Implementation of a Mobile Robot Platform Navigating in Dynamic Environment

Hadjira Belaidi[1], Hamid Bentarzi[1] and Mohamed Belaidi[2]

[1] *Signals and Systems Laboratory (SisyLab), Institute of Electrical and Electronic Engineering (IGEE), University M'hamed Bougara of Boumerdes (UMBB), 35000 Boumerdes, Algeria*
[2] *Infotronics Departement, Science Faculty, M'hamed Bougara University of Boumerdes (UMBB), Algeria*

Abstract. Currently, problems of autonomous wheeled mobile robots in unknown environments are great challenge. Obstacle avoidance and path planning are the back bone of autonomous control as it makes robot able to reach its destination without collision. Dodging obstacles in dynamic and uncertain environment is the most complex part of obstacle avoidance and path planning tasks. This work deals with the implementation of an easy approach of static and dynamic obstacles avoidance. The robot starts by executing a free optimal path loaded into its controller; then, it uses its sensors to avoid the unexpected obstacles which may occur in that path during navigation.

1 Introduction

Robotics helps human in difficult, repetitive or tedious tasks. Moreover, it is the dream of substituting the machine to man in these tasks. Currently, the faculties of robots perception and reasoning are progressing every day and more in the future, they will play an increasingly important role in our lives.

Navigation of mobile robots in dynamic environments still represents a challenge for real world applications. The robot should be able to gain its goal position navigating safely among moving people or vehicles, facing the implicit uncertainty of the surrounding world and the limits of its perception system.

The problem of autonomous navigation has been deeply studied in literature and several techniques have been developed. These techniques use probabilistic and complex algorithms which are hard to implement and requires specific sets of hardware and software [1], [2].

In this paper a technique that is easy to be implemented using low cost materials is introduced. First, the robot is supposed to follow a predefined optimal free path, generated using the method given by [3], to navigate safely in static environment. Then, a strategy enabling the robot to navigate in dynamic environment considering the uncertain obstacles is proposed. Thus sonar sensors are used for this purpose.

2 Path generation and obstacle avoidance in static environment

First, and prior to spraying the case of dynamic environment, the robot needs to generate a navigation path in a static environment. After that, this trajectory will be saved in the internal memory of a robot and run repeatedly; and at the same time avoiding unexpected obstacles (and dynamic ones).

2.1 Path planning approach

A path planning and obstacle avoidance approach using parametric curves was used for the generation of the predetermined path that the robot should follow [4]. It is assumed that the current position coordinates of the robot and of the target are known using exteroceptive sensors. The robot can measure the distance to nearby obstacles that are in the range of sensors and avoid them to achieve the goal by following an optimal smoothed path by running the algorithm proposed in [4], [5].

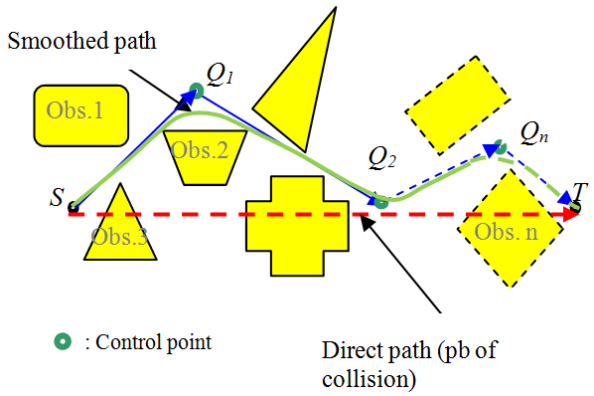

Figure 1. Path planning using parametric curves and control points insertion

In [5], a linear parametric curve is used for path planning, and then this curve is smoothed around the control point. The linear path connecting the initial

position (*S*) to the target (*T*) is considered first. If the trajectory hits an obstacle, a control point is introduced between the initial position and the target and an intermediate connection point is created as shown in Fig. 1. Then, the collision between the line (*QT*), connecting the control point *Q* to the target *T*, and the obstacle is tested. In case of collision, another control point must be added, and so on (see Fig. 1).

Non-uniform rational B-spline (NURBS) curves were used to smooth the path, as they preserve the C^2 continuity of the trajectory which is very important for the robot dynamic. Furthermore, the addition of control points to avoid obstacles does not affect the whole trajectory. The smoothed curve is shown in green in Fig. 1.

