Robotica (2014) volume 32, pp. 591–609. © Cambridge University Press 2013
doi:10.1017/S0263574713000878

Vision-based mobile robot motion control combining T^2 and ND approaches

Francisco Bonin-Font*, Javier Antich Tobaruela, Alberto Ortiz Rodriguez and Gabriel Oliver

Systems, Robotics and Vision Group, Department of Mathematics and Computer Sciences, University of the Balearic Islands, Palma de Mallorca, Islas Baleares, Spain

(Accepted August 2, 2013. First published online: September 6, 2013)

SUMMARY

Navigating along a set of programmed points in a completely unknown environment is a challenging task which mostly depends on the way the robot perceives and symbolizes the environment and decisions it takes in order to avoid the obstacles while it intends to reach subsequent goals. *Tenacity and Traversability* (T^2)[1]-based strategies have demonstrated to be highly effective for reactive navigation, extending the benefits of the artificial Potential Field method to complex situations, such as trapping zones or mazes. This paper presents a new approach for reactive mobile robot behavior control which rules the actions to be performed to avoid unexpected obstacles while the robot executes a mission between several defined sites. This new strategy combines the T^2 principles to escape from trapping zones together with additional criteria based on the Nearness Diagram (ND)[13] strategy to move in cluttered or densely occupied scenarios. Success in a complete set of experiments, using a mobile robot equipped with a single camera, shows extensive environmental conditions where the strategy can be applied.

KEYWORDS: Mobile robots; Visual navigation; Reactive systems; Motion control; Land autonomous robots.

1. Introduction

1.1. The navigation problem

In the context of autonomous mobile robots, the navigation problem deals with the ability of moving between several target points, avoiding all collisions and trying to maximize efficiency in terms of, typically, time and covered path length.

Current applications involving mobile autonomous agents entrust motion and behavior control to navigation strategies. Robust skills in mobility and navigation have a great influence in the system global performance and in the correct achievement of all the programmed objectives. It is not enough being able to detect and avoid obstacles and perfectly localize the robot while it moves, but also needs to head the robot toward the goal.

Classic localization and navigation approaches usually involve the use of range sensors such as laser scanners,[10] ultrasounds,[6] or even infrared-based systems.[16] Conventional laser scanners and ultrasounds emit in a limited horizontal field of view (FOV) and infrared light is almost useless under water. Lately, cameras have emerged as competitive alternatives, thanks to their relative low cost, the richness of the provided sensor data, and the larger spatial and temporal resolution available even under the water.

The literature is profuse with vision-based obstacle detection, navigation, and localization strategies.[2] Essentially, navigation approaches can be grouped as either map-based or mapless.

* Corresponding author. E-mail: francisco.bonin@uib.es

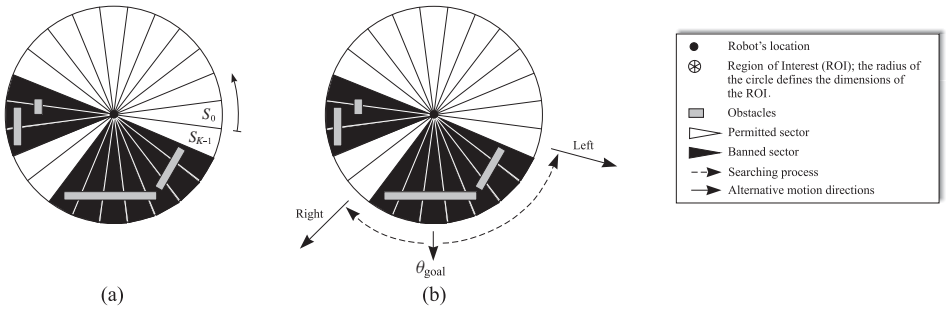

Fig. 1. Understanding the traversability principle: (a) Division of the region of interest into K sectors (S_0,..., S_{k-1}). White sectors are permitted and black sectors are banned. (b) Selection of two possible obstacle-free motion directions (labeled as left and right) when the goal direction (θ_{goal}) is occupied; both directions correspond to the first permitted sector at both sides of θ_{goal}.

Map-based systems incorporate the map of the environment for self-localization and navigation and they usually plan their motion according to this map, while mapless systems, the so-called reactive systems, do not assume any knowledge of the environment and apply the paradigm of *perception and action*. Among all of these, Bonin *et al.*[3,4] dealt simultaneously with both localization and obstacle avoidance problems in a reactive context, using for both tasks the same visual environmental data obtained using the concept of the Inverse Perspective Transformation (IPT).

This paper complements the work presented in Bonin *et al.*[3,4] with a new control architecture that integrates obstacle detector and localizer with an efficient behavior-based reactive navigation approach. T^2[21] and ND[13] are the techniques that provide the most important theoretical insights for the particular solution presented here. Combining the benefits of both in a single design and refusing their drawbacks increase the scope of hazardous situations that can be tackled by an autonomous system.

1.2. T^2 and ND

A local trapping zone is one of the most limiting situations to succeed in a reactive navigation task using the Potential Fields method.[12] In the presence of a certain type of obstacles, basically environment concavities (large U-, L-, G-shaped), robots can get stuck in a cyclic and infinite process of attraction and rejection caused by the goal and the obstacle, respectively.

One of the ways to escape from this situation is to plan a strategy to move the robot away from the target direction, and that is exactly what Antich *et al.*[1] proposed successfully complementing the Potential Fields method with two main principles: traversability and tenacity.

Roughly speaking, the traversability principle is supported by a region of interest of a pre-set size which holds the representation of surrounding scenario as a set of banned or permitted sectors, depending, respectively, on the presence or absence of obstacles. This representation of the surrounding environment is called the *navigation filter*. Distances between obstacles or between obstacles and the robot are not taken into consideration. The robot moves along the permitted sectors and avoids all banned sectors. If the goal direction is occupied by obstacles, the algorithm chooses two possible directions of motion, one on the left and another one on the right of the goal direction. Figure 1 illustrates the application of the traversability principle.

The tenacity principle consists, basically, in avoiding abrupt changes in the direction of motion when surpassing an obstacle. As a matter of fact and from a practical point of view, this turns into the special ability to follow the contour of an obstacle in a certain direction until it has been circumnavigated.

Figure 2 shows an illustration of how a virtual robot is able to overcome a trapping zone by applying both principles. Note that, in this case, the size of the region of interest is much smaller than the size of the obstacle. This restrains the portion of the obstacle perceptible by the robot, and forces to follow its contour until a free path to the goal is found.

The size of the navigation filter must be conveniently determined to perform local navigation, circumnavigating the obstacles at a cautious distance to avoid collisions but relatively close to the

