soll minimal sein})$$

$$H^T W y = H^T W H x^{LS}$$

$$x^{LS} = (H^T W H)^{-1} H^T W y$$

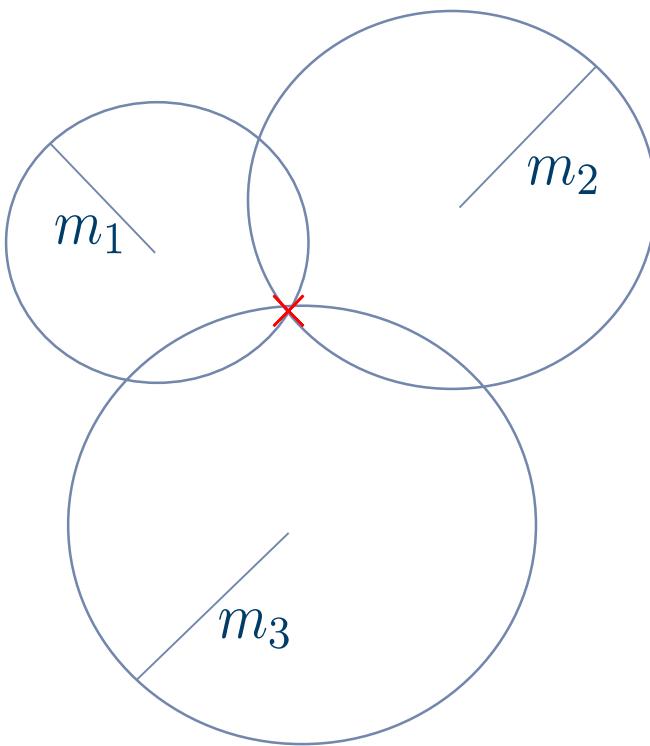
- a) How does the cost function of Least Squares change for the non-linear case?
- b) How could we use linear Least Squares to solve it?

a)

$$\mathbf{y} = h(\mathbf{x}) + \mathbf{e}$$
$$\mathbf{x}^{LS} = \underset{\mathbf{x}}{\operatorname{argmin}} \underbrace{\|\mathbf{y} - h(\mathbf{x})\|_{\mathbf{w}}^2}_{G(\mathbf{x})}$$

b) Approximate the problem into a linear form, i.e., get $y^* = \mathbf{H}^*x + e^*$, but remember that the error that is minimized is different now, i.e., $\|\mathbf{y} - h(\mathbf{x})\|^2 \neq \|y^* - \mathbf{H}^*x\|^2$.

How does trilateration work?



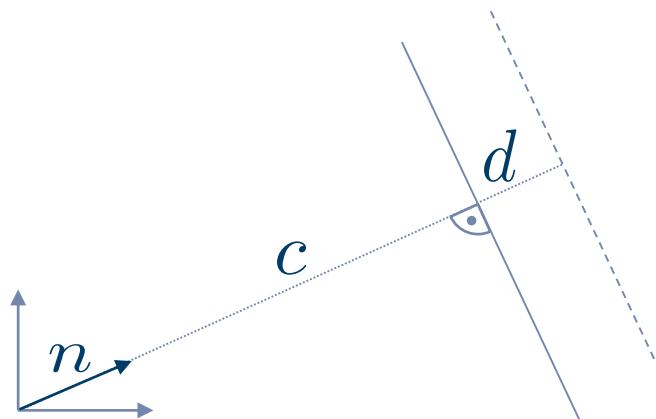
A wall in 2D space can be described using a normal vector $n = [n_1 \ n_2]^T$ (with $\|n\|_2 = 1$) and a scalar c

$$n_1 x_1 + n_2 x_2 = c$$

which holds true for all $x = [x_1 \ x_2]^T$ on the wall. Note that c in this case describes the shortest distance from the origin to the wall. So any shift d of c would describe all points within a distance d of the wall (see Figure below). So the formula

$$n_1 x_1 + n_2 x_2 = c + d$$

holds true for all x with distance d to the wall in the direction of n .



If the given vector \hat{n} is not a normal vector, so we have

$$\hat{n}_1 x_1 + \hat{n}_2 x_2 = \hat{c}$$

with $\|\hat{n}\|_2 \neq 1$, we need to normalize it first. This would result in the distance formula being

$$\frac{\hat{n}_1 x_1 + \hat{n}_2 x_2 - \hat{c}}{\|\hat{n}\|_2} = d$$

The objective is to estimate the two-dimensional object location $x = [x_1, x_2]^T \in \mathbb{R}^2$ using (noise-corrupted) distance measurements $d^i \in \mathbb{R}$ to N walls. The location of the i -th wall is given in normal form

$$n_1^i \cdot x_1^w + n_2^i \cdot x_2^w = c^i .$$

Assume n^i points to the half space where the object is located. Given are four walls with corresponding measurements:

i	n_1^i	n_2^i	c^i	distance d^i
1	-5	-1	-45	4.7
2	-1	-8	-70	5.2
3	-1	9	5	5.5
4	8	-1	7	4.5

- a) Please write a function which visualizes walls and measurements using different colors.

see notebook for solution

b) Formulate a linear measurement equation $y^i = \mathbf{H}^i x + e^i$, which relates the measurement to the i -th wall with x and the error e^i . In the same manner, formulate a measurement equation that relates N walls, i.e., the 1-st to N -th walls, with x and e . Note that the n^i vectors are not normalized.

$$d^i = \frac{n_1^i x_1 + n_2^i x_2 - c^i}{\sqrt{(n_1^i)^2 + (n_2^i)^2}} + e^i$$

The distance d^i to wall i

$$d^i \sqrt{(n_1^i)^2 + (n_2^i)^2} + c^i = n_1^i x_1 + n_2^i x_2 + e^i$$

$$\underbrace{d^i \sqrt{(n_1^i)^2 + (n_2^i)^2} + c^i}_{y^i} = \underbrace{\begin{bmatrix} n_1 & n_2 \end{bmatrix}}_{\mathbf{H}^i} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x + e^i$$

Distances to N walls

$$\underbrace{\begin{bmatrix} d^1 \sqrt{(n_1^1)^2 + (n_2^1)^2} + c^1 \\ \vdots \\ d^i \sqrt{(n_1^i)^2 + (n_2^i)^2} + c^i \\ \vdots \\ d^N \sqrt{(n_1^N)^2 + (n_2^N)^2} + c^N \end{bmatrix}}_y = \underbrace{\begin{bmatrix} n_1^1 & n_2^1 \\ \vdots & \vdots \\ n_1^i & n_2^i \\ \vdots & \vdots \\ n_1^N & n_2^N \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_x + \underbrace{\begin{bmatrix} e^1 \\ \vdots \\ e^i \\ \vdots \\ e^N \end{bmatrix}}_e$$

c) Could you calculate the unique location for the first case in 1b), if not, please explain. If an unique location could be obtained, which requirements are needed?

It is not possible to get an exact solution because we have 2 unknowns and only 1 formula. The exact solution can be calculated if $\text{rank}(\mathbf{H}) = 2$.

d) Based on the measurement equation formulated in 1b), write a function which calculates the least squares solutions based on the measurements. Using the function you implemented, calculate the least squares solutions $\hat{\mathbf{x}}_{12}$, $\hat{\mathbf{x}}_{34}$ and $\hat{\mathbf{x}}_{1234}$ based on the measurements (y_1, y_2) , (y_3, y_4) as well as (y_1, y_2, y_3, y_4) .

see notebook for solution

e) Given the true location \mathbf{x} , implement a function which calculates the estimation error e using Euclidean norm, e.g.,

$$e_{12} = \|\hat{\mathbf{x}}_{12} - \mathbf{x}\|, e_{34} = \|\hat{\mathbf{x}}_{34} - \mathbf{x}\|, e_{1234} = \|\hat{\mathbf{x}}_{1234} - \mathbf{x}\|$$

Assuming the true location $\mathbf{x} = [5, 5]^T$, calculate e_{12} , e_{34} and e_{1234} . What can you observe?

see notebook for solution

Consider a linear measurement equation

$$y = \mathbf{H}x + e$$

with measurement vector $y \in \mathbb{R}^3$, state vector $x \in \mathbb{R}^2$, error vector $e \in \mathbb{R}^3$ and measurement matrix

$$\mathbf{H} \in \mathbb{R}^{3 \times 2}, \text{ where } \mathbf{H} = \begin{pmatrix} 2 & 0 \\ 2 & 0 \\ 0 & 2 \end{pmatrix} \text{ and } y = \begin{pmatrix} 1 \\ 2 \\ 2 \end{pmatrix}.$$

- Visualize the column space spanned by \mathbf{H} using a three-dimensional plot.
- Illustrate the least squares estimate and the error vector in the three-dimensional plot. What is the meaning of the normal equation?
- Now assume $\mathbf{H} = \begin{pmatrix} 2 & 3 \\ 2 & 3 \\ 0 & 0 \end{pmatrix}$. Can you still calculate the LS estimate?

Solution – Normal Equation

$$\mathbf{H} = \begin{bmatrix} 2 & 0 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Weight Matrix kann weg-
gelassen werden, wird ange-
nommen, dass es die
Identity Matrix ist

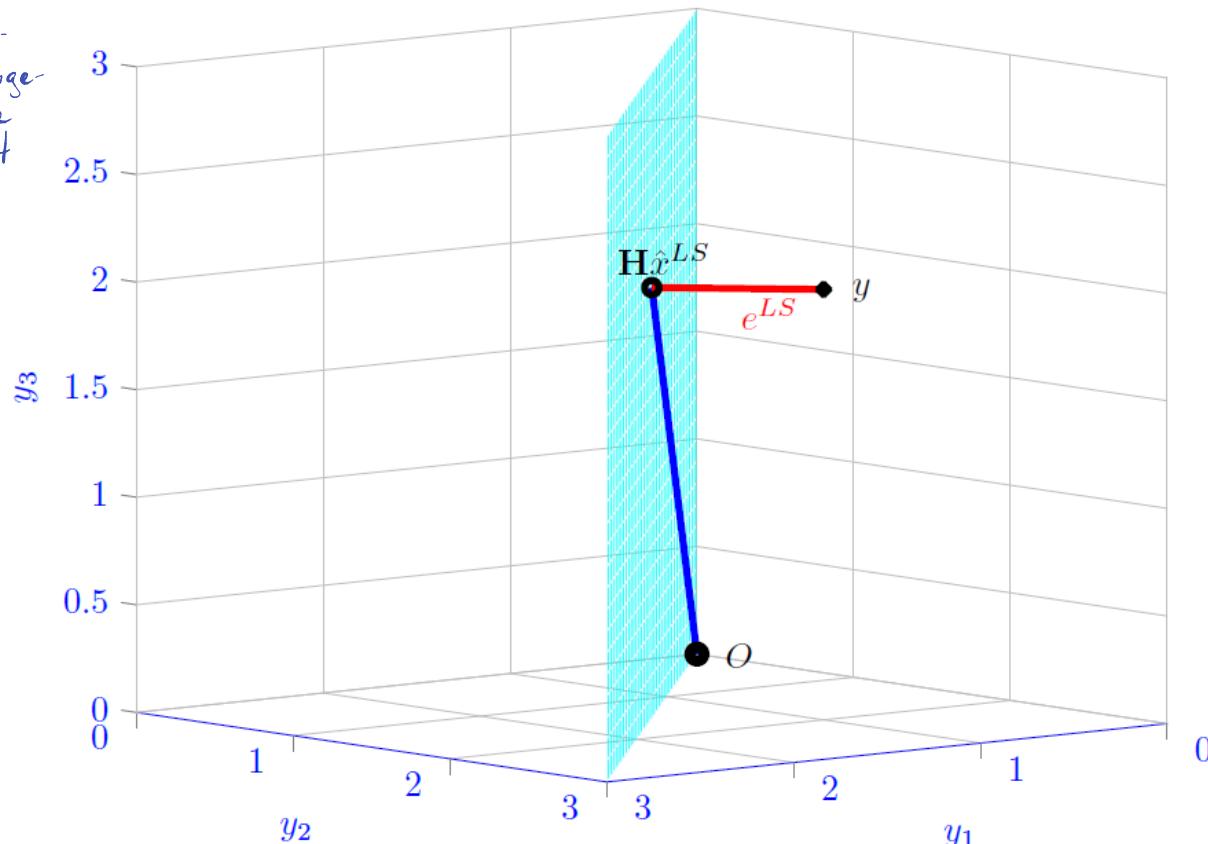
$$\hat{\mathbf{x}}^{\text{LS}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{y} = \begin{bmatrix} \frac{1}{8} & 0 \\ 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 6 \\ 4 \end{bmatrix} = \begin{bmatrix} 0.75 \\ 1 \end{bmatrix}$$

$$y^{\text{proj}} = \mathbf{H} \hat{\mathbf{x}}^{\text{LS}} = \begin{bmatrix} 1.5 \\ 1.5 \\ 2 \end{bmatrix}$$

$$e^{\text{LS}} = \mathbf{y} - \mathbf{H} \hat{\mathbf{x}}^{\text{LS}} = \begin{bmatrix} -0.5 \\ 0.5 \\ 0 \end{bmatrix}$$

$$c_1^T e^{\text{LS}} = 0$$

$$c_2^T e^{\text{LS}} = 0$$

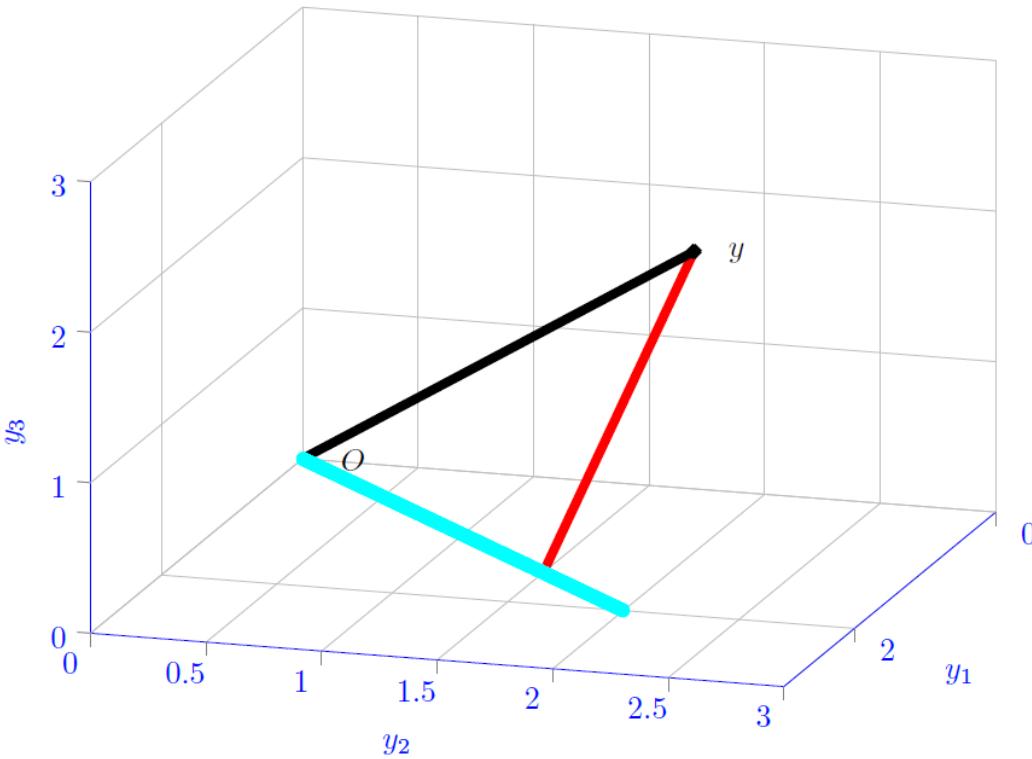


Anmerkung: $\text{Rank} \begin{pmatrix} 2 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 2 & 0 \end{pmatrix} = 1$

$$y^{\text{proj}} = x_1 \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} 3 \\ 3 \\ 0 \end{bmatrix}$$

No unique solution, but set one element of x to 0 and calculate the other using only part of \mathbf{H} .

$$\begin{aligned} y &= \hat{\mathbf{H}}x_1 + e_1 \\ \hat{\mathbf{H}} &= \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix} \\ x_1^{\text{LS}} &= (\hat{\mathbf{H}}^T \hat{\mathbf{H}})^{-1} \hat{\mathbf{H}}^T y = 0.75 \\ x_2 &= 0 \end{aligned}$$



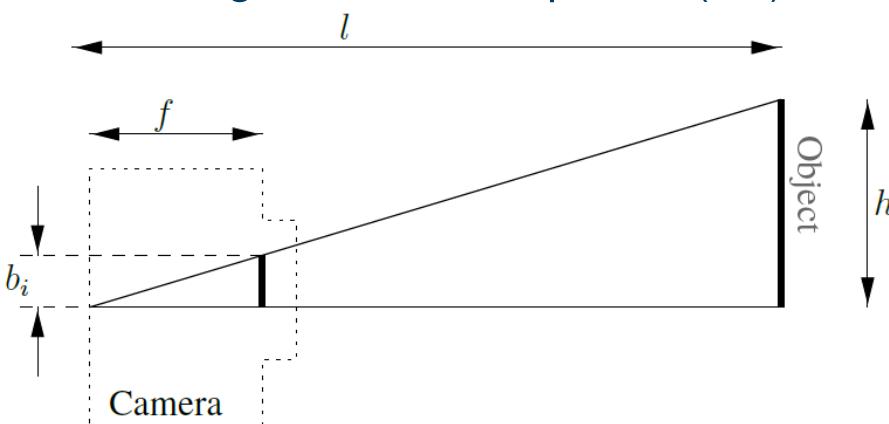
The objective is to determine the distance l from a camera to a planar object. For this purpose, features with height h_i are mapped according to

$$b_i = \frac{f}{l} h_i$$

to the image b_i , where $f = 4\text{mm}$ is the focal length. The pixel size is $2\mu\text{m}$. The following heights are measured

Feature height (cm)	1	5	15	25	50
Number of pixels	4	12	31	49	98

- Compute the distance l with an unweighted Least Squares (LS) estimator.



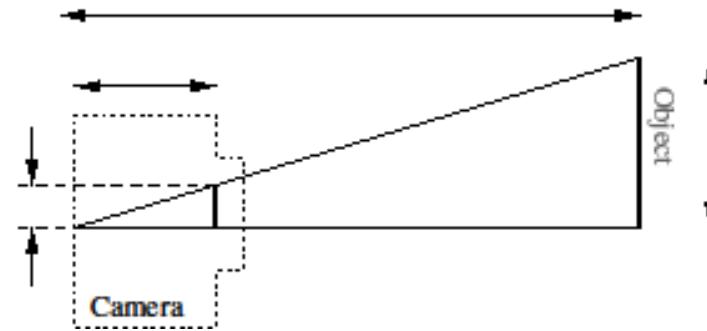
Given:

$$h = \begin{bmatrix} 1 \\ 5 \\ 15 \\ 25 \\ 50 \end{bmatrix} \times 1 \times 10^{-2} m, \quad b = \begin{bmatrix} 4 \\ 12 \\ 31 \\ 49 \\ 98 \end{bmatrix} \times 2 \times 10^{-6} m, \text{ and } f = 4 \times 10^{-3} m$$

$$b_i = \frac{f}{l} h_i$$

Want: l

$$\underbrace{h}_y = \underbrace{\frac{1}{f} b}_H \underbrace{l}_x + e$$



$$x^{\text{LS}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T y$$

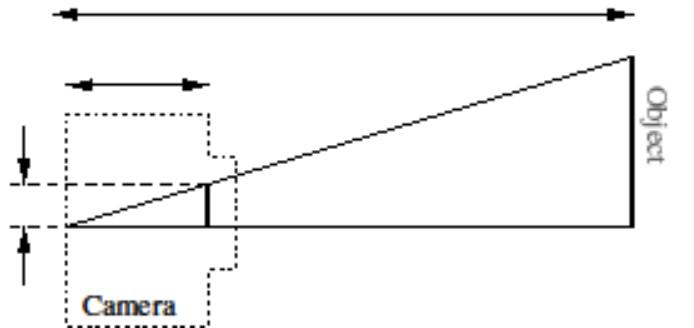
$$x^{\text{LS}} = \left(\left(\frac{2 \times 10^{-6}}{4 \times 10^{-3}} \right)^2 [4 \quad 12 \quad 31 \quad 49 \quad 98] \begin{bmatrix} 4 \\ 12 \\ 31 \\ 49 \\ 98 \end{bmatrix} \right)^{-1} \frac{2 \times 10^{-8}}{4 \times 10^{-3}} [4 \quad 12 \quad 31 \quad 49 \quad 98] \begin{bmatrix} 1 \\ 5 \\ 15 \\ 25 \\ 50 \end{bmatrix}$$

$$x^{\text{LS}} = \left(\left(\frac{10^{-3}}{2} \right)^2 (16 + 144 + 961 + 2401 + 9604) \right)^{-1} \frac{10^{-5}}{2} (4 + 60 + 465 + 1225 + 4900)$$

$$x^{\text{LS}} = \left(\frac{13126}{4000000} \right)^{-1} \frac{6654}{200000}$$

$$x^{\text{LS}} = \frac{0.03327}{0.0032815}$$

$$x^{\text{LS}} \approx 10$$



GPS consists of 24 satellites in orbit (20200km above mean sea level). Each satellite broadcasts its location (in spherical coordinates $[\theta, \phi, r]^T$) plus the emission time (see figures below). A GPS device receives at time $t = 0\text{s}$ the following four satellite signals:

$$p_1 = [0^\circ, 40^\circ, 20200\text{km}]^T,$$

$$p_2 = [10^\circ, 20^\circ, 20200\text{km}]^T,$$

$$p_3 = [10^\circ, -10^\circ, 20200\text{km}]^T,$$

$$p_4 = [-10^\circ, -20^\circ, 20200\text{km}]^T,$$

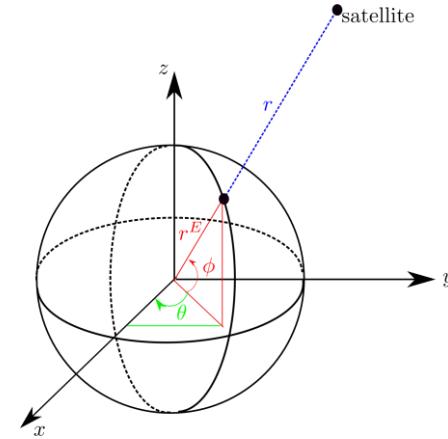
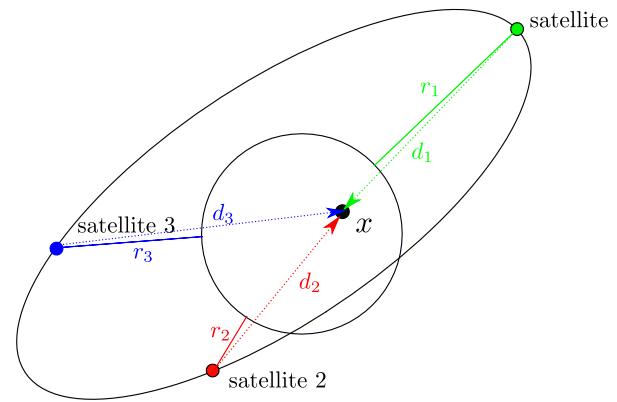
$$t_1 = -67.603\text{ms},$$

$$t_2 = -70.102\text{ms},$$

$$t_3 = -78.690\text{ms},$$

$$t_4 = -82.942\text{ms}.$$

Assume that the speed-of-light is $c = 3 \cdot 10^8 \text{m s}^{-1}$ and the Earth is an ideal sphere with radius $r^E = 6370\text{km}$.



a) Write a function which converts sphere coordinates (θ, ϕ, r) into Cartesian coordinates (x, y, z)

Hint:

The x coordinates could be obtained according to $x = (r^E + r) \cos \phi \cos \theta$;

You could use `math.radians(...)` to convert degrees into radians.

Using the function you implemented, calculate the Cartesian coordinates of these four satellites.

b) Please calculate the distance between satellites and GPS device.

Using the distance measurements, form the measurement equation like we did in the lecture.

c) Reformulate the non-linear measurement equation in 2b) into a linear measurement equation and calculate the least squares estimate.

d) Write a function which converts the Cartesian coordinates into sphere coordinates. Use the function you implemented, calculate the longitude and latitude of the GPS device, and find out where it is using using *GPS Visualizer*:

<http://www.gpsvisualizer.com/map?form=google>

For this purpose, use the format

type,latitude,longitude
W,PHI,THETA

where PHI and THETA are the calculated coordinates in degrees.

Sensor Data Fusion

Nonlinear Least Squares: Optimization

www.fusion.informatik.uni-goettingen.de



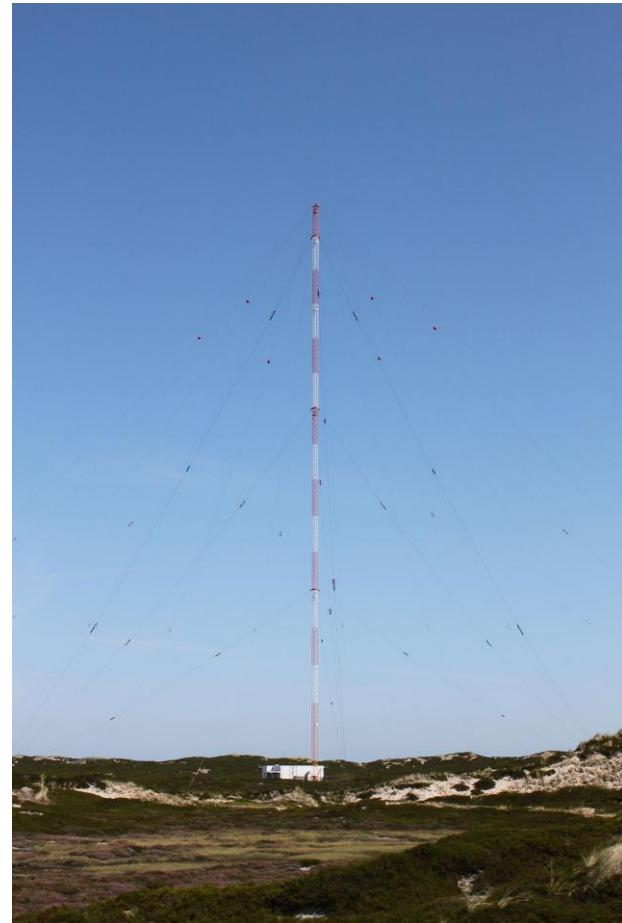
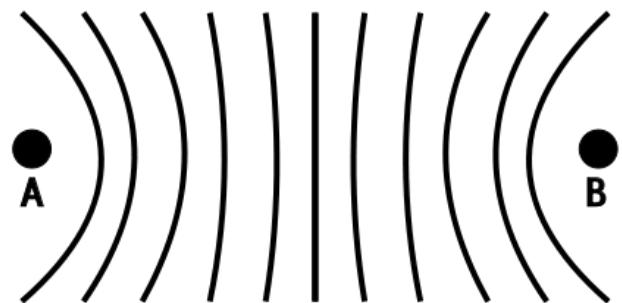
GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab

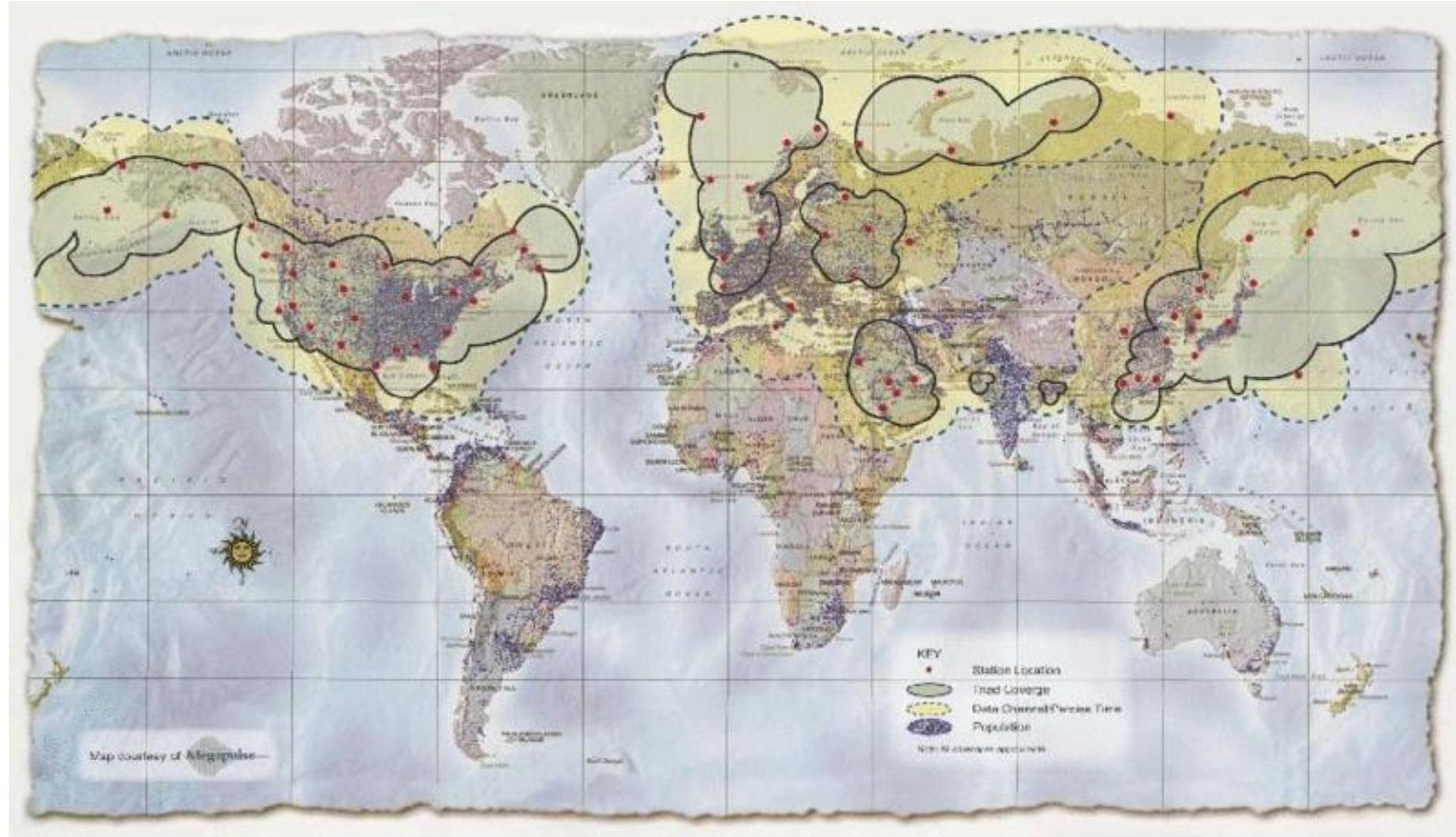
1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Dead reckoning
12. Dynamic models: Tracking (**live**)
13. Advanced Topic
14. Summary and Discussion (**live**)

- Localization based on distance differences
- Obtained from Time-Difference-of-Arrival (TDoA) measurements
- Cooperative surveillance (tracking):
Target object transmits signal to multiple receivers with synchronized clocks and known location
- Navigation:
Multiple synchronized senders transmit to target receiver
- Also known as hyperbolic localization

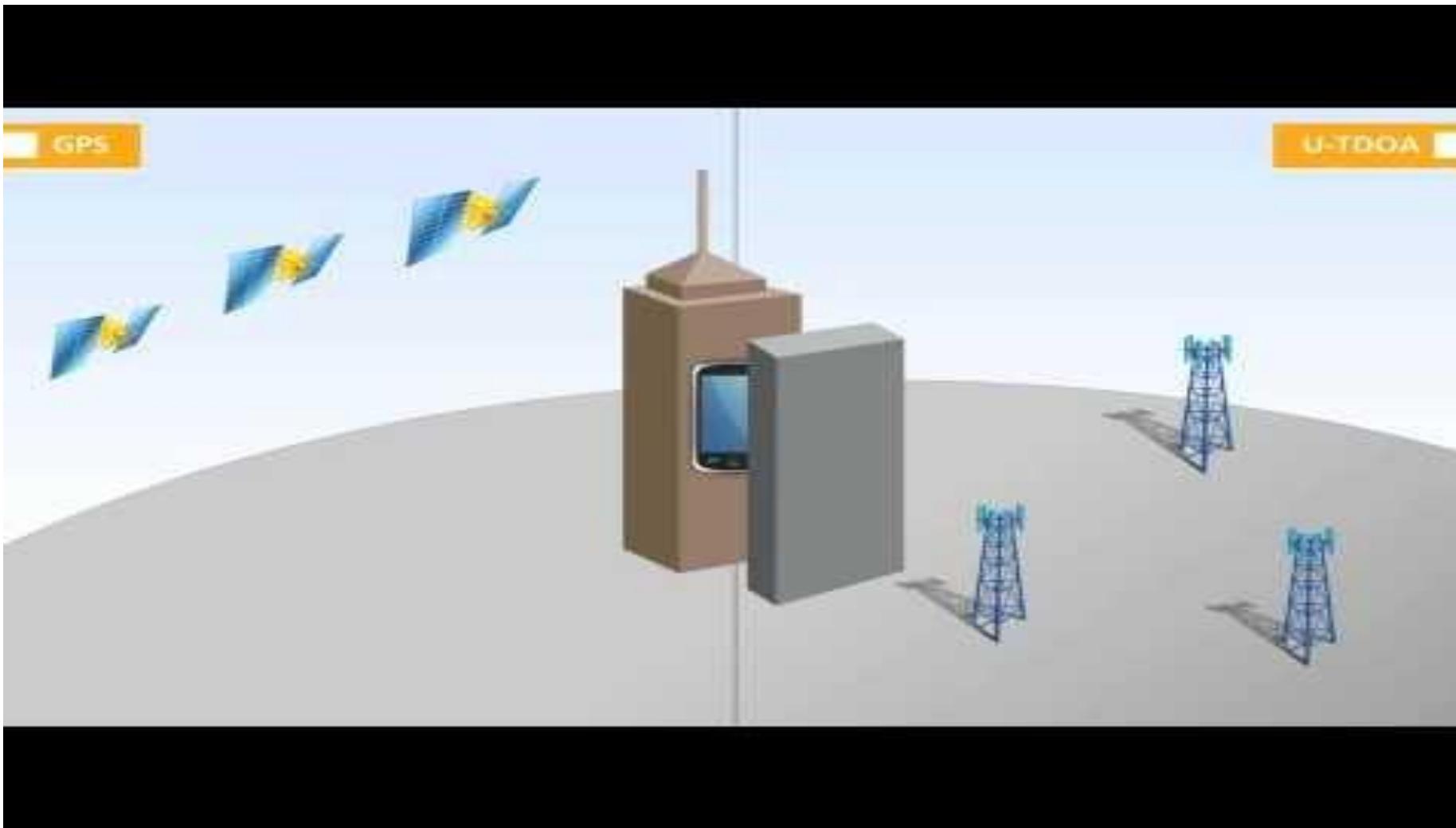
- First continuous operating LOng RAnge Navigation system
- Pulsed transmission
- Hyperbolic localization
- TDoA of signals from base stations
- Since 1958 developed by the US
- Accuracy ca. 0.25 to 1 mile
- Repeated measurements ca. 18-90 m
- Widely-used for maritime navigation
- Base station: 625'+ tall towers



Rantum, Sylt



Reference: Megapulse



<https://www.youtube.com/watch?v=MY9INdwj7tU>

Desired: Receiver position $x = [x_1, x_2]^T \in \mathbb{R}^2$

"consider un synchronized satellites at
known positions p_i "

Given:

- position $p_i = [p_{i,1}, p_{i,2}]^T \in \mathbb{R}^2$
- $\binom{m}{2}$ Time-Difference-of-Arrival (TDoA) measurements

$$\Delta T_{i,j} = t_i - t_j$$

$$= t_i + b - (t_j + b)$$

with time-of-arrivals t_i and t_j

unknown clock bias (major advantage)

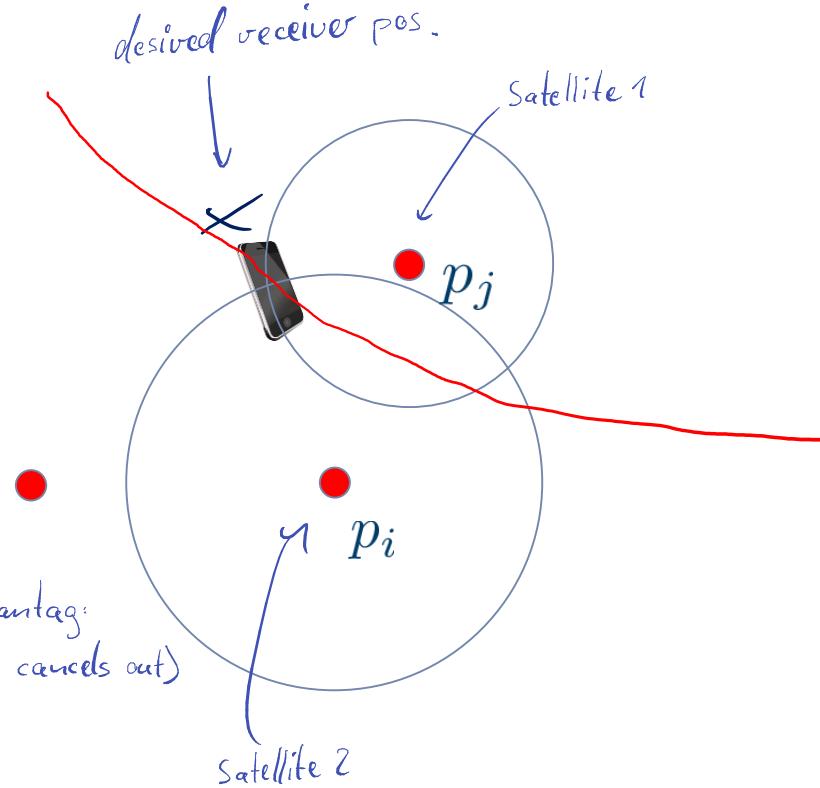
- distance difference $d_{i,j} = c \cdot \Delta T_{i,j}$ with wave speed c

with $i, j \in \{1, \dots, m\}$

Measurement equation: (non linear due to the norm operator, but closed-form expressions available)

$$d_{i,j} = \|x - p_i\| - \|x - p_j\| + e_i$$

↳ results in a hyperbola (red line in fig.)



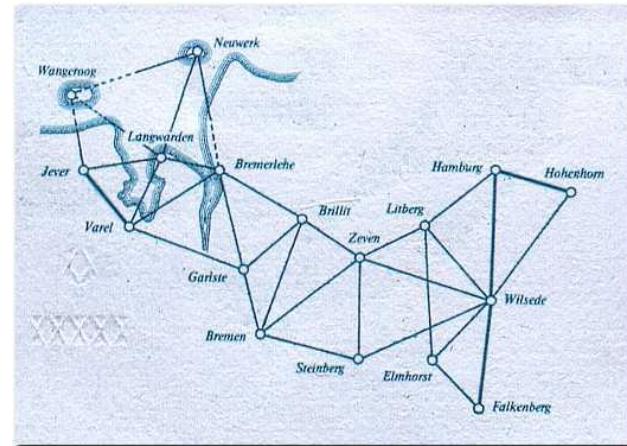
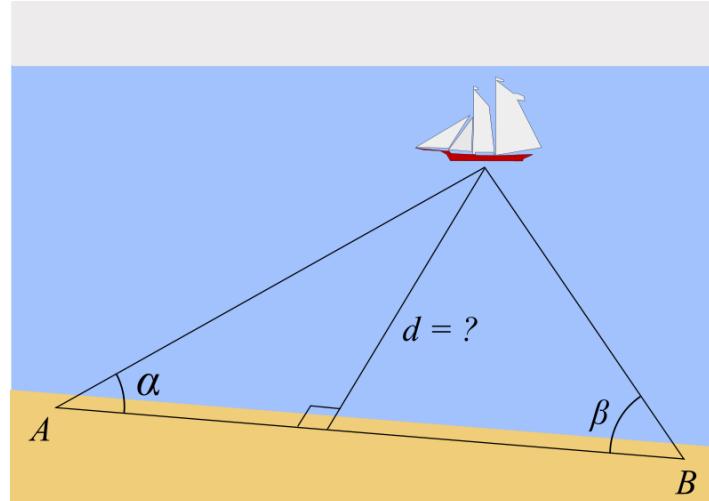
Time Diffs: calculate out Bias
from unsynchronous clocks
(keine abgestimmten Uhren nötig)

Triangulation

Process of determining the location with angle measurements

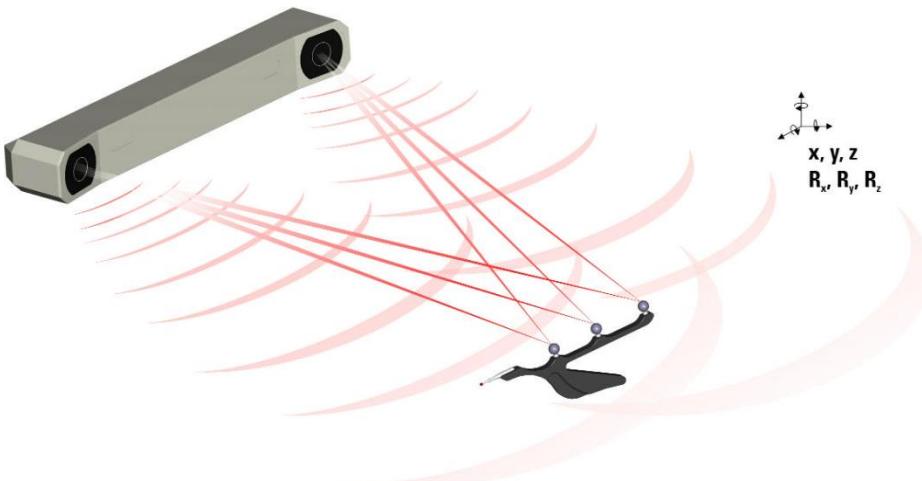
History

- Goes back to Thales, 6th century
- Later used for map making (geodesy)
- Carl Friedrich Gauss performed a triangulation of the kingdom of Hanover, 1821-1825. For this purpose he developed linear least squares.
- Large-scale triangulation networks are outperformed by GNSS
- Triangulation is, however, still heavily used in other applications



Parts of the triangulation network created by Gauss, scan from a 10DM note

- Localization of instruments, e.g., for computer-assisted surgery
- Position sensor sends out infrared signal
- Passive markers reflect infrared signal
- Position sensor detects reflections
- Triangulation gives pose



Source: www.ndigital.com

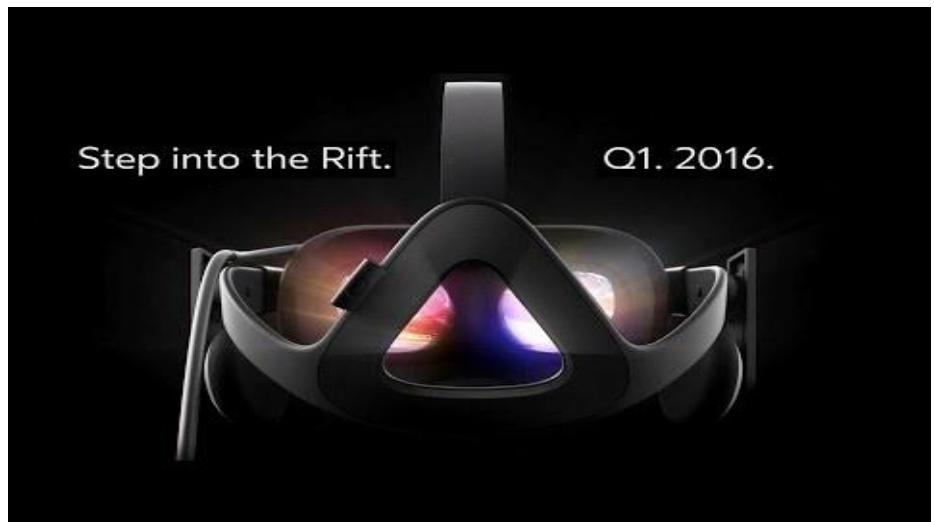
- Inside-Out:
Camera is mounted on target object
- HiBall consists of 6 infrared cameras
- Landmarks
 - Array of Infrared-LEDs
 - Mounted on ceiling
- Angular measurements are fused using a Kalman filter



Source: <http://www.3rdtech.com>

- Virtual reality head-mounted display (HMD)
- Developed by Oculus VR
- 1080x1200 pixels/eye
- 90 Hz refresh rate
- Pose tracking
 - Gyro, accelerometer, and magnetometer
 - LED on HMD
 - External infrared tracking sensor

<https://www.oculus.com/>



- **Desired:** Cartesian object position $x = [x_1, x_2]^T \in \mathbb{R}^2$

- **Given:** m angular measurements to m landmarks:

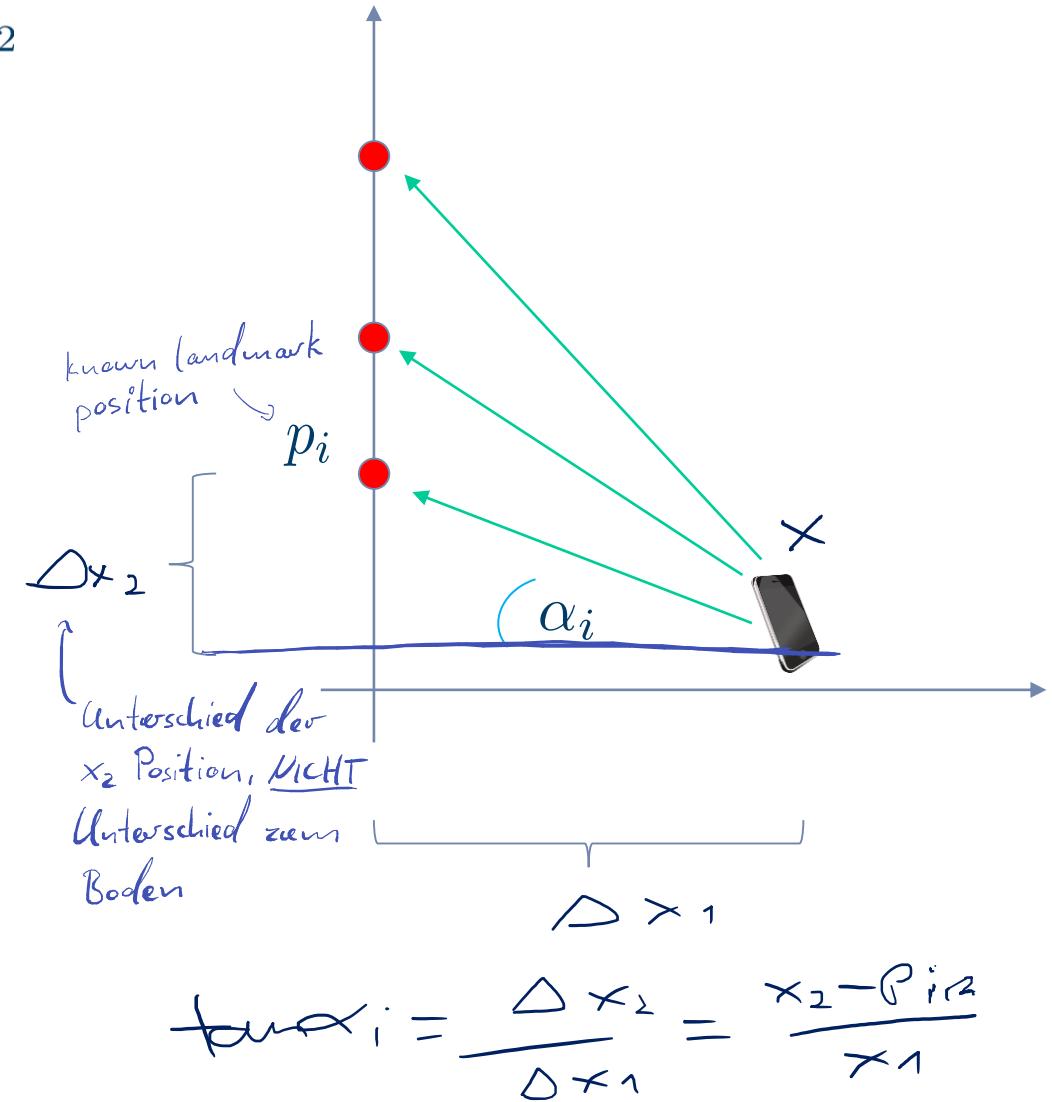
- Position $p_i = [0, p_{i,2}]^T \in \mathbb{R}^2$
- Angle $\alpha_i \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ to landmark i

for $i = 1, \dots, m$

- **Stacked measurement equation:**

$$\underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \text{atan}\left(\frac{x_2 - p_{1,2}}{x_1}\right) \\ \vdots \\ \text{atan}\left(\frac{x_2 - p_{m,2}}{x_1}\right) \end{bmatrix}}_{=:h(x)} + \underbrace{\begin{bmatrix} e_1 \\ \vdots \\ e_m \end{bmatrix}}_{=:e}$$

*Non linear
(poor quality in case of high noise)*

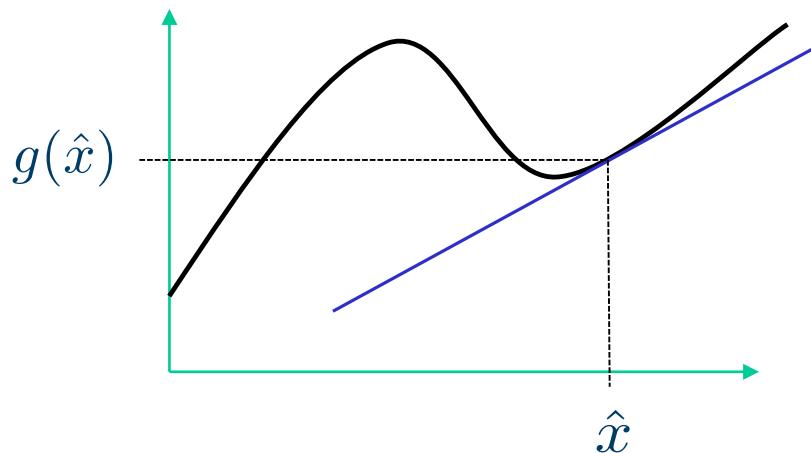


- Taylor series expansion of 1D function $g : \mathbb{R} \rightarrow \mathbb{R}$ with $\hat{x} \in \mathbb{R}$:

$$g(\hat{x}) + \frac{g'(\hat{x})}{1!}(x - \hat{x}) + \left[\frac{g''(\hat{x})}{2!}(x - \hat{x})^2 + \frac{g^{(3)}(\hat{x})}{3!}(x - \hat{x})^3 + \dots \right]$$

- Linearization of $g : \mathbb{R} \rightarrow \mathbb{R}$ around $\hat{x} \in \mathbb{R}$:
In unserem Anwendungsfall irrelevant

$$g(x) \approx g(\hat{x}) + g'(\hat{x})(x - \hat{x})$$



- **Objective:**

Find the root of a one-dimensional function, i.e.,

$$x : g(x) = 0$$

- **Algorithm:**

Succesively improve \hat{x}^i by linearization, i.e., solve

$$g(\hat{x}^i) + g'(\hat{x}^i)(x - \hat{x}^i) = 0$$

(\hookrightarrow Tangentengleichung)

which yields

$$\hat{x}^{i+1} = \hat{x}^i - \frac{g(\hat{x}^i)}{g'(\hat{x}^i)}$$

\uparrow
next step

minimize: $\|y - h(x)\|^2$

$$\begin{aligned} \Rightarrow g(\hat{x}^i) &= -g'(\hat{x}^i)(x - \hat{x}^i) \quad | \cdot (-1) \\ -g(\hat{x}^i) &= g'(\hat{x}^i)(x - \hat{x}^i) \\ \frac{-g(\hat{x}^i)}{g'(\hat{x}^i)} &= x - \hat{x}^i \quad | + \hat{x}^i \end{aligned}$$

- **Objective:**

Minimize a one-dimensional function, i.e.,

$$\hat{x} = \arg \min_x g(x)$$

$$\curvearrowleft \|y - h(x)\|^2$$

- **Algorithm:**

Successively improve \hat{x}^i by second-order Taylor series approximation, i.e., solve

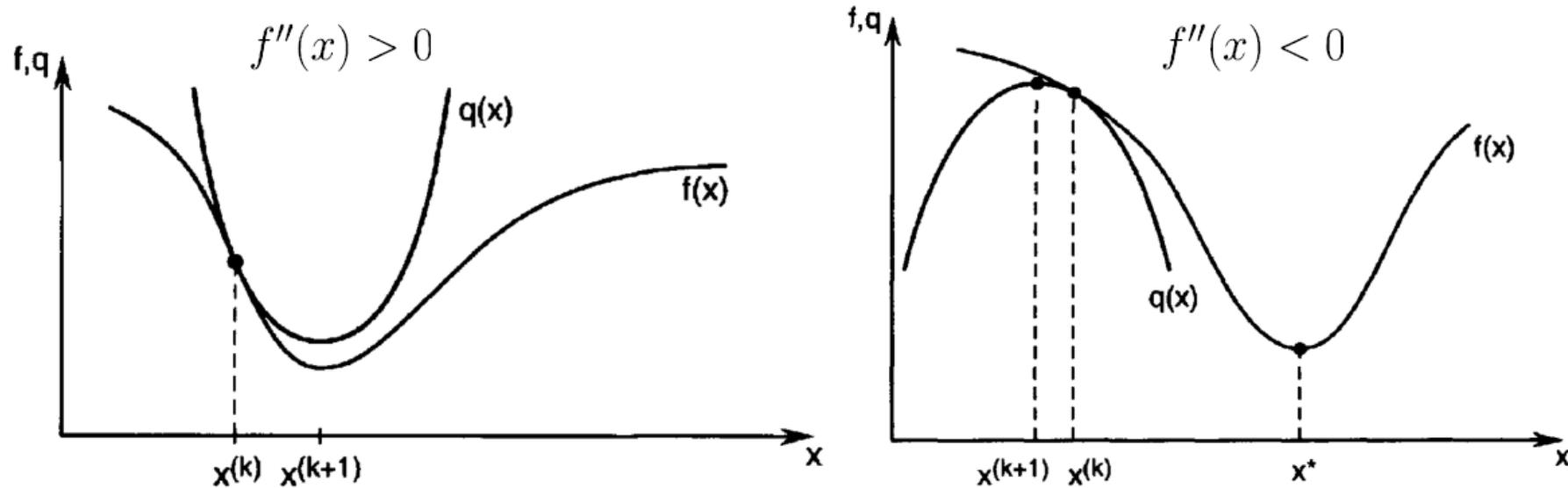
$$\hat{x}^{i+1} = \arg \min_x \left(g(\hat{x}^i) + g'(\hat{x}^i)(x - \hat{x}^i) + \frac{1}{2}g''(\hat{x}^i)(x - \hat{x}^i)^2 \right)$$

which yields

$$\hat{x}^{i+1} = \hat{x}^i - \frac{g'(\hat{x}^i)}{g''(\hat{x}^i)}$$

Requires Hessians for vector valued functions → Complex

- Newton's method works well for finding the minimum of $f(x)$ if $f''(x) > 0$ (positive curvature)
- If $f''(x) < 0$ (negative curvature), it may fail to converge



Reference: https://www.cs.cmu.edu/~wtchu/courses/2014s_OPT/

- $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with component functions $g = [g_1, \dots, g_m]^T$

- Linearization point $\hat{x} \in \mathbb{R}^n$

- Linearization:

$$g(x) \approx g(\hat{x}) + \mathbf{G}_{\hat{x}}(x - \hat{x})$$

Steigung d. Unterschieds
≈

with Jacobi matrix

The Jacobian Matrix is the matrix representing the best linear map approximation of f near a given point (in 2D (a, b))

$$\mathbf{G}_{\hat{x}} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1}(\hat{x}) & \frac{\partial g_1}{\partial x_2}(\hat{x}) & \dots & \frac{\partial g_1}{\partial x_n}(\hat{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_1}(\hat{x}) & \frac{\partial g_m}{\partial x_2}(\hat{x}) & \dots & \frac{\partial g_m}{\partial x_n}(\hat{x}) \end{bmatrix}$$

- **Objective:**

$$x^{LS} = \arg \min_x \|r(x)\|^2$$

$y - h(x)$

• y : measurement vector
• h : non-linear measurement function

Linear approximation of $r(x)$ around $\hat{x}^{(l)}$

$$\approx \arg \min_x \|r(\hat{x}^{(l)}) + \mathbf{J}_l(x - \hat{x}^{(l)})\|^2 = \|\mathbf{J}_l \cdot \hat{x}^{(l)} - r(\hat{x}^{(l)}) - \mathbf{J}_l \cdot x\|^2$$

$$\Rightarrow (\mathbf{J}_l \cdot \hat{x}^{(l)} - r(\hat{x}^{(l)})) = \mathbf{J}_l \cdot x$$

