Published in IEEE Int. Conf. on
Robotics and Automation
San Diego, USA, May 8-13 1994

Autonomous Navigation in Outdoor Environment : Adaptive Approach and Experiment

Simon Lacroix, Raja Chatila, Sara Fleury, Matthieu Herrb, Thierry Siméon

LAAS-CNRS
7, avenue du Colonel-Roche
31077 Toulouse Cedex - France

Abstract

This paper presents the approach, algorithms and processes we developed to perform cross-country autonomous navigation. After a presentation of the tele-programming context, we introduce an adaptive navigation approach, well suited for the characteristics of complex natural environments. The main perception, motion planning and decisional processes required by the robot during navigation are briefly presented. An on board control architecture that manages all these processes is then described, and first results of an experiment currently developed at LAAS are discussed.

1 Introduction - Context

Recently, several studies considering autonomous mobility in *natural unstructured* outdoor environments comes out [1]: various applications are considered, such as public safety (fire fighting, chemical disaster...), sub-sea intervention or exploration, and planetary exploration.

Several aspects make these kinds of interventions a demanding and difficult problem for robotics :

• The robot operates in a natural, unstructured, maybe hostile and a priori unknown environment ;

• The information on the robot and the environment is mostly acquired through the robot's sensors ;

• There might be interaction discontinuities with the robot because of communication breakdowns, important delays or low bandwidth.

These constraints rule out direct teleoperation as well as telerobotics approaches [2], and point towards robots with **important autonomous capacities**.

One approach that has been proposed [3, 4] is to use "simple" but completely autonomous robots, *without any control* from a distant operator. Such robots, using a behavior-based control scheme [5], can not be *programmed*, in the sense that once they started, there is no way to change or update their mission, or to recover any failure. We think this approach would be far from guaranteeing the success of the mission.

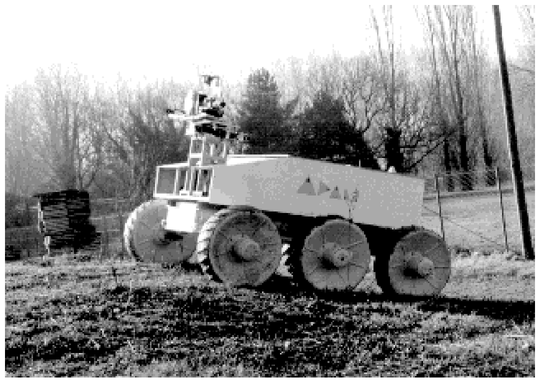
Figure 1: **The mobile robot ADAM in its environment**

We have a different approach to the challenge of remote intervention by mobile robots, which essentially ensues from the following considerations [2] :

• The intervention area may be very far from the robot's starting point. Hence the robot has to travel some distance (as far as tens of kilometers or more) to reach a *specific region* (not at random nor merely in a given direction).

• The mission is not defined once and for all : according to returned data, one should be able to change the objectives of the mission (when unexpected events occur for instance) or to decide the execution of a particular action (such as "pick this sample" in the case of a scientific exploration). To be able to remotely control the robot, it is necessary to know what it is doing, or where it is.

• Because the environment is poorly known, the mission can only be pre-defined at a *task-level* in general, and not in its every detail. Hence the robot must be able to *interpret* the mission according to its actual context during its autonomous execution.

• The robot could fall into difficult situations in which its perception, interpretation or decision making capacities are insufficient. Human intervention would then be necessary for troubleshooting.

According to these arguments, we propose a global architecture with two main parts [6] : an operating station for mission programming and supervision, and a remote robot system (not necessarily a single one) able to interpret and execute the mission autonomously.

The operating station includes the necessary functions that allow a human to build an *executable mission* that can be interpreted and executed by the robot (as opposed to a higher level description of objectives) ; and to supervise its execution, taking into account the delays and communication constraints.

