Lessons Learned in Integration for Sensor-Based Robot Navigation Systems

Luis Montesano   Javier Minguez   Luis Montano
I3A, Dpto. de Informática e Ing. de Sistemas
Universidad de Zaragoza, Spain
{montesano, jminguez, montano}@unizar.es

***Abstract*— This paper presents our work of integration during the last years within the context of sensor-based robot navigation systems. In our motion system, as in many others, there are functionalities involved, such as modeling, planning or motion control, that have to be integrated within an architecture. This paper addresses this problem. Furthermore, we also discuss the lessons learned while: (i) designing, testing and validating techniques that implement the functionalities of navigation system, and (ii) building the architecture of integration, and (iii) using the system on several robots equipped with different sensors in different laboratories.**

I. INTRODUCTION

Robots are being developed that operate under a wide variety of conditions including unknown, unstructured and dynamic scenarios. Mobility in such scenarios is a key issue to increase the degree of autonomy of a robotic system since it is the basis to incorporate more subsystems and functionalities. Thus, the performance of the motion system strongly affects the task carried out by the vehicle.

The capabilities required for the navigation of an autonomous robot are tied up with the specific application and vehicle. For instance, the a priori knowledge, the information provided by the on-board sensors, the motions constraints of the vehicle or the computational power. This usually leads to the development of specific navigation systems that accommodate the requirements of each application.

One important issue is to bound the scope of the mobility system, which is related to the differences between global and local navigation systems (Figure 1). In fact, the concerns of these systems are different. For instance, for global systems, the construction of accurate models and the tracking of the position of the vehicle are important to create global plans and to guarantee motion convergence, while real-time execution is not. However, for local systems, simpler local models and rough planning are enough, while motion constraints related to real-time or to the vehicle such as shape, kinematics and dynamics are important to guarantee robust obstacle avoidance.

Nevertheless, the mobility aspect is inherently related with some functionalities necessary for a fully autonomous operation (modeling, planning and reaction). More precisely, the topic of motion in evolving environments includes issues such as knowledge representation (model construction), global deliberation and reactivity. Navigational planning without considering execution is restricted to a small domain of the problem. This is because it becomes difficult to consider all contingencies and it is unrealistic to formulate plans that do not reflect a changing environment. On the other hand, reactive motion systems limit their scope of application to the perception-action paradigm, gaining flexibility and robustness of motions. Due to the nature of each methodology, the overall problem cannot be solved by these systems individually. The interest is focused on synthesizing a control mode that incorporates these methodologies, and not on extending both worlds separately [2]. Hybrid systems attempt to combine both paradigms by including the best of the artificial intelligence to represent and use the knowledge, with the best reactivity, robustness, adaptation and flexibility. Basically these schemes combine a planner (deliberation) and a reactor (execution). This work focuses on local navigation, where hybrid approaches have been used in several systems [30], [6], [25], [26]. Our integration scheme follows this approach combining modeling, planning and reactivity:

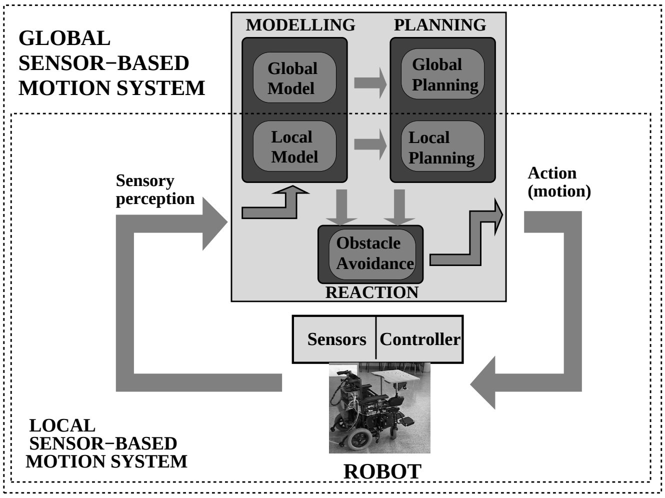


Fig. 1. Global versus Local Sensor-Based Navigation.

1) **Model builder**: construction of a model of the environment (to increase the spatial domain of the planning and used as local memory for obstacle avoidance) and tracking of the vehicle position.
2) **Planner**: extraction of the connectivity of the free space to increase the spatial domain of the solution (for instance used to avoid the cyclical motions and

trap situations).

3) **Reactive motion**: computation of the collision-free motion.

