

**SAKARYA UNIVERSITY  
INSTITUTE OF SCIENCE AND TECHNOLOGY**

**MOBILE ROBOT ODOMETRIC LOCALIZATION  
USING DECENTRALIZED KALMAN FILTER**

**M.Sc. THESIS**

**N'djadjo Romuald KOUAKOU**

**Department : ELECTRICAL AND ELECTRONICS  
ENGINEERING**  
**Field of Science : ELECTRONICS**  
**Supervisor : Prof. Dr. Aşkın DEMİRKOL**

**July 2018**

SAKARYA UNIVERSITY  
INSTITUTE OF SCIENCE AND TECHNOLOGY


MOBILE ROBOT ODOMETRIC LOCALIZATION  
USING DECENTRALIZED KALMAN FILTER

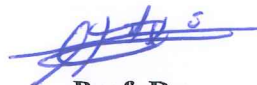
M.Sc. THESIS

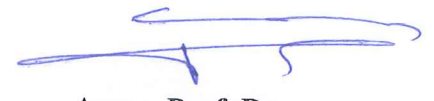
N'djadjo Romuald KOUAKOU

Department : ELECTRICAL AND ELECTRONICS  
ENGINEERING  
Field of Science : ELECTRONICS  
Supervisor : Prof. Dr. Aşkın DEMİRKOL

This thesis has been accepted unanimously by the examination committee on  
02.07.2018

  
Prof. Dr.  
Aşkın DEMİRKOL  
Head of Jury

  
Prof. Dr.  
Ali Fuat BOZ  
Jury Member

  
Assoc. Prof. Dr.  
Cenk YAVUZ  
Jury Member

## DECLARATION

I declare that all the data in this thesis was obtained by myself in academic rules, all visual and written information and results were presented in accordance with academic and ethical rules, there is no distortion in the presented data, in case of utilizing other people's works they were refereed properly to scientific norms, the data presented in this thesis has not been used in any other thesis in this university or in any other university.



N'djado Romuald KOUAKOU

02.07.2018

## **ACKNOWLEDGEMENT**

First of all, I would like to give thanks and glory to my God, who is inspiring and giving me an opportunity to meet my supervisor Prof. Dr. Aşkın DEMİRKOL. I would like to express my warm gratitude towards my Supervisor for his precious support, for the confidence that he accorded to me by accepting to work with me during this thesis. I thank him for his valuable advice and consistent encouragement throughout the course of my thesis. I also thank him for his vision and great effort that he offered me.

Forever grateful to the Turkish government and Turkish Scholarship Team for the scholarship which granted the opportunity to acquire scientific and academic knowledge. In the same way, I would like to acknowledge also the University of Sakarya, especially the Department of Electrical and Electronics Engineering, all of the Teaching staff for their effort put into my training.

I would like to express my appreciation to all those people who assisted me during this experience. Especially my friends, my new family in Turkish, the African community association and all of Prof. Dr. Aşkın DEMİRKOL's PhD students for their help.

Finally, I dedicate this thesis to my family: my dear father, my mother, sisters and brothers who supported me morally despite the distance that separate us. Without them I could never be where I am today.

## TABLE OF CONTENTS

ACKNOWLEDGEMENT .....	i
TABLE OF CONTENTS .....	ii
LIST OF SYMBOLS AND ABBREVIATIONS .....	v
LIST OF FIGURES .....	viii
LIST OF TABLES .....	x
SUMMARY .....	xi
ÖZET .....	xii
CHAPTER 1.	
INTRODUCTION .....	1
1.1. Thesis Introduction .....	1
1.2. Problem Statement.....	5
1.3. Thesis Objectives.....	6
1.4. Organization of the Thesis.....	7
CHAPTER 2.	
FIRST PART OF METHODS AND MATERIALS .....	8
2.1. Introduction.....	8
2.2. Sensors. ....	9
2.2.1. Wheel encoder.....	9
2.2.2. Telemeter.....	9
2.2.2.1. Radar.....	10
2.2.2.2. Ultrasound .....	10
2.2.2.3. Lidar .....	11
2.3. Camera. ....	13
2.3.1. The affine camera.....	13
2.3.1.1. The orthographic projection camera.....	15

2.3.1.2. The scaling orthographic projection camera .....	16
2.3.1.3. The weak projection camera.....	16
2.3.2. Camera transformation and geometry.....	17
2.3.2.1. Translation and rotation.....	17
2.3.2.2. Coordinates transformation .....	19
2.3.2.3. World and camera coordinates transformation.....	21
2.3.3. Geometry projection of the camera.....	22
2.3.4. The fundamental matrix .....	24
2.3.5. The random sample consensus algorithm (RANSAC) .....	25
2.4. Visual Odometry.....	27
2.4.1. Presentation.....	27
2.4.2. Selection of the features.....	28
2.5. Mobile Robot Locomotion Modelling.....	30
2.5.1. Kinematic of the two wheeled mobile robot.....	30
2.5.2. Inverse kinematic for mobile robot.....	32
2.5.3. Differential steering speed for four wheeled mobile robot .....	33
2.5.4. Dynamics model of four wheeled mobile robot.....	36
2.6. Literature Review .....	38
2.6.1. History of the robotics .....	38
2.6.2. Vehicle ego-motion estimation (Visual Odometry) .....	39
2.7. Summary.....	40

### CHAPTER 3.

SECOND PART OF METHODS AND MATERIALS.....	41
3.1. Introduction.....	41
3.2. Data Fusion Algorithm .....	42
3.2.1. Multi sensor system .....	43
3.2.2. Centralized data fusion architecture.....	44
3.2.3. Hierarchical data fusion architecture .....	44
3.2.4. Decentralized data fusion architecture.....	45
3.3. Kalman Filter .....	46
3.3.1. Standard Kalman Filter .....	47

3.3.2. Extended Kalman Filter (EKF).....	49
3.3.3. Decentralized Kalman Filter (DKF) .....	51
3.3.3.1. Presentation of DKF .....	51
3.3.3.2. Problem formulation.....	52
3.4. Literature Review .....	55
3.5. Summary.....	57
CHAPTER 4.	
EXPERIMENTS AND RESULTS.....	58
4.1. Part A: Experiments and Implementation With Matlab .....	58
4.1.1. Presentation of Simscape .....	58
4.1.2. Presentation of SolidWorks.....	59
4.1.3. Implementation with Matlab.....	61
4.1.3.1. DC motor modelling with Simulink.....	63
4.1.3.2. First local Kalman Filter with two wheels encoders ....	65
4.1.3.3. Second local Kalman Filter with a gyroscope.....	67
4.1.3.4. The global Kalman Filter.....	68
4.2. Part B: Results and Analysis.....	69
4.2.1. Analysis of the first local Kalman Filter .....	70
4.2.2. Analysis of the second local Kalman Filter .....	72
4.2.3. Analysis of the global Kalman Filter .....	73
CHAPTER 5.	
CONCLUSION AND SUGGESTION .....	76
5.1. Conclusion .....	76
5.2. Suggestion .....	76
REFERENCES .....	78
ANNEX.....	85
RESUME .....	87

## LIST OF SYMBOLS AND ABBREVIATIONS

BC	: Before Christ
CAD	: Computer Assisted Design
CCD	: Charge Coupled Device
CMOS	: Complementary Metal Oxide Semiconductor
d	: Robot wheels diameter
D	: Robot rear base
DC	: Direct Current
DFK	: Decentralized Kalman Filter
DLT	: Direct Linear Transformation
DOF	: Degree of freedom
EKF	: Extended Kalman Filter
F	: Fundamental matrix
F(t), A(t)	: Transition matrix
G(k), G <sub>i</sub> (k)	: Process disturbance matrix
G(t) , B(t)	: Control matrix
H	: homographic matrix
HD	: High Definition
H(k), H(t), C(t), C(k)	: Measurement or observation matrix
I	Identity matrix
ICR	: Instantaneous Center of rotation
J <sub>robot</sub>	: Robot inertia
k	: Cameras intrinsic or internal parameter
K, K <sub>if</sub> , K <sub>i</sub>	: Kalman gain matrix
KF	: Kalman Filter
LIDAR	: Light Detection And Ranging
LME	: Likelihood Maximum Estimation
LMS	: Least Means Square



LO	: LIDAR Odometry
LQG	: Linear Quadratic Gaussian
m	: Robot masse
MCL	: Monte Carlos Localization
Mi(k)	: Data fusion matrix
NASA	: National Aeronautics and Space Administration
P, P <sub>A</sub>	: Camera matrix
PF	: Particle Filter
PID	: Proportional Integral Derivative
Q, Q <sub>t</sub> , Q(k)	: White Gaussian process noise matrix
R	: Rotation matrix
R, R(t), R(k)	: White Gaussian measurement noise matrix
RADAR	: Radio Detection And Ranging
RANSAC	: Random Sample Consensus
Robotrom	: Robot Romi
T	: Translation matrix
t, k	: Time component
T <sub>[i,j]</sub> , H <sub>[i,j]</sub> , F <sub>[i,j]</sub> , D <sub>[i,j]</sub>	: Jacobean matrix
u(t)	: System input
V	: Linear velocity
V <sub>L</sub>	: Left linear velocity
VO	: Visual Odometry
V <sub>R</sub>	: Right linear velocity
V(t), V <sub>i</sub> (t), V(k)	: White Gaussian measurement noise
W(t), W <sub>i</sub> (k), W(k)	: White Gaussian process noise
X <sub>t</sub> , X(t), X <sub>k</sub> , X(k)	: State vector
Y(t), Z(t)	: Measurement or observation matrix
1D	: One dimension
2D	: Two dimension
3D	: Three dimension
Δt	: Time sample
φ	: Steering angle

$\omega$	: Angular velocity
$\omega_L$	: Left angular velocity
$\omega_R$	: Right angular velocity
$\theta$	: orientation or heading angle
$\hat{X}_i(k)$	: Estimated state vector

## LIST OF FIGURES

Figure 2.1. Ultrasound sensor operating mode .....	11
Figure 2.2. Velodyne HDL 64: 3D LIDAR sensors .....	12
Figure 2.3. Illustration of translation between two cartesian coordinates .....	17
Figure 2.4. Illustration of rotation in Cartesian coordinate.....	18
Figure 2.5. Two Coordinates transformation .....	19
Figure 2.6. An ideal homogenous transformation of camera.....	22
Figure 2.7. A model of projection plane from 3D plane to 2D Plane [4]......	22
Figure 2.8. The main components of Visual Odometry.....	27
Figure 2.9. Kinematic model of two wheel mobile robot .....	31
Figure 2.10. Differential steering speed for four wheeled mobile robot .....	33
Figure 2.11. Dynamics model for four wheeled mobile robot.....	37
Figure 3.1. Centralized data fusion architecture .....	44
Figure 3.2. Hierarchical data fusion architecture.....	45
Figure 3.3. Kalman Filter's mathematical foundation [47]. .....	47
Figure 3.4. Dynamic system state space block diagram [7].....	47
Figure 3.5. Kalman Filter algorithm .....	49
Figure 3.6. EKF operating algorithm.....	51
Figure 3.7. Structure of Decentralized Kalman Filter (DFK).....	52
Figure 3.8. Chronology of some contributors to estimation technology [47].....	56
Figure 4.1. Mobile robot drawn with SolidWorks 2017 software .....	60
Figure 4.2. Corresponding Robotrom block diagram on Simulink .....	61
Figure 4.3. Robotrom Simulink model .....	62
Figure 4.4. Electrical equivalent circuit of DC motor with Simulink.....	63
Figure 4.5. Simulated robot without slope.....	64
Figure 4.6. Simulated motor with a slope angle equal to 6 degree .....	64
Figure 4.7. Decentralized Kalman Filter model on Simulink .....	69

Figure 4.8. Local Kalman Filter 1: x,y positions .....	70
Figure 4.9. Linear speed on x and y axes, local KF 1 .....	71
Figure 4.10. Measurement and estimation errors of local KF 1 .....	71
Figure 4.11. Heading angle, measurement and estimation errors local KF 2 .....	72
Figure 4.12. Position on the x and y axes of the global Kalman Filter.....	73
Figure 4.13. Speed on the x and y axes of the global Kalman Filter .....	74
Figure 4.14. Heading angle of the global Kalman Filter .....	74
Figure 4.15. Measurement and estimation errors of global Kalman Filter .....	75

## LIST OF TABLES

Table 2.1. Comparison of lidar, radar and camera properties.....	12
Table 2.2. Algorithm of s Fundamental matrix with RANSAC algorithm [4].....	25
Table 2.3. RANSAC algorithm [4]. .....	26
Table 2.4. Feature Detector properties comparison .....	29
Table 4.1. Robotrom parameters.....	61
Table 4.2. Simulated motor characteristic for Robotrom .....	65

## **SUMMARY**

Keywords: Mobile Robot, Odometry, Localization, Decentralized Kalman Filter.

Robotics is the science that allows the realization and design of robots. This discipline brings together several fields such as Electronics, Mechanics, Computer Science, Computer Vision and several other basic sciences such as Mathematics and Physics. A robot is an assembly of several elements into a block equipped with sensors able to move dependently through a computer program that allows it to make decisions.

Today, robots are the first choice in several fields thanks to their performance and their tireless attitude to perform tedious work. Robots are found in the fields of technology, industry, education system, agriculture, medical system, automobile etc. There are several types of robots depending on the task assigned to them and their operating environment.

The objective of this thesis is to study the aspects of a mobile robot capable of evolving in any environment. During this research, the capabilities of a mobile robot operating in an unknown environment with MATLAB were highlighted by simulation. Based on the principles of odometry, differential drive, the robot will evolve in a noisy, static and dynamic environment. These parameters highlight important challenges that must be addressed in this thesis. In fact of its suitable functioning, the robot must be able to interact with its environment. For this fact, the robot is equipped with different wheel sensors for estimating the trajectory and location of the robot in order to be oriented. These sensors do not provide precision measurement, they can be complemented in their tasks by a camera. As for the camera, it has the ability to provide information in real time and in a compact way. This should be used according to the technique of visual odometry. It is important to know that the weather conditions can sometimes negatively influence the measurements provided by the camera. However, coupling the data provided by the wheel sensors and the camera with a third sensor such as lasers or ultrasonic can make it possible to optimize the system and to be able to estimate with less error the position of the robot.

Odometry is a technique for estimating the pose (position, orientation) of a moving robot. It is used when we design mobile robots evolving in the absence of GPS data or when it is hidden or poorly received by the robot. The advantage of this technique is that the robot can be able to evolve in almost any environment without having a prior knowledge of the field. There are several algorithms for estimating the robot pose's such as Particle Filter (PF), Kalman Filter (KF), and Monte Carlo Localization (MCL) etc. But, in this thesis only the Kalman Filter will be explore. The Decentralized Kalman Filter will be used for the fusion of the data provided by the different sensors.

# DAĞITIK KALMAN FİLTRESİ KULLANILARAK MOBİL ROBOT ODOMETRİK KONUMLANDIRMASI

## ÖZET

Anahtar Kelimeler: Mobil Robot, Odometri, Kalman Filtresi, Lokalizasyon, Dağıtık Kalman Filtresi.

Robotik, robotların gerçekleştirilmesini ve tasarlanmasını sağlayan bilimdir. Bu disiplin, Elektronik, Mekanik, Bilgisayar gibi birçok bilimi bir araya getirmektedir. Bir robot, karar vermesine izin veren bir bilgisayar programına bağımlı olarak hareket edebilen sensörlerle donatılmış çeşitli elemanların bir araya getirilmesidir. Günümüzde robotlar yoğun ve kompleks çalışmalardaki performanslarından dolayı çeşitli alanlarda ilk tercihtir. Robotlardan, teknoloji, endüstri, eğitim sistemi, tarım, sağlık sistemi, otomobil vb, alanlarda yararlanılmaktadır. Kendilerine ve evrimleşmesi gereken çevreye bağlı olarak çeşitli türlerde robotlar vardır.

Bu tezin amacı, herhangi bir ortamda geliştirilen bir mobil robotun incelenmesidir. Bu araştırma sırasında, MATLAB ile bilinmeyen bir ortamda çalışan bir mobil robotun yetenekleri simülasyon ile vurgulanmıştır. Odometri ve diferansiyel sürücü prensiplerine dayanarak, robot gürültülü bir ortamda geliştirilecektir. Bu parametreler, tezde ele alınması gereken önemli zorlukları vurgulamaktadır. Çünkü robotun uygun çalışabilmesi için, çevresi ile etkileşime girebilmesi gerekmektedir. Bunun için robot, yönlendirilmek üzere robotun yörüngesini ve yerini belirlemek için farklı tekerlek sensörleri ile donatılmıştır. Bu sensörler hassas ölçüm sağlamazlar, bu sorun bir kamera kullanımıyla kompanze edilmeye çalışılır. Kameraya gelince, bilgiyi gerçek zamanlı ve kompakt bir şekilde sağlama yeteneği vardır. Bu görsel odometri tekniğine göre kullanılacaktır. Hava koşulları, bazen kamera tarafından sağlanan ölçümleri olumsuz etkileyebilir. Bununla birlikte, tekerlek sensörleri ve kamera ile sağlanan verileri lazerler veya ultrasonik gibi üçüncü bir sensörle birleştirmek, sistemi optimize etmeyi ve robotun konumunu daha az hatayla tahmin etmeyi mümkün kılabilir.

Odometri, hareketli bir robotun pozunu (pozisyonunu, yönünü) tahmin etmede kullanılan bir tekniktir. Yer haritası ve GPS verilerinin yokluğunda veya robot tarafından gizlendiğinde veya kötü bir şekilde alınmasıyla geliştirilen robotların tasarımında kullanılır. Bu tekniğin avantajı, robotun hemen her alanda, ortam hakkında önceden bilgi sahibi olmadan kestirim yeteneğine sahip olmasıdır. Parçacık Filtresi (PF), Kalman Filtresi (KF) ve Monte Carlo Yerelleştirme (MCL) gibi bir robotun pozunu tahmin etmek için çeşitli algoritmalar vardır. Ancak bu tezde farklı sensörler ile sağlanan verilerin birleştirilmesi için Dağıtık Kalman Filtresi kullanılmıştır.

# **CHAPTER 1. INTRODUCTION**

## **1.1. Thesis Introduction**

Robots are the product of robotics, which is a science that involved in their designs and achievements. Defined as a machine consisting of sensors, actuators and a computer program, it has the ability to move with or without the help of an operator. The sensors are the sense organs of the robot, they allow it to interact with its environment to make decisions. As for the actuators, they are its driving organs. The master orchestra allowing coordination and synchronization of everything in a coherent action is a microcontroller embedding within it a computer program previously introduced by the designer of the robot. The computer program allows the robot to process the information received by the sense organs (sensors) and gives the possibility to the robot to send an order to the motor organs (motor, led, fan etc.). The programming language used depends on the designer and his specifications. Robots are skillful, precise and tireless for performing a tedious and tiring work, characteristics that the human does not possess because of his fatigability.

Robots are able to evolve in a variety of static or dynamic environments. Today, robots have become the companions of man. Thus, they are present in different fields such as agriculture, medicine, education, weapons, electronic gadget etc. There is a wide variety of robots depending on their employment environment, among them, can listed that industrial robots, domestic, military, transport, scientific, medical etc. The robots can be classified according to their structures and mode of operation. Thus, we have the humanoids, the manipulators, mobile robots like the automatic pilots (drone), virtual, probability robot [1 - 2].

The performance and speed of these robots in the tasks execution made them the unavoidable choice in several fields. They are granted with a very important place both



economically and scientifically, in a world where technology requires a high level of performance in terms of efficiency, speed, accuracy and above all at a lower cost.

The new technology used in robots today, tends to give more intelligence and autonomy to the robot to facilitate their interaction with humans with the least possible risk. To achieve this goal, robots are now equipped with much more robust, intelligent sensors and processors implementing very complex algorithms.

The purpose of this thesis is to make a mobile robot platform capable of self-localization using the data collected by the sensors that are embedded on it. This robot is equipped with various sensors such as wheel sensors, camera and / or laser or ultrasonic. All these sensors have almost the same purpose, allow the robot to react with its environment.

It is important to know that in most mobile robots, designers tend to use landmarks as a reference to allow the robot to locate itself in order to orientate itself. As part of this research, the landmarks option will not be appropriate because the goal is that the robot is able to operate in any environment without having a prior knowledge of it. To do this, our research will be based on the principles of odometry, which is the estimation of the pose of a robot in motion. There are several types of odometry depending on the type of sensors used to collect the information. Thus, one has the wheel odometry, which uses wheel encoders to determine the location of the robot. This type of odometry is very convenient when the robot evolves on a rough terrain. But the disadvantage is that it does not provide a very accurate result. There is next to the wheel odometry, visual odometry (VO), based on camera usage, visual odometry is much more accurate than wheel odometry [3]. This accuracy is due to the cameras' ability to provide real-time, compact and 3-dimensional information. The cameras offer several advantages in the visual odometry system, they are cheaper, economical and easy to embed. These different parameters have made them the most used optical sensors in imaging engineering, computers, mobile robots and many other similar applications. However, the cameras are very diverse and complex in overall. There are several varieties of cameras classified according to the technologies (CCD, CMOS), the resolution (HD), and the cadence (number of images / second) and also of the

geometrical configuration (perspective, affine etc.), and internal parameters which differ from one model to another [4]. The major disadvantage of VO is that the quality of ego-motion estimation is a function of the quality of the captured image. Thus, the weather conditions, the brightness of the scene can influence the quality of a VO system. However, there is another aspect of odometry called LO (Lidar Odometry) which is based on the use of laser sensors such as LIDAR. Like cameras, LIDAR laser sensors have the ability to deliver 3D information. The advantage with LO is that it is not influenced by the brightness of the scene and the weather, which allows it to give results much more expected. But, the problem that arises here is that unlike the camera, the data collected from the environment by the LIDAR are represented by dots thanks to the number of variables. The distortion of the motion can also disturb the data received by the LIDAR. The LO also depends on the correspondence scan, which can influence the data when the robot is operating in a degenerate environment.

Although there is a variety of odometry to estimate the pose of a moving robot, it can be remembered now that none of them provides a hundred percent reliable result. However, the design of a reliable and robust system based on odometry consist of combining different types of odometry in order to take profit of the advantages and offset the disadvantages of each type. Thus, one can combine the wheel odometry and the VO; the LO with the VO; or the pooling of the three types for a more robust system.

