UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCES
DEPARTMENT OF
MATHEMATICS AND INFORMATICS

Katarina Vidojević

Localization of vehicles using Factor Graphs and Belief Propagation

- Master thesis -

Mentor:
Dr Dejan Vukobratović

Novi Sad, 2024.

I would like to express my heartfelt gratitude to my mentor Dr Dejan Vukobratović for his advice, guidance, help, support, and expertise throughout the research and writing of this thesis. It has been a true privilege to work under his mentorship.

I would also like to extend my thanks to the other members of the board, Dr Nataša Krklec Jerinkić and Dr Mladen Kovačević for their valuable insights and feedback during the process. To all other professors, this master's program has been a wonderful experience that has enriched my academic and personal growth.

My special credit and recognition go to my friends and colleagues for their unwavering support, assistance, and understanding amid all of the challenges on this journey. Their companionship and collaboration have been invaluable, and I am truly grateful for their presence in my life. Especially to Milica and Marko, my deepest gratitude for the utmost support, and everything they have done for me.

I am beyond grateful to my mother Dragana, my father Vojislav, and my brother Pavle, who have provided me with unconditional love, understanding, and encouragement. Their belief in me has lifted my spirits, strength, and motivation throughout this process, and for that, I am eternally thankful.

Contents

1 Introduction

Throughout the ages, people have been looking for ways to move faster and easier. The first steps in that direction were simple, but revolutionary for their time. Thousands of years ago, our ancestors discovered that they could use sleds to transport goods over snow. This discovery was crucial, but the real breakthrough occurred with the invention of the wheel, which dates back to around 4200–4000 BC in Mesopotamia. Initially used for pottery, the true impact of the wheel was seen with the development of wheeled vehicles between 3500 and 3350 BC. The wheel enabled the creation of carts, chariots, and carriages, which used horses for traction. In ancient Egypt, Greece, and Rome, these chariots became the primary means of transporting people and goods, shaping trade routes and war strategies.

During the Middle Ages, vehicles were further developed. Carriages and wagons became more sophisticated, and horse-drawn vehicles were commonly seen on European roads.

However, the major advancement in vehicle development occurred during the Industrial Revolution in the 18th and 19th centuries. The introduction of steam engines led to the creation of the first steam locomotives and ships. These vehicles enabled faster and more efficient transport, revolutionizing trade and everyday life. At the end of the 19th century, the world saw yet another important moment—the advent of the first gasoline-powered cars. After working for different companies, Carl Benz and Gottlieb Daimler developed the first cars in history at Stuttgart and Mannheim, respectively. These vehicles were the first version of the modern cars we know today. There was a gap of several years before the invention of the automobile and its economic exploitation. Although initially luxurious and expensive, Henry Ford changed everything by introducing the assembly line in 1908. His Model T became affordable to the masses, making the car an essential part of everyday life. For more details and further insights on this topic, see [2].

The development of aviation at the beginning of the 20th century further expanded the horizons of transport. In 1903, the Wright brothers made the first successful flight with a powered airplane, laying the foundations for the development of commercial and military aviation. Additional information on this topic can be found in [1]. While airplanes conquered the skies, on the seas, container ships revolutionized maritime transport, enabling more efficient global trade.

Entering the 21st century, the world has witnessed incredible technological advances in the world of vehicles. Electric vehicles became increasingly popular as a response to environmental challenges. Companies like Tesla and Nissan led this revolution, developing vehicles that do not rely on fossil fuels. At the same time, the development of autonomous vehicles promises to revolutionize the way we travel, aiming to in-

crease road safety and efficiency. From package delivery to ocean exploration, drones, and autonomous underwater vehicles have become standard in many industries. These advanced systems enable new ways of exploration, research, surveillance, and transportation, adding another dimension to the world of vehicles.

Vehicles have been crucial to societal development throughout history. By enabling faster and more efficient transport, they have improved the quality of life, connected people and communities, and spurred economic growth. Whether it is cars, trains, planes, or ships, vehicles have shaped the world we live in. Today, with advances in technology, vehicles continue to play a key role in our daily lives, helping us move, explore, and connect in ways we could not have imagined just a few decades ago.

Vehicle localization, the process of accurately estimating a vehicle's position, has an essential role across numerous aspects of modern transportation and technology. In the field of autonomous vehicles, precise localization is indispensable. Self-driving cars depend on accurate position estimates to navigate roads safely, avoid obstacles, and adhere to traffic regulations. Without this capability, the promise of fully autonomous driving would remain unfulfilled, as the vehicle's ability to understand its surroundings and make informed decisions hinges on knowing exactly where it is at all times. Beyond autonomous driving, vehicle localization is at the heart of everyday navigation systems. Whether it is a driver using a GPS to find the fastest route or a logistics company optimizing delivery routes, the ability to specify a vehicle's location enables efficient and timely travel. In the context of vehicle-to-vehicle communication, accurate localization is crucial for cooperative driving systems. Vehicles need to understand their relative positions to each other to prevent collisions, facilitate platooning, and enhance driver-assistance systems. This interconnectedness between vehicles promises safer roads and more efficient traffic flow. Urban planners and traffic management systems also benefit from precise vehicle localization. By analyzing the movement of vehicles, they can devise strategies to reduce congestion and improve the flow of traffic in cities, leading to a smoother and more efficient transportation network. Safety and security are other critical areas where vehicle localization makes a difference. Emergency response systems can quickly locate vehicles involved in accidents, potentially saving lives by reducing response times. Additionally, localization technology is employed in theft recovery systems to track and recover stolen vehicles, offering peace of mind to vehicle owners. Finally, in the broader scope of Intelligent Transportation Systems (ITS), vehicle localization serves as a foundation. ITS integrates various technologies to enhance transportation efficiency, safety, and environmental sustainability. By providing accurate location data, these systems can manage traffic more effectively, disseminate real-time information, and ultimately improve the overall transportation experience.

In essence, vehicle localization is a cornerstone of modern transportation, enabling a wide range of applications that contribute to safer, more efficient, and more intelligent mobility solutions.

This study aims to present an algorithm for improving the precision of vehicle position estimation. To model the vehicle's environment, we will use a factor graph,

where the factor nodes represent sensor measurements such as GPS and odometry, and the variable nodes represent the vehicle's positions. The accuracy of vehicle position estimation will be based on the Kalman Filter and Belief Propagation algorithms. Measurement data will be simulated, and the algorithm will be implemented using the Python programming language.

2 Factor Graph

The graph G is an ordered pair (V, E), where $G = (V, E)$, and V and E are finite sets, with V being non-empty. The elements of the set V are called vertices, and the elements of the set E are edges. The edges are unordered pairs of vertices (not necessarily distinct). The set of neighbors of a vertex v in the graph G consists of all vertices u such that there exists an edge uv in G. If we denote this set as $N_G(v)$, then $N_G(v) = \{u|uv \in E(G)\}$. If it is clear which graph is being referred to, the index G can be omitted, and we can simply write $N(v)$ instead of $N_G(v)$. A bipartite graph is a graph in which the nodes can be divided into two disjoint sets such that there are no edges connecting nodes within the same set.

Factor graph is a bipartite graph $G = (F, V, E)$ that has two types of nodes: factor nodes $f_i \in F$ and variable nodes $x_j \in V$. Variable nodes represent random variables or unknowns in the model and they can be used to quantitatively describe an event. For example, we can use a random variable to denote if Pavle cooks. If Pavle cooks, the random variable takes a value of 1, and 0 if he does not cook. Each variable node typically corresponds to a specific element in the model. Factor nodes represent functions on subsets of the variables. They encode the dependencies or relationships between variables. Factors can be seen as conditional probability distributions or error functions that capture the information from the measurements or constraints in the model. The edges $e_{ij} \in E$ in the factor graph are always between variable nodes and factor nodes, indicating which variables are involved in each factor. This connectivity represents the dependencies or influences between variables. With these definitions, a factor graph G defines the factorization of a global function $f(X)$ as $f(X) = \prod_i f_i(X_i)$ where X_i is the set of all variables x_j connected by an edge to factor f_i.

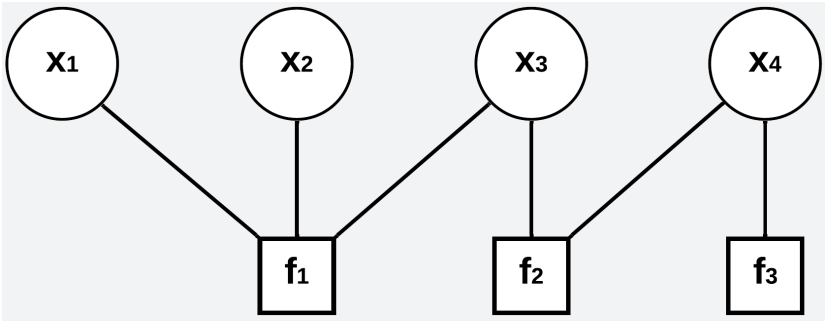


Figure 2.1: Example of factor graph.

$$f(x_1, x_2, x_3, x_4) = f_1(x_1, x_2, x_3) f_2(x_3, x_4) f_3(x_4)$$

It is used to represent the global function as a product of local functions, which allows efficient analysis and inference using algorithms such as Belief Propagation. The benefits of constructing a factor graph are visualization and analysis of the structure of the probabilistic model, understanding the dependencies between variables, and efficiently performing inference and optimization tasks using algorithms designed for factor graphs. Factor graphs are widely used in fields such as computer vision, robotics, and machine learning for tasks such as sensor fusion, state estimation, and inference. This chapter relies on [6] and [8] which have provided essential insights and foundational understanding for the concepts presented.

3 Belief Propagation on Factor Graphs

Belief Propagation (BP) is an algorithm developed by Judea Pearl[1] in the early 1980s as part of his research in artificial intelligence and probabilistic reasoning. Pearl introduced BP as a method for efficiently computing marginal probabilities in Bayesian networks but also works in other probabilistic graphical models such as Markov Random Fields and Factor graphs. For more details in this chapter we refer to [5].

