DVFF NAVIGATION: A COMBINED APPROACH TO REAL TIME MOBILE ROBOT NAVIGATION WITHOUT PRIOR KNOWLEDGE OF THE ENVIRONMENT

A. Oualid DJEKOUNE, Karim ACHOUR
*Robotics and Industrial Automation, Advanced Technologies Development Centre.
Lotissement 20 Août 1956, BP17 Baba Hassen, Algiers, Algeria. Tel: 213-021-35-10-40, Fax: 213-021-35-10-39. Email: odjekoune@cdta.dz

Redouane TOUMI
Electronics Department, Engineering Faculty, U.S.T.H.B., Algeria.
BP. 32 Bab Ezzouar, 16111, Alger, Algeria

Abstract: *Often autonomous mobile robots operate in environment for which prior maps are incomplete or inaccurate. They require a safe execution for a collision free motion to a goal position. This paper addresses a sensory based navigation method for a mobile robot that moves in unknown environments. Thus, a navigation method called DVFF combining the Virtual Force Field (VFF) obstacle avoidance approach and global path planning based on D* algorithm is proposed. While D* generates global path direction towards a goal position, the VFF local controller generates the admissible trajectories that ensure safe robot motion. Results and analysis from a battery of experiments with this method implemented on an ATRV2 mobile robot are shown.*

Key words: *Autonomous mobile robot, global navigation, VFF algorithm, reactive navigation, obstacle avoidance, D* algorithm.*

1. Introduction.

Some navigation systems integrate the local and global navigation systems in the sense that each one works independently, they interact to perform a complete navigation system: the global system pre-plan a global path and incrementally search new paths when discrepancy with the map occurs; the local system uses onboard sensors to detect and avoid unpredictable obstacles. The mobile robot executes an algorithm which permits to follow the optimal global path by tracking its geometric points from a start position to the goal. If an obstacle obstructs this path, the robot executes another algorithm (collision avoidance algorithm) allowing it to move around the perimeter until the nearest point of the obstacle to the geometric path point is found, or pre-plan another optimal global path to reach the goal.

This paper describes a sensory based navigation approach called *DVFF* for the *ATRV2* mobile robot. It combines two navigation methods, the global path planning algorithm *D** and the Virtual Force Field (*VFF*) algorithm proposed in [1] to produce an efficient collision-free navigation algorithm. The primary motivation behind this work is to develop a robust compete navigation algorithm which has the advantages of the two above methods, and at the same time, eliminates several of their drawbacks. The *D** algorithm is used to compute the global path direction which allows the mobile robot to head for the destination. The global path directions are the backpointer directions. The optimal path from any position in the environment can be determined simply by following the direction of the global path to reach the goal. The used *D** algorithm is applied in a known environment based on a modified Histogramic In Motion Mapping (*HIMM*) algorithm and updated using the robot's on-board sensors data, it takes into account the robot dimensions to reduce both the difficulty of the updating process and the computation time (the size of the obstacles are not enlarged). The *VFF* is used for the safety mobile robot motion and considered as a local controller algorithm to generate an admissible trajectory if new discrepancy obstacles are detected.

2. Prior work.

Nearly all researches in a complete navigation approach combine a global path planning module with a local obstacle avoidance module to perform navigation. While the global path planner determines a suitable path based on a map of the environment, the obstacle avoidance algorithm determines a suitable direction of motion based on recent sensor data. Obstacle avoidance is performed locally in order to ensure that real-time constraints are satisfied.

For example the Vector Field Histogram* algorithm (*VFH**) developed in [2], combines the *VFH+* [3] as local obstacle avoidance algorithm with the *A** search algorithm as global path planner. It is considered as a local obstacle avoidance algorithm that uses look-ahead verification to consider more than the robot's immediate surroundings. While *VFH** has the same obstacle avoidance performance as *VFH+* for regular obstacles, it is capable of dealing with problematic situations that would require the robot to substantially slow down or even stop [2]. The *VFH+* is a real-time local obstacle avoidance algorithm that looks for gaps in locally constructed polar histograms.

In the same way, several well known local obstacle avoidance algorithms such as Dynamic Window (*DW*) and Nearness Diagram (*ND*) are combined with a global path planner to perform navigation and to become respectively Global Dynamic Window (*GDW*) and Global Nearness Diagram (*GND*) complete navigation systems. They are combined with the global, local minima-free navigation function NF1 [4] using a wave propagation technique starting at the goal. However, these algorithms are unable to ensure the security of a mobile robot evolving in a dynamic environment. They use global reasoning essentially to overcome the shortcomings of the used local obstacle avoidance algorithm.

