MASTER THESIS

TITLE: Hybrid navigation using sensor fusion in Android devices

MASTER DEGREE: Master in Science in Telecommunication Engineering & Management

AUTHOR: David Calero Scanlan

DIRECTOR: Eduard Angelats Company

SUPERVISOR: Francesc Tarrés Ruiz

DATE: November 4th 2013
Resum

Aquest projecte examina el rendiment que es pot obtenir en la navegació utilitzant la càmera, els sensors incorporats en un telèfon mòbil (i inclòs el GPS (Global Positioning System) quan estigui disponible). El projecte ha desenvolupat una aplicació en un sistema operatiu Android per a l’adquisició i sincronització de les dades dels sensors i també incorpora algoritmes de processat d’imatge per tal d’extreu observacions útils per a la navegació. Per altra banda, el projecte inclou la implementació d’un model per utilitzar les observacions de les imatges en un software extern de navegació.

Per una bona navegació cal fer una prèvia etapa de calibratge i caracterització dels sensors que incorpora el mòbil, tals com la càmera, els acceleròmetres, els giroscopis i els magnetòmetres.

La finalitat del projecte es determinar la trajectòria del telèfon mòbil, és a dir, el temps, la posició, la velocitat i l’actitud del smartphone.

Per a la validació i verificació del sistema, és necessari dur a terme una sèrie de proves. Aquestes proves consisteixen en l’adquisició de les dades i el processat d’aquestes, mitjançant una eina de navegació del Institut de Geomàtica. Els resultats de la navegació son comparats amb la trajectòria estimada utilitzant una plataforma de referència.

La determinació de la navegació del telèfon, tant en entorns d’exterior com en entorns d’interior, és una etapa clau per a aplicacions tals com guitatge al destí, aplicacions de realitat augmentada, videojocs, monitoratge de persones grans, monitoratge d’esportistes, monitoratge d’automòbils, generació de mapes.
Overview

This project examines the navigation performance that can be achieved by using the camera and the embedded sensors in a mobile phone (and including GPS when is available). The project has developed an Android application in order to acquire and synchronize the data of the sensors and also includes the development of image processing algorithms for extracting useful observations for navigation. In addition, the project also includes the implementation of a navigation model based on extracted images observations, for being used by an external navigation software.

In order to achieve an ideal navigation, it is necessary to carry out a previous stage of calibration and characterization of the embedded mobile sensors, such as the camera, the accelerometers, the gyroscopes and the magnetometers.

The aim of this project is to determine the mobile phone trajectory, is to say, the time, position, velocity and attitude of the smartphone.

To verify and validate the system a series of tests have to be performed. This consists of acquiring the sensors data and processing them using a navigation tool from the Institute of Geomatics. The obtained results are compared with the ones obtained using a precise reference navigation platform.

The smartphone navigation trajectory determination in indoor and outdoor environments, is a mandatory step in many applications such as guidance to a destination, augmented reality representation, gaming, elderly people monitoring, sports monitoring, automotive monitoring, map building, to mention a few.
ACKNOWLEDGEMENTS

I would like to thank Eduard Angelats Company, from Institute of Geomatics, for offering this interesting theme to work on and for the given support.

I am also thankful to Professor Francesc Tarrés Ruiz, from UPC, for his aid in the process of developing this project.

I would like to thank Maria Angela Scanlan, for revising the English document.

I would also like to appreciate Eulalia Parès, from Institute of Geomatics, for her dedication time to obtain navigation tests results.

Last but not the least, also a sincere thanks to those that directly or indirectly help me fulfilling this project.
ÍNDICE

CHAPTER 1. INTRODUCTION ........................................................................................................ 1
  1.1. Project scope .................................................................................................................. 1
  1.2. Goals .............................................................................................................................. 1
  1.3. Document overview ...................................................................................................... 2

CHAPTER 2. STATE OF THE ART ............................................................................................. 3
  2.1. Smartphone technology ............................................................................................... 3
  2.2. Sensors .......................................................................................................................... 3
  2.3. Navigation techniques ................................................................................................. 5

CHAPTER 3. PROPOSED APPROACH ......................................................................................... 9

CHAPTER 4. SYSTEM ARCHITECTURE ..................................................................................... 11
  4.1. HW platform ................................................................................................................ 11
    4.1.1. Smartphone ............................................................................................................. 11
    4.1.2. Sensors .................................................................................................................... 12
      4.1.2.1. Camera image sensor ......................................................................................... 12
      4.1.2.2. Accelerometer ................................................................................................... 13
      4.1.2.3. Gyroscope ......................................................................................................... 14
      4.1.2.4. Magnetometer .................................................................................................. 14
      4.1.2.5. GPS Receiver .................................................................................................... 15
  4.2. SW platform .................................................................................................................. 15
    4.2.1. Mobile Navigation ................................................................................................. 15
      4.2.1.1. Data logging and time synchronization ............................................................. 17
      4.2.1.2. Inertial and magnetic calibration ......................................................................... 19
      4.2.1.3. Geometric camera calibration .......................................................................... 20
      4.2.1.4. Image feature detection, description and matching .......................................... 21
    4.2.2. Navega ..................................................................................................................... 26

CHAPTER 5. ALGORITHMS ...................................................................................................... 29
  5.1. Magnetometer orientation estimation ........................................................................... 29
LIST OF FIGURES

Fig. 2.1 WiFi position estimation by triangulation. ........................................... 6
Fig. 2.2 Image feature detection and matching [28]. ........................................ 7
Fig. 3.1 SmartNav system chain. ........................................................................ 10
Fig. 4.1 Samsung Galaxy SII [38]. .................................................................... 11
Fig. 4.2 Galaxy SII internal components [39]. .................................................. 12
Fig. 4.3 Samsung Galaxy SII image sensor [40]. .............................................. 13
Fig. 4.4 Samsung Galaxy SII accelerometer [42]. .......................................... 13
Fig. 4.5 Samsung Galaxy SII gyroscope [43]. .................................................. 14
Fig. 4.6 Samsung Galaxy SII magnetometer [44]. .......................................... 15
Fig. 4.7 Samsung Galaxy SII GPS receiver [45]. ............................................. 15
Fig. 4.8 Android [49]. ...................................................................................... 16
Fig. 4.9 Mobile Navigation SW scheme. .......................................................... 17
Fig. 4.10 Left image: barrel distortion; Right image: fish-eye distortion [52]. .... 21
Fig. 4.11 Feature detection, description and matching scheme. ...................... 21
Fig. 4.12 Feature detector example. ................................................................. 22
Fig. 4.13 Edge detector example[55]. .............................................................. 22
Fig. 4.14 Corner detector example [56]. .......................................................... 23
Fig. 4.15 Blob detector example [57]. .............................................................. 23
Fig. 4.16 Feature matching example. ............................................................... 25
Fig. 4.17 SS approach trajectory determination algorithm. ......................... 27
Fig. 5.1 Heading, pitch and roll representation [71]. ...................................... 30
Fig. 5.2 Geodetic to ECEF coordinates representation [72]. ......................... 30
Fig. 5.3 Chessboard Pattern [53]. .................................................................. 33
Fig. 5.4 Coplanarity constrain with epipolar geometry [73]. ......................... 36
Fig. 6.1 Camera screenshots used for calibration. ........................................ 37
Fig. 6.2 Feature detector, extractor and matching test samples. .................... 37
Fig. 6.3 Route plan reference points. ............................................................... 39
Fig. 6.4 Left image: Trolley platform; Right image: Level arms. .................... 40
Fig. 6.5 Left: Radial component of the distortion in May. Right: Tangential component of the distortion in May ............... 41
Fig. 6.6 Left: Radial component of the distortion in October. Right: Tangential component of the distortion in October. . 42
Fig. 6.7 Left: Original image. Right: Undistorted image. ............................. 42
Fig. 6.8 Example of features detected in two consecutives images using ORB method. ................................................................. 43
Fig. 6.9 Two consecutive matched points. ....................................................... 43
Fig. 6.10 X-axes accelerometer data (m/s^2) during 6 hours (s). ...................... 44
Fig. 6.11 X-axes gyroscope data (rad/s) during 6 hours (s). ......................... 44
Fig. 6.12 Y-axes magnetometer data (deg) during 6 hours (s). ..................... 45
Fig. 6.13 Z-axis gyroscope data (rad/s) in terms of temperature (°C). ....... 45
Fig. 6.14 Gyroscopes (rad/s) temperature (°C) compensation. ................. 45
Fig. 6.15 Accelerometer temporal drift. ......................................................... 46
Fig. 6.16 Gyroscope temporal drift. ............................................................... 46
Fig. 6.17 Magnetometer temporal drift. .......................................................... 46
Fig. 6.18 Estimated position with GPS (red line) and INS/GPS (green line). ... 48
Fig. 6.19 GPS uncertainty (m) in time (s). ....................................................... 48
Fig. 6.20 INS/GPS attitude estimation. .......................................................... 48
Fig. 6.21 Outdoor test matched points. .......................................................... 49
Fig. 6.22 Indoor test matched points. ................................................................. 50
Fig. 6.23 Left: Reference and smartphone gyroscope data. Right: Reference
and smartphone accelerometer data. .............................................................. 50
Fig. 6.24 Indoor navigation positioning results. .................................................. 51
Fig. 6.25 Indoor navigation heading results. ...................................................... 51
Fig. 9.1 Android internal architecture............................................................... 59
Fig. 9.2 Mobile Navigation main menu ............................................................. 60
Fig. 9.3 Sensors characterization and calibration ................................................. 60
Fig. 9.4 Camera calibration ............................................................................. 61
Fig. 9.5 Acquisition options ............................................................................. 62
Fig. 9.6 Viewing sensors data and artificial horizon ............................................. 62
Fig. 9.7 Selecting image detector and descriptor ................................................ 62
Fig. 9.8 Previewing detected points .................................................................... 63
Fig. 9.9 Google Maps view of the performed route ............................................ 63
Fig. 9.10 Route statistics .................................................................................. 64
Fig. 9.11 Selecting and plotting sensors data graphics ........................................ 64
Fig. 9.12 Visualizing the navigation images with the position and orientation .. 65
Fig. 9.14 Allan Variance on gyroscopes data ..................................................... 67
Fig. 9.15 Allan Variance on accelerometers data .............................................. 67
LIST OF TABLES

Table 4.1 Smartphone main characteristics .......................................................... 11
Table 4.2 Lens main characteristics ........................................................................ 13
Table 4.3 Accelerometer main characteristics ......................................................... 14
Table 4.4 Gyroscope main characteristics ............................................................... 14
Table 4.5 Magnetometer main characteristics ......................................................... 15
Table 6.1 Feature detector, extractor and matching combination tests ............... 38
Table 6.2 May test camera matrix results (in pixels) ............................................. 40
Table 6.3 May test distortion coefficients results (in pixels)............................... 41
Table 6.4 October test camera matrix results (in pixels) ..................................... 41
Table 6.5 October test distortion coefficients results (in pixels) ......................... 41
Table 6.6 Feature detector, extractor and matching test results ......................... 43
Table 6.7 Sensors characterization and calibration test results ......................... 47
Table 6.8 Sensors characterization with Allan Variance results ......................... 47
CHAPTER 1. INTRODUCTION

1.1. Project scope

David Calero is the author of this final project for Telecommunications Engineering and Master degree at the Polytechnic University from Catalonia.

