Master's Notes

Wellington Vieira M. de Castro

November 16, 2021

> O conhecido é finito, o desconhecido, infinito; intelectualmente estamos numa ilhota no meio de um oceano ilimitado de inexplicabilidade. Nossa função em cada geração é reinvidicar um pouco mais de terra firme.
>
> T. H. Huxley

TODO

- ~~Put notes in equations~~
- Rewrite the probability part in vector form
- Compare the different diff drive models
- Learn and write about the geneal kinematics model based on Twists
- ~~Write the theory behind Kalman Filters~~
- ~~Organize a Kalman filters taxonomy~~
- Write the Robot Model and Measurement Model Jacobians

Contents

1 Introduction

"If you can't explain it simply, you don't understand it well enough." - Albert Einstein

1.1 Why Active SLAM ?

"We do not just see, we look. And in the course, our pupils adjust to the level of illumination, our eyes bring the world into sharp focus, our eyes converge or diverge, we move our heads or change our position to get a better view of something, and sometimes we even put our spectacles" [1]. I also say that we walk and explore the world, according to our needs and wills, increasing our knowledge about its structure.

1.2 Markov Assumption

TODO

1.3 Linearization and Variance of the Gaussian Distribution

TODO

2 Mathematics

The way that the geometry in this section is addressed was heavily influenced by [4].

2.1 Notation to English

Matrices

A_{ij}	Element of the $\mathbf{A}$ matrix in the *ith* row and *jth* coloum.
$\mathrm{Tr}\,(\cdot)$	Trace operator applied to squared matrices.
$\mathbf{I}$	Identity matrix.
$\mathbf{0}_{m\times n}$	$m \times n$ null matrix

Vectors

Probability

2.2 Vocabulary

- **Degrees of Freedom (DoF):** The number of **degrees of freedom** of a robot is the smallest number real-valued coordinates needed to represent its configuration [4] (pose).

$\mathbf{x}$	Column vector.
$\dot{\mathbf{x}}$	Derivative of the $\mathbf{x}$ vector with respect to time. Newton's derivative notation.
$\boldsymbol{X}$	Random vector.
$\boldsymbol{x}$	Random vector realization or observed value.
$x_{1:n}$	Generic number sequence. Can be used as a column vector, or a random vector realization, for instance.

$p(\cdot)$	Probability Density Function (pdf)
$\mathbb{E}[\cdot]$	Expectation operator
$\boldsymbol{\mu_X}$	Expected value, or mean, of the random vector $\boldsymbol{X}$
$Cov(\cdot)$	Covariance of a random variable or vector
$\mathbf{K_X}$	Covariance matrix of the random vector $\boldsymbol{X}$

- **Configuration space** The n-dimensional space containing all possible configurations of the robot is called the **configuration space** (**C-space**). The configuration of a robot is represented by a point in its C-space. [4]

- **Free vector:** A **free vector** is a geometric quantity with a length and a direction. It is called free because it is not necessarily rooted anywhere, e.g. within a coordinate system; only its length and direction matter. A linear velocity can be viewed as a free vector.

2.3 Matrices

Matrices are the mathematics's soul of this work, they are as fundamental to this work as electricity is to a computer. In this Section, I will try to explain all the useful (for our purposes) tricks regarding matrices. It may have matrix specific subjects in other Sections because, it is hard sometimes, to separate definitions from applications.

2.3.1 The exponential of a matrix

At first glance the expression $e^{\mathbf{A}}$ may seen strange, because most of the times we correlate potentiating to successive multiplications, but how a matrix can dictate how many times we multiply e by itself, like the expression e^2, tell us? Well, actually it does not. But if we see this through the Taylor series expansion of e^x, Eq. 1, it gets less awkward.

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = \frac{1}{0!} + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^{n-1}}{(n-1)!} + \frac{x^n}{n!} \tag{1}$$

If we just plug $\mathbf{A}$ in the place of x, we have:

$$e^{\mathbf{A}} = \sum_{n=0}^{\infty} \frac{\mathbf{A}^n}{n!} = \frac{\mathbf{A}^0}{0!}^{\mathbf{I}} + \frac{\mathbf{A}^1}{1!} + \frac{\mathbf{A}^2}{2!} + \cdots + \frac{\mathbf{A}^{n-1}}{(n-1)!} + \frac{\mathbf{A}^n}{n!} \tag{2}$$

So the mystery is solved! The exponential of a matrix is just a matrix! Well, when you interpret the exponential through the lens of the Taylor series.

2.3.2 Derivative of $e^{\mathbf{A}t}$

We will see that the derivative of $e^{\mathbf{A}t}$, $\mathbf{A} \in \mathbb{R}^{m \times n}$ its very similar to its real counterpart e^{at}, $a \in \mathbb{R}$.

$$\begin{aligned}
\frac{\partial}{\partial t} e^{\mathbf{A}t} &= \frac{\partial}{\partial t}\left(\sum_{n=0}^{\infty} \frac{\mathbf{A}^n t^n}{n!}\right) = \frac{\partial}{\partial t}\left(\mathbf{I} + \frac{\mathbf{A}\,t}{1!} + \frac{\mathbf{A}^2 t^2}{2!} + \cdots + \frac{\mathbf{A}^n t^n}{n!}\right) \\
&= \cancel{\mathbf{I}}^{\mathbf{0}} + \frac{\mathbf{A} \cdot \cancel{1} t^0}{\cancel{1!}^{0!}} + \frac{\mathbf{A}^2 \cdot \cancel{2} t}{\cancel{2!}^{1!}} + \frac{\mathbf{A}^3 \cdot \cancel{3} t^2}{\cancel{3!}^{2!}} + \cdots + \frac{\mathbf{A}^n \cdot \cancel{n} t^{n-1}}{\cancel{n!}^{(n-1)!}} \\
&= \mathbf{A}\underbrace{\left(\mathbf{I} + \frac{\mathbf{A}\,t}{1!} + \frac{\mathbf{A}^2 t^2}{2!} + \cdots + \frac{\mathbf{A}^{n-1} t^{n-1}}{(n-1)!}\right)}_{e^{\mathbf{A}t}} \\
&= \mathbf{A}\, e^{\mathbf{A}t}
\end{aligned} \tag{3}$$

2.3.3 Commutativity of $\mathbf{A}$ and $e^{\mathbf{A}t}$

This is a straightforward from the development of Eq. 3, in the last but one step we could have put $\mathbf{A}$ in evidence in the right side, leading to $e^{\mathbf{A}t}\mathbf{A}$. Therefore:

$$\mathbf{A}\, e^{\mathbf{A}t} = e^{\mathbf{A}t}\, \mathbf{A} \tag{4}$$

2.3.4 Inverse of $e^{\mathbf{A}t}$

OBTER A INVERSA, ACHO QUE É POR LAPLACE

2.3.5 Product of $e^{\mathbf{A}}$ and $e^{\mathbf{B}}$

MOSTRAR QUE NO PRODUTO OS EXPOENTES PODEM SER SOMADOS

2.3.6 Trace

The Trace is a operation defined for squared matrices, it is simply the sum of the elements in the main diagonal. It is defined in Eq. 5

$$\mathrm{Tr}(\mathbf{A}) \triangleq \sum_i A_{ii} \tag{5}$$

2.3.7 Matrix with Vector Multiplication

Given a $m \times n$ matrix $\mathbf{A}$, and a $n \times 1$ vector $\mathbf{x}$, their multiplication is defined in Eq. 6.

$$\mathbf{Ax} = \begin{bmatrix} \sum_{j=1}^{n} A_{1j}\, x_j \\ \sum_{j=1}^{n} A_{2j}\, x_j \\ \vdots \\ \sum_{j=1}^{n} A_{mj}\, x_j \end{bmatrix} \tag{6}$$

We can also pre-multiply a matrix by a vector transposed vector, let $\mathbf{x}$ be a $m \times 1$ vector now.

$$\mathbf{x^T A} = \begin{bmatrix} \sum_{i=1}^{m} A_{i1}\, x_i & \sum_{i=1}^{m} A_{i2}\, x_i & \dots & \sum_{i=1}^{m} A_{in}\, x_i \end{bmatrix} \tag{7}$$

2.3.8 The dot product between vectors

Given two $n \times 1$ vectors $\mathbf{x}$ and $\mathbf{y}$, the dot (or internal) product between then is defined as

$$\mathbf{x^T y} = \sum_{i=1}^{n} x_i\, y_i \tag{8}$$

since both the sum and the multiplication are commutative operator, we have that

$$\mathbf{x^T y} = \mathbf{y^T x} \tag{9}$$

2.3.9 Vector and Matrix Differentiation

In this Section I will present some matrix expressions differentiation that will help understanding the math used in the techniques presented throughout the text. To build the expressions we will often fall in the following strategy: reduce the formulas to index notation, where we obtain the formula of each scalar in the vector or matrix, then we will derive this scalar with the rules already known from calculus.

The Gradient Vector

When we have a vector $\mathbf{x} \in \mathbb{R}^n$ related to a scalar $\alpha \in \mathbb{R}$ trough a function $f : \mathbb{R}^n \to \mathbb{R}$, the derivative of the scalar α in relation to the vector $\mathbf{x}$ is the gradient vector $\boldsymbol{\nabla} f$, defined in Eq. 10.

$$\frac{\partial \alpha}{\partial \mathbf{x}} = \boldsymbol{\nabla} f \triangleq \begin{bmatrix} \dfrac{\partial f(\mathbf{x})}{\partial x_1} \\ \dfrac{\partial f(\mathbf{x})}{\partial x_2} \\ \vdots \\ \dfrac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix}, \text{if } \alpha = f(\mathbf{x}) \tag{10}$$

The Gradient Matrix

Whereas the Gradient Vector is the derivative of a scalar with respect to a vector's components, the Gradient Matrix is the derivative of a scalar with respect each element of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$. Let $f : \mathbb{R}^{m \times n} \to \mathbb{R}$, the derivative of f with respect to the $\mathbf{A}$ matrix is defined in Eq. 11.

$$\frac{\partial f}{\partial \mathbf{A}} \triangleq \begin{bmatrix} \dfrac{\partial f}{\partial A_{11}} & \dfrac{\partial f}{\partial A_{12}} & \cdots & \dfrac{\partial f}{\partial A_{1n}} \\ \dfrac{\partial f}{\partial A_{21}} & \dfrac{\partial f}{\partial A_{22}} & \cdots & \dfrac{\partial f}{\partial A_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial f}{\partial A_{m1}} & \dfrac{\partial f}{\partial A_{m2}} & \cdots & \dfrac{\partial f}{\partial A_{mn}} \end{bmatrix} \tag{11}$$

The gradient matrix has the same number of rows and columns as $\mathbf{A}$ matrix.

The Jacobian Matrix

Whenever we have a vector $\mathbf{x} \in \mathbb{R}^n$ related to another vector $\mathbf{y} \in \mathbb{R}^m$ trough some function $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^m$, the derivative of $\mathbf{y}$ with respect to $\mathbf{x}$ is the Jacobian matrix $\mathbf{J}(\mathbf{x}) \in \mathbb{R}^{m \times n}$. Let

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} g_1(\mathbf{x}) \\ g_2(\mathbf{x}) \\ \vdots \\ g_m(\mathbf{x}) \end{bmatrix}$$

we have that

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{J}(\mathbf{x}) \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial g_1(\mathbf{x})}{\partial x_1} & \frac{\partial g_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial g_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial g_2(\mathbf{x})}{\partial x_1} & \frac{\partial g_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial g_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_m(\mathbf{x})}{\partial x_1} & \frac{\partial g_m(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial g_m(\mathbf{x})}{\partial x_n} \end{bmatrix} \tag{12}$$

The above expression is the Jacobian matrix of $\mathbf{y}$ with respect to $\mathbf{x}$.

Derivative of $\mathbf{y} = \mathbf{Ax}$ With Respect to $\mathbf{x}$

Let

$$\mathbf{y} = \mathbf{Ax}, \text{with } \mathbf{A} \in \mathbb{R}^{m \times n} \text{ and } \mathbf{x} \in \mathbb{R}^n$$

as can be seen in Eq. 6 the *ith* element of the $\mathbf{y}$ vector is given by

$$y_i = \sum_{j=1}^{n} A_{ij} x_j = A_{i1}x_1 + A_{i2}x_2 + \cdots + A_{in}x_n$$

so we have that

$$\frac{\partial y_i}{\partial x_k} = A_{ik}, \text{for } i = 1, 2, \ldots m \text{ and } k = 1, 2, \ldots, n$$

therefore we have that the Jacobian of the vector $\mathbf{y}$ in relation with the vector $\mathbf{x}$ is the matrix $\mathbf{A}$.

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \mathbf{A}, \text{if } \mathbf{y} = \mathbf{Ax} \tag{13}$$

Derivative of $\alpha = \mathbf{x^T Ax}$

It is very common to differentiate the scalar α obtained by the product $\mathbf{x^T Ax}$, with respect to the vector $\mathbf{x}$, for this first I will rewrite α:

$$\begin{aligned} \alpha &= \mathbf{x^T Ax} \\ &= \begin{bmatrix} \sum_{i=1}^{n} A_{i1}\, x_i & \sum_{i=1}^{n} A_{i2}\, x_i & \ldots & \sum_{i=1}^{n} A_{in}\, x_i \end{bmatrix} \mathbf{x} && \text{Applied Eq. 7} \\ &= \begin{bmatrix} \sum_{i=1}^{n} A_{i1}\, x_i & \sum_{i=1}^{n} A_{i2}\, x_i & \ldots & \sum_{i=1}^{n} A_{in}\, x_i \end{bmatrix} \mathbf{x} \\ &= \sum_{j=1}^{n} \left(\sum_{i=1}^{n} A_{ij}\, x_i \right) x_j && \text{Applied Eq. 8} \\ &= \sum_{j=1}^{n} \sum_{i=1}^{n} A_{ij}\, x_i x_j \end{aligned} \tag{14}$$

To derive α with respect to the *kth* component of the $\mathbf{x}$ vector, we have to derive the portion obtained when the *kth* component is present because of the left summation and when it is present because of the right summation:

$$\begin{aligned}\frac{\partial \alpha}{\partial x_k} &= \frac{\partial}{\partial x_k}\left(\sum_{i=1}^{n} A_{ik}x_i x_k + \sum_{j=1}^{n} A_{kj}x_k x_j\right) \\ &= \sum_{i=1}^{n} A_{ik}x_i \frac{\partial x_k}{\partial x_k} + \sum_{j=1}^{n} A_{kj}\frac{\partial x_k}{\partial x_k}x_j \\ &= \sum_{i=1}^{n} A_{ik}x_i 1 + \sum_{j=1}^{n} A_{kj}1 x_j \\ &= \sum_{i=1}^{n} A_{ik}x_i + \sum_{j=1}^{n} A_{kj}x_j\end{aligned} \tag{15}$$

for all components we have the following gradient vector

$$\frac{\partial \alpha}{\partial \mathbf{x}} = \mathbf{x^T A} + \mathbf{x^T A^T} = \mathbf{x^T}\left(\mathbf{A} + \mathbf{A^T}\right) \tag{16}$$

Matrix Product Derivative With Respect to Scalar

Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{B} \in \mathbb{R}^{n\times p}$ be matrices whose elements $A_{ij}(x)$ and $B_{ij}(x)$, respectively, are functions of a scalar x. We have that the ij element of the product between the two, $[\mathbf{AB}]_{ij}$, is

$$[\mathbf{AB}]_{ij} = \sum_{k=1}^{n} A_{ik}(x)B_{kj}(x)$$

and its derivative with respect to x can be written in terms of the product derivative rule, as following

$$\begin{aligned}\frac{\partial}{\partial x}[\mathbf{AB}]_{ij} &= \frac{\partial}{\partial x}\sum_{k=1}^{n} A_{ik}(x)B_{kj}(x) \\ &= \sum_{k=1}^{n} \frac{\partial}{\partial x}\left(A_{ik}(x)B_{kj}(x)\right) \\ &= \sum_{k=1}^{n} \underbrace{\left(\frac{\partial}{\partial x}A_{ik}(x)\right)}_{A'_{ik}(x)} B_{kj}(x) + A_{ik}(x)\underbrace{\left(\frac{\partial}{\partial x}B_{kj}(x)\right)}_{B'_{kj}(x)} \quad \text{Applied Product Derivative Rule} \\ &= \sum_{k=1}^{n} A'_{ik}(x)B_{kj}(x) + \sum_{k=1}^{n} A_{ik}(x)B'_{kj}(x) \\ &= \left[\left(\frac{\partial}{\partial x}\mathbf{A}\right)\mathbf{B}\right]_{ij} + \left[\mathbf{A}\left(\frac{\partial}{\partial x}\mathbf{B}\right)\right]_{ij}\end{aligned}$$

so according to the above, we can get the matrix product derivative with respect to a scalar x, as

$$\frac{\partial}{\partial x}(\mathbf{AB}) = \left(\frac{\partial}{\partial x}\mathbf{A}\right)\mathbf{B} + \mathbf{A}\left(\frac{\partial}{\partial x}\mathbf{B}\right) \tag{17}$$

which resembles a lot the product derivative rule for scalar functions.

Chain Rule in Linear Expressions

TODO

Trace Derivative With Respect to Matrix

Here I will show a property about trace derivative with respect to a matrix, followed by some examples that cover all the math, regarding to trace differentiation, needed to derive the SLAM algorithms used.