← previously used measurement equation

- **Algorithm:**

– Step 1: Choose initial estimate $\hat{x}^{(1)}$, $l = 1$

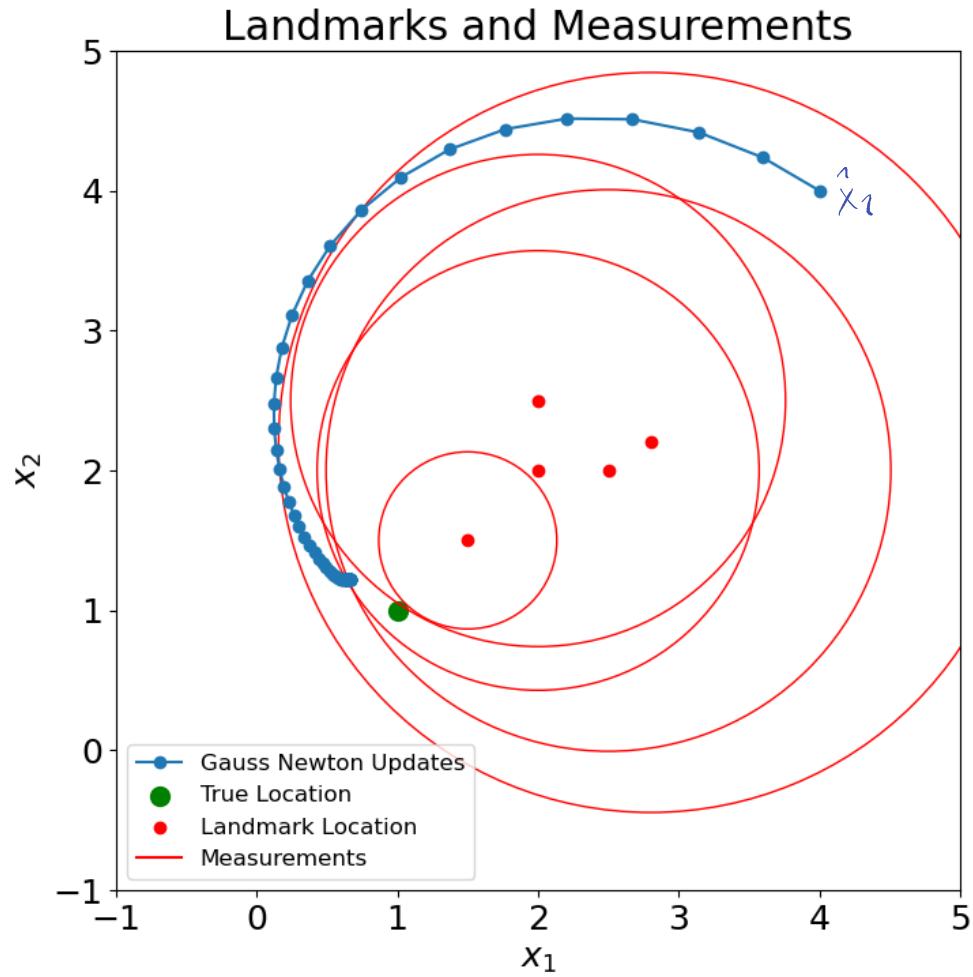
– Step 2: Use linear LS to get $\hat{x}^{(l+1)}$:

$$\begin{aligned} \hat{x}^{(l+1)} &= (\mathbf{J}_l^T \mathbf{J}_l)^{-1} \mathbf{J}_l^T (\mathbf{J}_l \hat{x}^{(l)} - r(\hat{x}^{(l)})) \\ &= \hat{x}^{(l)} - (\mathbf{J}_l^T \mathbf{J}_l)^{-1} \mathbf{J}_l^T r(\hat{x}^{(l)}) \end{aligned}$$

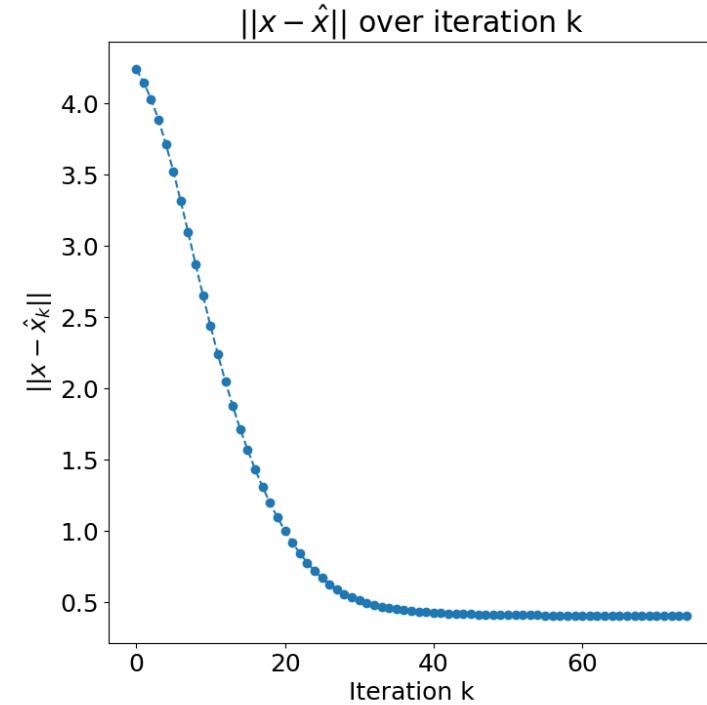
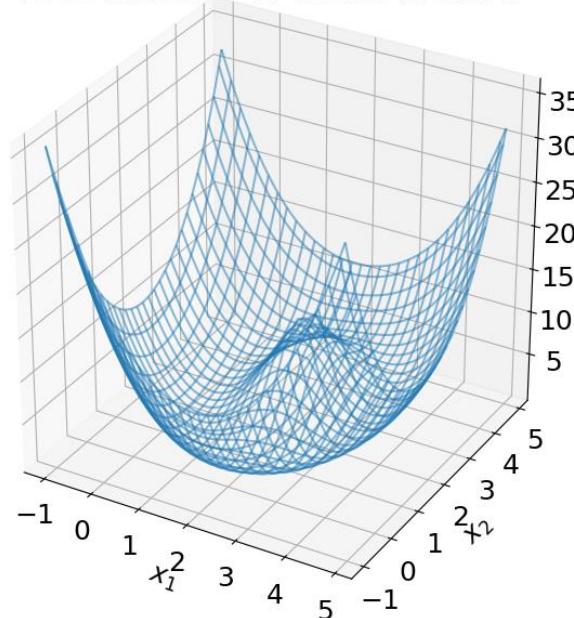
because $(\mathbf{J}_l^T \mathbf{J}_l)^{-1} \cdot \mathbf{J}_l^T \cdot \mathbf{J}_l = I$
(identity matrix)

– Step 3: Goto Step 2 until convergency is reached

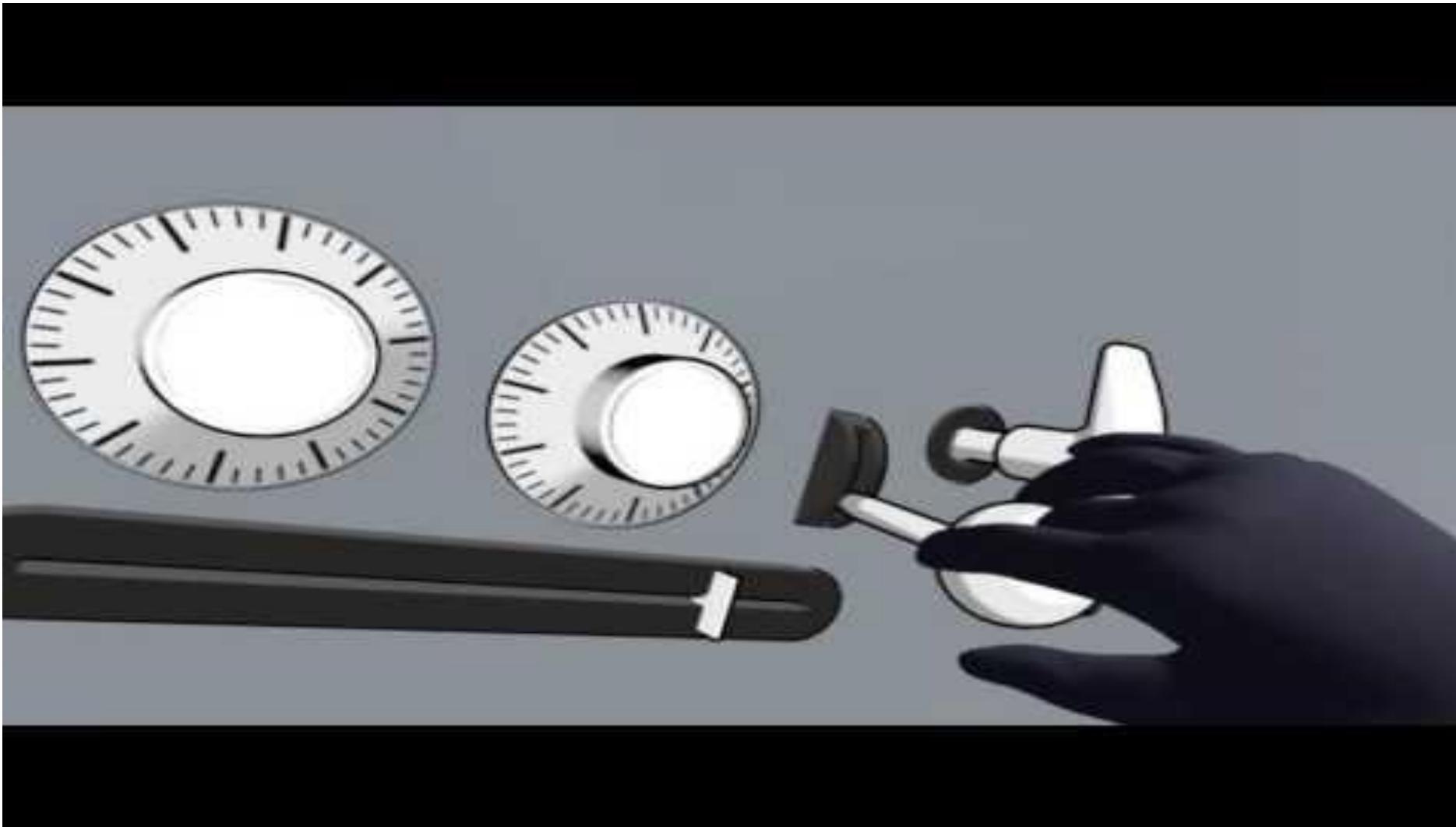
Example: Trilateration



Loss function of Gauss Newton



Images: Simon Steuernagel



<https://www.youtube.com/watch?v=QTz1zQAnMcU>

Questions?

→ Stud.IP, e-mail or live session

Sensor Data Fusion

Exercise 3

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab

- Review statistics
- Lecture review
- Questions
- Homework 2 solution
- Theory - Bancroft solution and closed-form solution for triangulation
- Problem 3 - Triangulation
- Homework 3 presentation

- Random variable $x \in \mathbb{R}$:
Random experiment whose outcome is associated with a real number.
- Cumulative distribution function (CDF):

$$F_x(x) = \text{Prob}(x \leq x)$$

where Prob is the probability measure

- $F_x(x)$ monotone increasing and $\lim_{x \rightarrow -\infty} F(x) = 0$, $\lim_{x \rightarrow +\infty} F(x) = 1$
- Probability density function $p_x(x)$:

$$F_x(x) = \text{Prob}(x \leq x) = \int_{-\infty}^x p_x(u) du$$

- If $F_x(x)$ differentiable:

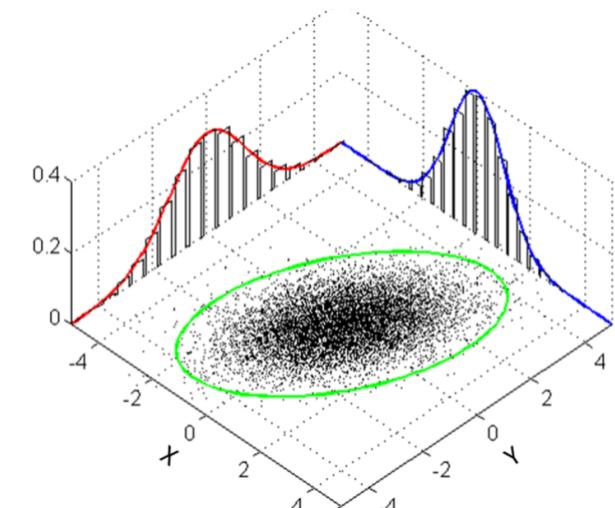
$$p_x(x) = \frac{d}{dx} F_x(x)$$

- Random vector $\mathbf{x} = [x_1, x_2]^T \in \mathbb{R}^2$: Column vector of scalar RVs
- Joint CDF function and joint PDF:

$$\text{Prob}(\mathbf{x}_1 < x_1, \mathbf{x}_2 < x_2) = F_{\mathbf{x}}(\mathbf{x}) = \int_{-\infty}^{x_1} \int_{-\infty}^{x_2} p_{\mathbf{x}}(u_1, u_2) du_1 du_2.$$

- If $F_{\mathbf{x}}(\mathbf{x})$ differentiable: $p(\mathbf{x}) = \frac{\partial^2}{\partial x_1 \partial x_2} F(\mathbf{x})$
- Marginal density:
 $p_{\mathbf{x}_1}(x_1) = \int p_{\mathbf{x}}(x_1, x_2) dx_2.$
- Independence: $p_{\mathbf{x}}(x_1, x_2) = p_{\mathbf{x}_1}(x_1) \cdot p_{\mathbf{x}_2}(x_2)$
- Conditioning on x_2 :

$$p_{\mathbf{x}_1}(x_1|x_2) = \frac{p_{\mathbf{x}}(x_1, x_2)}{p_{\mathbf{x}_2}(x_2)}$$



- Expectation:

$$\mathbb{E}[\mathbf{x}] = \int x p(x) dx = \begin{bmatrix} \mathbb{E}[x_1] \\ \mathbb{E}[x_2] \end{bmatrix} \text{ with } \mathbb{E}[x_i] = \int x_i p(x_i) dx_i$$

- Variance:

$$\text{Var}[x_i] = \mathbb{E}[(x_i - \mathbb{E}[x_i])^2]$$

- Covariance:

$$\text{Cov}[x_1, x_2] = \mathbb{E}[(x_1 - \mathbb{E}[x_1]) \cdot (x_2 - \mathbb{E}[x_2])]$$

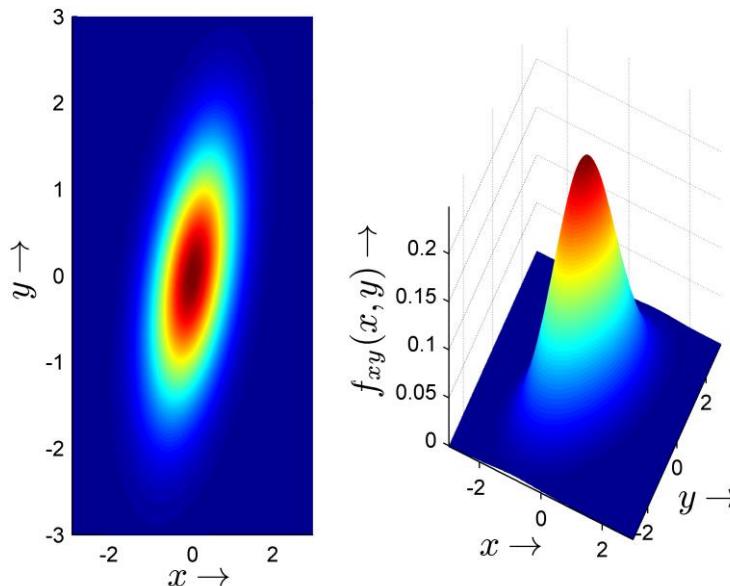
- Covariance matrix for n -dim. RV:

$$\begin{aligned} \text{Cov}[\mathbf{x}] &= \mathbb{E}[(\mathbf{x} - \mathbb{E}[\mathbf{x}]) \cdot (\mathbf{x} - \mathbb{E}[\mathbf{x}])^T] \\ &= \begin{bmatrix} \text{Var}[x_1] & \text{Cov}[x_1, x_2] & \dots & \text{Cov}[x_1, x_n] \\ \text{Cov}[x_2, x_1] & \text{Var}[x_2] & & \text{Cov}[x_2, x_n] \\ \vdots & & \ddots & \vdots \\ \text{Cov}[x_n, x_1] & \dots & & \text{Var}[x_n] \end{bmatrix} \end{aligned}$$

n -dimensional Gaussian distribution, i.e., $x \sim N(\mu, \mathbf{C})$

- Mean $\mu \in \mathbb{R}^n$
- Covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$

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



- $\mathbf{x} \in \mathbb{R}^n$
- $E[\mathbf{x}] = \hat{\mathbf{x}}$
- $\text{Cov}[\mathbf{x}] = \mathbf{C}$
- $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$

$$\begin{aligned} E[\mathbf{A}\mathbf{x} + b] &= \int_{\mathbb{R}^n} (\mathbf{A}\mathbf{x} + b)p(x) dx \\ &= \mathbf{A} \int_{\mathbb{R}^n} x p(x) dx + b \underbrace{\int_{\mathbb{R}^n} p(x) dx}_{=1} \\ &= \mathbf{A}\hat{\mathbf{x}} + b . \end{aligned}$$

$$\begin{aligned}\text{Cov}[\mathbf{A}\mathbf{x} + \mathbf{b}] &= \mathbb{E}\{(\mathbf{A}\mathbf{x} + \mathbf{b} - (\mathbf{A}\hat{\mathbf{x}} + \mathbf{b}))(\mathbf{A}\mathbf{x} + \mathbf{b} - (\mathbf{A}\hat{\mathbf{x}} + \mathbf{b}))^T\} \\ &= \mathbb{E}\{(\mathbf{A}\mathbf{x} - \mathbf{A}\hat{\mathbf{x}})(\mathbf{A}\mathbf{x} - \mathbf{A}\hat{\mathbf{x}})^T\} \\ &= \mathbb{E}\{\mathbf{A}(\mathbf{x} - \hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{A}^T\} \\ &= \mathbf{A} \text{Cov}[\mathbf{x}] \mathbf{A}^T \\ &= \mathbf{A} \mathbf{C} \mathbf{A}^T\end{aligned}$$

Important: If \mathbf{x} is Gaussian $\mathbf{A}\mathbf{x} + \mathbf{b}$ is Gaussian as well (no proof)

- Eigenvalue decomposition of SPD matrix \mathbf{C} :

$$\mathbf{C} = \mathbf{R}\mathbf{D}\mathbf{R}^T$$

with diagonal matrix \mathbf{D} and rotation matrix \mathbf{R}

$$\Rightarrow p(x) = \frac{1}{\sqrt{(2\pi)^n \det(\mathbf{C})}} \exp\left(-\frac{1}{2}(\mathbf{R}^T(x - \mu))^T \mathbf{D}^{-1}(\mathbf{R}^T(x - \mu))\right)$$

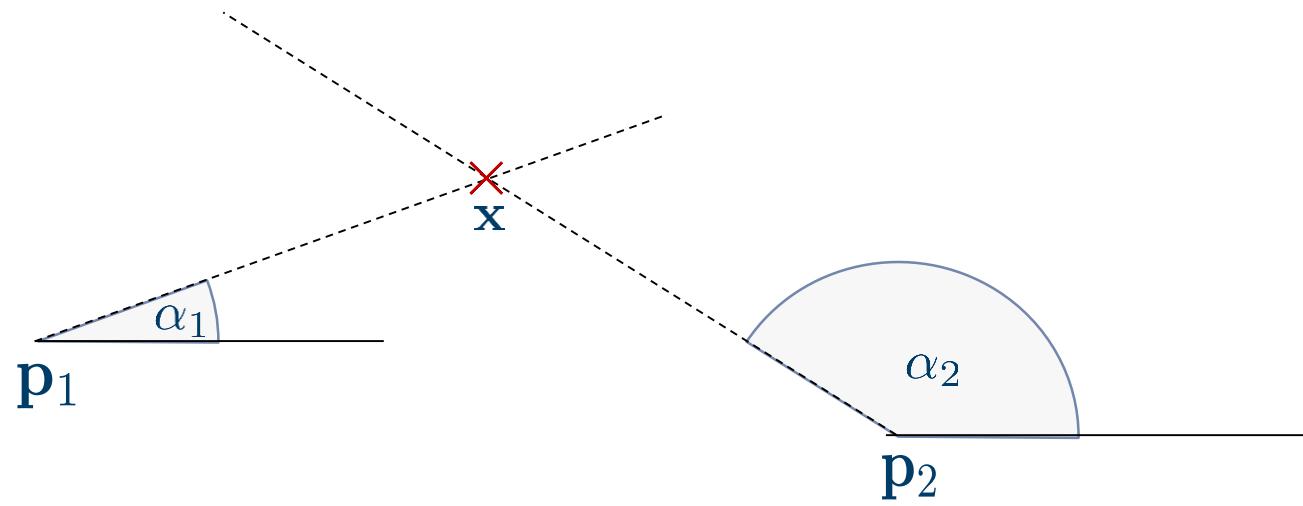
- Interpretation of $p(x) = c$?
- With $\mathbf{D} = \text{diag}(\sigma_1^2, \sigma_2^2)$ and $z := \mathbf{R}^T(x - \mu)$

$$z^T(\mathbf{D})^{-1}z = \tilde{c} \Leftrightarrow \frac{1}{\sigma_1^2}z_1^2 + \frac{1}{\sigma_2^2}z_2^2 = \tilde{c}$$

which is the equation of a scaled ellipse.

Can you explain how Triangulation works? Why can't you use the normal equation to solve it?

$$\alpha_i = \text{atan}2(x_2 - p_2, x_1 - p_1) + e_i$$



As the measurement equation is non-linear, we are unable to apply least square formula. We can still solve the problem by applying a closed-form solution or using iterative optimization.

Can you explain the steps of the Gauss-Newton algorithm?

- **Objective:**

$$x^{LS} = \arg \min_x \|r(x)\|^2 \approx \arg \min_x \|r(\hat{x}^{(l)}) + \mathbf{J}_l(x - \hat{x}^{(l)})\|^2$$

- **Algorithm:**

- **Step 1:** Choose initial estimate $\hat{x}^{(1)}$, $l = 1$
- **Step 2:** Use linear LS to get $\hat{x}^{(l+1)}$:

$$\begin{aligned}\hat{x}^{(l+1)} &= (\mathbf{J}_l^T \mathbf{J}_l)^{-1} \mathbf{J}_l^T \left(\mathbf{J}_l \hat{x}^{(l)} - r(\hat{x}^{(l)}) \right) \\ &= \hat{x}^{(l)} - (\mathbf{J}_l^T \mathbf{J}_l)^{-1} \mathbf{J}_l^T r(\hat{x}^{(l)})\end{aligned}$$

- **Step 3:** Goto **Step 2** until convergency is reached

What is a typical use for closed-form approximations?

- A closed-form approximation can be used as an initial guess for an iterative optimization method.
- For time critical applications, it might be necessary to apply an approximation instead of a time consuming optimization approach.

GPS consists of 24 satellites in orbit (20200km above mean sea level). Each satellite broadcasts its location (in spherical coordinates $[\theta, \phi, r]^T$) plus the emission time (see figures below). A GPS device receives at time $t = 0\text{s}$ the following four satellite signals:

$$p_1 = [0^\circ, 40^\circ, 20200\text{km}]^T,$$

$$p_2 = [10^\circ, 20^\circ, 20200\text{km}]^T,$$

$$p_3 = [10^\circ, -10^\circ, 20200\text{km}]^T,$$

$$p_4 = [-10^\circ, -20^\circ, 20200\text{km}]^T,$$

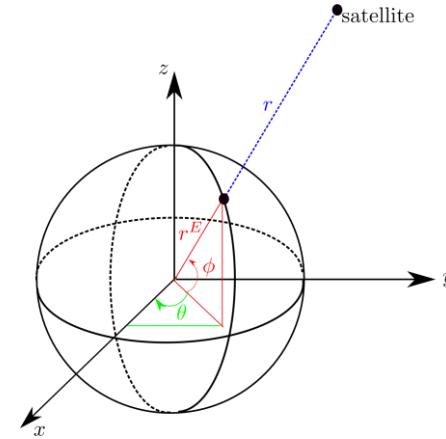
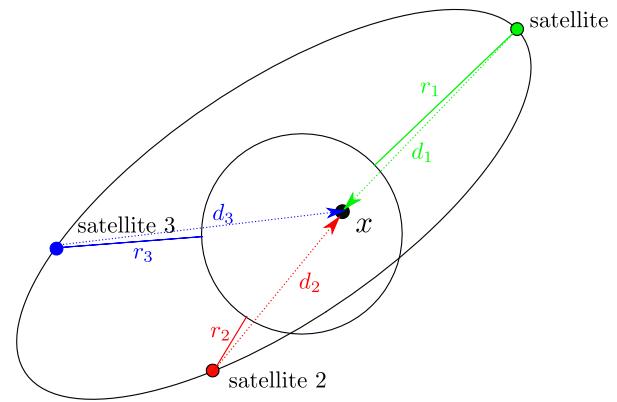
$$t_1 = -67.603\text{ms},$$

$$t_2 = -70.102\text{ms},$$

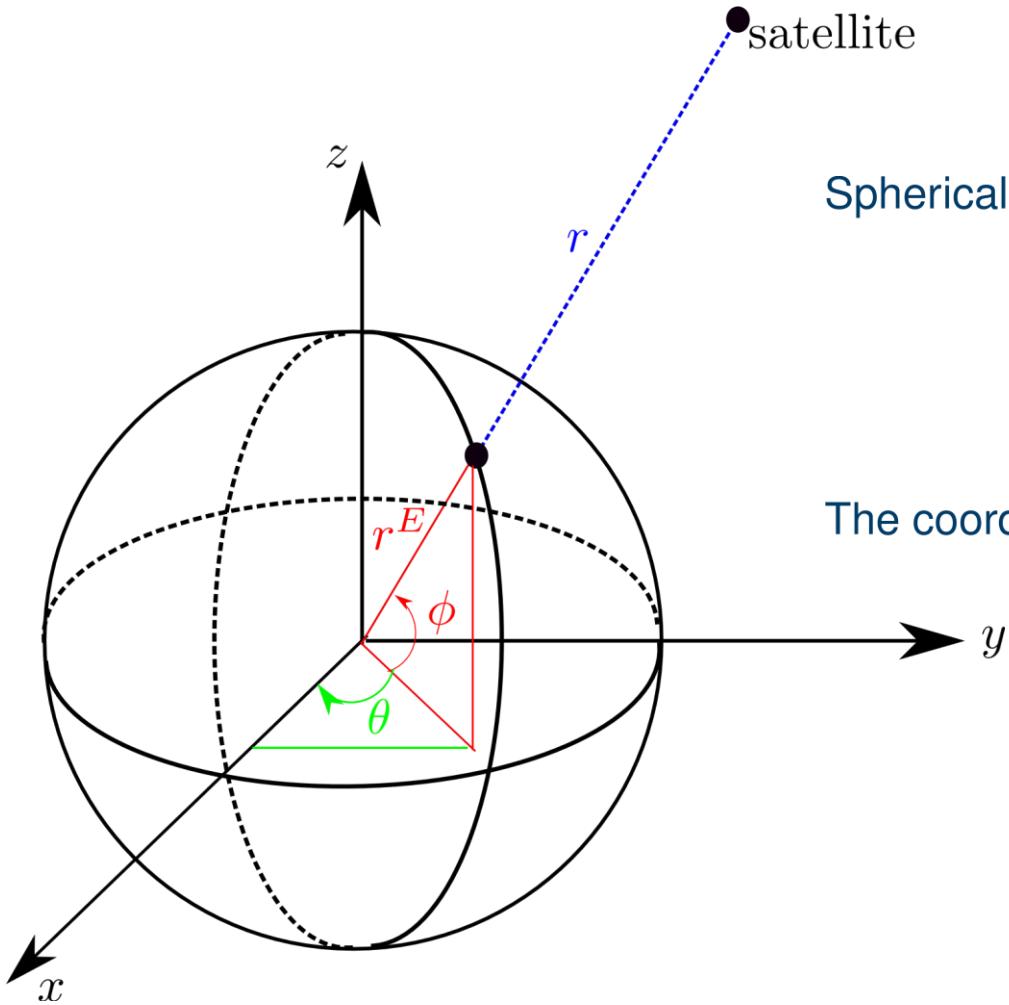
$$t_3 = -78.690\text{ms},$$

$$t_4 = -82.942\text{ms}.$$

Assume that the speed-of-light is $c = 3 \cdot 10^8 \text{m s}^{-1}$ and the Earth is an ideal sphere with radius $r^E = 6370\text{km}$.



a) Write a function which converts sphere coordinates (θ, ϕ, r) into Cartesian coordinates (x, y, z)



Spherical coordinates could be converted into Cartesian coordinates following

$$\begin{aligned} x &= (r^E + r) \cos \phi \cos \theta \\ y &= (r^E + r) \cos \phi \sin \theta \\ z &= (r^E + r) \sin \phi \end{aligned}$$

The coordinates of satellites are

$$\begin{aligned} P_1 &= [20354, 0, 17079]^T \\ P_2 &= [24588, 4336, 9087]^T \\ P_3 &= [25769, 4544, -4614]^T \\ P_4 &= [24588, -4336, -9088]^T \end{aligned}$$

b) Please calculate the distance between satellites and GPS device.
Using the distance measurements, form the measurement equation like we did in the lecture.

Get the distances between satellites and GPS device,

$$\begin{aligned}d_1 &= c \cdot |t_1| = 3 \times 10^8 \times 67.603 \times 10^{-3} = 20281\text{km} \\d_2 &= c \cdot |t_2| = 3 \times 10^8 \times 70.102 \times 10^{-3} = 21031\text{km} \\d_3 &= c \cdot |t_3| = 3 \times 10^8 \times 78.690 \times 10^{-3} = 23607\text{km} \\d_4 &= c \cdot |t_4| = 3 \times 10^8 \times 83.942 \times 10^{-3} = 24883\text{km}\end{aligned}$$

Measurement equation

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix} = \begin{bmatrix} \|\mathbf{x} - P_1\| \\ \|\mathbf{x} - P_2\| \\ \|\mathbf{x} - P_3\| \\ \|\mathbf{x} - P_4\| \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}$$

is non-linear but it could be solved with linear least squares directly by reformulation.

c) Reformulate the non-linear measurement equation in 2b) into a linear measurement equation and calculate the least squares estimate.

Squared measurement equation:

$$\begin{aligned}
 d_i^2 &= \|\mathbf{x} - P_i\|^2 + e_i^* \\
 &= (\mathbf{x}_x - P_{i,x})^2 + (\mathbf{x}_y - P_{i,y})^2 + (\mathbf{x}_z - P_{i,z})^2 + e_i^* \\
 &= \mathbf{x}_x^2 + \mathbf{x}_y^2 + \mathbf{x}_z^2 - 2\mathbf{x}_x P_{i,x} - 2\mathbf{x}_y P_{i,y} - 2\mathbf{x}_z P_{i,z} + P_{i,x}^2 + P_{i,y}^2 + P_{i,z}^2 + e_i^* ,
 \end{aligned}$$

where e_i^* is a new error term subsuming the transformed error e_i .
 Subtracting d_4^2 using d_1^2 , we have,

$$\begin{aligned}
 d_1^2 - d_4^2 &= \|\mathbf{x}\|^2 - \|P_1\|^2 - 2 \begin{bmatrix} P_{1,x} - P_{4,x} & P_{1,y} - P_{4,y} & P_{1,z} - P_{4,z} \end{bmatrix} \begin{bmatrix} \mathbf{x}_x \\ \mathbf{x}_y \\ \mathbf{x}_z \end{bmatrix} \\
 &\quad + \|P_1\|^2 - \|P_4\|^2 + e_1^* - e_4^*
 \end{aligned}$$

c) cont.

Moving $\|P_1\|^2 - \|P_4\|^2$ to the left side of the equation, we have

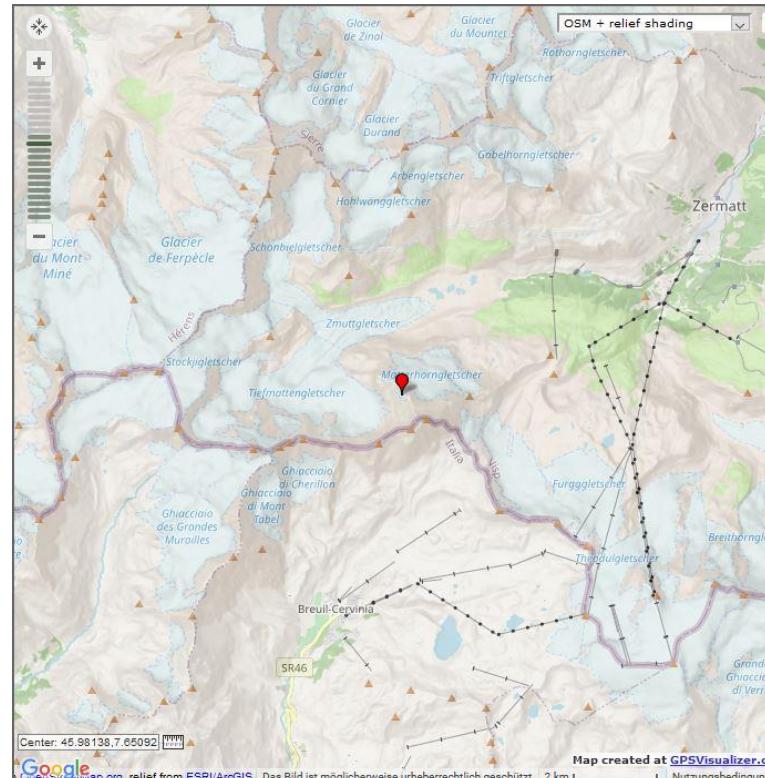
$$\underbrace{d_1^2 - d_4^2}_{:=\mathbf{y}_{1,4}} \underbrace{-\|P_1\|^2 + \|P_4\|^2}_{=0 \text{ in our case}} = \underbrace{-2 [P_{1,x} - P_{4,x} \quad P_{1,y} - P_{4,y} \quad P_{1,z} - P_{4,z}]}_{:=\mathbf{H}_{1,4}} \underbrace{\begin{bmatrix} \mathbf{x}_x \\ \mathbf{x}_y \\ \mathbf{x}_z \end{bmatrix}}_{\mathbf{x}} + e_1^* - e_4^*$$

In the similar manner we could have

$$\begin{bmatrix} \mathbf{y}_{1,3} \\ \mathbf{y}_{2,4} \\ \mathbf{y}_{3,4} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{1,4} \\ \mathbf{H}_{2,4} \\ \mathbf{H}_{3,4} \end{bmatrix} \mathbf{x} + \mathbf{e}$$

d) Write a function which converts the Cartesian coordinates into sphere coordinates. Use the function you implemented, calculate the longitude and latitude of the GPS device, and find out where it is using using *GPS Visualizer*:

<http://www.gpsvisualizer.com/map?form=google>



Nonlinear Measurement Equation with Additive Noise:

$$y = h(x) + e$$

Assumptions:

- Setting 1
- Overdetermined, $m > n$

Objective:

$$x^{LS} = \arg \min_x \| \underbrace{y - h(x)}_e \|_{\mathbf{W}}^2$$

General Solution Approaches:

- Iterative optimization
- Closed-form approximation by reformulation and linear LQ

- **Desired:** Receiver location $x \in \mathbb{R}^2$
- **Given:** Distances to m landmarks:
 - Position $p_i = [p_{i,1}, p_{i,2}]^T \in \mathbb{R}^2$
 - Distance $d_i \in \mathbb{R}$ to landmark i

- **Indiv. measurement equation:**

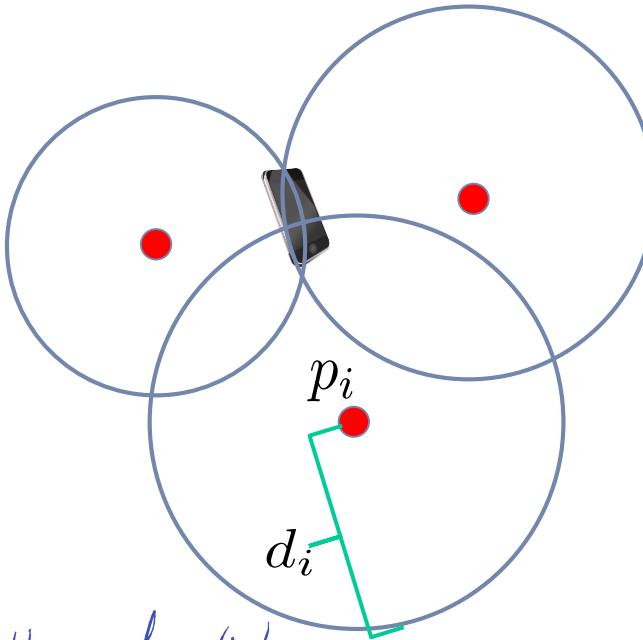
$$d_i = \|x - p_i\| + e_i$$

with measurement error $e_i \in \mathbb{R}$.

- **Stacked measurement equation:**

$$\underbrace{\begin{bmatrix} d_1 \\ \vdots \\ d_m \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \|x - p_1\| \\ \vdots \\ \|x - p_m\| \end{bmatrix}}_{=:h(x)} + \underbrace{\begin{bmatrix} e_1 \\ \vdots \\ e_m \end{bmatrix}}_{=:e}$$

Problem, wenn Bewegungsfunktion ausformuliert wird, ist die Funktion nicht linear (wegen Wurzel & Quadrat)
 => Idee: mit quadrierten Distanzen rechnen
 Fkt. wird dadurch linear



- Squared meas. equation:

$$\begin{aligned}
 d_i^2 &= \|x - p_i\|^2 + e_i^* \\
 &= (x_1 - p_{i,1})^2 + (x_2 - p_{i,2})^2 + e_i^* \\
 &= \underbrace{-2x_1 p_{i,1} - 2x_2 p_{i,2}}_{\text{kann mittels } H_1 \text{ dargestellt werden}} + \underbrace{\|p_i\|^2}_{\text{quadrieren}} + R^2 + e_i^*
 \end{aligned}$$

with $R^2 := \|x\|^2 = (x_1)^2 + (x_2)^2$

- Linear measurement equation for given R^2 :

$$y = \mathbf{H}_1 x + \mathbf{H}_2 R^2 + e^*$$

with

$$y = \begin{bmatrix} d_1^2 - \|p_1\|^2 \\ \vdots \\ d_m^2 - \|p_m\|^2 \end{bmatrix}, \quad \mathbf{H}_1 = \begin{bmatrix} -2p_{1,1} & -2p_{1,2} \\ \vdots & \vdots \\ -2p_{m,1} & -2p_{m,2} \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

- Least squares solution for a fixed R^2 : $\gamma = \mathbf{H}x$

$$\begin{aligned} x^{LS}(R^2) &= (\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T (y - \mathbf{H}_2 R^2) \\ &= z_1 + R^2 z_2 \end{aligned}$$

with $z_1 := (\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T y$ and $z_2 := -(\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T \mathbf{H}_2$

- What is R^2 ?

$$\begin{aligned} R^2 &\approx \|x^{LS}(R^2)\|^2 \\ &= (z_1 + R^2 z_2)^T \cdot (z_1 + R^2 z_2) \end{aligned}$$

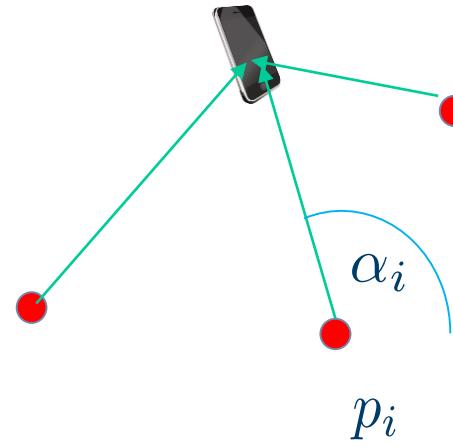
- Solve the following quadratic equation for R^2 :

$$0 = z_1^T z_1 + z_1^T z_2 R^2 + R^2 z_2^T z_1 + (R^2)^2 z_2^T z_2 - R^2$$

- Note that there can be multiple solutions to R^2 , but solutions outside the realm of interest could be discarded

- **Desired:** Cartesian object position $x = [x_1, x_2]^T \in \mathbb{R}^2$
- **Given:** m angular measurements to m landmarks:
 - Position $p_i = [p_{i,1}, p_{i,2}]^T \in \mathbb{R}^2$
 - Angle $\alpha_i \in [-\pi, \pi]$ to landmark i

for $i = 1, \dots, m$

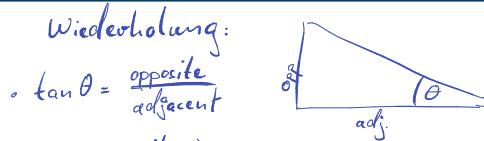


- **Stacked measurement equation:**

$$\underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \text{atan2}(x_2 - p_{1,2}, x_1 - p_{1,1}) \\ \vdots \\ \text{atan2}(x_2 - p_{m,2}, x_1 - p_{m,1}) \end{bmatrix}}_{=:h(x)} + \underbrace{\begin{bmatrix} e_1 \\ \vdots \\ e_m \end{bmatrix}}_{=:e}$$

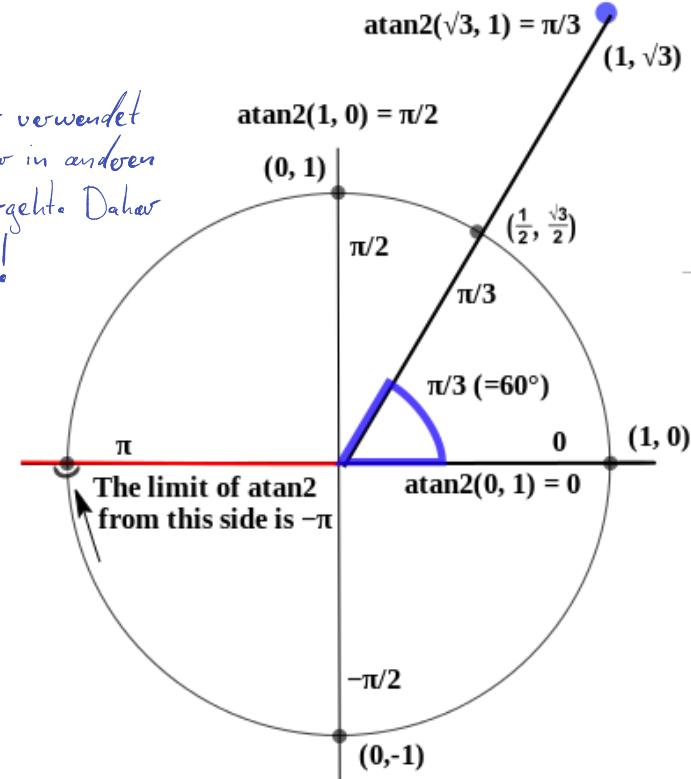
- Four quadrant tangent inverse
- Returns values in $(-\pi, \pi]$
- Standard inverse of tangent returns only values in $(-\pi/2, \pi/2)$

$$\text{atan2}(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geq 0, \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0, \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0. \end{cases}$$



- $\arctan\left(\frac{\text{opposite}}{\text{adjacent}}\right) = \theta$

\Rightarrow Methode kann nicht verwendet werden, wenn Vektor in anderen Quadranten übergeht. Daher atan2 verwenden!



<https://en.wikipedia.org/wiki/Atan2>

- Reformulation:

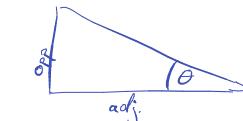
$$(x_1 - p_{i,1}) \cdot \tan(\alpha_i) = x_2 - p_{i,2}$$

- Linear measurement equation:

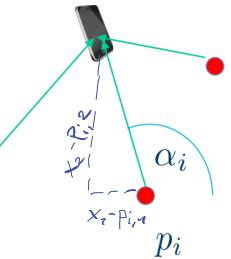
$$\underbrace{\begin{bmatrix} p_{1,1}\tan(\alpha_1) - p_{1,2} \\ \vdots \\ p_{m,1}\tan(\alpha_m) - p_{m,2} \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \tan(\alpha_1) & -1 \\ \vdots & \vdots \\ \tan(\alpha_m) & -1 \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1^* \\ \vdots \\ e_m^* \end{bmatrix}}_{=:e^*}$$

Wiederholung:
 • $\tan \theta = \frac{\text{opposite}}{\text{adjacent}}$

• $\arctan\left(\frac{\text{opposite}}{\text{adjacent}}\right) = \theta$



$$\tan \alpha_i = \frac{x_2 - p_{i,2}}{x_1 - p_{i,1}}$$



$$(x_1 - p_{i,1}) \cdot \tan(\alpha_i) = x_2 - p_{i,2}$$

$$x_1 \cdot \tan(\alpha_i) - p_{i,1} \cdot \tan(\alpha_i) = x_2 - p_{i,2}$$

$$p_{i,2} + p_{i,1} \cdot \tan(\alpha_i) = -x_1 \cdot \tan(\alpha_i) + x_2 \quad | \cdot -1$$

$$p_{i,1} \cdot \tan(\alpha_i) - p_{i,2} = x_1 \cdot \tan(\alpha_i) - x_2$$

Exact line from this equation

- Problem:

Measurement part of the measurement matrix, which introduces additional errors.

Problem 3 - Triangulation

Assume you receive the following angular measurements in radian from two transmitters:

$$\alpha_1 = \frac{\pi}{4}, \mathbf{p}_1 = [0 \quad 0]^T$$

$$\alpha_2 = \frac{3\pi}{4}, \mathbf{p}_2 = [4 \quad 0]^T$$

Calculate your position.

Hint: $\tan(0) = 0$, $\tan(\frac{\pi}{4}) = 1$, $\tan(\frac{3\pi}{4}) = -1$.

Visualize the measurements to confirm your results.

Next, assume a third measurement is received

$$\alpha_3 = 0, \mathbf{p}_3 = [0 \quad 2.5]^T$$

Update the estimate of your position.

Wiederholung:

$$\underbrace{\begin{bmatrix} p_{1,1}\tan(\alpha_1) - p_{1,2} \\ \vdots \\ p_{m,1}\tan(\alpha_m) - p_{m,2} \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \tan(\alpha_1) & -1 \\ \vdots & \vdots \\ \tan(\alpha_m) & -1 \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1^* \\ \vdots \\ e_m^* \end{bmatrix}}_{=:e^*}$$

⇒ Normal equation:

$$\boxed{\mathbf{H}^T \mathbf{W} \mathbf{y} = \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{x}}$$

- As \mathbf{H} has full column rank:

auf andere Seite ziehen

$$x^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}$$

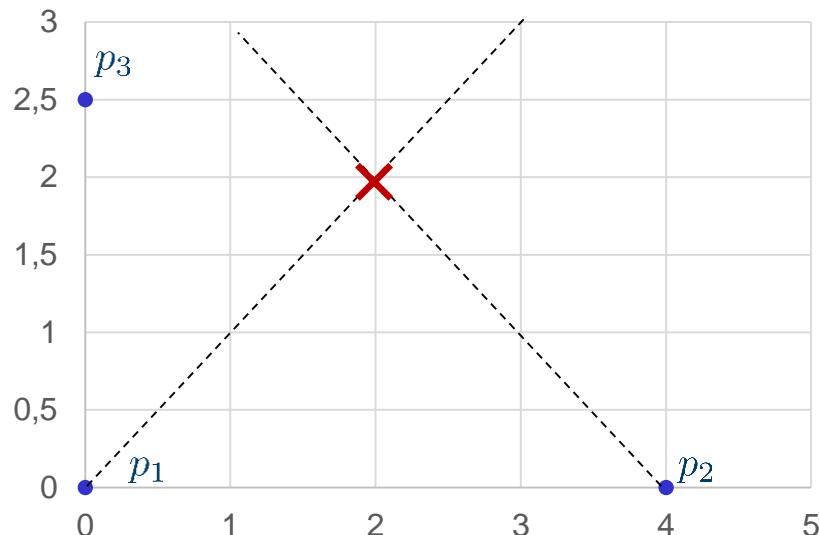
Problem 3: Solution

Reformulating, we get

$$\mathbf{y} = \begin{bmatrix} 0 \\ -4 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 1 & -1 \\ -1 & -1 \end{bmatrix}$$

Applying least squares:

$$\mathbf{x}^{LS} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 4 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$



Linear measurement equation:

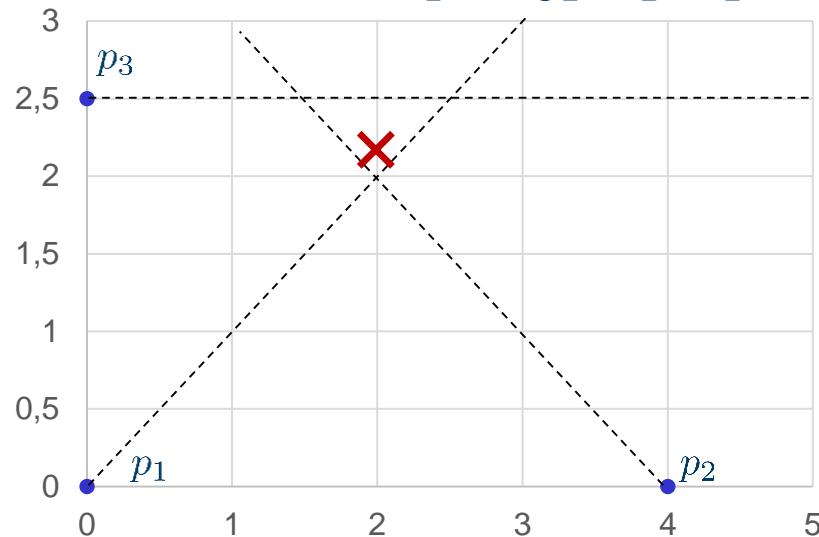
$$\underbrace{\begin{bmatrix} p_{1,1}\tan(\alpha_1) - p_{1,2} \\ \vdots \\ p_{m,1}\tan(\alpha_m) - p_{m,2} \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} \tan(\alpha_1) & -1 \\ \vdots & \vdots \\ \tan(\alpha_m) & -1 \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1^* \\ \vdots \\ e_m^* \end{bmatrix}}_{=:e^*}$$

Adding the new measurement, we get

$$\mathbf{y} = \begin{bmatrix} 0 \\ -4 \\ -2.5 \end{bmatrix}, \mathbf{H} = \begin{bmatrix} 1 & -1 \\ -1 & -1 \\ 0 & -1 \end{bmatrix}$$

Applying least squares:

$$\mathbf{x}^{LS} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{3} \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 6.5 \end{bmatrix} \approx \begin{bmatrix} 2 \\ 2.17 \end{bmatrix}$$



Look again at the GPS setting of last homework (satellites are 20200km above mean sea level, each satellite broadcasts its location in spherical coordinates $[\theta, \phi, r]^T$ plus the emission time (see figures below), assume that the speed-of-light is $c = 3 \cdot 10^8 \text{ m s}^{-1}$ and the Earth is an ideal sphere with radius $r^E = 6370\text{km}$)

This time, the device only receives the first three satellite signals at time $t = 0\text{s}$:

$$p_1 = [0^\circ, 40^\circ, 20200\text{km}]^T,$$

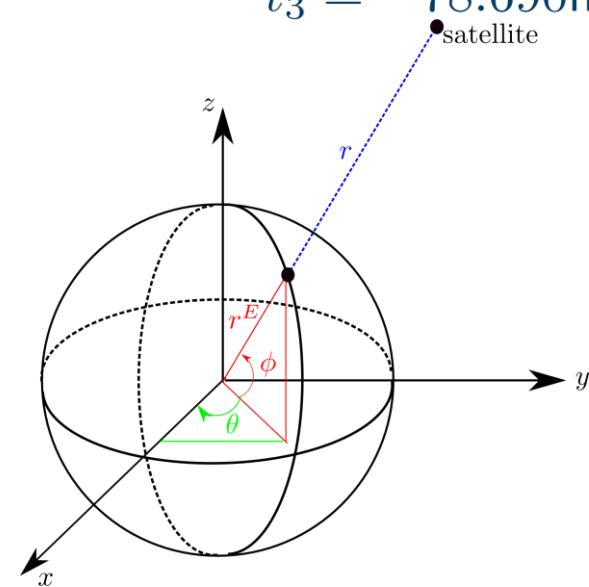
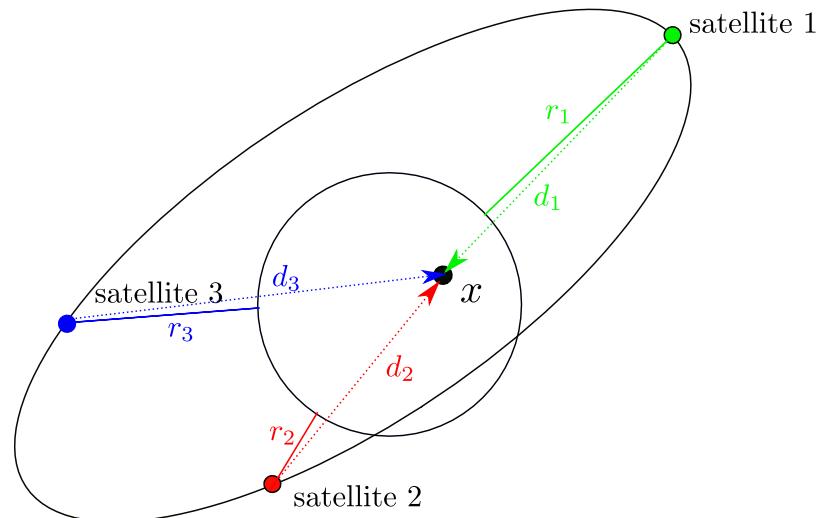
$$p_2 = [10^\circ, 20^\circ, 20200\text{km}]^T,$$

$$p_3 = [10^\circ, -10^\circ, 20200\text{km}]^T,$$

$$t_1 = -67.603\text{ms},$$

$$t_2 = -70.102\text{ms},$$

$$t_3 = -78.690\text{ms}.$$



- a) Would the reformulation from last times still work? If not, explain why.