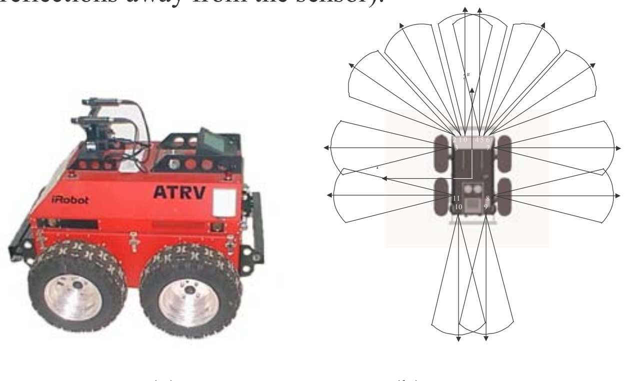
As used in [5], a complete navigation system was developed which integrates the *A** algorithm and the *DW* algorithm and then adapted for the use of focused *D** algorithm [6]. It is performed by a single path alignment measure based on geometric comparison between possible robot trajectories and the effective path. The effective path is determined according to the detection of a reference point on the global geometric path where the path direction starts changing significantly by observing path direction change points along the path. The length and orientation of the effective path directly determines optimal reference velocity vector in the next sampling instant that is a combined objective of obstacle clearance and path alignment. This navigation system is applied in a partially known dynamic indoor environment where the kinematic and dynamic robot constraints are taken into account using a C-space.

In [1] a real time obstacle avoidance based only on the *VFF* approach has been presented and tested on a real *ATRV2* mobile robot. However, based on several experiments, some shortcomings that are inherent to the concept of potential fields are discovered. The mobile robot was gotten trapped in local minima when it entered a dead end. In the literature, various efforts have been made to overcome the local minima problem. There are those which establish a new potential functions with a few or even no local minima, those which use certain techniques to escape from local minima [7], and those which use the global recovery. In [1], if the mobile robot has a global knowledge of the goal position, it would be easy for it to go out from the local minima. This global knowledge can be obtained using a global path planner algorithm.

3. Scene map building algorithm.

In order to create a scene map from ultrasonic range measurements, the environment must be scanned at first. The mobile robot *ATRV2* (Fig. 1.a) is equipped with 12 ultrasonic sensors that are mounted on a horizontal ring around the robot (six on the front, two on the back and two on both sides). Their measuring range spans distances from approximately 5 centimeters to 4 meters. The main lobe of the sensitivity function is contained within a solid angle Ω of 30° (Fig. 1.b).

With the robot's ultrasonic sensors, an approximate full 360° panorama can be acquired rapidly. However, all sensors are fired at once; unfortunately, this causes significant crosstalk (caused by low angular resolution and errors due to multiple reflections or specular reflections away from the sensor).



(a) (b)

Fig. 1. (a) *ATRV2* mobile robot, (b) the sensing coverage around the *ATRV2*.

3.1 Histogram grid for obstacles representation.

As spatial representation, the *HIMM* approach presented in [8] is modified in this work to improve the perception of the mobile robot. It uses a two dimensional Cartesian histogram grid for obstacle representation. Each cell in the histogram grid holds a *Certainty Value* "*CV*" that represents the confidence of the algorithm in the existence of an obstacle at that location. In [8] the *CVs* are increased or decreased by sensor data until predefined maximum or minimum values are reached for cells which only lie on the acoustic axis of the ultrasonic sensor. This algorithm does not take into account the angular error and supposes that the detected object is closer to the acoustic axis of the sensor than the periphery of the view conical field.

However, while considering that without complementary information on ultrasonic measure, all points of the cone's periphery are equivalent for a possible position of the obstacle; It appears to us that it is necessary to assign the same value to each cell representing the same state (free or occupied).

Our method uses this idea by decreasing by 1 the *CV* of cells corresponding to the free space until a predefined minimum value is reached, and increasing by 1 the *CV* of cells of the cone's periphery representing the obstacle until a predefined maximum value is reached. The *CVs* of the remaining cells are not modified. This idea has the advantage of avoiding the measurements treatment using probabilities laws which are heavy to manage.

The update rule is expressed according the Fig.2, as follows:

$$CV(X) = \begin{cases} CV(X)+1 & if\ X \in B \\ CV(X)-1 & if\ X \in A \\ CV(X) & if\ X \in C \end{cases} \quad (1)$$
$$CV_{Min} \le CV(X) \le CV_{Max}$$

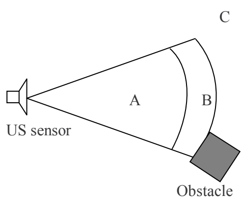


Fig. 2. The sonar data updating rule.

3.2 The mapping algorithm.

Initially, the histogram grid world model or the world map is set to 0 value. The *CV* cells values change if the obstacle can move and preserve their values otherwise (walls or static objects). When the robot moves, each detected obstacle increases the value of the corresponding cell. Progressively, while the robot moves, the obstacles are identified by cells whose values increase.