Trace Derivative Linearity

Let $\mathbf{A}(x)$ be a matrix dependent of the scalar x. I want to show that

$$\frac{\partial}{\partial x}\operatorname{Tr}(\mathbf{A}(x)) = \operatorname{Tr}\left(\frac{\partial}{\partial x}\mathbf{A}(x)\right)$$

for this we have that

$$\begin{aligned}\frac{\partial}{\partial x}\operatorname{Tr}(\mathbf{A}(x)) &= \frac{\partial}{\partial x}\sum_{i=1}^{n} A_{ii}(x) \\ &= \sum_{i=1}^{n}\frac{\partial}{\partial x}A_{ii}(x) \quad \text{Applied the } \frac{\partial}{\partial x} \text{ operator linearity property.}\end{aligned} \tag{18}$$

on the other rand we have that

$$\frac{\partial}{\partial x}\mathbf{A}(x) = \begin{bmatrix} \frac{\partial}{\partial x}A_{11}(x) & \frac{\partial}{\partial x}A_{12}(x) & \cdots & \frac{\partial}{\partial x}A_{1n}(x) \\ \frac{\partial}{\partial x}A_{21}(x) & \frac{\partial}{\partial x}A_{22}(x) & \cdots & \frac{\partial}{\partial x}A_{2n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x}A_{n1}(x) & \frac{\partial}{\partial x}A_{n2}(x) & \cdots & \frac{\partial}{\partial x}A_{nn}(x) \end{bmatrix}$$

then

$$\begin{aligned}\operatorname{Tr}\left(\frac{\partial}{\partial x}\mathbf{A}(x)\right) &= \frac{\partial}{\partial x}A_{11}(x) + \frac{\partial}{\partial x}A_{22}(x) + \cdots + \frac{\partial}{\partial x}A_{nn}(x) \\ &= \sum_{i=1}^{n}\frac{\partial}{\partial x}A_{ii}(x)\end{aligned} \tag{19}$$

since Equations 18 and 19 evaluate to the same expression, Eq. 20.

$$\frac{\partial}{\partial x}\operatorname{Tr}(\mathbf{A}(x)) = \operatorname{Tr}\left(\frac{\partial}{\partial x}\mathbf{A}(x)\right) \tag{20}$$

Useful Trace Derivatives Examples

1. Derivative of $\operatorname{Tr}(\mathbf{AXB})$ with respect to $\mathbf{X}$

   Let's say that $\mathbf{AXB} \in \mathbb{R}^{m\times m}$, that is: $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{X} \in \mathbb{R}^{n\times p}$, and $\mathbf{B} \in \mathbb{R}^{p\times m}$. First of all,

I will rewrite the $\mathrm{Tr}(\mathbf{AXB})$ in index notation:

$$\begin{aligned}\mathrm{Tr}(\mathbf{AXB}) &= \sum_{i=1}^{m} [\mathbf{AXB}]_{ii} \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}\,[\mathbf{XB}]_{ji} \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}\sum_{k=1}^{p} X_{jk}B_{ki} \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} A_{ij}X_{jk}B_{ki}\end{aligned}$$

The derivative of $\mathrm{Tr}(\mathbf{AXB})$ with respect to a particular element X_{jk} is given by

$$\frac{\partial}{\partial X_{jk}}\mathrm{Tr}(\mathbf{AXB}) = \sum_{i=1}^{n} A_{ij}B_{ki} = [\mathbf{BA}]_{kj} \tag{21}$$

from the above expression, we have that the derivative of this trace with respect to the $\mathbf{X}$ matrix is

$$\frac{\partial}{\partial \mathbf{X}}\mathrm{Tr}(\mathbf{AXB}) = \mathbf{BA}$$

but, $\mathbf{BA} \in \mathbb{R}^{p\times n}$ and as stated by Eq. 11 the Gradient Matrix must have the same dimension as the matrix whom the expression is derived with respect to, since $\mathbf{X} \in \mathbb{R}^{n\times p}$, we just need to transpose the product $\mathbf{BA}$ to get the dimension right. Equation 22 gives this trace's derivative with respect to the $\mathbf{X}$ matrix.

$$\begin{aligned}\frac{\partial}{\partial \mathbf{X}}\mathrm{Tr}(\mathbf{AXB}) &= (\mathbf{BA})^{\mathbf{T}} \\ &= \mathbf{A}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}}\end{aligned} \tag{22}$$

2. Derivative of $\mathrm{Tr}\left(\mathbf{AX}^{\mathbf{T}}\mathbf{B}\right)$ with respect to $\mathbf{X}$

   Let's define $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{X} \in \mathbb{R}^{p\times n}$ and $\mathbf{B} \in \mathbb{R}^{p\times m}$. Using the index notation as in the previous example, we have that:

$$\mathrm{Tr}\left(\mathbf{AX}^{\mathbf{T}}\mathbf{B}\right) = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} A_{ij}X_{kj}B_{ki}$$

   The derivative of $\mathrm{Tr}\left(\mathbf{AX}^{\mathbf{T}}\mathbf{B}\right)$ with respect to a particular X_{kj} is given by

$$\frac{\partial}{\partial X_{kj}}\mathrm{Tr}\left(\mathbf{AX}^{\mathbf{T}}\mathbf{B}\right) = \sum_{i=1}^{m} A_{ij}B_{ki} = [\mathbf{BA}]_{kj} \tag{23}$$

   therefore,

$$\frac{\partial}{\partial \mathbf{X}}\mathrm{Tr}\left(\mathbf{AX}^{\mathbf{T}}\mathbf{B}\right) = \mathbf{BA} \tag{24}$$

   which is in agreement with the dimensions of $\mathbf{X}$, so satisfies the Gradient Matrix dimension rule.

3. Derivative of $\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right)$ with respect to $\mathbf{X}$

   Instead of rewriting $\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right)$ in index notation and deriving with respect to a X_{jk} element, I will calculate this result using the Trace Derivative Linearity property (Eq. 20), the Matrix Product Derivative Rule (Eq. 17) and, the previous two examples. First lets apply the the two properties aforementioned:

$$\begin{aligned}
\frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) &= \mathrm{Tr}\left(\frac{\partial}{\partial X_{jk}}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right)\right) && \text{Applied Eq. 20}\\
&= \mathrm{Tr}\left(\left(\frac{\partial}{\partial X_{jk}}\mathbf{X}\right)\mathbf{A}\mathbf{X}^{\mathbf{T}} + \mathbf{X}\left(\frac{\partial}{\partial X_{jk}}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right)\right) && \text{Appled Eq. 17}\\
&= \mathrm{Tr}\left(\left(\frac{\partial}{\partial X_{jk}}\mathbf{X}\right)\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) + \mathrm{Tr}\left(\mathbf{X}\left(\frac{\partial}{\partial X_{jk}}\right)\mathbf{A}\mathbf{X}^{\mathbf{T}}\right)\\
&= \mathrm{Tr}\left(\left(\frac{\partial}{\partial X_{jk}}\mathbf{X}\right)\underbrace{\mathbf{A}\mathbf{X}^{\mathbf{T}}}_{\mathbf{B}}\right) + \mathrm{Tr}\left(\underbrace{\mathbf{X}\mathbf{A}}_{\mathbf{C}}\left(\frac{\partial}{\partial X_{jk}}\mathbf{X}^{\mathbf{T}}\right)\right)
\end{aligned}$$

   by Eq. 20 we can move the derivative operator out of the trace

$$\frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) = \frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{X}\mathbf{B}\right) + \frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{C}\mathbf{X}^{\mathbf{T}}\right)$$

   finally we can apply the results in Equations 23 and 23

$$\begin{aligned}
\frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) &= \frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{I}\mathbf{X}\mathbf{B}\right) + \frac{\partial}{\partial X_{jk}}\mathrm{Tr}\left(\mathbf{C}\mathbf{X}^{\mathbf{T}}\mathbf{I}\right)\\
&= \left[\mathbf{B}\,\mathbf{I}\right]_{kj} + \left[\mathbf{I}\,\mathbf{C}\right]_{kj}
\end{aligned}$$

   the identity matrices were inserted above just for the expression to look in the same format as in the results we have used. Therefore, as before, we can write

$$\frac{\partial}{\partial \mathbf{X}}\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) = \mathbf{I}^{\mathbf{T}}\,\mathbf{B}^{\mathbf{T}} + \mathbf{I}\,\mathbf{C}$$

   remembering that $\mathbf{B} = \mathbf{A}\,\mathbf{X}^{\mathbf{T}}$ and $\mathbf{C} = \mathbf{X}\,\mathbf{A}$, we have that

$$\frac{\partial}{\partial \mathbf{X}}\mathrm{Tr}\left(\mathbf{X}\mathbf{A}\mathbf{X}^{\mathbf{T}}\right) = \mathbf{X}\mathbf{A}^{\mathbf{T}} + \mathbf{X}\mathbf{A} \tag{25}$$

2.4 State Space representation of Dynamic Systems

There are two main ways of describe behaviour of a physical system: Dynamic and Kinematic models. A Dynamic Model models the system behaviour through the interaction of forces and torques. Whereas the Kinematic Model explains the system's behaviour through velocities, which are, often, the derivative the system's elements position.

To represent a kinematic model, we apply the use of state spaces notation. In a linear state space, the system's velocities is a linear function of the system state and the control input.

The $\mathbf{x}$ is commonly used to represent the vector state, with component $\{x_1, x_2, \ldots, x_n\}$, and $\mathbf{u}$ is used to represent the control vector, with components $\{u_1, u_2, \ldots, u_m\}$. Equation 26 is a general representation of a continuous linear time-variant system in the state space. The fact that the vectors $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{y}$ are functions of time is implicit.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A_t x} + \mathbf{B_t u} \\ \mathbf{y} &= \mathbf{C_t x} + \mathbf{D_t u} \end{aligned} \tag{26}$$

where

$\mathbf{x}$ is the system's state vector

$\mathbf{A_t}$ is named state matrix

$\mathbf{B_t}$ is the input or control matrix

$\mathbf{y}$ is the output vector

$\mathbf{C_t}$ is the output matrix

$\mathbf{D_t}$ is the feedforward matrix. Always $\mathbf{0}$ for physical systems.

In Equation 26 all the matrices has a sub index $\mathbf{t}$, the sub index is used to show that they can vary in time. Systems where the system, input, output and feedforward matrices do not vary in time are called time-invariant. Equation 27 is a state space of a continuous linear time-invariant system.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A x} + \mathbf{B u} \\ \mathbf{y} &= \mathbf{C x} + \mathbf{D u} \end{aligned} \tag{27}$$

One can say that the state space representation for linear systems is useless, since real-life systems are usually non-linear or are linear just inside an operation zone. Well, this is true. But, quoting Richard Feynman "Linear systems are important because we can solve them", what we usually do is linearize the non-linear systems so that we can work with them, therefore, linear state space representation is not useless.

State space is used to represent linear discrete systems, too. The notation is slightly different, and opposite to the continuous case, the state vector derivative with respect to time is not used. In the discrete case, the next state is a linear function of the current state and the current control input. Equation 28 is the representation of a discrete linear time-variant system.

$$\begin{aligned} \mathbf{x_k} &= \mathbf{A_k x_{k-1}} + \mathbf{B_k u_k} \\ \mathbf{y_k} &= \mathbf{C_k x_k} + \mathbf{D_k u_k} \end{aligned} \tag{28}$$

The sub index **k** is used with discrete systems, as in the continuous case, the system can be also time-invariant, that is, the system's matrices do not depend of the time step. Equation 29 is a representation of a discrete linear time-invariant system.

$$\begin{aligned} \mathbf{x_k} &= \mathbf{A x_{k-1}} + \mathbf{B u_k} \\ \mathbf{y_k} &= \mathbf{C x_k} + \mathbf{D u_k} \end{aligned} \tag{29}$$

Here we will work with discrete time-variant linear systems, since we are using computers to control our robot models.

2.5 Probability

Probability is a branch of mathematics used whenever uncertainty appears in a problem. As robots leave the well controlled and highly predictable assembly lines and enter in the real (open) world, a lot of uncertainty arises. It is impossible to pre-built a map for every scenario a vacuum cleaner robot will encounter while cleaning an endless variety of houses, or model every response of the Perseverance Rover in the *random* Mars environment.

Sebastian Thrun in [5] conjectures that "A robot that carries a notion of its own uncertainty and that acts accordingly, will do better than one that does not.", being a little socratic I would say that a robot aware of its own ignorance, is far more *wise* than a robot that does not. So, is more than natural that we incorporate probability in our robotics algorithms, this way they can have knowledge about their own ignorance about where they *think* they are, and how they *think* the world looks like.

The reader must encounter a lot of names/concepts as: random variable, expected value, variance, probability density function, and others. In this chapter, I will give a very informal introduction about then, just the necessary minimum to our purposes.

2.5.1 Probability density function

caracterizar uma funcao densidade de probabilidade

2.5.2 Random Variable Independence

We say that two random variables are statistically independent if and only if Equation 30 holds true.

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})\, p(\boldsymbol{y}) \tag{30}$$

2.5.3 Expected value (mean)

Given the probability density $p(\boldsymbol{x})$ function of the continuous random vector $\boldsymbol{X}$, the expected value $\mathbb{E}[\boldsymbol{X}]$ is the integral in Eq. 31.

$$\mathbb{E}[\boldsymbol{X}] = \boldsymbol{\mu_X} = \int_{-\infty}^{\infty} \boldsymbol{x}\, p(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \tag{31}$$

Often $\mathbb{E}$ is referred as the expectation operator. $\mathbb{E}[\boldsymbol{X}]$ is also known as the first moment of the random vector $\boldsymbol{X}$.

2.5.4 Random Variable Variance

The variance of a random variable measures the sparseness of the random variable realizations around the mean, and is given by Eq. 32. The variance is also represented by the σ_X^2 symbol.

$$Var(X) = \mathbb{E}\left[(X - \mu_X)^2\right] = \int_{-\infty}^{\infty} (x - \mu_X)^2\, p(x)\, dx \tag{32}$$

The expectation operator is defined in terms of a integral, so it is natural that the expectation operator inherit the algebraic properties of integrals. So working with Eq. 32 let us write the variance of a random variable in a different, and sometimes more convenient, way:

$$\begin{aligned} \mathbb{E}\left[(X - \mu)^2\right] &= \mathbb{E}\left[X^2 - 2\mu_X X + \mu_X^2\right] \\ &= \mathbb{E}\left[X^2\right] - 2\mu_X \underbrace{\mathbb{E}[X]}_{\mu_X} + \mu_X^2 \\ &= \mathbb{E}\left[X^2\right] - \mu_X^2 \end{aligned} \tag{33}$$

Equation 33 shows that the variance of a random variable is its second moment minus the first squared.

2.5.5 The Univariate Gaussian Distribution (Normal Distribution)

The Gaussian Distribution, also known as the Normal Distribution, is a probability density function that can be fully characterized by its first and second moments only. The single-variable Gaussian distribution is given by Eq. 34

$$p(x) = \frac{1}{\sqrt{2\pi\sigma_X^2}} \exp\left\{\left(-\frac{(x-\mu_X)^2}{2\sigma_X^2}\right)\right\} \tag{34}$$

Another way to refer to a Gaussian is by bell curve, the reason is easily seen in Fig. 1. By the way, the area under the curve, between$[\mu - \sigma, \mu + \sigma]$, corresponds to about 68% of the total area.

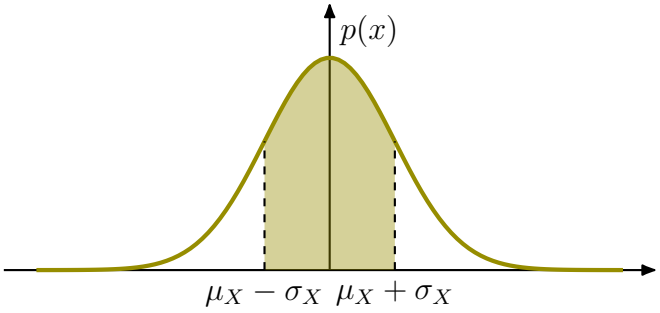

Figure 1: Gaussian distribution with the area under the curve between $-\sigma$ and $+\sigma$ around the mean highlighted.

The Gaussian distribution is, by far, the most important and used distribution in this work. Usually it is used to model noise of different natures: control, model and measurement. One could argue that model all noises as Gaussian is a mistake, but the central limit theorem of probability theory states that when independent random variables are added, the resultant distribution tends towards a normal distribution.

TODO: USAR O MAYBECK PARA JUSTIFICAR ISSO AQUI. RUÍDO BRANCO E TAL.