Probabilistic inference is the process of estimating statistical properties of unknown variables from known or observed quantities. It involves determining how variables of interest behave given the available evidence. Two forms of probabilistic inference are marginal inference and MAP (Maximum A Posteriori) inference. Marginal inference computes the marginal distribution of each variable. MAP inference computes the mode of the posterior distribution identifying the most probable values of the variables given the observed data.

Belief Propagation is a powerful and efficient algorithm for inference in graphical models, such as factor graphs. It operates through an iterative process where messages are exchanged between nodes in the graph. Each iteration of the BP algorithm begins with **Variable-to-Factor Message Passing**. During this phase, each variable node communicates its current belief, based on the messages received from all other factor nodes. Next, in the **Factor-to-Variable Message Passing** step, each factor node sends messages to its neighboring variable nodes. These messages are influenced by the information received from other adjacent variable nodes. Essentially, the factor node aggregates the information it has and sends it to the variable nodes. Following these exchanges, the **Belief Update** occurs. Each variable node recalculates its belief about itself by considering the messages it has received from all the factor nodes it is connected to. Together, these steps form the core of the Belief Propagation algorithm. The iterative nature of BP ensures that the beliefs about each variable and factor become more accurate over time.

Now, we will present the Sum-Product BP algorithm that solves the marginal inference problem.

[1] Judea Pearl (1936-) is an Israeli-American computer scientist.

3.1 Sum product algorithm

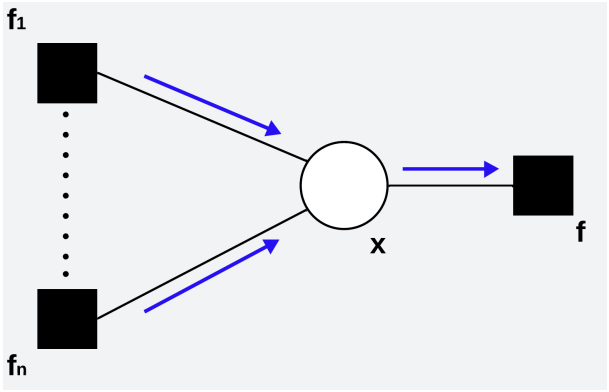
Acording to [9] we bring up the following explantion.

The Sum-Product Update Rule: The message sent from a node v on an edge e, is the product of the local function at v (or the unit function if v is a variable node) with all messages received at v on edges other than e , summarized for the variable associated with e.

We have the following notation:

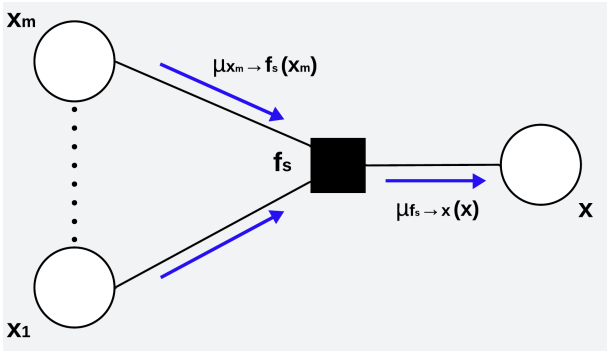
- $\mu_{x\to f}(x)$ - the message sent from node x to node f
- $\mu_{f\to x}(x)$ - the message sent from node f to node x
- $n(x)$ - the set of neighbors of a given node x in a factor graph

Variable-to-Factor node message:



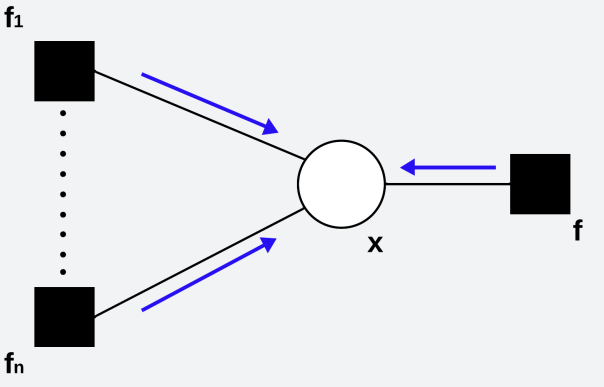
$$\mu_{x\to f}(x) = \prod_{h\in n(x)\setminus\{f\}} \mu_{h\to x}(x) \tag{3.1}$$

Factor-to-Variable node message:



$$\mu_{f_s\to x}(x) = \sum_{x_1}\cdots\sum_{x_m} f_s(x, x_1, ..., x_m) \prod_{x_l\in n(f_s)\setminus\{x\}} \mu_{x_l\to f_s}(x_l) \tag{3.2}$$

Belief update:



$$p(x) = \prod_{f_s \in n(x)} \mu_{f_s \to x}(x) \tag{3.3}$$

4 Normal distribution

The normal distribution plays a central role in describing uncertainty and probabilistic state estimation in vehicle localization using KF and BP. In the Kalman filter, each state (precited or updated) estimate is modeled as normal distribution. Similarly, Belief Propagation relies on message passing, which often involves normal distributions in linear systems. This assumption enables simple analytical processing, as the normal distribution has several key properties that make filtering and probability propagation highly efficient. In this chapter, we will discuss the fundamental characteristics of normal distribution.

Let $(\Omega, \mathcal{F}, P)$ be probability space. A function $\mathbf{X} : \Omega \to \mathbb{R}^n$ is called random variable if $\mathbf{X}^{-1}(S) \in \mathcal{F}$ for every Borel set $S \in \mathcal{B}_{\mathbb{R}^n}$. Random variable $\mathbf{X} = (X_1, ..., X_n)$ is absolutely continuous if there exists nonnegative integrable function $\varphi_{\mathbf{X}} : \mathbb{R}^n \to \mathbb{R}$, such that for every Borel set $S \in \mathcal{B}_{\mathbb{R}^n}$ it holds that

$$P(\{\mathbf{X} \in S\}) = \int \cdots \int_S \varphi_{\mathbf{X}}(x_1, ...x_n)\, dx_1 ... dx_n.$$

Function $\varphi_{\mathbf{X}}(\mathbf{x})$ is called density of random variable $\mathbf{X}$.

For every one-dimensional absolutely continuous random variable X we define its expectation and variance as follows $\mathbb{E}(X) = \int_{-\infty}^{\infty} x\varphi_X(x)\ dx$ and $Var(X) = \mathbb{E}(X^2) - \mathbb{E}^2(X)$.

If X and Y are two one-dimensional absolutely continuous random variables then their covariance is defined in the following way $cov(X, Y) = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y)$. For more details about probability theory, we refer to [10].

The normal distribution, or Gaussian[1] distribution holds the greatest significance among the probability distributions of absolutely continuous random variables. Many random variables that arise in relation to experiments have a normal distribution. A large number of random variables have an approximately normal distribution. If a random variable does not have a normal distribution, even approximately, it can be transformed into a normally distributed random variable through relatively simple transformations. It has a wide range of applications in mathematical statistics.

[1] Carl Friedrich Gauss (1777 – 1855) was a German mathematician, astronomer, geodesist, and physicist who contributed to many fields in mathematics and science.

Let X be a one-dimensional random variable. A random variable X has a normal distribution $\mathcal{N}(m, \sigma^2)$, $m \in \mathbb{R}$ and $\sigma > 0$, if its probability density function is

$$\varphi_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right), x \in \mathbb{R}.$$

The parameter m is called mean and represents an expectation $\mathbb{E}[X]$ and σ^2 is variance, i.e. $Var(X)$.

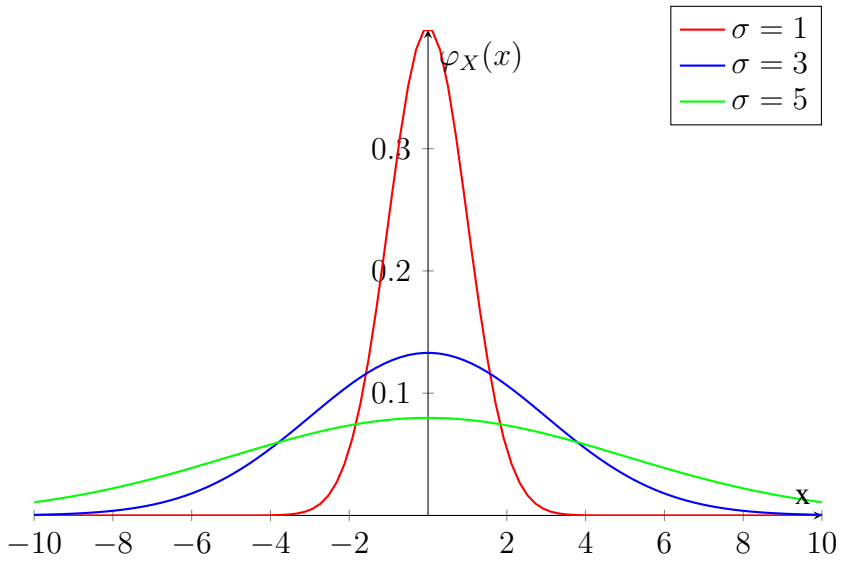


Figure 4.1: Normal distribution with parameters $m = 0$ i $\sigma \in \{1, 3, 5\}$

The graphs of the density curves vary depending on the values of the parameters m and σ , but some common characteristic features can be observed. All density curves are symmetric with respect to the line $x = m$. The maximum point is at $(m, \frac{1}{\sigma\sqrt{2\pi}})$. To the left and right of the maximum point, the density curve symmetrically decreases to zero. Changing the value of the parameter m will result in a translation of the density curve along the x-axis. Changing the value of the parameter σ will lead to a change in the flatness of the density curve. The larger the σ, the lower the maximum value of the curve, but the dispersion around the point $x = m$ is greater. All density curves have the shape of a bell.

The distribution function of the random variable X with normal $\mathcal{N}(m, \sigma^2)$ distribution is

$$F_X(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(t-m)^2}{2\sigma^2}\right) dt, x \in \mathbb{R}.$$

The probability that the random variable $X : \mathcal{N}(m, \sigma^2)$, takes a value within the interval (a, b)i s given by

$$P\{a < X < b\} = \int_{a}^{b} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) dx.$$

This integral represents the area under the normal distribution curve between a and b, which corresponds to the probability that the random variable X falls within that interval.

