A “Divide and Conquer” Strategy based on Situations to achieve Reactive Collision Avoidance in Troublesome Scenarios

Javier Minguez Javier Osuna Luis Montano
University of Zaragoza, Spain
{jminguez,osuna,montano}@unizar.es

Abstract— **This paper addresses the reactive collision avoidance for robots that move in arduous environments (i.e. very dense, complex and cluttered). To achieve this goal, the technique simplifies the difficulty of the navigation by a divide and conquer strategy, which is based on identifying navigational situations and applying the corresponding motion laws. The state of the art in reactive navigation still presents classic limitations such as the trap situations due to U-shape obstacles, the difficulty to achieve motion among very close obstacles, to obtain oscillation-free and stable motion, to move over directions far from the goal direction or towards the obstacles, or to tune the heuristic or internal parameters. This paper presents a method that overcomes all these limitations. As a result, navigation with this method is successfully achieved in scenarios where existing techniques present a high degree of difficulty to navigate. Outstanding navigation results are reported using a wheelchair vehicle and a discussion and comparison with existing techniques is provided.**

I. INTRODUCTION

Currently, applications such as emergency response, transportation, deminning, exploration, help care or ludic robots yield the motion control to autonomous navigation systems. The mobile devices, provided with these capabilities, greatly improve their performance and the security of the potential occupants or transported objects. Thus in mobile robotics, we are carrying out a vast effort to improve the robustness of the basic skills of the mobility task, since it is clear that they have a large impact over the whole performance of the system.

The navigation systems work over the vehicles with full autonomy or supervising the human operation. A good example is a robotic wheelchair, which must support both functionalities: complete autonomy (in high degrees of human disability) or supervising the operator orders (in lower degrees of disability). Furthermore, this vehicle requires motion in unstructured, unknown and dynamic scenarios with a high density of obstacles (e.g. moving among chairs, door traversal, humans moving around etc). The navigation performance is crucial for the normal development of this task (human transportation). This paper focuses on the motion generation aspect, which is the particular relevance here because collisions put in risk the safety of the occupant and sudden motions lead to an uncomfortable behavior. We present in this paper a technique able to achieve this objective.

Classically, the motion planning problem is solved by computing a geometrical trajectory avoiding known obstacles (see [1] for a review of techniques). However, the general methods for motion planning are not applicable if the environment is dynamic with a priori unknown behavior, or if it is gradually discovered. Moreover, when both the environment model and robot motion are uncertain (as in the real world, because of sensing inaccuracies) executing a theoretical geometric trajectory is not realistic and the robot is doomed to collide with obstacles. Hence, solving this problem involves sensing directly within the motion planning and control loop. Reactive navigation bridges the motion planning and control loop incorporating the sensory information, thus dealing with unknown and dynamic scenarios in a natural way. Thus, these techniques provide a robust framework to tackle with the mobility problem by taking into account the reality of the environment during the motion.

The classic drawback of these reactive approaches is the **local trap situations**, generally due to the motion between two close obstacles or due to U-shape obstacles. In both situations, the Potential Field Methods [2], [3] produce potential minima that trap the robot [4]. Similarly, the methods based on polar histograms [5], [6], [7] present the difficulty to navigate among close obstacles due to the tuning of a empirical threshold that has to be modified when the obstacle density changes. Traps due to the U-shape obstacles are not avoided by the methods that use constrained optimizations [8], [9], [10]. This is because the optimization loses the information of the environment structure necessary to solve these situations. The methods based on a given path deformed in real-time [11], [12] do not avoid traps when the path lies within U-shape obstacles dynamically created. Recently, some approaches incorporate the connectivity of the space (to avoid local traps) by applying a path-planning algorithm within the control loop [13], [14], [15], [16], [17]. However, these approaches require a large computational load, and thus they difficulty scale to be used in robotic applications with normal computational capabilities or where the resources are shared with other processes.

Other difficulties of existing approaches are: (i) to compute **oscillation free motion** (e.g. Potential Field Methods can produce oscillatory motion between two close obstacles or narrow corridors [4]), (ii) the selection of **motion directions far away from the goal direction**. The reactive methods that make a physical analogy use the goal location directly in the motion heuristic (e.g. [2], [3], [18], [19], [20], [21]), and in those that solve the problem with a constrained optimization (e.g. [8], [9], [17], [10]) one of the balance terms is the goal heading. Thus, with these methods, directions of motion far away from the goal direction are difficult to obtain (in all the situations where they are required). (iii) The selection of **motion directions towards the obstacles** (some methods explicitly prohibit these directions e.g. [6]). In addition, another difficulty found in most of the collision avoidance approaches is the **tuning of the internal parameters**, since in many methods it is difficult to find the optimum values for a good behavior in all the collision avoidance situations (e.g. the relation among the forces in the Potential Field methods [2], [3], [19], the internal threshold in the methods that use a polar histogram [5], [6], [7], or the weights of the terms in the methods that use constrained optimizations [8], [9]).