2.5.6 Random Variable Covariance

The covariance between two random variable X_1 and X_2, $Cov\left(X_1, X_2\right)$ or K_{X_1,X_2}, is given by the Expectation in Eq. 35.

$$K_{X_1,X_2} = \mathbb{E}\left[(X_1 - \mu_{X_1})(X_2 - \mu_{X_2})\right] \tag{35}$$

We can expand the above expression in:

$$\begin{aligned} K_{X_1,X_2} &= \mathbb{E}\left[X_1X_2\right] - \mathbb{E}\left[X_1\right]\mu_{X_2} - \mu_{X_1}\mathbb{E}\left[X_2\right] + \mu_{X_1}\mu_{X_2} \\ &= \mathbb{E}\left[X_1X_2\right] - \mu_{X_1}\mu_{X_2} - \cancel{\mu_{X_1}\mu_{X_2}} + \cancel{\mu_{X_1}\mu_{X_2}} \\ &= \mathbb{E}\left[X_1X_2\right] - \mu_{X_1}\mu_{X_2} \end{aligned} \tag{36}$$

2.5.7 Random Vector Covariance

Covariance is the equivalent for random vectors of the variance for random variables. The covariance, $Cov\left(\boldsymbol{X}\right)$ or $\mathbf{K}_{\boldsymbol{X}}$, of the random vectors $\boldsymbol{X} \in \mathbb{R}^n$ is given by

$$\begin{aligned} Cov\left(\boldsymbol{X}\right) &= \mathbb{E}\left[(\boldsymbol{X} - \boldsymbol{\mu_X})(\boldsymbol{X} - \boldsymbol{\mu_X})^{\mathbf{T}}\right] \\ &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^{\mathbf{T}} - \boldsymbol{X}\boldsymbol{\mu_X}^{\mathbf{T}} - \boldsymbol{\mu_X}\boldsymbol{X}^{\mathbf{T}} + \boldsymbol{\mu_X}\boldsymbol{\mu_X}^{\mathbf{T}}\right] \\ &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^{\mathbf{T}}\right] - \mathbb{E}\left[\boldsymbol{X}\right]\boldsymbol{\mu_X}^{\mathbf{T}} - \boldsymbol{\mu_X}\mathbb{E}\left[\boldsymbol{X}^{\mathbf{T}}\right] + \boldsymbol{\mu_X}\boldsymbol{\mu_X}^{\mathbf{T}} \\ &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^{\mathbf{T}}\right] - 2\boldsymbol{\mu_X}\boldsymbol{\mu_X}^{\mathbf{T}} + \boldsymbol{\mu_X}\boldsymbol{\mu_X}^{\mathbf{T}} \\ &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^{\mathbf{T}}\right] - \boldsymbol{\mu_X}\boldsymbol{\mu_X}^{\mathbf{T}} \end{aligned} \tag{37}$$

Equation 37 results in what is called the Covariance Matrix, shown in Eq. 38. Covariance Matrices are always symmetric.

$$Cov\left(\boldsymbol{X}\right) = \mathbf{K}_{\boldsymbol{X}} = \begin{bmatrix} \sigma_{X_1}^2 & K_{X_1,X_2} & \cdots & K_{X_1,X_n} \\ K_{X_2,X_1} & \sigma_{X_2}^2 & \cdots & K_{X_2,X_n} \\ \vdots & \vdots & \ddots & \vdots \\ K_{X_n,X_1} & K_{X_n,X_2} & \cdots & \sigma_{X_n}^2 \end{bmatrix} \tag{38}$$

2.5.8 The Multivariate Gaussian Distribution

TODO

2.5.9 Random Variable Correlation

The correlation, $Corr(X_1, X_2)$ or R_{X_1,X_2}, between two random variables is given by the expectation of the product of the two random variables

$$R_{X_1,X_2} = \mathbb{E}\left[X_1 X_2\right] \tag{39}$$

when we say that two random variables X_1 and X_2 are uncorrelated, this means that

$$\mathbb{E}\left[X_1 X_2\right] = \mu_{X_1}\mu_{X_2}$$

observing Eq. 36 we also have that

$$Cov\left(X_1, X_2\right) = K_{X_1,X_2} = 0$$

when the two random variables are uncorrelated.

2.5.10 Conditional probability

In this Section I remember the conditional probability definition and write it to an arbitrary number of variables.

I suppose you are already familiar with the definition of conditional probability in Eq. 40

$$p(x_1 \,|\, x_2) = \frac{p(x_1, x_2)}{p(x_2)} \tag{40}$$

it can be written as in Eq. 41 also.

$$p(x_1, x_2) = p(x_1 \,|\, x_2)\, p(x_2) \tag{41}$$

both can be rewritten to an arbitrary number of variables. Since I found it difficult to figure it out at first, they are shown below.

Equation 42 is the generalization of the expression in Eq. 40

$$p(x_1, x_2, \ldots x_i, \,|\, x_{i+1}, x_{i+2}, \ldots, x_n) = \frac{p(x_1, x_2, \ldots x_i, x_{i+1}, x_{i+2}, \ldots, x_n)}{p(x_{i+1}, x_{i+2}, \ldots, x_n)} \tag{42}$$

the generalization of Eq. 41 is called *chain rule* or *general product rule*, it is obtained by applying the conditional probability definition, in the remaining joint probability density function (*pdf*) recursively, until marginal *pdf* shows up. The generalization is in Eq. 43

$$\begin{aligned} p\left(x_1, x_2, \ldots, x_n\right) &= p\left(x_1 \,|\, x_2, x_3, \ldots, x_n\right)\, p\left(x_2, x_3, \ldots, x_n\right) \\ &= p\left(x_1 \,|\, x_2, x_3, \ldots, x_n\right)\, p\left(x_2 \,|\, x_3, x_4, \ldots, x_n\right) p\left(x_3, x_4, \ldots, x_n\right) \\ &\vdots \\ &= p\left(x_1 \,|\, x_2, x_3, \ldots, x_n\right)\, p\left(x_2 \,|\, x_3, x_4, \ldots, x_n\right) \cdots p\left(x_n\right) \\ &= \left(\prod_{i=1}^{n-1} p\left(x_i | x_{i+1}, \ldots, x_n\right)\right) p\left(x_n\right) \end{aligned} \tag{43}$$

there is a intuitive saying that helps me understand the chain rule "the probability of observing events E **and** F is the probability of observing F, multiplied by the probability of observing E, given that you observed F".

2.5.11 Manipulation of conditional probabilities

First, I will introduce some notation that will make the formulas less cluttered. Let $x_{i:n}$ be the sequence, or vector $\{x_i, x_{i+1}, \ldots, x_{i+j}, \ldots, x_n\}$. And $x_{i:n\setminus j}$ be the same vector, but without the *jth* element, that is $x_{i:n\setminus j} = \{x_i, x_{i+1}, \ldots, x_{i+j-1}, x_{i+j+1}, \ldots, x_n\}$. Thus we can write

$$p(x_j \,|\, x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n) = p\left(x_j \,|\, x_{1:n\setminus j}\right) \tag{44}$$

Sometimes it is interesting or convenient to rewrite one conditional *pdf* in terms of other *pdf*. Let's say that we want to calculate the value of $p(x_j \,|\, x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$, but we do not actually have the formula of $p\left(x_j \,|\, x_{1:n\setminus j}\right)$, one way is to rewrite it in terms of other *pdf* that we might know how to calculate. Equation 45 shows how to do that.

$$
\begin{aligned}
p\left(x_j \,|\, x_{1:n\setminus j}\right) &= \frac{p\left(x_{1:n}\right)}{p\left(x_{1:n\setminus j}\right)} \\
&= \frac{p\left(x_i \,|\, x_{1:n\setminus i}\right)\, p\left(x_{1:n\setminus i}\right)}{p\left(x_{1:n\setminus j}\right)} \\
&= \frac{p\left(x_i \,|\, x_{1:n\setminus i}\right)\, p\left(x_{1:n\setminus i}\right)}{p\left(x_{1:n\setminus j}\right)} \\
&= \frac{p\left(x_i \,|\, x_{1:n\setminus i}\right)\, p\left(x_j \,|\, x_{1:n\setminus\{i,j\}}\right)\, p\left(x_{1:n\setminus\{i,j\}}\right)}{p\left(x_{1:n\setminus j}\right)} \\
&= \frac{p\left(x_i \,|\, x_{1:n\setminus i}\right)\, p\left(x_j \,|\, x_{1:n\setminus\{i,j\}}\right)\, p\left(x_{1:n\setminus\{i,j\}}\right)}{p\left(x_i \,|\, x_{1:n\setminus\{i,j\}}\right)\, p\left(x_{1:n\setminus\{i,j\}}\right)} \\
&= \frac{p\left(x_i \,|\, x_{1:n\setminus i}\right)\, p\left(x_j \,|\, x_{1:n\setminus\{i,j\}}\right)}{p\left(x_i \,|\, x_{1:n\setminus\{i,j\}}\right)}
\end{aligned}
\tag{45}
$$

The above relation will be used further when deriving the measurement update equation of the Bayes Filter.

Other *pdf* manipulation of our interest is when we have a conditional distribution with two or more variables been conditioned and we want to split it in a product of two or more *pdf* each with just one conditioned variable.

$$
\begin{aligned}
p\left(x_{1:j} \,|\, x_{j+1:n}\right) &= \frac{p\left(x_{1:n}\right)}{p\left(x_{j+1:n}\right)} \\
&= \frac{\left(\prod_{i=1}^{n-1} p\left(x_i | x_{i+1}, \ldots, x_n\right)\right) p\left(x_n\right)}{p\left(x_{j+1:n}\right)} \qquad \text{Chain rule Eq. 43} \\
&= \frac{\left(\prod_{i=1}^{j} p\left(x_i | x_{i+1}, \ldots, x_j\right)\right) p\left(x_{j+1:n}\right)}{p\left(x_{j+1:n}\right)} \qquad = \prod_{i=1}^{j} p\left(x_i | x_{i+1}, \ldots, x_j\right)
\end{aligned}
\tag{46}
$$

2.5.12 Marginalization of probability density functions

Marginalization is another common operation on *pdf*, it consists of eliminating variables of the joint distribution. That is, we can obtain $p\left(x_{1:n\setminus k}\right)$ from $p\left(x_{1:n}\right)$, for example. For continuous random variables it is made by integration, as in Eq. 47.

$$
p\left(x_{1:n\setminus k}\right) = \int_{-\infty}^{+\infty} p\left(x_{1:n}\right) \mathrm{d}x_k \tag{47}
$$

Equation 48 shows that it can be also applied in conditional distributions, in the conditioned variables.

$$p\left(x_j \,|\, x_{1:n\setminus\{j,k\}}\right) = \int_{-\infty}^{+\infty} p\left(x_j, x_k \,|\, x_{1:n\setminus\{j,k\}}\right) \mathrm{d}x_k \tag{48}$$

2.6 The General Bayes Filter

In this Section I will present the General Bayes Filter algorithm, a recursive way to propagate probability density functions of the system estimated state through time. In the following Sections specific Bayes Filters will be derived, such as the Kalman Filter and the Information Filter. The derivation of the algorithm depends on the premise that the system state is complete.

A state x_k is called complete when its knowledge turns the past controls $\{u_1, u_2, \ldots, u_{k-1}\}$ and past measurements $\{z_1, z_2, \ldots, z_{k-1}\}$ useless for predicting the system future state. In other words, no variables prior to step k carry additional information for predicting the future more accurately when one knows the system predate state. Such systems are also known as Markov Chains [2, p. 21]. This statement is put in mathematical terms in Eq. 49.

$$p(\hat{x}_k \,|\, \hat{x}_{k-1}, z_{1:k-1}, u_{1:k}) = p(\hat{x}_k \,|\, x_{k-1}, u_k) \tag{49}$$

The $bel\,(\cdot)$ symbol is a more compact way to write the probability density function of the estimated stated conditioned to the measurements and controls up to step k. The $\bar{bel}(\cdot)$, also known as predicted belief, is almost the same but in this one the kth measurement is not incorporated in the prediction.

$$\begin{cases} bel(\hat{x}_k) & = p\left(\hat{x}_k \,|\, z_{1:k}, u_{1:k}\right) \\ \bar{bel}(\hat{x}_k) & = p\left(\hat{x}_k \,|\, z_{1:k-1}, u_{1:k}\right) \end{cases} \tag{50}$$

The η value is a normalization factor calculated to maintain the probability mass of the distribution equals to one.

First I will prove the measurement update equation, with the Bayes theorem and a little mathematical manipulation.

Algorithm 1 General Bayes Filter

1: **function** BAYES FILTER($bel(\hat{x}_{k-1}), u_k, z_k$)
2: $\quad \bar{bel}(\hat{x}_k) \leftarrow \int p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right) bel(\hat{x}_{k-1})\, \mathrm{d}\hat{x}_{k-1}$ ▷ prediction
3: $\quad bel(\hat{x}_k) \leftarrow \eta\, p(z_k \mid \hat{x}_k)\, \bar{bel}(\hat{x}_k)$ ▷ measurement update
4: $\quad$ **return** $bel(\hat{x}_k)$
5: **end function**

$$
\begin{aligned}
bel(\hat{x}_k) &= p\left(\hat{x}_k \mid z_{1:k}, u_{1:k}\right) \\
&= p\left(\hat{x}_k \mid z_k, z_{1:k-1}, u_{1:k}\right) \\
&= \frac{p\left(z_k \mid \hat{x}_k, z_{1:k-1}, u_{1:k}\right)\; p\left(\hat{x}_k \mid z_{1:k-1}, u_{1:k}\right)}{p\left(z_k \mid z_{k-1}, u_{1:k}\right)} \qquad \text{Applied Eq. 45} \\
&= \frac{p\left(z_k \mid \hat{x}_k, z_{1:k-1}, u_{1:k}\right)\; p\left(\hat{x}_k \mid z_{1:k-1}, u_{1:k}\right)}{p\left(z_k \mid z_{k-1}, u_{1:k}\right)} \\
&= \eta\, p\left(z_k \mid \hat{x}_k, z_{1:k-1}, u_{1:k}\right)\; p\left(\hat{x}_k \mid z_{1:k-1}, u_{1:k}\right) \\
&= \eta\, p\left(z_k \mid \hat{x}_k, z_{1:k-1}, u_{1:k}\right)\; \bar{bel}\left(\hat{x}_k\right)
\end{aligned}
\tag{51}
$$

Using the premise that the estimated state $\hat{x}_k$ is complete, we have that

$$
p(z_k \mid \hat{x}_k, z_{1:k-1}, u_{1:k}) = p(z_k \mid \hat{x}_k) \tag{52}
$$

because no past measurement or control adds new information. We can say that the predicted measurement is conditionally independent of the previous measurements and, controls (including the current), if we know the current state. From the above development, we get Eq. 53, which is the measurement update equation in line 3 of the General Bayes Filter algorithm.

$$
bel\left(\hat{x}_k\right) = \eta\, p(z_k|\hat{x}_k)\, \bar{bel}\left(\hat{x}_k\right) \tag{53}
$$

Now, the prediction step.

$$
\begin{aligned}
\bar{bel}\left(\hat{x}_k\right) &= p\left(\hat{x}_k \mid z_{1:k-1}, u_{1:k}\right) \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k, \hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right) \mathrm{d}\hat{x}_{k-1} \qquad \text{Applied Eq. 48} \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, z_{1:k-1}, u_{1:k}\right)\; p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right) \mathrm{d}\hat{x}_{k-1} \qquad \text{Applied Eq. 46} \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right)\; p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right) \mathrm{d}\hat{x}_{k-1} \qquad \text{Appied Eq. 49}
\end{aligned}
\tag{54}
$$

Now with $p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right)$ we use the assumption that the control in step k does not influence in the estimation of state in step $k-1$. This means that we are assuming that Eq. 55 holds

$$p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right) = p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k-1}\right) \tag{55}$$

with this we have

$$\begin{aligned}
\bar{bel}\left(\hat{x}_k\right) &= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right)\, p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k}\right) \mathrm{d}\hat{x}_{k-1} \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right)\, p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k-1}\right) \mathrm{d}\hat{x}_{k-1} \quad \text{Applied Eq. 55} \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right)\, p\left(\hat{x}_{k-1} \mid z_{1:k-1}, u_{1:k-1}\right) \mathrm{d}\hat{x}_{k-1} \\
&= \int_{-\infty}^{+\infty} p\left(\hat{x}_k \mid \hat{x}_{k-1}, u_k\right)\, bel\left(\hat{x}_k\right) \mathrm{d}\hat{x}_{k-1}
\end{aligned} \tag{56}$$

Equation 56 is the prediction equation in the second line of the Bayes Filter algorithm.

2.7 Kalman Filters

EXPLAIN THAT IN LINEAR CASES THE KALMAN FILTER IS THE OPTIMAL TRACKER

The Kalman Filter (KF) is a type of parametric Bayes Filter, the one we saw in Sec. 2.6. So it is an algorithm to estimate the system state, given the controls and measurements. This filter uses the hypotheses that, the system states and erros, can be represented as a Normal Distribution, which is parameterized by the mean and covariance.

EXPLICAR QUE ESSA ASSUNÇÃO FAZ SENTIDO DE ACOROD COM O TEOREMA DO LIMITE CENTRAL

2.7.1 Kalman Filter Taxonomy

Before getting into the specifics of the Kalman Filter and how it works, I will make a disclaimer about the naming of the different variants of Kalman Filters out there, a good understanding of the meaning of these names can lead the reader into a less cumbersome path when learning about this technique for the first time. More than that, it establishes a precise vocabulary to be used when talking about KF with other researchers, for instance. I will use the convention adopted in [3, p. 151], for this is important to know, first, how dynamic system are classified.

The authors classify systems with respect to the dynamic model, if the system state is a discrete function of time or if it is a continuous function of time. And if the measurements are continuous or discrete functions of time too. So if the system dynamics is continuous and its measurements are discrete, the system is called continuous-discrete. Hence, if the system dynamics is discrete and the measurements are also discrete, the system is called discrete-discrete, and so fourth. Besides the type in time of measurement and dynamics, we have linear and nonlinear dynamics and measurement models. This also has to be taken into consideration when deciding which type of KF to use.

For each combination of the aforementioned characteristics there is a Kalman Filter algorithm to be used. In this text I will derive which I call the "classic" Kalman Filter, which is the KF for *linear discrete-discrete* systems. And also the Extended Kalman Filter (EKF), which is the KF for *nonlinear (continuous/discrete)-discrete* systems.

2.7.2 Kalman Filter (classic)

As I said previously, the Kalman Filter is a type of Bayes Filter, presented in Section 2.6. This means that we could derive it from Algorithm 1 placing a Gaussian distribution in the probability density functions as in [2, p. 45], but I found it very cumbersome and with lots of mathematical tricks, instead, I will derive as the way that is classically done.

