IMU (Inertial Measurement Unit) Integration for the Navigation and Positioning of Autonomous Robot Systems

Alexandru Barnea*, Sid Ahmed Berrabah**
Cezar Oprisan*, Ioan Doroftei*

**"Gheorghe Asachi" Technical University of Iasi, Faculty of Mechanical Engineering, B-dul Prof.dr.docent Dimitrie Mangeron no.61-63, Iasi 700050, Romania (e-mail: alexandru.barnea@gmail.com).*
***Royal Military Academy of Brussels, Mechanical Department, Avenue de la Renaissance 30, 1000 Brussels, Belgium (email:sidahmed.berrabah@rma.ac.be)*

Abstract: An autonomous robot needs to be able to position itself in a given surrounding environment and to be able to accomplish a given navigational goal. This paper presents the integration of an Inertial Measurement Unit for the navigation and localization of an autonomous robot system. A generic control architecture is used for the CoRoBa architecture, using servant object component interfaces like *Sensor, Processor,* and *Actuator,* inherited from the Service interface abstraction based on the TAO implementation of the communication middleware CORBA. Different data integration modes were overviewed using several Kalman Filter implementations. The integration process for the Microstrain 3DM-GX2 IMU and the robot localization were designed for the ROBUDEM robot platform, and implemented as CoRoBa *sensor* and *processor* modules.

Keywords: autonomous, IMU, localization, CORBA middleware**,** navigation

1. INTRODUCTION

Autonomous robots are mobile robots which can perform desired tasks in unstructured environments without continuous human guidance. Many kinds of robots have some degree of autonomy. Different robots can be autonomous in different ways. A high degree of autonomy is particularly desirable in fields such as space exploration, major incidents or natural cataclysm area exploration, where communication delays and interruptions are unavoidable or GPS outage appears.

An autonomous robot needs to be able to position itself in the surrounding environment and accomplish the given navigational goal.

It is very important for the autonomous robot navigation and positioning systems to be able to provide accurate continuous data, based on previous and actual information gathered from the surrounding environment.

1.1 The necessity of different sensors on a mobile robot platform

The mobile robot, as an autonomous robot, needs a lot of information about the surrounding environment. The surrounding environment can be "discovered" from a multiple point of view. Each point of view is gathered using a specific sensor, which only obtains a partial view of the real world, and reacts to a certain stimulus from it.

Different sensors provide different kinds of information, which should be fused together in order to obtain a complete picture of the real world. More specifically, multi-sensor data fusion aims to overcome the limitations of individual sensors and produce accurate, robust and reliable estimate of the world state based on multi-sensory information.

It has been a big challenging task for robotics researchers to develop different algorithms used to interpret these sensory data before they can be used in control and navigation tasks of mobile robots. To obtain a more accurate estimation of the surrounding environment, it is preferred to have a more complex set of sensorial data, with smaller observation noise.

An autonomous robot needs to be able to position itself in the surrounding environment and accomplish the given navigational goal. Most of the robots positioning systems are based on the Global Positioning System (GPS), Inertial Measurement Units (IMU) and robot body odometry.

Sensing the variation of the outside elements and data gathering using specialized sensors is essential for acquiring a good knowledge and "understanding" of the surrounding environment, and allowing an autonomous robot to position itself and proceed for a successful navigation.

1.2 The sensing and the reasoning

To reach a reasonable degree of autonomy, two basic requirements are needed: sensing and reasoning. Sensing is provided by an on board sensorial system, that gathers information about the robot itself, as an entity, about the subcomponents and the surrounding environment.

The onboard sensorial system can be divided into several autonomous systems: IMU system, odometry system based on the wheels encoders, DGPS system, stereo vision camera framegrabber system and the sonar system.

