

The RAS Project

A Preliminary Study of a Surface Autonomous Robot

Andrea Carbone¹ and Giorgio Ugazio² and Paolo Fichera³

Abstract. The goal of this research, carried out by the University of Rome and ENEA, is to enhance an existing experimental Antarctica mobile platform with autonomous driving capabilities. The mobile robot (a snowcat) is already equipped with a remote driving system. The problems addressed in the realization of an intelligent AGV⁴ are strongly related to difficult outdoor environment and weather conditions which can affect the consistency of collected data. This preliminary report introduces all system components and the problems we are facing to tune and interpret signals coming from sensors which modelling will serve as a basis to the high level navigation level.

1 Introduction

The RAS Project has started at the end of 1992 in the frame of the "Italian Antarctic Research Program" (PNRA), with two main purposes:

- to provide the antarctic expeditions with a useful support for all the scientific and logistic activities during different weather conditions and/or hazardous regions for human operators;
- research and develop a set of technologies that can be applied in industrial sectors with a special spot on automotive industry and hi-tech autonomous service vehicles.

Therefore, the research areas that we are developing refers mainly to autonomous drive technologies, artificial vision, teleoperation, sensor integration and relative techniques of world modelling.

2 A Portrait of the Vehicle

2.1 The Platform

The RAS is realized upon a commercial vehicle platform, for instance the Kassboher *Pisten Bully* 260 Snowcat (see Figure 1). Such model is equipped to sustain the climatic condition of Antarctic Continent. The vehicle has been furtherly improved into the plants of Oto Breda located in La Spezia, Italy. These works consisted in:

- Interfacing the electro-mechanical parts with the Control System;
- Installation of the on-board computers which host the control routines;
- Some sensory subsystems.



Figure 1. The ENEA Kassboher PB260 snowcat.

2.2 Equipment

We now outline the equipment available on the RAS (without specifying coordination and fusion policies):

- Artificial Vision subsystem, composed by a couple of CCD cameras;
- A Laser Range Finder. Provides a depth scan of the surrounding environment;
- un sensore Range Finder Radar basato su una sorgente ad onde millimetriche che assicura una funzionalit di livello inferiore in termini di risoluzione rispetto al sensore Laser, ma che opera in condizioni ambientali pi ardue in cui la difficult di propagazione del segnale elettromagnetico rende inservibile il Laser;
- A GPR sensor (Ground Penetrating Radar). Detect crevasses and discontinuities in the superficial terrain layers;
- A high precision GPS OEM;
- Odometric speed sensors;
- Inertial platform device;
- Advanced tele-operation (supervision and/or drive) interface (MMI - Man Machine Interface).

2.3 Overall Architecture

From an architectural point of view, all subsystems hosted on the RAS can be decomposed into four abstraction layers:

¹ Universita degli studi di Roma "La Sapienza", ALCOR group, DIS. Email: carbone@dis.uniroma1.it

² Universita degli studi di Roma "La Sapienza", ALCOR group, DIS. Email: ugazio@dis.uniroma1.it

³ ENEA/FUS-ROB, CR Casaccia (Roma). Email: paolo.fichera@casaccia.enea.it

⁴ Autonomous Ground Vehicles

- Locomotion which collect all actuators and relative servo-controllers
- Pilot, implementing all low level routines such as the remote driving routines, speed control drive, motion primitives (linear and curve local paths). The Reactive module is distributed among the Pilot module and the Drive module.
- Drive, deals with the local navigation system (i.e. coordination and fusion among different behaviours). It sense the world via environmental sensors (radar, laser range finder) and a strict dialog with the local mapping task (presented in section 5)
- Navigator/Supervisor. In this module will reside the global mission planning and coordination. For example it must be able to drive the snowcat to a predefined location once the communication with the remote base has been lost.

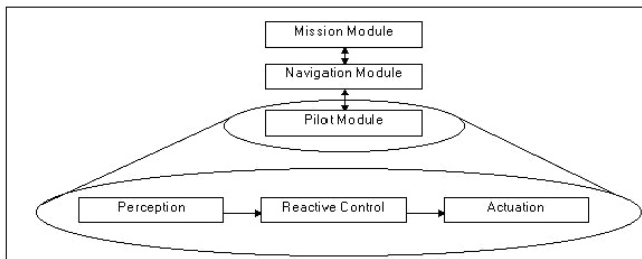


Figure 2. A sketch of

Two further blocks can be added to the ones listed above:

- MMI
- World representation.

