KALMAN FILTER

from the Ground Up

Alex Becker

First Edition

Revision history for the first edition

2023-05-01 First Release

2023-05-08 Minor Typo Updates

ISBN 978-965-598-439-2

KALMANFILTER.NET

First edition, May 2023

The road to learning by precept is long, by example short and effective.

Lucius Annaeus Seneca
A philosopher of Ancient Rome

Preface

Introduction

The Kalman Filter algorithm is a powerful tool for estimating and predicting system states in the presence of uncertainty and is widely used as a fundamental component in applications such as target tracking, navigation, and control.

Although the Kalman Filter is a straightforward concept, many resources on the subject require extensive mathematical background and fail to provide practical examples and illustrations, making it more complicated than necessary.

Back in 2017, I created an online tutorial based on numerical examples and intuitive explanations to make the topic more accessible and understandable. The online tutorial provides introductory material covering the univariate (one-dimensional) and multivariate (multidimensional) Kalman Filters.

Over time, I have received many requests to include more advanced topics, such as non-linear Kalman Filters (Extended Kalman Filter and Unscented Kalman Filter), sensors fusion, and practical implementation guidelines.

Based on the material covered in the online tutorial, I authored the "***Kalman Filter from the Ground Up***" e-book.

The original online tutorial will remain available for free access on the KALMANFILTER.NET website. The e-book "Kalman Filter from the Ground Up" and the source code for the numerical examples can be purchased online.

The book takes the reader from the basics to the advanced topics, covering both theoretical concepts and practical applications. The writing style is intuitive, prioritizing clarity of ideas over mathematical rigor, and it approaches the topic from a philosophical perspective before delving into quantification.

The book contains many illustrative examples, including 14 fully solved numerical examples with performance plots and tables. Examples progress in a paced, logical manner and build upon each other.

The book also includes the necessary mathematical background, providing a solid foundation to expand your knowledge and help to overcome your math fears.

This book is the solution for those facing challenges with the Kalman Filter and the underlying math.

Upon finishing this book, you will be able to design, simulate, and evaluate the performance of the Kalman Filter.

The book includes four parts:

- **Part 1** serves as an introduction to the Kalman Filter, using eight numerical examples, and doesn't require any prior mathematical knowledge. You can call it "The Kalman Filter for Dummies," as it aims to provide an intuitive understanding and develop "Kalman Filter intuition." Upon completing Part 1, readers will thoroughly understand the Kalman Filter's concept and be able to design a univariate (one-dimensional) Kalman Filter.
  This part is available for free access!
- **Part 2** presents the Kalman Filter in matrix notation, covering the multivariate (multidimensional) Kalman Filter. It includes a mathematical derivation of Kalman Filter equations, dynamic systems modeling, and two numerical examples. This section is more advanced and requires basic knowledge of Linear Algebra (only matrix operations). Upon completion, readers will understand the math behind the Kalman Filter and be able to design a multivariate Kalman Filter.
  Most of this part is available for free access!
- **Part 3** is dedicated to the non-linear Kalman Filter, which is essential for mastering the Kalman Filter since most real-life systems are non-linear. This part begins with a problem statement and describes the differences between linear and non-linear systems. It includes derivation and examples of the most common non-linear filters: the Extended Kalman Filter and the Unscented Kalman Filter.
- **Part 4** contains practical guidelines for Kalman Filter implementation, including sensor fusion, variable measurement uncertainty, treatment of missing measurements, treatment of outliers, and the Kalman Filter design process.

About the author

My name is Alex Becker. I am from Israel. I am an engineer with over 20 years of experience in the wireless technologies field. As a part of my work, I had to deal with Kalman Filters, mainly for tracking applications.

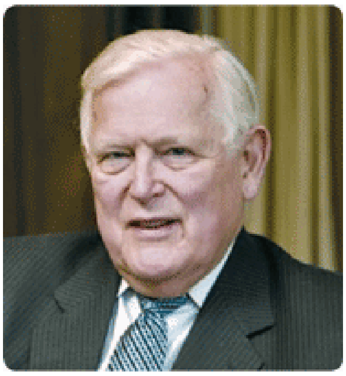
Constructive criticism is always welcome. I would greatly appreciate your comments and suggestions. Please drop me an email (alex@kalmanfilter.net).

R **The numerical examples in this book do not exemplify any modes, methodologies, techniques, or parameters employed by any operational system known to the author**.

About the Kalman Filter

Many modern systems utilize multiple sensors to estimate hidden (unknown) states through a series of measurements. For instance, a GPS receiver can estimate location and velocity, where location and velocity represent the hidden states, while the differential time of the arrival of signals from satellites serves as measurements.

One of the biggest challenges of tracking and control systems is providing an accurate and precise estimation of the hidden states in the presence of uncertainty. For example, GPS receivers are subject to measurement uncertainties influenced by external factors, such as thermal noise, atmospheric effects, slight changes in satellite positions, receiver clock precision, and more.



The Kalman Filter is a widely used estimation algorithm that plays a critical role in many fields. It is designed to estimate the hidden states of the system, even when the measurements are imprecise and uncertain. Also, the Kalman Filter predicts the future system state based on past estimations.

The filter is named after Rudolf E. Kálmán (May 19, 1930 – July 2, 2016). In 1960, Kálmán published his famous paper describing a recursive solution to the discrete-data linear filtering problem [1].

Contents

List of Figures

List of Tables

Acronyms

CPU Central Processing Unit
EKF Extended Kalman Filter
GNSS Global Navigation Satellite System
GPS Global Positioning System
INS Inertial Navigation System
KF Kalman Filter
LiDAR Light Detection and Ranging
LKF Linear Kalman Filter
LTI Linear Time Invariant
NASA National Aeronautics and Space Administration
PDF Probability Density Function
RMS Root Mean Square
RMSE Root Mean Square Error
SNR Signal to Noise Ratio
SPKF Sigma-point Kalman Filter
UAV Unmanned Air Vehicle
UKF Unscented Kalman Filter
UT Unscented Transform

Introduction to Kalman Filter

1. The Necessity of Prediction

Before delving into the Kalman Filter explanation, let us first understand the necessity of a tracking and prediction algorithm.

To illustrate this point, let's take the example of a tracking radar.