A local map is associated to the robot. Its center is the mobile robot gravity center. Its size is adjusted to the largest detected ultrasonic measure. When the robot moves, it uses its position, orientation, and the on-board sensor data to build a continuously-updated local map. The current data set is processed within the current local map using the update rule described in equation (1).While the robot moves, all local maps are integrated into one global map. Each local map is integrated with the previously built global map to get a new global map.

4. The D* algorithm for global path direction processing

The path planning problem of a mobile robot is to find a safe and an efficient path, given a start position, a goal position and a map of the workspace. The robot can go from the start position to the goal without colliding with any obstacle along the path.

Generally, a robot does not have complete map information. As a result, any path generated using its initial map may turn out to be invalid or suboptimal as it receives updated map information through its onboard sensors. It is thus important that the robot is able to update its map and replan optimal paths when new information arrives [9].

A number of algorithms exist for performing this replanning [10, 11]. Currently, *A** and *D** are the most widely used of these algorithms, due to their efficient use of heuristics and incremental updates. These algorithms guarantee optimal paths over grid-based representations of a robot's environment [9]. The *D** search algorithm is a dynamic version of *A**. It produces an optimal path from the start position to the goal by minimizing a predefined cost function. It has the capability of rapid replanning, and has been used in real time planning in partially known environment with challenging terrains. It gives also an estimate of the path cost to the goal from any state of the graph and a back pointer to one of its neighbors indicating the geometric direction to the goal.

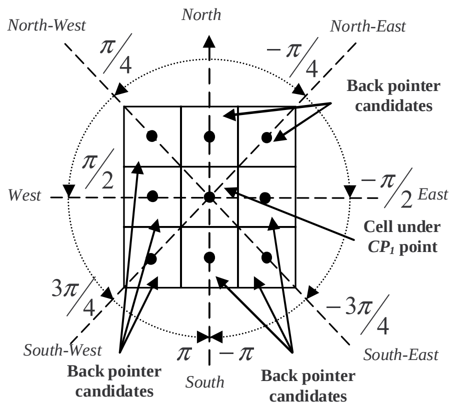


Fig. 3. Computing the global path direction according to the back pointer position.

In this work, *D** approach is used because; it allows replanning to occur incrementally and optimally in real time and gives the global path direction from any position in the environment towards the goal [10]. Applied on a map represented by two dimensional Cartesian histogram grid, some attributes are added to each cell of the grid which are *back pointer (b)*, *arc cost (c)*, *tag (t)*, *path cost (h)*, *key (k)*, in addition to the certainty values used in the *HIMM* algorithm (see section 3).

The *walkable* (*wc*) or *traversable* attribute is added for cells occupied by the detected obstacles. This attribute is used by the developed *D** algorithm to ensure its fast convergence, and to reduce the update processing time.

The C-space is not used, the mobile robot is not considered as a point, and the obstacles sizes are not enlarged by a safety margin in this work. The cells of the grid surrounding these obstacles (according to the robot dimensions) are not used by the developed D^* algorithm.

Each cell of the map except those surrounding obstacles includes an estimate of the path cost to the goal, and a back pointer to one of its neighbors indicating the geometric direction to the goal (north, south, east, west, north-west, south-west, north-east and south-east) called in our case the global path direction. Figure 3 shows how to calculate the global path direction according to the back pointer position. This direction is very important for a mobile robot moving in the environment. From any position in the map, the mobile robot can have the direction to use to head for the goal. Figure 4 shows the global path direction results obtained from simulated obstacles with a given start and goal positions. Following these directions, the mobile robot can reach the goal without carrying out a path tracking algorithm. That permits the robot saving time wasted in the path tracking operation.

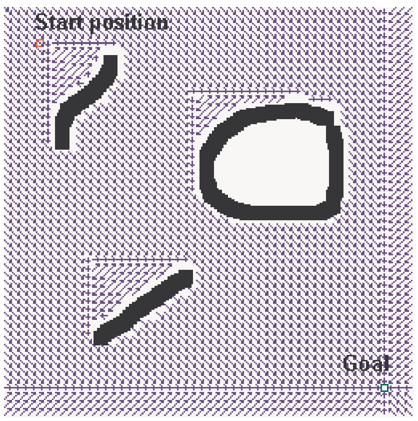


Fig. 4. Global path direction from any position in the environment.

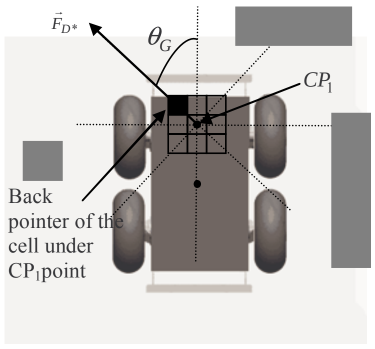


Fig. 5. Computing the global steering direction.