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

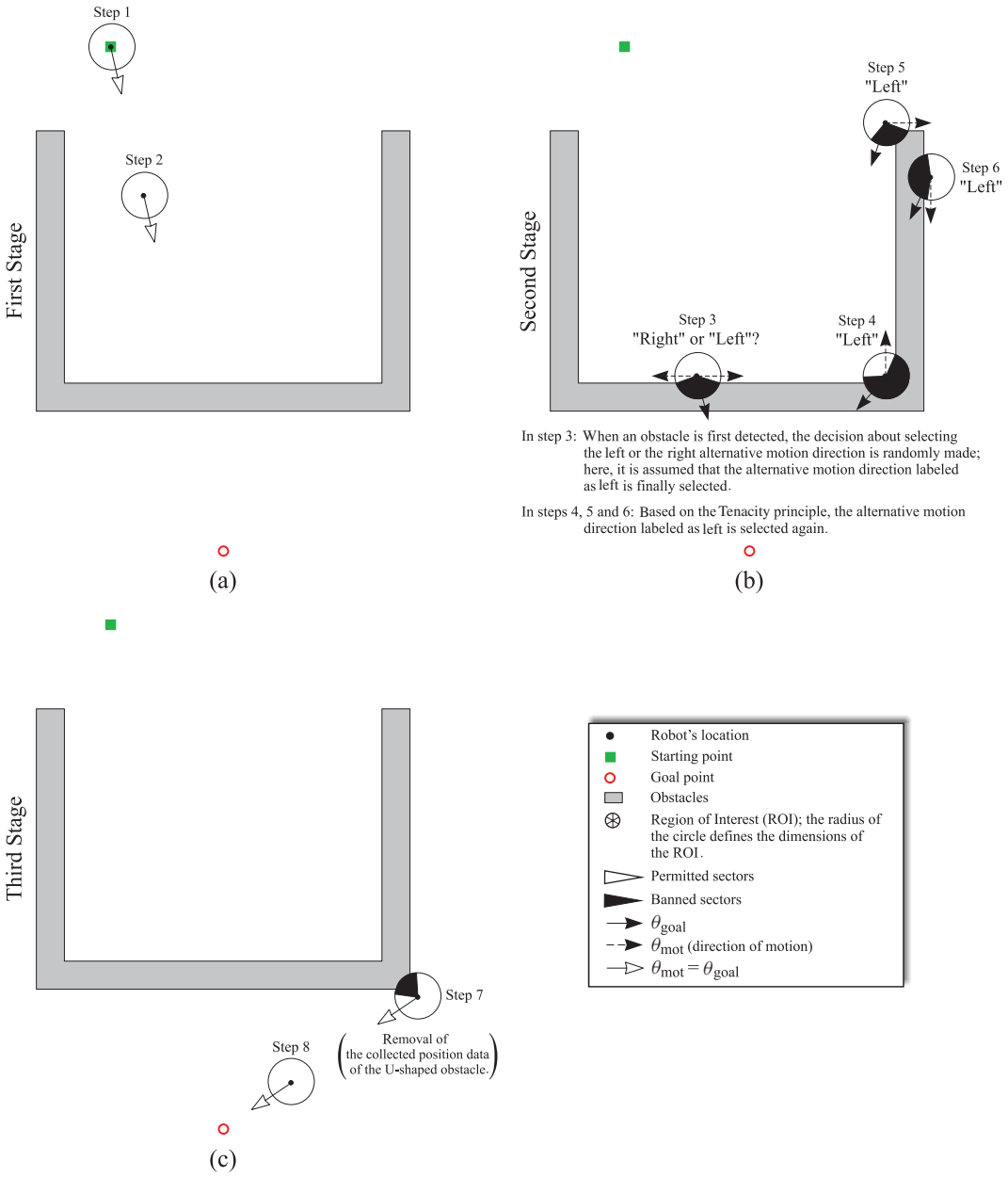

Fig. 2. (Colour online) Behavior pattern of a T^2-based reactive navigation strategy while escaping from a large U-shaped obstacle: (a) Direct path to the goal; (b) following the contour of the obstacle to the direction labeled as *left* by applying T^2; the system memorizes all the information of the detected obstacle(s) while it is moving out of the trapping area; (c) reset of the navigation filter since the goal direction is free.

obstacle contours to find all traversable gaps. In any case, for a local navigation approach, given the kind of data handled, it is clearly more efficient to anticipate a future collision situation.

T^2-based solutions are particularly interesting because: (a) in practice, with a relatively sparse set of low-cost sensor readings, it is possible to build an occupancy map sufficiently adequate to define the free and occupied zones around the robot, (b) they ensure the robot is able to escape from trapping zones, regardless their shape and size, and the size of the robot.

ND,[13] ND$^+$,[15] Smooth Nearness Diagram(SND),[8] or Closest Gap (CG)[14] tackle the problem of autonomous reactive navigation in troublesome or cluttered environments and all of them use a laser scanner as a unique sensor for real experiments. These strategies search for discontinuities between obstacles (gaps) wider than the robot width. The search of gaps is done by analyzing diagrams

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

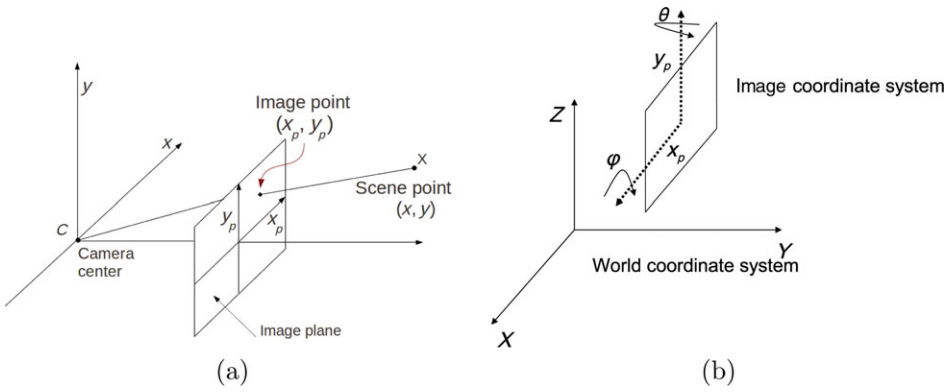

Fig. 3. (Colour online) (a) The perspective transformations; (b) coordinate frame conventions.

that represent the nearness of obstacles located between the robot and the maximum sensor range. Then navigable regions are built between two contiguous discontinuities and one of these navigable regions is chosen depending on the location of these areas with respect to the goal. This formulation avoids declaring environmental concavities as a navigable region. A security zone is defined as a region surrounding the robot, with a pre-set radius and where the risk of collision is considered very high. The strategy defines a set of situations with their corresponding actions, organized as a binary decision-tree, each being capable of uniquely characterizing the defined configuration of the group formed by obstacles, robot, and goal location.[13]

The aforementioned ND-based techniques continually consider all the environmental data captured along the full sensor range, being more likely to anticipate hazardous situations than the T^2-based strategies which perform local navigation. However, ND-based systems are not able to escape from local minima once the robot is inside one of them and its FOV is insufficient to perceive the whole obstructing area.

Our proposal consists in fusing the principal advantages of T^2 and ND in a single behavior-based control architecture which integrates the obstacle detection and the localization algorithms described in Bonin *et al.*:[3,4]

1. From T^2, we include the principles of traversability and tenacity and the concept of navigation filter.
2. From ND, we adapted the ability of finding discontinuities between obstacles, taking into consideration their length and relative position with respect to the robot, and the inclusion of the environmental data gathered between the robot and the goal.

1.3. Paper organization

This paper is structured as follows. Section 2 briefly outlines the IPT in the context of imaging geometry, the IPT-based feature classification method, and the algorithm to build local occupancy maps described in Bonin *et al.*[4] Section 3 describes the motion control algorithm; Section 4 discuses a wide range of experimental results; and finally, the conclusions are given in Section 5.

2. Obstacle Detection and Avoidance

2.1. Perspective transformations

The Direct Perspective Transformation (DPT) maps three-dimensional points onto a plane called the plane of projection. This transformation models the process of taking a picture. The line that connects a world point with the camera lens intersects the image plane defining the corresponding and unique image point of that world point (see Fig. 3(a)). The inverse process, that is, the projection of every image point back to the world is modeled by the IPT. The back projected point will be somewhere in the line that connects the image point with the center of projection (camera lens).

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

The direct and the inverse perspective projections are usually modeled assuming a pinhole camera.[7,9] Three coordinate systems are involved: the world, the camera, and the image coordinate systems. The linear mapping between world to image points, both expressed in homogeneous coordinates, can be written as[9]

$$\begin{bmatrix} x_p \\ y_p \\ f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} T_w^c \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{1}$$

where

$$T_w^c = \begin{bmatrix} R_{3x3} & T_{3x1} \\ 0_{1x3} & 1 \end{bmatrix}, \tag{2}$$

(x_p, y_p) are the image point coordinates, (x, y, z) are the corresponding scene point world coordinates, and T_w^c is the 4×4 transformation matrix that relates, via a rotational (3×3) matrix R and a translational (3×1) vector T, the world and the camera coordinate frames.

The scene point world coordinates corresponding to an image point can be calculated knowing either the distance between the camera and the scene point or any of the (x, y, z) world coordinates, as it is the case of, for example, points lying on the floor $(z = 0)$.[7]

The different coordinate systems and notations are illustrated in Fig. 3(b).

2.2. *Feature classification*

Let us consider a single camera mounted on a moving platform running on a flat ground, with an always known pose, combining translational with rotational motion, and capturing frames at consecutive time steps. The relative motion between the camera and the scene causes the displacement of salient features in the image, changing their image coordinates (x_p, y_p), but still representing the same scene point. Given an image feature, the world coordinates of its corresponding scene points can be computed in two consecutive images assuming that the world point lies on the ground. If the assumption is correct, the two resulting projections will coincide, otherwise the two projections turn out to be different to one another and the real world coordinates. Hence, one can distinguish if an image point belongs to an obstacle or to the floor projecting it onto a previously assumed flat ground $(z = 0)$ in two consecutive images and analyzing the distance between the resulting projections:

$$D \text{ (Discrepancy)} = \sqrt{(x_2^* - x_1^*)^2 + (y_2^* - y_1^*)^2} \quad \Rightarrow \begin{cases} \text{if } D > \beta \Rightarrow \text{obstacle,} \\ \text{if } D \leq \beta \Rightarrow \text{ground.} \end{cases} \tag{3}$$

where (x_1^*, y_1^*) and (x_2^*, y_2^*) correspond to the projection onto the ground of an image point (x_p, y_p) at instants t_1 and t_2, respectively, and β is the threshold for the maximum difference admissible between (x_1^*, y_1^*) and (x_2^*, y_2^*) to classify the feature as ground point. Ideally β should be 0.

The idea is illustrated in Fig. 4(a). Two frames of a scene are taken at instants t_1 and t_2. Point P_{2w} is on the floor. Its projection into the image plane at instants t_1 and t_2 generates the image points P_{2i0} and P_{2i1}, respectively. The inverse transformation of P_{2i0} and P_{2i1} generates a unique point P_{2w}. P_{1w} is an obstacle point. Its projection onto the image plane at t_1 and t_2 generates, respectively, points P_{1i0} and P_{1i1}. However, the inverse transformation of P_{1i0} and P_{1i1} back to the world assuming $z = 0$ (e.g., projection onto the ground plane) generates two different points on the ground, namely, P'_{1w} and P''_{1w}.

