SYSTEM BASED ON AN ANDROID PLATFORM AS A SENSOR TO ASSIST THE AUTONOMOUS LOCALIZATION

Luis Guilherme Duenhas Silva
Kelen Cristiane Teixeira Vivaldini
Marcelo Becker
Engineering School of São Carlos, University of São Paulo - USP. Av. Trabalhador São-carlense, 400, São Carlos - SP/Brazil.
guiduenhas@gmail.com; kriskaya@gmail.com; becker@sc.usp.br.

Abstract. *Project developed to study the use of mobile devices, smartphones, provided with the Android operating platform as a sensor to aid the autonomous localization of robotic systems in indoor environments. The development includes the location of the robotic system by the camera of the device, with which landmarks are detected, extracting information of position and angulation of the robotics system, and the aid to the navigation by the built-in sensors, such as accelerometer, magnetic compass and gyroscope, which also were submitted to a sensor fusion technique. So, was developed an application that can extract data and perform communication with the robotic system or any other external control system. Thus, tests and validations were performed, evaluating the accuracy and performance of the sensors and the algorithms used for data acquisition, which exceeded expectations, allowing a basic autonomous localization in a controlled environment. Finally, was concluded that the main objective of the work was accomplished, validating the use of mobile devices supplied with the Android OS as a system of sensors applicable to autonomous localization of robotic systems.*

Keywords: *Android, Mobile robotics, Mechatronics, Localization, Autonomous navigation.*

1. INTRODUCTION

In 2007, Google stunned the world with the launch of something that goes beyond a simple device: A comprehensive platform for mobile devices, called Android. Since then the system has become popular among the so-called "smartphones" and has found a place through the policy of encouraging the free development of applications.

Manufacturers of smartphones, in turn, has increasingly developed devices with high processing power, memory and other specific sensors such as cameras, accelerometer, magnetic compass, gyroscope, GPS, barometer, allowing high degree of customization and adaptation to the user's needs and desires. These sensors can be employed to assist the autonomous navigation of robotic systems (autonomous cars, robots for cleaning and maintenance, "toys", assistance to persons with disabilities, security systems, exploration or rescue).

1.1 PROBLEMS

In open air environment a GPS receiver is able to determine its position with very high accuracy. Inside a building, where the GPS signal is bad or unavailable, the position estimation from the GPS receiver is very erroneous and of no practical use.

The mobile robotics is an area of robotics that studies robotic systems capable of move and interact with the environment through sensors and actuators. The autonomous navigation is the fundamental problem of the segment. Thus, the autonomous navigation should be able to define a sequence of actions to be taken by the robotic system, equipped with a limited set of sensors, when exposed to an environment and having to meet a certain pre-specified task (Figueiredo, 1999).

Intimately related to autonomy of robotic systems are three primitive paradigms: feel, plan and act. And these are related to how the data are collected by sensors and then processed (THRUN et al., 2005). Thus, once the robotic system is ready for navigating in an environment, there are three classic questions by Leonard and Durrant-White that must be answered by its control algorithms:

- "Where am I?" Question that relates the robotic system with localization strategy or, as above, with the act of "feel".
- "Where am I going?" Asks that relates the robotic system to your goal or task.
- "How do I get there?" Question that relates the robotic system with navigation strategy, act of "planning".

1.2 PURPOSE

This paper presents the development of an application, to mobile phones with Android OS, which can assist the autonomous navigation by the localization of the robotic system in real time, contemplating the extraction of data from the sensors of the device, performing a data processing operation, and do the communication of such data to an external

control. So, being the developer of the robotic system responsible to make best use of the data to their own specific application, navigation.

The main objective of this paper was to examine the level of accuracy that can be achieved in indoor positioning by using built-in sensors in an Android smartphone. So, use the measurements of acceleration and orientation from the device's accelerometer, magnetometer and gyroscope, as the camera of the device, using the advantages of one sensor to compensate the drawbacks of the other and so locate a robotic system at an indoor environment.

2. OVERVIEW ABOUT ANDROID PLATFORM

Developed by the Open Handset Alliance, a consortium of over 30 companies in the sector of technology and communication, and operated by Google Inc., Android is a free, open and complete mobile platform, presenting features incorporated from other systems, well as countless innovative concepts to the mobile segment (Weiss, 2008).

Android has as main objectives the opportunity to customize the applications and components in your system, and the possibility of a quickly and modern development of personal applications, thereby enabling companies and developers to build applications that better fit to the everyday of the users, enhancing their skills, and, depending on the hardware configuration of the device, allowing the user to perform operations also performed on your personal computer.

Besides managing and developing, Google provides the whole system API, on a site dedicated solely to developers. Thus, it allows programmers to write freely in the Java language, software for the platform

The structure of the Android operating system is basically divided into four layers: Kernel, Libraries and Runtime, Framework and Applications. Each with internal modules specialized in their particular task, as illustrated in Fig. 1.