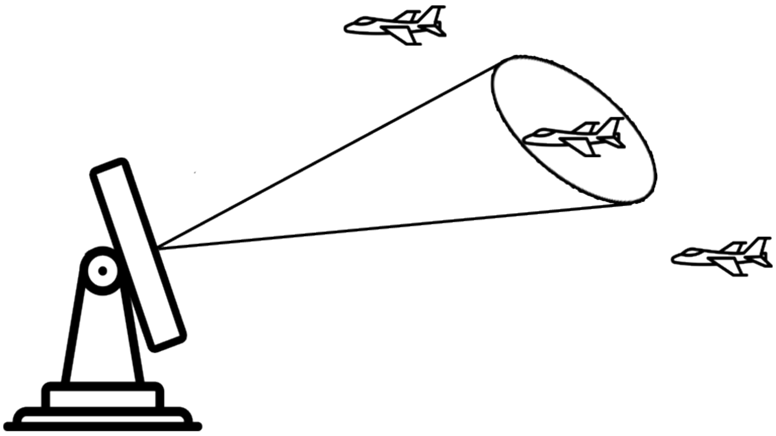


Figure 1.1: *Tracking radar.*

Suppose we have a track cycle of 5 seconds. At intervals of 5 seconds, the radar samples the target by directing a dedicated pencil beam.

Once the radar "visits" the target, it proceeds to estimate the current position and velocity of the target. The radar also estimates (or predicts) the target's position at the time of the next track beam.

The future target position can be easily calculated using Newton's motion equations:

$$x = x_0 + v_0 \Delta t + \frac{1}{2} a \Delta t^2 \tag{1.1}$$

Where:

x	is the target position
x_0	is the initial target position
v_0	is the initial target velocity
a	is the target acceleration
Δt	is the time interval (5 seconds in our example)

When dealing with three dimensions, Newton's motion equations can be expressed as a system of equations:

$$\begin{cases} x = x_0 + v_{x0}\Delta t + \dfrac{1}{2}\, a_x \Delta t^2 \\ y = y_0 + v_{y0}\Delta t + \dfrac{1}{2}\, a_y \Delta t^2 \\ z = z_0 + v_{z0}\Delta t + \dfrac{1}{2}\, a_z \Delta t^2 \end{cases} \tag{1.2}$$

The set of target parameters $[x, y, z, v_x, v_y, v_z, a_x, a_y, a_z]$ is known as the **System State**. The current state serves as the input for the prediction algorithm, while the algorithm's output is the future state, which includes the target parameters for the subsequent time interval.

The system of equations mentioned above is known as a **Dynamic Model** or **State Space Model**. The dynamic model describes the relationship between the input and output of the system.

Apparently, if the target's current state and dynamic model are known, predicting the target's subsequent state can be easily accomplished.

In reality, the radar measurement is not entirely accurate. It contains random errors or uncertainties that can affect the accuracy of the predicted target state. The magnitude of the errors depends on various factors, such as radar calibration, beam width, and signal-to-noise ratio of the returned echo. The random errors or uncertainties in the radar measurement are known as **Measurement Noise**.

In addition, the target motion is not always aligned with the motion equations due to external factors like wind, air turbulence, and pilot maneuvers. This misalignment between the motion equations and the actual target motion results in an error or uncertainty in the dynamic model, which is called **Process Noise**.

Due to the Measurement Noise and the Process Noise, the estimated target position can be far away from the actual target position. In this case, the radar might send the track beam in the wrong direction and miss the target.

In order to improve the radar's tracking accuracy, it is essential to employ a prediction algorithm that accounts for both process and measurement uncertainty.

The most common tracking and prediction algorithm is the **Kalman Filter**.

2. Essential background I

Before we start, I would like to explain several fundamental terms such as variance, standard deviation, normal distribution, estimate, accuracy, precision, mean, expected value, and random variable.

I expect that many readers of this book are familiar with introductory statistics. However, at the beginning of this book, I promised to supply the necessary background that is required to understand how the Kalman Filter works. If you are familiar with this topic, feel free to skip this chapter and jump to chapter 3.

2.1 Mean and Expected Value

Mean and **Expected Value** are closely related terms. However, there is a difference.

For example, given five coins – two 5-cent coins and three 10-cent coins, we can easily calculate the mean value by averaging the values of the coins.

Figure 2.1: *Coins.*

$$V_{mean} = \frac{1}{N}\sum_{n=1}^{N} V_n = \frac{1}{5}\left(5+5+10+10+10\right) = 8cent \tag{2.1}$$

The above outcome cannot be defined as the expected value because the system states (the coin values) are not hidden, and we've used the entire population (all 5 coins) for the mean value calculation.

Now assume five different weight measurements of the same person: 79.8kg, 80kg, 80.1kg, 79.8kg, and 80.2kg. The person is a system, and the person's weight is a system state.

Figure 2.2: *Man on scales.*

The measurements are different due to the random measurement error of the scales. We do not know the true value of the weight since it is a **Hidden State**. However, we can estimate the weight by averaging the scales' measurements.

$$W = \frac{1}{N}\sum_{n=1}^{N} W_n = \frac{1}{5}(79.8 + 80 + 80.1 + 79.8 + 80.2) = 79.98kg \tag{2.2}$$

The outcome of the estimate is the expected value of the weight.

The expected value is the value you would expect your hidden variable to have over a long time or many trials.

The mean is usually denoted by the Greek letter μ.

The letter E usually denotes the expected value.

2.2 Variance and Standard deviation

The **Variance** is a measure of the spreading of the data set from its mean.

The **Standard Deviation** is the square root of the variance.

The standard deviation is denoted by the Greek letter σ (sigma). Accordingly, the variance is denoted by σ^2.

Suppose we want to compare the heights of two high school basketball teams. The following table provides the players' heights and the mean height of each team.

	Player 1	**Player 2**	**Player 3**	**Player 4**	**Player 5**	**Mean**
Team A	1.89m	2.1m	1.75m	1.98m	1.85m	**1.914m**
Team B	1.94m	1.9m	1.97m	1.89m	1.87m	**1.914m**

Table 2.1: *Players' heights.*

As we can see, the mean height of both teams is the same. Let us examine the height variance.

Since the variance measures the spreading of the data set, we would like to know the data set deviation from its mean. We can calculate the distance from the mean for each variable by subtracting the mean from each variable.

The height is denoted by x, and the heights mean by the Greek letter μ. The distance from the mean for each variable would be:

$$x_n - \mu = x_n - 1.914m \tag{2.3}$$

The following table presents the distance from the mean for each variable.

	Player 1	**Player 2**	**Player 3**	**Player 4**	**Player 5**
Team A	-0.024m	0.186m	-0.164m	0.066m	-0.064m
Team B	0.026m	-0.014m	0.056m	-0.024m	-0.044m

Table 2.2: *Distance from the mean.*

Some of the values are negative. To get rid of the negative values, let us square the distance from the mean:

$$(x_n - \mu)^2 = (x_n - 1.914m)^2 \tag{2.4}$$

The following table presents the squared distance from the mean for each variable.