2.3. *Feature detection and tracking*

The first key step of the algorithm is to detect a sufficiently large and relevant set of image features and match them across consecutive images.

Establishing good correspondences between image points, for example between P_{2i0} and P_{2i1} in Fig. 4, is of crucial importance as incorrect correspondences lead to wrong classifications. Because of this, wrong correspondences between image points in consecutive frames are filtered out using *Random Sample Consensus* (RANSAC) and imposing the epipolar constraint: $x_p^T F x_p = 0$, where x_p^T and x_p are the point image coordinates in two consecutive frames, and F is the fundamental matrix.[9]

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

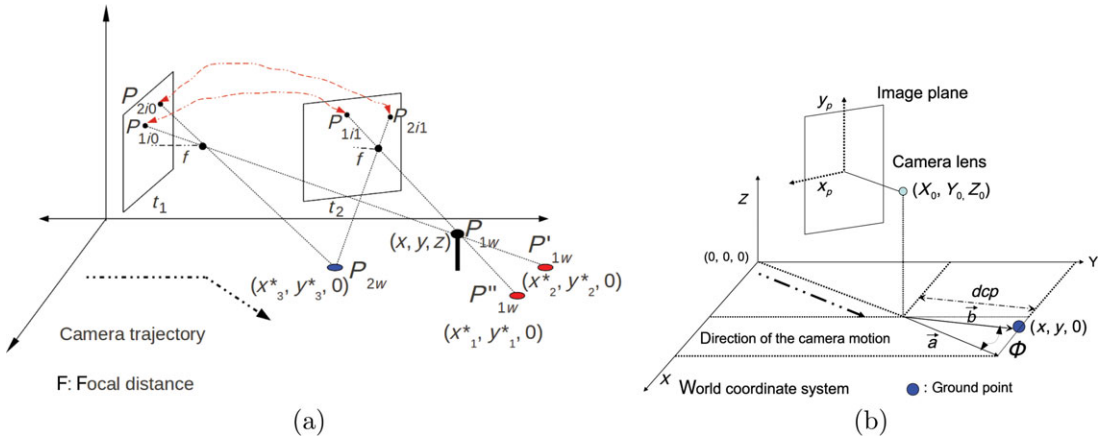

Fig. 4. (Colour online) (a) The IPT-based obstacle detection approach; (b) Distance and orientation of a ground point with respect to the camera.

2.4. Obstacle profiles and the occupancy map

Image features or corners are usually detected at regions of high gradient, thus they are likely to be near or belong to an edge. Besides, features classified as obstacle are most likely to be contained or near a vertical edge belonging to an obstacle. Hence, the next step of the algorithm is the computation of edge maps and the association of such edges with features classified as obstacle. This permits isolating the obstacle boundaries from the rest of edges and getting a qualitative perception of the environment.

Edges selected as obstacle boundaries are tracked down until the last edge pixel or a ground point is found. This will be considered as the point where the object rests on the floor.

Thus, the distance and the orientation of an obstacle with respect to the camera can be qualitatively estimated using the distance and the orientation of the ground points where the obstacle touches the floor. A semicircular area of a fixed radius, centered at the robot position and virtually located on the ground plane, is considered to be the ROI used for obstacle avoidance. Only obstacles detected inside this ROI are considered to be avoided. The idea is illustrated in Fig. 4(b), where ϕ and dcp are, respectively, the relative orientation and distance of the ground point with respect to the camera coordinates.

3. Motion Control for Reactive Navigation

3.1. Overview

The objective is to control the motion of the robot while navigating from a starting point to one or several subsequent preprogrammed goal points, detecting and avoiding all obstacles. Obstacle evidences captured by the sensors are placed in the navigation filter and are represented as points, in accordance to the range nature of the data handled.

Each execution of the algorithm involves a pair of images and outputs a steering vector. The systems use a navigation filter similar to the one described in Section 1.2, but with the following particularities:

1. In contrast to the region of a fixed radius proposed by Antich and Ortiz,[1] now the size of the navigation filter is dynamic, centered at the robot pose, and covering from the current robot coordinates up to the goal position. This ROI is divided into polar sectors of γ°. The aim is to consider at each time only those obstacles of the environment that can obstruct the way to the target.
2. Sections of the filter containing at least one obstacle point will be labeled as occupied, but, contrary to Antich and Ortiz,[1] where occupied zones are always banned, now occupied zones will not be automatically declared as forbidden directions unless they do not contain any gap traversable by the robot.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

3. Analogous to Antich and Ortiz,[1] sensor data world coordinates ranging between the robot and the goal are registered in the navigation filter to be used later in a qualitative way. This information is used to determine the traversability of the different areas of the environment and stored in the map until the current obstacle is overcome. Then the filter is reset and the process is started again.
4. When a sector of the navigation filter stores several obstacle data, the one closest to the robot will be the most restrictive one in terms of estimating the risk of collision.
5. To guarantee escaping from trapping situations, and of course, to overcome other obstacles, the principle of tenacity is applied until the goal direction is again free or the obstacle has been completely surpassed, which means, in practical terms, the obstacles currently avoided are behind the robot.

Furthermore, the strategy defines the following:

1. A safety distance S_{dist} outlining a safety area around the robot. If an obstacle is found inside this area, the situation is considered to be as high collision risk.
2. A set of different situations. Each situation involves a certain configuration of the robot pose, obstacles, and goal, and has associated a set of predefined behaviors: (a) Go to Goal, when the goal direction is free, (b) High Risk Obstacle Avoidance, when there are obstacles inside the safety area, and (c) Low Risk Obstacle Avoidance, when the goal direction is occupied and all obstacles are out of the safety area.

The strategy must guarantee escaping from trapping zones and, to a certain extent, the ability of anticipating hazardous situations.

3.2. Situations and actions

The objective of this section is to describe a set of situations and their respective associated actions. One assumption must be pointed out: Obstacles generate clusters or aggregations of points, and a cluster of points, distant from each other by less than the robot dimensions, represents an obstacle.

The algorithm analyzes which of the next possible situations can come out and associates each situation with a behavior:

(a) The sector of the navigation filter containing the goal direction is free of obstacles and there are no obstacles in front of the robot and inside the safety area. This situation activates the Go to Goal behavior. For relatively narrow sectors, let us say, for instance, between 10° and 15°, this situation is approximately equivalent to evaluate if the goal direction is free of obstacles. For wider sectors (for example 40° or 65°), the fact that the one containing the goal direction is occupied with obstacles would not significantly mean that the goal direction is not free.

Some verifications must be done before the navigation module decides whether the robot can take the goal direction or not. Starting at the sector corresponding to the goal direction, the algorithm searches to the left and to the right for the first sector at each side occupied by an obstacle.
Let us denote as follows:

- P_L: The closest obstacle point at the left of the goal direction.
- P_R: The closest obstacle point at the right of the goal direction.
- θ_{goal}: The goal direction with respect to the robot pose.
- θ_{mot}: The robot direction of motion.
- θ_{PL} and θ_{PR}: The polar directions of P_L and P_R, respectively, with respect to the robot pose.
- d_L: Euclidean distance between the robot and P_L.
- d_R: Distance between the robot and P_R.
- d_{RL}: Euclidean distance between P_R and P_L, defining the length of the gap.

If $d_{RL} < 2S_{\text{dist}}$, then the gap will be labeled as *forbidden*, otherwise the gap will be permitted and the robot will be directed, in principle, to pass through it: $\theta_{\text{mot}} = \theta_{\text{goal}}$.
However, following the goal direction can lead the robot to collide with one of the obstacles that delimit the gap.
Let us denote ϵ as:

$$\epsilon = \arcsin(S_{\text{dist}}/d_i), \tag{4}$$

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

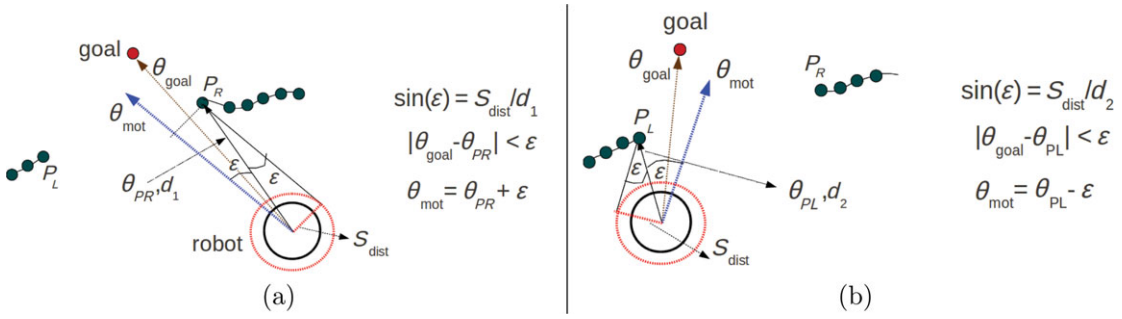

Fig. 5. (Colour online) (a) $\theta_{\text{mot}} = \theta_{PR} + \epsilon$, (b) $\theta_{\text{mot}} = \theta_{PL} - \epsilon$.