In order to study the odometry, a robot was designed to deepen our analysis. Robotrom is a mobile robot like-car designed with SolidWorks version 2016b, then simulated on Matlab R2016b and Simulink 3D simulation environment. The robot is based on the studying of the technique of forward and reverse kinematics, dynamics differential drive, and Ackermann steering model. Made up of four wheels, Robotrom will be able to evolve in almost any rugged environment. The motors are controlled by a PID controller [5]. Equipped with encoder, gyroscope, camera or laser, Robotrom will have the ability to localize itself. The design of this type of system has a lot of advantage in the sense that the operating environment of the robot does not require a landmarks. The environment of evolution of the robot being noisy, the probability of processing false data is great. Thus, to overcome this phenomenon, some filters and estimators

were studied during this thesis. The standard (KF), Extended Kalman (EKF) and Decentralized Kalman (DFK) filter have been investigated [6 - 7]. Next to the Kalman filter, we can use another filter like a Particle Filter (PF) in Robotics [8 - 9].

The Kalman filter is an optimally estimation technique using to estimate the state of a linear system. The optimality criterion for calculating the Kalman filter is a stochastic criterion. Since its invention, the Kalman filter has been used in several applications in the field of engineering. It is used to estimate the state of a system from information a priori on the evolution of the model and actual measurements. It is used to estimate unknown initial conditions (ballistics), to predict trajectories (mapping), to locate a vehicle (radar, navigation etc.) and also to implement control laws based on a state and state feedback estimator (LQG linear quadratic Gaussian control). There are several varieties of Kalman filter with precise specifications for their uses. For example, there is the Extended Kalman Filter (EKF), which is used for the study of nonlinear systems. The EKF is based on the calculation of a Jacobean matrix for the linearization of a nonlinear system. The calculation of the Jacobean can be expensive in terms of implementation. There is also the Decentralized Kalman Filter, which is based on the structure of the data fusion algorithm. DFK is generally used in multi-sensor systems for data fusion. The DFK was used in this thesis for the improvement of the robot pose's calculating by merging the estimated data provided by different types of sensors equipped on the robot platform. The DFK is incorporated by the local processor using the standard Kalman Filter (local filter) and a master filter which is the heart of the Decentralized Kalman Filter structure. This kind of Kalman filter using offers more accuracy and reliability system.

Like the Kalman filter, particle filter is also an estimator used in a linear system. But the particularity of particle filter is that it has the ability to treat both linear and nonlinear systems. Unlike the Kalman filter which is based on Gaussian data distributions, particle filter processes non-Gaussian signals. Particle filter is very expensive in terms of implementations. Even more expensive than the Extended Kalman Filter. So the use of this filter have to be when one is sure of inefficacy of the Kalman Filter.

The rest of the thesis begins by highlighting the problem statement, defining the objectives, research motivations, the used methods and materials in two chapters. Finally the experiments, results, conclusion and suggested future works are closed this research.

## **1.2. Problem Statement**

In the case of a robot operating in a predefined environment, the scenario is known in advance, when the designer is programming the robot, he is able to evaluate all possible cases of obstacles and scenario that the robot may encounter. In fact, we know in advance the obstacles and the scenario that the robot will face and also the operating environment of this one. Unfortunately, in the case of this research, this is very different and unachievable. The problem here, we do not know in what kind of environment the robot is going to work, it is almost impossible to predict the decisions to be learned by a simple synchronized program. However, it is possible to define the probably scenes that the robot can meet and with the help of sensors, to define the probably decisions that it have to make. The problem in this case of system design is much more probabilistic than deterministic because nothing is known in advance. Thus, the pose of the robot can be estimated through stochastic filters like Kalman Filter, Particle Filter and Monte Carlos Localization algorithms and so on.

To do this, one must ask the most eminent questions that are: What should the robot do if it encounters an obstacle on its path? The answer look like very simple to avoid it of course. But how, if the robot doesn't knows where is it? So the first answer here is to give an opportunity to the robot to localize itself in its operating environment. This first answer is the purpose of this thesis. The second question that can be ask is, if the obstacle is itself endowed with a motion? That is, the object in question has a relative moving speed, a relative position and an acceleration what can the robot do? The answer in this case become complex depending on operating environment of the robot. Is it in a static or dynamic environment? How many object have a dynamic

parameters? There are some kind of question that can help the design to improve his algorithm and program to model a mobile robot.

In the case of a static environment that mean any object aren't in motion, the problem can be easier to solve because of knowing of exactly position between the object and the robot, so that the robot can reorient itself using the information provided by odometry. In the second case, which is a dynamic environment, the problem is much more complicated especially if the robot meets several dynamic objects at the same time. Here, the robot must be able to calculate the relative position of each moving object and their speeds in order to compare them to itself position in order to avoid any case of collisions. But, another problem arises, would the robot have enough time to perform all these calculations before the crash time? For this fact, the robot must have a faster execution time than the time of the events. This aspect is a challenge that the designer have to rise above.

In order to reduce the time of calculation, one must define what could be an obstacle for a robot? Since it will evolve in an unknown and noisily environment consisting of several objects possible to be captured by the sensors of the robot. So, will be regarded as obstacle for the robot, only the objects being on its direction and being able to crash into the robot. Therefore, one can reduce the calculation timing and can localize properly the robot in order to run it in any operating environment. No matter the decision that the robot is going to take, in any cases, the robot have to know his position before any decision.

### **1.3. Thesis Objectives**

The purpose of this thesis is a priori pedagogical, since it makes it possible to highlight the importance and the place occupied by robots in the world, in a foreground. In the background being able to simulate a robot without having designed it in a physical way, is an important exploitable aspect for the development of students in robotics, electronics, computer science and many other fields related to the design of the robots.

## 1.4. Organization of the Thesis

This thesis is organized as:

Chapter 1 introduction: The introduction allowed to present the generality of this thesis, the problem statement and the reasons which motivated this research and the expected results.

Chapter 2 the first part of theoretical methods and Materials used and their literature review: In the second chapter, the focus was on starting of the theoretical part of this thesis. This theoretical part have been explained in this chapter and the next one. In the first part of theoretical materials and used methods, the focus was on the mechanical part of the mobile robot, sensors, visual odometry, and some image processing background like image reconstruction using the RANSAC algorithm has been explained. Then one has given the literature of the investigated previous work in mobile robot.

Chapter 3 is the second part of the theoretical methods and materials used and their literature review: This part has focused especially on the used algorithms like a Kalman Filter.

Chapter 4 Experiments and results: it allowed the implementation, simulation and analysis of the obtained results during the practical experiments by use of the Matlab and Simulink. The simulation is performed with Simulink's 3D simulation environment, which is the part of Matlab platform. The obtained results and graphical has been explained in this fourth chapter.

Chapter 5 Conclusion and Suggestion: In this section the realized work during this thesis was concluded, some perspectives and suggestions for future work was given.

## **CHAPTER 2. FIRST PART OF METHODS AND MATERIALS**

### **2.1. Introduction**

The first part of the theoretical principles and methods used during this research is highlighted in this section. These principles are numerous and varied according to their level of importance and intervention in this research. The robot is supposed to evolve in a perspective environment, so it has been equipped with sensors to obtain some information in its operating environment and several processing units to process these information provided by the sensors from its operating environment. The whole device (sensors, processing units) is the base that allows the robot to interact with its environment in order to locate itself, detect and avoid potential obstacles that may be in its path. For this fact, the sensors used are diverse and sometimes active (ultrasound, radar, laser etc.) or passive (camera etc.). As regards the processing unit, it consists of processors capable of executing a complex algorithms. Thus, in this chapter we will expose the different types of sensors, the robot's driving model have been studied in this research.

First of all, we will focus on rotary encoders that is used for wheel odometry, then the telemeters that are distance measurement sensors, then, make a development on the vision which is the camera studying. The cameras have become points of interest for many researchers and scientists because of their prize and accuracy.

Moreover, we will discuss the mechanical model of the robot which consists of kinematics, dynamics, and some driving model configurations used for wheeled mobile robots like the unique circle drive, the Ackermann model etc.

Finally the literature reviews have been presented followed by the partial conclusion of this chapter.

## **2.2. Sensors**

There are several types of sensors classify according to the nature of the processed signal, for example RADAR based on the processing of the radio waves, the LIDAR uses rays of light to provide information. One can also classifies the sensors according to their energy required to operate (active sensors) or not (passive). In the context of an odometry system for a mobile robot, the sensors generally used are incremental encoders, ultrasound, lidar, cameras and sometimes radar sensors.

### **2.2.1. Wheel encoder**

Wheels encoders are sensors for measuring the rotation around an axis to obtain information about the angle of the wheel. Then, we can deduce the velocity of the vehicle from the variation of the position with expected time. There are major types of encoders, incremental rotary encoders and absolute rotary encoders. The measures provided by the encoders can be influenced by the difference in the diameters of the wheels, a bump on the road etc. So the robot's designer have to remove that kind of error by the use of filters or combine other pose estimating technique to the encoders measurement in order to have a best results.

### **2.2.2. Telemeter**

The rangefinders are the active sensors measuring the distance, they can be ultrasonic, radar, laser, sonar types etc. The telemeters emit a wave moving in a direction of propagation and able to reflect on the surface of an object. The difference between the flight time of the incident wave and its echo makes it possible to calculate the distance between the surface forming the echo and the sensor.



### **2.2.2.1. Radar**

The principle of the radar is to emit an electromagnetic wave and estimate the distance with the reception of the echo formed by this electromagnetic wave on a conductive surface. However, if the object in question has a velocity relative to the radar, the reflected wave has a frequency different from the incident wave (it is assumed that the signal is monochromatic). This effect called Doppler, makes it possible to obtain a measurement of relative velocity between the reflecting object and the radar, by estimating the frequency shift.

The benefit of radar is their important precision in most weather conditions (Fog, rain, snow). Not being influenced by the weather conditions, the radar is often coupled in operation with other sensors. However, the radar has some limitations such as the low angle of view (approximately 10 degrees for a long-range radar). Moreover, any object outside the icon described by this angle of view is not perceptible. Because of the limitation of its angular resolution, the radar is not able to distinguish two objects located side by side. In addition, the analysis of radar data provides a lot of parasitic echo that can distort the calculations. These being, impose the use of filter for the removal of these noise for the reliability of the system.

### **2.2.2.2. Ultrasound**

Ultrasound sensors are the sensors generating high frequency sound pulses at regular intervals of time. These impulses make an echo when they encounter an obstacle in their direction of propagation. The received echo signal allows to calculate the distance between the sensor and the target. This calculation is a function of the flight time of the sound pulses. Ultrasound sensors are probably the most distance measurement sensor used for mobile robots because of their low cost. Their resolutions can range from 20 mm to 10 m. However, it should be remembered that ultrasound sensors have certain disadvantages [10] Ultrasound works with difficulty in environments where there is a clutter and in the presence of dust. They do not work in a vacuum and there is a dead zone to respect. The dead zone is the minimum distance of detection to keep

between the sensor and the target so that it can function well. The presence of unexpected objects or reflectors in this area may lead to errors measurement Figure 2.1. below illustrate the mode of operation of ultrasound.

$$v = \frac{d}{t}$$

$$\rightarrow d = v \cdot t$$

Where  $v=340\text{m/s}$  (speed of the sound)

$t$ : flight time of impulse and  $d$  the measured distance

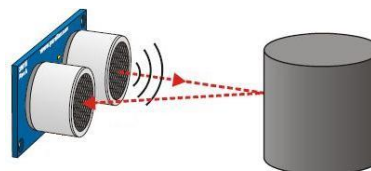


Figure 2.1. Ultrasound sensor operating mode

### 2.2.2.3. Lidar

The lidar is a rangefinder using the reflections of a ray of light for the measurement of distance. Just like the previous rangefinders, the lidar uses the principle of the flight time of the light wave for the estimation of the distance. The lidar has the ability to provide very accurate distance measurements over a long range. In addition to this, it offers the possibility to detect all types of objects (vehicle, pedestrian, cardboard etc.) provided that these objects have a power reflecting light. In a nutshell, it is the ideal rangefinder prototype for the major parts of today's applications. However, like any technology, lidar also has its disadvantages. As it consists of a single light beam, the lidar can only detect objects on its line of propagation. To overcome this imperfection, the lidar will be equipped with an electric motor controlling a rotating mirror thus to collinearity direct the light beam in several directions [11]. Thus, the angular resolution becomes important and precise with a much wider angle of view. This type of systems is called a scanning rangefinder. These rangefinders are very sensitive to the pitch of the robot. This is due to the fact that the scanning plane can cut the surface of the robot's road, detecting it as an obstacle. To correct this, one use multi-plane lasers that allows to reconstruct a partial 3D view of the scene. There are several types of lidar, for example the velodyne's models (Velodyne HDL-64, Veldyne HDL32, Velodyne HDL16). Figure 2.2. below presents the properties and characteristics of these sensors. The other major flaw of the lidar is that this sensor is very expensive.



Figure 2.2. Velodyne HDL 64: 3D LIDAR sensors

The Table 2.1. below gives a summary by comparing of some sensors

Table 2.1. Comparison of lidar, radar and camera properties

	LIDAR	RADAR	Camera
Sensing Dimension	3D	1D	2D
Range	+++	+++	-
Range Rate	++	+++	-
Field of View	+++	++	+
Width & Height	+++	-	+
3D Shape	+++	-	+
Object Rec @ Long Range	+++	-	-
Accuracy	+++	-	+
Rain, Snow, Dust	++	+++	-
Fog	+	+++	-
Pitch Darkness	+++	+++	-
Brigth sunlight	+++	+++	-
Ambient light independence	+++	+++	-
Read signs & see color	-	-	+++

### 2.3. Camera

Today, many systems such as robotics, remote monitoring, driver assistance and autonomous vehicles call for the expertise of computer vision engineers. The main goal searching through all these areas is to offer eyes to their systems. This could be achieved through the use of cameras. Cameras with their ability to provide real-time, compact information, and particularly at a lower cost, have become the most widely used optical sensors in image processing engineering, computers science, mobile robots and many other similar applications. . However, the cameras are very diverse and complex in overall. There are several types of camera to classify according to the technologies (CCD, CMOS), the resolution (HD), and the cadence (number of images/second) and also of their geometrical configuration (perspective, affine etc.), and of their internal parameters which differ from one model to another.

#### 2.3.1. The affine camera

An affine camera is a camera whose last line  $P^{3T}$  of its matrix  $P$  is of the form  $(0, 0, 0, 1)$ . It is called affine because of the fact that points at infinity are represented towards points at infinity [4]. The general form of the camera matrix  $P$  of an affine camera is as follows:

$$P_A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & t_1 \\ a_{21} & a_{22} & a_{23} & t_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

This matrix  $P$  has eight degrees of freedom, that is to say 8 elements that are non-zero and non-unitary. Let  $A_{2 \times 3}$  be the sub-matrix at the top left of  $P$ . Due to the constraint imposed by an affine camera, the matrix  $A_{2 \times 3}$  is of rank 2. This is due to the fact that the matrix  $P$  is of rank 3. Thus we can write  $P$  as following:

$$P_A = \begin{bmatrix} A & t^T \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & s & 0 \\ 0 & \alpha_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r^{1T} & t_1 \\ r^{2T} & t_2 \\ 0^T & 1 \end{bmatrix} \quad (2.2)$$

An affine camera is a composition of the effect of an affine transformation of 3 spaces, an orthographic projection of a 3D space towards an image and an affine transformation of this image. Thus, we can represent this as the matrix concatenation in the following form:

$$P_A = [3 \times 3 \text{ affine}] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} [4 \times 4 \text{ affine}] \quad (2.3)$$

This concatenation gives a 3x4 matrix resultant in the affine form. The projection of an affine camera in a linear and inhomogeneous plane with a translation is as following:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \quad (2.4)$$

This can be written in a contracted form:

$$\hat{x} = A_{2 \times 3} \hat{x} + \hat{t} \quad (2.5)$$

The point  $t = (t_1, t_2)$  is the image of the point of origin of the world.

In the case of an affine camera, the infinite plane of the space is projected towards infinite points of an image, which can be implemented in the following way:  $P_A = (x, y, z, 0)^T = (x, y, 0)^T$ . Unlike a perspective projection camera, the main plane of an affine camera is an infinite plane. Also, the optical center, which extends on the main plane, must imperatively extend on the infinite plane. In view of all of these assumptions above, one can deduce the properties defining an affine camera such as the following:

- Any projection of camera matrix whose main plane is an infinite plane is an affine matrix.

- The parallel lines of the world are projected towards parallel lines of the image. This follows from the fact that the lines of the world intercede in an infinite plane, and that this point of intercession is represented in infinite points in an image. As a result, the lines of the image are also parallel.
- $d$  is a vector such that  $A_2 \times_3 d = 0$ , is the direction of the parallel projection, and that  $(d^T, 0)^T$  of the center of a camera  $P_A = \begin{pmatrix} d \\ 0 \end{pmatrix} = 0$ .

Any camera resulting from the composition of the effect of an affine transformation, with a parallel projection is of the affine form. Starting from this observation, we can identify several types of camera affine like the orthographic projection camera, the Scaling orthographic projection camera, the weak projection camera, the para-perspective camera etc.

### 2.3.1.1. The orthographic projection camera

The orthographic camera is an affine camera whose projection is all along the Z axis. These are interpreted by the matrix P below:

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.6)$$

This representation has points of coordinates  $(x, y, z, 1)^T$  to image points  $(x, y, 1)^T$ , thus removing the coordinates of the Z axis. Thus, the 3D representation of an orthographic camera in a Euclidean coordinate is of the form:

$$H = \begin{bmatrix} R & t^T \\ 0^T & 1 \end{bmatrix} \quad (2.7)$$

Where  $t = (t_1, t_2, t_3)$  and  $R = (r^{1T}, r^{2T}, r^{3T})$ .

The general shape of the orthographic camera can simply be defined as:

$$P = \begin{bmatrix} r^{1T} & t_1 \\ r^{2T} & t_2 \\ 0^T & 1 \end{bmatrix} \quad (2.8)$$

This camera has 5 degrees of freedom, which are the 3 parameters of the rotation matrix  $R$  and the two parameters of the translation vector  $t$ . The camera matrix of the orthographic projection  $P = [A | t]$  is characterized by the fact that the last line of the matrix  $A$  is zero and that the first two lines are orthogonal with a norm equal to 1, and  $t_3$  equal to 1.

### 2.3.1.2. The scaling orthographic projection camera

The scaling orthographic projection is simply an orthographic projection with an isotropic factor. Thus, the general form of its camera matrix  $P$  can be written in the following way:

$$P = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r^{1T} & t_1 \\ r^{2T} & t_2 \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} r^{1T} & t_1 \\ r^{2T} & t_2 \\ 0^T & \frac{1}{f} \end{bmatrix} \quad (2.9)$$

The camera matrix  $P = [A | I]$  has six degrees of freedom. It is characterized by the fact that the last line of the matrix  $A$  is zero and that the first two lines are orthogonal with a norm equal to 1, and  $t_3$  equal to  $1/f$ .

### 2.3.1.3. The weak projection camera

It is a camera whose scaling factors of the two axial directions ( $x, y$ ) of the image are not equal. These being, the projection matrix is the following one:

$$P = \begin{bmatrix} \alpha_x & 0 & 0 \\ 0 & \alpha_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r^{1T} & t_1 \\ r^{2T} & t_2 \\ 0^T & 1 \end{bmatrix} \quad (2.10)$$

The camera matrix  $P = [A | I]$  has seven degrees of freedom. It is characterized by the fact that the last line of the matrix  $A$  is zero and that the first two lines are orthogonal but no need to have a unitary norm, and  $t_3$  equal to 1.

## 2.3.2. Camera transformation and geometry

### 2.3.2.1. Translation and rotation

In the case of a mobile robot, the points are constantly transferred from one coordinate to another. Thus, one must be able to express the same point in several coordinates, then moving from one to the other easily. Let  $B$  be a coordinate system (15, 7) defined in the reference  $A$  with  $T_{AB}$  a translation from  $A$  to  $B$ .  $P$  is a point defined in the  $B$  references, the position of  $P$  is a vector  $P_B$  starting from the origin of  $B$ , along the axes of  $B$  and ends at point  $P$ . The position of  $P$  expressed in the  $A$  reference is the sum of the both vectors  $P_B$  and  $T_{AB}$ . The diagram in Figure 2.3. illustrates it.

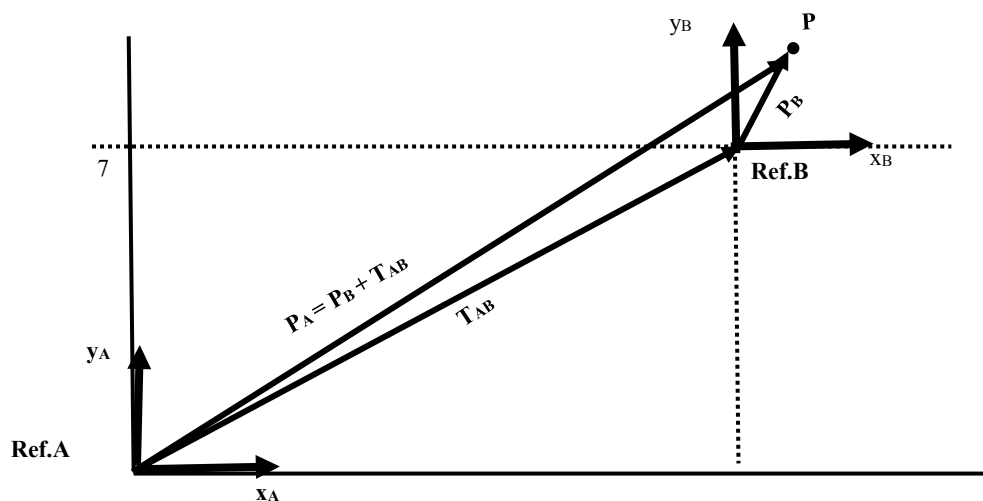


Figure 2.3. Illustration of translation between two cartesian coordinates

$$P_A = P_B + T_{AB}$$

(2.11)

$$\begin{bmatrix} P_{Ax} \\ P_{Ay} \end{bmatrix} = \begin{bmatrix} P_{Bx} \\ P_{By} \end{bmatrix} + \begin{bmatrix} T_{ABx} \\ T_{ABy} \end{bmatrix}$$



If the same point P in the two frames is rotated around the origin with an angle  $\theta$  counterclockwise, the equation (2.9) above is no longer true. Consequently, one must take into account the effect of the rotation which is described in Figure 2.4. below.