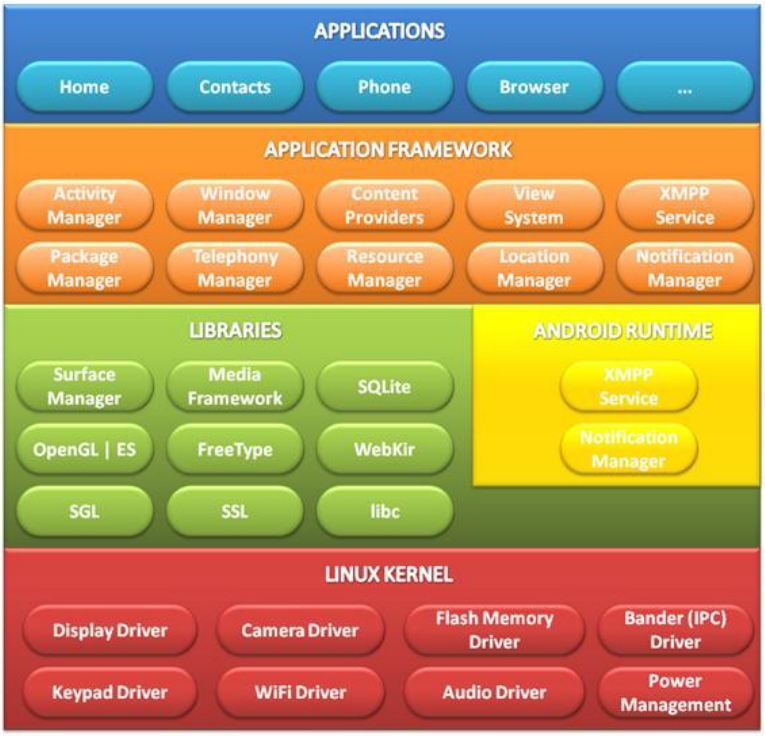

Figure 1. Structure of the Android operating system.

- Application layer: This layer contains the apps, written in Java language, makes contact with the users.
- Application framework: This layer is used to implement the standard structure of an application for the Android OS.
- Libraries: All available libraries from the Android OS are written in C/C++ and can be called through a Java interface. This layer contains the virtual machine Dalvik which operates like a translator between the application side and the operating system.
- Kernel: Based on the Linux kernel is used by Android for its device drivers, memory management, process management and networking.

In this project the development was restricted to the Application layer, thus not being worked on any of the other previous layers that could possibly allow greater speed of execution.

3. AVAILABLE COMPUTER VISION LIBRARY

The library ARToolkit is built in C/C++ and offers supports to the develop applications of Augmented Reality (AR) using the optical tracking, which implements computer vision techniques to identify and estimate in real time the position and orientation of a marker over the video capture device. Thus, the calculation of the correlation between the estimated data of the real marker and its image, makes possible aligned virtual objects with the image of the marker, enabling the rapid development of a wide range of application AR (Kato, 1999).

The ARToolKit basically consists of 4 modules:

- AR Module: central module with routines for tracking markers, calibration and acquisition of parameters;
- Video Module: a collection of routines to capture video frames of video input. This is a wrapper around the pattern of the video capture routine;
- Gsub Module: a collection of graphics routines based on the OpenGL and GLUT libraries (is now obsolete);
- Gsub_Lite Module: replaces the Gsub module with a more efficient collection of graphics routines. Works independent from any toolkit windows.

The Fig. 2 shows the hierarchical structure of the ARToolKit and its relation with the dependency libraries. To minimize the dependencies on the operating system, the ARToolKit uses the OpenGL librarie for part of rendering, the GLUT librarie to manipulate the appearance of the application and hardware dependencies and standard API of each platform.

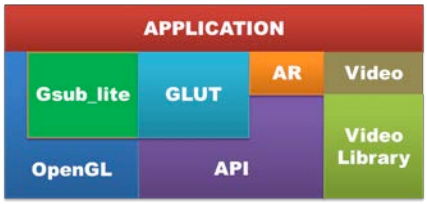

Figure 2. Hierarchical structure of the ARToolKit.

The modules are related in a direct sequence, as shown in Fig. 3, to be easily replaced if necessary by the developer.

Figure 3. Sequence of communication modules of the ARToolKit.

3.1 Structure of the markers

The tracking implemented in ARToolKit estimates the position marker relative to the camera, making it possible to develop applications that require knowing the position and orientation of elements.

The markers recognized by the ARToolkit consist of two square geometric figures, one larger black which contains in its interior a white smaller. The smallest contains in its interior symbols identifying the marker, as shown in Fig. 4. The symbols may or may not geometries, but the higher the complexity of this, the longer the processing time.

Figure 4. Example of marker.

3.2 Theory about the recognition of markers

The pattern recognition in ARToolKit identifies four vertices of square regions contained in the video image, and compares the symbols inside the marker with the templates previously registered by the user (CLAUS, 2005).

Once there is recognition of a marker, the tracking of ARToolKit extracts some information about the detection and identification of markers and to estimate their position and orientation.

Figure 5 schematizes the coordinate systems with which the ARToolKit work, and uses them to calculate the parameters required for the insertion of a virtual object in the image.

