OBSTACLE AVOIDANCE PROCEDURE FOR MOBILE ROBOTS

Marcelo Becker
PUC Minas – Mechatronic Engineering Department, Av. Dom José Gaspar, 500, 30.535-610, Belo Horizonte – MG. Brazil.
e-mail: marcelo.becker@pucminas.br

Carolina Meirelles Dantas
UFMG – Mechanical Engineering Department, , 30.535-610, Belo Horizonte – MG. Brazil.
e-mail: carol_md@ibest.com.br

Weber Perdigão Macedo
PUC Minas – Mechatronic Engineering Department, Av. Dom José Gaspar, 500, 30.535-610, Belo Horizonte – MG. Brazil.
e-mail: weber.perdigao@gmail.com

Abstract. *The common set of procedures provided by most of indoor mobile robots consists of Mapping Building, Auto-localization, and Navigation (Path Planning and Obstacle Avoidance). Considering that real indoor environments are commonly dynamic where known obstacles (e.g. chairs, tables, sofas, etc.) and unknown obstacles (persons walking close to the robot, unmapped environment characteristics, etc.) are constantly changing their positions, it is necessary to use sensor data to provide the robot a detailed real-time environmental view. Most of these obstacles may be located on a priori planned path and the robot controller must execute a skilful manoeuvre that avoids a collision and directs the robot to its goal position. Undoubtedly, it is essential that both navigation procedures (path planning and obstacle avoidance) work together and harmoniously, trying to solve possible conflicts between their two different behaviours (go to the goal position as soon as possible and avoid potential collisions during the path). In this article we describe the development of a new obstacle avoidance procedure based on the previous Obstacle Velocity approach. This procedure was denominated as BOA and was tested in a mobile robot in wide and narrow environments. The robot has been equipped with an intelligent control and navigation system and two SICK two-dimensional (2-D) laser range finder LMS 200.*

Keywords*: mobile robots, obstacle avoidance procedure, obstacle velocity.*

1. Introduction

Robotics has archived a great success in industrial manufacturing applications where manipulators are largely applied in assembly lines. They can make fast and precise movements to perform repetitive tasks such as spot welding, pick and place, and painting. Today there are numerous manipulator maker companies, most of them in the USA, Europe, and Japan. They make about US$ 2 billion a year selling robots and spare parts. However, these commercial robots have a considerable disadvantage: lack of mobility (Siegwart and Nourbakhsh, 2004). A fixed manipulator has a limited range of motions that depends on where it is bolted, its spare parts, and its degrees of freedom (quantity and type). If the commercial robots had the abilities to move in different indoor environments, extract their environment characteristics, adapt well to environmental changes, and work precisely on several tasks and assembly lines, they would improve considerably their versatility. This kind of robot is known as mobile robot.

The present work focuses one part of the navigation problem in autonomous mobile robots. Supported by the previous considerations, we focused our work on a different approach for local obstacle motion detection and robot speed control for autonomous and semiautonomous mobile robots that avoid a collision between the robot and obstacles. The robot's obstacle avoidance routine is able to deal with narrow and wide navigation areas, executing local maneuvers in narrow and cluttered spaces. We are assuming that the robots use a complete road map (previously provided or built) to apply some path planning routine that, based on known stationary obstacles, produced the desired trajectory.

2. Obstacle Avoidance

Recently, many researches turned their attention to obstacle avoidance problem developing interesting real-time approaches for narrow and cluttered spaces. However, there are some classic obstacle avoidance methods that must be cited (Borenstein and Koren, 1991): *edge-detection*, *certainty grids*, and *potential field methods*. The first one, edge-detection, is a very popular method that extracts the obstacle vertical edges and drives the robot around either one of the visible edges. This approach was early commonly combined with ultrasonic sensors. Due to the limited accuracy of the sensor, the approach presented some shortcomings: poor directionality, frequent misreadings, and specular reflections. On the other hand, Moravec and Elfes (1985) pioneered the concept of certainty grid, a map representation that is well suited for sensor data accumulation and fusion. Certainty grid is an obstacle probabilistic representation method that uses a grid-type world model. The robot's work area is modeled as a 2-D array of square elements, called cells. Each

cell has a certainty value (*CV*) that indicates the measure of confidence that an obstacle is within the cell area. The *CV* is a probability function that depends on the sensor characteristics. As each cell has its *CV* updated constantly by the sensor readings, after a period moving across an area, the robot has a fairly accurate map of that area. The method accuracy is a function of the cell size and may be considered as its drawback as well. The third method, potential field method is based on the idea that obstacles exert imaginary repulsive forces, while the goal position applies an imaginary attractive force to the robot. The resultant robot behavior is obtained summing all attractive and repulsive forces.