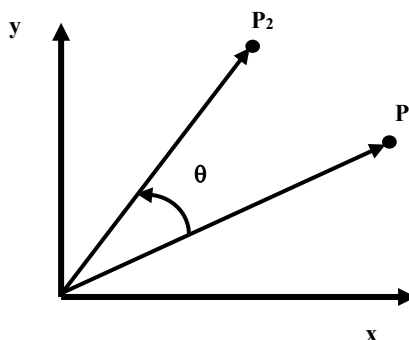


Figure 2.4. Illustration of rotation in Cartesian coordinate

$$R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \quad P_2 = RP_1 \quad (2.12)$$

$$\begin{bmatrix} P_{2x} \\ P_{2y} \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} P_{1x} \\ P_{1y} \end{bmatrix}$$

R is called rotation matrix, P1 is the initial position of point P and P2 the final position of this point to which a rotation of an angle  $\theta$  has been made.

### 2.3.2.2. Coordinates transformation

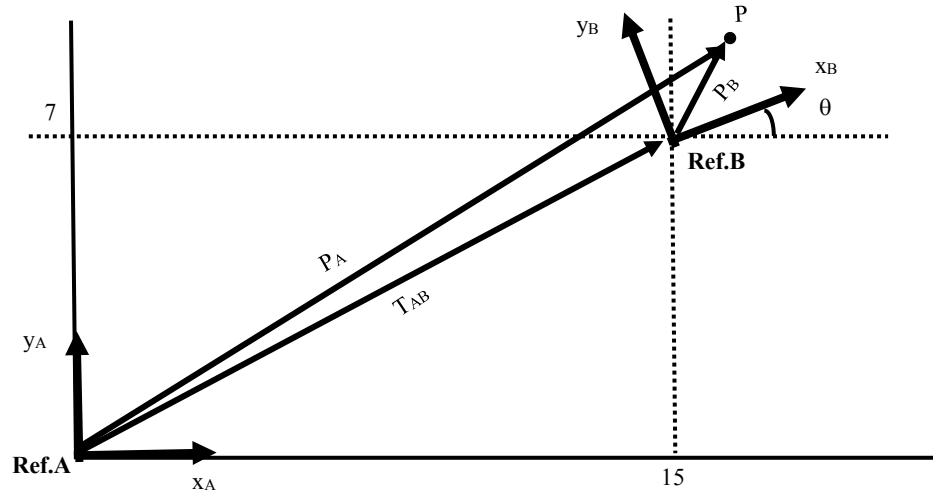


Figure 2.5. Two Coordinates transformation

The transformation between two coordinates can be defined as being a rotation movement followed by a translation. This is shown in the Figure 2.5. above.

$$P_A = R_{AB}P_B + T_{AB} \quad (2.13)$$

$$\begin{bmatrix} P_{Ax} \\ P_{Ay} \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} P_{Bx} \\ P_{By} \end{bmatrix} + \begin{bmatrix} T_{ABx} \\ T_{ABy} \end{bmatrix}$$

In a Cartesian coordinate system, the rotation is a multiplication and the translation an addition. On the other hand, in a homogeneous coordinate the rotation and the translation are both a multiplication.

A 2D translation in a Cartesian coordinate system is as following form:

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} T_{x1} \\ T_{y1} \end{bmatrix} \quad (2.14)$$

In a homogeneous coordinate, the 2D translation is a multiplication and it is in the following form:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_{x1} \\ 0 & 1 & T_{y1} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \quad P_2 = TP_1 \quad (2.15)$$

The rotation in a Cartesian coordinate system as in homogeneous coordinate is a multiplication. Thus the 2D rotation in a homogeneous coordinate system has the following form:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \quad P_2 = RP_1 \quad (2.16)$$

In a homogeneous coordinate system, the transformation (combination of rotation and translation) consists only of a multiplication operation which makes it possible to have a more elegant combination of operations such as  $P_2 = T_2R_2T_1R_1P$ , so that all the translations and rotations have been represented by the same matrix H.

$$P_2 = HP \quad \text{with } H = T_2R_2T_1R_1 \quad (2.17)$$

It is recalled that the matrix multiplication is not commutative, thus, the operations TR and RT will not give the same results. Thus, it is more convenient to use the combination TR.

The rotation in a 3D transformation in a homogeneous coordinate system differs according to the axis around which the rotation is made, so we have:

Rotation around the x axis

$$R(\theta)_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.18)$$

Rotation around the Y axis

$$R(\theta)_y = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.19)$$

Rotation around the Z axis

$$R(\theta)_z = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.20)$$

The 3D translation is of the following form

$$T = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.21)$$

The general form of a 3D transformation with the TR combination can be written in the following way:

$$H = TR = \begin{array}{cc} \text{Rotation} & \text{Translation} \\ \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} & \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \end{array} \quad (2.22)$$

### 2.3.2.3. World and camera coordinates transformation

To compute the position of a point (object) in the world with respect to a camera position, can be done by calculating the transformation H (rotation and translation) between the camera in the camera coordinate frame and the object located in the world

coordinate frame. The diagram in Figure 2.6. below shows a transformation model of a camera.

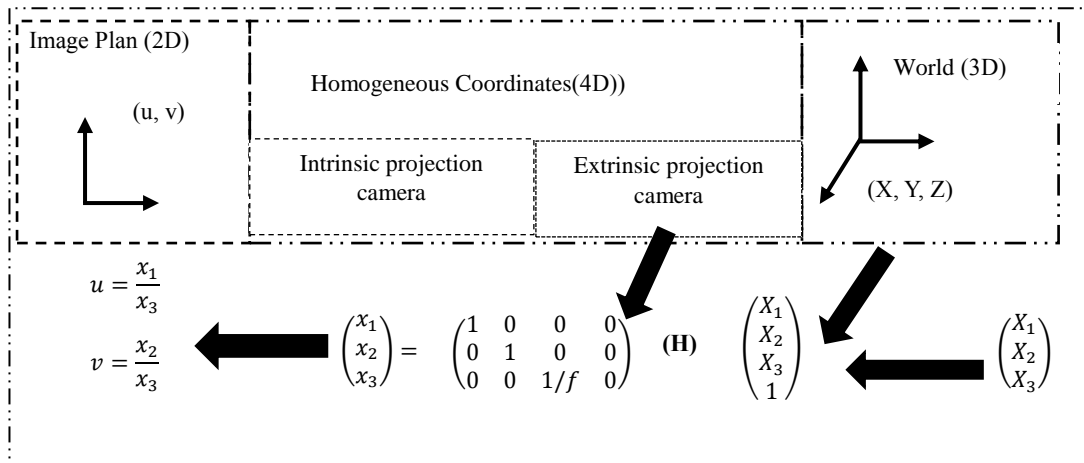


Figure 2.6. An ideal homogenous transformation of camera

This model is the ideal model of the camera. However, it may change depending on the technology and intrinsic parameters of the used camera.

### 2.3.3. Geometry projection of the camera

Projection is the passage from the three-dimensional world to a two-dimensional image, a process in which one dimension is removed like is shown in Figure 2.7.

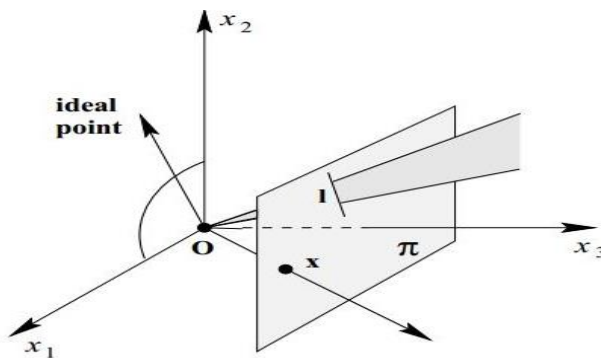


Figure 2.7. A model of projection plane from 3D plane to 2D Plane [4].

Projection in other words, is the procedure of forming an image according to the coordinates of an object. The image formed is the 2D representation of the 3D world object. This procedure allows to deduce the 3D structure of an object and to know how it would appear in an image. The application of a projective geometry in image processing, is the modeling of a 3D projective space, belonging to  $\mathbb{R}^3$  plane along a point to infinity. Simultaneously, the model of the image is a 2D projective in a  $\mathbb{R}^2$  plane. The method allowing the passage from a plane  $\mathbb{R}^3$  to a plane  $\mathbb{R}^2$  is called a central projection, where the ray coming from a point of space is drawn from a 3D point of the world through a fixed point of the space, it is a center of projection. This ray will intercede a precise plan in the space chosen as an image plan. The intersection of the ray with the image plane represents the image of the point. Consider  $\mathbb{R}^3$  having homogeneous coordinates as  $(X, Y, Z, 1)^T$  and the projection center be the origin  $(0, 0, 0, 1)$ . The coordinates of  $\mathbb{R}^2$  are  $(x, y, w)^T$ . Thus, the map is represented by a homogeneous matrix  $3 \times 4$ , the matrix P. This matrix P is called matrix of the camera. Finally, the projective of a camera on a point of space can be written in homogeneous linear form as following:

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = P_{3 \times 4} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.23)$$

However, if all points extend on the z axis, with  $z = 0$  then the equation is reduced as following:

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = H_{3 \times 3} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \quad (2.24)$$

By using projective geometry, it is possible in many circumstances to reconstruct a scene from a single image. However, the reconstruction of a scene of more than one image, that is to say that of two images for example can be done by the use of the fundamental matrix, which is an encapsulation of the pairs of matrix P and P'.

### 2.3.4. The fundamental matrix

The fundamental matrix is the algebraic representation of the epipolar geometry. In other words, the fundamental matrix is the encapsulation of the intrinsic projective geometry between two image views, which is nothing other than epipolar geometry. This fundamental matrix is derived from the mapping of a point and its epipolar line. In a given image pair, each point  $x$  in one of the images has a corresponding epipolar line in the other image. Any point  $x'$  in the second image matching the point  $x$  must extend on the epipolar line  $l'$ . The epipolar line is the projection of the rays of the point  $x$  through the center  $C$  of a camera in the second image. Thus represented as follows,  $x \mapsto l'$ , is the corresponding epipolar line in an image from one point in the other image. This representation is a singular correlation, it is a projective mapping from a point to a line represented by a matrix  $F$  called the fundamental matrix. It is a  $3 \times 3$  matrix of rank 2. Assume a point  $X$  in the space of 3 dimensions imaged by point  $x$  in the first image, and  $x'$  in the second one, then the image point satisfies the following equation:

$$x'^T F x = 0 \quad (2.25)$$

The fundamental matrix is in fact independent of the structure of the scene. However, it can be calculated from the correspondences of the image points of the scene alone, without knowing the internal parameters or the relative position of the camera. This property of fundamental matrix is suitable when the internal parameters of the used camera are unknown. The algorithm describing the computation of the fundamental matrix with the RANSAC algorithm is shown in Table 2.2.

Table 2.2. Algorithm of the Fundamental matrix with RANSAC algorithm [4].

<b>Objective:</b> Compute the fundamental matrix between two images.
<p><b>Algorithm</b></p> <p>(i) <b>Importance points:</b> Calculate importance points in each image.</p> <p>(ii) <b>Putative correspondences:</b> Calculate a set of importance point matches based on nearness and resemblance of their strength neighborhood.</p> <p>(iii) <b>RANSAC robust estimation:</b> Repeat for <math>N</math> samples, where <math>N</math> is determined adaptively such as in algorithm.</p> <p>(a) Select a random sample of 7 correspondences and compute the fundamental matrix <math>F</math>. There will be one or three real solutions.</p> <p>(b) Determine the distance <math>dL</math> for each putative correspondence.</p> <p>(c) Calculate the number of inliers consistent with <math>F</math> by the number of correspondences for which <math>dL &lt; t</math> pixels.</p> <p>(d) If there are three real solutions for <math>F</math> the number of inliers is calculated for each solution, and the solution with most inliers retained. Select the <math>F</math> with the largest number of inliers. In the case of ties select the solution that has the lowest standard deviation of inliers.</p> <p>(iv) <b>Non-linear estimation:</b> re-estimate <math>F</math> from all correspondences considered as inliers by minimizing a cost function, using the Levenberg–Marquardt algorithm.</p> <p>(v) <b>Guided matching:</b> Additional importance point correspondences are now calculated using the estimated <math>F</math> to describe a search strip about the epipolar line.</p>

The reconstruction of a scene with several images generates measurement errors that must be taken into consideration to obtain a reconstructed scene closer to the real scene. For this purpose, there are several algorithms such as RANSAC (RANDOM SAMPLE Consensus), LMS (Least Means Square) and MLE (Maximum likelihood Estimation) that are used to eliminate these errors.

### 2.3.5. The random sample consensus algorithm (RANSAC)

The RANSAC algorithm is a robust estimation algorithm used to determine the inliers of the correspondences included in some images. This kind of estimation has a tolerance in outliers. The RANSAC based on the Fischler and Bolles model is one of the largest robust estimators used [12]. This algorithm is capable of cope with a large proportion of outliers.



The idea is, two points are randomly selected; these points define a line. The support of this line is measured according to the number of points that extend with the distance threshold. This arbitrary selection repeats itself so many times and the line with the most support is deemed to be robust. The points with which the distance threshold is close to are the inliers (and constitute the consensus eponyms set). The trick is that if one of the points is an outlier, then the line will not get much support. Furthermore, the line whose racks have an added benefit are favored to have the best fits for scoring. As stated, Fischler and Bolles [12]. "The RANSAC procedure is opposite to that of the standardization of technical data: RANSAC uses, rather than using as much data as possible, as small an initial data set as possible". The application of RANSAC for the estimation of homography is described in the Table 2.3. below.

Table 2.3. RANSAC algorithm [4].

Objective :
Robust fitting of a model to data set $S$ which contains outliers.
Algorithms:
(i) Randomly, choose a sample of $s$ data points from $S$ and instantiate the model from this subgroup.
(ii) Calculate the set of data points $S_i$ which are within a distance threshold $t$ of the model. The set $S_i$ is the consensus set of the sample and describes the inliers of $S$ .
(iii) If the size of $S_i$ (Inliers numbers') is superior to some threshold $T$ , re-estimate the model using all the points in $S_i$ and end.
(iv) If the size of $S_i$ is inferior to $T$ , choose a new subgroup and repeat the above.
(v) After $N$ trials, the major consensus set $S_i$ is chosen, and the model is re-estimated using all the points in the subgroup $S_i$ .

The minimization of the points of the data  $S$  requires the free parameter instantiation of the model which are  $t$ ,  $T$  and  $N$ . This raises three types of very important question which are:

What is the distance threshold?

How much sample should I choose?

Finally, what is the acceptable width for the consensus set?

Although, these questions are very important for the proper understanding of the RANSAC algorithm, they are not dealing in this paper. Nevertheless, to find the answers to these questions in detail, be sure to refer to [4].

## 2.4. Visual Odometry

### 2.4.1. Presentation

Visual odometry is the estimation of the ego-motion of a person or object (vehicle, robot) using the input of one or more camera attached to it. The term odometry was used for the first time by Nister in 2004 in his article [13]. This term was inspired by wheel odometry, which by incrementally estimates the movement of a vehicle by integrating the number of turns of its wheels as a function of time. Similarly, visual incremental odometry, estimates the pose of a vehicle by examining the movement changes induced by the images of a camera embedded on it. It has been shown that visual odometry provides more accurate results than wheels because it is not influenced by wheel slip or terrain conditions. The Figure 2.8. below describes the steps of a visual odometry.

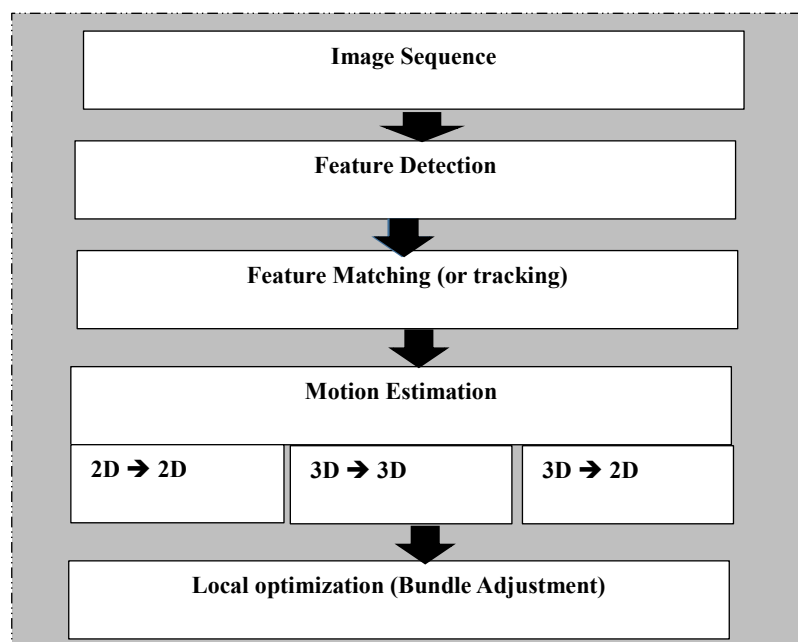


Figure 2.8. The main components of Visual Odometry

### 2.4.2. Selection of the features

There are two main approaches to find feature points and their correspondences. The first is to find feature in an image and then track them down in the next image using the local search technique, such as a correlation. The second is to detect independently feature in all images and then match them on the basis of metrical similarity that describe them. The first approach is more suitable when the image was taken close to the point of view. However, the last one is most suitable when large movement or point of view is supposed to change. During the detection of features, the image is searched according to the simple key points and which are likely to correspond well in to other images. A local feature is an image pattern that differs from its immediate neighbor in terms of texture, intensity, color, etc. There are several methods of detecting features like the Corner Detector, Blob Detector, Harris, Shi-Tomasi, FAST, SURF, CENSURE etc. These detectors are much more detailed in [3]. The Table 2.4. describes the properties of these detectors and compares them based on their performance.

Table 2.4. Feature Detector properties comparison

	Corner detector	Blob detector	Rotation invariant	Scale Invariant	Affine Invariant	Repeatability	Localization Accuracy	Robustness	Efficiency
Haris	x		x			+++	+++	++	++
Shi-Tomasi	x		x			+++	+++	++	++
FAST	x		x	x		++	++	++	++++
SIFT		x	x	x	x	+++	++	+++	+
SURF		x	x	x	x	+++	++	++	++
CENSURE		x	x	x	x	+++	++	+++	+++

## **2.5. Mobile Robot Locomotion Modelling**

Locomotion is the effect that allows a mobile robot to move autonomously. The locomotion of the mobile robot is based on mathematical equations allowing it to execute a processor instruction to perform a mechanical movement (forward, backward, turning). This locomotion can be translated into a mathematical model called kinematic. Kinematics is the mathematical study of motion. In a case of kinematics studying, the generating forces associated with locomotion of the robot are not considered. One can also express the locomotion of a robot from a dynamic model that is an important principle in the study of locomotion. Unlike kinematics, the dynamics of a mobile robot take into account all the forces allowing the robot to move according to the mass and dimensions of the robot. These forces are the set of speed and energy associated with the movement of the robot. The study of the dynamics is a very important process since it makes it possible to determine the necessary power that must provide an electric motor for the control of a robot, which is a capital information in the choice of the electric motor which will be used to drive the robot.

### **2.5.1. Kinematic of the two wheeled mobile robot**

Kinematics is the mathematical study of a mechanical movement without taking into consideration the forces that generate this movement. Kinematics is used in differential drive control. Most wheeled mobile robots use the differential drive mechanism. The principle of the differential drive, is that two fixed wheels are mounted on the same axis and the speed of each wheel is controlled independently to forward, backward or turn the robot. In another word, to modify the robots' pose (location and orientation). During the operation of the robot, the speed of each wheel is varied to change the direction of the robot. It is assumed that the robot rotates in a unique circle around a fixed point that extends along a common axis with the left and right wheels. Case in point for a robot with two wheels. The point around which the robot turns is called Instantaneous Center of Rotation (ICR). The ICR is the center of an imaginary circle of radius  $R$  that the robot travels. Thus, we can estimate the position and orientation of the robot with the relationship between the linear velocity  $V$ , the angular velocity  $\omega$

and the distance between the ICR point and the traveled path by the wheels on each side of the robot like is shown in Figure 2.9. [14].

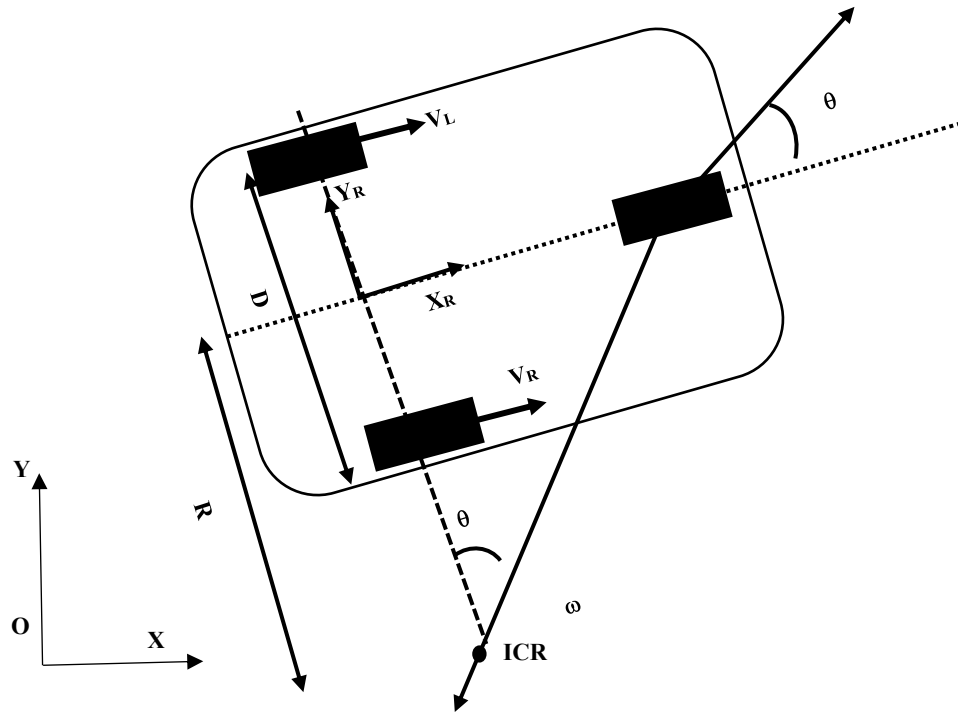


Figure 2.9. Kinematic model of two wheeled mobile robot

Where ICR: Instantaneous Center of Rotation, with a radius of rotation  $R$ ,  $D$  is the distance between the two rear wheels.  $V_L$ ,  $V_R$ , are respectively the linear speeds of the left and right rear wheel and  $\theta$  the orientation angle of the robot. By changing the speed of the wheels, we can change the trajectory taken by the robot. Since the rotation  $\omega$  around the ICR point is a constant for both wheels. Thus, the linear speed equation of the rear wheels can be written as the product of the angular velocity  $\omega$  and the scanning radius of each wheel as following:

$$\omega \left( R - \frac{D}{2} \right) = V_R \quad (2.26)$$

$$\omega \left( R + \frac{D}{2} \right) = V_L$$

Where  $D$  is the distance between the center of the two wheels,  $V_R$  and  $V_L$  are respectively the speed of the right and left wheels,  $R$  is the distance between the instantaneous center of rotation ICR and the midpoint of the segment  $D$  and the midpoint of the axis between the two wheels.  $R$  and  $\omega$  can be expressed as follows depending on the linear velocities of each wheel and  $D$ .

$$R = \frac{D}{2} \frac{V_R + V_L}{V_L - V_R} \quad ; \quad \omega = \frac{V_L - V_R}{D} \quad (2.27)$$

In this case, we can control three possible cases that are described below.