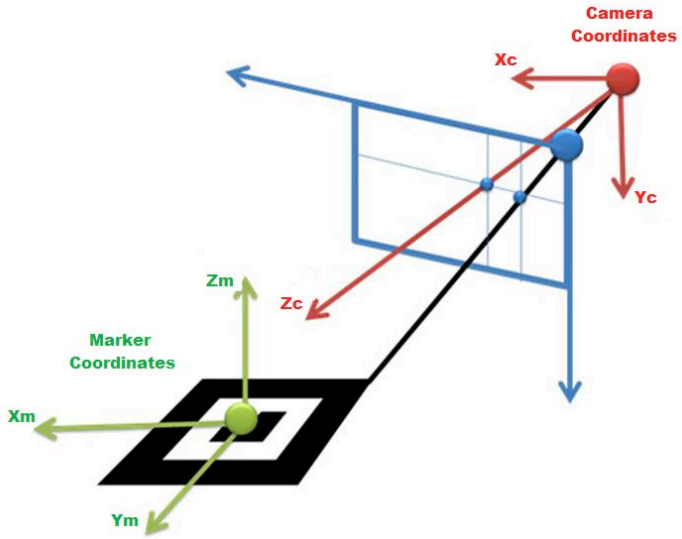

Figure 5. Relationship between the coordinate systems of the marker and the camera.

The marker coordinates and camera coordinates are related via a 3x4 matrix referred to as "transformation matrix". Multiplying the transformation matrix "T" for a 3D point marker (Xm, Ym, Zm), results in the corresponding point in the camera coordinate system (Xc, Yc, Zc).

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} = T \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} \quad (1)$$

In which we have, for each matrix element:

$$R_{11} = Cos(\theta).Cos(\varphi) \quad (2)$$

$$R_{12} = -Cos(\emptyset).Sen(\varphi) + Sen(\emptyset).Sen(\theta).Cos(\varphi) \quad (3)$$

$$R_{13} = Sen(\emptyset).Sen(\varphi) + Cos(\emptyset).Sen(\theta).Cos(\varphi) \quad (4)$$

$$R_{21} = Cos(\theta).Sen(\varphi) \quad (5)$$

$$R_{22} = Cos(\emptyset).Cos(\varphi) + Sen(\emptyset).Sen(\theta).Sen(\varphi) \quad (6)$$

$$R_{23} = -Sen(\emptyset).Cos(\varphi) + Cos(\emptyset).Sen(\theta).Sen(\varphi) \quad (7)$$

$$R_{31} = -Sen(\theta) \quad (8)$$

$$R_{32} = Sen(\emptyset).Cos(\theta) \quad (9)$$

$$R_{33} = Cos(\emptyset).Cos(\theta) \quad (10)$$

$$T_1 = X \quad (11)$$

$$T_2 = Y \quad (12)$$

$$T_3 = Z \quad (13)$$

The turning angles are as shown in Fig. 6.

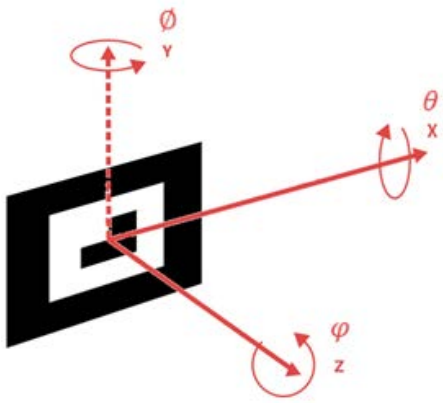

Figure 6. Degrees of freedom of the marker.

Once estimated the transformation matrix, the API "OpenGL" is used to adjust the virtual camera, position and draw the virtual object aligned with the real marker (PIEKARSKI, 2002).

3.3 Logic of operation

The standard operation of the library is described in Fig. 7, since the video capture to the insertion of a virtual object in it.

