# Autonomous Rover Navigation on Unknown Terrains Functions and Integration

Simon Lacroix, Anthony Mallet, David Bonnafous
Gérard Bauzil, Sara Fleury, Matthieu Herrb, and Raja Chatila
LAAS/CNRS
7, av. du Colonel Roche
F-31077 Toulouse Cedex 4 - France
Firstname.Lastname@laas.fr

**Abstract:** Autonomous long range navigation in partially known planetary-like terrain is an open challenge for robotics. Navigating several hundreds of meters without any human intervention requires the robot to be able to build various representations of its environment, to plan and execute trajectories according to the kind of terrain traversed, to localize itself as it moves, and to schedule, start, control and interrupt these various activities. In this paper, we briefly describe some functionalities that are currently running on board the Marsokhod model robot Lama at LAAS/CNRS. We then focus on the necessity to integrate various instances of the perception and decision functionalities, and on the difficulties raised by this integration.

## 1. Introduction

To foster ambitious exploration missions, future planetary rovers will have to fulfill tasks described at a high abstraction level, such as "`reach the top of that hill`" or "`explore this area`". This calls for the ability to navigate for several hundreds of meters, dealing with various and complex situations, without any operator intervention. Such an ability is still quite an open challenge: it requires the *integration* and *control* of a wide variety of autonomous processes, ranging from the lowest level servoings to the highest level decisions, considering time and resource constraints.

We are convinced that no simple autonomy concept can lead to the development of robots able to tackle such complex tasks: we believe in the efficiency of *deliberative* approaches [4], that are able to plan and control a variety of processes. Following such a paradigm and according to a general economy of means principle, we want the robot to autonomously *adapt* its decisions and behavior to the environment and to the task it has to achieve [5]. This requires the development of:

• Various methods to implement each of the perception, decision and action functionalities, adapted to given contexts;

• An architecture that allows for the integration of these methods, in which deliberative and reactive processes can coexist;

• Specific decision-making processes, that dynamically select the appropriate decision, perception and action processes among the ones the robot is endowed with.

In this paper, we present the current state of development of the robot Lama, an experimental platform within which our developments related to autonomous long range navigation are integrated and tested. We especially focus on the necessity to integrate various implementations of each of the main functionalities required by autonomous navigation (*i.e.* environment modeling, localization, path and trajectory generation). After a brief description of Lama and its equipment, the rest of the paper is split in two parts: the first part briefly presents the main functionalities required by long range navigation we currently consider (terrain modeling, path and trajectory planning, rover localization), while the second part insists on the problems raised by the *integration* of these functionalities.

## 2. The Robot Lama

Lama is a 6-wheels Marsokhod model chassis [10] that has been totally equipped at LAAS[1]. The chassis is composed of three pairs of independently driven wheels, mounted on axles that can roll relatively to one another, thus giving the robot high obstacle traversability capacities. Lama is $1.20m$ wide, its length varies from $1.60m$ to $2.20m$, depending on the axles configuration ($1.90m$ in its "nominal" configuration), and weighs approximately $160kg$. Each motor is driven by a servo-control board, and its maximal speed is $0.17m.s^{-1}$. Lama is equipped with the following sensors:

*Figure 1. The robot Lama on the experimentation site*

• Each wheel is equipped with a high resolution optical encoder, allowing fine speed control and odometry;

• Five potentiometers provide the chassis configuration;

• A 2 axes inclinometer provides robot attitude, a magnetic fluxgate compass and an optical fiber gyrometer provide robot orientation and rotational speed;

• A first stereo bench on top of a pan and tilt unit, is mounted on a $1.80m$ mast rigidly tied to the middle axle. This bench has a azimuthal field of view of approximately 60°, and is mainly dedicated to goal and landmarks tracking;

• A second stereo bench, also supported by a PTU, is mounted upon the front axle, at elevation of $0.80m$ . It has a azimuthal field of view of approximately 90°, and is mainly dedicated to terrain modeling in front of the robot;

• A differential carrier-phase GPS receiver[2] is used to qualify the localiza-

[1]Lama is currently lent to LAAS by Alcatel Space Industries.
[2]currently lent to LAAS by CNES.

tion algorithms.