3 Domain Specific Problems

The problems addressed in the RAS automation system fall within the general AGV automation category. These problems can be summarized as follows:

- Difficult terrain condition (slippage, trees, stones, slopes, skidding etc);
- Extremely different light and weather conditions that may affect the reliability of the vision devices, communication systems and other sensory data as the Laser Range Finder (see section 3.1);
- Poor confidence on perceived natural landmarks on the territory (in Antarctica a violent snow-storm can delete previous detected signs or tracks);
- normally high precision sensory devices can lose confidence depending on external factors. For example the GPS could go "blind" for 15 seconds before the re-establishing of a new set of satellites.

3.1 Interpreting signals

In the following we see in short the difficulties met to interpret the readings of the laser scans in a snowy environment and the approaches we are investigating to deal with such issue.

Here we recall in short the principles of functioning of a laser range finder: the phase shift of an amplitude-modulated laser signal can be measured and used to compute the distance of objects/obstacles spanned by the beam. The measurements taken have

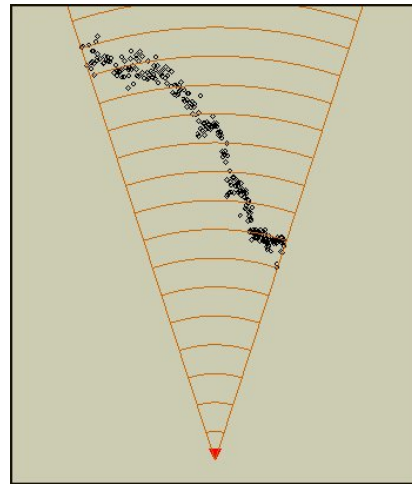


Figure 3. A clean Laser Scan transmitted by the onboard system to the Pilot module.

an extreme precision over a wide range of distances (36 meters in our case). This system anyway could lead to potential errors. In fact every 36 mt the phase returns to zero leading to a very short distance in place of the real one that is 36 meters longer. The first example shows the result of a laser scan taken from a measurement campaign on the Alps. The RAS was moving at the speed of about 20 Km/h over a snowy slope. Here the footprints of the returns form a clear design of what RAS sees frontally. In Figure 4 we can have an idea of how a tree is perceived by the laser. The laser returns are clustered around

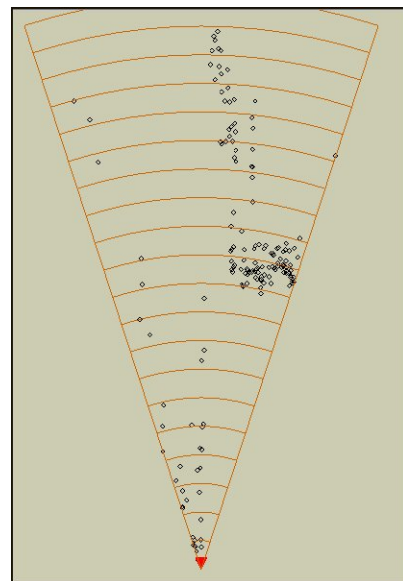


Figure 4. A tree as seen by the laser.

the object. As the tree is not a rigid body, the pings take the shape of a number of points. Some of them passing through the branches return distances from behind the tree. The nearest returns showed are far points (more than 36 meters away) that shows how a single scan can bring consistent data mixed with false returns. Filtering the laser scans represent a challenging issue: in fact every single measure has at least two corresponding candidates. The analysis leads us to or-

ganize a short-term memory of the past scans. Having an history of every single scan can give a chance to distinguish between false laser returns and true and confident measures.

4 Reactive Navigation

The reactive navigation is based on a behavioural based Fuzzy Logic Controller[6] (FLC). Reactive aptitudes are intended to shape high-level commands in real-time, thus leading to a virtual model of the world; further these aptitudes are built in low-level motion advice, forged on the real world perception. Behaviours[3, 1, 2] are modelled via a set of fuzzy rules. Differently from classical control methods like artificial potential field, edge detection and PID controllers, the FLC guarantees robust navigation in a dynamic and unstructured real world, such as indoor environments, and it naturally copes with data uncertainty and does not require an accurate model of the environment. A behaviour based navigation system can be characterized by few but significant elements, these are:

1. The set of behaviours that the system is able to perform;
2. How concurrent behaviours are merged together in order to produce a single motion command.

In RAS navigation system we distinguish between published and internal behaviours to outline the non one-to-one mapping between the set of fuzzy rule bases implemented and the high level behaviour exposed to the external Mission Planner level.

Rule bases are kept simple and light allowing a fast defuzzification process and an easy and modular composing of new higher level behaviours from simpler ones. The complexity of higher level behaviour (like the ones that we are going to list in the following) is achieved by a smart and accurate sequencing of this elementary fuzzy building blocks into a logical structure that we call behavioural stack. This logical structure helps to keep note of the current low level fuzzy behaviour and the whole state of the navigation (i.e. Resuming a path thraking task once an obstacle has been avoided.).