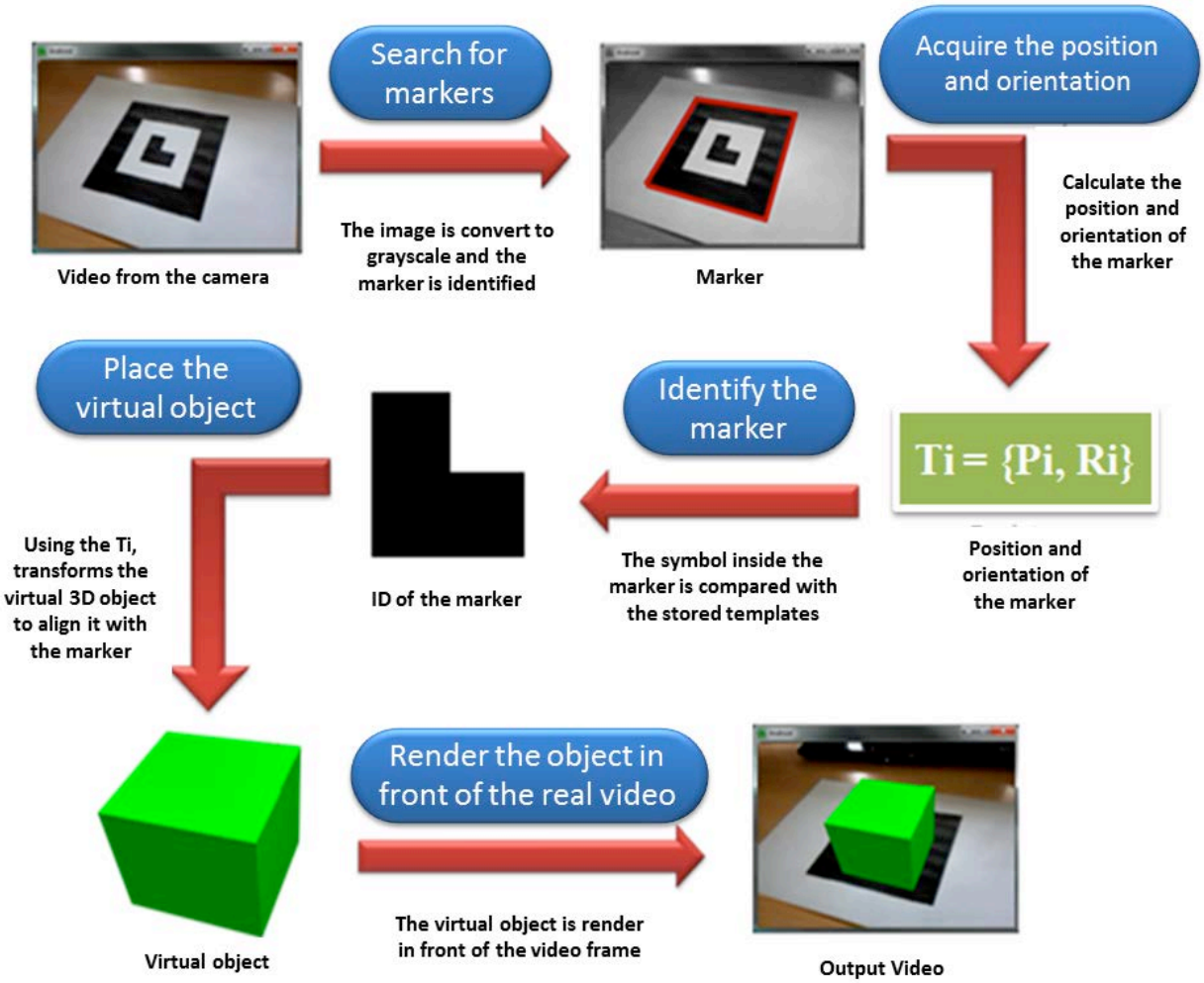

Figure 7. Operation of the ARToolKit.

4. APPLICATION DESIGNED

"TccDroid" was the name given to the app for devices with Android OS developed for this project. This app is able to read the sensor data from the mobile device (magnetic compass, accelerometer and gyroscope), apply some appropriate filters to improve the accuracy of the information, make the recognition of markers in the environment through the ARToolKit library, thus obtaining data from its position and orientation in an indoor controlled environment.

To make the application work correctly, the smartphone must be properly installed in the robotic system. The device can be oriented in such a way that can view the marks that are in front of the robotic system, in its side or even on the floor, as shown in Fig 8.

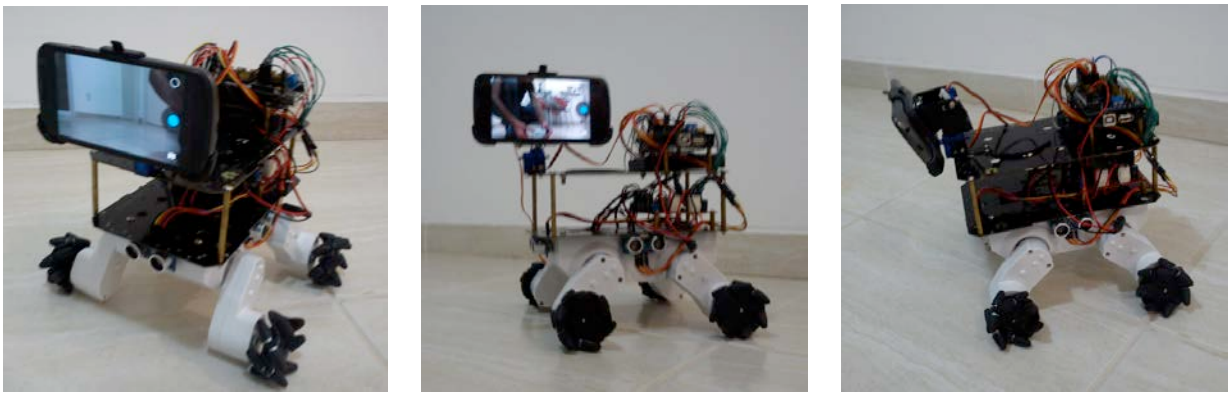

Figure 8. Position of the smartphone in the robotic system

The device turned to the front is the most usual to the standard operation. When used in applications with hallways, the best position for the smartphone is in the side. The markers can also be installed on the floor of the environment and then use the phone with the camera facing down. However, should be remembered that countless other options for positioning the mobile device and the markers are possible, leaving the developer of robotic system responsible for determining the best strategy for their application, as well as the change of coordinates from the device to the robotic system.

4.1 Development of markers

Some markers already have the patterns recorded in the TccDroid, but others can be used. For this purpose, the developer must follow some procedures (ARToolKit Marker Maker, 2012). After physically created the markers, is necessary to create the recognition ARToolKit files. Then must use the online version of the Marker Generator, (ARToolKit Marker Generator, 2012).