Each autonomous robot system should be able to auto localize in the surrounding environment. The localization process assumes that the robot has access to a Global Interactive System (GIS) map or to a predefined set of maps for a specific area. By using the sensorial system, the robot should be able to rebuild a map of the environment and also localize himself, by constructing a minimalistic description of the environment. This process is known as Simultaneous Localization and Mapping (SLAM). According to the environment state and the provided goals, the reasoning system must allow the robot to localize itself in the environment and seek for free possible paths to fulfil the goal assignment.

Both the sensing and the reasoning systems of the autonomous robot are adapted for the ROBUDEM robot and implemented as CoRoBa (COntrolling RObots with corBA) modules, in a distributed framework for integrating and controlling multi-sensor robotic systems.

2. THE CONTROL ARCHITECTURE OF THE AUTONOMOUS ROBOT ROBUDEM

The control architecture describes the strategy of combining the three main capabilities of an autonomous robot: the sensing, the reasoning and the actuation. These three capabilities have to be integrated in a coherent framework in order to accomplish certain tasks adequately.

2.1 The generic control architecture

The generic control architecture of the autonomous robot is divided into software architecture and hardware architecture. The control architecture has to be translated into a software architecture which manages the building blocks on a software level. This software architecture has to provide the flexibility of modular design while retaining a thorough structure, enabling an easy design process. All the different processes (sensor measurements, measurement processing, sensor fusion, map building, path planning, and task execution) have to be coordinated and interrelated in an efficient manner in order to allow the accomplishment of a higher goal.

The generic architecture has several key components such as sensors, processors and actuators: Sensors (*IMU*, DGPS, wheel encoders used for Odometry, Ultrasonic sensors), Position and Mapping Processors (Position Estimation, Visual Simultaneous Localization and Mapping, Global Path Planner), Behavioural Processors (Maximise Terrain Knowledge, Avoid Obstacles using SLAM, Stereo Vision and Sonar, Go To Goals). The generic control architecture is detailed in (Fig.1).

The output generated by the sensors must be processed by a processor or by a set of processors. All the processed information from the sensors is fused into a Behaviour Processor (Fuse Behaviours), from which the fused information is filtered in order to steer the RobuDem robot using a steering actuator (Robot Steering).

2.2 CoRoBa framework

CoRoBa, the distributed framework used, is based on the TAO implementation of the communication middleware CORBA. The implementation of the framework is based on several design patterns that make the design flexible, elegant and ultimately reusable (Colon 2006).

The elementary brick in CoRoBA is a component (Component-based Architecture Pattern) that exchanges data over a network, by invoking remote operations (Remote Method Call Pattern). Components are independent execution units and have separated interfaces for the configuration and the actual functionality they provide (Hierarchical Control Pattern). Components are loosely coupled, being connected only by the structure of the data they exchange (Data Bus Pattern), and can be discovered at run-time (Broker Pattern). According to the classical control theory, components are divided in three categories, *Sensors*, *Processors* and *Actuators*. They form a chain along which information is transferred via Notification Channels (Event Channels) and like in classic control schemes, the data flow is unidirectional (Channel Architecture Pattern)(Fig. 1).