Let's say we have a discrete linear time-variant model, like the one in Eq. 28, and since this model is from a physical system, the feedforward matrix $\mathbf{D_k}$ is null. Also, since this is a model from a physical system is reasonable that we have a modeling error, $\boldsymbol{\varepsilon_k}$, since that by definition models are not perfect. Here we will assume that this error can be modeled by a Gaussian distribution with zero mean and covariance $\mathbf{R_k}$, $\boldsymbol{\varepsilon_k} \sim \mathcal{N}(\mathbf{0}, \mathbf{R_k})$, that this errors are not correlated in time (the error in step k is not correlated with the error in step $k+1$, $\mathbb{E}[\boldsymbol{\varepsilon_k}\, \boldsymbol{\varepsilon_{k+1}}] = \mathbf{0}$), and that the error is not correlated with the systems state, therefore $\mathbb{E}[\mathbf{x_k}\, \boldsymbol{\varepsilon_k}] = \mathbf{0}$. Besides the modeling error, we also assume that our measurement $\mathbf{y_k}$ is not perfect too, due to sensor errors. The sensor error, $\boldsymbol{\delta_k}$, is also assumed to be time uncorrelated and have zero mean with covariance $\mathbf{Q_k}$, $\boldsymbol{\delta_k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q_k})$. Those are the basic assumptions needed to guarantee the Kalman Filter's optimality.

Summing it all, our system is described by Eq. 57

$$\begin{aligned} \mathbf{x_k} &= \mathbf{A_k x_{k-1}} + \mathbf{B_k u_k} + \boldsymbol{\varepsilon_k} \\ \mathbf{y_k} &= \mathbf{C_k x_k} + \boldsymbol{\delta_k} \end{aligned} \tag{57}$$

Since we do not know, exactly, our modeling error $\boldsymbol{\varepsilon_k}$, we can predict the mean of the current

state distribution, $\overline{\boldsymbol{\mu}}_{\mathbf{k}}$, using the previous state mean

$$\begin{aligned}
\overline{\boldsymbol{\mu}}_{\mathbf{k}} &= \mathbb{E}\left[\mathbf{A_k x_{k-1}} + \mathbf{B_k u_k} + \boldsymbol{\varepsilon}_{\mathbf{k}}\right] \\
&= \mathbb{E}\left[\mathbf{A_k x_{k-1}} + \mathbf{B_k u_k}\right] + \cancelto{\mathbf{0}}{\mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k}}\right]} \\
&= \mathbf{A_k}\mathbb{E}\left[\mathbf{x_{k-1}}\right] + \mathbf{B_k u_k} \\
&= \mathbf{A_k}\boldsymbol{\mu}_{\mathbf{k-1}} + \mathbf{B_k u_k}
\end{aligned} \tag{58}$$

A good thing to know is how wrong is this prediction in relation to the actual system state. For this, we could calculate the predictin error, $\bar{\mathbf{e}}_{\mathbf{k}} = \mathbf{x_k} - \overline{\boldsymbol{\mu}}_{\mathbf{k}}$, sparseness. In other words, we want to know the prediction error covariance matrix, $\bar{\mathbf{P}}_{\mathbf{k}}$, defined as the expectation in Eq. 59.

$$\bar{\mathbf{P}}_{\mathbf{k}} = \mathbb{E}\left[\bar{\mathbf{e}}_{\mathbf{k}}\,\bar{\mathbf{e}}_{\mathbf{k}}^{\mathbf{T}}\right] \tag{59}$$

The current prediction error can be rewritten as follows:

$$\begin{aligned}
\bar{\mathbf{e}}_{\mathbf{k}} &= \mathbf{x_k} - \overline{\boldsymbol{\mu}}_{\mathbf{k}} \\
&= \mathbf{A_k x_{k-1}} + \cancel{\mathbf{B_k u_k}} + \boldsymbol{\varepsilon}_{\mathbf{k}} - \mathbf{A_k}\boldsymbol{\mu}_{\mathbf{k-1}} + \cancel{\mathbf{B_k u_k}} \\
&= \mathbf{A_k}\left(\mathbf{x_{k-1}} - \boldsymbol{\mu}_{\mathbf{k-1}}\right) + \boldsymbol{\varepsilon}_{\mathbf{k}} \\
&= \mathbf{A_k e_{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k}}
\end{aligned} \tag{60}$$

Using the above "error dynamic system", we can write the prediction error covariance matrix as

$$\begin{aligned}
\bar{\mathbf{P}}_{\mathbf{k}} &= \mathbb{E}\left[\bar{\mathbf{e}}_{\mathbf{k}}\,\bar{\mathbf{e}}_{\mathbf{k}}^{\mathbf{T}}\right] \\
&= \mathbb{E}\left[\left(\mathbf{A_k e_{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k}}\right)\left(\mathbf{A_k e_{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k}}\right)^{\mathbf{T}}\right] \\
&= \mathbb{E}\left[\left(\mathbf{A_k e_{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k}}\right)\left(\mathbf{e}_{\mathbf{k-1}}^{\mathbf{T}}\mathbf{A}_{\mathbf{k}}^{\mathbf{T}} + \boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{T}}\right)\right] \\
&= \mathbf{A_k}\mathbb{E}\left[\mathbf{e_{k-1}}\,\mathbf{e}_{\mathbf{k-1}}^{\mathbf{T}}\right]\mathbf{A}_{\mathbf{k}}^{\mathbf{T}} + \mathbf{A_k}\underbrace{\mathbb{E}\left[\mathbf{e_{k-1}}\,\boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{T}}\right]}_{\mathbf{0}} + \underbrace{\mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k}}\mathbf{e}_{\mathbf{k-1}}^{\mathbf{T}}\right]}_{\mathbf{0}}\mathbf{A}_{\mathbf{k}}^{\mathbf{T}} + \mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k}}\,\boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{T}}\right] \\
&= \mathbf{A_k}\mathbb{E}\left[\mathbf{e_{k-1}}\mathbf{e}_{\mathbf{k-1}}^{\mathbf{T}}\right]\mathbf{A}_{\mathbf{k}}^{\mathbf{T}} + \mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k}}\,\boldsymbol{\varepsilon}_{\mathbf{k}}^{\mathbf{T}}\right] \\
&= \mathbf{A_k}\mathbf{P_{k-1}}\mathbf{A}_{\mathbf{k}}^{\mathbf{T}} + \mathbf{R_k}
\end{aligned} \tag{61}$$

From Eq. 61, we can see that the prediction error covariance grows between time steps, because in each time step a term $\mathbf{R_k}$ (positive definite) is always added. So the predicted mean diverges from the actual system state. What we want to have is a correction term $\boldsymbol{\Psi}_{\mathbf{k}}$ to correct our predicted mean towards the actual one.

$$\boldsymbol{\mu}_{\mathbf{k}} = \overline{\boldsymbol{\mu}}_{\mathbf{k}} + \boldsymbol{\Psi}_{\mathbf{k}} \tag{62}$$

But until now, we just used the system dynamics, when, we also have the system measurement. The idea is, somehow, to use the measurement to correct our prediction, even in presence of the measurement noise $\boldsymbol{\delta_k}$. Let's call by $\mathbf{z_k}$ the measurement we would expect to get from the sensor if the system state was close to the predicted mean.

$$\mathbf{z_k} = \mathbf{C_k}\,\overline{\boldsymbol{\mu}}_\mathbf{k} \tag{63}$$

We can use the error between the actual measurement, $\mathbf{y_k}$, and the predicted one, $\mathbf{z_k}$, in order to correct the predicted mean towards the actual system state.

The Kalman Filter proposes the following correction term, proportional to the difference between $\mathbf{y_k}$ and $\mathbf{z_k}$:

$$\boldsymbol{\Psi_k} = \mathbf{K_k}\,(\mathbf{y_k} - \mathbf{z_k}) \tag{64}$$

so the correction, or state update, would be given by Eq. 65.

$$\boldsymbol{\mu_k} = \overline{\boldsymbol{\mu}}_\mathbf{k} + \mathbf{K_k}\,(\mathbf{y_k} - \mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k}) \tag{65}$$

All the terms from the above Equation, the state update equation, are known, except for the $\mathbf{K_k}$ term, known as Kalman Gain. Further we will use this term to *minimize* the error between the estimated state mean, $\boldsymbol{\mu_k}$ and the actual system state, $\mathbf{x_k}$.

The estimated error, $\boldsymbol{\eta_k}$, between the system state and the updated (or corrected) mean, is given by Eq. 66.

$$\begin{aligned}
\boldsymbol{\eta_k} &= \mathbf{x_k} - \boldsymbol{\mu_k} \\
&= \mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k} - \mathbf{K_k}\,(\mathbf{y_k} - \mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k} - \mathbf{K_k}\,(\mathbf{C_k}\mathbf{x_k} + \boldsymbol{\delta_k} - \mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \mathbf{I}\mathbf{x_k} - \mathbf{I}\,\overline{\boldsymbol{\mu}}_\mathbf{k} - \mathbf{K_k}\,(\mathbf{C_k}\mathbf{x_k} + \boldsymbol{\delta_k} - \mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \mathbf{I}\mathbf{x_k} - \mathbf{I}\,\overline{\boldsymbol{\mu}}_\mathbf{k} - \mathbf{K_k}\mathbf{C_k}\mathbf{x_k} - \mathbf{K_k}\boldsymbol{\delta_k} + \mathbf{K_k}\mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k} \\
&= (\mathbf{I} - \mathbf{K_k}\mathbf{C_k})\,\mathbf{x_k} - (\mathbf{I} - \mathbf{K_k}\mathbf{C_k})\,\overline{\boldsymbol{\mu}}_\mathbf{k} - \mathbf{K_k}\boldsymbol{\delta_k} \\
&= (\mathbf{I} - \mathbf{K_k}\mathbf{C_k})\,(\mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) - \mathbf{K_k}\boldsymbol{\delta_k} \\
&= (\mathbf{I} - \mathbf{K_k}\mathbf{C_k})\,\bar{\mathbf{e}}_\mathbf{k} - \mathbf{K_k}\boldsymbol{\delta_k}
\end{aligned} \tag{66}$$

Now, we can calculate the estimated error covariance, $\mathbf{P_k}$.

$$
\begin{aligned}
\mathbf{P_k} &= \mathbb{E}\left[\boldsymbol{\eta}_\mathbf{k}\,\boldsymbol{\eta}_\mathbf{k}^\mathbf{T}\right] \\
&= \mathbb{E}\left[\left(\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{e}}_\mathbf{k}-\mathbf{K_k}\boldsymbol{\delta}_\mathbf{k}\right)\left(\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{e}}_\mathbf{k}-\mathbf{K_k}\boldsymbol{\delta}_\mathbf{k}\right)^\mathbf{T}\right] \\
&= \mathbb{E}\left[\left(\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{e}}_\mathbf{k}-\mathbf{K_k}\boldsymbol{\delta}_\mathbf{k}\right)\left(\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}-\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\mathbf{K_k^T}\right)\right] \\
&= \begin{array}{l}\mathbb{E}\Big[\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{e}}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}-\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{e}}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\mathbf{K_k^T}-\ldots \\ \qquad \mathbf{K_k}\boldsymbol{\delta}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}+\mathbf{K_k}\boldsymbol{\delta}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\mathbf{K_k^T}\Big]\end{array} \\
&= \begin{array}{l}\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right]\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}-\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\right]\mathbf{K_k^T}-\ldots \\ \mathbf{K_k}\mathbb{E}\left[\boldsymbol{\delta}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right]\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}+\mathbf{K_k}\mathbb{E}\left[\boldsymbol{\delta}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\right]\mathbf{K_k^T}\end{array} \\
&= \left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right]\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}-\mathbf{0}-\mathbf{0}+\mathbf{K_k}\mathbb{E}\left[\boldsymbol{\delta}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\right]\mathbf{K_k^T} \\
&= \left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right]\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}+\mathbf{K_k}\mathbb{E}\left[\boldsymbol{\delta}_\mathbf{k}\boldsymbol{\delta}_\mathbf{k}^\mathbf{T}\right]\mathbf{K_k^T} \\
&= \left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)\bar{\mathbf{P}}_\mathbf{k}\left(\mathbf{I}-\mathbf{K_k}\mathbf{C_k}\right)^\mathbf{T}+\mathbf{K_k}\mathbf{Q_k}\mathbf{K_k^T}
\end{aligned}
\tag{67}
$$

Expanding the above result, we have that:

$$
\mathbf{P_k} = \bar{\mathbf{P}}_\mathbf{k}-\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}\mathbf{K_k^T}-\mathbf{K_k}\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}+\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right]\mathbf{K_k^T}
$$

which is a quadratic in $\mathbf{K_k}$. Another interest thing to note, is that in $\mathbf{P_k}$ diagonal we have the estimated error variances, so its trace is the sum of the Mean Squared Errors between the estimated mean and the actual state. The best we can do is to minimize this quantity, in order to get the best estimation possible.

So, all we have to do is find the minimum of $\mathrm{Tr}(\mathbf{P_k})$ with respect to $\mathbf{K_k}$, since it is the only thing we can change. For this we will differentiate $\mathrm{Tr}(\mathbf{P_k})$ in relation to $\mathbf{K_k}$ using the results of Equations 24, 22 and 25.

$$
\begin{aligned}
\frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}(\mathbf{P_k}) &= \frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}\left(\bar{\mathbf{P}}_\mathbf{k}-\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}\mathbf{K_k^T}-\mathbf{K_k}\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}+\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right]\mathbf{K_k^T}\right) \\
&= \begin{array}{l}\cancelto{\mathbf{0}}{\frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}\left(\bar{\mathbf{P}}_\mathbf{k}\right)}-\frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}\left(\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}\mathbf{K_k^T}\right)-\frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}\left(\mathbf{K_k}\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\right)+\ldots \\ \frac{\partial}{\partial \mathbf{K_k}}\mathrm{Tr}\left(\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right]\mathbf{K_k^T}\right)\end{array} \\
&= -\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}-\bar{\mathbf{P}}_\mathbf{k}^\mathbf{T}\mathbf{C_k^T}+\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right]^\mathbf{T}+\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right] \\
&= 2\mathbf{K_k}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}+\mathbf{Q_k}\right]-2\bar{\mathbf{P}}_\mathbf{k}\mathbf{C_k^T}
\end{aligned}
\tag{68}
$$

Making the above equals to zero, give us the $\mathbf{K_k}$ matrix that minimizes the sum of the mean squared errors between the actual system state and our estimate, leading us to Eq. 69, which is the Kalman Gain equation.

$$\mathbf{K_k} = \bar{\mathbf{P}}_\mathbf{k}\mathbf{C}_\mathbf{k}^\mathbf{T}\left[\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C}_\mathbf{k}^\mathbf{T} + \mathbf{Q_k}\right]^{-1} \tag{69}$$

Algorithm 2 shows the five equations used in the Kalman Filter.

Algorithm 2 Discrete-Discrete Kalman Filter

1: **function** KALMAN FILTER($\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{P_{k-1}}, \mathbf{u_k}, \mathbf{y_k}$)
2: $\quad\overline{\boldsymbol{\mu}}_\mathbf{k} \leftarrow \mathbf{A_k x_{k-1}} + \mathbf{B_k u_k}$ ▷ Linear Prediction
3: $\quad\bar{\mathbf{P}}_\mathbf{k} \leftarrow \mathbf{A_k P_{k-1} A_k^T} + \mathbf{R_k}$ ▷ Covariance Prediction
4: $\quad\mathbf{K_k} \leftarrow \bar{\mathbf{P}}_\mathbf{k}\mathbf{C}_\mathbf{k}^\mathbf{T}\left(\mathbf{C_k}\bar{\mathbf{P}}_\mathbf{k}\mathbf{C}_\mathbf{k}^\mathbf{T} + \mathbf{Q_k}\right)^{-1}$ ▷ Kalman Gain
5: $\quad\boldsymbol{\mu}_\mathbf{k} \leftarrow \overline{\boldsymbol{\mu}}_\mathbf{k} + \mathbf{K_k}\left(\mathbf{y_k} - \mathbf{C_k}\overline{\boldsymbol{\mu}}_\mathbf{k}\right)$ ▷ State Update
6: $\quad\mathbf{P_k} \leftarrow \left(\mathbf{I} - \mathbf{K_k C_k}\right)\bar{\mathbf{P}}_\mathbf{k}\left(\mathbf{I} - \mathbf{K_k C_k}\right)^\mathbf{T} + \mathbf{K_k P_k K_k^T}$ ▷ Estimation Uncertainty
7: $\quad$**return** $\boldsymbol{\mu}_\mathbf{k}, \mathbf{P_k}$
8: **end function**

Equation 70 is the prediction error covariance more commonly encountered in books, instead of Eq. 67. But the later is a better alternative in presence of roundoff error, and is often used in actual software implementation according to [3, p. 73].

$$\mathbf{P_k} = \left(\mathbf{I} - \mathbf{K_k C_k}\right)\bar{\mathbf{P}}_\mathbf{k} \tag{70}$$

2.7.3 Discrete-Discrete Extended Kalman Filter (EKF)

The derivation of the EKF is pretty analogous to the KF. The EKF is all about non-linear systems, in a nutshell it linearizes the non-linear system equations around the estimated and predicted means of the system state distributions, and proceeds as the classic Kalman Filter for linear systems. Let Eq. 71 be the system that we want to estimate the state:

$$\begin{aligned}\mathbf{x_k} &= \boldsymbol{f}(\mathbf{x_{k-1}}, \mathbf{u_k}) + \boldsymbol{\varepsilon_k}\\ \mathbf{y_k} &= \boldsymbol{h}(\mathbf{x_k}) + \boldsymbol{\delta_k}\end{aligned} \tag{71}$$

where $\boldsymbol{\varepsilon_k} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R_k}\right)$ and $\boldsymbol{\delta_k} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{Q_k}\right)$, the assumptions that the errors are not correlated in time or with the system state are still necessary.

In order to use Kalman filtering we have to linearize the non-linear system in 71, for this we use Taylor series expansion and choose the previous state estimate ($\boldsymbol{\mu_{k-1}}$) to linearize about.