This project has been carried out via a collaboration contract with the Institute of Geomatics (IG) [1] and more specifically, in GIN¹ group. IG focuses on the promotion and development of Geomatics, through applied research, development and teaching. Some of the sciences and technology involved are the study, acquisition, storage, organization, analysis, dissemination, management and use of geographically-referenced spatial information. Geomatics disciplines include, among others, cartography, photogrammetry, remote sensing, sensor calibration and orientation, geodesy, topography, geographic information systems, civil engineering, deformation measurement and geomatic engineering, [2] and [3].

The main purpose of the project is to determine a navigation solution with the use of all embedded sensors in a handheld device using fusion techniques and to evaluate its performance.

There are multiple applications related to this project in indoor and/or outdoor environments, such as guidance to a destination, augmented reality applications, driving assistance, gaming, monitoring of the elderly, sports monitoring, motion recording, map building, automotive monitoring, to name a few.

1.2. Goals

The main goals of this project are:

- To design and implement a sensors data logger on an Android Operating System (OS).
- To time synchronize all the data of the embedded sensors.
- To characterize and calibrate the embedded positioning sensors.
- To geometrically calibrate the embedded camera.
- To study and implement image processing algorithms in order to extract suitable information for navigation.
- To implement a model for the image navigation.
- To process all the sensors data with a navigation software (SW), from the IG to obtain a unique trajectory.
- To verify and validate the system.

¹ Integrated Geodesy and Navigation Unit
1.3. Document overview

This document presents the design, implementation and validation of a smartphone [4] navigation system. The chapters are organized taking into account the different work blocks of the proposed system, which from now on will be referenced to as SmartNav².

Chapter 2 explores the state of the art of the current smartphone technology and its embedded sensors. It is also focused on presenting the current techniques and procedures for obtaining data from several sensors and fusing them for navigation.

Chapter 3 presents the main scheme of the SmartNav project, explaining the necessary steps to achieve a trajectory determination.

Chapter 4 defines the global system architecture, taking into account the hardware (HW) and SW platforms, explaining in detail the conceptual design of the SmartNav project.

Chapter 5 focuses on all the algorithms implemented in the SmartNav SW, such as the synchronization, characterization, calibration, image feature extraction and matching techniques and the image navigation model implemented.

Chapter 6 presents the tests performed during the project. The obtained results allow to verify and validate the SmartNav and evaluate its performance.

Chapter 7 summarizes the main conclusions of the project and offers an outlook for future developments.

² Smart Navigation
CHAPTER 2. STATE OF THE ART

2.1. Smartphone technology

Smartphone development is advancing at lightning speed. We are now at the point in the market where the features of Personal Computers (PC) were at in the late 90’s, but with better performance and new capabilities which are constantly evolving [5].

The mobile phone industry focuses on advancing in HW technology. All smartphones are based on a System-on-Chip (SoC) which is an integrated circuit that consists on a Central Processor Unit (CPU), memory blocks, Graphics Processing Unit (GPU) and other peripherals. The ARM mobile processor architecture [6] is the leading name in designing and licensing processors for mobile devices. At the moment, smartphones CPUs can achieve speeds up to 1.9GHz Quad-core (four simultaneously processors), and can also contain up to 1GB Random-Access Memory (RAM) [7].

Nowadays, modern mobile phones come with a variety of sensors [8] whose aim is to automate or facilitate many of our daily tasks. This field takes into account the presence of an accelerometer, a gyroscope, a compass, and a barometer. In addition, smartphones include a Global Navigation Satellites System (GNSS) receiver in order to determine their position. In indoor environments position can be estimated from Wireless Fidelity (WiFi) signals. Smartphones are also equipped with one or several small digital cameras. The image sensor quality is gradually increasing whereas the price of the smartphones is falling.

2.2. Sensors

The embedded sensors of a smartphone can be classified into inertial and non-inertial sensors:

Inertial sensors are measurement instruments that are based in motion sensing and rotation sensing. Basically, there are two types of inertial sensors: accelerometers and gyroscopes.

The accelerometers [9] measure linear acceleration in relation to one axis of an inertial frame of reference. Normally the integrated sensors are composed of three orthogonal accelerometers to receive information of linear acceleration in a 3 dimensional space. Accelerometers are based on Newton's second law “The acceleration of a body is parallel and directly proportional to the net force \( F \) and inversely proportional to the mass \( m \).”

\[
A = \frac{F}{m} \tag{2.1}
\]
Hybrid navigation using sensor fusion in Android devices

There are several types of accelerometers depending on their grade. High grade accelerometers are made from piezoelectric, piezoresistive and capacitive components such as piezoceramics or single crystals, such as quartz. Low grade accelerometers are the modern tendency making small sizes with the Micro Electro Mechanical System (MEMS) technology [10]. We have to take into account that all accelerometers on the earth’s surface are subject to the earth’s mean gravity acceleration value of $9.81 \text{ m/s}^2$.

The gyroscope [11] is an instrument that measures the angular increments from a reference orientation, so that a new orientation can be estimated by using mathematical equations. Like accelerometers sensors they are normally combined in a three orthogonal structure in order to receive information about rotation in three dimensional space. There are several types of electronic gyroscopes according to their technology. High quality gyroscopes like Fiber Optic Gyroscope (FOG) which use the interference of light to detect mechanical rotation. Mid grade gyroscopes such as Coriolis Vibratory Gyroscope (CVG) which use a resonator made with metallic alloys for measuring rotations using the Coriolis principle. The lower grade of gyroscopes is made of MEMS technology [12]. In the context of SmartNav, MEMS gyroscopes technology have been used, which are based on the Coriolis effect, that is to say, measuring the deflection force ($F_c$) of moving objects when they are rotating through a reference frame. MEMS gyroscopes contain a vibrating element used to sense the Coriolis effect. When the gyroscope coordinate system turns, Coriolis force induces a secondary vibration on the axis perpendicular to the measurement axis. This force depends on the mass ($m$) and on the cross product of the angular acceleration ($\Omega$) and the velocity of the particle in the rotating system ($v$).

$$F_c = -2m\Omega \times v \quad (2.2)$$

Nowadays MEMS gyroscopes cannot compete with precision of optical gyroscopes, nevertheless it is expected that technological progress will bring improvements in this area. For this reason it is of special importance to characterize the sensors in order to determine systematic errors and drifts, evaluating the sensors performance.

Non inertial sensors are instruments that do not measure motion of the object. The embedded sensors in smartphones are the magnetometer, the barometer, the image sensor and the GNSS receiver.

The magnetometer [13] is an instrument used to measure the strength and direction of the surrounding magnetic field. It is typically used for calculating the direction of an object by determining its angle with respect the north magnetic field of the earth. The principal disadvantage is that the magnetic field is variable and other magnetic elements can interfere with the sensor. In short, the magnetometer does not only measure the earth magnetic field it is also influenced by other external electronic devices that affect the surrounding magnetic field. In SmartNav, MEMS magnetometer technology sensors are
used. These are low cost sensors that are based on measuring the Lorentz-force or solid-state Hall Effect.

The barometer [14] is an instrument sensor used to measure atmospheric pressure. With a calibrated barometer the altitude of an object above the sea level can be estimated. The principal disadvantage of this instrument is the imprecision of the acquired data. The sensor depends on a reference measure of the sea level pressure and a correct determination of the surrounding temperature. In the context of the SmartNav, a barometer sensor has not been used because the smartphone selected did not have this feature.

The digital camera, concretely the image sensor is a device that converts an optical image into an electronic signal. The digital image sensors [15] can be classified into Charge-Coupled Device (CCD) or Complementary Metal–Oxide–Semiconductor (CMOS) active pixel sensors. Both are created from Metal–Oxide–Semiconductor (MOS) and are distributed in a matrix way. A CCD image sensor allows the analog to digital conversion to be done out of the image sensor chip, treating all pixels in the same way. This creates high quality sensors because reduces the noise from the images. A CMOS image sensor treats each pixel individually inside the image sensor chip, each pixel has its own analog to digital converter. CMOS sensors are easily to manufacture so are cheaper than CCD sensors. In terms of power consumption, CMOS image sensors require less power than CCD sensors. The image sensor used in the project is based on CMOS technology.

The GNSS receiver is composed of a GNSS antenna sensor that receives Radio Frequency (RF) signals from satellites and a processor that digitally processes the signals and obtains the messages from the different satellites. GNSS receivers determine the antenna position, velocity, and a precise time by processing the signals broadcasted by satellites. The GNSS receiver computes its distance to a set of satellites, by means of extracting the propagation time of the incoming signals traveling through space at the speed of light, according to the satellite and receiver local clocks. This time difference is transformed into a pseudorange measurement and by using triangulation techniques the current position is estimated. Because the satellites are constantly in motion, the receiver has to continuously acquire and track the signals from the satellites in view, in order to compute a continuous solution. High quality GNSS receivers can track several frequencies such as L1, L2 and L5 frequency band. In addition, the observations can be received in code pseudorange and in phase pseudorange. In the context of smartphones GNSS receivers, the GNSS signals received have a unique frequency channel L1 with a Coarse-Acquisition (C/A) code, used for the satellites tracking, achieving accuracies of 2-10m.

2.3. Navigation techniques

All the sensors mentioned above can be used for the trajectory estimation, that is to say, determining a time, position, velocity and attitude. Nowadays, several approaches are used to estimate a trajectory, but all of them are based on
measuring physical quantities that in some way are related with a geographical location or the change of it.

Dead reckoning [16] is a relative localization and orientation method that estimates the position and orientation by measuring motion and its direction from an initial position and orientation. For example, inertial navigation [17] is a dead reckoning technique that uses an Inertial Measurement Unit (IMU) [18] sensor that is composed of three orthogonal accelerometers plus three orthogonal gyroscopes. An Inertial Navigation System (INS) is an IMU sensor plus and a navigation SW. This SW computes orientation by integrating with respect time the angular rotation from the gyroscopes data. The position is computed by double integrating respect time the accelerometers data and by applying a rotation matrix, (change of direction) obtained from gyroscopes data. To improve performance, it requires accurate sensors in order not to accumulate drifts errors, and/or a good model implementation that can model the sensors behavior. At the beginning of the measurement, dead reckoning techniques require a fixed initial position and a fixed initial orientation.

Electromagnetic wave based techniques include visible and invisible light-based methods. Radio based techniques estimate position relative to known locations of radio emitters, such as satellites, wireless access points, Global System for Mobile Communications (GSM) base stations, etc. Regarding to GNSS techniques, the most prevalent solution today is the GPS, followed by GLONASS and GALILEO combined or alternative solutions. In the SmartNav project only GPS is referred to because of the smartphone compatibility. Satellite signals are easily obstructed by walls and have poor coverage in indoor areas. Regarding to radio methods, location is estimated by measuring the received signal strength from emitters using technologies like GSM, WiFi (Fig. 2.1), Blue-tooth, and ZigBee.

Remote Sensing [19] is basically remote perception, that is to say, a method that allows us to obtain information from an object or surface through the analysis of the received sensors data. It is based on the concept that each object or area issues energy in the electromagnetic spectrum, depending on the nature and the radiation that it receives. The reflected energy that it receives makes this object or surface distinguishable from another. Infrared, visible and thermal cameras, RAdio Detection And Ranging (RADAR), are example of remote sensors. Currently, there are several methods that are able to extract...
information from images and estimate position and orientation [20]. These techniques are known as optical navigation.

A first optical navigation approach for extracting useful information from an image, is marker detection [21]. This method consists of finding and scanning optical markers (such as barcodes, Quick Response (QR) codes, and ARTag [22] targets). A previous database with the markers identification and the geographical location is required.

A second optical processing method consists of feature detection [23] and matching (Fig. 2.2). This technique relies on computing the relative position and orientation of the camera from recognized features, such as interesting points, lines, circles, in a pair of input images. This method is computationally demanding however it can result in sub-meter localization precision.