The potential field method was later improved by Koren and Borenstein integrating its concept with the certainly grid concept. Based on the certainly grid data, a 2-D Cartesian histogram grid (bar graph in which the area of each bar is proportional to the frequency or relative frequency represented) is used to represent the probability of each cell contain an obstacle. After that, the potential field idea is applied to the histogram grid in order to obtain a fast reflexive obstacle avoidance behavior. This new method was named Virtual Force Field method (*VFF*). Nevertheless, after some experiments, it was abandoned due to the method instability and inability to pass through narrow passages like doors (local minima problem). Repulsive forces from the both sides of the doorway results on a force that pushes the robot away.

In the 1990s Koren and Borenstein (1991) developed the Vector Field Histogram (*VFH*) approach and afterward, Ulrich and Borenstein (1998, 2000) made some incremental improvements called VFH^+ and *VFH** approaches. The *VFH* methods create a local certainly grid map of the robot surround environment using sensor readings. Instead of a 2-D Cartesian, a Polar histogram (α-P) is built based on the certainly grid map. One should observe that α is the sensor angle and P is the probability that there is an obstacle in that direction. A probability threshold value is used in order to determine which directions may be considered as obstacle-free ones. Taking into account the robot's size and shape (configuration space), all obstacle-free directions are checked to verify if they are large enough for the robot to pass through. A masked polar histogram where the obstacles are enlarged is calculated. After that, the steering direction for the robot is chosen. In the VFH^+ improvement, the basic robot kinematics limitations were used to compute the robot possible trajectories using arcs or straight lines. Finally, in 2000 the *VFH** improvement proposed the look-ahead verification. The method analyses each possible direction provided by the VFH^+ approach, checking their consequences concerning the robot future positions. It projects the robot trajectory several steeps ahead, building a search tree where the end nodes correspond to a total projected distance.

Simultaneously, another method based on the admissible robot velocities was proposed. These methods are named *Steer Angle Field Approaches*. In 1997, the Dynamic Window Approach (*DWA*) was developed by Fox, Burgard and Thrun (1997). This approach takes into account robot kinematics constraints in order to calculate all possible sets of velocity vectors (v, ω) in the velocity space. One should observe that v and ω are the robot translational and rotational velocities, respectively. Considering the robot possible accelerations, the overall search velocity space is reduced to the dynamic window, which contains only the velocities that can be reached within the next time interval. The dynamic window is a rectangle centered on the robot present velocity and its vertex positions depend on the accelerations that can be applied. The dynamic window has a rectangular shape because it was assumed that the robot dynamic capabilities for translation and rotation are independent. All velocity vectors outside the dynamic window cannot be reached within the next time interval and thus should not be considered for the obstacle avoidance. The motion direction is chosen by applying an objective function to all admissible velocity vectors in the dynamic window. This objective function depends on the robot velocity, the distance between the robot and the closest obstacle, and the robot progress toward the goal position.

Brock and Khatib (1999) proposed a significant improvement to the dynamic window approach. They added a global thinking to the *DWA* by using the grassfire technique for finding routes in the certainly grid cells. Each cell is labeled with the distance to the robot's goal position (like a wave front expansion from the goal position outward). The desired trajectory is obtained by linking adjacent cells that are closer to the robot's goal position. This procedure allows the robot to improve their performance by using some of the advantages of global path planning without complete a priori knowledge. This procedure was named Global Dynamic Window Approach (*GDWA*).