- b) Instead of reformulating the measurement equation into a linear equation, implement the Gauss-Newton method to calculate the least squares solution.
Hint:
 1. you will need to use symbolic python
 2. you will need to use `sym.Matrix` instead of `np.array` for most variables: convert `np` arrays into `sympy` as necessary, the two packages don't mix well
 3. `sympy` provides `.jacobian(...)` for calculating the Jacobian
 4. `sympy` provides `.subs(...)` for symbolic substitution

- c) Alternatively, use the Bancroft solution to determine the position of the GPS device. How can we handle multiple solutions for R^2 in this case?
Hint: numpy provides `numpy.roots()` to find the roots of a polynomial

Sensor Data Fusion

Stochastic Least Squares

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Dead reckoning
12. Dynamic models: Tracking (**live**)
13. Advanced Topic
14. Summary and Discussion (**live**)

- (Linear) estimators
- Mean squared error
- Bias-variance decomposition
- Stochastic least squares
- Gauss-Markov theorem

Linear Transformation of RVs: Summary (see Exercise)



- $x \in \mathbb{R}^n$ with mean $E[x] = \hat{x}$ and $\text{Cov}[x] = \mathbf{C}_x$ (Covarianzmatrix)
random variable
- $y \in \mathbb{R}^n$ with mean $E[y] = \hat{y}$ and $\text{Cov}[y] = \mathbf{C}_y$
- $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$
l m dim Variable

- $E[Ax + b] = \underbrace{A\hat{x} + b}_{\text{will be random}}$ *wichtig!*
- $\text{Cov}[Ax + b] = A\mathbf{C}_x A^T$

- $E[x + y] = \hat{x} + \hat{y}$
- $\text{Cov}[x + y] = \mathbf{C}_y + \mathbf{C}_x$ (for uncorrelated x and y)

Random Variable

For any RV $z \in \mathbb{R}^n$:

Expectation of ...

mean of z

End up with semi positive Matrix

$$\begin{aligned}\text{Cov}[z] &= E[(z - E[z])(z - E[z])^T] \\ &= E[zz^T - zE[z]^T - E[z]z^T + E[z]E[z]^T] \\ &= E[zz^T] - E[z]E[z]^T\end{aligned}$$

Rechenregel:

$$E(E(z)^T) = E(z)^T$$

$$E(2) = 2$$

$$E(100) = 100$$

State: Quantity of interest (unknown, desired) $x \in \mathbb{R}^n$

Measurement: Observation received from the sensor (given) $y \in \mathbb{R}^m$

Meas. Error: Sensor error due to noise (unknown) $e \in \mathbb{R}^m$

Linear Measurement Equation with Additive Noise:

due to RVE, wind y and random $y = \mathbf{H}x + e$
 \uparrow is random, but we know..

with measurement matrix $\mathbf{H} \in \mathbb{R}^{m \times n}$

Three Settings:

1. No further statistical information available
2. Statistical information about the error is available (Fisher approach)
3. Statistical information about both error and state is available
(Bayesian approach)

$$\mathbb{E}[e] = 0 \text{ and } \text{Cov}[e] = \mathbf{C}$$

Example: Fusion of Two 1D Estimates

- **Desired:**

One-dimensional location $x \in \mathbb{R}$

1D Position

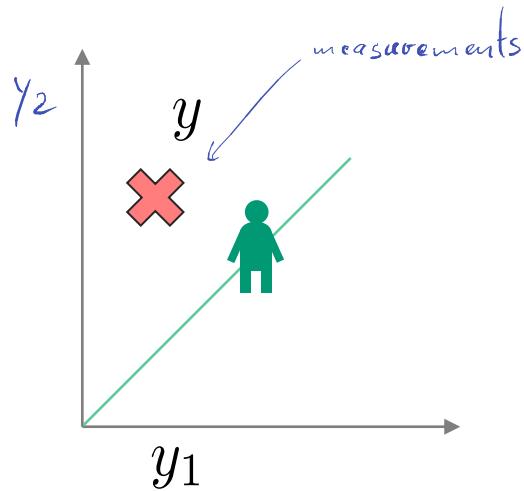


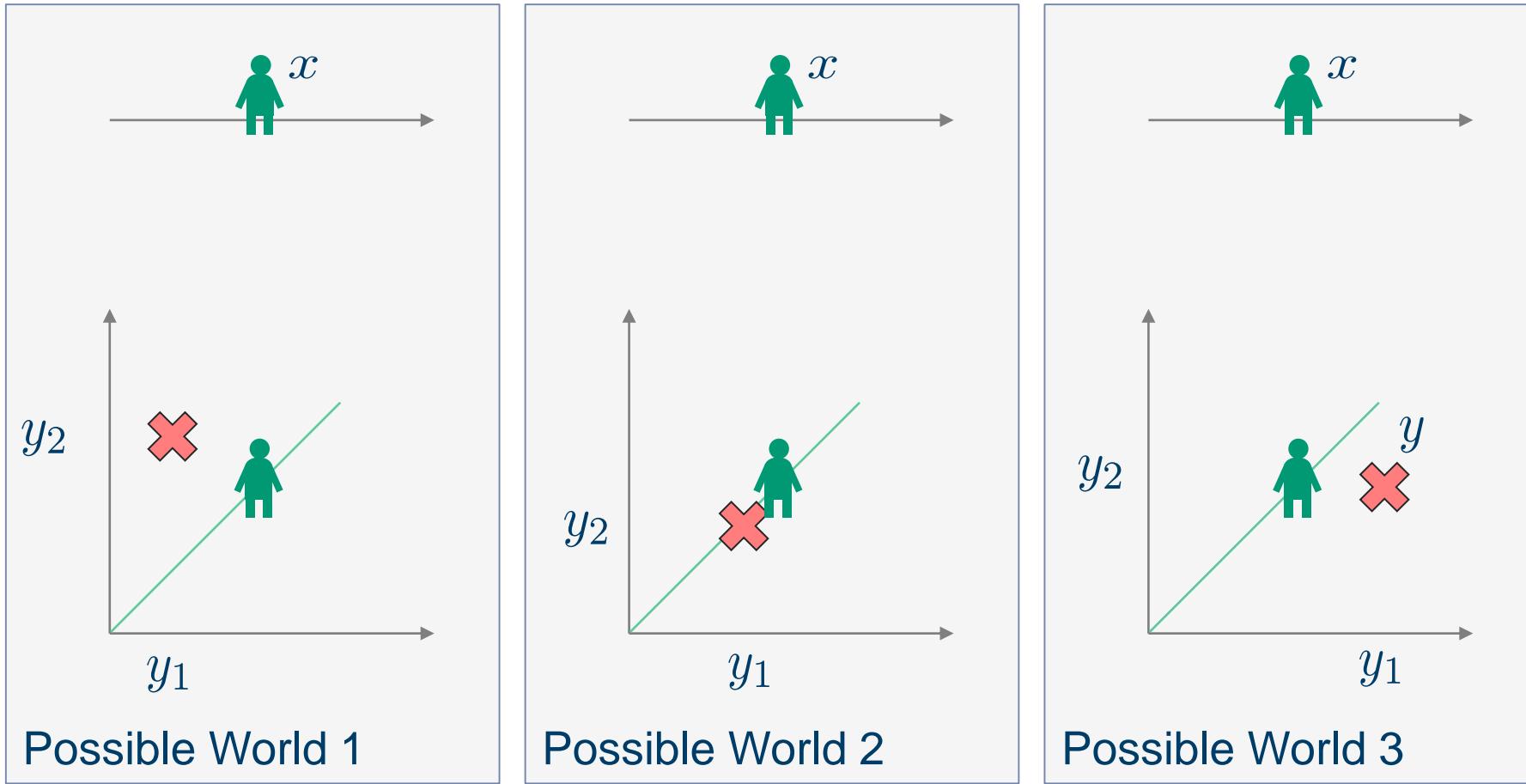
- **Given:**

- Meas. Sensor A: $y_1 \in \mathbb{R}$
- Meas. Sensor B: $y_2 \in \mathbb{R}$

- **Stacked measurement equation:**

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{=:y} = \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{=:H} x + \underbrace{\begin{bmatrix} e_1 \\ e_2 \end{bmatrix}}_{=:e}$$





↳ Many possible worlds with different possibilities (mean, Variance)

- Estimator: maps the measurement y to the unknown state x

Function $\theta : \mathbb{R}^m \rightarrow \mathbb{R}^n$

$\overset{\wedge}{\theta}$ _{theta}

Notation: $\theta(y) = \theta_y \in \mathbb{R}^n$

- Linear estimator:

(linear transformation of the measurement data to the state)

$$\theta_y = \mathbf{K}y + b \text{ with } \mathbf{K} \in \mathbb{R}^{n \times m} \text{ and } b \in \mathbb{R}^n$$

inverted measurement equation

offset X Korrektur aus VL

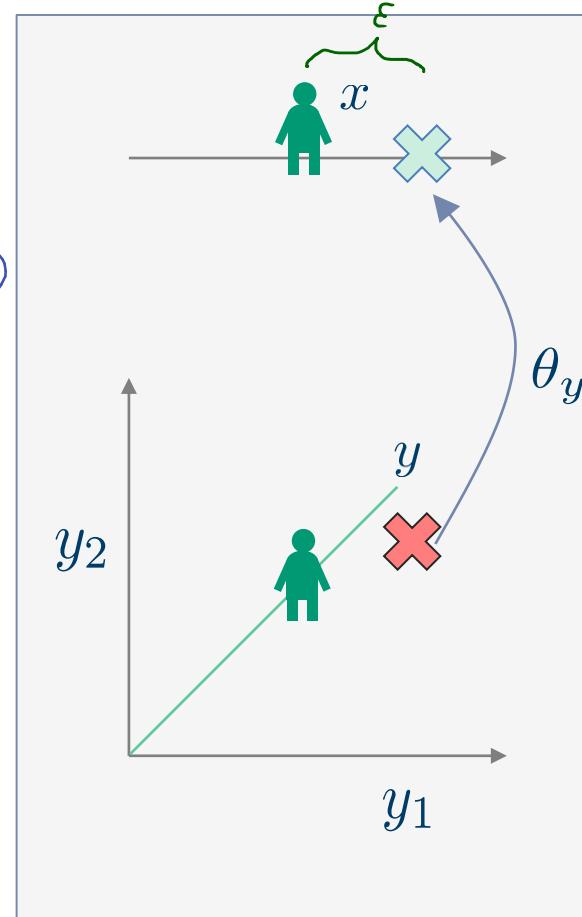
- Mean squared error:

↳ quality of an estimator

$$\text{MSE}(\theta_y) := E_y [\| \underbrace{\theta_y - x}_{\epsilon \text{ (estimation error)}} \|^2] = \text{Tr} (\text{Cov}(\theta_y))$$

mean = average squared error

- what's the average error of the estimator?
 - average over all possible worlds
- for unbiased estimators



- $\epsilon \in \mathbb{R}^n$
- Squared norm:

- Example $n = 2$: $\epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \end{bmatrix}$

$$\begin{aligned} \|\epsilon\|^2 &= [\epsilon_1, \epsilon_2] \cdot \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \end{bmatrix} \\ &= \text{Tr} \left\{ \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \end{bmatrix} \cdot [\epsilon_1, \epsilon_2] \right\} = \text{Tr} \left\{ \begin{bmatrix} \epsilon_1^2 & \epsilon_1 \epsilon_2 \\ \epsilon_1 \epsilon_2 & \epsilon_2^2 \end{bmatrix} \right\} \end{aligned}$$

(\|\epsilon\|^2) Trace: sum of diagonals

Bias-Variance Decomposition

- Bias-variance decomposition:

$$E[||\theta_y - x||^2] = \text{mean squared error} + \underbrace{\text{trace/Spur}}_{\beta} + \underbrace{||E[\theta_y] - x||^2}_{\text{Bias}}$$

(Mean-Variance-Decomposition of $\epsilon = \theta_y - x$)

- Bias: $\beta = E(\epsilon)$

$\beta = E(\epsilon) = E(\theta_y - x)$ true state (fixed/unknown)

$\beta = E[\theta_y] - x$ (usually depends on x)

- Unbiased:

matches state to all expectations

$E[\theta_y] = x$ for all possible x .

Wiederholung: Linear Estimator

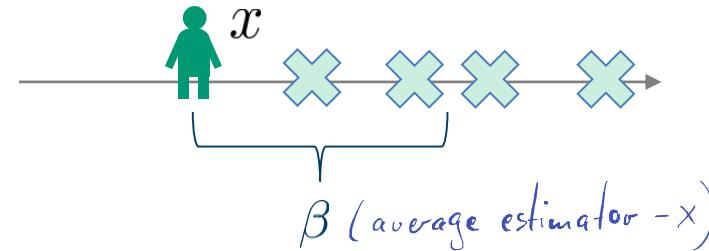
$\theta_y = Ky + b$ with $K \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$
Korrektur aus VL

$$\text{Cov}(\epsilon) = E(\epsilon \epsilon^\top) - E(\epsilon) \cdot E(\epsilon)^\top$$

$$E(\epsilon \epsilon^\top) = \text{Cov}(\epsilon) + E(\epsilon) \cdot E(\epsilon)^\top$$

$$\text{Tr}(E(\epsilon \epsilon^\top)) = \text{Tr}(\text{Cov}(\epsilon)) + ||E(\epsilon)||^2$$

Proof Bias Variance Decomposition:



BLUE

best linear unbiased estimator

- MVU estimator: minimum variance unbiased estimator
- "an estimator which is unbiased and minimizes the covariance of the error"
- "if we restrict the set of possible set of estimators to linear estimators, we get BLUE"

Mean squared error

Remember this!

- The MSE is a function of the unknown x
- For this reason, the MSE is in general difficult to minimize directly
- In case of unbiasedness:

$$\text{MSE}(\theta_y) = \text{Tr}\{\text{Cov}[\theta_y]\}$$

- ⇒ Minimum Variance Unbiased (MVU) Estimator
- ⇒ Best Linear Unbiased Estimator (BLUE) (= linear MVU)

Wiederholung:

$$\text{Cov}[z] = E[(z - E[z])(z - E[z])^T]$$

- Unbiased condition for a linear estimator $\theta_y = \mathbf{K}y + b$:

$$\mathbb{E}[\mathbf{K}y + b] = \mathbf{K}\mathbf{H}x + b$$

$\stackrel{\text{zero vector}}{\downarrow}$ $\stackrel{\mathbb{E}(\mathbf{I} + \mathbf{x} + \mathbf{e}) = \mathbf{H}x}{=} \mathbf{x}$ (for unbiasedness the mean of θ_y should correspond to x)

$\Rightarrow \mathbf{K}\mathbf{H} = \mathbf{I}$ and $b = \mathbf{0}$ ← wichtig, um zu prüfen ob estimator unbiased ist

- Covariance of a linear estimator $\theta_y = \mathbf{K}y + b$:

$$\begin{aligned} \text{Cov}[\theta_y] &= \text{Cov}[\mathbf{K}y + b] \\ &= \text{Cov}[\mathbf{K}(\mathbf{H}x + \mathbf{e}) + b] \\ &= \mathbf{K} \underline{\mathbf{C}} \mathbf{K}^T \end{aligned}$$

- Linear LS is a linear estimator:

↳ linear because it can be written as $\theta_y = Ky + b$

$$\theta_y = \underbrace{(\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y}_{=K_{LS}}$$

kommt neue place, ist
 aber unbiased

- Linear LS is unbiased: (condition: $KH = I$)

$$\begin{aligned} K_{LS} H &= (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{H} \\ &= I \end{aligned}$$

- Is linear LS also the best linear unbiased estimator?

• Trace of Covariance Matrix is minimal

$$\begin{aligned} \text{Cov}[\theta_y] &= K_{LS} C K_{LS}^T \\ &= (H^T C^{-1} H)^{-1} H^T C^{-1} C (H^T C^{-1} H)^{-1} H^T C^{-1} \\ &\quad \cancel{(H^T C^{-1} H)^{-1} H^T C^{-1} C} \quad \cancel{(H^T C^{-1} H)^{-1} H^T C^{-1}} \\ &\quad \text{first step: what is the Covariance Matrix?} \quad I \end{aligned}$$

BLUE if $\mathbf{W} = \mathbf{C}^{-1}$

am besten wenn Varianz möglichst gering

Theorem: Linear least squares is the BLUE if $\mathbf{W} = \mathbf{C}^{-1}$

- Assume $\mathbf{K} = \mathbf{K}^{LS} + \mathbf{B}$
- From unbiased:

$$\begin{aligned} & (\mathbf{K}^{LS} + \mathbf{B}) \cdot \mathbf{H} = \mathbf{I} \\ \Rightarrow & \underbrace{(\mathbf{K}^{LS} \mathbf{H} + \mathbf{B} \mathbf{H})}_{\mathbf{I}} = \mathbf{I} \\ \Rightarrow & \mathbf{B} \mathbf{H} = \mathbf{0} \end{aligned}$$

- Wurde in VL nur ange-
wissen. Nicht nötig komplett
nachzuverfolgen

- Covariance of the estimator:

$$\text{Cov}[\mathbf{K}y] = \mathbf{KCK}^T = \mathbf{K}_{LS}\mathbf{CK}_{LS}^T + \mathbf{K}_{LS}\mathbf{CB}^T + \mathbf{BCK}_{LS}^T + \mathbf{BCB}^T$$

- We know $\mathbf{BCK}_{LS}^T = \mathbf{BCC}^{-1}\mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-T} = \mathbf{0}$ as $\mathbf{BH} = \mathbf{0}$
- Hence, $\text{Cov}[\mathbf{K}y] = \underbrace{\mathbf{K}_{LS}\mathbf{CK}_{LS}^T}_{\text{Cov}[\mathbf{K}_{LS}y]} + \underbrace{\mathbf{BCB}^T}_{>0}$

Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 4

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de

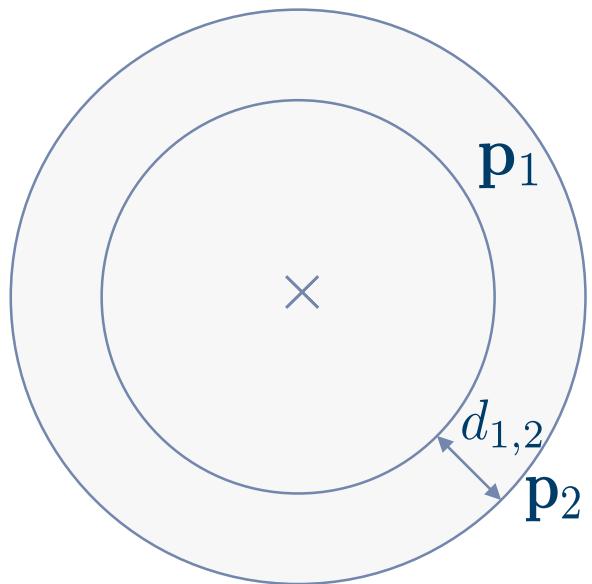


GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 3 solution
- Theory - TDOA closed-form
- Problem 4.1 - Silly estimator
- Problem 4.2 - Polynomial Regression
- Homework 4 presentation

Can you explain how Time-difference-of-Arrival works?



$$d_{1,2} = \| \mathbf{x} - \mathbf{p}_2 \| - \| \mathbf{x} - \mathbf{p}_1 \| + e$$

- no synchronized clocks between sender & receiver needed
- but synchronized clocks needed between sensors

What is an estimator?

An estimator is a function $\theta_y : \mathbb{R}^m \rightarrow \mathbb{R}^n$ providing an estimate for the desired variable x based on an observation y . A linear estimator has the form

$$\hat{\theta}_y = \mathbf{K}y + \mathbf{b} .$$

The quality of an estimator can be measured using the mean squared error (MSE)

$$MSE(\theta_y) = E[||\hat{\theta}_y - x||^2] .$$

How can the MSE be decomposed?

Note: Fisher approach, so no statistical information about \mathbf{x} .

$$\begin{aligned} E[||\theta_y - \mathbf{x}||^2] &= E[\text{Tr}(\theta_y - \mathbf{x})(\theta_y - \mathbf{x})^T] \\ &= \text{Tr } E[\theta_y^2 - 2\theta_y \mathbf{x} + \mathbf{x}^2] \\ &\stackrel{\text{wichtig: Betrag}}{=} \text{Tr } E[\theta_y^2] - 2E[\theta_y]\mathbf{x} + \mathbf{x}^2 + E[\theta_y]^2 - E[\theta_y]^2 \\ &\stackrel{\text{innerhalb von } E}{=} \text{Tr } E[\theta_y^2] - E[\theta_y]^2 + \text{Tr } E[\theta_y]^2 - 2E[\theta_y]\mathbf{x} + \mathbf{x}^2 \\ &= \text{Tr Cov}[\theta_y] + \underset{\text{Covariance}}{\parallel E[\theta_y] - \mathbf{x} \parallel^2} \\ &\quad \underset{\text{Bias}}{\parallel} \\ &\stackrel{\text{wichtig: Betrag außerhalb von } E}{=} \end{aligned}$$

Look again at the GPS setting of last homework (satellites are 20200km above mean sea level, each satellite broadcasts its location in spherical coordinates $[\theta, \phi, r]^T$ plus the emission time (see figures below), assume that the speed-of-light is $c = 3 \cdot 10^8 \text{ m s}^{-1}$ and the Earth is an ideal sphere with radius $r^E = 6370\text{km}$)

This time, the device only receives the first three satellite signals at time $t = 0\text{s}$:

$$p_1 = [0^\circ, 40^\circ, 20200\text{km}]^T,$$

$$p_2 = [10^\circ, 20^\circ, 20200\text{km}]^T,$$

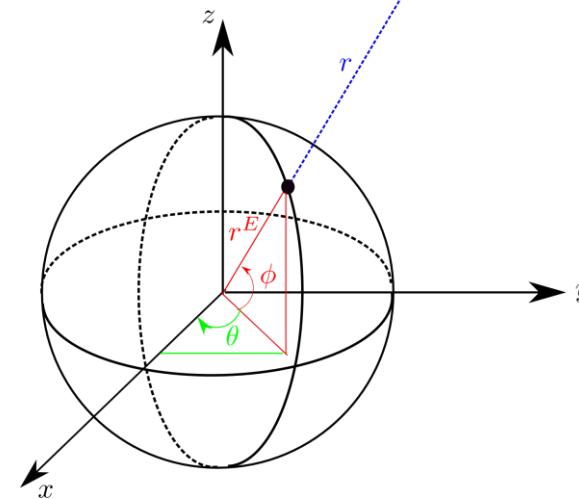
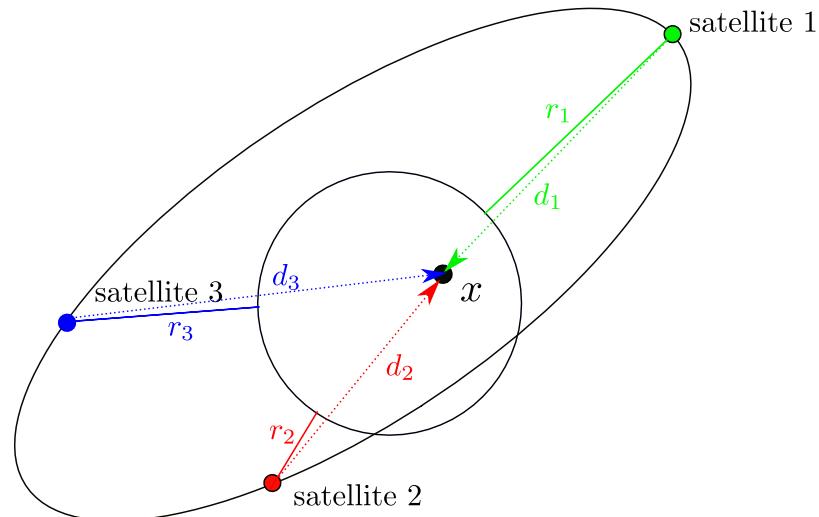
$$p_3 = [10^\circ, -10^\circ, 20200\text{km}]^T,$$

$$t_1 = -67.603\text{ms},$$

$$t_2 = -70.102\text{ms},$$

$$t_3 = -78.690\text{ms}.$$

*negativ weil: times
display when signal
was emitted (was
in past)*



- a) Would the reformulation from last time still work? If not, explain why.
- b) Instead of reformulating the measurement equation into a linear equation, implement the Gauss-Newton method to calculate the least squares solution.
- c) Alternatively, use the Bancroft solution to determine the position of the GPS device.

- Squared meas. equation:

$$\begin{aligned}
 d_i^2 &= \|x - p_i\|^2 + e_i^* \\
 &= (x_1 - p_{i,1})^2 + (x_2 - p_{i,2})^2 + e_i^* \\
 &= -2x_1 p_{i,1} - 2x_2 p_{i,2} + \|p_i\|^2 + R^2 + e_i^*
 \end{aligned}$$

with $R^2 := \|x\|^2 = (x_1)^2 + (x_2)^2$

- Linear measurement equation for given R^2 :

$$y = \mathbf{H}_1 x + \mathbf{H}_2 R^2 + e^*$$

with

$$y = \begin{bmatrix} d_1^2 - \|p_1\|^2 \\ \vdots \\ d_m^2 - \|p_m\|^2 \end{bmatrix}, \quad \mathbf{H}_1 = \begin{bmatrix} -2p_{1,1} & -2p_{1,2} \\ \vdots & \vdots \\ -2p_{m,1} & -2p_{m,2} \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

- Least squares solution for a fixed R^2 :

$$\begin{aligned}x^{LS}(R^2) &= (\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T (y - \mathbf{H}_2 R^2) \\&= z_1 + R^2 z_2\end{aligned}$$

with $z_1 := (\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T y$ and $z_2 := -(\mathbf{H}_1^T \mathbf{H}_1)^{-1} \mathbf{H}_1^T \mathbf{H}_2$

- What is R^2 ?

$$\begin{aligned}R^2 &= \|x^{LS}(R^2)\|^2 \\&= (z_1 + R^2 z_2)^T \cdot (z_1 + R^2 z_2)\end{aligned}$$

- Solve the following quadratic equation for R^2 :

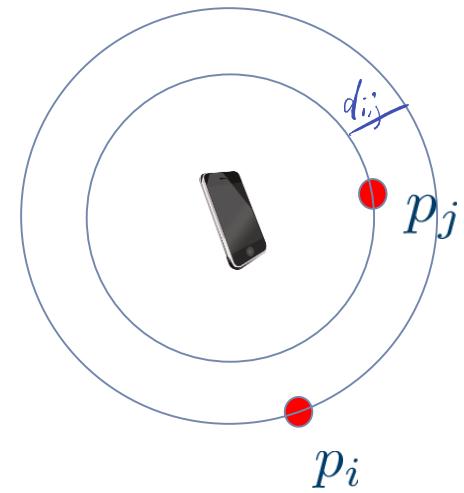
$$0 = z_1^T z_1 + z_1^T z_2 R^2 + R^2 z_2^T z_1 + (R^2)^2 z_2^T z_2 - R^2$$

Desired: Sender position $x = [x_1, x_2]^T \in \mathbb{R}^2$

Given: $\binom{m}{2}$ TDOAs to m landmarks:

- position $p_i = [p_{i,1}, p_{i,2}]^T \in \mathbb{R}^2$
- time difference $\Delta T_{i,j}$
- distance difference $d_{i,j} = c \cdot \Delta T_{i,j}$ with wave speed c

with $i, j \in \{1, \dots, m\}$



Measurement equation:

$$d_{i,j} = \underbrace{\|x - p_i\|}_{D_i} - \underbrace{\|x - p_j\|}_{D_j} + e_i$$

*time difference
of arrival (small d)*

- Define

$$(R^x)^2 := \|x\|^2 = (x_1)^2 + (x_2)^2$$

and

$$(R_i^p)^2 := \|p_i\|^2 = (p_{i,1})^2 + (p_{i,2})^2$$

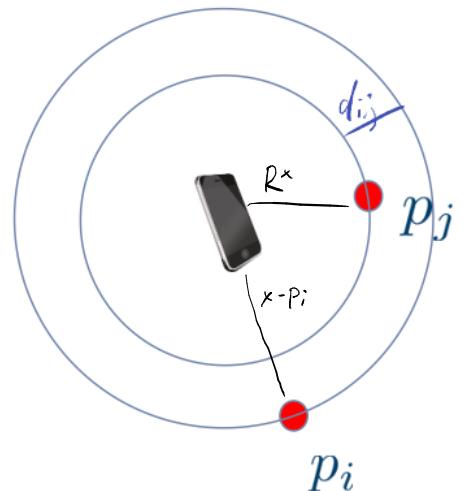
- Assume $p_1 = [0, 0]^T$; hence, $D_1 = R^x$ and $R_1^p = 0$.
- From

$$\begin{aligned} (d_{i,1} + R^x)^2 &= \|x - p_i\|^2 \\ &= (R^x)^2 + (R_i^p)^2 - 2p_i^T x \end{aligned}$$

we get

$$2(p_i)^T x + 2d_{i,1}R^x = (R_i^p)^2 - d_{i,1}^2$$

can be solved with Banerjee solution
γ: measurement



- Stacked measurement equation

where

$$y = \mathbf{H}_1 x + \mathbf{H}_2 R^x + e^* ,$$

Wichtig: man startet mit R_2 (we lose meas. from P_1)

$$y = \begin{bmatrix} (R_2^p)^2 - d_{2,1}^2 \\ \vdots \\ (R_m^p)^2 - d_{m,1}^2 \end{bmatrix}, \quad \mathbf{H}_1 = \begin{bmatrix} 2p_2^T \\ \vdots \\ 2p_m^T \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 2d_{2,1} \\ \vdots \\ -2d_{m,1} \end{bmatrix}$$

- Linear least squares solution: $x^{LS}(R^x)$
- Use $(R^x)^2 = \|x^{LS}(R^x)\|^2$ to get R^x

Problem 4.1 – Silly Estimator

Erwartungswert
Varianz

Given is a random variable $y \sim \mathcal{N}(x, 1)$. The objective is to estimate the deterministic mean x based on a single observation y . We consider two estimators:

- Natural estimator: $\theta_n(y) = y$
- Silly estimator: $\theta_s(y) = 0$

- a) Calculate the bias, variance, and MSE of both estimators.
- b) In which cases outperforms the silly estimator the natural estimator?

$$\theta_n : \beta = E[\theta_n(y)] - x = x - x = 0$$

siehe Erwartungswert in
↓ Det. von \mathcal{N}

$$\text{Var}[\theta_n(y)] = \text{Var}[y] = 1$$

$$MSE(\theta_n(y)) = 1 + 0 = 1$$

$$\theta_s : \beta = E[\theta_s(y)] - x = 0 - x = -x$$

$$\text{Var}[\theta_s(y)] = 0$$

$$MSE(\theta_s(y)) = 0 + || -x ||^2 = x^2$$

Wiederholung:

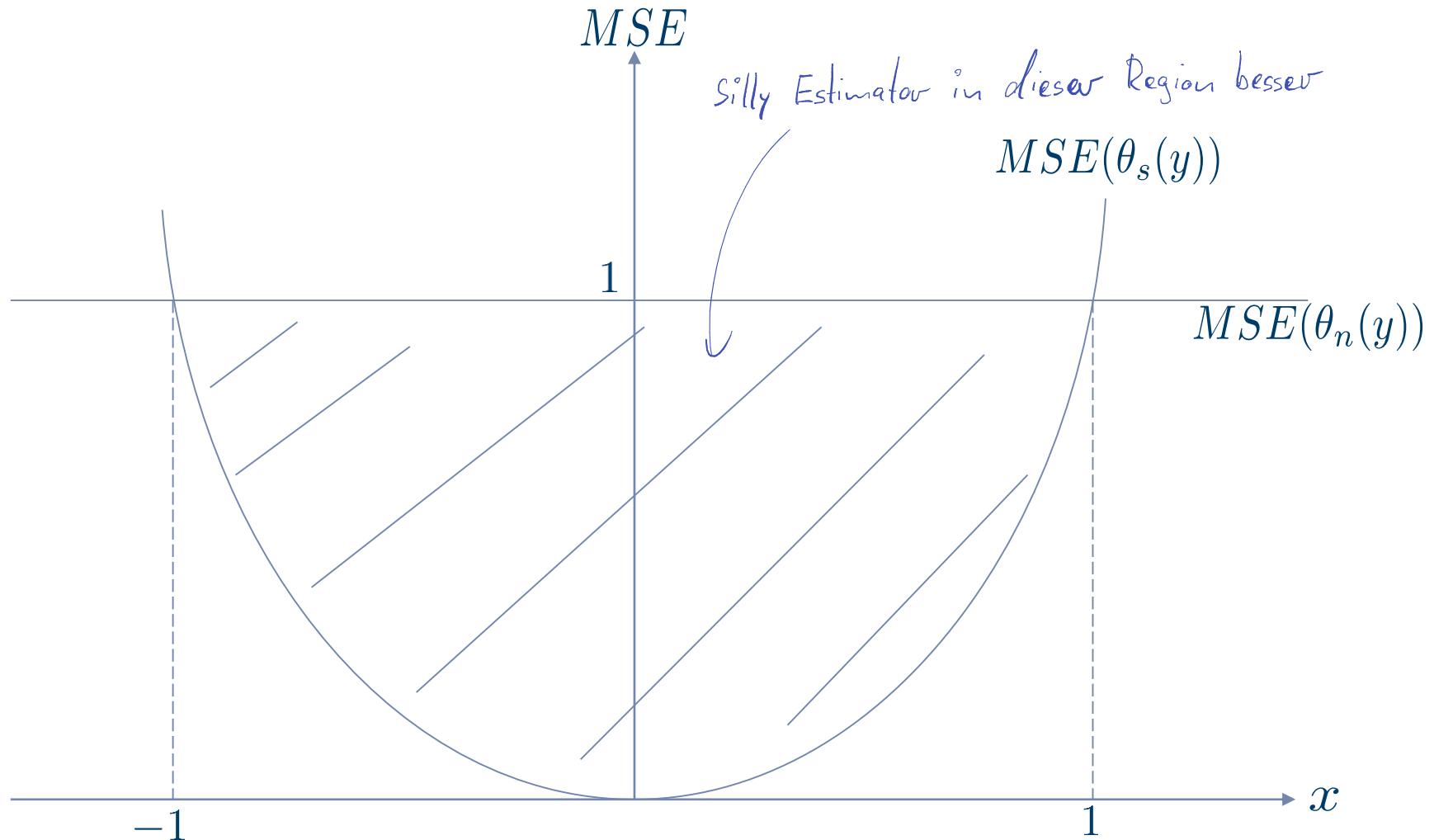
- Bias-variance decomposition:

$$E[||\theta_y - x||^2] = \underbrace{\text{Tr}[\text{Cov}[\theta_y]]}_{\substack{\text{mean squared} \\ \text{error}}} + \underbrace{||E[\theta_y] - x||^2}_{\substack{\text{trace / Spur} \\ \beta \text{ Bias}}}$$

(Mean-Variance-Decomposition of $\epsilon = \theta_y - x$)

- Bias: $\beta = E(\epsilon)$ true state (fixed/unknown)
- $\beta = E[\theta_y] - x$ (usually depends on x)
- Unbiased: $E[\theta_y] = x$ for all possible x .

Problem 4.1: Solution



Problem 4.2 - Polynomial Regression

Given are m data pairs $(y_i, x_i) \in \mathbb{R}^2$, $i = 1, \dots, m$. The objective is to find a polynomial function

$$f_a(x) := a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_nx^n$$

with $a = [a_0, \dots, a_n]^T$ that fits the data as good as possible, i.e., $\sum_i \|y_i - f_a(x_i)\|^2$ should be minimal.

- Formulate the fitting problem as a linear Least Squares (LS) problem, i.e., write down the linear measurement equation for this problem.
- Consider the values

i	1	2	3	4	5
x_i	1	2	5	7	8
y_i	4	1	2	3	4

i	1	2	3	4	5
x_i	1	2	5	7	8
y_i	4	1	2	3	4

i	1	2	3	4	5
x_i	1	2	5	7	8
y_i	4	1	2	3	4

i	1	2	3	4	5
x_i	1	2	5	7	8
y_i	4	1	2	3	4

$n=3$
 \nwarrow
 please perform a quadratic curve fit.

- Now we would like to perform a line fit. Assume the fitting passes through the origin, i.e., the equation of this line is $y = kx$. Given the two measured points $P_1 = [1.7 \ 0]^T$, $P_2 = [2.8 \ 1]^T$, please find the line using least squares. Visualize the fitted line and the column space spanned by the measurement matrix.

darauf least squares
anwenden

$$y = Hx + \epsilon$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & \cdots & & x_m^n \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} + \epsilon$$

Problem 4.2: Solution



$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_y = \underbrace{\begin{bmatrix} 1 & x_1 & \dots & x_1^n \\ 1 & x_2 & \dots & x_2^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & \dots & x_m^n \end{bmatrix}}_{\mathbf{H}} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} + \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_n \end{bmatrix}$$

Wiederholung:

⇒ Normal equation:

- As \mathbf{H} has full column rank:

$$x^{\text{LS}} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y$$

$$\boxed{\mathbf{H}^T \mathbf{W} y = \mathbf{H}^T \mathbf{W} \mathbf{H} x}$$

auf andere Seite ziehen

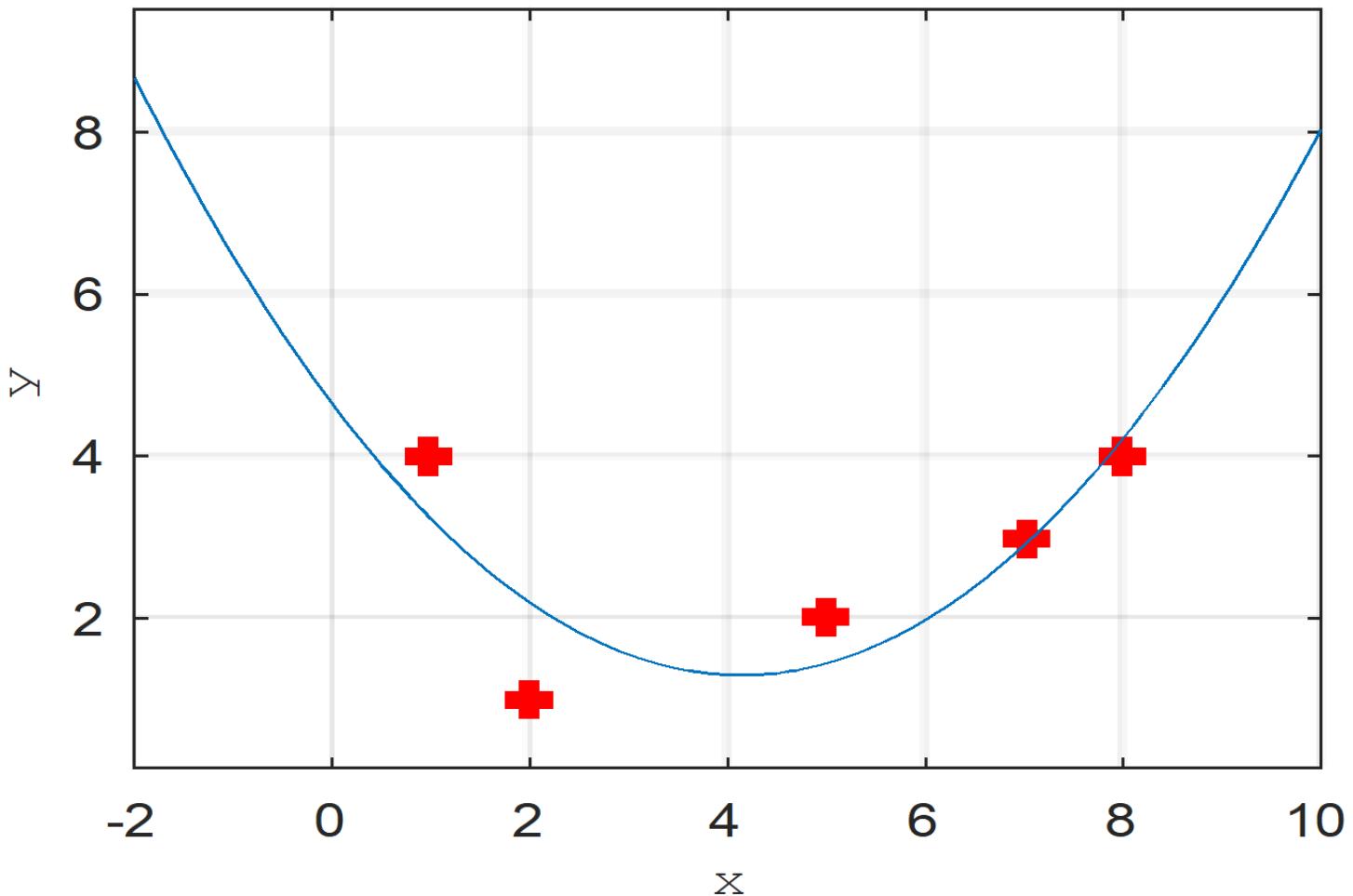
$$\underbrace{\begin{bmatrix} 4 \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}}_y = \underbrace{\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 5 & 25 \\ 1 & 7 & 49 \\ 1 & 8 & 64 \end{bmatrix}}_{\mathbf{H}} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}$$

$$a^{\text{LS}} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T y = \begin{bmatrix} 4.64740 \\ -1.61753 \\ 0.19557 \end{bmatrix}$$

$$f_a(x) = 4.64740 - 1.61753x + 0.19557x^2$$

Problem 4.2: Solution

Points and fitted line



Problem 4.2: Solution

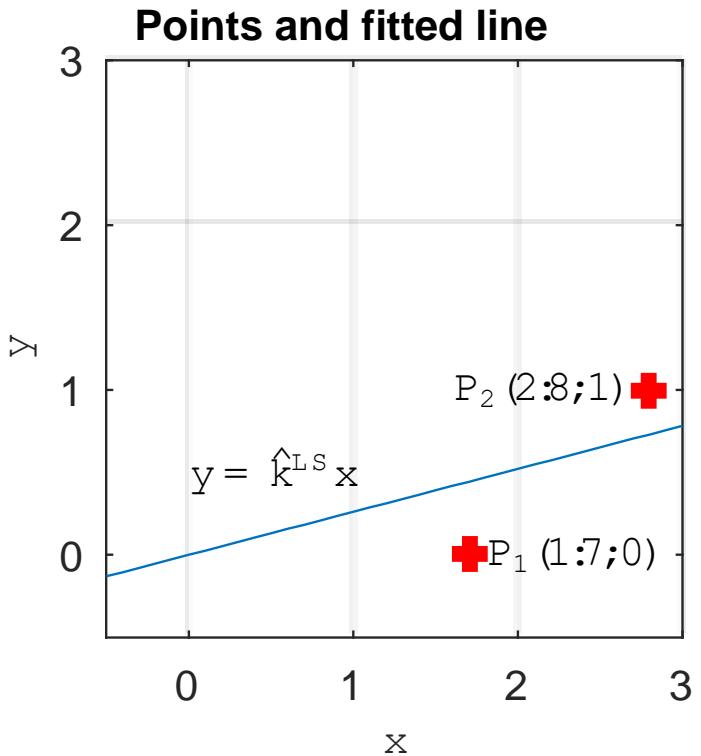
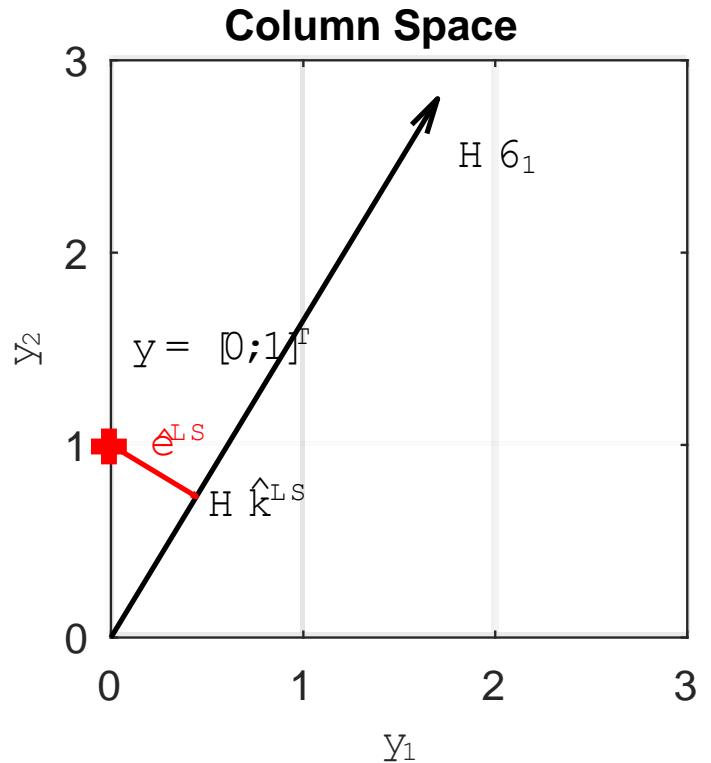


$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.7 \\ 2.8 \end{bmatrix} k + e$$

$$k^{\text{LS}} = ([1.7 \quad 2.8] \begin{bmatrix} 1.7 \\ 2.8 \end{bmatrix})^{-1} [1.7 \quad 2.8] \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$k^{\text{LS}} = 0.26095$$

Problem 4.2: Solution



Given are two one-dimensional (uncorrelated) sensor readings $y_1 \in \mathbb{R}$ and $y_2 \in \mathbb{R}$ of a one-dimensional location $x \in \mathbb{R}$. The variances of the errors are $\sigma_1^2 = 2$ for the first and $\sigma_2^2 = 3$ for the second sensor.

- a) Formulate the (joint) measurement equation for the fusion of the two measurements.
- b) Write a function that simulates $m = 1000$ measurements using the measurement model in a) with true $x = 2$.
- c) For each simulated measurement, calculate the weighted least squares (WLS) solution using $W^{-1} = \begin{pmatrix} 2 & 0 \\ 0 & c \end{pmatrix}$ with $c = 1, \dots, 10$. Which value for c provides the BLUE estimate?
- d) Derive an analytic formula for the variance of the weighted least squares solution, using the W providing the BLUE estimate.
- e) Check if the analytical solution coincides with the empirical variance of the WLS solution from c).

Sensor Data Fusion

Fundamental fusion formulas

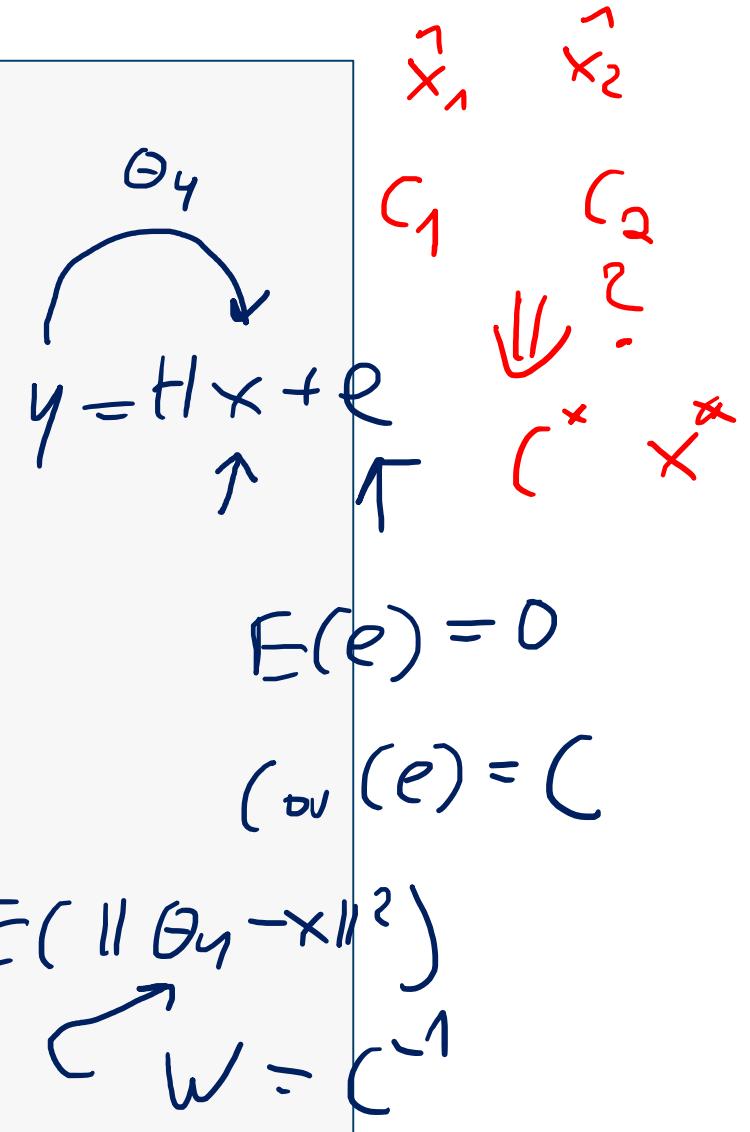
www.fusion.informatik.uni-goettingen.de



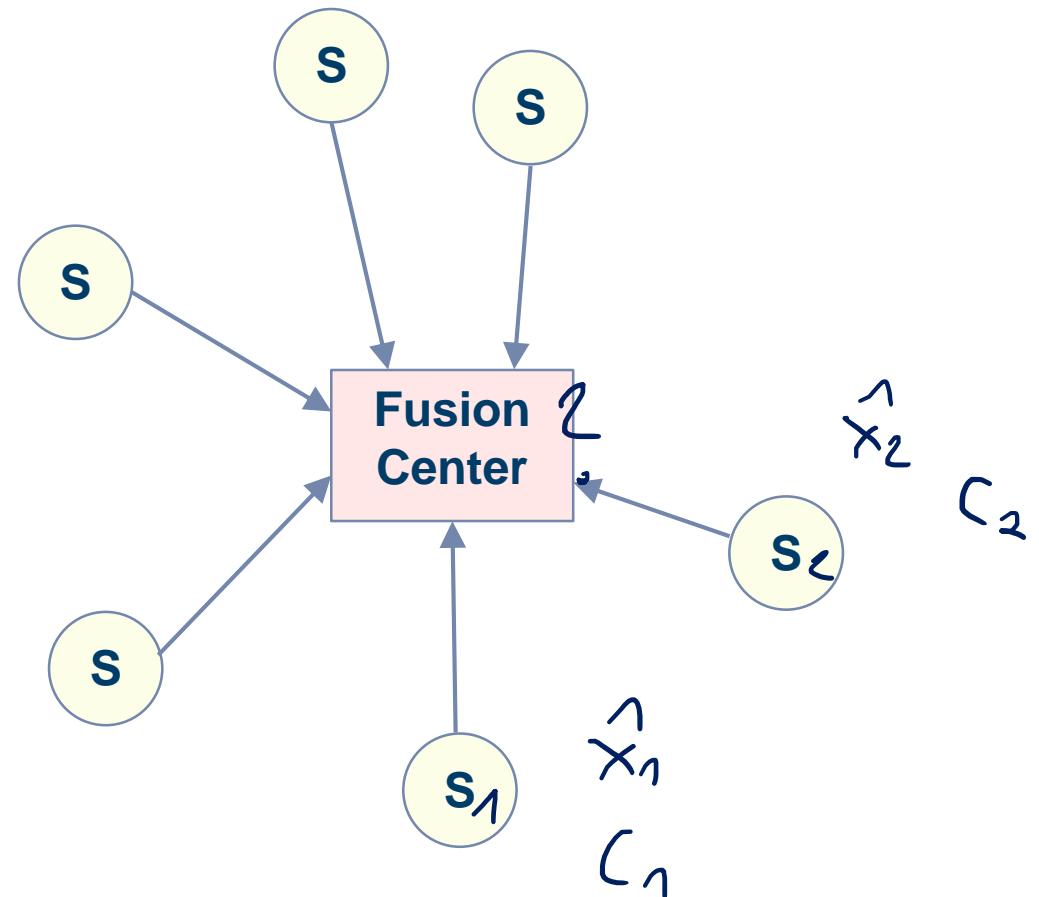
GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (live)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (live)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (live)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Tracking (live)
12. Advanced Topic
13. Advanced Topic (2)
14. Summary and Discussion (live)

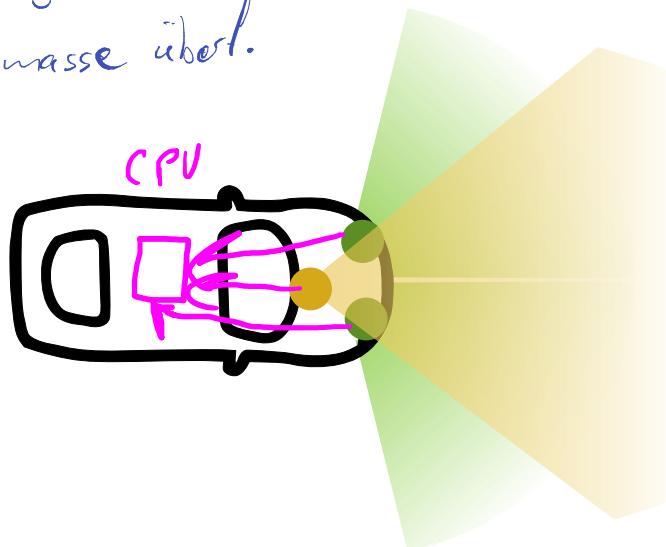


$y = t(x) + e$
 Θ_y
 $E(e) = 0$
 $C(e) = C$
 $E(\| \Theta_y - x \|^2)$
 $w = C^{-1}$
 x_1, x_2
 c_1, c_2



- Central fusion center
- Sensors communicate sensor data to fusion center

- nicht sehr robust
- Datenleitungen können mit Datenmasse über. sein



Fusion of Uncorrelated Measurements

Measurement $\hat{x}_1 \in \mathbb{R}^n$ with σ_1 and $E(\hat{x}_1 - x) = 0$

Measurement $\hat{x}_2 \in \mathbb{R}^n$ with σ_2 and $E(\hat{x}_2 - x) = 0$

e_1 and e_2
uncorrelated

Meas. eqn.: $\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = H \begin{bmatrix} I \\ I \end{bmatrix} x + \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}$

$C = \text{Cov}(\begin{bmatrix} e_1 \\ e_2 \end{bmatrix}) = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$
↳ joint covariance matrix

LS \Rightarrow inserting into LS $x^* = (G_1^{-1} + G_2^{-1})^{-1} \cdot (G_1^{-1} \hat{x}_1 + G_2^{-1} \hat{x}_2)$

$W = C^{-1}$
↳ because it gives the best solution

$$C^* = (H^T \cdot C^{-1} \cdot H)^{-1} = \left(\begin{bmatrix} I \\ I \end{bmatrix}^T \cdot \begin{bmatrix} \sigma_1^{-2} & 0 \\ 0 & \sigma_2^{-2} \end{bmatrix} \begin{bmatrix} I \\ I \end{bmatrix} \right)^{-1}$$

$$= (C^{-1} I I) \begin{bmatrix} \sigma_1^{-2} \\ \sigma_2^{-2} \end{bmatrix}^{-1} = (\sigma_1^{-2} + \sigma_2^{-2})^{-1}$$

- Is linear LS also the best linear unbiased estimator?

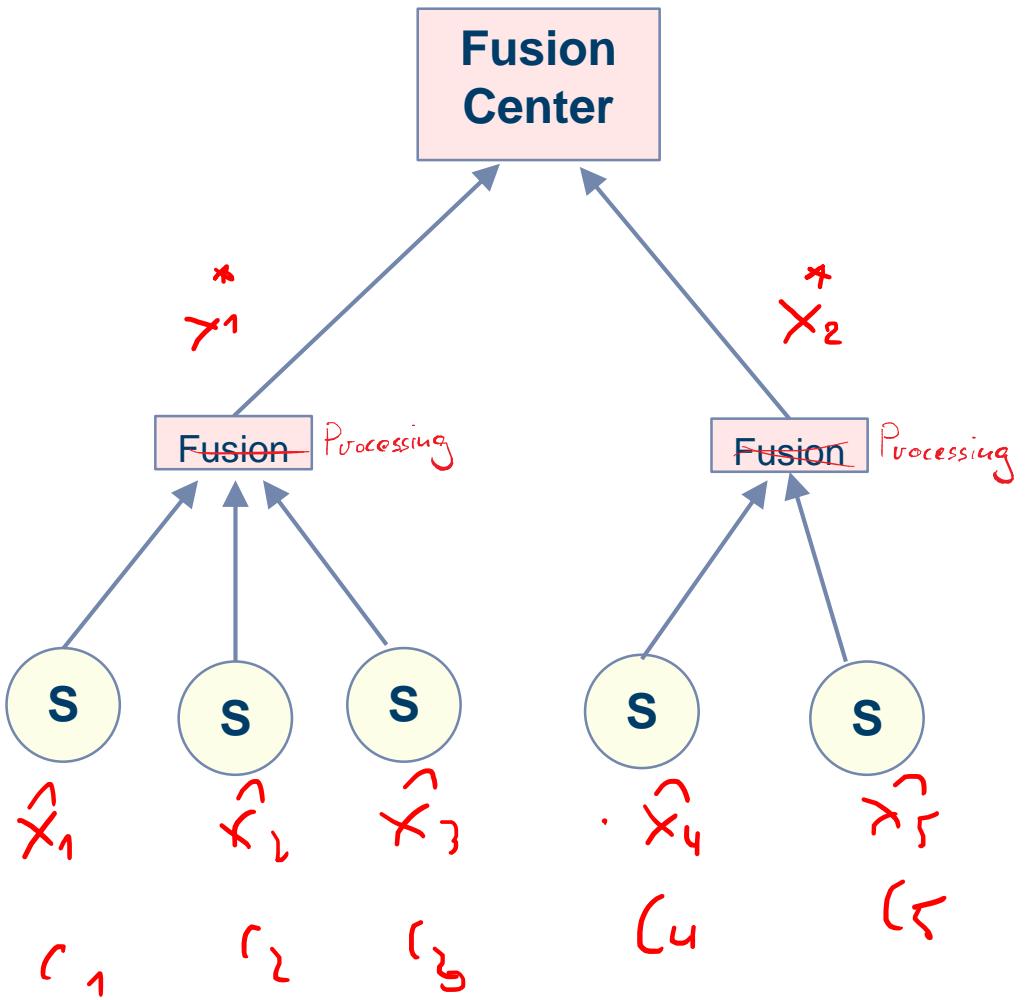
Trace of Covariance Matrix is minimal

$$\text{Cov}[\theta_y] = K_{LS} C K_{LS}^T$$

$$= (H^T C^{-1} H)^{-1} H^T C^{-1} C (H^T C^{-1} H)^{-1} H^T C^{-1} T$$

first step: what is the Covariance Matrix?