In a navigation task the FLC instinctively shuns obstacles while the robot achieves actions proposed by the cognitive level; there are different behaviours that can be performed by the FLC, the most important are:

1. Obstacle avoidance;
2. Point to point;
3. Path tracking;
4. Wandering

. Obstacle-avoidance behaviour is intrinsically endowed in every action; for each high level command the fuzzy inference system (FIS) includes a specific rule-base that is able to deal with the compromise between avoiding impacts and reaching the given goal. A collision free navigation is always guaranteed. In the FLC we can discriminate essentially between the obstacle-avoidance activity and all the other behaviours that can be viewed like particular cases, with appropriate tunings and data pre-elaborations, of a point-to-point navigation.

Furthermore, an intelligent navigation system should be able to embody more than a simple reactive behaviour. To accomplish this, an inference engine must be able to extend his knowledge beyond a strict local and instantaneous perception domain, that prevents any intelligent behavior. Basically, RAS navigation system is built upon two main layers: a Local Perceptual System and a semi-local Navigation System. While the first reacts in a reflexive fashion to events perceived by sensors, the latter dialogs with the mapping module

(which owns the local and global map). The collected metrical map, may cover or not the information that the navigation system needs in order to find a clear path. If such information is available (the robot has passed at least once in that region), the system has more chances to guess which is the most promising direction to take.

5 Localization and Mapping

5.1 Overview

Low level software architecture in RAS includes also the **Localization and Mapping** Process (in the rest of the document we'll name it **SLAM**). In this project we are realizing an intelligent SLAM system: it synthesizes a series of tasks useful for high level modules (will be explained later).

5.2 Sensor Fusion and Local Metric Mapping

The first step in a Mapping cycle is Sensor Interpretation. Sensor data are mapped onto local occupancy value. A Local Metric Map (RAS centered) is built using only current laser and radar readings. The occupancy value of each local cell is a function of the distance from cell center and the Sensor Polygon. Sensor Polygon is built with all sensors data in the following way: each sensor value becomes a point outside the robot, the points are connected with solid Lines (connecting each pair of nearest points), first and last sonars are connected with the center of RAS with a dashed line. Occupancy value of each local cell (the center of) is a costumed function of both distance and relative position with the polygon (solid and dashed line).

Sonar readings are used in a cyclical way because these sensors are in continuous polling. In subsection 5.5 we'll show how SLAM system chooses when (and where) take (and discard) the Local Metric Map made by sonars.

5.3 Bayesian Filtering and Global Metric Mapping

Local occupancy value must be integrated in time to build a single, global, Metric Map using a Bayesian Filter as explained in [4]and [5]. A Bayesian Filter is a recursive estimator we use to calculate sequences of posterior probability distribution over a quantity that cannot be observed directly: the Map. We name $b_{x,y}^t$ the subjective belief about the state of the cell (x,y), i.e. the probability that the cell is occupied conditioned of sensor readings:

$$b_{x,y}^t = P(occ_{x,y}/o^t, s^t) \quad (1)$$

Where o^t is the observation sequence and s^t is the robot state (x,y, θ) sequence (Robot Path). This desired probability can be computed in the following way:

$$b_{x,y}^t = 1 - \left(1 + \frac{P(occ_{x,y})}{1 - P(occ_{x,y})} \left[\prod_{i=1}^t \frac{P(occ_{x,y}/o_i, s_i)}{1 - P(occ_{x,y}/o_i, s_i)} \frac{1 - P(occ_{x,y})}{P(occ_{x,y})} \right] \right)^{-1} \quad (2)$$

Where o_i and s_i represent observation and robot position at i^{th} time step. The last equation can be written in term of $b/(1 - b)$, that leads to:

$$\frac{b_{x,y}^t}{1 - b_{x,y}^t} = \frac{P(occ_{x,y})}{1 - P(occ_{x,y})} \left[\prod_{i=1}^t \frac{P(occ_{x,y}/o_i, s_i)}{1 - P(occ_{x,y}/o_i, s_i)} \frac{1 - P(occ_{x,y})}{P(occ_{x,y})} \right] \quad (3)$$

This is a recursive formula:

$$\frac{b_{x,y}^t}{1 - b_{x,y}^t} = \frac{b_{x,y}^{t-1}}{1 - b_{x,y}^{t-1}} \frac{P(occ_{x,y}/o_i, s_i)}{1 - P(occ_{x,y}/o_i, s_i)} \frac{1 - P(occ_{x,y})}{P(occ_{x,y})} \quad (4)$$

Where the last term is the a-priori probability: if set to 0.5 for each cell, it can be omitted. The left hand side of the last equation ranges from zero to infinity, when the belief ranges from zero to one. We use this term in the Global Map.