An example of feature detection and matching is visual odometry [24]. This method can be done by combining two simultaneous cameras in a stereo combination [25] or by a sequence of two images from a single camera for the determination of navigation [26].

A second approach of feature detection can be done by detecting straight lines in images by applying a Hough transformation [27]. This method is useful for determining the camera orientation by determining planes on the image.

Optical based navigation needs a calibrated camera [28] for a robust navigation. This consists in determining the intrinsic parameters of the camera [29]. There is no standard method for calibrating Close-Range cameras, although the Conrady-Brown approach it is the most commonly used.

Hybrid navigation [30] or multisensor navigation is a navigation technique that consists of the fusion of several localization and orientation methods explained above in order to achieve more accurate navigation. All the navigation methods combined are used to obtain a unique solution using a Kalman Filter [31] or sequential least-squares algorithms.

Fig. 2.2 Image feature detection and matching [28].
The most common example of hybrid navigation in use today is INS/GNSS [32], which combines the inertial data and the GNSS data. There are several levels of integration modes: loosely coupled, tightly coupled and deeply coupled. The loosely integrated mode is the easiest and simplest approach because it is based on the independence of the GNSS and INS navigation functions. Tightly integrated mode is where a GNSS receiver provides pseudorange in real time for estimating position taking the IMU data in account. The deeply integrated mode is similar to the tightly coupled mode plus a closed loop-back of the estimated solution, in order to generate the new navigation data.

The combination of INS/GNSS/Camera [33] and [34] is another example of hybrid navigation, that combines the INS/GNSS with a position and attitude derived from image extracted features. The position and attitude is computed with a Least-Squares adjustment using the features coordinates. This position and attitude is used as an input for navigation correction in a Kalman Filter. The advantage of INS/GNSS/Camera is that a robust navigation solution can be achieved and that the camera can control the drift errors from the INS, without GPS, which is useful for indoor environments.

An example of a multisensor navigation with camera system using the feature detection is a Simultaneous Localization And Mapping (SLAM) [35] technique, used in autonomous vehicles to build a map in an unknown environment by determining its location simultaneously.

Google also implements multisensor navigation for building the street view application [36]. It uses an INS/GNSS plus a wheel encoder technique for the position and orientation determination and a system of nine directional cameras for 360° views at a height of 2.5m. The images are stitched making a panoramic georeferenced view. The street view system also includes a Laser Imaging Detection And Ranging (LIDAR) sensor for the 3D modeling.

Indoor navigation projects using the camera of the smartphone plus a positioning system already exist. An example of this approach has been done using the images for the smartphone attitude determination combining with several positioning systems is presented in [37].

Multiple combinations for hybrid navigations are usually variations of the INS/GNSS method by adding more sensors for a combined solution, such as a magnetometer, LIDAR, cameras, odometer, barometer, etc.
CHAPTER 3. PROPOSED APPROACH

The purpose of SmartNav is to design and implement the full chain of a hybrid navigation system in order to evaluate the smartphone performance in terms of navigation (Fig. 3.1).

The navigation approach is to develop a hybrid navigation technique based on sensor fusion. The main idea is to combine the INS, position, camera and magnetometer data to estimate the smartphone trajectory. The INS provides position, velocity and attitude. The position can be provided by GPS, WIFI or control point. The camera can generate relative position and attitude observations. The magnetometer provides attitude information.

The first step is to acquire data from all the available sensors of the smartphone and to store the information into disk. This step is realized using the Android API in the smartphone.

The second step is to build a calibration SW in order to detect and correct systematic errors on the sensor data. In terms of calibrating geometrically the camera, it has been selected the Conrady-Brown method, further details will be discussed in chapter 6. In relation to the integrated sensor misalignments, the bias and the noise components on the signals has been calculated. In addition, a temperature calibration has been performed in order to correct temperature drifts.

Some acquired data can be used directly as observations by the Navigation SW such as the IMU and the position data. Other data must be prepared such as the magnetometer data and images. For instance, from a pair of images, image points are extracted and matched by using image processing techniques.

In order to extract navigation observation from a pair of images, it is necessary to implement a mathematical model, named coplanarity. This pre-processing model allows relating pairs of matched points with the position and attitude of the smartphone. This model will be executed out of the smartphone in post processing mode.

In post processing, the sensor data has to be processed together in a navigation SW from the Institute of Geomatics to generate a navigation solution. This process will be executed on a Personal Computer (PC).

Finally, a navigation viewer platform needs to be built for the representation of the navigation trajectory.
In order to achieve all the required steps necessary to design and implement the whole system chain, open source APIs and library platforms have been selected.
CHAPTER 4. SYSTEM ARCHITECTURE

4.1. HW platform

4.1.1. Smartphone

The selected HW platform is a Samsung Galaxy S II mobile phone (GT-I9100) (Fig. 4.1). The smartphone main characteristics are presented in the following table.

Table 4.1 Smartphone main characteristics

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>1.2 GHz dual core</td>
</tr>
<tr>
<td>CPU architecture</td>
<td>ARM Cortex-A9</td>
</tr>
<tr>
<td>GPU</td>
<td>Mali-400 MP</td>
</tr>
<tr>
<td>RAM</td>
<td>1 GB</td>
</tr>
<tr>
<td>Internal memory</td>
<td>16 GB</td>
</tr>
<tr>
<td>External memory</td>
<td>64 GB</td>
</tr>
<tr>
<td>Weight</td>
<td>116 g</td>
</tr>
<tr>
<td>Dimensions</td>
<td>125.3×66.1×8.5 mm</td>
</tr>
</tbody>
</table>

In addition, the Samsung Galaxy SII display feature is a Super AMOLED Plus capacitive touch-screen with 16M colors. It is as wide as 4.3 inches with 480 x 800 pixels.

The Operating system is Android 4.0.4 (Ice Cream Sandwich).

Fig. 4.1 Samsung Galaxy SII [38].
4.1.2. Sensors

The embedded sensors used in SmartNav are the camera image sensor, the accelerometers, the gyroscopes, the magnetometer and the GPS receiver.

4.1.2.1. Camera image sensor

The Samsung Galaxy SII has two cameras, the front camera and the rear camera. The front camera is a 2 mega pixels (Mpix) camera, while the rear camera is an 8 Mpix camera. This project focuses only on the rear camera which is the one used for SmartNav.

The camera model name is IMX105 (Fig. 4.3) from Sony Manufacturer, and it has an 8Mpix quality, with a maximum size of (3264x2448) pixels. Its features are those which support autofocus and a LED flash. The image sensor is based on BSI-CMOS technology with a matrix dimension of (4.54x3.42) mm, is to say, a 1.4µm backside illuminated pixels.
In video mode, it is capable of recording High Definition (HD) video 1080p up to 30 frames per seconds.

The lens model IU105F2 is the one embedded in the Samsung Galaxy SII device. The main characteristics are the following:

**Table 4.2** Lens main characteristics

<table>
<thead>
<tr>
<th>Focal Length (35mm equivalent [41])</th>
<th>4 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zoom Ratio:</td>
<td>1.00x</td>
</tr>
<tr>
<td>Digital Zoom:</td>
<td>Yes</td>
</tr>
<tr>
<td>Aperture Range:</td>
<td>f/2.65 (fixed)</td>
</tr>
<tr>
<td>Dimensions:</td>
<td>8.5×8.5×5.67 mm</td>
</tr>
</tbody>
</table>

4.1.2.2. Accelerometer

The accelerometer that is embedded in the Samsung Galaxy SII is an ultra low-power three axes linear accelerometer from STMicroelectronics company, the model name is LIS3DH (Fig. 4.4).

The accelerometer is a MEMS technology category, a relatively inexpensive and small sensor. It includes an embedded temperature sensor to calibrate the sensor drifts in terms of temperature. The main characteristics of the accelerometer are:
Table 4.3 Accelerometer main characteristics

<table>
<thead>
<tr>
<th>Frequency sampling:</th>
<th>100 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range:</td>
<td>±16 g</td>
</tr>
<tr>
<td>Sensitivity:</td>
<td>12 mg/digit</td>
</tr>
<tr>
<td>Consumption:</td>
<td>0.03 mW</td>
</tr>
<tr>
<td>Dimensions:</td>
<td>3×3×1 mm</td>
</tr>
</tbody>
</table>

4.1.2.3. Gyroscope

The Samsung Galaxy SII gyroscope is a low-power three-axis angular rate sensor from STMicroelectronics, the model name is L3G4200D (Fig. 4.5). This gyroscope includes a clock source.

![Fig. 4.5 Samsung Galaxy SII gyroscope](image)

This gyroscope is also a MEMS technology category, relative inexpensive and small. The main characteristics of the embedded gyroscope are:

Table 4.4 Gyroscope main characteristics

<table>
<thead>
<tr>
<th>Frequency sampling:</th>
<th>100 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range:</td>
<td>±2000 deg/s</td>
</tr>
<tr>
<td>Sensitivity:</td>
<td>70 mdeg/s/digit</td>
</tr>
<tr>
<td>Non linearity:</td>
<td>4 deg/s</td>
</tr>
<tr>
<td>Consumption:</td>
<td>18.3 mW</td>
</tr>
<tr>
<td>Dimensions:</td>
<td>4×4×1.1 mm</td>
</tr>
</tbody>
</table>

4.1.2.4. Magnetometer

The embedded magnetometer in the Samsung Galaxy SII is a 3-axis electronic magnetometer integrated circuit with high sensitive Hall sensor technology; model AK8975 from AsahiKASEI Company (Fig. 4.6). It incorporates magnetic sensors for detecting terrestrial magnetism in the X-axis, Y-axis, and Z-axis, a sensor driving circuit, signal amplifier chain, and an arithmetic circuit for processing the signal from each sensor.
The AK8975 magnetometer, like the accelerometer and the gyroscope is a MEMS technology sensor. The main characteristics of the embedded magnetometer are:

Table 4.5 Magnetometer main characteristics

<table>
<thead>
<tr>
<th>Frequency sampling:</th>
<th>100 Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range:</td>
<td>+1200 yT</td>
</tr>
<tr>
<td>Sensitivity:</td>
<td>0.3 yT/LSB</td>
</tr>
<tr>
<td>Consumption:</td>
<td>3 mW</td>
</tr>
<tr>
<td>Dimensions:</td>
<td>4×4×0.75 mm</td>
</tr>
</tbody>
</table>

4.1.2.5. GPS Receiver

The embedded GPS receiver is a SiRFstarIV model GSD4t (Fig. 4.7). It is of relatively small size, 0.4 mm design that occupies less than 20 mm².

Some relevant specifications of the GPS receiver are the power consumption – 8 mW and the 160 dBm tracking sensitivity. It tracks a unique frequency L1, C/A code. In addition, SiRFstarIV detects, tracks and blocks up to 8 separate sources of interference in the GPS frequency band.

4.2. SW platform

4.2.1. Mobile Navigation

The Mobile Navigation SW is the platform that has been installed inside the smartphone. It has been developed in an Android OS, cross-compiled with a host computer with a Windows 7 OS. The NVIDIA Tegra Android Developer
Pack 2 [46] has been installed in the host computer. This pack is a developer tool configured for Android devices that includes an Eclipse development programming environment, Android Applications Programming Interfaces (API), Android SDK, Android NDK, and additional libraries such as OpenCV library [47]. The developer tool installs and configures the environment automatically.

Android (Fig. 4.8) is a Linux-based OS designed primarily for touch-screen portable devices such as smartphones and tablet computers [48].

Android is an open source OS and Google releases the code. This open source code and permissive licensing allows the SW to be freely modified and distributed by device manufacturers and developers.