Multivariate normal distribution

Let Σ be a symmetric, positive-definite $n \times n$ matrix, $\mathbf{x} = (x_1, \ldots, x_n)$ a column vector, $\mathbf{x}^T$ the transpose of $\mathbf{x}$, and $\mathbf{m} = (m_1, \ldots, m_n)$ a fixed column vector. Then

$$\varphi_{\mathbf{x}}(x_1, \ldots, x_n) = \frac{1}{(2\pi)^{n/2} \det(\Sigma)^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m})^T \Sigma^{-1} (\mathbf{x} - \mathbf{m})\right)$$

is the density of the n-dimensional normal distribution $\mathcal{N}(\mathbf{m}, \Sigma)$.

In the case $n = 2$, where $\mathbf{X} \sim \mathcal{N}(\mathbf{m}, \Sigma)$,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mathbf{m} = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \sigma_1 & \rho_{X_1X_2} \\ \rho_{X_1X_2} & \sigma_2 \end{bmatrix}$$

we have that $m_1 = \mathbb{E}[X_1]$, $m_2 = \mathbb{E}[X_2]$, $\sigma_1 = var(X_1)$, $\sigma_2 = var(X_2)$ and $\rho_{X_1X_2} = cov(X_1, X_2)$.

4.1 Product of Gaussian densities

We denote $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \Sigma) \propto \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})\right)$ if there exists $\alpha > 0$ such that $\varphi_X(\mathbf{x}) = \alpha \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})\right)$ where $\varphi_X(\mathbf{x})$ represents densitiy function for normal distribution $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \Sigma)$.

If we have two two-dimensional normal distributions $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)$ and $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_2, \Sigma_2)$, their product will also give normal distribution $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_3, \Sigma_3)$, where

$$\Sigma_3^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1} \text{ and } \boldsymbol{\mu}_3 = \Sigma_3 \left(\Sigma_1^{-1} \boldsymbol{\mu}_1 + \Sigma_2^{-1} \boldsymbol{\mu}_2\right).$$

In the following, we provide a proof of this statement.
Density function for first one: $\varphi_1(\mathbf{x}) = \frac{1}{2\pi\sqrt{\det(\Sigma_1)}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1)\right)$.
Density function for second one: $\varphi_2(\mathbf{x}) = \frac{1}{2\pi\sqrt{\det(\Sigma_2)}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_2)^T \Sigma_2^{-1} (\mathbf{x} - \boldsymbol{\mu}_2)\right)$.

The product of these densities is given with

$$\varphi_1(\mathbf{x}) \cdot \varphi_2(\mathbf{x}) = \frac{1}{(2\pi)^2 \sqrt{\det(\Sigma_1)\det(\Sigma_2)}} \exp\left(-\frac{1}{2}\left[(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) + (\mathbf{x} - \boldsymbol{\mu}_2)^T \Sigma_2^{-1} (\mathbf{x} - \boldsymbol{\mu}_2)\right]\right)$$

Below, we will consider just the exponent of the exponential function.

$$-\tfrac{1}{2}[(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) + (\mathbf{x} - \boldsymbol{\mu}_2)^T \Sigma_2^{-1} (\mathbf{x} - \boldsymbol{\mu}_2)]$$

$$= -\tfrac{1}{2}[\mathbf{x}^T \Sigma_1^{-1} \mathbf{x} - \mathbf{x}^T \Sigma_1^{-1} \boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T \Sigma_1^{-1} \mathbf{x} + \boldsymbol{\mu}_1^T \Sigma_1^{-1} \boldsymbol{\mu}_1 +$$

$$\mathbf{x}^T\Sigma_2^{-1}\mathbf{x} - \mathbf{x}^T\Sigma_2^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_2^T\Sigma_2^{-1}\mathbf{x} + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2]$$

$$\overset{[1]}{=} -\tfrac{1}{2}[\mathbf{x}^T\Sigma_1^{-1}\mathbf{x} - 2\boldsymbol{\mu}_1^T\Sigma_1^{-1}\mathbf{x} + \boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \mathbf{x}^T\Sigma_2^{-1}\mathbf{x} - 2\boldsymbol{\mu}_2^T\Sigma_2^{-1}\mathbf{x} + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2]$$

$$= -\tfrac{1}{2}[\mathbf{x}^T(\Sigma_1^{-1} + \Sigma_2^{-1})\mathbf{x} - 2(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})\mathbf{x} + (\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2)]$$

$$= -\tfrac{1}{2}[\mathbf{x}^T(\Sigma_1^{-1} + \Sigma_2^{-1})\mathbf{x} - 2(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})\mathbf{x} + (\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2)$$

$$\pm(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right)] = (*)$$

Now, if we take $\Sigma_3^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1}$ and $\boldsymbol{\mu}_3 = \Sigma_3\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right)$ we get

$$(*) = -\tfrac{1}{2}[\mathbf{x}^T\Sigma_3^{-1}\mathbf{x} - 2\boldsymbol{\mu}_3^T\Sigma_3^{-1}\mathbf{x} + \boldsymbol{\mu}_3^T\Sigma_3^{-1}\boldsymbol{\mu}_3]$$

$$+\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1}+\Sigma_2^{-1})^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right) - \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2)$$

$$= -\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_3)^T\Sigma_3^{-1}(\mathbf{x} - \boldsymbol{\mu}_3) + C,$$

where previous constant C is given with

$$C = \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1}+\Sigma_2^{-1})^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right) - \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2).$$

In the step [1] we use the fact that when B is symmetric matrix then $\mathbf{a}^T B\mathbf{c} = \mathbf{c}^T B\mathbf{a}$. Matricies Σ_1^{-1} and Σ_2^{-1} are symmetric as inverse matricies of symmetric matricies Σ_1 and Σ_2 (based on our assumption).

We have shown that the product of two normal distributions is again a normal distribution $\mathcal{N}(\mathbf{x}, \boldsymbol{\mu_3}, \Sigma_3) \propto \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_3})^T\Sigma_3^{-1}(\mathbf{x} - \boldsymbol{\mu_3})\right)$, i.e.

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_2, \Sigma_2) \propto \mathcal{N}(\mathbf{x}, \boldsymbol{\mu_3}, \Sigma_3) \tag{4.1}$$

where $\Sigma_3^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1}$ and $\boldsymbol{\mu}_3 = \Sigma_3\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right)$.

Now, we will find that

$$\int\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{y}, A\mathbf{x}, \Sigma_2)\, d\mathbf{x} \propto \mathcal{N}(\mathbf{y}, A\boldsymbol{\mu_1}, A\Sigma_1 A^T + \Sigma_2) \tag{4.2}$$

where

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1) \propto \exp\left(-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^\top\Sigma_1^{-1}(\mathbf{x} - \boldsymbol{\mu}_1)\right)$$

$$\mathcal{N}(\mathbf{y}, A\mathbf{x}, \Sigma_2) \propto \exp\left(-\tfrac{1}{2}(\mathbf{y} - A\mathbf{x})^\top\Sigma_2^{-1}(\mathbf{y} - A\mathbf{x})\right).$$

Firstly, we will show that $\mathcal{N}(\mathbf{y}, A\mathbf{x}, \Sigma_2) \propto \mathcal{N}(\mathbf{x}, A^{-1}\mathbf{y}, (A^T\Sigma_2^{-1}A)^{-1})$.

$$\begin{aligned}
\mathcal{N}(\mathbf{y}, A\mathbf{x}, \Sigma_2) &\propto \exp\left(-\frac{1}{2}(\mathbf{y}-A\mathbf{x})^\top\Sigma_2^{-1}(\mathbf{y}-A\mathbf{x})\right) \\
&= \exp\left(-\frac{1}{2}\left((-A)(\mathbf{x}-A^{-1}\mathbf{y})\right)^T\Sigma_2^{-1}\left((-A)(\mathbf{x}-A^{-1}\mathbf{y})\right)\right) \\
&= \exp\left(-\frac{1}{2}(\mathbf{x}-A^{-1}\mathbf{y})^TA^T\Sigma_2^{-1}A(\mathbf{x}-A^{-1}\mathbf{y})\right) \\
&\propto \mathcal{N}(\mathbf{x}, A^{-1}\mathbf{y}, (A^T\Sigma_2^{-1}A)^{-1})
\end{aligned}$$

Now, we have that

$$\int\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{y}, A\mathbf{x}, \Sigma_2)\,d\mathbf{x} \propto \int\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{x}, A^{-1}\mathbf{y}, (A^T\Sigma_2^{-1}A)^{-1})\,d\mathbf{x}$$

For simplicity during calculations we will use the notation $\mu_2 = A^{-1}\mathbf{y}$, $\Sigma_2 = (A^T\Sigma_2^{-1}A)^{-1}$ and we will revert to original notation after final calculations.

$$\begin{aligned}
&\int\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{x}, A^{-1}\mathbf{y}, (A^T\Sigma_2^{-1}A)^{-1})\,d\mathbf{x} \\
&= \int\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma_1)\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_2, \Sigma_2)\,d\mathbf{x} \\
&\propto \int\exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_3)^T\Sigma_3^{-1}(\mathbf{x}-\boldsymbol{\mu}_3)+C\right)\,d\mathbf{x} \\
&= \int\exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_3)^T\Sigma_3^{-1}(\mathbf{x}-\boldsymbol{\mu}_3)\right)\cdot\exp(C)\,d\mathbf{x} \\
&= \exp(C)\int\exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_3)^T\Sigma_3^{-1}(\mathbf{x}-\boldsymbol{\mu}_3)\right)\,d\mathbf{x} \\
&\overset{*}{=} \sqrt{\det(2\pi\Sigma_3)}\exp(C) \propto \exp(C)
\end{aligned}$$

We have denoted

$$C = \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1}+\Sigma_2^{-1})^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1+\Sigma_2^{-1}\boldsymbol{\mu}_2\right)-\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2),$$

and in the step * we have used that $\int\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)\,d\mathbf{x} = \sqrt{\det(2\pi\boldsymbol{\Sigma})}$.