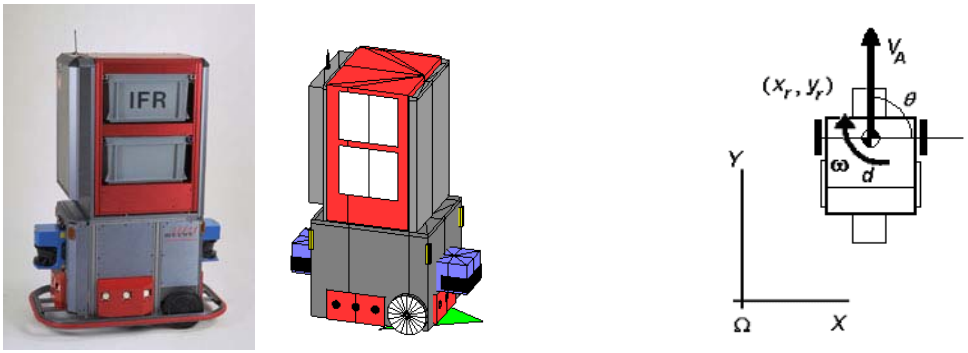
3. Velocity Obstacle Approach

Other well-known *Steer Angle Field* Method is the *Velocity Obstacle Approach* (VOA). This method was proposed by Fiorioni and Shiller in 1993, but Prassler, Scholz and others proposed some improvements later. Initially the VOA was implemented in MAid – a robotic intelligent wheelchair fully capable of driving in narrow and wide areas, avoiding collisions with pedestrians and retaining user independence. As the present work was based on the *Velocity Obstacle Approach*, this method is focused ahead.

3.1 Robots General Motion Equations

Let $x_r(t)$ and $y_r(t)$ represent the robot's position at time t in some global coordinate system (Ω), and let the robot's orientation (heading direction) be represented by $\theta(t)$. The triplet (x_r, y_r, θ) describes the robot's kinematic configuration. In our case, the robot used (Fig. 1) has two independently drive wheels at a distance d, with fixed

maximum wheel velocity V_{max} and maximum wheel acceleration a_{max}. Let a_l and a_r be respectively the accelerations of the left and right wheels, based on Fig. 1-b the kinematic equations that describe the robot motion are as follow:



(a) (b)

Figure 1. The mobile robot used (a) MOPS and its MatLab model and (b) its kinematic model: Ω is the global coordinate system.

$$\dot{X} = V_A.cos\theta \tag{1}$$

$$\dot{Y} = V_A.sin\theta \tag{2}$$

$$\varpi = \dot{\theta} \tag{3}$$

$$\dot{V} = (a_r + a_l)/2 \tag{4}$$

$$\dot{\varpi} = (a_r - a_l)/d \tag{5}$$

One may observe that the robot motion state is given by its translational and angular velocities, respectively V_A and ω. Consequently, robot's state is given by (x, y, θ, V_A, ω).

3.2 Evasive Maneuvers for VOA

For simplicity, the robot and the obstacles (moving or stationary ones) are modeled as circles where they were inscribed on taking into account their sizes and shapes. Given the robot and an obstacle, represented by circles A and B_i, respectively, as shown in Fig. 2-a at time t_0, with velocities V_A and V_{Bi}, the robot is considered as a point ($\hat{A}$) and the B_i circle (the obstacle) is enlarged by the A circle radius, resulting in $\hat{B}_i$ circle:

$$\hat{r}_{Bi} = r_{Bi} + r_A \tag{6}$$

After that, a *Collision Cone*, CC_{ABi}, is calculated. The CC_{ABi} is defined as the set of colliding velocities between $\hat{A}$ and $\hat{B}_i$:

$$CC_{ABi} = \left\{V_{ABi} \mid \lambda_{ABi} \cap \hat{B} \neq \emptyset\right\} \tag{7}$$

Where V_{ABi} is the relative velocity of Â with respect to $\hat{B}_i$ is defined as:

$$V_{ABi} = V_A - V_{Bi} \tag{8}$$

And λ_{ABi} is the line of V_{ABi}. The CC_{ABi} is the light gray sector with vertex in $\hat{A}$, bordered by the two tangents λ_f and λ_r from $\hat{A}$ to $\hat{B}_i$ (concentric to B_i), shown in Fig. 2-b. The collision cone represents the set of colliding *relative* velocities between $\hat{A}$ and $\hat{B}_i$.

