NAVIGATION OF AN AUTONOMOUS CAR USING VECTOR FIELDS AND THE DYNAMIC WINDOW APPROACH

Danilo Alves de Lima[*]
daniloalvesdelima@yahoo.com.br

Guilherme Augusto Silva Pereira[*]
gpereira@ufmg.br

[*]Grupo de Pesquisa e Desenvolvimento de Veículos Autônomos, Escola de Engenharia,
Universidade Federal de Minas Gerais, Av. Antônio Carlos, 6626, 31270-901,
Belo Horizonte, MG, Brasil.

ABSTRACT

This work presents a safe navigation approach for a car-like robot. The approach relies on a global motion planning based on Velocity Vector Fields along with a Dynamic Window Approach for avoiding unmodeled obstacles. Basically, the vector field is associated with a kinematic, feedback-linearization controller whose outputs are validated, and eventually modified, by the Dynamic Window Approach. Experiments with a full-size autonomous car equipped with a stereo camera show that the vehicle was able to track the vector field and avoid obstacles in its way.

KEYWORDS: Safe navigation, Car-like robot, Vector Field, Dynamic Window Approach.

RESUMO

Este trabalho aborda o problema de navegação segura de um carro autônomo. Para tanto, é utilizado um planejamento de movimento por meio de Campos Vetoriais de Velocidade aliado ao Método da Janela Dinâmica para o desvio de obstáculos não modelados. Basicamente, o campo vetorial é associado a um controlador cinemático baseado em linearização por realimentação de estados cujas saídas são validadas, e eventualmente modificadas, pelo Método da Janela Dinâmica. Resultados experimentais com um automóvel autônomo equipado com uma câmera estéreo mostraram que a metologia foi capaz de guiar um carro autônomo pelo campo vetorial, fazendo-o desviar de obstáculos em seu caminho.

PALAVRAS-CHAVE: Navegação segura, Carro autônomo, Campo Vetorial, Método da Janela Dinâmica.

1 INTRODUCTION

Several research groups around the world have been working in the development of autonomous or semi-autonomous cars (Nothdurft et al., 2011; Levinson et al., 2011; Milanes et al., 2010). Part of this development was motivated by the DARPA Grand Challenges, competitions promoted by the American's Defence Advanced Research Projects Agency (DARPA) between 2004 and 2007. In these competitions, unmanned cars should perform autonomous tasks in a desert rally or in a urban environment race. Besides the competitions, one of the main goals of the research in autonomous or semi-autonomous vehicles is to deliver new technologies. In a near future, these technologies may be integrated to the commercial automobiles, equipping this vehicles with systems that increase the comfort and, mainly, the safety of their drivers and passengers. Some of these systems are already commercially available, such as the autonomous parking system, which allow the car to autonomously search and enter a parking spot, and the adaptive cruise control systems, which control the vehicle speed and keep it in a safe distance from other cars in the road. Several other applications have been foreseeing by the researchers (Vermaas et al., 2009; McBride et al., 2008).

In Brazil, including the group of the authors at the Universidade Federal de Minas Gerais (UFMG), there are at least five groups working on the development of autonomous cars. As the result of their work, some vehicles were developed: Driving4U (Honório et al., 2010), at UNIFEI and UFJF; VERO (Mirisola et al., 2011), at CTI; SENA (Megda et al., 2011) and CARINA (Fernandes et al., 2010), at USP/SC; and

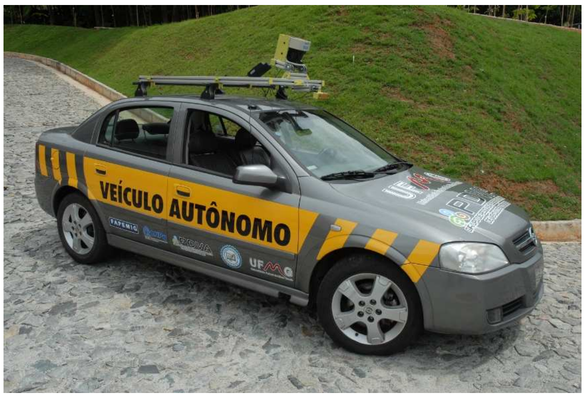


Figure 1: CADU: The autonomous car under developed at UFMG.

CADU (Freitas et al., 2009) at UFMG. The last one is shown in Figure 1. Although those groups are from different backgrounds and have distinct scientific objectives, all of them are developing technologies to make a car to drive itself controlled by a computer system that relies solely on the vehicle on-board sensors.

Actually, to allow a car to drive itself safely, it must be equipped with a complete navigation system. This system includes, among several other subsystems, a *planner*, which computes a path from its origin to its destination, a *perception system*, which detects the drivable path in front of the vehicle and obstacles in this path, and a *controller*, responsible to drive the vehicle through the path planned by taking into account the information gathered by the perception system. These three subsystems, their computational implementation and integration are the focus of this paper. The localization system, very important to make the car aware of its global position, and the low level controllers, that in fact actuate the car, are not deeply covered by the paper. To the best of the authors knowledge, although the paper does not present new methodologies for planning, perception, and control, it presents a novel integration of three previous published techniques to compose a complete navigation system. This navigation system allows an autonomous car to navigate in an environment with obstacles. Moreover, an important contribution of the authors is the implementation and evaluation of the proposed solution in the full-size autonomous car under development at UFMG (see Figure 1). This paper is an extended version of the one published in Portuguese by the authors (Lima and Pereira, 2011)

Several solutions where presented in the literature to solve the problem of autonomous navigation. A review that compare some of these methodologies can be found in (Lima, 2010). Among them, most use the idea of precomputing a path or trajectory that is followed with aid of a specific controller (Thrun et al., 2006; Braid et al., 2006). When unmodeled obstacles are detected in the path of the vehicle, a new path is computed to avoid the obstacles. Different from these works, in order to avoid path replanning and to simplify the vehicle controllers, the proposition of this paper is to use velocity vector fields to integrate planning and control in the same algorithm. Since the autonomous vehicles are, generally, velocity controlled, the idea is to use the vector field as input for the car. When an obstacle is detected, the field is locally modified without the need of replanning.

Therefore, the solution proposed in this work was conceived given the principle that the low level controllers of the car should receive velocity setpoints. We then propose the composition of two main techniques: (i) a Velocity Vector Field (Pereira et al., 2009) and (ii) the Dynamic Window Approach (DWA) (Fox et al., 1997). The first is the main responsible for the vehicle motion planning. It associates a velocity vector for each allowed vehicle configuration. The second technique is the one responsible for dealing with unmodeled obstacles encountered in the robot's path. It validates the velocities given by the vector field based on a local map constructed with the vehicles's on-board sensors. When a velocity vector given by the vector field can make the vehicle to collide, the DWA computes a new, free-of-collision vector without changing the original vector field. To construct the local map used by the DWA, it is used an Occupancy Grid (Elfes, 1989). This may be constructed with any set of range sensors available on the vehicle. In the experimental results presented in this paper, data from an stereo camera was applied in this task.