	Player 1	Player 2	Player 3	Player 4	Player 5
Team A	0.000576m^2	0.034596m^2	0.026896m^2	0.004356m^2	0.004096m^2
Team B	0.000676m^2	0.000196m^2	0.003136m^2	0.000576m^2	0.001936m^2

Table 2.3: *Squared distance from the mean.*

To calculate the variance of the data set, we need to find the average value of all squared distances from the mean:

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2 \tag{2.5}$$

For team A, the variance would be:

$$\sigma_A^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

$$= \frac{1}{5}(0.000576 + 0.034596 + 0.026896 + 0.004356 + 0.004096) = 0.014m^2$$

For team B, the variance would be:

$$\sigma_B^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

$$= \frac{1}{5}(0.000676 + 0.000196 + 0.003136 + 0.000576 + 0.001936) = 0.0013m^2$$

We can see that although the mean of both teams is the same, the measure of the height spreading of Team A is higher than the measure of the height spreading of Team B. Therefore, the Team A players are more diverse than the Team B players. There are players for different positions like ball handler, center, and guards, while the Team B players are not versatile.

The units of the variance are meters squared; it is more convenient to look at the standard deviation, which is a square root of the variance.

$$\sigma = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2} \tag{2.6}$$

- The standard deviation of Team A players' heights would be 0.12m.
- The standard deviation of Team B players' heights would be 0.036m.

Now, assume that we would like to calculate the mean and variance of all basketball players in all high schools. That would be an arduous task - we would need to collect data on every player from every high school.

On the other hand, we can estimate the players' mean and variance by picking a big data set and making the calculations on this data set.

The data set of 100 randomly selected players should be sufficient for an accurate estimation.

However, when we estimate the variance, the equation for the variance calculation is slightly different. Instead of normalizing by the factor N, we shall normalize by the factor $N-1$:

$$\sigma^2 = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \mu)^2 \tag{2.7}$$

The factor of $N-1$ is called Bessel's correction.

You can see the mathematical proof of the above equation on visiondummy or Wikipedia.

2.3 Normal Distribution

It turns out that many natural phenomena follow the **Normal Distribution**. The normal distribution, also known as the **Gaussian** (named after the mathematician Carl Friedrich Gauss), is described by the following equation:

$$f\left(x;\mu,\sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}}\,exp\left(\frac{-\left(x-\mu\right)^2}{2\sigma^2}\right) \tag{2.8}$$

The Gaussian curve is also called the **PDF (Probability Density Function)** for the normal distribution.

The following chart describes PDFs of the pizza delivery time in three cities: city 'A,' city 'B,' and city 'C.'

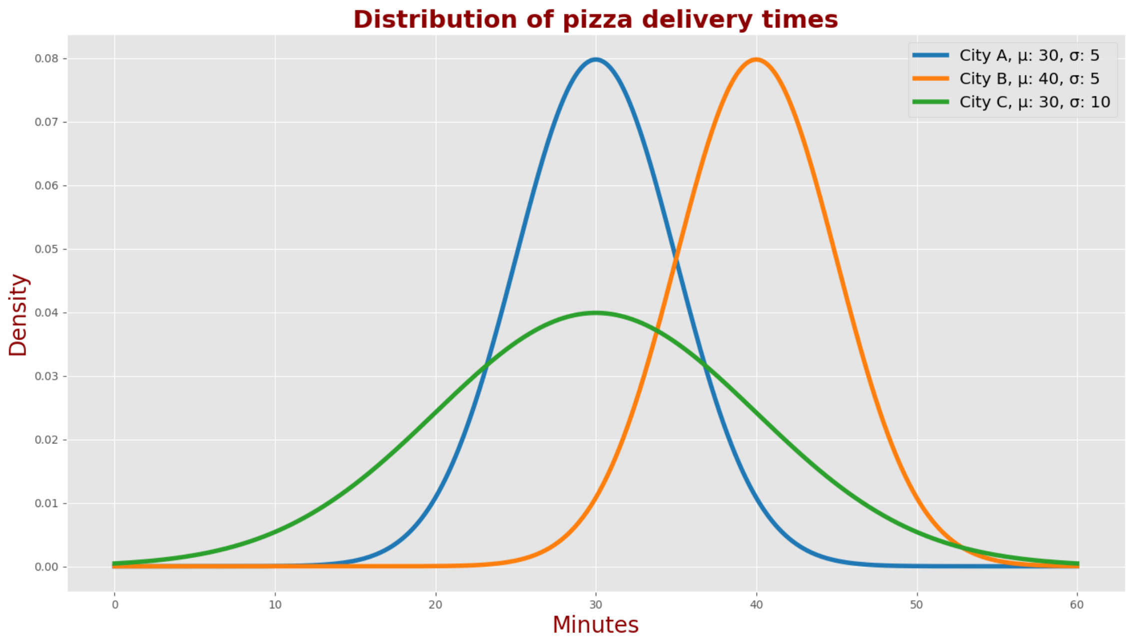


Figure 2.3: *Normal distribution PDFs.*

- In city 'A,' the mean delivery time is 30 minutes, and the standard deviation is 5 minutes.
- In city 'B,' the mean delivery time is 40 minutes, and the standard deviation is 5 minutes.
- In city 'C,' the mean delivery time is 30 minutes, and the standard deviation is 10 minutes.

We can see that the Gaussian shapes of the city 'A' and city 'B' pizza delivery times are identical; however, their centers are different. That means that in city 'A,' you wait for pizza for 10 minutes less on average, while the measure of spread in pizza delivery time is the same.

We can also see that the centers of Gaussians in the city 'A' and city 'C' are the same; however, their shapes are different. Therefore the average pizza delivery time in both cities is the same, but the measure of spread is different.

The following chart describes the proportions of the normal distribution.