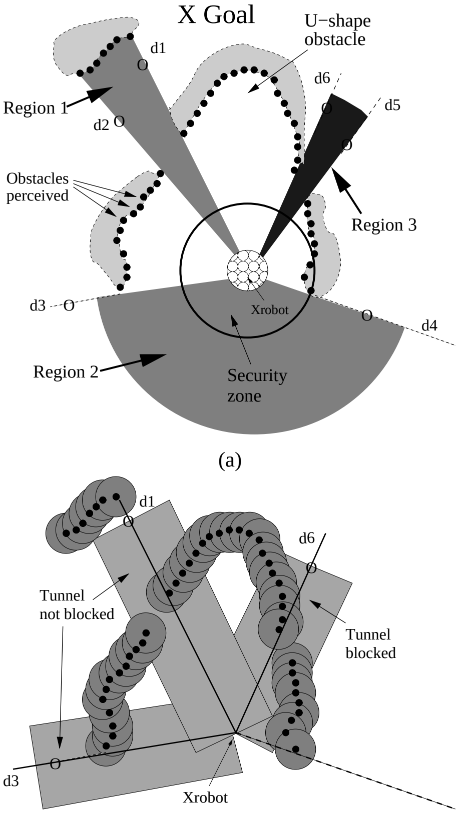
Easy-to-find obstacle configurations such as doors, corridors, humans or tables and chairs forming a U-shape structure could reveal some of the above mentioned limitations of the existing techniques. In this case, a non desirable behavior could be obtained seriously penalizing the task development. We present in this paper a reactive method that overcomes all these limitations. With this method, the trap situations due to very close obstacles or U-shape obstacles are avoided, the motion is oscillatory-free in narrow places, maneuvers that require motion towards the obstacles or far away from the goal direction are obtained when it is required, and there are not heuristic or internal parameters to tune. The technique is based on a "divide and conquer" strategy based on situations to simplify the difficulty of the navigation. Next, specific actions are implemented to each navigation context that allow to avoid the above mentioned difficulties. This idea was previously developed with the ND method [22]. Here, we extend the situations with a new case and we reformulate all the motion laws (leading to the ND+ method). We also discuss here the outstanding navigation results obtained using a robotic wheelchair, and we compare this method with other existing approaches on the basis of the mentioned limitations.

II. THE "DIVIDE AND CONQUER" STRATEGY FOR REACTIVE NAVIGATION

The *situated-activity* paradigm [23] is a design methodology used to design behaviors in robotics. This paradigm is based on defining a set of situations (exclusive and complete) and on designing an action (to accomplish the task) associated to each situation. In execution time, the perception and other information relative to the task is used to identify one situation, and the corresponding action is carried out.

Our idea is to design a reactive navigation method following this paradigm. In general, these methods follow a *perception - action* process repeated at a high rate: based on the sensory information (perception) they compute the collision-free motion command (action) to converge the robot towards the destination. Fig. 2 outlines our reactive navigation method design following the mentioned paradigm, that is, the set of situations and the associated actions. The situations are represented in such a way that only one is identified each time based on the sensory information and the goal location. Next, the corresponding action is carried out computing the motion commands (that drive the vehicle towards the destination whilst collisions are avoided). The motion is executed and the process is resumed.

This scheme is a "divide and conquer" strategy based on situations whose objective is to simplify the difficulty of the navigation, because there is a division in navigation cases (situations) with specific motion heuristics (actions) for each case. The next Section describes the situations and the actions of the reactive method.



(a)

(b)

Fig. 1. (a) This Figure depicts the regions computed in a scenario full of obstacles. (b) The configuration space of the previous scenario with some of the *tunnels* to check whether a region is "navigable".

III. THE REACTIVE COLLISION AVOIDANCE METHOD

We present in this Section the collision avoidance method (ND+ method), assuming a circular and holonomic robot, and the obstacle information available in the form of points (usually the sensory information is given in this form, e.g. laser sensors).

The reactive method works as follows: first, some intermediate entities are identified from the sensory information (described in Subsection III-A), which are used to select one situation of a given set (presented in Subsection III-B). Then, the associated action is carried out computing the motion commands (described in Subsection III-C). Subsequently this process is resumed.

A. The intermediate tools

The intermediate tools are devices that describe relations among the entities of the reactive navigation problem: the robot, the obstacles and the goal location.

Robot and obstacle relation. The first device is the *security zone* used to discriminate whether an obstacle is dangerous