d_i being the shortest of d_R or d_L. ϵ defines the minimum angular deviation for the robot to overcome the obstacle passing with the border of its safety area tangential to the closest obstacle point.
If θ_{goal} fulfills the requirement,

$$|\theta_{\text{goal}} - \theta_{PX}| > \epsilon \tag{5}$$

being θ_{PX} either θ_{PL} or θ_{PR} (between P_R or P_L, take the closest to the robot), then the direction of motion will be $\theta_{\text{mot}} = \theta_{\text{goal}}$. If requirement (5) is not accomplished, then $\theta_{\text{mot}} = \theta_{PX} \pm \epsilon$. The sum or subtraction will be applied depending on the relative orientation of the robot with respect to the goal. See in Fig. 5 two cases concerning this situation. The border of the safety distance is in red, the goal direction is in brown, the final motion is in blue. The goal direction is free, the gap is wide enough for being traversed by the robot, but going direct to the goal would lead to a sure collision. In both cases the closest obstacle point touches the border of the safety area.
If P_R and P_L are one behind the robot and the other is in front of it, the algorithm focuses on the point which is in front of the robot, since this will be the one limiting the robot motion toward the target.
(b) There are obstacles in front of the robot and inside the safety area. This situation activates the high risk obstacle avoidance behavior. In this case, the direction of the motion will be the result of applying the Potential Fields method only with the obstacles located inside the safety area.
(c) The sector of the navigation filter containing the goal direction is occupied by obstacles and they are outside the safety area:

$$d_L > S_{\text{dist}} \text{ and } d_R > S_{\text{dist.}} \tag{6}$$

This situation activates the low risk obstacle avoidance behavior.
In this case, the system will propose, similar to Antich and Ortiz,[1] two possible solutions for the next steering vector: one on the right of the goal direction, and another on its left. The algorithm will search from the goal direction, sector by sector, clockwise and counterclockwise if one of the next cases is fulfilled at each side:

- Case (1): n ($n \in \mathbb{N}^+$) contiguous sectors of the navigation filter are occupied by obstacle points. Find two points, P_L and P_R, both in front of the robot with no other points in-between, satisfying the condition $d_{RL} \geqslant 2S_{\text{dist}}$.
  If two points fulfilling this requirement are found, it will be assumed that they form a traversable gap. If several permitted gaps are found, take the closest to the goal direction.
  Then $\theta_{\text{mot}} = \theta_{PX} \pm \epsilon$, PX being either P_L or P_R (the closest to the robot), and ϵ as defined in Eq. (5) (see Fig. 6).
- Case (2): n contiguous sectors free of obstacles bordered by one occupied sector on each side, forming a gap.
  Let us denote again points P_L and P_R as the extremes of the obstacles that define the gap. Again, if $d_{RL} \geqslant 2S_{\text{dist}}$, then the gap is labeled as permitted, otherwise it is labeled as forbidden and the corresponding sector(s) is (are) banned.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

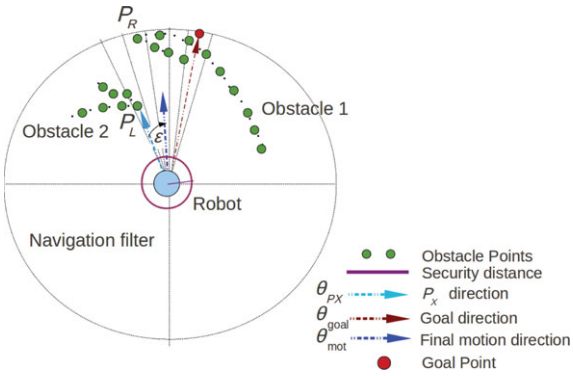

Fig. 6. (Colour online) The goal direction is occupied. There are two contiguous sectors occupied by two obstacle points with a gap in-between. $\theta_{\text{mot}} = \theta_{PL} - \epsilon$.

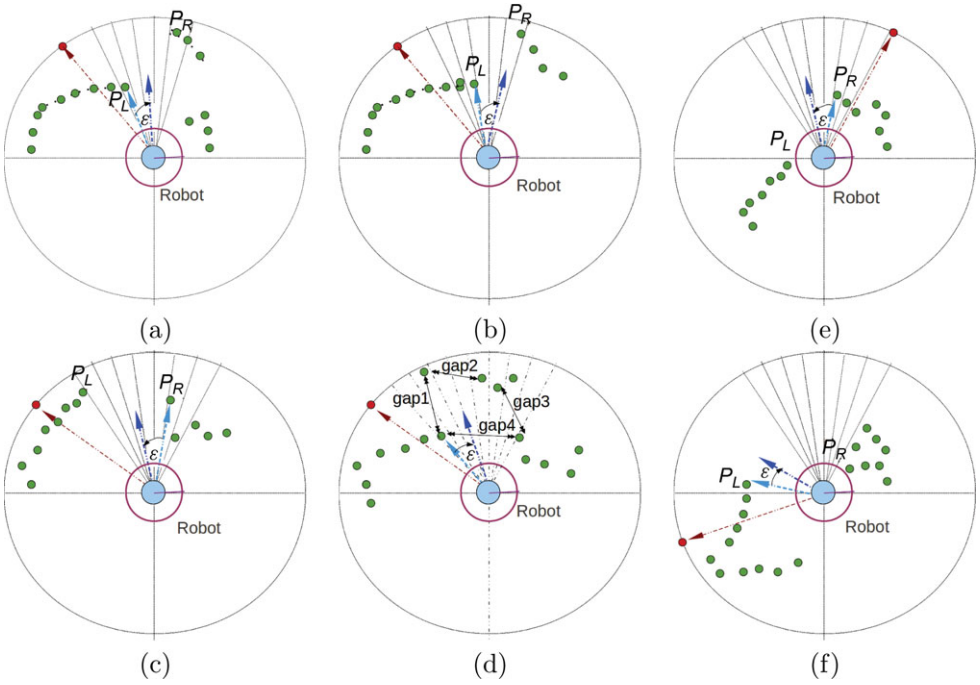

Fig. 7. (Colour online) Goal direction: occupied. (a), (b), and (c): there is only a traversable gap in front of the robot. (d) Gaps 1 and 4: permitted; gaps 2 and 3: banned. In (e) and (f) the goal direction is occupied and one of the extremes of the gap is behind the robot. In (e) $\theta_{\text{mot}} = \theta_{PR} + \epsilon$; in (f) $\theta_{\text{mot}} = \theta_{PL} - \epsilon$.

Analogously, $\theta_{\text{mot}} = \theta_{PX} \pm \epsilon$, PX being either P_R or P_L (the closest to the robot), always moving the steering vector from the closest obstacle point to the furthest obstacle point that forms the gap. The idea is illustrated in Figs. 7(a)–(d). The blue vector represents θ_{mot}, the red circle represents the goal point, the green circles represent the obstacle points, and the light blue vector represents the *PX* direction.

- Case (3): One of the points, either P_L or P_R, of an existing gap is in front of the robot and the other one is behind it. This case represents the situation in which one of the obstacles delimiting the gap is behind the robot and the other is still in its front.
  Then $\theta_{\text{mot}} = \theta_{PX} \pm \epsilon$, θ_{PX} being the point which is in front of the robot. The summation or subtraction will be chosen in order to displace θ_{mot} away from the obstacle point, which is in front of the robot (see Figs. 7(e) and (f)).

In cases (1) and (2), if the resulting vector steers the robot toward a sector occupied by an obstacle point which shall fall inside the safety area, the gap is discarded.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

At this point the algorithm may have two possible steering vectors, one to the right and another to the left of the goal direction. The *decision-maker* module will choose according to the following criteria:

1. The vector closer to the goal direction is chosen in two cases: (a) In the first motion of the robot, and (b) the navigation filter has been reset after the current obstacle is left behind.
2. Contrarily, if the current obstacle has not yet circumnavigated, the tenacity principle is applied and the direction of turn (clockwise or counterclockwise) is the same as the previous decision.

The full algorithm is illustrated in Fig. 8.

Nearly all the sectors between the robot position and the goal point are occupied by obstacle points (sensor readings). In plot (a) two possible directions of motion are proposed: V1 (in blue) and V2 (in red). V1 is chosen because it is closer to the goal direction. In (b), two possibilities are also generated, V1 and V2. Applying the tenacity concept, the robot chooses again V1. In (c) the target direction is still occupied. The immediate obstacle has been already overcome. V1 is chosen again still applying tenacity. The gaps on the right of the robot are not permitted because they are narrower than the safety distance. The first available gap to the right is indicated as V2 in red. In (d) the robot still chooses V1 applying the concept of tenacity, since the upcoming obstacle has still not been overcome. In (e) the goal point is perfectly visible, its direction is completely free, and the current obstacle has been overcome. Immediately, both navigation filter and tenacity boolean condition are reset. The robot directly faces the goal. Note how the navigation filter moves with the robot and reduces its dimensions as it approaches the target, checking each time only the part of the environment between the robot and the goal point.

While the tenacity principle is being applied, the navigation filter holds all existing points and places the new ones. Once tenacity is no longer applied, the filter is reset and filled again.