We focus here on the robot system, as an entity capable of performing autonomous navigation in a initially unknown cross-country environment, and configured to allow an interaction with the operating station. This papers presents the principles we retained to conceive and develop the software of the robot, and is articulated as follow : the general adaptive approach we choose to tackle with complex unstructured environments is first introduced. This approach induces needs for several different functionalities, that are briefly presented in sections 3 and 4. Section 5 presents how these functionalities are embedded in an architecture that fulfills both the autonomy and the tele-programming requirements. Some preliminary experimental results with the robot Adam[1](figure 1) are presented.

2 An Adaptive Approach for Outdoor Navigation

The complexity of outdoor natural environments comes essentially from their diversity and lack of structure : some areas can be totally flat (maybe cluttered with easily detectable obstacles - big rocks lying on a prairie for instance), whereas others area can be much more cluttered, such as sand dunes or uneven rocky areas. This variety induces several different behaviors, and constrains both the perception and motion planning processes.

Several systems for outdoor navigation have already been achieved with the ALV [7] and UGV projects, as well as at CMU (Navlab [8] and Ambler [9]) or at Hughes Research Lab. [10]. In summary, the main difference between these approaches and ours are related to the environment representations used, and to the integration of various motion planning algorithms, depending on the nature of the area to cross : according to a general economy of means principle, we favor an *adaptive* approach [11]. We chose to adapt the robot behavior to the nature of the terrain, and hence consider three navigation modes :

• A **2D planned** navigation mode : Applied when the terrain is mostly flat - or with an admissible slope for the robot -, it relies on the execution of a planned 2D trajectory, using a binary description of the environment in terms of *Crossable/Non-Crossable* areas.

• A **3D planned** navigation mode : Applied when an uneven area has to be crossed, it requires a precise model of the terrain, on which a 3D trajectory is planned and executed.

• And a **reflex** navigation mode : The robot locomotion commands are determined on the basis of *(i)* a goal (heading or position) and *(ii)* the information provided by "obstacle detector" sensors. This navigation mode does not require any modeling of the terrain, that has simply been labeled as "essentially obstacle-free".

Our approach to determine which navigation mode can be applied is based on a quick analysis of the raw 3D data produced either by the Laser Range Finder (LRF) or by a stereovision algorithm. This quick analysis provides a description of the terrain in term of navigation classes, that is incrementally fused to maintain a *global qualitative representation* of the environment. All "strategic" decisions are taken on the basis of this global representation. They concern the determination of the intermediate positions, the choice of the navigation mode to apply to reach them, as well as the definition of the next perception task to execute (Which sensor to use ? With what operating modalities ? How should the data be processed ?). Such an approach involves the development of various perception and motion planning processes, and emphasizes the importance of the *navigation planner*, which is in charge of the strategic decisions.

3 Perception Processes[2]

3.1 Terrain Representations : A Structural Scheme

Various kind of informations are required during navigation : a localization procedure requires a description of landmarks, whereas a trajectory planner procedure requires a continuous description of the zone to cross for instance. We believe that aiming at building a "universal" terrain model that contains all the necessary informations is extremely difficult and not

[1]ADAM : Advanced Demonstrator for Autonomy and Mobility, is property of Framatome and Matra Marconi Space, currently lent to LAAS ; its chassis was built by VNII Transmach (S^t Petersburg, Russia)

[2]This section and the following are a brief presentation of the different functionalities required by the robot during navigation. If the reader is especially interested by a particular point, we invite him to refer to the corresponding references.

Figure 2: **Terrain representations used in the system and their constructive relations**

3.2 Building Representations

3.2.1 Global Qualitative Representations

Fast Terrain Analysis

Applied each time 3D data are acquired, this process produces a description of the perceived areas in term in *terrain classes* [13]. It relies on a specific discretization of the perceived area that respects the sensor resolution. The discretization defines "cells", on which different global mean characteristics are determined : density (number of points contained in a cell compared with a nominal density defined by the discretization rates), mean altitude, variance on the altitude, mean normal vector and corresponding variances. These informations allow to heuristically label each cell as one of {*Flat, Uneven, Obstacle, Unknown*}. Thanks to statistical figures, a confidence value on the labeling of each cell is determined. This value, that obviously decreases with the distance of the cell to the sensor, constitutes the useful model of the logical sensor "terrain classifier"

The classification procedure takes less than half a second[3] to process a 10.000 3D points image. It has proved its robustness on a large number of different images, produced either by the LRF or a stereovision correlation algorithm, and is especially weakly affected by the sensor noise (uncertainties, artifacts and errors).

