Low Cost Inertial-based Localization System for a Service Robot

Rúben José Simões Lino

Dissertação apresentada na Faculdade de Ciências e Tecnologia
da Universidade Nova de Lisboa para a obtenção do grau de
Mestre em Engenharia Electrotécnica e de Computadores

Orientador: Doutor Pedro Alexandre da Costa Sousa

Júri

Presidente: Doutor José António Barata de Oliveira

Vogais: Doutor João Paulo Branquinho Pimentão

Doutor Pedro Alexandre da Costa Sousa

Março de 2011

UNIVERSIDADE NOVA DE LISBOA

Faculdade de Ciências e Tecnologia

Departamento de Engenharia Electrotécnica e de Computadores

Sistema de Localização Inercial de Baixo Custo para um Robô de Serviço

Por:

Rúben José Simões Lino

Dissertação apresentada na Faculdade de Ciências e Tecnologia
da Universidade Nova de Lisboa para a obtenção do grau de
Mestre em Engenharia Electrotécnica e de Computadores

Orientador: Prof. Pedro Alexandre da Costa Sousa

Lisboa
2011

NEW UNIVERSITY OF LISBON

Faculty of Sciences and Technology

Electrical Engineering Department

Low Cost Inertial-based Localization System for a Service Robot

Rúben José Simões Lino

Dissertation presented at Faculty of Sciences and Technology of the New University of Lisbon to attain the Master degree in Electrical and Computer Science Engineering

Supervisor: Prof. Pedro Alexandre da Costa Sousa

Lisbon

2011

Low Cost Inertial-based Localization System for a Service Robot

Acknowledgements

For all persons that, in some way, had contributed to this dissertation I want to express my deepest acknowledgments.

First of all I want to manifest a sincerely gratitude to Prof. Pedro Sousa, my dissertation supervisor, for providing the conditions to do this work, giving me this opportunity and for all advices and guidance along the way.

I can not forget my colleagues Pedro Gomes and Tiago Ferreira for all the interesting discussions, critics, comments and casual talking. To João Lisboa I have to thank all the wise observations and special critics and also his availability. Non forgetting also all my university friends and coworkers of Holos, SA.

I would like to thank Neuza from all of comprehension, support and encouragement that she always transmitted me, even along the toughest times.

At last, I have to refer one of the greatest values that we could have that is the family. For my parents and my sister I have to express the deepest gratitude for all of support and encouragement given during the whole life.

Abstract

The knowledge of a robot's location it's fundamental for most part of service robots. The success of tasks such as mapping and planning depend on a good robot's position knowledge.

The main goal of this dissertation is to present a solution that provides a estimation of the robot's location. This is, a tracking system that can run either inside buildings or outside them, not taking into account just structured environments. Therefore, the localization system takes into account only measurements relative.

In the presented solution is used an AHRS device and digital encoders placed on wheels to make a estimation of robot's position. It also relies on the use of Kalman Filter to integrate sensorial information and deal with estimate errors.

The developed system was testes in real environments through its integration on real robot. The results revealed that is not possible to attain a good position estimation using only low-cost inertial sensors. Thus, is required the integration of more sensorial information, through absolute or relative measurements technologies, to provide a more accurate position estimation.

Keywords: localization, INS, service robots, mobile robots, Kalman filter, AHRS

Resumo

O conhecimento da sua localização é fundamental para a maior parte dos robôs de serviço. O sucesso de funções como construção de mapas e planeamento dependem de um bom conhecimento sobre a posição do robô.

O principal objectivo desta dissertação é apresentar uma solução que realize uma estimativa sobre a localização do robô. Um sistema de localização capaz de funcionar quer dentro de edifícios quer no seu exterior não tendo em conta ambientes estruturados. Deste modo, o sistema de localização tem em conta, somente, medições relativas.

Na solução apresentada é utilizado um dispositivo AHRS e *encoders* digitais nas rodas para realizar uma estimativa da posição do robô. Recorre-se ainda à utilização de Filtro de Kalman para integrar informação sensorial e lidar com erros provenientes da estimativa.

O sistema desenvolvido foi testado em ambientes reais através da sua integração num robô físico. Os resultados indicam que não é possível obter uma boa estimativa de localização utilizando apenas sensores inerciais de baixo-custo. Sendo necessária a integração de informação sensorial, através de medidas absolutas ou relativas, de modo a fornecer uma estimativa com menos erro.

Palavras-chave: localização, INS, robôs de serviço, robôs móveis, filtro de Kalman, AHRS

Contents

List of Figures

List of Tables

Symbols and Notations

Symbols	Description
AHRS	Attitude and Heading Reference System
AOA	Angle Of Arrival
DGPS	Differential Global Positioning System
DTG	Dynamically Tuned Gyroscope
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
ENU	East-North-Up
FOG	Fiber Optic Gyroscope
GPS	Global Positioning System
IFR	International Federation of Robotics
IMU	Instrument Measurement Unit
INS	Inertial Navigation System
kNN	*k*-Nearest Neighbor
LPT	Local Tangent Plane
MEMS	Microelectromechanical Systems
NED	North-East-Down
RFID	Radio Frequency Identification
RPY	Roll-Pitch-Yaw
RSS	Radio Signal Strengths
RTOF	Roundtrip Time Of Flight
SLAM	Simultaneous Localization and Mapping
SMP	Smallest M-vertex Polygon
SVM	Support Vector Machine
TDOA	Time-Difference Of Arrival
TOA	Time of Arrival
UWB	Ultra-Wide Band
F	Specific force vector
A	Acceleration vector
G	Gravitational acceleration vector
ϕ	Roll Angle
θ	Pitch Angle
ψ	Yaw Angle

Ω	Angular speed
Ω	Angular speed of the Earth
L	Latitude
Λ	Longitude
H	Altitude
C_{ned}^{ecef}	Transformation matrix from ECEF to NED
C_{ecef}^{ned}	Transformation matrix from NED to ECEF
C_{ecef}^{eci}	Transformation matrix from ECI to ECEF
P	Estimate error covariance matrix
Q	Process noise covariance matrix
R	Measurement noise covariance matrix
A	State transition matrix
B	Control matrix

1. Introduction

In earlier 20th century, more precisely in 1920, the word robot was introduced by Karel Capek in his play RUR, Rossum's Universal Robot. Capek's idea of these beings was that robots were artificial automated workers with their own soul, but they were meant to serve. Since then, the development in robotic area allowed the use of robots in diverse areas making them an integrating part of our society.

In these days, robots are helping humans in monotonous, dangerous and difficult to achieve (like accurate or heavy weight related) tasks. Cleaning and housekeeping, elderly care, interplanetary exploration, humanitarian demining, tour-guide, underwater exploration and surveillance are some possible applications of a robot.

"From static, non-autonomous robots in the beginning, robots now become more and more mobile and autonomous" [Negenborn, 2003]. Autonomy and mobility are such important characteristics for robots that need to move freely to do their tasks, like service robots. The International Federation of Robotics (IFR) defines a service robot as *"(…) a robot which operates semi- or fully autonomously to perform services useful to the well-being of humans and equipment, excluding manufacturing operations*" [IFR, 2009].

Navigation is a task that service robots must be able to do. Navigation can be divided in localization, obstacle detection and avoidance and planning . Therefore, they can move safely from its location to another one. According to Leonard and Durrant-Whyte [Leonard and Durrant-Whyte, 1991], the general proposition of navigation can be resumed in three questions: "where am I?", "where am I going?" and "how should I get there?".

The question "where am I?" is the focus of this thesis and relies with robot localization, which is one of the problems of service robots. Once autonomy and mobile are principal features of these robots, the knowledge about robot's position and orientation is absolutely needed. Thus, it is possible to solve other problems such as mapping, obstacle avoidance and planning and, thereby, have an approach to answer the problem of navigation.

It's common to see localization issues being solved by technologies like GPS, inertial systems or vehicle's odometry,

Shortly, to solve the localization problem there are various sensors and techniques that are divided in absolute position measurements and relative measurements. The relative measurements have the great vantage of being self contained in the robot and not jammable but have as major drawback it position error be unbounded with the usage over the time. In other hand, the absolute position measurements have a bounded error with its resolution depending on the technology used.

Its major disadvantage is that the environment has to be structured because this kinds of technologies need to be placed in vehicle and outside it. This means that references have to be placed strategically (in the environment where the robot will operate) in order to have a better error/cost relation. With low price and accuracy of these days' GPS technology these measurements are, essentially, applied in indoor environments where GPS is unsuitable.

Nowadays, location is usually obtained from combining the short-term accuracy of relative measurements methods with the time-independent bounded error of the absolute measurements methods. "*The actual trend is to exploit the complementary nature of these two kinds of sensorial information to improve the precision of the localization procedure*" [Ippoliti et al., 2005].

Simultaneous localization and Mapping is being, in these days, a subject of research in mobile robots area. The basic idea of SLAM is to place an autonomous vehicle at an unknown location in an unknown environment and have it build a map, using only relative observations, and then using this map to navigate [Dissanayake et al., 2001]. However, this thesis doesn't try to contribute to this field of research.

In this work, the goal is to answer the question "where am I?" using only dead-reckoning systems[1]. A reliable estimate of a robot's will be attempted with a low-cost inertial measurement unit as main sensor. Furthermore, sensorial odometric information will be added to improve the estimate's precision.

A priori, it is possible to say that this error will be unbounded along the time since won't be used absolute position measurements.

1.1 Problem Statement

The main goal of this dissertation is to present a localization solution including only relative measurements of an attitude and heading reference system and digital encoders placed in wheels. Indoor and outdoor environments are included once the system only relies in relative measurements. Therefore, systems such as GPS, which is widely used to do localization, is discarded and a low cost inertial measurement unit and digital encoder information will be used.

The rotational speed of the Earth is one problem of localization through a strapdown inertial system once that it inflicts particular forces that cause noise on acceleration and gyroscopes (gyroscope drift). These particular forces are known as local gravitational force and Coriolis acceleration.

[1] Dead-reckoning systems estimates the current position based upon knowledge of the last position. Through data like acceleration, velocity and time elapsed is estimated the current position. The error of these systems increase over time because position estimate is obtained from the previous position.

Dead-reckoning systems have as main disadvantage the cumulative error of position. So it is needed a method that deals with errors and could combine the data from various sources.

Another problem is information integration of various sources. It is needed to acquire data from sensors in real-time in order to do a better estimation of position.

1.2 Solution Prospect

In this thesis will be presented different solutions that will use a inertial measurement unit and digital encoders placed on wheels. This combined with navigation equations and Kalman filters will provide some position estimations.

To correct acceleration by the influences of the Earth's movement it will be implemented the navigation equations. It takes in account the rotational speed of the earth to translate the platform's acceleration to an Earth fixed frame. With navigation equations it is possible to know the exact value of Coriolis acceleration and local gravitational force and, thus, deduct it to the acceleration vector.

Kalman filter is a tool widely use in control and prediction areas and also in autonomous or assisted navigation. It uses the all available information to predict the state of system and to project the error ahead. After that, with complemental measurements it corrects the estimate. In this way, it will be used a Kalman filter to estimate the robot's position and dealing with cumulative dead-reckoning errors.

To attempt to solve the localization of a mobile robot, with an inertial sensor as the main reference, will be implemented and tested two methods: an algorithm referred as double integration algorithm, which only uses the data provided by inertial sensor combined with navigation equations to obtain a estimate of robot's velocity and position; Kalman filter appears as another solution, implementing it with data received from inertial sensor and aided by velocity obtained by digital encoder. All of this, integrated in a Player module.

The Player is a software tool capable of communicate with hardware normally used in robot's projects. It is modular, this means that it's separated in different logical modules which helps the integration of new modules (such as drivers or algorithms). It has itself a messaging system to communicate between modules.

1.3 Dissertation Outline

This dissertation is organized as follows:

Chapter 1: introduces the reader to the problem of robot localization;

Chapter 2: exposes a brief overview of the state of the art about localization in general and then giving the focus to robot localization;

Chapter 3: provides an overview of supporting mechanisms used in robot localization;

Chapter 4: description of algorithms implemented;

Chapter 5: presents the experimental results;

Chapter 6: encompasses the conclusions about the developed work and presents some future work possibilities.

2. State of the Art

In this Chapter will be presented and discussed the main techniques to do mobile robot localization. In section 2.1 there are the most common techniques to do localization using absolute position measurements, in section 2.2 will be described localization techniques using relative position measurements. In section 2.3 are presented localization methods that combine relative position measurements with absolute position measurements to achieve a more accurate estimate of position.

Over the years, with the development of robotic industry and appearance of mobile robots the robot localization became an emergent problem to solve. Since then, a variety of systems, sensors and techniques for mobile robot positioning are being subject of interest for researchers and engineers. According to Borenstein et al. [Borenstein et al., 1997], an accurate knowledge of a vehicle's position is fundamental in mobile robot application.

Borenstein et al. divides the localization techniques into Relative Position Measurements and Absolute Position Measurements [Borenstein et al., 1997]. He also refers to Relative Position Measurements as dead-reckoning and to Absolute Position Measurements as reference-based systems.

2.1 Absolute Position Measurements

Absolute localization requires a structured environment where the robot is operating. Having sensors in the robot (under a structured environment) providing data to it or have some outside references that help localizing robot is considered as absolute measurements. Among the most known and used are the GPS and other wireless systems (such as WLAN, cellular based and UWB). Another system widely use in robot localization is the usage of landmarks which could be natural (such as trees and room corners) or artificial (barcodes, pattern images, etc…).

2.1.1 Wireless

Radio signal propagation in indoor environments suffers from severe multipath and there is a low probability for availability of line of sight. Environment characteristics such as wall thickness, floor layout and reflecting surfaces need to be considered to model the radio propagation. So it's

common to use triangulation, positioning algorithms using scene analysis or proximity methods in order to have an estimate for location discarding the measurement errors.

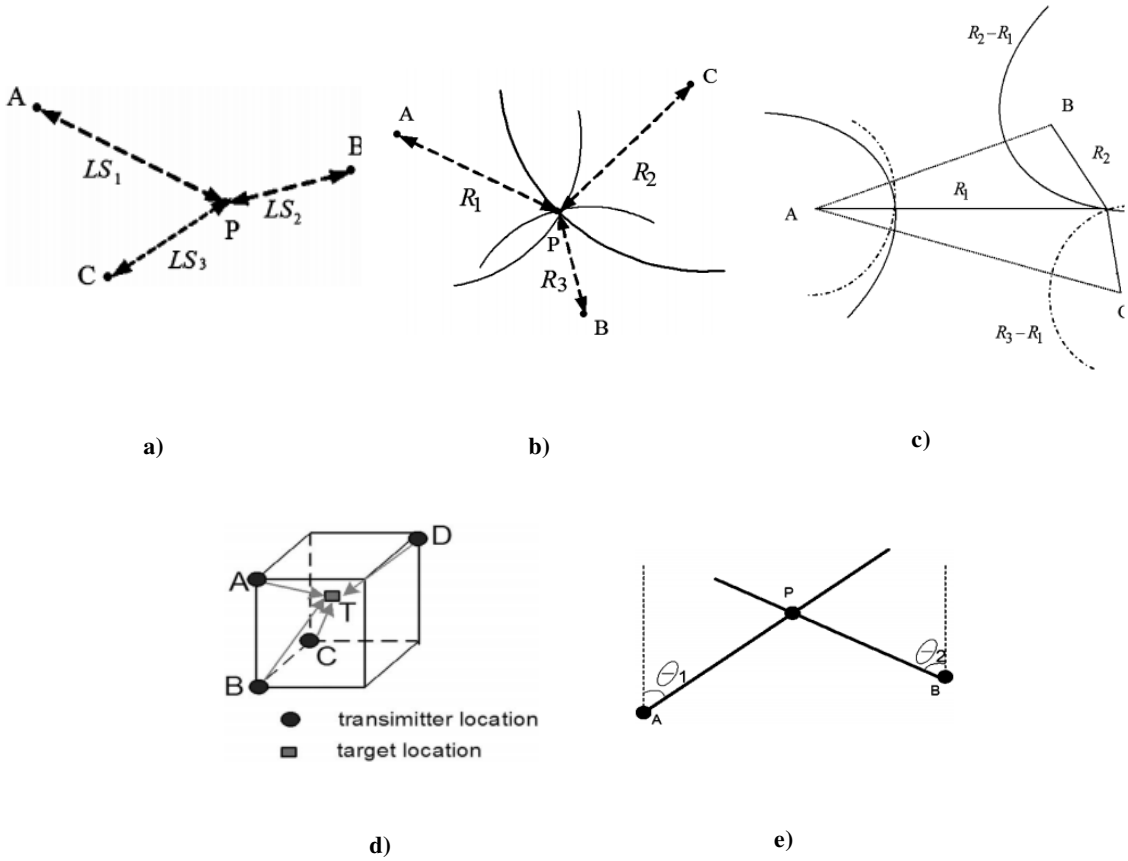


Figure 2.1 – a) positioning based on RSS, where LS_1, LS_2 and LS_3 denote measured path loss; b) positioning based on TOA/RTOF measurements; c) positioning based on TDOA measurements; d) positioning based on signal phase; e) positioning based on AOA; adapted from Liu et al. [Liu et al., 2007]

Shortly, triangulation is based on geometric properties of triangles to estimate the target location. It could be divided in lateration and angulation. Lateration estimates the position of an object by measuring its distance from multiple reference points (it includes RSS-based, TOA – *Time of Arrival*, TDOA – *Time difference of Arrival*, RTOF – *Roundtrip Time of Flight* and received signal phase method). In angulation techniques (AOA) the location of the target is estimated by the intersection of several pairs of angle direction lines. Liu et al. gives an more precise explanation of these techniques [Liu et al., 2007].

Scene analysis is associated to fingerprinting location. This means that includes algorithms that estimate an object's location by matching the measurement with the closest *a priori* location fingerprint (algorithms that first collect features). So, they are composed by two stages: the offline stage (location coordinate/labels and respective signal strengths from nearby stations/measuring units are collected) and the online stage (where it's used the observed signal strengths and previous information collected to estimate the location [Liu et al., 2007]. According to Liu et al., there are at least five types fingerprinting based positioning algorithms using this pattern recognition:

probabilistic methods, *k*-nearest neighbor (*kNN*), neural networks, support vector machine (SVM) and smallest M-vertex polygon (SMP).

Proximity algorithms are very simple to implement and provide symbolic relative location information [Liu et al., 2007]. The simplest application of proximity is when a mobile target is detected by a single antenna and, knowing the location of antenna, it's possible to have an approximate location of mobile target. Infrared radiation (IR), radio frequency identification (RFID) and cell identification (Cell-ID) are some proximity localization methods.

Wireless systems are common used in indoor localization. Liu et al., provides an overview of the existing wireless indoor positioning solutions, explaining the different techniques and systems such as WLAN, RFID, UWB, GPS, Bluetooth, Ultrasounds, ZIGBEE and GSM. In Figure 2.2 it's presented the principal wireless location systems and its accuracy. Some of these will be approached ahead.

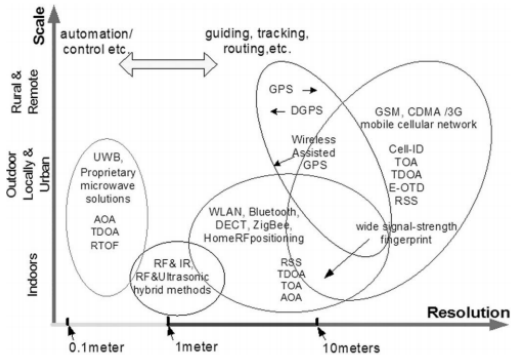


Figure 2.2 - Overview of principal positioning wireless techniques [Liu et al., 2007]

WLAN

As referred before, scene analysis' methods are widely used in wireless systems. In WLAN localization the most common technique is fingerprinting location with radio signal strengths.

System/solution	Wireless technology	Positioning algorithm	accuracy	Precision
Microsoft Radar	RSS	*k*NN	3-5m	50% within around 2.5m and 90% within around 5.9m
DIT	RSS	MLP, SVM	3m	90% within 5.12m for SVM and 90% within 5.4m for MLP
Ekahau	RSS	Probabilistic method	1m	50% within 2m
MultiLoc	RSS	SMP	2.7m	50% within 2.7m

Table 1 - Some WLAN-based indoor positioning systems and solutions, adapted from Liu et al [Liu et al., 2007]

The system developed by Caceres et al. is one example of scene analysis method. They've developed a WLAN-based real time vehicle location system [Caceres et al., 2009]. Using as positioning algorithm a trained neural network with a map of received power fingerprints from WLAN Access Points, they have an approximate error of 7 meters.