2.2 Static trajectory data extraction

There are several ways to generate a path to follow for a robot such as: Arc trajectory [4], which is easy to be generated; but the combination of obstacles avoidance with other factor changes in a dynamic environment is difficult because of the circular arc. Thus, the trajectory generation using Bezier curves is introduced [6], which is characterized by defining the orientation and direction of the robot. However, the major drawback of this method is the mathematical complexity of Bezier curves, which makes it difficult to create paths a robot can follow with high speed and the same result is presented in straight line guide where the robot must stop walking to turn.

To reduce these problems, in our work, the robot must follow the trajectory generated using NURBS. First, the following parameters must be extracted from the NURBS curve generated using Visual Basic instruction [5]:

- The tangent from the initial position, which defines the initial orientation of the robot.
- The distance (*di*) between each two control point which defines the distance that the robot has to move and which can be calculated using the following formula:

$$d_i = \sqrt{\left(Q_{(i+1)x} - Q_{ix}\right)^2 + \left(Q_{(i+1)y} - Q_{iy}\right)^2} \quad (1)$$

- The tangent at each control point *Qi* to define the orientation α_i of the robot (see Fig. 2(a)).
- The speed of the robot is defined by the derivative of the NURBS curve at each control point. Between the control points, the robot has to move with a constant speed.

Initially, NURBS is presented as a database containing the coordinates (x_i, y_i) of each control point Q_i. Next, the distances d_i and the orientations α_i are computed as explained above and as shown in Fig. 2(b). After extracting these data, they will be recorded as an Excel database (* .xls) to send them to the platform of the robot via USB (universal serial bus) port, which allows series data transfers from the PC to the robot microcontroller.

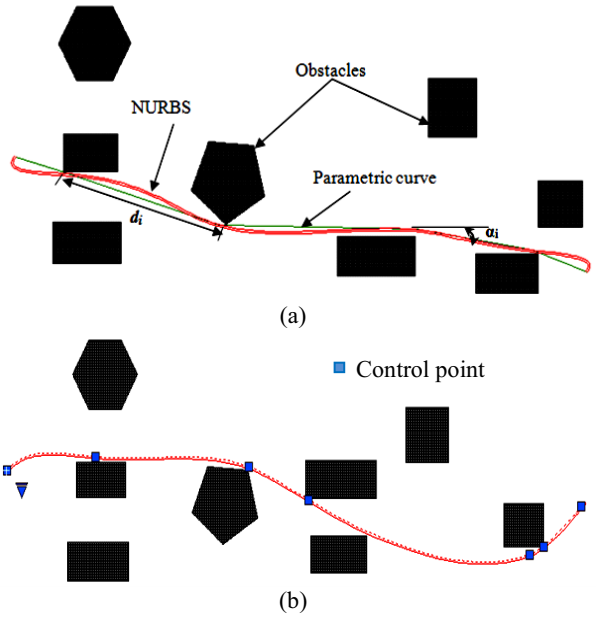

Figure 2. AutoCAD simulation results. (a) Path planning in static environment. (b) Static trajectory data extraction

Table 1 is an illustration of database containing the coordinates (x_i, y_i) of control points Q_i constituting the free path, and Table 2 illustrates the database that contains the translation of these data to distances and orientations.

Table 1. Illustration of database extraction from static free path (the control points (x, y) coordinates).

x_i(mm)	y_i(mm)
892,8212	1228,2581
1451,0873	1300,5873
2413,4103	1248,5391
4179,7141	795,9584
2791,8635	1075,9884
4270,0362	838,2637
4506,3260	1043,2394

Table 2. Translation of Q_i coordinates to distances and orientations.

d_i	α_i
1518,4688	30,252
1948,6358	0
2717,2410	-35,456
4254,8278	-45,001
2992,0315	10,983
4351,5394	20,980
4625,5078	45,032

This free path data will be executed by the robot platform. If the robot encounters an unexpected obstacle so it sends an error message to the microcontroller to notify the existence of an obstacle not point in the map of the environment.