BLUE if $W = C^{-1}$

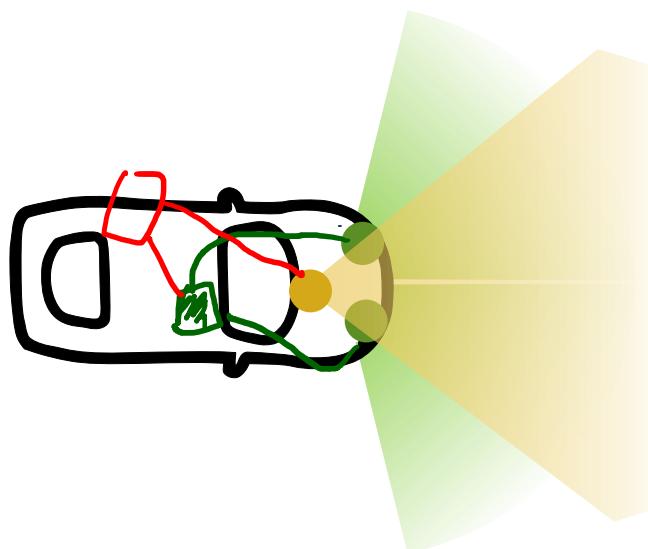


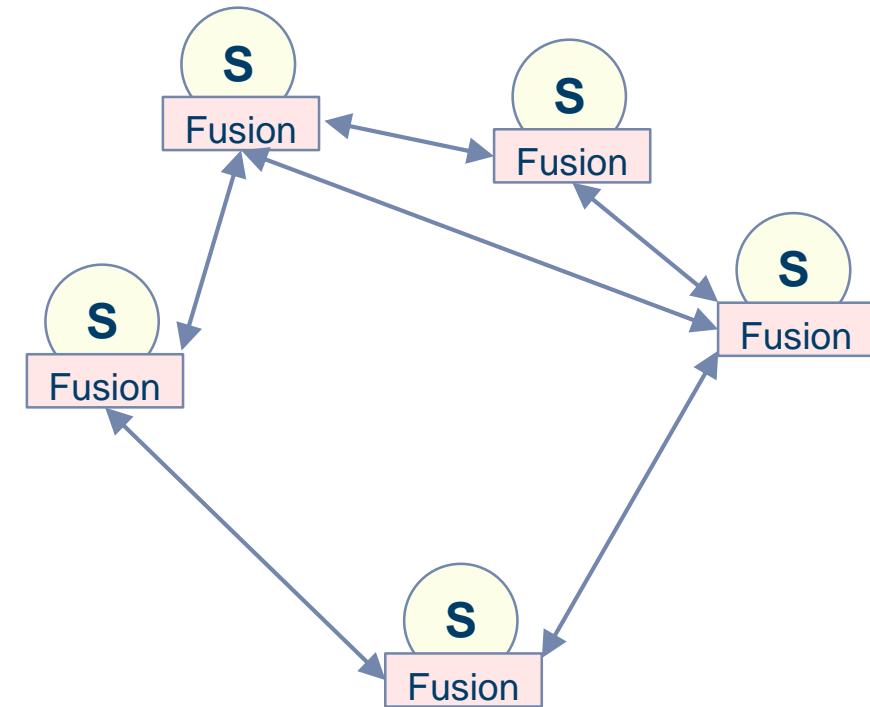
Properties:

1. Several computing nodes
2. Data is processed on nodes
3. Central node required for final result

Advantages:

- Scalable





Properties:

1. No central fusion center
2. Each node fuses data
2. Node-to-node communication
3. Network topology unknown to sensors

Advantages:

- Scalable
- Dynamic topology
- Robust against failures

Challenges:

- Rumor problem, i.e., double-counting of information

Covariance Bounds (1)

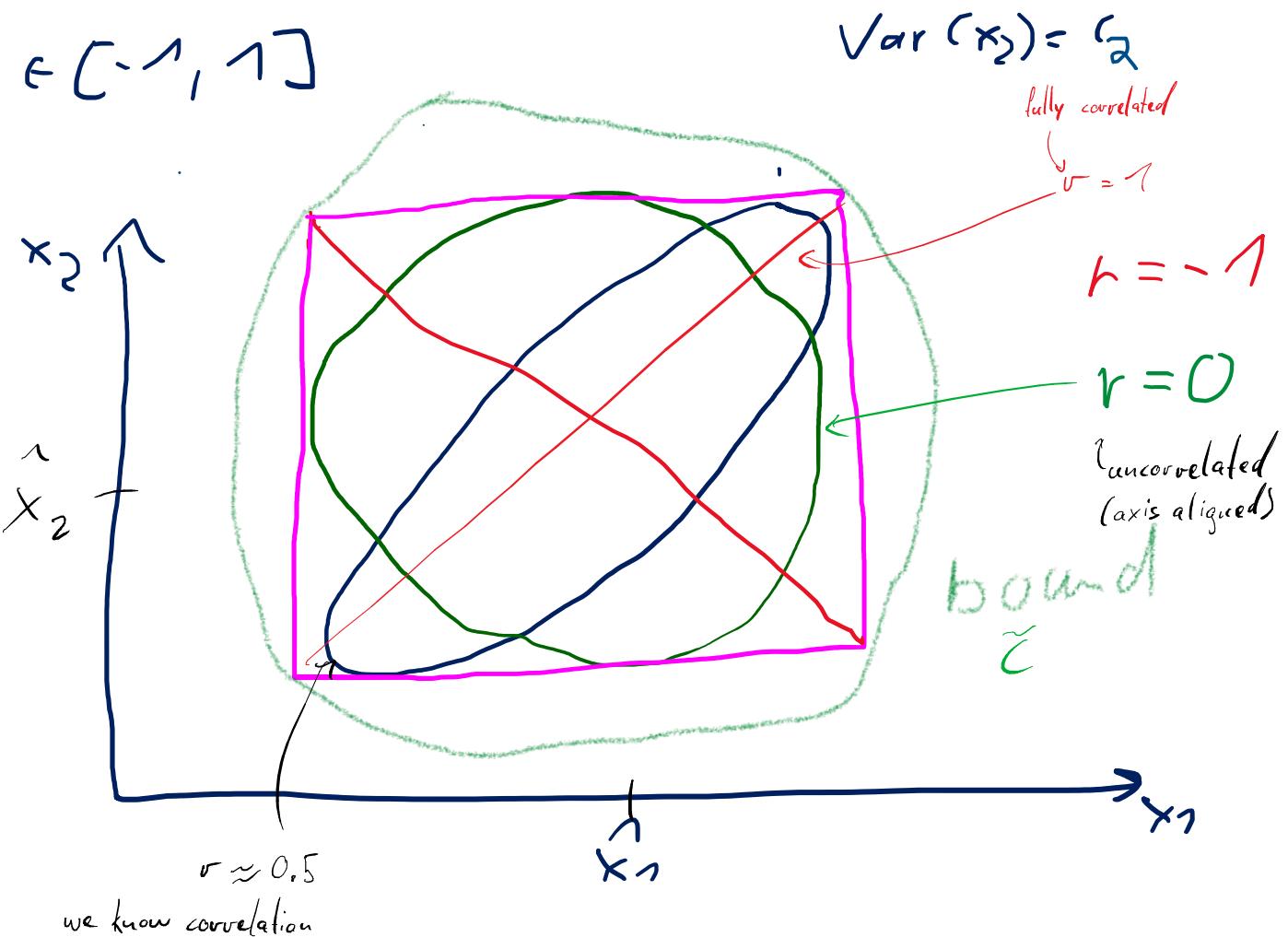
- Two scalar RV x_1, x_2 $E(x_1) = \hat{x}_1$ $E(x_2) = \hat{x}_2$ $\text{Var}(x_1) = G_1$

- Correlation coefficient $r \in [-1, 1]$

$$\text{Cov}(x_1, x_2) = \begin{bmatrix} G_1 & r \cdot \sqrt{G_1 G_2} \\ r \cdot \sqrt{G_1 G_2} & G_2 \end{bmatrix}$$

correlation coefficient

- if unknown = \tilde{C}
- Kreis umfasst alle anderen Ellipsen



$$C \geq \tilde{C} \iff \underline{C - \tilde{C}} \quad \text{pos. semi def.}$$

equivalent to

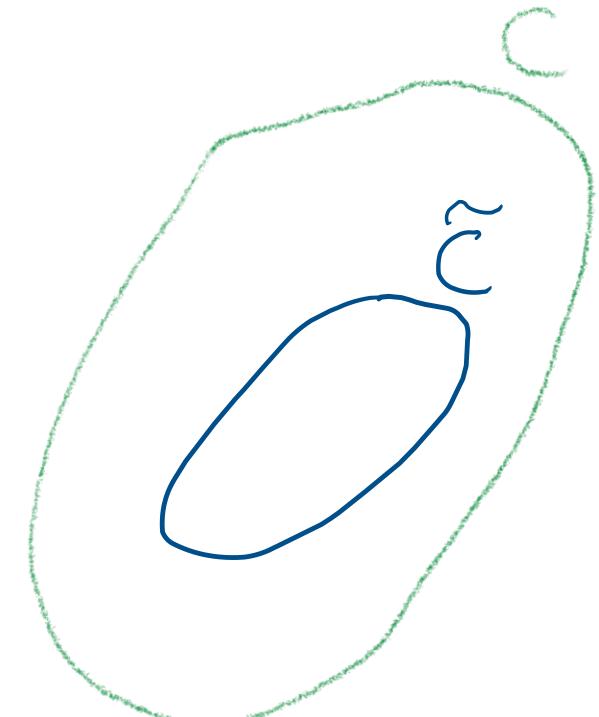
$$\iff \tilde{C}^{-1} \geq C^{-1}$$

$$x^T (\tilde{C}^{-1} - C^{-1}) x \geq 0 \quad \text{f.a. } x$$

$$\Downarrow$$

$$x^T \tilde{C}^{-1} x \geq x^T C^{-1} x$$

$$x^T \tilde{C}^{-1} x \leq 1 \Rightarrow x^T C^{-1} x \leq 1$$

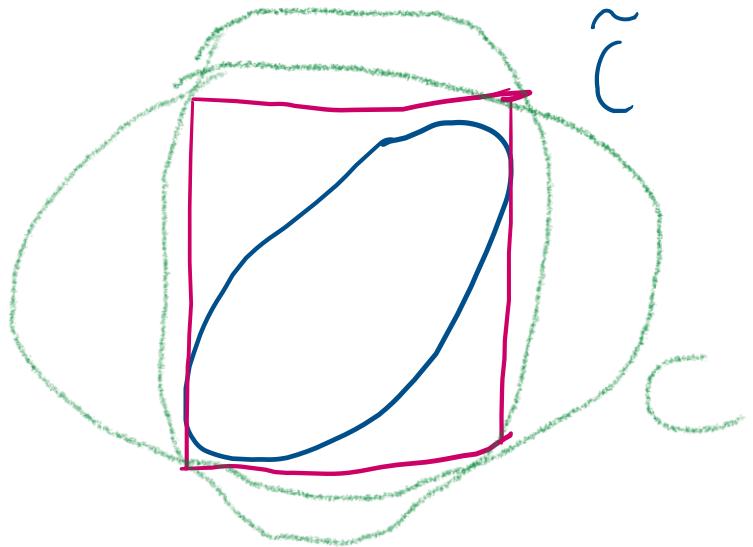


Covariance Bounds (3)

$$\text{Covariance } \tilde{C} = \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}$$

\Rightarrow A „large“ matrix C with $C \geq \tilde{C}$ is given by

$$C = \begin{bmatrix} \frac{1}{0.5-\alpha} C_{xx} & 0 \\ 0 & \frac{1}{0.5+\alpha} C_{yy} \end{bmatrix} \quad \begin{array}{l} \text{f.a. } \alpha \in (-0.5, 0.5) \\ \text{for any } \end{array}$$



Covariance Bounds (4)

Proof:

$$C - \tilde{C} = \begin{bmatrix} \left(\frac{1}{0.5-\alpha} - 1\right) \cdot C_{xx} & -C_{xy} \\ -C_{yx} & \left(\frac{1}{0.5+\alpha} - 1\right) C_{yy} \end{bmatrix} = \begin{bmatrix} \frac{0.5+\alpha}{0.5-\alpha} C_{xx} & C_{xy} \\ -C_{yx} & \frac{0.5-\alpha}{0.5+\alpha} C_{yy} \end{bmatrix}$$

$$Q = \begin{bmatrix} x^T & y^T \end{bmatrix} \begin{bmatrix} \lambda C_{xx} & -C_{xy} \\ -C_{yx} & \frac{1}{\lambda} C_{yy} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \lambda = \frac{0.5+\alpha}{0.5-\alpha} \in (0, \infty)$$

$$= \left[\sqrt{\lambda} x^T - \frac{1}{\sqrt{\lambda}} y^T \right] \underbrace{\begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}}_{\tilde{C}} \begin{bmatrix} \sqrt{\lambda} x \\ -\frac{1}{\sqrt{\lambda}} y \end{bmatrix} \geq 0$$

Measurement Fusion with unknown Correlation



Meas. \hat{x}_1 with C_1
Meas \hat{x}_2 with C_2

$$\text{Cov} \begin{pmatrix} p_1 \\ e_2 \end{pmatrix} = \begin{bmatrix} C_1 & ? \\ ? & C_2 \end{bmatrix}$$

$$\leq \begin{bmatrix} \frac{1}{0.5+2} C_1 & 0 \\ 0 & \frac{1}{0.5+2} C_2 \end{bmatrix}$$

Apply LS with bounded covariance

$$\Rightarrow \hat{x}^*(\omega) = \dots$$

$$C^*(\omega) = \dots$$

$\lambda = 2$
 $\text{Trace}(C^*) \rightarrow \text{minimal}$

Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 5

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 4 solution
- Example - LS as an estimator
- Problem 5 - non-Gaussian Noise
- Homework 5 presentation

What is BLUE?

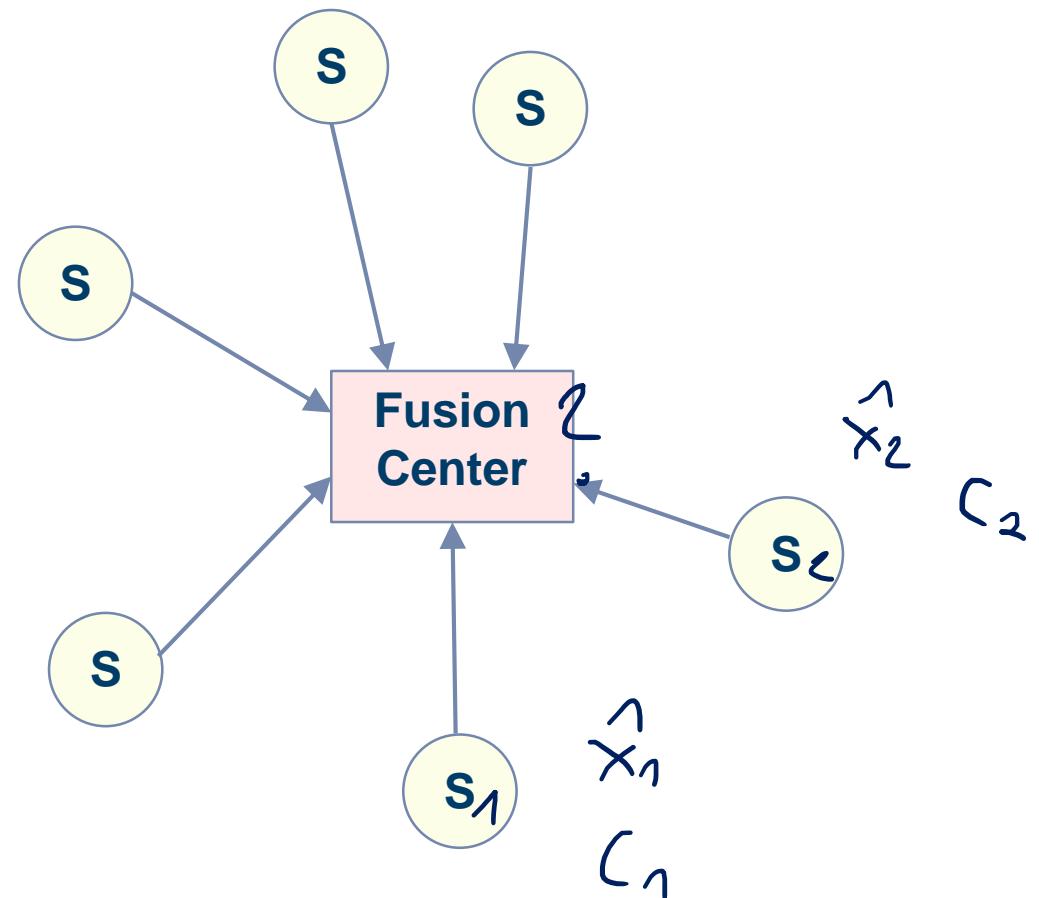
- Quality of an estimator measured via MSE
- Goal: Minimum MSE Estimator
- Problem: unknown \mathbf{x}
- Solution: assume unbiasedness

$$MSE(\theta_y) = \text{Tr Cov}[\theta_y] + \|\mathbf{E}[\theta_y] - \mathbf{x}\|^2$$

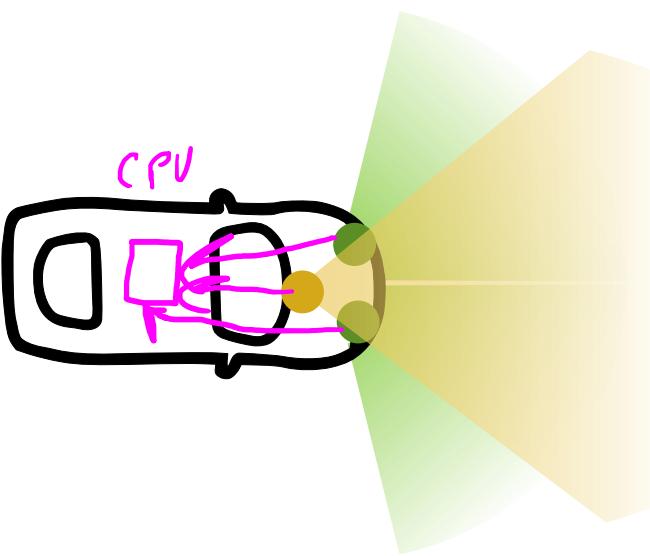
- New goal: Minimum Variance Unbiased (MVU) Estimator
- If MVU estimator is linear, we have the Best Linear Unbiased Estimator (BLUE)

What are types of sensor network topologies?

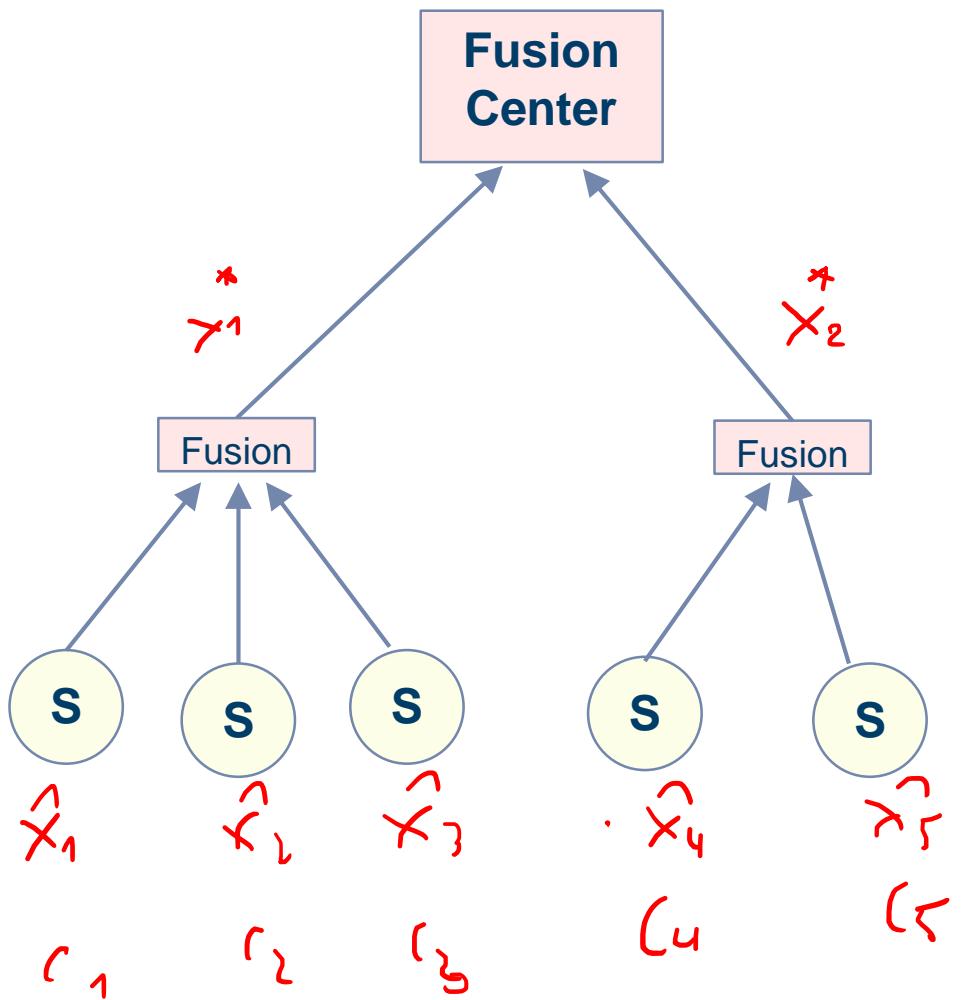
Centralized Fusion Architecture



- Central fusion center
- Sensors communicate sensor data to fusion center



Distributed Fusion Architecture

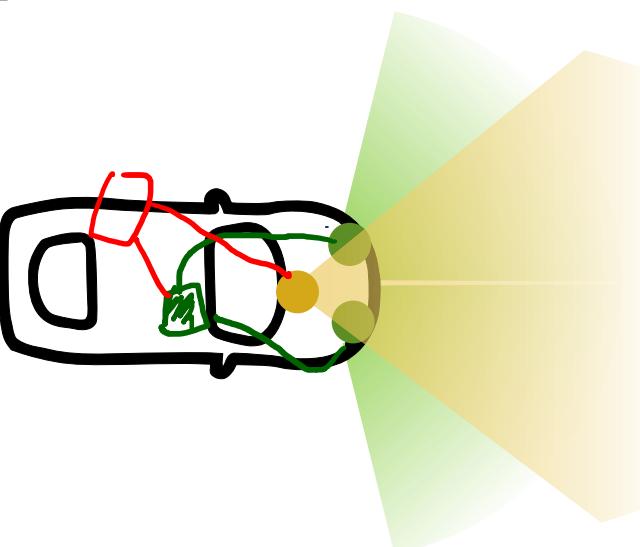


Properties:

1. Several computing nodes
2. Data is processed on nodes
3. Central node required for final result

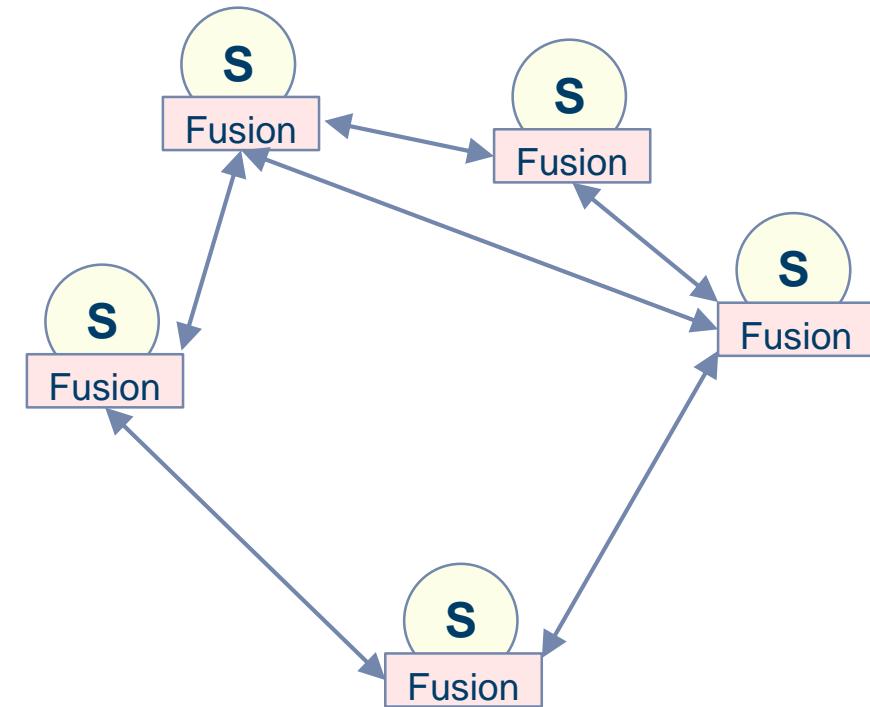
Advantages:

- Scalable



Decentralized Fusion

PREVIOUS
LECTURE



Properties:

1. No central fusion center
2. Each node fuses data
3. Network topology unknown to sensors

Advantages:

- Scalable
- Dynamic topology
- Robust against failures

Challenges:

- Rumor problem, i.e., double-counting of information

How can unknown correlation between estimates be handled?

Find uncorrelated ellipse encapsulating correlated one.

$$\tilde{\mathbf{C}} = \begin{bmatrix} \mathbf{C}^{xx} & \mathbf{C}^{xy} \\ \mathbf{C}^{yx} & \mathbf{C}^{yy} \end{bmatrix}$$

$$\mathbf{C} > \tilde{\mathbf{C}}$$

$$\mathbf{C} = \begin{bmatrix} \frac{1}{0.5-\alpha} \mathbf{C}^{xx} & 0 \\ 0 & \frac{1}{0.5+\alpha} \mathbf{C}^{yy} \end{bmatrix}$$

Given are two one-dimensional (uncorrelated) sensor readings $y_1 \in \mathbb{R}$ and $y_2 \in \mathbb{R}$ of a one-dimensional location $x \in \mathbb{R}$. The variances of the errors are $\sigma_1^2 = 2$ for the first and $\sigma_2^2 = 3$ for the second sensor.

- a) Formulate the (joint) measurement equation for the fusion of the two measurements.
- b) Write a function that simulates $m = 1000$ measurements using the measurement model in a) with true $x = 2$.
- c) For each simulated measurement, calculate the weighted least squares (WLS) solution using $W^{-1} = \begin{pmatrix} 2 & 0 \\ 0 & c \end{pmatrix}$ with $c = 1, \dots, 10$. Which value for c provides the BLUE estimate?
- d) Derive an analytic formula for the variance of the weighted least squares solution, using the W providing the BLUE estimate.
- e) Check if the analytical solution coincides with the empirical variance of the WLS solution from c).

Given are two one-dimensional (uncorrelated) sensor readings $y_1 \in \mathbb{R}$ and $y_2 \in \mathbb{R}$ of a one-dimensional location $x \in \mathbb{R}$. The variances of the errors are $\sigma_1^2 = 2$ for the first and $\sigma_2^2 = 3$ for the second sensor.

- a) Formulate the (joint) measurement equation for the fusion of the two measurements.

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}}_y = \underbrace{\begin{pmatrix} 1 \\ 1 \end{pmatrix}}_H x + \underbrace{\begin{pmatrix} e_1 \\ e_2 \end{pmatrix}}_e$$

$$e \in \mathcal{N}(0, C)$$

$$C = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}$$

- b) Write a function that simulates $m = 1000$ measurements using the measurement model in a) with true $x = 2$.
- c) For each simulated measurement, calculate the weighted least squares (WLS) solution using $W^{-1} = \begin{pmatrix} 2 & 0 \\ 0 & c \end{pmatrix}$ with $c = 1, \dots, 10$. Which value for c provides the BLUE estimate?

$$x^{LS} = \underbrace{(H^T W H)^{-1} H^T W}_K y = \left(\frac{1}{2} + \frac{1}{c}\right)^{-1} \cdot \left(\frac{y_1}{2} + \frac{y_2}{c}\right)$$

Using $c = 3$, we get the BLUE estimate, because then we get $W^{-1} = C$.

d) Derive an analytic formula for the variance of the weighted least squares solution, using the W providing the BLUE estimate.

$$\begin{aligned} \text{Cov}\{x^{LS}\} &= KCK^T \\ &= (H^TWH)^{-1}H^TWCWH(H^TWH)^{-1} \\ &= (H^TC^{-1}H)^{-1}H^TC^{-1}CC^{-1}H(H^TC^{-1}H)^{-1} \\ &= (H^TC^{-1}H)^{-1}H^TC^{-1}H(H^TC^{-1}H)^{-1} \\ &= (H^TC^{-1}H)^{-1} \end{aligned}$$

e) Check if the analytical solution coincides with the empirical variance of the WLS solution from c).

- General: $\theta_y = \mathbf{K}\mathbf{y} + \mathbf{b}$
- Unbiased: $E[\mathbf{K}\mathbf{y} + \mathbf{b}] = E[\mathbf{K}\mathbf{y}] + \mathbf{b} = \mathbf{K}\mathbf{H}\mathbf{x} + \mathbf{b} \stackrel{!}{=} \mathbf{x} \iff \mathbf{K}\mathbf{H} = \mathbf{I} \wedge \mathbf{b} = \mathbf{0}$
- Therefore, we have

$$\mathbf{x}^{LS} = \underbrace{(\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}}_{\mathbf{K}}$$
$$\mathbf{K}\mathbf{H} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{H} = \mathbf{I}$$

- Gauss-Markov: If the error is zero mean with covariance \mathbf{C} , then Least Squares is BLUE if $\mathbf{W} = \mathbf{C}^{-1}$
- Assume \mathbf{K}^{LS} to be calculated using $\mathbf{W} = \mathbf{C}^{-1}$ and any different $\mathbf{K} = \mathbf{K}^{LS} + \mathbf{B}$
- The different \mathbf{K} must still fulfill the unbiased condition

$$\mathbf{KH} = \mathbf{I}$$

$$\underbrace{\mathbf{K}^{LS}\mathbf{H}}_{\mathbf{I}} + \mathbf{BH} = \mathbf{I}$$

$$\mathbf{BH} = \mathbf{0}$$

- The MSE given the new \mathbf{K} is then the trace of

$$\begin{aligned}\text{Cov}[\mathbf{K}\mathbf{y}] &= \mathbf{K}\mathbf{C}\mathbf{K}^T \\ &= \underbrace{\mathbf{K}^{LS}\mathbf{C}(\mathbf{K}^{LS})^T}_{\text{Cov}[\mathbf{K}^{LS}\mathbf{y}]} + \cancel{\mathbf{K}^{LS}\mathbf{C}\mathbf{B}^T} + \cancel{\mathbf{B}\mathbf{C}(\mathbf{K}^{LS})^T} + \underbrace{\mathbf{B}\mathbf{C}\mathbf{B}^T}_{\geq 0}\end{aligned}$$

$$\geq \text{Cov}[\mathbf{K}^{LS}\mathbf{y}]$$

$$\begin{aligned}\mathbf{B}\mathbf{C}(\mathbf{K}^{LS})^T &= \underbrace{\mathbf{B}\mathbf{C}\mathbf{E}^{-1}\mathbf{H}}_0(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1} \\ &= \mathbf{0}\end{aligned}$$

Four independent sensors measure the temperature in a fridge. Their results are

Sensor	1	2	3	4
Measurement	6	3	1	2

Calculate the unweighted least squares solution.

$$x^{LS} = (H^T \omega H)^{-1} H^T \omega y$$

$$\begin{bmatrix} 6 \\ 3 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} x + e$$
$$x^{LS} = \frac{12}{4} = 3$$

To improve the estimate, the sensors errors are determined. They are zero-mean with variances

Sensor	1	2	3	4
Variance	2	1	1	0.66

Calculate the result of the BLUE.

$$\mathbf{W} = \mathbf{C}^{-1} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.66 \end{bmatrix}^{-1}$$

$$x_{\text{BLUE}}^{\text{LS}} = \left(\frac{1}{2} + 1 + 1 + \frac{3}{2} \right)^{-1} \left(\frac{6}{2} + 3 + 1 + 1.5 \cdot 2 \right) = \frac{10}{4} = 2.5$$

Stochastic Least squares:

• linear LS is a linear estimator: $\hat{\theta}_y = (H^T \omega H)^{-1} H^T \tilde{\omega}_y$ w ist hierbei invert. Covariance Matrix zum BLUE zu erfüllen

Consider a one-dimensional state $x \in \mathbb{R}$ and measurements $y_k \in \mathbb{R}, k = 1, \dots, m$ with linear measurement equation

$$y_k = x + e_k$$

where $e_k \sim \text{Exp}(\mu)$ is exponential distributed according to

$$p(e_k) = \begin{cases} \frac{1}{\mu} e^{-\frac{1}{\mu} e_k} & e_k \geq 0 \\ 0 & e_k < 0 \end{cases},$$

where $\mu > 0$ is the scale parameter and e is the base of the natural logs.

Hint: For exponential distribution e_k with scale parameter μ , we have $E[e_k] = \mu$ and $\text{Var}[e_k] = \mu^2$

- What is the BLUE of x for $m = 1$?
- What is the variance of the BLUE of x for $m = 1$?
- What is the BLUE and its variance of x for $m > 1$?

Problem 5: Solution

Schritt 1:



a) Reformulate the measurement equation

$$y_k - \mu = x + e_k^* - \mu$$

um unbiased zu sein

y_k^* e_k^*

Blue braucht Bias von 0. In der urspr. Version ist dies nicht gegeben. $\theta_y = k_y + b$ soll sein $\theta_y = I(x - e_k) + 0$. $\theta_y = x - e_k$ ist nicht unbiased da $E(e_k) = \mu$. Heißt, wir müssen bei der ursprünglichen Measurement equation den Bias herausrechnen (auf beiden Seiten, damit alles gleich bleibt)

to get a zero-mean error. Then, apply least squares as the BLUE with $H = 1$ to get

$$\hat{x}_1^{\text{BLUE}} = y_k - \mu$$

Herleitung: $x^{LS} = (H^T \cdot W \cdot H)^{-1} \cdot H^T \cdot W \cdot y_k^*$, W ist Cov von e_k^*
also $\text{Cov}(e_k^*) = \mu^2 - \mu$, dann in x^{LS} Formel einsetzen
 $x^{LS} = (I \cdot \mu^2 - \mu \cdot I)^{-1} \cdot I \cdot \mu^2 \cdot \mu \cdot y_k^* \Rightarrow y_k^* = y_k - \mu$

b) As $\hat{x}_1^{\text{BLUE}} = x + e_k^*$ and $E[\hat{x}_1^{\text{BLUE}}] = x$, its variance is $\text{Var}[\hat{x}_1^{\text{BLUE}}] = \text{Var}[e_k^*] = \mu^2$.

c) For the multidimensional measurement case, we again stack the measurement equations, resulting in

$$\hat{x}_m^{\text{BLUE}} = \frac{\sum_{k=1}^m y_k}{m} - \mu ,$$

with variance $\text{Var}[\hat{x}_m^{\text{BLUE}}] = \frac{\mu^2}{m}$.

Consider a 1D state x and estimates x_1 and x_2 . The errors of the estimates are correlated with

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.8\sqrt{1 \cdot 4} \\ 0.8\sqrt{1 \cdot 4} & 4 \end{bmatrix}\right)$$

- Write a function which visualizes the joint covariance matrix. Use it to draw the matrix along with the column space in the measurement space. Then assume the correlation coefficient 0.8 from the description is unknown and define a covariance which ignores it. Draw that covariance and calculate and visualize the solution of the BLUE.
- Now write a function which calculates a matrix which is bigger than the input matrix, determined by a parameter α as in the lecture. Calculate the BLUE estimate for different α values and determine which works best. Visualize the results.

Sensor Data Fusion

Bayesian Estimation

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Tracking (**live**)
12. Advanced Topic
13. Advanced Topic (2)
14. Summary and Discussion (**live**)

- Prior probability density function:

"The Bayesian approach assumes that a prior density function $p(x)$ for the state x is available."

$$y = Hx + e$$

$p(x) = \mathcal{N}(x; \mu_x, \Sigma_x)$

$$p(e) = \mathcal{N}(e; 0, \Sigma_e)$$

↑
probability density function
of error

- Likelihood function (as a function of x):

$$p(y|x)$$

specifies the probability that the measurement y is observed when x is given. Hence it can be used to measure how good the current state estimate fits to the measured data.

$$p(y|x) = p_e(y - Hx)$$

↑ "p of y given x " ↓ probability density of error

- Joint probability density function:

$$p(x, y) = p(y|x) \cdot p(x)$$

↓ hier keine RV sondern fester Wert

likelihood times prior

Prior: Represents initial beliefs before observing data

Posterior: After observing data, we update beliefs about param.
given both the prior distribution & the obs. data using bayes' theorem

Bayes' theorem: $P(A|B) = (P(B|A) \cdot P(A)) / P(B)$

Bayesian Mean Square Error (1)

expected value of squared norm of the estimation
error & meaning $\theta_y - x$.

$$\text{BMSE}(\theta_y) = \mathbb{E}_{x,y}\{\|\underbrace{\theta_y - x\|}_e^2\}$$

Quality of Bayesian Estimator

$$= \int \int \|\theta_y - x\|^2 p(x, y) dx dy$$

:= e joint density operator

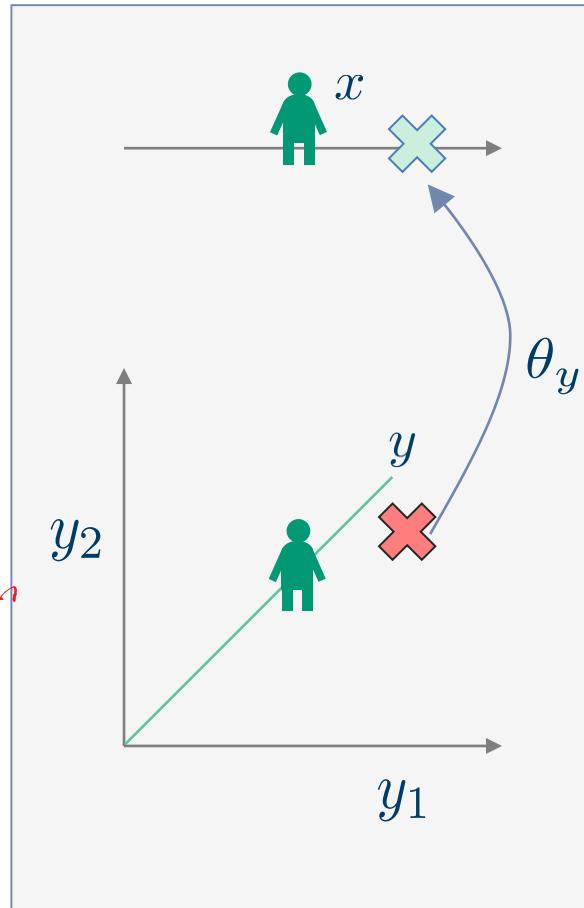
$$= \int \int \|\theta_y - x\|^2 p(y|x) dy p(x) dx$$

MSE($\theta_y|x$) ← non bayesian mean squared error
averaged according to prior $p(x)$

$$= \int \int \|\theta_y - x\|^2 \underbrace{p(x|y) dx p(y) dy}_{\substack{\text{posterior mean} \\ \text{squared error}}} \uparrow \text{averaged over all posterior mean squared errors}$$

joint probability transf.
from previous slide

← wieso kann man hier die Reihen.
vertauschen?



Bayesian Mean Square Error (2)

Note: Bias does not depend on specific x , because the mean of x is used in the definition of the bias.

- (Bayesian) Bias: Expectation Error (not squared)

$$\text{beta} \rightarrow \beta = E_{x,y}\{\epsilon\} = E(x) - E(\theta_y)$$

↑
expectation over the estimation error ϵ with respect to x & y

- Unbiased:

$$E_{x,y}\{\epsilon\} = 0$$

- Error covariance:

$$\Sigma_\epsilon = \text{Cov}[\epsilon] = E_{x,y}\{(\epsilon - \beta)(\epsilon - \beta)^T\}$$

- Bias-Variance-Decomposition:

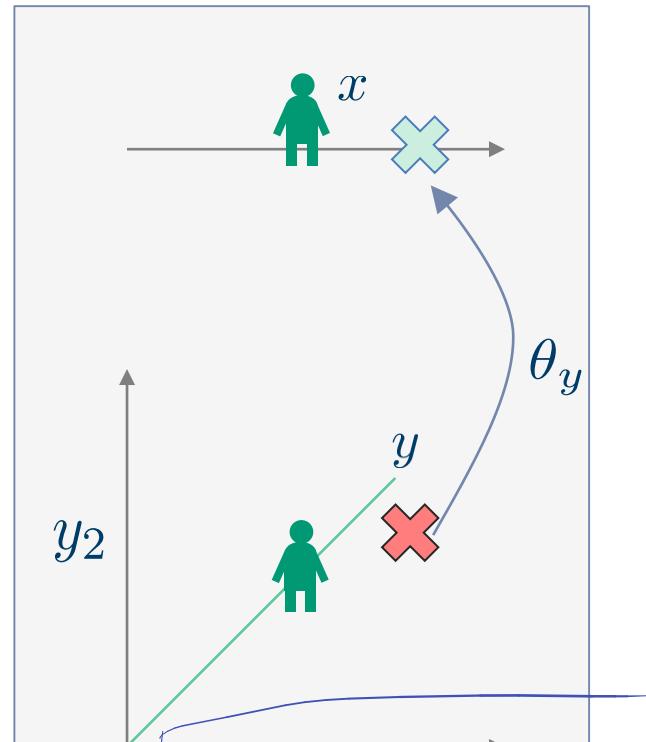
$$\begin{aligned} \text{BMSE}(\theta_y) &= \text{Tr}[E\{\epsilon\epsilon^T\}] \\ &= \text{Tr}[\Sigma_\epsilon] + \text{Tr}[\beta\beta^T] \end{aligned}$$

↑ Covariance of bias

↑ trace of the covariance of the estimation error

$\neq \text{Cov}(\theta_y)$

↑ involves cov. mat. from Estimation error, not of the estimator



Wiederholung: Fischer approach

- Bias-variance decomposition:

$$\text{E}[||\theta_y - x||^2] = \underbrace{\text{Tr}[\text{Cov}[\theta_y]]}_{\text{mean squared error}} + \underbrace{\|\underbrace{E[\theta_y] - x}\|_2^2}_{\text{trace / spur}}$$

(Mean-Variance-Decomposition of $\epsilon = \theta_y - x$)
- Bias: $\beta = E(\epsilon)$
 $\beta = E[\theta_y] - x$ (usually depends on x)
- Unbiased: $E[\theta_y] = x$ for all possible x .

Folie fehlt: Example - Variance of Estimator vs Error Variance

Variance of Estimator $\hat{\theta}_Y$ does not coincide with Epsilon

$$\theta_y^{opt} = E[x|y] = \int x p(x|y) dx$$

y gegeben bzw. wird
 als Param. reingereicht & bildet auf x ab

Proof:

With $\theta_y^{opt} - x = \theta_y^{opt} - c + c - x$ we get

• Proof was skipped in lecture

$$\begin{aligned}
 E[(\theta_y^{opt} - c + c - x)^T (\theta_y^{opt} - c + c - x)|y] = \\
 E[(\theta_y^{opt} - c)^T (\theta_y^{opt} - c) + (\theta_y^{opt} - c)^T (c - x) + \\
 (c - x)^T (\theta_y^{opt} - c) + (c - x)^T (c - x)|y]
 \end{aligned}$$

With $c := E[x|y]$, the cross-terms vanish:

$$E[(\theta_y^{opt} - E[x|y])^T (\theta_y^{opt} - E[x|y]) + (E[x|y] - x)^T (E[x|y] - x)|y]$$

which becomes minimal for $\theta_y^{opt} = E[x|y]$

How to Compute the Optimal Bayesian Estimator?

$$\hat{G}_y^{\text{opt}} = E(x|y)$$

- Posterior (Bayes rule):

$$p(x|y) \xrightarrow[\text{so called 'evidence'}]{\substack{\text{probability of } x \text{ given } y \\ (\text{posterior})}} \frac{p(y|x) \cdot p(x)}{p(y)} \xrightarrow[\text{prior}]{\substack{\text{likelihood} \\ \text{so called 'evidence'}}} \frac{P(x,y)}{P(y)}$$

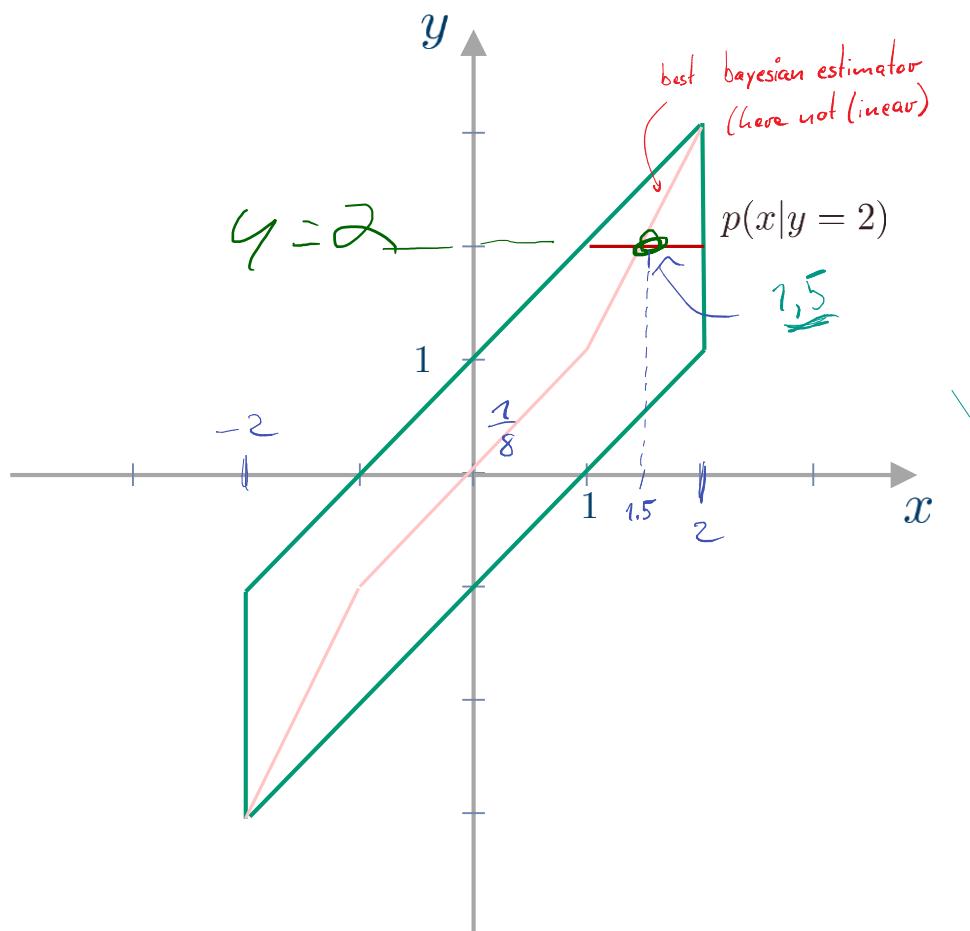
- Posterior mean:

$$E[x|y] = \int xp(x|y)dx$$

- Problems:

- No general closed form solution (*integral problematic in higher Dimensions*)
- Complete description of prior and likelihood required

1D-Example: Uniform Prior and Uniform Noise



Measurement equation $y = x + e$ with

- $x \xrightarrow{\text{prior}} U(-2, 2)$

- $e \sim U(-1, 1)$

\uparrow uniform distribution

$$p(y|x) = U(x-1, x+1) ,$$

$$p(x,y) = p(y|x) \cdot p(x)$$

$$= \begin{cases} \frac{1}{2} \cdot \frac{1}{4} & \text{if } -2 \leq x \leq 2 \text{ and } x-1 \leq y \leq x+1 \\ 0 & \text{otherwise} \end{cases}$$

Conditioning on $y = 2$:

$$p(x|y=2) = \begin{cases} 1 & \text{if } 1 \leq x \leq 2 \\ 0 & \text{otherwise} \end{cases}$$

\checkmark sonst würde $y = x + e$ nicht mehr passen. Werte werden für Integralgrenzen verwendet (siehe vorherige Seite)

Optimal Bayesian Estimator: $E[x|y=2] = 1.5$ $\leftarrow \int_{-2}^2 x \cdot p(x|y=2) dx = \int_{-2}^2 x dx = 7.5$
(not Linear - consists out of 3 estimators)

Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 6

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 5 solution
- Example - Woodbury Identity proof
- Problem 6 - Silly Estimator with Prior
- Homework 6 presentation

What is the difference between Bayesian and Fisher approach?

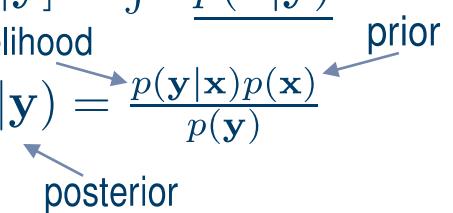
- We have $\mathbf{y} = \mathbf{Hx} + \mathbf{e}$
- Fisher: $\mathbf{x}?, \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- Bayesian: $\mathbf{x} \in \mathcal{N}(\boldsymbol{\mu}_x, \mathbf{C}_{xx}), \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- In general
 - Prior: $p(\mathbf{x})$
 - Likelihood: $p(\mathbf{y}|\mathbf{x})$
 - $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$

How can you measure the quality of an estimator in a Bayesian setting?

Bayesian Mean Square Error (BMSE)

$$\begin{aligned}\text{BMSE}(\theta_y) &= \mathbb{E}_{x,y}(\|\underbrace{\theta_y - x}_{\epsilon}\|^2) \\ &= \text{Tr}(\text{Cov}(\epsilon)) + \text{Tr}(\mathbb{E}_{x,y}(\epsilon)\mathbb{E}_{x,y}(\epsilon)^T)\end{aligned}$$

What is the optimal Bayesian estimator and how can you compute it?

- $\theta_y^{opt} = E[\mathbf{x}|\mathbf{y}]$
- Given $p(\mathbf{x})$, $p(\mathbf{y}|\mathbf{x})$
- $E[\mathbf{x}|\mathbf{y}] = \int \mathbf{x} p(\mathbf{x}|\mathbf{y}) d\mathbf{x}$
- $$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$


Consider a 1D state x and estimates x_1 and x_2 . The errors of the estimates are correlated with

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.8\sqrt{1 \cdot 4} \\ 0.8\sqrt{1 \cdot 4} & 4 \end{bmatrix}\right)$$

- Write a function which visualizes the joint covariance matrix. Use it to draw the matrix along with the column space in the measurement space. Then assume the correlation coefficient 0.8 from the description is unknown and define a covariance which ignores it. Draw that covariance and calculate and visualize the solution of the BLUE.
- Now write a function which calculates a matrix which is bigger than the input matrix, determined by a parameter α as in the lecture. Calculate the BLUE estimate for different α values and determine which works best. Visualize the results.

From the Woodbury Matrix Identity it follows:

For matrices A, C and V with suitable dimensions, the following holds:

$$(A + V^T C V)^{-1} = A^{-1} - A^{-1} V^T (C^{-1} + V A^{-1} V^T)^{-1} V A^{-1},$$

As $(A + UCV)^{-1} \stackrel{!}{=} \left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right)$, we must show that

$$(A + UCV)\left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) = I$$

$$\begin{aligned}
 & (A + UCV)\left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) \\
 = & \left(I - \underline{U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} \right) + \left(\underline{UCVA^{-1}} - UCVA^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) \\
 = & (I + UCVA^{-1}) - \left(\underline{U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} + \underline{UCVA^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} \right) \\
 = & I + UCVA^{-1} - (U + \underline{UCVA^{-1}U})(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \\
 = & I + UCVA^{-1} - UC(C^{-1} + VA^{-1}U)(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \\
 = & I + UCVA^{-1} - UCVA^{-1} \\
 = & I
 \end{aligned}$$

Consider the Silly Estimator from last exercise. This time, assume x is distributed according to

$$x \sim \mathcal{N}(\mu_x, \sigma_x^2).$$

- a) Which estimator is now better, the natural or the silly estimator?
- b) Is there a better estimator?

Given is a random variable $y \sim \mathcal{N}(x, 1)$. The objective is to estimate the deterministic mean x based on a single observation y . We consider two estimators:

- Natural estimator: $\theta_n(y) = y$
 - Silly estimator: $\theta_s(y) = 0$
- a) Calculate the bias, variance, and MSE of both estimators.
 - b) In which cases outperforms the silly estimator the natural estimator?

a)

You can write y as $y = x + e$ with $e \sim \mathcal{N}(0, 1)$. Then you get for

- Natural

$$\beta = 0; \text{Var}[\theta_n(y) - x] = \text{Var}[e] = 1$$

$$\text{BMSE}(\theta_n(y)) = 0 + 1 = 1$$

- Silly

$$\beta = \text{E}[\theta_s(y) - x] = -\mu_x; \text{Var}[\theta_s(y) - x] = \sigma_x^2$$

$$\text{BMSE}(\theta_s(y)) = \sigma_x^2 + \mu_x^2$$

- Silly is better if $\mu_x^2 + \sigma_x^2 < 1$

- b) As we are in Bayesian setting (statistical information about x and e given), the best estimator is the optimal Bayesian estimator

$$\theta_{opt}(y) = \text{E}[x|y] .$$

Consider again the setup of the Silly Estimator exercise with $x \sim \mathcal{N}(1, 1)$ and $y = x + e$ with $e \sim \mathcal{N}(0, 1)$.

- a) Write a function which calculates a possible value for x based on its distribution and then takes a measurement from that value using the measurement noise.
- b) Consider the natural estimator $\theta_n(y) = y$. Calculate the empirical mean square error of 1000 runs using this estimator and compare it with the analytic solution.
- c) Repeat the process with the silly estimator $\theta_s(y) = 0$.