In this context the **integration of functionalities** plays a crucial role. On one hand, the three issues enumerated above are active research areas where the community continuously proposes new methods improving the current state of the art. Thus, the integration architecture must allow a quick module replacement with the most appropriated technologies for each module (they might have different properties that allow to address problems with different nature within the same context). On the other hand, all functionalities must be integrated within an architecture for specification, coordination and failure detection and recovery. The integration must have a clear specification of the interaction of the modules and time constraints. Everything together favors the portability between different platforms and sensors and the easy module replacement to add or change technologies. As a result, the architecture reduces the effort required to upgrade, test and validate new developments.

Existing works only address partially the integration issues of these navigation systems [30], [6], [25], [26]. This paper presents the evolution of our work during the last years within the context of local sensor-based navigation systems focusing on those aspects related to the integration architecture. Moreover, we show experimental results obtained with different real robots that illustrate the benefits of using an architecture of integration.

The work is organized as follows: first we present the evolution of our navigation system (Section II). Section III describes the architecture and Section IV presents the experimental results. Finally, we draw the conclusions in Section V.

II. EVOLUTION OF THE SENSOR-BASED NAVIGATION SYSTEM

The objective of a local motion system is to drive the vehicle among locations while avoiding collisions with obstacles. The operation is governed by a perception - action process repeated at a high frequency (Figure 1). Sensors gather information of the environment (obstacles) and the robot. This information is then processed to compute the motion. The vehicle executes the motion and the process restarts. The result is an on-line sequence of motions that drive the vehicle to the destination without collisions.

In this section we describe an historical perspective of the selection of the techniques, which are closely related with the problems that might be addressed to design a local motion system (Table I).

A. The seed of motion: M_1

Some years ago we started to deal with the mobility problem of autonomous robots. For obstacle avoidance, we selected a potential field method (PFM in short) [9]. Our experience with this obstacle avoidance method [23] confirmed the problems that were described for these type of methods [11]. In fact, at that time many methods exhibited problems to address the motion in troublesome scenarios. Thus, we understood that the first step was to design a method to close the research gap of reactive motion in dense, complex and cluttered scenarios.

B. Motion in Troublesome scenarios: M_2

To address this issue we developed the *Nearness Diagram Navigation* method (ND) [17], [22]. This technique employs a "divide and conquer" strategy to simplify the navigation by identifying situations and applying the corresponding motion laws. The set of situations represents all the cases between robot positions, obstacles and the goal (navigational situations). In addition, for each of these cases a motion law (action) is associated.

The advantage of this method is that it employs a divide and conquer strategy based on situations to simplify the difficulty of navigation. Thus, this technique is able to deal with more complex navigation cases than other methods (usually these cases arise in environments where there is little space to maneuver like for example a narrow door). In particular, the ND method avoids most of the problems that other techniques present in these circumstances (see [17] for a discussion on this topic).

With this new technique, we were able to address motion in places where it was difficult to maneuvers vehicles. However, the problem of trap situations and cyclic behaviors were unavoidable due to the local nature of the obstacle avoidance methods.

C. Trap situations and Cyclic behaviors: M_3

With this problematic in mind, we realized that it was necessary to integrate local planning with obstacle avoidance. Besides, building a local model would also increase the spatial domain of the planner while acting as a memory for the obstacle avoidance method (sensor visibility constraints). The necessity of integrating these functionalities was the beginning of the work described in this paper. Then, we proposed the *Global Nearness Diagram Navigation* (GND) [20], [18].

The GND implements a hybrid architecture with three layers (modeling, planning and reaction). The modeler constructs a representation of the environment integrating the sensory information, which is the base for the rest of modules. We used a robot-centred binary occupancy grid updated whenever a new sensory measurement is available. The planner computes tactical information to direct the vehicle. We implemented the *Navigation Function 1* (NF1 in short, [4]), which is free of potential minima, can work on a grid (existing representation), and can be efficiently executed in real time. The obstacle avoidance computes the collision free motion. This was performed with the ND since it is efficient and robust in environments with little space to maneuver.

The key result was the integration of the modules in a unified system (Section IV provides a detailed description of the architecture). This integration concentrates the best of the deliberative and reactive worlds, since the planning

TABLE I

A SUMMARIZED EVOLUTION OF THE SENSOR-BASED NAVIGATION SYSTEM.