UWB

As indicated in Figure 2.2 Ultra-Wide Band is one of the best resolution wireless localization system. UWB is based on sending ultrashort pulses [Liu et al., 2007]. It uses frequencies from 3.1 to 10.6 GHz and transmits simultaneously a signal over the multiple bands. As well as the others wireless systems used in location, UWB sensors must be placed at known places and techniques as TDOA and AOA (or both together) are used to obtain a real-time 2D or 3D position. UWB location exploits the characteristics of time synchronoization of UWB communication to achieve a very high indoor localization (20 cm).

It short duration pulses are easily filtered and determined which signals are correct and which derive from multipath. The signal of UWB passes easily through walls, equipment and clothing, but has metallic elements or liquid materials will cause interferences on signal.

A well known system is the one developed by Ubisense. Using a combination of TDOA and AOA it provides and 2D/3D position with an accuracy of 15 cm where 99% of time with an error below of 0.3 m [Liu et al., 2007]. González et al. developed a combination of UWB and GPS for indoor-outdoor vehicle localization [González et al., 2009] that will be focused in section 2.3.

System/solution	Wireless technology	Positioning algorithm	accuracy	Precision
Ubisense	UWB (TDOA+AOA)	Least Square	15 cm	99% within 0.3m
Sappire Dart	UWB (TDOA)	Least Square	<0.3m	50% within 0.3m

Table 2 – Some UWB indoor positioning systems and solutions, adapted from Liu et al [Liu et al., 2007]

RFID

Using RFID it's possible to determine the location of an object through proximity or scene analysis methods. Using passive tags placed at known locations and a reader placed in the object it's possible to know approximate position of the object. LANDMARC [Ni et al., 2004] and SpotON [Hightower et al., 2000] methods use active tags and fingerprinting locating through RSS.

System/solution	Wireless technology	Positioning algorithm	accuracy	Precision
SpotON	Active RFID RSS	Ad-Hoc Lateration	Depends on the size of cluster	N/A
LANDMARC	Active RFID RSS	*k*NN	3m	50% within 1m

Table 3 – Some RFID-based indoor positioning systems and solutions

GPS

Global Positioning Systems (GPS) are, probably, the most used approach to do navigation. It was developed by the US Department of Defense. The system is composed by 24 satellites in the Earth's orbit and ground stations. The position of the GPS receiver is computed based on the principle of trilateration after receiving information about satellites' localization combined with the time of flight of radio frequency signal [Ashkenazi et al., 2000].

In mobile robot localization is not very common it appears a stand-alone method to do localization or navigation once that it could only be used outdoors in areas covered by technology's satellites. In urban of dense forest environments the GPS suffers from signal blockage and multipath interference. Besides, a regular GPS provides an accuracy of <10m [Ashkenazi et al., 2000] which, could not match the requirements of application.

There is an augmentation of this technology which is, commonly, used in mobile robot localization – DGPS. The Differential Global Positioning System has a pre-determined reference stations and can offer an accuracy of 1-2m [Ashkenazi et al., 2000]. Because of the poor coverage of signal for indoor environments its accuracy is low and it is unsuitable to be used to indoor position estimation. Another augmentation developed to overcome limitations of conventional GPS is the A-GPS (Assisted GPS). Using a location server with a reference GPS receiver, it combines the weak GPS signals with mobile station (or wireless handset) in order to produce a position estimate. It has 5-50 m accuracy in indoor environments [Liu et al., 2007].

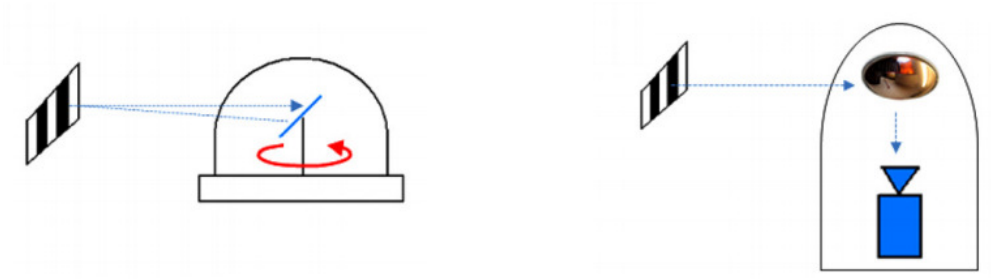
Because it is widely used its price is decreasing over the time, which is, associated to characteristics, probably the main advantage. In other side, the incapacity of being used indoors is one of its problems.

2.1.2 Landmarks

In indoor or outdoor environments landmarks utilization is a very common method used in mobile robot localization since robots became mobile and autonomous. Basically, there are two kinds of landmarks: natural and artificial landmarks. Natural landmarks are those objects or features that are already in the environment and have a function other than robot navigation while artificial landmarks are specially designed to do navigation and placed in the specific locals with this unique propose [Borenstein et al.,1997].

As referred before and mention by Loevsky and Shimshoni robot localization is essential for path planning to a desired location [Loevsky and Shimshoni, 2010] and it's a key to many applications. They presented an indoor localization method based on artificial landmarks and triangulation method.

In these methods the environment has to be structured. This is, the landmarks have to be placed at known locations in order to known the robot's position when it sees the landmarks. These landmarks can be detected in various ways. The usage of an omindirectional camera (Figure 2.3 a)) or a rotating laser beam (Figure 2.3 b)) are two possible methods and they are illustrated on Figure 2.3



(a) (b)

Figure 2.3 - Two possible methods to detect artificial landmarks: (a) laser beam is sent towards the landmark and received; (b) the landmark is detected by the omnidirectional camera image.

In both cases the sensor used must be mounted on robot. The usage of laser beams or cameras makes allows to identify the landmarks and measure their bearings. Using a laser scanner and artificial landmarks made of retro-reflective material so they can be detected easily detected by photodetector. According to Loevsky and Shimshoni the landmarks can be single-stripped or in form of a barcode [Loevsky and Shimshoni, 2010]. In form of a barcode the returning beam is analyzed and landmarks can be identified. To identify landmarks using an omnidirectional camera (Figure 2.3 b)) the captured image has to be processed and landmarks must be distinguishable from other objects in the scene for an image processing algorithm can detect them.

The orientation of the robot can be obtained measuring the bearings of the landmarks and location is then obtained by triangulation.

2.2 Relative Position Measurements

These methods are based on the data provided by sensors belonging only to robot and whose measure the vehicle's behavior. These methods are very often known as dead-reckoning and have the particularity of its estimate accumulates error along the time while it is being used. Methods using inertial measurements and odometry are the most common in robot localization.

2.2.1 Inertial measurement unit

Inertial measurement units consist in having a set of accelerometers and a set o gyroscopes. Both disposed orthogonally in order to obtain inertial accelerations and angular speeds in the three axes.

Through the acceleration and angular velocity obtained it's possible to compute the variation of position and angle using the following equations:

$$p = \int_{t_0}^{t_1} \int_{t_0}^{t_1} a \, dt \, dt$$

$$\theta = \int_{t_0}^{t_1} \omega dt$$

Though, it is necessary the usage of methods to compensate the attitude of sensor and gravity.

It is very common see inertial navigation system applications in ships, submarines, aircrafts, spacecrafts and guided missiles. They have the advantage of being self-contained and not jammable. However, inertial sensors are mostly unsuitable for accurate positioning over extended period of time [Borenstein et al., 1997]. Due to the price of high-quality INS it isn't very common to use in robot applications. Thus, it is often used low-cost inertial sensors combined with other sensorial information as we will see in Section 2.3.

2.2.2 Odometry

Odometry is based on simple equations which hold true when wheel revelations can be translated accurately into linear displacement. Through wheel encoders' information and using the vehicle's correspondent kinematic model it is possible to know the velocity and position.

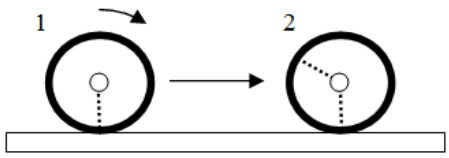


Figure 2.4 **- Single axis odometry**

The basic concept of how to obtain position and speed information (just for one axis) through an encoder placed on wheel is presented by the following explanation and Figure 2.4. Having a

digital encoder that counts the ticks, knowing the wheel's radius (r) and the ticks per revolution (t_r) it's possible to measure the number of ticks (n_t) in a time interval (t). So distance and speed are given, respectively, by:

$$distance = \frac{n_t \cdot 2\pi r}{t_r}$$

$$speed = \frac{distance}{t}$$

Nevertheless, in order to obtain 2D speed and distance information there is the need to implement the kinematic model of the robot which depends on its configuration.

Wheel slippage and cracks (systematic errors) or kinematic imperfections (non-systematic errors) are the main reason why this system is not so used alone [Borenstein et al., 1997]. Another type of odometry is using camera's sequence image to compute de vehicle's movement. This is a method which doesn't suffer with wheel slippage. Normally, it is combined with other sensorial data. In Section 2.3 it is presented some methods that use odometry as complementary information.

2.3 Robot Localization

Once that in robot localization is wanted the most accurate knowledge of position appeared the need to combine different sensors and techniques resulting in, mostly, "hybrid" localization methods. This means that is, often, combined relative position measurements with absolute position measurements to achieve the robot's position. This combination has the purpose of use the best of each group and hide its drawbacks: the difficulty of being always under structured environments and the error associated to the relative measurements position's methods. Thus, it's added the accuracy of absolute systems with the estimate (and independence of absolute measurements) of the relative methods.

Combination of Absolute Measurements Technologies

As referred before, González et al. developed a system that combines UWB with GPS for indoor-outdoor vehicle localization [González et al., 2009]. They've combined a widely used system for vehicle localization (GPS) with a technology used for efficient wireless communication (UWB) in order to be capable of localizing a vehicle in indoor or outdoor. Using the Monte-Carlo Localization algorithm (also known as particle filter). Their system has a GPS receiver and a UWB master antenna. With 3 slave UWB antennas placed indoors, they compare three localization

methods travelling in a indoor-outdoor circuit. The first one only relies on odometry, in the second the particle filter also has in consideration UWB and in the third GPS data is added to the particle filter. Figure 2.5 shows the errors of the three different methods over the time. The shadowed areas represent the part in navigation where GPS and UWB data were combined.

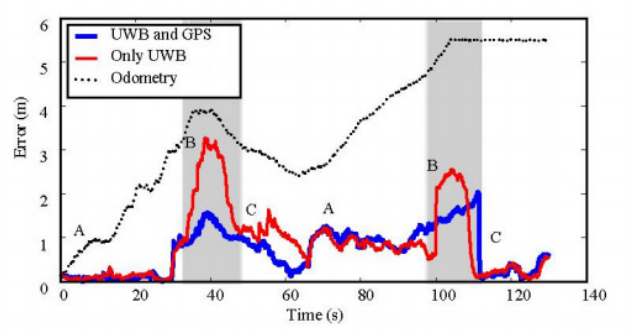


Figure 2.5 - UWB and GPS errors [González et al., 2009]

Combination of Absolute/Relative Measurements Technologies

Panzieri et al have presented an outdoor solution that fuses data from a GPS and from an inertial unit [Panzieri et al., 2002]. Their work presents a localization algorithm based on Kalman filtering that fuses information from an inertial platform and an inexpensive single GPS. It also uses map based data in their algorithm. In more details, their system have encoders measuring rotation of each motor, a inertial navigation system that provides measures of the robot linear accelerations and angular velocities, a laser scanner which measures the distances between a fixed point on the robot body and obstacle surfaces in the environment and a GPS measuring absolute position in the geodetic coordinates.

This work uses a very simple inertial model in the Kalman filter since inertial' input is $u_k = (a_k^x, \omega_k^z)^T$. This is, it uses only the data from x accelerometer and angular velocity from z gyroscope which only allows moving in a planar ground without error. Without attitude information the acceleration from gravity could not be deducted to x accelerometer. So this system has a very weak position estimate and the correction is made through the measurements: encoders' velocity, position from GPS and lased data. Panzieri's system is described in Figure 2.6.

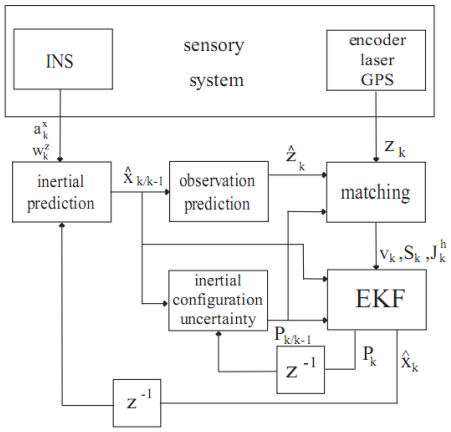


Figure 2.6 - Structure of Panzieri et al. localization system.

Agrawal and Konolige refer that in outdoors an accurate position can be obtained using a differential GPS and/or a high-quality, expensive inertial navigation system [Agrawal and Konolige, 2006]. So they developed an inexpensive localization solution using stereovision and GPS complemented by IMU/vehicle odometry. They've tested three methods: a solution that combines IMU and vehicle odometry, visual odometry only by itself and visual odometry combined with GPS as referred before. In their tests the last method has showed the best performance followed by raw visual odometry. With the biggest percentage error appeared the position obtained by vehicle odometry combined with an inexpensive inertial navigation sensor.

Their system relies on stereo vision to estimate frame-to-frame motion in real-time. Aside, it uses a Kalman filter based on inertial measurements that fuses wheel encoders and GPS data to estimate the motion when visual odometry fails.

Table 4 shows the percentage of error for the three methods in the different closed-circuits. The principal characteristics of each one and conclusions of its results are discussed in their work.

Run number	**1**	**2**	**3**	**4**
Distance (meters)	82.4	141.6	55.3	51.0
Method	Percentage Error			
IMU-Vehicle Odometry	1.3	11.4	11.0	31.0
Raw Visual Odometry	2.2	4.8	5.0	3.9
Visual Odometry & GPS	2.0	0.3	1.7	0.9

Table 4 - Agrawal et al. results: loop closure error in percentage

Liu et al. say that mobile robot localization in the general outdoor conditions is much more difficult than fairly flat and structured indoor environments [Liu et al., 2005]. According with Liu et al. in the uneven surfaces wheels encoders suffers much larger systematic errors and the outdoor environments are often semi-structured or totally unstructured.

They developed a multi-aided inertial navigation system for outdoor ground vehicles which uses a Kalman filter to fuse data from an INS and two wheel encoders that provide data to a kinematic model in order to extract a velocity estimate to be used by the filter. Also has been used GPS data for INS/GPS method developed. The general idea of the filter for aided INS is presented in Figure 2.7.

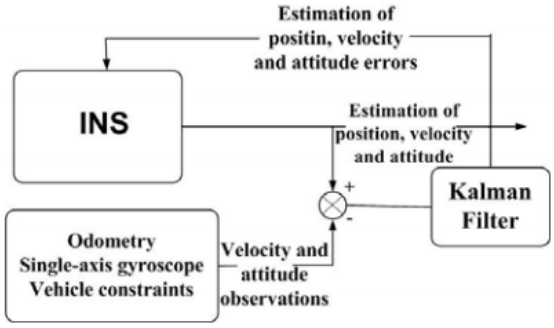
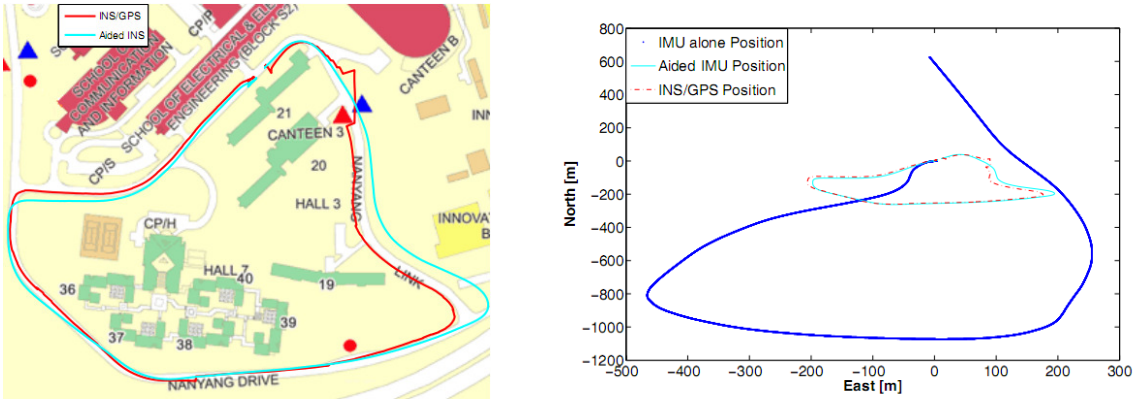


Figure 2.7 - Multi-aided inertial navigation system developed by Liu et al. [Liu et al., 2005]

Although the system has been developed to outdoor environments and implemented on a pickup truck, the same idea (without GPS) could be applied to an indoor vehicle either once the system doesn't use structured environments.

The tests for the three methods presented were made in a circuit (Figure 2.8 a)) of a total path around 1.1 Km that take approximately 3 minutes to be completed. Without any kind of aiding, the free INS can only last for a limited period of time as expected accurately as shown in b). They consider INS/GPS position as to be the ground truth even the trajectories present some discontinuities due to GPS errors [Liu et al., 2005]. The mean errors for the other methods in order to INS/GPS are shown in Table 5.



a) b)

Figure 2.8 - a) testing environment; b) the general paths from three methods [Liu et al., 2005]

	North (m)	East (m)
IMU	345	106
Aided IMU	3.7	7.5

Table 5 - Liu et al. errors from IMU and aided IMU methods.

Ippoliti's et al. work is a low-cost localization system for a robot using only internal sensors, i.e., relative position measurements. Their system is composed by odometers and an optical fiber gyroscope.

They've presented 3 different algorithms. In the three methods is achieved x and y position's estimate as well as robot's orientation. In the first one the position's estimate is only based on information received from wheels' encoders and kinematic of vehicle.

In the algorithm 2 the orientation is calculated using angular measures of a fiber optic gyroscope. The x and y position's estimate is computed applying this new orientation to the kinematic model used before. In the third algorithm a state-space approach is adopted to have a more general method of merging information. The adopted algorithm was an extended Kalman filter. Their results were obtained from a 108 meter indoor closed circuit.

Table 6 shows the results for the three methods. The value on table to each mark corresponds to the difference between the real mark and the position estimate – error of position's estimate. Because of using the fiber optic gyroscope, algorithms 2 and 3 present very similar errors.

	Mk 1	Mk 2	Mk 3	Mk 4	Mk 5	Mk 6	Stop
Algorithm1	0.014	0.143	0.690	4.760	1.868	3.770	6.572
Algorithm2	0.012	0.041	0.042	0.164	0.142	0.049	0.187
Algorithm3	0.012	0.037	0.035	0.150	0.106	0.030	0.161

Table 6 - Estimation errors (in meters) in correspondence of the marker configurations (distance between the marker and estimated position) [Ippoliti et al., 2005]

With only one gyroscope and odometric information, this system is only able to provide a reliable estimate 2d position in a planar ground.

Santana's work introduces the development of a strapdown inertial navigation system designed to reconstruct vehicle terrain trajectories [Santana, 2006]. Using an inertial measurement unit, an odometer and artificial landmarks, the system uses a Kalman filter for sensor data fusion. Its system is inertial-based and aided by observations of car's odometer and artificial landmarks. There were tested three configurations of the system: using only inertial-based system without any measurements, inertial-based system aided by velocity from odometer and inertial-based system combined with velocity from odometer and landmarks.

The tests were made in 2800 meters closed-circuit in a planar ground. Tests had an average duration of 5 minutes. The results obtained are shown in Table 7.

Method	Error (m)
Kalman filter with no reference	53208,3
Kalman filter with speed observation	97,98
Kalman filter with speed and landmark observations (spaced by 280 m)	8,95

Table 7 – Santana's results for the three different Kalman filter configuration.

3. Supporting Concepts

In this chapter it's presented all the theory and tools to support the implementation: in section 3.1 there is an overview of the coordinate systems commonly used in localization through strapdown inertial systems; in section 3.2 is described all the frames transformations that are needed; a summarized study of navigation equations is done in section 3.3; a brief overview of what a strapdown inertial system is and an explanation of the main differences between inertial navigation systems, attitude and heading reference systems and inertial measurement units is done in section 3.4; section 3.5 shows some theory about Kalman filter and describes its algorithm; at last, in section 3.6 is done a brief description of The Player Project.