Android consists of a kernel based on Linux kernel version 2.6 and, from Android 4.0 Ice Cream Sandwich onwards, the Linux kernel version is 3.0.15. It includes a middleware, libraries and APIs written in C programming language, and the application SW running on an application framework which includes Java-compatible. The main HW platform for Android is the ARM architecture, but there is also support for x86 architectures.

The Android SW Development Kit (SDK) includes a comprehensive set of development tools for programming in Android. These include a debugger, libraries, a handset emulator, documentation, sample code, and tutorials. It must be emphasized that the SDK is a Java programming language.

The Android Native Development Kit (NDK) includes libraries written in C programming language and can be compiled using GNU Compiler Collection (GCC) and installed into an ARM architecture using the native code. Writing code in NDK is more complicated but can be useful in order to accelerate the processor and also for using C or C++ programming language libraries.

Additional information about Android architecture is explained in the Annexes.

The Mobile Navigation SW is the platform in charge of calibrating the sensors, acquiring, recording, pre-processing and sending the sensor data and visualizing all the recorded navigation statistics. The mobile navigation main scheme is presented on (Fig. 4.9). The Mobile Navigation SW is divided into 3 different modules:
1. Sensors characterization and calibration.
2. Sensors acquisition (acquiring, recording, preprocessing and communicating via TCP/IP protocol).
3. Visualization (map, statistics, plots and navigation preview).

![Mobile Navigation SW scheme](image)

Fig. 4.9 Mobile Navigation SW scheme

An extension of this section is explained in the annexes, as a user manual of the Mobile Navigation SW.

### 4.2.1.1. Data logging and time synchronization

In order to access to the accelerometers, gyroscopes and magnetometer data in the mobile navigation SW, it is necessary to import the Android library. The Android library is implemented in Java. It is declared a sensor manager class which can manage the internal sensors from the smartphone.

The sensors are initialized to work in different threads at the highest priority, giving a sample rate of approximately 100Hz. The reason of using threads is to simultaneously acquire all sensor data and to achieve a better performance.

All sensor data received is time stamped. This timestamp is related to the time when the measure is generated. The acquired data is synchronized with the internal processor clock of the smartphone. It is very important to timestamp the sensors data because all the data needs to be time sorted in order to perform the navigation processing.
The IMU data is generated by combining the gyroscope measurements and the accelerometer measurements. The timestamp related to the IMU data is the mean of the accelerometer timestamp and the gyroscope timestamp. The IMU data format is the following:

\[ \text{Timestamp, Gyr}_x, \text{Gyr}_y, \text{Gyr}_z, \text{Acc}_x, \text{Acc}_y, \text{Acc}_z \]

For generating the attitude data it is necessary to combine the accelerometer data and the magnetic field data. This algorithm is explained in further details in the algorithms section. The timestamp related to the attitude data is the mean of the accelerometers timestamp and the magnetometer timestamp.

The attitude data resulting has the following format:

\[ \text{Timestamp, Heading, Pitch, Roll} \]

In order to obtain GPS information in android, it is necessary to register a GPS location manager, from Android library and to specify the GPS localization provider when initializing.

Subsequently a thread for the GPS acquisition is created, which detects the reception of the first location data. After the first data received, the thread will only receive new GPS data if the device has a new location. If no data is received, it is assumed that the smartphone is static. If there is always a new location, because the smartphone is in motion, then the GPS data is updated at a maximum rate of 1 Hz.

The received GPS data is expressed in geodetic coordinates and it is necessary to transform to Earth-Centered Earth-Fixed (ECEF) coordinates. This transformation is explained in detail in the algorithms section.

The GPS data includes an uncertainty number specifying the radius of error at one sigma (\( \sigma \)) in meters of the received location. This uncertainty number is used later on for estimating the trajectory coordinates.

The GPS data generated has the following structure:

\[ \text{Timestamp, X, Y, Z, Uncertainty}_X, \text{Uncertainty}_Y, \text{Uncertainty}_Z \]

Other information that it can be received is the GPS velocity but it is not used for the SmartNav.

The GPS time information can be useful to synchronize the processor’s time device. From the instant when the first GPS data is received, the internal clock time is read and then related to the UTC time provided by the GPS. Synchronizing the processor time is useful for compensating the smartphone internal clock drift. Synchronization has the following format:

\[ \text{Processor time, UTC time} \]
In order to acquire images from the smartphone camera, it is necessary to use the camera class from the Android’s library. An initialization is required in order to set the parameters from the camera, such as the camera mode (camera or video camera), color effects, etc. These options are previously introduced in the acquisition options menu before starting the acquisition. Once the camera class is initialized, the SW can start taking images.

To synchronize the images from the camera, the internal processor clock is used to timestamp every acquired frame from the camera. When acquiring pictures, synchronization is made in the shutter time. In video camera mode, the synchronization is made when the frame is received in the smartphone. It is important to emphasize that in the video camera mode there is an extra delay between the image taken and the timestamp when the frame is delivered.

Once the image is acquired, it is stored on the smartphone µSD memory, with “.jpg” compression using the following filename:

\( \text{Image\_FrameNumber\_Timestamp.jpg} \)

Regarding to the camera acquisition, an optional but recommended step to be performed is to correct the distortion on the acquired image. It is only possible to remove distortion if the camera has been previously calibrated. This image correction takes into account the distortion coefficients and the interior orientation matrix obtained from the calibration process. The distortion removed from images has also been implemented with the OpenCV library in C++ programming language. The output corrected image is stored in the smartphone with “.jpg” compression using the following format for the filename:

\( \text{ImageUndistorted\_FrameNumber\_Timestamp.jpg} \)

In order to perform a temperature calibration from the positioning sensors it is necessary to acquire temperature data. The acquisition is done through receiving information about voltage and temperature levels from the battery module. The temperature is not related to the ambient temperature, it is related with the smartphone internal temperature. The temperature changes with one degree of resolution.

\( 4.2.1.2. \quad \text{Inertial and magnetic calibration} \)

In order to increase the performance of the sensors, more precisely the accelerometer, gyroscope and magnetometer, it is necessary to characterize their behavior. This characterization is carried out in order to compensate systematic errors and thus, calibrate the sensors. Each sensor is analyzed at different axis level (X,Y,Z) separately.

The main idea is to carry out static acquisitions and then analyze the behavior of the sensors data. This statistic approximation is more accurate as more acquisition time is used.
The first step is to characterize the sensor data in terms of temperature. This step is carried out because when the smartphone starts to acquire the sensor data, its temperature starts to rise, reaching a temperature difference up to 15 degrees during the first hour. After this, temperature is more or less stabilized. The output temperature drift can be modeled with a first order linear regression equation that relates a given temperature with the associated error value.

Once the temperature drift factor has been taking into account, the second step is to estimate the sensor time drift. Another long period static acquisition is required. A first order linear regression equation is estimated for each sensor.

This linear regression estimates the sensor time drift. This drift has to be compensated in order to extract the mean value and the noise estimation of the sensors.

Another alternative method implemented in order to characterize the sensor noise is the Allan Variance [50]. This method is widely used for the IMU characterization and it can provide information on types and magnitude of various noise terms. The IG has already implemented an Allan Variance SW that is used in the SmartNav.

![Allan Variance representation](image)

**Fig. 4.10** Allan Variance representation [51].

### 4.2.1.3. Geometric camera calibration

The objective of the geometric camera calibration is to estimate the parameters that relate the coordinates of a 3D point and its projection to the image plane. In order to determine a realistic projection, it is necessary to detect and remove the distortion present on the image, thus, determine the distortion coefficients. The distortion coefficients taken into account are the radial and tangential factors. In addition, to determine the camera matrix it is necessary to estimate the focal length and the principal point.

If there is presence of radial distortion it manifests in form of a “barrel” or “fish-eye” effect (Fig. 4.10).
Fig. 4.10 Left image: barrel distortion; Right image: fish-eye distortion [52].

The tangential distortion occurs because the image lenses are not perfectly parallel to the imaging plane.

The estimation of the focal length and the principal point is also part of image sensor characterization. Geometric camera calibration is important in order to determine a relationship between the camera pixel units and the real world units (such as millimeters).

The smartphone camera has been geometrically calibrated using the OpenCV library programmed in C++ language [53]. OpenCV calibration tool is based in the Conrady-Brown method, which attempts to estimate image distortion coefficients and the camera matrix. The Conrady-Brown method is explained in detail in the algorithms chapter.

The distortion coefficients and the camera matrix are stored in xml format file (camera_calibration.xml).

4.2.1.4. Image feature detection, description and matching

In this section the image processing techniques implemented in the Mobile Navigation SW are presented. These techniques have been implemented using the OpenCV library in C++ native code. More specifically, the image processing tools used are the feature detector, descriptor and matching [54].

All acquired images need to be processed in the Mobile Navigation SW, in order to extract relations between pairs of images, identifying common points. From these common points, relatives rotations and translations can be extracted. This image processing step composes feature detection, description and matching tasks (Fig. 4.11).

Fig. 4.11 Feature detection, description and matching scheme.
In image processing, feature detection (Fig. 4.12) refers to methods that aim to compute abstractions of image information. Feature detectors make local decisions at every image point whether there is an image feature of a given type at that point or not. A feature is defined as an interesting part of an image.

Feature detection is a low-level image processing operation. It examines every pixel of an image to see if there is a feature present at that pixel. There are several types of image features such as edge features, corners features and blobs features.

![Feature detector example](image1)

**Fig. 4.12** Feature detector example.

Edges (Fig. 4.13) are usually defined as sets of points in the image which have a strong gradient magnitude.

![Edge detector example](image2)

**Fig. 4.13** Edge detector example[55].

Corners features (Fig. 4.14) are referred to as points which have a local two dimensional structure. First edge detection is performed and then the edges are analyzed to find rapid changes in direction.
Blobs features (Fig. 4.15) provide a complementary description of image structures in terms of regions. Blob detectors can detect areas in an image which are too smooth to be detected by a corner detector.

The following feature detectors have been implemented in the Mobile Navigation SW:

1. Features from Accelerated Segment Test (FAST): Based on Corner detector [58].
2. Good Features To Track (GFTT): Based on the Shi and Tomasi corner detector method [59].
3. Maximally Stable Extremal Regions (MSER): Based on Blob detector [60].
4. STAR: Based on Center Surround Extrema (CenSurE), a multi scale corner detector with full spatial resolution [61].
5. Canny: Uses a Gaussian filter to convolve with the raw image, making black and white images based on Edge detector useful for detecting lines, circles and other geometries [62].

Once the features have been detected, a local image patch around the feature can be extracted. This process is called feature extraction. This extraction may be quite complex and computably expensive. After performing the feature description on an image, a list of vectors descriptors are generated, that is to say, a set of attributes related to that point. These attributes depend on the algorithm used, for example, edge orientation and gradient magnitude in edge detection and the polarity and the strength of the blob in blob detection. Feature
descriptors search to be invariant to scale, rotation, view point changes and illumination.

There are two different feature descriptors implemented in the Mobile Navigation SW:

1. Binary Robust Independent Elementary Features (BRIEF) which is based on a binary feature descriptor making a small feature vector [63].
2. Fast Retina Keypoint (FREAK) which is based also on binary feature descriptor, with the advantage that it is faster and more robust than BRIEF [64].

There are also some feature detectors and descriptors combinations used in the project:

1. Binary Robust Invariant Scalable Keypoints (BRISK) [65].
2. Oriented FAST and Rotated BRIEF (ORB) is a combination of FAST detector and BRIEF descriptor [66].