4.2 Information about marker recognition

From the camera the TccDroid takes 4 information: Identification of the marker; distance on the X axis between the mobile device and marker; distance in the Z axis between the mobile device and the marker; angles between the coordinates of the mobile device and the coordinates of the marker, as shown Fig. 9.

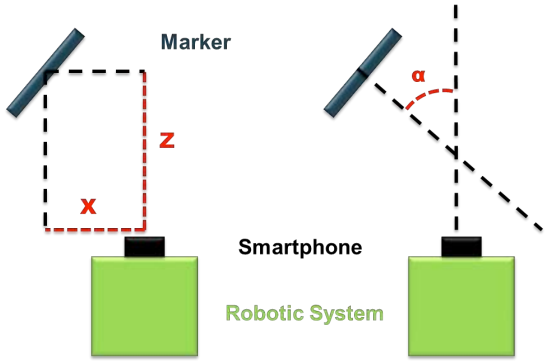

Figure 9. Schematic representation of the collected data from the camera

4.3 Information about magnetic compass

A magnetometer is an instrument used to measure the strength and/or direction of the magnetic field in the surrounding area of the instrument.

The magnetometer measures the strength of the ambient magnetic field in micro-Tesla (μT), in the x, y and z axes. To collect the data from the mobile device, were defined degrees of freedom in three-dimensional space as shown in Figure 10.

Figure 10. Definition of degrees of movement in three-dimensional space.

4.4 Information about accelerometer

An accelerometer is a device that measures the proper acceleration of the device. This is not necessarily the same as the coordinate acceleration (change of velocity of the device in space), but is rather associated with the phenomenon of weight experienced by a test mass that resides in the frame of reference of the accelerometer device

The accelerometer measures the acceleration in three axes in m/s2. It outputs the acceleration applied to the device by measuring forces applied to the sensor. The measured acceleration is always influenced by the force of the earth's gravity.

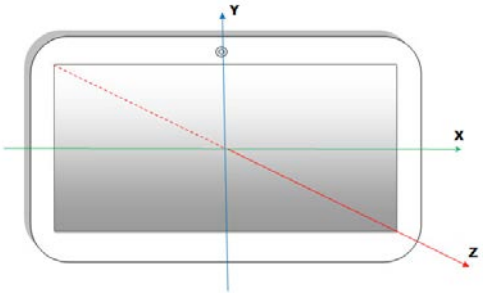

Figure 11. Definition of the orthogonal system used.

4.5 Information about gyroscope

Generally, a gyroscope is a device for measuring or maintaining orientation, based on the principles of conservation of angular momentum.

The gyroscope values are in radians/second and measure the rate of rotation around the x, y and z axis. Rotation is positive in the counter-clockwise direction.

4.6 Sensor Fusion implementation

Sensor fusion is the combining of sensory data derived from sensory data from disparate sources such that the resulting information is in some sense more accurate and robust than would be possible when these sources were used individually.

In order to provide a generic system, that can provide better information about the localization of the robotic system, was implemented a sensor fusion to the orientation data with the sensors: Accelerometer, Magnetic compass and Gyroscope. The data from landmark recognition was not included into the sensor fusion because its data is not available all the time, with a regular frequency, only when the landmark is visible. So, is responsible of the robotic

system control take actions based on the information provided by the built-in sensors and the information provided by the landmark recognition.

The Android OS provide by default a called “Orientation Sensor”, that is a software method which uses readings from the accelerometer and the magnetometer to compute the phone’s orientation based on the rotation matrix. In simple terms, the accelerometer provides the gravity vector (the vector pointing towards the center of the earth) and the magnetometer works as a compass. The Information from both sensors suffice to calculate the device’s orientation. The reference coordinate system is defined according the Fig. 12.

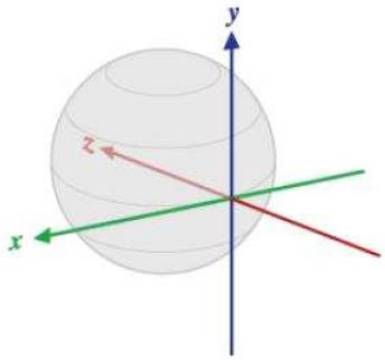

Figure 12. Reference coordinate system of orientation by the “Orientation Sensor”

However both sensor outputs are inaccurate, especially the output from the magnetic field sensor which includes a lot of noise arising from changes in the magnetic field generated by metal components, which are common indoors environments.

The gyroscope in the device is far more accurate and has a very short response time but its downside is the drift. To get the actual orientation the angular rotation speeds values need to be integrated over time. This is done by multiplying the angular speeds with the time interval between the last and the current sensor output.

$$\alpha_n = \sum_{i=0}^{n} (\omega_i * \Delta t) \tag{14}$$

During this process small errors are introduced in each iteration. These small errors add up over time resulting in a constant slow rotation of the calculated orientation, the drift. To compensate for the drift a correction is applied:

$$\omega_i = \omega_i - \omega_i * (1 - \varepsilon'_{x,y,z}) \tag{15}$$

where ε’ is the estimated error of the angular speed around each axis.

To improve the accuracy of the system was applied one complementary filter. The gyroscope output is applied only for orientation changes in short time intervals, while the magnetometer/accelerometer data is used as support information over long periods of time. This is equivalent to low-pass filtering of the accelerometer and magnetic field sensor signals and high-pass filtering of the gyroscope signals, as presented at Fig. 13.