For the system model $\boldsymbol{f}()$ we have that

$$\boldsymbol{f}(\mathbf{x_{k-1}}, \mathbf{u_k}) = \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \underbrace{\left.\frac{\partial \boldsymbol{f}(\mathbf{x_{k-1}}, \mathbf{u_k})}{\partial \mathbf{x_{k-1}}}\right|_{\mathbf{x_{k-1}}=\boldsymbol{\mu}_{\mathbf{k-1}}}}_{\mathbf{F_k}}(\mathbf{x_{k-1}} - \boldsymbol{\mu_{k-1}}) + \text{h.o.t.}$$

ignoring the higher order terms we have a first order linear approximation, $\tilde{\boldsymbol{f}}$ to the system model in Eq. 72

$$\tilde{\boldsymbol{f}}(\mathbf{x_{k-1}}, \mathbf{u_k}) = \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \mathbf{F_k}(\mathbf{x_{k-1}} - \boldsymbol{\mu_{k-1}}) \tag{72}$$

the state transition equation in Eq. 71 is then approximated to:

$$\begin{aligned} \mathbf{x_k} &\approx \tilde{\boldsymbol{f}}(\mathbf{x_{k-1}}, \mathbf{u_k}) + \boldsymbol{\varepsilon_k} \\ &\approx \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \mathbf{F_k}(\mathbf{x_{k-1}} - \boldsymbol{\mu_{k-1}}) + \boldsymbol{\varepsilon_k} \end{aligned} \tag{73}$$

Using the above equation as the system model, we can calculate the state predicted by EKF:

$$\begin{aligned} \overline{\boldsymbol{\mu}}_\mathbf{k} &= \mathbb{E}\left[\mathbf{x_k}\right] \\ &= \mathbb{E}\left[\tilde{\boldsymbol{f}}(\mathbf{x_{k-1}}, \mathbf{u_k}) + \boldsymbol{\varepsilon_k}\right] \\ &= \mathbb{E}\left[\boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \mathbf{F_k}(\mathbf{x_{k-1}} - \boldsymbol{\mu_{k-1}})\right] + \cancelto{\mathbf{0}}{\mathbb{E}\left[\boldsymbol{\varepsilon_k}\right]} \\ &= \mathbb{E}\left[\boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k})\right] + \mathbf{F_k}\left(\mathbb{E}\left[\mathbf{x_{k-1}}\right] - \mathbb{E}\left[\boldsymbol{\mu_{k-1}}\right]\right) \end{aligned}$$

at this point we must remember that $\mathbf{u_k}$ and $\boldsymbol{\mu_{k-1}}$ are vectors with a defined value and not random vectors. It means that $\mathbb{E}\left[\boldsymbol{\mu_{k-1}}\right]$ is simply $\boldsymbol{\mu_{k-1}}$ and $\mathbb{E}\left[\boldsymbol{f}\left(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}\right)\right]$ is equal to $\boldsymbol{f}\left(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}\right)$, this follows from the expected value definition in Eq. 31. Then,

$$\begin{aligned} \overline{\boldsymbol{\mu}}_\mathbf{k} &= \mathbb{E}\left[\boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k})\right] + \mathbf{F_k}\left(\mathbb{E}\left[\mathbf{x_{k-1}}\right] - \mathbb{E}\left[\boldsymbol{\mu_{k-1}}\right]\right) \\ &= \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \mathbf{F_k}\left(\underbrace{\mathbb{E}\left[\mathbf{x_{k-1}}\right]}_{\boldsymbol{\mu_{k-1}}} - \boldsymbol{\mu_{k-1}}\right) \\ &= \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) + \mathbf{F_k}\left(\boldsymbol{\mu_{k-1}} - \boldsymbol{\mu_{k-1}}\right) \\ &= \boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k}) \end{aligned} \tag{74}$$

Just as was done with the classic Kalman Filter, we can calculate the prediction error, $\bar{\mathbf{e}}_\mathbf{k} = \mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k}$, sparseness.

$$\bar{\mathbf{P}}_\mathbf{k} = \mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\, \bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right] \tag{75}$$

The current approximate prediction error can be rewritten as follows:

$$\begin{aligned} \bar{\mathbf{e}}_\mathbf{k} &= \mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k} \\ &= \cancel{\boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k})} + \mathbf{F_k}(\mathbf{x_{k-1}} - \boldsymbol{\mu_{k-1}}) + \boldsymbol{\varepsilon_k} - \cancel{\boldsymbol{f}(\boldsymbol{\mu_{k-1}}, \mathbf{u_k})} \\ &= \mathbf{F_k}(\mathbf{x_{k-1}} - \tilde{\boldsymbol{\mu}}_{\mathbf{k-1}}) + \boldsymbol{\varepsilon_k} \\ &= \mathbf{F_k}\mathbf{e_{k-1}} + \boldsymbol{\varepsilon_k} \end{aligned} \tag{76}$$

Using the above error system, we can calculate the prediction error covariance matrix as follows:

$$\begin{aligned}
\bar{\mathbf{P}}_\mathbf{k} &= \mathbb{E}\left[\bar{\mathbf{e}}_\mathbf{k}\,\bar{\mathbf{e}}_\mathbf{k}^\mathbf{T}\right] \\
&= \mathbb{E}\left[\left(\mathbf{F}_\mathbf{k}\mathbf{e}_{\mathbf{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k-1}}\right)\left(\mathbf{F}_\mathbf{k}\mathbf{e}_{\mathbf{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k-1}}\right)^\mathbf{T}\right] \\
&= \mathbb{E}\left[\left(\mathbf{F}_\mathbf{k}\mathbf{e}_{\mathbf{k-1}} + \boldsymbol{\varepsilon}_{\mathbf{k-1}}\right)\left(\mathbf{e}_{\mathbf{k-1}}^\mathbf{T}\mathbf{F}_\mathbf{k}^\mathbf{T} + \boldsymbol{\varepsilon}_{\mathbf{k-1}}^\mathbf{T}\right)\right] \\
&= \mathbf{F}_\mathbf{k}\mathbb{E}\left[\mathbf{e}_{\mathbf{k-1}}\mathbf{e}_{\mathbf{k-1}}^\mathbf{T}\right]\mathbf{F}_\mathbf{k}^\mathbf{T} + \mathbf{F}_\mathbf{k}\underbrace{\mathbb{E}\left[\mathbf{e}_{\mathbf{k-1}}\boldsymbol{\varepsilon}_{\mathbf{k-1}}^\mathbf{T}\right]}_{\mathbf{0}} + \underbrace{\mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k-1}}\mathbf{e}_{\mathbf{k-1}}^\mathbf{T}\right]}_{\mathbf{0}}\mathbf{F}_\mathbf{k}^\mathbf{T} + \mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k-1}}\boldsymbol{\varepsilon}_{\mathbf{k-1}}^\mathbf{T}\right] \\
&= \mathbf{F}_\mathbf{k}\mathbb{E}\left[\mathbf{e}_{\mathbf{k-1}}\mathbf{e}_{\mathbf{k-1}}^\mathbf{T}\right]\mathbf{F}_\mathbf{k}^\mathbf{T} + \mathbb{E}\left[\boldsymbol{\varepsilon}_{\mathbf{k-1}}\boldsymbol{\varepsilon}_{\mathbf{k-1}}^\mathbf{T}\right] \\
&= \mathbf{F}_\mathbf{k}\mathbf{P}_{\mathbf{k-1}}\mathbf{F}_\mathbf{k}^\mathbf{T} + \mathbf{R}_\mathbf{k}
\end{aligned} \tag{77}$$

As we did with the classic KF, it is time to include the measurement information to have a better estimate of the system state. To do so, we need to linearize the measurement model too, it is linearized around the predicted mean ($\overline{\boldsymbol{\mu}}_\mathbf{k}$):

$$\boldsymbol{h}(\mathbf{x}_\mathbf{k}) = \boldsymbol{h}\left(\overline{\boldsymbol{\mu}}_\mathbf{k}\right) + \underbrace{\left.\frac{\partial \boldsymbol{h}(\mathbf{x}_\mathbf{k})}{\partial \mathbf{x}_\mathbf{k}}\right|_{\mathbf{x}_\mathbf{k}=\overline{\boldsymbol{\mu}}_\mathbf{k}}}_{\mathbf{H}_\mathbf{k}}\left(\mathbf{x}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k}\right) + \text{h.o.t.}$$

and, ignoring the higher order terms, we have the first order approximation of the measurement model in Eq. 78.

$$\tilde{\boldsymbol{h}}(\mathbf{x}_\mathbf{k}) = \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbf{x}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) \tag{78}$$

Then the linearized measurement equation in

$$\mathbf{y}_\mathbf{k} \approx \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbf{x}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) + \boldsymbol{\delta}_\mathbf{k} \tag{79}$$

and the expected system measurement would be

$$\begin{aligned}
\mathbf{z}_\mathbf{k} &= \mathbb{E}\left[\boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbf{x}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) + \boldsymbol{\delta}_\mathbf{k}\right] \\
&= \mathbb{E}\left[\boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k})\right] + \mathbf{H}_\mathbf{k}\mathbb{E}\left[(\mathbf{x}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k})\right] + \cancelto{\mathbf{0}}{\mathbb{E}\left[\boldsymbol{\delta}_\mathbf{k}\right]} \\
&= \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbb{E}\left[\mathbf{x}_\mathbf{k}\right] - \overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbb{E}\left[\tilde{\boldsymbol{f}}(\mathbf{x}_{\mathbf{k-1}}, \mathbf{u}_\mathbf{k} + \boldsymbol{\varepsilon}_\mathbf{k})\right] - \overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\mathbb{E}\left[\tilde{\boldsymbol{f}}(\mathbf{x}_{\mathbf{k-1}}, \mathbf{u}_\mathbf{k} + \boldsymbol{\varepsilon}_\mathbf{k})\right] - \overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H}_\mathbf{k}(\overline{\boldsymbol{\mu}}_\mathbf{k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) \\
&= \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k})
\end{aligned} \tag{80}$$

The EKF update equation is given by:

$$\begin{aligned}
\boldsymbol{\mu}_\mathbf{k} &= \overline{\boldsymbol{\mu}}_\mathbf{k} + \mathbf{K}_\mathbf{k}\left(\mathbf{y}_\mathbf{k} - \mathbf{z}_\mathbf{k}\right) \\
&= \overline{\boldsymbol{\mu}}_\mathbf{k} + \mathbf{K}_\mathbf{k}\left(\mathbf{y}_\mathbf{k} - \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k})\right)
\end{aligned} \tag{81}$$

The EKF estimate error is:

$$
\begin{aligned}
\boldsymbol{\eta}_k &= \mathbf{x}_k - \boldsymbol{\mu}_k \\
&= \mathbf{x}_k - (\overline{\boldsymbol{\mu}}_k + \mathbf{K}_k (\mathbf{y}_k - \boldsymbol{h}(\overline{\boldsymbol{\mu}}_k))) \\
&= \mathbf{x}_k - \left(\overline{\boldsymbol{\mu}}_k + \mathbf{K}_k \left(\cancel{\boldsymbol{h}(\overline{\boldsymbol{\mu}}_k)} + \mathbf{H}_k(\mathbf{x}_k - \overline{\boldsymbol{\mu}}_k) + \boldsymbol{\delta}_k - \cancel{\boldsymbol{h}(\overline{\boldsymbol{\mu}}_k)}\right)\right) \\
&= \mathbf{x}_k - \mathbf{K}_k\mathbf{H}_k\mathbf{x}_k - \overline{\boldsymbol{\mu}}_k + \mathbf{K}_k\mathbf{H}_k\overline{\boldsymbol{\mu}}_k + \mathbf{K}_k\boldsymbol{\delta}_k \\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{x}_k - (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\overline{\boldsymbol{\mu}}_k + \mathbf{K}_k\boldsymbol{\delta}_k \\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,(\mathbf{x}_k - \overline{\boldsymbol{\mu}}_k) + \mathbf{K}_k\boldsymbol{\delta}_k \\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\bar{\mathbf{e}}_k + \mathbf{K}_k\boldsymbol{\delta}_k
\end{aligned}
\tag{82}
$$

Again, we want to calculate the estimate error covariance matrix in order to minimize it with respect to the Kalman Gain. Some steps are omitted below because it is really analogous to what was done in the classic KF in Eq. 67.

$$
\begin{aligned}
\mathbf{P}_k &= \mathbb{E}\left[\boldsymbol{\eta}_k\boldsymbol{\eta}_k^{\mathbf{T}}\right] \\
&= \mathbb{E}\left[((\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\bar{\mathbf{e}}_k + \mathbf{K}_k\boldsymbol{\delta}_k)\,((\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\bar{\mathbf{e}}_k + \mathbf{K}_k\boldsymbol{\delta}_k)^{\mathbf{T}}\right] \\
&= \mathbb{E}\left[((\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\bar{\mathbf{e}}_k + \mathbf{K}_k\boldsymbol{\delta}_k)\,(\bar{\mathbf{e}}_k^{\mathbf{T}}(\mathbf{I} - \mathbf{H}_k\mathbf{K}_k)^{\mathbf{T}} + \boldsymbol{\delta}_k^{\mathbf{T}}\mathbf{K}_k^{\mathbf{T}})\right] \\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbb{E}\left[\bar{\mathbf{e}}_k\bar{\mathbf{e}}_k^{\mathbf{T}}\right](\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathbf{T}} + \mathbf{K}_k\mathbb{E}\left[\boldsymbol{\delta}_k\right]\boldsymbol{\delta}_k^{\mathbf{T}}\mathbf{K}_k^{\mathbf{T}} \\
&= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\overline{\mathbf{P}}_k(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathbf{T}} + \mathbf{K}_k\mathbf{Q}_k\mathbf{K}_k^{\mathbf{T}}
\end{aligned}
\tag{83}
$$

Expanding the terms of the above equation, we obtain a quadratic form in $\mathbf{K}_k$:

$$
\mathbf{P}_k = \overline{\mathbf{P}}_k + \mathbf{K}_k\left(\mathbf{H}_k\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}} + \mathbf{Q}_k\right)\mathbf{K}_k^{\mathbf{T}} - \mathbf{K}_k\mathbf{H}_k\overline{\mathbf{P}}_k - \overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}}\mathbf{K}_k^{\mathbf{T}} \tag{84}
$$

Then we minimize the trace of $\mathbf{P}_k$ with respect to $\mathbf{K}_k$, this way, we obtain the minimum sum of the mean squared error of the estimate.

$$
\begin{aligned}
\frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}(\mathbf{P}_k) &= \frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\overline{\mathbf{P}}_k + \mathbf{K}_k\left(\mathbf{H}_k\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}} + \mathbf{Q}_k\right)\mathbf{K}_k^{\mathbf{T}} - \mathbf{K}_k\mathbf{H}_k\overline{\mathbf{P}}_k - \overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}}\mathbf{K}_k^{\mathbf{T}}\right) \\
&= \cancel{\frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\overline{\mathbf{P}}_k\right)}^{\mathbf{0}} + \frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\mathbf{K}_k\left(\mathbf{H}_k\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}} + \mathbf{Q}_k\right)\mathbf{K}_k^{\mathbf{T}}\right) + \dots \\
&\quad - \frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\mathbf{K}_k\mathbf{H}_k\overline{\mathbf{P}}_k\right) - \frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}}\mathbf{K}_k^{\mathbf{T}}\right) \\
&= \frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\mathbf{K}_k\left(\mathbf{H}_k\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}} + \mathbf{Q}_k\right)\mathbf{K}_k^{\mathbf{T}}\right) - 2\,\frac{\partial}{\partial\mathbf{K}_k}\operatorname{Tr}\left(\mathbf{K}_k\mathbf{H}_k\overline{\mathbf{P}}_k\right)
\end{aligned}
\tag{85}
$$

Applying the same principles used in Eq. 68 we have that the Kalman Gain for the EKF is:

$$
\mathbf{K}_k = \overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}}\left[\mathbf{H}_k\overline{\mathbf{P}}_k\mathbf{H}_k^{\mathbf{T}} + \mathbf{Q}_k\right]^{-1} \tag{86}
$$

Equations 74, 77, 81, 83 and 86 define the Extended Kalman Filter. They look very similar to the ones used in the classic Kalman Filter, but the later is guaranteed the best possible estimate, while the EKF estimate is not. This happens because to use the EKF we have to linearize the system model around the state estimate $\boldsymbol{\mu_k}$ and the measurement model around the state prediction $\overline{\boldsymbol{\mu}}_\mathbf{k}$.

In summary, the EKF is the Kalman Filter applied to the linear system in Eq. 87, the jacobian notation $\mathbf{G_k}$ and $\mathbf{H_k}$ is a little trick, because the first is the jacobian of the system model with respect to the previous *estimated* mean ($\tilde{\boldsymbol{\mu}}_{\mathbf{k-1}}$), while the latter is the jacobian of the measurement model with respect to the current *predicted* mean ($\overline{\boldsymbol{\mu}}_\mathbf{k}$).

$$\begin{aligned} \mathbf{x_k} &\approx \boldsymbol{f}(\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{u_k}) + \mathbf{F_k}(\mathbf{x_{k-1}} - \boldsymbol{\mu}_{\mathbf{k-1}}) + \boldsymbol{\varepsilon_k} \\ \mathbf{y_k} &\approx \boldsymbol{h}(\bar{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H_k}(\mathbf{x_k} - \overline{\boldsymbol{\mu}}_\mathbf{k}) + \boldsymbol{\delta_k} \end{aligned} \tag{87}$$

Algorithm 3 shows the five EKF equations in order.