3.1 Coordinate Systems used in Inertial Navigation

To say where a body is it is needed a coordinate system, so it is possible to refer, exactly, where it is. Though, to define an object's location it is needed to know the coordinate system that it's used. According to Grewal [Grewal, 2001] "navigation makes use of coordinates that are natural to the problem at hand". This is, inertial coordinates for inertial navigation, satellite orbital coordinates for GPS navigation and earth-fixed coordinates to locate a place in the earth. Among all possible coordinate systems to represent object's location, the most used in inertial navigation are the following:

Earth-Centered Inertial (ECI) – this is an inertial coordinate system which its origin is in the earth's gravity center. As shown in Figure 3.2, the z axis passes through the north pole, the x axis points to the sun and passes through the equatorial line. The direction of the y axis is given by the right hand rule.

Earth-Centered, Earth Fixed (ECEF) – as in the ECI system, the origin of the ECEF system also is in the earth's center of gravity and the z axis passes through the north pole. This coordinate systems differs from the previous one in the x axis, which is fixed to the earth and is in the intersection of Greenwich meridian and equatorial line. Once more, the y axis is deduced by the right and rule.

In ECEF system *longitudes* and *latitudes* are measured in degrees and are used to locate a place in the earth. *Longitudes* take values from 180º (East) to -180º (West) and the 0º is in the prime meridian that passes through Greenwich while *latitudes* takes values from 90º (North) to -90º

(South) with Equatorial line being the 0º. As shown in Figure 3.1, *longitude* (λ) is the angle between Greenwich meridian and P and *latitude* (L) is the angle between Equator and the point P.

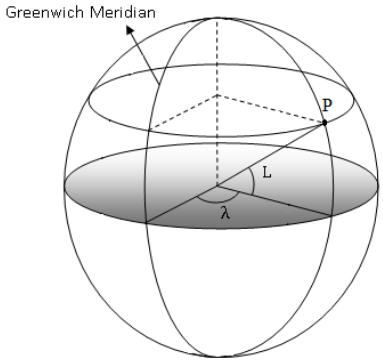


Figure 3.1- Representation of *latitude* and *longitude*.

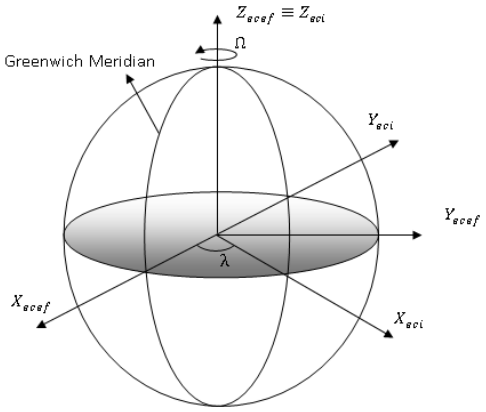


Figure 3.2 - ECI and ECEF coordinate systems

Local Tangent Plane (LPT) – LTP systems are a local coordinate system that represents the Earth as it was in the first idea of Earth – as a plane. This is, LPT is a local representation using the coordinate systems like North-East-Down and East-North-Up.

North-East-Down (NED) and East-North-Up (ENU) – as said before, these are LPT coordinate system. They are two common right-handed systems and their names refer the axis orientation. For both of the systems, the origin could be defined in anywhere and they have an axis pointing to North and another axis in East direction. The NED system uses the z Axis pointing to the center of the Earth whereas the ENU system uses the z axis pointing up (Figure 3.3).

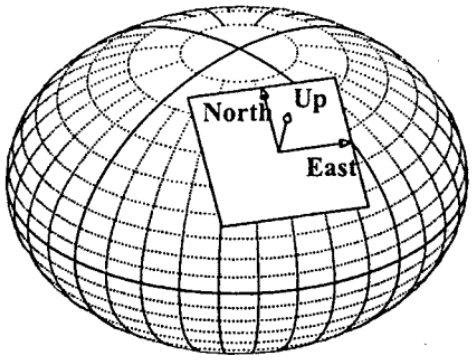


Figure 3.3 - ENU coordinate system [Grewal, 2001]

Roll-Pitch-Yaw (RPY) – this is a vehicle-fixed coordinate system. This means that this system changes as the attitude of the platform is altered. As shown in Figure 3.4 the roll axis matches with the motion axis (x), the pitch axis comes out of the right side and the yaw axis is set in a manner that turning right is positive. According to Grewal [Grewal, 2001] these systems are used for surface ships and ground vehicles.

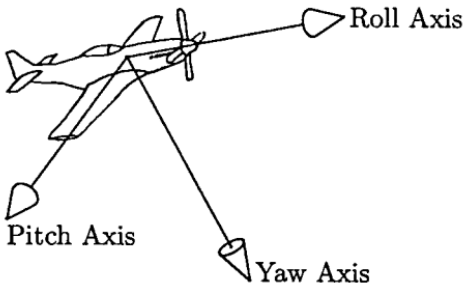


Figure 3.4 - RPY coordinate system [Grewal, 2001]

Vehicle Attitude Euler Angles – Euler angles could be defined as rotation angles between the earth-fixed referential and the object coordinate system. These angles are one of the possible methods to characterize the orientation (or attitude) of an object.

There are three angles:

Φ - Roll – rotation over the X axis (sensor's longitudinal axis);

Θ - Pitch – rotation over the Y axis, also known as elevation. It is measured positive from the horizontal plane upwards;

Ψ - Yaw – rotation over the Z axis, also called heading. The yaw is measured clockwise (to east) from north.

3.2 Coordinate Systems Transforms

The coordinate transforms are used to satisfy the need of convert a coordinate vector from the respective coordinate system to another one. Normally, these transformations involve translations and rotations in the different axes.

These transforms between two different coordinate systems may be represented by a matrix. Normally, this matrix is represented by C_{to}^{from}. This matrix gives the transform of a coordinate frame (denoted by *from*) to another coordinate frame (designed *to*).

To do such transformations there are methods like Euler Angles, Rotation Vectors, Direction Cosine Matrix and Quaternion each with its advantages and disadvantages.

3.2.1 Direction Cosine Matrix – Rotation Matrix (RPY to NED)

In Inertial Navigation Systems the rotation matrixes, also known as cosines matrixes, have an important role once that they allow the transformation of vector's components from one coordinate system to another one. The utilization of this kind of matrixes on inertial systems is due to the need to transform vector components from sensor's axis system to an earth fixed axis system. The following equations shows the construction processo of one of these matrixes in a two dimensional frame.

In Figure 3.5 there is P vector which represents the vector to be transformed to a new referential and has as original referential X1,Y1 with an inclination angle α. The rotation angle between the two frames it is represented by ψ and P' represents the P vector transformed from referential X1 ,Y1 to the coordinate system X2 ,Y2.

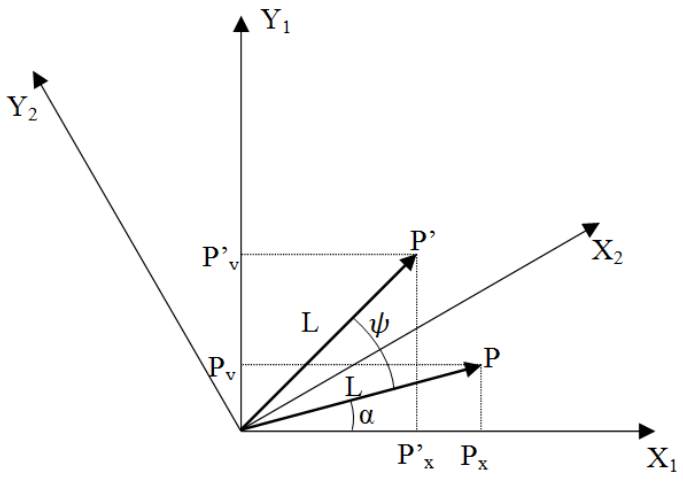


Figure 3.5 - Example of coordinate system transform

Using α and ψ angles it is possible to obtain the P vector projections in X1,Y1 and the P' projections in X2 ,Y2:

$$P_x = L \cdot \cos(\alpha)$$
$$P_y = L \cdot \sin(\alpha)$$
$$P'_x = L \cdot \cos(\alpha + \psi)$$
$$P'_y = L \cdot \sin(\alpha + \psi)$$

Through Euler's formula it is possible to represent the addition of angles as follows:

$$\cos(x + y) = \cos(x) \cdot \cos(y) - \sin(x) \cdot \sin(y)$$
$$\sin(x + y) = \cos(y) \cdot \sin(x) + \cos(x) \cdot \sin(y)$$

Using the previous expressions and P and P' vectors' projections in the respective coordinate systems:

$$P'_x = L \cdot \cos(\alpha) \cdot \cos(\psi) - L \cdot \sin(\alpha) \cdot \sin(\psi)$$
$$P'_y = L \cdot \cos(\psi) \cdot \sin(\alpha) + L \cdot \cos(\alpha) \cdot \sin(\psi)$$
$$P'_x = P_x \cdot \cos(\psi) - P_y \cdot \sin(\psi)$$
$$P'_y = P_x \cdot \sin(\psi) + P_y \cdot \cos(\psi)$$

If the two previous equations be placed in matrix form, we have

$$P' = \begin{bmatrix} P'_x \\ P'_y \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \cdot \begin{bmatrix} P_x \\ P_y \end{bmatrix}$$

this is, to obtain P' projections from P, it is multiplied the component's vector by transformation matrix.

$$P' = C \cdot P$$

Though, C is the transformation matrix

$$C = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix}$$

Considering the right hand rule and a three axes frame, once we have a rotation with a angle ψ in the Z axis, the matrix could be re-written in the following form

$$R_\psi^z = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In a similar way it is obtained the rotation matrixes to all axes:

Z Rotation

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

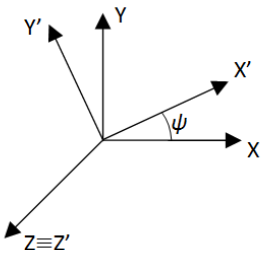


Figure 3.6 - Rotation over Z axis

Y Rotation

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

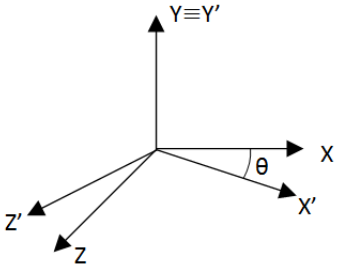


Figure 3.7 - Rotation over Y axis

X Rotation

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

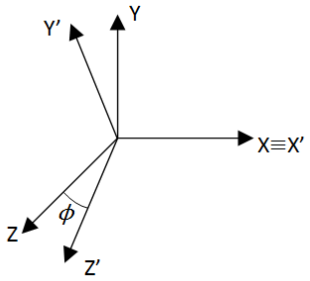


Figure 3.8 - Rotation over X axis

Combining all three possible rotations on the three different axes, each one with respective rotation angle, the following equations can be used

$$C = R_{\psi}^{Z} \cdot R_{\theta}^{Y} \cdot R_{\phi}^{X}$$

$$C = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix}$$

$$C = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

3.2.2 NED/ECEF and ENU/ECEF Transformations

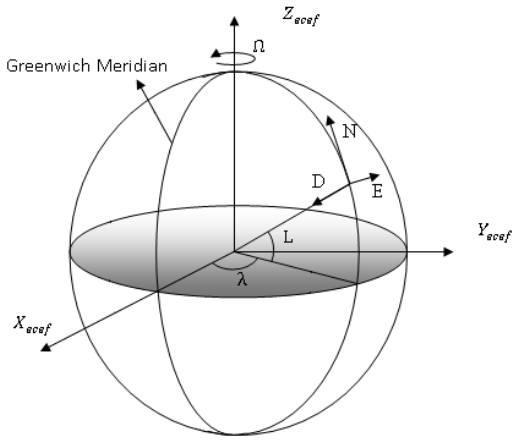


Figure 3.9 - Representation of NED coordinate system on ECEF

In applications involving inertial navigation it is, normally, used NED system which coincides with RPY (Figure 3.4) system when vehicle is facing. In this kind of applications, the use of ENU systems is also common. So, it is needed a transformation between the Local Tangent Plane used and a frame like ECEF to have the platform's position in Earth. These transformations are obtained through rotation matrixes using latitude (L) and longitude (λ).

The following equations show the transform matrixes between ECEF and NED systems:

$$C_{ned}^{ecef} = \begin{bmatrix} \cos\lambda & -\sin\lambda & 0 \\ \sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos L & 0 & -\sin L \\ 0 & 1 & 0 \\ \sin L & 0 & \cos L \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

$$C_{ned}^{ecef} = \begin{bmatrix} -\cos\lambda \cdot \sin L & -\sin\lambda & -\cos\lambda \cdot \cos L \\ -\sin\lambda \cdot \sin L & \cos\lambda & -\sin\lambda \cdot \cos L \\ \cos L & 0 & -\sin L \end{bmatrix}$$

$$C_{ecef}^{ned} = \begin{bmatrix} -\cos\lambda \cdot \sin L & -\sin\lambda \cdot \sin L & \cos L \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\lambda \cdot \cos L & -\sin\lambda \cdot \cos L & -\sin L \end{bmatrix}$$

Once that transformation matrix from NED to ENU and ENU to NED could be represented by

$$C_{ned}^{enu} = C_{enu}^{ned} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

the ECEF/ENU transformation can be computed in a similar way as ECEF/NED or computed from it. This is, the C_{enu}^{ecef} is result of multiplying the matrix C_{ned}^{enu} by C_{ecef}^{ned}:

$$C_{enu}^{ecef} = C_{enu}^{ned} \cdot C_{ned}^{ecef} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} -\cos\lambda \cdot \sin L & -\sin\lambda & -\cos\lambda \cdot \cos L \\ -\sin\lambda \cdot \sin L & \cos\lambda & -\sin\lambda \cdot \cos L \\ \cos L & 0 & -\sin L \end{bmatrix}$$

$$C_{enu}^{ecef} = \begin{bmatrix} -\sin\lambda & -\cos\lambda \cdot \sin L & \cos\lambda \cdot \cos L \\ \cos\lambda & -\sin\lambda \cdot \sin L & \sin\lambda \cdot \cos L \\ 0 & \cos L & \sin L \end{bmatrix}$$

$$C_{ecef}^{enu} = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\cos\lambda \cdot \sin L & -\sin\lambda \cdot \sin L & \cos L \\ \cos\lambda \cdot \cos L & \sin\lambda \cdot \cos L & \sin L \end{bmatrix}$$

3.2.3 Transformation between ECEF and ECI

ECEF and ECI coordinate systems are very similar. As the Figure 3.2 shows, the only difference is that in ECEF system the x axis is pointing to the Greenwich meridian while the x axis of ECI is pointing to the local meridian. So, to transform coordinates of one system to another only the longitude has to be considered. These transformations are a single rotation over the z axis (which is common to both systems).

The transform from ECI to ECEF and ECEF to ECI are represented by the following matrixes respectively:

$$C_{ecef}^{eci} = \begin{bmatrix} \cos\lambda & -\sin\lambda & 0 \\ \sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad C_{eci}^{ecef} = \begin{bmatrix} \cos\lambda & \sin\lambda & 0 \\ -\sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.3 Navigation Equations

Navigation equations are used, generally, in inertial navigations systems to know a body's localization with a respect to a fixed frame. The following explanation is based on Titterton [Titterton, 2004].

By Newton's second law of motion it is known that a body under a specific force will accelerate. The acceleration of a generic point in an inertial frame is given by:

$$\boldsymbol{a_i} = \left.\frac{\boldsymbol{d^2 r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} \tag{3.1}$$

Assuming that we have accelerometers in the three axes providing the specific force (**f**) at the body:

$$\boldsymbol{f} = \left.\frac{\boldsymbol{d^2 r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} - \boldsymbol{g} \tag{3.2}$$

Note that gravitational force (**g**) is always present (once we consider that the body is in the Earth) and its effect is always measured by accelerometers.

Rearranging the Equation (3.2) and taking in consideration Equation (3.1) it is quite noticeable that the acceleration measured contains the specific force and gravitational force.

$$\left.\frac{\boldsymbol{d^2 r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} = \boldsymbol{f} + \boldsymbol{g} \tag{3.3}$$

Equation (3.3) can be considered a navigation equation once that it is possible to obtain quantities of velocity and position by integration. So, the first integral of the acceleration gives the velocity and the second integral gives the position.

$$\boldsymbol{v_i} = \left.\frac{\boldsymbol{dr}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} \tag{3.4}$$

When the navigation is in the Earth there are some things that must be taken in account. Beyond the gravitational force there is the Earth's rotational velocity. This phenomenon causes additional apparent forces acting on the body. So, it is used the Coriolis theorem that relates the

velocity of the body over an inertial frame and a fixed frame. Therefore, a frame fixed to the Earth is needed and Earth's rotational velocity must be considered.

In Equation (3.5) $\boldsymbol{v_e}$ can be obtained from the inertial velocity (Equation (3.4)) using the theorem of Coriolis:

$$\boldsymbol{v_e} = \left.\frac{\boldsymbol{dr}}{\boldsymbol{dt}}\right|_{\boldsymbol{e}} = \boldsymbol{v_i} - \boldsymbol{\omega_{ie}} \otimes \boldsymbol{r} \tag{3.5}$$

The rotational velocity of the Earth with respect to an inertial frame is represented by $\boldsymbol{\omega_{ie}} = [\mathbf{0} \quad \mathbf{0} \quad \boldsymbol{\Omega}]^{\boldsymbol{T}}$.

Combining Equations (3.4) and (3.5) in order to translate the velocity from a fixed frame to a inertial frame. This is, computing the inertial velocity from the ground speed and Coriolis equation, we have:

$$\left.\frac{\boldsymbol{dr}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} = \left.\frac{\boldsymbol{dr}}{\boldsymbol{dt}}\right|_{\boldsymbol{e}} + \boldsymbol{\omega_{ie}} \otimes \boldsymbol{r} \tag{3.6}$$

Differentiating this expression and writing $\left.\frac{dr}{dt}\right|_e = v_e$,

$$\left.\frac{\boldsymbol{d^2r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} + \left.\frac{\boldsymbol{d}}{\boldsymbol{dt}}[\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}]\right|_{\boldsymbol{i}} \tag{3.7}$$

Applying the Coriolis equation in the form of Equation (3.6) to the second term in Equation (3.7), we have

$$\left.\frac{\boldsymbol{d^2r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} + \boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} + \left.\frac{\boldsymbol{d}}{\boldsymbol{dt}}[\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}]\right|_{\boldsymbol{i}} + \boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}] \tag{3.8}$$

where $\frac{\boldsymbol{d}}{\boldsymbol{dt}}\boldsymbol{\omega_{ie}} = 0$, once it's assumed that the turn rate of Earth is constant. The previous equation can be simplified:

$$\left.\frac{\boldsymbol{d^2r}}{\boldsymbol{dt^2}}\right|_{\boldsymbol{i}} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} + \boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} + \boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}] \tag{3.9}$$

Combining the previous equation with (3.3):

$$\boldsymbol{f} + \boldsymbol{g} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} + \boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} + \boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}] \tag{3.10}$$

$$\left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} = \boldsymbol{f} - \boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} - \boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}] + \boldsymbol{g} \tag{3.11}$$

In this equation, **f** represents the specific force of the acceleration acting on the body, the term $\boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e}$ is the Coriolis acceleration which is the acceleration caused by body's velocity over the surface of the rotating Earth, $\boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}]$ is the centripetal acceleration (also caused by the rotational velocity) and it is not distinguishable from the gravitational acceleration **g**. The sum of these two last terms (centripetal force and gravitational force) is also known as local gravity vector. It takes the symbol $\mathbf{g_l}$ and could be represented by:

$$\boldsymbol{g_1} = \boldsymbol{g} - \boldsymbol{\omega_{ie}} \otimes [\boldsymbol{\omega_{ie}} \otimes \boldsymbol{r}] \tag{3.12}$$

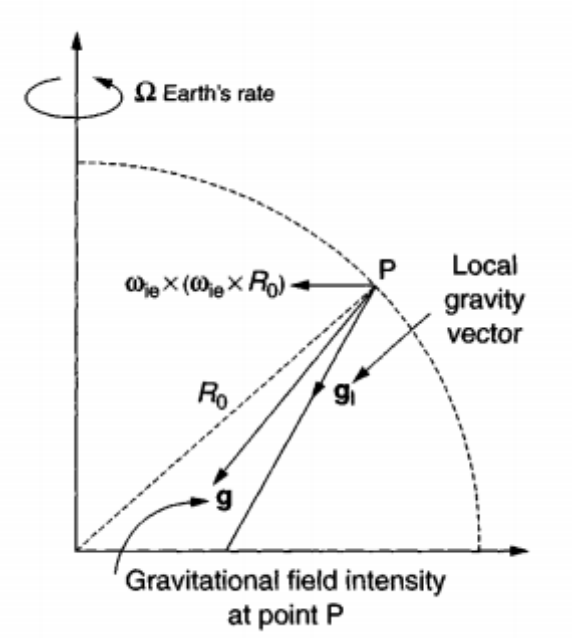


Figure 3.10 - Diagram showing the components of the gravitational field [Titterton, 2004]

3.3.1 Earth frame mechanization

Using the known ground speed in the inertial frame and the Coriolis equation, it's possible to obtain the ground speed in Earth-fixed frame ($\boldsymbol{v_e}^{\boldsymbol{e}}$):

$$\left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{e}} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} - \boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} \tag{3.13}$$

Merging the (3.11), (3.12) and (3.13), we have:

$$\left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{e}} = \boldsymbol{f} - 2\boldsymbol{\omega_{ie}} \otimes \boldsymbol{v_e} + \boldsymbol{g_1} \tag{3.14}$$

3.3.2 Local frame mechanization

In a similar way of Earth frame mechanization, the ground speed in local frame (as NED) it is obtained from ground speed in inertial frame (from Equation (3.11)). To do this, it's necessary to consider the turn rate of the Earth plus the turn rate of the local geographic frame with respect to the Earth-fixed frame. The following equation describes the

$$\left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{n}} = \left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{i}} - [\boldsymbol{\omega_{ie}} + \boldsymbol{\omega_{en}}] \otimes \boldsymbol{v_e} \tag{3.15}$$

Merging the (3.11), (3.12) and (3.15), we have

$$\left.\frac{\boldsymbol{dv_e}}{\boldsymbol{dt}}\right|_{\boldsymbol{n}} = \boldsymbol{f} - 2[\boldsymbol{\omega_{ie}} + \boldsymbol{\omega_{en}}] \otimes \boldsymbol{v_e} + \boldsymbol{g_1} \tag{3.16}$$

3.4 Strapdown Inertial Navigation Systems

Considering Newton's laws of motion (especially the 1st) that says that a body will move in uniform movement in a straight line or stay still in its state of rest unless it has a force impressed on it. Newton's second law tells us that a force over a body will produce a proportional acceleration of the body. So, to mathematically be able to obtain velocity and position by successive integration it is needed the acceleration, which can be obtained from accelerometers. Gyroscopes and magnetometers are, also, useful instruments to perceive body's orientation and, thus, to do inertial navigation.