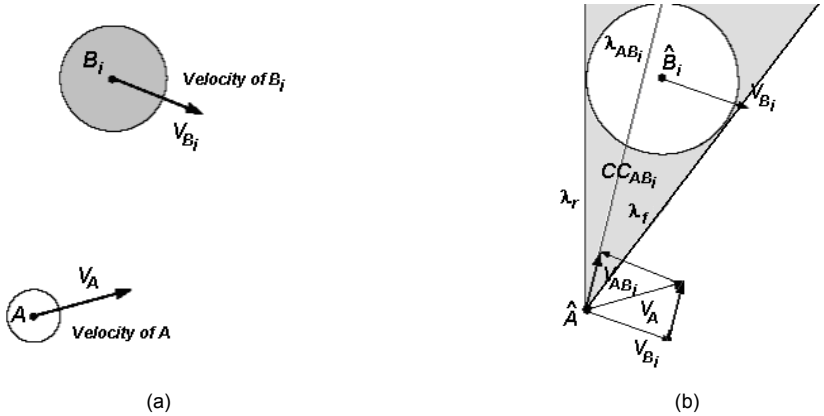


Figure 2. Velocity Obstacle Approach: (a) the robot is represented by A and the obstacle, by B_i. (b) Collision Cone between $\hat{A}$ and $\hat{B}_i$: CC_{ABi}.

With this representation, any relative velocity that lies between the two tangents to $\hat{B}_i$, λ_f and λ_r will cause a collision between A and B_i. As the CC_{ABi} is specific to a robot / obstacle pair, for multiple obstacles, it is easier to calculate the best V_A velocity using the VO_{Bi} instead the CC_{ABi}. The VO_{Bi} is computed by translating each point of the CC_{ABi} between A and B_i, by vector V_{Bi}:

$$VO_{Bi} = CC_{ABi} \oplus V_B \tag{9}$$

Where $\oplus$ is the Minkowski vector sum operator. After that, individual VO_{Bis} are combined to form:

$$VO = \cup_{i=1}^{n} VO_{Bi} \tag{10}$$

Where n is the number of obstacles.

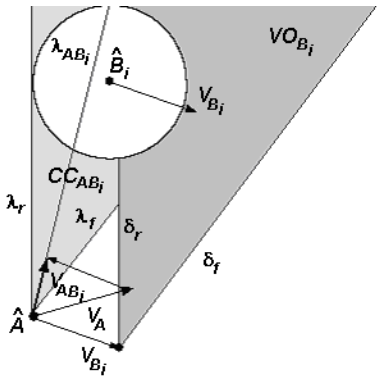


Figure 3. Velocity Obstacle Approach: VO_{Bi}.

The VO of A with respect to B is the dark gray sector shown in Fig. 3. The velocities V_A, whose tips are inside the VO, are called *dangerous velocities* because the will cause a collision between the robot and an obstacle in future. The velocities V_A, whose tips are outside the VO, are called avoidance velocities. In addition, the velocities V_A, whose tips are on the borders of VO would result in A grazing the obstacle B_i. Therefore, since VOA is based on a linear approximation of each obstacle trajectory, obstacles with imminent collision are avoided first. The use of a suitable *Time Horizon*, T_h, the collision avoidance procedure is limited to those occurring at a time $t < T_h$.

To guarantee that the avoidance maneuvers are executable by the robot, the robot dynamics must be considered; i.e., it is necessary to verify if the robot motors have enough torque to achieve the desired velocity within the allowed time interval. On account of that, the set of avoidance velocities must be limited to those velocities that satisfies the robot dynamic constraints. Similarly to *DWA*, it is used a polygon to represent the set of *reachable avoidance velocities* (*RAV*). This set is graphically determined, by superimposing to the velocity obstacle VO a polygon representing the set of all *reachable velocities* and computing by difference the RAV set. The result is represented by the polygon PQRS

shown in Fig. 4, where the outer parallelogram represents the set of all reachable velocities, the inner gray area represents a *VO* segment, and the remaining clear area is the *RAV* set.

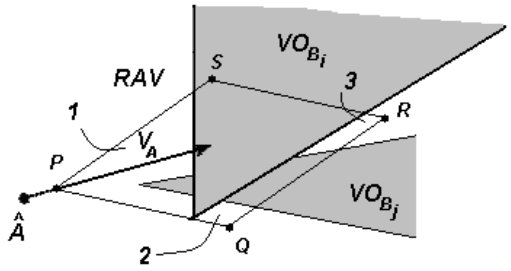


Figure 4. Velocity Obstacle Approach: The Reachable Avoidance Velocities (RAV) polygon represented by white areas inside the PQRS polygon and the present robot velocity (V_A).