Related to the methodology presented in this paper are the works by Brock and Khatib (1999) and Rebai et al. (2007). The first one also integrates a global motion planning algorithm to the DWA. Since the authors were using holonomic or sincro-drive robots, the standard DWA was directly used. Several other details, such as the construction of the plan during the navigation, makes that work very different from the present one. On the other hand, Rebai et al. (2007) present a DWA modified for car-like robots. This approach is directly used in the present work, but now, it is integrated to a global motion planning approach given by a vector field methodology. Other important differences between the works are the model used for the car, the variables used to define the dynamic window, and the absence of real world experiments in (Rebai et al., 2007).

The complete navigation solution proposed in this paper was segmented into two complementary parts: workspace perception and navigation, as shown in the block diagram of Figure 2. All the blocks in Figure 2 will be detailed in the next sections of the paper that is divided as followed: Section 2 presents the kinematic model of the car used in the paper. Sections 3 and 4

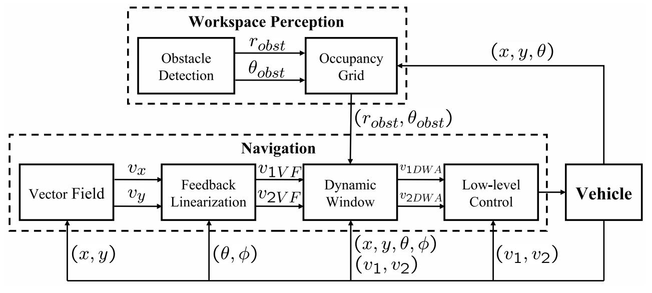


Figure 2: Block diagram of the proposed navigation solution.

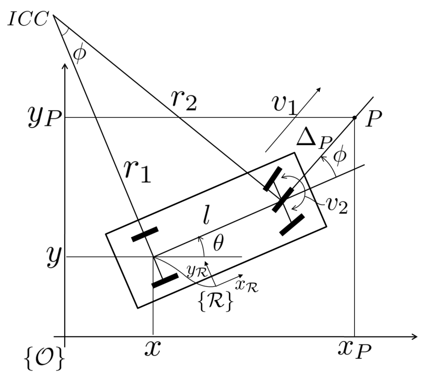


Figure 3: Cinematic model of the vehicle (refer to Equation (1)). In this model the vehicle move about circular trajectories defined by instantaneous center of curvature(ICC). Point $P = (x_P, y_P)$ is the projection required by the static feedback linearization controller, as seen in Section 4.3.

present the two dashed blocks in Figure 2. Details of the experimental setup used to validate the methodology along with simulated and experimental results are presented in Section 5. Finally, section 6 presents conclusions and perspectives for future work.

2 VEHICLE MODELING

This work considers a front-wheel drive car. It can be kinematically represented as (Luca et al., 1998):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi/l \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \,, \tag{1}$$

where the vehicle configuration is given by $(x,\ y,\ \theta,\ \phi)$, where $(x,\ y)$ and θ are the position and orientation of the car's reference frame $\{\mathcal{R}\}$ in relation to a static world reference frame $\{\mathcal{O}\}$, and ϕ is the steering angle of the vehicle. The steering angle may be computed as the average of the angles of each front wheel following the Ackerman approximation (Luca et al., 1998). Figure 3 illustrates these variables. Notice in this figure that the origin of $\{\mathcal{R}\}$ is located at the midpoint between the two rear wheels while its x axis points to the front of the vehicle. Also, observe that in the midpoint of the front axle is a virtual wheel related to the Ackerman approximation, whose orientation in relation to $\{\mathcal{R}\}$ is given by ϕ.

The inputs of model (1) are the linear velocity of the front wheels, v_1, and the steering velocity, v_2. Constant l represents the distance between rear and front axles. It is important to notice that the linear velocity of the virtual wheel, which corresponds to the input v_1, is different from the linear velocity of the vehicle that is given by $v_1 \cos\phi$.

Once the time derivative of the configuration is computed using (1), it is possible to predict the robot movement by integrating the computed values over time. The prediction of the robot movement will be important to construct the dynamic window (DW) for collision avoidance, as will be shown in Section 4.2. However, it is important to notice that Equation (1) is a kinematic model that is based on some strong assumptions related to the vehicle, such as its rigidity and immunity to wheel slippage. In the real world, most of this assumptions may be considered valid only at low speeds. Therefore, in high speed applications, a dynamic model must be considered. In the experiments presented in this work, the vehicle maximum speed was set to be 25 km/h. At this speed, the car was, apparently, well approximated by the model in Equation (1), mainly regarding localization and collision avoidance. Regarding the vehicle control, this kinematic model was only applied as a base to a feedback linearization strategy that transforms velocity vectors given by the vector field into the model inputs v_1 and v_2, as will be explained in Section 4.1. The low level controllers that guarantee that v_1 and v_2 will be followed by the vehicle, on the other hand, relies on dynamic models that are out of the scope of this paper. This controllers are briefly discussed in Section 4.3.

3 WORKSPACE PERCEPTION

Workspace perception is an important task to guarantee that the vehicle will not collide with previously unmodeled obstacles. In this work, an occupancy grid is the main tool used in this task, as shown in Figure 2.

Occupancy Grid (Elfes, 1989) is a probabilistic technique for workspace mapping using sensor data collected during the robot movement. It implicitly includes data filtering and fusion when several sensors are used. In this technique, the map is stored as a data matrix (grid), where each cell represents the probability of occupation of a given position of the workspace. Therefore, the access to the data is very simple and efficient.

However, due to the grid based representation, occupancy grids for large environments, such as outdoor environments, may yield in high memory and processing consumption. The solution adopted in this work was to reduce the grid to a small window around the vehicle, then composing a Local Occupancy Grid. This window is constantly updated with the car movement and may be written as the problem of finding a *posteriori* probability of a map cell from a series of data, as follow:

$$p(m_i|z_{1:t}, x_{1:t})\,, \tag{2}$$

where m_i is the grid cell with index i, $z_{1:t}$ is the sensors' measurement set at time t, and $x_{1:t}$ is the path followed by the vehicle.

In our implementation, it is not necessary to define a specific sensor, once the local occupancy grid only requires information about the detected obstacles on a two dimensional input given by $(r_{obst}, \theta_{obst})$, which represents distance and bearing in relation to the vehicle reference frame. This allows for the use of several sensors, which would be implicitly fused by the methodology.

In the experimental results presented in this paper, a stereo camera (a pair of cameras mounted in a rigid aluminum case) pointing to the front of the vehicle is used to detect obstacles and generate information for the computation of the occupancy grid. The images from the stereo camera are used to construct a V-Disparity Map (Broggi et al., 2005), which allows an easy classification of the images into obstacles or drivable areas. Details of this methodology can be found in (Lima and Pereira, 2010).