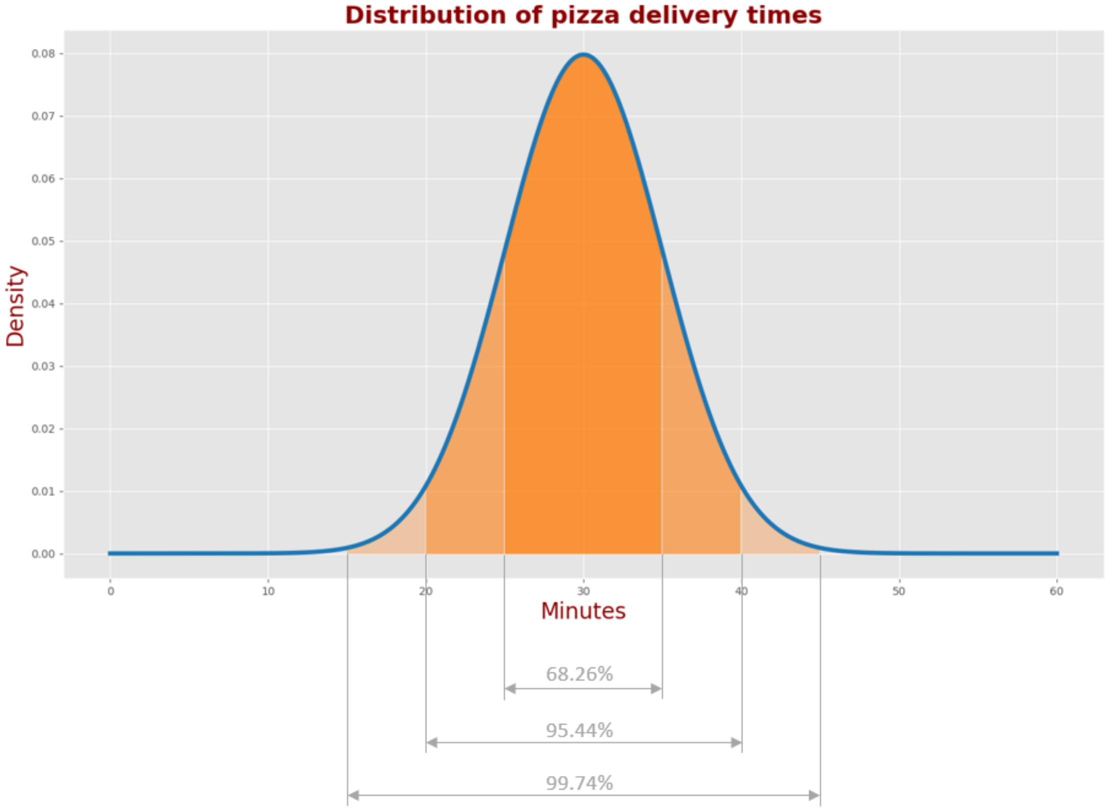


Figure 2.4: *Proportions of the normal distribution.*

- 68.26% of the pizza delivery times in City A lie within $\mu \pm \sigma$ range (25-35 minutes)
- 95.44% of the pizza delivery times in City A lie within $\mu \pm 2\sigma$ range (20-40 minutes)
- 99.74% of the pizza delivery times in City A lie within $\mu \pm 3\sigma$ range (15-45 minutes)

Usually, measurement errors are distributed normally. The Kalman Filter design assumes a normal distribution of the measurement errors.

2.4 Random Variables

A **random variable** describes the hidden state of the system. A random variable is a set of possible values from a random experiment.

The random variable can be continuous or discrete:

- A continuous random variable can take any value within a specific range, such as battery charge time or marathon race time.
- A discrete random variable is countable, such as the number of website visitors or the number of students in the class.

The random variable is described by the probability density function. The probability density function is characterized by **moments**.

The moments of the random value are expected values of powers of the random variable. We are interested in two types of moments:

- The k^{th} raw moment is the expected value of the k^{th} power of the random variable: $E\left(X^k\right)$.
- The k^{th} central moment is the expected value of the k^{th} power of the random variable distribution about its mean: $E\left((X-\mu_X)^k\right)$.

In this book, the random variables are characterized by the following:

- The first raw moment $E(X)$ – the mean of the sequence of measurements.
- The second central moment $E\left((X-\mu_X)^2\right)$ – the variance of the sequence of measurements.

2.5 Estimate, Accuracy and Precision

An **Estimate** is about evaluating the hidden state of the system. The true position of the aircraft is hidden from the observer. We can estimate the aircraft position using sensors, such as radar. The estimate can be significantly improved by using multiple sensors and applying advanced estimation and tracking algorithms (such as the Kalman Filter). Every measured or computed parameter is an estimate.

Accuracy indicates how close the measurement is to the true value.

Precision describes the variability in a series of measurements of the same parameter. Accuracy and precision form the basis of the estimate.

The following figure illustrates accuracy and precision.

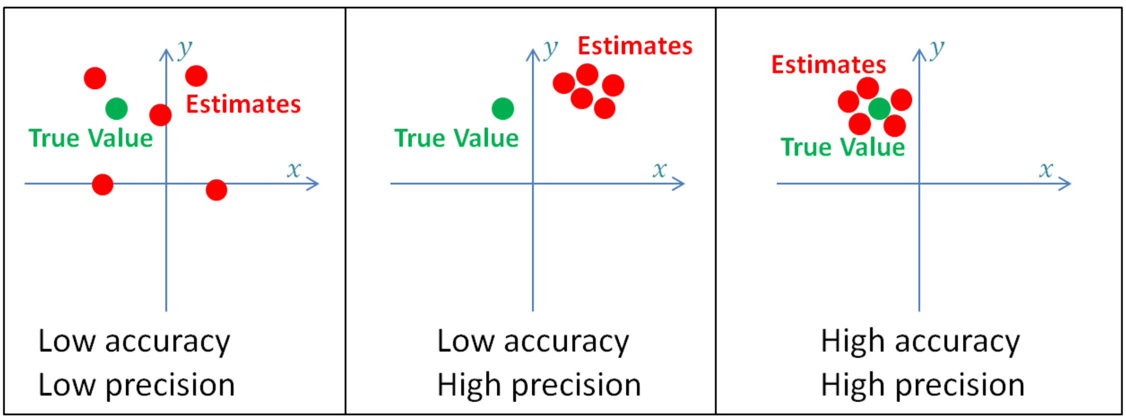


Figure 2.5: *Accuracy and Precision.*

High-precision systems have low variance in their measurements (i.e., low uncertainty), while low-precision systems have high variance in their measurements (i.e., high uncertainty). The random measurement error produces the variance.

Low-accuracy systems are called **biased** systems since their measurements have a built-in systematic error (bias).

The influence of the variance can be significantly reduced by averaging or smoothing measurements. For example, if we measure temperature using a thermometer with a random measurement error, we can make multiple measurements and average them. Since the error is random, some measurements would be above the true value and others below the true value. The estimate would be close to the true value. The more measurements we make, the closer the estimate will be.

On the other hand, a biased thermometer produces a constant systematic error in the estimate.

All examples in this book assume **unbiased** systems.

2.6 Summary

The following figure represents a statistical view of measurement.