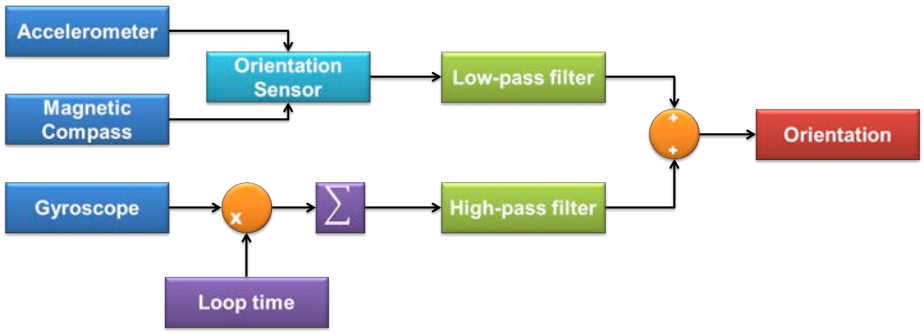

Figure 13. Complementary filter implemented on TccDroid.

An example of the routine for filter the acquire data is presented in the Fig. 14.

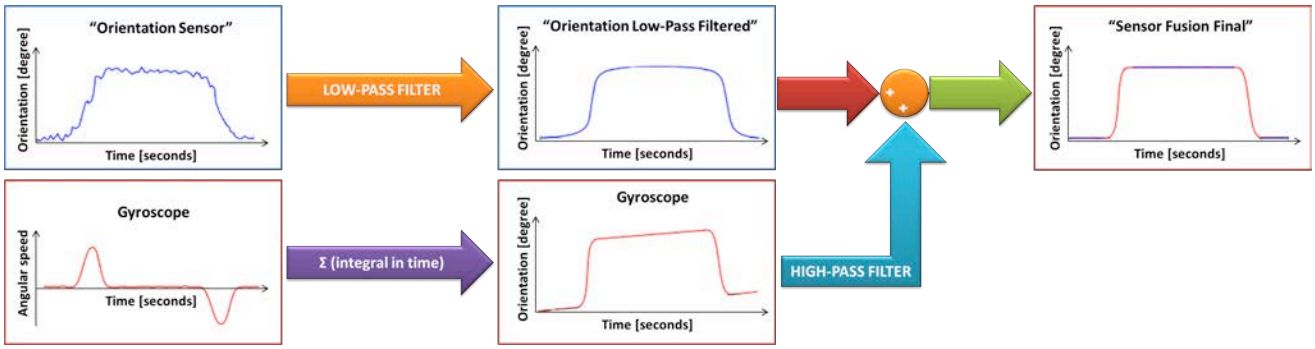

Figure 14. Example of the application of the complementary filter.

4.7 Interface and comunication

The TccDroid interface allows the user to choose which kind of response, modes and external component which wants to communicate the collected data, as presented in the Fig. 15.

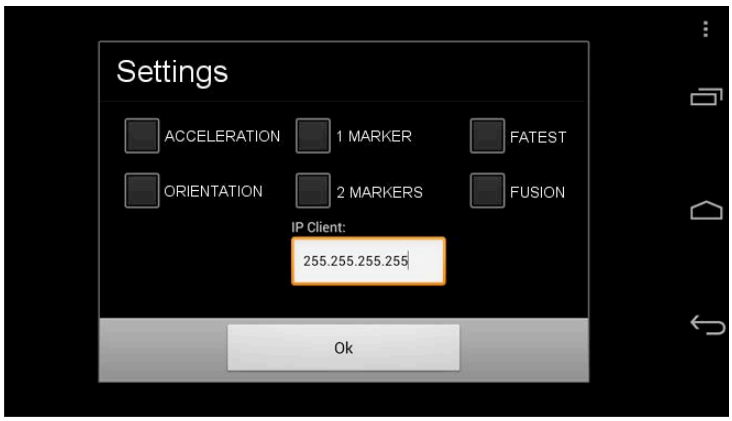

Figure 15. Initial screen of TccDroid.

The “ACCELERATION” and “ORIENTATION” buttons enables respectively the sending data from the sensors accelerometer and magnetic compass. If the “Fusion” button is enable, data from sensor fusion algorithm will be sent instead of magnetic compass results.

The “1 MARKER” button should be enabling when there is no physical possibility of the mobile device locating more than one marker at the same time. This option speeds up the processing and reduces the size of the string communication over the “2 MARKERS” button. The markers should be positioned in order to prevent the smartphone visualize over two markers simultaneously, which can lead to inconsistencies.

In Android it is not possible to choose the sampling frequency and read directly from the inertial sensors, instead a sensor event is generated every time the sensor values changes, but is allowed to set the priority of the task of reading. So if is required a high frequency the “FATEST” button should be enable.

When the “FUSION” button is enable, the TccDroid applies the complementary filter in the acquire data, then the sending data is no more from the magnetic compass output, but from the orientation filter output.

The “IP Client” field correspond to the IP from the external system, that will use the data provide by the TccDroid.

The sensor data formatted in a string to streamline and standardize the communication, as shown in Tab. 1.