Algorithm 3 Discrete-Discrete Extended Kalman Filter

1: **function** EXTENDED KALMAN FILTER($\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{P_{k-1}}, \mathbf{u_k}, \mathbf{y_k}$)
2: $\quad\overline{\boldsymbol{\mu}}_\mathbf{k} \leftarrow \boldsymbol{g}(\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{u_k})$ ▷ Nonlinear Prediction
3: $\quad\overline{\mathbf{P}}_\mathbf{k} \leftarrow \mathbf{G_k P_{k-1} G_k^T} + \mathbf{R_k}$ ▷ Covariance Prediction
4: $\quad\mathbf{K_k} \leftarrow \overline{\mathbf{P}}_\mathbf{k}\mathbf{H_k^T}\left(\mathbf{H_k}\overline{\mathbf{P}}_\mathbf{k}\mathbf{H_k^T} + \mathbf{Q_k}\right)^{-1}$ ▷ Kalman Gain
5: $\quad\boldsymbol{\mu_k} \leftarrow \overline{\boldsymbol{\mu}}_\mathbf{k} + \mathbf{K_k}\left(\mathbf{y_k} - \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k})\right)$ ▷ State Update
6: $\quad\mathbf{P_k} \leftarrow (\mathbf{I} - \mathbf{K_k H_k})\,\overline{\mathbf{P}}_\mathbf{k}$ ▷ Estimation Uncertainty
7: $\quad$**return** $\boldsymbol{\mu_k}, \mathbf{P_k}$
8: **end function**

2.8 Information Filters

Information Filters are more suitable for multi-robot system according to [2, p. 78].

TODO

2.8.1 Classic Information Filter

Before getting into the specifics of the information filter, it is important to introduce the Information Vector (Eq. 88) and, the Information Matrix (Eq. 89).

$$\boldsymbol{\xi} = \mathbf{P^{-1}}\boldsymbol{\mu} \tag{88}$$

$$\mathbf{\Omega} = \mathbf{P^{-1}} \tag{89}$$

Algorithm 4 Information Filter

function INFORMATION FILTER($\boldsymbol{\xi_{k-1}}, \mathbf{\Omega_{k-1}}, \mathbf{u_k}, \mathbf{z_k}$)
  $\bar{\mathbf{\Omega}}_\mathbf{k} \leftarrow \left(\mathbf{A_k}\, \mathbf{\Omega}_{\mathbf{k-1}}^{-1}\, \mathbf{A_k^T} + \mathbf{R_k}\right)^{-1}$
  $\bar{\boldsymbol{\xi}}_\mathbf{k} \leftarrow \bar{\mathbf{\Omega}}_\mathbf{k}\, \left(\mathbf{A_k}\, \mathbf{\Omega}_{\mathbf{k-1}}^{-1}\, \boldsymbol{\xi}_{\mathbf{k-1}} + \mathbf{B_k}\, \mathbf{u_k}\right)$
  $\mathbf{\Omega_k} \leftarrow \bar{\mathbf{\Omega}}_\mathbf{k} + \mathbf{C_k^T}\, \mathbf{Q_k^{-1}}\, \mathbf{C_k}$
  $\boldsymbol{\xi}_k \leftarrow \bar{\boldsymbol{\xi}}_k + \mathbf{C_k^T}\, \mathbf{Q_k^{-1}}\, \mathbf{z_k}$
  return $\boldsymbol{\xi_k}, \mathbf{\Omega_k}$
end function

2.8.2 Extended Information Filter

Algorithm 5 Extended Information Filter

function EXTENDED INFORMATION FILTER($\boldsymbol{\xi_{k-1}}, \mathbf{\Omega_{k-1}}, \mathbf{u_k}, \mathbf{y_k}$)
  $\boldsymbol{\mu}_{\mathbf{k-1}} \leftarrow \mathbf{\Omega}_{\mathbf{k-1}}^{-1}\, \boldsymbol{\xi}_{\mathbf{k-1}}$
  $\bar{\mathbf{\Omega}}_\mathbf{k} \leftarrow \left(\mathbf{G_k}\, \mathbf{\Omega}_{\mathbf{k-1}}^{-1}\, \mathbf{G_k^T} + \mathbf{R_k}\right)^{-1}$
  $\bar{\boldsymbol{\xi}}_\mathbf{k} \leftarrow \bar{\mathbf{\Omega}}_\mathbf{k}\, \boldsymbol{g}(\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{u_k})$
  $\overline{\boldsymbol{\mu}}_\mathbf{k} \leftarrow \boldsymbol{g}(\boldsymbol{\mu}_{\mathbf{k-1}}, \mathbf{u_k})$
  $\mathbf{\Omega_k} \leftarrow \bar{\mathbf{\Omega}}_\mathbf{k} + \mathbf{H_k^T}\, \mathbf{Q_k^{-1}}\, \mathbf{H_k}$
  $\boldsymbol{\xi}_k \leftarrow \bar{\boldsymbol{\xi}}_k + \mathbf{H_k^T}\, \mathbf{Q_k^{-1}}\, [\mathbf{y_k} - \boldsymbol{h}(\overline{\boldsymbol{\mu}}_\mathbf{k}) + \mathbf{H_k}\, \overline{\boldsymbol{\mu}}_\mathbf{k}]$
  return $\boldsymbol{\xi_k}, \mathbf{\Omega_k}$
end function

2.8.3 Sparse Extended Information Filter

2.9 Geometry

2.9.1 Cross product and the skew-symmetric matrix

Here I introduce a "matrix form" of calculating the cross product between two vectors. This form will be useful later to calculate the rotation matrix given an axis and the amount of rotation around that axis.

The following, in Equation 90, is the determinant form to calculate the cross product between two vectors $\vec{\omega}$ and $\vec{p}$.

$$\vec{\omega} \times \vec{p} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \omega_x & \omega_y & \omega_z \\ p_x & p_y & p_z \end{vmatrix} = \begin{vmatrix} \omega_y & \omega_z \\ p_y & p_z \end{vmatrix} \hat{i} - \begin{vmatrix} \omega_x & \omega_z \\ p_x & p_z \end{vmatrix} \hat{j} + \begin{vmatrix} \omega_x & \omega_y \\ p_x & p_y \end{vmatrix} \hat{k}$$
$$\vec{\omega} \times \vec{p} = \begin{bmatrix} \omega_y p_z - \omega_z p_y \\ \omega_z p_x - \omega_x p_z \\ \omega_x p_y - \omega_y p_x \end{bmatrix} \tag{90}$$

The above was just for remembering purposes. First of all, a skew-symmetric matrix is a square matrix that its transpose equals its negative, in other words, $A^T = -A$. The 3×3 skew-symmetric matrices can be used to express cross products.

The skew-symmetric matrix of a vector $\vec{\omega}$ ($[\vec{\omega}]$) is defined in Equation 91.

$$[\vec{\omega}] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{91}$$

Now with this, we can redefine the cross product in Equation 90 by a product between the skew-symmetric matrix of the vector $\vec{\omega}$ and the vector $\vec{p}$, as shown in Equation 92.

$$\vec{\omega} \times \vec{p} = [\vec{\omega}]\,\vec{p} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \omega_y p_z - \omega_z p_y \\ \omega_z p_x - \omega_x p_z \\ \omega_x p_y - \omega_y p_x \end{bmatrix} \tag{92}$$

2.9.2 The Rodrigues' rotation formula

In this section we will derive the Rodrigues' rotation formula. This formula gives the recipe to calculate the rotation matrix that describes a rotation of θ radians around an arbitrary unit axis $\hat{\omega}$, represented in Figure 2.

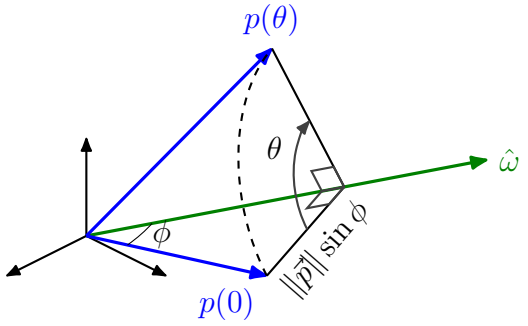

Figure 2: Rotation of the position vector $\vec{p}$ around the $\hat{w}$ axis for θ radians.

Lets imagine a point placed at the tip of vector $\vec{p}$ at the instant 0 ($p(0)$) travelling along the dashed arc and achieving the position denoted by $p(\theta)$. One, of many ways, to achieve this rotation is imagining that the point p travels with constant angular velocity of $\omega = 1\,\text{rad/s}$ during the time period θ seconds. Let ϕ be the constant angle between the rotation axis $\hat{\omega}$ and the vector $\vec{p}(t)$, $t \in [0, \theta]$. Regarding to the point's linear velocity, its norm is equivalent to $\|\vec{p}\| \sin\phi$, because it travels along a circle of radius $\|\vec{p}\| \sin\phi$ with unit angular velocity. The point's velocity direction must be tangent to the circular trajectory, therefore orthogonal to $\vec{p}(t)$. All of this can be addressed by Eq. 93.

$$\dot{p} = \hat{\omega} \times \vec{p}(t) \tag{93}$$

Using the skew-symmetric form, Eq. 92, we have:

$$\dot{p} = [\hat{\omega}]\, p(t) \tag{94}$$

Equation 94 is a ordinary differential equation (ODE) just like 99, therefore, its solution is of the form of Eq. 100.

$$p(t) = e^{[\hat{\omega}]t}\, p(0)$$

Replacing t for θ, we end up with Eq. 95.

$$p(\theta) = e^{[\hat{\omega}]\theta}\, p(0) \tag{95}$$

Now, using the Taylor series expansion, Eq. 2, and the fact that $[\hat{\omega}]^3 = -[\hat{\omega}]$, we have

$$\begin{aligned} e^{[\hat{\omega}]\theta} &= \mathbf{I} + [\hat{\omega}]\,\theta + \frac{[\hat{\omega}]^2\,\theta^2}{2!} + \frac{\cancel{[\hat{\omega}]^3}^{-[\hat{\omega}]}\,\theta^3}{3!} + \frac{\cancel{[\hat{\omega}]^4}^{-[\hat{\omega}]^2}\,\theta^4}{4!} + \frac{\cancel{[\hat{\omega}]^5}^{[\hat{\omega}]}\,\theta^5}{5!} + \cdots \\ &= \mathbf{I} + [\hat{\omega}] \underbrace{\left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \cdots\right)}_{\sin(\theta)} + [\hat{\omega}]^2 \underbrace{\left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \frac{\theta^8}{8!} + \cdots\right)}_{1-\cos(\theta)} \\ &= \mathbf{I} + [\hat{\omega}]\,\sin(\theta) + (1 - \cos(\theta))\,[\hat{\omega}]^2 \end{aligned} \tag{96}$$

which is known as the Rodrigues' rotation formula.

2.9.3 Coordinate system

A coordinate system $\{s\}$ is composed of an origin and a basis. The basis is a minimum set of vectors which can generate any other vector through their linear combination. It is important

to emphasize that any set of n linear-independent vectors can describe the entirety of a $\mathbb{R}^n$ space, in other words, there is an infinity number of bases in any space.

$$\{s\} = \{(o_x, o_y, o_z), \{\vec{s_1}, \vec{s_2}, \vec{s_3}\}\} \tag{97}$$

All the coordinate systems used in this work have an orthonormal basis. A orthonormal basis have all its vectors of length one and they all are perpendicular to each other.

2.9.4 TODO: Changing coordinates of a Twist

A **twist** is the combination of an angular and a linear velocity. In the plane it is represented by the following triplet:

$$\mathcal{S} = (\omega, v_x, v_y) \tag{98}$$

2.10 Differential equations

"*Since Newton, mankind has come to realize that the laws of physics are always expressed in the language of differential equations.*" - Steven Strongatz

2.10.1 The simplest, yet important, linear differential equation: $\frac{\partial}{\partial t}x(t) = ax(t)$

The above differential equation is one, if not, of the most used equations in control systems. Therefore is very important to know its solution. This same equation can be found in other forms, like the following:

$$\dot{x} = ax \tag{99a}$$

$$x'(t) = ax(t) \tag{99b}$$

Before the math starts, it is important to think about the nature of the function $x(t)$, what other function we know that is *very similar* to its derivative? Well, moving forward, we can solve this by manipulating the expression and then integrating it.

$$\frac{x'(t)}{x(t)} = a$$

The key insight here, is assume that $x(t) > 0$ and realize that the right side of the above equation is the derivative of $\ln(x(t))$. After that, all we need to do is integrate both sides.

$$\frac{\partial}{\partial t}(\ln x(t)) = a$$
$$\int \frac{\partial}{\partial t}(\ln x(t))\, dt = \int a\, dt$$
$$\ln x(t) = at + c$$
$$\exp(\ln x(t)) = \exp(at + c)$$
$$x(t) = \exp(at) \cdot \cancelto{c_0}{\exp(c)}$$
$$x(t) = e^{at}\, c_0$$

The above development provides a family of solutions to $x(t)$, but if one is pursuing an unique solution, knowing a point $(t_0, x(t_0)$ that belongs to the curve is a must. Because of the following:

$$x(t_0) = e^{at_0}\, c_0$$
$$c_0 = \frac{x(t_0)}{e^{at_0}}$$
$$= e^{-at_0} x(t_0)$$

It is very common to write $x(0)$ as x_{t_0}. So we have the general solution:

$$\dot{x} = ax \implies x(t) = e^{a(t-t_0)}\, x_{t_0} \tag{100}$$

So far I showed that Equation 100 is a solution to the Equation 99. But is it the only one? Lets suppose that there is another solution $y(t)$, and lets also suppose a function $z(t) = e^{-at}\, y(t)$. Now we derive $z(t)$, of course.

$$z'(t) = \frac{\partial}{\partial t}(e^{-at}\, y(t))$$
$$z'(t) = -ae^{-at}y(t) + \cancelto{ay(t)}{y'(t)}e^{-at}$$
$$z'(t) = ay(t) \underbrace{(-e^{-at} + e^{-at})}_{0}$$
$$z'(t) = 0$$

The above development shows that z(t) is a constant, because its time derivative is zero. So we have that:

$$z(t) = C = e^{-at}y(t),\ C \in \mathbb{R}$$
$$y(t) = Ce^{at}$$

So the supposed new solution $y(t)$, is indeed of the form Ce^{at}, therefore, nothing new in the front. With that it is shown that Equation 100 is the only solution to 99.

3 The differential drive model dynamics

In this section two dynamic models will be derived for the differential drive robot, and I will show which one is the most adequate and why.

3.1 The differential drive model

A differential drive robot is a two-wheeled robot, the wheels are aligned and their velocities can be different from each other. When both wheels have the same velocity, the robot moves forward or backwards. If they have the same speed but in opposite directions, the robot spins around the wheels axle's midpoint, clockwise or counterclockwise. Any configuration different than these makes the robot follow a curved trajectory.

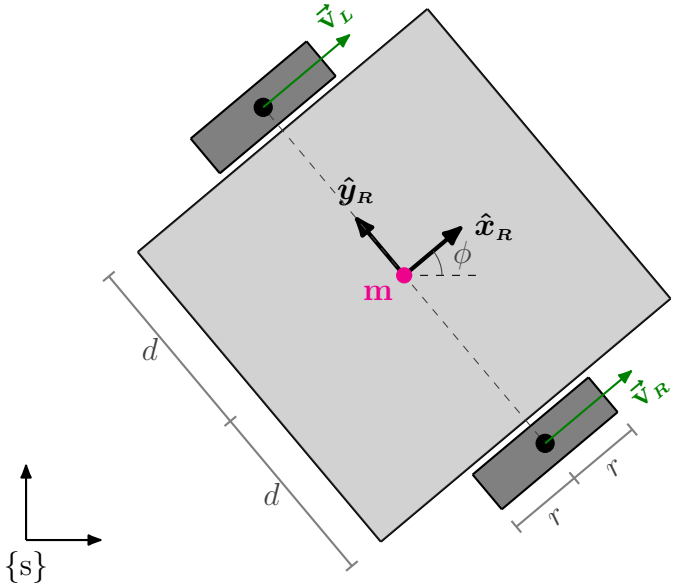

Figure 3: Differential drive robot schematic top view. The z-axis is pointing out of the sheet. The robot's moving coordinates frame is placed in the middle point between the wheels axle. {s} is a static frame of reference.

3.2 The differential drive kinematics

Let's first define the robot state by a state vector $\mathbf{x}$, Eq. 101, composed by the robot's heading and position.

$$\mathbf{x} = \begin{bmatrix} \phi \\ x \\ y \end{bmatrix} \tag{101}$$

The goal is to derive a model that relates the wheel velocities (inputs), in rad/s, to the vector representing the robot's state. Something with the shape of Eq. 102. Where $\mathbf{u}$ is the input vector.

$$\dot{\mathbf{x}} = \boldsymbol{g}\left(\begin{bmatrix} g_1(\mathbf{x}, \mathbf{u}) \\ g_2(\mathbf{x}, \mathbf{u}) \\ g_3(\mathbf{x}, \mathbf{u}) \end{bmatrix}\right) \tag{102}$$

The input vector is the pair (u_L, u_R), the angular velocities of the left and right wheels, respectively. The velocities signs are defined in a way that when both are positive, the robot moves in the x-axis positive direction of its own coordinate frame. The wheels linear velocities is given by Eq. 103, where r is the wheels radius.

$$\begin{cases} \|\vec{\mathrm{v}}_L\| = u_L \cdot r \\ \|\vec{\mathrm{v}}_R\| = u_R \cdot r \end{cases} \tag{103}$$

From now on, all the calculations will be made in the robot coordinates frame. The axle middle point is the robot position. One also must know the relation between linear and angular speeds in Eq. 104.

$$V = \omega R \tag{104}$$

where

V is the linear speed

ω is rotation speed, aka the angular speed

R is the radius of rotation

First, I will derive the robot linear velocity in function of the wheels velocities. By Figure 3 it is clear that any wheel displacement will make the middle point translates in the $\hat{y}_R$ direction, therefore we have that $\dot{y} = 0$. The linear velocity in the $\hat{x}_R$ direction is given by Eq. 105,

$$\begin{aligned} \dot{x} &= \frac{1}{2}\left(\mathrm{v}_R + \mathrm{v}_L\right) \\ &= \frac{r}{2}\left(u_R + u_L\right) \end{aligned} \tag{105}$$

to see that, just observe that when one wheel is stopped and the other is spinning the middle point linear velocity will be half of the point in the wheel, because the middle point will describe a circle with half the radius of the circle described by the spinning wheel point around the stopped wheel. Then, just sum the contribution of each wheel.