An important observation is that, since the occupancy grid relies on current and past data, the vehicle perception of the environment is increased with the use of this technique, specially if static or quasi-static environments are considered. In this way, even using a sensor with narrow field-of-view (FOV), such as the camera used in this paper, the vehicle may have a 360 degrees perception during its movement. An illustration of a local occupancy grid can be seen on the right hand side of Figure 4, where dark regions represent free workspace and white regions represent obstacles with maximum probability. This figure also shows, on the left hand side, a snapshot of a simulation where an autonomous car moves in a map with modeled (large box) and unmodeled obstacles (small ellipses). In this simulation, the occupancy grid was constructed with a 43° FOV vision system. This FOV is represented in the map of the left hand side of Figure 4 by the pink dots. Notice in this figure that, although the beginning of the obstacle close to the car (the black ellipse) is not inside the FOV of the sensor, this part of the obstacle is still represented in the local occupancy. This is only possible because the occupancy grid is also based on past data. In this simulation the vehicle was following a vector field, represented by

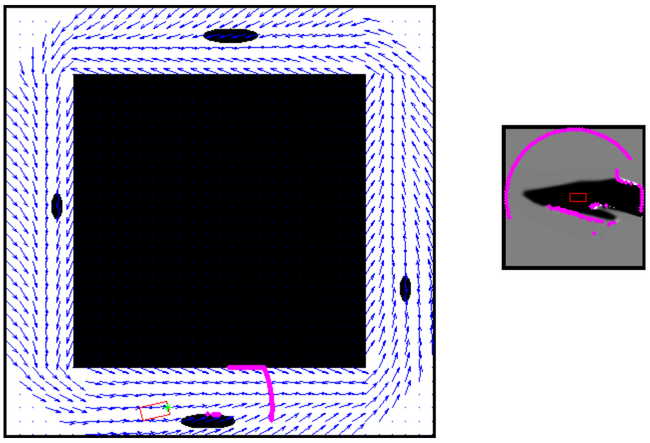


Figure 4: Simulation that shows an autonomous car equipped with a narrow field-of-view (FOV) sensor navigating around obstacles. On the left it is shown the map overlayed by the vector field that guides the vehicle (blue arrows). Still in this figure, the dark regions represent obstacles and the pink dots represent the current sensory data. On the right is the local occupancy grid for the current car position. In this figure, dark regions represent free workspace (null probability of obstacles) and white regions represent obstacles with maximum probability. The pink dots are still the sensory data but now, current and past data are plotted in order to show the increased FOV provided by the occupancy grid.

the blue arrows in the left hand side of Figure 4. This navigation strategy will be described in the next section.

4 NAVIGATION

This section aims to present the navigation control blocks presented in Figure 2. We basically adopt a hybrid arquitecture where a deliberative, global motion planning strategy based on vector fields is constantly validated by a local reactive strategy based on the dynamic window approach. These two steps are presented next.

4.1 Global motion planning

The deliberative strategy used in our solution may be any global vector field based methodology. Vector field methodologies are those where a robot, by means of a function or a table, is able to compute, for each free configuration in its configuration space, a vector, that in general may be interpreted as a force or velocity vector acting on the robot. Among the known vector field methodologies are the standard Potential Function approach proposed by Khatib (1986) and its evolution called Navigation Function (Rimon and Koditschek, 1992), which is a potential function without local minima. In these approaches the vector field is computed as the descended gradient of a function. Several important properties of most vector field based approaches are robustness to small localization errors, since neighboring vectors are generally very similar, and the possibility of local replanning by slightly modifying a given vector to avoid unmodeled obstacles.

Although any vector field based approach could be applied in the solution presented in this paper, the one used in the experiments presented in Section 5 was proposed by the author's group in (Pereira et al., 2009). The main advantages of this methodology over the standard ones are: (i) it can be easily computed for any polygonal workspace, being this space convex or not; (ii) only the regions of the space in which the vehicle is able to go, such as streets and parking lots, may be considered in the computation; (iii) it includes information about the terrain, adapting the vehicle speed depending of the region been traversed; and (iv) it can be modified to make the vehicle perform loops in a closed path, what is useful for some applications and demonstrations (Pereira et al., 2008).

Basically, the methodology proposed by Pereira et al. (2009) has three steps: (i) the workspace is discretized with a set of triangles; (ii) a sequence of consecutive triangles is selected using a graph search algorithm, composing a corridor; and (iii) a continuous vector field is computed inside the corridor. The first two steps constitute a higher level planner responsible for computing the main route of the vehicle. The basic idea is that this planner select corridors where the car could drive safely except for the presence of some unmodeled obstacles, such as other cars and pedestrians. These steps are, generally, computed off-line. However, if for some reason the map has changed and the planned corridor is completely obstructed, step (ii) could be re-executed to find a new sequence of triangles, and thus a new corridor. In the present work it is assumed that this situation will never happen and, therefore, steps (i) and (ii) are only executed once, before the movement of the car begins.

The third step, the computation of the field itself, is executed on-line given the vehicle position (x, y). This is done by the simple interpolation of three base vectors located at the vertices of each triangle of the corridor as:

$$\mathbf{u}(x,y) = \frac{A_i\mathbf{u}_i + A_j\mathbf{u}_j + A_k\mathbf{u}_k}{A_i + A_j + A_k}\,, \tag{3}$$

where A_i, A_j, and A_k are the areas of the sub-triangles formed between the original vertices of the triangles and the vehicle position (x, y), represented by the black dot in Figure 5, and $\mathbf{u}_i$, $\mathbf{u}_j$, and $\mathbf{u}_k$ are the vertices' base vectors. The vector $\mathbf{u}(x,y) = (v_x, v_y)$ correspond to the velocity vector that must be followed by the vehicle. The computation of this vector for all valid positions compose the vector field. Notice that, since two adjacent triangles share two base vectors (the ones at the common edge), the field is continuous between two triangles.

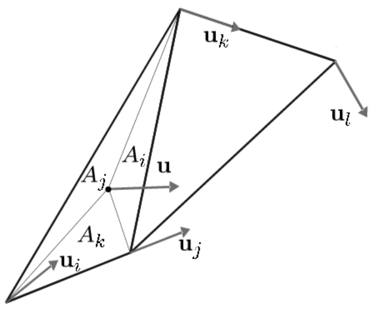


Figure 5: The convex combination of the base vectors of each triangle is used to compute a vector field inside the triangle. The small black dot represent the position of the vehicle and the arrow is the velocity vector related to this position using Equation (3). Since two consecutive triangles share two vectors, the vector field is continuous inside the corridor.

The choice of the base vectors $\mathbf{u}_i$, $\mathbf{u}_j$, and $\mathbf{u}_k$ is critical to establish a resultant field that will keep the vehicle inside the corridor and moving in the correct direction. As pointed out in (Pereira et al., 2009) any vector that has both negative projection on the outward normal vectors of the corridor boundaries and positive projections on the outward normal vectors of the intersection edge between two consecutive triangles could be chosen. When both constraints cannot be satisfied simultaneously new triangles must be created. Details of this methodology can be found in (Pereira et al., 2009). It is important to notice that the computed vector field is provable correct in the sense that it guarantees that, provided that there is no actuation or localization errors, the car will never leave the corridor and will keep moving in the correct direction. However, in case of such errors, to prevent the car to leave the corridor, an external vector field that points inside the corridor must be considered. The main issue related to this is the discontinuity in the field at the borders of the corridor, which may cause large accelerations to the vehicle.

The blue arrows in Figure 4 represent an example of a continuous vector field computed in the vehicle workspace using this technique. In this figure, the field computation did not considered the small oval obstacles. This obstacles, along with the vehicle dimensions will be considered only by the local reactive strategy presented in the next section.