All the computing equipment is in a VME rack mounted on the rear axle of the robot. The rack contains four CPUs (two PowerPcs and two 68040) operated by the real-time OS VxWorks.

The terrain on which we test the navigation algorithms is approximately $100 \times 50m^2$. It contains a variety of terrain types, ranging from flat obstacle-free areas to rough areas, including gentle and steep slopes, rocks, gravel, trees and bushes.

# Navigation functionalities

## 3. Environment Modeling

Perceiving and modeling the environment is of course a key capacity for the development of autonomous navigation. Environment models are actually required for several different functionalities: to plan paths, trajectories and perception tasks (section 4), to localize the robot (section 5), and also to control the execution of trajectories. We are convinced that there is no "universal" terrain model and that we must build different *multi-layered and heterogeneous* representations adapted to their use.

### 3.1. Qualitative Model

We developed a method that produces a description of the terrain in terms of *navigability classes*, on the basis of stereovision data [7]. Most of the existing contributions to produce similar terrain models come to a data segmentation procedure (*e.g.* [15, 9]), that produce a *binary* description of the environment, in terms of traversable and non-traversable areas. Our method is a classification procedure that produces a probabilistically labeled polygonal map, close to an occupancy grid representation. It is an *identification* process, and does not require any threshold determination (a tedious problem with segmentation algorithms).

Our method relies on a specific discretisation of the perceived area, that defines a *cell image* (figure 2). It "respects" the sensors characteristics: cell resolution decreases with the distance according to the decrease of the data resolution.

Every cell is labelled with a supervised Bayesian classifier: a probability for each cell to correspond to a pre-defined traversability class is estimated. Figure 3 shows a classification result, with two terrain classes considered (flat and obstacle). There are several extensions to the method: the discretisation can be dynamically controlled to allow a finer description, and the classification results can be combined with a terrain physical nature classifier using texture or color attributes. One of its main advantages is that thanks to the probabilistic description, local maps perceived from different viewpoints can be very easily merged into a global description. The produced terrain model can be either used to generate elementary motions on rather obstacle-clear terrains (section 4.1), or to reason at the path level (section 4.3).

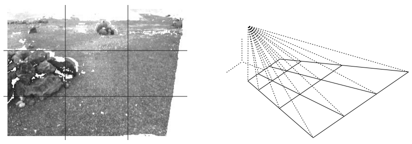

*Figure 2.* *Discretisation of a 3D stereo image. Left: regular Cartesian discretisation in the sensor frame; right: its projection on the ground (the actual discretisation is much finer).*

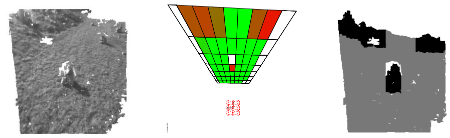

*Figure 3.* *An example of classification result. From left to right: image from stereo, partial probabilities of the cells to be an obstacle (represented as gray levels), and reprojection of the cells in the sensor frame, after the application of a symmetric decision function.*

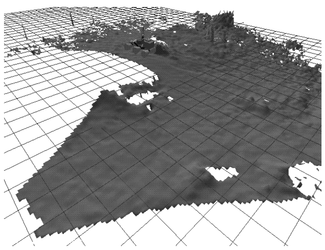

*Figure 4.* *A digital elevation map built by Lama during a 20 meter run using 50 stereovision pairs.*

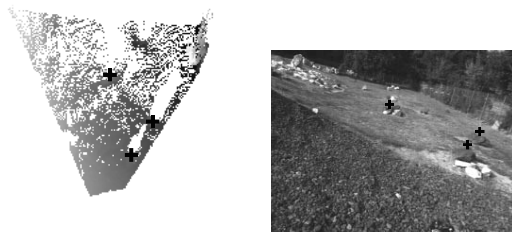

*Figure 5.* *Landmarks (black + signs) detected on the locally built digital elevation maps (left), and reprojected in the camera frame (right). Landmarks are here from 3 to 10 meters away.*

### 3.2. Digital Elevation Map

Although there has been several contributions to the problem of building digital elevation maps ([11, 2]), we think that it has still not been addressed in sufficiently satisfactory way: the main difficulty comes from the uncertainties on the 3D input data, that can be fairly well estimated, but hardly propagated throughout the computations and represented in the grid structure.