Now the angular velocity, just as with the linear velocity, I will analyse the contribution of each wheel and then, sum the effects. When the right wheel spins, the midpoint, $\boldsymbol{m}$ in Fig. 3, rotates in the counterclockwise direction around the left wheel with linear velocity $\frac{\mathrm{v}_R}{2}$ and radius d. Therefore the angular velocity contribution of the right wheel is $\dot{\phi}_R = \frac{\mathrm{v}_R}{2\,d}$, in the counterclockwise direction. The contribution of the left wheel is analogous, but when it spins forward, the middle point moves clockwise around the right wheel, since I am adopting the right-handed coordinates system, the clockwise sense is negative and the counterclockwise is positive. The robot angular velocity is given by Eq. 106

$$\begin{aligned}\dot{\phi} &= \dot{\phi}_R + \dot{\phi}_L \\ &= \frac{\mathrm{v}_R}{2\,d} - \frac{\mathrm{v}_L}{2\,d} \\ &= \frac{r}{2\,d}\left(u_R - u_L\right)\end{aligned} \tag{106}$$

Wrapping everything matrix form, we end up with Eq. 107

$$\begin{bmatrix}\dot{\phi} \\ \dot{x} \\ \dot{y}\end{bmatrix}_{\{\mathrm{R}\}} = \begin{bmatrix}-r/2d & r/2d \\ r/2 & r/2 \\ 0 & 0\end{bmatrix}\begin{bmatrix}u_L \\ u_R\end{bmatrix} \tag{107}$$

Equation 107 is expressed in the robot frame, that is moving with it. To express it with respect to a static frame, as depicted in Fig. 3, we need to write the linear velocities in terms of the static frame base vectors. Equation 108 gives the basis change matrix from the robot frame to the static frame.

$$\boldsymbol{R}_{\mathbf{SR}} = \begin{bmatrix}1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi\end{bmatrix} \tag{108}$$

One must remember that the z-axis of both frames is pointing out of the sheet, that's why the first column is $\begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$. Using Eq. 107 and 108 we obtain Eq. 109 which is the robot dynamic model in the static frame. Note that this is a nonlinear model, since we have sine and cosine function of the robot heading, one of the system's states.

$$\begin{aligned}\begin{bmatrix}\dot{\phi}\\ \dot{x}\\ \dot{y}\end{bmatrix}_{\{S\}} &= \boldsymbol{R}_{\mathbf{SR}}\begin{bmatrix}\dot{\phi}\\ \dot{x}\\ \dot{y}\end{bmatrix}_{\{R\}}\\ &= \begin{bmatrix}-r/2d & r/2d\\ (r/2)\cos\phi & (r/2)\cos\phi\\ (r/2)\sin\phi & (r/2)\sin\phi\end{bmatrix}\begin{bmatrix}u_L\\ u_R\end{bmatrix}\end{aligned} \tag{109}$$

To use this model with the Kalman Filter, we need to linearize it around the predicted state and then discretize the model.

3.3 From The Continuouns To The Discrete System

When one has a continuous system and wants to discretize it there is two ways: The first is to first linearize the model about a nominal trajectoty or equilibrium point and then discretize exactly, resulting in a linear and discrete model. The other one is to just make an approximate discretization, using forward Euler method, for example. In the following both methods will be derived for the differential drive model.

3.3.1 Linearization and Discretization

Model Linearization

One way of linearizing a nonlinear dynamic model is through its Taylor Series Expansion around some nominal trajectory $(\mathbf{x}, \bar{\mathbf{u}})$. Equation 110 is the first order Taylor approximation of a function $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ around $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$.

$$\boldsymbol{g}(\mathbf{x}, \mathbf{u}) \approx \boldsymbol{g}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) + \frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\bigg|_{(\bar{\mathbf{x}}, \bar{\mathbf{u}})}(\mathbf{x} - \bar{\mathbf{x}}) + \frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}\bigg|_{(\bar{\mathbf{x}}, \bar{\mathbf{u}})}(\mathbf{u} - \bar{\mathbf{u}}) \tag{110}$$

In the Extended Kalman Filter, the model is linearized around the estimated state $\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}$ and the input $\mathbf{u_k}$. Here the hat symbol is used to refer to the estimated state, please do not confuse with when it is used to represent unit vectors. Rewriting Eq. 110 around $(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k})$, we have Eq. 111.

$$\dot{\mathbf{x}} \approx \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k}) + \underbrace{\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\bigg|_{(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k})}}_{\mathbf{A_c}(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k})}(\mathbf{x} - \hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}) + \underbrace{\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}\bigg|_{(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k})}}_{\mathbf{B_c}(\hat{\mathbf{x}}_{\mathbf{k},\mathbf{k}}, \mathbf{u_k})}(\mathbf{u} - \mathbf{u_k}) \tag{111}$$

Using the model in Eq. 109, we get the following state matrix, Eq. 112.

$$\mathbf{A_c}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) = \left.\frac{\partial \mathbf{g}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\right|_{(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k})} = \begin{bmatrix} 0 & 0 & 0 \\ -\frac{r}{2}\sin\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & 0 & 0 \\ \frac{r}{2}\cos\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & 0 & 0 \end{bmatrix} \tag{112}$$

Now, all that is left is the model discretization.

Model Discretization

A common assumption when control by computer is employed, is that the input signal is constant between the control interval, also known as sampling time.

$$\frac{\partial \mathbf{u}(t)}{\partial t} = 0,\ kT_s \leq t < (k+1)T_s \tag{113}$$

This fact enables us to derive a linear discrete dynamics from the linear continuous dynamics in Eq. 111. In the following, I will refer to the matrix $\mathbf{A_c}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k})$ as $\mathbf{A_c}$ and to $\mathbf{B_c}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k})$ as $\mathbf{B_c}$,

$$\begin{aligned}
\dot{\mathbf{x}}(t) &\approx \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) + \mathbf{A_c}(\mathbf{x}(t) - \hat{\mathbf{x}}_{\mathbf{k,k}}) + \mathbf{B_c}(\mathbf{u}(t) - \mathbf{u_k}) \\
\dot{\mathbf{x}} &\approx \mathbf{A_c}\mathbf{x}(t) + \mathbf{B_c}\mathbf{u}(t) + \underbrace{\boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k}}_{\text{constant}} \\
\dot{\mathbf{x}} - \mathbf{A_c}\mathbf{x}(t) &\approx \mathbf{B_c}\mathbf{u}(t) + \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k} \\
e^{-\mathbf{A_c}t} \cdot (\dot{\mathbf{x}} - \mathbf{A_c}\mathbf{x}(t)) &\approx e^{-\mathbf{A_c}t} \cdot (\mathbf{B_c}\mathbf{u}(t) + \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k}) \\
\frac{\partial}{\partial t}\left(e^{-\mathbf{A_c}t}\,\mathbf{x}(t)\right) &\approx e^{-\mathbf{A_c}t} \cdot (\mathbf{B_c}\mathbf{u}(t) + \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k}) \\
\int_{kT_s}^{(k+1)T_s} \frac{\partial}{\partial t}\left(e^{-\mathbf{A_c}t}\,\mathbf{x}(t)\right) dt &\approx \int_{kT_s}^{(k+1)T_s} e^{-\mathbf{A_c}t} \cdot (\mathbf{B_c}\mathbf{u}(t) + \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k})dt
\end{aligned}$$

$$\begin{aligned}
e^{-\mathbf{A_c}(k+1)T_s}\,\mathbf{x}((k+1)T_s) - \ldots \\
- e^{-\mathbf{A_c}kT_s}\,\mathbf{x}(kT_s)
\end{aligned} \approx \begin{aligned}
&\int_{kT_s}^{(k+1)T_s} e^{-\mathbf{A_c}t}\mathbf{B_c}\ \underbrace{\mathbf{u}(t)}_{\text{constant}}\ dt\ + \ldots \\
&\int_{kT_s}^{(k+1)T_s} e^{-\mathbf{A_c}t}(\boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k})dt
\end{aligned}$$

$$\begin{aligned}
e^{-\mathbf{A_c}(k+1)T_s}\,\mathbf{x}((k+1)T_s) \approx\ & e^{-\mathbf{A_c}kT_s}\,\mathbf{x}(kT_s) + \left(\int_{kT_s}^{(k+1)T_s} e^{-\mathbf{A_c}t}dt\right)(\mathbf{B_c}\,\mathbf{u}(kT_s)\ + \ldots \\
& \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \mathbf{B_c}\mathbf{u_k})
\end{aligned}$$

for the sake of readability, I will write $\mathbf{x}((k+1)T_s)$ as $\mathbf{x_{k+1}}$, $\mathbf{x}(kT_s)$ as $\mathbf{x_k}$, and $\mathbf{u}(kT_s)$ as $\mathbf{u_k}$,

$$e^{-\mathbf{A_c}(k+1)T_s}\,\mathbf{x_{k+1}} \approx e^{-\mathbf{A_c}kT_s}\,\mathbf{x_k} + \left(\int_{kT_s}^{(k+1)T_s} e^{-\mathbf{A_c}t}dt\right)\left(\cancel{\mathbf{B_c}\,\mathbf{u_k}} + \boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}} - \cancel{\mathbf{B_c}\,\mathbf{u_k}}\right)$$

multiplying both sides by the inverse of $e^{-\mathbf{A_c}(k+1)T_s}$, which is $e^{\mathbf{A_c}(k+1)T_s}$ (see Eq. **??**),

$$\begin{aligned}
\mathbf{x_{k+1}} &\approx e^{\mathbf{A_c}T_s}\,\mathbf{x_k} + \left(\int_{kT_s}^{(k+1)T_s} e^{\mathbf{A_c}((k+1)T_s - t)}dt\right)\left(\boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}}\right) \\
&\approx e^{\mathbf{A_c}T_s}\,\mathbf{x_k} + \left(\int_0^{T_s} e^{\mathbf{A_c}\tau}d\tau\right)\left(\boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}}\right) \\
&\approx \underbrace{e^{\mathbf{A_c}T_s}}_{\mathbf{A_d}}\mathbf{x_k} + \left(\int_0^{T_s} e^{\mathbf{A_c}\tau}d\tau\right)\left(\boldsymbol{g}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k}) - \mathbf{A_c}\hat{\mathbf{x}}_{\mathbf{k,k}}\right)
\end{aligned} \tag{114}$$

Some things to be reminded.

T_s is the interval between two control inputs

$\mathbf{x_k}$ is the same as $\mathbf{x}(kT_s)$

$\mathbf{u_k}$ is the same as $\mathbf{u}(kT_s)$

$\hat{\mathbf{x}}_{\mathbf{k,k}}$ is the filter estimation of the state $\mathbf{x}_k$

$\mathbf{A_c}$ is the $\mathbf{A_c}(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k})$ matrix from Eq. 111, which is the jacobian of $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ with respect to $\mathbf{x}$ in $(\hat{\mathbf{x}}_{\mathbf{k,k}}, \mathbf{u_k})$

With the continuous state matrix obtained in 112, using the matrix exponential definition in Eq. 2 and noting that this particular matrix squared is null, Eq. 115 gives the discrete state matrix.

$$e^{\mathbf{A_c}T_s} = \exp\left(\begin{bmatrix} 0 & 0 & 0 \\ -\frac{r}{2}\sin\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & 0 & 0 \\ \frac{r}{2}\cos\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & 0 & 0 \end{bmatrix} T_s\right) = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{r}{2}\sin\hat{\phi}_{k,k}(u_{L_k} + u_{R_k})T_s & 1 & 0 \\ \frac{r}{2}\cos\hat{\phi}_{k,k}(u_{L_k} + u_{R_k})T_s & 0 & 1 \end{bmatrix} \tag{115}$$

Equation 116 is the integral of $\exp(\mathbf{A_c}\tau)$ from 0 to T_s.

$$\int_0^{T_s} e^{\mathbf{A_c}\tau}d\tau = \begin{bmatrix} T_s & 0 & 0 \\ -\frac{r\,T_s^2}{4}\sin\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & T_s & 0 \\ \frac{r\,T_s^2}{4}\cos\hat{\phi}_{k,k}(u_{L_k} + u_{R_k}) & 0 & T_s \end{bmatrix} \tag{116}$$

3.3.2 Discretization through approximation

Another method to discretize a continuous model is to use the forward Euler approximation, this approximation can be obtained by the manipulation of the Taylor series expansion of a function $y(\cdot)$ around some point t_0.

$$\begin{aligned} y(t) &= \sum_{n=0}^{\infty} \left.\frac{\partial^n y(t)}{\partial t^n}\right|_{t_0} \frac{(t-t_0)^n}{n!} \\ &= y(t_0) + y'(t_0)\,(t-t_0) + \textit{high order terms} \end{aligned} \tag{117}$$

Ignoring the high order terms, we get the following approximation:

$$y(t) \approx y(t_0) + y'(t_0)\,(t-t_0) \tag{118}$$

Rewriting the above equation with $h = t - t_0$, we obtain the first order approximation to the derivative at the point t_0

$$\begin{aligned} y(t_0+h) &\approx y(t_0) + y'(t_0)\,h \\ y'(t_0) &\approx \frac{y(t_0+h) - y(t_0)}{h} \end{aligned} \tag{119}$$

So, with Eq. 119 we can take the continuous time model in Eq. 102 and obtain its discrete first order approximation

$$\underbrace{\dot{\mathbf{x}}(kT_s)}_{\boldsymbol{g}(\mathbf{x}(kT_s),\mathbf{u}(kT_s))} \approx \frac{\mathbf{x}(kT_s+T_s) - \mathbf{x}(kT_s)}{T_s}$$

writing $\mathbf{x}(kT_s)$ and $\mathbf{x}(kT_s+T_s)$ as $\mathbf{x_k}$ and $\mathbf{x_{k+1}}$, we get Eq. 120

$$\begin{aligned} \boldsymbol{g}(\mathbf{x_k}, \mathbf{u_k}) &\approx \frac{\mathbf{x_{k+1}} - \mathbf{x_k}}{T_s} \\ \mathbf{x_{k+1}} &\approx \mathbf{x_k} + T_s\,\boldsymbol{g}(\mathbf{x_k}, \mathbf{u_k}) \end{aligned} \tag{120}$$

where T_s is the sampling period or step size.

With the above expression, we get the following discrete differential drive kinematics from Eq. 107

$$
\begin{bmatrix} \phi_{k+1} \\ x_{k+1} \\ y_{k+1} \end{bmatrix} \approx \begin{bmatrix} \phi_k \\ x_k \\ y_k \end{bmatrix} + T_s \begin{bmatrix} -r/2d & r/2d \\ (r/2)\cos\phi_k & (r/2)\cos\phi_k \\ (r/2)\sin\phi_k & (r/2)\sin\phi_k \end{bmatrix} \begin{bmatrix} u_{L_k} \\ u_{R_k} \end{bmatrix}
$$

$$
\mathbf{x_{k+1}} \approx \begin{bmatrix} \phi_k \\ x_k \\ y_k \end{bmatrix} + \begin{bmatrix} -r/2d & r/2d \\ (r/2)\cos\phi_k & (r/2)\cos\phi_k \\ (r/2)\sin\phi_k & (r/2)\sin\phi_k \end{bmatrix} \begin{bmatrix} \cancel{T_s u_{L_k}} \to \Delta\theta_{L_k} \\ \cancel{T_s u_{R_k}} \to \Delta\theta_{R_k} \end{bmatrix} \tag{121}
$$

$$
\mathbf{x_{k+1}} \approx \begin{bmatrix} \phi_k - \dfrac{r}{2d}\Delta\theta_{L_k} + \dfrac{r}{2d}\Delta\theta_{R_k} \\ x_k + \dfrac{r}{2}\cos(\phi_k)\Delta\theta_{L_k} + \dfrac{r}{2}\cos(\phi_k)\Delta\theta_{R_k} \\ y_k + \dfrac{r}{2}\sin(\phi_k)\Delta\theta_{L_k} + \dfrac{r}{2}\sin(\phi_k)\Delta\theta_{R_k} \end{bmatrix}
$$

3.4 Another differential drive model

$$
\mathbf{x_{k+1}} = \begin{bmatrix} \phi_k \\ x_k \\ y_k \end{bmatrix} + \begin{bmatrix} \dfrac{r}{2d}\Delta\theta_{L_k} - \dfrac{r}{2d}\Delta\theta_{R_k} \\ \dfrac{\Delta\theta_{R_k} + \Delta\theta_{L_k}}{\Delta\theta_{R_k} - \Delta\theta_{L_k}}\left(\sin(\phi_k + \alpha) - \sin(\phi_k)\right) \\ \dfrac{\Delta\theta_{R_k} + \Delta\theta_{L_k}}{\Delta\theta_{R_k} - \Delta\theta_{L_k}}\left(-\cos(\phi_k + \alpha) + \cos(\phi_k)\right) \end{bmatrix} \tag{122}
$$