OpenCV also provides two additional feature detectors and extractors Scale Invariant Feature Transform (SIFT) [67] and Speeded Up Robust Features (SURF) [68]. These features detector and extractors are not available in the OpenCV library for Android. The idea is to test the newest faster algorithms, thus, SIFT and SURF has not been used in SmartNav.

When two different images have been processed by a feature detector and descriptor, then it is possible to make a comparison between the descriptor points vectors with the use of matchers. Feature matching (Fig. 4.16) compares if a descriptor is similar to another descriptor from a list. The basic method of matching is using a convolution mask on all feature vectors of the image pair, and search for the maximum response in the image to be detected. This technique can be easily performed on grey images or edge images. The convolution output will be highest at places where the image structure matches the mask structure, where large image values get multiplied by large mask values. In the Mobile Navigation SW the BruteForce [69] matcher has been used. This compares the query descriptor with all the others descriptors from the list.

The resulting match is a class vector containing parameters of every matched feature such as the pixel coordinates feature point on the first image, the pixel coordinates feature point on the second image and the distance (matched quality parameter) of the matched features.
From all the detected matches, a selection of the best good matches must be performed. The proposed method consists of selecting the fifteen matched points with the least distance. In addition to this, the matched points selected need to accomplish a maximum distance at three times the smallest distance feature match. It is a restrictive method but useful for minimize outliers.

After selecting the good matches, a RANdom SAmple Consensus (RANSAC) method is used to detect outliers, and remove some false matches.

The related matches between two images are in pixel coordinates and it is necessary to convert these pixel measurements into metric coordinates. This process is explained in the annexes. Once the matches have a metric unit, the Mobile Navigation SW generates a match file. This file contains two time tags, the processor time referring to the previous image and the processor time referring to the most recent image, the previous image coordinates and the newest image coordinates in µm:

\[ \text{Timestamp0}, \text{Timestamp1}, x0, y0, x1, y1 \]

In order to visualize the correctness of the matching process, matched points are drawn with straight lines unifying the points (Fig. 4.16). The pair of images and the matched points are stored in the smartphone µSD with “.jpg” compression using the following filename:

\[ \text{MatchedImage \_ FrameNumber.jpg} \]

For evaluating the matching algorithm, the following definitions are taking into account:

- True positives (TP) are the number of correct matches.
- False negatives (FN) are the number of matches that were not correctly detected.
- False positives (FP) are the proposed matches that are incorrect.
- True Negatives (TN) are the number of matches that were correctly rejected.

This numbers can be converted into unit rates in order to evaluate their performance:
\[ TPR = \frac{TP}{(TP + FN)} : \text{True Positive Rate} \]  
\[ FPR = \frac{FP}{(FP + TN)} : \text{False Positive Rate} \]  
\[ PPV = \frac{TP}{(TP + FP)} : \text{Positive Predictive Value} \]  
\[ FPR = \frac{(TP + TN)}{(TP + FN + FP + TN)} : \text{Accuracy} \]

4.2.2. Navega

Navega is a SW platform from the Institute of Geomatics. The aim of the SW is to achieve the optimal determination of trajectories driven by measurements and their associated dynamic or static models.

Navega has evolved from an INS/GNSS navigation method, into the above more general concept to accommodate the various instrument and sensor configurations of modern navigation and orientation systems. Thus, Navega is relatively modular, it can be configured for GNSS alone based navigation systems, for classical hybrid INS/GNSS systems, for INS/GNSS systems augmented with other ancillary navigation sensors or for INS/GNSS systems with multiple IMU or multiple GNSS receivers to mention a few examples.

Navega allows for real time and post process trajectory determination.

The main purpose of the Navega platform is to provide a trajectory solution plus instrument calibration parameters from positioning/orientation sensors such as IMUs, GNSS receivers, barometric altimeters, odometers, digital compasses, cameras, etc. It delivers time, position, velocity, attitude and calibration information (expectations and covariance's) and other related information (reliability, incidents, etc.)

In order to do this, a State-Space (SS) approach is implemented. This approach is a two-step process. The first one involves the integration of the observations related to dynamic models to get an estimation of the states. The second involves the update of the estimated states through an updating algorithm. This updating step is usually computed through a least squares adjustment implementation (usually a filter of the Kalman Filter family).
In order to integrate the camera data into the Navega SW it is necessary to develop a model. The modeling consists of obtaining a relationship between the matched points between two images and the navigation solution. This model is based on the coplanarity constrain explained in details in the next chapter.

In the SmartNav context, the Navega SW estimates the trajectory using four different models. The first model is the mechanization equations based on the IMU observations [17]. These mechanization equations estimate the position, velocity and attitude. In addition, they can also estimate the accelerometer and gyroscopes biases, scale factors and misalignment factors. The second model is used for the position observations in order to update the position state. The third model implemented is the coplanarity, that generates constrains of relative positions and orientations. The last model allows forcing velocity updates during a period of time. All the models can be used together, or a subset of them, to estimate a unique trajectory.

The Navega SW has been used in post processing for obtaining trajectory estimation within the SmartNav.
CHAPTER 5. ALGORITHMS

5.1. Magnetometer orientation estimation

This algorithm explains how to obtain orientation Euler angles from magnetometer measurements and accelerometer measurements.

The first step is to transform the mobile phone device vector coordinates \((x,y,z)\) to the world’s coordinate system \((X,Y,Z)\). This transformation is implied by generating a \((3\times3)\) rotation matrix using the accelerometers data and the magnetic field data. The rotation matrix transforms the smartphone Forward Right Down (FRD) reference frame into a local East North Up (ENU) reference frame. The Inclination matrix is a rotation matrix that transforms the geomagnetic vector into the same coordinate space as gravity, that is to say, a simple rotation on the \(X\) axis.

\[
\begin{bmatrix}
0 \\
0 \\
g
\end{bmatrix} = \begin{bmatrix}
A_x \\
A_y \\
A_z
\end{bmatrix} \begin{bmatrix}
R_{mat}^{ENU}_{bFRD}
\end{bmatrix}
\]

(5.1)

\[
\begin{bmatrix}
0 \\
\phi \\
0
\end{bmatrix} = \begin{bmatrix}
Mag_x \\
Mag_y \\
Mag_z
\end{bmatrix} \begin{bmatrix}
R_{mat}^{ENU} \\
I_{mat}
\end{bmatrix}
\]

(5.2)

\(g\) = gravity module
\(\phi\) = magnitude of geomagnetic field
\(I_{mat}\) = Inclination matrix
\(R_{mat}\) = Rotation matrix
\(A\) = Acceleration
\(Mag\) = Magnetic flux

This rotation matrix will be valid only if the smartphone is not in free-falling and if the smartphone is not close to the magnetic north. Otherwise if the device is accelerating or placed into strong magnetic field, the rotation matrix will be inaccurate.

Another step is to convert the ENU reference frame rotation matrix into a North East Down (NED) reference frame in order to be compatible with the heading pitch roll nomenclature.

The following step is to transform the NED rotation matrix into three angles (Fig. 5.1):

1. Yaw/Azimuth/Heading: Rotation angle on Z axis \((\alpha)\)
2. Pitch: Rotation angle on X axis \((\beta)\)
3. Roll: Rotation angle on Y axis \((\gamma)\)
Where \( X \) is the vectorial product of \( Y^*Z \), \( Y \) is tangential to ground and orientated to north and \( Z \) is pointing towards the center of earth perpendicular to ground.

\[
R_{\text{Mat}} = \begin{pmatrix}
\cos \alpha \cos \beta & \cos \alpha \sin \beta \sin \gamma - \sin \alpha \cos \gamma & \cos \alpha \sin \beta \cos \gamma + \sin \alpha \sin \gamma \\
\sin \alpha \cos \beta & -\sin \alpha \sin \beta \sin \gamma + \cos \alpha \cos \gamma & \sin \alpha \sin \beta \cos \gamma - \cos \alpha \sin \gamma \\
-\sin \beta & \cos \beta \sin \gamma & \cos \beta \cos \gamma
\end{pmatrix}
\]  \( (5.3) \)

Fig. 5.1 Heading, pitch and roll representation [70].

5.2. Geodetic to ECEF coordinates transformation

The received GPS data is expressed in geodetic coordinates (longitude \( \lambda \), latitude \( \phi \) and height \( h \)) and it is necessary to transform this to a ECEF coordinate system \((X,Y,Z)\). The reference frame is the WGS84.

Fig. 5.2 Geodetic to ECEF coordinates representation [71].
This transformation is realized in the following way:

\[
\begin{align*}
X &= (N(\phi) + h)\cos \phi \cos \lambda \\
Y &= (N(\phi) + h)\cos \phi \sin \lambda \\
Z &= (N(\phi)(1 - e^2) + h)\sin \phi
\end{align*}
\] (5.4)

Where \( N(\phi) \) is called the Normal and is defined as the distance from the center of the earth to the earth surface along the Z-axis. The mathematical definition of the Normal is the following formula:

\[
N(\phi) = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}}
\] (5.5)

Where \( a \) is the semi-major axis equal to 6378.137 km and \( e^2 \) the first numerical eccentricity squared of the Earth ellipsoid respectively 6.69437999010^-3.

### 5.3. Bias and noise extraction

This section explains how the sensors data, specifically, accelerometer, gyroscope and magnetometer data, is characterized in order to understand the their behavior.

The bias can be calculated averaging the static data. The mean value is calculated as the mean of all the acquired data.

\[
A = \frac{1}{n} \sum_{i=1}^{n} a_i
\] (5.6)

This bias estimation can change due to the smartphone position and orientation configuration, because this is directly proportional to the smartphone attitude. In order to calibrate the sensors bias it is necessary to make a static acquisition using a calibration board perpendicular to the earth’s surface.

The noise value is estimated as the standard deviation of all the acquired data.

\[
\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2}
\] (5.7)

This value does not depend on the smartphone attitude. It is very useful in order to characterize the sensor data quality. This noise is introduced as an input into the Navega SW in order to estimate a trajectory.
5.4. Conrady-Brown camera calibration

As mentioned before, OpenCV calibration algorithm is based in the Conrady-Brown method, allowing the estimation of the image distortion coefficients. The distortion coefficients taken into account are the radial and tangential factors.

The radial distortion corrects the image distortion that propagates from the center to the corners of the image in a radial way. The following formula is applied:

\[
x_{\text{undistorted}} = x_{\text{distorted}}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)
\]
\[
y_{\text{undistorted}} = y_{\text{distorted}}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)
\]
\[
r = \sqrt{(x_{\text{distorted}}^2 + y_{\text{distorted}}^2)}
\]

So for an original pixel point at \((x, y)\) coordinate in the input image, for a corrected output image its position will be \((X_{\text{undistorted}}, Y_{\text{undistorted}})\). \(r\) is the radius distance in pixels from the center.

In order to estimate the tangential errors from the camera, the following formulas have been applied:

\[
x_{\text{undistorted}} = x_{\text{distorted}} + [2 p_1 x y + p_2 (r^2 + 2 x^2)]
\]
\[
y_{\text{undistorted}} = y_{\text{distorted}} + [p_1 (r^2 + 2 y^2) + 2 p_2 x y]
\]

Finally, five distortion parameters are required for the image distortion calibration, which are known as distortion coefficients:

\[
\text{Dist}_{\text{coeff}} = (k_1, k_2, p_1, p_2, k_3)
\]

For determining the relationship between the camera units pixels \((x, y)\) and the real world units \((X, Y, Z)\), a transformation matrix named camera matrix is applied.

\[
\begin{pmatrix}
x \\
y \\
w
\end{pmatrix} =
\begin{pmatrix}
f_x & 0 & c_x & |X| \\
0 & f_y & c_y & |Y| \\
0 & 0 & 1 & |Z|
\end{pmatrix}
\]

For determining the camera matrix, the three unknown parameters are the focal length \(f\) (but expressed as \(f_x\) and \(f_y\) and \(P(c_x, c_y)\), the optical principal point, both expressed in pixels coordinates.