Sensor Data Fusion

Linear Bayesian Estimation

Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter
11. Dynamic models: Tracking (**live**)
12. Advanced Topic
13. Advanced Topic (2)
14. Summary and Discussion (**live**)

Linear meas. equation

$$y = Hx + \rho$$

1. All probability densities are Gaussian
→ Posterior is Gaussian and can be derived analytically
2. Linear estimator
→ Linear MMSE can be derived analytically

Both special cases lead to the same result:

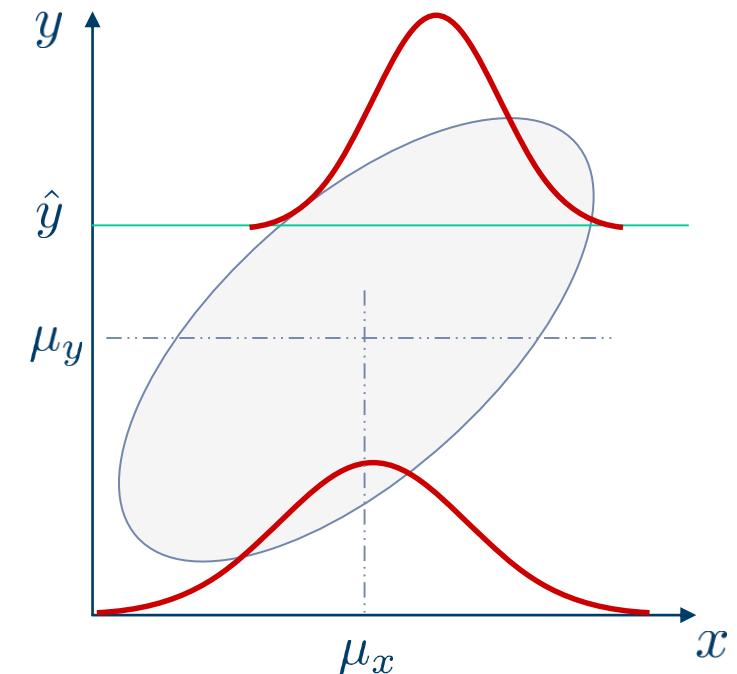
Kalman Filter Update Equations

- Optimal Bayesian estimator in case of Gaussian densities
- Best linear estimator otherwise

Gaussian joint density:

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \underbrace{\begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{bmatrix}}_{\mathbf{C}}\right)$$

$$\begin{aligned} \mu_x^+ &= \mu_x + \mathbf{C}_{xy} \mathbf{C}_{yy}^{-1} (\hat{y} - \mu_y) \\ \mathbf{C}^+ &= \mathbf{C}_{xx} - \mathbf{C}_{xy} \mathbf{C}_{yy}^{-1} \mathbf{C}_{yx} \end{aligned}$$



$$y = Hx + e$$

Conditioning on y :

$$p(x|y = \hat{y}) = \mathcal{N}(\mu_x^+, \mathbf{C}^+)$$

Todo: Posterior & Prior madschlagen

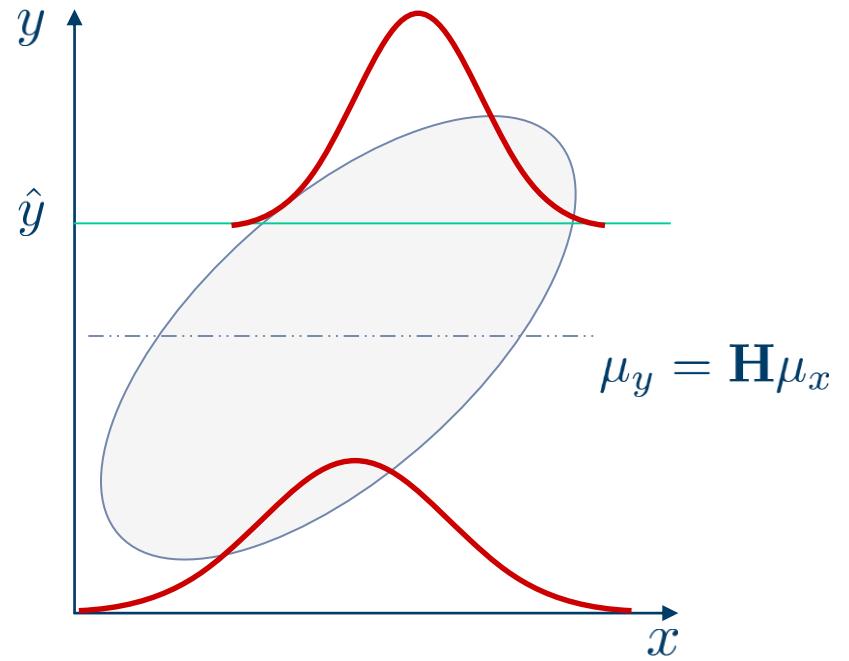
Linear Measurement Equation, Gaussian Prior and Noise

$y = \mathbf{H}x + e$ with

- prior $x \sim \mathcal{N}(\mu_x, \mathbf{C}_{xx})$
- noise $e \sim \mathcal{N}(\mu_e, \mathbf{C}_e)$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{H} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} x \\ e \end{bmatrix}$$

$$\begin{aligned} \text{Cov}\begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{H} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{C}_{xx} & 0 \\ 0 & \mathbf{C}_{ee} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{H} & \mathbf{I} \end{bmatrix}^T \\ &= \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{H} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xx}\mathbf{H}^T \\ 0 & \mathbf{C}_{ee} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xx}\mathbf{H}^T \\ \mathbf{H}\mathbf{C}_{xx} & \mathbf{H}\mathbf{C}_{xx}\mathbf{H}^T + \mathbf{C}_{ee} \end{bmatrix} \end{aligned}$$



$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \underbrace{\begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{bmatrix}}_C\right)$$

- Measurement equation:

$$y = \mathbf{H}x + e$$

$$\|y - \mathbf{H}x\|^2$$

- $y \in \mathbb{R}^m$ given
- $x \in \mathbb{R}^n$ and $e \in \mathbb{R}^m$ deterministic and unknown
- $\text{Rank}(\mathbf{H}) = n$ and $m > n$
- SPD weight \mathbf{W}
- LS estimate:

$$x^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} y$$

\downarrow
 C_{pq}^{-1}



- Measurement equation:

$$y = \mathbf{H}x + e$$

- $y \in \mathbb{R}^m$ given
- $x \in \mathbb{R}^n$ deterministic and unknown
- e is a RV with $E[e] = 0$ and $\text{Cov}[e] = \mathbf{C}_{ee}$
- $\text{Rank}(\mathbf{H}) = n$ and $m > n$
- BLUE estimate:

$$x^{BL} = (\mathbf{H}^T \mathbf{C}_{ee}^{-1} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{C}_{ee}^{-1} y$$

$$\text{Cov}[x^{BL}] = (\mathbf{H}^T \mathbf{C}_{ee}^{-1} \mathbf{H})^{-1}$$

- Measurement equation:

$$y = \mathbf{H}x + e$$

- x is a RV with $E[x] = \hat{x}$ and $\text{Cov}[x] = \mathbf{C}_{xx}$
- e is a RV with $E[e] = 0$ and $\text{Cov}[e] = \mathbf{C}_{ee}$
- Estimate (formulation 1):

$$\hat{x}^+ = \hat{x} + \mathbf{C}_{xx} \mathbf{H}^T (\mathbf{H} \mathbf{C}_{xx} \mathbf{H}^T + \mathbf{C}_{ee})^{-1} (y - \mathbf{H} \hat{x})$$

$$\mathbf{C}_{xx}^+ = \mathbf{C}_{xx} - \mathbf{C}_{xx} \mathbf{H}^T (\mathbf{H} \mathbf{C}_{xx} \mathbf{H}^T + \mathbf{C}_{ee})^{-1} \mathbf{H} \mathbf{C}_{xx}$$

- Estimate (formulation 2):

$$\hat{x}^+ = \hat{x} + \mathbf{C}_{xx}^+ \mathbf{H}^T \mathbf{C}_{ee}^{-1} (y - \mathbf{H} \hat{x})$$

$$\mathbf{C}_{xx}^+ = (\mathbf{C}_{xx}^{-1} + \mathbf{H}^T \mathbf{C}_{ee}^{-1} \mathbf{H})^{-1}$$

- Optimal for Gaussian densities, best linear estimator otherwise

From the Woodbury Matrix Identity:

For matrices A, C and V with suitable dimensions, the following holds:

$$(A + V^T C V)^{-1} = \underline{A^{-1}} - \underline{A^{-1} V^T} \underline{(C^{-1} + V \underline{A^{-1} V^T})^{-1} V} \underline{A^{-1}},$$

Proof: Exercise

$$\mathbf{C}_{xx}^+ = \underline{\mathbf{C}_{xx}} - \underline{\mathbf{C}_{xx} \mathbf{H}^T} (\mathbf{H} \underline{\mathbf{C}_{xx}} \mathbf{H}^T + \underline{\mathbf{C}_{ee}})^{-1} \underline{\mathbf{H} \mathbf{C}_{xx}} = (\mathbf{C}_{xx}^{-1} + \mathbf{H}^T \mathbf{C}_{ee}^{-1} \mathbf{H})^{-1}$$

- Measurement equations:

$$y_k = \mathbf{H}x + e_k \text{ for } k \in \mathbb{N} \text{ with}$$

$$x \sim \mathcal{N}(\hat{x}_0, \mathbf{C}_0)$$

$$e_k \sim \mathcal{N}(\mathbf{0}_m, \mathbf{C}_{ee})$$

and $\text{Cov}[x, e_k] = \mathbf{0}$, $\text{Cov}[e_l, e_k] = \mathbf{0}$, $l \neq k$.

- Invariants:

$$\hat{x}_k = \mathbb{E}[x|y_1, \dots, y_k]$$

$$\mathbf{C}_k = \text{Cov}[x|y_1, \dots, y_k]$$

- Update:

$$\hat{x}_{k+1} = \hat{x}_k + \mathbf{C}_k \mathbf{H}^T (\mathbf{H} \mathbf{C}_k \mathbf{H}^T + \mathbf{C}_{ee})^{-1} (y_k - \mathbf{H} \hat{x}_k)$$

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mathbf{C}_k \mathbf{H}^T (\mathbf{H} \mathbf{C}_k \mathbf{H}^T + \mathbf{C}_{ee})^{-1} \mathbf{H} \mathbf{C}_k$$

$$Y = X + E \quad H = I$$

- Special cases for $H = I$, i.e.,

$$\begin{cases} \hat{x}_{k+1} = \hat{x}_k + C_k(C_k + C_{ee})^{-1}(y_k - \hat{x}_k) \\ C_{k+1} = C_k - C_k(C_k + C_{ee})^{-1}C_k \end{cases}$$

$$\begin{cases} \hat{x}_{k+1} = \hat{x}_k + C_{k+1}C_{ee}^{-1}(y_k - \hat{x}_k) \\ C_{k+1} = (C_k^{-1} + C_{ee}^{-1})^{-1} \end{cases}$$

- $C_k = C_{ee}$:

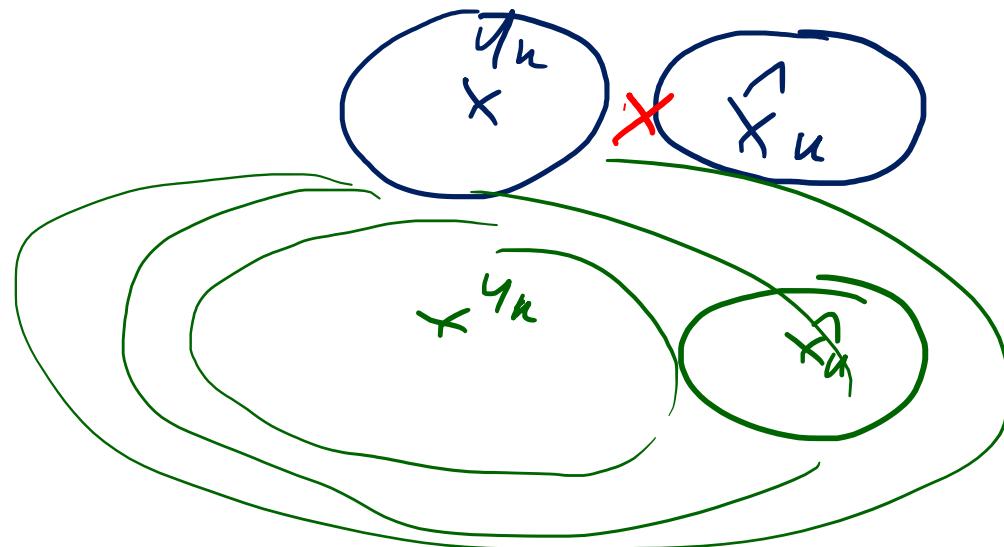
$$\begin{aligned} \hat{x}_{k+1} &= \frac{1}{2}(y_k + \hat{x}_k) \\ C_{k+1} &= \frac{1}{2}\underline{C_k} \end{aligned}$$

- $\text{Tr}[C_{ee}] \rightarrow \infty$

$$\begin{aligned} \hat{x}_{k+1} &= \hat{x}_k \\ C_{k+1} &= C_k \end{aligned}$$

- $\text{Tr}[C_k] \rightarrow \infty$

$$\begin{aligned} \hat{x}_{k+1} &= y_k \\ C_{k+1} &= C_{ee} \end{aligned}$$



Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 6

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 5 solution
- Example - Woodbury Identity proof
- Problem 6 - Silly Estimator with Prior
- Homework 6 presentation

What is the difference between Bayesian and Fisher approach?

- We have $\mathbf{y} = \mathbf{Hx} + \mathbf{e}$
- Fisher: $\mathbf{x}?, \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- Bayesian: $\mathbf{x} \in \mathcal{N}(\boldsymbol{\mu}_x, \mathbf{C}_{xx}), \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- In general
 - Prior: $p(\mathbf{x})$
 - Likelihood: $p(\mathbf{y}|\mathbf{x})$
 - $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$

How can you measure the quality of an estimator in a Bayesian setting?

Bayesian Mean Square Error (BMSE)

$$\begin{aligned}\text{BMSE}(\theta_y) &= \mathbb{E}_{x,y}(\|\underbrace{\theta_y - x}_{\epsilon}\|^2) \\ &= \text{Tr}(\text{Cov}(\epsilon)) + \text{Tr}(\mathbb{E}_{x,y}(\epsilon)\mathbb{E}_{x,y}(\epsilon)^T)\end{aligned}$$

What is the optimal Bayesian estimator and how can you compute it?

- $\theta_y^{opt} = E[\mathbf{x}|\mathbf{y}]$
- Given $p(\mathbf{x})$, $p(\mathbf{y}|\mathbf{x})$
- $E[\mathbf{x}|\mathbf{y}] = \int \mathbf{x} p(\mathbf{x}|\mathbf{y}) d\mathbf{x}$
- $$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$

likelihood prior

posterior

Measurement Equation gives us the likelihood

Consider a 1D state x and estimates x_1 and x_2 . The errors of the estimates are correlated with

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.8\sqrt{1 \cdot 4} \\ 0.8\sqrt{1 \cdot 4} & 4 \end{bmatrix}\right)$$

- Write a function which visualizes the joint covariance matrix. Use it to draw the matrix along with the column space in the measurement space. Then assume the correlation coefficient 0.8 from the description is unknown and define a covariance which ignores it. Draw that covariance and calculate and visualize the solution of the BLUE.
- Now write a function which calculates a matrix which is bigger than the input matrix, determined by a parameter α as in the lecture. Calculate the BLUE estimate for different α values and determine which works best. Visualize the results.

From the Woodbury Matrix Identity it follows:

For matrices A, C and V with suitable dimensions, the following holds:

$$(A + V^T C V)^{-1} = A^{-1} - A^{-1} V^T (C^{-1} + V A^{-1} V^T)^{-1} V A^{-1},$$

As $(A + UCV)^{-1} \stackrel{!}{=} \left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right)$, we must show that

$$(A + UCV)\left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) = I$$

$$\begin{aligned}
 & (A + UCV)\left(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) \\
 = & \left(I - \underline{U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} \right) + \left(\underline{UCVA^{-1}} - UCVA^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \right) \\
 = & (I + UCVA^{-1}) - \left(\underline{U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} + \underline{UCVA^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}} \right) \\
 = & I + UCVA^{-1} - (U + \underline{UCVA^{-1}U})(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \\
 = & I + UCVA^{-1} - UC(C^{-1} + VA^{-1}U)(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \\
 = & I + UCVA^{-1} - UCVA^{-1} \\
 = & I
 \end{aligned}$$

*Wurde in VL bzw. Übung
nur kurz erwähnt, aber
nicht erklärt.*

Consider the Silly Estimator from last exercise. This time, assume x is distributed according to

$$x \sim \mathcal{N}(\mu_x, \sigma_x^2).$$

- a) Which estimator is now better, the natural or the silly estimator?
- b) Is there a better estimator?

Given is a random variable $y \sim \mathcal{N}(x, 1)$. The objective is to estimate the deterministic mean x based on a single observation y . We consider two estimators:

- Natural estimator: $\theta_n(y) = y$
 - Silly estimator: $\theta_s(y) = 0$
- a) Calculate the bias, variance, and MSE of both estimators.
 - b) In which cases outperforms the silly estimator the natural estimator?

Problem 6: Solution

a)

You can write y as $y = x + e$ with $e \sim \mathcal{N}(0, 1)$. Then you get for

- Natural

$$\beta = 0; \text{Var}[\theta_n(y) - x] = \text{Var}[e] = 1$$

$$\text{BMSE}(\theta_n(y)) = 0 + 1 = 1$$

silly is only better
if $\mu_x^2 + \sigma_x^2 < 1$

- Silly

since silly Est. - always 0

$$\beta = \text{E}[\theta_s(y) - x] = -\mu_x; \text{Var}[\theta_s(y) - x] = \sigma_x^2$$
$$\text{BMSE}(\theta_s(y)) = \sigma_x^2 + \mu_x^2$$

- Silly is better if $\mu_x^2 + \sigma_x^2 < 1$

- b) As we are in Bayesian setting (statistical information about x and e given), the best estimator is the optimal Bayesian estimator

$$\theta_{opt}(y) = \text{E}[x|y] .$$

Consider again the setup of the Silly Estimator exercise with $x \sim \mathcal{N}(1, 1)$ and $y = x + e$ with $e \sim \mathcal{N}(0, 1)$.

- a) Write a function which calculates a possible value for x based on its distribution and then takes a measurement from that value using the measurement noise.
- b) Consider the natural estimator $\theta_n(y) = y$. Calculate the empirical mean square error of 1000 runs using this estimator and compare it with the analytic solution.
- c) Repeat the process with the silly estimator $\theta_s(y) = 0$.

Sensor Data Fusion

Introduction to the Kalman Filter

Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

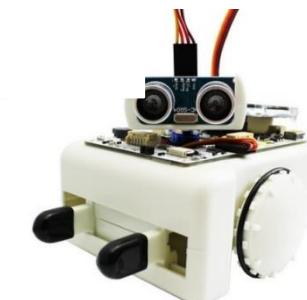
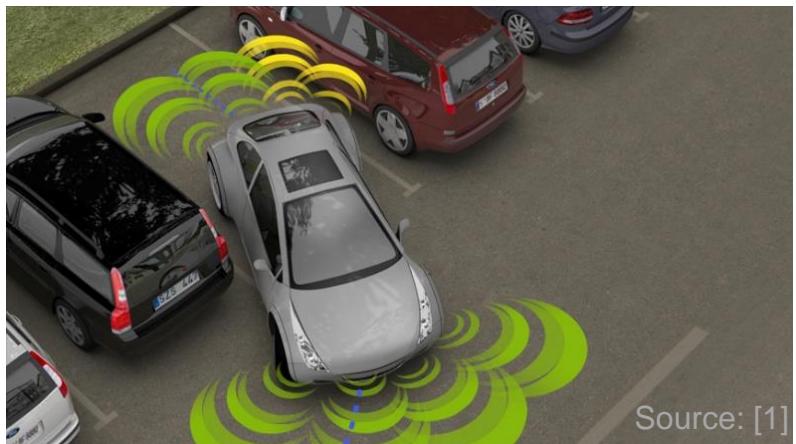
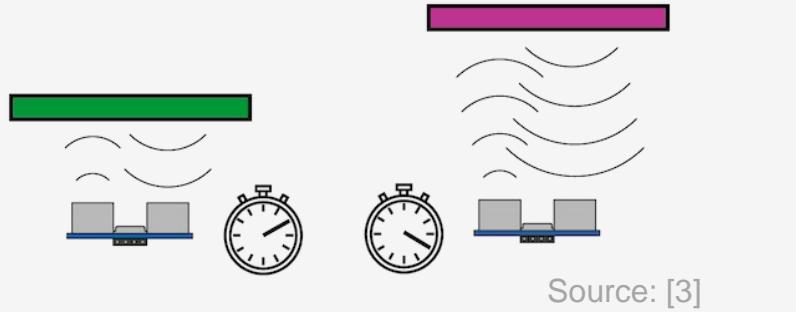
**DATA
FUSION Lab**



After the lecture you will be in the position to...

- fuse noisy measurements and state estimates,
- predict future states based on the current state estimate,
- explain the basic concept of the Kalman filter for the special case of scalar systems.

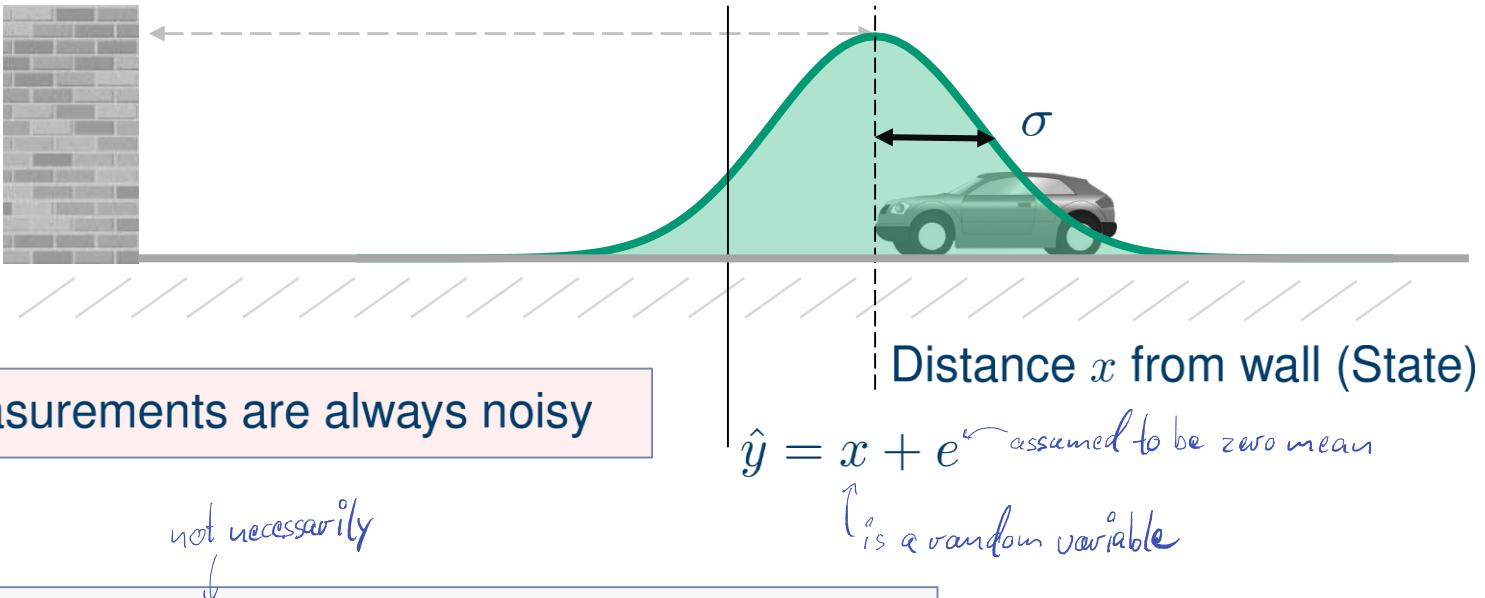
- Speaker sends out sound waves
- Microphone measures time-of-flight
- Distance can be calculated based on the speed of sound



[1] <https://www.valeo.de>

[2] <https://www.parrot.com/de/drohnen/parrot-bebop-2>

[3] <http://arcbotics.com/products/sparki>



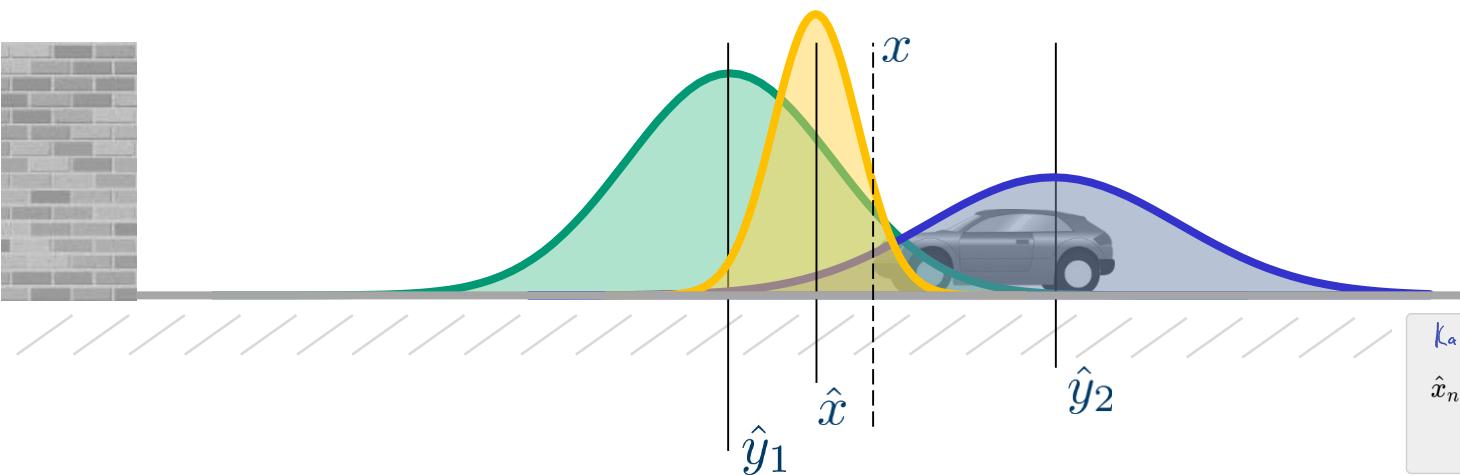
Measurement error e is (often) Gaussian distributed:

- Mean $E[e] = 0$
- Variance $\text{Var}[e] = E[(e - E[e])^2] = E[e^2] = \sigma^2$
- larger variance \rightarrow larger mean square error

\Rightarrow A measurement \hat{y} is unbiased, i.e., $E[\hat{y} - x] = 0$

Key idea:
Combine, i.e., fuse, multiple measurements to reduce the error

Update: Fusion of Two Noisy Measurements



Note: in the recursive update step use

$$\hat{x}_{\text{new}} = \hat{x}_{\text{old}} + \alpha \cdot y_n$$

↑
no alpha needed

Kalman State Update (mean)

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \frac{p_{n,n-1}}{p_{n,n-1} + r_n} (z_n - \hat{x}_{n,n-1})$$

Kalman Gain

Kalman State Upd. (variance)

$$p_{n,n} = (1 - K_n) p_{n,n-1}$$

↳ same as

$$\sigma_{x_{k,k}}^2 = \sigma_{x_{k,k-1}}^2 - \frac{\sigma_y^2}{\sigma_{x_{k,k-1}}^2 + \sigma_y^2}$$

Meas. / Estimate	Error Covariance
\hat{y}_1	$\sigma_1^2 = E[e_1^2] = E[(y_1 - x)^2]$
\hat{y}_2	$\sigma_2^2 = E[e_2^2] = E[(y_2 - x)^2]$
$\hat{x} = (1 - \alpha)\hat{y}_1 + \alpha\hat{y}_2$	$\sigma_x^2 = E[(\hat{x} - x)^2] = (1 - \alpha)\sigma_1^2$

is actually a weighted average

y_3 (another measurement \hat{y}_3 could be fused with \hat{x} directly)
with the same formulas

$$\alpha = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

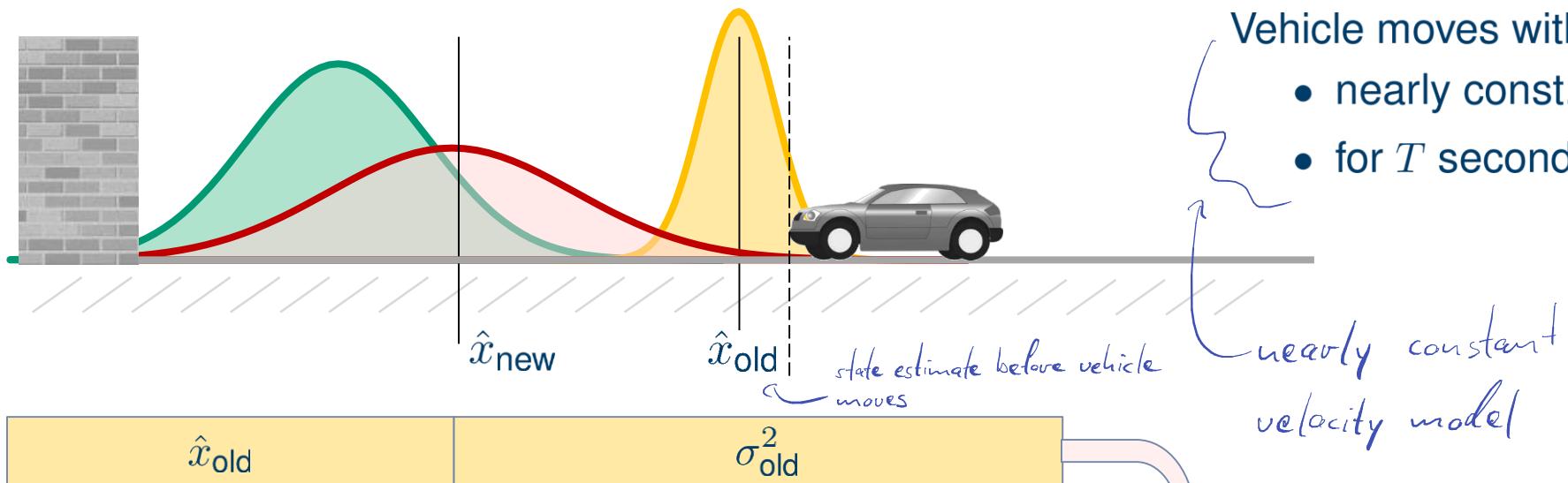
Sigma

- "A natural first estimate for the variance would be σ_x^2 of x , since it is an unbiased estimate."

Linear estimator

Variance decreases

Can be applied recursively



Discrete-time Motion Model:

we assume a nearly constant velocity for T seconds

$$x_{\text{new}} = x_{\text{old}} + T \cdot (v + e_v)$$

with velocity v and zero-mean noise e_v with variance σ_v^2

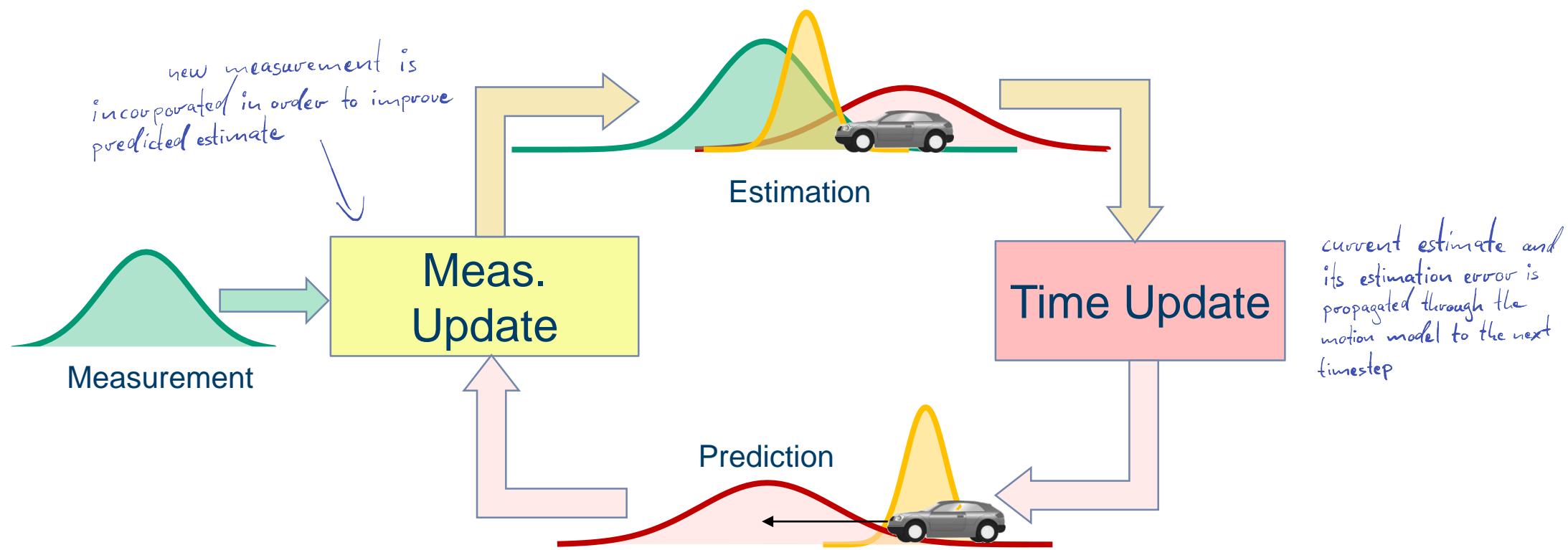
$$\hat{x}_{\text{new}} = \hat{x}_{\text{old}} + T \cdot v$$

$$\sigma_{\text{new}}^2 = \sigma_{\text{old}}^2 + T^2 \sigma_v^2$$

Variance increases

Next measurement: fusion with prediction

(One-dimensional) Kalman Filter



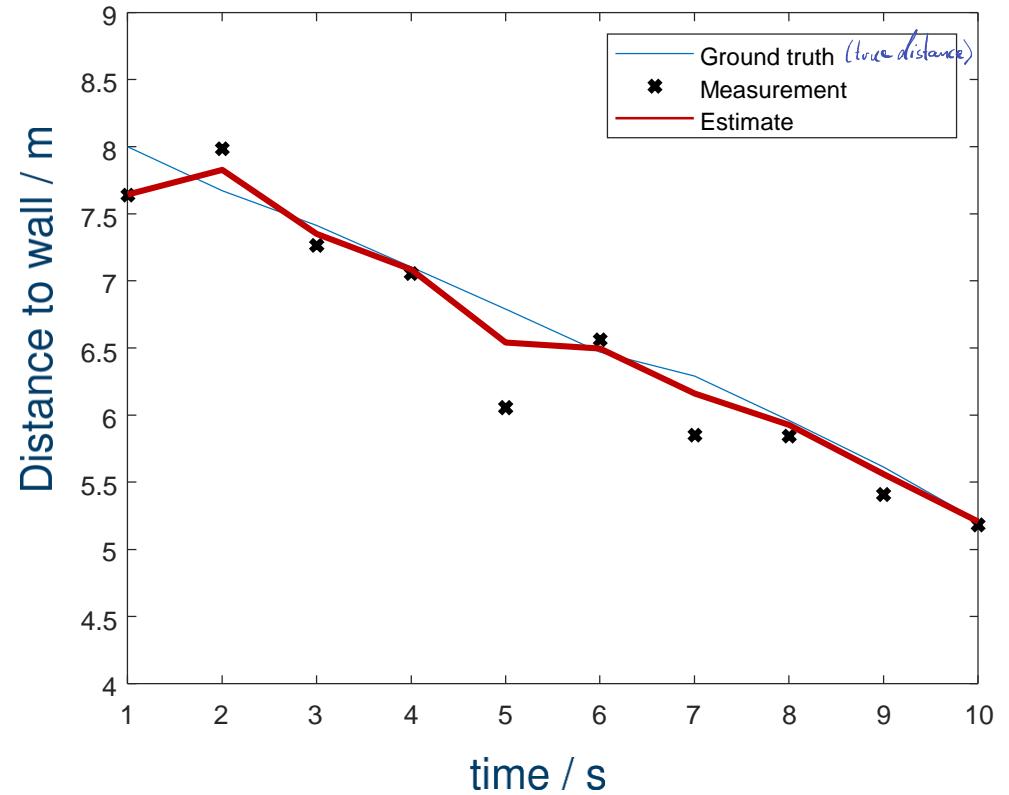
- Alternating measurement and time updates
- Recursive filter
- Optimal for linear systems with (white) Gaussian noise,

otherwise best linear unbiased estimator, BLUE: you only need mean & variance of error in order for the kalman filter to be BLUE

Ref: Super Tutorial unter www.kalmanfilter.net/kalman1d.html

	Equation	Equation Name	Alternative names used in the literature
State Update	$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n (z_n - \hat{x}_{n,n-1})$	State Update	Filtering Equation
	$p_{n,n} = (1 - K_n) p_{n,n-1}$	Covariance Update	Corrector Equation
	$K_n = \frac{p_{n,n-1}}{p_{n,n-1} + r_n}$	Kalman Gain	Weight Equation
State Predict	$\hat{x}_{n+1,n} = \hat{x}_{n,n}$ (For constant dynamics)	State Extrapolation	Predictor Equation Transition Equation
	$\hat{x}_{n+1,n} = \hat{x}_{n,n} + \Delta t \hat{\dot{x}}_{n,n}$		Prediction Equation
	$\hat{x}_{n+1,n} = \hat{x}_{n,n}$ (For constant velocity dynamics)		Dynamic Model State Space Model
	$p_{n+1,n} = p_{n,n}$ (For constant dynamics)	Covariance Extrapolation	
	$p_{n+1,n}^x = p_{n,n}^x + \Delta t^2 \cdot p_{n,n}^v$		Predictor Covariance Equation
	$p_{n+1,n}^v = p_{n,n}^v$ (For constant velocity dynamics)		

Numerical Example



• Filter: 'Because it filters out the noise.'

- Time interval $T = 1\text{s}$
- Velocity $v = -0.3 \frac{\text{m}}{\text{s}}$
- Meas. noise $\sigma_e = 0.3 \text{m/s}$
- System noise $\sigma_v = 0.05 \text{m/s}$

Given is a scalar random variable x with $E[x] = \mu_x$ and $\text{Var}[x] = \sigma_x^2$

- Multiplication with a constant a :

$$E[a \cdot x] = a\mu_x \text{ and } \text{Var}[a \cdot x] = a^2 \sigma_x^2$$

important

- Sum of x and constant b :

$$E[x + b] = \mu_x + b \text{ and } \text{Var}[x + b] = \sigma_x^2$$

var. not changed by sum

- Sum of x and (independent) random variable z with $E[z] = \mu_z$ and $\text{Var}[z] = \sigma_z^2$:

$$E[x + z] = \mu_x + \mu_z \text{ and } \text{Var}[x + z] = \sigma_x^2 + \sigma_z^2$$

Update Formulas:

$$\hat{x} = (1 - \alpha)\hat{y}_1 + \alpha\hat{y}_2 \quad \sigma_x^2 = E[(\hat{x} - x)^2] = (1 - \alpha)\sigma_1^2$$

↑ unbiased? (Blue cond. 1)

$$\alpha = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

is it optimal? (blue)
(and. 2)

Unbiased:

- $E[\hat{x}] = E[(1 - \alpha)(x + e_1) + \alpha(x + e_2)] = (1 - \alpha)E[x] + \alpha E[x] = E[x]$

$\underbrace{\hat{y}_1}_{\uparrow}$ $\underbrace{\hat{y}_2}_{\uparrow}$

Minimum Mean Squared Error (as $\text{Var}[\hat{x} - x] = E[(\hat{x} - x)^2]$):

- Error Variance:

$$\text{Var}[\hat{x} - x] = \text{Var}[(1 - \alpha)(x + e_1) + \alpha(x + e_2) - x] = (1 - \alpha)^2\sigma_1^2 + \alpha^2\sigma_2^2$$

because \hat{x} unbiased coincides with mean squ. error

- Derivative w.r.t. α shall be zero:

with respect to α

$$-2(1 - \alpha)\sigma_1^2 + 2\alpha\sigma_2^2 = 0 \rightarrow \text{abgeleitet nach } \alpha$$

e_1 & e_2 independent

Discrete-time Motion Model:

$$x_{\text{new}} = x_{\text{old}} + T \cdot (v + e_v)$$

with speed v and zero-mean noise e_v with variance σ_v^2

$$\hat{x}_{\text{new}} = \hat{x}_{\text{old}} + T \cdot v$$

$$\sigma_{\text{new}}^2 = \sigma_{\text{old}}^2 + T^2 \sigma_v^2$$

Unbiased:

- $E[\hat{x}_{\text{new}}] = E[\hat{x}_{\text{old}} + T \cdot v] = E[x_{\text{old}}] + T \cdot v = \underline{E[x_{\text{new}}]}$

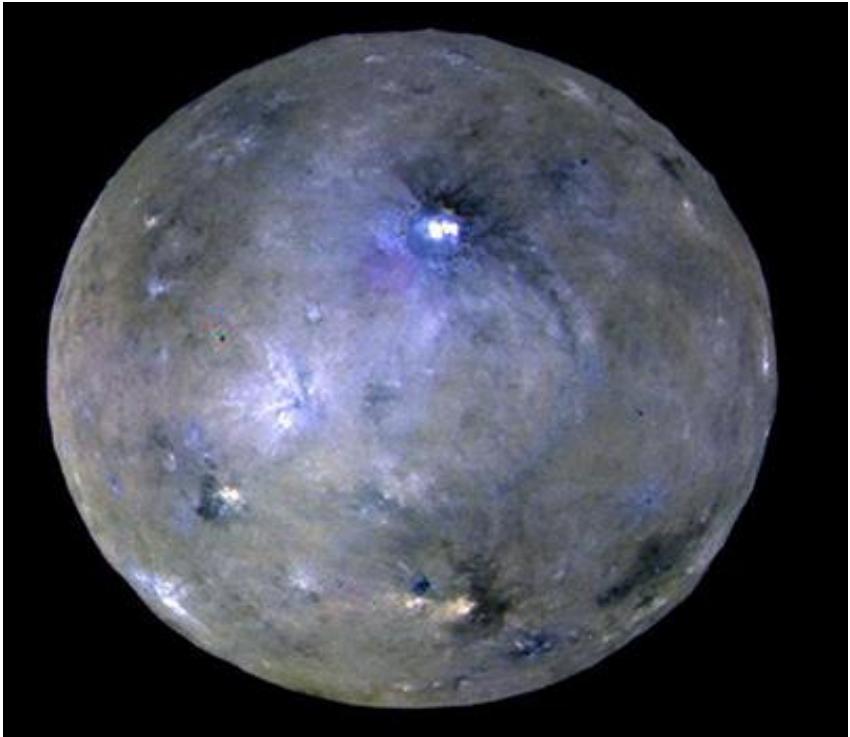
Mean Squared Error:

- $E[(x_{\text{new}} - \hat{x}_{\text{new}})^2] = E[(x_{\text{old}} + T \cdot (v + e_v) - \hat{x}_{\text{old}} - T \cdot v)^2]$
 $= \sigma_{\text{old}}^2 + T^2 \sigma_v^2$

$$\begin{array}{c} \uparrow \\ x_{\text{old}} - \hat{x}_{\text{old}} \end{array}$$

- On January 1, 1800, Giuseppe Piazzi, discovered Ceres in Palermo
- He recorded its position
- Over 42 days, Piazzi collected 19 measurements
- On February 12, the object disappeared
- Total motion covered an arc of 3°

- C. F. Gauss (24 years) predicted the Ceres orbit in 1801
- Ceres was rediscovered by Franz X. von Zach on 31 December
- Gauss used the method of least squares (developed in 1795 with 18 years)



Source:

<https://solarsystem.nasa.gov/missions/dawn/science/ceres/>

- Born in Budapest, 1930
- Emigrated to the US in 1943
- BA (1953) and MA (1954) from MIT
- PhD, 1957, Columbia University, NYC
- Professor at Stanford University (1964-1971), University of Florida (1971-1992) and ETH (since 1973)



- First publication of the Kalman Filter in 1960
- First implementation by Stanley F. Schmidt (NASA Ames Research Center)
- Used for trajectory estimation in the Apollo Program
- Many co-inventors and related works
(e.g., Swerling, Bucy, Schmidt, Stratonovich, Gauss, ...)

Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 7

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 6 solution
- Example - Optimal Bayesian estimator
- Problem 7 - MMSE Estimator
- Homework 7 presentation

Why is it difficult to obtain the optimal Bayesian estimator?

- No closed-form solution (in general)
- complete prior and likelihood information required
- Kalman filter is optimal Bayesian filter if
 - Linearity is given
 - All distributions are Gaussian

What happens in the Kalman filter update for $\mathbf{H}=\mathbf{I}$ if the noise of the prior is much higher than the measurement noise and vice versa?

Regard the update formula:

$$x_{k+1|k+1} = x_{k+1|k} + \underbrace{\mathbf{C}_{k+1|k} \mathbf{H}^T (\overbrace{\mathbf{H} \mathbf{C}_{k+1|k} \mathbf{H}^T + \mathbf{R}}^{\mathbf{S}})^{-1} (y - \mathbf{H} x_{k+1|k})}_\mathbf{K}$$

With the prior noise $\mathbf{C}_{k+1|k}$ being much larger than the measurement noise \mathbf{R} , $\mathbf{K}\mathbf{H}$ gets close to being the identity matrix, so that the state is subtracted and the update will only use the measurement to set the new state. Vice versa, \mathbf{S}^{-1} would be close to zero, leaving only the prior state.

What can be said about the posterior distribution if all other distributions involved are Gaussian?

The posterior $p(x|y)$ is also Gaussian.

Consider again the setup of the Silly Estimator exercise with $x \sim \mathcal{N}(1, 1)$ and $y = x + e$ with $e \sim \mathcal{N}(0, 1)$.

- Write a function which calculates a possible value for x based on its distribution and then takes a measurement from that value using the measurement noise.
- Consider the natural estimator $\theta_n(y) = y$. Calculate the empirical mean square error of 1000 runs using this estimator and compare it with the analytic solution.
- Repeat the process with the silly estimator $\theta_s(y) = 0$.

From exercise 6:

$$\text{BMSE}(\theta_n(y)) = \sigma_e^2 = 1$$

$$\text{BMSE}(\theta_s(y)) = \sigma_x^2 + \mu_x^2 = 2$$

\uparrow Variance \uparrow squared mean of x

Todo: Einfügen: Example Gaussian Joint Density Conditioning

We want to find θ_y^{opt} minimizing

$$\text{BMSE}(\theta_y) = \int \underbrace{\int ||\theta_y - \mathbf{x}||^2 p(\mathbf{x}|\mathbf{y}) d\mathbf{x}}_{E[||\theta_y - \mathbf{x}||^2 | \mathbf{y}]} p(\mathbf{y}) d\mathbf{y}$$

As $p(\mathbf{y})$ is always positive, we only need to minimize the inner integral for all \mathbf{y} .

$$\begin{aligned} & E[(\theta_y^{opt} - \mathbf{x})^T (\theta_y^{opt} - \mathbf{x}) | \mathbf{y}] \\ &= E[(\theta_y^{opt} - \mathbf{c} + \mathbf{c} - \mathbf{x})^T (\theta_y^{opt} - \mathbf{c} + \mathbf{c} - \mathbf{x}) | \mathbf{y}] \\ &= E[(\theta_y^{opt} - \mathbf{c})^T (\theta_y^{opt} - \mathbf{c}) + (\theta_y^{opt} - \mathbf{c})^T (\mathbf{c} - \mathbf{x}) + (\mathbf{c} - \mathbf{x})^T (\theta_y^{opt} - \mathbf{c}) \\ &\quad + (\mathbf{c} - \mathbf{x})^T (\mathbf{c} - \mathbf{x}) | \mathbf{y}] \quad | \mathbf{c} = E[\mathbf{x} | \mathbf{y}] \\ &= E[(\theta_y^{opt} - E[\mathbf{x} | \mathbf{y}])^T (\theta_y^{opt} - E[\mathbf{x} | \mathbf{y}]) | \mathbf{y}] \\ &\quad + E[(E[\mathbf{x} | \mathbf{y}] - \mathbf{x})^T (E[\mathbf{x} | \mathbf{y}] - \mathbf{x}) | \mathbf{y}] \quad | \theta_y^{opt} = E[\mathbf{x} | \mathbf{y}] \\ &= E[(E[\mathbf{x} | \mathbf{y}] - \mathbf{x})^T (E[\mathbf{x} | \mathbf{y}] - \mathbf{x}) | \mathbf{y}] \end{aligned}$$

Unbiasedness:

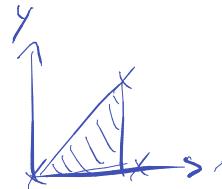
$$\begin{aligned} E_{\mathbf{y}}[E[\mathbf{x}|\mathbf{y}]] &= \int E[\mathbf{x}|\mathbf{y}]p(\mathbf{y})d\mathbf{y} \\ &= \int \int \mathbf{x}p(\mathbf{x}|\mathbf{y})d\mathbf{x}p(\mathbf{y})d\mathbf{y} \\ &= \int \mathbf{x}p(\mathbf{x})d\mathbf{x} \\ &= E[\mathbf{x}] \end{aligned}$$

$$\begin{aligned} E[\theta_y^{opt} - \mathbf{x}] &= \underbrace{E_{\mathbf{y}}[E[\mathbf{x}|\mathbf{y}]]}_{E[\mathbf{x}]} - E[\mathbf{x}] \\ &= 0 \end{aligned}$$

Problem 7 – MMSE Estimation

Consider a two-dimensional joint density

$$p(x, y) = \begin{cases} 2 & \text{for } [x, y] \in S \\ 0 & \text{otherwise} \end{cases}$$



where the set S is a triangle specified by the points $[0, 0]$, $[1, 0]$, and $[1, 1]$.

- What is the conditional distribution $p(x|y)$?
- What is the MMSE estimator of x given y ?
✓ measurement mean sqa. error
- What is the conditional MSE associated to the MMSE estimator?
- What is the unconditional MSE associated to the MMSE estimator?

$$\text{MSE}(\hat{\theta}_Y) = E_{x,y} (\|\hat{\theta}_Y - x\|^2)$$

$$\text{MSE}(\hat{\theta}_Y | Y) = E_x (\|\hat{\theta}_Y - x\|^2 | Y)$$

If $x \sim \mathcal{U}(a, b)$, we have

- $p(x) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$

- $E[x] = \frac{1}{2}(a + b)$

- $\text{Var}[x] = \frac{1}{12}(b - a)^2$

Problem 7: Solution

a) $p(x|y) = \begin{cases} \frac{1}{1-\hat{y}} & \text{for } x \in [\hat{y}, 1] \\ 0 & \text{otherwise} \end{cases}$

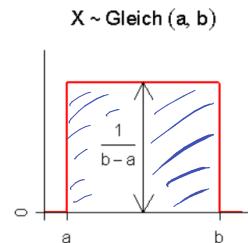
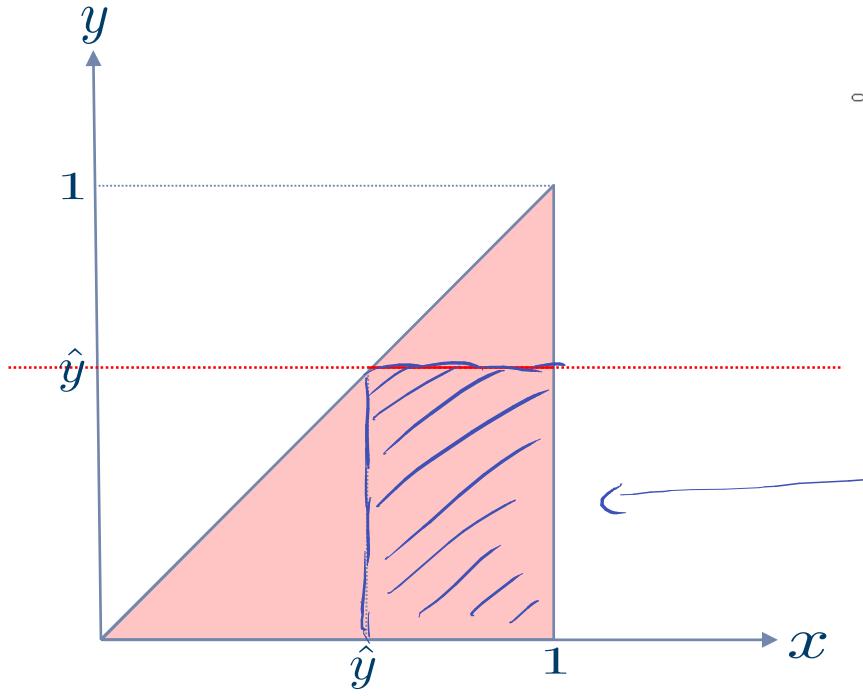
b) $\theta_y = E[x|y] = \frac{1}{2}(y + 1)$
under
bounds

c) $MSE(\theta_y|y) = E[(x - E(x|y))^2|y] = \text{Var}[x|y] = \frac{1}{12}(1-y)^2$
estimator

denote $p(x|y) = p(x,y)/p(y)$

If $x \sim U(a,b)$, we have

- $p(x) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$
- $E[x] = \frac{1}{2}(a+b)$
- $\text{Var}[x] = \frac{1}{12}(b-a)^2$



Problem 7: Solution

$$\iint \|x - \theta_y\|^2 \cdot p(x, y) dx dy$$

d) Non Conditional MSE

$$\text{MSE}(\theta_y) = E[||x - \theta_y||^2]$$

$$= E_y[E[||x - E[x|y]||^2|y]]$$

$$= E_y[\text{MSE}(\theta_y|y)]$$

$$= E_y\left[\frac{1}{12}(1-y)^2\right]$$

$$= \int \frac{1}{12}(1-y)^2 p(y) dy$$

$$= \int_0^1 \frac{1}{12}(1-y)^2 2(1-y) dy$$

$$= -\frac{1}{6} \frac{1}{4}(1-y)^4 \Big|_0^1$$

$$= \frac{1}{24}$$

$$p(x, y) = \begin{cases} 2 & , \text{ for } y \in [0, 1], x \in [y, 1] \\ 0 & , \text{ otherwise} \end{cases}$$

$$p(y) = \int p(x, y) dx$$

$$= \begin{cases} \int_y^1 2 dx, & y \in [0, 1] \\ 0, & \text{otherwise} \end{cases}$$

$$= \begin{cases} 2(1-y), & y \in [0, 1] \\ 0, & \text{otherwise} \end{cases}$$

Assume a robot in 2D-space. Its position is modeled as a Gaussian random variable. The prior has $\hat{\mathbf{x}}_0 = [0 \quad 0]^T$ and

$$\mathbf{C}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

a) A sensor measures the robot's true position. Formulate and implement a measurement equation assuming independent zero-mean Gaussian noise with

$$\mathbf{R} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix} .$$

b) Implement a function which samples a true position of \mathbf{x} from the prior and then generates a measurement from the true position.

c) Implement the Kalman update formula to calculate the posterior distribution.

Homework 7

- d) Now, assume the sensor will provide 5 measurements in a row. Use the Kalman filter update formulas to update the robot's state recursively.
- e) Visualize the robot's covariance matrix as an ellipse and observe how it changes with each update.

Sensor Data Fusion

Kalman Filter (Part 2)

Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

1. Introduction (**live**)
2. Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas (**live**)
7. Bayesian approach
8. Bayesian approach: Linear and Gaussian case (**live**)
9. Kalman filter
10. Extended Kalman filter *- great because it only needs mean & variance
- optimal for gaussian & BLUE for non-gaussians*
11. Dynamic models: Tracking
12. Advanced Topic (**live**)
13. Advanced Topic (2) (**live**)
14. Summary and Discussion (**live**)

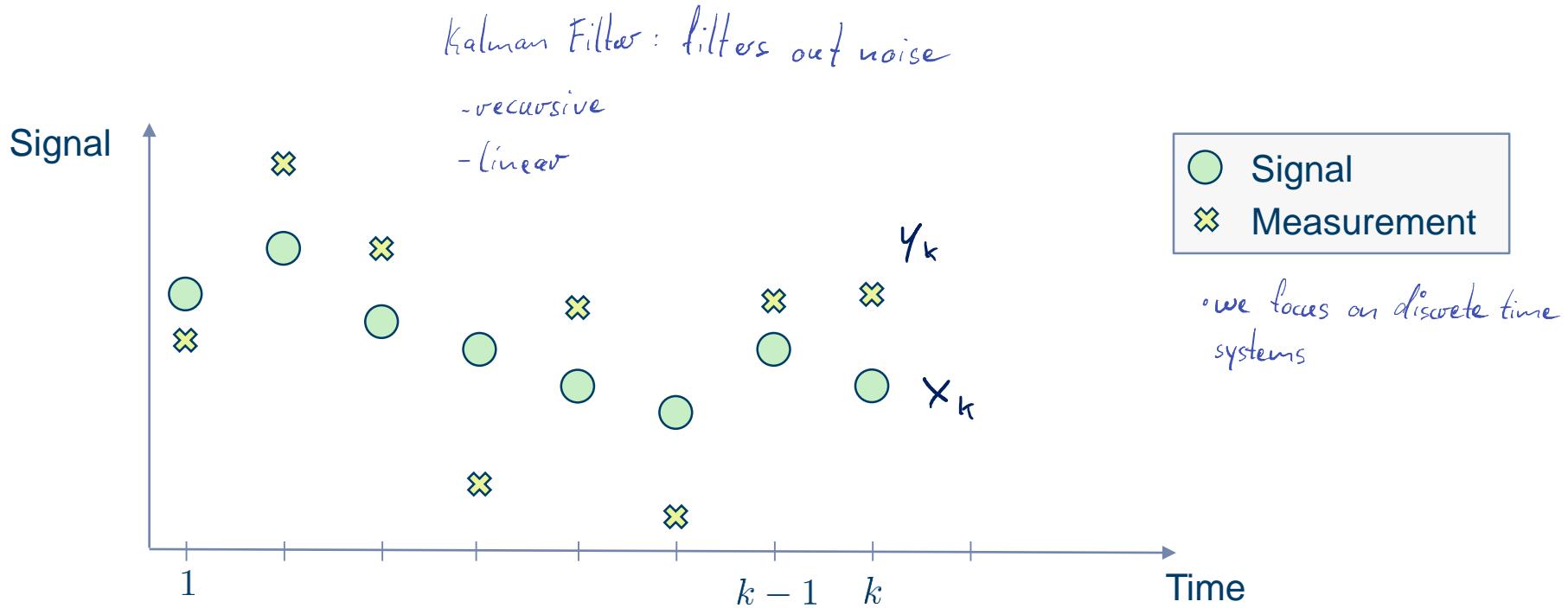
- Born in Budapest, 1930
- Emigrated to the US in 1943
- BA (1953) and MA (1954) from MIT
- PhD, 1957, Columbia University, NYC
- Professor at Stanford University (1964-1971), University of Florida (1971-1992) and ETH (since 1973)



- First publication of the Kalman Filter in 1960
- First implementation by Stanley F. Schmidt (NASA Ames Research Center)
- Used for trajectory estimation in the Apollo Program
- Many co-inventors and related works (e.g., Swerling, Bucy, Schmidt, Stratonovich, Gauss, ...)



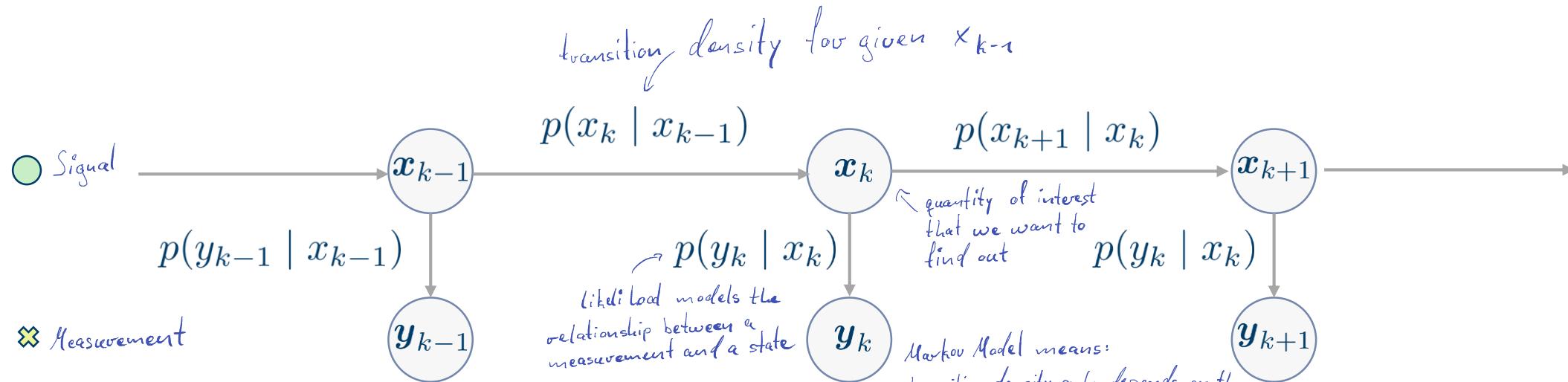
- The navigation system of Eagle lunar module in 1969 was based on an Extended Kalman Filter (EKF)
- The dynamic model was Newton's gravitation law.
- The measurements at lunar landing were the radar readings.
- On rendezvous with the command ship the orientation was computed with gyroscopes and their biases were also compensated with the radar.