When a robot approximates a trapping zone much bigger than its size and the sensor range, having a global vision the complete problematic area is not possible. In this case the robot is not able to anticipate a motion to avoid an obstacle because nothing is perceived until some (small) part of the obstacle falls inside the sensor FOV. However, at that moment, the robot is already deeply inside the trapping area. This is one of the cases where the tenacity principle can be typically applied to solve the situation.

For the sake of illustration, Fig. 9 shows a simulation of escaping from a U-shaped obstacle with an aperture on one of its sides.

Three algorithms have been tested to compare their performances: ND, whose result is shown in plot (b); T^2, shown in plot (c); and the strategy outlined in this paper, shown in plot (d). All three simulations have been run through the 2D stage simulator over the player platform.[11] In all these plots, the blue circle represents the starting point and the red circle represents the goal point. The small red box represents the robot and the wide red line depicts the robot trajectory. The robot in Fig. 9(b) is equipped with a simulated laser scanner because all experiments presented in refs. [13, 15] use a robot equipped with a laser, while the robot of Fig. 9(c) is equipped with a simulated sonar, since T^2 does not require sensor information as rich as a laser scan.[1] In the simulation of the combined ND/T^2 algorithm (plot (d)) no cameras have been simulated, but sonar readings are used as for T^2 (the obstacle detection step is assumed finished and its results are the sonar readings).

Table I shows the relevant settings for the three experiments: the sensor for obstacle detection, the ROI (portion of the environment centered at the robot that is used for (local) navigation), the motion control loop execution frequency and the number of sensor beams. For the ND simulation, the frequency of the control loop is maximum that permits the CPU. In all four plots, the transparent navy blue area shows the ROI, which corresponds to the range of the simulated laser in plots (a) and (b), the range of the region containing the obstacle data supplied to the navigation filter in (c) (it does not necessarily coincide with the sonar range), and the FOV_H in (d) (again not necessarily coincident with the sonar range). Note that in all the cases the blue area is much smaller than the obstacle size. In (b), the robot running ND directs to the goal but it gets trapped in the frontal wall. Although the robot using the T^2 strategy (c) is not able to detect the gap on the U side, it is able to reach the goal, but performing a longer trajectory since the robot of plot (d) is able to reach the goal passing through the lateral gap.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

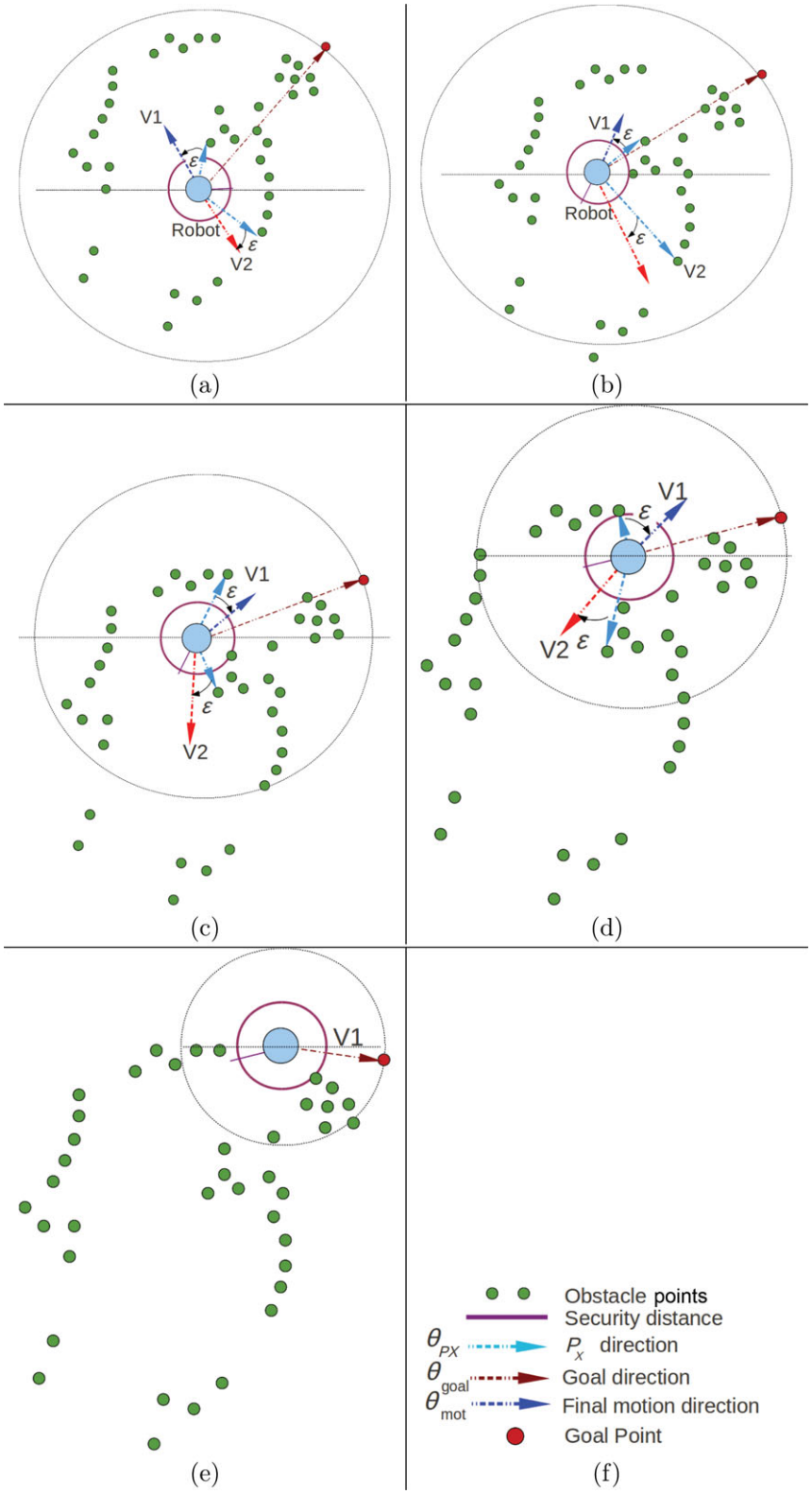

Fig. 8. (Colour online) Sequence of a complete route: (a)–(d) Traversing the permitted gaps, applying the criteria of minimum turn and tenacity; (e) the robot directly faces the goal.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

Table I. Basic settings for the simulation.

	ND	T^2	Combined ND/T^2
Sensor	Laser	Sonar	Sonar
ROI	3 m/180°	3 m/180°	3 m/180°
Frequency	Maximum	4 Hz	4 Hz
Nbr of beams/samples	361 samples	10 beams	10 beams

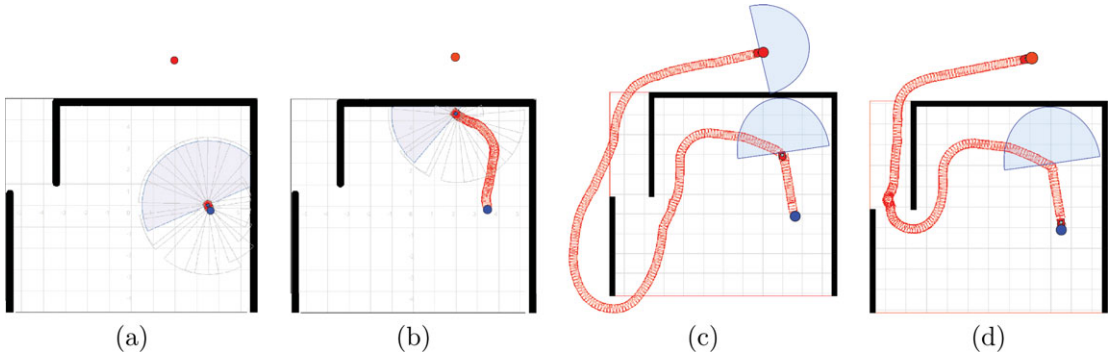

Fig. 9. (Colour online) Solving a U-shaped obstacle: (a) laser range for ND simulation, (b) ND simulation, (c) T^2 simulation, (d) simulation for our strategy.

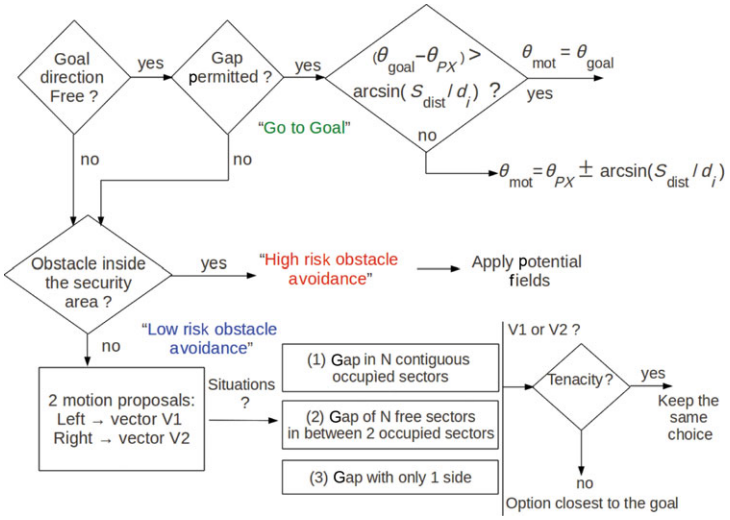

Fig. 10. (Colour online) Overview of the complete motion control strategy.