One should notice that the P, Q, R, and S vertices correspond to the maximum accelerations and velocities that are related to maximum wheel velocity V_{max} and maximum wheel acceleration a_{max}. Observing Fig. 4 it is possible to verify that *RAV* may consist of many different areas, in this case, areas 1 to 3. These areas produce distinct robot behaviors during obstacle avoidance maneuvers. For example, choosing velocities from area 1 would result in decelerating the robot to stay behind the obstacles B_i and B_j. On the other hand, choosing velocities form areas 2 or 3 would result in accelerating the robot to pass the obstacle B_i and avoiding the obstacle B_j on the right or on the left side respectively. Based on this, some heuristics may be created to satisfy a hierarchy of objectives, such optimizing some trajectory reaching the goal, optimizing some trajectory parameter, etc. It is important to emphasize that the purpose of the heuristic search is to find a good local solution, if one exists.

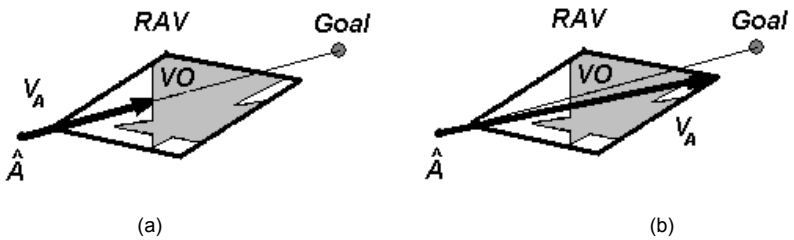


Figure 5. Velocity Selection Heuristics: *TG* (a) and *MV* (b) strategies.

There are some heuristics in the literature, for example, the *TG* and *MV* strategies, respectively, *Towards Goal* and *Maximum Velocity* strategies (Fig. 5). The first one chooses the velocities along the straight line to the goal that would ensure reaching the goal; otherwise, the second one chooses the highest feasible velocity in the general direction of the goal that would reduce the motion time.

3.3 Evasive Maneuvers for *BOA* Procedure

Initially we assumed that the robot's navigator is decomposed into the *path planning* and *obstacle avoidance* routines that are both always active. Robot's path planning routine is constantly calculating the optimal trajectory on a given map that drives the robot from its present position to its goal position. The path planning output signals are related to the wheels velocities and accelerations. Every time step this routine provides the robot with the desirable robot's velocity, which modulus is $\|V_A\|$ and direction is ϕ, that drives the robot to its goal position. Concerning the obstacle avoidance routine, we worked on the *VOA* set of assumptions about the robot and obstacles modeling and CC_{ABi} and VO_{Bi} calculus. When it comes to the *RAV* application to choose the robot's velocity, we used a new approach named *BOA*. This approach is based on the heuristics of the constant cruiser velocity, i.e., the robot should keep constant its velocity modulus as long as possible during the trajectory to its goal position. Only if a collision is imminent, the robot's velocity direction is changed. Taking into account robot's maximum translational and rotational accelerations it is possible to determine the limit values of robot's velocity modulus and direction centered on $\|V_A\|$ and ϕ (previously provided by the path planning routine) that define the area limited by ϕ_l, ϕ_r, $\|V_{Amax}\|$, and $\|V_{Amin}\|$. This area represents the robot *reachable velocities* for the next time step based on the path planning output. If we assume that the robot dynamic capabilities for translation and rotation are independent, this area is a circle sector as represented on Fig. 6-a by the dark

gray area. Following the heuristics, the *BOA* procedure tries to keep the robot's velocity modulus ($\|V_A\|$) constant during the avoidance maneuver, varying its direction (ϕ) within the interval [ϕ_l, ϕ_r].

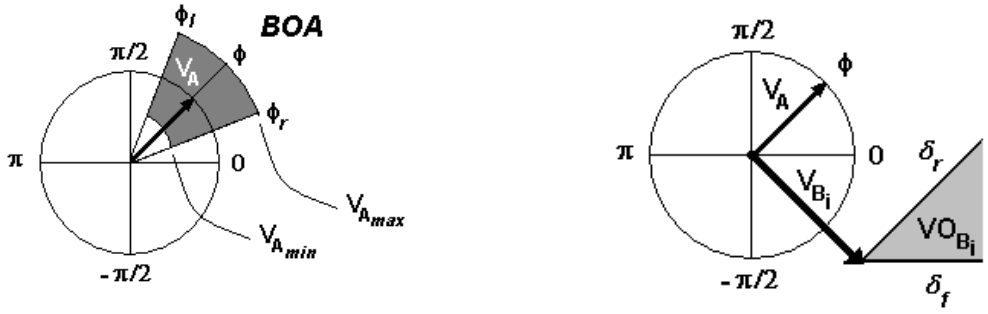


Figure 6. *BOA* Procedure: (a) the circle of radius $\|V_A\|$ represents the velocity space and the dark gray area limited by ϕ_l, ϕ_r, V_{Amax}, and V_{Amin} represents the set of *reachable velocities*. (b) The use of VO_{Bi} to calculate the *dangerous velocities*.