Since the computation of vector (v_x, v_y) depends on the vehicle position, in a practical point of view, an estimate of (x, y) is necessary. In the experiments presented in this paper, a localization system based on an Extended Kalman Filter (EKF) was used to combine information from GPS (Global Positioning System), IMU (Inercial Measurement Unit), wheel velocity sensors and steering angle sensor to produce a good estimate of the vehicle's position and orientation. Details of this system can be found in (Santos, 2009).

Another practical issue is related to the fact that the velocity commands $(v_x,\, v_y)$ cannot be directly applied to control the non-holonomic car. In this case, the solution adopted was to convert $(v_x,\, v_y)$ into the pair of actuation velocities $(v_{1VF},\, v_{2VF})$, composed by the vehicles' linear and angular velocities, using a static feedback linearization technique (FBL) (Luca et al., 1998). This is based on the inversion of the car model in order to find its inputs in function of its outputs.

A natural output choice for the car is the the origin of its its referential $\{\mathcal{R}\}$. In this way, the model to be used by the FBL would be composed by the two first lines of model (1) as:

$$\dot{z} = \begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}\cos\theta\cos\phi & 0\\ \sin\theta\cos\phi & 0\end{bmatrix}\begin{bmatrix}v_{1VF}\\ v_{2VF}\end{bmatrix} = A(\theta,\phi)\begin{bmatrix}v_{1VF}\\ v_{2VF}\end{bmatrix},$$

where z is the output vector. Since matrix $A(\theta,\phi)$ is singular, the model is not invertible and the FBL fails to solve the input-output linearization problem. A way to circumvent this problem is to redefine the system output to be the point $P = (x_P, y_P)$ in Figure 3. This point moves together with the main axis of Ackerman's virtual wheel and is located at distance Δ_P from the midpoint of the vehicle's front axle. It is given by:

$$z = \begin{bmatrix}x_P\\ y_P\end{bmatrix} = \begin{bmatrix}x + l\cos(\theta) + \Delta_P\cos(\theta+\phi)\\ y + l\sin(\theta) + \Delta_P\sin(\theta+\phi)\end{bmatrix}. \quad (4)$$

By differentiating Equation (4) with respect to time, the vehicle model may be written as:

$$\dot{z} = \begin{bmatrix}\cos(\theta)\cos(\phi) - \sin(\theta)\sin(\phi) - \dfrac{\Delta_P\sin(\theta+\phi)\sin(\phi)}{l}\cdots\\ \sin(\theta)\cos(\phi) + \cos(\theta)\sin(\phi) + \dfrac{\Delta_P\cos(\theta+\phi)\sin(\phi)}{l}\cdots\end{bmatrix}$$
$$\begin{matrix}\cdots - \Delta_P\sin(\theta+\phi)\\ \cdots\quad \Delta_P\cos(\theta+\phi)\end{matrix}\Bigg]\begin{bmatrix}v_{1VF}\\ v_{2VF}\end{bmatrix} = A(\theta,\phi)\begin{bmatrix}v_{1VF}\\ v_{2VF}\end{bmatrix}.$$

Since matrix $A(\theta,\phi)$ is now invertible, the actuation velocities of the car, $(v_{1VF},\, v_{2VF})$, may be computed from the velocity vector $(v_x,\, v_y)$ as:

$$\begin{bmatrix}v_{1VF}\\ v_{2VF}\end{bmatrix} = A^{-1}(\theta,\phi)\begin{bmatrix}v_x\\ v_y\end{bmatrix}.$$

Next section shows how $(v_{1VF},\, v_{2VF})$ are locally modified to deal with practical considerations related to vehicle and the environment, such as unmodeled obstacles.

4.2 Reactive control

The vector field methodology presented in the previous section does not guarantee safeness to the car movement once it does not consider the workspace dynamics and the vehicle dimensions. These characteristics are taken into account in the reactive control step presented in this section.

By following the solution presented in Figure 2, it can be noticed that the velocities $(v_{1VF},\, v_{2VF})$ are not directly applied to the vehicle. Actually, they are validated and modified by a Dynamic Window (DW) (Fox et al., 1997) that relies on the local occupancy grid presented in Section 3 and on the current state of the vehicle.

The dynamic window approach used in this work is similar to the one proposed by Rebai et al. (2007). The authors define a two dimensional window with coordinates given by the linear velocity v_1 and the angular velocity ω, similarly to the original proposition of the method (Fox et al., 1997). Points in this window may be used to compute the steering angle ϕ of the car considering the following relation:

$$\dot{\theta} = \omega = v_1\frac{\sin\phi}{l}. \quad (5)$$

Alternatively, in the present work we directly define a window with coordinates v_1 and ϕ. This simplifies the process of combining the global navigation strategy with the DWA once v_2, one of the actuation velocities given by the global strategy, is the time derivative of ϕ. This makes the computation of ϕ a simple integration.

Therefore, for each time interval $\triangle t$ a two-dimensional window, called Dynamic Window (DW), with all reachable points $(v_1,\, \phi)$ is built. Basically, considering the current state of the vehicle, the DW is defined as the set of points V_{DW} limited by the maximum linear velocity and the maximum steering angle in the next time interval:

$$V_{DW} = \{(v_1,\phi)|v_1 \in [v_{1a} - \dot{v}_{1max}\triangle t, v_{1a} + \dot{v}_{1max}\triangle t],$$
$$\phi \in [\phi_a - v_{2max}\triangle t, \phi_a + v_{2max}\triangle t]\}, \quad (6)$$

where $\dot{v}_{1max}$ is the maximum linear acceleration of the vehicle and v_{2max} is its maximum steering velocity.

Among the points in the DW, some are valid and some are invalid in the sense of collision to obstacles. In this way, invalid points are those pairs $(v_1,\, \phi)$ that would cause a collision. For computing those points, it is used the occupancy grid presented in Section 3 and a function $dist(v_1,\phi)$. In this work, a function $dist(v_1,\phi)$ was constructed using the collision detection methodology proposed by Arras et al. (2002), which is applied to polygonal robots describing circular paths. Basically, notice that each vertex of the polygon that represents the vehicle (in this work a rectangle) draws a circumference in the workspace when it moves with v_1 and ϕ. Collisions are detected by verifying the intersection between these circumferences and the obstacles in the local occupancy grid. If no collisions are detected, function $dist(v_1,\phi)$ returns the smallest of the four distances between the circumferences and the obstacles. Besides function $dist(v_1, \phi)$, if the breaking acceleration of the vehicle is also considered, it is possible to verify whether it can stop before a collision and thus determine a valid set of points as:

$$V_{OBS} = \{(v_1, \phi) | v_1 \cos(\phi) \leq \sqrt{2 \cdot dist(v_1, \phi) \cdot \dot{v}_{1b}}, \\ v_1 \frac{\sin\phi}{l} \leq \sqrt{2 \cdot dist(v_1, \phi) \cdot \dot{v}_{1b} \frac{\sin\phi}{l}} \} , \quad (7)$$

where $\dot{v}_{1b}$ is the maximum breaking acceleration of the vehicle. The idea behind Equation (7) is to use Torricelli's equation to determine the limits of (v_1, ϕ) which guarantee that is still possible to break the car before a collision with the closest obstacle. Notice that the Torricelli's equation is used both for the car's linear $(v_1 \cos(\phi))$ and angular $(v_1 \frac{\sin\phi}{l})$ velocities.