However, a quite realistic model can be easily built by computing the mean elevation of the data points on the grid cells, using only the points that are provided with precise coordinates. With our stereovision algorithm for instance, 3D points whose depth is below $15m$ can be used to build a realistic $0.05 \times 0.05m^2$ cell digital elevation map. Provided the robot is localized with a precision of the order of the cell size, data acquired from several view-points can be merged into a global map (figure 4). This model is then used to detect landmarks (section 3.3) and to generate elementary trajectories (section 4.2).

### 3.3. Finding Landmarks

An efficient way to localize a rover is to rely on particular elements present in the environment, referred to as *landmarks* (section 5.3). Several authors presented 3D data segmentation procedures to extract salient objects, and our first attempts were based on a similar principle [3]. However, such techniques are efficient only in simple cases, *i.e.* in scenes were sparse rocks lie on a very flat terrain, but rather fragile on rough or highly cluttered terrains for instance.

To robustly detect such local peaks, we are currently investigating a technique that relies on the computation of similarity scores between a digital elevation map area and a pre-defined 3D peak-like pattern (a paraboloid for instance), at various scales. First results are encouraging (figure 5), and the detected landmark could be used to feed a position estimation technique (section 5.3).

## 4. Trajectory generation

A generic trajectory planner able to deal with any situation should take into account all the constraints, such as rover stability, rover body collisions with the ground, kinematic and even dynamic constraints. The difficulty of the problem calls for high time-consuming algorithms, which would actually be quite inefficient in situations where much simpler techniques are applicable. We therefore think it is worth to endow the rover with various trajectory generation algorithms, dedicated to the kind of terrain to traverse. Section 7 describes how they are actively started and controlled.

### 4.1. On easy terrains

On easy terrains, *i.e.* rather flat and lightly cluttered, dead-ends are very unlikely to occur. Therefore, the robot can efficiently move just on a basis of a goal to reach[3], and of a terrain model that exhibits non-traversable areas, using techniques that evaluates elementary motions [7].

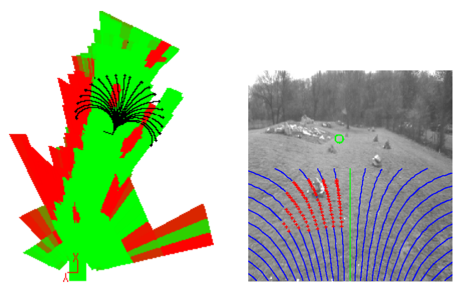

*Figure 6. A set of circle arcs to evaluate on the global probabilistic model (left), and reprojection of the arcs in the camera view (right)*

To generate motions in such terrains, we use an algorithm that evaluates circle arcs on the basis of the global qualitative probabilistic model. Every cycle, the algorithm is run on an updated terrain model. It consists in evaluating the interest (in terms of reaching the goal) and the risk (in terms of terrain traversability) of a set of circle arcs (figure 6). The arc that maximizes the interest/risk ratio is chosen.

### 4.2. On rough terrains

On uneven terrain, the notion of obstacle clearly depends on the capacity of the locomotion system to overcome terrain irregularities, and on specific constraints acting on the placement of the robot over the terrain. These constraints are the stability and collision constraints, plus, if the chassis is articulated, the configuration constraints (figure 7). To evaluate such constraints, the probabilistic qualitative model is not anymore sufficient and we use a digital elevation map.

We developed a planner (a stripped down version of [8]) that evaluate elementary trajectories, in a way very similar to section 4.1 : a set of circle arcs is produced, and for each arc, a discrete set of configurations are evaluated. Each

[3] not necessarily the distant global goal, it can be a formerly selected sub-goal - see section 4.3

arc is then given a *cost* that integrates the elementary costs ("dangerousness") of the successive configurations it contains, the arc to execute being the one that maximizes the interest/cost ratio.