	Modalities					
	M_1	M_2	M_3	M_4	M_5	M_6
Modeling	—	Local laser memory	Binary local grid	Binary local grid	Binary local grid + IDC	Probabilistic map Tracking objects MbICP
Planning	—	—	NF1	NF1	Gap Navigation	D^*
Reaction	PFM	ND	ND	ND+ Abstr. layers or MG	ND+ Abstr. layers or MG	ND+ Abstr. layers or MG
Architecture	no	no	yes	yes	yes	yes

information helps to guide the motion toward zones without traps, and the reactive component quickly directs the execution according to the evolution of the environment. The advantage of this system was to perform robust and trustworthy navigation in difficult scenarios.

D. The vehicle constraints: M_4

At this moment, we addressed the portability of the motion system to different platforms. In order to generate robust obstacle avoidance, the vehicle constraints (shape, kinematics and dynamics) could not be ignored. For this reason we included the vehicle constraints within the obstacle avoidance paradigm with the *Abstraction Layers* [16], [19] and a *Motion Generator* (MG) [3].

The ND and many existing techniques assume that the robot is a point free of any constraint (omnidirectional motion). The idea behind the abstraction layers is to abstract these constraints from the usage of the avoidance methods. A solution is to encapsulate the constraints within the spatial representation. By doing this, we transform the tridimensional obstacle avoidance problem with shape, kinematics and dynamic constraints into the simple problem of moving a point in a bidimensional space without constraints (usual approximation in obstacle avoidance). Thus, many existing methods that do not address these constraints can be applied in this representation. The consequence is that the methods take into account the vehicle constraints without being redesign (the information is implicitly represented in the application space). Alternatively, the motion generator is based on a dynamic motion controller that converts the solution of the obstacle avoidance method into a command that complies with the vehicle kinematic and dynamics.

With these new techniques integrated in the previous system, we take into account the vehicle constraints in the obstacle avoidance module. In parallel, we ameliorated our previous ND version leading to the ND+ [22]. The ND+ method improves the previous method with new navigational situations and a new design of the motion laws (to have motion continuity in the most common transitions between situations). Another advantage of the ND+ method is its efficiency which liberates computational resources for the other modules of the architecture.

E. Local correction of the vehicle localization and time requirements: M_5

At this point in time, the precision of the localization of the vehicle became a serious limitation. In order to deal with vehicles with bad odometry information, it was necessary to correct the robot pose. Models built only with odometry accumulate errors. As the model is the base of the planning and obstacle avoidance methods, it strongly affects the performance of the system. Another important issue at this point was time constraints. The planning method was computationally very demanding and we investigated more efficient planners that do not penalize the reactivity and modeling performance of the system.

To improve the localization of the vehicle, we integrated a scan matching technique that improves the odometry readings using the information provided by the sensors. We used the *Iterative Dual Correspondence* (IDC) algorithm [13]. This technique does not require to extract any specific kind of features and, consequently, is well suited to unstructured environments. Although these techniques do not guarantee global consistency in the model, its precision is enough to build the local map needed by the other modules.

Furthermore, we implemented a planner [21] similar to the *Gap Navigation Trees* [29]. The idea behind this planner is to construct a graph of reachable points of the space, instead of an analytical path as many classical planners do. The graph contains enough tactical information to avoid the trap situations. The advantage of this planner is the computation time since in average is more efficient than computing a local path from scratch with a navigation function.

With this new system we ameliorated the efficiency and the robustness of the local navigation system. However, the performance could still be improved specially in dynamic scenarios.

F. Dynamic Scenarios: M_6

The previous systems do not differentiate between the static structure of the environment and the moving objects. Reactivity against changes in the environment is achieved through a high sensing frequency. However, when dealing with dynamic scenarios, taking into account the nature of the obstacles might ameliorate the performance of the system. A reliable solution must address both: a module