The complete DW, which considers the actual speed of the vehicle, its accelerations, the obstacles in the workspace, and also the physical limits of the vehicle is then computed as:

$$\tilde{V}_{DW} = V_{DW} \cap V_{OBS} \cap V_{CAR} , \quad (8)$$

where V_{CAR} is the set of points that satisfy the maximum speed and the maximum steering angle of the car.

Figure 6 is an example of a DW. It was computed for the local occupancy grid in Figure 4 and have considered that the maximum speed of the car is about 2, 78 m/s (10 Km/h) and its maximum steering angle is 29 degrees. The external box of Figure 4 represents these limits. The small box is the set V_{DW} computed around the current pair (v_{1c}, ϕ_c) and the darker regions are those points that are invalid because may cause collisions to obstacles. Notice that $\tilde{V}_{DW}$, in this case, is computed by removing those invalid points from V_{DW}.

Once the DW is constructed, the velocities came from the vector field (v_{1VF}, v_{2VF}) are transformed into a pair v_1 e ϕ, which is then validated with the DW. If the pair is valid, (v_{1VF}, v_{2VF}) is directly used to control the vehicle. On the other hand, if the pair is invalid, a valid pair is chosen by maximizing the objective function in Equation (9). This function represents the weighted sum of three other functions: the orientation of the vehicle in relation to the field $(vf1)$; the clearance of the path $(dist)$; and the vehicle linear velocity $(velocity)$. In this equation, α, β and γ are the weights of each function.

$$G(v_1, \phi) = \alpha \cdot vf1(v_1, \phi) + \beta \cdot dist(v_1, \phi) \\ + \gamma \cdot velocity(v_1, \phi) . \quad (9)$$

The adequate choice of α, β and γ in Equation (9) makes the vehicle to follow the vector field as close as possible with a safer distance from the obstacles and at high speeds.

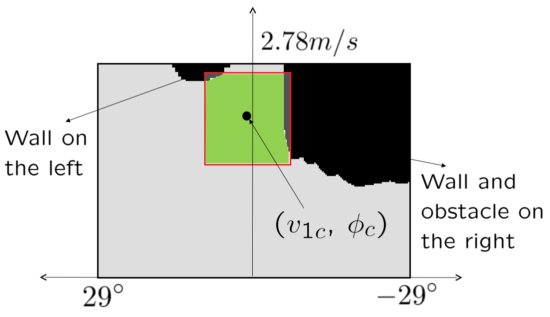


Figure 6: Dynamic window for the simulation in Figure 4. The external box is the set V_{CAR} that satisfies the vehicle constraints, the small box is the set V_{DW} of reachable points in the next time step, and the darker regions are invalid points that may cause collisions with the obstacles. The pair (v_{1c}, ϕ_c) represents the current linear velocity and steering angle of the vehicle.

Given the optimal values, (v_{1otm}, ϕ_{otm}), obtained from the maximization of $G(v_1, \phi)$, they are converted into the vehicle inputs (v_{1DWA}, v_{2DWA}) by:

$$\begin{cases} v_{1DWA} = v_{1otm} , \\ v_{2DWA} = \dfrac{\phi_{otm} - \phi_c}{\triangle t} . \end{cases} \quad (10)$$

The linear velocity v_{1DWA} and the steering velocity v_{2DWA} are setpoints for low-level controllers that actually actuate the car. They are discussed next.

4.3 Low-level control

The vehicle must be equipped with low level controllers that will guarantee that the linear velocity v_{1DWA} and the steering velocity v_{2DWA} will be followed. Regarding the linear velocity v_{1DWA}, the car used in the experiments section of this paper is equipped with a fuzzy system based controller empirically adjusted to imitate the human behavior (Freitas and Pereira, 2010). This controller actuates on the car using a linear motor that directly press the break pedal and an electronic system that simulates the potentiometer of the accelerator pedal using PWM signals. It also measures the car speed using encoders installed on the wheels. The sensor and the actuators are sampled/controlled by microcontrollers that communicate with a standard laptop using the USB port. In this specific implementation, the computer communicates with the microcontrollers in a frequency of 10 Hz, what determines its minimum sampling time. Since the time constant of the vehicle is approximately 5 seconds (Dias et al., 2012), this sampling time is sufficient to control the car, even at high speeds. However, it limits the frequency of the higher level controllers, that must be executed at lower frequencies to avoid coupling between the loops.

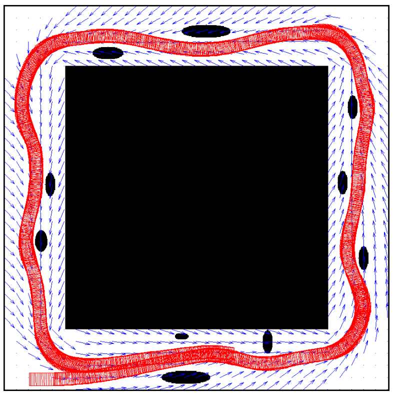


Figure 7: Simulation with 10 unmodeled obstacles (black elipses). The vector field (blue arrows) is overlaid by the vehicle path (sequence of red rectangles). The car starts at the bottom left of the figure and moves in the counterclockwise direction.

Regarding the steering velocity v_{2DWA}, it is used a commercial system by Maxon Motors, which is composed by a DC motor with encoder and a power driver. The motor is mechanically connected to the steering wheel using toothed wheels and a roller chain. The motor driver has an internal PID controller that receives velocity setpoints from a computer using serial communication. For keeping the time compatibility with the rest of the system, the communication with the motor's driver was chosen to be at 10 Hz using a USB/Serial converter. However, in this case, the PID controller that runs in the firmware of the power driver has as sampling rate of 1 kHz, what guarantees that the 10 Hz setpoint will be followed if the gains of the controller are properly tunned.

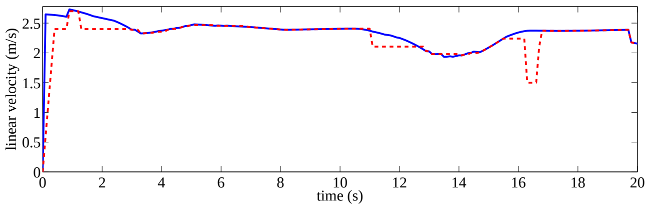
Next section will present experimental results that illustrates the proposed methodology.

5 EXPERIMENTAL RESULTS

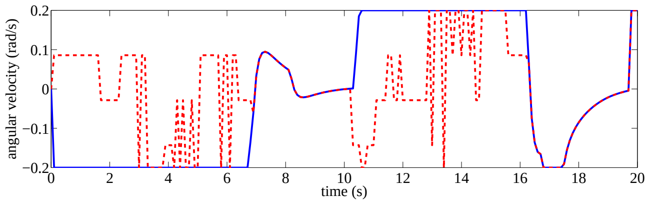
The methodology presented in the previous section was implemented both in simulation and in an actual vehicle. The results obtained are shown next.