\(w\) is a non scaled distance because an homography coordinate system is used, assuming that all points remain in the same plane. The distance between points is normalized to \(w=1\).
While the distortion coefficients are the same regardless of the camera pixels resolutions used, the camera matrix must be scaled. The image is scaled depending on the calibrated resolution and the current resolution because the pixel size changes.

The process of determining the distortion coefficients and the camera matrix is called camera calibration. To calculate these parameters, OpenCV uses a (6x9) size black-white chessboard pattern (Fig. 5.3) in order to automatically recognize the intersection points \((X,Y,Z)\) and compare the same 3D points between pairs of images. The target is fixed in a static position while the images need to be taken at different distances and perspectives from the target.

![Chessboard Pattern](image)

The advantage of using a planar target is that the 3D coordinates of the chessboard are known. The 3D points can be solved using a local reference coordinate.

The position and orientation (rotation and translation) for each image must be calculated in order to determine the camera matrix and the distortion coefficients. This matrix is a \((3x4)\) size that contains the \((3x3)\) rotation matrix and the \((1x3)\) translator vector.

A scene view is formed by projecting 3D points into the image plane using a perspective transformation:

\[
\begin{bmatrix}
  u \\
  v \\
  1
\end{bmatrix}
= 
\begin{bmatrix}
  f_x & 0 & c_x & r_{11} & r_{12} & r_{13} & t_1 \\
  0 & f_y & c_y & r_{21} & r_{22} & r_{23} & t_2 \\
  0 & 0 & 1 & r_{31} & r_{32} & r_{33} & t_3
\end{bmatrix}
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

\[(5.12)\]

\[
\begin{bmatrix}
  u \\
  v \\
  1
\end{bmatrix}
= CAM_M (R | T)_M
\begin{bmatrix}
  X \\
  Y \\
  Z \\
  1
\end{bmatrix}
\]

\((X,Y,Z)\) : Are the coordinate of a 3D point in the world coordinate space, with a reference point as an origin in the chessboard pattern. All these coordinates are
known because the chessboard pattern has pre-defined measurements. In the (6x9) chessboard pattern there are a total of 54 known coordinate points.

\((u,v)\) : Are the pixel coordinates projection in the image. A total of 54 measurements are automatically detected from the chessboard pattern. 54 pixel coordinates are used for each image, that is to say, the number of equations is 54*Num images.

\(\text{CAM}_M\) : The camera matrix, containing the principal point \((cx,cy)\) and the focal distance \(f\), a total of three constant unknowns.

\(\left( R|T \right)_M\) : Rotation and Translation matrix containing the 6 exterior orientation parameters \((X_0, Y_0, Z_0, \omega, \varphi, \kappa)\). These parameters depend on the position and orientation of every image. So the total number of unknowns depends on 6 times the number of images to calibrate.

This perspective transformation can also be expressed as the following equation:

\[
\begin{pmatrix}
  x \\
  y \\
  1
\end{pmatrix} = R_M \begin{pmatrix}
  X \\
  Y \\
  Z
\end{pmatrix} + T_v
\]  

(5.13)

Where:

\[
x' = x(1 + k_1r^2 + k_2r^4 + k_3r^6) + 2p_1xy + p_2(r^2 + 2x^2)
\]

\[
y' = y(1 + k_1r^2 + k_2r^4 + k_3r^6) + 2p_2xy + p_1(r^2 + 2y^2)
\]

\[
r^2 = x^2 + y^2
\]

\[
u = f_xx' + c_x
\]

\[
v = f_yy' + c
\]

In these equations the five distortion parameters \((k_1,k_2,p_1,p_2,k_3)\) are taken into account.

The camera matrix and the distortion coefficients are calculated simultaneously. A RANSAC method is used for detecting outliers.

In practice, for achieving good results, 10 images are required and it is necessary to be taken in different distances and perspectives.

5.5. **Coplanarity modeling**

There are several methods in order to estimate the exterior orientation using feature detection, description and matching output. These consist of estimating six parameters \((X_{\text{rot}}, Y_{\text{rot}}, Z_{\text{rot}}, X_{\text{trans}}, Y_{\text{trans}}, Z_{\text{trans}})\) which describe the spatial position...
and orientation of a camera reference frame with respect to a global object reference frame.

The selected approach is the coplanarity model. The coplanarity algorithm describe the transformations equations needed to link object coordinates with their corresponding image coordinates. These coordinates are related as function of the exterior orientation parameters, three translations and three rotations. Specifically, coplanarity constrain can only determine five from the six parameters of orientation, the translation is scaled by an unknown factor. For this reason, it is only possible to determine the direction and not the magnitude of the translation. So these observations are useful for determining the camera pose, that is, the three angular rotations and three the translation distances scaled.

This modeling is carried out outside the Mobile Navigation SW, to be more precise it is done inside the Navega SW, as a new model.

The orthogonal rotation matrix is the resultant of three independent camera rotations around the local coordinate’s axis (X,Y,Z). It is a different nomenclature as the Heading, Pitch and Roll because it is not expressed in global coordinates, it represents a relative orientation respect the three angles rotations of the smartphone.

\[
R_{mat} = R_xR_yR_z = \begin{pmatrix}
    r_{11} & r_{12} & r_{13} \\
    r_{21} & r_{22} & r_{23} \\
    r_{31} & r_{32} & r_{33}
\end{pmatrix} : \text{Rotation matrix} \quad (5.22)
\]

The translation vector is represented as a 3D coordinate point that represents the spatial position of the image.

\[
T_{vec} = \begin{pmatrix}
    X_t \\
    Y_t \\
    Z_t
\end{pmatrix} = \begin{pmatrix}
    X_t / Z_t \\
    Y_t / Z_t \\
    1
\end{pmatrix} : \text{Translation vector} \quad (5.23)
\]

The coplanarity constrain defines the projection of an image point into a corresponding object point, similar to the Homography camera calibration method. The main difference is that homography projects all object points in the same 2D plane, while coplanarity projects all points in a 3D space.

The first assumption for the Coplanarity model is an epipolar constraint (Fig. 5.4). In two different images planes (\(\pi_L, \pi_R\)), the same (X,Y,Z) coordinate Point (P) is detected in different pixel coordinates. Epipolar geometry is based on the origin point of the left image, the origin point of the right image and the detected point (P) together creating an epipolar plane.
Then, the \((Pr)\) vector can be related with the \((Pl)\) vector, taking into account the rotation matrix and the translation vector between the images.

\[
P_r = R_{Mat} (P_l - T_{vec})
\]  

\[
(P_l - T_{vec}) = R_{Mat}^{-1} P_r = R_{Mat}^{T} P_r
\]

In the coplanarity model \((Pl, T, \text{and} \ P_r)\) vectors are coplanar. One property of the coplanar vector is the following equation:

\[
(P_l - T_{vec})^{T} \cdot T_{vec} \times P_l = 0
\]

So developing the equation the following expression is obtained:

\[
(R_{Mat}^{T} P_r)^{T} \cdot T_{vec} \times P_l = (P_r^{T} R_{Mat}) \cdot (T_{vec} \times P_l) = 0
\]

The Navega SW solves all the above equations, together with the equations from other models, using a non-linear sequential least squares.
CHAPTER 6. VERIFICATION AND VALIDATION

6.1. Tests description

6.1.1. Camera calibration

In order to calibrate the camera sensor, ten images (Fig. 6.1) of the chessboard pattern in different perspectives have been used.

![Fig. 6.1 Camera screenshots used for calibration.](image)

Different tests have been carried out taking images at the maximum resolution in order to accurately extract the distortion coefficient parameters and the camera matrix (intrinsic parameters). The calibration process can be carried out inside the smartphone SW, but these tests were performed in a PC. The reason is the time consumption of the chessboard detection algorithm at high resolution images. The documented tests consists of two different camera calibration tests which were carried out at different periods, in May and October, in order to observe a geometric calibration time dependence.

6.1.2. Feature detection, extraction and matching test

This test is used to evaluate and compare several feature detectors and extractors implemented with the OpenCV library. A sequence of ten images from two different indoor navigation tests was used. The performance was computed using the indices introduced in subsection 4.2.1 (Fig. 6.2).

![Fig. 6.2 Feature detector, extractor and matching test samples.](image)
The following combinations have been tested in the Mobile Navigation SW:

<table>
<thead>
<tr>
<th>Combination</th>
<th>Feature detector + extractor + matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FAST + FREAK + BruteForce</td>
</tr>
<tr>
<td>2</td>
<td>STAR + BRIEF + BruteForce</td>
</tr>
<tr>
<td>3</td>
<td>GFFT + FREAK + BruteForce</td>
</tr>
<tr>
<td>4</td>
<td>ORB + BruteForce</td>
</tr>
<tr>
<td>5</td>
<td>BRISK + BruteForce</td>
</tr>
<tr>
<td>6</td>
<td>MSER + FREAK + BruteForce</td>
</tr>
</tbody>
</table>

### 6.1.3. Positioning sensors calibration and characterization test

This consists of a 6 hour static test in order to characterize the embedded smartphone positioning sensors. This data has been acquired from accelerometer, gyroscope, magnetometer and temperature sensor. The GPS receiver has not been characterized because the received data is already processed. The aim is to calibrate temperature drifts and evaluate the noise and bias from the sensors. In addition, a time drift component has been estimated.

### 6.1.4. Outdoor navigation test

The outdoor test consists of acquiring all the possible sensor data embedded on the smartphone. The realized test consist of 37 minutes of a car trajectory, acquiring data from accelerometer, gyroscope, magnetometer and GPS sensors. In addition, images have been acquired and processed by a feature detector description and matching algorithm. The aim of this test is to validate the acquisition, synchronization and pre-processing of the sensor data.

In this test, two different navigation have been estimated. A real time navigation can be derived combining the position from the GPS and the attitude derived from the magnetometer and the accelerometer sensors. Alternatively, in post processing, a trajectory has been estimated using a hybrid navigation SW. In this case the navigation is based on INS/GPS. This test allows us to compare both solutions.

### 6.1.5. Indoor navigation test

This consists in an indoor test walking within the IG building using a trolley where the smartphone and a reference navigation system are installed. It is important to remark that this indoor test is a different test from the ones used to evaluate the image feature detectors and extractors. The aim of this indoor test
is to evaluate the quality of smartphone navigation solution. In order to evaluate the smartphone navigation solution, the estimated trajectory is compared with the one estimated by using a reference navigation platform.

The reference navigation system consists of a high grade INS named FJI from IMAR company [73]. With this IMU system data, a reference navigation solution is generated in order to compare the smartphone navigation solution.

The route trajectory inside the building is planned a priori (Fig. 6.3). Some points, marked with a yellow cross, within the trajectory are measured in a global reference frame. These points positions are measured in an ECEF reference frame and are used as reference points. In the known positions, the trolley stops and remains static for approximately 10 seconds in every known position. These point positions are introduced into the Navega SW as a simulated GPS receiver in order to perform a hybrid navigation. In addition, a Zero velocity Update (ZUPT) is applied during the static periods.

The test has an initial static part of three minutes. This static period is used to determine the initial position and attitude.

During the test, accelerometer, gyroscope, magnetometer data are recorded. In addition, images are acquired and processed. No GPS data is recorded because the test is carried out in an indoor environment.
In the trolley platform (Fig. 6.4) the equipments are installed in a known position. The level arms (Fig. 6.4) between the different navigation equipments are measured and introduced into the Navega SW.