A strapdown inertial navigation system is composed by sets of these instruments in the three different axes. Some authors refer to these systems as: "A strapdown inertial navigation system is basically formed from a set of inertial instruments and a computer" [Titterton, 2004]. Walchko and Mason says that a "strapdown system is a major hardware simplification of the old gimbaled system" [Walchko and Mason, 2002].

The main difference between inertial navigation systems and other types of navigation systems is that inertial systems are entirely self-contained within a vehicle and, for that reason, they do not depend on signal transmission from the vehicle to a base station where decisions due to navigation could, eventually, be taken. However, because of that absence of communication, this kind of systems only can be used to obtain a relative estimation of position. To estimate the body's localization in the earth it will depend on an accurate knowledge of vehicle's position at the start of navigation.

With the development of the technology that provided advances in sensing technology, appeared different implementations of strapdown systems. Nevertheless, the purpose and principles of operation are the same. In the original applications of inertial systems the inertial sensors are mounted on a stable platform and are mechanical isolated from the rotational motion of vehicle. Although its size and weight, this approach is still used, particularly for those who need a very accurate data as submarines and ships [Titterton, 2004].

An inertial navigation system can be divided into instrument cluster, instrument electronics, attitude computing and navigation computing. As said before, an inertial navigation system contains a set of inertial instruments (accelerometers, gyroscopes and magnetometers) divided in the three orthogonal axes as it is illustrated in Figure 3.11 – instrument cluster. For these inertial sensors be able to collect data it is necessary the instrument electronics. Power supplies, analogue-to-digital conversion electronics and output interface conditioning are some examples.

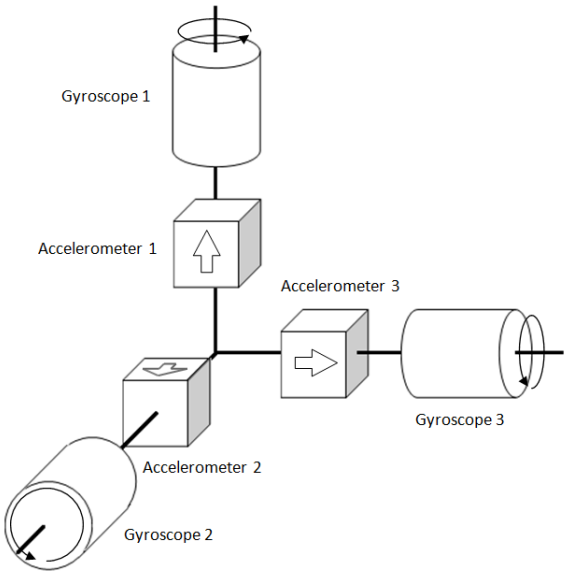


Figure 3.11 - Orthogonal instrument cluster arrangement, adapted from Titterton [Titterton, 2004]

The instrument cluster and the instrument electronics combined together is called an IMU (Inertial Measurement Unit). Combining the information obtained in IMU with a processor to do attitude computation we have an AHRS (Attitude and Heading Reference System). Finally, the Inertial Navigation System is the result of combining an AHRS with the Navigation Equations Computing as illustrated in Figure 3.12.

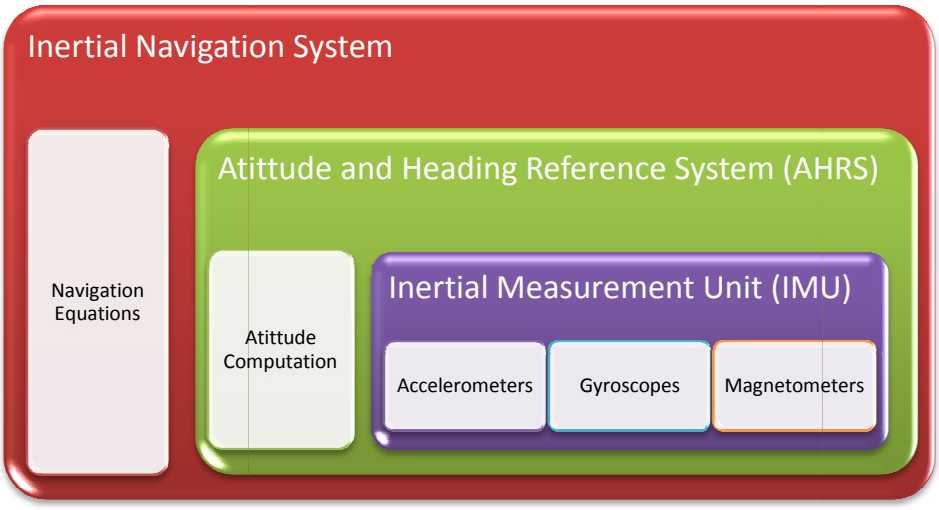


Figure 3.12 - Inertial Navigation System representation, adapted from Titterton [Titterton, 2004]

3.5 Kalman Filter

The Kalman filter is a mathematical method introduced by Rudolf Kalman in 1960. In [Kalman, 1960], Kalman formulates and solves the Wiener problem and introduces the Kalman filter as it is known today. Since then, many generalizations and extensions were developed and, thus, making this method one tool used in many applications to estimate the true state of a system. According to Welch and Bishop [Welch, 2006], the area of autonomous or assisted navigation was one of the most responsible for the study and research that led to generalizations and extensions mentioned before.

Maybeck describes the filter as a "(…) simply an optimal recursive data processing algorithm." [Maybeck, 1979]. In this method, it is possible to add any information that is available to estimate the true state of the system. This means that Kalman Filter processes all available measurements regardless of their precision in order to estimate the current value of the variables of interest. To estimate variable's state the Kalman filter uses its knowledge of system and measurement device dynamics, the statistical description of noises, measurement errors and uncertainly in the dynamic models and also takes in account all possible available information of initial values of variables of interest.

The following explanation of Kalman filter is based on the one given by Welch [Welch, 2006]. The state of the system is a vector *x* with *n* variables which describe interesting characteristics of the system. To estimate the state of system, $x \in \Re^n$, the Kalman filter uses the linear stochastic[2] difference equation:

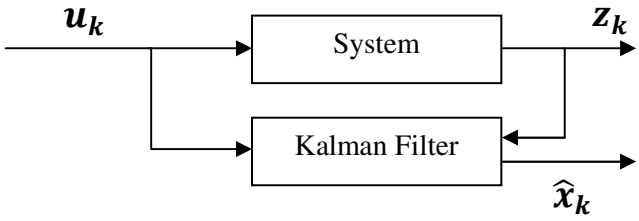
$$\boldsymbol{x_k = Ax_{k-1} + Bu_k + w_{k-1}} \tag{3.17}$$

In linear stochastic equation (3.17) there are matrices A and B. The matrix A is a $n \times n$ matrix designed as state transition matrix and represents the state evolution over time. This is, this matrix relates the state at $k-1$ with the state at k. B is known as control matrix, is a $n \times l$ matrix and relates the control input, $u \in \Re^l$, with the state *x*.

The measurement is represented by $\mathrm{z_k} \in \Re^m$

[2] A stochastic process is a non-deterministic process. In these processes, the next system's state is predicted having in consideration the previous state, processes' actions and a random variable.

$$\boldsymbol{z_k = Hx_k + v_k} \tag{3.18}$$



In (3.18), the true measurement $z_k \in \Re^m$ at time *k* depends linearly on the state of system x_k. The H matrix is used to relate the state x_k with the measurement z_k. This means, given a state the, H tells what the real measurement should be with no errors. However, once those measurements are obtained through sensors they are noisy and this noise is modeled by $v_k \in \Re^m$.

The random variables $\boldsymbol{w_k}$ and $\boldsymbol{v_k}$ represent, respectively, process and measurement noise. It's assumed that they are independent, white and with normal distributions and are represented by:

$$p(w) \sim N(0, Q)$$
$$p(v) \sim N(0, R)$$

where Q and R represents, respectively, process noise covariance and measurement noise covariance.

3.5.1 Computational Consideration

The *a priori* estimate of the estate at step *k* is represented by $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}$, where the '^' denotes that is an estimate, '-' denotes that is *a priori* and 'k' identifies the step of process the value belongs. $\hat{\boldsymbol{x}}_{\boldsymbol{k}}$ represents the *a posteriori* estimate of the system current state. Negenborn refers to *a priori* and *a posteriori* estimates as beliefs [Negenborn, 2003]. In his work, $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}$ is also referred as the *prior* belief ($Bel^{-}(x_k)$) which represents the belief about system's state after incorporating all information up to step *k*, including the last relative measurement but non including the absolute measurement z_k in step *k*. The *a posteriori* estimate corresponds to the *posteriori* belief ($Bel(x_k)$) and represents the belief about system's state after has included the absolute measurement

The error of *a priori* estimate and *a posteriori* estimate could be represented, respectively, as:

$$e_k^{-} \equiv x_k - \hat{x}_k^{-}$$
$$e_k \equiv x_k - \hat{x}_k$$

Thus, we have the *a priori* estimate error covariance and *a posteriori* estimate error covariance:

$$P_k^- \equiv E[e_k^- e_k^{-T}]$$

$$P_k = E[e_k {e_k}^T]$$

These covariance matrices reflect the uncertainty level in the respective estimates of the current state.

In the Kalman filter, an *a posteriori* estimate of system's state is represented by the following equation:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad \textbf{(3.19)}$$

The difference $(z_k - H\hat{x}_k^-)$ is called measurement innovation. This matrix expresses the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. This matrix also can be called residual and if it takes a zero value it means that predicted state is equal to the measured.

Returning to equation (3.19), the *a priori* estimate of state is a combination of the system's state obtained in the previous step (k-1) and the difference referred before weighted by K_k, Kalman gain matrix – Equation (3.20). This matrix minimizes the *a posteriori* error covariance when used in (3.19) once it aims to reduce the difference between the *a posteriori* state estimate (computed in (3.19)) and the true state of system. One of the more popular forms to represent this matrix is presented by the following equation:

$$K_k = {P_k}^- H^T (H {P_k}^- H^T + R)^{-1} \quad \textbf{(3.20)}$$

3.5.2 Kalman Filter Algorithm

As shown in Figure 3.13, the Kalman filter has two important steps: the *time update*, where the filter predicts the state vector and the covariance using the process information from $k-1$, and *measurement update*, where it is corrected the estimate based on the actual measurement.

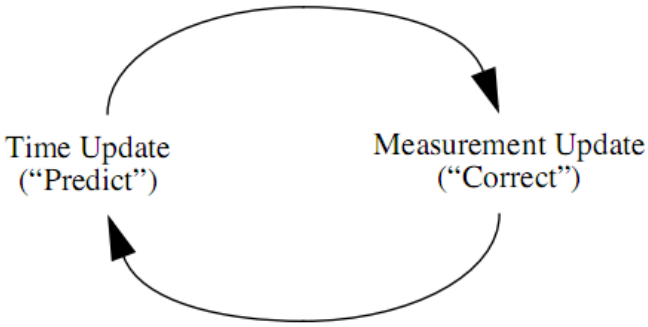


Figure 3.13 - The cycle of discrete Kalman filter algorithm [Welch, 2006]. *Time update* predicts the current state estimate while *measurement update* corrects the projected estimate using the measured data.

As we can see in the Figure 3.14 a more detailed picture of the cycle of discrete Kalman filter algorithm is presented. In the beginning of the algorithm information about system's state must be taken in account giving an initial estimate of system's state $\hat{\boldsymbol{x}}_{\boldsymbol{k-1}}$. The initial estimate error covariance matrix also must be given to the process can start.

Prediction Equations

In the step *time update*, also known as process or predictor, there are the process equations represented in (3.21) and (3.22). Here, the first step to be done is to project forward (in time) the *a priori* estimates of the current state ($\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} \in \Re^n$) and the second step is to project the estimate error covariance ($\boldsymbol{P}_{\boldsymbol{k}}^{-}$).

$$\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} = \boldsymbol{A}\hat{\boldsymbol{x}}_{\boldsymbol{k-1}} + \boldsymbol{B}\boldsymbol{u}_{\boldsymbol{k}} \tag{3.21}$$

$$\boldsymbol{P}_{\boldsymbol{k}}^{-} = \boldsymbol{A}\boldsymbol{P}_{\boldsymbol{k-1}}\boldsymbol{A}^{\boldsymbol{T}} + \boldsymbol{Q} \tag{3.22}$$

Correction Equations

In the *measurement update* it is computed the Kalman gain (Equation (3.23)) used after in the update of the estimate (Equation (3.24)) with the reference measurement, z_k, read at the step *k*. Right after, is updated the estimate error covariance in Equation (3.25) in order to use in the following *a priori* estimate projection.

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{3.23}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (Z_k - H \hat{x}_k^-) \tag{3.24}$$

$$P_k = (I - K_k H)\, P_k^- \tag{3.25}$$

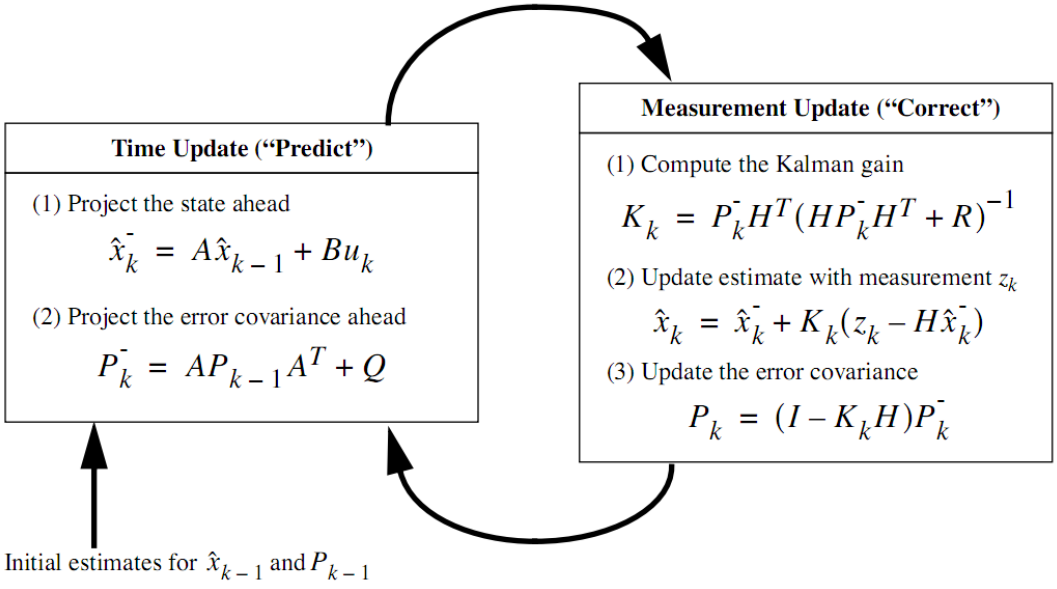


Figure 3.14 - A complete picture of the Kalman filter algorithm [Welch, 2006]

3.6 **Player**

The Player/Stage Project provides Open Source tools that simplify controller development for Multi-Robot Systems (multiple robot, distributed robot and sensor network). It began in University of South California Robotics Research Labs in 1999. Developed by researchers to help them with interfacing and simulation for Multi-Robot Systems is, in these days, widely used in robot control around the world.

Player could be considered as hardware abstraction layer (HAL) for robotic devices once that, according with Gerkey et al., "player is a robot device server that provides network transparent robot control" [Gerkey et al.,2003], it provides a clean and simple interface to the robot's sensor and actuators over the IP network.

Along with the Player appeared Stage and Gazebo which are, respectively, 2D and 3D multi-robot simulators. These simulators are prepared to provide virtual robots that make use of simulated devices instead of physical sensors, taking advantage of several sensor models including, laser rangefinders, cameras, etc. [Gomes, 2010].

In the Figure 3.15 it's shown an overview of the Player's structure and how clients, proxies and drivers are connected

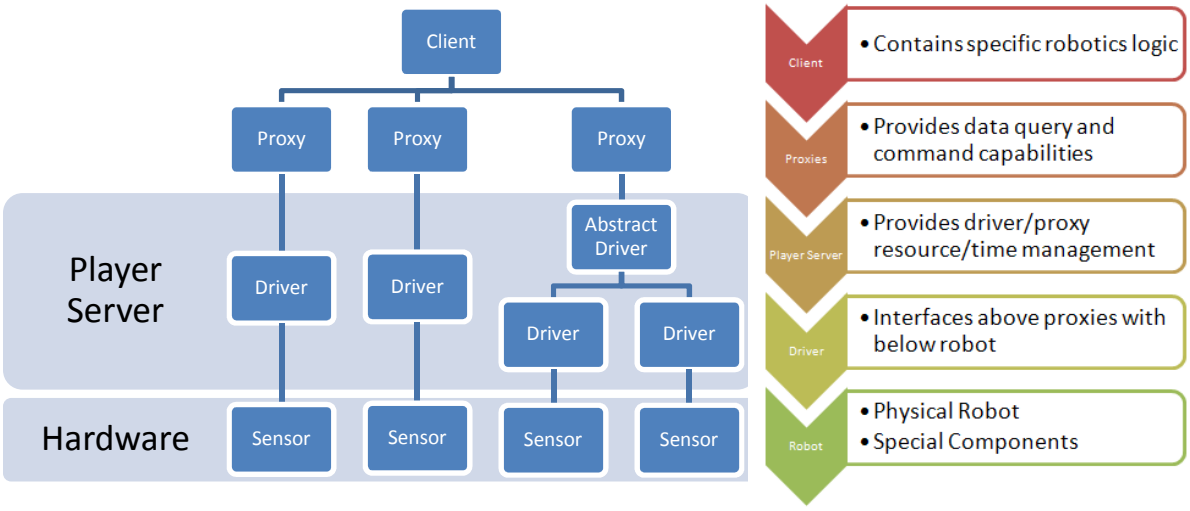


Figure 3.15 – General structure of Player.

A **sensor** is a physical component of the robot. Normally, is a sensor which its data needs to be delivered to higher levels of control or an actuator that is supposed to be used.

To operate a sensor or actuator it is needed a driver. A **driver** is a piece of software that talks to a robotic sensor, actuator, or algorithm, and translates its inputs and outputs to conform to one or more interfaces. The driver's job is hide the specifics of any given entity by making it appear to be the same as any other entity in its class [Player Manual, 2010].

Above the driver there is the **abstract driver**. Although they are very similar, an abstract driver doesn't communicate directly with the hardware. This is, an abstract driver uses other drivers, instead of hardware, as data sources. The use of this kind of drivers is very common to encapsulate an algorithm so it can be easily reused. Every driver follows the layout presented in Figure 3.16.