Similarly to *VOA*, we use the VO_{Bi} instead the CC_{ABi} to calculate the *dangerous velocities directions -DVDs* (Fig. 6-b). The *DVDs* are obtained through the intersection points between the velocity space circle and the two lines δ_f and δ_r that border the VO_{Bi} (Fig. 7-a). The circle arcs limited by α_1, α_2, α_3, and α_4 represent the *DVD* sets (Fig. 7-b). Any velocity, whose direction (ϕ) lies within [α_1, α_2] or [α_3, α_4], would cause a future collision between the robot and the B_i obstacle. If there is no intersection between de velocity space circle and the VO_B, there is no imminent collision. In case of multiple obstacles, multiple VO_Bs are used in order to obtain all the *dangerous velocity direction* sets (Fig. 8).

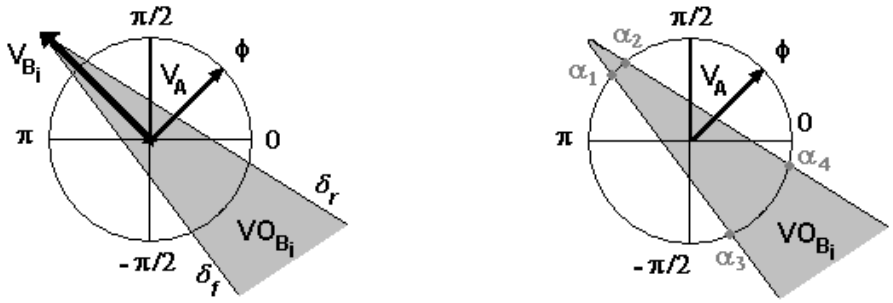


Figure 7. *BOA* Procedure: (a) and (b) the *dangerous velocity direction* calculus in the velocity space.

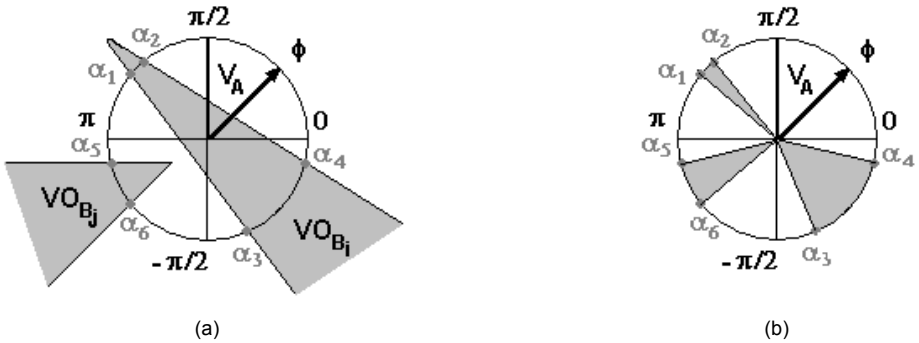


Figure 8. *BOA* Procedure: (a) two obstacles represented by two VO_Bs (VO_{Bi} and VO_{Bj}). (b) Velocity Space and the 3 *dangerous velocity direction* sets obtained.

If it is not possible to avoid a collision only varying the velocity direction within [ϕ_l, ϕ_r], the *BOA* procedure check variations on robot's velocity modulus ($\|V_A\|$) and direction within ϕ_l, ϕ_r, V_{Amax}, and V_{Amin} limits, searching for possible

velocities to avoid the collision. The *BOA* procedure accelerates, decelerates, and even stops the robot waiting until the group of obstacles blocking its way had passed and then restarts its trajectory. Figure 9 shows how variations on velocity modulus affect the *DVD* sets. Accelerating the robot would decrease the *DVD* set interval from [α_3, α_4] to [α_3', α_4'] and would eliminate the [α_1, α_2] interval (Fig. 9 – circle II). On the other hand, decelerating the robot would increase both *DVD* set intervals from [α_1, α_2] and [α_3, α_4] to [α_1'', α_2''] and [α_3'', α_4''], respectively (Fig. 9 – circle III). As *VOA*, the *BOA* may use some heuristics to decide which velocity would be used to avoid the collision. We decided to use the heuristics that results in the robot grazing the obstacles. It is important to notice that due to sensor uncertainty, noise and misreadings, sometimes it is difficult to know exactly the obstacle shapes and sizes. For this reason, in order to add an additional safety margin to the *BOA* routine, the *DVD* sets are diminished by an uncertain coefficient (*uc*). The *uc* value is a function of the sensor type, reliability and accuracy.