Based on tips and tricks in [4] we aim to bring the expression $\exp(C)$ into the form of $\exp(-\frac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2}))$. First, it holds that

$$\begin{aligned}
&\exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\right). \\
&= \exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_1}-2\boldsymbol{\mu_1}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_2}+\boldsymbol{\mu_2}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_2})\right)
\end{aligned}$$

We will now focus on simplifying the exponent C of the exponential function.

$$C = \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1}+\Sigma_2^{-1})^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1+\Sigma_2^{-1}\boldsymbol{\mu}_2\right)-\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1+\boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2)$$

$$
\begin{aligned}
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})(\Sigma_1^{-1}(I + \Sigma_1\Sigma_2^{-1}))^{-1}\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right) \\
&\quad -\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})(I + \Sigma_1\Sigma_2^{-1})^{-1}\Sigma_1\left(\Sigma_1^{-1}\boldsymbol{\mu}_1 + \Sigma_2^{-1}\boldsymbol{\mu}_2\right) \\
&\quad -\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1} + \boldsymbol{\mu}_2^T\Sigma_2^{-1})(I + \Sigma_1\Sigma_2^{-1})^{-1}\Sigma_1\Sigma_1^{-1}\left(\boldsymbol{\mu}_1 + \Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2\right) \\
&\quad -\tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})((\Sigma_2+\Sigma_1)\Sigma_2^{-1})^{-1}\left(\boldsymbol{\mu}_1 + \Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2\right) - \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}+\boldsymbol{\mu}_2^T\Sigma_2^{-1})\Sigma_2(\Sigma_2+\Sigma_1)^{-1}\left(\boldsymbol{\mu}_1 + \Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2\right) - \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\Sigma_2+\boldsymbol{\mu}_2^T)\Sigma_2^{-1}\Sigma_2(\Sigma_2+\Sigma_1)^{-1}\left(\boldsymbol{\mu}_1 + \Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2\right) - \tfrac{1}{2}(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= -\tfrac{1}{2}[-(\boldsymbol{\mu}_1^T\Sigma_1^{-1}\Sigma_2+\boldsymbol{\mu}_2^T)(\Sigma_2+\Sigma_1)^{-1}\left(\boldsymbol{\mu}_1 + \Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2\right) + (\boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2) \\
&= -\tfrac{1}{2}[-\boldsymbol{\mu}_1^T\Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T\Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2 \\
&\quad -\boldsymbol{\mu}_2^T(\Sigma_2 + \Sigma_1)^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^T(\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1}\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1^T\Sigma_1^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T\Sigma_2^{-1}\boldsymbol{\mu}_2] \\
&= -\tfrac{1}{2}[\boldsymbol{\mu}_1^T(\Sigma_1^{-1}-\Sigma_1^{-1}\Sigma_2(\Sigma_2+\Sigma_1)^{-1})\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T(\Sigma_2^{-1}\Sigma_1)^{-1}(\Sigma_2+\Sigma_1)^{-1}(\Sigma_2\Sigma_1^{-1})^{-1}\boldsymbol{\mu}_2 \\
&\quad -\boldsymbol{\mu}_2^T(\Sigma_2 + \Sigma_1)^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T(\Sigma_2^{-1} - (\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1})\boldsymbol{\mu}_2] \\
&= -\tfrac{1}{2}[\boldsymbol{\mu}_1^T(\Sigma_1^{-1} - \Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1})\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T(\Sigma_2\Sigma_1^{-1}(\Sigma_2 + \Sigma_1)\Sigma_2^{-1}\Sigma_1)^{-1}\boldsymbol{\mu}_2 \\
&\quad -\boldsymbol{\mu}_2^T(\Sigma_2 + \Sigma_1)^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T(\Sigma_2^{-1} - (\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1})\boldsymbol{\mu}_2] \\
&= -\tfrac{1}{2}[\boldsymbol{\mu}_1^T(\Sigma_1^{-1} - \Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1})\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1^T(\Sigma_1 + \Sigma_2)^{-1}\boldsymbol{\mu}_2 \\
&\quad -\boldsymbol{\mu}_2^T(\Sigma_2 + \Sigma_1)^{-1}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2^T(\Sigma_2^{-1} - (\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1})\boldsymbol{\mu}_2]
\end{aligned}
$$

By equivalent transformations we obtain $\Sigma_1^{-1} - \Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1} = (\Sigma_2 + \Sigma_1)^{-1}$.

$$
\begin{aligned}
(\Sigma_1^{-1} - \Sigma_1^{-1}\Sigma_2(\Sigma_2 + \Sigma_1)^{-1})(\Sigma_2 + \Sigma_1) &= (\Sigma_2 + \Sigma_1)^{-1}(\Sigma_2 + \Sigma_1) \\
\Sigma_1^{-1}(\Sigma_2 + \Sigma_1) - \Sigma_1^{-1}\Sigma_2 &= I \\
\Sigma_1^{-1}\Sigma_2 + I - \Sigma_1^{-1}\Sigma_2 &= I \\
I &= I
\end{aligned}
$$

Analogously it can be shown $(\Sigma_2^{-1} - (\Sigma_2 + \Sigma_1)^{-1}\Sigma_1\Sigma_2^{-1}) = (\Sigma_2 + \Sigma_1)^{-1}$.

It holds that

$$-\boldsymbol{\mu}_1^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu}_2-\boldsymbol{\mu}_2^T(\Sigma_2+\Sigma_1)^{-1}\boldsymbol{\mu}_1=-2\boldsymbol{\mu}_1^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu}_2,$$

because the sum of two symmetric matrices is a symmetric matrix and when B is a symmetric matrix then we know that $\mathbf{a}^TB\mathbf{c}=\mathbf{c}^TB\mathbf{a}$.

We have derived the desired form.

$$\begin{aligned}\exp(C)&=\exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_1}-2\boldsymbol{\mu_1}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_2}+\boldsymbol{\mu_2}^T(\Sigma_1+\Sigma_2)^{-1}\boldsymbol{\mu_2})\right)\\&=\exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\right)\end{aligned}$$

To finish our proof, we revert the original notation

$$\boldsymbol{\mu}_2=A^{-1}\mathbf{y},\quad \Sigma_2=(A^T\Sigma_2^{-1}A)^{-1}$$

and rearrange the expression to the desired form.

$$\begin{aligned}&\exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})^T(\Sigma_1+\Sigma_2)^{-1}(\boldsymbol{\mu_1}-\boldsymbol{\mu_2})\right)\\&=\exp\left(-\tfrac{1}{2}(\boldsymbol{\mu_1}-A^{-1}\mathbf{y})^T(\Sigma_1+(A^T\Sigma_2^{-1}A)^{-1})^{-1}(\boldsymbol{\mu_1}-A^{-1}\mathbf{y})\right)\\&=\exp\left(-\tfrac{1}{2}(-A^{-1}(\mathbf{y}-A\boldsymbol{\mu_1}))^T(\Sigma_1+A^{-1}\Sigma_2A^{-T})^{-1}(-A^{-1}(\mathbf{y}-A\boldsymbol{\mu_1}))\right)\\&=\exp\left(-\tfrac{1}{2}(\mathbf{y}-A\boldsymbol{\mu_1})^TA^{-T}(A^{-1}(A\Sigma_1A^T+\Sigma_2)A^{-T})^{-1}A^{-1}(\mathbf{y}-A\boldsymbol{\mu_1})\right)\\&=\exp\left(-\tfrac{1}{2}(\mathbf{y}-A\boldsymbol{\mu_1})^TA^{-T}A^T(A\Sigma_1A^T+\Sigma_2)^{-1}AA^{-1}(\mathbf{y}-A\boldsymbol{\mu_1})\right)\\&=\exp\left(-\tfrac{1}{2}(\mathbf{y}-A\boldsymbol{\mu_1})^T(A\Sigma_1A^T+\Sigma_2)^{-1}(\mathbf{y}-A\boldsymbol{\mu_1})\right)\\&\propto\mathcal{N}(\mathbf{y},A\boldsymbol{\mu_1},A\Sigma_1A^T+\Sigma_2)\end{aligned}$$

5 Kalman Filter

In this chapter, we will present the Kalman Filter (KF) relying on the information provided in [11] and [7]. The Kalman filter is a powerful algorithm used to estimate unknown variables given the measurements observed over time. It stands as one of the most significant advancements in statistical estimation theory, and arguably one of the greatest discoveries of the twentieth century. It is known for its simplicity and low computational requirements. Developed by Rudolf E. Kálmán[1] in the 1960s, this powerful mathematical tool is used in a wide range of applications for estimating the state of dynamic systems. Whether it is in engineering, finance, robotics, or navigation, the Kalman Filter offers an efficient way to make predictions and updates about a system's state based on noisy and uncertain measurements. The key strength of the Kalman Filter lies in its ability to provide accurate estimates based on incomplete and imprecise data. This makes it an essential tool for scenarios where real-time decision-making is crucial, such as tracking the position of a vehicle, predicting stock prices, or controlling the movement of a robot. Additionally, the Kalman filter is used for predicting future states of systems, making it useful in areas like meteorology, economics, and navigation. What makes the Kalman filter special is its ability to optimally combine previous predictions with new data, thereby reducing estimation errors. This algorithm operates in a recursive manner, meaning it continuously refines its estimates as new data becomes available. The fact that it supports estimations of past, present, and even future states, and it can be used even when the precise nature of the modeled system is unknown makes this filter very powerful.