When the mobile robot moves, it uses the back pointer of the cell under the point *CP₁*, determines its direction θ_G (global path direction) and applies a fictitious vector $\vec{F}_{D^*}$ in this direction (Fig.5). This direction is used to calculate the motion command that generates a collision free motion while driving the robot towards the goal.

CP₁ is a point located on the longitudinal axis of the robot; its optimal location differs for different mobile robots [12].

The cost of traversing from cell *Y* to *X* is a positive number given by the *arc cost* function $c(X,Y)$ by the following equation:

$$c(X,Y)=\begin{cases}10 \text{ if } Y \text{ is horizontal or vertical to } X\\ 14 \text{ if } Y \text{ is on the diagonal to } X\\ X \text{ and } Y \text{ are neigbors}\end{cases} \quad (2)$$

For each cell *N*, the *D** algorithm maintains an estimate of the sum of the arc costs from *X* to the goal cell given by the *path cost* function *h*:

$$h(N)=|x_N-x_G|+|y_N-y_G| \quad (3)$$

where (x_N,y_N) and (x_G,y_G) are the coordinates of the cell *N* and the goal cell respectively.

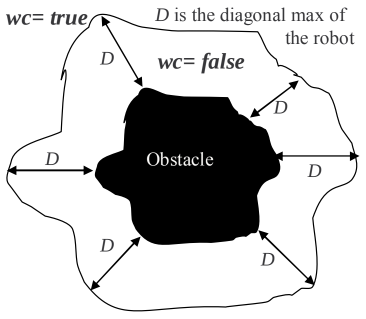


Fig. 6. The *wc* value according to the mobile robot dimensions.

Description of the algorithm

The implemented *D** algorithm is the same as presented in [10], but varies in that, it does not take into account the cells which surround obstacles. They are recognized by the value of their attribute *wc*. *wc* is a boolean variable that takes the value false if the cell is near from an obstacle according to the robot dimensions, and true otherwise (Fig.6).

The used D^* algorithm consists of two functions: *Init_D** and *Case_D** presented below (the processes I, II and III are the same as presented in [10]).

Init_D* is used only at the beginning of the algorithm to compute the initial optimal path from the start position to the goal (*G*), and Case_*D** is used when a new obstacle is detected by the mobile robot (arc cost function of a state is changed) to compute the new optimal path from the robot position to the goal. These two functions use a new function called W*alcableCase* that gives for a given cell the value of its *wc* attribute.

```
bool  WalkableCase(X).
L1   $wc = true$.
L2    for each neighbor $Y$ of $X$ :
L3      if ($Y$ is not walkable or
L4         $Y$'s $CV \succ 0$ or
           $Y$ is near from the neirest obstacle
             according to the robot dimensions)
               $wc = false$.
L5    return($wc$).

Init_D*.
L1   Do {
L2     $X = MinState()$.
L3    if ($X == Null$)  break.
L4    $k_{old} = GetKmin()$.
L5    Delete($X$).
L6    $wc = WalkableCase(X)$.
L7    if ($wc$) {
L8       if ( $k_{old} \prec h(X)$ )
L9          ProcessI($X, k_{old}$).
L10       else if ( $k_{old} == h(X)$ )
L11              ProcessII($X, k_{old}$).
L12       else  ProcessIII($X, k_{old}$).
L13         }
L14     $k_p = GetKmin()$.
L15    }while($k_p != -1$).
```

Case_D* **(X_{Rob}).**

```
L1   The tag value of all states is set to New.
L2   The backpointer of all states is set to −1.
L3   The goal G in the OpenList.
L4   Do {
L5     $X = MinState()$.
L6     if (($X == Null$) or ($X == X_{Rob}$)) break.
L7     $k_{old} = GetKmin()$.
L8     $Delete(X)$.
L9     $wc = WalkableCase(X)$.
L10    if ($wc$) {
L11      if ( $k_{old} \prec h(X)$ )
L12        $ProcessI(X, k_{old})$.
L13      else if ( $k_{old} == h(X)$ )
L14        $ProcessII(X, k_{old})$.
L15      else $ProcessIII(X, k_{old})$.
L16      }
L17    $k_p = GetKmin()$.
L18  }while($k_p != -1$).
```

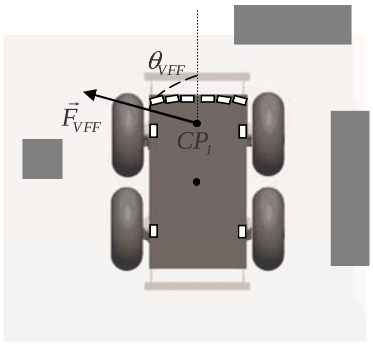


Fig. 7. Computing the local repulsive force $\vec{F}_{VFF}$.