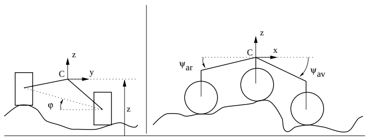

*Figure 7. The chassis internal configuration angles checked on the digital elevation map.*

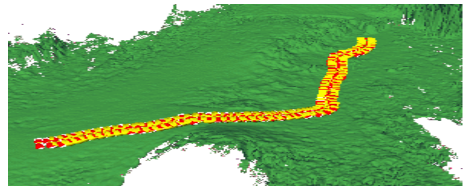

*Figure 8. A trajectory resulting from the application of the rough terrain local planner (approximately 30 cycles).*

### 4.3. Planning Paths

The two techniques described above are not able to efficiently deal with highly cluttered areas and dead-ends. For that purpose, we use a *path* planner, that reasons on the global qualitative model to find sub-goals and perception tasks [12].

The global qualitative model, which is built upon a bitmap structure, includes a set of regions and a graph that connects them. A search algorithm provides an *optimal* path to reach the global goal The path is then analyzed to produce a sub-goal to reach: it is the last node of the path that lies in a traversable known area. The "optimality" criterion takes here a crucial importance: it is a linear combination of time and consumed energy, weighted by the terrain class to cross *and the confidence of the terrain labeling*. Introducing the labeling confidence in the crossing cost of an arc amounts to *implicitly* consider the modeling capabilities of the robot.

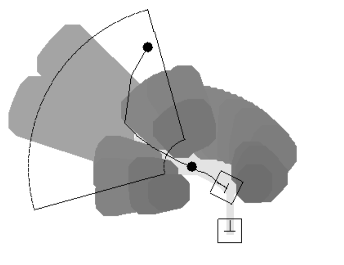

*Figure 9. A result of the navigation planner on the qualitative model.*

## 5. Localization

A position estimate is not only necessary to build coherent environment models, it is also required to ensure that the given mission is successfully being achieved, or to control motions along a defined trajectory. Robot self-localization is actually one of the most important issue to tackle in autonomous navigation.

One can distinguish various algorithm categories that compute the robot's position: *(i) motion estimation* techniques, that integrate data at a very high pace as the robot moves (odometry, inertial navigation, visual motion estimation - sections 5.1 and 5.2), *(ii) position refinement* techniques, that rely on the matching of landmarks perceived from different positions, and *(iii) absolute localization* with respect to an initial global model of the environment. All these algorithms are complementary, and provide position estimates with different characteristics. We are convinced that an autonomous rover must be endowed with at least one instance of each category.

### 5.1. Odometry on Natural Terrain

We use 3D odometry with Lama by incorporating the attitude informations provided by the 2 axes inclinometer[4] to the translations measured by the encoders of the central wheels. Due to skid-steering, the angular orientation measured by the odometers is not reliable: the information provided by the integration of the gyrometer data is much better, and do not drift significantly before a few tens of minutes.

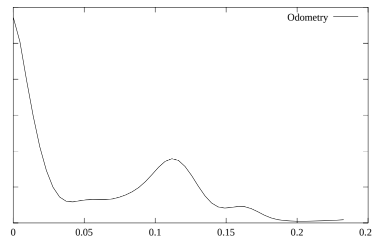


*Figure 10.* *Histogram of odometry errors measured every* $0.5m$ *steps during a 50 meter run with Lama on various kinds of terrain.*

To have a quantitative idea of the precision of odometry, we gathered some statistics, using a carrier-phase DGPS as a reference. Figure 10 show that odometry can hardly be modeled an estimator with Gaussian uncertainties: some *gross errors* (due to lateral slippages) actually occur quite often. We are currently investigating the possibility to analyze on line a set of proprioceptive data in order to be able to dynamically qualify the odometry, and especially to detect such errors. These data are the 6 wheel encoders, the measured currents, the two attitude parameters and the five chassis configuration parameters.

### 5.2. Visual motion estimation

We developed an exteroceptive position estimation technique that is able to estimate the 6 parameters of the robot displacements in any kind of environments, provided it is textured enough so that pixel-based stereovision works well: the presence of no particular landmark is required [13]. The technique computes the motion parameters between two stereo frames on the basis of a set of 3D point to 3D point matches, established by tracking the corresponding pixels in the image sequence acquired while the robot moves.