Introduction to the Discrete Kalman Filter Algorithm

The Kalman filter deals with the challenge of estimating the state $x \in \mathbb{R}^n$ of a discrete-time controlled system that follows a linear stochastic difference equation

$$x_{k+1} = A_k x_k + B u_k + w_k \tag{5.1}$$

with a measurement $z_k \in \mathbb{R}^m$ that is

$$z_k = H_k x_k + v_k \tag{5.2}$$

Vector x_k represents the state vector at time k. It represents the current assessment of the system state (e.g. position, speed, orientation of the vehicle). The state transition matrix, denoted as A_k, is $n \times n$ matrix that describes how the current state x_k transitions

[1] Rudolf E. Kálmán (1930-2016) was a Hungarian-born American mathematician, electrical engineer, and inventor.

to the next state x_{k+1} without considering driving functions or process noise. Vector $u_k \in \mathbb{R}^l$ represents the control input vector at time k and $n \times l$ matrix B represents the control input matrix.Random variable that represents unpredictable disturbances or errors in the model that are not captured by the deterministic parts of the equation, w_k, is called process noise. Usually, it is assumed to be zero-mean Gaussian with the covariance Q_k, i.e., $w_k \sim \mathcal{N}(0, Q_k)$. Thus, w_k is added to the model to capture the uncertainties and inaccuracies in the system. The formula (5.1) describes how the state of the system changes from k to moment $k+1$ taking into account the previous state, control inputs, and process noise.

The second formula (5.2) describes the relationship between the state and the measurement at the current time step k. Vector z_k is denotes the measurement at the moment k. These are actual measurements obtained using sensors or some other data source in the system (e.g. GPS, radar, camera readings, etc.). Matrix that relates the current state of the system x_k to the measurement z_k, denoted with H_k, is $m \times n$ measurement matrix. It defines how the state of the system affects the measurements and can be different at each time point k. Random variable that represents measurement errors often caused by sensor imprefections or external disturbances, v_k, is the measurement noise vector. Usually, v_k is assumed to be white noise with zero mean and known covariance R_k, i.e. $v_k \sim \mathcal{N}(0, R_k)$.

The role of KF is to provide estimate x_k given the initial estimate of x_0, the series of measurement, $z_1, ..., z_k$, and the information of the system described by A_k, B, H_k, Q_k, and R_k.

Kalman Filter Algorithm

In the sequel, we define

- $\hat{x}_k^- \in \mathbb{R}^n$ to be our a priori state estimate at step k, also called predicted estimate,
- $\hat{x}_k \in \mathbb{R}^n$ to be a posteriori state estimate or updated estimate after incorporating the new measurement at time k.

Kalman filter algorithm consists of two stages: prediction (I) and update (II-IV) (see illustration 5.1).

Prediction stage:

$$\hat{x}_k^- = A_{k-1}\hat{x}_{k-1} + Bu_{k-1} \tag{5.3}$$

$$P_k^- = A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1} \tag{5.4}$$

The predicted state estimate is derived from the previously updated state estimate. The a priori estimate error is $x_k^- - \hat{x}_k^-$ and the a priori estimate error covariance is then $P_k^- = \mathbb{E}[(x_k^- - \hat{x}_k^-)(x_k^- - \hat{x}_k^-)^T]$. An uncertainty in the Kalman filter's prediction before receiving a new measurement is quantified by P_k^- . A smaller P_k^- suggests that the filter has greater confidence in its prediction, indicating reliable model assumptions

and previous estimates. Conversely, a larger P_k^-, implies higher uncertainty, leading the filter to rely more on the new measurement to correct the state estimate. The matrix P_k^- is a crucial factor in determining the Kalman gain K_k and is computed using the previous state covariance P_{k-1} and the process noise covariance Q_{k-1}.

Update stage:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{5.5}$$

$$\hat{x}_k = \hat{x}_k^- + K_K(z_k - H_k \hat{x}_k^-) \tag{5.6}$$

$$P_k = (I - K_k H_k) P_k^- \tag{5.7}$$

The first task during the measurement update is to compute the Kalman gain. The matrix K_k is chosen to be the gain or blending factor that minimizes the a posteriori error covariance P_k.

The diferenece $(z_k - H_k \hat{x}_k^-)$, i.e., the weighted differenece between an actual measurement z_k and a measurement prediction $H_k \hat{x}_k^-$, is called the measurement innovation or residual. It quantifies how much the new measurement deviates from what was expected based on the current state estimate. A large innovation indicates that the predicted state may be inaccurate, whereas a small innovation suggests that the predicted state is close to the actual state. For example, residual zero means that two are in complete agreement. The residual is then multiplied by the Kalman gain, K_k, to apply a correction, $K_k(z_k - H_k \hat{x}_k^-)$, to the predicted estimate $\hat{x}_k^-$. After obtaining the updated state estimate, the Kalman filter calculates the updated error covariance P_k, which will be used in the next step. The a posteriori covariance error P_k reflects the reduced uncertainty in this updated estimate. It is influenced by the Kalman gain K_k, which determines how much the measurement z_k affects the updated state estimate. Notably, the updated error covariance is smaller than the predicted error covariance, indicating that the filter is more confident in the state estimate after incorporating the measurement in the update stage.

The a posteriori state estimate error is the difference between the true state x_k and the posteriori state estimate $\hat{x}_k$, i.e. $x_k - \hat{x}_k$. The a posteriori state error covariance is then $\mathbb{E}[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$. It holds that the result P_k of (5.7) is exatctly what the expected error value $\mathbb{E}[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$ describes, i.e. error covariance after state update.

It holds that $p(x_k|z_k) \sim \mathcal{N}(\mathbb{E}[(x_k)], \mathbb{E}[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]) = \mathcal{N}(\hat{x}_k, P_k)$, i.e. Kalman Filter will give us $x_k \sim \mathcal{N}(\hat{x}_k, P_k)$.

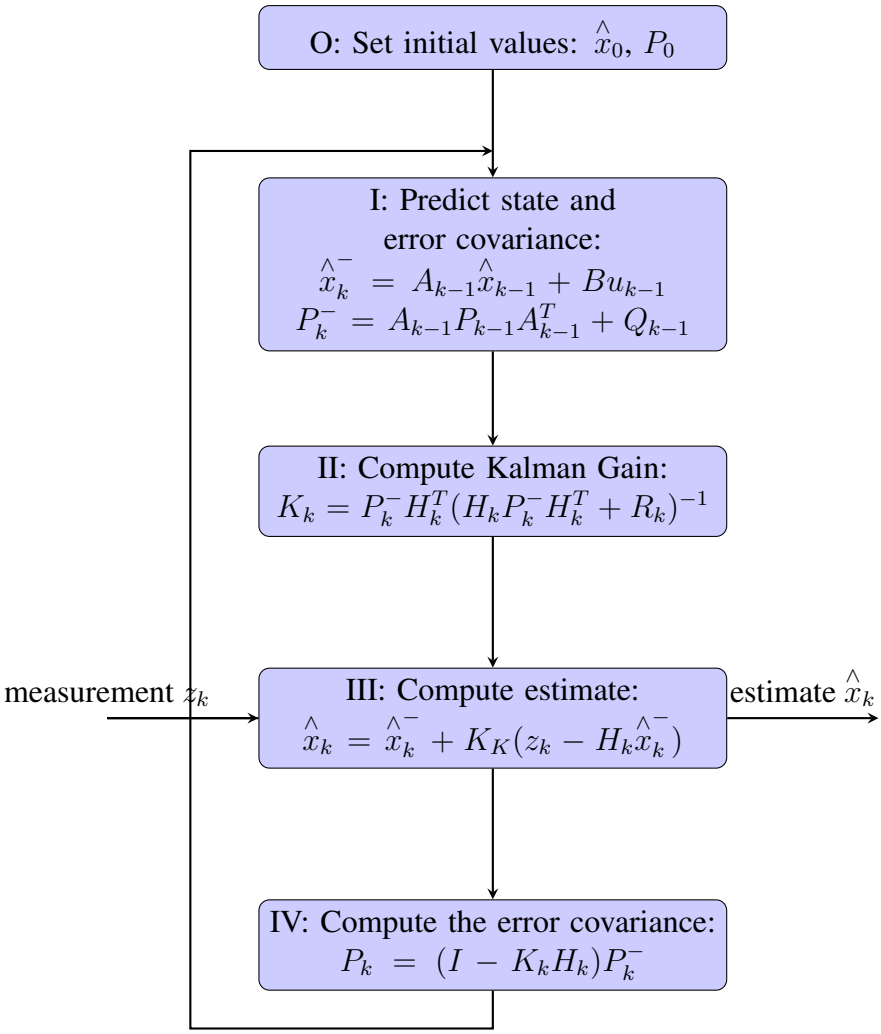


Figure 5.1: Ilusttration of the algorithm

6 Kalman Filtering using Belief Propagation

Based on [9] we derive Kalman Filter as an instance of the sum-product algorithm operating in the factor graph corresponding to a discrete-time linear dynamical system

$$\mathbf{x}_{j+1} = A_j \mathbf{x}_j + B\mathbf{u}_j + \mathbf{w}_j$$

$$\mathbf{y}_j = C_j \mathbf{x}_j + \mathbf{v}_j$$

with $\mathbf{x}_j, \mathbf{y}_j, \mathbf{u}_j$, $\mathbf{w}_j$, $\mathbf{v}_j$ denoting the time-j state, output, input, and two noise (two-dimensional) vector variables, respectively. Further, A_j, B, and C_j are assumed to be known time-varying matrices of appropriate dimensions. In the sequel, we will neglect the input part. We assume that w_j and v_j are independent Gaussian noise sequences with zero mean.

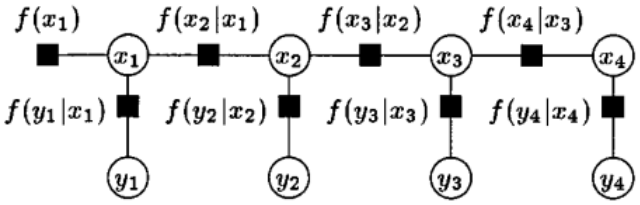


Figure 6.1: A hidden Markov model. Taken from [9].