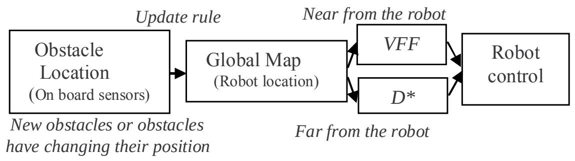

Update rule
Near from the robot
Obstacle Location (On board sensors)
Global Map (Robot location)
VFF
D*
Robot control
New obstacles or obstacles have changing their position
Far from the robot

Fig. 8. Diagram of the *DVFF* navigation system.

5. *VFF* approach for real time obstacle avoidance.

Real time obstacle avoidance presents the problem of navigating around known or unknown objects in a dynamic environment [13]. The obstacle avoidance algorithm used in this work is the same as the one developed in [1]. It uses the *VFF* method to calculate both the frontal and the lateral corrective measures. These measures are used to calculate the local steering direction to protect the robot from collisions with the detected obstacles.

Figure 7 shows the force $\vec{F}_{VFF}$ applied at the point CP_1 on the mobile robot which represents the corrected repulsive force pushing the mobile robot away according to the direction θ_{VFF} from the frontal and lateral detected obstacles.

6. The *DVFF* navigation algorithm.

By combining the global path planning *D** and the Virtual Force Field (*VFF*) algorithms, we propose a navigation system called *DVFF* that drives the mobile robot robustly among locations. The structure of this navigation system is shown in Fig. 8. The mobile robot motion generation process has two principal steps:

- *VFF* module: Operates when new discrepancy obstacles are detected near to the mobile robot. It generates a local steering direction θ_{VFF} that protects the mobile robot from collision with the obstacles.
- *D** module: Computes the global path direction θ_G which permits to the mobile robot to head for the destination.

The *D** module can be easily used with the *VFF* module because both of them generate a direction. These directions can be used separately or combined to give one direction. The selected direction will be used to calculate the motion command that generates a collision free motion while simultaneously driving the robot towards the goal

In [14] the authors show that when a mobile robot uses the resultant of these two directions, it will be able to navigate without collisions with the obstacles. The experimental tests with a real *ATRV2* mobile robot have proved to be successful except if the robot is in a local minimum. The authors use two coefficients for each direction. If they increase only the value of the coefficient used for the global direction, the mobile robot reaches the goal but avoids badly the obstacles. And if they increase only the value of the coefficient used for the local direction, the mobile robot avoids well the obstacles but the trajectory is not optimal. The choice of these coefficients isn't simple because they must be chosen experimentally [14].

To overcome this shortcoming, the mobile robot will use separately these two directions. The mobile robot uses the global path direction from the *D** module if it exists (the mobile robot is far from the obstacles, see Fig. 8). This direction is used to calculate the motion command to move the robot towards the goal. When the mobile robot is near either to a frontal or to a lateral detected obstacle, the *VFF* module calculates a local direction to generate a collision free motion to move the robot around the obstacles. Each of these two directions is used to calculate the motion command that generates a collision free motion for the next sampling period *T*, while simultaneously driving the robot towards the goal. Additionally, the mobile robot uses its onboard sensors to update the global map of the environment for calculating the new global path direction from its actual position towards the goal if the detected obstacles are new.

The following algorithm illustrates in details how the mobile robot combines local and global directions in a complete navigation way to permit a safe navigation from an initial position to a goal position.

Combinaison of global and local direction algorithm :

```
L1   Compute the initial optimal path using the Init_D* function.
L2   The mobile robot starts to follow the global path direction θ_G.
L3   While the mobile robot moves and while the goal is not reached, it :
L4         Reads the odometry data.
L5         Reads the US data.
L6         If new obstacles are detected, the mobile robot uses Case_D*.
L7         Reads the new global path direction θ_G under its CP_1 point.
L8         If θ_G exists, the mobile robot uses this direction to move.
L9         Else, it calculates the local direction θ_VFF to move.
L10  If the target is reached, the mobile robot stops.
```

7. Motion control.

The mobile robot *ATRV2* is a four wheeled differential drive skid steering configuration. Its kinematics model can be expressed by the following equations:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} w \qquad (4)$$

where $(\dot{x}, \dot{y})$ is the Cartesian velocity, and (v, w) the linear and angular velocity. The velocities v and w are expressed as follow:

$$\begin{cases} v = \dfrac{v_R + v_L}{2} \\ w = \dfrac{v_R - v_L}{2d} \end{cases} \qquad (5)$$

where v_R, v_L are the linear velocities of the right and left wheels and d the width of the mobile robot..