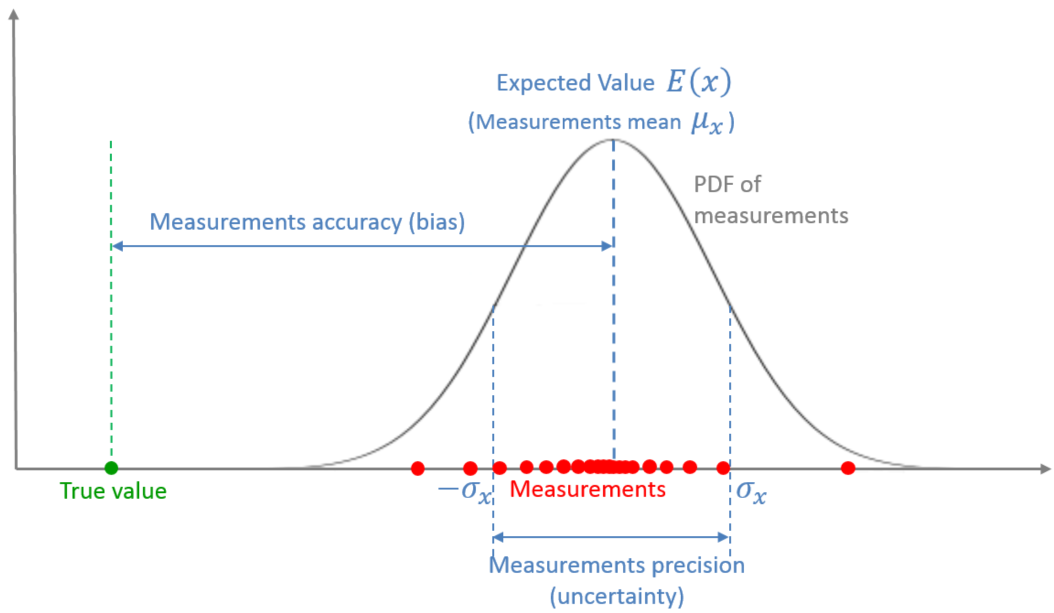


Figure 2.6: *Statistical view of measurement.*

A measurement is a **random variable** described by the **PDF**.

The mean of the measurements is the **Expected Value** of the random variable.

The offset between the mean of the measurements and the true value is the **accuracy of the measurements**, also known as **bias** or **systematic measurement error**.

The dispersion of the distribution is the measurement **precision**, also known as the **measurement noise**, **random measurement error**, or **measurement uncertainty**.

3. The $\alpha - \beta - \gamma$ filter

This chapter is introductory, and it describes the $\alpha - \beta$ and $\alpha - \beta - \gamma$ filters. These filters are frequently used for time series data smoothing. The principles of the $\alpha - \beta(-\gamma)$ filter are closely related to the Kalman Filter principles.

3.1 Example 1 – Weighting the gold

Now we are ready for the first simple example. In this example, we estimate the state of the static system. A static system is a system that doesn't change its state over a reasonable period. For instance, the static system could be a tower, and the state would be its height.

In this example, we estimate the weight of the gold bar. We have unbiased scales, i.e., the measurements don't have a systematic error, but the measurements do include random noise.

Figure 3.1: *Gold Bars.*

The system is the gold bar, and the system state is the weight of the gold bar. The dynamic model of the system is constant since we assume that the weight doesn't change over short periods.

To estimate the system state (i.e., the weight value), we can make multiple measurements and average them.

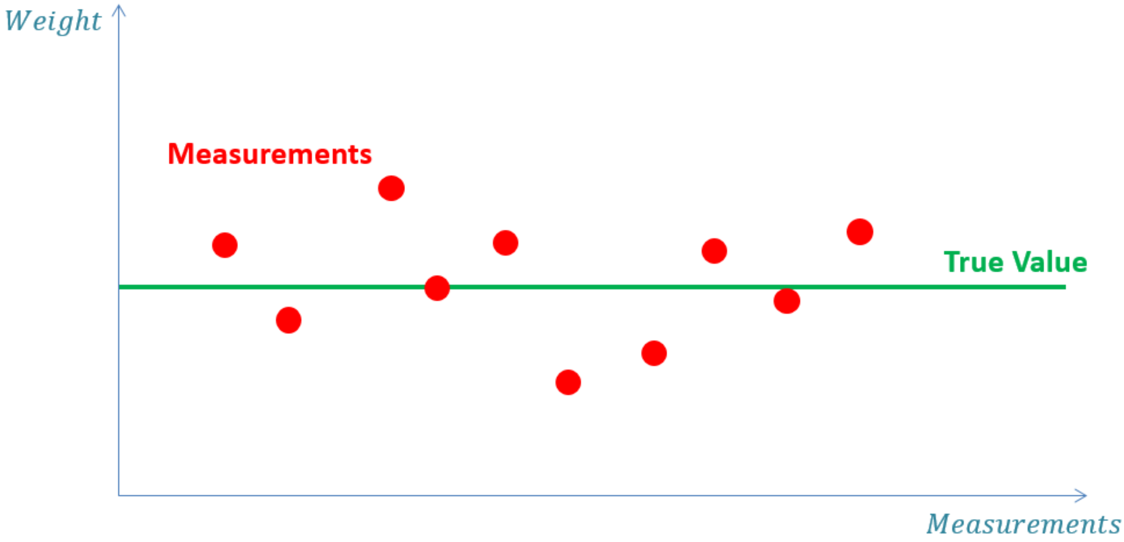


Figure 3.2: *Measurements vs. True value.*

At the time n, the estimate $\hat{x}_{n,n}$ would be the average of all previous measurements:

$$\hat{x}_{n,n} = \frac{1}{n}\left(z_1 + z_2 + \ldots + z_{n-1} + z_n\right) = \frac{1}{n}\sum_{i=1}^{n}\left(z_i\right) \tag{3.1}$$

Example Notation:

x	is the true value of the weight
z_n	is the measured value of the weight at time n
$\hat{x}_{n,n}$	is the estimate of x at time n (the estimate is made after taking the measurement z_n)
$\hat{x}_{n+1,n}$	is the estimate of the future state $(n+1)$ of x. The estimate is made at the time n. In other words, $\hat{x}_{n+1,n}$ is a predicted state or extrapolated state
$\hat{x}_{n-1,n-1}$	is the estimate of x at time $n-1$ (the estimate is made after taking the measurement z_{n-1})
$\hat{x}_{n,n-1}$	is a prior prediction - the estimate of the state at time n. The prediction is made at the time $n-1$

R In the literature, a caret (or hat) over a variable indicates an estimated value.

The dynamic model in this example is static (or constant) since the weight of gold doesn't change over time, therefore $\hat{x}_{n+1,n} = \hat{x}_{n,n}$.

Although the Equation 3.1 is mathematically correct, it is not practical for implementation. In order to estimate $\hat{x}_{n,n}$ we need to remember all historical measurements; therefore, we need a large memory. We also need to recalculate the average repeatedly if we want to update the estimated value after every new measurement. Thus, we need a more powerful Central Processing Unit (CPU).

It would be more practical to keep the last estimate only ($\hat{x}_{n-1,n-1}$) and update it after every new measurement. The following figure exemplifies the required algorithm:

- Estimate the current state based on the measurement and prior prediction.
- Predict the next state based on the current state estimate using the Dynamic Model.