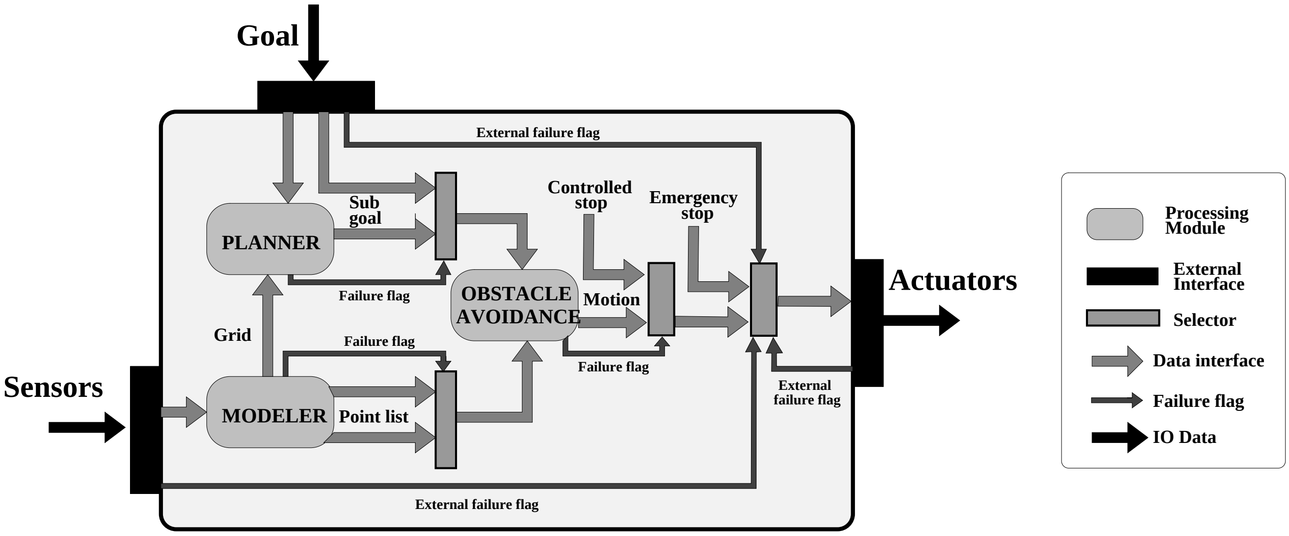


Fig. 2. Overview of the architecture: interaction between modules and data flows.

able to model the static and dynamic parts of the scenario, and a way to use this information within the system.

First, we designed a modeling module that carries out the detection and tracking of moving objects and the mapping of the static parts at the same time [24]. We used a maximum likelihood approach which complies with the spatial and time constraints of the local navigation system. As a result we obtain a map of static obstacles and a separate map of dynamic objects and their velocities. Within this process, we integrated a new scan matching approach [14], the *Metric-based Iterative Closest Point* (MbICP), that ameliorates the IDC performance.

The dynamic/static information is selectively used by the other modules. The role of the tactical planner is to determine at each cycle the main cruise to direct the vehicle. Therefore, the planner only uses the map of static features. The obstacle avoidance method generates the collision-free motion to align the vehicle toward the cruise (computed by the planner). Here we use the map of static obstacles, since all the obstacles included in the map must be avoided. Furthermore, we use information of the dynamic obstacles, but taking advantage of their velocity by projecting their position to the collision point with the vehicle.

At the same time we explored the use of the D^* Lite planner [10]. The principle of this planner is to locally modify the previous path (available from the previous step) using only the changes in the environment. This strategy is by far more efficient than re-computing the path from scratch (up to two orders of magnitude [27]).

This is the current state of the art of our system.

III. ARCHITECTURE DESIGN

This section describes the architecture of the navigation system focusing on those aspects related with the integration of the different functionalities and their interactions.

The system has been designed to work on a single node. This is because many applications in which autonomous motion systems operate are *safety-critical* ([12], [28]) and involve real-time constraints. The architecture is composed of three modules executed following the modeler - planner - reactor sequence dictated by the flow of data between modules (Figure 2). This flow is unidirectional, from the modeling module toward the planner and obstacle avoidance modules. The exact data of each flow depend on the technologies used. These flows define the interactions and dependencies among the modules. Replacing a module requires to comply with the interface and usually does not require to redefine it. For instance, sets of points are a common way to represent obstacles for several obstacle avoidance methods (model-obstacle avoidance interface). The interface between the planner and the obstacle avoidance is just a subgoal location (tactical information). With respect to the model-planner interface, we use a grid. Although there exist many other representations, grids are commonly used to compute navigation functions and are able to represent dense information. The bandwidth required by each flow also varies depending on the modules but remains reasonable (under kilobytes per second).

The modules are executed synchronously. This is important to avoid inconsistencies in time that would arise using asynchronous strategies (the model is used for local planning and obstacle avoidance and must be consistent in time with both modules). Furthermore, we assigned time outs to each module to close the motion control loop at the desired sensor rate. The main purpose of these timeouts is to assure that the obstacle avoidance module is executed every cycle. This is important since the motion of the system is always generated by the avoidance method (assuring collision free motion). Figure 3 shows the control flow of the architecture.