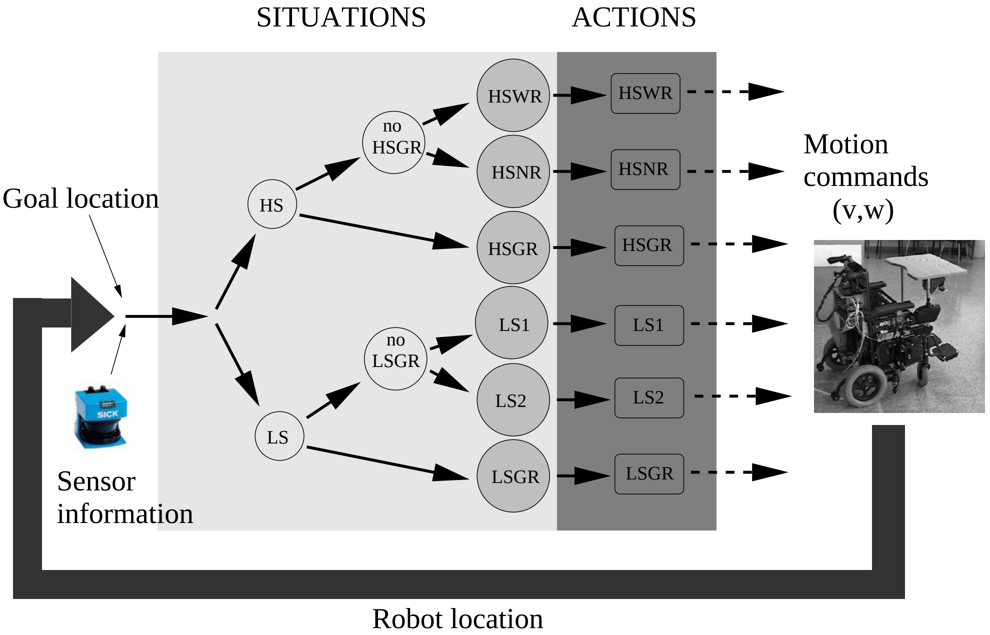


Fig. 2. This Figure illustrates the reactive method design following the *situated-activity* paradigm. Using the sensory information, the robot location and the destination, one situation is selected, and next, the associated action is executed computing the motion commands. Subsequently this process is resumed.

for the robot. We define this area with a security distance, D_s, around the robot bounds. Fig. 1a depicts some obstacles within the security zone in the right-hand side of the vehicle.

Robot and goal relation. The *free walking area* is a device that relates both entities by means of the obstacle distribution structure. To identify it we proceed as follows:

1) We look for *discontinuities* in the obstacle distribution: (i) between two angular contiguous obstacle points whose distance is greater than the robot diameter (e.g. d_1 and d_2 in Fig. 1a), or (ii) between an obstacle point and the absence of obstacles in the angular consecutive correspondent (e.g. $d_3 \ldots d_6$).
2) With two contiguous discontinuities we build a *region* (e.g. Regions $1 \ldots 3$). Notice that this formulation avoids to have regions within the U-shape obstacles (Fig. 1a).
3) We check whether the regions can be crossed by the robot (the algorithm is described in [22]). As a result, we know whether a region can be crossed by the robot (e.g. Regions 1 and 2) or not (e.g. Region 3). In short, this algorithm computes the existence of a path that joins the robot location and significant points of the region in a local portion of the configuration space (that we call *tunnel*). For instance, in Fig 1b the *tunnel* that joins the robot location and the point over d_1 is not blocked (it exist a path joining both configurations), thus Region 1 is “navigable”. However the *tunnel* of the point over d_6 is blocked, and thus Region 3 cannot be crossed (notice that the robot does not fit between the obstacles that create this region).
4) Finally, we select the “navigable” region closest to the goal location. We call this region the free walking area (Region 1 in Fig. 1a).

Next, we use these tools to design the set of situations.

B. The situations

The objective here is to find a set of situations that: (i) fully describe all possible obstacle configurations, and robot and goal locations (the situation representation has to be *complete*), (ii) given an obstacle configuration and a robot and goal locations, there is only one situation that represents it (the representation is also *exclusive*).

To fulfill both premises, we represent the situations using a binary *decision tree* (Fig. 2). The inputs of the tree are the obstacles (sensory information) and the robot and goal locations, which allow us to identify one situation (output). The tree is traversed using binary decision rules based on criteria that depend on the inputs and their relations (intermediate tools, previous Subsection). We describe next the five criteria:

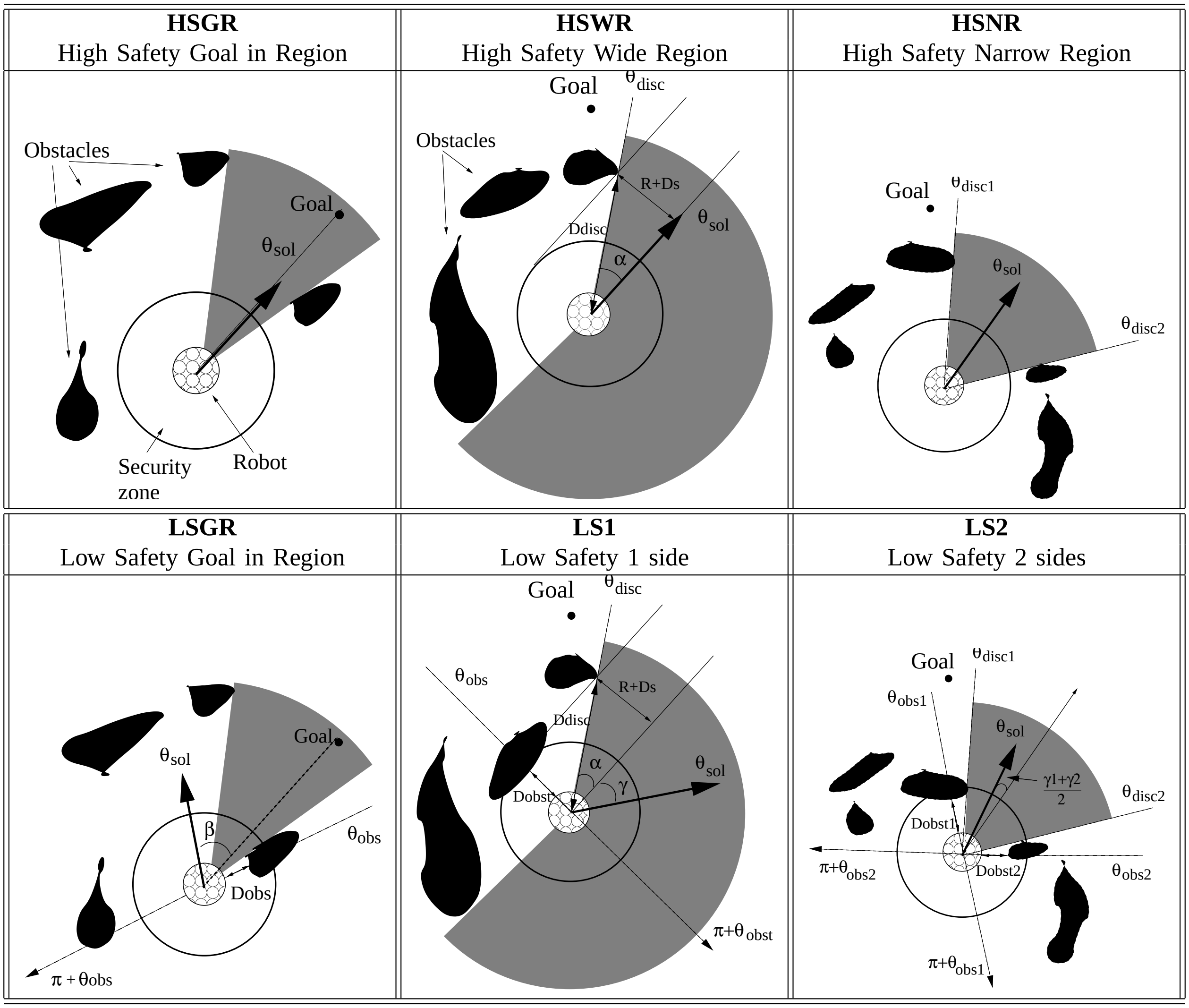
Criterion 1: Safety criterion. The are two safety situations depending on whether there are obstacles within the security zone: (i) if there are not obstacles (High Safety, HS in Fig. 2), see the upper figures of Fig. 3, or (ii) otherwise (Low Safety, LS), see the bottom figures of Fig. 3.
There are three situations in High Safety (HS), see Fig. 2. We obtain the first of them by applying the next criterion:

Criterion 2: Goal within the free walking area criterion.

1) **High Safety Goal in Region** (HSGR): The situation is HSGR when there are not obstacles within the security zone, and the goal location is within the free walking area (Fig. 3).

If not, we obtain the next two situations by applying the following criterion:

Criterion 3: Free walking area width criterion. To differ between *wide* or *narrow* free walking area we use a fix angular width.



HSGR
High Safety Goal in Region

HSWR
High Safety Wide Region

HSNR
High Safety Narrow Region

LSGR
Low Safety Goal in Region

LS1
Low Safety 1 side

LS2
Low Safety 2 sides

Fig. 3. This Figure shows examples of the situations and the actions.

2) **High Safety Wide Region** (HSWR): The situation is HSWR if the free walking area is wide (Fig. 3).
3) **High Safety Narrow Region** (HSNR): The situation is HSNR if the free walking area is narrow (Fig. 3).

There are three situations in Low Safety (LS), see Fig. 2. We obtain the first of them by applying the next criterion:

Criterion 4: Goal within the free walking area criterion.

4) **Low Safety Goal in Region** (LSGR): The situation is LSGR when there are obstacles within the security zone, and the goal location is within the free walking area.

If not, we obtain the last two situations by applying the following criterion:

Criterion 5: Dangerous obstacle distribution criterion.

5) **Low Safety 1 side** (LS1): The situation is LS1 when there are obstacles within the security zone, but only on one side of the discontinuity (closest to the goal) of the free walking area (this discontinuity is θ_{disc} in the LS1 figure of Fig. 3).
6) **Low Safety 2 sides** (LS2): The situation is LS2 when there are obstacles within the security zone on the two sides of the discontinuity (closest to the goal) of the free walking area (this discontinuity is θ_{disc_1} in the LS2 figure of Fig. 3).

This set of situations is exclusive and complete to cover all the possibilities among the obstacle distribution and robot and goal location. In other words, there is only one navigational situation given an obstacle distribution, robot location and destination. Once a situation is selected, the next step is to execute the associated action to compute the motion. We address the design of the actions in the next Subsection.

C. The actions

The actions associated to each situation compute the motion commands to carry out the main task (avoid collisions whilst moving the vehicle to the destination), but adapted to the navigation context represented by each situation. Next, we illustrate the computation of the motion direction, θ_{sol} in each situation.

In High Safety there is no need to avoid obstacles, because none of them is dangerous. The solutions drive the vehicle towards the inside of the free walking area:

1) **High Safety Goal in Region** (HSGR): the direction solution is to the goal location $\theta_{sol} = \theta_{goal}$.
2) **High Safety Wide Region** (HSWR): the direction solution drives the vehicle towards the side of the free walking area (closest to the goal), plus a given angle to prevent that the obstacle enters in the security zone.

$$\theta_{sol} = \theta_{disc} \pm \alpha \tag{1}$$

$$\alpha = \arcsin(\frac{R + D_s}{D_{disc}}) \tag{2}$$

where D_{disc} and θ_{disc} are the distance and direction of the discontinuity (side of the free walking area). Notice that α is the deviation that allows the robot to avoid that the obstacle (that creates the discontinuity) enters in the security zone.