The Markov structure of the system Figure 6.1 permits us to write the conditional joint probability density function of the state variables $\mathbf{x_1}, ..., \mathbf{x_k}$ given $\mathbf{y_1}, ..., \mathbf{y_k}$ as

$$f(\mathbf{x_1}, ..., \mathbf{x_k}|\mathbf{y_1}, ..., \mathbf{y_k}) = \prod_{j=1}^{k} f(\mathbf{x}_j|\mathbf{x}_{j-1}) f(\mathbf{y}_j|\mathbf{x}_j), \tag{6.1}$$

where $f(\mathbf{x}_j|\mathbf{x}_{j-1})$ is a Gaussian density with mean $A_j\mathbf{x}_{j-1}$ and variance Q_j (process noise covariance) and $f(\mathbf{y}_j|\mathbf{x}_j)$ is a Gaussian density with mean $C_j\mathbf{x}_j$ and variance R_j (measurement noise covariance).

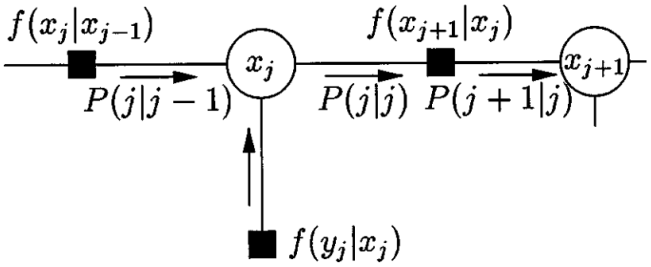


Figure 6.2: A portion of the factor graph corresponding to (6.1). Taken from [9].

In Figure 6.2 is shown a portion of the factor graph and messages that are passed in the operation of the sum-product algorithm. With $P_{j|j-1}(\mathbf{x}_j)$ we denote the message passed to $\mathbf{x}_j$ from $f(\mathbf{x}_j|\mathbf{x}_{j-1})$. This message is always of the form $\mathcal{N}(\mathbf{x}_j, \mathbf{m}_{j|j-1}, \Sigma_{j|j-1})$. Vector $\mathbf{m}_{j|j-1}$ is interpreted as the MMSE (Minimal Mean Square Error) prediction of the $\mathbf{x}_j$ given the set of observations up to time $j-1$. So, $P_{j|j-1}(\mathbf{x}_j)$ represents the prediction based on information up to time $j-1$, before the new measurement is taken into account, i.e. represents the prior probability estimate for the state $\mathbf{x}_j$ at time j.

According to the product rule, applying (4.1), we have:

$$P_{j|j}(\mathbf{x}_j) = P_{j|j-1}(\mathbf{x}_j) \cdot f(\mathbf{y}_j|\mathbf{x}_j)$$

$$= \mathcal{N}(\mathbf{x}_j, \mathbf{m}_{j|j-1}, \Sigma_{j|j-1})\mathcal{N}(\mathbf{y}_j, C_j\mathbf{x}_j, R_j)$$

$$\propto \mathcal{N}(\mathbf{x}_j, \mathbf{m}_{j|j-1}, \Sigma_{j|j-1})\mathcal{N}(\mathbf{x}_j, C_j^{-1}\mathbf{y}_j, C_j^{-1}R_jC_j^{-T})$$

$$\propto \mathcal{N}(\mathbf{x}_j, \mathbf{m}_{j|j}, \Sigma_{j|j})$$

where

$$\Sigma_{j|j} = (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} \tag{6.2}$$

$$\mathbf{m}_{j|j} = \Sigma_{j|j}(\Sigma_{j|j-1}^{-1}\mathbf{m}_{j|j-1} + C_j^T R_j^{-1}\mathbf{y}_j) \tag{6.3}$$

.
The posterior probability estimate for the state $\mathbf{x}_j$ at time j is given by $P_{j|j}(\mathbf{x}_j)$.

From (6.2) we can express $\Sigma_{j|j-1}^{-1}$ as follows:

$$\Sigma_{j|j}(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j) = I$$

$$\Sigma_{j|j}\Sigma_{j|j-1}^{-1} = I - \Sigma_{j|j}C_j^T R_j^{-1} C_j$$

$$\Sigma_{j|j-1}^{-1} = \Sigma_{j|j}^{-1}(I - \Sigma_{j|j}C_j^T R_j^{-1} C_j)$$

$$\Sigma_{j|j-1}^{-1} = \Sigma_{j|j}^{-1} - C_j^T R_j^{-1} C_j$$

This can be substituted into (6.3).

$$\mathbf{m}_{j|j} = \Sigma_{j|j}((\Sigma_{j|j}^{-1} - C_j^T R_j^{-1} C_j)\mathbf{m}_{j|j-1} + C_j^T R_j^{-1}\mathbf{y}_j)$$

$$\mathbf{m}_{j|j} = \Sigma_{j|j}\Sigma_{j|j}^{-1}\mathbf{m}_{j|j-1} - \Sigma_{j|j}C_j^T R_j^{-1} C_j\mathbf{m}_{j|j-1} + \Sigma_{j|j}C_j^T R_j^{-1}\mathbf{y}_j$$

$$\mathbf{m}_{j|j} = \mathbf{m}_{j|j-1} + \Sigma_{j|j}C_j^T R_j^{-1}(\mathbf{y}_j - C_j\mathbf{m}_{j|j-1})$$

$$\mathbf{m}_{j|j} = \mathbf{m}_{j|j-1} + (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1}(\mathbf{y}_j - C_j\mathbf{m}_{j|j-1})$$

In the last equation, the value $K_j = (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1}$ is Kalman gain.

$$(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1}$$

$$= (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1}(R_j C_j^{-T})^{-1}$$

$$= (R_j C_j^{-T}(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j))^{-1}$$

$$= (R_j C_j^{-T}\Sigma_{j|j-1}^{-1} + C_j)^{-1}$$

$$= ((R_j C_j^{-T} + C_j\Sigma_{j|j-1})\Sigma_{j|j-1}^{-1})^{-1}$$

$$= \Sigma_{j|j-1}C_j^T(R_j + C_j\Sigma_{j|j-1}C_j^T)^{-1}$$

Which perfectly matches with (5.5).

Now, by applying elementary transformations, we will show that the right-hand side of equations (6.2) and (5.7) are equal.

$$(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} = (I - K_J C_j)\Sigma_{j|j-1}$$

$$(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} = (I - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j)\Sigma_{j|j-1}$$

$$(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} = \Sigma_{j|j-1} - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j\Sigma_{j|j-1}$$

$$I = (\Sigma_{j|j-1} - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j\Sigma_{j|j-1})(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)$$

$$I = I - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j + \Sigma_{j|j-1}C_j^T R_j^{-1} C_j$$

$$-(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j\Sigma_{j|j-1}C_j^T R_j^{-1} C_j$$

$$I = I + \Sigma_{j|j-1}C_j^T R_j^{-1} C_j - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j(I + \Sigma_{j|j-1}C_j^T R_j^{-1} C_j)$$

$$I = (I - (\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1} C_j^T R_j^{-1} C_j)(I + \Sigma_{j|j-1}C_j^T R_j^{-1} C_j)$$

$$\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j = ((\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j) - C_j^T R_j^{-1} C_j)(I + \Sigma_{j|j-1}C_j^T R_j^{-1} C_j)$$

$$\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j = \Sigma_{j|j-1}^{-1}(I + \Sigma_{j|j-1} C_j^T R_j^{-1} C_j)$$

$$\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j = \Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j$$

Likewise, applying (4.2), we have

$$P_{j+1|j}(\mathbf{x}_{j+1}) = \int P_{j|j}(\mathbf{x}_j)\mathcal{N}(\mathbf{x}_{j+1}, A_j\mathbf{x}_j, Q_j)\, d\mathbf{x}_j \propto \mathcal{N}(\mathbf{x}_{j+1}, \mathbf{m}_{j+1|j}, \Sigma_{j+1|j})$$

where

$$\mathbf{x}_{j+1|j} = A_j\mathbf{m}_{j|j} = A_j\mathbf{m}_{j|j-1} + A_j K_j(\mathbf{y}_j - C_j\mathbf{m}_{j|j-1})$$

$$\Sigma_{j+1|j} = A_j\Sigma_{j|j}A_j^T + Q_j = A_j(\Sigma_{j|j-1}^{-1} + C_j^T R_j^{-1} C_j)^{-1}A^T + Q_j$$

which correspond to (5.3) and (5.4), respectively.

7 Sensor Technologies and Implementation

7.1 GPS and Odometry

Over the years, people have used a variety of techniques to navigate across the globe. Traditionally, people relied on stars and landmarks to travel between different areas, while maps and compasses helped to keep people from getting lost. The Global Positioning System (GPS) is a component used to estimate location and has made it much easier for people to find their way around. It became fully functional in 1994. It was originally developed for military applications but was made available for public use and became widely adopted in civilian applications, including automobile navigation systems, (recreational orienteering, and inventory tracking for transportation companies). The GPS system consists of a network of 24 active satellites (and 8 spares) located nearly 20,000 km above the Earth's surface. Every satellite sends out distinct signals that can be traced and analyzed by a GPS receiver on Earth to pinpoint the exact location of the satellite. Even if you are in the desert, the ocean, or Antarctica, GPS receivers can determine your position anywhere on Earth 24 hours a day in all weather conditions (rain, fog, shine, etc.). Also, satellite signals can travel through most plastics and glass, but cannot penetrate through solid objects, such as wood, rock, or concrete. GPS receivers can be found anywhere, are reasonably priced, and come in a variety of sizes and forms. These days, GPS receivers are present in a huge range of devices, including computers, watches, phones, tablets, and automobiles. While GPS signals can theoretically be received almost anywhere on Earth, practical factors can affect the accuracy of GPS measurements. For instance, although fog itself does not directly interfere with GPS signals, other factors such as dense clouds, buildings, trees, or terrain that obstruct the line of sight to satellites can impact signal quality. Moreover, GPS signals pass through the atmosphere, and delays caused by the ionosphere and troposphere can lead to inaccuracies in position measurements. Thus, while GPS is available nearly everywhere, the accuracy of measurements can vary depending on environmental and atmospheric conditions. However, GPS alone can be imprecise, especially in urban areas with signal obstructions.