4 The Range-Bearing measurements

4.1 The Range-Bearing measurement direct model

Equation 123 is the laser scanner coordinates in the map frame.

$$
\begin{cases} x_l = x + d\,\cos\phi \\ y_l = y + d\,\sin\phi \end{cases} \tag{123}
$$

Equation 124 is the range-bearing measurement model of the *jth* landmark. That is, the measurement that should be obtained if the estimated system state was exactly $\mathbf{x}$, in other

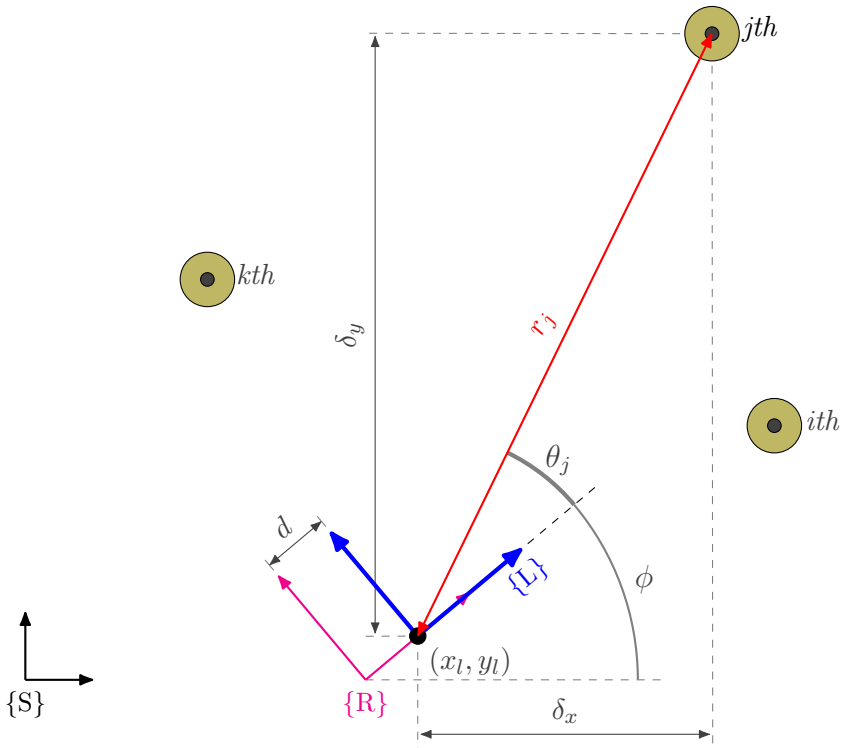

Figure 4: The Range-Bearing measurement scheme. The robot frame is represented in magenta, the sensor frame in blue. The measurement (r_j, θ_j) refers to the *jth* landmark in the map.

words, if the estimated state was equal the true state.

$$\boldsymbol{h_j}(\mathbf{x}) = \begin{bmatrix} r_j \\ \theta_j \end{bmatrix} = \begin{bmatrix} \sqrt{(m_{j,x} - x_l)^2 + (m_{j,y} - y_l)^2} \\ \arctan\left(\dfrac{m_{j,y} - y_l}{m_{j,x} - x_l}\right) - \phi \end{bmatrix} \tag{124}$$

4.1.1 Direct Model Linearization

In order to use the non-linear model described in the previous section with the Kalman or Information filters, it must be linearized. Equation 125 is the jacobian of Eq. 124.

$$\frac{\partial}{\partial \mathbf{x}} \boldsymbol{h_j}(\mathbf{x}) = \begin{bmatrix} \begin{bmatrix} \dfrac{d}{r_j}\left(\delta_x \sin\phi - \delta_y \cos\phi\right) & -\dfrac{\delta_x}{r_j} & -\dfrac{\delta_y}{r_j} \\ -\dfrac{d}{r_j^2}\left(\delta_y \sin\phi + \delta_x \cos\phi\right) - 1 & \dfrac{\delta_y}{r_j^2} & -\dfrac{\delta_x}{r_j^2} \end{bmatrix} & \mathbf{0}_{2\times 2(j-1)} & \begin{bmatrix} \dfrac{\delta_x}{r_j} & \dfrac{\delta_y}{r_j} \\ -\dfrac{\delta_y}{r_j^2} & \dfrac{\delta_x}{r_j^2} \end{bmatrix} & \mathbf{0}_{2\times 2(M-j)} \end{bmatrix} \tag{125}$$

4.2 The Range-Bearing measurement inverse model

A measurement $\mathbf{y}$ is defined as:

$$\mathbf{y} = \begin{bmatrix} r + \delta_r \\ \theta + \delta_\theta \end{bmatrix} \tag{126}$$

and the expected measurement $\mathbf{z}$ is

$$\mathbf{z} = \mathbb{E}\left[\mathbf{y}\right] = \mathbb{E}\left[\begin{bmatrix} r + \delta_r \\ \theta + \delta_\theta \end{bmatrix}\right] = \begin{bmatrix} r \\ \theta \end{bmatrix} \tag{127}$$

The range bearing

$$\boldsymbol{g}(\mathbf{x}, \mathbf{z}) = \begin{bmatrix} m_x \\ m_y \end{bmatrix} = \begin{bmatrix} x_l + r\cos(\phi + \theta) \\ y_l + r\sin(\phi + \theta) \end{bmatrix} \tag{128}$$

$$\frac{\partial}{\partial \mathbf{x}} \boldsymbol{g}(\mathbf{x}, \mathbf{z}) = \begin{bmatrix} \begin{bmatrix} -d\sin\phi - r\sin(\phi + \theta) & 1 & 0 \\ d\cos\phi + r\cos(\phi + \theta) & 0 & 1 \end{bmatrix} & \mathbf{0}_{2\times n-3} \end{bmatrix}_{2\times n} \tag{129}$$

$$\frac{\partial}{\partial \mathbf{z}} \boldsymbol{g}(\mathbf{x}, \mathbf{z}) = \begin{bmatrix} \cos(\phi + \theta) & -r\sin(\phi + \theta) \\ \sin(\phi + \theta) & r\cos(\phi + \theta) \end{bmatrix}_{2\times 2} \tag{130}$$

5 EKF SLAM

TODO

5.1 Landmark Initialization

When using the Extended Kalman Filter to solve the SLAM problem, it is necessary to initialize new landmarks as the robot moves through the environment and discovers new unseen places. In practice the state vector size of the EKF becomes dynamic, it grows when a new landmark is incorporated into the map. This operation of incorporating new map data into the state vector is exclusive of the EKF-SLAM algorithm, such operation is not performed in a regular use of the standard Extended Kalman Filter.

$$x^*_{1:n+2} = \boldsymbol{\sigma}(x_{1:n}, \mathbf{z_j}) \tag{131}$$

The $\boldsymbol{\sigma}(\cdot)$ function above, takes the current state vector of size n and the measurement $\mathbf{z_j}$ of the new *jth* landmark and returns a new state vector of size $n + 2$ with the new landmark (x, y) position, calculated through the inverse measurement model $\boldsymbol{g}(\cdot)$, appended at the end.

When the robot sees a new landmark, we can expect that the estimated error of this new landmark position has to account the robot positioning error and the sensor error. It is not fair to initialize it with ∞ like [2, p. 317] says so, because the robot already have a guess about where it is when it makes the new observation, so the error can not be ∞.

The aforementioned $\boldsymbol{\sigma}(\cdot)$ function appends the new landmark $(m_{j,x}, m_{j,y})$ position into the map, using the inverse measurement model (Section 4.2). It is defined in Eq. 132

$$\boldsymbol{\sigma}(\mathbf{x_k}, \mathbf{z_j}) = \begin{bmatrix} \mathbf{x_k} \\ \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j}) \end{bmatrix} \tag{132}$$

As $\boldsymbol{g}(\cdot)$ is nonlinear, we need to linearize it in order to use with EKF. We will linearize around $(\boldsymbol{\mu_k}, \mathbf{y_j})$, that is, the current estimate mean and the new landmark measurement.

$$\begin{aligned} \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j}) &= \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) + \underbrace{\frac{\partial \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j})}{\partial \mathbf{x_k}}\bigg|_{\mathbf{x_k}=\boldsymbol{\mu_k}}}_{\boldsymbol{\Sigma_x}} (\mathbf{x_k} - \boldsymbol{\mu_k}) + \underbrace{\frac{\partial \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j})}{\partial \mathbf{z_j}}\bigg|_{\mathbf{z_j}=\mathbf{y_j}}}_{\boldsymbol{\Sigma_z}} (\mathbf{z_j} - \mathbf{y_j}) + \text{h.o.t.} \\ &= \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) + \boldsymbol{\Sigma_x}\boldsymbol{\eta_k} + \boldsymbol{\Sigma_z}\boldsymbol{\delta_j} + \text{h.o.t.} \end{aligned} \tag{133}$$

Then the approximated append function is:

$$\tilde{\boldsymbol{\sigma}}(\mathbf{x_k}, \mathbf{z_j}) \approx \begin{bmatrix} \mathbf{x_k} \\ \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) + \boldsymbol{\Sigma_x}\boldsymbol{\eta_k} + \boldsymbol{\Sigma_z}\boldsymbol{\delta_j} \end{bmatrix} \tag{134}$$

The expectation of the append operation is:

$$\begin{aligned} \boldsymbol{\mu}_\mathbf{k}^* &= \mathbb{E}\left[\tilde{\boldsymbol{\sigma}}(\mathbf{x_k}, \mathbf{z_j})\right] \\ &= \begin{bmatrix} \mathbb{E}\left[\mathbf{x}\right] \\ \mathbb{E}\left[\boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j})\right] + \boldsymbol{\Sigma_x}\cancelto{\mathbf{0}}{\mathbb{E}\left[\boldsymbol{\eta_k}\right]} + \boldsymbol{\Sigma_z}\cancelto{\mathbf{0}}{\mathbb{E}\left[\boldsymbol{\delta_j}\right]} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{\mu_k} \\ \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) \end{bmatrix} \end{aligned} \tag{135}$$

The append error is

$$\begin{aligned} \boldsymbol{\alpha} &= \mathbf{x}_\mathbf{k}^* - \boldsymbol{\mu}_\mathbf{k}^* \\ &= \begin{bmatrix} \mathbf{x_k} \\ \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j}) \end{bmatrix} - \begin{bmatrix} \boldsymbol{\mu_k} \\ \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) \end{bmatrix} = \begin{bmatrix} \mathbf{x_k} - \boldsymbol{\mu_k} \\ \boldsymbol{g}(\mathbf{x_k}, \mathbf{z_j}) - \boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j}) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{\eta_k} \\ \cancel{\boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j})} + \boldsymbol{\Sigma_x}\boldsymbol{\eta_k} + \boldsymbol{\Sigma_z}\boldsymbol{\delta_j} - \cancel{\boldsymbol{g}(\boldsymbol{\mu_k}, \mathbf{y_j})} \end{bmatrix} \qquad \text{Applied Eq. 134} \\ &= \begin{bmatrix} \boldsymbol{\eta_k} \\ \boldsymbol{\Sigma_x}\boldsymbol{\eta_k} + \boldsymbol{\Sigma_z}\boldsymbol{\delta_j} \end{bmatrix} \end{aligned} \tag{136}$$

The append error covariance is given by

$$\begin{aligned}\mathbf{P}_{\mathbf{k}}^{*} &= \mathbb{E}\left[\boldsymbol{\alpha}\,\boldsymbol{\alpha}^{\mathbf{T}}\right] \\ &= \mathbb{E}\left[\begin{bmatrix}\boldsymbol{\eta}_{\mathbf{k}} \\ \boldsymbol{\Sigma}_{\mathbf{x}}\boldsymbol{\eta}_{\mathbf{k}} + \boldsymbol{\Sigma}_{\mathbf{z}}\boldsymbol{\delta}_{\mathbf{j}}\end{bmatrix}\begin{bmatrix}\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}} & \boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{x}}^{\mathbf{T}} + \boldsymbol{\delta}_{\mathbf{j}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{\mathbf{T}}\end{bmatrix}\right] \\ &= \begin{bmatrix}\mathbb{E}\left[\boldsymbol{\eta}_{\mathbf{k}}\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\right] & \mathbb{E}\left[\boldsymbol{\eta}_{\mathbf{k}}(\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{x}}^{\mathbf{T}} + \boldsymbol{\delta}_{\mathbf{j}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{\mathbf{T}})\right] \\ \mathbb{E}\left[(\boldsymbol{\Sigma}_{\mathbf{x}}\boldsymbol{\eta}_{\mathbf{k}} + \boldsymbol{\Sigma}_{\mathbf{z}}\boldsymbol{\delta}_{\mathbf{j}})\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\right] & \mathbb{E}\left[(\boldsymbol{\Sigma}_{\mathbf{x}}\boldsymbol{\eta}_{\mathbf{k}} + \boldsymbol{\Sigma}_{\mathbf{z}}\boldsymbol{\delta}_{\mathbf{j}})(\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{x}}^{\mathbf{T}} + \boldsymbol{\delta}_{\mathbf{j}}^{\mathbf{T}}\boldsymbol{\Sigma}_{\mathbf{z}}^{\mathbf{T}})\right]\end{bmatrix}\end{aligned} \tag{137}$$

with the fact that $\mathbb{E}\left[\boldsymbol{\delta}_{\mathbf{j}}\,\boldsymbol{\eta}_{\mathbf{k}}^{\mathbf{T}}\right] = \mathbb{E}\left[\boldsymbol{\eta}_{\mathbf{k}}\,\boldsymbol{\delta}_{\mathbf{j}}^{\mathbf{T}}\right] = \mathbf{0}$, we end up with:

$$\mathbf{P}_{\mathbf{k}}^{*} = \begin{bmatrix}\mathbf{P}_{\mathbf{k}} & \mathbf{P}_{\mathbf{K}}\boldsymbol{\Sigma}_{\mathbf{x}}^{\mathbf{T}} \\ \boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{P}_{\mathbf{k}} & \boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{P}_{\mathbf{k}}\boldsymbol{\Sigma}_{\mathbf{x}}^{\mathbf{T}} + \boldsymbol{\Sigma}_{\mathbf{z}}\mathbf{Q}_{\mathbf{j}}\boldsymbol{\Sigma}_{\mathbf{z}}^{\mathbf{T}}\end{bmatrix} \tag{138}$$

that is, the new covariance matrix is the old covariance matrix $\mathbf{P_k}$ appended with the cross-covariances ($\boldsymbol{\Sigma_x}\mathbf{P_k}$ and $\mathbf{P_k}\boldsymbol{\Sigma_x^T}$) and the new landmark position covariance, highlighted in magenta. The new landmark position covariance is a combination of the system covariance, up to its insertion, and the sensor error covariance.

6 Appendix

6.1 Properties of the inverse matrix

1. For any $n \times n$ invertible matrices $\mathbf{A}$ and $\mathbf{B}$, the following holds true:

$$\mathbf{A^{-1}B^{-1}} = (\mathbf{BA})^{-1} \tag{139}$$

this is true because

$$\begin{aligned}\mathbf{A^{-1}B^{-1}BA} &= \mathbf{I} \\ \mathbf{A^{-1}B^{-1}}\,(\mathbf{BA}) &= \mathbf{I} \\ \mathbf{A^{-1}B^{-1}}\,(\mathbf{BA})\,(\mathbf{BA})^{-1} &= \mathbf{I}\,(\mathbf{BA})^{-1} \\ \mathbf{A^{-1}B^{-1}} &= (\mathbf{BA})^{-1}\end{aligned}$$

6.2 Sherman/Morrison formula

The Sherman/Morrison formula, also known as the specialized inversion lemma, is stated below from [2, p. 50]

Lemma 1. *For any invertible quadratic matrices* $\mathbf{R}$ *and* $\mathbf{Q}$ *and any matrix* $\mathbf{P}$ *with appropriate dimensions, the following holds true:*

$$\left(\mathbf{R}+\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right)^{-1}=\mathbf{R}^{-1}-\mathbf{R}^{-1}\mathbf{P}\left(\mathbf{Q}^{-1}+\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\right)^{-1}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1} \tag{140}$$

assuming that all above matrices can be inverted as stated.

Proof. Define $\mathbf{\Psi}=\left(\mathbf{Q}^{-1}+\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\right)^{-1}$. It suffices to show that

$$\left(\mathbf{R}^{-1}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\right)\left(\mathbf{R}+\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right)=\mathbf{I}$$

this is shown through a series of manipulations:

$$\begin{aligned}
&=\mathbf{R}^{-1}\mathbf{R}+\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{R}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{I}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}-\mathbf{R}^{-1}\mathbf{P}\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{Q}^{-1}\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{Q}^{-1}\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right] \qquad (141)\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\left(\mathbf{Q}^{-1}-\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\right)\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\left(\mathbf{Q}^{-1}-\mathbf{P}^{\mathbf{T}}\mathbf{R}^{-1}\mathbf{P}\right)\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{\Psi}\mathbf{\Psi}^{-1}\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}+\mathbf{R}^{-1}\mathbf{P}\left[\mathbf{Q}\mathbf{P}^{\mathbf{T}}-\mathbf{Q}\mathbf{P}^{\mathbf{T}}\right]\\
&=\mathbf{I}
\end{aligned}$$

□

References

[1] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[2] Josh Bongard. Probabilistic robotics. sebastian thrun, wolfram burgard, and dieter fox.(2006, mit press.) 647 pages, 2006.

[3] Frank L Lewis, Lihua Xie, and Dan Popa. *Optimal and robust estimation: with an introduction to stochastic control theory*. CRC press, 2017.

[4] K.M. Lynch and F.C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.

[5] Sebastian Thrun. Probabilistic algorithms in robotics. *Ai Magazine*, 21(4):93–93, 2000.