If both wheels are driving forward with the same speed, the robot moves forward. If they are driving in opposite directions, the robot will hovers around itself, but if they are driving forward with different speeds, the robot will follow a curve on the side moving with the lower speed. In this case, the mobile robot describes an arc of circle of radius R around the Instantaneous Center of Rotation (ICR) (Fig.9). R can be written as follow:

$$R = d\,\frac{v_R + v_L}{v_R - v_L} \qquad (6)$$

When the desired steering direction θ_{Dev} (θ_{VFF} or θ_G) obtained from the navigation algorithm detailed above and the mobile robot orientation θ_{Rob} are known, the steering direction θ_c given to the robot is processed as follows:

$$\Delta\theta = \theta_{Dev} - \theta_{Rob} \qquad (7)$$

$$\begin{aligned} &if\ (-\pi \le \Delta\theta \le \pi)\quad \theta_c = \Delta\theta. \\ &else\ \ if\ (\Delta\theta \prec -\pi)\quad \theta_c = 2\pi + \Delta\theta. \\ &\quad else\ \ if\ (\Delta\theta \succ \pi)\quad \theta_c = \Delta\theta - 2\pi. \end{aligned} \qquad (8)$$

The desired respective linear and angular velocities v_d and w_d, can then be computed as follows according to the steering direction θ_c:

$$if\ \left(-\pi/180 \le \theta_c \le \pi/180 \right) \begin{cases} move\ \ straight\ \ on. \\ v_d = v_c. \\ w_d = 0. \end{cases} \qquad (9)$$

$$else\ \ if\ \left(\pi/180 \le \theta_c \prec \pi/2 \right) \begin{cases} turn\ \ on\ \ the\ \ left. \\ R = D/tg\,\theta_c\,. \\ v_d = \dfrac{v_c}{2}\left(1 + \dfrac{R-d}{R+d}\right). \\ w_d = \dfrac{v_c}{2d}\left(1 - \dfrac{R-d}{R+d}\right). \end{cases} \qquad (10)$$

$$else\ \ if\ \left(\theta_c = \pi/2 \right) \begin{cases} turn\ \ on\ \ the\ \ left. \\ v_d = 0. \\ w_d = \dfrac{v_c}{d}. \end{cases} \qquad (11)$$

$$else\ \ if\ \left(\pi/2 \le \theta_c \le 175\pi/180 \right) \begin{cases} turn\ \ on\ \ the\ \ left\ \ then \\ straight\ \ on \\ v_d = 0. \\ w_d = \dfrac{v_c}{d}. \end{cases} \qquad (12)$$

$$else\ \ if\ \left(\theta_c = -\pi/2 \right) \begin{cases} turn\ \ on\ \ the\ \ right. \\ v_d = 0. \\ w_d = -\dfrac{v_c}{d}. \end{cases} \qquad (13)$$

$$else\ \ if\ \left(-\pi/2 \prec \theta_c \le -\pi/180 \right) \begin{cases} turn\ \ on\ \ the\ \ right. \\ R = D/tg\,\theta_c \\ v_d = \dfrac{v_c}{2}\left(1 + \dfrac{R+d}{R-d}\right). \\ w_d = \dfrac{v_c}{2d}\left(\dfrac{R+d}{R-d} - 1\right). \end{cases} \qquad (14)$$

$$else\ \ if\ \left(-175\pi/180 \le \theta_c \le -\pi/2 \right) \begin{cases} turn\ \ on\ \ the\ \ right\ \ then \\ straight\ \ on \\ v_d = 0. \\ w_d = -\dfrac{v_c}{d} \end{cases} \qquad (15)$$

$$else\ \ Blocking\ \ Situation. \qquad (16)$$

where the constant v_c is set experimentally.

8. Experimental results.

The complete mobile robot navigation algorithm called *DVFF* presented in this paper has been implemented and tested on the *ATRV2* mobile robot manufactured by *iRobot* (Fig.1.a). Four trials runs that illustrate different navigations of the *ATRV2* mobile robot through obstacle courses are examined in this section.

To test the performance limits of the developed *DVFF* navigation method, the environment and the obstacle locations are completely unknown and carried out in our laboratory. Both of the start position and the goal position are given in advance to the robot. The translation velocity is limited to $v_c = 20\, cm/s$ and the rotational velocity is limited to $0{,}6\, rd/s$. The experiments are shown in Fig. 10 to Fig.13.

In each experiment, initially the mobile robot calls the function *Init_D** to compute the initial optimal path from the start position to the goal. The mobile robot uses then the global path direction of the cell under its point CP_1 if it exists to have the direction θ_G to follow until it arrives to the goal. If θ_G does not exist, in this case the mobile robot is close to a frontal or lateral obstacles, it computes then the direction θ_{VFF} of the corrected repulsive force which will allow it avoiding these obstacles.

As the mobile robot drives, it receives information through its onboard ultrasonic sensors. If an obstacle obstructs the path of the robot, the robot calls the function *Case_D** to replan a new optimal path to permit it moving around this obstacle and going towards the goal. The new information received by the sensors is used, to update the map of the environment and to process new global path direction using the function *Case_D**.

The first experiment is shown in Fig.10; the start position and the goal position are taken aligned except where there are some obstacles which cross the path in front of the robot. The reproduced histogram grid of the global map with the robot shape and its gravity center trajectory is shown in Fig.10.b.