3) **High Safety Narrow Region** (HSNR): the direction of motion is to the central part of the free walking area:

$$\theta_{sol} = \frac{\theta_{disc_1} + \theta_{disc_2}}{2} \tag{3}$$

where θ_{disc_1} and θ_{disc_2} are the directions of the discontinuities (the sides of the free walking area).

In Low Safety there are risky obstacles to avoid. The solutions drive the vehicle as it would be in High Safety but with some additional terms that depend on the closer obstacles:

4) **Low Safety Goal in Region** (LSGR): the direction is to the goal location (solution in HSGR) plus a deviation that depends on the distance to the closest obstacle:

$$\theta_{sol} = \theta_{goal} \pm \beta \tag{4}$$

$$\beta = \frac{D_s - D_{obs}}{D_s}.|(\pi + \theta_{obs}) - \theta_{goal}| \tag{5}$$

where D_{obs} and θ_{obs} are the distance (to the robot bound) and the direction of the closest obstacle. The angle β is proportional to the angle comprised between the goal direction, θ_{goal}, and the opposite direction to the closest obstacle, $\pi + \theta_{obs}$ (complete avoidance of this obstacle).

5) **Low Safety 1 side** (LS1): the direction is the same that would be obtained in HSWR plus a deviation that depends on the distance to the closest obstacle:

$$\theta_{sol} = \theta_{disc} \pm (\alpha + \gamma) \tag{6}$$

$$\gamma = \frac{D_s - D_{obs}}{D_s}.|(\pi + \theta_{obs}) - |\theta_{disc} - \alpha|| \tag{7}$$

where α is given by Eq. (2) and the deviation γ is proportional to the angle comprised between the direction of the discontinuity θ_{disc} minus α, and the opposite direction to the closest obstacle.

6) **Low Safety 2 sides** (LS2): the direction is the same that would be obtained in HSNR plus a deviation that depends on the distances to the two closest obstacles:

$$\theta_{sol} = \frac{\theta_{disc_1} + \theta_{disc_2}}{2} \pm \frac{\gamma_1 + \gamma_2}{2} \tag{8}$$

where γ_1 and γ_2 by are given by Eq. (7) for the closest obstacles on both sides of the discontinuity (closest to the goal) of the free walking area. The deviation here is the bisector of the deviations obtained for each obstacle separately.

In summary, we have presented in this Section the design of a reactive method that uses a “divide and conquer” strategy based on situations. We have discussed the design of the situations and the corresponding motion laws in each navigation case. In the next Section we show the experimental results.

IV. EXPERIMENTAL RESULTS

For experimentation, we used a commercial wheelchair that we have equipped with two on-board computers, and with a SICK laser. The vehicle is rectangular $(1.2 \times 0.7 meters)$ with two tractor wheels that work in differential-driven mode. We set the maximum operational velocities to $(v_{max}, w_{max}) = (0.3 \frac{m}{sec}, 0.7 \frac{rd}{sec})$ due to the application context (human transportation). The reactive method run at more than $2000Hz$ on the on-board $PentiumIII 850Mhz$. However, we reduced this frequency to $5Hz$ (frequency of the laser) sleeping the process in order to have a *perception-action* scheme.

Notice that we have presented the reactive method assuming that the robot can move in any direction and it is circular. In order to adapt this technique to the rectangular and nonholonomic robot, we followed the solution proposed in [24]: in a first step, the direction solution computed by the reactive method is converted into commands that comply with the kinematics and dynamics using a kinodynamic model of the robot. Next, these commands are modified if collisions appear due to the robot shape (rectangular here). As a result, the commands computed tend to align the vehicle with the instantaneous direction of motion whilst avoiding collisions.

We outline next three experiments carried out in unknown, unstructured and dynamic office-type environment, where the goal location was the only information provided in advance.

In the first experiment, the robot moved avoiding two large U-shape structures (formed by furniture) to reach the goal location (see a snapshot in Fig. 4a and the complete trajectory and the laser points in Fig. 4d). First, the robot turned right-hand to avoid the frontal U-shape structure, following motion directions far away from the goal direction (some of these directions differ in more than 90°). Next, the vehicle moved to the bottom part to cross the narrow corridor, whilst avoided moving within the U-shape structure to the right-hand side. Within the corridor the vehicle performed a smooth trajectory without oscillations or any unstable behavior. Finally the vehicle turned right-hand to reach the goal. During the experiment, directions of motion towards obstacles were selected in almost all the experiment, since the laser has a large field of view (8meters) which was about the office size. The computational load of the algorithm in each cycle was almost constant ($< 0.5msec$), the experiment was accomplished in $68sec$ and the velocity profiles are shown in Fig. 4g.