Global Metric Map is realized by the same C++ Class of every kind of Local Metric Map.

5.4 Localization Problem

RAS is equipped with a pair of Encoder (one for each actuated belt) and an Inertial Platform. Our goal is to integrate the estimated position of each sensors and to take care of what the RAS perceives while exploring. In first approximation we can assert that our localization problem is to correct odometric errors (due to slippage and drift) and Inertial Platform errors (due to high vibration and electromagnetic fields): this kind of problem is often named, in literature, **Position Tracking Problem** where errors are assumed small.

Our approach follows the idea in [4] and it is not an approach purely probabilistic: at each computation step, the perceptual error is calculated (estimated), and then corrected. No error distribution will be maintained in memory. This works well when odometers and inertial platform errors are not so big (as in our case). These are restrictive hypotheses because a precise position is needed to build a consistent map. This estimation process tries to minimize the following functional:

$$\begin{aligned} J = & +\beta_1[(x_{robot}^o - x_{robot})^2 + (y_{robot}^o - y_{robot})^2] \\ & +\beta_2(\theta_{robot}^o - \theta_{robot})^2 \\ & +\beta_3[(x_{robot}^{pi} - x_{robot})^2 + (y_{robot}^{pi} - y_{robot})^2] \\ & +\beta_4(\theta_{robot}^{pi} - \theta_{robot})^2 \\ & -\beta_5 Corr(x_{robot}, y_{robot}, \theta_{robot}) \\ & -\beta_6 \sigma(\alpha(\theta_{wall}, \theta_{robot}, \theta_{new})) \end{aligned} \quad (5)$$

Where the symbols β_1, \dots, β_6 represents positive parameter to be tuned for minimization. The first two terms measures the odometric error, the second two measures the inertial platform error and the last two is respectively a measure of the correlation between the local metric map and the global one (a measure of map matching), and a measure of the alignment between robot and environment. In this way it is possible to correct small errors and it works well under the constraint that localization processes will be done frequently (that implicitly means small error each time). Minimization of functional J is done with the well known Gradient Descent algorithm.

We are working to make this Localization Process more intelligent as explain in section 5.5. Now we are testing sensor goodness, in translational and rotational errors, in order to optimize beta factors.

5.5 Intelligent Tasks

Now we want to show some of our Intelligent tasks accomplished (or to be accomplished) by RAS SLAM System. Our goal is that SLAM System should supply to the other systems detailed information about the state: for example nearest unexplored location on the map (that can be used by a planner as default behavior). This task

is accomplished by an independent thread under the SLAM⁵ Process named Value Iteration Thread as the relative algorithm.

This leads to an important consideration: the RAS needs a decisional support to discriminate if a situation is interesting for Mapping or Not. SLAM System Implement a Control Routine that evaluates, for each situation, what should be done with read data. In unknown environment exploration, everything you see is new information and must be unconditionally added to your knowledge, while in known environment you must match what you see and what you should see to determine if there's something wrong. RAS SLAM System implements a routine to determine if it knows enough about local environment and if it can use read data to confirm its knowledge without adding new knowledge. This is done to avoid error caused from a Non-Markovian environment where errors does not compensate each others in means (where measurement errors are not independent) but not only: it realizes an human way of reasoning based on how "I am sure about what I know". Technically speaking, SLAM Control Sub-system analyzes RAS path (sequence of mapping positions during its Life) and actual position to decide if data read must be used to increase current knowledge or to confirm current localization.

REFERENCES

- [1] Rodney A. Brooks, 'Elephants don't play chess', *Robotics and Autonomous Systems*, **6**(1&2), 3-15, (June 1990).
- [2] E. H. Ruspini, 'Fuzzy logic in the Flakey robot', in *Proc. of the Int. Conf. on Fuzzy Logic and Neural Networks (IZUKA)*, pp. 767-770, Iizuka, JP, (1990).
- [3] L. Steels, 'The artificial life roots of artificial intelligence', *Artificial Life*, **1**, 75-110, (1994). citeseer.nj.nec.com/steels93artificial.html.
- [4] S. Thrun, 'Learning metric-topological maps for indoor mobile robot navigation', *Artificial Intelligence*, **99**(1), 21-71, (1998).
- [5] S. Thrun, 'Robotic mapping: A survey', in *Exploring Artificial Intelligence in the New Millenium*, eds., G. Lakemeyer and B. Nebel, Morgan Kaufmann, (2002). to appear.
- [6] L. Zadeh, 'Fuzzy sets', *Information and Control*, **8**, 338-353, (1965).

⁵ Simultaneous Localization And Mapping.