Estimation of a dynamic process

- Time-varying signal
 - Noisy measurements
- Determine the signal based on the measurements

State Space Model (Hidden Markov Model)



- State vector $x_k \in \mathbb{R}^n$ for $k \in \mathbb{Z}$
 \hookrightarrow hidden variable
- Observations/Measurements: $y_k \in \mathbb{R}^m$

- Prior: $x_0 \sim p(x_0)$
- Transition probability density function: $p(x_k | x_{k-1})$
- Likelihood function: $p(y_k | x_k)$

$p(x_{k+1} | y_{1:k})$
 y_1 to y_k (all states until x)

π
Measur. equation to specify
Likelihood function

- State vector at discrete time $k \in \mathbb{N}_0$:

„Kalman Filter is tailored to linear systems, corrupted by additive noise.“

$$x_k \in \mathbb{R}^n$$

just denotation, no further meaning

with initial mean $E[x_0] = \hat{x}_0$ and covariance $\text{Cov}[x_0] = \mathbf{C}_0^{xx}$

t_{initial}

- Process model:

control input (fixed value) e.g. steering angle, acceleration etc.
 ↓ motion noise (mean 0)

$$x_{k+1} = \mathbf{A}_k x_k + \mathbf{B}_k u_k + w_k$$

- State transition matrix $\mathbf{A}_k \in \mathbb{R}^{n \times n}$
- Control input $u_k \in \mathbb{R}^p$ ← deterministic value, directly available to filter
- Control input matrix $\mathbf{B}_k \in \mathbb{R}^{n \times p}$
- Zero-mean white process noise $w_k \in \mathbb{R}^n$:

$$x_{k+1} = \mathbf{A}_k x_k + \mathbf{B}_k (u_k + \omega_k)$$

- Kalman F. requires linearity
- RV mustn't be gaussian

Control input matrix \mathbf{B}_k :
 models the effect of u_k on x_k

$$\rightarrow * E[w_k] = \mathbf{0}_n \quad \text{mean zero}$$

$$* E[w_k w_k^T] = \mathbf{C}_k^{ww} \quad \text{covariance known, can be written this way, because mean 0}$$

$$\rightarrow * E[w_k w_l^T] = \mathbf{0}_{n \times n} \text{ for } k \neq l$$

↳ noise uncorrelated from time to time

- Measurement model:

$$y_k = \mathbf{H}_k x_k + v_k$$

- Measurement matrix $\mathbf{H}_k \in \mathbb{R}^{m \times n}$
- Zero-mean white measurement noise $v_k \in \mathbb{R}^m$ with

- * $E[v_k] = \mathbf{0}_n$ *white noise important for recursive iteration*

- * $E[v_k v_k^T] = \mathbf{C}_k^{vv}$

- * $E[v_k v_l^T] = \mathbf{0}_{m \times m}$ for $k \neq l$
↳ measurement noise uncorrelated over time

- Initial state, and all noise vectors mutually uncorrelated

- Recursively calculates estimate \hat{x}_k and error covariance C_k^{xx} for time k

- Time update:

$$\hat{x}_{k+1|k} = A_k \hat{x}_k + B u_k$$

state transition matr. A_k
 (const.) bzw. expected
 Bewegung
 auch Motion model
 current estimate
 \hat{x}_k

Control input matrix B_k :
 models the effect of
 u_k on x_k
 u_k : steering

typically grows

prediction is more unprecise (only when noise > 0)

$$C_{k+1|k}^{xx} = A_k C_k^{xx} A_k^T + C_k^{ww}$$

state covariance
 cov from steering
 error covariance

- Measurement update:

$$\hat{x}_{k+1} = \hat{x}_{k+1|k} + K_{k+1} (y_{k+1} - H \hat{x}_{k+1|k})$$

$$C_{k+1}^{xx} = C_{k+1|k}^{xx} - K_{k+1} H C_{k+1|k}^{xx} K_{k+1}^T$$

predicted/expected measurement
 difference between measurement & expected measurement
 (also called "innovation")

prediction
 measurement matrix
 prediction

with Kalman gain

states how much the "prediction"

is updated by the "innovation"!

$$K_{k+1} = C_{k+1|k}^{xx} H^T (H C_{k+1|k}^{xx} H^T + C_{k+1}^{vv})^{-1}$$

inverse not divided

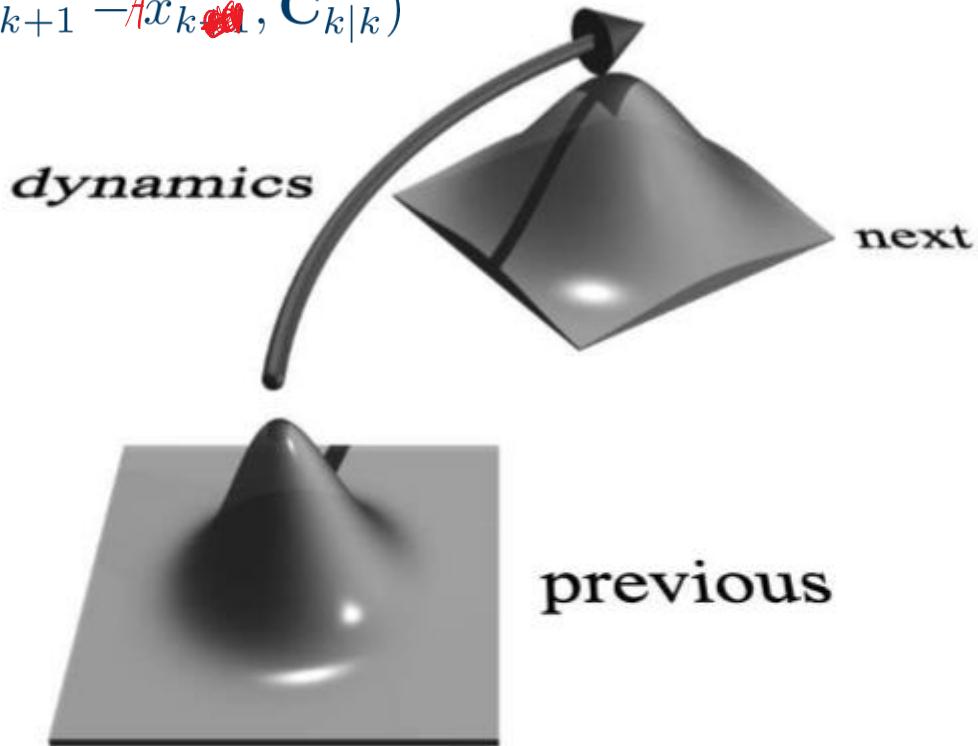
special case: if measurement noise $C_{k+1|k}^{xx}$ is extremely large, the prediction won't be modified at all by the Kalman Filter update formulas

for linear dynamic systems

Anmerkung: Formeln müssen nicht auswendig gelernt werden

$$\boldsymbol{x}_{k+1} = \mathbf{A} \cdot \boldsymbol{x}_k + \boldsymbol{w}_k$$

$$p(\boldsymbol{x}_{k+1} | \boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{x}_{k+1} - \mathbf{A}\boldsymbol{x}_k, \mathbf{C}_{k|k})$$

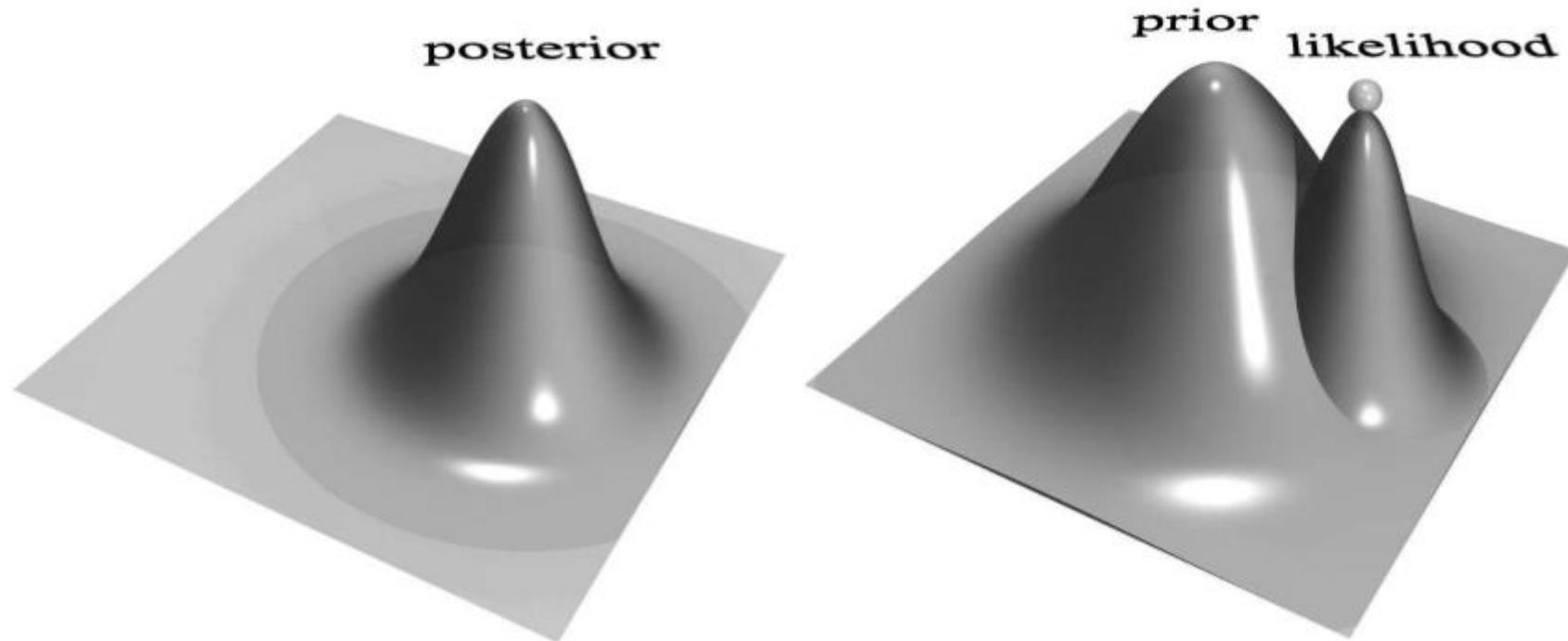


$$p(\boldsymbol{x}_{k+1} | \boldsymbol{y}_{1:k}) = \int p(\boldsymbol{x}_{k+1} | \boldsymbol{x}_k) \cdot p(\boldsymbol{x}_k | \boldsymbol{y}_{1:k}) d\boldsymbol{x}_k$$

$$p(\boldsymbol{x}_{k+1}, \boldsymbol{x}_k | \boldsymbol{y}_{1:k})$$

Simo Särkkä, Bayesian Filtering and Smoothing. Cambridge University Press, 2013

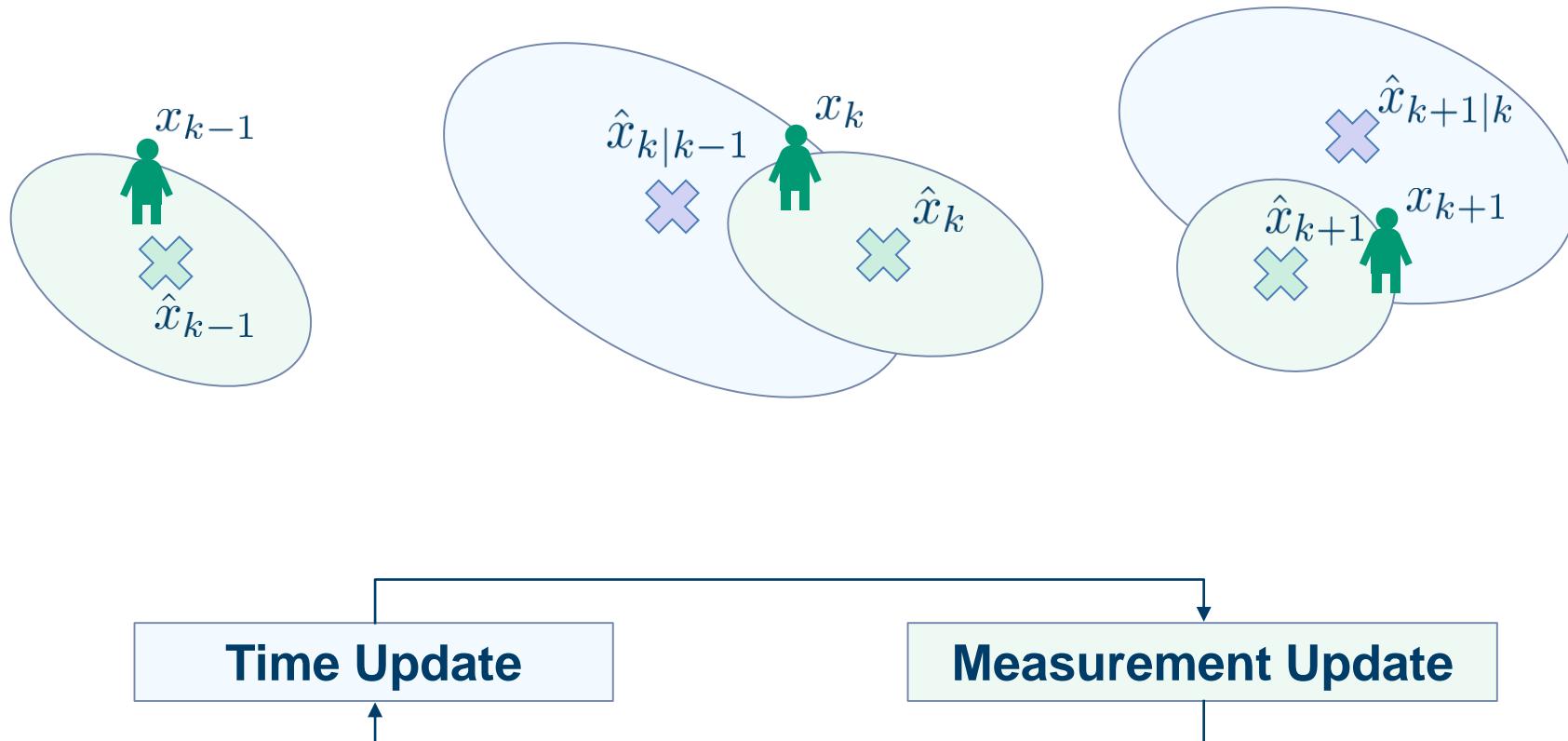
$$p(x_k|y_{1:k}) \propto p(x_k|y_{1:k-1}) \cdot p(y_k|x_k)$$



$$\mathbf{y}_k = \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{v}_k$$

$$p(y_k|x_k) = \mathcal{N}(y_k - \mathbf{H}_k \cdot \mathbf{x}_k, \mathbf{C}_k^v)$$

Simo Särkkä, Bayesian Filtering and Smoothing. Cambridge University Press, 2013



- Define random vector

$$\begin{bmatrix} x_k \\ w_k \end{bmatrix}$$

and matrix

$$[\mathbf{A}_k \quad \mathbf{I}]$$

- Mean and covariance of linear transformation:

$$\begin{aligned} E[\begin{bmatrix} \mathbf{A}_k & \mathbf{I} \end{bmatrix} \begin{bmatrix} x_k \\ w_k \end{bmatrix} + \mathbf{B}u_k] &= [\mathbf{A}_k \quad \mathbf{I}] \begin{bmatrix} \hat{x}_k \\ \mathbf{0}_m \end{bmatrix} + \mathbf{B}u_k \\ &= \mathbf{A}_k \hat{x}_k + \mathbf{B}u_k \end{aligned}$$

$$\begin{aligned} \text{Cov}[\begin{bmatrix} \mathbf{A}_k & \mathbf{I} \end{bmatrix} \begin{bmatrix} x_k \\ w_k \end{bmatrix} + \mathbf{B}u_k] &= [\mathbf{A}_k \quad \mathbf{I}] \begin{bmatrix} \mathbf{C}_k^{xx} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_k^{ww} \end{bmatrix} [\mathbf{A}_k \quad \mathbf{I}]^T \\ &= \mathbf{A}_k \mathbf{C}_k^{xx} \mathbf{A}_k^T + \mathbf{C}_k^{ww} \end{aligned}$$

- If RVs are Gaussian, the prediction is Gaussian, too.

- State vector at discrete time $k \in \mathbb{N}_0$:

$$x_k \in \mathbb{R}^n$$

with initial mean $E[x_0] = \hat{x}_0$ and covariance $\text{Cov}[x_0] = \mathbf{C}_0^{xx}$

- Process model:

$$x_{k+1} = a_k(x_k, u_k) + w_k$$

state control input

- State transition function $a_k : \mathbb{R}^n \times \mathbb{R}^p \rightarrow \mathbb{R}^n$
- Control input $u_k \in \mathbb{R}^p$
- Zero-mean white (Gaussian) process noise $w_k \in \mathbb{R}^n$:
 - * $E[w_k] = \mathbf{0}_n$
 - * $E[w_k w_k^T] = \mathbf{C}_k^{ww}$
 - * $E[w_k w_l^T] = \mathbf{0}_{n \times n}$ for $k \neq l$

- Measurement model:

$$y_k = h_k(x_k) + v_k$$

- Measurement function $h_k : \mathbb{R}^n \rightarrow \mathbb{R}^{n_{\text{out}}}$
- Zero-mean white (Gaussian) measurement noise $v_k \in \mathbb{R}^m$ with
 - * $\mathbb{E}[v_k] = \mathbf{0}_n$
 - * $\mathbb{E}[v_k v_k^T] = \mathbf{C}_k^{vv}$
 - * $\mathbb{E}[v_k v_l^T] = \mathbf{0}_{m \times m}$ for $k \neq l$
- Initial state, and all noise vectors mutually uncorrelated

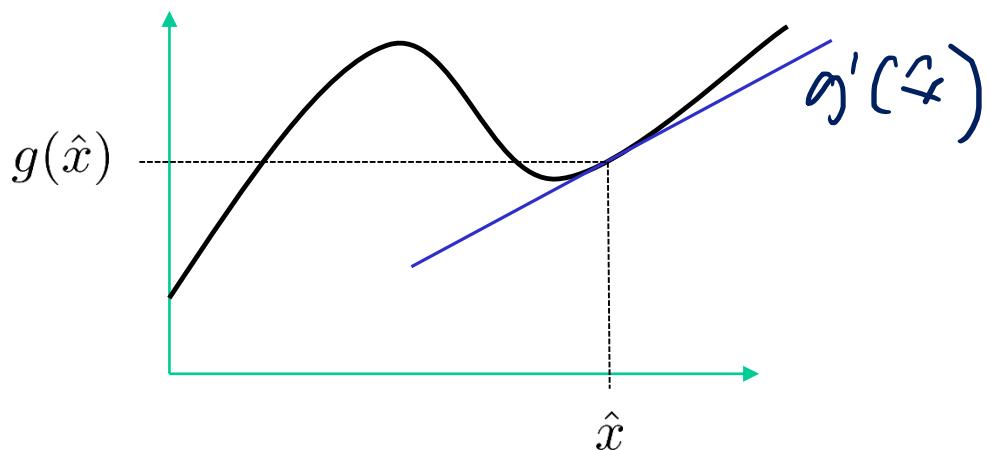
Taylor Series Expansion (1D)

- Taylor series expansion of 1D function $g : \mathbb{R} \rightarrow \mathbb{R}$ with $\hat{x} \in \mathbb{R}$:

$$g(\hat{x}) + \frac{g'(\hat{x})}{1!}(x - \hat{x}) + \frac{g''(\hat{x})}{2!}(x - \hat{x})^2 + \frac{g^{(3)}(\hat{x})}{3!}(x - \hat{x})^3 + \dots$$

- Linearization of $g : \mathbb{R} \rightarrow \mathbb{R}$ around $\hat{x} \in \mathbb{R}$:

$$g(x) \approx g(\hat{x}) + g'(\hat{x})(x - \hat{x})$$



Linearization of Multivariate Functions

PREVIOUS
LECTURE

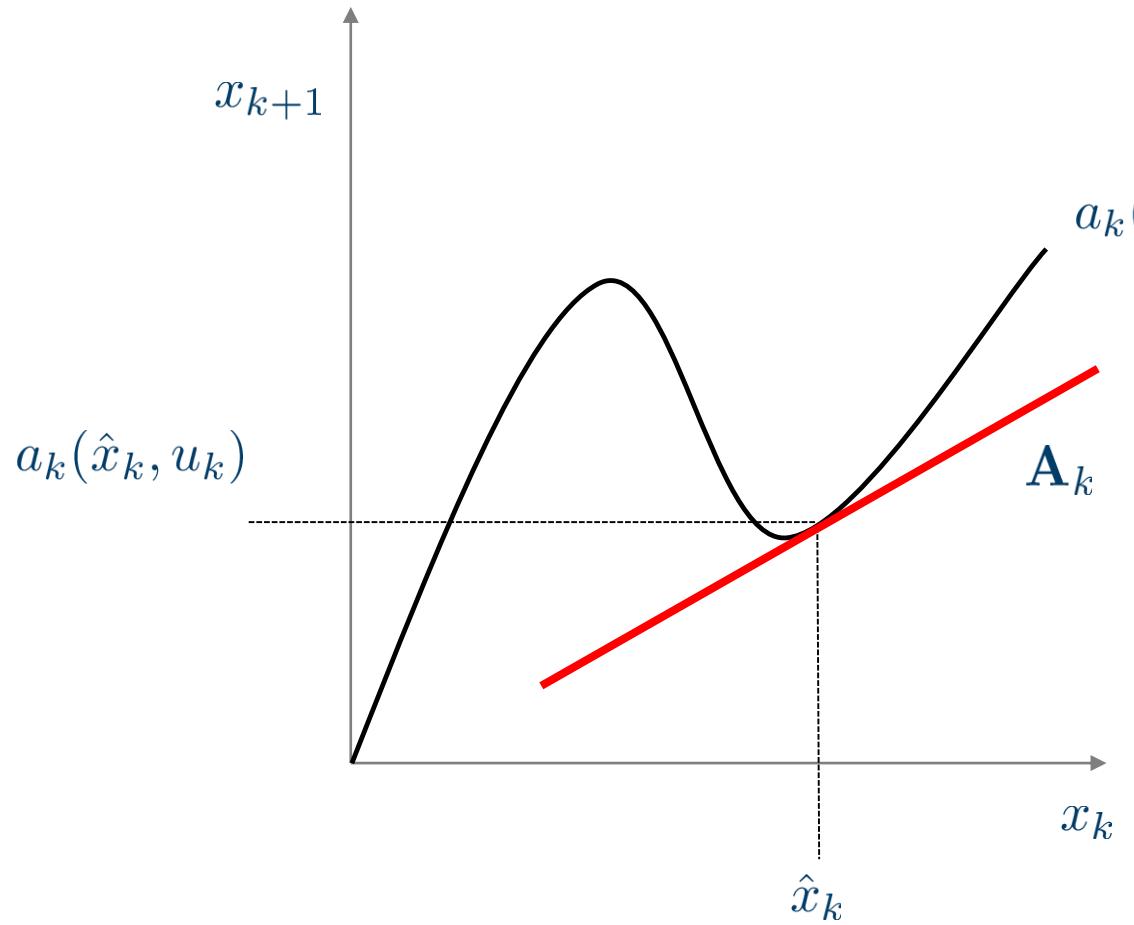
- $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with component functions $g = [g_1, \dots, g_m]^T$
- Linearization point $\hat{x} \in \mathbb{R}^n$
- Linearization:

$$g(x) \approx g(\hat{x}) + \mathbf{G}_{\hat{x}}(x - \hat{x})$$

with Jacobi matrix

$$\mathbf{G}_{\hat{x}} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1}(\hat{x}) & \frac{\partial g_1}{\partial x_2}(\hat{x}) & \dots & \frac{\partial g_1}{\partial x_n}(\hat{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_1}(\hat{x}) & \frac{\partial g_m}{\partial x_2}(\hat{x}) & \dots & \frac{\partial g_m}{\partial x_n}(\hat{x}) \end{bmatrix}$$

Extended Kalman Filter: Time Update - Intuition



$$x_{k+1} = a_k(x_k, u_k) + w_k$$

g(x_k) non linear part
(needs to be linearized)
 \downarrow
noise

$$x_{k+1} = a_k(\hat{x}_k, u_k) + A_k(x_k - \hat{x}_k) + w_{k+1}$$

A_k constant

=> linear with respect to x_k

- Linearization at current estimate \hat{x}_{k-1} :

$$a(x_{k-1}, u_k) \approx a(\hat{x}_{k-1}, u_k) + \mathbf{A}_k(x_{k-1} - \hat{x}_{k-1})$$

with Jacobi Matrix

$$\mathbf{A}_k = \frac{\partial a}{\partial x_{k-1}}(\hat{x}_{k-1}, u_k)$$

- Kalman filter prediction:

$$\begin{aligned}\hat{x}_{k|k-1} &= a(\hat{x}_{k-1}, u_k) \\ \mathbf{C}_{k|k-1} &= \mathbf{A}_k \mathbf{C}_{k-1}^{xx} \mathbf{A}_k^T + \mathbf{C}_k^{ww}\end{aligned}$$

system noise
↑ *↑ old covariance*
Jac. Mat.

$$y = \underline{h(x)} + v$$

- Linearization at prediction $\hat{x}_{k|k-1}$:

$$h_k(x_k) \approx h_k(\hat{x}_{k|k-1}) + \mathbf{H}_k(x_k - \hat{x}_{k|k-1})$$

with Jacobi Matrix

$$\mathbf{H}_k = \frac{\partial h_k}{\partial x_k}(\hat{x}_{k|k-1})$$

$$\begin{aligned} y - h(\hat{x}_{k|k-1}) + H \hat{x}_{k|k-1} \\ = \underline{y} = H x_k + v_k \end{aligned}$$

- Kalman filter measurement update: $(y^* - H \hat{x}_{k|k-1})$

$$\hat{x}_k = \hat{x}_{k|k-1} + \mathbf{K}_k(y_k - h(\hat{x}_{k|k-1}))$$

$$\mathbf{C}_k^{xx} = \mathbf{C}_{k|k-1}^{xx} - \mathbf{K}_k \mathbf{H}_k \mathbf{C}_{k|k-1}^{xx}$$

with Kalman gain

$$\mathbf{K}_k = \mathbf{C}_{k|k-1}^{xx} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{C}_{k|k-1}^{xx} \mathbf{H}_k^T + \mathbf{C}_k^{vv})^{-1}$$

Todo: Folie fehlt (important)

Radar measurement equation

Questions?

→ Stud.IP or e-mail

Sensor Data Fusion

Exercise 8

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 7 solution
- Example - Gaussian Joint Density
- Problem 8 - LMMSE estimator
- Homework 8 presentation

What are the two special cases which need to be fulfilled for the Kalman filter to be optimal?

- linearity
- Gaussian densities

Two Special Cases

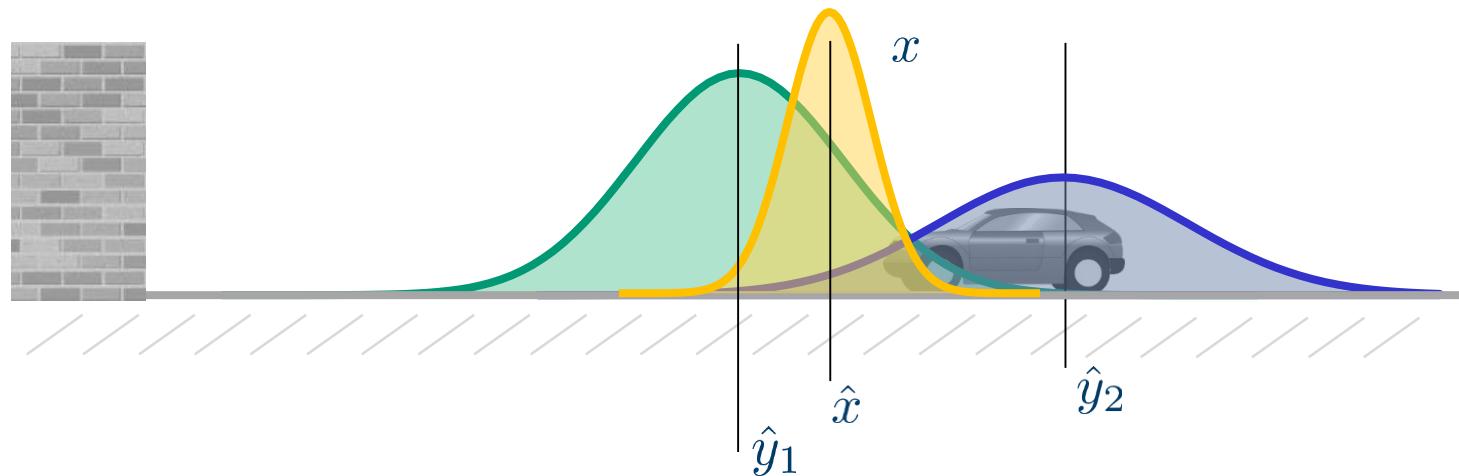
1. **All probability densities are Gaussian**
→ Posterior is Gaussian and can be derived analytically
2. **Linear estimator**
→ Linear MMSE can be derived analytically

Both special cases lead to the same result:

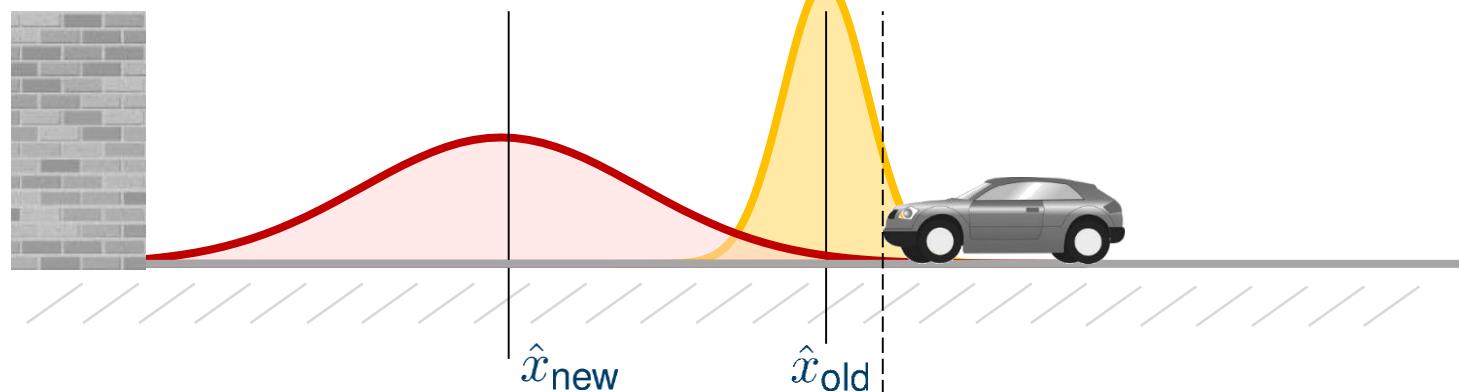
Kalman Filter Update Equations

- Optimal Bayesian estimator in case of Gaussian densities
- Best linear estimator otherwise

What happens to the state variance during measurement update? What happens during time update?



measurement update:
variance decreases



time update:
variance increases

Assume during the time update, the state moves with a noise corrupted velocity. What would happen to the state covariance if we want to estimate the state for $T \rightarrow \infty$?

We have $\sigma_{x_{k+1}}^2 = \sigma_{x_k}^2 + T^2 \sigma_v^2$.

For $T \rightarrow \infty$, we get $\sigma_{x_{k+1}}^2 \rightarrow \infty$.

Todo Folie ein.

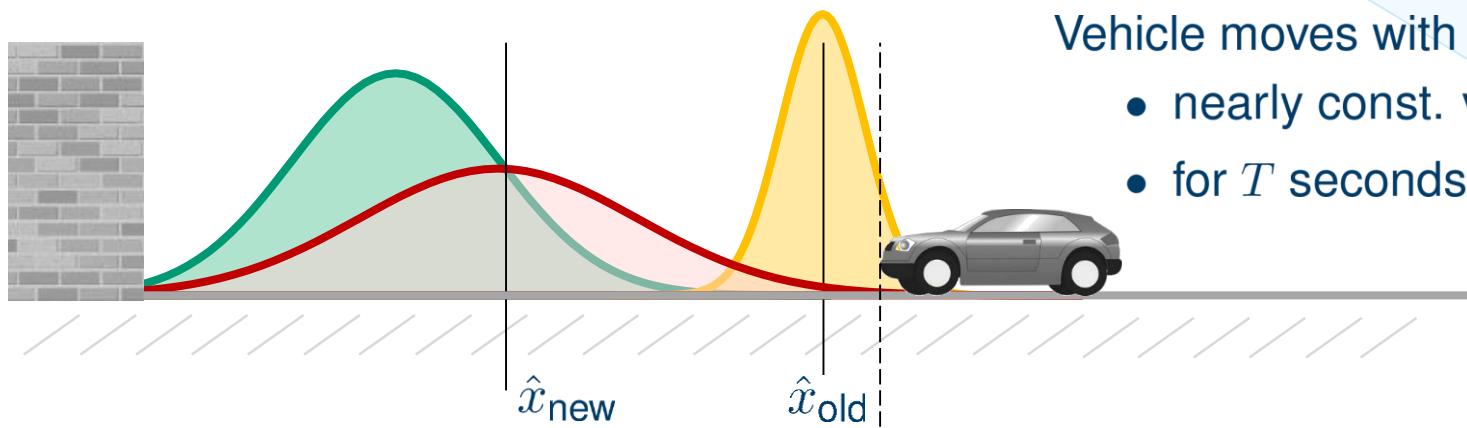
What's the idea of the Ext. Kalman Filter

Setting X

Todo Folie ein.

Example EKF issues

Time Update: Prediction of an Estimate



Vehicle moves with

- nearly const. velocity v
- for T seconds.

$$\hat{x}_{\text{old}}$$

$$\sigma_{\text{old}}^2$$

Discrete-time Motion Model:

$$x_{\text{new}} = x_{\text{old}} + T \cdot (v + e_v)$$

with velocity v and zero-mean noise e_v with variance σ_v^2

$$\hat{x}_{\text{new}} = \hat{x}_{\text{old}} + T \cdot v$$

$$\sigma_{\text{new}}^2 = \sigma_{\text{old}}^2 + T^2 \sigma_v^2$$

Variance increases

Next measurement: fusion with prediction

Assume a robot in 2D-space. Its position is modeled as a Gaussian random variable. The prior has $\hat{\mathbf{x}}_0 = [0 \quad 0]^T$ and

$$\mathbf{C}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

a) A sensor measures the robot's true position. Formulate and implement a measurement equation assuming independent zero-mean Gaussian noise with

$$\mathbf{R} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix} .$$

b) Implement a function which samples a true position of \mathbf{x} from the prior and then generates a measurement from the true position.

c) Implement the Kalman update formula to calculate the posterior distribution.

- d) Now, assume the sensor will provide 5 measurements in a row. Use the Kalman filter update formulas to update the robot's state recursively.
- e) Visualize the robot's covariance matrix as an ellipse and observe how it changes with each update.

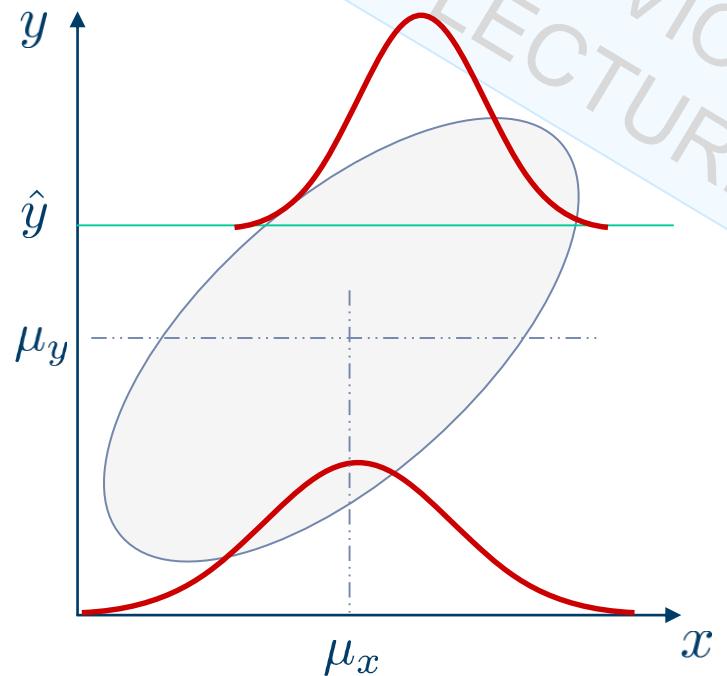
Example – Gaussian Joint Density: Conditioning

Gaussian joint density:

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \underbrace{\begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{bmatrix}}_{\mathbf{C}}\right)$$

$$\mu_x^+ = \mu_x + \mathbf{C}_{xy} \mathbf{C}_{yy}^{-1} (\hat{y} - \mu_y)$$

$$\mathbf{C}^+ = \mathbf{C}_{xx} - \mathbf{C}_{xy} \mathbf{C}_{yy}^{-1} \mathbf{C}_{yx}$$



Conditioning on y :

$$p(x|y = \hat{y}) = \mathcal{N}(\mu_x^+, \mathbf{C}^+)$$

Remark 1:

$$\underbrace{\begin{bmatrix} \mathbf{I} & -\mathbf{C}_{xy}\mathbf{C}_{yy}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{T}} \cdot \underbrace{\begin{bmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{bmatrix}}_{\mathbf{C}} \cdot \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx} & \mathbf{I} \end{bmatrix}}_{\mathbf{T}^T} = \underbrace{\begin{bmatrix} \mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{yy} \end{bmatrix}}_{\mathbf{D}}$$

Hence $\mathbf{C} = \mathbf{T}^{-1}\mathbf{D}\mathbf{T}^{-T}$ and $\mathbf{C}^{-1} = \mathbf{T}^T\mathbf{D}^{-1}\mathbf{T}$

- Conditioning:

$$\begin{aligned} p(x|y) &= \frac{p(x,y)}{p(y)} \\ &= \frac{\sqrt{(2\pi)^m}}{\sqrt{(2\pi)^{m+n}}} \frac{\sqrt{\det(\mathbf{C}_{yy})}}{\sqrt{\det(\mathbf{C})}} \frac{\exp\left(-\frac{1}{2}\begin{bmatrix}x - \mu_x \\ y - \mu_y\end{bmatrix}^T \mathbf{C}^{-1} \begin{bmatrix}x - \mu_x \\ y - \mu_y\end{bmatrix}\right)}{\exp\left(-\frac{1}{2}(y - \mu_y)^T \mathbf{C}_{yy}^{-1} (y - \mu_y)\right)} \end{aligned}$$

- With Remark 1:

$$\det\{\mathbf{C}\} = \det\{\mathbf{C}_{xx} - \mathbf{C}_{xy} \mathbf{C}_{yy}^{-1} \mathbf{C}_{yx}\} \det\{\mathbf{C}_{yy}\}$$

As

$$\begin{aligned}
 & \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix}^T \mathbf{C}^{-1} \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix} \\
 &= \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix}^T \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{yy} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{I} & -\mathbf{C}_{xy}\mathbf{C}_{yy}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix} \\
 &= \begin{bmatrix} x - \mu_x^+ \\ y - \mu_y \end{bmatrix}^T \begin{bmatrix} \mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{yy} \end{bmatrix}^{-1} \begin{bmatrix} x - \mu_x^+ \\ y - \mu_y \end{bmatrix} \\
 &= (x - \mu_x^+)^T (\mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx})^{-1} (x - \mu_x^+) + (y - \mu_y)^T \mathbf{C}_{yy}^{-1} (y - \mu_y)
 \end{aligned}$$

with $\mu_x^+ = \mu_x + \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}(y - \mu_y)$, the conditional distribution from the previous slide simplifies to the desired formula.

Consider two scalar random variables $x \in \mathbb{R}$ and $y \in \mathbb{R}$. Derive the linear minimum mean square error (LMMSE) estimator of x given y , i.e., find $K \in \mathbb{R}$ and $b \in \mathbb{R}$, so that the linear estimator $\theta_y = Ky + b$ minimizes

$$\mathsf{E}_{x,y}\{(x - \theta_y)^2\} .$$

Assume the random variables expectations $\mathsf{E}[x] = \hat{x}$ and $\mathsf{E}[y] = \hat{y}$, variances $\text{Var}[x] = C_{xx}$ and $\text{Var}[y] = C_{yy}$, and covariance $\text{Cov}[x, y] = C_{xy}$ to be given.

We aim to minimize $E[(x - (Ky + b))^2]$. From

$$\frac{\partial}{\partial b} E[(x - (Ky + b))^2] = E[-2(x - (Ky + b))] \stackrel{!}{=} 0$$

should be
↓

we get $b = E[x] - K E[y]$. Repeating the process for K leads us to

$$\frac{\partial}{\partial K} E[(x - (Ky + b))^2] = E[-2y(x - (Ky + b))] \stackrel{!}{=} 0$$

solving that, we get

$$E[-2xy + 2Ky^2 + 2by] = 0$$

$$2K E[y^2] = 2 E[xy] - 2b E[y]$$

$$K \text{Var}[y] + K E[y]^2 = E[xy] - b E[y]$$

$$K \text{Var}[y] = E[xy] - b E[y] - K E[y]^2$$

Now substituting b with the previously determined formula, we get

$$K \operatorname{Var}[y] = E[xy] - (E[x] - K E[y]) E[y] - K E[y]^2$$

$$K \operatorname{Var}[y] = E[xy] - E[x] E[y] + K E[y]^2 - K E[y]^2$$

$$K = \operatorname{Cov}[x, y] \operatorname{Var}[y]^{-1}$$

$$K = C_{xy} C_{yy}^{-1}$$

Substituting K with that result brings us $b = \hat{x} - C_{xy} C_{yy}^{-1} \hat{y}$. Now substituting K and b with their formulas in θ_y gives us the optimal estimator

$$\theta_y = \hat{x} + C_{xy} C_{yy}^{-1} (y - \hat{y})$$

Assume a robot in 1D-space at position x moving at time k with velocity v_k forward. The prior of x at time $k = 0$ is a Gaussian with $\hat{x}_0 = 5m$ and $\sigma_{x,0}^2 = 2m^2$.

- a) Draw x_0 from the prior. The robot moves $x_{k+1} = x_k + T(v_k + e_v)$ with equidistant time steps $T = 1s$ and velocity error $e_v \sim \mathcal{N}(0\frac{m}{s}, 0.5(\frac{m}{s})^2)$. Write a function which moves the robot for one time step with constant input $v_k = 1\frac{m}{s}$.

Hint: When using `np.random.normal`, you need to pass the standard deviation as the scale parameter. Remember how standard deviation and variance (which is given here) are related, and make sure you use `np.sqrt(...)` as necessary.

- b) In each time step, a sensor measures the robot's position. Implement a measurement equation assuming independent zero-mean Gaussian noise of $e_s \sim \mathcal{N}(0m, 0.2m^2)$.

- c) Now, implement the time update formulas from the lecture to get the next predicted state and variance for a single time step.
- d) Next, implement the measurement update to get the updated state and variance after a measurement was received.
- e) Finally, put the code of all functions together to run the simulation and create a visualization for it.

Hint: all plotting functions are already implemented, along with the necessary variable definitions. However, you still need to fill out certain small blocks of code that are responsible for generating the initial state of x , and the measurement and time update steps.

Sensor Data Fusion

Basic Motion Models for Tracking

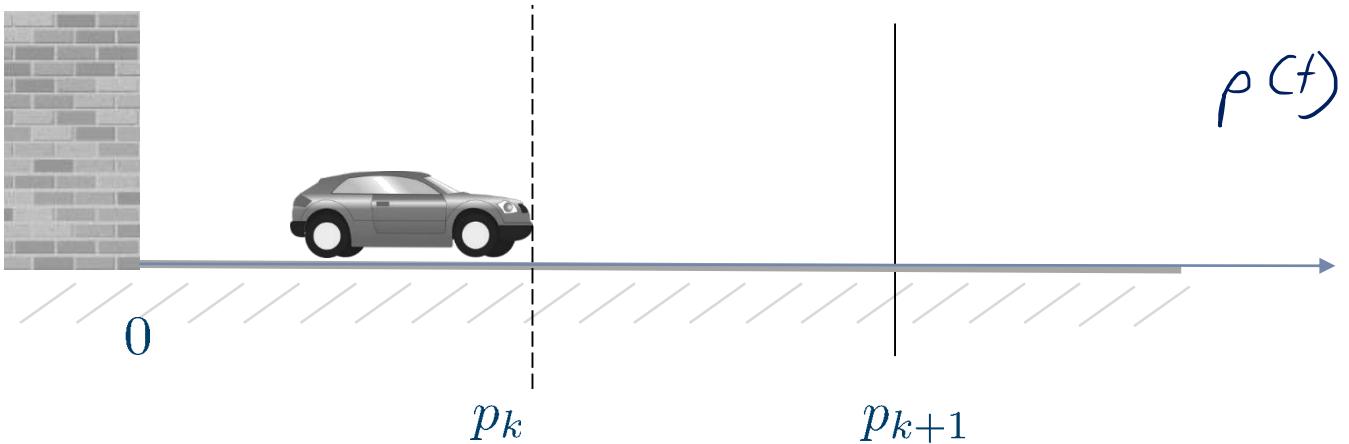
Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**



How to model the motion of the vehicle from time t_k to t_{k+1} ?

- 1D-position at discrete time k :

$$p_k = p(t_k) \in \mathbb{R}$$

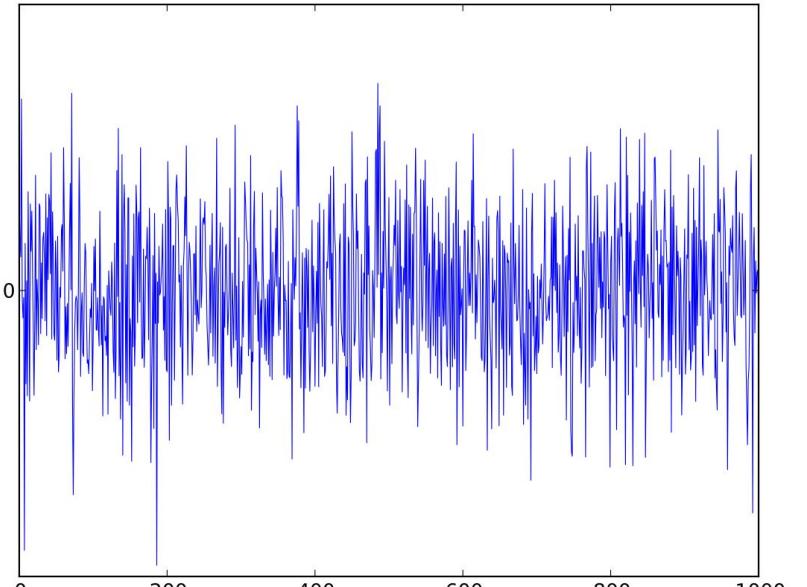
- Time period $T = t_{k+1} - t_k$

The scalar discrete time process $w_k \in \mathbb{R}$ with $k > 0$ is white with variance σ_w^2 if

- $E[w_k] = 0$ (mean of noise at every timestep is zero)
- $E[w_k w_l] = 0$ for $l \neq k$
- $E[w_k^2] = \sigma_w^2$

uncorrelated between
different time steps

for all $l, k > 0$



Source: https://en.wikipedia.org/wiki/White_noise

Discrete White Noise Velocity model

- 1D-position at discrete time k : $p_k = p(t_k) \in \mathbb{R}$
- Time period $T = t_{k+1} - t_k$
continuous time position in t_k
- Assumptions on velocity
specifies *lsl derivation*

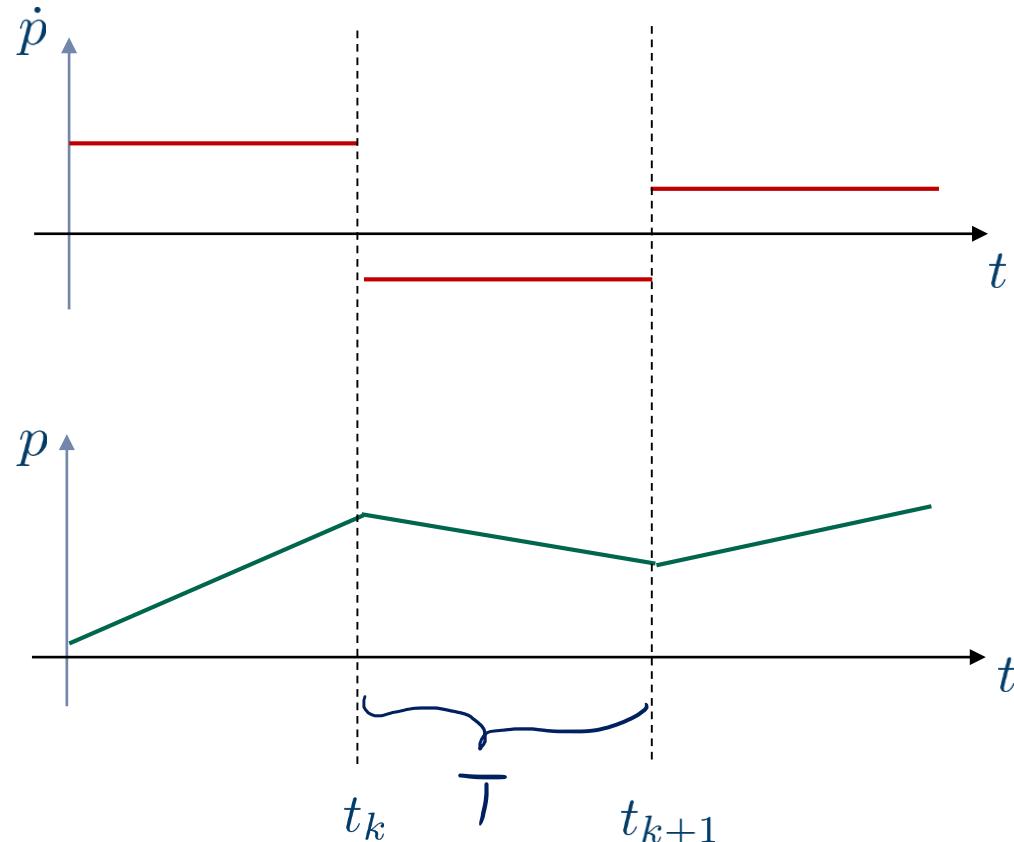
- Constant during time t_k and t_{k+1}
- Zero-mean white noise process with variance σ_w^2

$$\dot{p}_k = w_k \text{ with } [w_k] = \frac{m}{s}$$

white noise

- For $t > t_k$:
- $$p(t) = p_k + w_k \cdot t$$
- pos at t_k*

- At time t_{k+1} :
- $$p_{k+1} = p_k + \underbrace{w_k \cdot T}_{\text{noise: } T^2 \sigma_w^2}$$



White Noise Acceleration Model

(white noise on the acceleration)

- 1D-position at discrete time k : $p_k = p(t_k) \in \mathbb{R}$

- Time period $T = t_{k+1} - t_k$

- Assumptions on acceleration

- Constant during time t_k and t_{k+1}
 - Zero-mean white noise process with variance σ_w^2

- For $t > t_k$:

$$\ddot{p}_k = w_k \text{ with } [w_k] = \frac{m}{s^2}$$

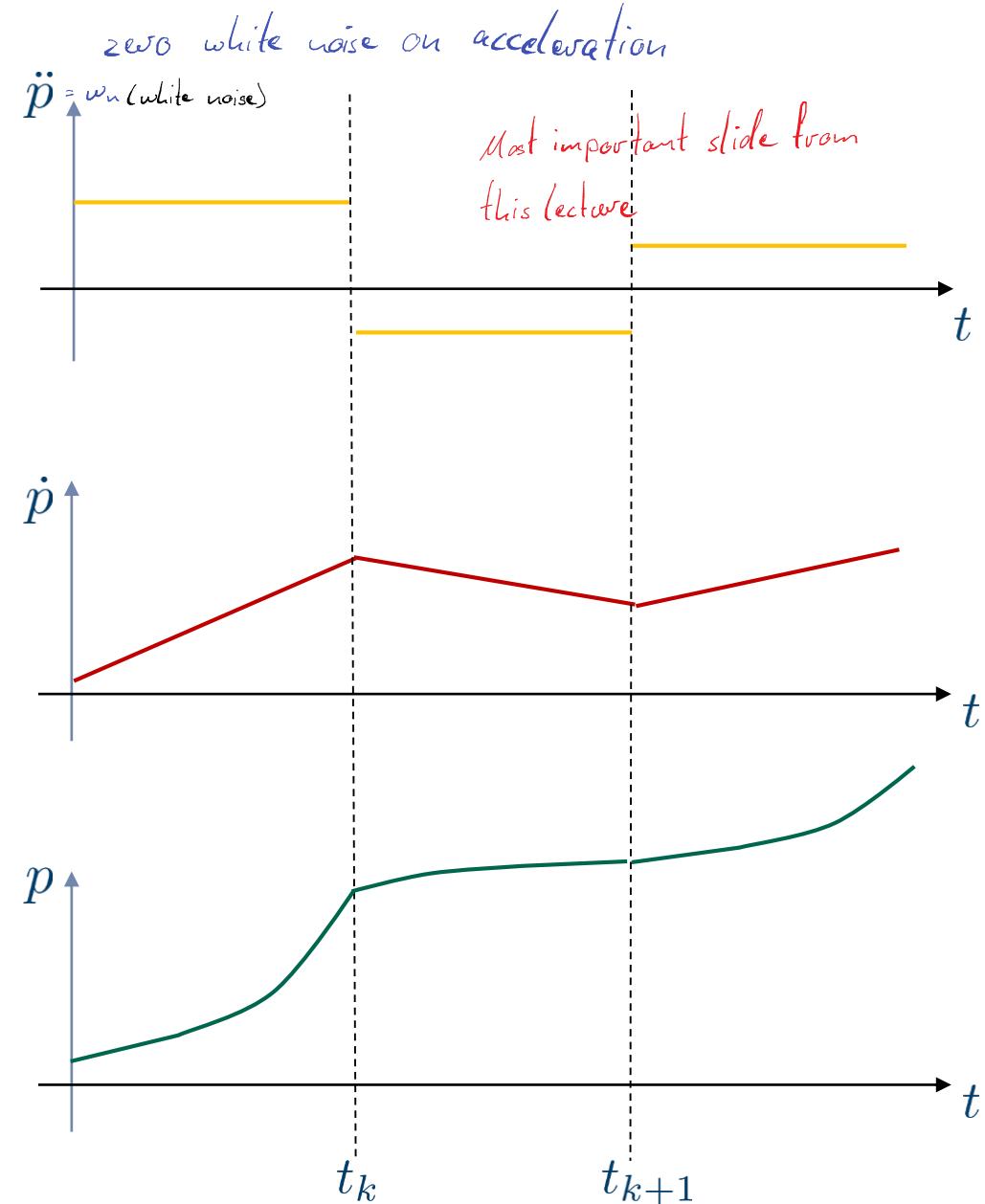
random acceleration $\dot{p}(t) = \dot{p}_k + w_k \cdot t$
 integration step to get position $p(t) = p_k + \dot{p}_k \cdot t + \frac{1}{2} \cdot w_k \cdot t^2$

- At time t_{k+1} :

$$\dot{p}_{k+1} = \dot{p}_k + w_k \cdot T$$

$$p_{k+1} = p_k + \dot{p}_k \cdot T + \frac{1}{2} \cdot w_k \cdot T^2$$

T is fixed



- State vector:

$$x_k = \begin{bmatrix} p_k \\ \dot{p}_k \end{bmatrix} \quad \begin{array}{l} \text{position} \\ \text{velocity} \end{array}$$

- Assume acceleration is white noise that is constant during a sampling period, i.e., w_k and $E[w_k^2] = \sigma_w^2$:

$$x_{k+1} = \mathbf{F}x_k + \mathbf{G}w_k$$

this system equation can be directly used
in the Kalman Filter for the prediction

with

$$\mathbf{F} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

system matrix

$$\mathbf{G} = \begin{bmatrix} \frac{1}{2}T^2 \\ T \end{bmatrix}$$

noise gain

and hence

$$\begin{aligned} \text{Cov}[\mathbf{G}w_k] &= \mathbf{G}\sigma_w^2\mathbf{G}^T &= \left[\begin{array}{c|c} \frac{1}{2}\tau^2 & \tau \\ \hline \tau & T \end{array} \right] \cdot \left[\begin{array}{cc} \frac{1}{2}\tau^2 & \tau \\ \tau & T \end{array} \right] \cdot \sigma_w^2 \\ &= \sigma_w^2 \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 \\ \frac{1}{2}T^3 & T^2 \end{bmatrix} := \mathbf{C}_{\text{cv}} \end{aligned}$$

can be explicitly calculated



- Acceleration is a Wiener process, i.e.,

$$\ddot{p}_{k+1} = \ddot{p}_k + w_k$$

(w_k is independent of T)

- Assume noise w_k is constant during sampling period with $E[w_k^2] = \sigma_w^2$