5.1 Simulations

A simulated environment similar to the one in Figure 4 were created with the objective of evaluate the proposed methodology and adjust their parameters prior to the implementation in the actual vehicle. In this environment the simulated model is exactly the one in Equation (1) with $l = 2.61\,\text{m}$. For the FBL controller, the distance Δ_P was set to 0.5 m. In all simulations of these subsection, the vehicle's maximum speed and steering angle are the ones in Figure 6, i.e. 10 km/h and $\pm 29^o$, respectively.



(a)

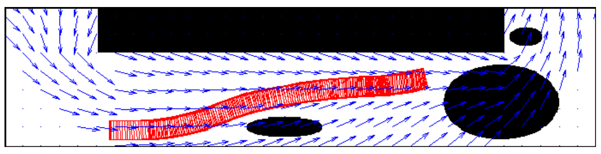


(b)

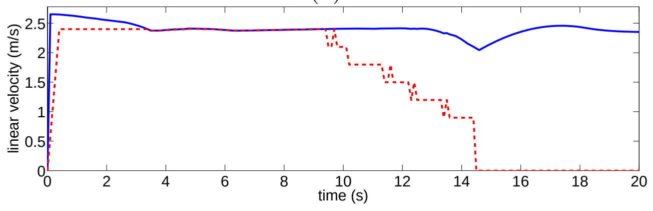
Figure 8: Velocity profiles related to simulation in Figure 7: (a) linear velocity given by the vector field and the FBL controller – v_{1VF} (solid blue line) and the one modified by the DWA – v_{1DWA} (dashed red line); (b) steering velocities: v_{2VF} (solid blue line) and v_{2DWA} (dashed red line).

Figure 7 shows the path of the simulated car in environment with 10 unmodeled obstacles. With this simulation the parameters α, β and γ of the DWA were empirically set to 0.04, 0.2 and 0.4, respectively. These values were also used during the actual experiment of the next subsection. Figure 8 shows linear and steering velocities for the first 20 s of the simulation. In this figures it is possible to see how the output of the global motion planning composed by the vector field and the feedback linearization controller (solid line) is modified by the DWA (dashed line) to make the car to avoid the unmodeled obstacles. At this point, it is important to say that the velocities given by the DWA changes in steps because the dynamic window was implemented as a discrete, low resolution map (grid) (see (Lima, 2010) for implementation details).

Another simulation is shown in Figure 9. In this simulation, a larger unmodeled obstacle was introduced in the vehicle's path in a way that it is impossible to the car to avoid it. In Figure 9(a) it is possible to see that the vehicle moves towards a narrow passage until it finally stops in front of the obstacle. Notice in Figure 9(b) that the global motion planning is not aware of the presence of the large obstacle, and therefore, the vector field imposes an almost constant velocity to the car (solid line). On the other hand, the reactive control implemented by the DWA was able to detect the obstacle and, gradually, reduce the vehicle speed until zero (dashed line). In situations like this, the vehicle will remain stopped in front of the obstacle until the obstacle is removed. A practical solution is to execute a higher level planner that would select a new corridor and recompute the vector field.



(a)



(b)

Figure 9: Simulation where the path of the vehicle is blocked with an unmodeled obstacle (larger ellipse): (a) the vector field overlaid by the vehicle path (sequence of red rectangles). (b) linear velocity given by the vector field and the FBL controller – v_{1VF} (solid blue line) and the one modified by the DWA – v_{1DWA} (dashed red line).

5.2 Real world implementation

The methodology was also implemented in the autonomous car under development at UFMG shown in Figure 1. For the experiments presented here, the computational configuration used is shown in Figure 10. It is composed of:

- A stereo vision camera Bumblebee2 connected to an IEEE1394 (Firewire) hub used solely as a camera's power supply;
- A 1.83 GHz Intel Pentium Core II Duo with 4G bytes of RAM running Windows Vista and all algorithms for obstacle detection based on the stereo camera information;
- A 1.66 GHz Intel Pentium Core II Duo with 2G bytes of RAM memory running Windows Vista. This computer receives obstacle information via Ethernet and implements all others tasks of Figure 2. It also actuate the car via USB ports.

With this hardware configuration, the software responsible for the low level controllers executed at 10 Hz and the other tasks are executed at 5 Hz. During the experiments, the linear velocity of the car was limited to 25 Km/h by the DWA. This value was chosen due to the limited field of view and resolution of the stereo camera, that is not able to detect obstacles farther then 17 meters. With the maximum speed of 25 Km/h and a conservative breaking acceleration of about 2 m/s^2, it is guaranteed that the car would completely stops before colliding with an obstacle at this distance.

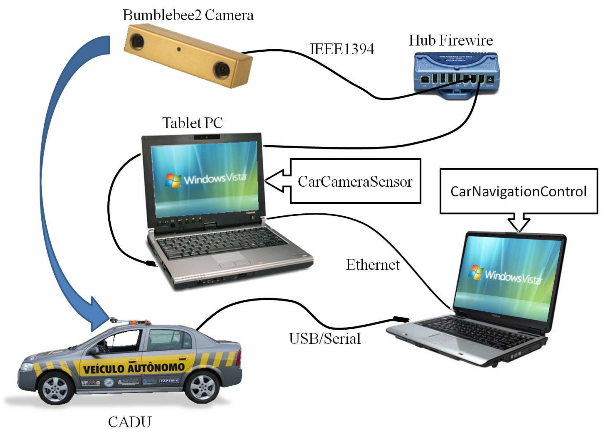
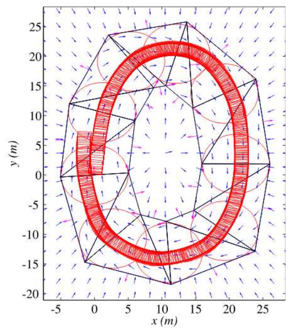
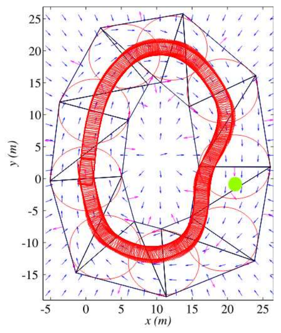


Figure 10: Computational configuration adopted on the CADU experiments. *CarCameraSensor* is the application responsible by the obstacle detection and *CarNavigationControl* is responsible for all other tasks.



(a)



(b)

Figure 11: Path followed by the vehicle in two situations: (a) withot and (b) with an obstacle. The movement starts at position $x = y = 0$ and occurs in the clockwise direction. The vehicle pose along the movement is represented by the red rectangles. In (b) the obstacle position is represented in green.

For the experiment presented here, a vector field were created inside a "ellipsoidal" corridor. The car followed the vector field in two situations. In the first one, whose path is shown in Figure 11(a), there were no unmodeled obstacles inside the field. In the second one, an obstacle was added to the corridor. Notice in Figure 11(b) that the vehicle reacted to the presence of this obstacle, represented by the small yellow circle, by modifying its path but still keeping itself inside the corridor. Notice that the vehicle is not trying to come back to its original path after avoiding the obstacle. This is a characteristic of this specific vector field, which is not designed to make the car track a specific trajectory, but only to keep it moving inside the corridor.