An odometry is a form of localization that uses data from sensors like encoders to derive an estimated position relative to a starting point. The process typically relies on measuring parameters such as wheel rotations, angular rates, or accelerations to estimate how the vehicle has moved. By integrating these measurements over time, odometry provides an estimation of the vehicle's trajectory and its relative changes in position and orientation. Odometry is used in various applications such as robotics, autonomous vehicles, and navigation systems.

Achieving precise and reliable vehicle localization is paramount for ensuring safety and efficiency. This is where sensors like GPS and odometry play crucial roles. In this chapter we refer to [3] and [8].

In our work, we will focus on GPS and odometry providing an overview of their functionalities and discussing their integration into our system. Although RADAR sensors are also relevant in the context of vehicle localization, they will not be included in our implementation, since their complexity exceeds the theme of this work. We will, however, briefly mention their role and potential benefits in the broader scope of sensor technologies. RADAR (RAdio Detection And Ranging) is used for detecting and tracking objects using radio waves. It operates by emitting radio waves and then measuring the time it takes for the waves to bounce back after hitting an object. This allows RADAR to determine the distance, direction, and speed of the objects. RADAR sensors are typically mounted on the vehicle and provide information about the range (distance), angle, and possibly the velocity of the detected vehicles. RADAR measurements can be incorporated as factors in factor graphs or probabilistic graphical models to improve the accuracy of vehicle localization. The RADAR measurements help constrain the estimation process by providing additional information about the relative positions and motions of the vehicles.

7.2 Implementation with Simulated Data

In this section we will implement an algorithm to estimate vehicle positions by simulating true positions, applying GPS and odometry measurements with noise, and using our equations.
We want to estimate position of vechile, $\mathbf{x}_k$, at state k, i.e. $\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix}$ where x_k and y_k are the vehicle coordinates at time k, i.e. x_k represents latitude and y_k longitude .

GPS measurements also provide two-dimensional position data, so the measurement vector $\mathbf{z}_{GPS,k}$ would be $\mathbf{z}_{GPS,k} = \begin{bmatrix} z_{gx,k} \\ z_{gy,k} \end{bmatrix}$, where $z_{gx,k}$ and $z_{gy,k}$ are the GPS measurements for x_k and y_k.
The measurement model for GPS can be represented as

$$\mathbf{z}_{GPS,k} = H_{GPS}\mathbf{x_k} + \mathbf{v}_{GPS,k}$$

where $H_{GPS} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $\mathbf{v}_{GPS,k}$ is the measurement noise with covariance matrix R_{GPS}.

Odometry itself can be nonlinear due to the complexity of measurement and calculation of the vehicle's trajectory. However, in the context of Kalman filters and other state estimation methods, it is often useful to linearize the problem to simplify analysis and computation. By using a linearized model for odometry we approximate the odometric

measurements in a linear form. If we assume we get estimates of traveled distance in x and y directions, the odometry measurement vector would be $\mathbf{z}_{O,k} = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$. The measurement model for odometry can be represented as

$$\mathbf{z}_{O,k} = H_O \mathbf{x_k} + \mathbf{v}_{O,k}$$

where $H_O = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $\mathbf{v}_{O,k}$ is the measurement noise with covariance matrix R_O.

Hence, we have

$$f(\mathbf{z}_{GPS,k}|\mathbf{x}_k) \propto \mathcal{N}(\mathbf{z}_{GPS,k}, H_{GPS}\mathbf{x}_k, R_{GPS}) \text{ and } f(\mathbf{z}_O|\mathbf{x}_k) \propto \mathcal{N}(\mathbf{z}_{O,k}, H_O\mathbf{x}_k, R_O)$$

Applying product rule we get

$$P_{k|k}(\mathbf{x}_k) = P_{k|k-1}(\mathbf{x}_k) f(\mathbf{z}_{GPS,k}|\mathbf{x}_k) f(\mathbf{z}_{O,k}|\mathbf{x}_k).$$

$$\propto \mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k-1}, \Sigma_{k|k-1}) \mathcal{N}(\mathbf{z}_{GPS,k}, H_{GPS}\mathbf{x}_k, R_{GPS}) \mathcal{N}(\mathbf{z}_{O,k}, H_O\mathbf{x}_k, R_O)$$

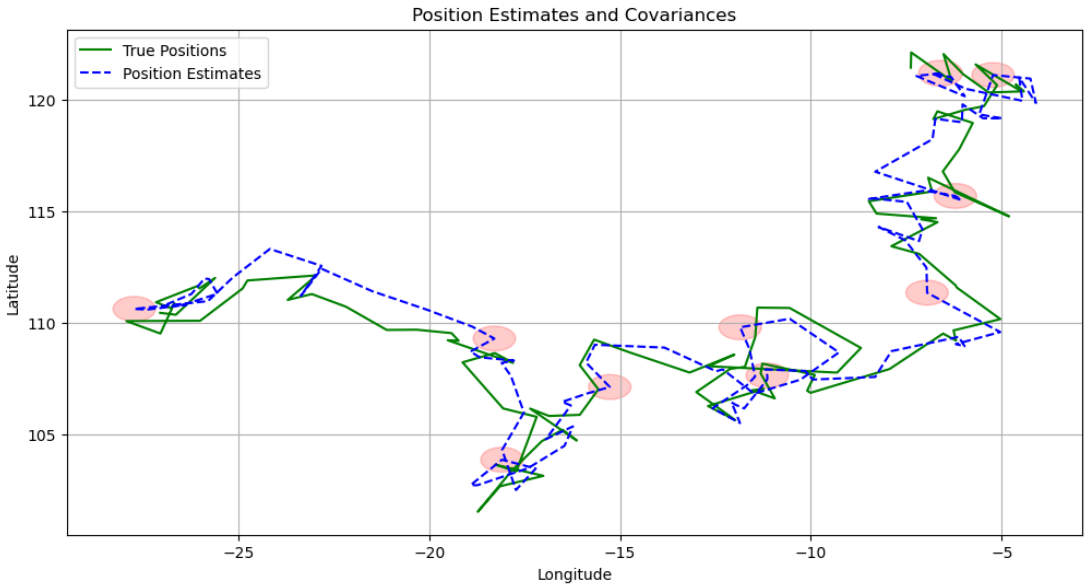
$$\propto \mathcal{N}(\mathbf{x}_k, \mathbf{m}_{k|k-1}, \Sigma_{k|k-1}) \mathcal{N}(\mathbf{x}_k, H_{GPS}^{-1}\mathbf{z}_{GPS,k}, (H_{GPS}^T R_{GPS}^{-1} H_{GPS})^{-1}) \mathcal{N}(\mathbf{x}_k, H_O^{-1}\mathbf{z}_{O,k}, (H_O^T R_O^{-1} H_O)^{-1})$$

where based on (4.1)

$$\Sigma_{k|k}^{-1} = \Sigma_{k|k-1}^{-1} + R_{GPS}^{-1} + R_O^{-1}$$

$$\mathbf{m}_{k|k} = \Sigma_{k|k}(\Sigma_{k|k-1}^{-1}\mathbf{m}_{k|k-1} + R_{GPS}^{-1}\mathbf{z}_{GPS,k} + R_O^{-1}\mathbf{z}_{O,k})$$

Numerical results



7.3 The Python code

```
import numpy as np
import matplotlib.pyplot as plt

# Simulation parameters
num_steps = 100
lat_min, lat_max = -90, 90
lon_min, lon_max = -180, 180
true_positions = np.zeros((num_steps, 2))  # Simulated true
    positions
estimates = np.zeros((num_steps, 2))  # Position estimates (mean)
covariances = np.zeros((num_steps, 2, 2))  # Covariances

# Model parameters
R_GPS = np.diag([1.0, 1.0])  # GPS measurement covariance
R_O = np.diag([1.0, 1.0])    # Odometry measurement covariance
Q = np.diag([0.5, 0.5])      # Process covariance
A = np.eye(2)                # Dynamic system model (identity
    assumption)
P = np.eye(2)                # Initial covariance

# Function to clamp latitude values
def clamp_latitude(lat):
    return np.clip(lat, lat_min, lat_max)

# Function to wrap longitude values
def wrap_longitude(lon):
    if lon > 180:
        return lon - 360
    elif lon < -180:
        return lon + 360
    return lon

# Simulate true positions
true_positions[0] = np.array([
    np.random.uniform(lat_min, lat_max),  # Latitude
    np.random.uniform(lon_min, lon_max)   # Longitude
]) # Initial position

for k in range(1, num_steps):
    delta_pos = np.random.normal(0, 1, 2)  # Change in position
    new_pos = true_positions[k-1] + delta_pos
    # Limit coordinates to geographical boundaries
    new_pos[0] = clamp_latitude(new_pos[0])
    new_pos[1] = wrap_longitude(new_pos[1])
    true_positions[k] = new_pos

# Set initial values
m_prior = true_positions[0]  # Initial estimate (mean)
P_prior = P  # Initial covariance

for k in range(num_steps):
    # Simulate measurements
    z_GPS = true_positions[k] + np.random.multivariate_normal([0,
```

```
        0], R_GPS)  # GPS measurement with noise
    z_O = true_positions[k] + np.random.multivariate_normal([0, 0],
        R_O)    # Odometry measurement with noise

    # A posteriori estimate
    S= np.linalg.inv(np.linalg.inv(P_prior) + np.linalg.inv(R_GPS) +
         np.linalg.inv(R_O))

    m_posterior = S @ (np.linalg.inv(P_prior) @ m_prior + np.linalg.
        inv(R_GPS) @ z_GPS + np.linalg.inv(R_O) @ z_O)
    P_posterior = S

    # Store results
    estimates[k] = m_posterior
    covariances[k] = P_posterior

    # A priori estimate for the next step
    m_prior = A @ m_posterior
    P_prior = A @ P_posterior @ A.T + Q

# Visualization
plt.figure(figsize=(12, 6))

# Plot true positions and estimates
plt.plot(true_positions[:, 0], true_positions[:, 1], 'g-', label='
    True Positions')
plt.plot(estimates[:, 0], estimates[:, 1], 'b--', label='Position
    Estimates')

# Plot uncertainty as ellipses
from matplotlib.patches import Ellipse
for i in range(0, len(true_positions), 10):  # Plot every 10 steps
    for clarity
    cov = covariances[i]
    eigenvalues, eigenvectors = np.linalg.eig(cov[:2, :2])
    angle = np.arctan2(*eigenvectors[:, 0][::-1])
    width, height = 2 * np.sqrt(eigenvalues)
    ellipse = Ellipse(xy=estimates[i], width=width, height=height,
        angle=np.degrees(angle), color='r', alpha=0.2)
    plt.gca().add_patch(ellipse)

plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.title('Position Estimates and Covariances')
plt.legend()
plt.grid(True)
plt.show()
```

8 Conclusion

Vehicle localization is crucial for numerous applications, ranging from autonomous driving to navigation systems. Accurate localization ensures that vehicles can determine their precise position in real-time, which is fundamental for safe and efficient operation. It enables advanced driver assistance systems, collision avoidance, and route optimization. Moreover, in the context of autonomous vehicles, robust localization is essential for effective decision-making and interaction with other vehicles and infrastructure. By improving localization accuracy through techniques like factor graphs and BP, we enhance the overall reliability and performance of vehicle systems.