To proxies be able to interact with drivers, Player defines a set of standard **interfaces** (Position2d, position3d, IMU or GPS are some examples). This is a specification of how to interact with a certain class of sensor, actuator or some algorithm. The interface defines the syntax and semantics of all messages that can be exchanged between drivers or between drivers and clients.

A **device** is a driver that is bound to an interface. In other words, it is a driver together with an interface, so Player can interact to device directly either it is a real device or it belongs to a simulated robot.

An example to a better understanding these concepts is given by Owen [Owen, 2010] and it is adapted to the specific case of study in this work. Consider that you have a specific IMU, it can be from several different producers and, therefore, their specifications differ from one device to another. Consider the *xsensmt* driver, this is a particular driver that is capable to interact with an xsens device, it was developed to interact with those specific devices and, therefore it is able to retrieve data and to control device's options. The driver also translates the data retrieved in the format defined by the IMU interface.

As mentioned before a device is a driver bound to an interface. So a device could be created by xsensmt driver and the imu interface is what connects directly with player to data exchange.

Between drivers and the client there are proxies. A **proxy** is used to transfer data and commands from the specific device in the robot to the client. "*Proxies are standard C++ classes defined in the client library that provide methods to pass commands and/or query data*" [Player Manual, 2010].

At last and on the top of the structure illustrated in Figure 3.15 it is represented a **client**. A client is a high level piece of software used to control the robot. It uses proxies to retrieve information and to control the robot.

Although Player has several drivers and abstract drivers already developed the user may not have the specific hardware or might not want to use those algorithms. So, there is the need of developing a driver and it's important to know the methods contained on a driver and its run-time process [Gomes, 2010].

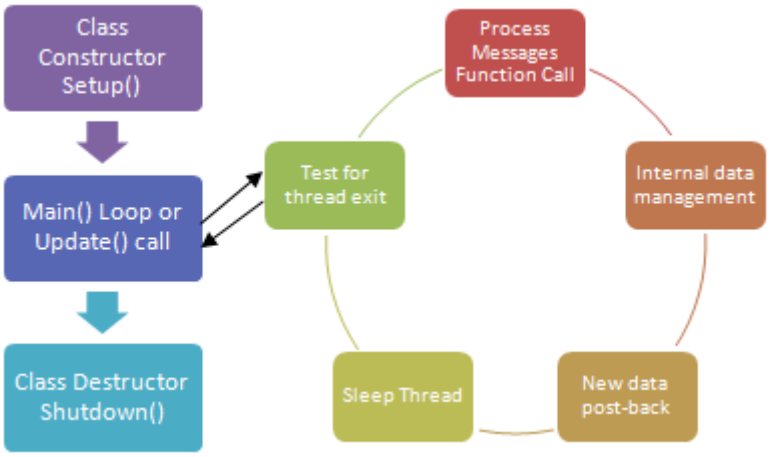


Figure 3.16 - The Run-Time Layout of a Player Driver [PsuRobotics, 2010]

4. Localization Solutions

In this chapter will be described the implemented solutions to the localization problem: in section 4 it's explained the double integration algorithm, which uses only inertial measurements combined with navigation equations (to prevent some acceleration errors) in order to achieve the goal; in section 4.2 it's described the Kalman filter implemented, forming what is known in literature as aided INS, to estimate the velocity and position; finally, in section 4.3 its presented how to construct a module capable to do the algorithms implemented using information from different sources using Player.

Basic concept of localization using acceleration

According to the Newton's second law that says that a body will accelerate with an acceleration proportional to the force that it is submitted and inversely proportional to its mass, the sum of all forces applied on an object is equal to its mass multiplied by its acceleration.

$$\Sigma F = ma$$

Once that acceleration is the derivative of the speed and speed is the derivative of position

$$\frac{dv}{dt} = a \tag{4.1}$$

$$\frac{dp}{dt} = v \tag{4.2}$$

In this way, it is possible to obtain variations of speed by integrating the acceleration in time

$$\dot{v} = \int_{t_0}^{t_1} a\, dt$$

In the same way, variations of position can be obtained from integrating the speed in a time interval:

$$\dot{p} = \int_{t_0}^{t_1} v\, dt$$

4.1 Double Integration Algorithm

This method consists in integrating the acceleration, given by the inertial sensor, to obtain the platform's speed and position. So, accelerations inflicted by the Earth have to be considered to obtain platform's final position.

The navigation equation (3.14) can be rewritten to:

$$a_e^n = C_n^b \cdot f^b - 2\omega_{ie}{}^{n} \otimes v_e{}^{n} + g_1 \tag{4.3}$$

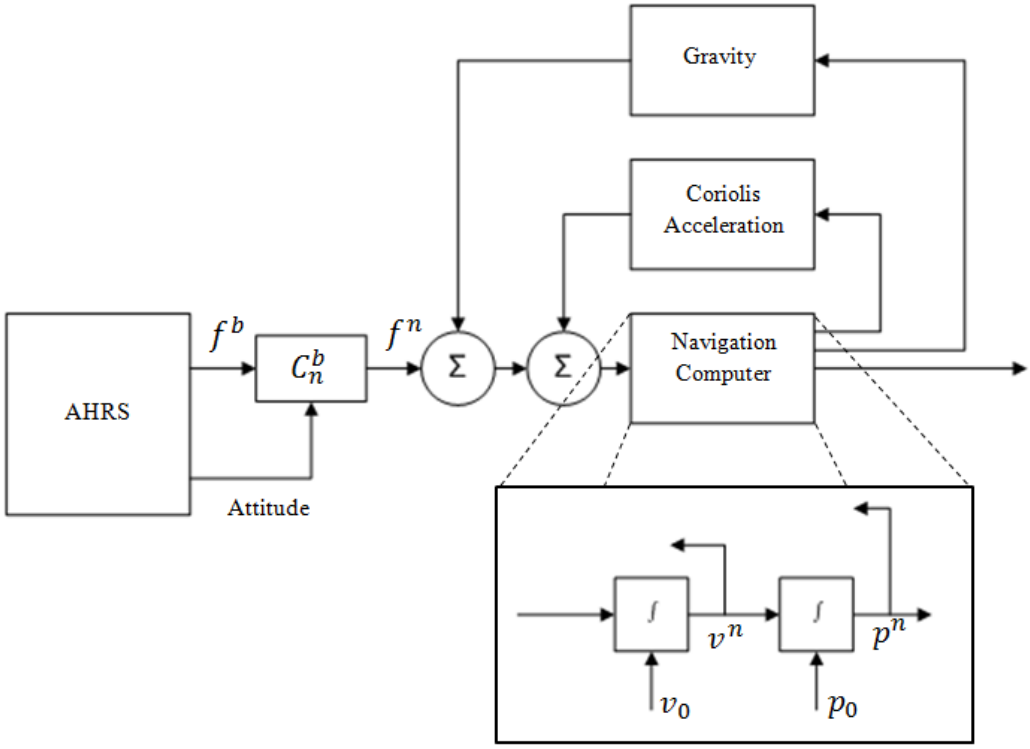


Figure 4.1 - Double integration algorithm with an AHRS

The transformation from force in body to force with the respect the navigation frame is represented in the equation (4.3) by the matrixC_n^b.

The double integration algorithm is represented in Figure 4.1 and it shows how it is possible to estimate a vehicle's position knowing its acceleration and its attitude (although heading isn't included in attitude's definition, we know from AHRS' definition that it provides heading too). So, a double integration over body's acceleration with the respect to the navigation frame can solve this problem. A drawback of these types of sensors is the fact that they measure all of accelerations over the body. This is, accelerations inflicted by the Earth as gravitational force or Coriolis

acceleration are also measured and must be considered in this algorithm. Therefore, these accelerations must be deducted from measured acceleration.

4.1.1 Acceleration Corrections

Remembering the Equation (3.11) it is quite noticeable that is needed to take out the Coriolis acceleration and the two components of gravitational field from the acceleration read from accelerometers.

To compute those accelerations it is needed ${\omega_{ie}}^n$ which is turn rate of Earth expressed in the local geographic frame. It is deduced in the next equation:

$${\omega_{ie}}^n = C_{ecef}^{ned} \cdot \omega_{ie} = \begin{bmatrix} -\cos\lambda \cdot \sin L & -\sin\lambda \cdot \sin L & \cos L \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\lambda \cdot \cos L & -\sin\lambda \cdot \cos L & -\sin L \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix} = \begin{bmatrix} \Omega\cos L \\ 0 \\ -\Omega\cos L \end{bmatrix}$$

Another element that is necessary is the turn rate of the local geographic frame with respect to the Earth-fixed frame. This is described and developed in Titterton [Titterton, 2004]:

$${\omega_{en}}^n = \begin{bmatrix} \frac{v_E}{R_0 + h} & -\frac{v_N}{R_0 + h} & -\frac{v_E}{R_0 + h}\tan L \end{bmatrix}^T$$

4.1.1.1 Gravity acceleration

To correct the acceleration from the local gravitational acceleration it's necessary to subtract g_1 from the Equation (3.4) to the acceleration measured. This gravitational acceleration represents the influence of gravity and centripetal forces over a body and can be represented by:

$${g_1}^n = g^n - {\omega_{ie}}^n \otimes ({\omega_{ie}}^n \otimes r) = \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix} - \begin{bmatrix} \Omega\cos L \\ 0 \\ -\Omega\cos L \end{bmatrix} \otimes \left(\begin{bmatrix} \Omega\cos L \\ 0 \\ -\Omega\cos L \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 0 \\ R_0 + h \end{bmatrix} \right)$$

$$= \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix} - \begin{bmatrix} \frac{\Omega^2 \sin(2\mathrm{L})(R_0 + h)}{2} \\ 0 \\ \frac{\Omega^2(1 + \cos(2\mathrm{L}))(R_0 + h)}{2} \end{bmatrix}$$

In this equation, g^n represents the gravitational acceleration with respect to the local frame.

$$g^n = \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix} (m/s^2)$$

4.1.1.2 Coriolis acceleration

In the same way, Coriolis acceleration must be taken out in order to correct the measured acceleration. Developing the correspondent term of Coriolis acceleration, we have:

$$a_{coriolis} = -2{\omega_{ie}}^n \otimes {v_e}^n = -2\begin{bmatrix} \Omega\cos L \\ 0 \\ -\Omega\cos L \end{bmatrix} \otimes \begin{bmatrix} v_N \\ v_E \\ v_D \end{bmatrix} = -2\begin{bmatrix} v_E \Omega \sin \mathrm{L} \\ -\Omega(v_D \cos \mathrm{L} + v_N \sin \mathrm{L}) \\ v_E \Omega \cos \mathrm{L} \end{bmatrix}$$

4.2 **Kalman filter**

As said before, the acceleration can be obtained by velocity's derivative or, in other words, the velocity can be obtained through the integration of the acceleration. Once that the IMU gives its acceleration, we have that acceleration measured by the unit ($a_{measured}$) contains the real acceleration (a_{real}) of the unit as well as errors (δa):

$$a_{measured} = a_{real} + \delta a \tag{4.4}$$

Considering the Equation (4.1) we have that the derivative of velocity is equal to the real acceleration of the body. So, it is possible to combine with Equation (4.4):

$$\frac{dv}{dt} = a_{measured} - \delta a$$

Integrating the previous equation in order to have the actual speed, v:

$$\int_{v_0}^{v} dv = \int_{t_0}^{t_1} (a_{measured} - \delta a)\, dt$$

$$v = v_0 + (a_{measured} - \delta a)t \tag{4.5}$$

Where, v_0 represents the vehicle's velocity before a new data arrives. The second member of t is the interval of time between t_1 and t_0, the time difference between the last sample received and the previous one.

From Equation (4.2) it's known that position can be obtained by integrating the vehicle's speed. So, in a similar way, it is possible to have the position from the velocity obtained in Equation (4.5):

$$\frac{dp}{dt} = v_{0+}(a_{measured} - \delta a)t$$

$$\int_{p_0}^{p} dp = \int_{t_0}^{t_1} v_0\, dt + \int_{t_0}^{t_1} (a_{measured} - \delta a)t\, dt$$

$$p = p_0 + v_0 t + \frac{1}{2}(a_{measured} - \delta a)t^2$$

Once we have velocity and position equations depending on acceleration given by inertial sensor and time of each sample it is possible to discretize those equations:

$$p_k = p_{k-1} + v_{k-1}T + \frac{1}{2}a_{measured_{k-1}}T^2 - \frac{1}{2}\delta a_{k-1}T^2 \quad \textbf{(4.6)}$$

$$v_k = v_{k-1} + a_{measured_{k-1}}T - \delta a_{k-1}T \quad \textbf{(4.7)}$$

Where, the subscript *k* denotes the actual information while *k-1* refers to the past information.

Using an inertial sensor capable of measuring the accelerations in the three different axis it is possible to evaluate velocity and position for each axes. And, placing the equations (4.6) and (4.7) in matrixial form:

$$\begin{pmatrix} P_x \\ P_y \\ P_z \\ V_x \\ V_y \\ V_z \\ \delta a_x \\ \delta a_y \\ \delta a_z \end{pmatrix}_k = \begin{pmatrix} 1 & 0 & 0 & T & 0 & 0 & \frac{T^2}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 & T & 0 & 0 & \frac{T^2}{2} & 0 \\ 0 & 0 & 1 & 0 & 0 & T & 0 & 0 & \frac{T^2}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & T & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \\ V_x \\ V_y \\ V_z \\ \delta a_x \\ \delta a_y \\ \delta a_z \end{pmatrix}_{k-1} + \begin{pmatrix} \frac{T^2}{2} & 0 & 0 \\ 0 & \frac{T^2}{2} & 0 \\ 0 & 0 & \frac{T^2}{2} \\ T & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & T \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}_k$$

Now, we have the equation (3.21) represented where the state transition matrix, A, is the 9x9 matrix and the control matrix, B, is the 9x3 matrix.

To project the error covariance ahead (Equation (3.22)) which is $\boldsymbol{P_k}^-$, we have from the state transition matrix $\boldsymbol{A}$ and its transpose $\boldsymbol{A^T}$. From the last iteration we have the previous error covariance $\boldsymbol{P_{k-1}}$ (in the first iteration we consider that $\boldsymbol{P_{k-1}}$ is a 9x9 identity matrix). The matrix Q is difficult to obtain. It requires a lot of testing and tuning and how much elements has the state vector much hard is this task. There are some methods as the one presented by Brown [Brown, 1997]:

Having the system's equations:

$$x_{k+1} = \phi_k \cdot x_k + w_k \tag{4.8}$$

$$\dot{x} = F \cdot x + G \cdot u \tag{4.9}$$

where, ϕ is the matrix that relates x_k to x_{k+1} in the absence of a forcing function u_k as shown in Equation (4.8).

According to Brown the matrix Q_k must satisfy the matrix differential equation:

$$Q_k(t, t_k) = F(t)Q_k(t, t_k) + Q_k(t, t_k)\, F^T(t) + G(t)WG^T(t)$$

with W representing the power spectral density matrix

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

First is constructed A,

$$A = \begin{bmatrix} -F & GWG^T \\ 0 & F^T \end{bmatrix} T$$

Second, it's used A to obtain B through e^A,

$$B = \begin{bmatrix} \dots & \phi^{-1}Q_k \\ 0 & \phi^T \end{bmatrix}$$

Finally, Q_k is obtained from the upper-right element of B

$$Q_k = \phi \cdot \phi^{-1}Q_k$$

After the Q_k matrix is obtained it's possible to compute the covariance error ahead, $P_k{}^-$ using the following equation:

$$P_k{}^- = AP_{k-1}A^T + Q$$

Assuming that there is Gaussian distributed noise in the measurement represented in Equation (4.10):

$$v_k \sim N(\hat{v}_k, R_k) \tag{4.10}$$

Where,

$$\hat{v}_k = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

So, the measurement noise covariance matrix, R_k, can be obtained by

$$R_k = E[(v_k - \hat{v}_k) \cdot (v_k - \hat{v}_k)^T]$$

$$= \begin{bmatrix} \sigma^2_{v_x,k} & \sigma_{v_y,k}\sigma_{v_x,k} & \sigma_{v_z,k}\sigma_{v_x,k} \\ \sigma_{v_x,k}\sigma_{v_y,k} & \sigma^2_{v_y,k} & \sigma_{v_z,k}\sigma_{v_y,k} \\ \sigma_{v_x,k}\sigma_{v_z,k} & \sigma_{v_y,k}\sigma_{v_z,k} & \sigma^2_{v_z,k} \end{bmatrix}$$

The Kalman gain matrix, $\boldsymbol{K_k}$ (Equation (3.23), is computed using the error covariance calculated previously, matrix $\boldsymbol{H}$ (which is used to relate the measurement read with the *a priori* system's state) and $\boldsymbol{R}$. Once that velocity read from digital encoders is the only measurement to correct the system the $\boldsymbol{H}$ matrix could be represented by

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

thus, it's possible to compute the *a posteriori* system state, $\hat{\boldsymbol{x}}_{\boldsymbol{k}}$ (Equation (4.11)), based on the *a priori* system state estimate obtained in Equation (3.21), Kalman gain (Equation (3.23)) and using the measurement $\boldsymbol{Z_k}$. The velocity acquired by the digital encoders is decomposed in the local three axes in order to have $\boldsymbol{Z_k} = [\mathrm{v_x} \quad \mathrm{v_y} \quad \mathrm{v_z}]^{\mathrm{T}}$. This velocity used as a reference will prevent that Kalman Filter generates large position errors because of noise present in data measured by the inertial sensor. In other hand, this reference measurement also is a source of errors. Any tire slippage or a non-ideal tire pressure can lead to a difference between real and measured velocity.

$$\hat{\boldsymbol{x}}_{\boldsymbol{k}} = \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} + \boldsymbol{K_k}(\boldsymbol{Z_k} - \boldsymbol{H}\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}) \qquad \textbf{(4.11)}$$

At last, according to the Kalman filter algorithm presented in 3.5.2, it's updated the error covariance through Equation (4.12):

$$\boldsymbol{P_k} = (\boldsymbol{I} - \boldsymbol{K_k H})\, \boldsymbol{P_k}^{-} \qquad \textbf{(4.12)}$$

Where $\boldsymbol{I}$ is a 9x9 identity matrix.

4.3 Player Integration

To implement a module capable of doing vehicle's position estimation was developed a new module in The Player. This module consists in an *abstract driver* which, as discussed in chapter 0, communicates with other drivers instead of communicate directly with hardware devices.

The use of existing interfaces defined in The Player and its message communication system makes the integration of data from different types of sensors, providing to the user real-time exchange data between modules.

The driver (CalcPosDriver) was developed in C++ and incorporates the double integration algorithm and the Kalman filter in order to achieve position estimation. It receives data from two other drivers: the motor controller driver (ControllerDriver) and ahrs (ImuDriver). From ImuDriver it receives packet's time of arrival, platform's x acceleration, platform's y acceleration, platform's z acceleration, roll, pitch and yaw. The driver ControllerDriver provides the encoder measured velocity of the robot.

Adapting the Figure 3.15 for the specific case we have the following figure, where a client can decide the motion of the robot requesting a motion with specific speed and turnrate through a function provided by Position2dProxy and can receive the result of position estimation from the Position3dProxy by requesting it.

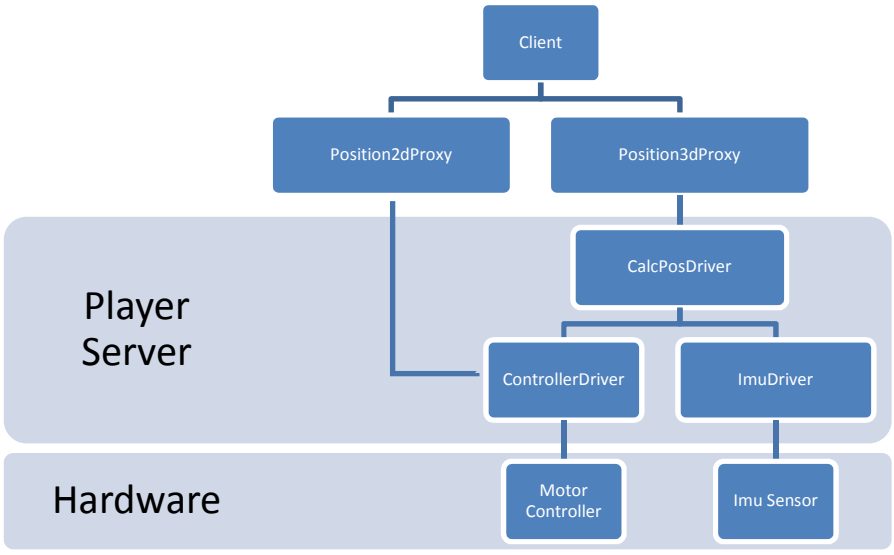


Figure 4.2 - Specific structure of Player for position estimation

The specific driver's layout is represented in Figure 4.3. The constructor retrieves the options from the configuration file and initializes the needed data. Then, there is the MainSetup() (omitted in the figure) that is responsible for subscribing any driver that this module needs. The main loop will be up till the end of driver's job. In the main loop we have process messages function call - *ProcessMessage()*, that deals with incoming messages. Either data from the other drivers (*PLAYER_MSGTYPE_DATA*) or requests (*PLAYER_MSGTYPE_REQ*). To both of the message types received a message PLAYER_MSGTYPE_RESP_ACK must be published to ensure the communication protocol.