Figure 10 summarizes the main points of the algorithm and the corresponding actions performed in each case.

4. Experimental Results

4.1. Settings

A robot Pioneer 3DX equipped with a calibrated camera was programmed to move from a starting point to a defined goal, or to a set of subsequent defined goal points, in different scenarios with different structures, different illumination conditions, with regular and irregular shaped obstacles and textured and untextured floors.

One of the important differences between the sensorial equipment used in the experiments presented in refs. [8, 13, 15] and a camera is the way the environment is scanned. Laser range-finders are usually

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

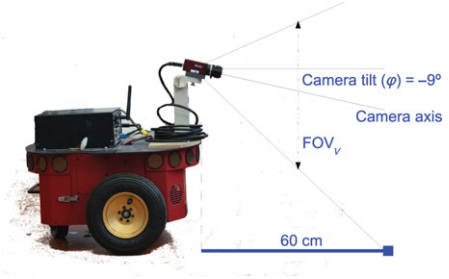

Fig. 11. (Colour online) Robot platform with the camera. Blind zone and FOV_V.

mounted on rotational platforms that scan the environment in considerably wide horizontal ranges. For example, the Hokuyo URG-04LX horizontal measuring area covers 240°. For covering such horizontal ranges with a standard lens, it is necessary to capture several frames rotating the camera around its vertical z-axis. In consequence, unless a fisheye lens or an omnidirectional system is used, a camera will need several shots to sweep horizontal area that can be covered by a single laser scan. Besides, being FOV_H the lens horizontal FOV expressed in degrees, the robot will have a blind zone of $(180^\circ - \text{FOV}_H)/2$ at both left and right sides, increasing the need of storing recently captured obstacle information and imposing additional navigation rules to carefully explore both sides of the robot.

Another important issue is that laser-scanned environments can generate dense maps, favoring the detection of obstacles and discontinuities with a considerable accuracy. However, the aim of our system was not to represent a map of the environment as accurate as possible, but to deal with a qualitative representation. Our obstacle detector tried to emulate a visual sonar, placing in the navigation filter the set of obstacle-to-ground contact points described in Section 2 and observed from the robot pose to the goal point.

The environmental parameters were (a) camera height = 430 mm, (b) camera pitch and yaw angles = -9° and 0°, respectively, (c) focal length = 3.720 mm, (d) image original resolution: 1024 $\times$768 square pixels, being afterwards down-sampled to a resolution of 256 $\times$ 192 pixels to reduce processing time.

Figure 11 shows the robot platform. With the current settings of focal length, original resolution, and pixel size, the FOV_H turns out to be 65.5° and the FOV_V is 51.5°, leaving 60 cm of blindness between the robot front and the bottom of the camera's vertical FOV.

The camera height was imposed by the platform and the gimbal that append the camera to the robot, but the tilt angle was chosen to obtain a reasonably short blind area in front of the robot combined with a considerable vertical visual range. All these variables directly conditioned the criteria for setting S_{dist}, and the value of S_{dist} in turn determines which gaps are traversable and which are not. S_{dist} was set to 95 cm, which includes the blind area plus the approximate robot radius (27.5 cm).

The width of one navigation filter sector was set to 12°, which means a total of 30 sectors. This value experimentally demonstrated to be adequate to manage the free and occupied areas in front of the platform.

The limited camera FOV reinforced the idea of storing in memory the environmental information during a limited time period.

4.2. Experiments

In all the experiments included below, the robot detected the obstacles and set the occupancy of the navigation filter according to the procedures described in Section 2. The motion orders were generated applying the strategy reported in Section 3.

The feature-tracking process previous to the feature classification was performed by means of the pyramidal implementation of the KLT tracker[17] because (a) the algorithm is faster than other detectors, (b) it generates a sufficiently large number of features with high repeatability, (c) feature descriptors were not significantly affected by changes in scale since differences in consecutive frames were negligible at the used frame rates, (d) more than 90% of the tracked features were inliers and well classified, and (e) they were usually found in corners, facilitating the detection of obstacle boundaries.[5]

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

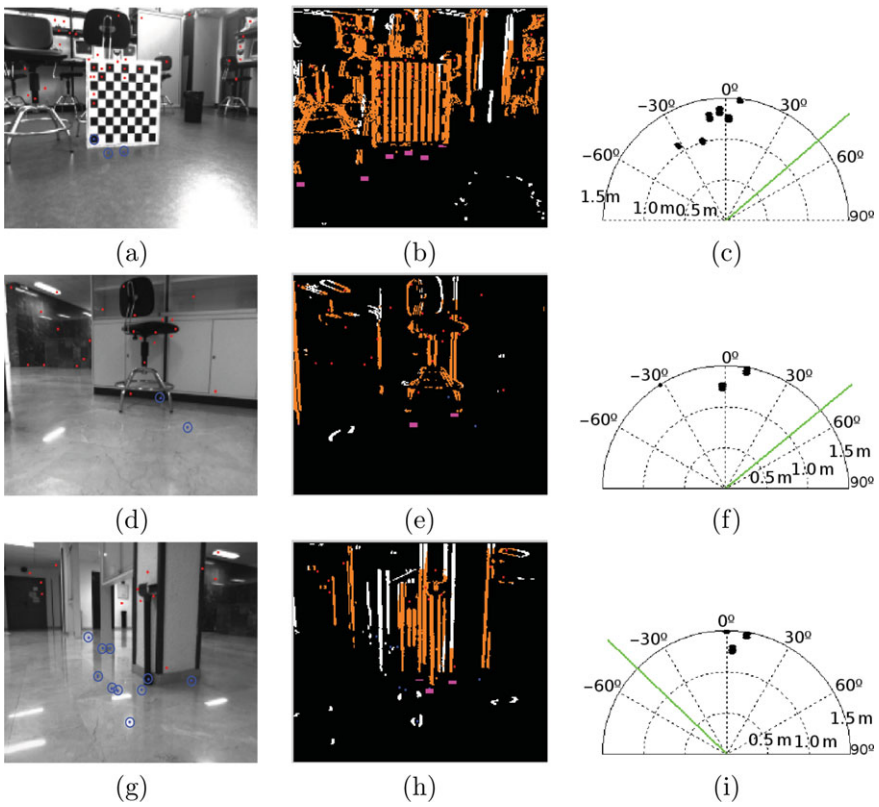

Fig. 12. (Colour online) Examples of images captured during an experiment and the occupancy maps computed from them.

Figure 12 shows some examples of qualitative occupancy maps built from the images captured during the robot motion in different environments. Images in Figs. 12(a), (d), and (g) show the captured images with obstacle points in red and the ground points circled in blue. Images in Figs. 12(b), (e), and (h) show the obstacle boundaries highlighted in orange and the obstacle-to-ground contact points in pink. Plots (c), (f), and (i) show the resulting occupancy maps with a radius limited to 1.5 m to facilitate its presentation.

In all next plots (a) the starting point is colored blue, (b) the goal points are colored red, (c) the robot trajectory points were obtained from the wheel odometry data and are pictured as tiny empty red circles with the corresponding direction vector in blue, and (d) all length units are expressed in millimeters.

First navigation experiments were planned inside a laboratory, where some scenarios were particularly prepared to test certain fundamental behaviors, such as, for example, escaping from potential trapping areas or navigating through environments where the distances between obstacles where approximately equal to S_{dist}.

Figure 13 shows some of these experiments. Plot (a) shows a route where the goal point is just behind a wall and there is only one free path to reach it. Plot (b) shows an example of going and returning to the same departure point. Plots (c) and (d) show two examples of anticipation and avoidance of local minima. If the robot had fallen into a deep trapping canyon due to, for example, the impossibility of detecting it in a reasonable number of frames, the application of the tenacity concept would have helped to escape from it.

Images shown in Figs. 14(a) and (c) correspond to the plot of Fig. 13(c). Images shown in Fig. 14(e) and (g) correspond to the plot of Fig. 13(d). The corresponding obstacle boundaries and obstacle-to-ground contact points are depicted in orange and pink, respectively, in Figs. 14(b), (d), (f), and (h).