If  $V_L = V_R$ , then we have a linear motion in front of the robot on a straight line.  $R$  is equal to infinite and  $\omega$  is equal to zero, which translates as, no rotation of the robot.

If  $V_L = -V_R$ , then  $R = 0$  and we have a rotation around the center point of the wheel axis, thus, the robot turns on the same place.

If  $V_R = 0$ , then we have a rotation around the right wheel. In this case,  $R = d / 2$ . The same effects are identical if  $V_L = 0$ , this time the rotation will be around the left wheel. Robots based on differential driving are sensitive by a small change in the speed of each wheel. The wheel speed error can influence the robot's trajectory. Also the robots using the differential drive are sensitive to the difference of the ground plans.

### 2.5.2. Inverse kinematic for mobile robot

One can express in a general way the pose of a robot by using the inverse kinematics equations. The position of the robot moving in a particular direction  $\theta_t$  at a given speed  $V(t)$  is expressed in the following equations.

$$x(t) = \int_0^t V(t) \cos[\theta(t)] dt$$

$$y(t) = \int_0^t V(t) \sin[\theta(t)] dt \quad (2.28)$$

$$\theta_t = \int_0^t \omega(t) dt$$

### 2.5.3. Differential steering speed for four wheeled mobile robot

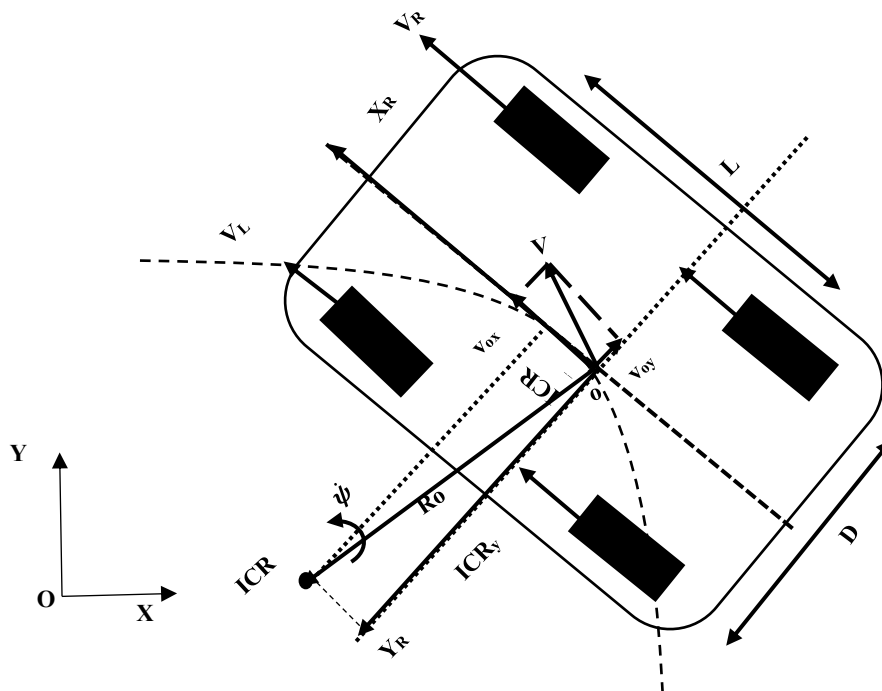


Figure 2.10. Differential steering speed for four wheeled mobile robot

ICR is the instantaneous center of rotation as we defined previously. But the difference of this model is that unlike the previous model, where the ICR extended on the same axis as the rear wheels, here the ICR is parallel to the axis  $Y_R$  [15- 17] like is shown in Figure 2.10. above. So, in the local coordinate system of the robot, the longitudinal location  $ICR_x$  and the lateral location  $ICR_y$  respect the following constraint:



$$ICR_x = -\frac{v_{oy}}{\dot{\psi}_o} \quad (2.29)$$

$$ICR_y = \frac{v_{ox}}{\dot{\psi}_o}$$

Where  $v_{ox}$  and  $v_{oy}$  are the longitudinal velocity and lateral velocity of the robot's center of mass. Let's set  $V_L$  and  $V_R$ , as the linear speed of the left and right wheels. Therefore, the radius of rotation can be calculated according to each triangle formed by the following equations.

$$v_{ox} = \frac{V_L + V_R}{2}$$

$$\dot{\psi}_o = \frac{V_L - V_R}{D} \quad (2.30)$$

$$ICR_y = \frac{D V_L + V_R}{2 V_L - V_R}$$

The radius of the instantaneous center of rotation (ICR) of the robot can be calculated as following:

$$R_o = \sqrt{ICR_x^2 + ICR_y^2} \quad (2.31)$$

We can summarize all of these equations in the local frame of the robot as following:

$$\begin{bmatrix} v_{ox} \\ v_{oy} \\ \dot{\psi}_o \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ ICR_x & -ICR_x \\ \frac{D}{1} & -\frac{D}{1} \end{bmatrix} \begin{bmatrix} V_L \\ V_R \end{bmatrix} \quad (2.32)$$

Wheel sliding plays an important role in the control of differential drive. It can be seen that during a turn, the steering angle of the robot's trajectory has a non-linear relationship with the different wheel speeds on each side of the robot. It is difficult to determine the desired turn angle in this nonlinear relationship. So, to solve this problem, Xiaodong Wu et al, introduced in their article [18] the notion of a non-dimensional coefficient, in order to create a relationship between the steering angle and the rotational velocity of the robot wheels. This non-dimensional coefficient denoted  $\lambda$  is the ratio of the sum and the difference of the angular rotation velocity of the left and right wheels. This coefficient is expressed as below.

$$\lambda = \frac{\omega_R + \omega_L}{\omega_R - \omega_L} \quad (2.33)$$

According to the results of the experiments carried out by Xiaodong et al [18], the steering angle is a linear trajectory with the parameter  $\lambda$ . Consequently the linear steering angle is defined as following:

$$R = K\lambda = K \frac{\omega_R + \omega_L}{\omega_R - \omega_L} \quad (2.34)$$

Where K is a proportionality coefficient determinable by calculation with experiment data with the curvature function. If  $\omega_R = \omega_L$ , then  $R = \infty$ , the robot moves in a straight line. If  $\omega_R = -\omega_L$ ,  $R=0$ , the robot rotates in place around its central axis without longitudinal movement.

Since the steering angle is now coupled to the speed of the wheels on both sides, the steering is now sensitive to the speed of the vehicle. During a change of direction, the acceleration or deceleration with a variation of the speed can create a phenomenon of oversteering or understeering. Hence, to improve cornering enhancement, Xiaodong et al [18] have defined the speed variation on each side as,  $\Delta_x$  and  $\Delta_y$  to maintain the steering angle stable.

$$\frac{\omega_R + \Delta_x + \omega_L + \Delta_y}{\omega_R + \Delta_x - \omega_L - \Delta_y} = \frac{\omega_R + \omega_L}{\omega_R - \omega_L} \quad (2.35)$$

If the speed variation  $\Delta x$  and  $\Delta y$  satisfy the proportionality relation above, the steering angle will be stabilized and unchanged. So, the equation (2.35) has been simplified and named SSC (Steering Speed Control) [18].

$$\frac{\Delta_x}{\Delta_y} = \frac{\omega_R}{\omega_L} \quad (2.36)$$

#### 2.5.4. Dynamics model of four wheeled mobile robot

Unlike kinematics, which does not take into account forces associated to robot's locomotion, dynamics is the studying of the mechanical movement that taking into account the different forces applied to generate the robot's motion. The dynamic model is very important for the analysis of a simulation and the modeling of many control algorithms. The Figure 2.11. below illustrates the dynamics model for four wheeled mobile robot.

It is assumed that the center of mass of the vehicle is the center point of the whole vehicle. As a result, the 3 DOFs (degrees of freedom) of the equation of the dynamics  $L(x, y, z)$  of the whole vehicle can be expressed in the following equations, where, the indices  $x$  and  $y$  ( $F_{Lfx}$ ,  $F_{Rfy}$ ), indicates the longitudinal and lateral forces. The first letter of each force  $F$ , refers to left (L) or right (R) and the second letter to front (f) or backward (r). For example,  $F_{Lfx}$ , is the longitudinal force actuated on the left front wheel,  $m$  and  $I$  are the mass and the moment of inertia of the robot, and finally,  $D$  and  $L$  are respectively the width and length of the robot.

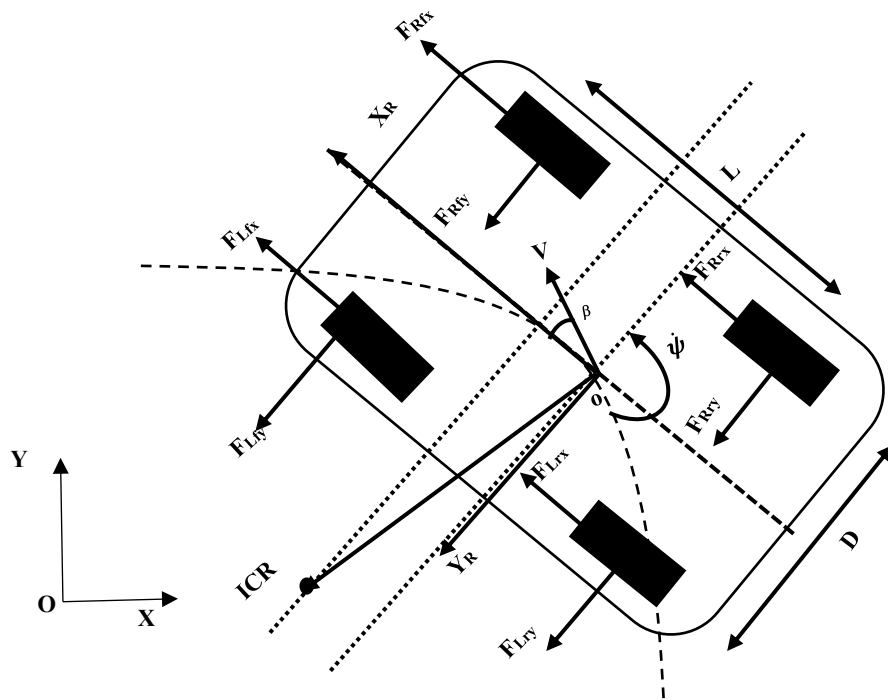


Figure 2.11. Dynamics model for four wheeled mobile robot

$$m\ddot{x} = F_{Lfx} + F_{Rfx} + F_{Lrx} + F_{Rrx} + m\dot{\psi}_o \dot{y}$$

$$m\ddot{y} = F_{Lfy} + F_{Rfy} + F_{Lry} + F_{Rry} + m\dot{\psi}_o \dot{x}$$

(2.37)

$$I\ddot{\psi}_o = (F_{Lfx} + F_{Rfx} + F_{Lrx} + F_{Rrx})\frac{D}{2} (F_{Lfy} + F_{Rfy} + F_{Lry} + F_{Rry})\frac{L}{2}$$

$\beta$  is the angle formed between the axis of the robot body and the direction of movement of the robot. We can say that  $\beta$  is the wheel slipping angle equivalent to the angle formed by the direction of robot movement and the surface revolution of the wheels. Suppose that,  $r$  as the radius of the wheels, the longitudinal and lateral force can extend as following.

$$J\dot{\omega} = T - F_x r \quad (2.38)$$

$$F_y = C_\alpha \beta$$

Where, T is the driving torque/braking of the engine J is the inertia of the wheel,  $\omega$  is the angular velocity of the wheel,  $C_\alpha$  is the stability power of each wheel.

## 2.6. Literature Review

### 2.6.1. History of the robotics

Robotics, since the early civilization was one of man's ambition has been to create artifacts in their image. So one, can see in the legend of Titan Prometheus, who modelled humankind from clay, or that of giant Talus, the bronze slave forged by Hephaestus (3500 BC), testify to this quest in the Greek mythology. The Egyptian's oracle statues hiding priests inside (2500 BC) were perhaps the processor of man modern thinking machine. The clepsydra water clock introduced by Babylonians (1400 BC) was one of the first automated mechanical artifacts [2].

The concept of the physical robot was established during the course of twentieth century. In 1920, the term 'robot' derived from 'robota' which means subordinate labour in Slav language. It was first introduced by the Czech playwright Karel Capek in his play "Rossum's Universal Robots". In 1940, Isaac Asimov, the Russian science-fiction writer in his novel "Runaround", introduced the ethics of interaction between robot and human was envisioned to be governed by his well-known three fundamental laws. The early robot built in 1960s, stemmed the union of two technologies (numerical control for precise manufacturing and tele operator for remote radioactive material handling). These robots named industrial robots, became essential components in the automation of flexible manufacturing in 1970s. In 1980s, robotics was defined as the science which studies the intelligent connection between perception and action. In 1990s, research was boosted by the need to resort to robot to address human safety in dangerous environment, or to increase the human operator ability and

reduce human tiredness, or else by the desire to develop products with wide potential markets aimed at improving the quality of life.

Today, robot plays an important I many field like medicine, automotive where the intelligent vehicle is main point of interest. Give an ability of robot to interaction closer to human are the challenge of the tomorrow's robots.

### **2.6.2. Vehicle ego-motion estimation (Visual Odometry)**

The history of the vehicle pose estimation based on vision has been proposed for the first time [3] by Moravec in 1980s described in [19] The works of Moravec was the motion-estimation pipeline which the main functioning block, testify are still used today and describing the earliest corner detector after the first one proposed in 1974 by Hannah [20] which is denoted today by Moravec corner detector [21], processor of one presented by Fostner [22] and Harris and Stephens [23 - 24]. Most of the early research [21, 25] was done for planetary rovers and was motivated by the NASA Mars exploration program to prove all terrain rover that have an ability to measure their 6-degrees of freedom (DOF) motion in the presence of wheel slippage uneven and rough terrain. Because of the visual odometry has based on processing of image from a camera and scene reconstruction. Many algorithms used in image processing fields are important in for the good scene reconstruction in visual odometry system modelling.

There are many algorithms used to estimate a camera parameters and scene reconstruction. Sutherland used the original DLT application for a camera computation [26]. Hartley has highlighted the term normalization in computer vision in [27]. Slama spoke of the case of minimizing geometric errors in his photogrammetry manual [28]. Torr and Murray [29] presented the very first application mentioning the use of robust estimation algorithms for the estimation of a fundamental matrix using RANSAC; Zhang, Deriche, Faugeras and Luong [30] instead used LMS. Fischler and Bolles uses the RANSAC robust estimation algorithms to solve the pose problem from 3 points in computer vision [12]. Torr and Zisserman have described the automatic ML estimation of homographic [31]. Criminisi Reid and Zisserman, gave numerous examples for

calculating the covariance of the transfer points as a match for determining homographic by varying the number and the position in [32]. Zeller [33] has studied the problem of calibrating a perspective, affine and Euclidean projection camera. In contrast to him, Sturm, Vieville and D. Lingrand, opted for the uncalibrated camera analysis respectively [34 - 35]. Sturm, used an uncalibrated and a self-calibrated camera for Euclidean reconstruction for a mobile mono-camera [36] Springer focused on geometry and analysis in a perspective space [37]. Devernay and Faugeras [38] , in the same context, have evoked the problem of reconstructing an image from a projective projection to a Euclidean coordinate. Wolfe and Mathis, Tsai and Huang have contributed in the context of a three-view perspective respectively [39 - 40]. The multiple perspective view of a camera in motion for Euclidean has been investigated by Christy and Horaud [41]. In the same way, Richard Hartley and Zisserman approached the geometry of the camera [4] Finally, Tordoff and Murray invested in the reactive zoom control properties of an affine camera to track an object [42].

## **2.7. Summary**

This chapter allowed the description of the first part of the used material and method. In this section, we have strengthened the theoretical bases used for the design of a mobile robot, from used sensor to the mobile robot locomotion (kinematics, dynamics). In the next chapter the second part of materials and methods based especially on estimation algorithms such as Kalman Filter have been considered.

## **CHAPTER 3. SECOND PART OF METHODS AND MATERIALS**

### **3.1. Introduction**

In this chapter, the continued theoretical part of materials and methods used in this thesis have been investigated, the first part was discussed in the previous chapter (Chapter 2). This part is principally based on the different algorithms used. It will be questions of algorithms using to process the information provided by the sensors that allow the mechanical block of the robot to move and estimate its position and orientation. These algorithms are the Kalman filter, the particle filter and so on.

The Kalman filter which is a powerful estimator for the estimation of linear systems is the first choice among the algorithms highlighted during this research. However, since the evolution of the robot can raise problems of non-linearity, for example during a turning, the steering radius is a non-linear curvature. The Extended Kalman Filter and the Particle Filter will be used for their ability to analyze non-linear systems. The particularity of the EKF is as mentioned above its ability to study the cases of non-linearity of system, but this filter is very expensive in implementation. Besides the Extended Kalman Filter, the Particle Filter is an ideal and adequate answer for non-linearity issues. It can offer a much more satisfying result than the standard and extended Kalman filter in the case of non-linear systems studying. But the big problem of the Particle Filter, is unlike the standard and Extended Kalman Filter, it is very expensive in implementation. So one has to use the Particle Filter when one is sure of ineffectiveness of the Kalman filters.

To compensate the measurement errors of the robot's pose, different sensors have been combined to obtain a more reliable results. Thus the Decentralized Kalman Filter (DKF) is the algorithm used to merge all of the data provided by these different sensors. Based on the principle of decentralized data fusion algorithm, the DKF is a



parallel form of Kalman Filter allows the implementation of multi-sensor or multi-tracking system. Because of the use of different sensors during this research, the multi-sensor technology have been interested and one was also focused on its different configurations and architectures. The multi-sensor system is a technique that allows the design of a more reliable dynamic system for locating the robot, prompt detection and obstacle.

The rest of this chapter is organized in the following way, an overview has been established on the algorithms of data fusion with its different configurations. After which, the presentation of the Kalman Filter and then of the Particle Filter. At the end of the chapter, a review of the literature and the partial conclusion of this chapter will close this section.

### **3.2. Data Fusion Algorithm**

In a system using more than one sensor, the major problem is the fusion of data after each processor has finished its own estimate and must transfer its data to the central processor for the final decision. Data fusion is the method of combining data from different sensors. As in a multi-sensor system, the objective is to evaluate the measurements of the same phenomenon with different sensors to obtain a better result. So the data fusion in other words, is the method allowing the condensation of all measurements provide by each processor in a single instruction. Fusion algorithms can be classified into three categories: centralization, hierarchical and decentralization structures. Among these three without neglecting the effectiveness of the first two cities, the decentralized configuration is the most adequate in a multi-sensor system. Because in a decentralized structure, the processors evolve independently without the possibility of data exchange between the processors of different subroutines. These subroutines are called local filter. The local filters are the particularity of the decentralized structure, because of the capacity of these filters to estimate themselves their own data and transmit it to the master processor.

### 3.2.1. Multi sensor system

In view of the more evolved and complex system design used by the applications developed today by engineers, single sensor usage has become unable to provide quality and satisfactory information. So, to offset for this defect, the use of multi-sensors have made their appearances. Multi-sensors have become inevitable in the complex applications that have become demanding in terms of quantity of data and its precision. Multi-sensor systems are the combination of different types of sensors that can provide information in a real-time environment for optimizing a system. One can find multi-sensors in the field of robotics, navigation, telephony, aeronautics, military applications and many other fields. Unlike a single-sensor system, multiple sensors offer several advantages over both reliability and accuracy of the system. The principle is to measure the same phenomenon with several sensors of the same or different technologies sensors, in fact to provide a satisfying measurements. This mode of use of sensors allows a better evaluation of the errors of measurement and the robustness of the system.

The multi-sensors are much more advantageous than the use of single sensor, in the sense that, if one of the sensors was to stop working, the system meanwhile will be able to function. Although the breakdown of this sensor will affect the reliability and accuracy of the system, but it cannot cause an absolute shutdown of the system. Also, since one can use several sensors of different technologies, the choice of sensors becomes extensive and it can allow the designer to complement the sensors on the basis of their price, a factor that can allow the design of a cheaper applications. However, multi-sensors are not without drawbacks, the disadvantage of the implementation of such a system requires the resolution of the problem of associated data and also the order of communication of the sensors with the central system (master). One answer of this problem is a data fusion algorithm introduced above.

### 3.2.2. Centralized data fusion architecture

In the centralized data fusion configuration, all the sensors are directly connected to a main and only one processor. So, the processor reads the state of each sensor and processes the information provided by these sensors then decides what should be for the proper functioning of the system. This is the simplest of the three configurations.

However, this configuration shows some limitations. In the case where the processor would no longer work, the whole system would fail because the processor is the main communication node. The Figure 3.1. below illustrates the centralized architecture of a data fusion. The centralized architecture calls a big computation problem [7]. Since the processor is the only one that has to do everything by itself in the system.

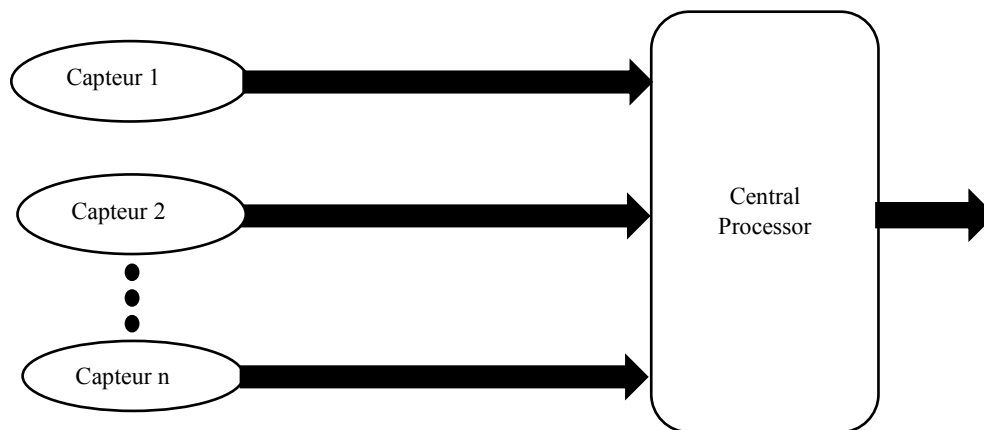


Figure 3.1. Centralized data fusion architecture

### 3.2.3. Hierarchical data fusion architecture

The Figure 3.2. below shows the hierarchical configuration of the data fusion algorithm. It solves the problem of direct connections of the sensors to the processor and also the computation problem by providing a hierarchical transmission data to the main processor. In a hierarchical configuration, one has a central processor, a fusion center and sensors.