Figure 12 shows the images from the stereo camera in the moment that the vehicle detected the obstacle (a

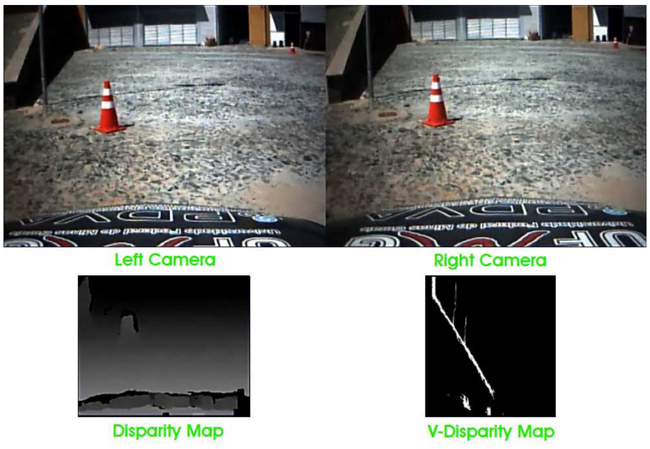


Figure 12: Vehicle perception: The stereo images are shown on the top, and their respective Disparity and V-Disparity Map are on the bottom.

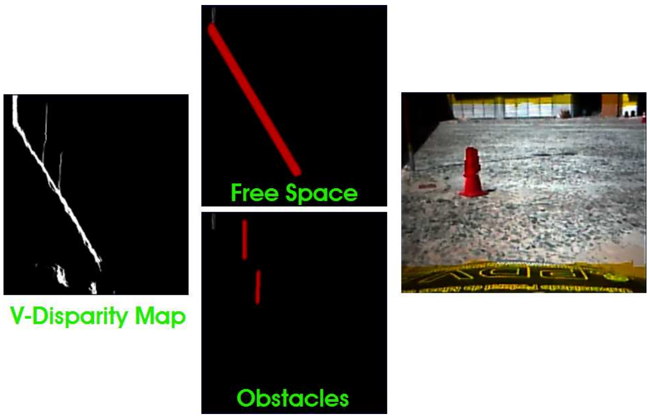


Figure 13: Obstacle detection: The V-Disparity Map, on the left, is segmented into obstacles and free space. On the right, the detected obstacles are marked in one of the original images. In this image, closer obstacles are shown in red.

stripped cone). In this figure it is also possible to see the disparity map obtained from these images and it respective V-Disparity Map. In the V-Disparity map, obstacles are detected as vertical lines as can be see in Figure 13 that shows the obstacle segmentation. The detected obstacle is used to compose the local occupancy grid.

Videos of the demos presented here and other experiments with the autonomous car are available in the following web address: `http://coro.cpdee.ufmg.br/`.

6 CONCLUSIONS AND FUTURE WORK

This paper presented a solution for safe navigation of autonomous cars. The solution is based on the integration of a Vector Field based methodology with the Dynamic Window Approach (DWA). The Dynamic Window Approach relies on an occupancy grid that, in this work, was constructed using information from a stereo camera. The solution was successfully implemented and tested in a full size autonomous car.

In relation to the vector field, it is important to mention that different methodologies could be used. Notice that the vehicle non-holonomic constraints and its dimensions were not taken into account by the field, allowing for a large variety of methodologies. The field presented in this paper and used in the experiments was chosen because it is easy to be computed for large workspaces and may considers the properties of the terrain to determine the vehicle speed. However, it has a serious problem that is the discontinuity of the field in the borders of the corridor where it is computed, as can be observed in Figure 11. If the vehicle crosses the border (what may happen in the case of localization errors or when an obstacle is obstructing the corridor), it may be subject to large accelerations that will try to make it come back the corridor. To avoid this, totally continuous velocity fields, such as the one by Gonçalves et al. (2010), could be used to follow closed paths.

Another important observation is that, although in this work a stereo camera was the only sensor used to avoid obstacles, several other sensors could be easily incorporated into the methodology, once the DWA uses only intermediate information from the occupancy grid. Therefore, one of the next steps of our work is to introduce laser range finders to improve obstacle detection. With the stereo camera, it was observed that some small obstacles were not detected due to the poor resolution of the cameras. This resolution also limits the vehicle speed, as observed before. With the addition of laser sensors, the authors believe that the autonomous car would be able to navigate safely in low traffic roads, such as the desert roads crossed by the competitors of the first DARPA Grand Challenge (DARPA, 2005). For more complex environments, it is important to add more constraints into the DW and to improve the decision making process that uses this window. This will allow, for example, the vehicle to decide between avoid an obstacle or stop in front of it depending of the obstacle or even on the traffic laws.

Still related to sensing and obstacle avoidance, notice that the occupancy grid methodology and the DWA assume that all obstacles in the environment are static. Therefore, movable and fixed obstacles are treated in the same way. In the occupancy grid, this may be a problem once movable obstacles could create the false indicative that an obstacle is larger than it really is. On the other hand, if the trajectory of the movable obstacle is not considered in the DWA, the decision making process may lead the vehicle to collide with this obstacle. The solutions for these problems depend on a semantic interpretation of the vehicles' environment and must be considered as a future work.

The autonomous car used in this paper, after the conclusions of the experiments presented here, received a completely new hardware and software architecture (Arruda, 2012). This is based on a set of four computers that uses industrial networks to communicate with sensors and actuators. The new solution solved most of the practical problems that were originated from the multiple USB devices connected to a single computer in the old architecture. However, since the new system is based on Linux computers and the current navigation solutions were build using Windows, the authors future work include the portability of the code to the new system. It includes the use of the Robot Operating System (ROS) (Quigley et al., 2009) in order to make the vehicle easy to be programed by an end user.

ACKNOWLEDGMENTS

This project was financed by Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG). During the realization of the project, Danilo A. Lima held a master's scholarship from CAPES. Guilherme A. S. Pereira holds a scholarship from CNPq. This project would not be possible without the hard work of several students and professors from UFMG that spent part of their time making CADU an autonomous vehicle. In this way, the authors sincerely thank Elias Freitas, Érica Campedelli, Frederico Vória, Guilherme Becker, Guilherme Castro, Hélio Guerra Júnior, Jullierme Dias, Leandro Batista, Leonardo Torres, Luciano Pimenta, Marco Túlio Pires, Marlon Vieira, Matheus Vinti, Maurício Baleeiro, Michelle Santos, Paulo Iscold, Pedro Cotta, Tiago Arruda, Tiago Mendonça, and Vitor Sabbagh. Finally, the authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

REFERENCES

Arras, K., Persson, J., Tomatis, N. and Siegwart, R. (2002). Real-time obstacle avoidance for polygonal robots with a reduced dynamic window, *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 3, Washington, DC, USA, pp. 3050–3055.

Arruda, T. A. (2012). *Arquitetura de hardware e software para supervisão e controle de um carro autônomo*, Master's thesis, Universidade Federal de Minas Gerais. Available in portuguese at `http://coro.cpdee.ufmg.br`.

Braid, D., Broggi, A. and Schmiedel, G. (2006). The TerraMax autonomous vehicle, *Journal of Field Robotics* **23**(9): 693–708.

Brock, O. and Khatib, O. (1999). High-speed navigation using the global dynamic window approach, *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, Michigan, USA, pp. 341–346.