![Image](image.png)

**Fig. 6.4** Left image: Trolley platform; Right image: Level arms.

After the test is performed, two different navigation solutions are computed on the Navega SW. The first solution is based on an INS/GPS method using the FJI IMU and the reference points as simulated GPS coordinates. This first solution is the reference navigation. The second solution computed is the smartphone navigation solution, a hybrid solution taking into account the IMU data, the image observations, the reference points and the ZUPTs periods.

### 6.2. Tests results

#### 6.2.1. Camera Calibration

In May, several camera calibration tests were performed. The intrinsic parameters obtained were similar in all the tests that were carried out.

The nominal central point should be $c = (1632,1224)$ pixels, but the estimated principal point is deviated to $c' = (1619,1234)$ pixels. The obtained focal length is $f = 4.08$ mm, quite close to the nominal (4 mm).

**Table 6.2** May test camera matrix results (in pixels)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>fx</td>
<td>0</td>
<td>cx</td>
</tr>
<tr>
<td>0</td>
<td>fy</td>
<td>cy</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\[
\begin{bmatrix}
2935.64 & 0 & 1619 \\
0 & 2944.08 & 1234 \\
0 & 0 & 1
\end{bmatrix}
\]
The following formula is applied to obtain the focal length in millimeters units from pixels units:

\[ f = f_x \cdot \text{pixel size } x = f_y \cdot \text{pixel size } y \quad (6.1) \]

The estimated distortion coefficients are:

**Table 6.3** May test distortion coefficients results (in pixels)

<table>
<thead>
<tr>
<th>k1</th>
<th>k2</th>
<th>p1</th>
<th>p2</th>
<th>k3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.28E-01</td>
<td>-9.73E-01</td>
<td>-2.32E-04</td>
<td>-1.29E-03</td>
<td>9.51E-01</td>
</tr>
</tbody>
</table>

The distortion coefficients can be separately represented with a radial component plot and a tangential component plot (Fig. 6.5).

Fig. 6.5 Left: Radial component of the distortion in May. Right: Tangential component of the distortion in May.

A second camera calibration test was carried out in October. The estimated principal point estimated is \( c' = (1634,1200) \) pixels. The obtained focal length is similar as the previous may test result, concretely \( f = 4.11 \text{mm} \). The principal point has changed significantly.

**Table 6.4** October test camera matrix results (in pixels)

<table>
<thead>
<tr>
<th>fx</th>
<th>0</th>
<th>Cx</th>
</tr>
</thead>
<tbody>
<tr>
<td>2929.26</td>
<td>0</td>
<td>1634</td>
</tr>
<tr>
<td>0</td>
<td>fy</td>
<td>Cy</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 6.5** October test distortion coefficients results (in pixels)

<table>
<thead>
<tr>
<th>k1</th>
<th>k2</th>
<th>p1</th>
<th>p2</th>
<th>k3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.92E-01</td>
<td>-8.56E-01</td>
<td>-3.31E-03</td>
<td>1.25E-03</td>
<td>9.04E-01</td>
</tr>
</tbody>
</table>

The radial and the tangential components of distortion of the October camera calibration results (Fig. 6.6) have changed with respect to the May calibration results. It can be observed that the radial component value is very similar to the May test while the tangential component has changed significantly.
Looking at the calibration results, the geometric camera calibration is time depending.

![Radial Component of the Distortion Model](image1)

![Tangential Component of the Distortion Model](image2)

**Fig. 6.6** Left: Radial component of the distortion in October. Right: Tangential component of the distortion in October.

With the camera matrix and the distortion coefficients the camera is calibrated and it is possible to remove distortions on the images (Fig. 6.7).

![Original Image](image3)
![Undistorted Image](image4)

**Fig. 6.7** Left: Original image. Right: Undistorted image.

Looking at the border lines of the original image (Fig. 6.7 left), a small fish-eye distortion effect can be observed.

### 6.2.2. Feature detection and extraction test

All the different feature detectors and extractors algorithms, presented on section 6.1.2 have been tested with a subset of images from an indoor test. After carrying out the feature detectors and extractor process, interest points are detected. Each feature detector and extractor has their own threshold parameters. For this test, the default values are used.
Subsequently, the interest points from the two images are compared and the resulting final matches are drawn with lines between them (Fig. 6.9).

![Fig. 6.9 Two consecutive matched points.](image)

In order to evaluate the different features detectors and descriptors, the TPR has been computed for each combination. In addition, a timing benchmark has been carried out in order to evaluate the time consumption of each algorithm. Another consideration taken into account is the minimum five true matched points, in order to be possible to resolve the equations, and thus, extract navigation measurements.

<table>
<thead>
<tr>
<th>Table 6.6 Feature detector, extractor and matching test results</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTAL</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>% TPR</td>
</tr>
<tr>
<td>% image pairs with minimum 5 true matches</td>
</tr>
<tr>
<td>Medium Time (s)</td>
</tr>
</tbody>
</table>

The best results are obtained with STAR and ORB algorithms. On the one hand, STAR algorithm is reliable in terms of true positive rate and it is really fast in terms of time consumption, however, its weakness is that it generates less than the half of all measurements. On the other hand, ORB algorithm is still reliable and generates more measurements but it is three times slower. The BRISK results show the lowest TPR value and the highest medium time. This performance can be due to the used default threshold values. The results
suggest us that threshold values different from the defaults, might be necessary for the BRISK method.

As a consequence of these results, the STAR detector and extractor is selected in the indoor navigation test as the main reference algorithm. In order to perform continuous image navigation, when the STAR detector and extractor cannot extract the required matches, another feature detector and extractor replace it. This image data combination is carried out in post processing. The pair of matched points is used as input observation for the coplanarity model.

6.2.3. Positioning sensors calibration and characterization test

A static acquisition test was carried out in order to characterize the internal smartphone sensors. The following figures show the accelerometer and gyroscopes x-axes and the magnetometer y-axes.

![Fig. 6.10 X-axes accelerometer data (m/s²) during 6 hours (s).](image)

![Fig. 6.11 X-axes gyroscope data (rad/s) during 6 hours (s).](image)
Chapter 6. Verification and validation

Fig. 6.12 Y-axes magnetometer data (deg) during 6 hours (s).

It has been observed that the gyroscope sensor is affected by the smartphone internal temperature component (Fig. 6.13), while this behavior was not observed in accelerometers and magnetometers data. The temperature range measured is from 33 degrees up to 48 degrees Celsius. The following graph shows the gyroscope data in terms of internal temperature. Theoretically, the gyro mean value should be “0 rad/s”.

Fig. 6.13 Z-axis gyroscope data (rad/s) in terms of temperature (ºC).

It can be clearly observed that there is a temperature drift in the gyroscope data. This temperature drift can be approximated to a linear equation using a first-order linear regression algorithm. Knowing that the gyroscope output data of a static acquisition should be 0 rad/s, it is possible to determine the gyroscope error in function of the temperature. The following graph represents three different regression lines that approximate the temperature drift related to the three gyroscope sensor axis.

Fig. 6.14 Gyroscopes (rad/s) temperature (ºC) compensation.
Observing at Fig. 6.14, it seems that approximately at 40 degrees, the three axes have minimum error.

In addition, a temporal drift component has also been estimated for each sensor and each axis. The temporal drift observed in the accelerometer sensor and in the gyroscope sensor is not significant (Fig. 6.15 and Fig. 6.16).

![Fig. 6.15 Accelerometer temporal drift.](image)

![Fig. 6.16 Gyroscope temporal drift.](image)

Regarding the magnetometer temporal drift (Fig. 6.17), the x-axis component is very susceptible to time, achieving a difference of one degree at five hours of acquisition.

![Fig. 6.17 Magnetometer temporal drift.](image)
Once the temperature and temporal drift has been calibrated, it is possible to characterize the sensors in terms of bias and noise. All static tests carried out reached to similar noise values, verifying the sensors performance.

Table 6.7 Sensors characterization and calibration test results

<table>
<thead>
<tr>
<th></th>
<th>ACCELEROMETERS (m/s²)</th>
<th>GYROSCOPES (rad/s)</th>
<th>MAGNETOMETERS (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noise X-axis</td>
<td>0.038626</td>
<td>0.002199964</td>
<td>0.889523149</td>
</tr>
<tr>
<td>Noise Y-axis</td>
<td>0.029747</td>
<td>0.002519801</td>
<td>0.228508635</td>
</tr>
<tr>
<td>Noise Z-axis</td>
<td>0.043502</td>
<td>0.00307757</td>
<td>0.17578425</td>
</tr>
<tr>
<td>BIAS X-axis</td>
<td>0.396373</td>
<td>0.001616033</td>
<td>-8.0171093</td>
</tr>
<tr>
<td>BIAS Y-axis</td>
<td>-0.068177</td>
<td>0.000677086</td>
<td>-2.34326702</td>
</tr>
<tr>
<td>BIAS Z-axis</td>
<td>0.123613</td>
<td>-0.004887857</td>
<td>-0.402904615</td>
</tr>
</tbody>
</table>

In order to validate the obtained noise results, an Allan Variance has been computed for the IMU data. The white noise results obtained by applying the Allan Variance are similar, in the same order of magnitude to the characterization results obtained before. Additional Allan variance plots can be found in the annexes section.

Table 6.8 Sensors characterization with Allan Variance results

<table>
<thead>
<tr>
<th></th>
<th>ACCELEROMETERS (m/s²)</th>
<th>GYROSCOPES (rad/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>White noise X</td>
<td>0.065</td>
<td>0.0030</td>
</tr>
<tr>
<td>White noise Y</td>
<td>0.050</td>
<td>0.0035</td>
</tr>
<tr>
<td>White noise Z</td>
<td>0.080</td>
<td>0.0045</td>
</tr>
</tbody>
</table>

The noise estimation is used later on as an input for the navigation SW.

6.2.4. Outdoor navigation test

The outdoor navigation test was performed in a car. The smartphone was located in the front window, next to the co-driver’s seat. The trajectory starts on Cervello village and finalizes on Vallbona d’Anoia village, with 37 minutes duration. A total of 3882 images were acquired, achieving a frame rate of 1.6 frames per second. The accelerometer, gyroscope and magnetometer data were acquired at 100Hz while the GPS data was acquired at 1 Hz.

A real time navigation has been performed using the position provided by the GPS sensor and the attitude derived from magnetometer and accelerometer data. Due to time synchronization, the acquired images are positioned and orientated using the previous navigation solution.

In post processing, the smartphone trajectory has been estimated using a INS/GPS hybridization for estimating position, velocity and attitude.

In order to visualize the trajectories, the estimated position by the GPS receiver (red line) and the estimated position with the INS/GPS (green line) are drawn on
a map using Google Maps (Fig. 6.18). The INS/GPS trajectory has a continuous solution while the GPS trajectory have gaps in no GPS signal situation such as tunnels or bridges. The INS/GPS output frame rate is 100 Hz while GPS based position frame rate is 1Hz. In Fig. 6.18, the INS/GPS position is down sampled to 2.5Hz for visualization purposes.

**Fig. 6.18** Estimated position with GPS (red line) and INS/GPS (green line).

The uncertainty of the estimated position by the GPS alone solution remains between 5 and 10 meters (Fig. 6.19). The performance of the INS/GPS solution has not been quantified in the outdoor test because there is not a reference navigation system. However the estimated position is close to the main road and its shape is very similar to the performed route.