3 Obstacles avoidance in dynamic environment

A function called "dynamic" is created to support the dynamic obstacle avoidance. This function helps the robot to navigate in an uncertain environment with four (04) ultrasonic sensors that will collect the environmental information, locate the obstacles and avoid them.
The principle of this function is as follows:

- Read the data of the front sensor;
- If there is an obstacle, the robot moves backward for a distance that allows it to avoid the obstacle without collision by a test of the rear sensor;
- Read the data of the right sensor;
- If there is no obstacle the robot turns right and reads the right sensor data until it exceeds the obstacle then it recovers the initial trajectory by the inverse performance of data travelled;
- If there is an obstacle the robot tests the left sensor, otherwise it avoids the obstacle from the left.
- If both sensors indicate that the path is not free an error message will be sent to the controller to assign that "the way is occupied";
- If a moving obstacle moves toward the robot, the robot will move backward with sending a message that contains the value # DANGER#. The flowchart in Fig. 3 summarizes the proposed approach.

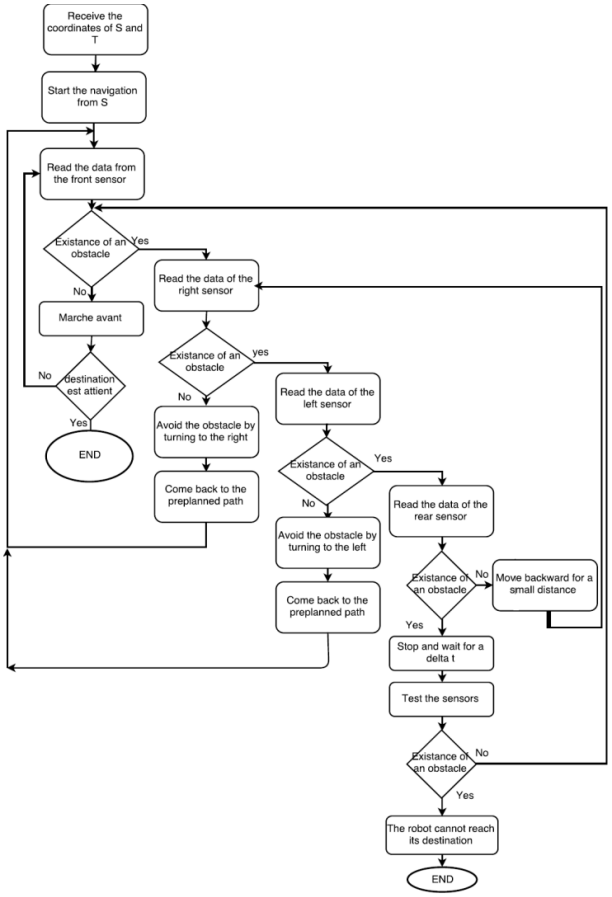

Figure 3. Flowchart simurizing the approach of obstacles aviodance in dynamic environment.

4 The robot platform

Fig. 4 shows the general electronic circuit connecting the various components constituting the robot platform. The denotation of each component is given below.

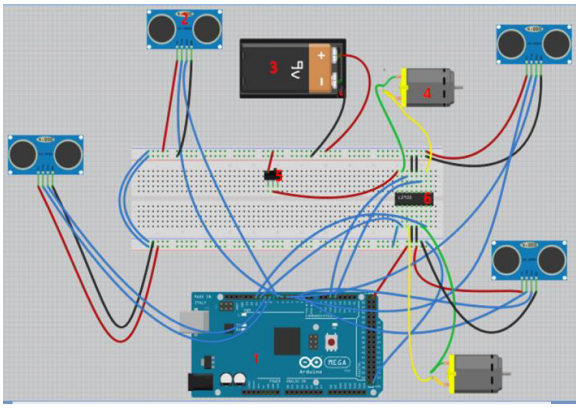

Figure 4. The robot circuit Platform. 1: Arduino MEGA2560 card, it contains the program that allows the robot to navigate in its environment; 2: Ultrasonic sensors used to calculate the distance between the robot and the obstacles; 3: 6V battery energy source; 4: DC-motors; 5: On / off switch; 6: L293D used to drive the DC motors