When the mobile robot starts moving, it detects obstacles in the front and in the right. The robot makes the correct decision and turns to the left avoiding the detected obstacles successfully and reaches the goal.

The second experiment is a typical experiment to show the performance of the developed *DVFF* method when the mobile robot is trapped in local minima. In this experiment, the start position and the goal position were taken aligned too. The mobile robot starts to move on a straight line. During its displacement, it detects obstacles forming a dead end using its onboard ultrasonic sensors. It calls then the function *Case_D** and follows the new processed global path direction. Figure 11 shows the mobile robot avoiding perfectly the detected dead end.

In the third experiment, the start position and the goal position are aligned to the left of the mobile robot obstructed by an obstacles wall. The mobile robot starts to turn on the left according to the global path direction of the cell under the CP_1 point. As it moves, it detects obstacles from its frontal ultrasonic sensors. The function *Case_D** makes a correct decision by indicating the right direction to the mobile robot until it reaches the goal. Figure 12 shows the mobile robot avoiding perfectly the wall and reaches the goal represented by a small rectangle.

The last experiment is a typical experimental run using the algorithm developed in [1]. The mobile robot is in the same situation as in the second experiment. The aim of this experiment is to show the mobile robot behavior when it uses only the local information from the onboard sensors for navigation. Figure 13 shows when the mobile robot entered a dead end, it is trapped automatically in local minima.

At each point along the robot trajectory, a local map of the environment was obtained and integrated into a global map. The global map is represented by a matrix containing information of the different cells in the map. The map size is given by the size of the matrix and the area that is to be included in the map. In our case, the max size of the covered area is 10×10 meters and the size of each cell is 10×10 centimeters.

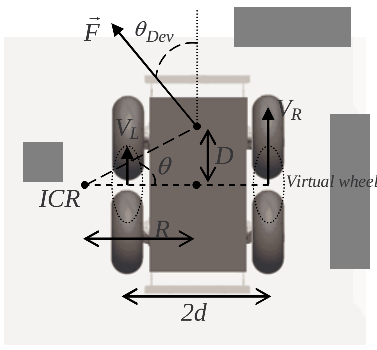


Fig. 9. The *ATRV2* kinematics model and its motion command.

9. Conclusion and future work.

Based on the idea of combining the global path planning based on *D** algorithm and a real time obstacle avoidance algorithm based on the *VFF* approach, our method called *DVFF* performs a path to reach a goal position in unknown environments.

No prior knowledge about the environment is assumed in this method. Such knowledge can be provided in form of the environment model or be acquired during motion through sensing. The map is built using a modified *HIMM* algorithm based on range data sampled only by onboard ultrasonic sensors. The updating rule does not use any probability functions and takes into account all the grids inside a sector for each sonar reading.

Satisfactory results have been obtained concerning the problem of mobile robot navigation in unknown environments but some improvements can be brought like using information from the onboard stereo pair camera which will certainly enhance the mobile robot navigation quality.

In the near future, we would like to use this new approach with developed algorithms such as those in [15,16] in outdoor environments for monitoring our center site.

References

1. Djekoune, A. O., AchourK., Toumi, R. : *Ultrasonic sensing based navigation for mobile robot with obstacles avoidance.* In: Proceedings of the IEEE International

Computer Systems & Information Technology Conference, IEEE CSIT'05, jul.2005, Algiers, Algeria, pp.193-198,

2. Borenstein, J., Ulrich, I.: *VFH*: local obstacle avoidance with look-ahead verification*. In: Proceedings of the IEEE Int. Conf. Robotics Automation, April. 2000, San Francisco, CA, pp2505-2511.

3. Borenstein, J., Ulrich, I.: *Reliable obstacle avoidance for fast mobile robots*. In: Proceedings of the IEEE Int. Conf. on Robotics and Automation, May 16–21, 1998, Leuven, Belgium, pp. 1572 – 1577.

4. Brock, O., Khatib, O.: *High speed navigation using the global dynamic window approach*. In: Proceedings of the IEEE Int. Conf. on Robotics and Automation, May 1999, Detroit, Michigan, USA, pp.341-346.

5. Macek, K., Petrovic, I., Ivanijko, E.: *An approach to motion planning of indoor mobile robots*. In: Proceedings of the IEEE International Conference on Industrial Technology, ICIT'03, 2003, Maribor, Slovenia, pp.969-973.

6. Seder, M. Macek, K., Petrovic, I.: *An integrated approach to real time mobile robot control in partially known indoor environments*. In: Proceedings of the 31st Annual Conference of the IEEE Industrial Electronics SocietyNov. 2005, Raleigh, North Carolina, USA, pp.1785-1790.

7. Xi-yong, Z., Jing, Z.:*Virtual local target method for avoiding local minimum in potential field based robot navigation*. In: Journal of Zhejiang University SCIENCE, ISSN 1009-3095, V.4 (2003), No. 3, May-June 2003, pp.264-269.