- Discrete-time state vector:

$$x_k = \begin{bmatrix} p_k \\ \dot{p}_k \\ \ddot{p}_k \end{bmatrix}$$

- System equation:

$$x_{k+1} = \begin{bmatrix} p_k + T\dot{p}_k + \frac{1}{2}(\ddot{p}_k + w_k)T^2 \\ \dot{p}_k + T(\ddot{p}_k + w_k) \\ \ddot{p}_k + w_k \end{bmatrix}$$

$$x_{k+1} = \mathbf{F}x_k + \mathbf{G}w_k$$

with

$$\mathbf{F} = \begin{bmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} \frac{1}{2}T^2 \\ T \\ 1 \end{bmatrix}$$

and hence

$$\begin{aligned} \text{Cov}[\mathbf{G}w_k] &= \mathbf{G}\sigma_w^2\mathbf{G}^T \\ &= \sigma_w^2 \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 & \frac{1}{2}T^2 \\ \frac{1}{2}T^3 & T^2 & T \\ \frac{1}{2}T^2 & T & 1 \end{bmatrix} \end{aligned}$$

• generalizing to the 2D case

- State vector:

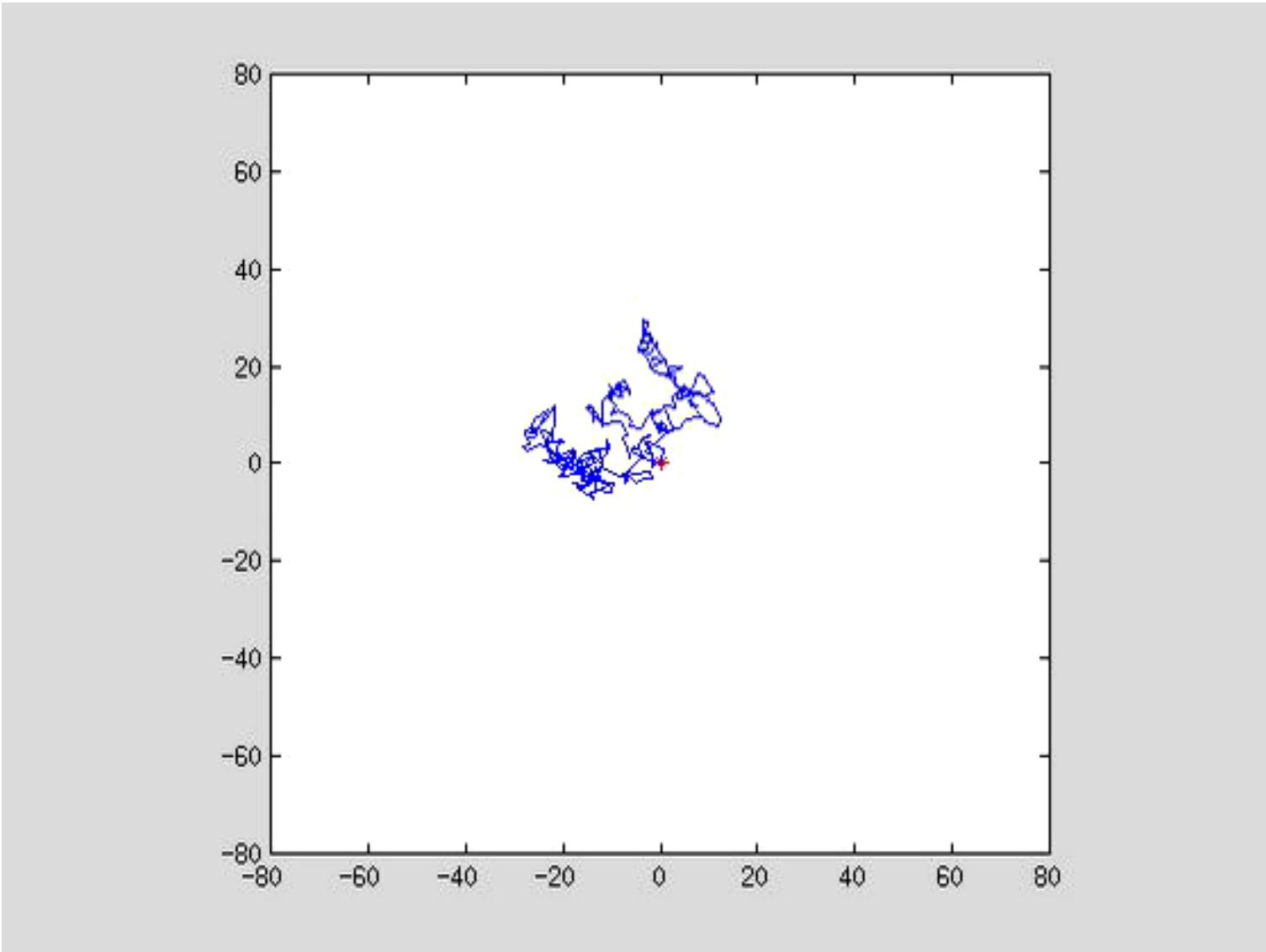
$$x_k = \begin{bmatrix} p_k \\ \dot{p}_k \\ r_k \\ \dot{r}_k \end{bmatrix}$$

- System model:

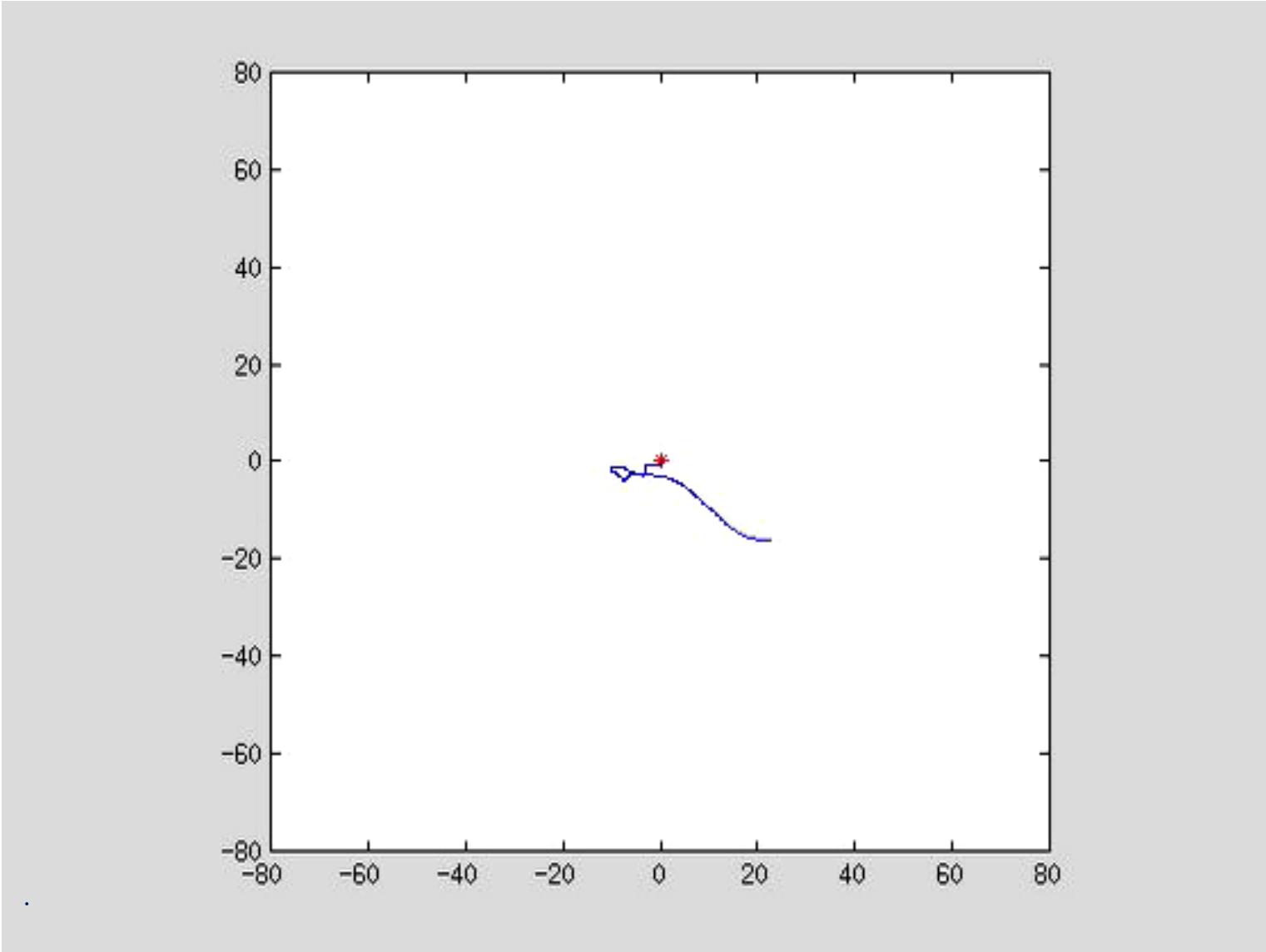
$$x_{k+1} = \begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0} & \mathbf{F} \end{bmatrix} x_k + w_k^*$$

with $\text{Cov}[w_k^*] = \begin{bmatrix} \mathbf{C}_{\text{cv}} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{\text{cv}} \end{bmatrix}$ where \mathbf{F} and \mathbf{C}_{cv} are matrices from the one-dimensional model

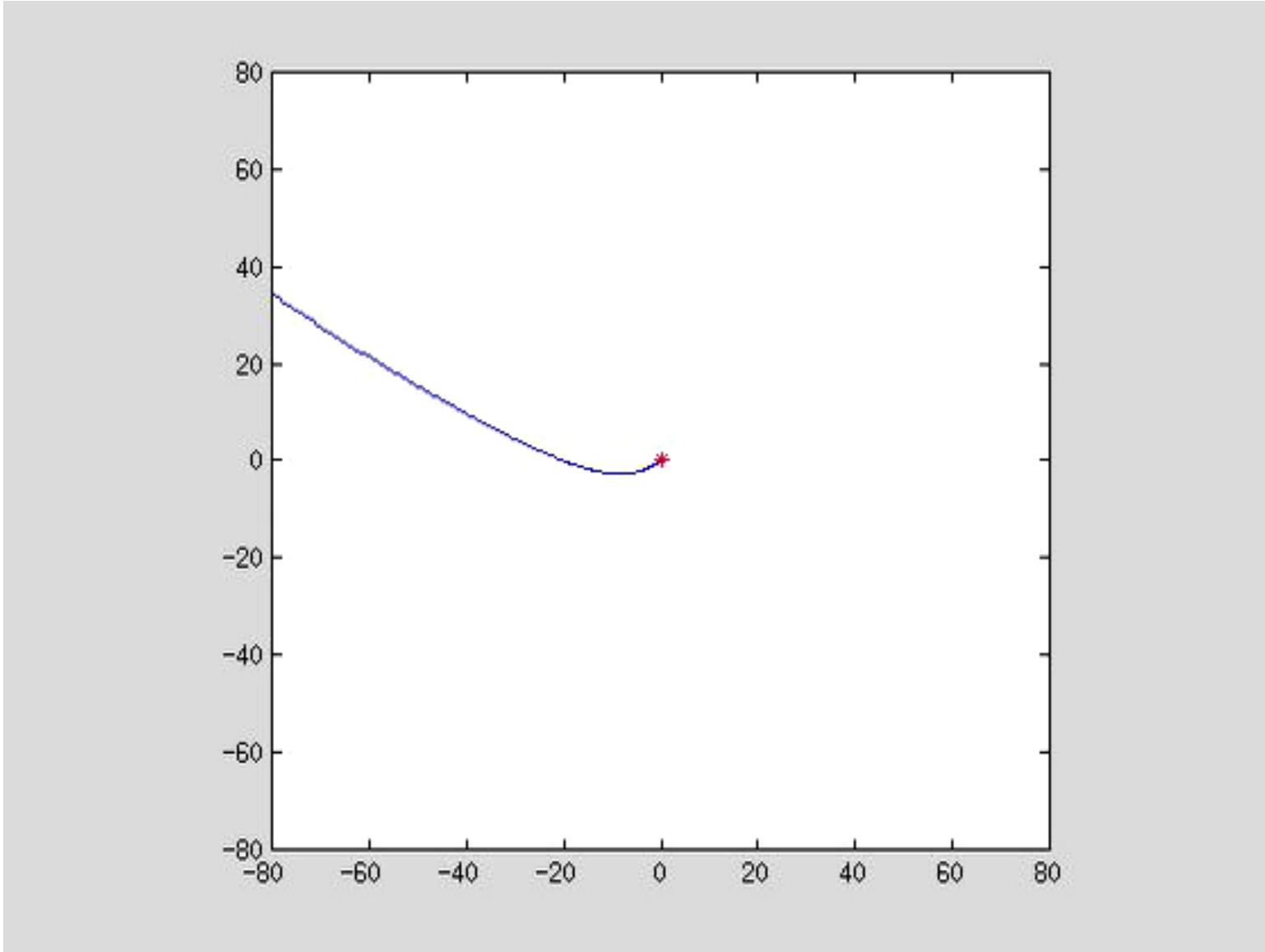
Video: White Noise Velocity Model (Wiener Process)



Video: Nearly Constant Velocity



Video: Nearly Constant Acceleration



Sensor Data Fusion

Exercise 9

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

www.fusion.informatik.uni-goettingen.de

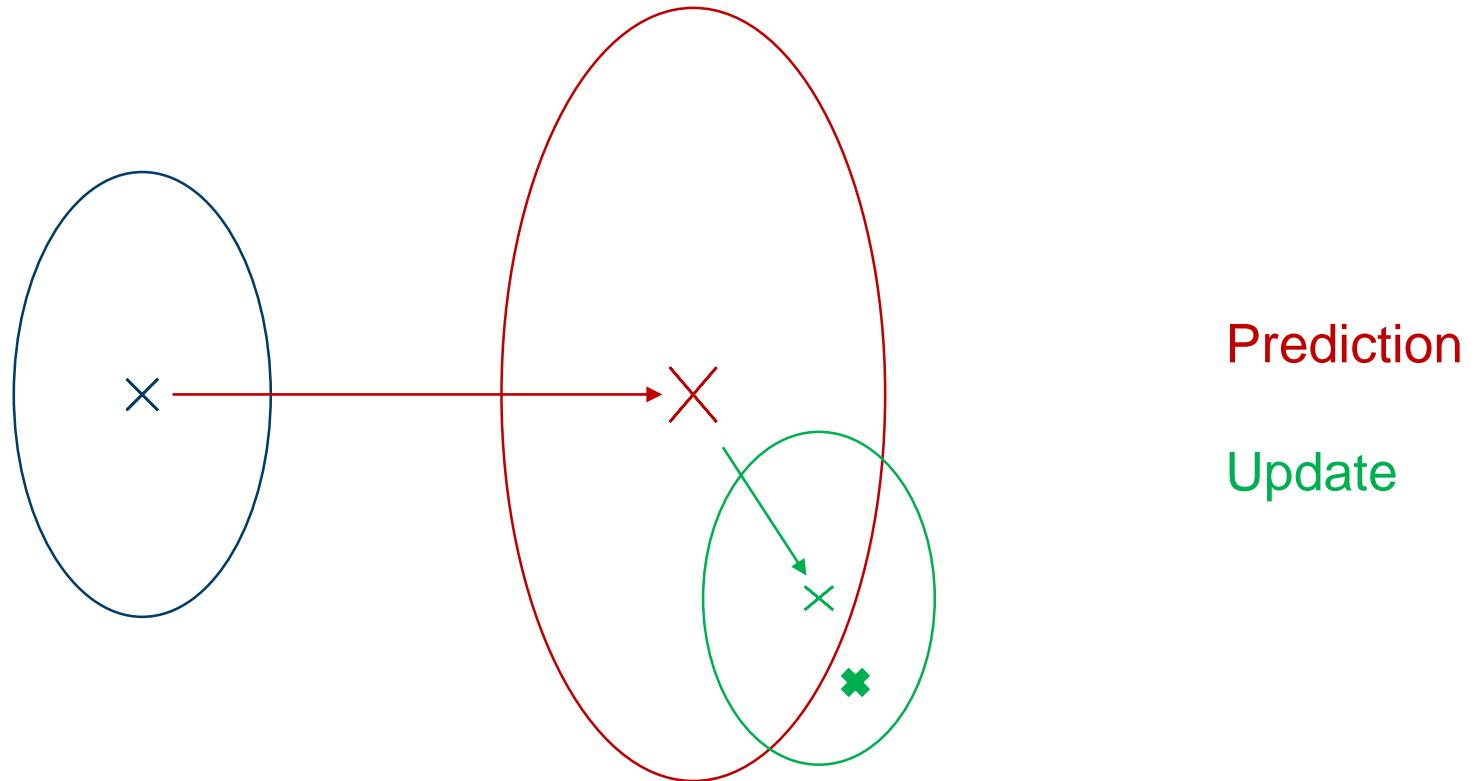


GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab

- Lecture review
- Questions
- Homework 8 solution
- Example - Steady state gain
- Problem 9 - Motion model
- Homework 9 presentation

Can you intuitively explain the Kalman filter?



Prediction

Update

What are disadvantages of the EKF?

- Might diverge for severe nonlinearities
- Need to compute Jacobians

Can you explain the white noise velocity model?

- State p_k
- Velocity as white noise $\dot{p}_k = w_k \sim \mathcal{N}(0, \sigma_w^2)$
- $p_{k+1} = p_k + T w_k$ with $T = t_{k+1} - t_k$

What is the assumption of the noise during sampling periods in the white noise acceleration model?

The noise is constant during sampling periods.

Assume a robot in 1D-space at position x moving at time k with velocity v_k forward. The prior of x at time $k = 0$ is a Gaussian with $\hat{x}_0 = 5m$ and $\sigma_{x,0}^2 = 2m^2$.

- a) Draw x_0 from the prior. The robot moves $x_{k+1} = x_k + T(v_k + e_v)$ with equidistant time steps $T = 1s$ and velocity error $e_v \sim \mathcal{N}(0\frac{m}{s}, 0.5(\frac{m}{s})^2)$. Write a function which moves the robot for one time step with constant input $v_k = 1\frac{m}{s}$.

Hint: When using `np.random.normal`, you need to pass the standard deviation as the scale parameter. Remember how standard deviation and variance (which is given here) are related, and make sure you use `np.sqrt(...)` as necessary.

- b) In each time step, a sensor measures the robot's position. Implement a measurement equation assuming independent zero-mean Gaussian noise of $e_s \sim \mathcal{N}(0m, 0.2m^2)$.

see notebook for solution

- c) Now, implement the time update formulas from the lecture to get the next predicted state and variance for a single time step.

Note for the solution:

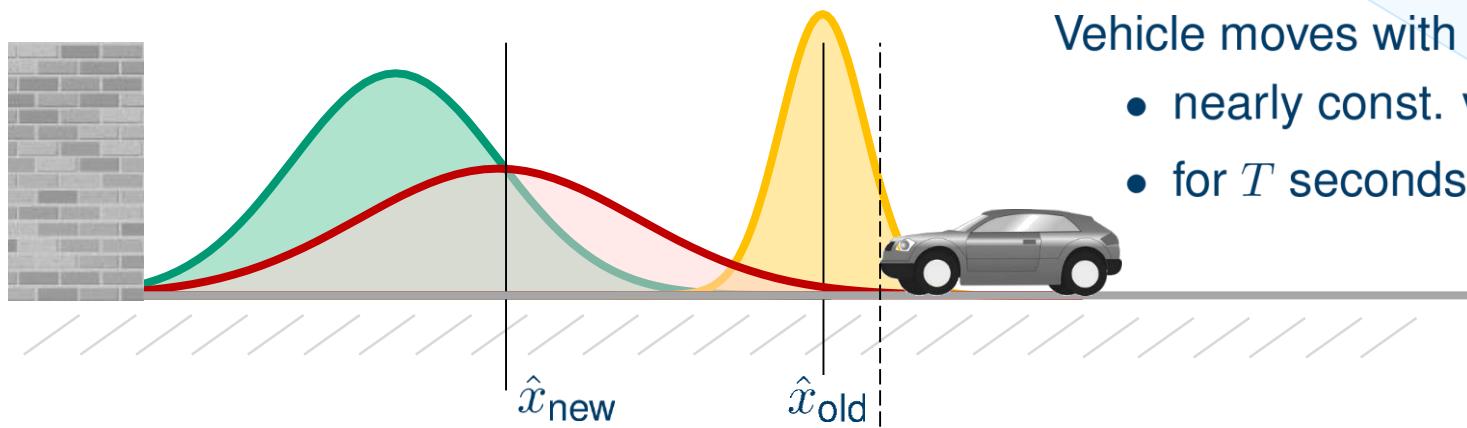
$$\hat{x}_{k,k-1} = \hat{x}_{k-1,k-1} + T v_k$$
$$\sigma_{x_{k,k-1}}^2 = \sigma_{x_{k-1,k-1}}^2 + T^2 \sigma_{e_v}^2$$

- d) Next, implement the measurement update to get the updated state and variance after a measurement was received.

Note for the solution:

$$\hat{x}_{k,k} = \hat{x}_{k,k-1} + \frac{\sigma_{x_{k,k-1}}^2}{\sigma_{x_{k,k-1}}^2 + \sigma_{e_s}^2} (y_k - \hat{x}_{k,k-1})$$
$$\sigma_{x_{k,k}}^2 = \sigma_{x_{k,k-1}}^2 - \frac{\sigma_{x_{k,k-1}}^4}{\sigma_{x_{k,k-1}}^2 + \sigma_{e_s}^2}$$

Time Update: Prediction of an Estimate



\hat{x}_{old}

σ_{old}^2

Discrete-time Motion Model:

$$x_{\text{new}} = x_{\text{old}} + T \cdot (v + e_v)$$

with velocity v and zero-mean noise e_v with variance σ_v^2

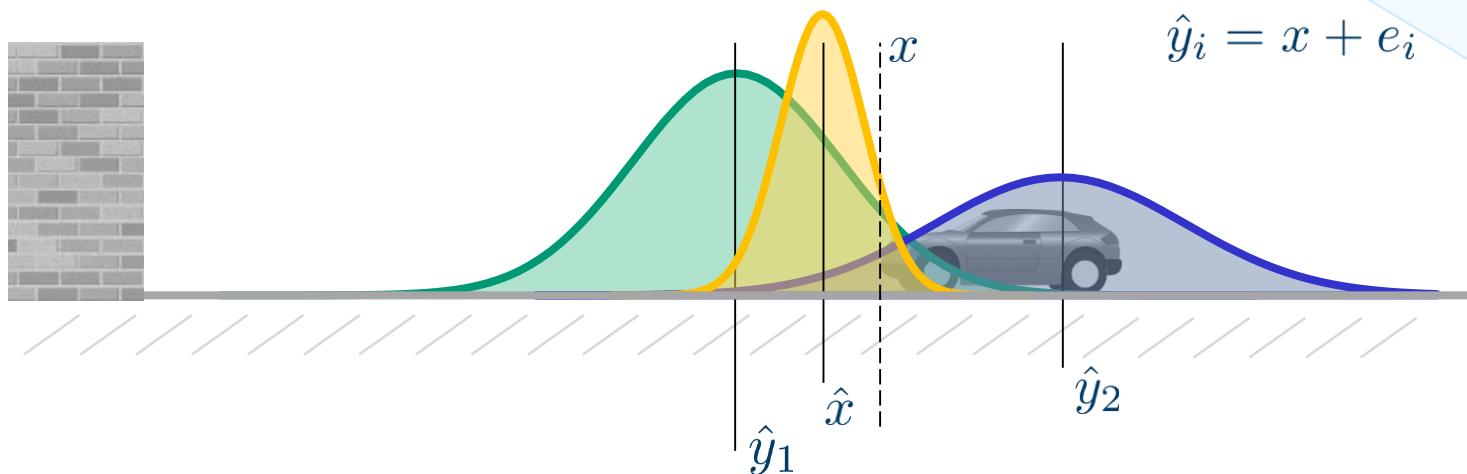
$$\hat{x}_{\text{new}} = \hat{x}_{\text{old}} + T \cdot v$$

$$\sigma_{\text{new}}^2 = \sigma_{\text{old}}^2 + T^2 \sigma_v^2$$

Variance increases

Next measurement: fusion with prediction

Update: Fusion of Two Noisy Measurements



Meas. / Estimate	Error Covariance
\hat{y}_1	$\sigma_1^2 = E[e_1^2] = E[(y_1 - x)^2]$
\hat{y}_2	$\sigma_2^2 = E[e_2^2] = E[(y_2 - x)^2]$
$\hat{x} = (1 - \alpha)\hat{y}_1 + \alpha\hat{y}_2$	$\sigma_x^2 = E[(\hat{x} - x)^2] = (1 - \alpha)\sigma_1^2$

$$\alpha = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

Linear estimator

Variance decreases

Can be applied recursively

- e) Finally, put the code of all functions together to run the simulation and create a visualization for it.

Hint: all plotting functions are already implemented, along with the necessary variable definitions. However, you still need to fill out certain small blocks of code that are responsible for generating the initial state of x , and the measurement and time update steps.

see notebook for solution

Consider the state variance time and measurement update

$$\begin{aligned}\mathbf{C}_{k+1|k} &= \mathbf{F}\mathbf{C}_{k|k}\mathbf{F}^T + \mathbf{Q} \\ \mathbf{C}_{k|k} &= \mathbf{C}_{k|k-1} - \mathbf{K}_k \mathbf{H} \mathbf{C}_{k|k-1}\end{aligned}$$

with Kalman gain $\mathbf{K}_k = \mathbf{C}_{k|k-1} \mathbf{H}^T (\mathbf{H} \mathbf{C}_{k|k-1} \mathbf{H}^T + \mathbf{R})^{-1}$. We assume that \mathbf{R} and \mathbf{Q} are constant. Combining the time and measurement update, we get

$$\mathbf{C}_{k+1|k} = \mathbf{F}(\mathbf{C}_{k|k-1} - \mathbf{K}_k \mathbf{H} \mathbf{C}_{k|k-1})\mathbf{F}^T + \mathbf{Q}$$

which is the discrete time algebraic Riccati equation. If the pair (\mathbf{F}, \mathbf{H}) is observable and $(\mathbf{F}, \mathbf{Q}^{\frac{1}{2}})$ is controllable, the covariance converges

$$\lim_{k \rightarrow \infty} \mathbf{C}_{k+1|k} = \overline{\mathbf{C}}$$

so subsequently, if enough time passes, the Kalman gain will be constant

$$\lim_{k \rightarrow \infty} \mathbf{K}_k = \overline{\mathbf{K}} = \overline{\mathbf{C}} \mathbf{H}^T (\mathbf{H} \overline{\mathbf{C}} \mathbf{H}^T + \mathbf{R})^{-1}$$

In the following, we will only describe the observability aspect. The n -dimensional state x is observable, if the observability matrix \mathbf{O}_n has full rank n . \mathbf{O}_n is defined in the following. Considering expected measurements over multiple time steps

$$\mathbf{y}_0 = \mathbf{H}\mathbf{x}_0$$

$$\mathbf{y}_1 = \mathbf{H}\mathbf{x}_1 = \mathbf{H}\mathbf{F}\mathbf{x}_0$$

$$\mathbf{y}_2 = \mathbf{H}\mathbf{F}^2\mathbf{x}_0$$

we get

$$\underbrace{\begin{bmatrix} \mathbf{H} \\ \mathbf{H}\mathbf{F} \\ \vdots \\ \mathbf{H}\mathbf{F}^{n-1} \end{bmatrix}}_{\mathbf{O}_n} \mathbf{x}_0 = \begin{bmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_{n-1} \end{bmatrix}$$

Finding the steady state gain $\bar{\mathbf{K}}$ saves computational power. The initial covariance can be set accordingly and the gain does not need to be recalculated each measurement update.

As an example, consider again a robot in 1D. The state consists of the robot position and velocity $\mathbf{x}_k = [x_k \quad \dot{x}_k]^T$. The robots movement is noise corrupted with w_k and a sensor generates a measurement of the robots position each time step corrupted with noise v_k . The time difference between measurements is a constant t . So we have

$$\mathbf{x}_k = \underbrace{\begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}} \mathbf{x}_{k-1} + w_k$$
$$y = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{\mathbf{H}} \mathbf{x} + v_k$$

For our 2D state, we have the observability matrix

$$\mathbf{O}_2 = \begin{bmatrix} \mathbf{H} \\ \mathbf{HF} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & t \end{bmatrix}$$

\mathbf{O}_2 has full rank if

$$\begin{aligned}\det \mathbf{O}_2 &\neq 0 \\ 1 \cdot t - 1 \cdot 0 &\neq 0 \\ t &\neq 0\end{aligned}$$

If time passes, the speed can be observed as well from the position measurement.

We are interested in implementing a (linear) Kalman filter for tracking of the position and velocity of a ship. The 4D state is $[x \ y \ \dot{x} \ \dot{y}]^T$. In order to implement the prediction step of the filter, we need to define our process noise \mathbf{Q} . For this, we need to consider two things: First, we model inaccuracies in our motion model using a zero-mean white noise acceleration model. This is described by σ_a , which is the same for acceleration in both x and y direction. Second, we additionally want to take into account that waves can move the ship. We assume that wave movement only affects the position of the ship, but not its velocity. This is described by σ_w , again for both x and y direction. As usual, we assume that the time passed is described by T .

How can \mathbf{Q} be calculated, taking both models into account?

Hints:

- Remember that for a single multivariate process noise factor, we used a matrix \mathbf{G} in the following fashion: $\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{G}\nu$. Hence, the Kalman filter update would look like this:
$$\mathbf{P}_{k+1} = \mathbf{F}\mathbf{P}_k\mathbf{F}^T + \underbrace{\mathbf{G}\Sigma_\nu\mathbf{G}^T}_\mathbf{Q}$$
- Consider what Σ_ν would need to include in our case. You can also consider the dimensions that \mathbf{G} and ν (and Σ_ν) should have. Remember: For the given setting, $\mathbf{Q} \in \mathbb{R}^{4 \times 4}$!

Solution: We define $\Sigma_\nu = \begin{bmatrix} \Sigma_a & 0 \\ 0 & \Sigma_w \end{bmatrix}$, a block matrix built from $\Sigma_a = \mathbf{I}_{2 \times 2} \cdot \sigma_a^2$ and $\Sigma_w = \mathbf{I}_{2 \times 2} \cdot \sigma_w^2$.

Note that $\Sigma_\nu \in \mathbb{R}^{4 \times 4}$. To define \mathbf{G} , we use a combined representation: $\mathbf{G} = [\mathbf{G}_a \quad \mathbf{G}_w]$.

For \mathbf{G}_a , we have

$$\mathbf{G}_a = \begin{bmatrix} \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{2}T^2 \\ T & 0 \\ 0 & T \end{bmatrix},$$

which is the two-dimensional version of the corresponding matrix used in the zero-mean white noise acceleration model.

For \mathbf{G}_w , we have

$$\mathbf{G}_w = \begin{bmatrix} T & 0 \\ 0 & T \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

as the wave movement linearly affects the two positional entries of the state, but does not affect the velocity at all. This results in $\mathbf{Q} = \mathbf{G}\Sigma_\nu\mathbf{G}^T$ with \mathbf{G} and Σ_ν defined above.

Assume a robot in 2D-space at position $[x_1 \quad x_2]^T$ moving with velocity \dot{x}_1 in x_1 direction and \dot{x}_2 in x_2 direction. Its state is defined as $\mathbf{x} = [x_1 \quad x_2 \quad \dot{x}_1 \quad \dot{x}_2]^T$.

- a) Formulate a motion model for the robot, assuming the acceleration on both dimensions as independent zero-mean Gaussian noise. Write a function implementing the motion model.
- b) In each time step, a sensor measures the robot's position. Formulate a measurement equation assuming independent zero-mean Gaussian noise and implement it as well.
- c) Use the functions from a) and b) to implement a simulation using initial state $\hat{\mathbf{x}}_{\text{init}} = [0\text{m} \quad 0\text{m} \quad 2\text{m s}^{-1} \quad 2\text{m s}^{-1}]$ (for each run, draw the true state from the prior), 10 time steps of length 1s, and covariances for the initial state \mathbf{C}_{init} , the transition noise Σ_a and the measurement noise \mathbf{R} as

$$\mathbf{C}_{\text{init}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \Sigma_a = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}.$$

- d) Now based on a) and b), implement a predict and an update function for a Kalman filter. Use the Kalman filter to track the robot simulated with your function from c).
- e) Write a function which calculates the root mean square error (RMSE) of $n = 100$ simulation runs with the error as the Euclidean norm at the last time step.
- f) Finally, assume a worse sensor with noise covariance

$$\mathbf{R}_2 = \begin{bmatrix} 2.0 & 0 \\ 0 & 0.2 \end{bmatrix} .$$

Calculate the RMSE as in e). To deal with the noise, add a second sensor. Assume the measurements to be independent of each other and update the simulation and Kalman filter accordingly and observe the difference in RMSE. Now, instead of using the same type of sensor, use the sensor with \mathbf{R}_2 along a second sensor with noise covariance

$$\mathbf{R}_3 = \begin{bmatrix} 0.2 & 0 \\ 0 & 2.0 \end{bmatrix} .$$

Bonus:

B.a) This time, instead of measuring the position directly, we measure the distance to a landmark P_i . Formulate the measurement equation and write a function implementing it. We do not provide a measurement matrix H this time, but instead the landmark P .

For now, we will assume only a single landmark is used. In that case, the measurement noise will be a 1×1 matrix.

B.b) To use the measurement in the Kalman filter, we utilize the EKF formulas. Calculate the Jacobian of the measurement function from a) and implement the EKF measurement update.

Again, we assume that for now only one landmark exists, and \mathbf{R} will be a 1×1 matrix. Run the simulation again assume $\mathbf{R} = [0.1]$ and $\mathbf{P}_1 = [0, -5]^T$.

B.c) To improve your estimate, use a second landmark $P_2 = [-5 \ 0]^T$. Formulate a stacked measurement equation and calculate the corresponding Jacobian. Run the simulation again.

Sensor Data Fusion

Uncertainty Modeling (1 & 2)

Prof. Dr.-Ing. Marcus Baum

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Oral exams (approx. 20 minutes)
- Please register at FlexNow until
July 13
- and make an appointment via Stud.IP until
July 17
- All available course material at Stud.IP is relevant for the exam
- **No auxiliary materials** allowed during the exam, i.e., no slides, no notes etc.
- Place: Room 3.115, Institute of Computer Science

1. Introduction
2. Measurement Equation and Linear least squares
3. Nonlinear least squares: Analytic approaches
4. Nonlinear least squares: Optimization algorithms
5. Stochastic least squares
6. Fundamental fusion formulas
7. Bayesian approach: Fundamentals
8. Bayesian approach: Fusion, Linear and Gaussian case
9. Kalman filter: Introduction
10. (Extended) Kalman filter
11. Dynamic models
12. Uncertainty modeling (1)
13. Uncertainty modeling (2)
14. Summary and Discussion

Reference:

Branko Ristic, Christopher Gilliam, Marion Byrne, Alessio Benavoli,
A tutorial on uncertainty modeling for machine reasoning,
Information Fusion,
Volume 55, 2020, Pages 30-44, ISSN 1566-2535,
<https://doi.org/10.1016/j.inffus.2019.08.001>.

Available online:

<https://www.sciencedirect.com/science/article/pii/S1566253519301976>



Information Fusion
Volume 55, March 2020, Pages 30-44



A tutorial on uncertainty modeling for machine reasoning

Branko Ristic , Christopher Gilliam , Marion Byrne , Alessio Benavoli 

Show more 

+ Add to Mendeley  Share  Cite

<https://doi.org/10.1016/j.inffus.2019.08.001> 

Under a Creative Commons license

[Get rights and content](#)

[open access](#)

Highlights

- A review of the prevalent methods for reasoning and decision making under uncertainty.
- Covers Bayesian probabilistic, random sets, possibilistic, belief function and imprecise probability theory.
- Includes numerous examples with MATLAB solutions.

“How high is the Eiffel tower?”

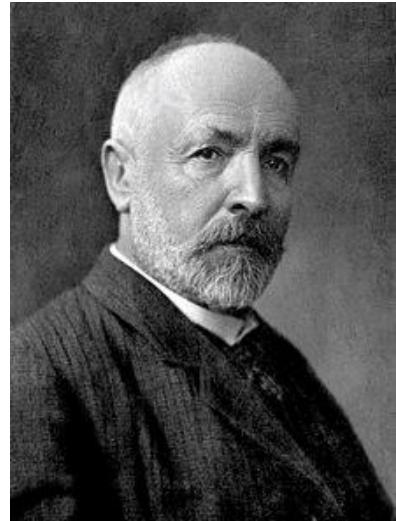
- **Imprecise information**
 - Expressed as a set of possible values (crisp set)
 - Examples:
 - (i) more than 270 m
 - (ii) in the interval from 300 m to 350 m
- **Information affected by variability due to random effects (aleatory uncertainty)**
 - Typically expressed by a probability distribution
 - Example:
Probability mass function (PMF) whose support is {321 m, 322 m , 323m} with the probabilities 0.4, 0.5, and 0.1
- **Erroneous information (epistemic uncertainty)**
 - Model-mismatch situation
 - Systematic errors
- **Vague information**
 - Typically expressed as a fuzzy set
 - Example (vague and imprecise):
“The height of the Eiffel tower is at least approximately 325 m”

Brief introduction to

- Fuzzy theory
(vague information)
- Random Finite Set (RFS) theory
(imprecise likelihoods within Bayes)
- Dempster-Shafer Theory
(imprecise information)

“A set is a gathering together into a whole of definite, distinct objects of our perception or our thought—which are called elements of the set.”

Georg Cantor



https://en.wikipedia.org/wiki/Georg_Cantor

Examples:

- Set of natural numbers smaller than 3
- Unit disc in two-dimensional space

- Enumeration of elements

$$S_1 = \{1, 2, 3\}$$

- Definition by property

$$S_2 = \{x \in \mathbb{R}^2 \mid \|x\| \leq 1\}$$

- Characteristic function

Indicator
membership

$$\chi_A(x) : X \rightarrow \{0, 1\}$$

$$\chi_A(x) = \begin{cases} 1 & x \text{ is a member of } A \\ 0 & \text{otherwise} \end{cases}$$

- Sensor 1:
“Eiffel tower is larger than 200m”

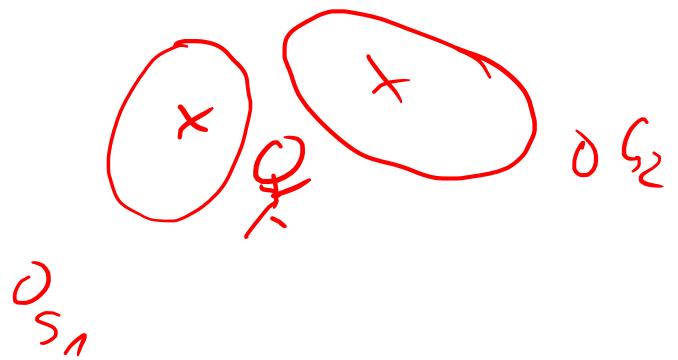
$$x \in (200m, \infty)$$

- Sensor 2:
“Eiffel tower is less than 400m”

$$x \in (-\infty, 400m)$$

- Fusion:

$$x \in (200m, 400m)$$

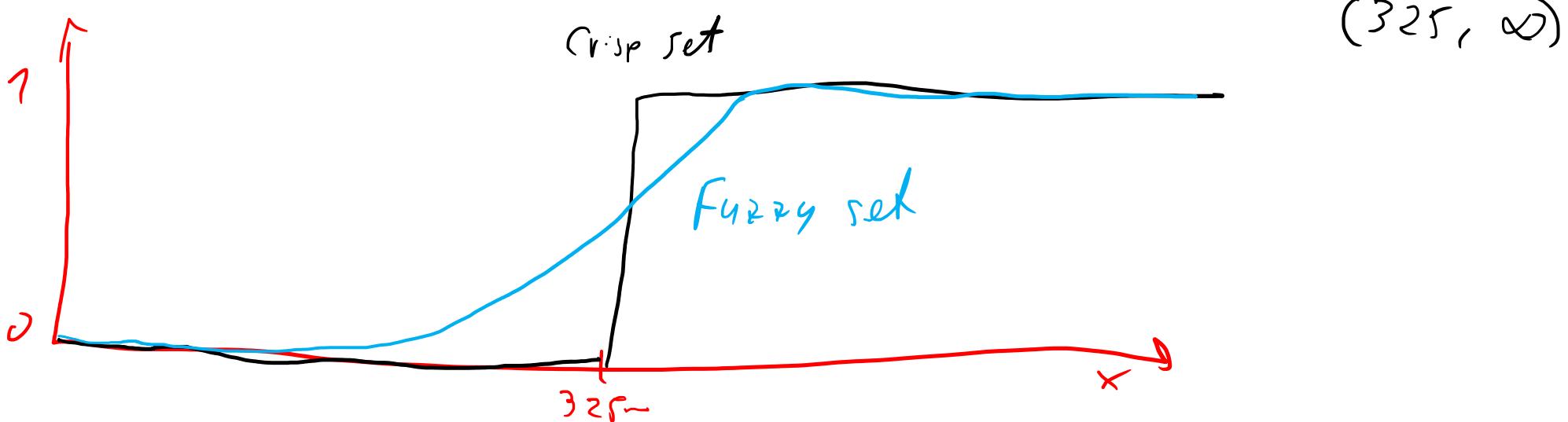


“The height of the Eiffel tower is larger than (approximately) 325 m”



- (Crisp) Sets are not suitable to represent vague information
- Fuzzy sets are sets with elements that have a degree of membership, i.e., membership function can take any value in $[0,1]$
- Introduced by Lotfi A. Zadeh (1965)

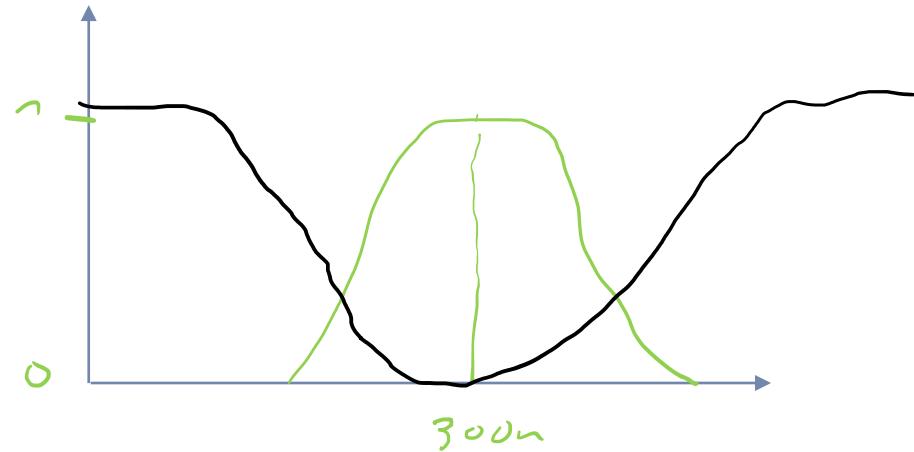
https://en.wikipedia.org/wiki/Lotfi_A._Zadeh



Operations on Fuzzy Sets

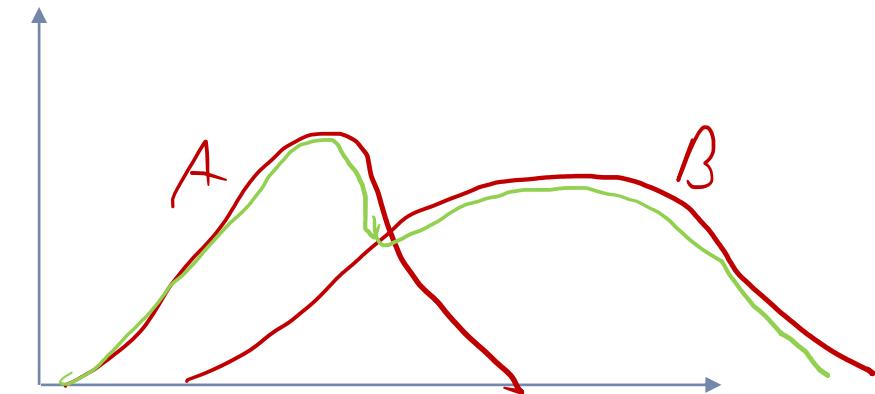
- Complement

$$1 - \mu(x)$$



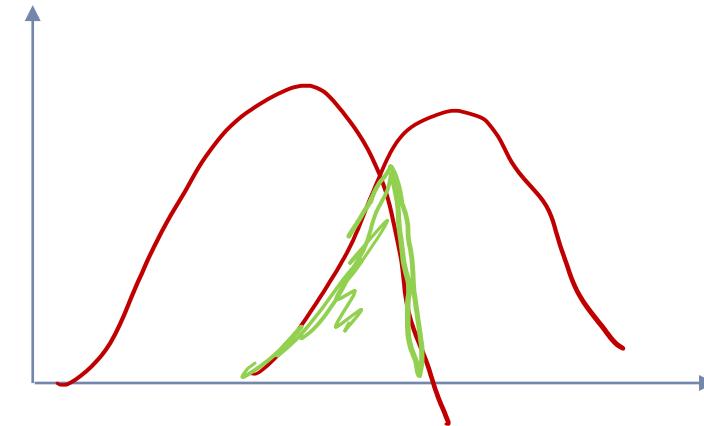
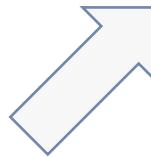
- Union

$$\max\{\mu_A(x), \mu_B(x)\}$$



- Intersection

$$m = \{M_A(x), M_B(x)\}$$



Can be used for fusion

- Classes Random variable $X \in \mathcal{X}$
 $\mathcal{X} = \{x_1, \dots, x_N\}$ with $N > 1$
- Features Z either discrete or continuous
- Prior Probability mass function $p(x)$
- Likelihood function $g(z|x), z \in \mathcal{Z}$
- Posterior $p(x|z) = \frac{g(z|x) \cdot p(x)}{\sum_{x' \in \mathcal{X}} g(z|x') \cdot p(x')}$
- Axioms Event $A \subset \mathcal{X}$
 - I. $P(A) \geq 0$
 - II. $P(\emptyset) = 1$
 - III. $P(A_1 \cup A_2) = P(A_1) + P(A_2)$

- Sequence of feature measurements

$$z^{1:k} = z^1, z^2, z^3, \dots, z^k$$

- Recursive calculation of the posterior

$$p(x|z^{1:k}) = \frac{g(z^k|x) \cdot p(x|z^{1:k-1})}{\sum_{x' \in X} g(z^k|x') \cdot p(x'|z^{1:k-1})}$$

with $p(x|z^{1:0}) = p(x)$

- Classes

$$N=3$$

$$X = \{x_1, x_2, x_3\}$$

- Features

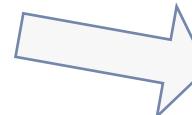
$$\mathcal{Z} = \{z_1, z_2, z_3\}$$

- Prior

$$P(x_1) = P(x_2) = \frac{2}{5} \quad P(x_3) = \frac{1}{5}$$

- Recursive classification

- Measurements generated according to confusion matrix
- True class
- 2000 Monte Carlo runs

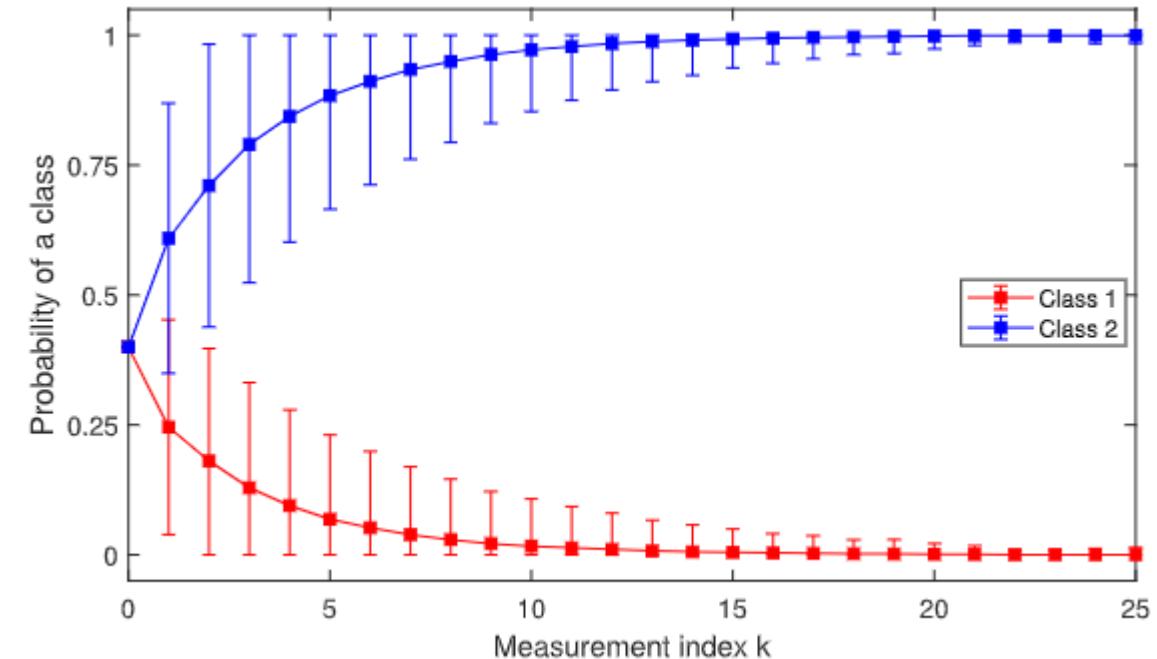


Confusion matrix

$g(z_i x_j)$	x_1	x_2	x_3
z_1	5/7	1/7	1/7
z_2	1/7	5/7	1/7
z_3	1/7	1/7	5/7

Posterior

	z_1	z_2	z_3
x_1	10/13	2/13	2/13
x_2	2/13	10/13	2/13
x_3	1/13	1/13	5/13



- Classes $X = \{x_1, x_2, x_3\}$

- Features $Z = \{z_1, z_2, z_3\}$

- Prior $p(x) = \frac{1}{3}$

- No explicit confusion matrix available, instead imprecise likelihood

(a) Class x_1 can cause z_1 only

(b) Class x_2 can cause z_1 or z_2

(c) Class x_3 can cause z_1, z_2 or z_3

Radon sets

$$\sum_{x_1} = \{z_1\}$$

$$\sum_{x_2} = \{z_1, z_2\}$$

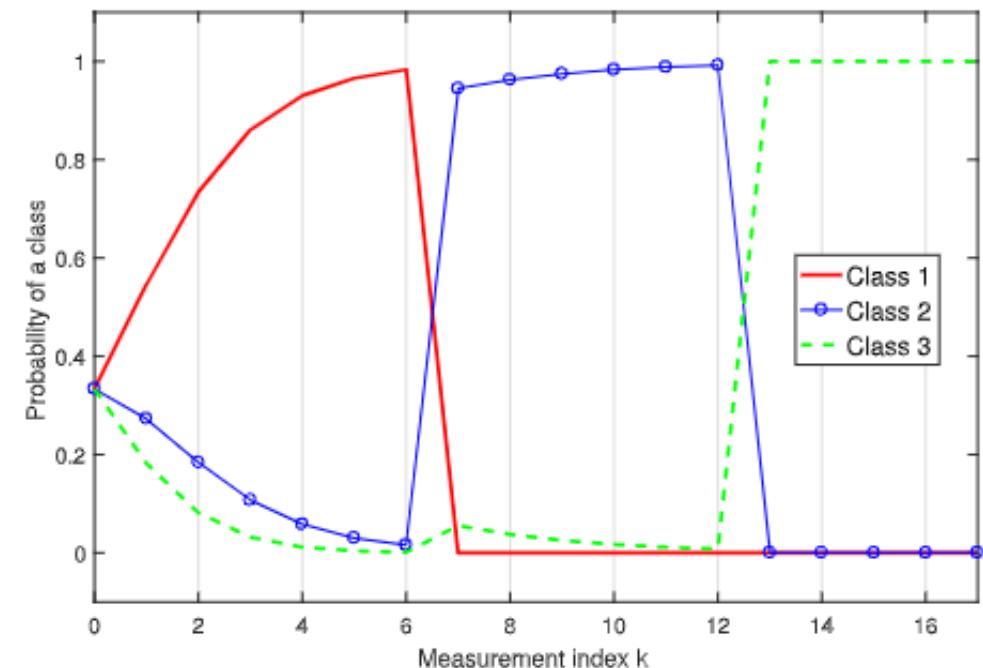
$$\sum_{x_3} = \{z_1, z_2, z_3\}$$

- Principle of maximum entropy
- → Uniform distribution on sets

Confusion matrix

$g(z_i x_j)$	x_1	x_2	x_3
z_1	1	1/2	1/3
z_2	0	1/2	1/3
z_3	0	0	1/3

- Recursive classification
 - Measurements
 - z_1 except $z^7 = z_2$ and $z^{13} = z_3$
 - 2000 Monte Carlo runs



- In a standard Bayesian setting imprecise and random information is modelled identically
- Example: A lidar sensor detects an object
 - „The object is a vehicle or a pedestrian with equal chances“ $\leftarrow P(V) = 0.5$
 $P(P) = 0.5$
 - „I have no idea about the type of the object“ $\leftarrow \{V, P\} \rightarrow$
- A framework for combining set-valued (imprecise) and random information is desired

- Random set:

A random variable whose outcome is a set

$$\Sigma \in \mathcal{Z}^{\mathcal{Z}} = \{\emptyset, \{z_1\}, \{z_1, z_2\}, \dots\}$$

$$P(\Sigma)$$

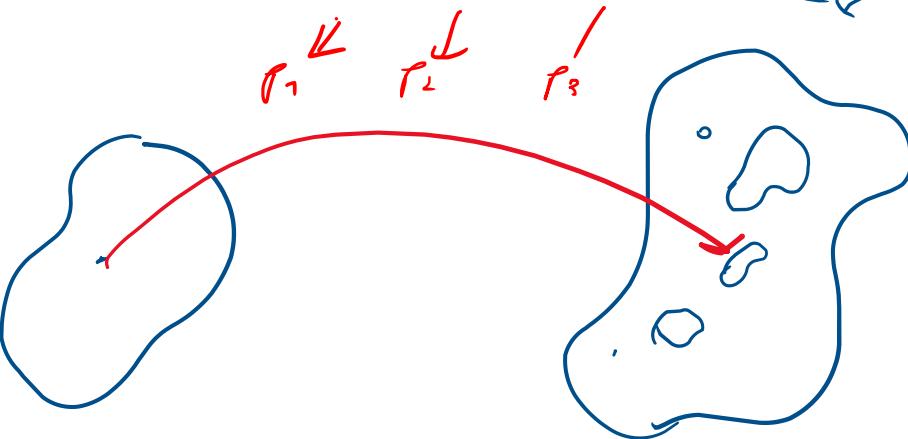
$$P(\{z_1, z_2\}) = 0.7$$

- Generalized likelihood function

$$\tilde{g}(z|x) = P(z \in \Sigma_x) \Rightarrow \text{Bayes}$$

- Still in a Bayesian framework

$$\mathcal{Z} = \{z_1, z_2, z_3\} \quad \mathcal{Z}^{\mathcal{Z}} = \{\emptyset, \{z_1\}, \{z_2\}, \{z_3\}, \{z_1, z_2\}, \{z_1, z_3\}, \{z_2, z_3\}, \{z_1, z_2, z_3\}\}$$



Scenario 3: Random Set Approach

- Sets in the measurement space

$$\Sigma_{x_1} = \{z_1\}$$

$$\Sigma_{x_2} = \{z_1, z_2\}$$

$$\Sigma_{x_3} = \{z_1, z_2, z_3\}$$

\Rightarrow Likelihood of sets

- Probabilities

$$P(z_1 \in \Sigma_{x_1}) = 1 \quad P(z_1 \in \Sigma_{x_2}) = 1 \quad P(z_1 \in \Sigma_{x_3}) = 1$$

$$P(z_2 \in \Sigma_{x_1}) = 0 \quad P(z_2 \in \Sigma_{x_2}) = 1 \quad P(z_2 \in \Sigma_{x_3}) = 1$$