Table 1. Formatting characters for sensor data.

Sensor / Response	Data	Integer Characters	Separation Character	Decimal Characters	Total of characters
Accelerometer / Acceleration	X	3	1	2	6
	Y	3	1	2	6
	Z	3	1	2	6

Magnetic Compass / Orientation	Yaw	4	1	1	6
	Pitch	4	1	1	6
	Roll	4	1	1	6
Camera	Marker ID	2	0	0	2
	X Distance	4	1	1	6
	Z Distance	4	1	1	6
	Angle	3	1	1	5

The first character of the integer characters represents the sign value, "0" if the data is positive and "-" if negative. Each data is separated from the next one by a space character. And the sensor that wasn't enable by the user is ignored by the app. Thus, can have for example the following string:

- Selected sensors:
  Accelerometer, Magnetic compass and Camera (1 marker)
- Generated string:
  "000.41 009.03 -00.18 0081.2 -009.5 0173.4 01 0938.2 -045.4 -02.1

Accelerometer Magnetic Compass Camera (Marker 1)

5. RESULTS

In order to validate the development, in first time was tested the efficiency, accuracy and reliability of the sensor fusion implemented, and some results are presented at Fig. 16 and Table 2. From this it is apparent that the sensor fusion implemented improves the accuracy of results, compared to using only the magnetic compass, decreasing the response variation and removing the gyroscope drift caused by the integration of the same in time.

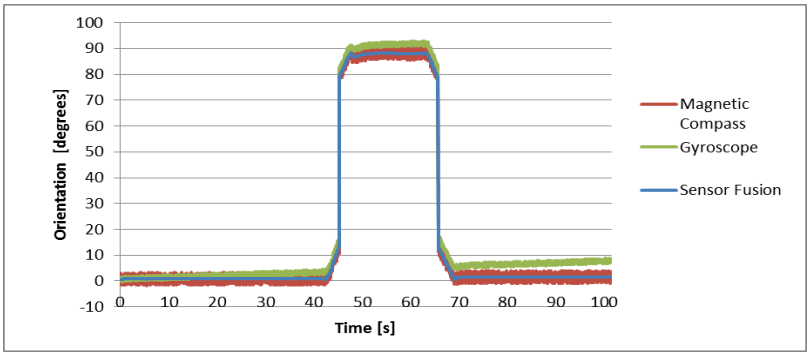

Figure 16. Test, in bench, of sensor fusion implementation

Table 2. Variation of orientation response, in static test, bench.

Sensor	Variation of orientation
Magnetic Compass	±2.39º
Gyroscope	±1.57º
Sensor Fusion	±0.71º

On a second wave of tests, was tested the application with all features, including the sensor fusion, on a robotic system. The final test, presented below, has been structured with 5 markers, allowing the robotic system to move around two corridors, as shown in Fig. 17.

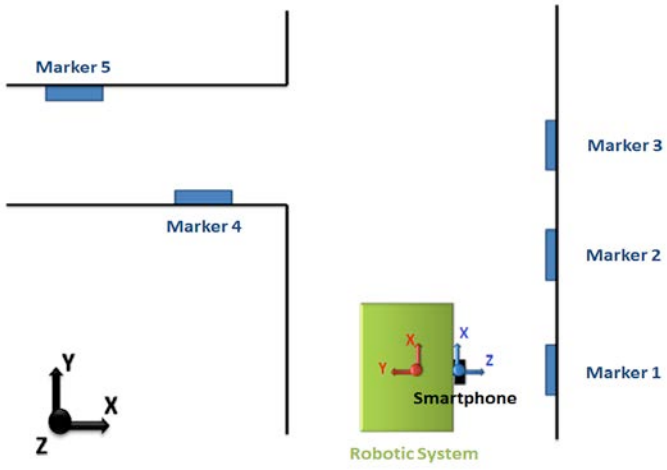

Figure 17. Schematic map of the environment for validation test.

As interaction routine, the robotic system was programmed to walk straight and perform the following commands when recognized one marker:

- Marker 1: Wait 5 seconds,
- Marker 2: Wait 5 seconds,
- Marker 3: Rotate 90 ° in a counter-clockwise direction;
- Marker 4: Stop;
- Marker 5: Rotate 180 ° in a counter-clockwise direction.

In this test was acquiring the following data:

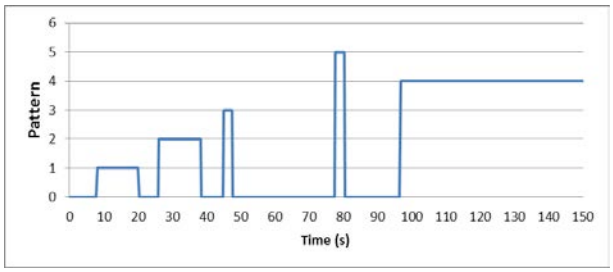

Figure 18. Graph indicating the marker viewed by the mobile device.

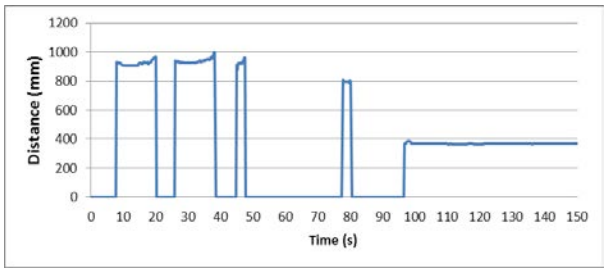

Figure 19. Graph showing the distance between the mobile device and marker.