The system also has to manage possible failures (Figure 2). Currently our architecture includes the following exceptions:

- Hardware failure: The architecture monitors the inputs of the sensors and engines. In case of bad operation,

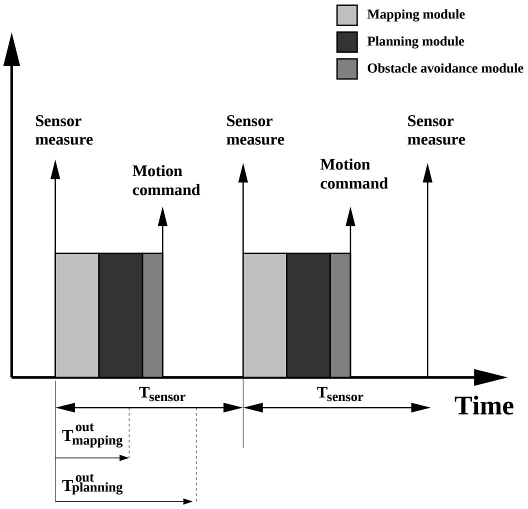


Fig. 3. Control flow of the system.

an emergency stop is executed stopping the vehicle as fast as possible.

- Modeling module failure: Failures of this module are hard to identify and they depend on the implementation. When such a failure occurs or the time out is launched, the usual strategy is to use odometry to keep track of the vehicle position and to re-initialize the map with the last measurement.
- Planning module failure: A failure of this module arises when the planner does not find a solution, either because it does not exist (for example when the goal falls on an obstacle) or because the time out is launched. In this case the information of the planner is not used and the obstacle avoidance tries to move the vehicle directly toward the goal.
- Reactive module failure: the robot is completely surrounded by obstacles when there are no areas of motion free of collision. The vehicle does not progress until a new passage is detected.

Summarizing, the proposed architecture decouples the functional modules necessary for the motion generation and assures their correct interaction and coordination. It also specifies the interfaces with the external modules and hardware devices. The benefits of the integration within the architecture are: (i) to ease the integration of research works developed by different people in different domains, (ii) to improve the software engineering processes specially the final stages of the software life cycle, (iii) to facilitate portability issues.

IV. EXPERIMENTAL VALIDATION

The purpose of this section is to demonstrate that the navigation system successfully carries out the motion task and to show some of the benefits of the architecture: (i) how the architecture allows to easily replace modules and to stay in the cutting edge technologies for local sensor-based motion systems, and (ii) to discuss the portability among different platforms on the basis of the experimental validation obtained in different laboratories (Figure 4).

Firstly, we integrated and tested the system in seven robots at three different laboratories [15]. The results were very satisfactory from the motion execution point of view. The vehicles successfully achieved the motion task in unknown, unstructured and dynamic scenarios, where maneuvering was a determinant factor. One of these implementations has been used daily in a museum for several months [7], and others are daily used for demonstrations [1], [24].

Secondly, with this architecture we have been able to integrate our on going research. This is a key issue in developing time. Thanks to this architecture we have been able to design, integrate, test and validate in real systems more than 20 different technologies in the last four years in our robots and in robots of other laboratories (Table I and Figure 4).

Thirdly, another important issue is the portability among different vehicles. This includes the following aspects:

- Vehicle constraints: the shapes of the robots are circular, square or rectangular. The kinematics are holonomic or differential-drive. The dynamics are also different for all the vehicles.
- Sensors: The sensors used include ultrasounds, 2D and 3D laser range finders, and a stereo vision system.
- Operating systems and computer capabilities: In the robots were installed Linux, Solaris, VxWorks and Windows. The power of the on board computers ranged from a single Pentium II at 200MHz to a Pentium IV at 800MHz.