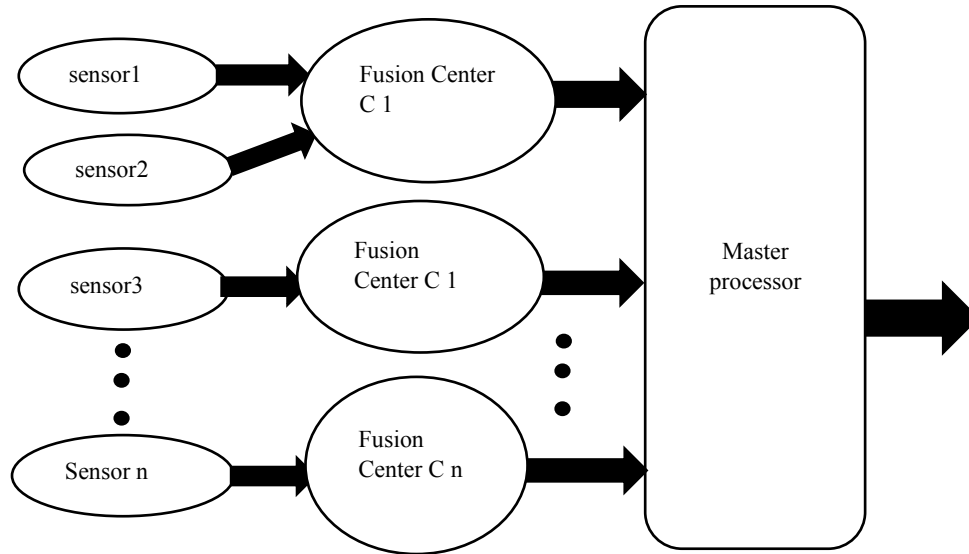


Figure 3.2. Hierarchical data fusion architecture

The fusion centers locally process the information received from the sensors and transmit it to the central processor. Many applications nowadays are structured on the hierarchical data fusion model. Among other, one can mention robotics, remote monitoring and much more [43]. This configuration offers the advantage by distributing the loads in the implementation. However, one can observe certain disadvantage of this configuration, which is based on the model of the centralized fusion. In addition to this problem, there is also the vulnerability of the hierarchical model in terms of overcrowding in communication [44 - 45].

#### 3.2.4. Decentralized data fusion architecture

The use of the complete decentralized data fusion model is a resolution of the said problems of the both previous models namely hierarchical and centralized models [46]. The advantage offered by a completely decentralized model is, among other things, one of the reasons for choosing this model of decentralized data configuration in this thesis. The Figure 3.7. gives a typical illustration of this model.

In contrast to the both previous models, in a decentralized structure, the processors evolve independently without the possibility of data exchange between the processors of different subprograms. These subprograms are the local filters. The local filters are

the meticulousness of the decentralized structure of data fusion algorithm. Because these filters have the ability to estimate by themselves their own data and transmit these estimated data to the main processor or master.

### **3.3. Kalman Filter**

The Kalman Filter was developed by Rudolf E. Kalman in 1960 and used for the first time in the Apollos project. This filter is based on fundamental principles of mathematics, statistics, probability, and theory of dynamic systems study [47]. These mathematical theoretical background of Kalman filter are illustrated in Figure 3.3. below.

The Kalman filter is an optimal algorithm for estimating. Theoretically, it is used to solve linear-quadratic problems, which are themselves instantaneous problems of the state of a dynamic system disturbed by a white noise (Gaussian noise). These linearly reported measurements of this noisy state are used. The result obtained is an optimized statistical estimator that meets all expectations of a quadratic function of an error estimate.

In practice, the Kalman filter is one of the greatest discoveries in history that has allowed human to create incredible systems that may not have been possible without this filter. It is used in very complex dynamic system control applications such as aviation systems, offshore navigation, military applications etc. The Kalman filter is much more common in applications such as object avoidance, navigation systems, computer vision systems, signal processing and image processing. In all these applications, the Kalman Filter is used as an estimation tool and a powerful analysis estimator. It is also used for the prediction in the future of the behavior of a dynamic system whose human cannot have direct access to its control such as the trajectory of stars in space for example.

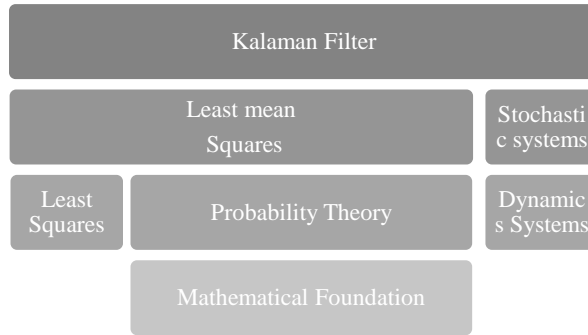


Figure 3.3. Kalman Filter's mathematical foundation [47].

### 3.3.1. Standard Kalman Filter

The block diagram Figure 3.4. below describes the general state space equations of a dynamic system.

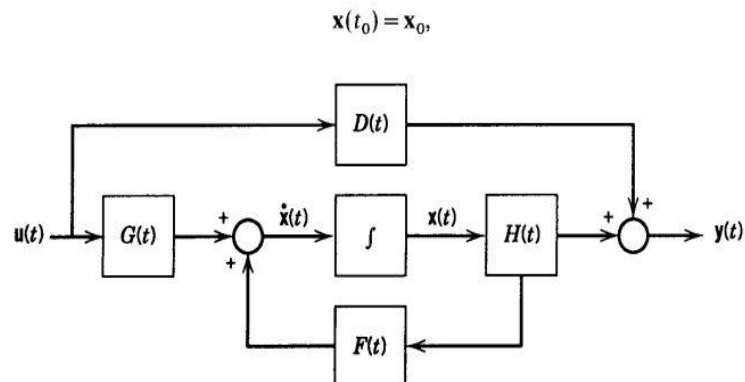


Figure 3.4. Dynamic system state space block diagram [7].

$$\dot{x}(t) = F(t)x(t) + G(t)u(t) \quad (3.1)$$

$$y(t) = H(t)x(t) + D(t)u(t) \quad (3.2)$$

$\dot{x}(t)$  is the state vector,  $F(t)$  is the state transition matrix,  $G(t)$  is the control matrix,  $y(t)$  is the measurement vector and  $H(t)$  is the matrix of measurement, generally,  $D(t)$  equals to 0. For the rest of the notation of this thesis,  $F(t)$ ,  $G(t)$ ,  $y(t)$  will be respectively replaced by  $A$ ,  $B$  and  $Z$ .

From the general model of the state equation of a linear time-varying system, the Kalman filter can be formulated in the following manner by adding measurement noise and process disturbance.

$$X_t = AX_{t-1} + Bu_t + W_t \quad (3.3)$$

$$Z_t = HX_t + V_t \quad (3.4)$$

The Kalman Filter algorithm is divided into two main parts as follows: prediction and update of measurements.

The prediction step

$$X_t = AX_{t-1} + Bu_t + W_t \quad (3.5)$$

$$P_t = AP_{t-1}A^T + Q_t$$

The update step

$$K = \frac{P_t H}{HP_t H^T + R} \quad (3.6)$$

$$X_t = X_{tp} + K[Z_t - HX_{tp}]$$

Where,  $X_t$ : state vector  $t$  describing the system concisely (speed, position, acceleration etc.),  $u_t$ : input control vector (orientation angle);  $A$ : state transition matrix between  $t-1$  and  $t$ ;  $B$ : input control matrix;  $W_t$ : state noise vector;  $Z_t$ : measurement vector. The Figure 3.5. below describes the algorithm of Kalman filter for more comprehending of concept of this filter.

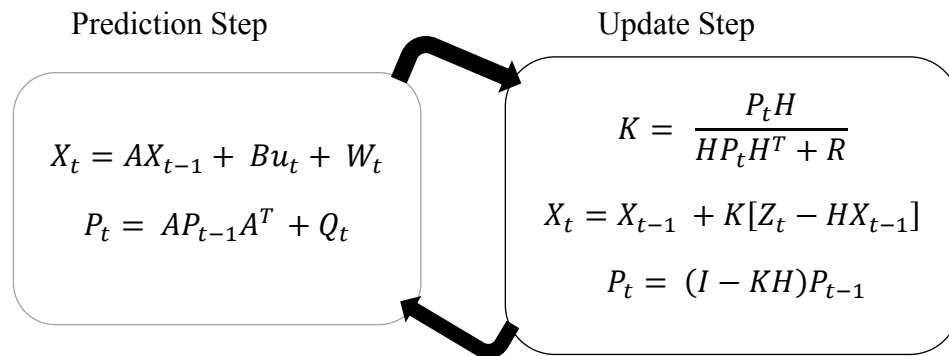


Figure 3.5. Kalman Filter algorithm

In addition to the standard Kalman filter, there are several other types of Kalman filter such as Decentralized Kalman Filter (DKF), Extended Kalman Filter (EKF).

### 3.3.2. Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) is the filter used for nonlinear systems, recalled that the standard Kalman filter is only reliable for linear systems. The EKF has been developed to fill this gap to also allow the studying of the nonlinear systems. The EKF is generally applied in the localization of mobile robots because of the fact that, the relation of nonlinearity observed during a robot turning. It is clear that the steering angle is not a linear curvature. For this fact, the linearization of that will be necessary for more reliable estimation. So the EKF plays the important role in a robot pose estimation. Therefore, the different equations used in the EKF are expressed as following [6].

$$X_k = f(X_{k-1}, u_{k-1}, W_{k-1}), \quad (3.7)$$

$$Z_k = h(X_k, V_k) \quad (3.8)$$

In practice, we do not know the value of noise  $W_k$  and  $V_k$ . For this, one can express the equations of state and measurement without taking them into account.



$$\hat{X}_k = f(\hat{X}_{k-1}, u_{k-1}, 0) \quad (3.9)$$

$$\hat{Z}_k = h(\hat{X}_k, 0) \quad (3.10)$$

Where  $X_k$ , is the posteriori estimate of the state, of the previous step of  $k$ .

$$X_k \approx \hat{X}_k + F(X_{k-1} - \hat{X}_{k-1}) + TW_{k-1} \quad (3.11)$$

$$Z_k \approx \hat{Z}_k + H(X_k - \hat{X}_k) + DV_k$$

Where  $T$  is the Jacobean matrix of the partial derivative of the function  $f$  respect to  $W$ , and  $F$  the Jacobean matrix of function  $f$  respect to  $X$ .  $H$  is the Jacobean matrix of the partial derivative of function  $h$  respect to  $X$  and  $D$  the Jacobean matrix of function  $h$  respect to  $V$ .

The Jacobean matrix  $F$  can express as following:

$$F_{[i,j]} = \frac{\partial f_{[i]}}{\partial X_{[j]}}(\hat{X}_{k-1}, u_{k-1}, 0) \quad (3.12)$$

The Jacobean matrix  $T$  can express as following:

$$T_{[i,j]} = \frac{\partial f_{[i]}}{\partial W_{[j]}}(\hat{X}_{k-1}, u_{k-1}, 0) \quad (3.13)$$

The Jacobean matrix  $H$  can express as following:

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial X_{[j]}}(\hat{X}_k, 0) \quad (3.14)$$

The Jacobean matrix  $H$  can express as following:

$$D_{[i,j]} = \frac{\partial h_{[i]}}{\partial V_{[j]}}(\hat{X}_k, 0) \quad (3.15)$$

The Figure 3.6. below gives a summary of the heard Kalman filter by highlighting the equations of state prediction and measurement correction.

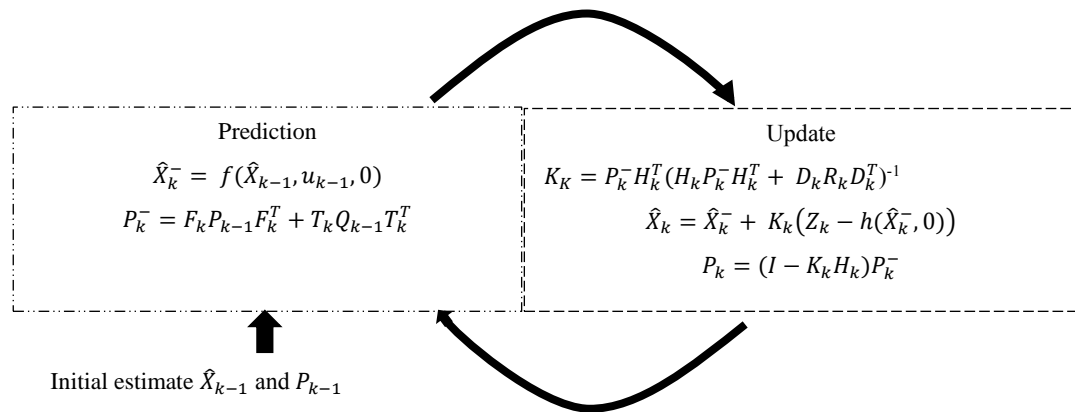


Figure 3.6. EKF operating algorithm

### 3.3.3. Decentralized Kalman Filter (DKF)

#### 3.3.3.1. Presentation of DKF

The Decentralized Kalman Filter can be defined as the Kalman Filter based on the decentralized model of the data fusion algorithm. In another word, the DKF is an answer the problem of data merging of multi-sensor the data fusion. The structure of the Decentralized Kalman filter is shown in Figure 3.7. below. The principle is to operate Kalman Filter independently of each other through the local filter, containing of the Kalman algorithm and the sensors. Each filter operates independently of other local filters and then reports to a central filter called master or main filter. So, one creates a parallel execution of the system simulating the idea of multitasking used in computers.

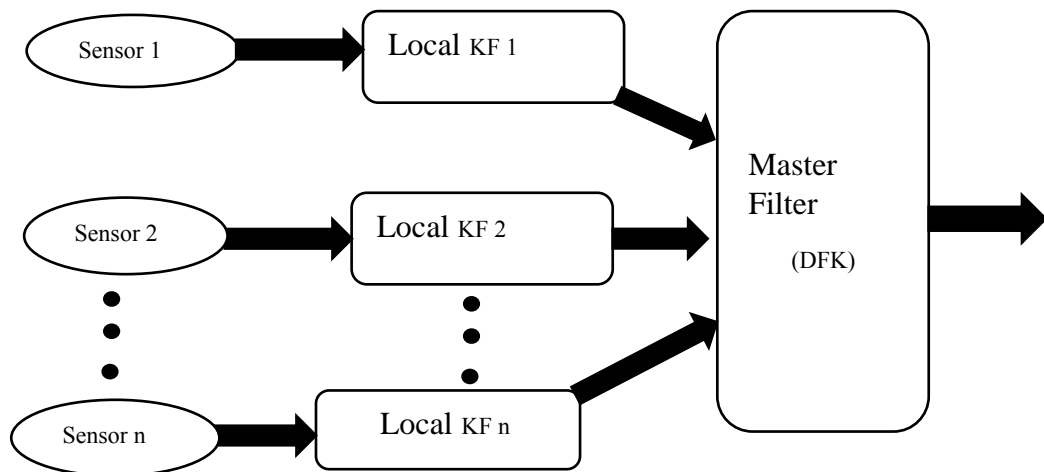


Figure 3.7. Structure of Decentralized Kalman Filter (DFK)

### 3.3.3.2. Problem formulation

Consider the discrete model of dynamic system as follows [48 - 49]:

$$X_i(k + 1) = A_i(k + 1, k)X_i(k) + G_i(k + 1)W_i(k + 1) \quad (3.16)$$

It is assumed that there is a measurement  $Z(k)$ , observing the linear combination of the state  $X(k)$  at a time  $k$  in the presence of the Gaussian noise. Therefore  $Z(k)$  can be expressed as follows:

$$Z_i(k) = C_i(k)X_i(k) + V_i(k), (i=1, 2, \dots, n) \quad (3.17)$$

Where,  $n$  is the sensor number;  $X_i(k)$  is the state vector of the  $i$ -th sensor;  $Z_i(k)$  is the measurement vector of  $i$ -th sensor;  $C_i(k)$  is the matrix of measurement corresponding to the  $i$ -th sensor;  $W_i(k+1)$  is the process noise of the  $i$ -th sensor,  $G_i(k+1)$  is the process noise matrix and  $V_i(k)$  is the measurement noise of the  $i$ -th sensor. The process noise  $W_i(k+1)$  and the measurement noise  $V_i(k)$  are assumed to be Gaussian white noise with zero mean and the standard derivative covariance  $Q$  and  $R$  respectively.

Since it not possible to observe directly the state of  $X(k)$ , the Kalman filter is an ideal tool used to estimate the state of  $\hat{X}(k)$ .

$$\hat{X}_f(k) = E(X(k)/Z_1(j), \dots, Z_k(j) \text{ } j=1, \dots, k) \quad (3.18)$$

As the Decentralized Kalman Filter is constitute of the local and the master filter, on can implement the global model of the Kalman filter based on a global model of all the measurements [50]. Therefore one can express this global model as follows:

$$X(k + 1) = A(k + 1, k)X(k) + G(k + 1)W(k + 1) \quad (3.19)$$

$$Z_i(k) = C_i(k)X_i(k) + V_i(k) , (i=1, 2 \dots n)$$

The local model of the Kalman filter is as follows:

$$X_i(k + 1) = A_i(k + 1, k)X_i(k) + G_i(k + 1)W_i(k + 1) \quad (3.20)$$

$$Z_i(k) = H_i(k)X_i(k) + V_i(k) , (i=1, 2 \dots n)$$

It is assumed that there is exists a matrix  $M_i$ , where  $i=1, \dots, n$  such that:

$$C_i(k + 1) = H_i(k + 1)M_i(k + 1) \quad (3.21)$$

$$Z(k) = \begin{bmatrix} Z_1(k) \\ \dots \\ Z_k(k) \end{bmatrix} \quad (3.22)$$

$$C(k) = \begin{bmatrix} C_1(k) \\ \dots \\ C_k(k) \end{bmatrix} \quad (3.23)$$

$$R = blkdiag(R_1(k), \dots, R_k(k)) \quad (3.24)$$

$$P_f(k + 1/k) = A(k + 1, k)P_f(k/k)A(k + 1, k)^T + G(k + 1)Q(k + 1)G^T(k + 1, k) \quad (3.25)$$

$$P_f(k + 1/k + 1) = \left[ I - \sum_{i=1}^k K_i(k + 1)C_i(k + 1) \right] P_f(k + 1/k) \quad (3.26)$$

Where  $K_i$  is the global gain of the Kalman filter expressed as follows:

$$K_i(k + 1) = P_f(k + 1/k)C_i^T(k + 1) \times [C_i(k + 1)P_f(k + 1)C_i^T(k + 1, k) + R_i(k + 1)]^{-1} \quad (3.27)$$

$$\Phi_f(k + 1) = \left[ I - \sum_{i=1}^k K_i(k + 1)C_i(k + 1) \right] A(k + 1, k) \quad (3.28)$$

$$T_i(k + 1) = \Phi_f(k + 1)F_i(k) - F_i(k + 1)\Phi_{if}(k + 1) \quad (3.29)$$

$$F_i(k + 1) = P_f(k + 1/k + 1)M^T_i(k + 1)P_{if}^{-1}(k + 1/k + 1) \quad (3.30)$$

$$\Gamma(k + 1/k + 1) = \Phi_f(k + 1)\Gamma(k/k) + \sum_{i=1}^k T_i(k + 1)\hat{X}_{if}(k/k) \quad (3.31)$$

$$\hat{X}_f(k + 1/k + 1) = \Gamma(k + 1/k + 1) + \sum_{i=1}^k F_i(k + 1)\hat{X}_{if}(k + 1/k + 1) \quad (3.32)$$

The estimation of the state of the different sensor can be calculated in a local Kalman filter in the following way:

$$P_{if}(k + 1/k) = A_i(k + 1, k)P_{if}(k/k)A_i(k + 1, k)^T + G_i(k + 1)Q_i(k + 1)G_i^T(k + 1, k) \quad (3.33)$$

$$P_{if}(k + 1/k + 1) = [I - K_{if}(k + 1)H_{if}(k + 1)]P_{if}(k + 1/k) \quad (3.34)$$

Where  $K_{if}$  is the partial gain of the each local Kalman filter is as the following form:

$$K_{if}(k + 1) = P_{if}(k + 1/k)H_i^T(k + 1) \times [H_i(k + 1)P_{if}(k + 1)H_i^T(k + 1) + R_i(k + 1)]^{-1} \quad (3.35)$$

$$\Phi_{if}(k + 1) = [I - K_{if}(k + 1)H_{if}(k + 1)]A_i(k + 1, k) \quad (3.36)$$

The estimation of the state vector locally is as follows:

$$\hat{X}_{if}(k + 1/k + 1) = \Phi_{if}(k + 1)\hat{X}_{if}(k/k) + K_{if}(k + 1)Z_i(k + 1) \quad (3.37)$$

### 3.4. Literature Review

The Kalman filter is the result of the evolution of an ideology and thought advocated by many scientists since the beginning of time. In this literature review, we will present some contributions that have led to the birth of the concept of optimal estimation, until reaching the Kalman filter since the history of mathematics [47]. It is known that the list of these scientists is not exhaustive in this presentation. We have listed the best known and the most important in terms of revolutionary contribution. Because of the number of these scientists who have marked history a simple table cannot summarize all them. The Figure 3.8. below gives an overview of the times and period of evolution of the estimation. For a much more advanced detail on the optimal estimation history, one can refer to the following authors, Kailatch [51 - 52], Lainiotis [53], Mendel and Gieseking [54], Sorenson [55 - 56], Battin [57] and Schmidt [58].