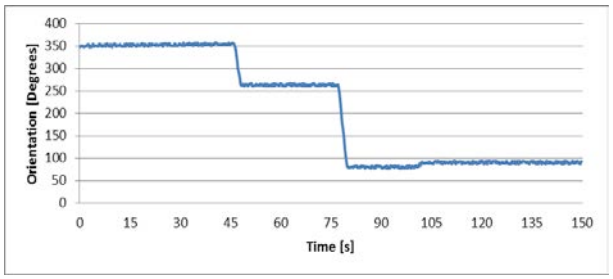

Figure 20. Orientation of the mobile device with respect to Yaw.

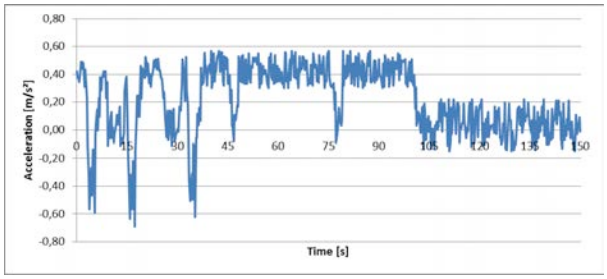

Figure 21. Acceleration of the mobile device with respect to X.

From Fig. 18, follows that the identification and distinction of the markers was extremely positive, with the quick recognition and with no confusion between markers.

From Fig. 19, follows that the determination of the distance between the marker and the mobile device is also adequate, and found the error was smaller than expected, even at a distance greater than the maximum recommended by the developer of ARToolKit

The Fig. 20 and Fig 21. prove that the accelerometer and the magnetic compass generate favorable results to assist an autonomous navigation, but it is necessary to examine the degree of precision required by the robotic system.

From this study should be scored as caveats the determination of the angle between the marker coordinates and the coordinates of the mobile device, which was excessively noisy and the method of administration of the magnetic compass, which can receive noise from metallic material commons in indoors environments.

Finally, can be considered achieved the main objective of this project: the validation of a mobile device provided with the Android operating system as a sensor applicable to autonomous navigation of robotic systems.

6. CONCLUSIONS AND FUTURE PRESPECTIVE

The main aim that characterize this paper, validation of a mobile device powered by an Android OS as a sensor applicable to autonomous robotic system, can be considered reached, because:

- The localization, identification, distinction and extraction of information from the markers was extremely positive, once that did not occur confusion between markers and the extracted data contained an error acceptable to indoor environments;
- The data from built-in sensors is also reliable and accurate to indoor environments, but some care must be taken with regard to the presence of iron magnetic materials near the device;
- Is remarkable that constant movements tend to generate better results, with less noise. So, should be prioritize strategies of movement with rectilinear motion and constant velocity or acceleration. Should also be careful with the attachment on the robot, not too rigid fixation may generate noise to the sensors.

- The application of sensor fusion is fundamental to increase the accuracy and reliability of the system. In this development was utilized a complementary filter, but other approaches and techniques, more robust, can be studied.

There are a number of different aspects of further developing to this project. The most significant are:

- Improve the accuracy of the acceleration data, through statistical filters;
- Implement a model with Kalman filter to integrate the different information sensors and then improve the accuracy of the position estimation;
- Implement a mapping and location of the magnetic field generated by the ferromagnetic materials presents in the environment and capture by the magnetic compass;
- Utilize other types of sensor, as encoder on the wheels of the robot, or methods, as triangulation of wireless signals;
- Utilize the mobile device to process all the data from the sensors and from the robot, transferring all the control of the robot to the device, removing the need for a processor to the robotic system.

7. REFERENCES

AndAR. (2012). Android Augmented Reality. Available in <http://code.google.com/p/andar/>. Accessed at Mach 20, 2012.

ARToolKit Marker Generator Online. (2012). Available in <http://flash.tarotaro.org/blog/2008/12/14/artoolkit-marker-generator-online-released/>. Accessed at March 26, 2012.

ARToolKit Marker Maker. (2012). Available in <http://www.roarmot.co.nz/ar/>. Accessed at March 26, 2012.

Developer SDK - Android. (2012). Available in <http://developer.android.com/sdk/index.html>. Accessed at March 19, 2012.

FIGUEIREDO M., 1997, “Redes neurais nebulosas aplicadas em problemas de modelagem e controle autônomo”, tese de doutorado, Faculdade de Engenharia Elétrica e de Computação, Unicamp, Campinas, SP.

THRUN, S; BURGARD, W. & Fox, D. (2005). Probabilistic Robotic. The MIT Press.

KATO, H. (1999). Marker Tracking and HDM Callibration for a Video-based Augmented Reality Conferencing System. Proceedings of the 2nd IEEE and ACM Internacional Workshop on Augmented Reality.

SILVA, L. G. D. (2012). Sistema de auxílio a navegação autônoma baseado em uma plataforma Android. 102p. Trabalho de Conclusão de Curso. Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2012