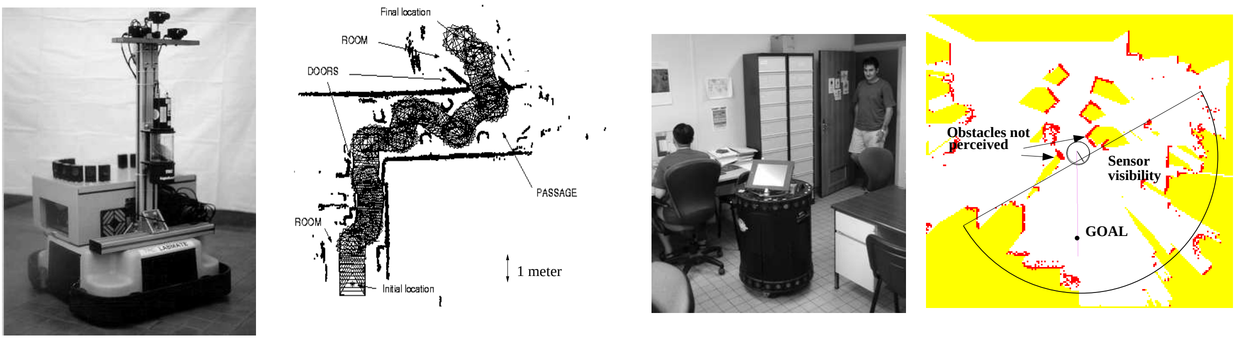
In order to integrate the navigation system in the different vehicles, the first important issue was to take into account the vehicle shape, kinematics and dynamics for obstacle avoidance. This was easily achieved by activating/deactivating the abstraction layers or the motion generator that take into account these issues.

The usage of different sensors required to use the appropriate sensor interfaces. Nevertheless, if the type of data needed a specific processing, the modeling module had to be replaced too. For instance, for the lasers we used the solution described in this paper and we adopted other solutions for the ultrasounds [5] and for the cameras [8]. The important point is that changes affected only to the model builder and the corresponding interfaces.

The different operative systems and computer capabilities were not a problem. It was necessary to build the connexions between the architecture and the external devices. However, since the architecture was developed in standard ANSI C, it was straightforward used in several operative systems. Regarding the different computational power of the platforms, the time outs of the modules balanced the load in order to comply with the real-time requirements and perception-action cycle constraints.

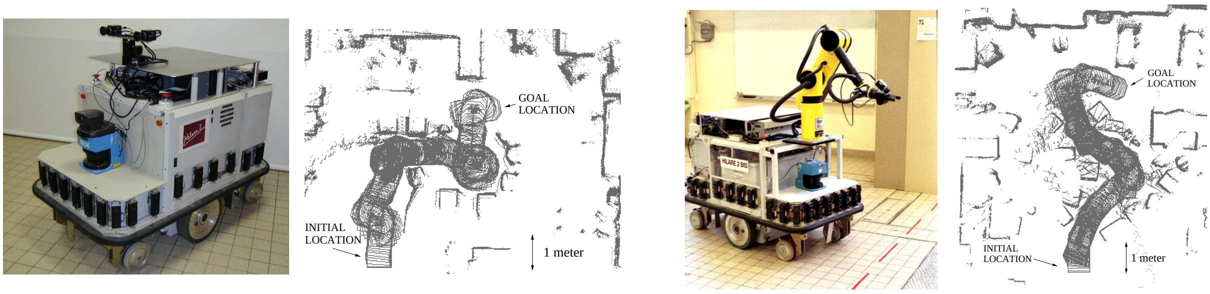
V. CONCLUSIONS

In this paper we have proposed an architecture to integrate the functionalities required to perform local sensor-based navigation. The architecture decouples the main



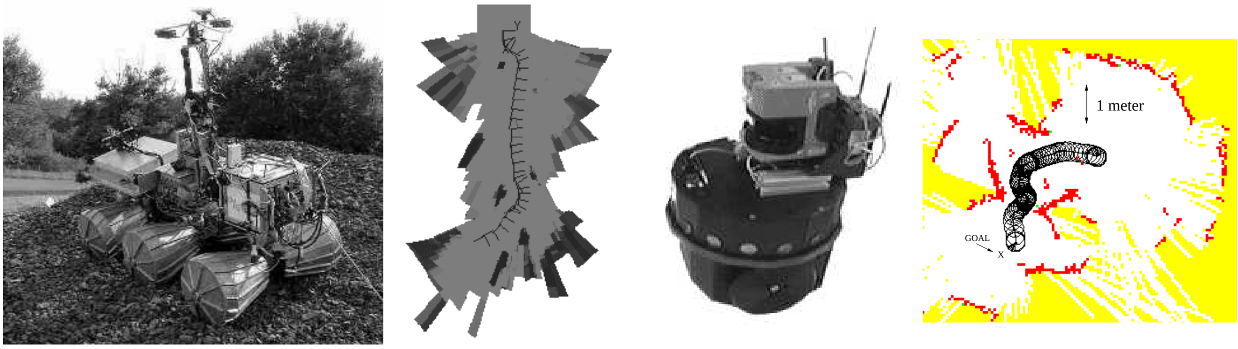
(a) Labmate platform (Univesity of Zaragoza)
Rectangular differential drive, TRC 3D laser