Global Model Construction

The global terrain model is a bitmap structure, in which the main informations provided by the classification are encoded (label and corresponding confidence, elevation). Such a structure is simple, adapted to the lack of geometrical structure of the environment and to the elevation maps structure.

Fusion of the classifier output is a simple and fast procedure : each cell is written in the bitmap using a polygon filling algorithm. When a pixel has already been perceived, the possible conflict with the new perception is solved by comparing the label confidence values. Many experiments have proved the robustness of this fusion method.

Connection Graph

Once the global representation is updated, it is structured thanks to region merging and contour following algorithms. The constrained areas (Uneven, Obstacle and Unknown) are first growed by the robot radius : indeed, the robot is considered to be in such an area when any of its part is. Regions are then merged : they are areas of uniform label, elevation and confidence. The graph is deduced from the adjacency relations between the regions : each node corresponds

[3]In this paper, all the execution times correspond to an execution on a Sparc-10 workstation

to the middle point of a border, and each arc corresponds to the crossing of a region. This structuration procedure is executed within a few seconds (essentially depending on the ratio of constrained area to grow) ; as an example, the fusion of four acquisitions covering an area of approximately 200 square meter produces 400 regions, and the graph contains 1500 nodes.

3.2.2 Elevation Map

To build this fine terrain model, we use a generic interpolation method [14] that builds a discrete representation $z = f(x, y)$ on a regular Cartesian grid from a 3D spherical image.

Local Elevation Map (LEM) Building

The method relies on the analysis of all sets of four neighboring points in the spherical image : they define *patches* in the Cartesian robot's redressed frame. Thanks to the fine grid resolution, a planar approximation is sufficient to represent a patch. The interpolation problem is then reduced to finding the intersection between each (x, y) "vertical" line and the plane that best approximate the patch. A test based on depth discontinuities allows to decide whether a patch can be interpolated or not, and leads to an estimation of the elevation Z for the (x, y) interpolated points. An accuracy on each computed elevation is estimated, using Jacobian matrix of the sensor model to estimate variances on the raw Cartesian measurements, and a Kalman Filter to compute variances on the plane parameters.

Global Elevation Map Building

A fusion of different LEM in a global elevation map may be needed for trajectory planning if the uneven area can not be entirely perceived from a single viewpoint. Once the estimation of the new robot's position is achieved (section 3.2.3), we combine the new LEM and the former Global Elevation Map into a new global map. The new elevation is updated using a weighted average.

3.2.3 Localization Model

We developed a localization process that relies on a peak detection method [15], well suited for unstructured environments.

The specific terrain representation used here is a B-Spline surface based model, built upon an elevation map thanks to a least-square approximation. Such a model is very rich and compact, and provides a hierarchical description of the environment : a coarse level B-Spline representation is first computed on a uniform mesh, and a test based on the least-square errors points out the areas where some refinement is needed. A new mesh with smaller size patches is then defined, and a new B-Spline representation is computed, which ultimately leads to a tree model, in which each node corresponds to a B-Spline surface.

This analytic model allows to extract features such as high curvature points, valleys or ridges. We currently implemented a peak extraction procedure based on a quick analysis of the *matrix* expression of the B-Spline surfaces. Once the peaks are extracted, a feature matching localization method is applied to focus an iconic one : the iconic method is only performed in the neighborhood of the detected features. Hence, using small correlation windows, we avoid the long processing time usually encountered with such methods.

4 Planning Processes

4.1 Navigation Planning

The navigation planner is a key component : mixing procedural knowledge and knowledge about the environment, it performs the decisions that provide the robot with a "smart" behavior. These decisions include *perception strategies* and *motion strategies*, that imply the definition of intermediate goals and the choice of navigation modes to apply to reach them.