Depending on this driver's purpose, the internal data management only starts when the CalcPosDriver has the data provided by ControllerDriver and ImuDriver. Once that the Controller driver takes longer to send its data, the *calcposition()* function starts when data from ControllerDriver arrive.

The function *calcposition()* handles the internal management data. This is, sub-functions of the double integration algorithm and Kalman filter are part of this function.

Terminated all computations needed and having the 3D position estimate it's time to publish the new data to others can have access. The message type is PLAYER_POSITION3D_DATA_STATE and it is posted with the function *publish()*.

In the end of a main loop, it must be given a sleep to the thread.

The destructor of the class must destruct all data used by the driver.

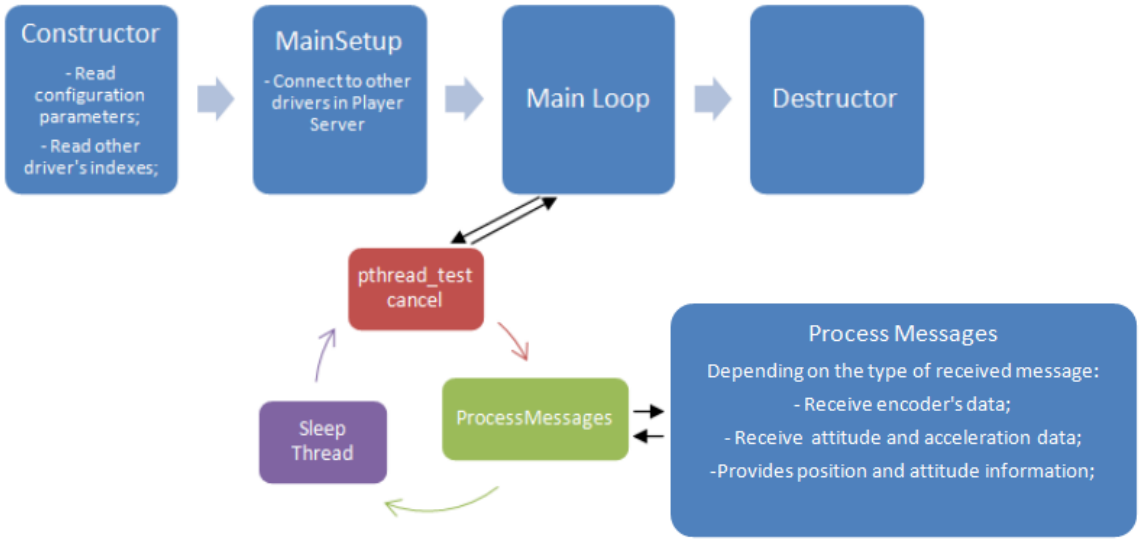


Figure 4.3 - Run-time process of CalcPosDriver, adapted from Gomes [Gomes, 2010]

5. Experimental Results

This chapter presents, in section 5.1, the equipment used to test the localization solutions presented in this work. In section 5.2 it's described and discussed the tests made and their results.

5.1 Equipment

The platform used to implement the localization system is the robot presented in Figure 5.1. The robot has being developed in Holos, SA and, in interest of localization problem, it uses an inertial sensor, a motor controller, two motors and two optical digital encoders (one per motor). Aside these components, it's used a mini-itx form factor computer (ZOTAC GerForce 9300 with a 3 GHz Intel Q9650 Core2Quad processor and 4 GB of RAM DDR2) to run the Player and a laptop (with a 2.2GHz Core2Duo processor and 2 GB of RAM DDR2) to run the client. Both running Linux OS, more precisely, distribution Ubuntu 9.10.

Figure 5.1 - Platform in which localization system is implemented

Inertial Sensor - XSens MTi

The inertial sensor used is the XSENS MTi [Xsens, 2008] which is an AHRS. It provides calibrated data from 3 axes such as accelerations, angular speeds, magnetic fields and temperature. This sensor also gives its orientation through quartenions, direct cosine matrix or Euler angles. It

has itself a Kalman filter for gyro drift compensation which could integrate magnetometers data in order to stabilize the yaw angle.

Figure 5.2 - XSens MTi inertial sensor

Depending on the MTi model the characteristics as communication interface, full scale acceleration and full scale rate of turn might be different.

The accuracies of the orientation provided by this AHRS, namely, yaw, roll and pitch are presented in the Table 8.

Attitude and Heading	
Static accuracy (yaw)[3]	< 1 deg
Static accuracy (roll/pitch)	< 0.5 deg
Dynamic accuracy[4]	2 deg RMS
Angular resolution[5]	0.05 deg
Dynamic range:	
• **Pitch**	± 90 deg
• **Roll/Heading**	± 180 deg

Table 8 - Attitude and heading accuracies

The interface used on this sensor is RS-232 and the full scales of acceleration and rate of turn are defined in Table 9. In the same table is given a detailed specification for each sensor present in the MTi where it is information such as full scale, bias stability and noise density, among others.

MTi's instrument cluster is composed by MEMS solid state accelerometer gyroscopes and accelerometers as well as thin film magnetoresistive magnetometers (Table 10).

[3] In homogeneous magnetic environment
[4] Under condition of a stabilized Xsens sensor fusion algorithm
[5] one standard deviation of zero-mean angular random walk

		Rate of turn	acceleration	Magnetic field	temperature
Unit		[deg/s]	[m/s^2]	[mGauss]	[°C]
Dimensions		3 axes	3 axes	3 axes	-
Full Scale	[units]	+/- 300	+/- 50	+/- 750	-15…+125
Linearity	[% of FS]	0.1	0.2	0.2	<1
Bias stability[6]	[units 1σ]	1	0.02	0.1	0.2[12]
Scale factor stability[7]	[% 1σ]	-	0.03	0.5	-
Noise density	[units/$\sqrt{\text{Hz}}$]	0.05[13]	0.002	0.5(1σ)[14]	-
Alignment error	[deg]	0.1	0.1	0.1	-
Bandwith	[HZ]	40	30	10	-
A/D resolution	[bits]	16	16	16	12

Table 9 - Individual sensor specifications

MTi Sensor Fact Table	
Accelerometers	MEMS[8] solid state, capacitive readout
Gyroscope	MEMS solid state, monolithic, beam structure, capacitive readout
Magnetometer	Thin film magnetoresistive

Table 10 – Sensor's type used in MTi

The Player driver that corresponds to *ImuDriver* explained in 4.3 is the *xsensmt* that was developed specifically to deal with MTi and MTi-g.

Motor Controller – Roboteq Ax3500

The controller has the function to power and control the motors. It controls the motors according with the orders received by RS232 or Radio. It receives the feedback from the encoders and, thus, it's possible to estimate the velocity of the platform as well as estimate the position using kinematics.

The controller used in this robot is a Roboteq AX3500 [Roboteq, 2007]. It is shown on the Figure 3.1.

[6] Non-considering environment
[7] Deviation over operating temperature range 1σ
[8] MEMS – micro-electro-mechanical systems

Figure 5.3 - the motor controller - Roboteq AX3500

In Player, was adapted an existing driver from another similar motor controller to this specific controller in order to assume communication issues and to provide upwards the needed information. The developed driver is called *h_roboteq* and corresponds to the *ControllerDriver* presented in section 4.3.

Optical Incremental Digital Encoders

Attach to the motor controller are two motors Maxon with 150W each one with an optical incremental encoder with 500 ppr (points per revolution). The motor and the encoder attached is shown in Figure 5.4. The information provided by encoders is given to Roboteq AX3500.

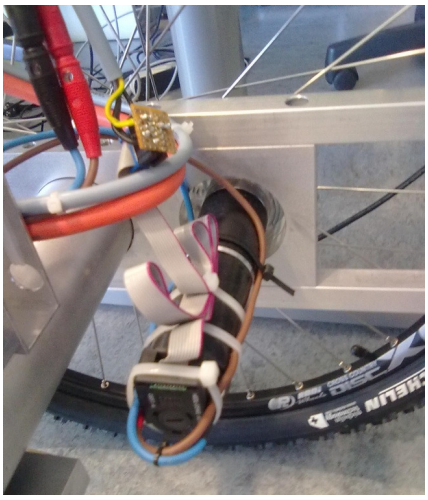


Figure 5.4 - The optical digital encoders used in the robot

5.2 Tests

In this dissertation are presented two ways of estimate position. The first approach presented was the double integrator, which uses only the data from low-cost AHRS as sensorial information. The second approach uses a Kalman filter with AHRS and odometric data.

The reason why the double integrator was developed consists on the fact that the AHRS could be used as stand-alone device to estimate position.

To see the performance and compute the efficiency of both approaches there were three kinds of tests:

1. Straight line tests under strong magnetic interferences;
2. Outdoor circular tests in a planar ground;
3. Outdoor circuit.

The first test is intended to study the magnetic consequences on the two approaches performances. It consists in indoor environments, under a metallic stairway, that inflicts strong magnetic fields. The test was repeated ten times in straight line at constant speed and covered 20 meters.

Circular tests have the purpose to check the system's behavior in circular trajectories in both ways in curves with yaw variation. It is intended to see how yaw is affected by the gyro bias and/or magnetic fields and how the trajectories are affected by yaw's precision.

The third type of tests is meant to determine the overall behavior of localization system under outside non-structured environments and, thus, compare with other similar methods explored in state of the art (Chapter 2). To perform the outdoor tests it was used the platform presented before and the circuit shown on Figure 5.5. The black dots are references to compare the true position with the estimated position. Their positions are referenced with the origin (red square) and are denoted by the local coordinate (x,y) in meters. Physically, these marks are 5cm x 5cm squares where the robot has to pass in order to have a controlled test.

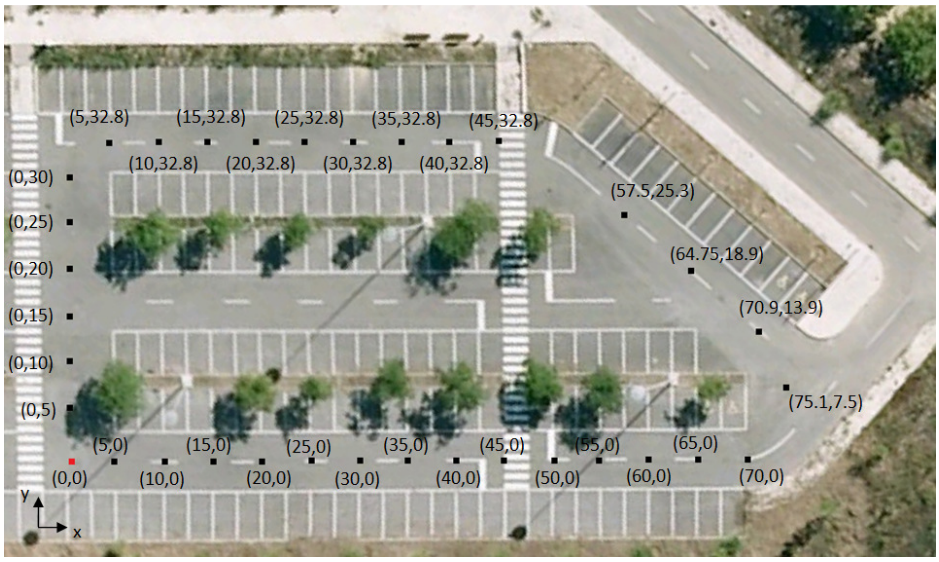


Figure 5.5 - Tests' circuit

5.2.1 Indoor - straight line

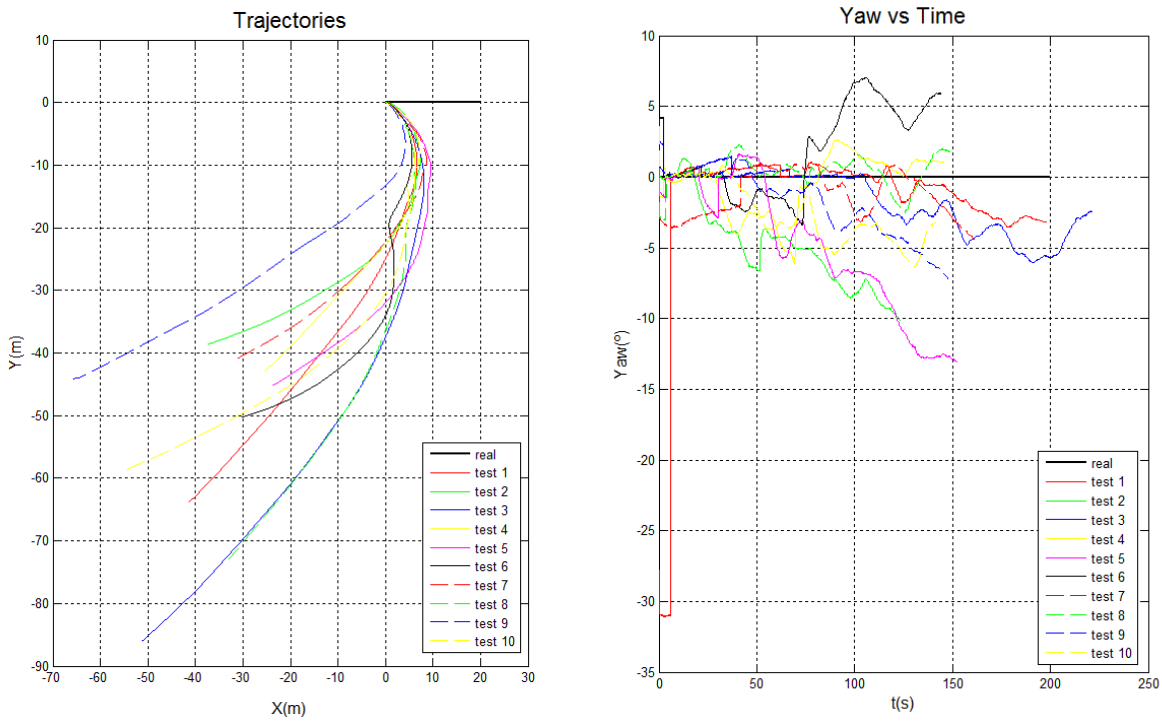
In order to check the estimated position given by the double integrator algorithm and the Kalman filter in indoor environments there were performed ten straight line tests at the same constant speed. There were made 5x5 cm marks on floor spaced by 1 meter in order to control the distance covered and to be possible to compare the estimated position with the real position.

It was followed the same procedure for all of the tests which include a waiting period before the test starts followed by a yaw angle reset. The goal of this part of procedure is to let the AHRS adapt to the magnetic field and stabilize the yaw angle. The reset was made to ensure that the robot is moving in x direction. During the test was marked the exactly moment that the robot passed through the 5x5 cm marks to allow a comparison between estimated and real position.

Double Integration Algorithm Results

.

The next figure shows the estimated trajectories obtained by a stand-alone method. The results of the double integrator application are shown in Figure 5.6 a) with the respective yaw variation in b).



(a) **(b)**

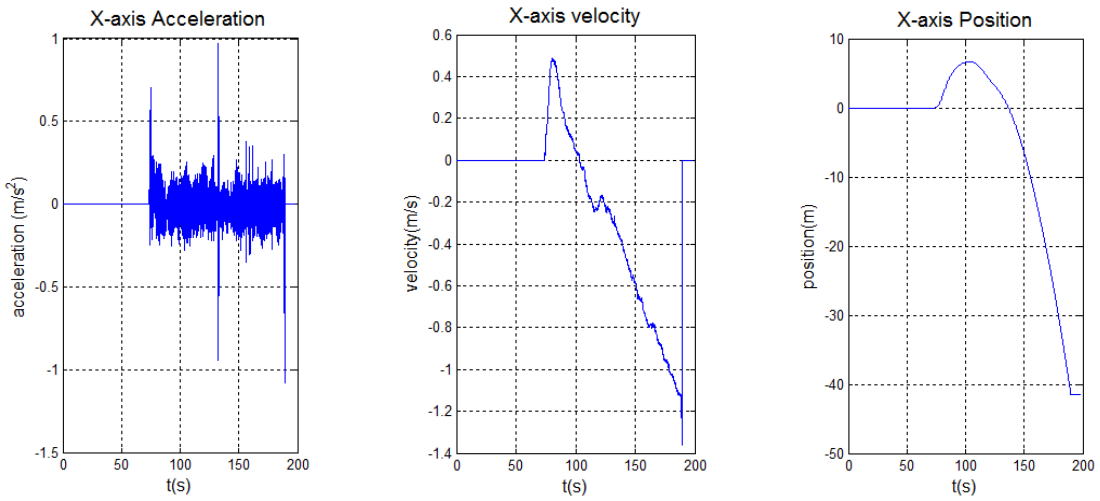
Figure 5.6 - 20 meters indoor with double integration algorithm: (a) the xy trajectory (b) the yaw variance over time

As it's possible to see in (a) all estimated trajectories and final positions are far away from the real. So it's possible to conclude that double integration algorithm using only this AHRS is unsuitable to have a reasonable position estimate.

Double Integration Algorithm Issues

To see exactly why a reasonable estimate couldn't be obtained using only this AHRS and a double integration algorithm let's analyze the Figure 5.7 (a). This figure presents the acceleration read from accelerometer in x axis, which, by integrating it it's obtain the velocity shown in (b). Position in (c) it's obtained by integrating the velocity. Although the double integration algorithm takes in account Coriolis acceleration and the local gravity vector the acceleration read isn't accurate enough to obtain an acceptable velocity. When comparing the velocity obtained by acceleration (Figure 5.7 (b)) with the one read by encoders (Figure 5.6) it is quite noticeable the difference. The encoder's velocity is much closed to the real speed and this is possible to affirm because of all calibration done to match these speeds (encoder's speed and real speed).

Thus, these tests suggest that acceleration of this AHRS isn't accurate enough to attain velocity using only the integration principle.



(a) **(b)** **(c)**

Figure 5.7- double integration algorithm behavior in 20 meters indoor test: (a) acceleration; (b) speed; (c) position

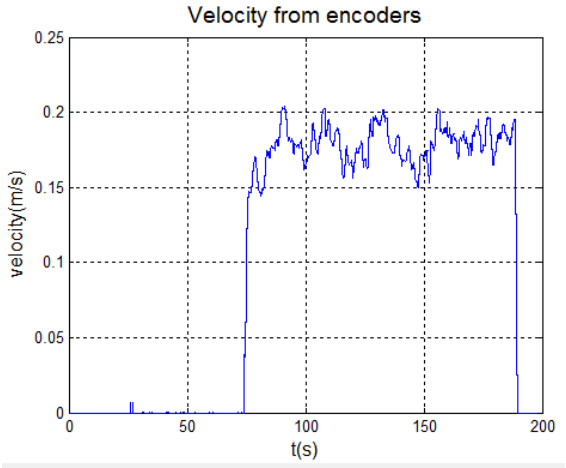


Figure 5.8 - Velocity read by encoders

Kalman Filter Results

The Figure 5.9 shows the XY trajectory for all of the tests and the respective yaw, roll and pitch variation. It is possible to determine that the robot followed approximately the same path in every tests once that roll and pitch variations are very similar when comparing with the different tests. Analyzing all the trajectories in Figure 5.9 (a) it is not possible to found a pattern in their deviation. Besides that, yaw doesn't seem to present a pattern either, but looking attentionally it is

quite noticeable that between 5 and 10 meters there are significant changes in yaw. This may be caused by the existence of metallic stairway which perturbs the magnetic field. Once that pitch and roll have similar behaviors for all tests it seems to be yaw variation the great responsible for xy trajectories.