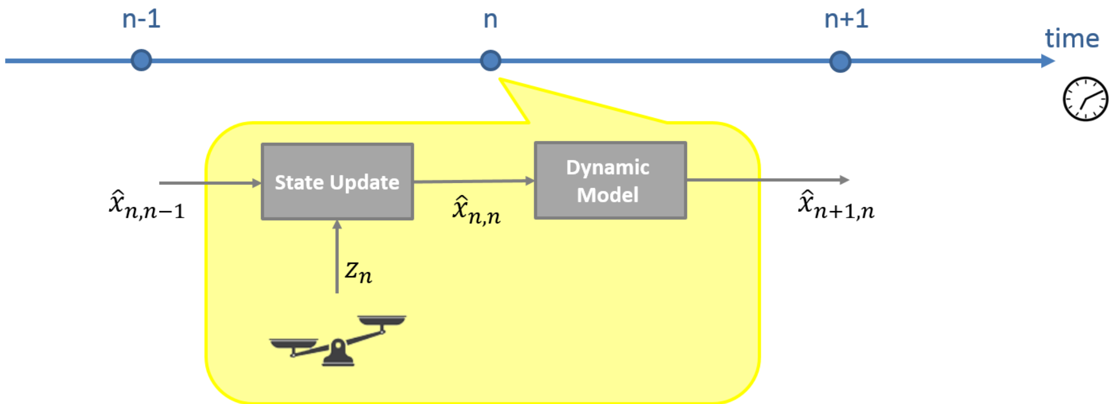


Figure 3.3: *Example Notation.*

We can modify the averaging equation for our needs using a small mathematical trick:

Equation	Notes
$\hat{x}_{n,n} = \frac{1}{n}\sum_{i=1}^{n}(z_i)$	Average formula: sum of n measurements divided by n
$= \frac{1}{n}\left(\sum_{i=1}^{n-1}(z_i) + z_n\right)$	Sum of the $n-1$ measurements plus the last measurement divided by n
$= \frac{1}{n}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$	Expand
$= \frac{1}{n}\frac{n-1}{n-1}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$	Multiply and divide by term $n-1$
$= \frac{n-1}{n}\frac{1}{n-1}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$	Reorder. The 'orange' term is the prior estimate
$= \frac{n-1}{n}\hat{x}_{n-1,n-1} + \frac{1}{n}z_n$	Rewriting the sum
$= \hat{x}_{n-1,n-1} - \frac{1}{n}\hat{x}_{n-1,n-1} + \frac{1}{n}z_n$	Distribute the term $\frac{n-1}{n}$
$= \hat{x}_{n-1,n-1} + \frac{1}{n}(z_n - \hat{x}_{n-1,n-1})$	Reorder

Table 3.1: *Averaging equation.*

$\hat{x}_{n-1,n-1}$ is the estimated state of x at the time $n-1$, based on the measurement at the time $n-1$.

Let's find $\hat{x}_{n,n-1}$ (the predicted state of x at the time n), based on $\hat{x}_{n-1,n-1}$ (the estimation at the time $n-1$). In other words, we would like to extrapolate $\hat{x}_{n-1,n-1}$ to the time n.

Since the dynamic model in this example is static, the predicted state of x equals the estimated state of x: $\hat{x}_{n,n-1} = \hat{x}_{n-1,n-1}$.

Based on the above, we can write the **State Update Equation**:

State Update Equation

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \frac{1}{n}\left(z_n - \hat{x}_{n,n-1}\right) \tag{3.2}$$

The State Update Equation is one of the five Kalman filter equations. It means the following:

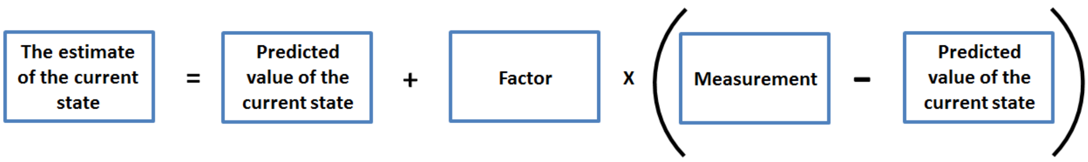


Figure 3.4: *State Update Equation.*

The factor $^1/_n$ is specific to our example. We will discuss the vital role of this factor later, but right now, I would like to note that in "Kalman Filter language," this factor is called the **Kalman Gain**. It is denoted by K_n. The subscript n indicates that the Kalman Gain can change with every iteration.

The discovery of K_n was one of Rudolf Kalman's significant contributions.

Before we get into the guts of the Kalman Filter, we use the Greek letter α_n instead of K_n.

So, the State Update Equation looks as follows:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \alpha_n\left(z_n - \hat{x}_{n,n-1}\right) \tag{3.3}$$

The term $(z_n - \hat{x}_{n,n-1})$ is the "measurement residual," also called **innovation**. The innovation contains new information.

In this example, $^1/_n$ decreases as n increases. In the beginning, we don't have enough information about the current state; thus, the first estimation is based on the first measurement $\frac{1}{n}|_{n=1} = 1$. As we continue, each successive measurement has less weight in the estimation process, since $^1/_n$ decreases. At some point, the contribution of the new measurements will become negligible.

Let's continue with the example. Before we make the first measurement, we can guess (or rough estimate) the gold bar weight simply by reading the stamp on the gold bar. It is called the **Initial Guess**, and it is our first estimate.

The Kalman Filter requires the initial guess as a preset, which can be very rough.

3.1.1 Estimation algorithm

The following chart depicts the estimation algorithm that is used in this example.

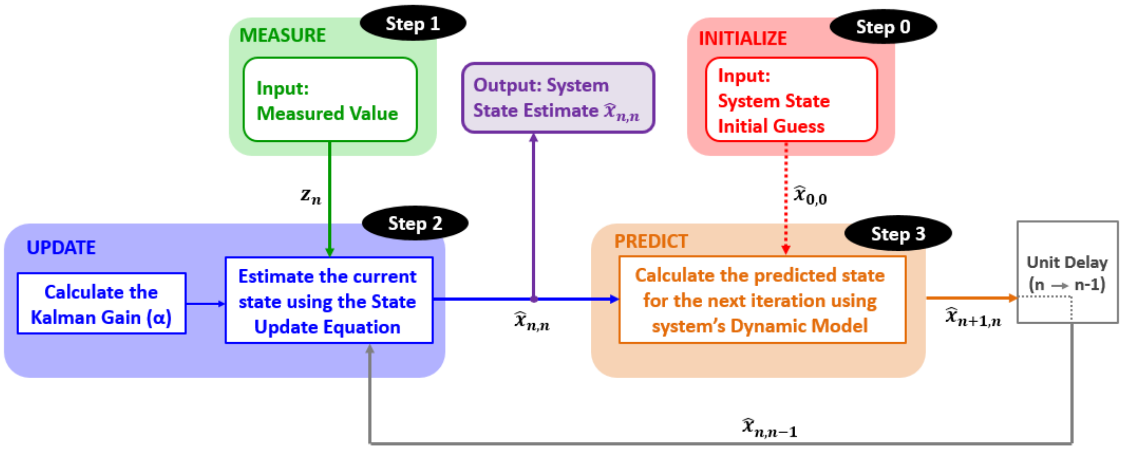


Figure 3.5: *State Update Equation.*

Now we are ready to start the measurement and estimation process.