The basic technique to plan a route in the known environment relies on the execution of an A^* search in the connection graph. The search selects a path, *i.e.* a succession of connected regions, that defines the intermediate goal and the motion mode to activate. The valuation of the arcs is obviously determinant to implement different strategies. Our valuation is currently a heuristic mix between these three criteria :

• **Arc label** : to plan a route that minimizes the execution time, the region label are taken into account. The planner then avoids to cross uneven areas when possible.

• **Arc confidence** : considering only the former constraint, the artifacts raised by the classification procedure (essentially badly labeled "obstacle" cells) would mislead the robot navigation. The arc label criterion is therefore weighted by its confidence : this allows the planner to cross some obstacles areas for instance, which actually triggers the execution of a new perception task when facing such areas.

• A **localization ability value**, derived from to the localization model is taken into account during the search to select paths along which a localization is possible.

Once the optimal path is found in a graph, it is analyzed to determine the next *perceptual task*, thanks to simple procedures :

• A localization task is planned when the position estimation uncertainty overrides a threshold (this uncertainty estimation is simply derived from the path length and crossed region labels) ;

• A "discover" task (classification) is planned when reaching unknown areas : it is determined in order to perceive the unknown areas that corresponds to the path returned by the search ;

• A model refining task is planned when reaching constrained areas labeled with a low confidence (classification), or when a confidently labeled uneven area has to be crossed (fine modeling, to initiate the 3D planned mode).

4.2 Trajectory Planning

4.2.1 Reactive Navigation

when the terrain to be crossed is flat and obstacle free (which is detected by the classification procedure), the reactive navigation mode is activated : it skips the global modeling and motion planning steps, the robot heads directly to its goal. If an obstacle is detected by a proximity sensor (the surveillance mode of the LRF), the robot stops[4], and the nominal procedure of terrain modeling (*ie.* fusion of the classification results and structuration) is started.

4.2.2 Flat Terrain

The trajectory is searched with a simplified and fast method, based on bitmap and potential fields techniques. The robot is approximated by a circle, and its configuration space is two dimensional, corresponding to the robot's position in the horizontal plane. First, a binary bitmap *free/obstacle* is extracted from the global bitmap model over the region to be crossed ; a distance propagation method then produces a distance map and a discrete voronoï diagram, from which a path reaching the sub-goal is extracted. To provide an executable trajectory, a set of line segments is derived from this first path.

Search time depends only on the bitmap discretization, and not on the complexity of the environment. The final trajectory is obtained in at most 2.5 seconds on a 256×256 bitmap.

4.2.3 Uneven Terrain

On uneven terrain, irregularities may alter the attitude and the motions of the robot. The path planner therefore requires a 3D description of the terrain relief (based on the elevation map), and a model of the robot geometry in order to produce trajectories verifying collision-free constraints, and that guarantees terrain irregularities absorption and the stability of the vehicle. This planner, first described in [16], computes a motion verifying these constraints by exloring a graph of configurations obtained by applying sequences of constant controls (rotations and straight line segments in the case of ADAM).

[4]Local obstacle avoidance is this reactive mode is under development.

In the case of incremental exploration of the environment, an additional constraint must be taken into account : the existence of unknown areas on the terrain elevation map. Indeed, any terrain irregularity can hide a part of the ground. When it is possible (this caution constraint can be more or less relaxed), the path planner must avoid such unknown areas. If not, it must search the best way through unknown areas, and provide the best perception point of view on the way to the goal. This is performed with the following operations :

1. The unknown areas of the elevation map are filled by an interpolation operation, which provides a continuous elevation map.

2. The planner then searchs a way through unknown parts of the map. However, the avoidance of unknown areas is obtained by an adapted weight of the arc cost.

3. In order to improve the heuristic guidance of the search, a new heuristic distance to the goal is built by bitmap techniques : a cost bitmap is computed, including the difficulty of the terrain, and the proportion of unknown areas around the current patch.

4. A potential propagation from the goal generates a distance information corresponding to the best way to the goal through terrain relief and unknown areas.

Finally, once a trajectory reaching the goal is obtained, it is truncated at the last point that enables to perceive the first unknown area to cross.

Search time strongly depends on the difficulty of the terrain. The whole procedure takes between 40 seconds to a few minutes, on an Indigo R4000 Silicon Graphics workstation. Figure 4 shows a trajectory computed on a real terrain, where darker areas correspond to interpolated unknown terrain.