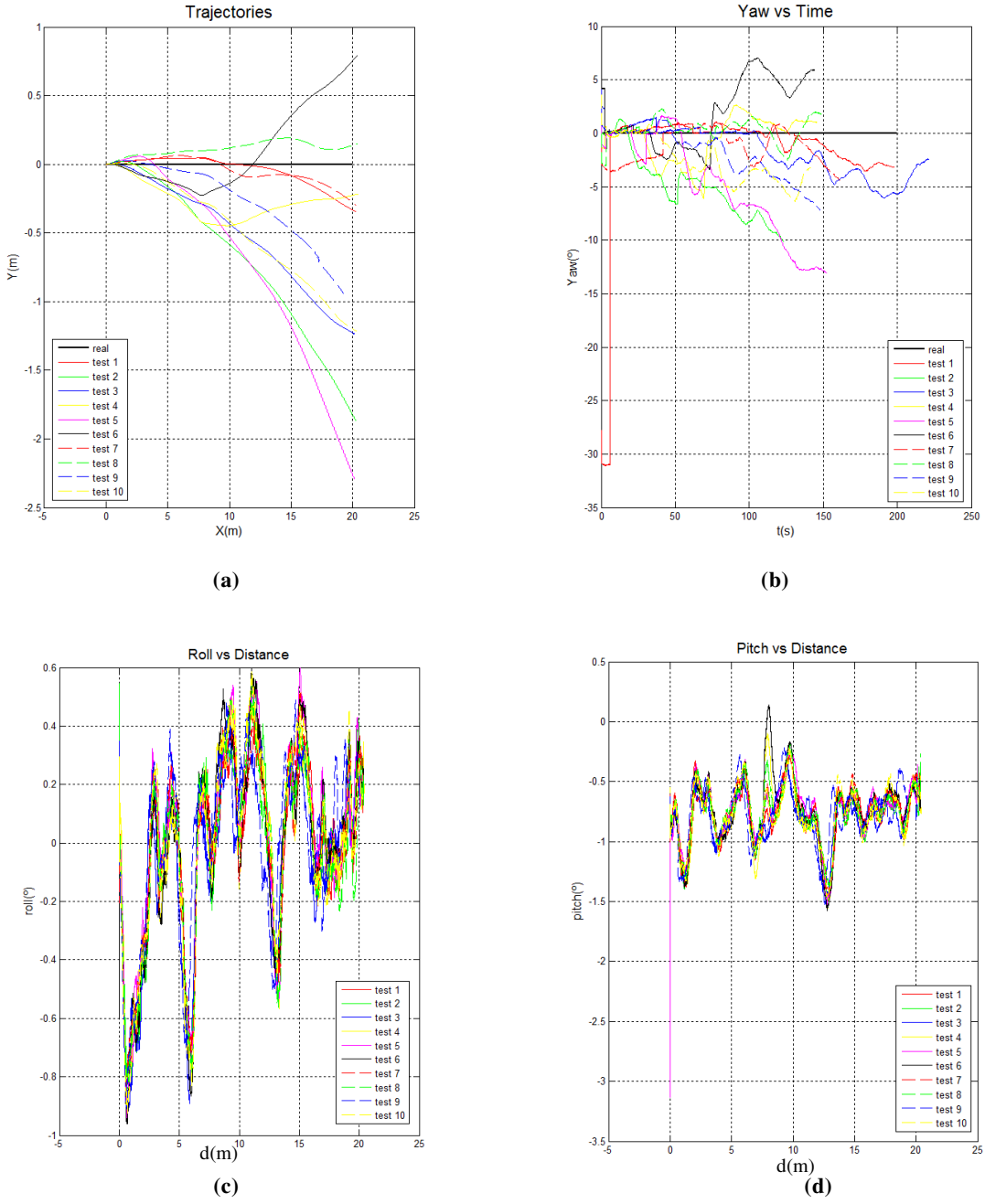


Figure 5.9 - 20 meters indoor with Kalman filter: (a) the xy trajectory; (b) the yaw variance over the distance (c) the roll variance over the distance; (d) the pitch variance over the distance

In this set of tests we have a minimum error of 1.9% and a maximum of 11.9% being the average error for the tests set 5.1%.

The Figure 5.10 shows for each test the true position versus the real position when passing in each mark. In the graphic's x axis it is represented the robot's true position as in y axis it is represented the estimated position. The gap between the beginning marks is 2 meters once that was given space for robot reaches the constant speed used.

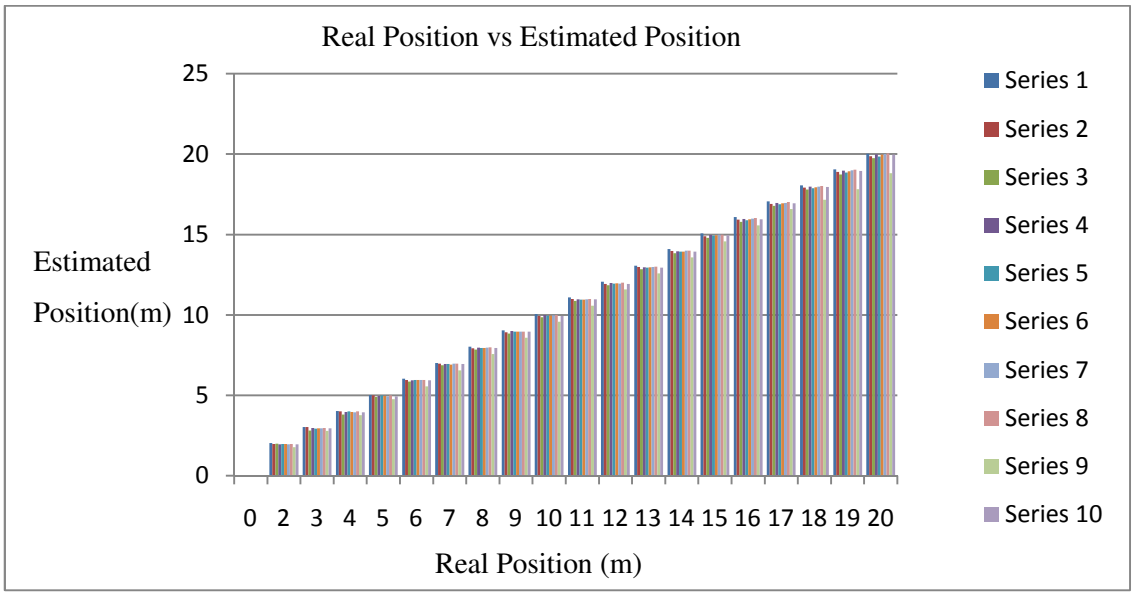


Figure 5.10 – One dimension's test – real distance versus estimated distance

The results show proximity between the estimated and real position. Yaw's deviation is responsible by the gap between estimated and true position of some series.

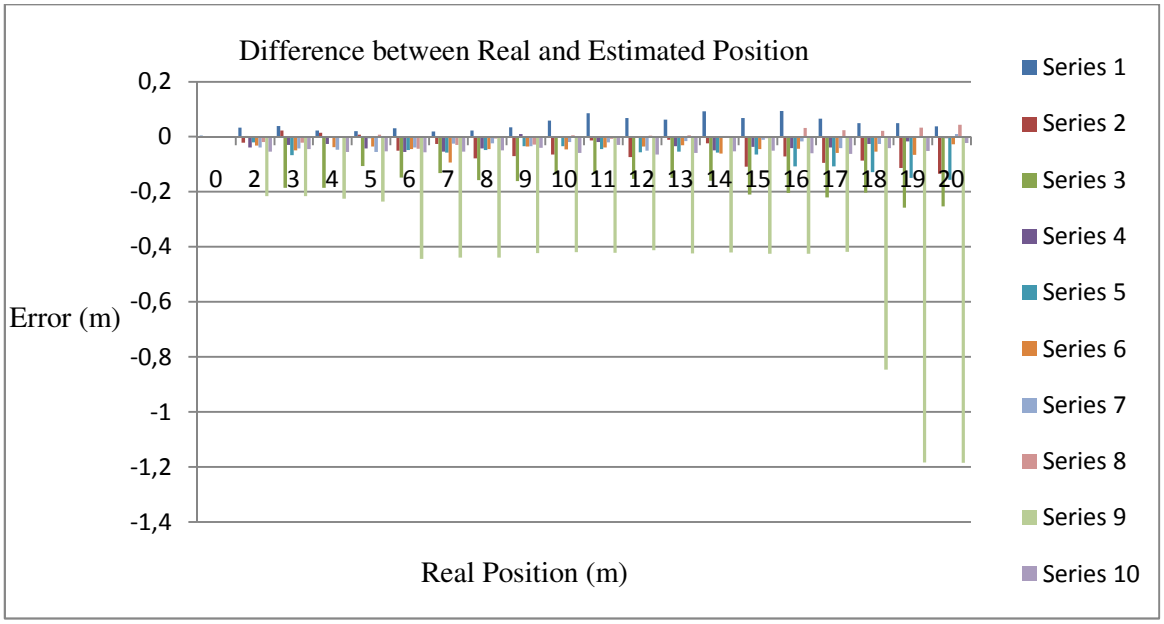


Figure 5.11 - Error amount over the distance

In Figure 5.11 it's shown the amount of error along the path for all of tests. In graphic's x axis it is represented the true position in meters and in y axis it is represented de difference between the true position and the estimated position.

The fact that error in the x axis is negative for most of the tests is explained by the deviation of yaw angle. Therefore, the distance should have been added only to x axis (once the test is realizes only in this direction) instead of being distributed by the two axes (x and y). Thus, it is obvious that the estimated distance (in x) is slightly lower than the true distance.

In straight trajectories is noticeable that the error will accumulate along time. New position estimation errors will always be added to the previous error.

5.2.2 Circular Trajectories

To see how the system deals with curvilinear trajectories have been made tests which the trajectory was a circumference. Outdoors, in a planar ground conditions, there were accomplished 10 tests in each ways. They all were executed at the same constant speed and fulfilled twice the same exactly circular trajectory.

The real trajectory should be a perfect circle with an approximately diameter of 3.5 meters and yaw must run from 0º to 180º and from -180º to 0º for a single turn.

In Figure 5.12 are shown the anti-clockwise tests and as it's possible to see in (b), (c) and (d) respectively trajectories' yaw, roll and pitch their variation over time is very alike. But looking to yaw (b)) it's noticeable that there is a decoupling between the different tests over time. Once that each tests performed two turns, this effect is very visible in yaw's first to second transition (from 180º to -180º). Because all the tests were done at the same constant speed, the yaw should vary at the same pace and this decoupling shouldn't exist.

Although the AHRS has an embedded Kalman filter to correct yaw's deviation caused by gyroscope bias it is, yet, noticeable. This will produce a slightly difference in yaw at same robot's position (in the two different rounds) and consequently between the first and second turns of estimated trajectories.

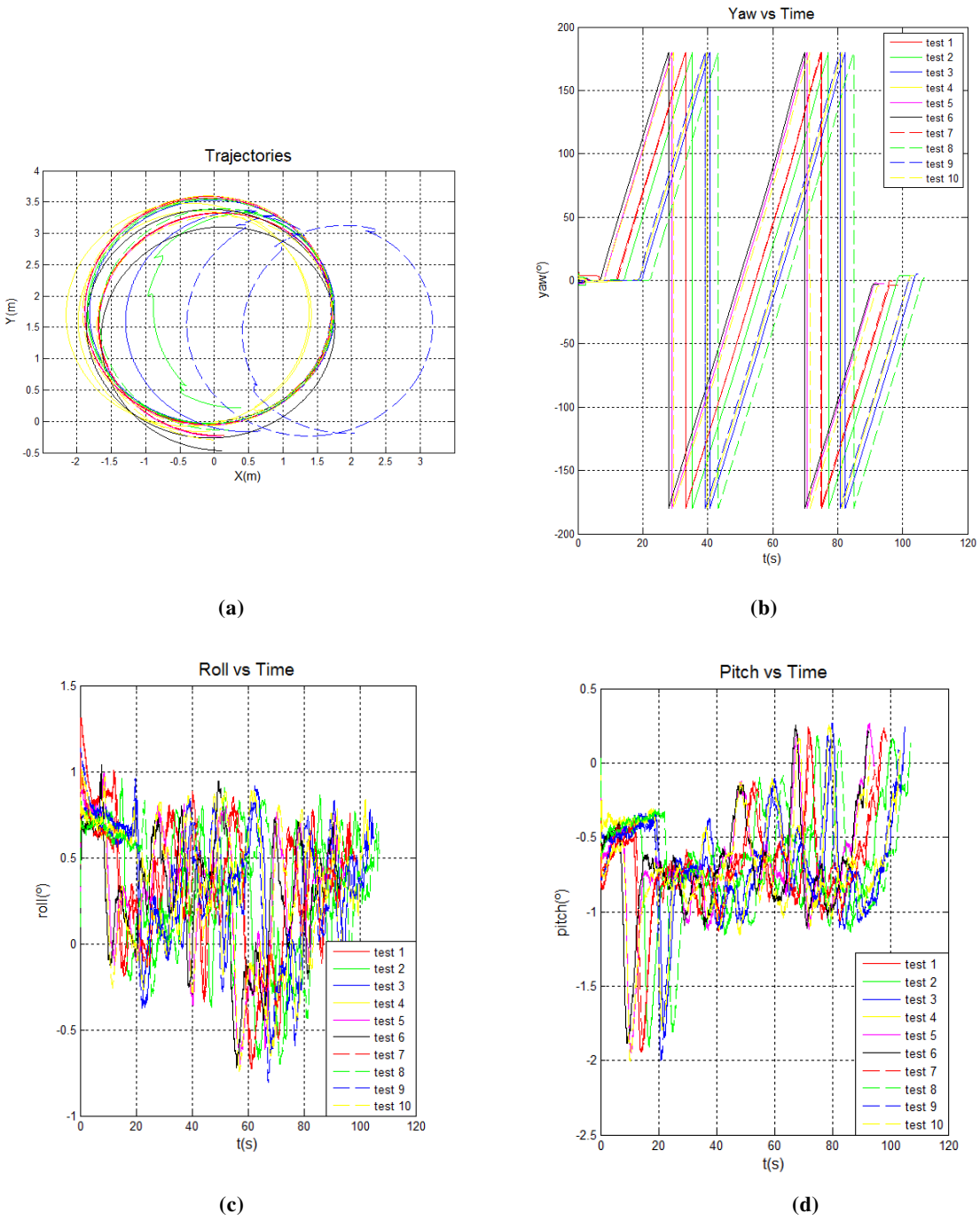


Figure 5.12 - anti-clockwise circular trajectory with Kalman filter: (a) the xy trajectory; (b) the yaw variance over the distance (c) the roll variance over the distance; (d) the pitch variance over the distance

In (a) it is possible to see that there are two traces that have a very different behave from all others. The green and blue dashed lines have some "jumps" which don't match with the real trajectory and have such important in the reconstruction of trajectory. Besides these two lines there are others with the same kind of errors but not so noticeable and preponderant in the trajectory.

In

Figure 5.13 shows that all the sudden changes in x position coincide with a velocity error. There are two obvious causes for the this errors in velocity: malfunctions on Player's messages that lead to a wrong velocity data transmission; due to the way that data is received from the quadrature encoders, it is only possible to compute forward velocity. If the wheel goes in the opposite way the velocity calculated in the *controllerDriver* don't match the real speed. Due the conditions of the tests (constant speed) and the sudden zeros the first option is more probable.

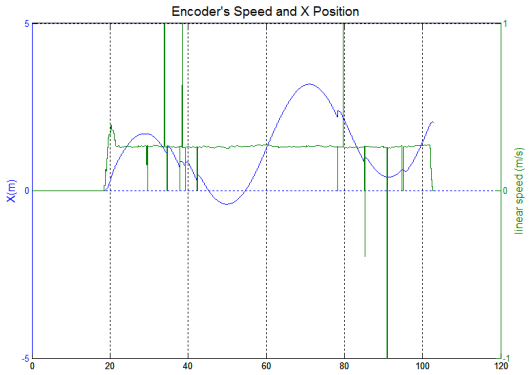
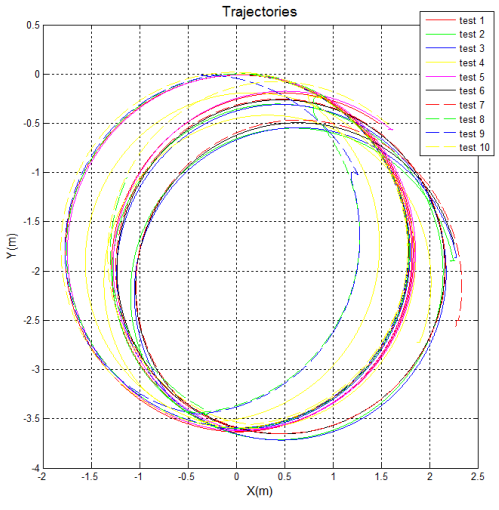


Figure 5.13 - X position versus the velocity received from motor controller driver

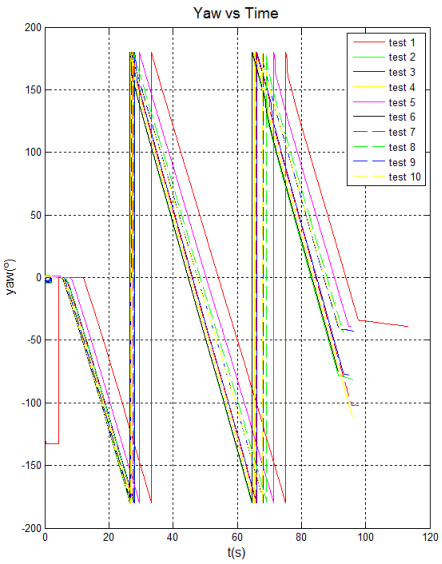
Other characteristic of these results is that in the first turn the estimated trajectory is very close to the real while in the second turn, with the accumulated error and deviation of yaw angle yaw, the estimated trajectory has a bigger difference from the real.

In relation to the Figure 5.14 and its clockwise circular tests the same considerations and conclusions can be taken. The system has the same behavior in both ways.

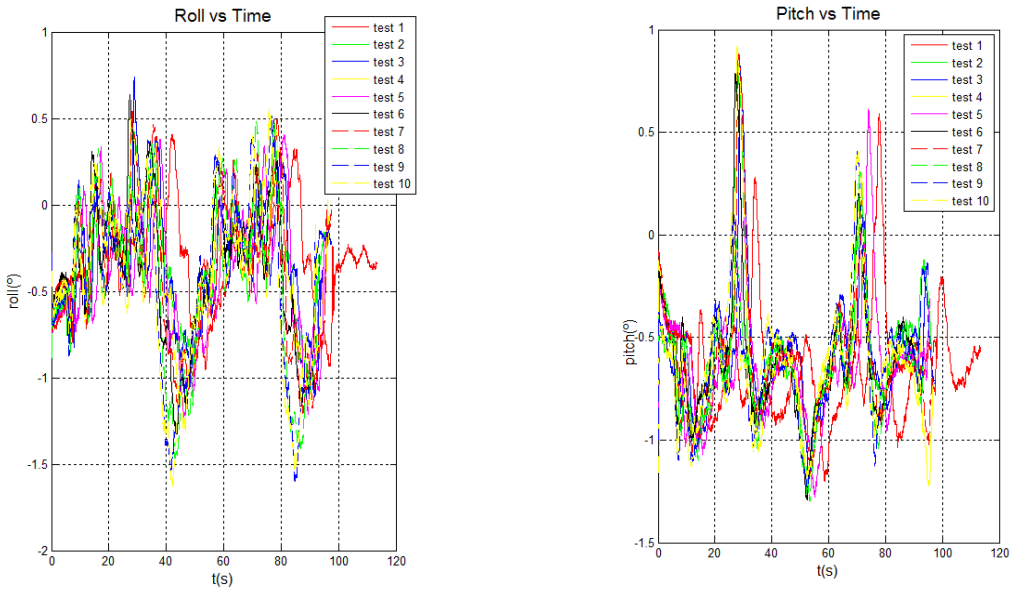
Similarly to the anti-clockwise tests, the clockwise real trajectory should be a perfect circle with an approximately diameter of 3.5 meters but yaw must run from 0º to -180º and from 180º to 0º for a single turn.



(a)



(b)



(c) **(d)**

Figure 5.14 - clockwise circular trajectory with Kalman filter: (a) the xy trajectory; (b) the yaw variance over the distance (c) the roll variance over the distance; (d) the pitch variance over the distance

5.2.3 Closed Circuit

Outdoor tests were done in the park shown in Figure 5.5. The principal goal of these tests is to study the behavior of the system in more general conditions than the previous tests. The circuit has 197 meters perimeter with some slight climbs and falls. The tests were done at constant speed and their average time was 11 minutes and 30 seconds.

Once more, we can see in Figure 5.15 (a) the estimated trajectories computed by the Kalman filter, in (b) the yaw variation over the time, in (c) and (d) the roll and pitch variation over time. Roll and Pitch angles variation are very similar which means that the real trajectories are approximately the same. Again, yaw angle seems to be a great influence in the final result as we can see by observing (a) and (b). Looking to the firsts 300 seconds in (b) it's quite noticeable that there are some lines which are slightly above the zero (as red, dashed blue and black representing the most visible) and others slightly below zero (as dashed yellow and green). When observing (b) it's clear that the ones below zero yaw angle (in (b)) have the tendency to deviate to the right. In other hand, the lines above zero yaw angle tend to go left when doing a straight forward movement. In this matter, it's notorious the influence of yaw in the error of first 70 meters.