(b) Nomadic platform (LAAS-CNRS)
Circular Holonomic, SICK Laser



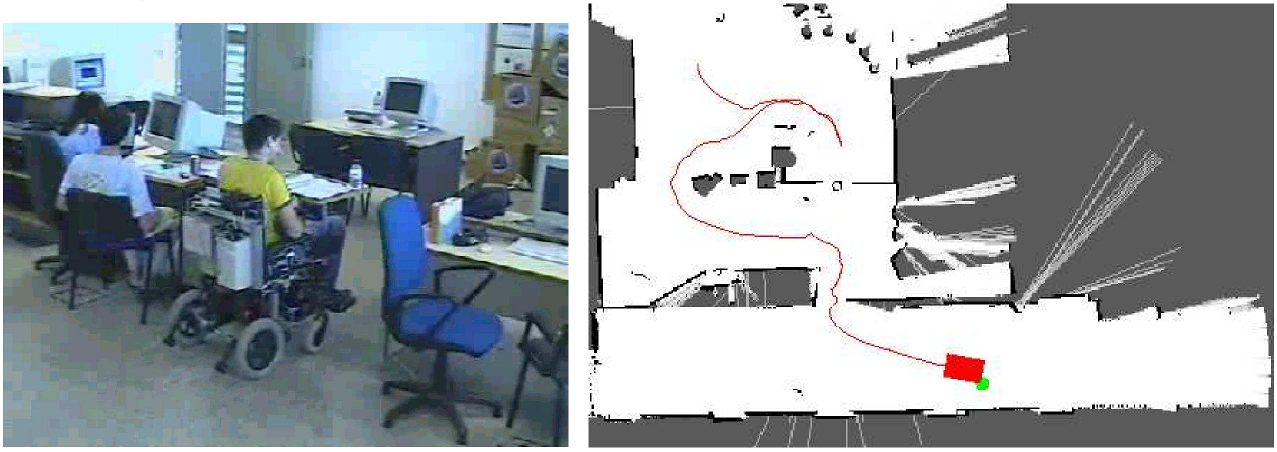
(c) Hilare platform (LAAS-CNRS)
Rectangular differential drive, SICK Laser

(d) Hilare platform with a manipulator (LAAS-CNRS)
Rectangular differential drive, SICK Laser



(e) Lama platform (LAAS-CNRS)
Rectangular differential drive, stereo vision

(f) Scout platform (Technical University of Lisbon)
Circular differential drive, Ultrasounds



(g) Robotic wheelchair platform (University of Zaragoza)
Rectangular differential drive, SICK Laser

Fig. 4. The architecture has been validated using seven different robots. The figure shows each platform and an example of the model and trajectories followed by the robot to reach the goal.

functionalities of the system, defines their interfaces and assures their correct interactions. It provides a framework to continuously upgrade the system with new developments in the field, to ease the development process and to migrate it among different platforms. In addition, we have presented an historical perspective of the technologies and their main characteristics together lessons we learned.

VI. ACKNOWLEDGMENTS

This work was partially supported by MCYT DPI2003-7986.

REFERENCES

[1] R. Alami, I. Belousov, S. Fleury, M.Herb, F. Ingrand, J. Minguez, and B. Morisset. Diligent: Towards a human-friendly navigation system. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2094–2100, Takamatsu, Japan, 2000.

[2] R. Arkin. Towards the unification of navigational planning and reactive control. In *Working Notes of the AIII Spring Symposium on Robot Navigation*, pages 1–6, Stanford University, 1989.

[3] J. Asensio and L. Montano. A Kinematic and Dynamic Model-Based Motion Controller for Mobile Robots. In *15th IFAC World Congress*, Barcelona, Spain, 2002.

[4] J. Barraquand and J.C. Latombe. On nonholonomic mobile robots and optimal maneuvering. In *Intelligent Symposium on Intelligent Control*, pages 340–346, Albany, 1989.

[5] J. Borenstein and Y. Koren. Histogramic in-Motion Mapping for Mobile Robot Obstacle Avoidance. *IEEE Journal on Robotics and Automation*, 7(4):535–539, 1991.