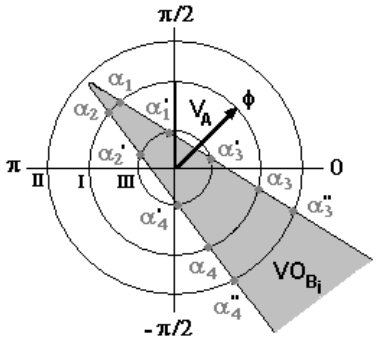


Figure 9. *BOA* Procedure: illustration of the velocity modulus variation effect on *DVD* sets.

4. Results

The *BOA* procedure was tested using real sensor data provided by the mobile robot MOPS. MOPS has been equipped with an intelligent control and navigation system and two SICK two-dimensional (2-D) laser range finder LMS 200, front, rear and lateral sonar rings, and touch sensor bumper. The robot performed several maneuvers in wide and narrow environments. The results obtained were encouraged. During the maneuvers the *DVD* sets were obtained using the LMS data (see Figures 10-a and 11). But one should notice that using only the LMS data there are two large blind lateral areas. Due to this, for real navigation procedures, the MOPS' sonar rings and data fusion algorithms are essential in order to detect obstacles in robots vicinities. Figure 10 shows MOPS driving in a small corridor that was previously mapped. Only two obstacles did not match with map features and represent potential mobile obstacles that should be verified (tracked) for a future classification into mobile or static obstacles. Figure 11 shows the *DVD* sets behavior when mobile obstacles (in this case, 1 and 2 persons, respectively for Figure 11-a and 11-b) are detected close to MOPS. As we used the heuristics that results in the robot grazing the obstacles, the robot would firstly try to avoid the obstacles grazing them. If it is not possible, it starts to search for possible velocities to avoid the collision.

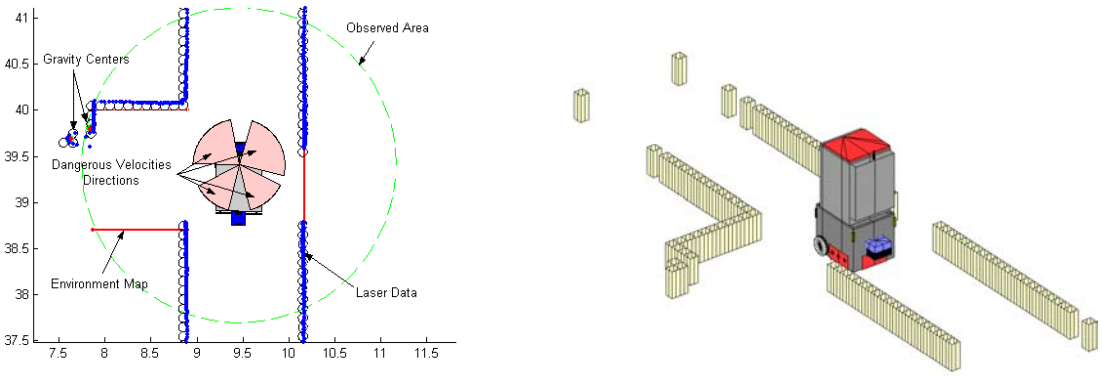


Figure 10. *BOA* Procedure (a): illustration of the *DVD* sets (salmon areas), laser data (blue dots), environment map (red lines), the unmatched obstacles gravity center (red filled circles) and the observed area (green circle). 3-D illustration of the MOPS vicinity (b).