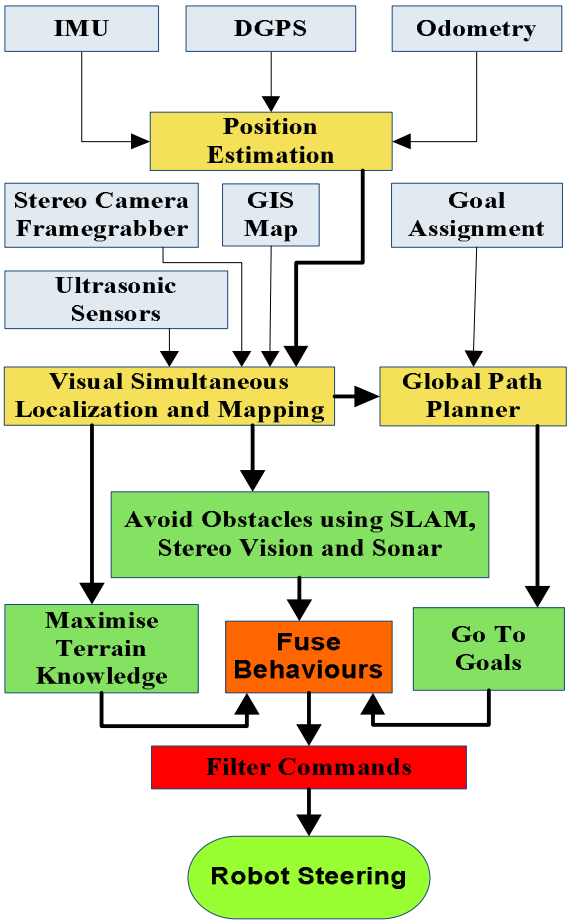

Fig. 1. Generic control architecture of the autonomous robot ROBUDEM

2.3 Data exchange between the basic interfaces

The base interface in CoRoBa code is organized in two parallel hierarchies: the IDL interfaces and the CoRoBa classes implementing these interfaces. The *Service* interface represents the base interface, all other interfaces being derived from it. The operations defined by the *Service*

interface are implemented by a class named RMA_Service_i, which is inherited by all the components. There are three basic interfaces: *Sensor, Processor,* and *Actuator* that inherit from the Service interface. The classes implementing these interfaces are considered as abstract and may not be instantiated. In CoRoBa all servant objects (sensors, processors and actuators) are derived classes of the class *RMA_Service_i*. For the handling of different management operations like: *start(), pause(), wakeup(), stop(),* and also for the interaction with the *Service* interface, a special component called Component Remote Control (CRC) was used. The *Sensor* and the *Actuator* components manage the link between the *Processor* components and the physical elements mounted on the hardware platform. The *Sensors* are connected to physical sensorial systems and retrieve information that is forwarded to the processing components.

Inside the CoRoBa framework, three different running modes are possible for the transmission of events between *Sensors, Processors,* and *Actuators*: PERIODIC, SYNCHRO and TRIGGER. The specific running modes for each basic interface are detailed in Table 1. (Colon 2006).

Table 1. Actions performed by each interface on the specific running modes

Running mode	Sensor	Processor	Actuator
SYNCHRO	Push an event each time new hardware sensor values are available	Processes the data and pushes events each time a new event is received	Sends data to an external system each time a new event is received
PERIODIC	Reads the sensor values and pushes events at given periodic intervals	Processes the data and pushes events at periodic intervals	Sends data periodically to an external system
TRIGGER	Reads hw. Sensor values and pushes events each time it is externally triggered	Processes the data and pushes external events when it is externally triggered	Sends data to an external system only when it is externally triggered

3. INERTIAL SYSTEMS INTEGRATION FOR THE NAVIGATION AND LOCALIZATION OF AUTONOMOUS ROBOT SYSTEMS

3.1 Localization and posing

The localization of an autonomous robot system refers mainly to the precise determination of the coordinates where the system is present at a certain moment of time. The localization problem for autonomous robot systems can be divided into two sub-tasks: global and local localization. In many applications, the orientation and an initial estimation of the robot position are known, being supplied directly or indirectly by the user or the supervisor. During the execution of the tasks, the robot must update this estimation using measurements from its sensors. This is known as local localization (Jensfelt P. (2001)).

Using only sensors that measure relative movements, the error in the pose estimation increases over time as errors are accumulated. Therefore external sensors are needed to provide information about the absolute pose of the robot. This is achieved by matching the sensors measurements with a model of the environment.

On the other hand, global pose estimation (Se et al, 2002), is the ability to determine the robot's pose in an a priori or previously learned map, given no other information than that the robot is somewhere on the map. Global localization is considerably more difficult than pose tracking because of the data association problems.