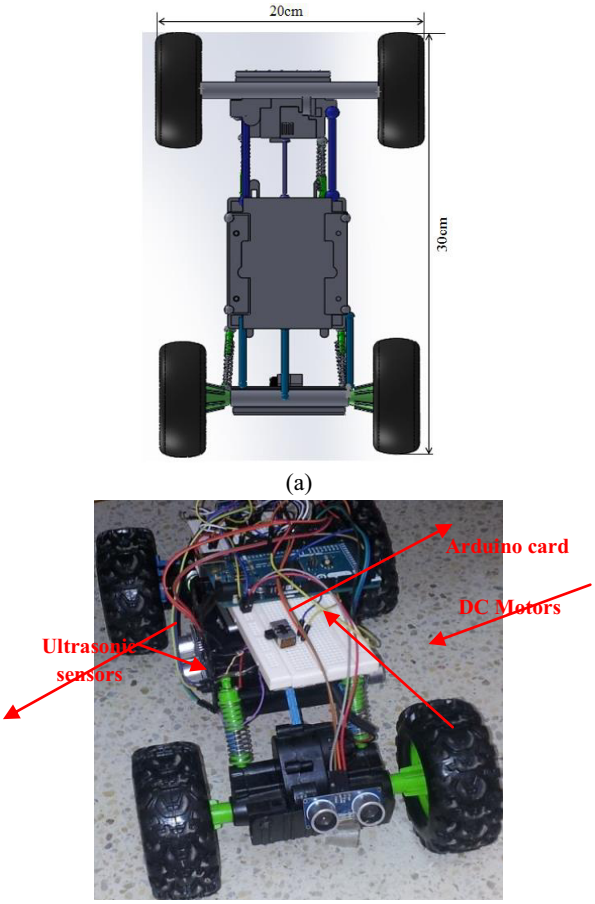

Figure 5. The used Autonomous mobile robot platform. (a) Platform geometry of the used autonomous mobile robot. (b)The image of the robot

The prototype represents an autonomous mobile robot platform which moves in three dimensional space. The platform has two rear driving wheels and two front

orientation wheels. The wheels are supplied by two independent DC motors. The geometry of the platform is depicted in Fig. 5(a). Hence an image of the robot platform is given in Fig. 5(b).

5 Experimental results

To validate our approach, it has been tested in several scenarios.

In the scenario shown in Fig. 6, the robot follows its path planned previously, and loaded into its controller; once an unexpected obstacle appears, the robot stops, go back (for a small distance which permits it to avoid the obstacle) then turns right and avoids the obstacle. If no other obstacle occurs, the robot turns left and come back to its initial path.

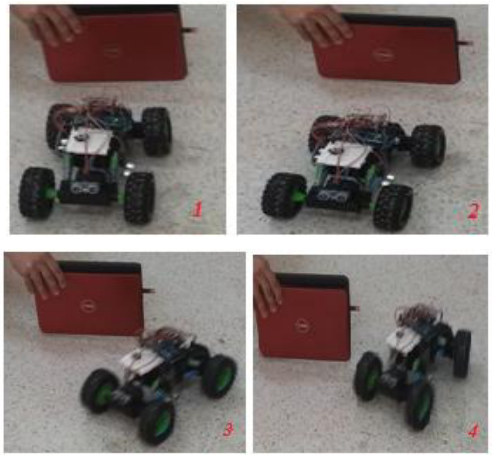

Figure 6. Obstacle avoidance scenario

Before turning, the robot tests the two side sensors. Hence, the robot avoids the obstacle by the side free of obstacles, and it will not come back to its initial path until the sensor face of the obstacle be free.

One important case to notice is when the robot avoids the obstacles to a specific side and detects another obstacle in that way. The robot automatically goes back and tries the other side. If the two sides are blocked, the robot backs off in a straight line and avoids it again.

When the robot is driven out of its path by many obstacles, it always tries to rejoin its path.

5.1 Results and discussion

This robot is equipped by ultrasonic sensors which make it able to detect transparent objects, and navigate in an indoor environment. So this robot can be used in our dignified house, offices, supermarkets etc .... For example the robot can detect glass sliding doors.

In addition, the robot can navigate in outdoor environments where the surface can be rough, because of its chassis with suspension (see Fig. 5) and the approach that allow it to navigate in uncertain environments.

However, there are some critical cases that this robot can encounter and which are not taken into consideration in our work. The following section discusses some of these cases.

5.2 Some critical cases

Our robot encounters great difficulties to detect circular objects and obstacles with a width lower than 5cm due to the characteristics of ultrasonic sensors when sending sound waves. These waves will hit the obstacle and change direction and they do not return to the sensor, so the obstacle avoidance is not very accurate in this case.