3.1.2 The numerical example

3.1.2.1 Iteration Zero

Initialization

Our initial guess of the gold bar weight is 1000 grams. The initial guess is used only once for the filter initiation. Thus, it won't be required for successive iterations.

$\hat{x}_{0,0} = 1000g$

Prediction

The weight of the gold bar is not supposed to change. Therefore, the dynamic model of the system is static. Our next state estimate (prediction) equals the initialization:

$\hat{x}_{1,0} = \hat{x}_{0,0} = 1000g$

3.1.2.2 First Iteration

Step 1

Making the weight measurement with the scales:

$z_1 = 996g$

Step 2

Calculating the gain. In our example $\alpha_n = {}^1\!/_n$, thus:

$$\alpha_1 = \frac{1}{1} = 1$$

Calculating the current estimate using the State Update Equation:

$$\hat{x}_{1,1} = \hat{x}_{1,0} + \alpha_1 (z_1 - \hat{x}_{1,0}) = 1000 + 1\,(996 - 1000) = 996g$$

> **R** The initial guess could be any number in this specific example. Since $\alpha_1 = 1$, the initial guess is eliminated in the first iteration.

Step 3

The dynamic model of the system is static; thus, the weight of the gold bar is not supposed to change. Our next state estimate (prediction) equals to current state estimate:

$$\hat{x}_{2,1} = \hat{x}_{1,1} = 996g$$

3.1.2.3 Second Iteration

After a unit time delay, the **predicted estimate** from the previous iteration becomes the **prior estimate** in the current iteration:

$$\hat{x}_{2,1} = 996g$$

Step 1

Making the second measurement of the weight:

$$z_2 = 994g$$

Step 2

Calculating the gain:

$$\alpha_2 = \frac{1}{2}$$

Calculating the current estimate:

$$\hat{x}_{2,2} = \hat{x}_{2,1} + \alpha_2 (z_2 - \hat{x}_{2,1}) = 996 + \frac{1}{2}(994 - 996) = 995g$$

Step 3

$$\hat{x}_{3,2} = \hat{x}_{2,2} = 995g$$

3.1.2.4 Third Iteration

$$z_3 = 1021g$$

$$\alpha_3 = \frac{1}{3}$$

$$\hat{x}_{3,3} = \ 995 + \frac{1}{3}(1021 - 995) = 1003.67g$$

$$\hat{x}_{4,3} = 1003.67g$$

3.1.2.5 Fourth Iteration

$$z_4 = 1000g$$

$$\alpha_4 = \frac{1}{4}$$

$$\hat{x}_{4,4} = 1003.67 + \frac{1}{4}(1000 - 1003.67) = 1002.75g$$

$$\hat{x}_{5,4} = 1002.75g$$

3.1.2.6 Fifth Iteration

$$z_5 = 1002g$$

$$\alpha_5 = \frac{1}{5}$$

$$\hat{x}_{5,5} = 1002.75 + \frac{1}{5}(1002 - 1002.75) = 1002.6g$$

$$\hat{x}_{6,5} = 1002.6g$$

3.1.2.7 Sixth Iteration

$$z_6 = 1010g$$

$$\alpha_6 = \frac{1}{6}$$

$$\hat{x}_{6,6} = 1002.6 + \frac{1}{6}(1010 - 1002.6) = 1003.83$$

$$\hat{x}_{7,6} = 1003.83g$$

3.1.2.8 Seventh Iteration

$$z_7 = 983g$$

$$\alpha_7 = \frac{1}{7}$$

$$\hat{x}_{7,7} = 1003.83 + \frac{1}{7}(983 - 1003.83) = 1000.86g$$

$$\hat{x}_{8,7} = 1000.86g$$

3.1.2.9 Eighth Iteration

$$z_8 = 971g$$

$$\alpha_8 = \frac{1}{8}$$

$$\hat{x}_{8,8} = 1000.86 + \frac{1}{8}(971 - 1000.86) = 997.125g$$

$$\hat{x}_{9,8} = 997.125g$$

3.1.2.10 Ninth Iteration

$$z_9 = 993g$$

$$\alpha_9 = \frac{1}{9}$$

$$\hat{x}_{9,9} = 997.125 + \frac{1}{9}(993 - 997.125) = 996.67g$$

$$\hat{x}_{10,9} = 996.67g$$

3.1.2.11 Tenth Iteration

$$z_{10} = 1023g$$

$$\alpha_{10} = \frac{1}{10}$$

$$\hat{x}_{10,10} = 996.67 + \frac{1}{10}(1023 - 996.67) = 999.3g$$

$$\hat{x}_{11,10} = 999.3g$$

We can stop here. The gain decreases with each measurement. Therefore, the contribution of each successive measurement is lower than the contribution of the previous measurement. We get pretty close to the true weight, which is 1000g. If we were making more measurements, we would get closer to the true value.

The following table summarizes our measurements and estimates, and the chart compares the measured values, the estimates, and the true value.

n	1	2	3	4	5	6	7	8	9	10
α_n	1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}
z_n	996	994	1021	1000	1002	1010	983	971	993	1023
$\hat{x}_{n,n}$	996	995	1003.67	1002.75	1002.6	1003.83	1000.86	997.125	996.67	999.3
$\hat{x}_{n+1,n}$	996	995	1003.67	1002.75	1002.6	1003.83	1000.86	997.125	996.67	999.3

Table 3.2: *Example 1 summary.*

3.1.3 Results analysis

The following chart compares the true, measured, and estimated values.