3.2 Classification of the inertial systems

Inertial systems can be classified into three different types. This classification could cause certain confusion. The less complex system type, providing raw data like acceleration and angular velocity from the inertial sensors, is the Inertial Sensor Assembly (ISA). Generally, ISA consists of Accelerometers and Gyroscopes that output uncompensated information. Usually the ISA might contain an temperature sensor, used at a higher level by the IMU for the error compensation. The Inertial Measurement Unit (IMU), receives the raw data from the ISA and manages the post processing of the data for error compensations like bias scale factors. To be able to determine position, velocity and attitude, the compensated acceleration and angular rotation are processed using different navigation algorithms. The entire process is called Inertial Navigation Systems (INS). The interdependence between INS, IMU and ISA can be observed in the schematic presented below (Fig. 2).

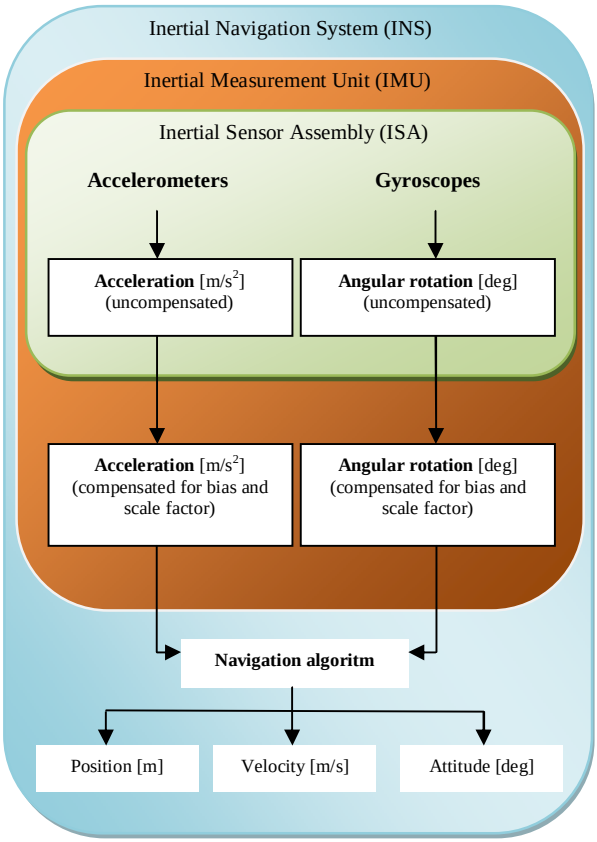

Fig. 2. Differences between ISA, IMU and INS

3.3 Data integration for robot localization

In order to increase the accuracy of the robot positioning, a Kalman Filter is used to integrate the data received from the DGPS system with the data from the Inertial Navigation System (INS) and from the robot encoders. This integration will allow also maintaining the robot positioning even if no satellites are visible or the data received from the satellites is corrupted.

The Kalman Filter is an extremely effective and versatile procedure for combining noisy sensor outputs, and to estimate the state of a system with uncertain dynamics. In the case of GPS, INS, and wheel Encoders integration, the output given by the noisy sensors includes the GPS receivers, INS and Wheel Encoder components.

The system state includes the position, velocity, acceleration, attitude, and attitude rate of a vehicle. Uncertain dynamics include different unpredictable disturbances generated either by the host autonomous robot, or by the unpredictable changes that may appear internally, in the sensor parameters or structure (eg. the sensors temperature dependence). The position, velocity, and attitude errors, as well as errors in the inertial and GPS measurements are optimally estimated by the Kalman filters.

The data output given by the GPS receivers (usually at 1 Hz) is relatively low and might not fulfil the accuracy requirements necessary for autonomous robots positioning, where in some cases the accuracy of less than 5 centimetres is requested. The problem is more serious and different other options should be taken into account in situations like: a potential temporary loss of GPS signal, slips occurrence due to uncertain terrain, or controlled electromagnetic jamming that might affect the functionality of the robot internal sensors.