**Fig. 6.19** GPS uncertainty (m) in time (s).

The estimated INS/GPS attitude is the following:

**Fig. 6.20** INS/GPS attitude estimation.
The estimated attitude seems to be coherent with the performed attitude. The roll component remains close to 0 degrees. This value is coherent with the car’s dynamics. The pitch component is also coherent with the road slope changes. Analyzing the solution and looking to the performed route, the heading solution is coherent with the performed heading. For instance, the car was pointing close to East at the beginning, and the estimated heading was 85 degrees. In a roundabout, the estimated heading shows that the car turns 320° degrees.

On the acquired images, the feature detector description and matching methods were applied (Fig. 6.21). However, the coplanarity model has not been used for the navigation estimation due to a high number of false matches. The high number of false matches can be explained for three reasons. The sky occupies half part of the image and it is not suitable for detecting static objects. The second reason is the presence of other moving cars. The last reason is the time space between the matched images when the smartphone moves at high speeds (120 km/h).

![Outdoor test matched points](image)

Fig. 6.21 Outdoor test matched points.

### 6.2.5. Indoor navigation test

The test duration was 20 minutes and about 1912 images were acquired. A total of 1840 images have been matched. The achieved camera frame rate is 1.25Hz. In addition, the accelerometer, gyroscope and magnetometer data were acquired at 100Hz.

As mentioned before, in order to improve the feature detectors and descriptors algorithms reliability, the image data combination was performed in post processing mode for testing and validating the coplanarity model.
In order to evaluate the performance of the smartphone trajectory, a reference IMU was used. The IMU data from the reference system (green) and the smartphone system (red) the z-axis gyroscope and the z-axis accelerometer are plot (Fig. 6.23). It can be observed that the smartphone accelerometer and gyroscope sensors are much noisy than the reference ones.

In order to validate the coplanarity model two different smartphone navigation approaches were used. The first approach is an INS/Position navigation method (red line), while the second approach is an INS/Position/Camera navigation method (green line). The estimated trajectories plus the reference trajectory (blue line) are shown in (Fig. 6.24).
Analyzing the results it can be observed that the position cannot properly be estimated with the smartphone. The coplanarity model improves the position determination, but the estimated solution is far from the reference one.

The estimated attitude from the indoor test is better than expected. An example of a heading estimation is presented in the following figure:

The attitude provided by the INS/Position approach is very similar to the reference attitude. In the other hand, the INS/Position/Camera approach deteriorates the attitude estimation.

These results presented are still preliminary, further research must be done to improve the results of the INS/Position/Camera based navigation.
CHAPTER 7. CONCLUSIONS AND OUTLOOK

The main conclusions obtained from the SmartNav project are the following:

- A full hybrid navigation system chain has been developed and tested, named SmartNav, using open source platforms and a Navigation SW.

- A data logger platform has been developed in the smartphone in order to acquire and synchronize all the embedded sensors data.

- The smartphone camera has been geometrically calibrated. It has been also observed a time dependence of the camera calibration. This phenomenon might be because a low cost sensor technology is used and/or because the smartphone can receive easily shocks. For that reason, it is recommended to calibrate the camera before carrying out a test.

- The embedded smartphone sensors have been characterized in order to detect and correct bias and to determine noise values. In addition, the internal gyroscopes temperature drift has been compensated. In a near future, calibration tests must be carried out in order to check a noise and bias time dependence.

- The feature detector descriptor and matching methods consume quite amount of processing time. This time consumption is proportional to the image size, the number of detected features and the selected feature detector and descriptor algorithm.

- The feature detectors, descriptors and matching algorithms that OpenCV includes were compared. There is no a 100% reliable algorithm, thus not always can be extracted navigation observations from a pair of images. Further research must be done for determining the optimal threshold parameters of the feature detector and descriptor methods.

- A hybrid INS/GPS navigation method has been implemented and tested in an outdoor environment. INS/GPS method can estimate position, velocity and orientation at a high frame rate, while the GPS alone method only determines position and velocity at 1 Hz.

- The performance orientation obtained by the accelerometer and magnetometer sensors, in real time, was not validated. Preliminary results indicates us that the obtained orientation is coherent during static periods. Further research will include a validation performance.

- An indoor trajectory was estimated using an INS/Position/Camera hybrid solution. The implemented image model was the coplanarity model. The preliminary results shows that the position estimation is improved, but
less than desired. For the attitude estimation, the coplanarity model does not improve the INS/Position attitude.

- The obtained results show that the smartphone attitude is well estimated using the inertial sensors. These results reveals a promising potential of the smartphone sensors for applications that require orientation.

In order to evolve the SmartNav, some implementations are identified:

- Implement de-noising filters to the accelerometer and gyroscope data signals in order to reduce the IMU data noise.

- Implement a WiFi positioning system inside the smartphone in order to obtain position for indoor environments.

- Refine the optical navigation modeling and search alternatives to the coplanarity model.

- Change the optical navigation strategy. A first solution can be the implementation of a Hough transform to detect lines from a single input image and derive an attitude determination. A second optical navigation strategy can be the implementation of positioned coded targets in order to estimate position and attitude from a single image.

- Implement a real time hybrid navigation using a TCP/IP communication between the mobile navigation SW and the navigation SW.

Last but not least, the trajectory estimation is a key step for many applications, such as guidance to a destination, augmented reality applications, driving assistance, monitoring of the elderly, to name a few.
CHAPTER 8. REFERENCES

[1] Institute of Geomatics: www.ideg.es
[7] Smartphones processors:
[38] Samsung Galaxy SII: http://www.samsung.com/es/support/model/GT-I9100L5KAFOP
[40] Sony Cameras: http://www.sony.net/SonyInfo/News/Press/201010/10-137E/
[41] Equivalent focal length: http://en.wikipedia.org/wiki/35_mm_equivalent_focal_length
[47] OpenCV: http://opencv.org/


[57] Blob detector: http://www.kixor.net/school/2008spring/comp776/assn1/


[70] Heading, pitch and roll: http://www.emeraldinsight.com/journals.htm?articleid=1917078&show=html


[73] FJI: http://www.cbil.co.uk/products/imar/inav-fji/


[76] Google sensor fusion: http://www.youtube.com/watch?v=C7JQ7Rpwn2k
CHAPTER 9. ANNEXES

9.1. Android architecture

Android is divided into four different layers which include five groups:

• Application layer: This layer is the one used by end phone users. Applications can run simultaneously (multitasking) and they are written in Java language.

• Application framework: The SW framework used to implement the standard structure of an application for the Android OS.

• Libraries: The available libraries are written in C/C++ and they are called through a Java interface.

• Android runtime: The Android runtime consists of two components. First a set of core libraries which provides most of the functionality in the Java core libraries. And second the virtual machine Dalvik which operates as a translator between the application side and the OS.

• The kernel: This Linux based kernel is used by Android for its device drivers, memory management, process management and networking.

[Fig. 9.1 Android internal architecture]

The main menu allows choosing between the different modules. The user should follow this sequence: first do a calibration, second realize an acquisition and third visualize the recorded navigation.

![Mobile Navigation main menu](image1)

**Fig. 9.2** Mobile Navigation main menu

In the characterization and calibration activity step, it is possible to characterize the accelerometers, gyroscopes and the compass sensors independently or at the same time. To characterize the sensors, a static test is realized during a long period of time. The SW calibrates the temperature drift of the sensors. In addition, the noise, bias and time drift of the embedded sensors are characterized.

![Sensors characterization and calibration](image2)

**Fig. 9.3** Sensors characterization and calibration
In addition, the camera can be geometrically calibrated by estimating the intrinsic parameters. The radial and tangential distortion coefficients can be estimated by taking several photographs of a chessboard target pattern.

![Camera calibration](image)

**Fig. 9.4 Camera calibration**

This calibration is done by using some functionality from the OpenCV library in native code. OpenCV implements feature detectors for detecting the chessboard and implements the calibration functionality based on the Conrady-Brown method. The results of the calibration activity are four calibration files, one for every sensor.

In the Acquisition activity, the first step is to configure all the navigation and camera options. According to the navigation options, these include storing sensor data and the images into the µSD card. A TCP/IP communication could also be enabled in order to transmit data to an external computer for real time monitoring. The GPS receiver can be enabled or disabled, depending on the environment. The last navigation option is to enable or disable the image feature extraction and matching.

Regarding the camera options, if the camera has been previously calibrated, the option of removing the distortion in the images can be selected. The camera options also include switching between camera mode or video camera mode, the frame rate and the image resolution. There are additional camera options such as color effects, flash modes, scene modes and white balance. Once the options are set, then the navigation acquisition starts.
During the acquisition there is visualization on the screen of the smartphone showing the camera scene. There are several options in the menu button to visualize and manipulate the SW. The sensor data can be visualized in real time for monitoring the acquisition. Also, the attitude data can be represented with an artificial horizon drawing instrument. This artificial horizon is used to indicate the object orientation relative to the Earth’s horizon. In addition, several image detectors and descriptors can be selected and image features detection can be enabled or disabled.
Once the acquisition is finished, the output result files saved depend on the acquisition options selected.

In the visualization activity, several computed navigation routes can be selected. In the first tab, a Google map view is used to show the estimated trajectory.

The second tab shows some trajectory statistics, such as total time, total distance, total images, maximum speed, maximum acceleration, maximum rotation, maximum and minimum height.
The third tab is for plotting the navigation data respect time. The different axes of the accelerometers, gyroscopes and compass data can be selected to analyze the results.

The last tab is for previewing the navigation, where all the recorded images can be played, stopped, paused, forwarded or rewound observing the associated time and orientation data. By clicking on the image, the map tab shows where the related image is located in the map.
Fig. 9.12 Visualizing the navigation images with the position and orientation

9.3. Mobile Navigation files

The Mobile Navigation SW generates the following files:

**GPS data file**: Which includes processor time, UTC time, latitude, longitude, altitude and accuracy.

**Sync file**: Contains a relationship between processor time and UTC time.

**IMU data file**: Contains processor time, accelerometer and gyroscope data.

**Compass data file**: Contains processor time and the compass data.

**Camera data file**: Contains the processor time of the captured images and the matched images coordinates.

**Image files**: The raw acquired images with a processor time associated.

**Undistorted image files**: The calibrated images with a processor time associated.

**Matched image files**: The two matched images in grey scale with the matched points drawn.
9.4. Pixels to distance conversion

When the image pixels are detected after a feature matching process, it is necessary to convert the correctly matched pixels points into distance points. Knowing that the CMOS matrix size is (4.54x3.42) mm, depending on the image resolution the matched point will be translated into CMOS matrix distance.

Fig. 9.13 Translation from pixels to distance.

\[ P(x, y) \rightarrow P(X, Y) \] \hspace{1cm} (9.1)

The image pixel coordinate origin is on the image top-left while in the matrix distance the origin is in the center. Firstly the origin is displaced by applying the follow transformation:

\[ X' = x - \frac{\text{Image Width}}{2} \]
\[ Y' = \frac{\text{Image Height}}{2} - y \] \hspace{1cm} (9.2)

Then the conversion from pixels to µm is done with the following formula:

\[ X = X' \frac{\text{Matrix Width}}{\text{Image Width}} \]
\[ Y = Y' \frac{\text{Matrix Height}}{\text{Image Height}} \] \hspace{1cm} (9.3)
9.5. **Allan variance results**

In order to validate the obtained noise results, an Allan Variance has been computed for the IMU data (Fig. 9.13 and Fig. 9.14). The same static data has been processed with the Allan Variance method in order to estimate the white noise value.

![Fig. 9.13 Allan Variance on gyroscopes data.](image)

![Fig. 9.14 Allan Variance on accelerometers data.](image)