The following experiments intended to simulate an ordinary situation in which the robot had to execute a mission over two defined points in a relatively crowded daily used indoor environment. All

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

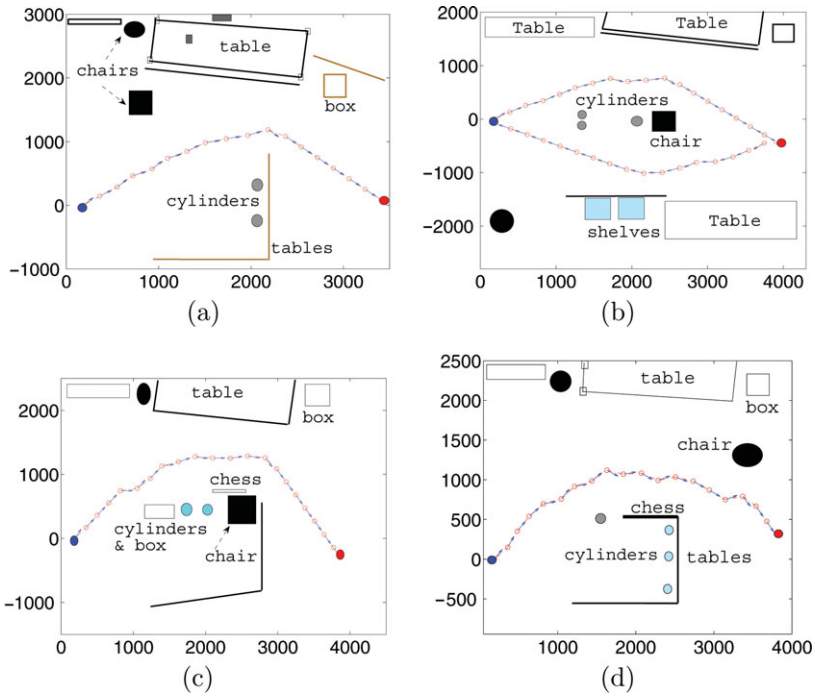

Fig. 13. (Colour online) Tests in a laboratory: (a) From starting to the goal point in a relatively dense environment; (b) go and return; (c) and (d) avoiding trapping zones.

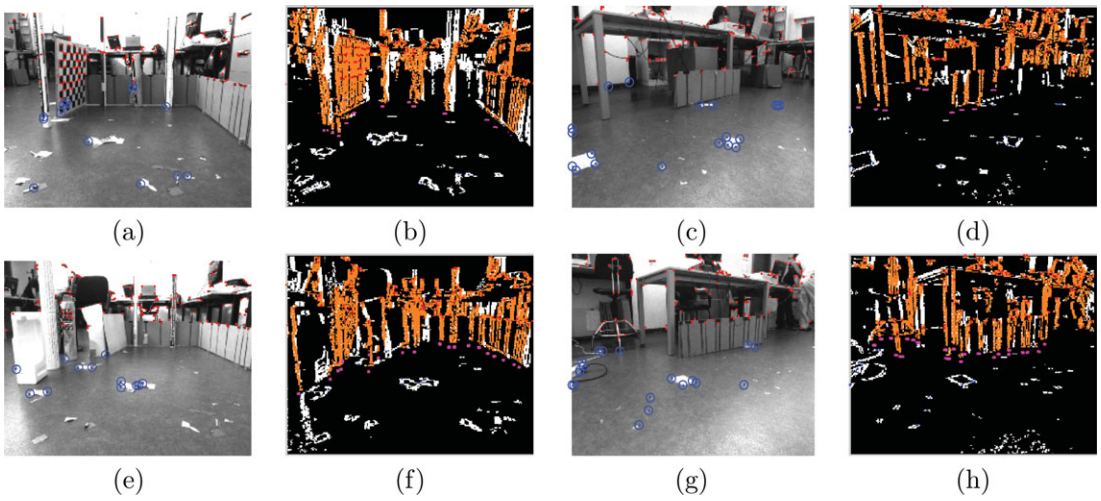

Fig. 14. (Colour online) Avoiding a trapping zone. (a), (c), (e), and (g): frames with obstacle and ground points. (b), (d), (f), and (h): edge maps with obstacle boundaries in orange and obstacle-to-ground contact points in pink.

tests tried to demonstrate the suitability of our navigation strategy in common scenarios such as, for example, offices, corridors, halls, sidewalks, or warehouses.

Figures 15 and 16 plot four different trajectories conducted by the robot in a considerable busy hall, located inside a university building. Obstacles are clearly labeled in all pictures.

Note how the robot is able to pass through all available apertures in order to advance along the free space toward the goal. Figure 15(b) shows three goal points, enumerated according to the order in which they had to be reached. Figure 17 shows some images recorded during the experiments of Figs. 15 and 16, together with the obstacle and ground points and with the corresponding edge maps highlighting obstacle contours.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

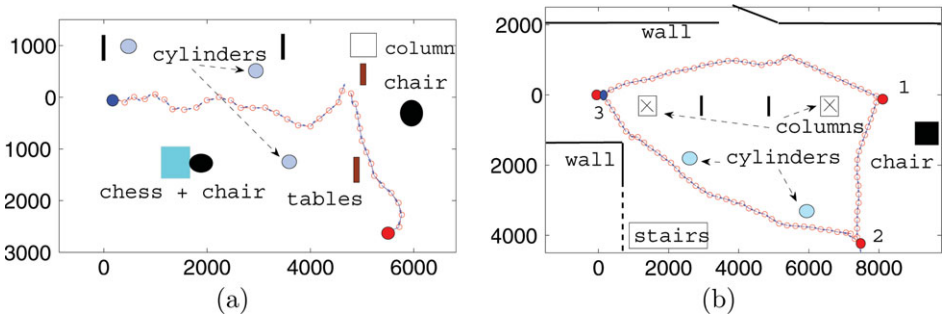

Fig. 15. (Colour online) Experiments in a hall: (a) One goal point, and (b) three consecutive goal points forming a closed loop.

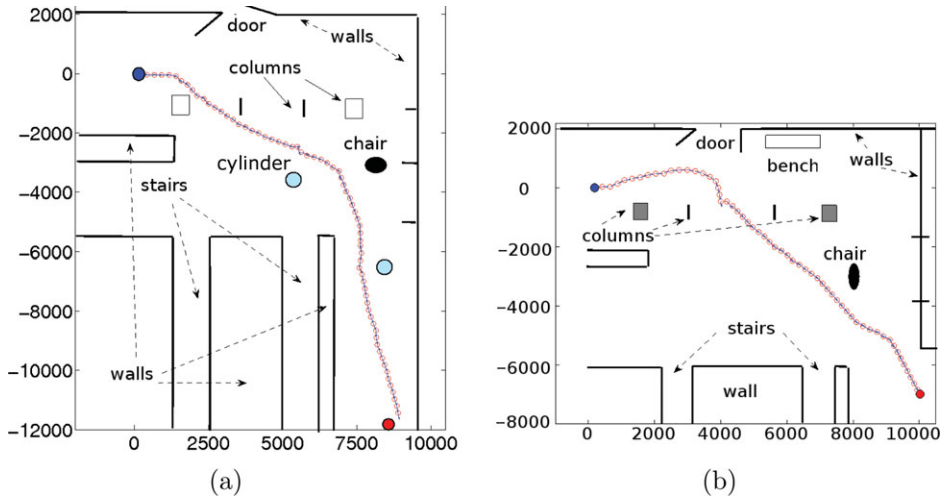

Fig. 16. (Colour online) Experiments in a hall with longer trajectories.

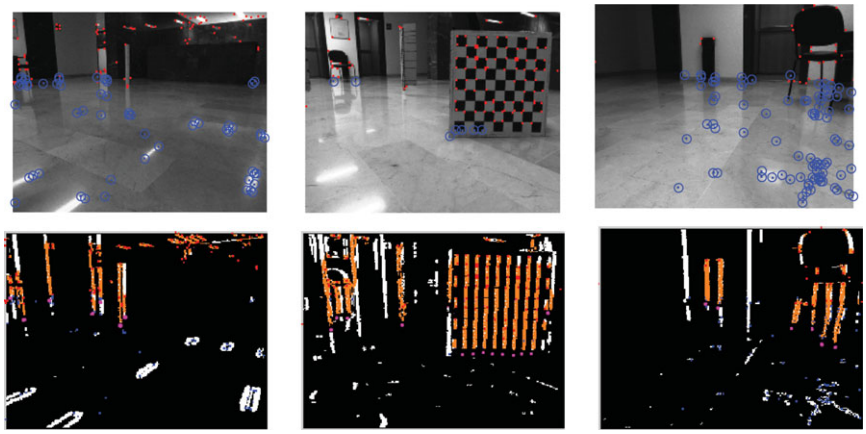

Fig. 17. (Colour online) Experiments in a hall. Images recorded online during the route.

These tests evidence the capacity of a complete system to provide a robust obstacle detection module integrated in a highly effective reactive navigation architecture.

Navigating through long and relatively narrow spaces with a considerably high number of randomly placed obstacles could be a challenge for a robot of the size of Pioneer 3DX. In Fig. 18, the starting and goal points were separated 25 m in the x-axis of a long corridor located in the university building. This corridor was filled with obstacles distributed along the whole path. Again, the robot was able to cover the whole distance avoiding all the incoming obstacles and walls. Walls of the corridor are

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

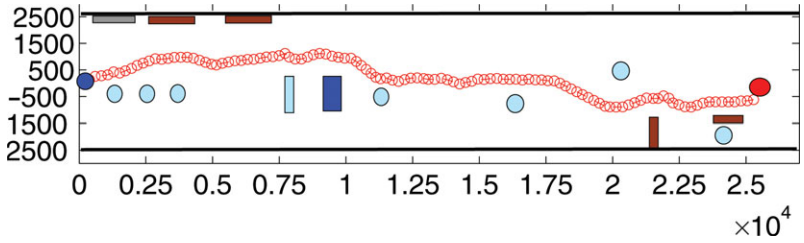

Fig. 18. (Colour online) The longest trajectory along a corridor with several obstacles placed along the path.