The next experiment was similar to the previous one but with two added difficulties that converted the scenario in a

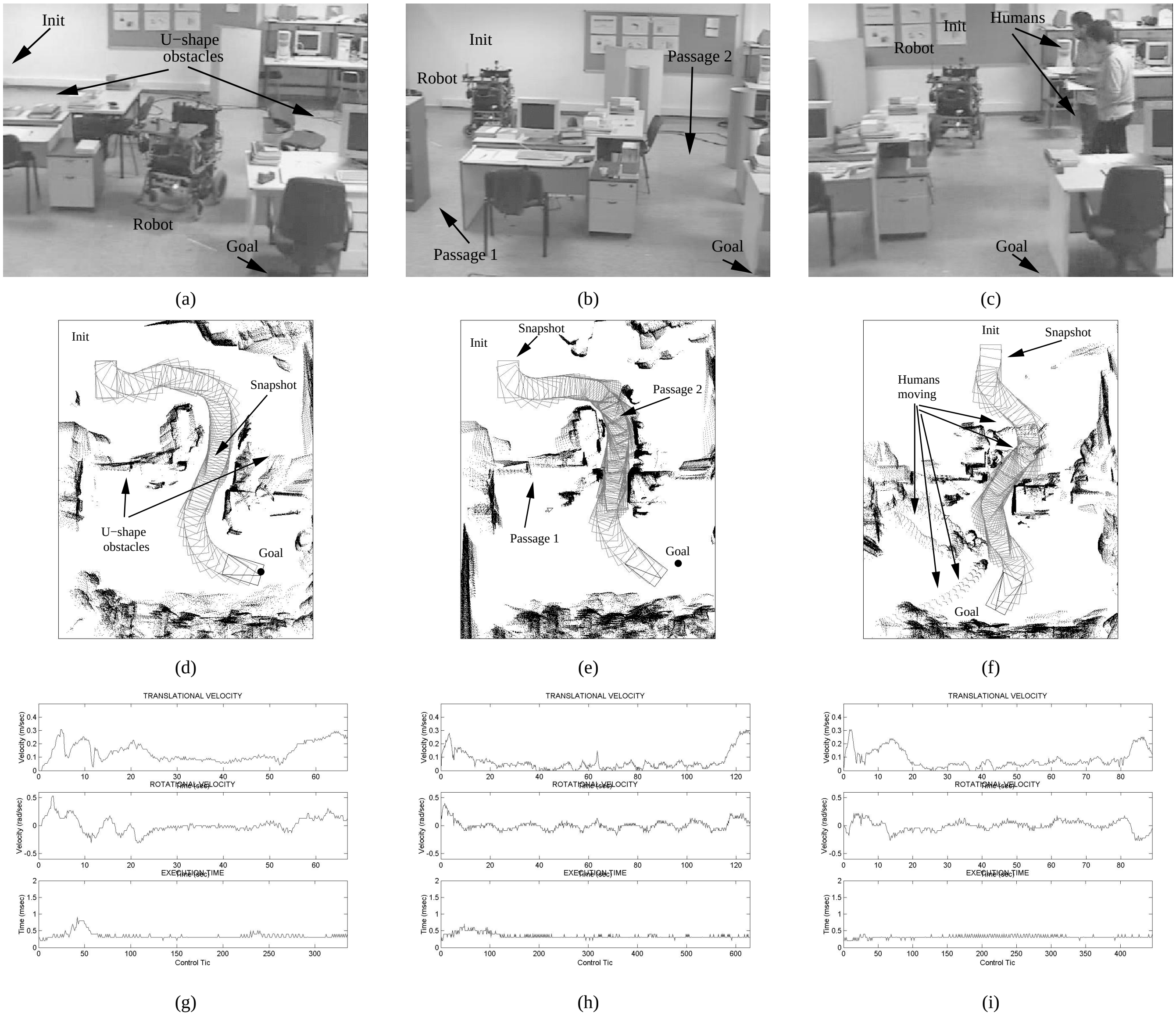


Fig. 4. Experiments carried with a wheelchair vehicle in typical office enviroments.

particularly challenging one. The first one was that we opened a narrow passage (*Passage* 1 in Fig. 4b) where the robot did not fit in (in Fig. 4e it seems to be closed because it is blurred due to the robot drift). On-line, the robot detected that it did not fit in and moved to the right-hand avoiding the large U-shape structure formed by the furniture and this narrow passage. Notice that directions of motion far from the goal direction were required to solve this situation. The next difficulty was found in the central corridor, since it was extremely narrowed (*Passage* 2 in Figs. 4b,e). Again on-line, the vehicle detected that it was possible to cross this narrow place and proceed to do it. While the vehicle was moving within the corridor there was very little room to maneuver ($< 10cm$ on both sides). However, there were not oscillations or unstable behaviors. Finally the robot reached the goal location without collisions. The computational load of the algorithm in each cycle was almost constant ($< 0.5msec$), the experiment was accomplished in $125sec$ and the velocity profiles are shown in Fig. 4h.

In the last experiment the vehicle was driven in the office but with humans moving around (what converted the scenario in a dynamic place) (Figs. 4c,f). In the first part of the experiment, the vehicle maneuver to the right-hand to avoid the humans that were moving towards the table on the left-hand. Then, it was checked that the vehicle fitted in the very narrow passage (the same size of the previous experiment) and thus proceed to cross it. Finally other humans hindered the vehicle motion at the exit of the passage. However, collisions were avoided with smooth motions (see the velocities profiles, Fig. 4i) until the vehicle reached the goal location. The experiment was accomplished in $88sec$.

V. DISCUSSION

We discuss next the advantages of this method with respect to existing techniques on the basis of the difficulties and limitations mentioned in Section I, some questions about the method design and the portability among different vehicles and sensors.