8. Borenstein, J., Koren, Y.: *Histogramic in-motion mapping for mobile robot obstacle avoidance*. In: IEEE Journal of Robotics and Automation, Vol. 7 (1991), N° 4, pp. 535-539, Sscramento. California.

9. Ferguson, D., Stentz, A.: *The Delayed D* Algorithm for efficient path replaning*. In: Proceedings of the IEEE International Conference on Robotics and Automation, April, 2005, Barcelona, Spain, pp. 2045-2050.

10. Stenz, A.:*Optimal and efficient path planning for partially-known environments*. In: Proceedings of the IEEE International Conference on Robotics and Automation, 1994, San Diego, California, pp.3310-3317.

11. Koenig, S., Likhachev, M.: *Improved fast replanning for robot navigation in unknown terrain*. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2002, Washington, DC, USA, Volume 1, pp. 968- 975.

12. Borenstein, J., Raschke, U.: *Real-time Obstacle Avoidance for Non-Point Mobile Robots*. In: Proceedings of the Fourth World Conference on Robotics Research, Sept 1991, Pittsburgh, PA, pp. 2.1–2.9.

13. Gourley, C., Trivedi, M.M.: *Sensor Based Obstacle Avoidance and Mapping for Fast Mobile Robots*. In: Proceedings of the International Conference on Robotics and Automation, ICRA'94, May 1994, San Diego, CA, USA, pp.1306-1311.

14. Djekoune, A. O., Achour, K., Toumi, R.: *A new approach for mobile robot navigation*. In: Proceedings of the Conference sur le Génie Electrique CGE'05, 16-17 avril 2007, Ecole Militaire Polytechnique, Bordj El Bahri, Algeria.

15. Djekoune, A. O., Achour, K.: *Incremental Hough Transform: An Improved Algorithm for Digital Device Implementation*. In: Real-Time Imaging, Vol. 10 (2004), Issue 6, Dec. 2004, pp. 351-363.

16. Achour, K., Djekoune, A. O.: *Localisation and guidance with an embarked camera on a mobile robot*. In: Advanced Robotics, Vol. 16 (2002), No. 1, 2002, pp. 87-102.

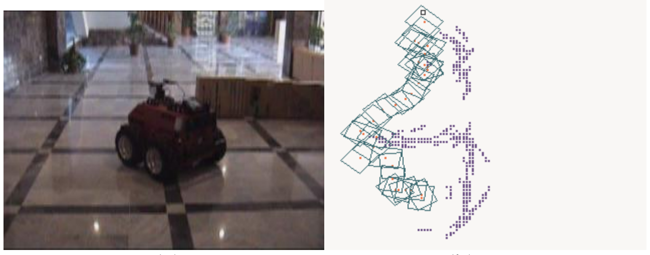

(a) (b)

Fig. 10. First experiment.

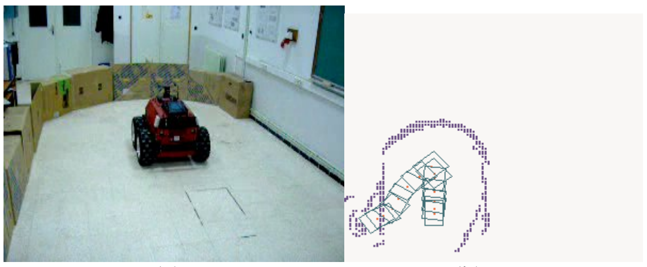

(a) (b)

Fig. 11. Second experiment.

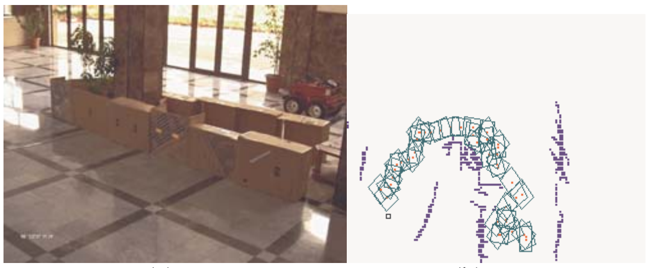

(a) (b)

Fig. 12. Third experiment.

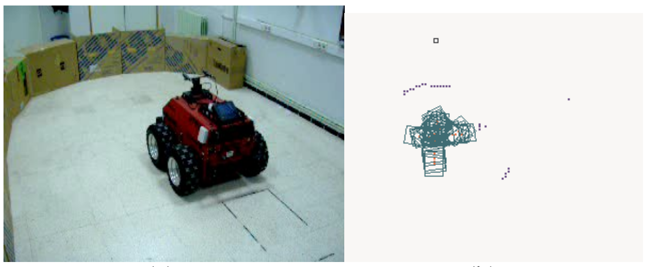

(a) (b)

Fig. 13. Fourth experiment.