5 System Architecture and Control

The generic control architecture for the autonomous mobile robots developed at LAAS [6] is instantiated in the case of the EDEN experiment as shown in figure 3. The higher task planning level, not currently used in the experiment, plans the mission specified by the operator in terms of tasks, with temporal constraints, executable by the robot.

Let's describe here the functional and decisional levels, and the way they are integrated.

5.1 The Functional Level

The Functional Level includes the functions for acting (wheels, perception platform), sensing (laser, cameras, odometry and inertial platform) and for various data processing (feedback control, image processing, trajectory computation, ...). To control robot functionalities and underlying resources, all these func-

Figure 3: **Global control architecture. Connections between the modules at the functional level show data flow.**

tions are embedded into modules defined in a systematic and formal way [17], according to data or resources sharing. Thus, modules are servers which are called via a standard interface, and allow to combine or to redesign easily the functions. The connections between modules in the figure 3 are dynamically established by the decisional level according to the context.

5.2 The Decisional Level

This level includes the navigation planner and a supervisor that establishes at run-time the dependencies between modules. It also controls their execution according to the context and the robot state, and installs the conditions/reactions in case of external events (watching for obstacles when executing a trajectory for instance).

The Supervisor

It receives the task to execute, described in terms of actions to be carried out and modalities. If the task is not directly executable, the navigation planner refines it. The supervisor watches for events (obstacles, time-out, etc.) and reacts to them as planned, according to the dynamics of the situation. It sends to the Executive the different sequences of actions which correspond to the task, and sends back to the task planner (or the operator) the informations related to the task (*e.g.* specific data, and the report about its execution,etc.).

Navigation Planner

In general, it is necessary to refine the task since the environment is roughly known at the moment of the task planning. The purpose of the refinement planner is to produce a sequence of elementary actions, at run-time, on the basis of the current environment state, the modalities of the task, and the state of the robot. It executes specific procedures (section 6) and performs the decisions about the navigation modes and strategies.

The Executive

The executive launches the execution of actions by sending the related requests to the functional level modules. It manages the access to resources and the coherence of multiple requests at the syntactic level, and sends back to the supervisor reports about the fulfillment of those basic actions. The parallelism of some sequences is taken into account : for instance the robot moves toward a sub-goal while watching for obstacles.

5.3 Integration

In our current implementation, the three entities of the decisional level have been simplified and merged together, using a Procedural Reasoning System (PRS [18]), an environment to develop and execute operational procedures, well designed to cope with the constraints of the architecture. The procedural representation is convenient to define the different actions needed to carry out a task, or to introduce some heuristic reasoning in case of non-nominal situations. The choice of the best procedure to solve a given task can be stated on the basis of the data corresponding to the context of execution, or deduced by a meta-procedure that reasons about the best applicability of such procedures.

6 The EDEN Experiment

6.1 The Robot

ADAM has six motorized non directional wheels with passive suspensions. Its internal localization system is based on the fusion of informations provided by a 3-axis inertial platform and 12 position encoders (one odometry and one suspension position coder for each wheel). We have equipped ADAM with a 3D scanning laser range finder with a deflecting mirror and two color cameras, mounted on a 1-axis pan platform.

6.2 Hardware

The on-board computing architecture is composed of two VME racks running under the real time operating system VxWorks, connected to the operating station (a Sun SparcStation 10-41) by an Ethernet link. The first rack includes two 68030 CPUs and various I/O boards, and is dedicated to localization and locomotion[5]. The second rack is composed of two 68040 CPUS, three Datacube boards and some I/O. It is dedicated to sensing activities : video image acquisition, laser range finder command and acquisition, local processing of data.

During the experiments, most of the "high level" computing processes are run on the operating workstation to take benefit of a better debugging environment and of the pre-existence of the softwares under Unix. However, we have the possibility to embark all the softwares in a near future : some are already ported under VxWorks, and it is possible to use an on-board Sparc CPU under Sun-OS.