Broggi, A., Caraffi, C., Fedriga, R. I. and Grisleri, P. (2005). Obstacle detection with stereo vision for off-road vehicle navigation, *Proceedings of the International IEEE Workshop on Machine Vision for Intelligent Vehicles*, San Diego, CA, USA, pp. 1–8.

Dias, J. E. A., Pereira, G. A. S. and Palhares, R. M. (2012). Identificação do modelo dinâmico longitudinal de um carro autônomo, *Anais do Congresso Brasileiro de Automática*, Campina Grande, PB.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation, *Computer* **22**(6): 46–57.

Fernandes, L. C., Dias, M. A., Osório, F. and Wolf, D. F. (2010). A driving assistance system for navigation in urban environments, *in* A. Kuri-Morales and G. Simari (eds), *Advances in Artificial Intelligence - IBERAMIA 2010*, Vol. 6433 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 542–551.

Fox, D., Burgard, W. and Thrun, S. (1997). The dynamic window approach to collision avoidance, *IEEE Robotics and Automation Magazine* **4**: 23–33.

Freitas, E. J. R. and Pereira, G. A. S. (2010). *Controle longitudinal de um veículo autônomo*, Trabalho de Conclusão de Curso, Escola de Engenharia da Universidade Federal de Minas Gerais. Available in portuguese at `http://coro.cpdee.ufmg.br`.

Freitas, E. J. R., Vinti, M. N. W., Santos, M. M., Iscold, P., Tôrres, L. A. B. and Pereira, G. A. S. (2009). Desenvolvimento de automação embarcada para um robô móvel baseado em um carro de passeio, *Anais do Simpósio Brasileiro de Automação Inteligente*, Brasilia, DF, pp. 1–6.

Gonçalves, V. M., Pimenta, L. C. A., Maia, C. A. and Pereira, G. A. S. (2010). Navegação de robôs móveis utilizando curvas implícitas, *Controle & Automação* **21**(1): 43–57.

Honório, L. M., Vermaas, L. L. G., Gonçalves, L. M. and Vidigal, M. (2010). Uma metodologia para aprendizado supervisionado aplicada em veículos inteligentes, *Anais do XVIII Congresso Brasileiro de Automática*, Bonito, MS, pp. 1028–1035.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots, *The International Journal of Robotics Research* **5**(1): 90–98.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J., Langer, D., Pink, O., Pratt, V., Sokolsky, M., Stanek, G., Stavens, D., Teichman, A., Werling, M. and Thrun, S. (2011). Towards fully autonomous driving: Systems and algorithms, *Proceedings of the IEEE Intelligent Vehicles Symposium*, Alcalá de Henares, Spain, pp. 163–168.

Lima, D. A. (2010). *Navegação segura de um carro autônomo utilizando campos vetoriais e o método da janela dinâmica*, Master's thesis, Universidade Federal de Minas Gerais. Available in portuguese at `http://coro.cpdee.ufmg.br`.

Lima, D. A. and Pereira, G. A. S. (2010). Um sistema de visão estéreo para navegação de um carro autônomo em ambientes com obstáculos, *Anais do XVIII Congresso Brasileiro de Automática*, Bonito, MS, pp. 224–231.

Lima, D. A. and Pereira, G. A. S. (2011). Navegação segura de um carro autônomo utilizando campos vetorias e o método da janela dinâmica, *Anais do X Simpósio Brasileiro de Automação Inteligente*, São João del Rei, MG, pp. 1167–1172.

Luca, A. D., Oriolo, G. and Samson, C. (1998). *Robot Motion Planning and Control*, Vol. 229, Springer Berlin / Heidelberg, chapter Feedback control of a nonholonomic car-like robot, pp. 171–253.

McBride, J. R., Ivan, J. C., Rhode, D. S., Rupp, J. D., Rupp, M. Y., Higgins, J. D., Turner, D. D. and Eustice, R. M. (2008). A perspective on emerging automotive safety applications, derived from lessons learned through participation in the darpa grand challenges, *Journal of Field Robotics* **25**: 808–840.

Megda, P., Esteves, B. and Becker, M. (2011). Determining forbidden steering directions for a passenger car in urban environments based on the velocity obstacle approach and use of trackers, *Proceedings of the IX IEEE Latin American Robotics Symposium*, Bogota, Colombia., pp. 1 –6.

Milanes, V., Llorca, D., Vinagre, B., Gonzalez, C. and Sotelo, M. (2010). Clavile: Evolution of an autonomous car, *Proceedings of the13th International IEEE Conference on Intelligent Transportation Systems*, Funchal, Portugal, pp. 1129–1134.

Mirisola, L., Azevedo, H., Ramos, J. J. G., Bueno, S., Paiva, E. and Azinheira, J. (2011). Validação experimental de um veículo robótico terrestre para ambientes externos, *Anais do X Simpósio Brasileiro de Automação Inteligente*, São João del Rei, MG, pp. 1322–1327.

Nothdurft, T., Hecker, P., Ohl, S., Saust, F., Maurer, M., Reschka, A. and Bohmer, J. (2011). Stadtpilot: First fully autonomous test drives in urban traffic, *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems*, Washington, DC, USA, pp. 919–924.

Pereira, G. A. S., Pimenta, L. C. A., Chaimowicz, L., Fonseca, A. R., de Almeida, D. S. C., de Q. Corrêa, L., Mesquita, R. C. and Campos, M. F. M. (2009). Robot navigation in multi-terrain outdoor environments, *The International Journal of Robotics Research* **28**(6): 685–700.

Pereira, G. A. S., Rebelo, D. R., Iscold, P. and Torres, L. A. B. (2008). A vector field approach to guide small UAVs through a sequence of waypoints, *Anais do XVII Congresso Brasileiro de Automática*, Juiz de Fora, MG, pp. 1–6.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T. B., Leibs, J., Wheeler, R. and Ng, A. Y. (2009). ROS: an open-source robot operating system, *Proceedings of the International Conference on Robotics and Automation: workshop on Open-Source Software*, Anchorage, Alaska.

Rebai, K., Azouaoui, O., Benmami, M. and Larabi, A. (2007). Car-like robot navigation at high speed, *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, Sanya, P. R. China, pp. 2053–2057.

Rimon, E. and Koditschek, D. (1992). Exact robot navigation using artificial potential functions, *IEEE Transactions on Robotics and Automation* **8**(5): 501–518.

Santos, M. M. (2009). *Desenvolvimento de um sistema de localização para um veículo terrestre*, Master's thesis, Universidade Federal de Minas Gerais. Available in portuguese at `http://coro.cpdee.ufmg.br`.

DARPA (2005). DARPA Grand Challenge web site, accessed July 6, 2012, `http://archive.darpa.mil/grandchallenge05/`.

Thrun et al. (2006). Stanley: The robot that won the DARPA Grand Challenge, *Journal of Field Robotics* **23**(9): 661–692.

Vermaas, L. L. G., de Melo Honório, L., de Jesus, E. O., Freire, M. and Barbosa, D. A. (2009). Intelligent vehicle survey and applications, *in* G. L. Torres, J. M. Abe, J. I. da Silva Filho and H. G. Martins (eds), *Advances in technological applications of logical and intelligent systems*, IOS Press, pp. 205–235.