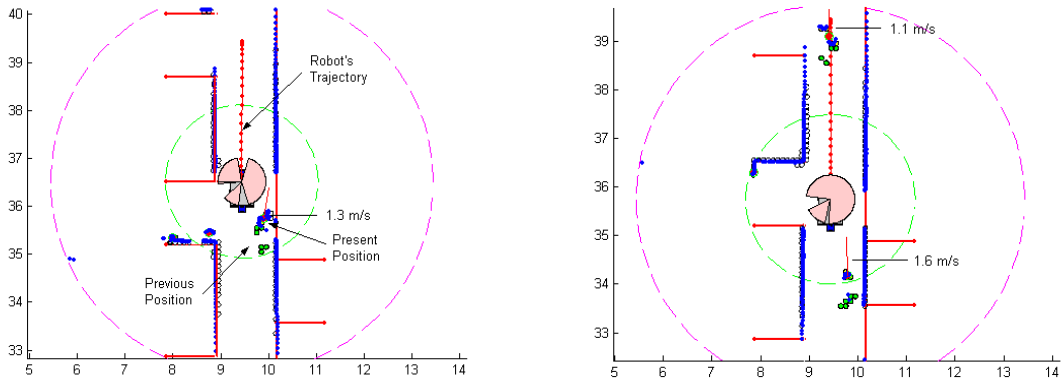


Figure 11. *BOA* Procedure: illustration of the *DVD* sets (salmon areas), laser data (blue dots), environment map (red lines) a person walking close to MOPS at 1.3 m/s (a) and 2 persons walking close to MOPS at 1.1 m/s and 1.6 m/s (b).

5. Conclusions

The article described the development of a new obstacle avoidance procedure based on the previous Obstacle Velocity approach. This procedure was denominated as *BOA* and was tested in a mobile robot (MOPS) in wide and narrow environments. The robot has been equipped with an intelligent control and navigation system, two SICK two-dimensional (2-D) laser range finder LMS 200, front, rear and lateral sonar rings, and touch sensor bumper. The results obtained were encouraging. The MOPS was able to drive in narrow and large environments calculating the *DVD* sets and avoiding collisions. Nevertheless, the results also indicated that it is necessary to use different kind of sensors and multi-sensor data fusion in order to extract more environment features and avoid blind areas. That happened because the robot uses these features to match obstacles and classify them into mobile and static ones. We are planning in a near future the implementation of particle filters aiming the multi-sensor data fusion procedure.

6. Acknowledgements

This work was supported by the Minas Gerais State Agency of Science, Research, and Technology (FAPEMIG-Brazil) under grant no. 1971/03. The development of the *BOA* approach was carried out in part at the Institute of Robotics, Swiss Federal Institute of Technology, Zurich (IfR – ETHZ). We would like to thank Ing. Felix Wullschleger, Dr. Sjur Vestli, and Prof. Gerhard Schweitzer for their support, help and the IfR's facilities.

7. References

Borenstein, J. and Koren, Y., 1991, "The vector field histogram – fast obstacle avoidance for mobile robots", IEEE Journal of Robotics and Automation, vol. 7, n. 3, pp. 278-288.

Brock, O. and Khatib, O., 1999, "High speed navigation using the global dynamic window approach", Proceedings of the 1999 IEEE International Conference on Robotics & Automation, Detroit, MI, pp. 341-346.

Fox, D., Burgard, W. and Thrun, S., 1997, "The Dynamic Window Approach to Collision Avoidance", IEEE Robotics & Automation Magazine, March 1997, pp. 23-33.

Fiorioni, P. and Shiller, Z., 1993, "Motion Planning in Dynamic Environments using the Relative Velocity Paradigm", Proceedings of the 1993 IEEE International Conference on Robotics & Automation, Atlanta.

Movarec, H.P. and Elfes, A., 1985, "High Resolution Maps from Wide Angle Sonar", IEEE Conference on Robotics and Automation, Washington D.C., pp. 116-121.

Prassler, E., Scholz, J., Strobel, M. and Fiorini, P., 1999, "An Intelligent (Semi-) Autonomous Passenger Transportation System", IEEE, ISBN 0-7803-4975-X, pp. 374-379.

Siegwart, R. and Nourbakhsh, I.R., 2004, "Introduction to Autonomous Mobile Robots", ISBN 0-262-19502-X, 331 pp.

Ulrich, I. and Borenstein, J., 1998, "VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots", Proceedings of the 1998 IEEE International Conference on Robotics & Automation, Leuven, Belgium, May 1998, pp. 1572-1577.

Ulrich, I. and Borenstein, J., 2000, "VFH*: Local Obstacle Avoidance with look-ahead Verification", Proceedings of the 2000 IEEE International Conference on Robotics & Automation, San Francisco, CA, April 2000.

8. Responsibility notice

The authors are the only responsible for the printed material included in this paper.