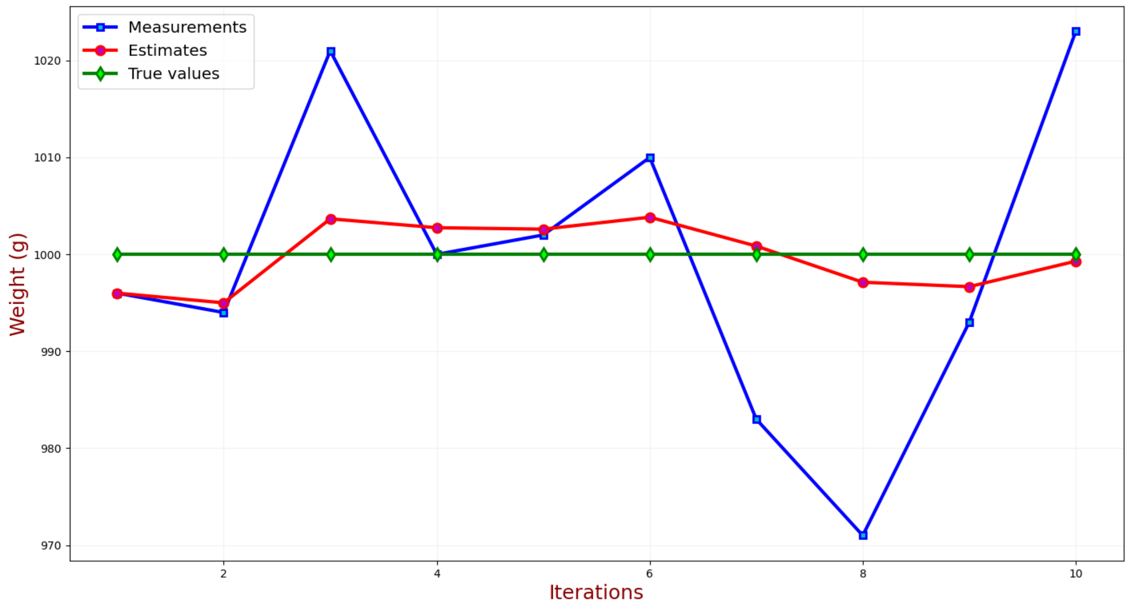


Figure 3.6: *Example 1: Measurements vs. True value vs. Estimates.*

The estimation algorithm has a smoothing effect on the measurements and converges toward the true value.

3.1.4 Example summary

In this example, we've developed a simple estimation algorithm for a static system. We have also derived the state update equation, one of the five Kalman Filter equations. We will revise the state update equation in subsection 4.1.4.

Bibliography

Articles

[1] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Trans. of the ASME - Journal of Basic Engineering* (1960), pages 35–45 (cited on page 5).

[2] Tarunraj Singh Dirk Tenne. "Optimal design of $\alpha - \beta - (\gamma)$ filters". In: *Proceedings of the American Control Conference 6* 6 (Feb. 2000), pages 4348–4352. DOI: 10.1109/ACC.2000.877043 (cited on page 73).

[3] P. R. Kalata. "A generalized parameter for $\alpha - \beta$ and $\alpha - \beta - \gamma$ target trackers". In: *The 22nd IEEE Conference on Decision and Control* (Dec. 1983). DOI: 10.1109/CDC.1983.269580 (cited on page 73).

[4] Wayne E. Hoover. "Algorithms For Confidence Circles and Ellipses". In: *NOAA Technical Report NOS 107 CGS 3* (Sept. 1984) (cited on page 148).

[5] L. Campo Y. Bar-Shalom. "The Effect of the Common Process Noise on the Two-Sensor Fused-Track Covariance". In: *Advances in Control Systems* 3 (1966), pages 293–340. DOI: https://doi.org/10.1016/B978-1-4831-6716-9.50011-4 (cited on page 237).

[6] S. F. Schmidt L. A. Mcgee. "Discovery of the Kalman filter as a practical tool for aerospace and industry". In: *ech. Rep., NASA-TM-86847* (1985). DOI: 10.1109/TAES.1986.310815 (cited on page 237).

[7] Kai O. Arras. "An Introduction To Error Propagation: Derivation, Meaning and Examples of Equation Cy=FxCxFxT". In: *EPFL-ASL-TR-98-01 R3* (1998). DOI: https://doi.org/10.3929/ethz-a-010113668 (cited on page 246).

[8] Jeffrey K. Uhlmann Simon J. Julier. "New extension of the Kalman filter to nonlinear systems". In: *Proc. SPIE 3068, Signal Processing, Sensor Fusion, and Target Recognition VI* (July 1997). DOI: https://doi.org/10.1117/12.280797 (cited on pages 283, 284, 337).

[9] Eric A. Wan Rudolph Van Der Merwe. "Sigma-point kalman filters for probabilistic inference in dynamic state-space models". In: *Oregon Health and Science University* (2004). DOI: https://doi.org/10.6083/M4Z60KZ5 (cited on pages 285, 337).

[10] H.F. Durrant-Whyte S. Julier J. Uhlmann. "A new method for the nonlinear transformation of means and covariances in filters and estimators". In: *IEEE*

Transactions on Automatic Control 45.3 (Mar. 2000), pages 477–482. DOI: 10.1109/9.847726 (cited on page 298).

[11] Herman Bruyninckx Tine Lefebvre and Joris De Schutter. “Comment on A new method for the nonlinear transformation of means and covariances in filters and estimators”. In: *IEEE Transactions on Automatic Control* 47.8 (Aug. 2002), pages 1406–1408. DOI: 10.1109/TAC.2002.800742 (cited on page 298).

[12] Rudolph van der Merwe Eric A. Wan. “The Unscented Kalman Filter for Nonlinear Estimation”. In: *IEEE Proceedings of IEEE 2000 adaptive systems for signal processing, communication and control symposium* (Oct. 2000). DOI: https://doi.org/10.3390/s21020438 (cited on page 320).

[13] René van de Molengraft Jos Elfring Elena Torta. “Particle Filters: A Hands-On Tutorial”. In: *Sensors 2021, 21(2)* (2021), page 438 (cited on page 337).

[14] Y. Bar-Shalom K. C. Chang R. K. Saha. “On optimal track-to-track fusion”. In: *IEEE Transactions on Aerospace and Electronic Systems* 33.8 (Oct. 1997), pages 1271–1276. DOI: doi:10.1109/7.625124 (cited on page 348).

[15] L. Campo Y. Bar-Shalom. “The Effect of the Common Process Noise on the Two-Sensor Fused-Track Covariance”. In: *IEEE Transactions on Aerospace and Electronic Systems* AES-22.6 (Nov. 1986), pages 803–805. DOI: 10.1109/TAES.1986.310815 (cited on page 348).

[16] P.A. Bromiley. “Products and Convolutions of Gaussian Probability Density Functions”. In: *Internal Report* (Aug. 2014) (cited on pages 421, 427).

Books

[17] Joseph P.D. Bucy R.S. *Filtering for Stochastic Processes with Applications to Guidance, Chapter 16*. Interscience, New York, 1968 (cited on page 182).

[18] Uhlmann Jeffrey. *Dynamic map building and localization: new theoretical foundations, Thesis (Ph.D.)*. University of Oxford, 1995 (cited on pages 283, 320).

[19] Tom M. Apostol. *Calculus, theorem 8.3*. John Wiley and Sons, Inc, 1967 (cited on page 404).

Index