We pay a lot of attention to the selection of the pixels to track: in order to avoid wrong correspondences, one must make sure that they can be faithfully tracked, and in order to have a precise estimation of the motion, one must choose pixels whose corresponding 3D points are known with a good accuracy. Pixel selection is done in three steps: an *a priori* selection is done on the basis of the stereo images; an empirical model of the pixel tracking algorithm is used to discard the dubious pixels during the tracking phase; and finally an outlier rejection is performed when computing an estimate of displacement between two stereo frames (*a posteriori selection*).

We evaluated the algorithm on many runs (totalizing several hundreds of meters) and it gives translation estimates of about 4% of the distance. Work related to this algorithm is still under way, with the goal of reaching a precision on translation estimates of about 1%.

### 5.3. Landmark Based Localization

We are currently investigating Set Theoretic approaches for landmark-based localization [14]. No statistical assumptions are made on sensor errors: the

[4]after the application of a slight smoothing filter on its data.

only hypothesis is that errors are bounded in norm. Estimates of the robot and landmarks positions are derived in terms of *feasible uncertainty sets*, defined as regions in which the robot and the landmarks are guaranteed to lie, according to all the available informations.

Some simulation results using realistic bounds, using the landmarks detection algorithms presented in section 3.3 are promising. The integration of these algorithms on board Lama is currently under way.

# Integration

Our research group has been working for several years on the definition and development of a generic software and decisional architecture for autonomous machines [1]. This architecture has been successfully instantiated in multi-robot cooperation experiments, indoor mobile robotics experiments, and autonomous satellite simulations [6]. Within this architecture, we addressed the problem of running concurrent localization algorithms (section 6) and we also implemented navigation strategies (section 7).

## 6. Integration of Concurrent Localization Algorithms

To tackle the coexistence of several localization algorithms in a generic and reconfigurable way, we developed a particular module named PoM (position manager), that receives all the position estimates produced by localization as inputs, and produces a single consistent position estimate as an output. PoM addresses the *Sensor geometrical distribution* issue (section 6.1) as well as the *localization modules asynchronism* and the *fusion of the various position estimates* (section 6.2).

### 6.1. Internal situation

Distributing geometrical information is not satisfying: some information is hard-coded and duplicated within the modules, and it complicates the porting of a module to another robot or another sensor. For that purpose, we use InSitu, a centralized geometrical description of a robot. InSitu reads a configuration file upon startup, and provides the necessary frame coordinates to any module when the robot navigates. The configuration file is the textual description of a geometrical graph: the nodes are frames coordinates that need to be exported.

Every data acquisition module reads the frame configuration which it relates to, and associates to its data a “tag” structure. Thanks to this tagging, clients using such data do not have to care from where the data comes, since all the necessary geometrical and time information is contained in the data itself. The tag is propagated along with the data between modules, thus making inter-module data communication very flexible.

### 6.2. Position Management

Positions computed by the various position estimators are always produced with some delay, that depends on the computation time required to produce a particular position. PoM maintains the time consistency between the various position estimates by managing one time chart for each position estimator handled by PoM, plus one particular chart for the *fused* position (fusion of every position estimator). Actually, no data fusion algorithm is currently implemented within PoM: given the individual position estimator characteristics, we indeed consider that a *consistency check* (fault detection) has to be performed formerly to any fusion. Up to now, an estimate selection is performed on the basis of a confidence (real value between 0.0 and 1.0, 1.0 being the best) that is hard-coded for each position estimator.

## 7. Navigation strategies

The integration of the various algorithms presented in the first part of the paper requires specific decisional abilities, that are currently instantiated as Tcl scripts. The following simple strategy is currently applied: the three environment models (qualitative map, digital map and landmark map) are *continuously* updated every time new data are gathered, and the two integrated localization algorithms (odometry and visual motion estimate) are also continuously running[5]. The selection of the trajectory generation algorithm is the following: given a global goal to reach, the easy terrain algorithm is applied until no feasible arcs can be found. In such cases, the rough terrain algorithm is applied. It is run until either the easy terrain algorithm succeeds again, or until no feasible arcs are found in the elevation map. In the latter case (that can be assimilated to a dead end), the path planning algorithm is run, to select a sub-goal to reach. The whole strategy is then applied to reach the sub-goal, and so on.