6.3 Experiments

Adam's environment is a 20 by 50 meters area, composed of flat, sloppy, uneven rocky areas, and of big obstacle rocks. The canonical task to execute is "GoTo Landmark", the environment being initially *totally unknown*. It is run according to the following procedure (parentheses indicate the corresponding module) :

1. A video image is acquired in the direction of the target (Video acquisition).

2. If identified in the video image, the target is localized in a global reference frame (Target recognition), which defines the final goal.

3. A 3D image is acquired (Laser range-finder).

4. Using the laser image, the terrain is classified and the global representation updated (Terrain classif.). From this representation, the navigation planner selects a sub-goal within the navigable regions, and a navigation mode to execute.

4-bis. If an uneven region is to be crossed, the corresponding elevation map is computed (Elevation Map).

5. The selected trajectory planner is used to find a trajectory reaching the sub-goal (2D traj. planner or 3D traj. planner).

6. The trajectory is executed with a feedback control loop on the position given by the internal localization system and the obstacle detector (Motion Control).

7. If necessary, the position of the robot is updated (Robot Localization).

We have already performed several "reach that goal" experiments using only the 2D motion planner in the crossable zones, and an "discovery" strategy. After a few "perceive - analyze - plan" steps, (from 3 to 10, depending on the chosen path) Adam reaches the target located at an approximatively 30 meter distance from its starting point. The whole processing time does not exceed half a minute at each step, but due to the slow motion of the robot (its maximum speed is 28 cm/s) and the LRF image acquisition time, ADAM takes generally about 20 minutes to execute its mission.

[5] It was provided with the associated software "GNC" (Guidance, Navigation and Control) by Matra Marconi Space and Framatome

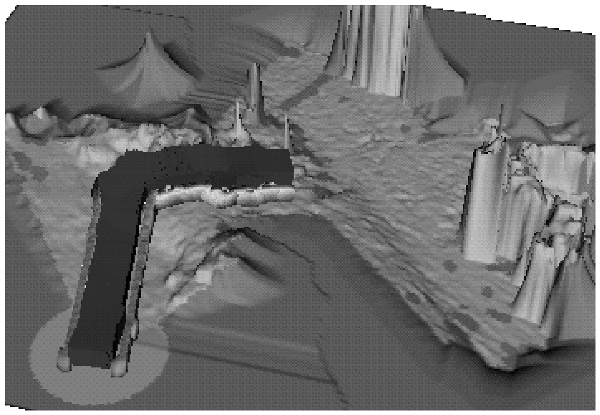
Figure 4: **A 3D trajectory planned on a real elevation map**

The 3D navigation mode has also been integrated (figure 4) and tested on the uneven areas of the terrain, the processing time is here sensibly longer (refer to [19] for more details), which convinced us to persist on the integration of the different navigation modes.

7 Conclusion and Future Work

We have presented a complete autonomous system able to perform autonomous navigation in a natural, unstructured and totally unknown environment. The system is endowed with several complex capabilities : environment modeling, localization, trajectory planning and exploring strategies. An ambitious experimental project, still under way, validates the different processes we developed and benefits to the development of highly demanding robotic applications, in particular planetary exploration.

A lot of difficult tasks have nevertheless still to be achieved, among which we retain the followings :

• The integration of the different navigation modes is still to be achieved : the definition of "smart" navigation strategies is here crucial to enhance considerably the robustness and efficiency of the robot. This topic needs particularly to be better formalized and tested.

• Besides the software integration of the whole system, each process performance needs to be improved and better adapted to the mission context. Feedback provided by the experiments is here an essential information.

• The integration of a stereovision correlation algorithm would enhance the perception capabilities, by providing dense 3D and color data on a particular area. We could then address natural landmark recognition, and estimate the physical nature of the soil during the classification procedure.

Finally, the integration of a task-level tele-programming and supervising interface is an fundamental topic we are currently tackling with : any robot, however smart it is, could not perform faithfully and efficiently a very complex mission without being supervised by an operator.

Acknowledgments

The authors wish to thank all the people who participated to the definition and building of the EDEN experiment : R. Alami, B. Dacre-Wright, P. Fillatreau, P. Gaborit, P. Grandjean, C. Proust, F. Nashashibi... and the "technical gurus" : G. Bauzil, C. Lemaire and G. Vialaret.