Using the INS output information, the autonomous robot can calculate the dynamics of motion between different GPS epochs, at a high temporal resolution and also complements the discrete nature of GPS and wheel encoders when a cycle slips or signal loss occurs. Additionally, the positioning with INS requires the integration with respect to time of the accelerations and the angular rates, the measurement noise accumulates and resulting in long wavelength errors. Opposite to INS, the GPS errors do not accumulate, but they get relatively larger in short term and the recorded measurements are characterised by a poorer resolution.

3.4 Data integration modes overview

The types of integration can be categorized by the extent to which data from each component aid the other's function. First one is the coupling of the systems and depends on the architecture of the system. The second one refers to the category of parameters used by the method of combining or fusing the data to obtain position coordinates.

The system architecture is generally understood in two ways, tight coupling and loosely coupling; where no coupling implies no data feedback from either instruments to the other, for the purpose of improving its performance.

Tightly coupled sensors are treated as belonging to a single system producing complementary types of data. The obtained data are produced simultaneously and optimally, and used to enhance the function of individual sensor components where possible. In a loosely coupled system, processed data from one instrument are fed back in an aiding capacity to improve the utility of the other's performance, but each instrument still has its own individual data processing algorithm (Kocaman 2003).

Inertial navigation improves if the GPS solution functions as an update in a Kalman filter estimation of the systematic errors in the inertial sensors. Similarly, GPS positions and velocities may be used to aid the INS solution in a high-dynamic situation by providing a better reference for propagating error states based on the linear approximation (Jekeli, 2000).

There are two basic categories of processing algorithms that are centralized and de-centralized. In centralized processing, the raw sensor data are combined optimally using one central processor to obtain a position solution.

This kind of processing is usually associated with tight system integration. Decentralized processing is a sequential approach to processing, where processors of individual systems provide solutions that subsequently are combined with various degrees of optimality by a master processor. In principle, if the statistics of the errors are correctly propagated, the optimal decentralized and centralized methods should yield identical solutions (Jekeli, 2000).

The centralized approach provides the best performance for the navigation solutions than a single robust Kalman filter model. Different forms of integration are evaluated in Table 2.

Table 2. Different forms of the Kalman Filter (KF) implementation

Implementation	Advantages	Disadvantages
Open loop	KF may be run externally, suitable for navigation based on INS	Non-linear error model due to large second-order effect, needs an EKF
Closed loop	For the INS, the linear model is sufficient. Suitable for software integration	Requires more complex processing. GPS errors may affect INS performance
Loosely-coupled (cascade, decentralised)	Flexible and modular combination, small KF with faster processing, for parallel processing	Sub-optimal performance, unrealistic covariance, minimum four satellites needed for a stable solution. INS data is not used for ambiguity estimation
Tightly-coupled (centralised)	Optimal solution with one error state model, GPS measurements can be used with less than 4 satellites, direct INS aiding throughout GPS outages, faster ambiguity estimation	Large size of the eror state model, with more complex processing requested

Integrated systems can provide a more complex system, with superior performances in comparison with either an DGPS, an INS or an Odometry system. A comparison regarding the main strengths and weakness of INS and DGPS systems versus an integrated INS and DGPS system is summarized in Table 3. (Skaloud 1999 and Kocaman 2003).

Table 3. Summarised comparison between INS, DGPS and integrated INS and DGPS systems

INS	DGPS	INS and DGPS
high position and velocity accuracy over the short term	high position and velocity accuracy over the long term	high position and velocity accuracy
accurate attitude information	noisy attitude information (due to multiple antenna arrays)	precise attitude determination
accuracy decreasing with time	uniform accuracy, independent of time	accuracy decreasing with time during long GPS signal outages
high measurement output rate	low measurement output rate	high data output rate
autonomous	non-autonomous	navigational output during GPS signal outages
no signal outages	cycle slip and loss of lock	cycle slip detection and correction
affected by gravity	not sensitive to gravity	gravity vector determination

In the situations when a long temporary loss of the GPS signal appears, the localization update, and also the positioning and the navigation based only on the INS system can lead to a inaccurate solution. Therefore, an additional correction of the localization data is recommended, using the wheel encoders data output, combined with a frequent bias recalculation based on the temperature fluctuations recorded by the onboard temperature sensor of the ISA. For this recalculation of the bias, the temperature dependencies given by the manufacturer could be used.