## 8. Conclusion

We insisted on the fact that to efficiently achieve autonomous long range navigation, various algorithms have to be developed for each of the basic navigation functions (environment modeling, localization and motion generation). Such a paradigm eventually leads to the development of a complex integrated system, thus requiring the development of integration tools, at both the functional and decisional levels. We are convinced that such tools are the key to implement efficient autonomy on large time and space ranges.

There are however several open issues. Among these, we believe that the most important one is still localization. In particular, the system must be robust to extremely large uncertainties on the position estimates, that will eventually occur: this requires the development of landmark *recognition* abilities to tackle the data association problem, and also the development of terrain model structures that can tolerate large distortions. Note that both problems should benefit from the availability of an initial terrain map, such as provided by an orbiter, whose spatial consistency is ensured. Indeed, the development of algorithms that match locally built terrain models with such an initial map would guarantee bounds on the error of the position estimates.

## References

[1] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An architecture for autonomy. *Special Issue of the International Journal of Robotics Research*

[5] landmark-based localization integration is under way, but is should also be continuously run, while *actively* controlling image acquisition.

*on Integrated Architectures for Robot Control and Programming*, 17(4):315–337, April 1998. Rapport LAAS 97352, Septembre 1997, 46p.

[2] P. Ballard and F. Vacherand. The manhattan method : A fast cartesian elevation map reconstruction from range data. In *IEEE International Conference on Robotics and Automation, San Diego, Ca. (USA)*, pages 143–148, 1994.

[3] S. Betge-Brezetz, R. Chatila, and M.Devy. Object-based modelling and localization in natural environments. In *IEEE International Conference on Robotics and Automation, Nagoya (Japan)*, pages 2920–2927, May 1995.

[4] R. Chatila. Deliberation and reactivity in autonomous mobile robots. *Robotics and Autonomous Systems*, 16(2-4):197–211, 1995.

[5] R. Chatila and S. Lacroix. A case study in machine intelligence: Adaptive autonomous space rovers. In A. Zelinsky, editor, *Field and service Robotics*, number XI in Lecture Notes in Control and Information Science. Springer, July 1998.

[6] J. Gout, S. Fleury, and H. Schindler. A new design approach of software architecture for an autonomous observation satellite. In *5th International Symposium on Artificial Intelligence, Robotics and Automation in Space, Noordwijk (The Netherlands)*, June 1999.

[7] H. Haddad, M. Khatib, S. Lacroix, and R. Chatila. Reactive navigation in outdoor environments using potential fields. In *International Conference on Robotics and Automation, Leuven (Belgium)*, pages 1232–1237, May 1998.

[8] A. Hait, T. Simeon, and M. Taix. Robust motion planning for rough terrain navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, Kyongju (Korea)*, pages 11–16, Oct. 1999.

[9] L. Henriksen and E. Krotkov. Natural terrain hazard detection with a laser rangefinder. In *IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico (USA)*, pages 968–973, April 1997.

[10] A. Kemurdjian, V. Gromov, V. Mishkinyuk, V. Kucherenko, and P. Sologub. Small marsokhod configuration. In *IEEE International Conference on Robotics and Automation, Nice (France)*, pages 165–168, May 1992.

[11] I.S. Kweon and T. Kanade. High-resolution terrain map from multiple sensor data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):278–292, Feb. 1992.

[12] S. Lacroix and R. Chatila. Motion and perception strategies for outdoor mobile robot navigation in unknown environments. In *4th International Symposium on Experimental Robotics, Stanford, California (USA)*, July 1995.

[13] A. Mallet, S. Lacroix, and L. Gallo. Position estimation in outdoor environments using pixel tracking and stereovision. In *IEEE International Conference on Robotics and Automation, San Francisco, Ca (USA)*, pages 3519–3524, April 2000.

[14] M. Di Marco, A. Garulli, S. Lacroix, and A. Vicino. A set theoretic approach to the simultaneous localization and map building problem. In *39th IEEE Conference on Decision and Control, Sydney (Australia)*, Dec. 2000.

[15] L. Matthies, A. Kelly, and T. Litwin. Obstacle detection for unmanned ground vehicles: A progress report. In *International Symposium of Robotics Research, Munich (Germany)*, Oct. 1995.