The **local trap situations** due to U-shape obstacles or due to the motion among close obstacles are overcome with this method. The vehicle avoids entering and getting trapped within U-shape obstacles because there are no regions within these structures, and thus these areas are not selected for motion. Fig. 1a depicts this situation, where there is not region within the U-shape obstacle located between the vehicle and the goal. Thus, these situations are avoided in a natural way. In the beginning of the first two experiments the vehicle avoided the large U-shape structures moving to the right-hand side, since the free walking area was located in this direction. On the other hand, there is no difficulty to move among very close obstacles because: (i) the possibility of whether the vehicle fits is checked with the free walking area (see e.g. how *Passage* 1 is not selected for motion and the vehicle proceed through *Passage* 2 in Fig. 4b,e), and (ii) the action in this situation (LS2) centers the vehicle among the closest obstacles, that “a priori” is the safest motion (notice how the vehicle navigated within the narrow *Passage* 2 in Fig. 4b,e). Furthermore, when moving among very close obstacles we observed **motion free of oscillations or unstabilities** (see the smooth paths generated in Figs. 4d,e,f and the velocity profiles of the experiments Figs. 4g,h,i).

With this method, the **selection of directions far away from the goal direction** when required is not a problem. This is because the goal direction is only used in two of the six situations (both when the goal is within the free walking area), which do not exhibit any difficulty. Furthermore, any deviation from the goal direction can be obtained with this method. This property was determinant to successfully accomplish the experiments. For example in the first two experiments, the only way to reach the goal was to move right-hand that implies directions far from the goal direction. In addition, in the action formulation of the method nothing prohibits the **selection of directions of motion towards the obstacles**, thus they were selected during all the experiments almost every time. This is because the laser has a field of view of 8 meters and thus moving forward implies to move towards obstacles. However, this issue was critical when moving in narrow corridors, because sometimes only this type of directions allows to move safely.

There are not **internal parameters** with this method. Only the security distance has to be set with a coherent value that we chose ($D_s = 0.75m$ the shorter side of the vehicle), and the width of the free walking area (90°).

We have seen how this method overcomes the limitations and problems of previous works, which leads to the outstanding results obtained in scenarios that remain very troublesome for existing techniques. On the other hand, there are still some open questions regarding the method design, such as (i) how continuous is the motion among the transitions between situations, and (ii) the implications of the computational load.

The changes in the direction of motion in the transition between situations are smooth because, working at a high rate, the structure of the scenario and the robot location do not have significant changes, and as the situation and the corresponding action are computed based on these variables, changes in direction are also smooth. Furthermore, **the actions are continuous between the common transitions among the situations** (HSGR⟺LSGR or HSWR⟺LS1 or HSNR⟺LS2). For instance a typical situation is when the robot is navigating in HSGR situation, then the direction of motion is towards the goal location (Fig. 3). Then, if an obstacle shows up in the security zone the situation turns to be LSGR, and thus there is a deviation from the goal direction in proportion of how close is the obstacle from the robot (Fig. 3). This deviation is applied until the security zone is cleared and the situation turns to be HSGR again resuming the motion towards the goal. This resulted in a smooth motion since the HSGR and LSGR actions are continuous. Similar conclusions derive for HSWR⟺LS1 or HSNR⟺LS2 actions (Fig. 3). This is illustrated in the smooth velocity profiles obtained in the real experiments (Figs. 3g,h,i), which in addition rely on a comfortable motion with the wheelchair.

Another remaining point is the computational load because it sets the reactivity of the system. The execution time of the reactive method was less than $0.5msec$ (Figs. 4g,h,i) on a $PentiumIII850Mhz$. In other words, once a new sensor reading is available, in less than $0.5msec$ there is a motion command ready to be executed (thus **the reaction is immediate**). Furthermore, another advantage is that the processor is free the majority of the time (the reactive method consumes $< 2.5\%$), and then it could be employed by other modules that require higher computational loads (e.g. path planning, simultaneous location and map building, supervisors, etc). These experiments illustrate the advantage of using these techniques in unknown and dynamic places, where the sensory information is collected and the motion rapidly computed to react to the change. Notice that using other techniques that require higher computational loads (e.g. path planning methods) do not have these benefits, and thus they have a limited applicability in such circumstances.

An important issue is the method usage in different vehicles. In this paper we have adopted the solution proposed in [24], that allows to convert a direction of motion to commands that comply with the shape, kinematic and dynamic constraints of a given vehicle. This procedure is based on a kinodynamic model, and it has been followed to use a reactive method in robots of different laboratories (five indoor and one outdoor robots). Similarly, this procedure could be followed to install the method in other vehicle (**easy portability**). Recently, it has been proposed a solution to adapt reactive navigation methods to vehicles that exhibit non circular shapes, and kinematic and dynamic constraints [25]. This technique is based on a spatial representation that abstracts the vehicle characteristics from the reactive method. As a result, when the reactive methods are used over this spatial representation (abstraction layer) the motions comply with all these issues. This solution could also be adopted here. Finally, very linked with the vehicles are the on-board sensors. We have presented the method abstracted from the sensor to achieve the maximum generality. Thus, issues as the sensor noise have not been addressed. This is because we believe that external modules should process the sensory information in order to deal with noisy sensors (e.g. [26], [27]). However, strategies such as increasing the security distance according to the sensor uncertainty could be designed.