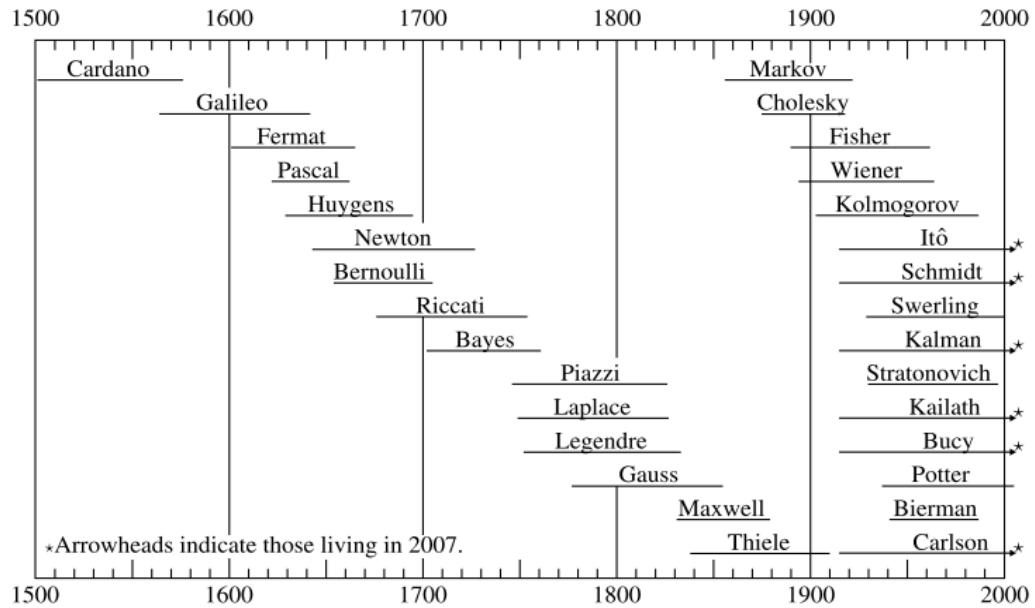


Figure 3.8. Chronology of some contributors to estimation technology [47].

The first implemented method for an optimal estimate from noisy data is the least square method. Its discovery was attributed in 1795 to Carl Friedrich Gauss (1777-1855). The inevitability of measuring errors had been recognized since the time of Galileo Galilei (1564-1642), but it was the first official method of approaching the subject. Although it was more used for a linear problem estimation, Gauss is the one he used for the first for a non-linear problem for solving an estimation problem in astronomical mathematics. This has been an important part of the history of astronomy. This has been attested in several sources among them that of Makemson [59].

In this same period, the method of least squares was independently discovered and published by Andrien-Marie Legendre (1775-1833) in France, and Robert Adrian (1775-1855) in the United States [51]. It is also known that the least squares method was used even before the birth of Gauss by Swiss-German physicist Johan Heinrich Lambert (1728-1777). It must be said that the phenomenon of the simultaneous discovery of the theory of estimation will not make the exception of the Kalman filter. Since the same coincidences have been repeated between the Wiener-Kolmogorov filter and the Kalman filter [47]. Indeed, Norbert Wiener (1894-1964) is one of the famous researchers as Kalman marked the twentieth century by his discovery of his filter called Winner Filter.

Rodolph Emile Kalman born May 19, 1930, then died on July 2, 2016 is the one in 1960 presented the recursive solution to the problem of discrete linear data filtering called the Kalman Filter in his article [60]. Son of a Hungarian migrant family in the United States, during the Second World War. In 1943, when the war in the Mediterranean was intense, they traveled in an exodus movement to Turkey and Africa, which he eventually led to Youngstown, Ohio, in 1944 [47]. Kalman received the Bachelor's and Master's degree in Electrical Engineering at MIT respectively in 1953 and 1954. His supervisor was Ernst Adolph Guillemin, with his thesis topic was based on the resolution of second-degree differential equations [61]. A general introduction to the concept of the Kalman filter was presented in Chapter 1 of [62] by Maybeck. One can also refer to the following articles for more on the Kalman filter literature, presented by Gelb [63] Lewis [64], Jacobs [65] Brown and Hwang [66], and Grewal and Andrews [67].

The Kalman filter has been used extensively for tracking the graphical interaction of computers, where Welch, Bishop et al, used it in their HiBall Tracking System project, using the simple time constraint of the Kalman Filter [6]. It is also used for motion prediction by Azuma and Bishop [68] and again by Azuma [69]. The literature of the Kalman Filter offers a largesse in terms of literature because of its widely used. For who wants to know more about a Kalman Filter, can also refer to the following articles: Foxlin [70], Van Pabst and Krekel [71], Azarbayejani and Pentland [72], Emura and Tachi [73], and finally Mazuryk and Gervautz [74].

### **3.5. Summary**

This chapter, come to end the second part of the methods, materials used and the theoretical part of this thesis. The next chapter is more practical and presents the implementation, simulation, and analysis part of this research.



## **CHAPTER 4. EXPERIMENTS AND RESULTS**

In this chapter, the practical phase of this thesis, which allows the localization of a mobile robot (Robotrom) has been presented. The robot's platform is equipped with two wheels encoders for measuring wheel odometry and a gyroscope for measuring the angle of orientation. As the system is equipped with different sensors, the use of an algorithm with a single sensor is not appropriate, it cannot provide quality and satisfactory information. Consequently, to offset this deficiency, we set up a multi-sensor system with the Decentralized Kalman Filter to locate the robot positions.

The remainder of this chapter is arranged in two parts: A and B. Part A presents the Mobile Robotrom Platform drawn on SolidWorks and the method using to export the 3D model to Simscape. Simscape is an under application of Simulink allowing the modeling of compact and multi-body component in 3D dimension. Then, it implements the different algorithms with Matlab and Simulink. The implemented algorithm consists of two local Kalman filters and a central, global or master Kalman Filter to merge data from local filters. Finally, part B focuses on analyzing the obtained results after simulation.

### **4.1. Part A: Experiments and Implementation With Matlab**

#### **4.1.1. Presentation of Simscape**

Simscape is an integrated module of Simulink in the Matlab software. It allows the physical modeling of components in hydraulics, electricity, mechanics using block diagram. Based on the philosophy of diagram blocks programming like LabVIEW, and many other similar software, Simscape offers to a designer an advanced capability to evaluate at the same time the robustness and errors of his project. Also, Simscape offers the designer the ability to create their own block diagram using the Simscape

programming language. The possibility of importing a library into the Simscape environment is possible and very advantageous. Simscape has a 3D simulation environment for complex mechanical systems such as aircraft modeling, vehicle suspension, robot, mechanical equipment etc. The modeling of these complex systems is possible by the use of diagram blocks such as joints, constraints, the force of the elements, the sensors. All parameters used in Simscape are editable using functions and scripts written on Matlab and in addition the resulting block diagram models can interact with any other block diagrams in Simulink. In a nutshell, Simscape is an ideal simulation environment for systems designing and control implementation.

However, in practice the use of Simscape for complex system modeling as a robot requires a great deal of manual calculation which sometimes by their complexities can lead to errors and also take a lot of time in the implementation phase. To overcome this problem, Simscape offers a possibility to the designer to import 3D models already drawing on other CAD Platform like SolidWorks, Catia etc. This imported 3D files must have followed characteristic: the model must be completely assembled, contain all the measurements of masses, inertia, joints, constraints and geometrical measurements in 3D. Simscape automatically creates a 3D model corresponding to the imported model and offers the possibility to visualize the dynamics of the created model [75] The Matlab version used in this thesis is the 2016b version.

#### **4.1.2. Presentation of SolidWorks**

SolidWorks is CAD software for the physical modeling of any object. It is used for equipment modeling in the aeronautics, automotive, biomedical electronics etc. SolidWorks consists of several software's to draw, simulate, and even estimate the cost of production in some advanced version. Its use is intuitive and easier to understand. It does not necessarily require having an artistic talent and / or great knowledge of technical drawing. The specificity of SolidWorks is its speed to design and test 3D models. In this thesis, it was used to draw Robotrom, the mobile robot analyzed in this research, in order to avoid complex calculations that Simscape

reserved for us. The drawn and assembled robot on SolidWorks is shown in Figure 4.1. below.

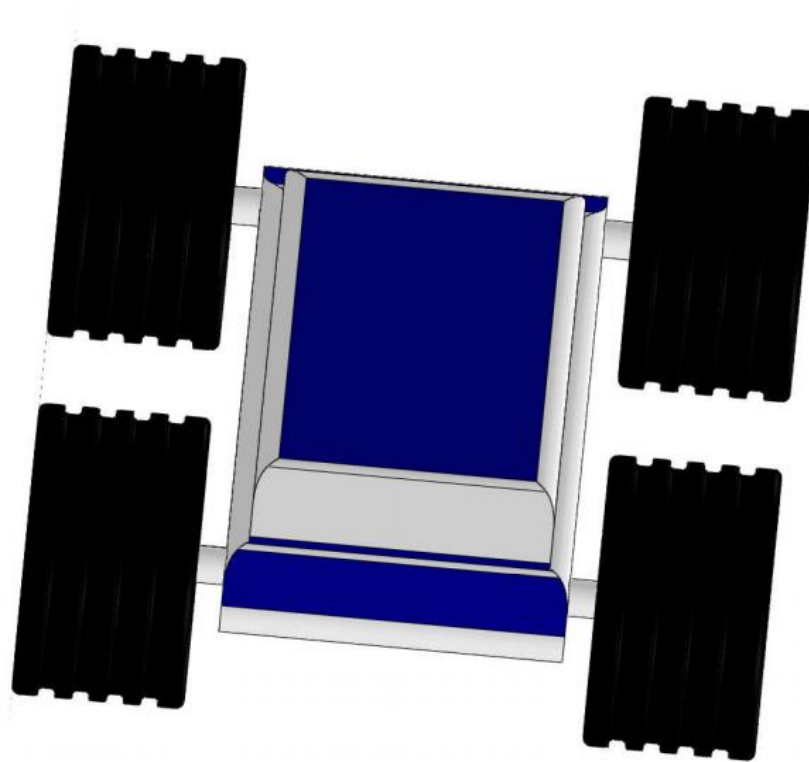


Figure 4.1. Mobile robot drawn with SolidWorks 2017 software

Now the 3D model is ready to be exported to Simulink's 3D Simulation environment, using the functions offers by Simscape in Simulink. For this fact, it is necessary to export the assembled file of the robot from SolidWorks with the extension xml by means of a plugin installed on Matlab. To open this folder in Matlab, we use the 'smimport' function of Matlab by passing in parameter the name of the xml exported file. After that, Matlab automatically opens Simscape and creates the corresponding exported SolidWorks model. Before any manipulation, one must be reassured of the veracity of the file transferred in Simscape. For this fact, it is necessary to check the joins, and the constraints imposed on the robot during its design in SolidWorks by applying a gravity test. The 3D model diagram block obtained from SolidWorks to Simscape is shown in Figure 4.2.

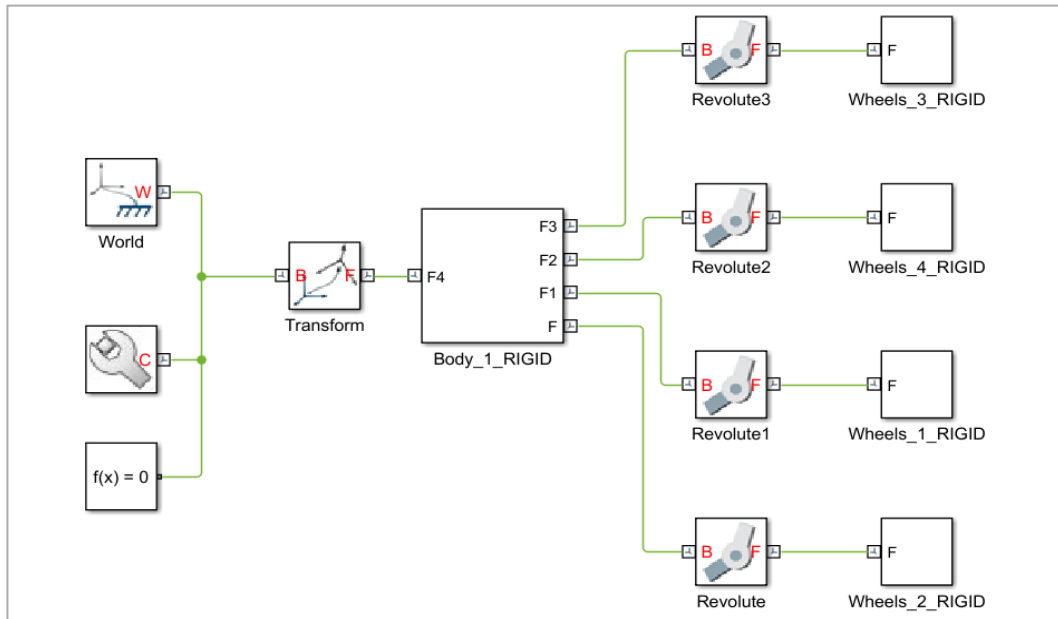


Figure 4.2. Corresponding Robotrom block diagram on Simulink

The table 4.1. below shows the characteristics of the robot Robotrom.

Table 4.1. Robotrom parameters

Weight of the robot (kg)	1.07
Wheel diameter (m)	0.70
Maximum speed of the robot (m / s)	0.5
Maximum acceleration	0.25
The greatest angle slope to frank (°)	25
Distance between the longitudinal wheels L: back & front (m)	0.12
Distance between side wheels D: left & right (m)	0.112

#### 4.1.3. Implementation with Matlab

This section is based on the implementation of algorithm allowing control and evaluation of the robot operating. Figure 4.3. below shows the full block diagram of the robot. The Simulink model of the robot consists of two parts. The 3D physical model of the robot, and the Kalman filter. The pose of the robot is expressed by the following vector  $[x, y, \theta, \varphi]$ , the encoders measure the relative angular velocity of the wheels which allows to deduce the position of the robot using the principle of kinematics. The rotation matrix makes it possible to pass local coordinates of the robot

to the world referential. As for the orientation angle, it is measured with a gyroscope which axis of rotation is the Z axis. By combining the data of the different sensors, it was estimated the pose of the robot. Since these measurements may be subject to noise disturbance, the Kalman filter is then a powerful instrument allowing the elimination of these noises in order to obtain the result close to the actual measurement. The pose of the robot is as follows:

$$\frac{d}{dt} \begin{bmatrix} X_x(t) \\ X_y(t) \\ \theta(t) \\ \varphi(t) \end{bmatrix} = \begin{bmatrix} V(t) \cos(\theta(t)) \\ V(t) \sin(\theta(t)) \\ \frac{V(t)}{L} \tan(\varphi(t)) \\ \omega(t) \end{bmatrix}$$

Where  $V_R$  and  $V_L$  are the speeds on each side of the robot,  $V(t)$  the linear speed of the robot which is the average speed of  $V_R$  and  $V_L$  and  $D$  the distance between the rear wheels of the robot.  $X_x(t)$  and  $X_y(t)$  are the positions of the robot on the x and y axes.  $\theta(t)$  is the orientation of the robot, and  $\varphi(t)$  the steering angle of the robot. In practice  $\varphi(t)$  is not estimated, so the complete pose of mobile robot is:

$$X = [P_x, P_y, \theta, \dot{P}_x, \dot{P}_y, \dot{\theta}]^T.$$

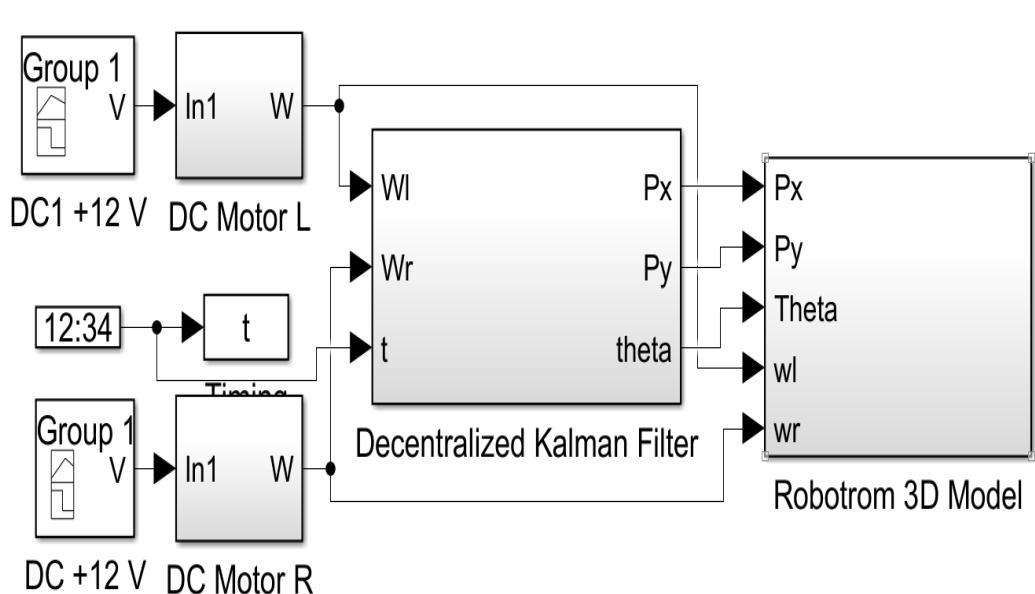


Figure 4.3. Robotrom Simulink model

#### 4.1.3.1. DC motor modelling with Simulink

Any mobile robot to operate needs an actuator like a DC motor. These being the choice of engine that is needed is important. The power required for the normal operation of the robot must first of all be studied in order to choose the most suitable motor for the robot. A DC motor is characterized by electrical energy at the stator and mechanical energy at the rotor. This electrical energy can be defined by a source of electrical voltage passing through a resistor and inductance included in the motor at the stator. At the rotor, there is a rotational movement creating a mechanical energy. The wiring diagram in the Figure 4.4. below is an illustration of a DC motor modeled on Simulink.

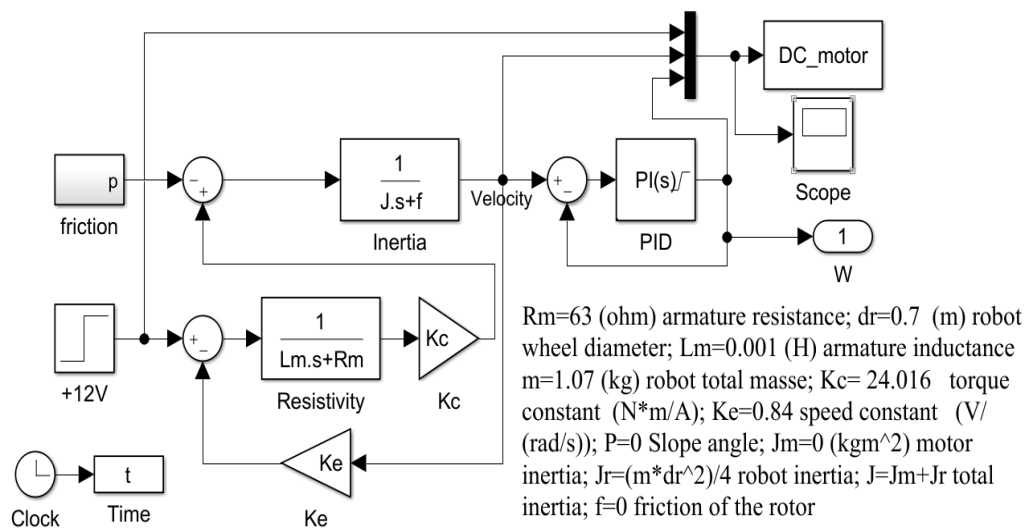


Figure 4.4. Electrical equivalent circuit of DC motor with Simulink

The +12V block is the voltage source provided in this case 12V, the Resistivity block is the transfer function of the electrical part of the motor,  $K_c$  and  $K_e$  are the torque and speed constant of the motor respectively, the friction block is characterized by a slope, wind that the robot can meet and finally the Inertia block is the sum of the viscous friction  $f$  and the total inertia  $J$  which is the addition of the motor and robot inertias. The motor inertia's and characteristics, are provided in the data sheet of the motor designer by against the robot inertia is calculated as follows:

$$J_{robot} = \frac{md^2}{4} \quad (4.1)$$

Where  $m$  is the total mass of the robot,  $d$  the diameter of the wheels and  $J_{robot}$  the inertia of the robot. The simulation results of a motor corresponding to Robotrom characteristics with and without disturbances can be observed in Figure 4.5. and Figure 4.6. below.

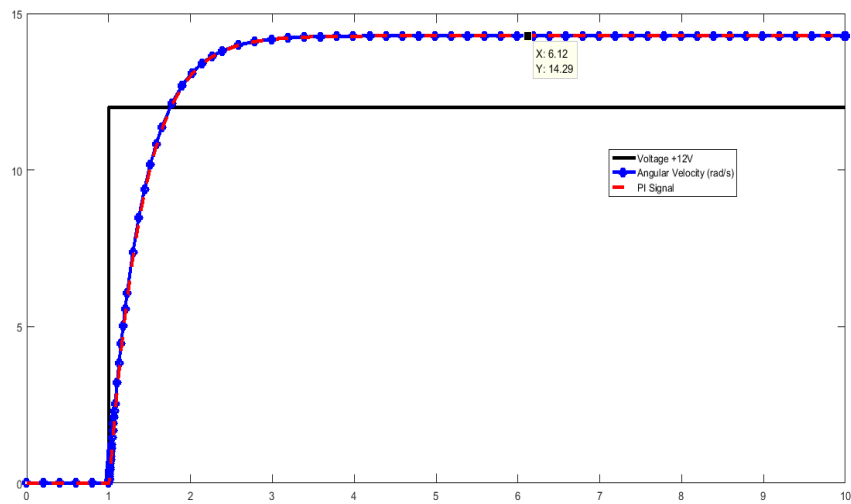


Figure 4.5. Simulated robot without slope

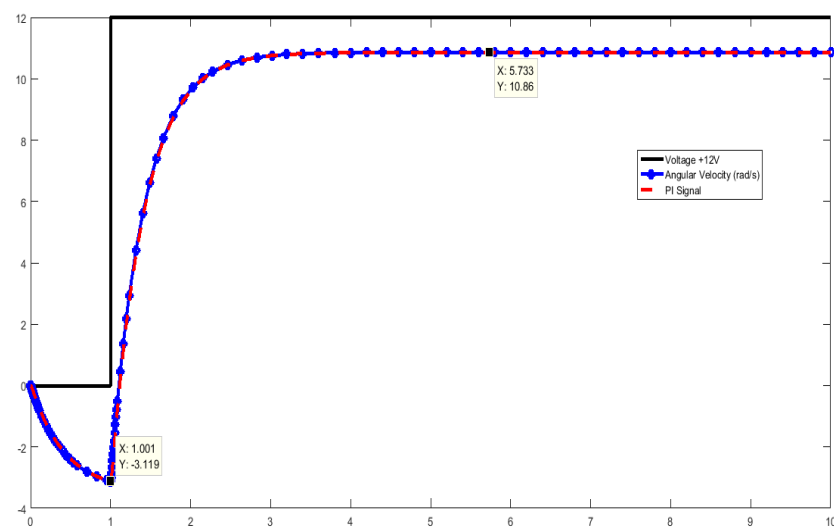


Figure 4.6. Simulated motor with a slope angle equal to 6 degree

It can be seen from the Figure 4.6. that the robot speed is negative ( $-3.119\text{rad/s}$ ) before applying the voltage ( $12\text{V}$ ) to the motor terminal. After application of the voltage, the speed becomes positive again but this time cannot reach the  $14.29\text{rad/s}$  observed in the previous Figure 4.5. The disturbance imposes a negative force on the movement of the robot thus, if this force is greater than the maximum power of the robot, it will be in free fall. This can be compared by a vehicle riding an incline, if the angle of the slope is high and create a great friction force, this force will roll back the vehicle.