4. KALMAN FILTER MODELLING AND IMU INTEGRATION IN THE CoRoBa FRAMEWORK

The IMU integration for the navigation and the positioning of an autonomous robot system is applied to a mobile car-like robot named "ROBUDEM", robot that travels through the environment using its sensors for self localization. The IMU is accessible to the CoRoBa framework via a basic *interface* named *Sensor.*

A sensor interface named “RMA_Sensor_GX2_i” was developed to allow the data aquisition from the externally connected Microstrain 3DM-GX2 IMU sensor. All the three running modes are available for this sensor, and furthermore, due to the internal microprocessor of the 3DM-GX2, complex data acquisition can reach up to 110 groups of samples per second. The hardware sensor values are read by the CoRoBa *Sensor* and released in the Event Channel as pushed events. A *processor* interface named “Position Estimation” should receive the event and process the information. In parallel, the processor listens for other events generated by the DGPS *sensor* or the Odometry *sensor*. According to the information received at a specific moment of time, the *processor* launches into the event channel specific information necessary to the *actuators* to accomplish their purpose.

For the navigation and the localization of the ROBUDEM autonomous robot, two coordinate frames can be assumed: a world coordinate frame **W** and a robot coordinate frame **R**. The **W** coordinate frame is defined such that the X and Z axes are on the ground plane, and the Y axis points vertically upwards. The ROBUDEM robot system state vector $\boldsymbol{y_R}$ is defined in this case as (1), with the 3D position vector $\boldsymbol{r = (y_Z, y_X, y_Y)}$ in the gravity centre of the robot (supposed in the middle of the axis between the rear wheels), in the world frame coordinates **W**. The roll, pitch and yaw robot orientations correspond to the rotations around the Z, X, and Y axes, respectively (γ,θ,φ).

$$y_R = \begin{bmatrix} y_Z \\ y_X \\ y_Y \\ g \\ q \\ j \end{bmatrix} \tag{1}$$

The motion model is obtained from the relationship between the past state of the robot, noted $\mathbf{y_R^{t-1}}$, and the current state of the robot, $\mathbf{y_R^t}$, being given a control input $\mathbf{u^t}$ and , $\mathbf{v^t}$ as a random vector describing unmodelled aspects of the robot (eg. Process noise such as wheel sleep or odometry errors):

$$\mathbf{y}_{\mathbf{R}}^{\mathbf{t}} = \mathbf{f}\left(\mathbf{y}_{\mathbf{R}}^{\mathbf{t-1}}, \mathbf{u}^{\mathbf{t}}, \mathbf{v}^{\mathbf{t}}\right) \tag{2}$$

where $\boldsymbol{f}$ is a function representing the mobility, kinematics and dynamics of the robot (transition function). Considering the control $\boldsymbol{u}$ as identity, the dynamic system model, in our case, is given by (3):

$$y_R^t = \begin{bmatrix} y_Z^t \\ y_X^t \\ y_Y^t \\ g^t \\ q^t \\ j^t \end{bmatrix} = \begin{bmatrix} y_Z^{t-1} + (v^{t-1} + V)\cos(g^{t-1})\Delta t \\ y_X^t + (v^{t-1} + V)\sin(g^{t-1})\Delta t \\ y_Y^{t-1} \\ g^{t-1} + (w^{t-1} + \Omega)\Delta t \\ q^{t-1} \\ j^{t-1} \end{bmatrix} \tag{3}$$

where $\boldsymbol{v}$ and ω are the linear and the angular velocities respectively, and $\boldsymbol{V}$ and Ω represent the Gaussian distributed perturbations to the linear and angular velocity of the robot.