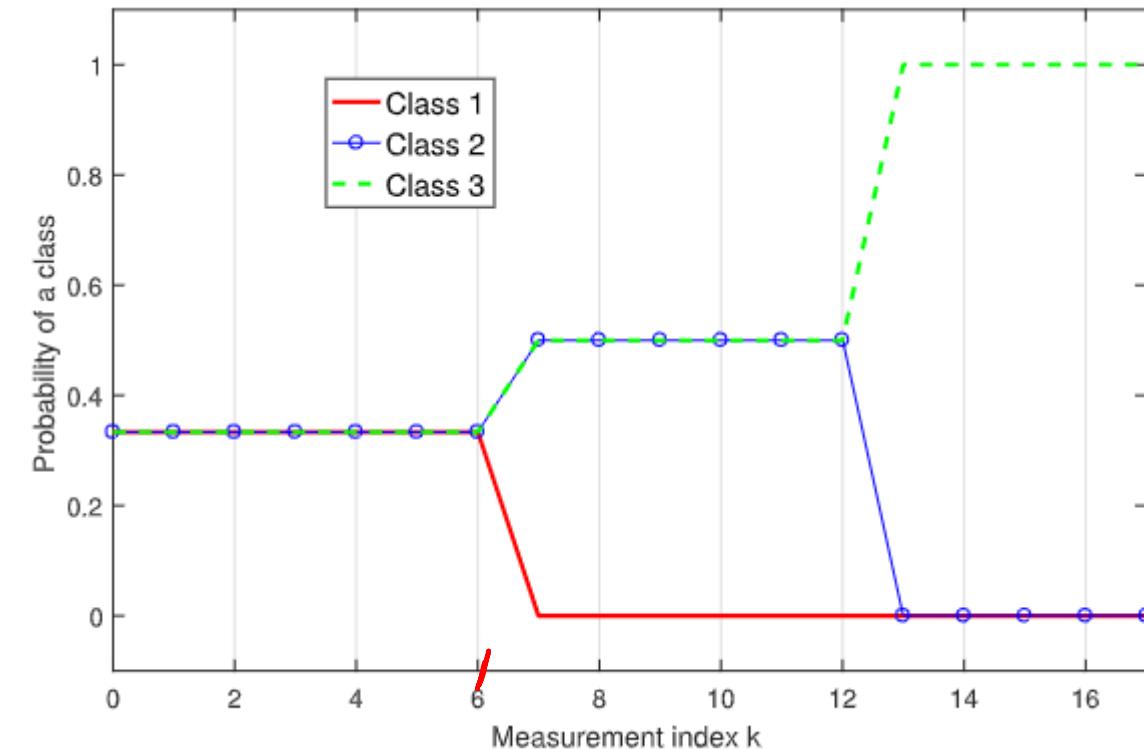
$$P(z_3 \in \Sigma_{x_1}) = 0 \quad P(z_3 \in \Sigma_{x_2}) = 0 \quad P(z_3 \in \Sigma_{x_3}) = 1$$

- Recursive classification
(same setting as Bayesian approach)

- Identical results for scenarios 1 and 2

Confusion matrix

$\hat{g}(z_i x_j)$	x_1	x_2	x_3
z_1	1	1	1
z_2	0	1	1
z_3	0	0	1



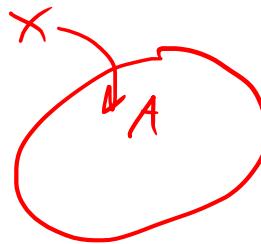
- Universe:

A finite set $\underline{\mathcal{X}} = \{x_1, \dots, x_n\}$

- Basic Belief Assignment (BBA):

Mapping $m(\cdot)$ from 2^X to $[0, 1]$ with

$$\sum_{A \subseteq 2^X} m(A) = 1$$



and

$$m(\emptyset) = 0$$

- Interpretation of $\underline{m(A)}$:

Evidence that the quantity of interest belongs (exactly) to A

Example: Environment Perception

- A lidar sensor detects an object, where its type can be

- Vehicle
- Pedestrian
- Traffic Light
- Lane

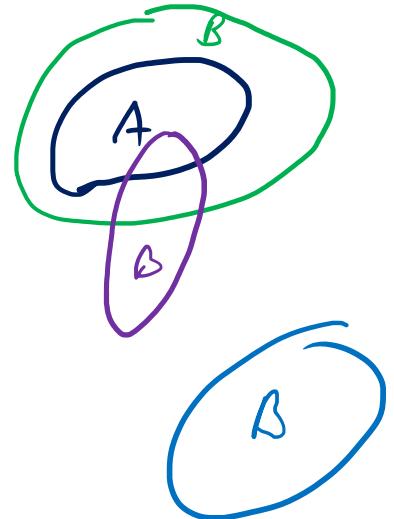
$$\mathcal{X} = \{\underline{V}, P, T, L\}$$

- The sensor says the object is either a pedestrian or a traffic light, where with 10% chance the sensor did not work (e.g., due to bad weather)

$$\begin{cases} m(\{T, P\}) = 0.9 \\ m(\mathcal{X}) = 0.1 \end{cases} \quad \Leftrightarrow \quad \begin{array}{l} \text{Believe} \\ \text{Plausibility} \end{array}$$

$$m(\{V, P, T, L\}) = 0.9$$

- $A = \{T, P\} \subset X$ $m(A) = 1$
- $B \subset X$



1. $A \subseteq B$ $x \in A \Rightarrow x \in B$ for sure
 „ B is supported by A “
2. $A \cap B \neq \emptyset$ $x \in A$ we cannot exclude that $x \in B$
 „ B is consistent with the evidence“
3. $A \cap B = \emptyset$ $x \in B$ is impossible
 (inconsistent)

- Belief function:
Probability that a proposition is supported by the evidence

$$\text{bel}(A) = \sum_{B|B \subseteq A} m(B)$$

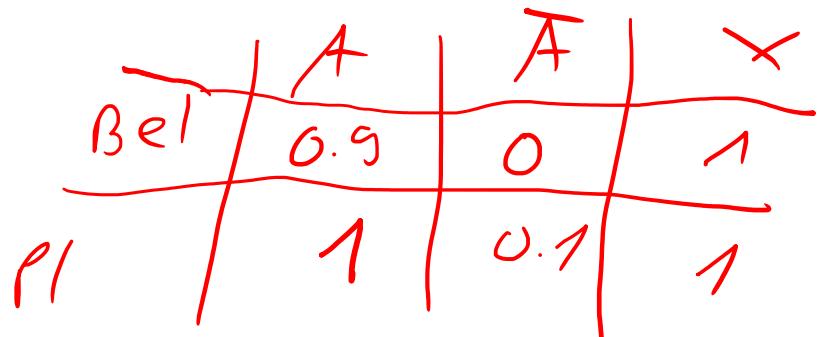
$$X = \{V, P, T, \neg\}$$

$$m(A) = 0.9 \quad A = \{T, P\}$$

$$m(\neg X) = 0.1 \quad \neg A = \{V, \neg L\}$$

- Plausibility function:
Probability that proposition is consistent with the evidence

$$\text{pl}(A) = \sum_{B|B \cap A \neq \emptyset} m(B)$$

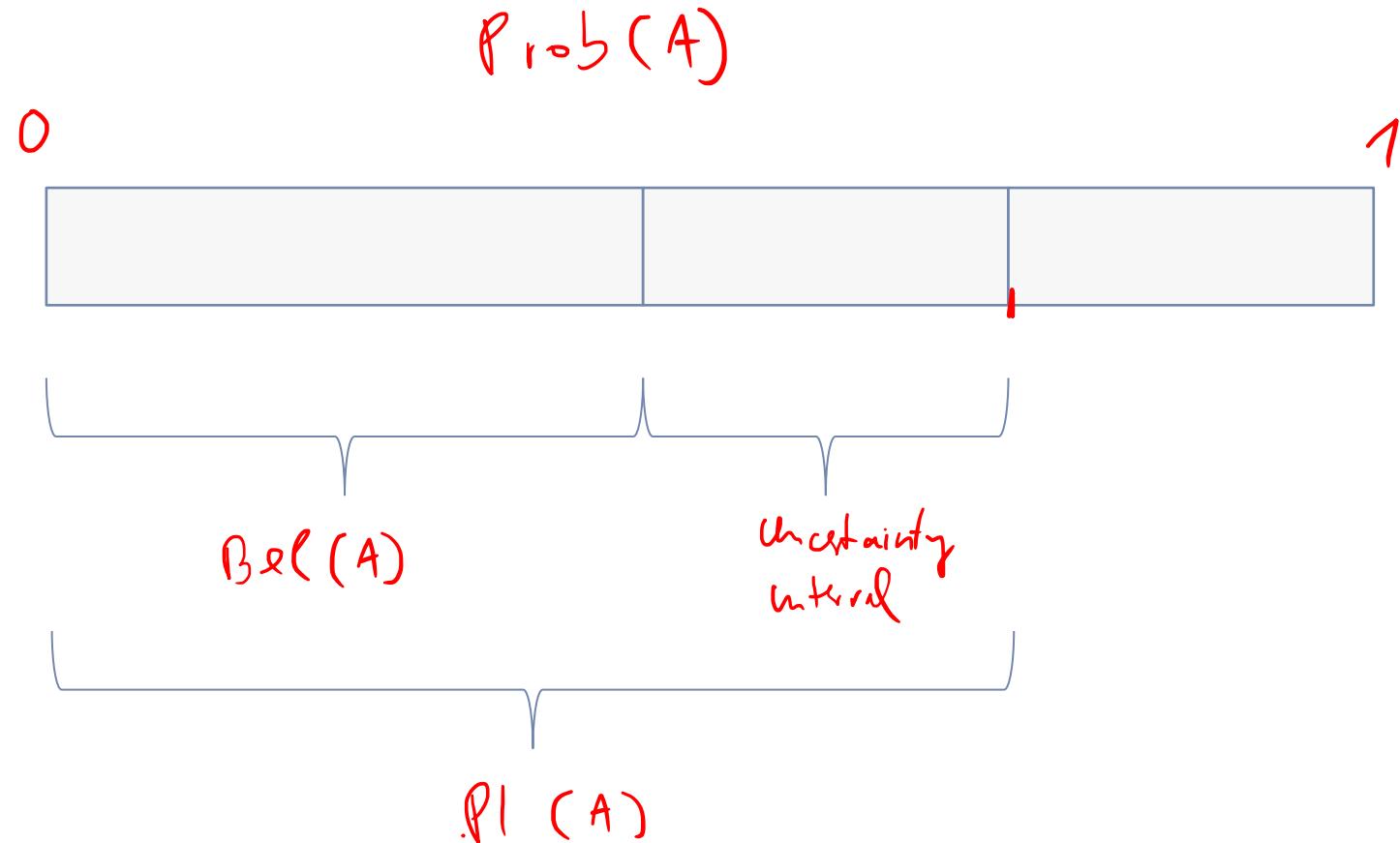


- Relationship

$$\text{pl}(A) = 1 - \text{bel}(\neg A).$$

- Specifies upper and lower probability:

$$\text{bel}(A) \leq P(A) \leq \text{pl}(A)$$



Example: Environment Perception (Continued)

- Lidar measurement results in

$$m_1(\{T, P\}) = 0.9$$

$$m_1(\mathcal{X}) = 0.1$$

- Camera measurement results in

$$m_2(\{T, V\}) = 0.8$$

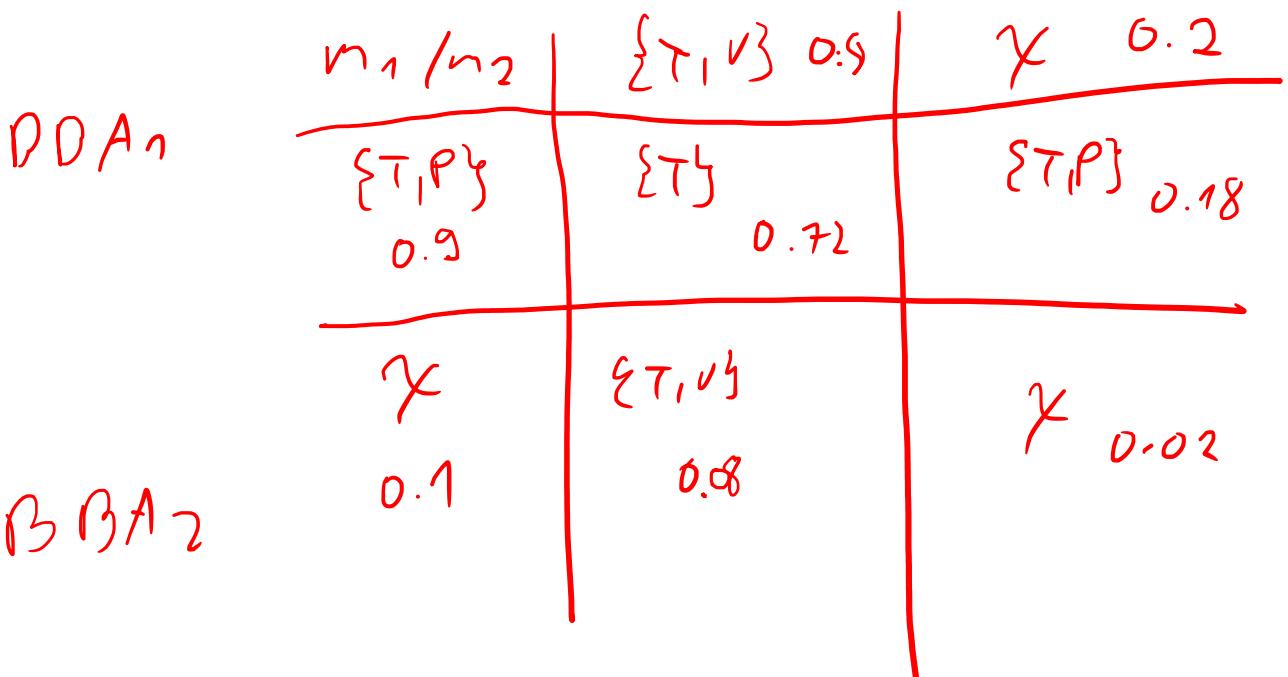
$$m_2(\mathcal{X}) = 0.2$$

Fused DBA₃
 \Rightarrow

$$m_3(\{\mathcal{T}\}) = 0.72 \quad m_3(\mathcal{X}) = 0.02$$

$$m_3(\{\mathcal{T}, V\}) = 0.08$$

$$m_3(\{\mathcal{T}, P\}) = 0.18$$

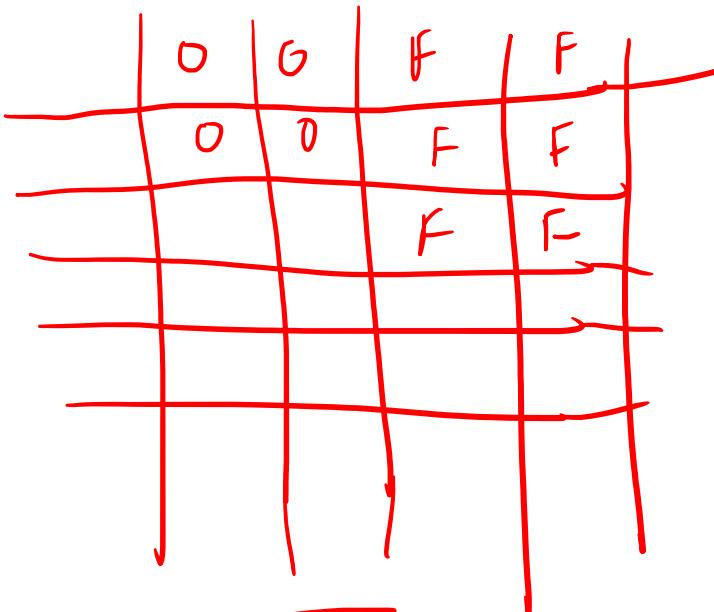


The combination of the two BBAs m_1 and m_2 according to Dempster's rule results in

- $m_*(\emptyset) = 0$
- $m_*(A) = \frac{1}{1-K} \sum_{B \cap C = A \neq \emptyset} m_1(B)m_2(C)$ with

$$K = \sum_{B \cap C = \emptyset} m_1(B)m_2(C)$$

Application: Evidential Occupancy Grids



$$\begin{aligned} P(O) &= 0.5 \\ P(F) &= 0.5 \end{aligned}$$

<https://github.com/ika-rwth-aachen/EviLOG>

unicaragile | Peter Wohlleben INSTITUTE FOR AUTOMOTIVE ENGINEERING AND MECHANICAL SYSTEMS

ika | RWTH AACHEN UNIVERSITY

32nd IEEE Intelligent Vehicles Symposium

A Simulation-based End-to-End Learning Framework for Evidential Occupancy Grid Mapping

11 – 17 July, 2021

Raphael van Kempen,
Bastian Lampe, Timo Woopen, Lutz Eckstein

Institute for Automotive Engineering (ika) – RWTH Aachen University

© IEEE. All rights reserved.

$$\begin{aligned} m(\{O\}) \\ m(\{O, F\}) \\ m(\{F\}) \end{aligned}$$

Sensor Data Fusion

Exercise 10

Prof. Dr.-Ing. Marcus Baum
M.Sc. Kolja Thormann

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

**DATA
FUSION Lab**

- Lecture review
- Questions
- Homework 9 solution
- Example - EKF issues
- Example - Multiplicative noise with EKF
- Problem 10 - Radar measurements with EKF
- Homework 10 presentation

What are disadvantages of the EKF?

- Might diverge for severe nonlinearities
- Need to compute Jacobians

Can you explain the white noise velocity model?

- State p_k
- Velocity as white noise $\dot{p}_k = w_k \sim \mathcal{N}(0, \sigma_w^2)$
- $p_{k+1} = p_k + T w_k$ with $T = t_{k+1} - t_k$

What is the assumption of the noise during sampling periods in the white noise acceleration model?

The noise is constant during sampling periods.

Assume a robot in 2D-space at position $[x_1 \ x_2]^T$ moving with velocity v_1 in x_1 direction and v_2 in x_2 direction. Its state is defined as $\mathbf{x} = [x_1 \ x_2 \ v_1 \ v_2]^T$.

- a) Formulate a motion model for the robot, assuming independent zero-mean Gaussian noise on each state element. Write a function implementing the motion model.

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \nu, \text{ with } \nu \sim \mathcal{N}(0, \mathbf{Q}) \text{ and } \mathbf{F} = \begin{bmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- b) In each time step, a sensor measures the robot's position. Formulate a measurement equation assuming independent zero-mean Gaussian noise and implement it as well.

$$y = \mathbf{H}\mathbf{x}_k + \mu, \text{ with } \mu \sim \mathcal{N}(0, \mathbf{R}) \text{ and } \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

c) Use the functions from a) and b) to implement a simulation using initial state

$\hat{\mathbf{x}}_{\text{init}} = [0 \text{m} \quad 0 \text{m} \quad 2 \text{m s}^{-1} \quad 2 \text{m s}^{-1}]$ (for each run, draw the true state from the prior), 10 time steps of length 1s, and covariances for the initial state \mathbf{C}_{init} , the transition noise \mathbf{Q} and the measurement noise \mathbf{R} as

$$\mathbf{C}_{\text{init}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{Q} = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0.2 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}.$$

d) Now based on a) and b), implement a predict and an update function for a Kalman filter. Use the Kalman filter to track the robot simulated with your function from c).

$$\mathbf{x}_{k+1|k} = \mathbf{F}\mathbf{x}_{k|k}$$

$$\mathbf{C}_{k+1|k} = \mathbf{F}\mathbf{C}_{k|k}\mathbf{F}^T + \mathbf{Q}$$

$$\mathbf{x}_{k+1|k+1} = \mathbf{x}_{k+1|k} + \underbrace{\mathbf{C}_{k+1|k}\mathbf{H}^T(\overbrace{\mathbf{H}\mathbf{C}_{k+1|k}\mathbf{H}^T + \mathbf{R}}^{\mathbf{S}})^{-1}}_{\mathbf{K}}(y - \mathbf{H}\mathbf{x}_{k+1|k})$$
$$\mathbf{C}_{k+1|k+1} = \mathbf{C}_{k+1|k} - \mathbf{K}\mathbf{S}\mathbf{K}^T$$

- e) Write a function which calculates the root mean square error (RMSE) of $n = 100$ simulation runs with the error as the Euclidean norm at the last time step.
- f) Finally, assume a worse sensor with noise covariance

$$\mathbf{R}_2 = \begin{bmatrix} 2.0 & 0 \\ 0 & 0.2 \end{bmatrix} .$$

Calculate the RMSE as in e). To deal with the noise, add a second sensor. Assume the measurements to be independent of each other and update the simulation and Kalman filter accordingly and observe the difference in RMSE. Now, instead of using the same type of sensor, use the sensor with \mathbf{R}_2 along a second sensor with noise covariance

$$\mathbf{R}_3 = \begin{bmatrix} 0.2 & 0 \\ 0 & 2.0 \end{bmatrix} .$$

Consider the state $\mathbf{x} = [x_1 \ x_2 \ v_1 \ v_2]^T$ consisting of position and velocity vectors. It uses the discrete linear motion model $\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \nu$ to move t seconds into the future with

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Due to the shape of the transition matrix, the uncertainty in the velocities influences the corresponding position uncertainty. Looking at the following scenario with high velocity uncertainty in both directions (as they are both completely wrong), we see that the covariance handles the true position being far off.



Now consider the state $\mathbf{x} = [x_1 \ x_2 \ v \ \alpha]^T$ consisting of position and polar velocity with speed v and orientation α . It moves with the discrete non-linear motion model

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k) = [x_1 + v \cos(\alpha) \ x_2 + v \sin(\alpha) \ v \ \alpha]^T + \nu .$$

The EKF would linearize using Jacobian

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \cos(\alpha) & -v \sin(\alpha) \\ 0 & 1 & \sin(\alpha) & v \cos(\alpha) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

For the same scenario as before, we would have a low variance in v , but a high in α . Linearizing at the current estimate with $\alpha = 0$, we get

$$\mathbf{F} = \alpha = 0 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & v \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

Linearized at this point, the x_1 uncertainty does not grow much due to the certain v . x_2 will get a larger uncertainty, but the resulting covariance does not cover the true position well. The reason is how the linearization handles the angular uncertainty of the velocity vector. The true uncertainty (green banana shape) would cover the true position well.



Assume the zero-mean Gaussian noise ν in the motion model is not additive, but multiplicative, resulting in a transition function

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \nu) .$$

The EKF is able to deal with this setting as well, applying the same linearization technique. Given the random variables are distributed with $\mathbf{x}_k \sim \mathcal{N}(\hat{\mathbf{x}}_k, \mathbf{C}_k)$ and $\nu \sim \mathcal{N}(0, \mathbf{Q})$, we get a covariance update

$$\mathbf{C}_{k+1} = \mathbf{F}\mathbf{C}_k\mathbf{F}^T + \mathbf{L}\mathbf{Q}\mathbf{L}^T ,$$

with

$$\mathbf{F} = \frac{\partial f}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\hat{\mathbf{x}}_k}, \mathbf{L} = \frac{\partial f}{\partial \nu} \Big|_{\mathbf{x}=\hat{\mathbf{x}}_k} .$$

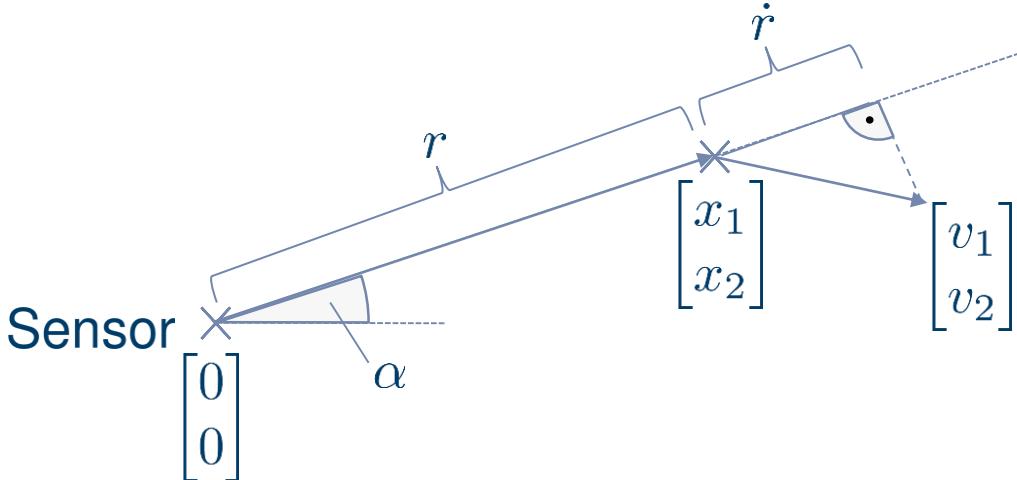
The same can be applied for multiplicative noise in the measurement model.

Problem 10 – Radar measurements with EKF

Consider a point target $\mathbf{x} = [x_1 \quad x_2 \quad v_1 \quad v_2]^T$ consisting of 2D position and velocity. The target is measured via radar (located at $[0 \quad 0]^T$), giving us measurements

$$\mathbf{y} = \begin{bmatrix} r \\ \alpha \\ \dot{r} \end{bmatrix},$$

consisting of range r , angle α (relative to the x_1 -axis), and range rate \dot{r} with the latter being a projection of the velocity vector onto the measurement direction $\dot{r} = \left[\frac{x_1}{\sqrt{x_1^2+x_2^2}} \quad \frac{x_2}{\sqrt{x_1^2+x_2^2}} \right]^T \cdot [v_1 \quad v_2]^T$. Formulate the measurement equation and calculate the Jacobian necessary for the EKF measurement update.



Hint:

$$\frac{\partial}{\partial x} \text{atan2}(y, x) = \frac{-y}{x^2+y^2}$$
$$\frac{\partial}{\partial y} \text{atan2}(y, x) = \frac{x}{x^2+y^2}$$

Problem 10: solution

$$h(\mathbf{x}) = \begin{bmatrix} \sqrt{x_1^2 + x_2^2} \\ \text{atan2}(x_2, x_1) \\ (v_1 x_1 + v_2 x_2) / \sqrt{x_1^2 + x_2^2} \end{bmatrix}$$
$$H = \begin{bmatrix} \frac{x_1}{\sqrt{x_1^2+x_2^2}} & \frac{x_2}{\sqrt{x_1^2+x_2^2}} & 0 & 0 \\ \frac{-x_2}{\sqrt{x_1^2+x_2^2}} & \frac{x_1}{\sqrt{x_1^2+x_2^2}} & 0 & 0 \\ \frac{-x_1(v_1 x_1 + v_2 x_2)}{\sqrt{x_1^2+x_2^2}^3} + \frac{v_1}{\sqrt{x_1^2+x_2^2}} & \frac{-x_2(v_1 x_1 + v_2 x_2)}{\sqrt{x_1^2+x_2^2}^3} + \frac{v_2}{\sqrt{x_1^2+x_2^2}} & \frac{x_1}{\sqrt{x_1^2+x_2^2}} & \frac{x_2}{\sqrt{x_1^2+x_2^2}} \end{bmatrix}$$

Consider the same motion model as in the last homework:

Assume a robot in 2D-space at position $[x_1 \ x_2]^T$ moving with velocity v_1 in x_1 direction and v_2 in x_2 direction. Its state is defined as $\mathbf{x} = [x_1 \ x_2 \ v_1 \ v_2]^T$.

- This time, instead of measuring the position directly, we measure the distance to a landmark P_i .
Formulate the measurement equation and write a function implementing it.

For now, we will assume only a single landmark is used. In that case, the measurement noise will be a 1×1 matrix.

- To use the measurement in the Kalman filter, we utilize the EKF formulas. Calculate the Jacobian of the measurement function from a) and implement the EKF measurement update.
Again, we assume that for now only one landmark exists, and \mathbf{R} will be a 1×1 matrix.

- c) Your implementation will be used to track the robot, getting measurements from a single landmark $P_1 = [5 \ 0]^T$.

The measured distance is noise corrupted with Gaussian noise having zero mean and variance 0.1.

For the initial state and the process covariance, we will use

$$C_{\text{init}} = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}, Q = \begin{bmatrix} 0.001 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0.02 & 0 \\ 0 & 0 & 0 & 0.02 \end{bmatrix}$$

For this, you will first need to implement the motion model of the robot.

d) To improve your estimate, use a second landmark $P_2 = [0 \quad 5]^T$. Formulate a stacked measurement equation and calculate the corresponding Jacobian.

You will need to implement new versions of your `measurement_model` and your `update` functions.

Hints:

- since we are now dealing with a multi-dimensional scenario, you should consider using `np.random.multivariate_normal` instead of `np.random.normal`.
- make sure you calculate the distances between x and P_i across the correct axis, since P is now multi-dimensional too.

Sensor Data Fusion

Exercise 11

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab

- Homework 10
- Intuition on the Kalman filter
- Sample exam 1

Assume the setting from the lecture. We have three classes $X = \{x_1, x_2, x_3\}$ and three possible measurements $Z = \{z_1, z_2, z_3\}$ with measurement sets

- $\Sigma_{x_1} = \{z_1\}$
- $\Sigma_{x_2} = \{z_1, z_2\}$
- $\Sigma_{x_3} = \{z_1, z_2, z_3\}$

- a) Implement a function to recursively track the class probability given the standard approach.
- b) Use the function from a) to track the probabilities over 20 time steps. Let $z^{(k)}$ be the measurement at time step k . Assume you receive measurements $z^{(7)} = z_2$, $z^{(13)} = z_3$, and $z^{(k)} = z_1$ for $k \in \{1, \dots, 20\} \setminus \{7, 13\}$. Plot the development of the class probabilities.
- c) Now implement a function to recursively track the class probabilities based on the random set approach, defining and using a generalized likelihood function. Use the same measurements as in b) to track the class probabilities. Visualize the results.

see notebook for solution

Sample Exam 1

The following questions represent a sample exam intended to refresh your knowledge about the discussed topics. Questions in the exam can be different. Keep in mind that everything from the lecture and exercises is relevant.

Also note that it is possible that some exam questions require you to do math.

Why do we need multiple sensors? Name and explain two sensor fusion concepts.

Sensors have advantages and disadvantages. Using multiple different sensors can balance these out.

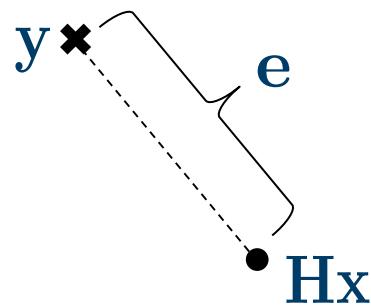
Two concepts are competitive fusion (multiple sensors measure the same space to increase accuracy) and complementary fusion (multiple sensors measure different spaces to increase completeness).

What is a measurement equation? Could you write down the linear measurement equation in a general form and explain the components in it?

The measurement equation projects the state onto the measurement space and describes the measurement as the sum of the projected state and an error.

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{e}$$

with state $\mathbf{x} \in \mathcal{R}^n$, measurement $\mathbf{y} \in \mathcal{R}^m$, measurement matrix $\mathbf{H} \in \mathcal{R}^{m \times n}$, and error $\mathbf{e} \in \mathcal{R}^m$.



What is the objective/cost function of least squares?

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

$$\mathbf{x}^{LS} = \underset{\mathbf{x}}{\operatorname{argmin}} \underbrace{\|\mathbf{y} - \mathbf{Hx}\|_{\mathbf{W}}^2}_{G(\mathbf{x})}$$

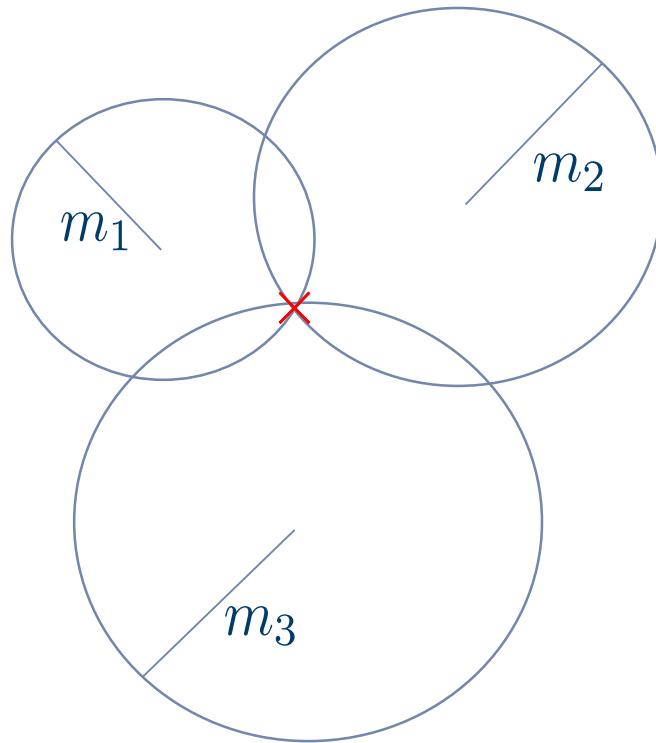
$$G(\mathbf{x}) = (\mathbf{y} - \mathbf{Hx})^T \mathbf{W} (\mathbf{y} - \mathbf{Hx})$$

$$G'(\mathbf{x}^{LS}) \stackrel{!}{=} 0$$

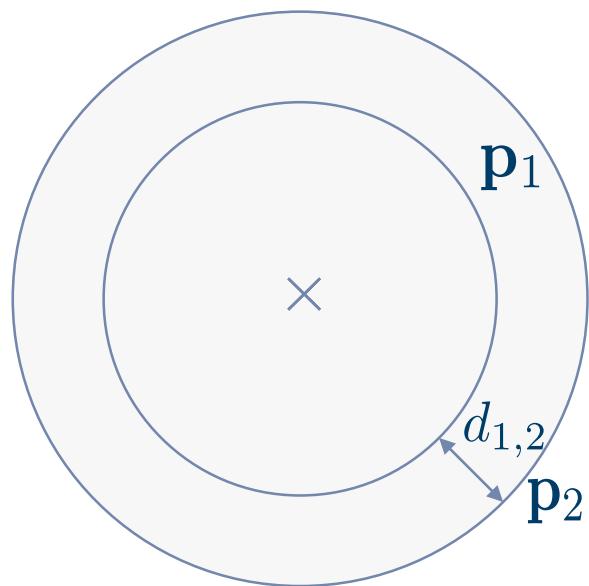
$$\mathbf{H}^T \mathbf{W} \mathbf{y} = \mathbf{H}^T \mathbf{W} \mathbf{H} \mathbf{x}^{LS}$$

$$\mathbf{x}^{LS} = (\mathbf{H}^T \mathbf{W} \mathbf{H})^{-1} \mathbf{H}^T \mathbf{W} \mathbf{y}$$

Can you outline schematically how trilateration works?



Can you explain how Time-difference-of-Arrival works?



$$d_{1,2} = \|x - p_2\| - \|x - p_1\| + e$$

Can you explain the steps of the Gauss-Newton algorithm?

$$\mathbf{x}^{LS} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\| \underbrace{\mathbf{y} - h(\mathbf{x})}_{e(\mathbf{x})} \right\|^2 \quad \mathbf{J} = e'(\hat{\mathbf{x}})$$

$$\begin{aligned} e(\mathbf{x}) &\approx e(\hat{\mathbf{x}}) + e'(\hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}}) \\ &= e(\hat{\mathbf{x}}) + \mathbf{J}\mathbf{x} - \mathbf{J}\hat{\mathbf{x}} \\ &= -(\mathbf{J}\hat{\mathbf{x}} - e(\hat{\mathbf{x}}) - \mathbf{J}\mathbf{x}) \end{aligned}$$

$$\mathbf{x}^{LS} \approx \underset{\mathbf{x}}{\operatorname{argmin}} \left\| \underbrace{\mathbf{J}\hat{\mathbf{x}} - e(\hat{\mathbf{x}})}_{\mathbf{y}} - \underbrace{\mathbf{J}}_{\mathbf{H}} \mathbf{x} \right\|^2$$

The resulting least squares estimate is based on an initial guess $\hat{\mathbf{x}}$. The result is then used in a next iteration step as the new guess. This is repeated until convergence.

What is BLUE?

- Quality of an estimator measured via MSE
- Goal: Minimum MSE Estimator
- Problem: unknown \mathbf{x}
- Solution: assume unbiasedness

$$MSE(\theta_y) = \text{Tr Cov}[\theta_y] + \cancel{\|E[\theta_y] - \mathbf{x}\|^2}$$

- New goal: Minimum Variance Unbiased (MVU) Estimator
- If MVU estimator is linear, we have the Best Linear Unbiased Estimator (BLUE)

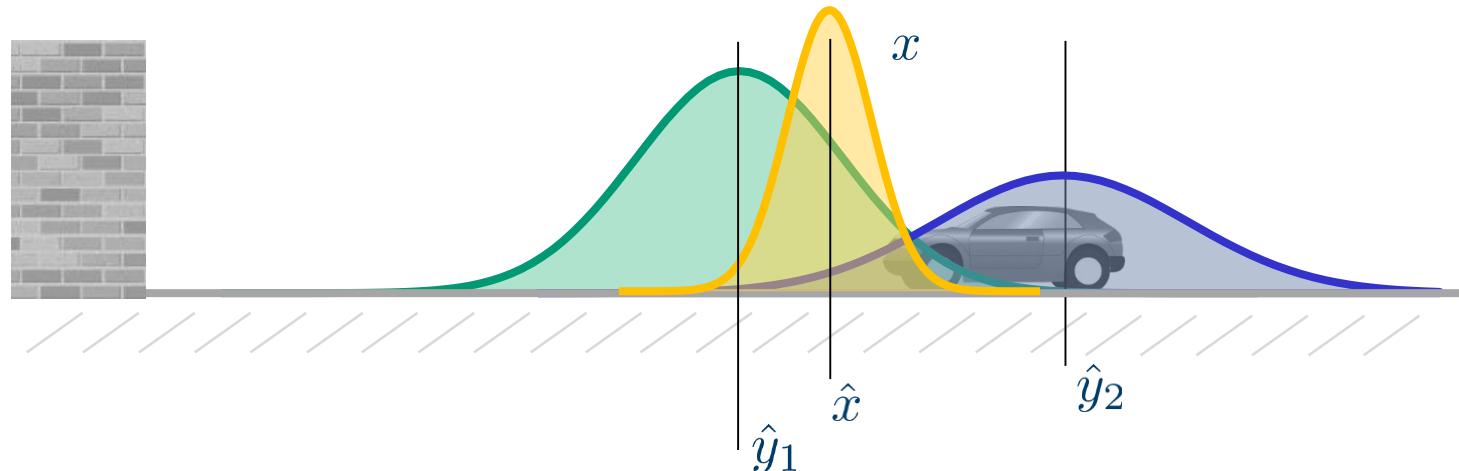
What is the difference between Bayesian and Fisher approach?

- We have $\mathbf{y} = \mathbf{Hx} + \mathbf{e}$
- Fisher: $\mathbf{x}?, \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- Bayesian: $\mathbf{x} \in \mathcal{N}(\boldsymbol{\mu}_x, \mathbf{C}_{xx}), \mathbf{e} \in \mathcal{N}(\mathbf{0}, \mathbf{C}_{ee})$
- In general
 - Prior: $p(\mathbf{x})$
 - Likelihood: $p(\mathbf{y}|\mathbf{x})$
 - $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$

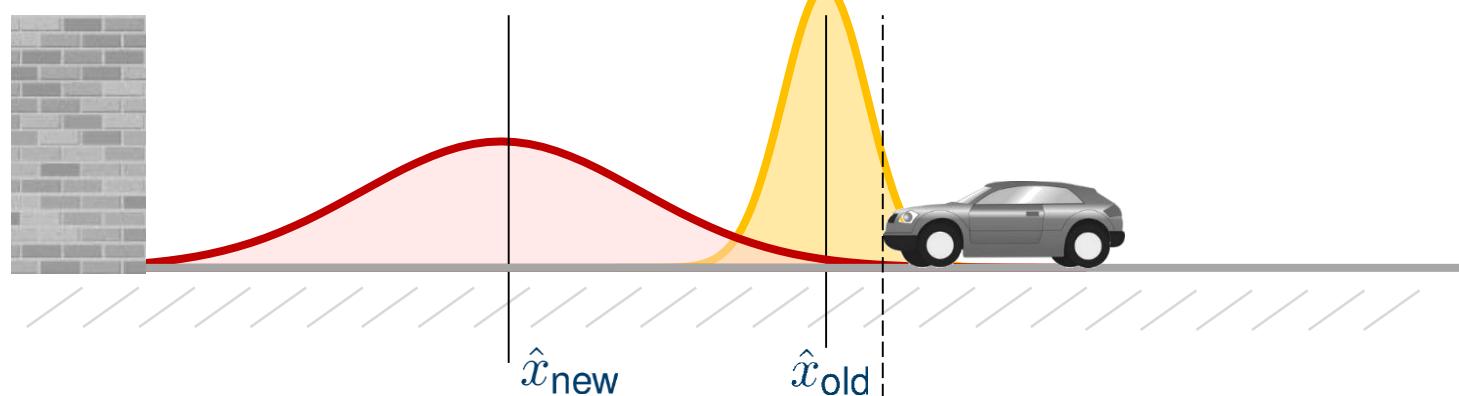
Why is it difficult to obtain the optimal Bayesian estimator?

- No closed-form solution (in general)
- complete prior and likelihood information required
- Kalman filter is optimal Bayesian filter if
 - Linearity is given
 - All distributions are Gaussian

In the Kalman filter, what happens to the state variance during measurement update? What happens during time update?



measurement update:
variance decreases



time update:
variance increases

Assume during the time update, the state moves with a noise corrupted velocity. What would happen to the state covariance if we want to estimate the state for $T \rightarrow \infty$?

We have $\sigma_{x_{k+1}}^2 = \sigma_{x_k}^2 + T^2 \sigma_v^2$.

For $T \rightarrow \infty$, we get $\sigma_{x_{k+1}}^2 \rightarrow \infty$.

How can unknown correlation between estimates be handled?

Find uncorrelated ellipse encapsulating correlated one.

$$\tilde{\mathbf{C}} = \begin{bmatrix} \mathbf{C}^{xx} & \mathbf{C}^{xy} \\ \mathbf{C}^{yx} & \mathbf{C}^{yy} \end{bmatrix}$$

$$\mathbf{C} > \tilde{\mathbf{C}}$$

$$\mathbf{C} = \begin{bmatrix} \frac{1}{0.5-\alpha} \mathbf{C}^{xx} & 0 \\ 0 & \frac{1}{0.5+\alpha} \mathbf{C}^{yy} \end{bmatrix}$$

with $\alpha \in (-0.5, 0.5)$.

What assumptions does the discrete time Nearly Constant Velocity model make about the noise?

The acceleration is assumed to be zero-mean white noise which is constant during each sampling period.

Sensor Data Fusion

Exercise 12

Prof. Dr.-Ing. Marcus Baum

Dr. Kolja Thormann

M.Sc. Simon Steuernagel

www.fusion.informatik.uni-goettingen.de



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

DATA
FUSION Lab

- Lecture review
- Questions
- Sample exam 2

Sample Exam 2

The following questions represent a sample exam intended to refresh your knowledge about the discussed topics. Questions in the exam can be different. Keep in mind that everything from the lecture and exercises is relevant.

Also note that it is possible that some exam questions require you to do math.

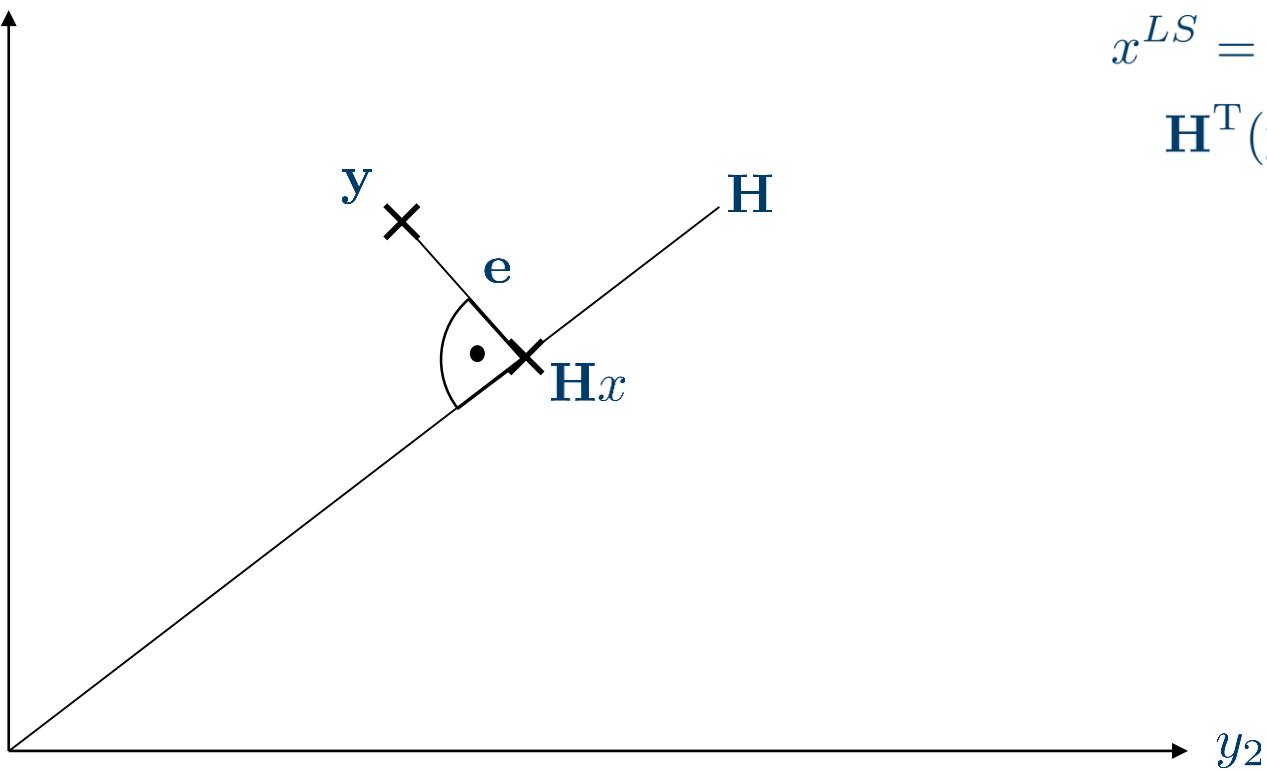
What are the assumptions for least squares?

$$\mathbf{y} = \mathbf{Hx} + \mathbf{e}$$

$$\mathbf{H} \in \mathbb{R}^{m \times n}, \quad m \geq n, \quad \text{rank}(\mathbf{H}) = n$$

Can you explain the normal equation intuitively?

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} 1 \\ 1 \end{bmatrix}}_{\mathbf{H}} x + \mathbf{e}$$

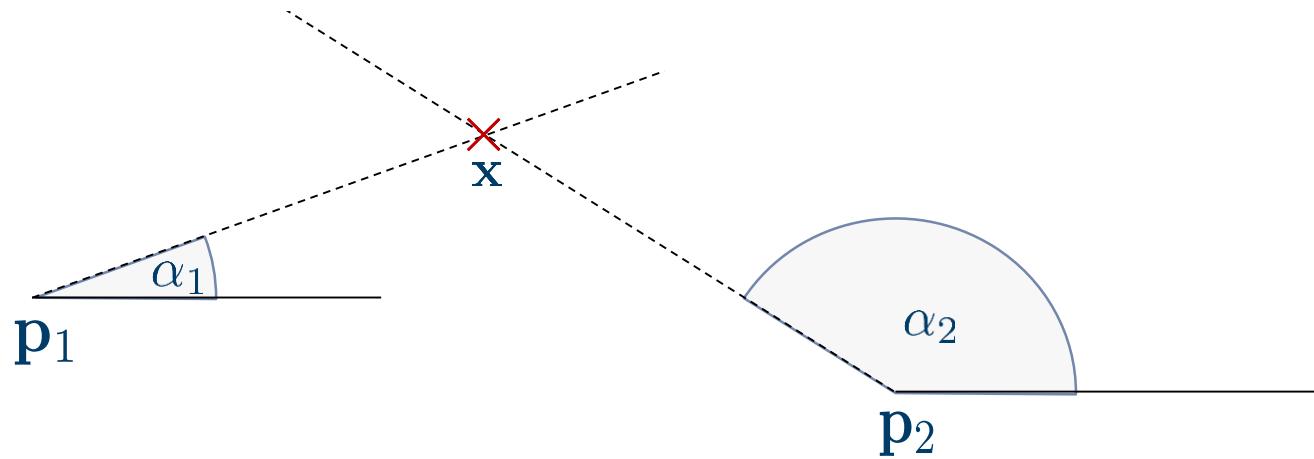


for $\mathbf{W} = \mathbf{I}$, we have

$$x^{LS} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{y}$$

$$\mathbf{H}^T(\mathbf{y} - \mathbf{H}x) = 0$$

Can you explain how Triangulation works? Why can you not use the normal equation to solve it?



As the measurement equation is non-linear, we are unable to apply least square formula. We can still solve the problem by applying a closed-form solution or using iterative optimization.

What options are there to solve non-linear least squares?

You can use iterative optimization, which are precise but suffer from local minima, or closed-form solutions, which are computationally efficient but can be bad if the noise is too high. A closed-form approximation can be used as an initial guess for an iterative optimization method.

What is an estimator?

An estimator is a function $\theta_y : \mathbb{R}^m \rightarrow \mathbb{R}^n$ providing an estimate for the desired variable \mathbf{x} based on an observation \mathbf{y} . A linear estimator has the form

$$\theta_y = \mathbf{K}\mathbf{y} + \mathbf{b} .$$

The quality of an estimator can be measured using the mean squared error (MSE)

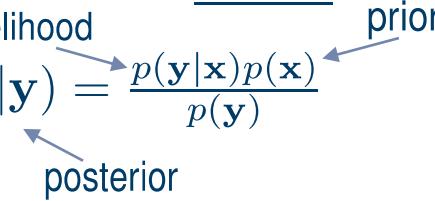
$$MSE(\theta_y) = E[||\theta_y - \mathbf{x}||^2] .$$

How can the MSE be decomposed in the Fisher approach?

Note: Fisher approach, so no statistical information about \mathbf{x} .

$$\begin{aligned} E[||\theta_y - \mathbf{x}||^2] &= E[\text{Tr}((\theta_y - \mathbf{x})(\theta_y - \mathbf{x})^T)] \\ &= \text{Tr}(E[\theta_y^2] - 2\theta_y \mathbf{x} + \mathbf{x}^2) \\ &= \text{Tr}(E[\theta_y^2] - 2E[\theta_y]\mathbf{x} + \mathbf{x}^2 + E[\theta_y]^2 - E[\theta_y]^2) \\ &= \text{Tr}(E[\theta_y^2] - E[\theta_y]^2) + \text{Tr}(E[\theta_y]^2 - 2E[\theta_y]\mathbf{x} + \mathbf{x}^2) \\ &= \underset{\text{Covariance}}{\text{Tr}(\text{Cov}[\theta_y])} + \underset{\text{Bias}}{||E[\theta_y] - \mathbf{x}||^2} \end{aligned}$$

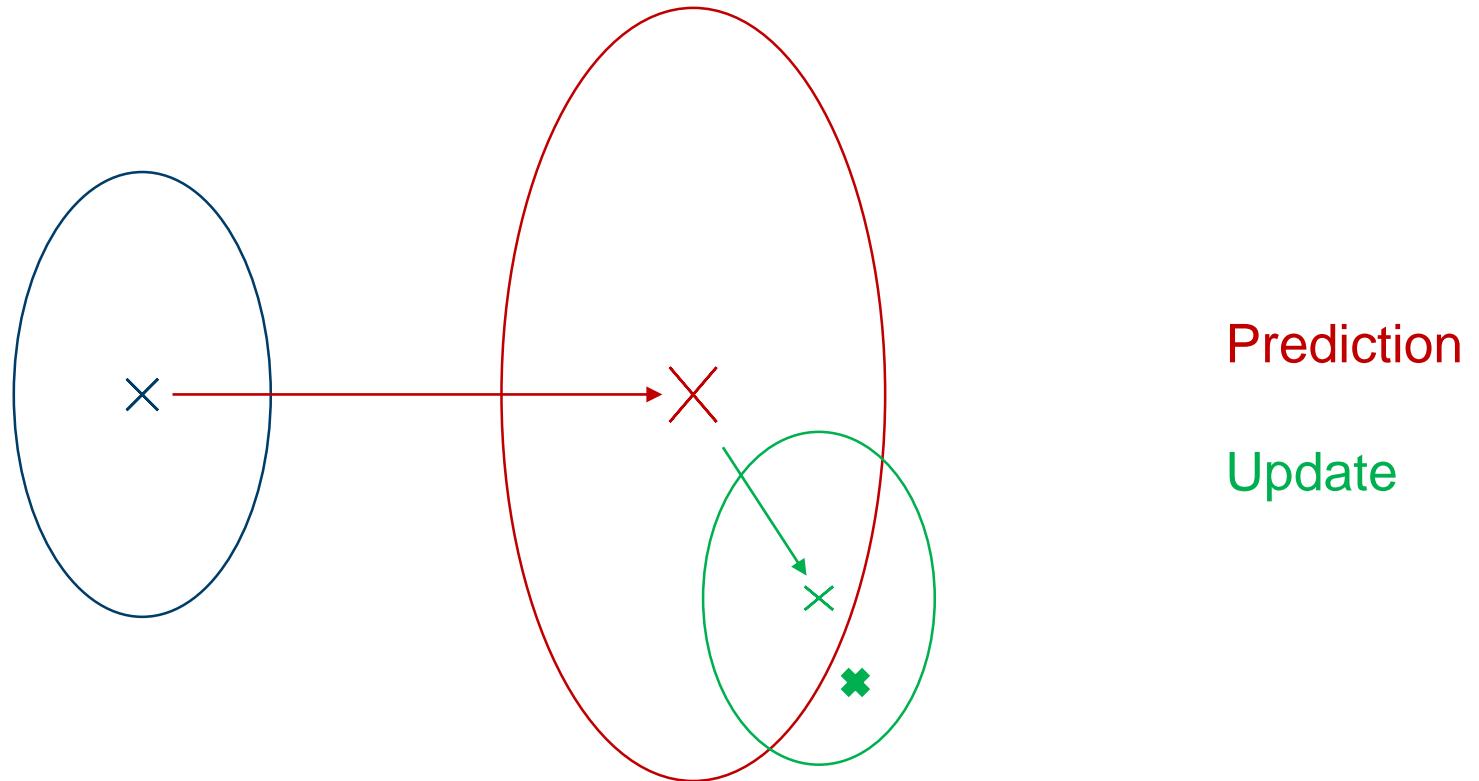
What is the optimal Bayesian estimator and how can you compute it?

- $\theta_y^{opt} = E[\mathbf{x}|\mathbf{y}]$
- Given $p(\mathbf{x})$, $p(\mathbf{y}|\mathbf{x})$
- $E[\mathbf{x}|\mathbf{y}] = \int \mathbf{x} p(\mathbf{x}|\mathbf{y}) d\mathbf{x}$
- $$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}$$


What can be said about the posterior distribution if all other distributions involved are Gaussian?

The posterior $p(x|y)$ is also Gaussian.

Can you intuitively explain the Kalman filter?



What happens in the Kalman filter update for $\mathbf{H}=\mathbf{I}$ if the noise of the prior is much higher than the measurement noise and vice versa?

- Special cases for $\mathbf{H} = \mathbf{I}$, i.e.,

$$\hat{x}_{k+1} = \hat{x}_k + \mathbf{C}_k(\mathbf{C}_k + \mathbf{C}_{ee})^{-1}(y_k - \hat{x}_k)$$

$$\mathbf{C}_{k+1} = \mathbf{C}_k - \mathbf{C}_k(\mathbf{C}_k + \mathbf{C}_{ee})^{-1}\mathbf{C}_k$$

$$\hat{x}_{k+1} = \hat{x}_k + \mathbf{C}_{k+1}\mathbf{C}_{ee}^{-1}(y_k - \hat{x}_k)$$

$$\mathbf{C}_{k+1} = (\mathbf{C}_k^{-1} + \mathbf{C}_{ee}^{-1})^{-1}$$

- $\text{Tr}[\mathbf{C}_{ee}] \rightarrow \infty$

$$\hat{x}_{k+1} = \hat{x}_k$$

$$\mathbf{C}_{k+1} = \mathbf{C}_k$$

- $\text{Tr}[\mathbf{C}_k] \rightarrow \infty$

$$\hat{x}_{k+1} = y_k$$

$$\mathbf{C}_{k+1} = \mathbf{C}_{ee}$$

What is the idea of the EKF? Describe it using the time update formulas.

- Setup: nonlinear transition function $\mathbf{x}_{k+1} = a(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}$ with $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$
- Problem: we require a transition matrix \mathbf{A}_k to transform the state covariance
- Solution: linearization around estimate $\hat{\mathbf{x}}_k$

$$a(\mathbf{x}_k, \mathbf{u}_k) \approx a(\hat{\mathbf{x}}_k, \mathbf{u}_k) + \mathbf{A}_k(\mathbf{x}_k - \hat{\mathbf{x}}_k), \mathbf{A}_k = \frac{\partial a(\mathbf{x}_k, \mathbf{u}_k)}{\partial \mathbf{x}_k}$$

$$\hat{\mathbf{x}}_{k+1} = a(\hat{\mathbf{x}}_k, \mathbf{u}_k)$$

$$\mathbf{C}_{k+1} = \mathbf{A}_k \mathbf{C}_k \mathbf{A}_k^T + \mathbf{Q}$$

What are disadvantages of the EKF?

- Might diverge for severe nonlinearities
- Need to compute Jacobians

How can you classify uncertain information?

- Imprecise information, e.g., more than ..., an interval, ...
- Affected by random effect, e.g., probability function
- Vague information, e.g., approximate values
- Erroneous information, e.g., systematic errors

What is the difference between the random set based method to the classical approach?

The random set based methods replace the likelihood by a generalized likelihood, which describes the probability of the measurement belonging to a set. Each state x is then mapped to a set in the measurement space.