In the case 1 of Fig. 7, the distance measurement is accurate because the obstacle is perpendicular and in front of the sensor, however in case 2 it will generate an inaccurate measurement since the inclined position sensor. Among the critical cases also, the positions which generate a measuring antagonism as shown in Fig. 7 case 3.

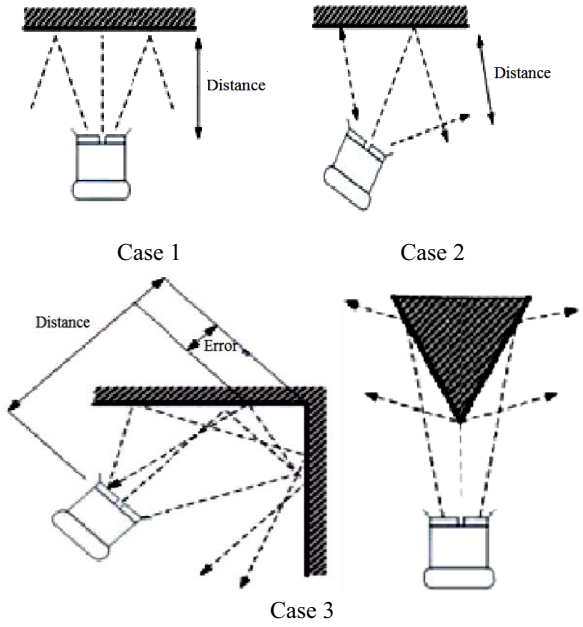

Figure 7. Some critical cases which cannot be avoided accurately by the robot

Apparently, the robot will not detect objects that have a vacuum of more than 10cm high and in the case where the object has a substantial length. This length may be a function of speed. This situation can create a robot damage (crushing) Fig. 8.

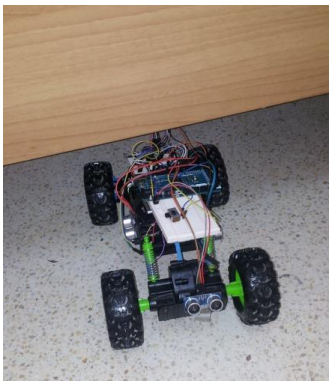

Figure 8. Objects having a vacuum of more than 10cm high.

Another drawback to notice is that high-speed obstacles cannot be avoided. The obstacle velocity should be less or equal to the robot's velocity.

6 Conclusion

In this work, the design, simulation and real time testing of an autonomous mobile robot in an unknown environment have been successfully carried out. We were able to come up with a strategy of navigation and try its effectiveness in a real time changing indoor environment.

Though some defects, the results were satisfying and the robot was able to navigate safely in dynamic environment.

Hence, in the future work, the robot's performance can be enhanced by installing a GPS module to localize the robot at any given moment. This will simplify the program and drive the robot to any goal with high precision.

Moreover, the ultrasonic sensors can be replaced by laser sensors which have a wider range to recognize the exact shape and position of obstacles. We can go further in customizing with adding up a Radar sensor to work in outdoor environment (rough environment) since the ultrasonic sensors performance is limited.

References

1. H. Cho, et al., "A Multi-Sensor Fusion System for Moving Object Detection and Tracking in Urban Driving Environments," 2014 IEEE International Conference on Robotics & Automation (ICRA), Hong Kong Convention and Exhibition Center, Hong Kong, China, (May 31-June 7, 2014)
2. C. Saranya, et al., "Path Planning Through Partially Occupied Cells using Modified D* Algorithm," ISTE-ACEEE Int. J. in Computer Science, **1**(1), (March 2014)
3. B.J. Lee, S.O. Lee and G.T. Park, "Trajectory Generation and Motion Tracking Control for the Robot Soccer Game," Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems, (1999)
4. H. Belaidi, "Optimal control and strategy of autonomous mobile robot 2D/3D simulation," Magister thesis, Electrical and Electronic Institution, UMBB, (June 2009)
5. H. Belaidi, A. Hentout, B. Bouzouia , H. Bentarzi, A. Belaidi, "NURBs Trajectory Generation and Following by an Autonomous Mobile Robot Navigating in 3D Environment," 4th Annual International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2014 IEEE, Hong Kong, (4-7 June 2014), pp. 168–173.
6. J.C. Wolf, "Fast Autonomous Mobile Robot Platform," A report submitted to the University of Plymouth In partial fulfillment for the degree in B Eng (Hons) Robotics and Automated Systems, (2003-2004)