References

[1] G.Giralt, "Outdoor mobile robots", *International Conference on Robots for Competitive Industries, Brisbane (Australie)*, July 1993.

[2] G. Giralt, R. Chatila, and R. Alami, "Remote Intervention, Robot Autonomy, And Teleprogramming: Generic Concepts And Real-World Application Cases", *in IEEE - IROS '93, Yokohama, (Japan)*, July 1993.

[3] D. P. Miller, R. S. Desai, E. Gat, R. Ivlev, and J. Loch, "Experiments with a small behaviour controlled planetary rover", *in Misions, Technologies and Design of Planetary Mobile Vehicles, CNES, Toulouse, France*, Sept. 1992.

[4] C. M. Angle and R. A. Brooks, "Small planetary rovers", *in IEEE - IROS '90, Tsuchiura (Japan)*, 1990.

[5] R. A. Brooks, "A robust layered control system for a mobile robot", *IEEE Journal of R. and A.*, Apr. 1986.

[6] R. Alami, R. Chatila, and B. Espiau, "Designing an intelligent control architecture for autonomous robots", *in Third International Symposium on Experimental Robotics, Kyoto, Japan, Oct. 28-30*, 1993.

[7] M. Daily, J. Harris, D. Kreiskey, K. Olion, D. Payton, K. Reseir, J. Rosenblatt, D. Tseng, and V. Wong, "Autonomous cross-country navigation with the alv", *in IEEE Int. Conf. on R. A., Philadelphia (USA)*, 1988.

[8] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, "Vision and navigation for the carnegie-mellon navlab", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 362–373, May 1988.

[9] E. Krotkov, J. Bares, T. Kanade, T. Mitchell, R. Simmons, and R. Whittaker, "Ambler: A six-legged planetary rover", *in '91 Int. Conf. on Advanced Robotics (ICAR),Pisa (Italy)*, pp. 717–722, June 1991.

[10] D. Payton, J. Rosenblatt, and D. Keirsey, "Plan Guided Reaction", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, pp. 1370–1382, Nov./Dec. 1990.

[11] R. Chatila, S. Fleury, M. Herrb, S. Lacroix, and C. Proust, "Autonomous navigation in natural environment", *in Third International Symposium on Experimental Robotics, Kyoto, Japan, Oct. 28-30*, 1993.

[12] S. Betge-Brezetz, R. Chatila, and M.Devy, "Natural scene understanding for mobile robot navigation", *in IEEE Int. Conf. on R. A., San Diego, California*, 1994.

[13] S. Lacroix, P. Fillatreau, F. Nashashibi, R. Chatila, and M. Devy, "Perception for Autonomous Navigation in a Natural Environment", *in Workshop on Computer Vision for Space Applications, Antibes, France*, Sept. 1993.

[14] F. Nashashibi and M. Devy, "3D Incremental Modeling and Robot Localization in a Structured Environment using a Laser Range Finder", *in IEEE Int. Conf. on R. A., Atlanta (USA)*, May 1993.

[15] P. Fillatreau, M. Devy, and R. Prajoux, "Modelling of Unstructured Terrain and Feature Extraction using B-spline Surfaces", *in '93 International Conference on Advanced Robotics (Tokyo (Japan)*, Nov. 1993.

[16] T. Siméon and B. Dacre Wright, "A Practical Motion Planner for All-terrain Mobile Robots", *in IEEE - IROS '93 Japan*, July 1993.

[17] S. Fleury, M. Herrb, and Chatila R, "Design of a modular architecture for autonomous robot", *in IEEE Int. Conf. on R. A., San Diego, California*, 1994.

[18] M.P.Georgeff and F. F. Ingrand, "Decision-Making in an Embedded Reasoning System", *in 11th International Joint Conference on Artificial Intelligence (IJCAI), Detroit, Michigan (USA)*, 1989.

[19] F. Nashashibi, P. Fillatreau, B. Dacre-Wright, and T. Simeon, "3d autonomous navigation in a natural environment", *in IEEE Int. Conf. on R. and A., San Diego, California*, 1994.