Knowing all the characteristics of the motor to use summarized in Table 4.2. below, one can pass to the control of it by the use of a PID controller [5].

Table 4.2. Simulated motor characteristic for Robotrom

Motor speed (rad/s)	14.29
Motor torque (Nm)	0.1599
Mechanical power of the motor (w)	2.2843
Supply voltage (V)	12
Current (A)	0.1904
Armature Resistance (ohm)	63
Armature Inductance (H)	0.001
Torque constant	24.016
Speed constant (V/(rad/s))	0.84

#### 4.1.3.2. First local Kalman Filter with two wheels encoders

The Kalman filter used is based on the measurement model of two rotary encoders providing the rotational speed of each wheel, then by integration as a function of time of these speeds, one obtains the positions on the X and Y axes. The device consisting of the two encoders will be the first local filter of our overall system, in which the position data will be merged with the orientation angle measurements provided by the gyroscope which makes it the second local filter. The discrete model of Kalman for the local filter 1 is described as follows:



$$\hat{X}[k] = \begin{bmatrix} \hat{X}_x[k] \\ \hat{X}_y[k] \\ \hat{X}_x[k] \\ \hat{X}_y[k] \end{bmatrix}$$

$\hat{X}_x[k]$  The estimated position of the robot on the axe X,  $\hat{X}_y[k]$  the estimated position of the robot on the y axis  $\hat{X}_x[k]$  is the estimated longitudinal speed of the robot (x axis)  $\hat{X}_y[k]$  is the estimated lateral speed of the robot (y-axis) We can recall the Kalman model on the following form:

$$\hat{X}[k + 1] = A\hat{X}[k] + Gw[k]$$

$$Z[k] = H\hat{X}[k] + V[k]$$

The matrices A, G and H are of the following form:

$$A1 = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$G1 = \begin{bmatrix} \Delta t/2 & 0 \\ 0 & \Delta t/2 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Where  $\Delta t$  is the sampling time equal to 20 milliseconds (20ms). The covariance error matrix Q is proportional to the angular velocity  $\omega_R$  and  $\omega_L$  respectively of the left and right wheels. Their variances is then equal to  $\rho\omega_R^2$  and  $\rho\omega_L^2$ . Where  $\rho$  is a constant obtained by experiments in our case equal to  $10^{-4}$ . The matrix Q is defined as follows:

$$Q_1 = \begin{bmatrix} \rho\omega_R^2 & 0 \\ 0 & \rho\omega_L^2 \end{bmatrix}$$

#### 4.1.3.3. Second local Kalman Filter with a gyroscope

The second Kalman local filter consists of a gyroscope oriented on the Z axis, to obtain the relative orientation angle of the robot. The operating principle of the gyroscope is to integrate the robot's angular velocity over the time to obtain the orientation angle [76]. The typical model of a gyroscope is illustrated in the following equation:

$$\begin{aligned} \omega(t) &= \omega_a(t) + \omega_e(t) \\ \varphi &= \int_0^t \omega(t) + C_0 \end{aligned} \tag{4.2}$$

With  $C_0$ , the initial orientation of the robot,  $\omega_a(t)$  is the angular velocity measured by the gyro with an error  $\omega_e(t)$ . Thus the definitive measurement  $\omega(t)$  contains both signals the true measurement and the measurement error. Thus the model of Kalman can be as follows:

$$X[k] = \begin{bmatrix} \omega[k] \\ \dot{\omega}[k] \end{bmatrix}$$

$$A2 = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

$$G2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H2 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

To regularize the size of the overall covariance matrix of the master filter, the size of the distribution matrix G2 and the measurement matrix H2 were considered as a two-measure matrix.

#### 4.1.3.4. The global Kalman Filter

As presented in chapter 3 the matrices of the global model are the constitution of the matrices of the different local filters.

$$X = [X1, X2]^T = [0 \ 0 \ 0 \ 0 \ 0 \ 0]^T$$

$$A = blkdiag(A1, A2) = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$G = blkdiag(G1, G2) = \begin{bmatrix} \Delta t/2 & 0 & 0 & 0 \\ 0 & \Delta t/2 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$C = [H_1 M_1, H_2 M_2]^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The last line of the matrix C is always equal to zero. This line is used for the correction of the matrix size error. It has been assumed that the measurement in the second local filter 2 consists of a gyroscope and an imaginary measurement which itself is set to zero in order to obtain local filters of two sensors on either side. As a result, the implementation of the global covariance matrix is of the following form:

$$Q = \text{blkdiag}(Q1, Q2) = \begin{bmatrix} q1 & 0 & 0 & 0 \\ 0 & q1 & 0 & 0 \\ 0 & 0 & q2 & 0 \\ 0 & 0 & 0 & q2 \end{bmatrix}$$

Where  $q1$  and  $q2$  the standard deviation of the filters 1 and 2 sensors. Consequently the measurement error of each sensor can be represented in the matrix  $R$  as follows:

$$R = \text{blkdiag}(R1, R2) = \begin{bmatrix} r1 & 0 & 0 & 0 \\ 0 & r1 & 0 & 0 \\ 0 & 0 & r2 & 0 \\ 0 & 0 & 0 & r2 \end{bmatrix}$$

The model of Kalman decentralized filter made on Simulink is shown in Figure 4.7. below. After simulation on Matlab and Simulink, the various obtained results are analyzed in Part B below.

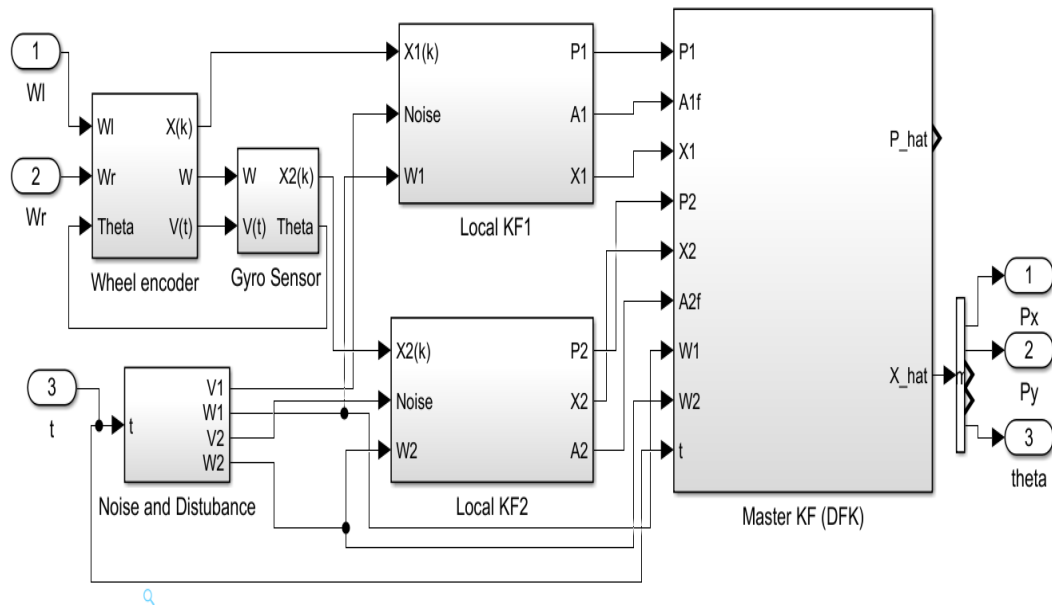


Figure 4.7. Decentralized Kalman Filter model on Simulink

## 4.2. Part B: Results and Analysis

This section presents the results obtained from the experience explained in part A of this chapter. An experiment that highlights the use of the Decentralized Kalman Filter

for the data fusion of different sensors measurement to estimate the mobile robot pose's by wheel odometry.

#### 4.2.1. Analysis of the first local Kalman Filter

Figure 4.8., Figure 4.9., Figure 4.10. below shows the results of the Kalman Local Filter 1 used for estimating the position of the robot. In Figure 4.8. and Figure 4.12. we can see the plot of the position of the robot and its estimate on the x and y axes. Figure 4.9. shows the estimated linear velocity of each axis x and y. Finally, the Figure 4.10. presents the measurement and estimation errors of the local filter. One can see immediately the weakness of the local Kalman filter 1. This weakness can be explained by the fact that the Kalman Filter is more assigned to the linear system which is not the case here. But remarkable thing is the estimate of the global filter after merging the data of all measurement shown in Figure 4.12.

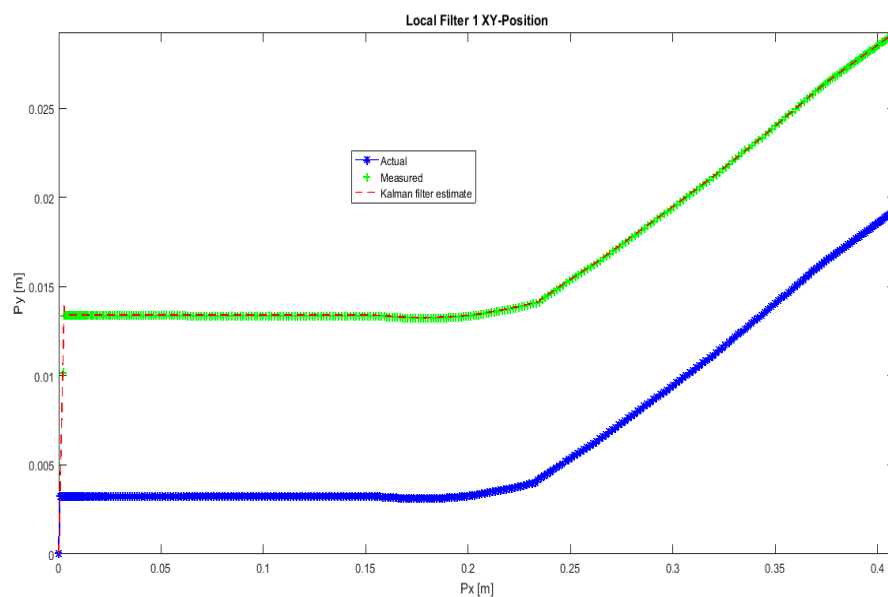


Figure 4.8. Local Kalman Filter 1: x,y positions

Actual, measured and estimated robot's position with the First local Kalman Filter on x and y axis. The blue graphic is the robot's real position. The green graphic is the

measured position, and the red dotted one is estimated position by using the Decentralized Kalman Filter's first local filter.

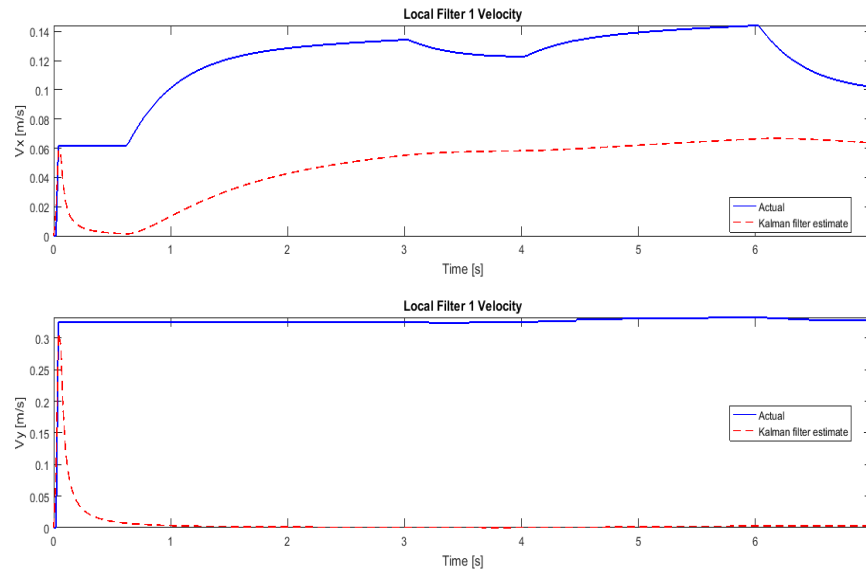


Figure 4.9. Linear speed on x and y axes, local KF 1

First Local Kalman filter estimated velocity on X and Y axis. The blue graphics are the robot's real velocities on x and y axis. The red dotted one is estimated velocity by using the Decentralized Kalman Filter's first local filter.

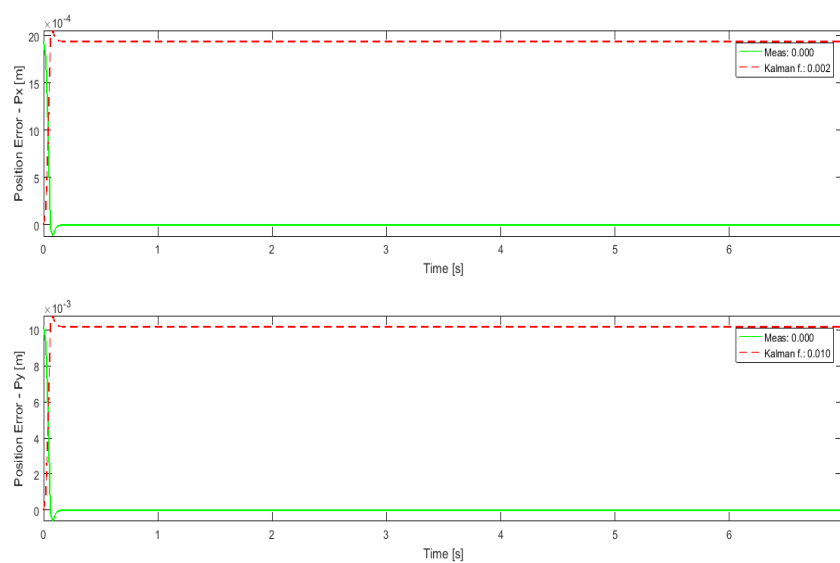


Figure 4.10. Measurement and estimation errors of local KF 1

Measured and estimated position errors on x-y axis by the first Local Kalman Filter. The green graphics is the measurement error and the red dotted one is the estimated errors.

#### 4.2.2. Analysis of the second local Kalman Filter

The second local filter is focused on estimating the angle of orientation of the robot. The results as well as the local filter 1 are also of undesirable quality. Unlike those obtained with the global filter in Figure 4.14. The results of second local filter can be seen in Figure 4.11.

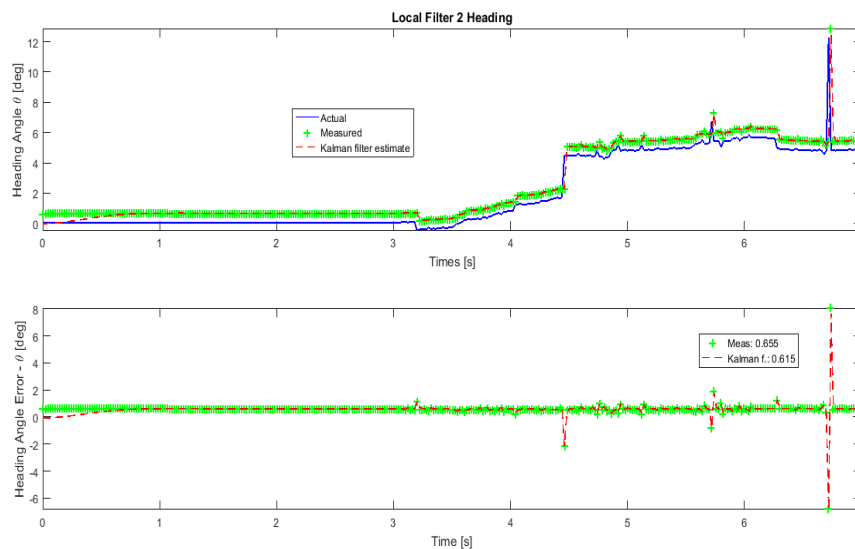


Figure 4.11. Heading angle, measurement and estimation errors local KF 2

The first subfigure up is an actual, measured and estimated heading angle of the robot using the Decentralized Kalman Filter's second local filter. The blue graphic is the robot's true orientation. The green graphic is the measured orientation. The red dotted one is estimated orientation. The second subfigure down describes the measured and estimated errors highlighted in green and red respectively.

### 4.2.3. Analysis of the global Kalman Filter

The Figure 4.12., Figure 4.13., Figure 4.14., and Figure 4.15. are the results obtained after merging the data obtained from the different local filters. And we can see that unlike the result previously obtained with each local filter, these results are much more accurate. Figure 4.12. shows the position estimated and the speed is shown in Figure 4.13., the orientation angle can be observed in Figure 4.14. and finally the measurement and estimation errors are presented in Figure 4.15.

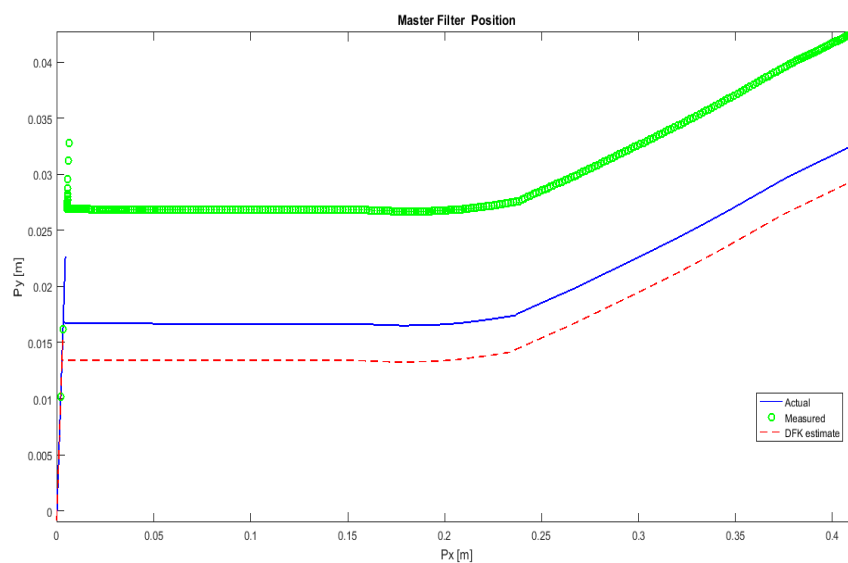


Figure 4.12. Position on the x and y axes of the global Kalman Filter

Actual, measured and estimated position with the Decentralized Kalman Filter's global filter. The blue signal is the robot's real position on x and y axis. The green graphic is the measured position. The red dotted one is estimated position by the use of the Decentralized Kalman Filter.



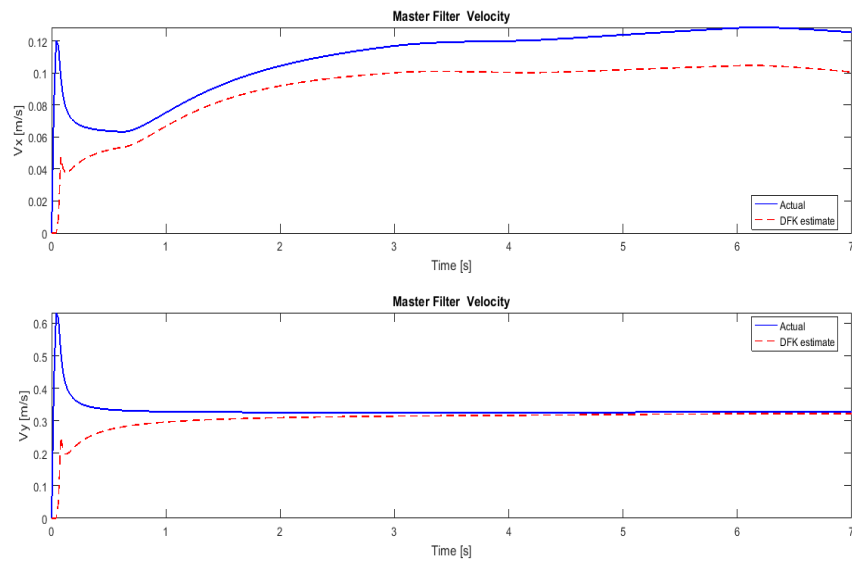


Figure 4.13. Speed on the x and y axes of the global Kalman Filter

Estimated velocity with the Decentralized Kalman Filter's global filter. The blue graphics in the subfigures up and down are the robot's real velocities on x and y axis respectively. The red dotted one is estimated velocity by the use of the Decentralized Kalman Filter's global filter.

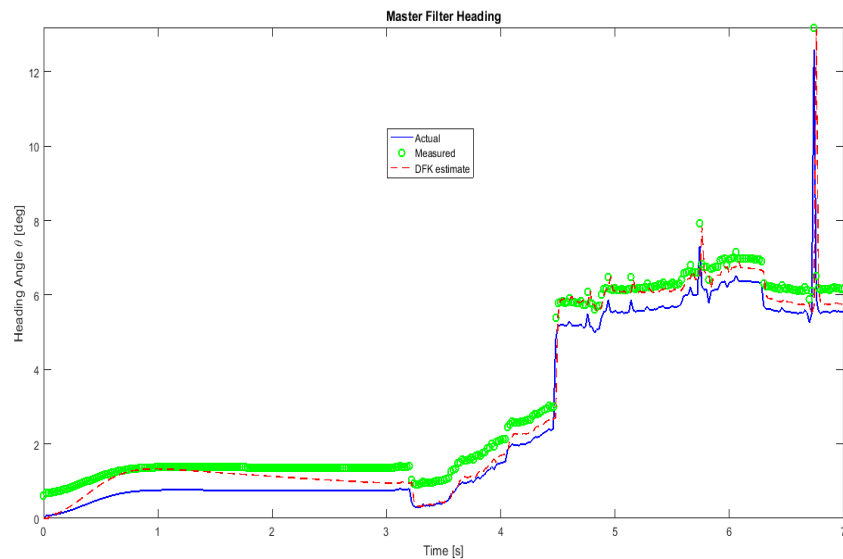


Figure 4.14. Heading angle of the global Kalman Filter

Actual, measured and estimated heading angle with the Decentralized Kalman Filter's global filter. The blue graphic is the robot's orientation. The green graphic is the measured orientation. Finally, the red dotted one is estimated orientation by using the Decentralized Kalman Filter.