[6] O. Brock and O. Khatib. High-Speed Navigation Using the Global Dynamic Window Approach. In *IEEE Int. Conf. on Robotics and Automation*, pages 341–346, Detroit, MI, 1999.

[7] A. Clodic, S. Fleury, R. Alami, M. Herrb, and R. Chatila. Supervision and interaction: Analysis from an autonomous tour-guide robot deployment. In *International Conference on Advanced Robotics*, Seattle, USA, 2005.

[8] H. Haddad, M. Khatib, S. Lacroix, and R. Chatila. Reactive Navigation in Outdoor Environments Using Potential Fields. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1232–1237, Leuven, Belgium, 1998.

[9] O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. Journal of Robotics Research*, 5:90–98, 1986.

[10] S. Koenig and M. Likhachev. Improved fast replanning for robot navigation in unknown terrain. In *International Conference on Robotics and Automation*, Washington, USA, 2002.

[11] Y. Koren and J. Borenstein. Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1398–1404, Sacramento, CA, 1991.

[12] N. Leveson. *Safeware: System Safety and Computers*. Reading, MA: Addison-Wesley, 1995.

[13] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Intelligent and Robotic Systems*, 18:249–275, 1997.

[14] J. Minguez. Integration of planning and reactive obstacle avoidance for sensor-based navigation. In *IEEE Int. Conf. on Intelligent Robot and Systems*, Edmonton, Canada, 2005.

[15] J. Minguez and L. Montano. Robot Navigation in Very Complex Dense and Cluttered Indoor/Outdoor Environments. In *15th IFAC World Congress*, Barcelona, Spain, 2002.

[16] J. Minguez and L. Montano. The Ego-KinoDynamic Space: Collision Avoidance for any Shape Mobile Robots with Kinematic and Dynamic Constraints. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 637–643, Las Vegas, USA, 2003.

[17] J. Minguez and L. Montano. Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, 2004.

[18] J. Minguez and L. Montano. Autonomous sensor-based motion control in unknown, dynamic and troublesome scenarios. *To appear in Journal of Robotics and Autonomous Systems*, 2005.

[19] J. Minguez, L. Montano, and J. Santos-Victor. Abstracting the Vehicle Shape and Kinematic Constraints from the Obstacle Avoidance Methods. *To appear in Journal on Autonomous Robots*, 2005.

[20] J. Minguez, L. Montano, N. Simeon, and R. Alami. Global Nearness Diagram Navigation (GND). In *IEEE International Conf. on Robotics and Automation*, pages 33–39, Seoul, Korea, 2001.

[21] J. Minguez, L. Montesano, and L. Montano. An architecture for sensor-based navigation in realistic dynamic and troublesome scenarios. In *IEEE Int. Conf. on Intelligent Robot and Systems*, Sendai, Japan, 2004.

[22] J. Minguez, J. Osuna, and L. Montano. A Divide and Conquer Strategy to Achieve Reactive Collision Avoidance in Troublesome Scenarios. In *IEEE International Conference on Robotics and Automation*, Minessota, USA, 2004.

[23] L. Montano and J. Asensio. Real-Time Robot Navigation in Unstructured Environments Using a 3D Laser Rangefinder. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, volume 2, pages 526–532, Grenoble, France, 1997.

[24] L. Montesano, J. Minguez, and L. Montano. Modeling the static and the dynamic parts of the environment to improve sensor-based navigation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[25] R. Philipsen and R. Siegwart. Smooth and efficient obstacle avoidance for a tour guide robot. In *IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan, 2003.

[26] C. Stachniss and W. Burgard. An Integrated Approach to Goal-directed Obstacle Avoidance under Dynamic Constraints for Dynamic Environments. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 508–513, Switzerland, 2002.

[27] A. Stenz. The Focussed D^* Algorithm for Real-time Replanning. In *Proceedings of the Intenational Joint on Artifcial Intelligence*, pages 1652–1659, 1995.

[28] N. Storey. *Safety-Critical Computers Systems*. Reading, MA: Addison-Wesley, 1996.

[29] B. Tovar, L. Guilamo, and S.M. LaValle. Gap navigation trees: Minimal representation for visibility-based tasks. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2004.

[30] I. Ulrich and J. Borenstein. VFH*: Local Obstacle Avoidance with Look-Ahead Verification. In *IEEE Int. Conf. on Robotics and Automation*, pages 2505–2511, San Francisco, USA, 2000.