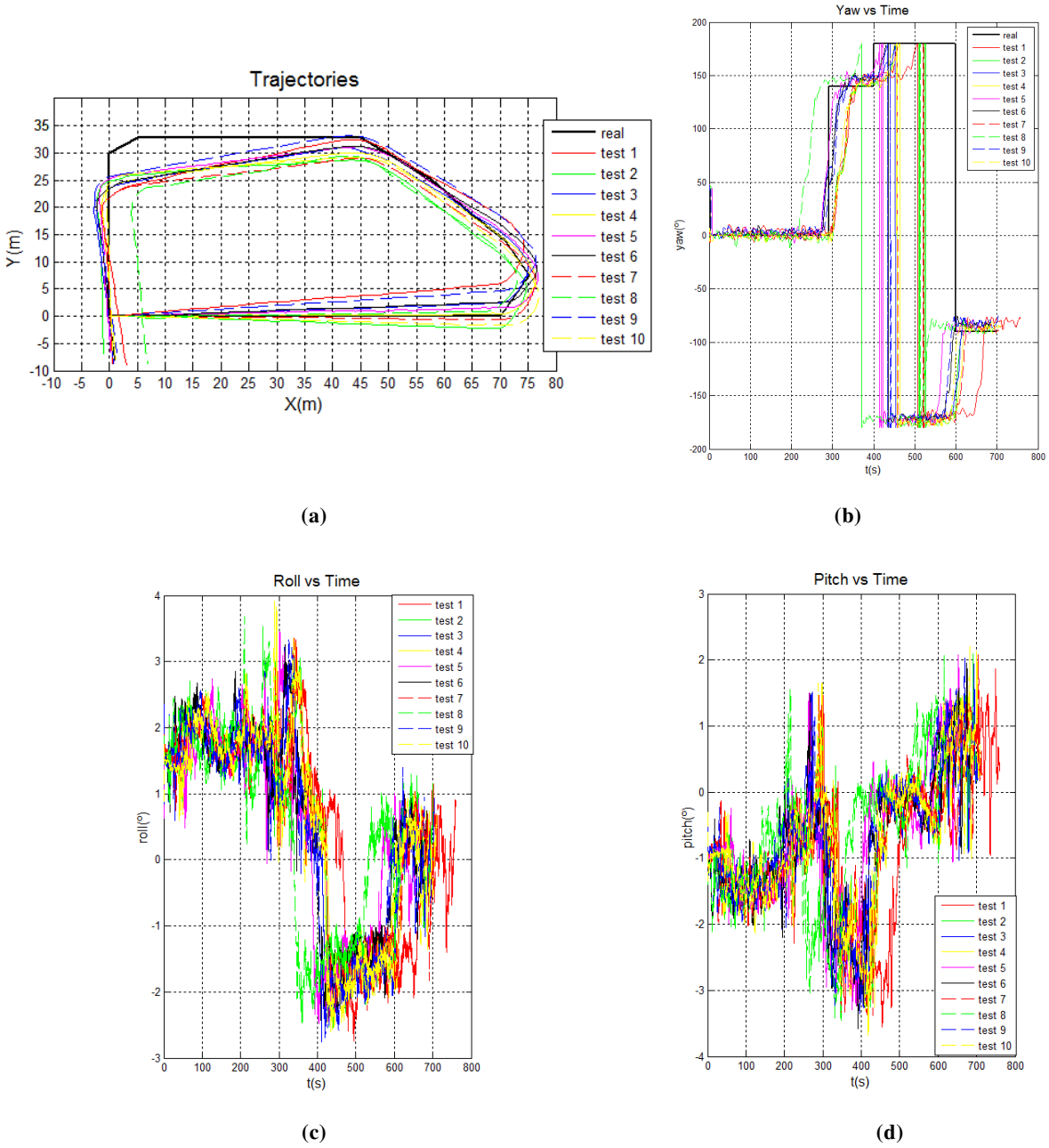


Figure 5.15 - closed circuit trajectory with Kalman filter: (a) the xy trajectory; (b) the yaw variance over the distance (c) the roll variance over the distance; (d) the pitch variance over the distance

In the ten tests roll and pitch have practically the same variation over the time as shown in (c) and (d). In Figure 5.16 it's presented the amount of the error along the circuit. The error is obtained by the difference between true position (in the marks of Figure 5.5) and estimated position at the same time. As visible the error increases along the distance covered which is normal when using dead-reckoning methods do estimate position. Around the 16th mark the error decreases and this phenomenon is explained by the circuit's characteristics and deviation of yaw angle: the fact of being a closed circuit makes that estimated trajectory cross the real route.

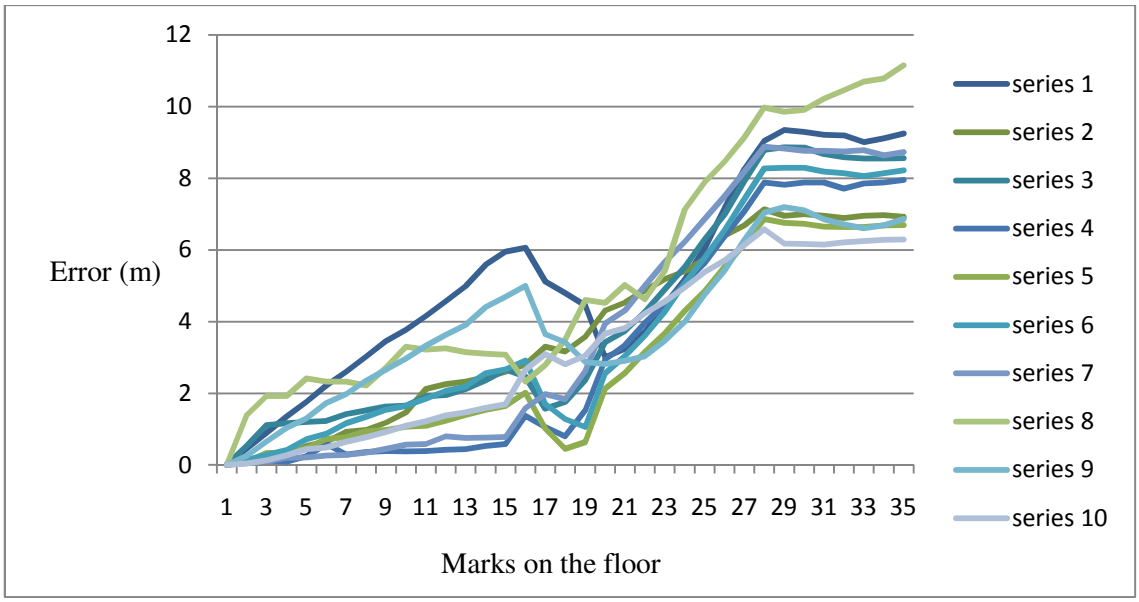


Figure 5.16 - Error amount along circuit's trajectories

Also is noticeable by the figure that, under these conditions, the final position error has a mean value of approximately 8 meters. This corresponds to a 4% of error after covering the 197 meters in 11,5 minutes. As it's possible to see Figure 5.16 the average error is growing over the time (increasing in each mark) excluding around the 17th mark (because of loop trajectories) ending with the maximum error attained.

Table 11 shows the principal characteristics of similar methods, cited in 2.3, that are inertial-based and are aided by odometry data. All the methods use closed loops to test its approaches. The different set of sensors and specific conditions in each method makes the comparison not so trivial and, thus, it's necessary to consider all the parameters.

	distance covered (m)	distance error (m)	% of error	Time (minutes)	Inertial Sensor
This method	197	8	4	11,5	XSENS MTI
[Agrawal and Konolige, 2006]	141,6	11,4	8,1	3,4	XSENS MTI
[Liu et al., 2005]	1100	8,36	0,76	3	DMARS-I IMU with DTG and a digital Honeywell compass
[Ippoliti et al., 2005]	108	0,161	0,15	Not known	Hitachi HOF-1 with FOG
[Santana, 2006]	2800	97,98	3,5	5	CROSSBOW IMU 3 FOG and 3 MEMS ACCELEROMETER

Table 11 – Comparison of position error of the various methods

Although the conditions (distance, velocity, circuit's nature, etc…) were not the same it's clear that an error of 4% is in the typical values of this kind of approaches.

The fact of that this method's tests have the biggest duration (remembering that dead-reckoning methods loose accuracy along the time) means that 4% of error is a good quality indicator for this method. This error could have been reduced if a higher velocity was used.

Agrawal and Konolige's method serves as a good comparison to this method. Mainly because it was used the same AHRS. When comparing this result with Agrawal and Konolige's method it was obtained a half percentage of error in a longer test (distance and time). Apart of this method, the other methods use more accurate inertial sensors equipped with DTG and FOG technology instead of MEMS. This is one more parameter that should be taken in consideration when comparing results. FOG and DTG gyroscopes are more expensive and more accurate than MEMS.

The results of Ippoliti's were obtained in an indoor environment in a planar ground using a Hitachi FOG which bias drift is ±0.05 deg/s [Hitachi HOF-1, 2010] while XSENS MTi gyroscopes have ±1 deg/s. Santana used a Crossbow VG700 IMU which gyroscopes have a ±0.03 deg/s maximum bias drift [Crossbow VGA700, 2010]. DMARS-I such as the one used by Liu et al. have a maximum bias drift of ± 1 deg/h (0.003 deg/s) [DMARS-I, 2010].

The major part of these methods, that serve as a comparison with this, are just intermediate methods before it were applied absolute measurements technology to bound the error.

6. Conclusions and Future Work

This chapter summarizes this dissertation, providing a set of conclusions about the localization system developed through its results. Possible improvements are also a theme that will be focused as future work.

6.1 Conclusions

This thesis presented two solutions for a robot localization using only relative measurements technologies: a double integration algorithm using the AHRS and an inertial-based Kalman filter.

Using the attitude and orientation data, the accelerations are translated from body-frame to earth-frame and, then, the accelerations were integrated once to obtain velocity and twice to attain position. Coriolis acceleration and local gravity were taken out during the process in order to have a more reliable acceleration and, thus, a better position estimation.

An inertial-based Kalman filter was applied using encoder's velocity as observations.

By studying the results obtained with this system it's possible to do some conclusions.

A low-cost IMU as the one used in this thesis is not suitable, only by itself, for localization. Thus, other sensorial information must be used to aid an inertial localization system using a low-cost IMU.

Using an AHRS, such as a XSENS MTi, has some advantages: to have attitude and orientation information which facilitates the body to earth coordinate transformation; its size, weight, low power consumption and a low price. As major disadvantages are the susceptibility to magnetic fields in orientation and its lack of accuracy when comparing with high-cost inertial sensors. Even so, it has a set of Kalman filter configurations that provide orientation for different scenarios.

As referred before, a strapdown inertial system is not suitable for location with XSENS MTi. As the results show, the velocity obtained in double integration algorithm does not represent the real velocity. This happens, mainly, because of the low accelerometer precision which, when is integrated, causes a wrong velocity and, then, a wrong position as well. When aided by encoder's data the precision of position estimation increases drastically. Principally, due to the speed included as measurement in Kalman filter.

Orientation data has a major influence on position error as demonstrated in all the three kinds of tests done. Particularly, in Section 5.2.1 – Indoor Straight Line tests where results show that yaw data suffers from magnetic perturbations.

The results revealed a 4% error in the final position which fits in the bounds of the other similar methods' error. The comparison between the methods presented is difficult because each one was tested under specific conditions. A considerable advantage of this test resides in the fact of being the one with the largest duration.

The fact that this method does not depend on absolute relative measurements allows it to work in indoor/outdoor environments. It doesn't have the need of using structured environments, hence the robot can be localized even in unknown environments. However, since there are only used dead-reckoning technologies, every estimation error from the beginning of its usage will affect the whole position estimation. Depending on robot's application it may be very harmful, especially in high- precision tasks.

6.2 Future work

Several measures can be applied some corrections and for a future improves of this location system:

- A more depth study should be done in process and measurement covariance matrixes for a better estimation of Kalman filter and, thus, a less error added along the time;

- Once that the main issue is yaw's precision, it should be study other ways to obtain orientation. Combining the vehicle's kinematics with the existing yaw is one possible way to achieve it. Another solution can be obtained by the inclusion of one high-precision gyroscope to integrate in Kalman filter. Landmarks also can be used for improving yaw's estimation. This can be obtained by using artificial landmarks or by exploring features collection using a camera;

- The sudden changes in circular trajectories are caused by errors in the velocity read. This may be caused by defective messaging on Player or by an error while reading the encoders. Filtering the velocity in order to take out its sudden changes and absurd values is one solution. Another solution could be achieved by detecting an eventual error when using Player's messages to data exchange;

- The inclusion of new sensorial information to add consistency to the estimation is an important part of future work. The inclusion of more technologies to aid the process will give more reliability. Either relative or absolute measurements can increase its

performance. However, the inclusion of an absolute measurement technology will bound the error allowing it to have a time-independent precision.

Bibliography

[Agrawal and Konolige, 2006] Agrawal, M. and Konolige, K. (2006). Real-time Localization in outdoor Environments using Stereo Vision and Inexpensive GPS. In *18th International Conference on Pattern Recognition*, pages 1063-1068.

[Ashkenazi et al., 2000] Ashkenazi, V., Park, D. and Dumville, M. (2000). Robot Positioning and Global Navigation Satellite System. *Industrial Robot*, 27: 419-426.

[Borenstein et al.,1997] Borenstein, J., Everett, H.R., Feng, L. and Wehe, D. (1997). Mobile Robot Positioning – Sensors and Techniques. *Journal of Robotic Systems, Special Issue on Mobile Robots*, 14(4): 231-249.

[Brown and Hwang, 1997] Brown, R. G. and Hwang, P. (1997) **Introduction to Random Signals and Applied Kalman Filtering**. New York: John Wiley & Sons, Inc., 2001. ISBN 0-471-12839-2. [Brown, 1997]

[Caceres et al., 2009] Caceres, M., Sottile, F., and Spirito, M. A. (2009). WLan-Based Real Time Vehicle Locating System. In *2009 IEEE Vehicular Technology Conference*, pages 2165-2169.

[Craig, 2005] Craig, John J. (2001). **Introduction To Robotics – Mechanics and Control**. New Jersey: Pearson Education, Inc., 2005. ISBN 0-13-123629-6.

[Crossbow VGA700, 2010] Crossbow VGA700, *Fiber Optic Vertical Gyro System*, consulted in 10/11/2010, http://saba.kntu.ac.ir/eecd/ecourses/instrumentation/projects/reports/Poly%20Gyroscope/Producers/Crossbow/vg/6020-0039-02_A_VG700AA.pdf.

[Dissanayake et al., 2001] Dissanayake, M.W.M.G., Newman, P., Clark, S., Durrant-Whyte, H.F. and Csorba, M. (2001). A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. IEEE Transactions on Robotics and Automation, 17(3): 229-241.

[DMARS-I, 2010] DMARS-I, *DigitalMiniature Attitude Reference System,* consulted in 10/11/2010, http://www.inertialscience.com/manuals/DMARS_I_broc.PDF.

[Gerkey et al.,2003] Gerkey, B., Vaughan R. and Howard A.(2003). The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor System. In *Proceedings of the International Conference on Advanced Robotic*, pages 317-323.

[Gomes, 2010] Gomes, P. LADAR Based Mapping and Obstacle Detection System for a Service Robot. Master's thesis. New University of Lisbon, Lisboa, Portugal, 2010.

[González et al., 2009] González, J., Blanco, J. L., Galindo, C., Ortiz-de-Galisteo, A., Fernández-Madrigal, J. A., Moreno, and F.A. Martínez, J. L. (2009). Mobile Robot Localization Based on Ultra-Wide-Band Ranging: A Particle Filter Approach. *Robotics and Autonomous Systems*, 57(5): 497-507.

[Grewal, 2001] Grewal, Mohinder S., Weil, Lawrence R., Andrews, Angus P.(2001). **Global Positioning Systems, Inertial Navigation, and Integration**. New York: John Wiley & Sons, Inc., 2001. ISBN 0-471-20071-9.

[Hightower et al., 2000] Hightower, J., Want, R. and Borrielo, G. "SpotON: An Indoor 3D Location Sensing Technology Based on RFID Signal Strength", Univ. Washington, Seattle, Technical Report UW CSE 2000-02-02, 2000.

[Hitachi HOF-1, 2010] Hitachi, *Fiber Optic Gyroscope*, consulted in 10/11/2010, http://www.hitachi-cable.co.jp/en/products/optical/gyro/gyro.html.

[IFR] International Federation of Robotics (IFR), *Service Robot*, consulted in 05/11/2009, *http://www.ifr.org/service-robots/*.

[Ippoliti et al., 2005] Ippoliti, G., Jetto, L. and Longhi, S. (2005). Localization of Mobile Robots: Development and Comparative Evaluation of Algorithms Based on Odometric and Inertial Sensors. *Journal of Robotics Systems*, 22(12): 725-735.

[Kalman, 1960] Kalman, Rudolf E. (1960). A New Aproach to Linear Filtering and Prediction Problems. *Transactions* ASME *Journal of Basic Engineering,* 82: 35-44.

[Leonard and Durrant-Whyte, 1991] Leonard, J., and Durrant-Whyte, H. (1991). Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3): 376-382.

[Liu et al., 2005] Liu, B., Adams, M. and Ibañez-Guzmán J.(2005). Multi-aided Inertial Navigation for Ground Vehicles in Outdoor Uneven Environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4703-4708.

[Liu et al., 2007] Liu H., Darabi H., Banerjee P. and Liu J. (2007). Survey of Wireless Indoor Positioning Techniques and Systems. *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, 37(6):1067-1080

[Loevsky and Shimshoni, 2010] Loevsky, I., and Shimshoni, I. (2010). Reliable And Efficient Landmark-Based Localization For Mobile Robots. *Robotics and Autonomous Systems*, 58(5): 520-528.

[Maybeck, 1979] Maybeck, Peter S.(1979). **Stochastic Models, Estimation and Control**. New York: Academic Press, Inc., 1979. ISBN 0-12-480701-1.

[Negenborn, 2003] Negenborn, R.R. Robot Localization and Kalman Filters. Master's thesis INF/SCR-0309. Utrecht University, Utrecht, The Netherlands, 2003.

[Ni et al., 2004] Ni, L.M., Liu, Y., Lau, Y.C. and Patil, P. (2004). LANDMARC: Indoor Location Sensing Using Active RFID. *Wireless Networks*, 10(6):701-710.

[Owen, 2010] Owen, J. (2010), How to Use Player/Stage, 2nd Edition, retrieved in April 2010, http://www-users.cs.york.ac.uk/~jowen/player/playerstage-tutorial-manual.pdf.

[Panzieri et al., 2002] Panzieri, S., Pascucci, F. and Ulivi, G. (2002). An Outdoor Navigation

System Using GPS and Inertial Platform. *IEEE-ASME Transactions on Mechatronics*, 7(2): 134-142.

[Player Manual, 2010] The Player Project, *The Player Robot Device Interface*, retrieved in May 2010, *http://playerstage.sourceforge.net/wiki/Main_Page*.

[PsuRobotics, 2010] Psu Robotics, *Player/Stage Drivers*, retrieved in May 2010, *http://www.psurobotics.org/wiki/index.php?title=Player/Stage_Drivers*.

[Roboteq, 2007] Roboteq (2007). AX3500 Motor Controller User's Manual. Roboteq, Inc.

[Santana, 2006] Santana, D. D. S. (2006). Estimação de Trajetórias Terrestres Utilizando Central Inercial e Fusão Sensorial. In *Proceedings of the 2006 IEEE INDUSCON*, Recife - PE.

[Titterton, 2004] Titterton, David H., and Weston, John L.(2004). **Strapdown Inertial Navigation Technology**. United Kingdom: The Institution of Electrical Engineers, 2004. ISBN 0-86341-358-7.

[Walchko et al., 2003] Walchko, K., Nechyba. C, Schwartz, E. and Arroyo, A. (2003). Embedded Low Cost Inertial Navigation System. In *Florida Conference on Recent Advances in Robotics*, Dania Beach – FL.

[Welch, 2006] Welch, G., and Bishop. G. (2006). An Introduction to the Kalman Filter, University of North Carolina at Chapel Hill.

[Xsens, 2008] Xsens (2008). MTi and MTx User Manual and Technical Documentation. Xsens Technologies B.V.