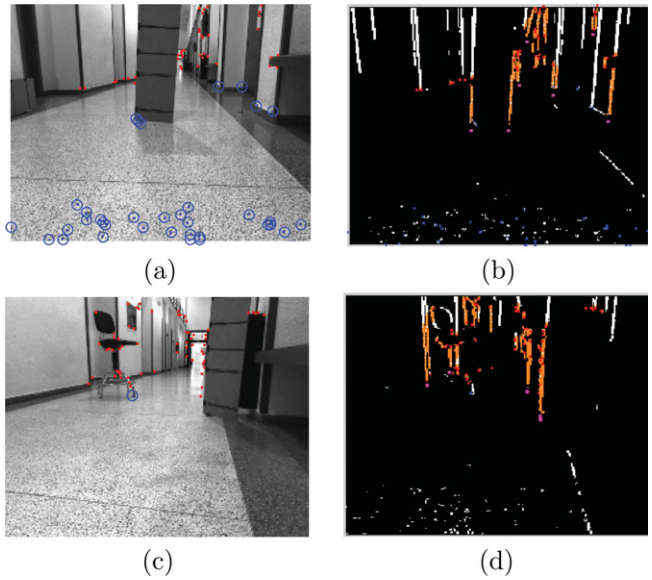

Fig. 19. (Colour online) Navigation along a corridor: (a) and (c) frames show obstacle and ground points. (b) and (d) show edge maps with obstacle boundaries in orange and the obstacle-to-ground contact points in pink.

clearly indicated as well as obstacles colored in blue, soft blue, and brown. Figure 19 shows some pictures captured during this experiment.

In some of these plots, there is a slight difference, caused by odometry errors between the red circle, which represents the programmed goal point, and the point where the robot really ends its trajectory.

4.3. Discussion

Although experimental results have shown a good performance of the proposed navigation strategy using a camera, the control algorithm presented in Section 3 is not restricted to the use of a visual sensor, but it could also work with any range sensor that returns the perceived environment in the form of discrete readings or points. As mentioned in the Introduction of this paper, cameras are usually cheaper and outperform standard laser or sonars in temporal and/or spacial resolution, so they finally become a preferable alternative. However, it is worth to analyze briefly the advantages and inconveniences of applying our motion control strategy with different sensors, and whether a combination of these could be useful to improve the overall system.

Table II compares three main features that characterize the new control strategy and are directly affected by the type of the used sensor. On the one hand, considering different sensors in approximately the same price range, robots equipped with cameras will perceive larger areas than sonars or lasers (consider from now on only laser in 2D since 3D lasers exceed considerably the price of a standard camera), increasing their ability to anticipate obstacles or hazardous situations. Perceiving at longer distances also permits the planning and generation of shorter and smoother trajectories. Cameras have, in addition, the capacity to infer data in 3D, contrary to lasers or sonars.

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

Table II. Featuring the new strategy with different sensors.

Feature	Camera	Laser	Sonar
Passing through extremely narrow gaps	Difficult	Yes	Difficult
Perceiving in 3D	Yes	No	No
Anticipate obstacles. Better paths in terms of length and smoothness	Yes	Difficult	Difficult

On the other hand, sonars and cameras using the visual algorithm described in Section 2 return a sparse set of environmental points, which complicates the detection of exact dimensions of gaps, and thus the navigation through apertures with a length close to robot dimensions. Contrarily, lasers give highly dense set points permitting to maneuver even in extremely cluttered environments.

In consequence, the results of the proposed navigation strategy could be improved by combining a camera with a low-cost laser. The laser could be used to support the perception of the environment in the blind zones of the camera and to measure with a higher degree of reliability the dimensions of gaps close to the robot. In this way, the strategy would permit to anticipate obstacles, escape from complex trapping situations, plan shorter and smoother trajectories, and, furthermore, pass through gaps of length similar to the robot size.

5. Conclusions

The image feature classifier presented in Bonin *et al.*[4] permits to discriminate obstacle from ground points. Obstacle points are used to build a semi-circular qualitative occupancy map with a pre-defined diameter which contains the points where obstacles touch the ground. The main advantages of this method are: (a) the possibility of using a feature-tracking-based obstacle detection system constrained as less as possible to environmental conditions (illumination, reflections, colors, textures, etc), and (b) the performance of an occupancy map with a sparse set of points, sufficiently effective to represent the occupied and navigable space.

Apart from detecting obstacles, there is also an important need of moving safely between different points of the environment, without excessive deviations due to the avoidance of successive obstacles. To this end, this paper details a behavior-based control architecture which combines some of the advantages of T^2[1] and ND[13] in a unique system.

From T^2 our proposal inherits the capability of overcoming obstacles and deep deadlocks. From ND our system acquires (1) the ability of dealing with gaps between obstacles in scenarios densely occupied (with distances between obstacles close to S_{dist}), and (2) the methodology to plan the forthcoming motion toward the goal. Definitely, the combination of all these capabilities in a single visual solution permits to increase the scope of situations that can be overcome by the robot.

Acknowledgment

The authors are specially grateful to José Guerrero Sastre, a co-member of the Systems, Robotics and Vision Group, for his inestimable support and contribution with the configuration and execution of the simulations performed with the Player/Stage environment. This work is partially supported by the Spanish Ministry of Education under contracts PTA2011-05077 and DPI2011-27977-C03-02, FEDER funding, and Govern Balear (Ref 71/2011).

References

1. J. Antich and A. Ortiz, "Extending the Potential Fields Approach to Avoid Trapping Situations," *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Alberta, Canada, Aug. 2–6 (2005).
2. F. Bonin, A. Ortiz and G. Oliver, "Visual navigation for mobile robots: A survey," *J. Intell. Robot. Syst.* **3**(53), 263–296 (2008).
3. F. Bonin, A. Burguera, A. Ortiz and G. Oliver, "Concurrent visual navigation and localization using the inverse perspective transformation," *Electron. Lett.* **48**(5), 264–266 (2012).
4. F. Bonin, A. Ortiz and G. Oliver, "A Novel Inverse Perspective Transformation-based Reactive Navigation Strategy," *Proceedings of the 4th European Conference on Mobile Robots (ECMR)*, Mlini/Dubrovnik, Croatia, Sep. 23–25 (2009).

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press

5. F. Bonin, A. Ortiz and G. Oliver, "Experimental Assessment of Different Feature Tracking Strategies for an IPT-Based Navigation Task," *Proceeding of IFAC Intelligent Autonomous Vehicles Conference (IAV)*, Lecce, Italy, Sep. 6–8 (2010).
6. A. Burguera, Y. Gonzalez and G. Oliver, "The UspIC: performing scan matching localization using an imaging sonar," *Sensors* **12**(6), 7855–7885 (2012).
7. R. O. Duda and P. Hart, *Pattern Classification and Scene Analysis* (John Wiley, New York, NY, 1973).
8. J. W. Durham and F. Bullo, "Smooth Nearness-Diagram Navigation," *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS)*, Nice, France (2008).
9. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision* (Cambridge University Press, Cambridge, UK, 2003).
10. C. H. Kuo, Y. S. Syu, T. C. Tsai and T. S. Chen, "An Embedded Robotic Wheelchair Control Architecture with Reactive Navigations," *Proceedings of the IEEE Conference on Automation Science and Engineering* (2011).
11. B. Gerkey, R. Vaughan and A. Howard, "The player project," available at: http://playerstage.sourceforge.net/
12. Y. Koren and J. Borenstein, "Potential Fields Methods and Their Inherent Limitations for Mobile Robot Navigation," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)* (1991).
13. J. Minguez and L. Montano, "Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios," *IEEE Trans. Robot. Autom.* **23**(1), 45–59 (2004).
14. M. Mujahad, D. Fischer, B. Mertsching and H. Jaddu, "Closest Gap Based (CG) Reactive Obstacle Avoidance Navigation for Highly Cluttered Environments," *Proceedings of IEEE International Workshop on Intelligent Robots and Systems (IROS)* (2010).
15. J. Minguez, J. Osuna and L. Montano, "A Divide and Conquer Strategy Based on Situations to Achieve Reactive Collision Avoidance in Troublesome Scenarios," *Proceedings of IEEE ICRA* (2004).
16. D. S. O. Correa, D. F. Sciotti, M. G. Prado, D. O. Sales, D. F. Wolf and F. S. Osorio, "A Mobile Robots Navigation in Indoor Environments Using Kinect Sensor," *Proceedings of IEEE Brazilian Conference on Critical Embedded Systems* (2012).
17. J.-Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker," available at: http://robots.stanford.edu/cs223b04/algo_tracking.pdf (Intel Corporation Microprocessor Research Labs, 2000).

https://doi.org/10.1017/S0263574713000878 Published online by Cambridge University Press