The Kalman Filter maintains the state vector $\boldsymbol{y_R}$ based on sensors measurements, and the covariance matrix $\boldsymbol{P}$, which includes the uncertainties from the various states as well as correlations between the states. For each time stamp of the filter, we obtain the predicted state $\boldsymbol{y_R}$ and the covariance $\boldsymbol{P}$ using the state transition function (4)

$$\begin{aligned} \mathbf{y}_{\mathbf{R}}^{\mathbf{t}|\mathbf{t-1}} &= \mathbf{f}\left(\mathbf{y}_{\mathbf{R}}^{\mathbf{t-1}|\mathbf{t-1}}, \mathbf{u}^{\mathbf{t}}, \mathbf{v}^{\mathbf{t}}\right) \\ \mathbf{P}^{t|t-1} &= FP^{\,t-1|t-1}F^{\,t} + Q^{\,t-1} \end{aligned} \tag{4}$$

where:

$$F = \frac{\partial f}{\partial y_R}\Big| y_{\mathbf{R}}^{t-1|t-1} \quad (5)$$

represents the Jacobian of $\boldsymbol{f}$ with respect to the state vector $y_{\mathbf{R}}$ and $\boldsymbol{Q}$ is the process noise covariance. Considering a constant velocity model for a smooth robot motion, $\boldsymbol{F}$ becomes (6):

$$\frac{\partial f}{\partial y_R}\Big| y_{\mathbf{R}}^{t-1|t-1} = \begin{bmatrix} 1 & 0 & -\sin\left(g^{t-1}\right)\left(v^{t-1}+V\right)\Delta t \\ 0 & 1 & \cos\left(g^{t-1}\right)\left(v^{t-1}+V\right)\Delta t \\ 0 & 0 & 1 \end{bmatrix} \quad (6)$$

The measurement model is needed for the Kalman Filter in order to relate the observation to unknown parameters.

5. CONCLUSIONS

In this paper, an Inertial Measurement Unit integration method for the navigation and the localization of an autonomous robot system is presented. The developed CoRoBa interfaces and the presented methods show that for a more accurate navigation and localization of the robot, supplementary information are necessary, for example wheel encoders and sonar data, as well as access to a set of GIS maps of an given environment. As all of the experiments were done indoor and partially on a simulated environment, future work aims for the usage of the robot with the CoRoBa framework in outdoor different real-time situations, such that the robot could gather environmental information and also react in real-time to the dynamic changes of the environment. Future research will enable and allow the robot to navigate autonomously and perform self reasoning on a 3D model of the environment, by the integration of map-building, map-updating and path-planning techniques.

ACKNOWLEDGEMENT

This paper was realised with the support of POSDRU CUANTUMDOC "DOCTORAL STUDIES FOR EUROPEAN PERFORMANCES IN RESEARCH AND INOVATION" ID79407 project funded by the European Social Found and Romanian Government.

REFERENCES

Colon E. (2006). *CoRoBa, a multi mobile robot control and simulation framework*, PhD thesis, Vrije Universiteit Brussel - Koninklijke Militaire School, Brussels, Belgium.

Jensfelt P. (2001). *Approches to mobile robot localization in indoor environment*, PhD thesis, Department of Signals, Sensors and Systems, Royal Institute of Technology, Stockholm, Sweden.

Kocaman S. (2003). *GPS and INS Integration with Kalman Filtering for Direct Georeferencing of Airborne Imagery*, Geodetic seminar report, Institute of Geodesy and Photogrammetry, Zurich, Switzerland.

Li L., Wang FY. (2007), *Advanced Motion Control and Sensing for Intelligent Vehicles*, Springer, New York, USA.

Se S., Lowe D.G. and Little J.(2002), *Global localization using distinctive visual features*, International Conference on Intelligent Robots and Systems, IROS, pp.226-231, Lausanne, Switzerland.

Skaloud J. (1999). *Optimizing Georeferencing of Airborne Survey Systems by INS/DGPS*, Ph.D. Thesis, Dept. of Geomatics Engineering, The University of Calgary, Calgary, Alberta, Canada