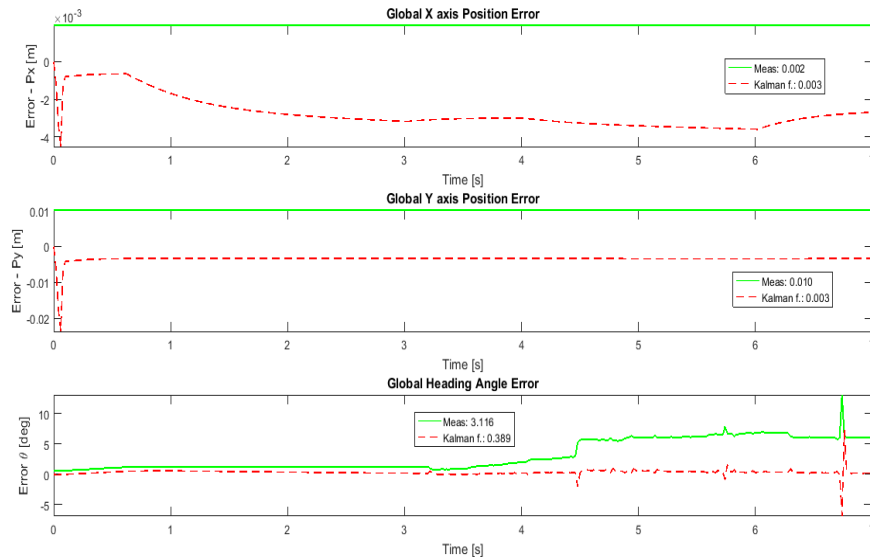


Figure 4.15. Measurement and estimation errors of global Kalman Filter

Global filter measured and estimated errors. The green graphics are the measured errors and the red dotted one the estimated errors of Decentralized Kalman Filter's global filter.

In view of all these results, it can clearly be seen that the global filter provides much more accurate results than those provided by each local filters of Decentralized Kalman Filter.

## **CHAPTER 5. CONCLUSION AND SUGGESTION**

### **5.1. Conclusion**

This thesis presented the localization of a mobile robot using odometry and Decentralized Kalman Filter. It was demonstrated during this research, the important role that the Decentralized Kalman Filter played in the data fusion stage. Comparative analyzes between the local filters and the global filter showed that after the data is merged, a more accurate estimate is obtained than the standard local mode filter. After simulation with Matlab and Simulink 3D environment, the following observations were made: The Decentralized Kalman Filter can be used to merge hybrid sensor model in multivariable system. The central filter of DFK provides a more accurate estimate than the local filters.

### **5.2. Suggestion**

The points listed below should be taken into consideration for future works:

- Optimize and improve the filtering algorithm by using more advanced simulation tools like Gazebo.
- Take advantage of interconnection options Gazebo and Simulink through the use of ROS (Robot operating System) to test in a more optimal virtual world taking into account several parameters of the real world.
- Model the detection and avoidance object algorithm. To provide an autonomy of the robot to evolve in an environment containing multiple obstacles.
- Model an algorithm for multiple tracking of objects.
- Model the visual odometry mono, stereo or omnidirectional camera, so that the robot can have a broader and more precise overview of its working environment.

- Estimate the pose of the robot through data provided by a laser sensor.
- Associate with the system a GPS sensor or an IMU sensor for the measurement of the position and the angle of orientation.
- Model the measurement fault detection algorithm of a sensor in order to take into consideration measurements provided by the latter at the moment when the malfunction was found.
- Test the robot with other non-Kalman algorithm such as particle filter to highlight a comparative study of these filters.
- Realization of the hardware part of the Robotrom robot in order to analyze it in real time.
- Uses the synchronized speed control technique for four wheeled mobile robot. A much more robust technique allowing the robot to function in very rugged environments.

## REFERENCES

- [1] Thrun, S., Burgard, W., Fox, D., Probabilistic Robotics, MIT Press, 2006.
- [2] Siciliano, B., Khatib, O., Springer Handbook of Robotics, Springer-Verlag Berlin Heidelberg, 2008.
- [3] Nister, D., Naroditsky, O., Bergen, J., Visual odometry, in in Proc.Int. Conf. Computer Vision and Pattern Recognition, pp. 652–659, 2004.
- [4] Hartley, R., Zisserman, A., Multiple view geometry in computer vision, Second ed., Cambridge University Press, 2003.
- [5] Åström, K., Hägglund, T., Advanced PID Control, ISA, Research Triangle Park, NC, August 2005.
- [6] Greg, W., Gary, B., An Introduction to the Kalman Filter, University of North Carolina, Department of Computer Science, TR 95-041, 1995.
- [7] Siouris, G., M. An engineering approach to optimal control and estimation theory, John Wiley & Sons, Inc. 1996.
- [8] Rekleitis Ioannis, M., A particle filter tutorial for mobile robot localization, PhD thesis, School of Computer Science, McGill University, Montréal, Quebec, Canada, February 2003.
- [9] Rekleitis Ioannis, M., Cooperative Localization and multi-robot exploration, PhD thesis, School of Computer Science, McGill University, Montréal, Quebec, Canada, February 2003.
- [10] <https://www.microsonic.de/en/support/ultrasonic-technology/operating-modes.htm>, Access Date: 06/02/2018.
- [11] <http://velodynelidar.com/downloads.html>, Access Date: 09/02/2018.
- [12] Fischler, M., A., Bolles, R., C., Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Comm. Assoc. Comp. Mach, vol. 24, no. 6, pp. 381-395, 1981.

- [13] Friedrich, F., Scaramuzza, D., Visual Odometry Part I: The First 30 Years and Fundamentals, IEEE ROBOTICS & AUTOMATION MAGAZINE, DECEMBER 2011.
- [14] Dudek, G, Jenkin, M., Computational Principles of Mobile Robotics, 2nd Edition Cambridge University Press, New York, USA 2010.
- [15] Wang, H., Zhang, J., Song, D., Jayasuriya, S., Liu Yi, J, Kinematic Modeling and Analysis of Skid-Steered Mobile Robots With Applications to Low-Cost Inertial-Measurement-Unit-Based Motion Estimation, in IEEE Transactions on Robotics, vol. 25,no. 5, pp.1087-1097, 20009.
- [16] Wong J.,Y., Chiang, C., F., A general theory for skid steering of tracked vehicles on firm ground, Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering,, vol. 215, no. 3, pp. 343-355, 2001.
- [17] Shamah, B., Experimental Comparison of Skid Steering Vs. Explicit Steering for a Wheeled Mobile Robot, Master Thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh Pennsylvania, 1999.
- [18] Xiaodong, W., Xu M., Wang, L., Differential Speed Steering Control for Four-Wheel Independent Driving Electric vehicle, International Journal of Materials, Mechanics and Manufacturing, vol. 1, no. 4, November 2013.
- [19] Moravec, H., Obstacle avoidance and navigation in the real world by a seeing robot rover, Ph.D. dissertation, Stanford Univ., Stanford, CA, 1980.
- [20] Hannah, M., Computer matching of areas in stereo images, Ph.D. dissertation, Stanford Univ., Stanford, CA, 1974.
- [21] Moravec, H, Towards automatic visual obstacle avoidance, in in Proc.5th Int. Joint Conf. Artificial Intelligence, Aug. 1977, pp. 584.
- [22] Forstner, W., A feature based correspondence algorithm for image matching, in Int. Arch. Photogrammetry, vol. 26, no. 3, pp. 150–166, 1986.
- [23] Harris, C., Pike, J., 3d positional integration from image sequences, in Proceedings, Alvey Vision Conf., 1988, pp. 87–90.
- [24] Stephens, M., Harris, C. A combined corner and edge detector, in in Proceedings, Alvey Vision Conf., 1988, pp. 147–151.

- [25] Olson, C., Matthies, L., Schoppers, M., Maimone, M., W., Robust stereo ego-motion for long distance navigation, in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2000, pp. 453–458.
- [26] Sutherland, I., E., Sketchpad: Aman-machine graphical, Technical Report 296, MIT Lincoln Laboratories, 1963. Also published by Garland Publishing, New York, 1980.
- [27] Hartley, R., I., In defense of the eight-point algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 6, pp. 580-593, 1997.
- [28] Slama, C., Manual of Photogrammetry, 4th ed., Falls Church, VA, USA: American Society of Photogrammetry, 1980.
- [29] Torr, P., H., S., Murray, D., W., Outlier detection and motion segmentation, In Proc SPIE Sensor Fusion VI, pp. 432–443, September 1993.
- [30] Zhang, Z, Deriche, R., Faugeras, O., D., Luong, Q., A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry, Artificial Intelligence, vol. 78, pp. 87–119, 1995.
- [31] Torr, P., H., S., Zisserman, A., Robust computation and parameterization of multiple view relations, InProc. 6thInternational Conference on Computer Vision, pp. 727–732, January 1998.
- [32] Criminisi, A., Reid, I, Zisserman, A., A plane-measuring device,” Image and Vision Computing, vol. 17, no. 8, pp. 625–634, 1999.
- [33] Zeller, C., Projective, Affine and Euclidean Calibration in Computer Vision and the Application of Three Dimensional Perception, PhD thesis, Robot,Vis Group, INRIA Sophia-Antipolis, 1996.
- [34] Sturm, P., Vision 3D non calibrée : Contributions à la reconstruction projective et étude des mouvements critiques pour l’auto-calibrage, PhD thesis, INRIA Rhone-Alpes, 1997.
- [35] Vieville, T., Lingrand, D., Using singular displacements for uncalibrated monocular vision systems, Technical Report 2678, I.N.R.I.A., 1995.
- [36] Sturm, P., Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction, In Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1100–1105, June 1997.
- [37] Springer, C., E., Geometry and Analysis of Projective Spaces, San-Francisco: Freeman, 1964.

- [38] Devernay, F., Faugeras, O., D., From projective to euclidean reconstruction, In Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 264–269, 1996.
- [39] Wolfe, W., J., Mathis, D., Sklair, C., Magee, M., The perspective view of three points, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 1, pp. 66–73, January 1991.
- [40] Tsai, R., Y., Huang, T., S., The perspective view of three points, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 6, pp. 13–27, 1984.
- [41] Christy, S., Horaud, R., Euclidean shape and motion from multiple perspective views by affine iteration, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 11, pp. 1098–1104, November 1996.
- [42] Tordoff, B., Murray, D., W., Reactive zoom control while tracking using an affine camera, In Proc. British Machine Vision Conference, vol. 1, pp. 53–62, 2001.
- [43] Abdien Ali Abdulla, F., Decentralized Kalman Filter Approach for Multi-sensor Multi-target Tracking Problems, Master thesis, Department Electricals and Electronics Engineering, Sakarya University, June 2016.
- [44] Bramwell, B., “centralized and decentralized Kalman filter technique for tracking navigation and control,” in Vol. 32, no. 4, pp.859-886, 2005.
- [45] Murphy, K., P., Proposed design for gR, a graphical models toolkit for R. Interface, September, 2003.
- [46] Pinchinat, S., Riedweg, S., On the architecture in decentralized supervisory control, in Proc. 44th IEEE Conf. Decis. Control. Eur. Control Conf. CDC-ECC’ 05, vol. 2005, pp. 12-17, 2005.
- [47] Grewal, M., Andrews, A., P., Kalman Filtering Theory and Practice Using MATLAB, third Edition, Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2008.
- [48] Brumback, B., Srinath, M., A Fault-Tolerant Multi-sensor Navigation System Design, in IEEE Transactions on Aerospace and electronic systems vol. aes-23, no. 6, November 1987.



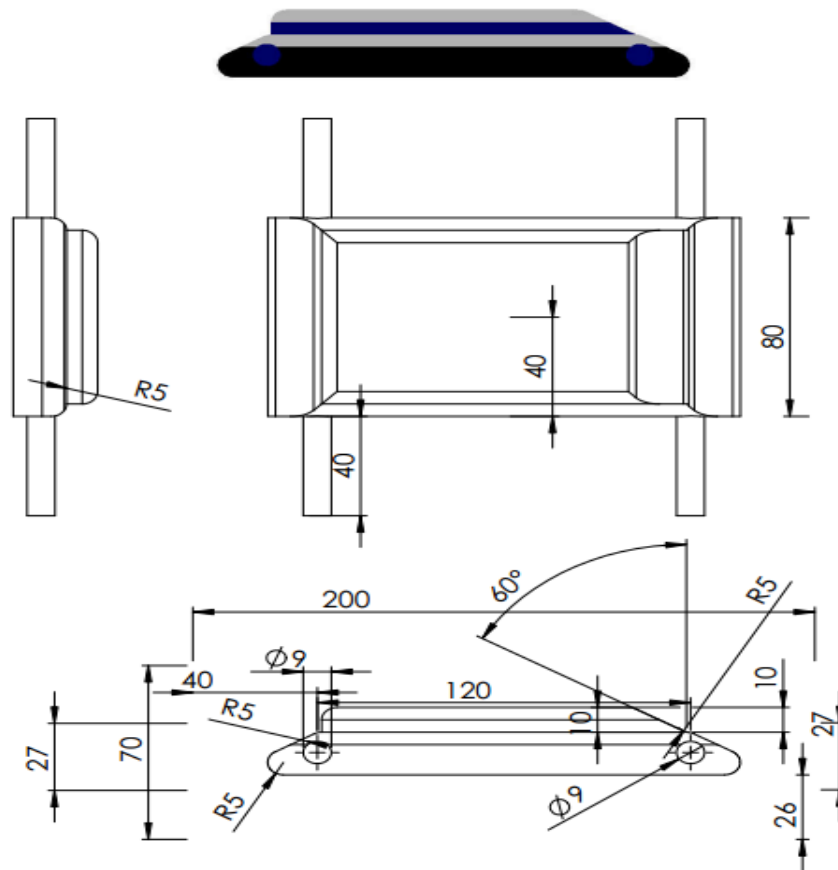
- [49] Chung, Y., Emre, E., Gustafson, D., A global modeling approach for multi-sensor problems, in 306 SPIE Vol. 1481 Signal and Data Processing of Small Targets 1991.
- [50] Sage, A., Melsa, J., Estimation Theory with Applications to Communication and Control, New York: McGraw-Hill, 1971.
- [51] Kailath, T., A view of three decades of linear filtering theory, in IEEE Transactions on Information Theory, Vol. IT-20, No.2, pp. 146–181, 1974.
- [52] Kailath, T., Linear Least Squares Estimation, in Dowden, Hutchinson and Ross, Stroudsburg, PA, 1977.
- [53] Lainiotis, D., G., Estimation: A brief survey,” in Information Sciences, Vol. 7, pp. 191–202, 1974.
- [54] Geiseking, D., Mendel, J., M., Bibliography on the linear-quadratic-Gaussian problem, in IEEE Transactions on Automatic Control, Vol. AC-16, pp. 847–869, 1971.
- [55] Sorenson, H., W., Least-squares estimation: From Gauss to Kalman, in IEEE Spectrum, Vol. 7, pp. 63–68, 1970.
- [56] Sorenson, H., W., Kalman Filtering: Theory and Application, IEEE Press, New York, 1985.
- [57] Battin, R., H., Space guidance evolution—a personal narrative, AIAA Journal of Guidance and Control, vol. 5, pp. 97-110, 1982.
- [58] Schmidt, S., F., The Kalman filter: Its recognition and development for aerospace applications, AIAA Journal of Guidance and Control, vol. 4, no. 1, pp. 4-8, 1981.
- [59] Baker, R., M., L., Makemson, M., W., An Introduction to Astrodynamics, Academic Press, New York, 1960.
- [60] Kalman, R., E., A New Approach to Linear Filtering and Prediction Problems, Transaction of the ASME—Journal of Basic Engineering, 82 (Series D), pp 35-45, 1960.
- [61] Kalman, R., E., Phase-plane analysis of nonlinear sampled-data servomechanisms, M.Sc. thesis, Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 1954.

- [62] Maybeck, P., S., Stochastic Models, Estimation, and Control, Volume 1, Academic Press, Inc, 1979.
- [63] Gelb, A., Applied Optimal Estimation, MIT Press, Cambridge, MA, 1974.
- [64] Lewis Richard, Optimal Estimation with an Introduction to Stochastic Control Theory, ohn Wiley & Sons, Inc, 1986.
- [65] Jacobs, O., L., R., Introduction to Control Theory, 2nd Edition, Oxford University Press 1993.
- [66] Brown, R., G., Hwang, P., Y., C., Introduction to Random Signals and Applied Kalman Filtering, 2nd Edition, John Wiley & Sons, Inc, 1992.
- [67] Grewal, M., S., Andrews, A., Kalman Filtering Theory and Practice Using MATLAB, Second Edition. New York, NY USA: John Wiley & Sons, Inc, 2001.
- [68] Azuma, R., Gary, B., Improving Static and Dynamic Registration in an Optical See-Through HMD,” in SIGGRAPH 94 Conference Proceedings, Annual Conference Series, pp. 197-204, ACM SIGGRAPH, Addison Wesley, July 1994. ISBN 0-201-60795-6.
- [69] Azuma, R., Predictive Tracking for Augmented Reality, Ph.D. dissertation, University of North Carolina at Chapel Hill, TR95-007.
- [70] Foxlin, E, Inertial Head Tracking, Master’s Thesis Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1993.
- [71] Van Pabst, J., V., L., Krekel, P., F., C., Multi Sensor Data Fusion of Points, Line Segments and Surface Segments in 3D Space, in TNO Physics and Electronics Laboratory, The Hague, the Netherlands. [Cited 19 November 1995]. Available from <http://www.bart.nl/~lawick/index.html>, 1995.
- [72] Pentland, A., Azarbayejani, Ali., Recursive Estimation of Motion, Structure, and Focal Length, in IEEE Trans. Pattern Analysis and Machine Intelligence, 17(6), June 1995.
- [73] Emura, S., Tachi, S., Sensor Fusion based Measurement of Human Head Motion,” in Proceedings 3rd IEEE International Workshop on Robot and Human Communication, RO-MAN’94 NAGOYA (Nagoya University, Nagoya, Japan), 1994.

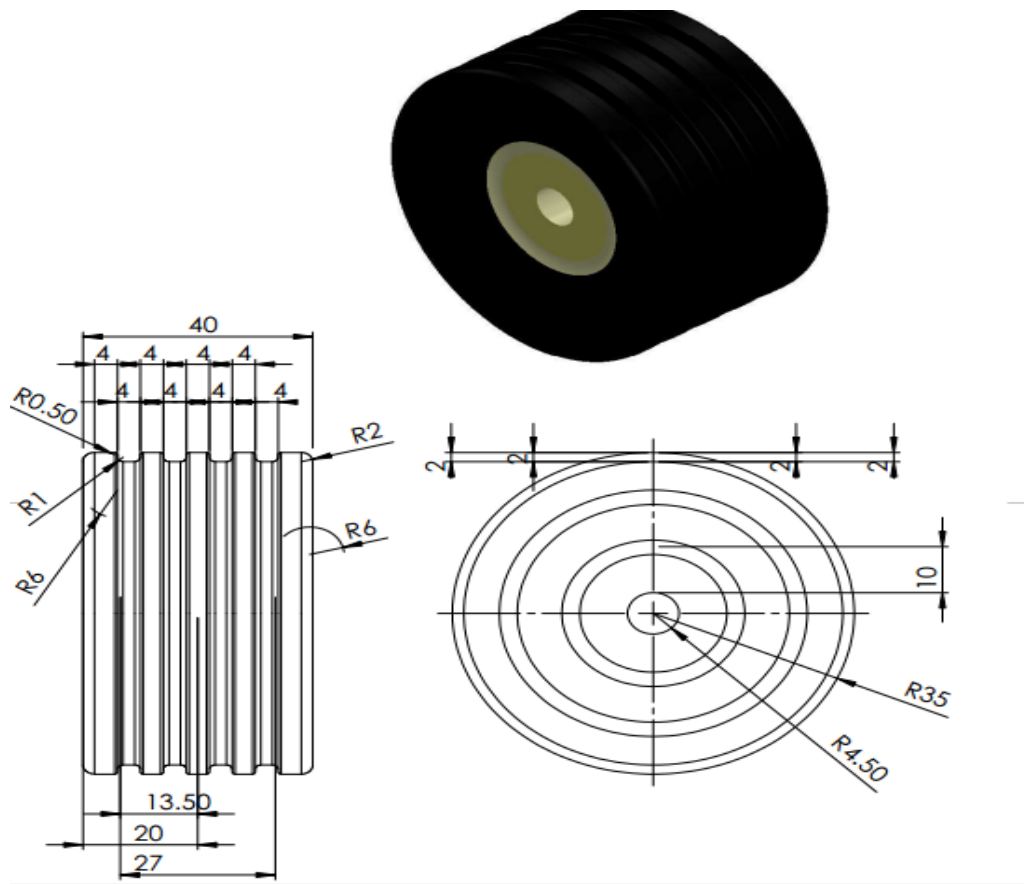
- [74] Mazuryk, T., Gervautz, M., Two-Step Prediction and Image Deflection for Exact Head Tracking in Virtual Environments, in EUROGRAPHICS '95, Vol. 14, No. 3, pp. 30-41, 1995.
  
- [75] <https://www.mathworks.com/help/physmod/sm/index.html>, Access Date: 15/01/2018.
  
- [76] Sven Rönnbäck, On a Matlab/Java Testbed for Mobile Robots, EISLAB. Dept. of Computer Science and Electrical Engineering Luleå University of Technology Luleå, Sweden, 2004.

# ANNEX

## Annex 1: Robotrom Dimension



## Annex 2 : Robotrom Wheel Dimension



## **RESUME**

N'djadjo Romuald Kouakou was born in Côte d'Ivoire (Ivory Coast) 1988. He studied his secondary and high school, in Lycée Mixte of Yamoussoukro, Collège Orioux of N'douci, and a Technical high school of Yopougon, Abidjan, Côte d'Ivoire (Ivory Coast) respectively. He received the Baccalaureate degrees in Biochemical in 2011 and the degree of High Technician in Electronics from National Institute Specialized of professional Training (INSFP 500 Logements) of Sétif, Algeria, Department of Industrial Electronics in 2015. He got his Turkish language education from Sakarya University's Turkish Language Center for foreigner from 2015 to 2016. He started studying in M.Sc in Sakarya University, Sakarya, Turkey, Department of Electrical and Electronics Engineering from 2016. He is interested in control and automation system, robotics, signal and image processing, tracking, microcontroller, programming with Matlab, C, C# and Java, 3D modelling and simulation.