Factor graphs allow complex functions to be represented as a product of simpler, local functions. This makes it easier to visualize and understand the connections between different parts of the problem and can be applied in a wide range of application areas. The sum-product algorithm, which is based on only a single conceptually simple computational rule, can encompass an enormous variety of practical algorithms, including KF. Based on the study presented in [9], which demonstrates KF as an instance of a sum-product algorithm in the one-dimensional case, we extend this framework to a two-dimensional scenario, considering vehicle position coordinates. Representing the Kalman Filter as an instance of Belief Propagation is important because it highlights how KF can be understood within a broader framework of probabilistic inference. This perspective reveals the underlying connections between different methods for state estimation and message propagation. It simplifies the integration of KF with other BP-based techniques and helps in adapting KF to more complex, non-linear scenarios. However, this framework serves as a basis that can be extended to handle non-linear cases and incorporate additional sensors, such as RADAR. In our work, we utilized normal distributions along with GPS and odometry measurements to establish a baseline approach.

Looking ahead, cooperative localization is a promising area for future research. Expanding our approach to include collaborative methods between multiple vehicles could enhance accuracy and reliability, addressing challenges posed by non-linear dynamics and complex vehicle interactions.

Bibliography

[1] $https : //airandspace.si.edu/collection - objects/1903 - wright - flyer/nasm_a19610048000$.

[2] $https : //group.mercedes - benz.com/company/tradition/company - history/1886 - 1920.html$.

[3] $https : //www.nasa.gov/directorates/somd/space - communications - navigation - program/gps/$.

[4] P. AHRENDT, *The Multivariate Gaussian Probability Distribution*, Danmarks Tekniske Universitet, 2005.

[5] C. M. BISHOP, *Pattern Recognition and Machine Learning*, Springer, 2006.

[6] F. DELLAERT AND M. KAESS, *Factor Graphs for Robot Perception*, 2017.

[7] M. GREWAL AND A. ANDREWS, *Kalman filtering: theory and practice using matlab*, New York: John Wiley and Sons, 14 (2001).

[8] D. GULATI, F. ZHANG, D. CLARKE, AND A. KNOLL, *Vehicle infrastructure cooperative localization using Factor Graphs*, in 2016 IEEE Intelligent Vehicles Symposium (IV), 2016, pp. 1085–1090.

[9] F. KSCHISCHANG, B. FREY, AND H.-A. LOELIGER, *Factor graphs and the sum-product algorithm*, IEEE Transactions on Information Theory, 47 (2001), pp. 498–519.

[10] D. RAJTER-ĆIRIĆ, *Verovatnoća, drugo dopunjeno izdanje*, Univerzitet u Novom Sadu, Prirodno-matematički fakultet u Novom Sadu, 2009.

[11] G. WELCH AND G. BISHOP, *An introduction to the Kalman filter*, Tech. Rep. 95-041, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

Biography

Katarina Vidojević was born on the 9th of May, 1997 in Loznica, Serbia. She attended elementary school “Anta Bogićević” and gymnasium “Vuk Karadžić” in Loznica. During her schooling, she participated in municipal, regional, and national competitions, as well as competitions organized by the Regional Center for Talented Youth Belgrade in the field of physics, which resulted in diplomas for achievements in that area. She received her Bachelor’s degree in Teaching Mathematics at the Faculty of Sciences, University of Novi Sad in 2021 with a GPA of 9,08. Due to her academic achievements, she received the City Award of Loznica for exceptional students. She enrolled with her Master’s studies in Data Science in 2021 at the same faculty and passed all exams in 2023 with a GPA of 9.44. During her master’s studies, in 2022, she found herself her first employment as a teaching associate at the Faculty of Technical Sciences, University of Novi Sad. She is a member of the research team for the PROMIS project “OPTIC” since 2024, funded by the Science Fund of the Republic of Serbia.

UNIVERZITET U NOVOM SADU
PRIRODNO–MATEMATIČKI FAKULTET
KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj:
RBR

Identifikacioni broj:
IBR

Tip dokumentacije: Monografska dokumentacija
TD

Tip zapisa: Tekstualni štampani materijal
TZ

Vrsta rada: Master rad
VR

Autor: Katarina Vidojević
AU

Mentor: Dr Dejan Vukobratović
MN

Naslov rada: Lokalizacija vozila upotrebom faktor grafova i Belief Propagation algoritma
NR

Jezik publikacije: Engleski
JP

Jezik izvoda: Engleski
JI

Zemlja publikovanja: Republika Srbija
ZP

Uže geografsko područje: Vojvodina
UGP

Godina: 2024.
GO

Izdavač: Autorski reprint
IZ

Mesto i adresa: Departman za matematiku i informatiku, Prirodno–matematički fakultet, Univerzitet u Novom Sadu, Trg Dositeja Obradovića 4, Novi Sad
MA

Fizički opis rada: 8 poglavlja, 38 strana, 11 citata, 9 slika
FO

Naučna oblast: Matematika
NO

Naučna disciplina: Primenjena matematika
ND

Ključne reči: faktor graf, Belief propagation algoritam, Kalman filter, normalna distribucija, lokalizacija vozila
KR

Univerzalna decimalna klasifikacija:
UDK:

Čuva se: Biblioteka Departmana za matematiku i informatiku, Prirodno–matematički fakultet, Univerzitet u Novom Sadu
ČU

Važna napomena:
VN

Izvod: Ovaj rad istražuje predstavljanje veze između Kalman filter (KF) i Belief Propagation (BP) algoritama u okviru probabilističkog zaključivanja. Na osnovu studije koja demonstrira KF kao Sum-product algoritam u jednodimenzionalnom slučaju, proširujemo ovaj pristup na višedimenzionalni scenario, uzimajući u obzir koordinate položaja vozila. Kroz primenu normalnih distribucija, GPS i odometrijskih merenja, razvijamo osnovu za procenu pozicije vozila. Ovaj okvir ne samo da pojednostavljuje razumevanje i integraciju KF-a sa drugim BP metodama, već omogućava i proširenje na nelinearne slučajeve i dodatne senzore, poput RADAR-a. U budućem radu, ovaj model bi mogao da se proširi na kooperativnu lokalizaciju vozila, čime bi se značajno unapredila preciznost i pouzdanost procene u sistemima sa više vozila i u složenijim dinamičkim uslovima.
IZ

Datum prihvatanja teme od strane NN veća:
DP

Datum odbrane:
DO

Članovi komisije:
KO

Predsednik:	Dr Nataša Krklec Jerinkić, redovni profesor, Prirodno-matematički fakultet, Novi Sad
Mentor:	Dr Dejan Vukobratović , redovni profesor, Fakultet tehničkih nauka, Novi Sad
Član:	Dr Mladen Kovačević, vandredni profesor, Fakultet tehničkih nauka, Novi Sad

UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCES
KEY WORDS DOCUMENTATION

Accession number:
ANO

Identification number:
INO

Document type: Monograph type
DT

Type of record: Printed text
TR

Contents Code: Master's thesis
CC

Author: Katarina Vidojević
AU

Mentor: Dr Dejan Vukobratović
MN

Title: Localization of the vehicles using Factor Graphs and Belief Propagation
TI

Language of text: English
LT

Language of abstract: English
LA

Country of publication: Republic of Serbia
CP

Locality of publication: Vojvodina
LP

Publication year: 2024.
PY

Publisher: Author's reprint
PU

Publ. place: Novi Sad, Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, Novi Sad
PP

Physical description: 8 chapters, 38 pages, 11 references, 9 figures
PD

Scientific field: Mathematics
SF

Scientific discipline: Applied mathematics
SD

Key words: factor graphs, Belief propagation, Kalman filter, normal distribution, localization of the vehicles
KW

Universal decimal classification:
UDC:

Holding data: The Library of the Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad
HD

Note:
N

Abstract: This paper explores the relationship between the Kalman Filter (KF) and the Belief Propagation (BP) algorithm within the framework of probabilistic inference. Based on a study that demonstrates KF as a Sum-product algorithm in the one-dimensional case, we extend this approach to a multi-dimensional scenario, taking into account vehicle position coordinates. By applying normal distributions, GPS, and odometry measurements, we develop a foundation for vehicle state estimation. This framework not only simplifies the understanding and integration of KF with other BP methods but also allows extension to non-linear cases and additional sensors, such as RADAR. In future work, this model could be expanded to cooperative vehicle localization, significantly improving the accuracy and reliability of estimation in systems with multiple vehicles and more complex dynamic conditions.
AB

Accepted by the Scientific Board on:
ASB

Defended:
DE

Thesis defend board:
DB

Chairperson:	Dr Nataša Krklec Jerinkić, full professor, Faculty of Sciences, University of Novi Sad
Mentor:	Dr Dejan Vukobratović, full professor, Faculty of Technical Sciences, University of Novi Sad
Member:	Dr Mladen Kovačević, associate professor, Faculty of Technical Sciences, University of Novi Sad