VI. CONCLUSIONS

This paper presents a reactive collision avoidance method that simplifies the difficulty of the navigation by a divide and conquer strategy, which is based on identifying a navigation situation and applying the corresponding motion heuristic. This type of idea was previously proposed in [22] by means of the design of a reactive method in symbolic level, in such a way that reactive methods are implemented following these design guidelines. The reactive method presented in this paper extends the situation definition and proposes a design of the actions to have continuity among them.

The advantage of the proposed technique is that it avoids the common limitations identified in existing techniques, thus it significantly overcomes the results obtained with other methods. Although this is the advantage of the method, another key point is the simplicity. Taking a look to Fig. 3 one can devise how simple is the method formulation and thus the real implementation. Furthermore, similar methods have been straightforward extended to different vehicles using the same solution adopted here. Thus, the authors think that it might also be direct to use this technique in other platforms (portability).

The usage of this method does not avoid the problems inherent to the local nature of reactive navigation methods: the global trap situations persist. This issue is far beyond the scope of this work, however, this method could be used together with techniques that aim to increase the locality of reactive methods, such as those described in [7], [13], [14], [15]. Then, these undesirable situations would be mitigated.

VII. ACKNOWLEDGMENTS

The authors want to thank José Luis Villarroel of the University of Zaragoza (Spain) for his valuable comments and discussions. This work was partially supported by MCYT DPI2000-1272.

REFERENCES

[1] J. C. Latombe, *Robot Motion Planning,* Kluwer Academic, 1991.

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

[3] B. H. Krogh and C. E. Thorpe, "Integrated Path Planning and Dynamic Steering control for Autonomous Vehicles," in *IEEE Int. Conf. on Robotics and Automation*, San Francisco, USA, 1986, pp. 1664–1669.

[4] Y. Koren and J. Borenstein, "Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation," in *IEEE Int. Conf. on Robotics and Automation*, Sacramento, CA, 1991, vol. 2, pp. 1398–1404.

[5] J. Borenstein and Y. Koren, "The Vector Field Histogram–Fast Obstacle Avoidance for Mobile Robots," *IEEE Transactions on Robotics and Automation*, vol. 7, pp. 278–288, 1991.

[6] I. Ulrich and J. Borenstein, "VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots," in *IEEE Int. Conf. on Robotics and Automation*, 1998, pp. 1572–1577.

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

[8] D. Fox, W. Burgard, and S. Thrun, "The Dynamic Window Approach to Collision Avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, 1997.

[9] R. Simmons, "The Curvature-Velocity Method for Local Obstacle Avoidance," in *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, USA, 1996, pp. 3375–3382.

[10] W. Feiten, R. Bauer, and G. Lawitzky, "Robust Obstacle Avoidance in Unknown and Cramped Environments," in *IEEE Int. Conf. on Robotics and Automation*, San Diego, USA, 1994, pp. 2412–2417.

[11] S. Quinlan and O. Khatib, "Elastic Bands: Connecting Path Planning and Control," in *IEEE Int. Conf. on Robotics and Automation,* Atlanta, USA, 1993, vol. 2, pp. 802–807.

[12] O. Brock and O. Khatib, "Real-Time Replanning in High-Dimensional Configuration Spaces using Sets of Homotopic Paths," in *IEEE Int. Conf. on Robotics and Automation*, San Francisco, USA, 2000, pp. 550–555.

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

[14] J. Minguez, L. Montano, N. Simeon, and R. Alami, "Global Nearness Diagram Navigation (GND)," in *IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, 2001, pp. 33–39.

[15] 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*, Switzerland, 2002, pp. 508–513.

[16] N. Roy and S. Thrun, "Motion Planning through Policy Search," in *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Switzerland, 2002, pp. 2419–2424.

[17] K. Arras, J. Persson, N. Tomatis, and R. Siegwart, "Real-time Obstacle Avoidance for Polygonal Robots with a Reduced Dynamic Window," in *IEEE Int. Conf. on Robotics and Automation*, Washington, USA, 2002, pp. 3050–3055.

[18] R. B. Tilove, "Local Obstacle Avoidance for Mobile Robots Based on the Method of Artificial Potentials," in *IEEE Int. Conf. on Robotics and Automation*, Cincinatti, OH, 1990, vol. 2, pp. 566–571.

[19] J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989.

[20] K. Azarm and G. Schmidt, "Integrated mobile robot motion planning and execution in changing indoor environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Munchen, Germany, 1994, pp. 298–305.

[21] A. Masoud, S. Masoud, and M. Bayoumi, "Robot navigation using a pressure generated mechanical stress field, the biharmonical potential approach," in *IEEE International Conference on Robotics and Automation*, San Diego, USA, 1994, pp. 124–129.

[22] J. Minguez and L. Montano, "Nearness Diagram Navigation (ND): Collision Avoidance in Troublesome Scenarios," *IEEE Transactions on Robotics and Automation (February)*, vol. 20, no. 1, pp. 45–59, 2004.

[23] R.C. Arkin, *Behavior-Based Robotics*, The MIT Press, 1999.

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

[25] 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*, Las Vegas, USA, 2003, pp. 637–643.

[26] J. Borenstein and Y. Koren, "Histogramic in-Motion Mapping for Mobile Robot Obstacle Avoidance," *IEEE Journal on Robotics and Automation*, vol. 7, no. 4, pp. 535–539, 1991.

[27] A. Elfes, "Sonar-based Real-world Mapping and Navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.