Autonomous wheelchair navigation with real time obstacle detection using 3D sensor

Article *in* Automatika · February 2017

DOI: 10.7305/automatika.2017.02.1421

3 authors:

Emna Baklouti
University of Sfax
8 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Nader Ben Amor
Ecole Nationale d'Ingénieurs de Sfax
50 PUBLICATIONS **102** CITATIONS
SEE PROFILE

Mohamed Jallouli
University of Sfax
42 PUBLICATIONS **67** CITATIONS
SEE PROFILE

Online ISSN 1848-3380, Print ISSN 0005-1144
ATKAFF 57(3), 761–773(2016)

Emna Baklouti, Nader Ben Amor, Mohamed Jallouli

Autonomous wheelchair navigation with real time obstacle detection using 3D sensor

DOI 10.7305/automatika.2017.02.1421
UDK [681.51.07:510.644.4]:629.312.072.1-83-52

Original scientific paper

Autonomous wheelchairs operating in dynamic environments need to sense its surrounding environment and adapt the control signal, in real-time, to avoid collisions and protect the user. In this paper we propose a robust, simple and real-time autonomous navigation module that drives a wheelchair toward a desired target, along with its capability to avoid obstacles in a 3D dynamic environment. To command the mobile robot to the target, we use a Fuzzy Logic Controller (FLC). For obstacle avoidance, we use the Kinect Xbox 360 to provide an actual map of the environment. The generated map is fed to the reactive obstacle avoidance control Deformable Virtual Zone (DVZ). Simulations and real world experiments results are reported to show the feasibility and the performance of the proposed control system.

Key words: Wheelchair, Autonomous navigation, FLC, Obstacle avoidance, 3D environment, DVZ, 3D sensors

Autonomna navigacija za invalidska kolica s detekcijom prepreka u stvarnom vremenu korištenjem 3D senzora. Autonomna invalidska kolica koja se kreću u dinamičkim okruženjima moraju biti sposobna detektirati prepreke u svojoj okolini, te prilagoditi upravljački signal u stvarnom vremenu kako bi se izbjegli sudari i zaštitio korisnik. U ovom radu predlaže se jednostavan, robustan modul za autonomnu navigaciju u stvarnom vremenu koji vodi invalidska kolica prema željenom odredištu, te omogućuje izbjegavanje prepreka u 3D okruženju. Za upravljanje koristi se regulator baziran na neizravnoj logici (FLC). Za izbjegavanje prepreka koristi se Kinect Xbox 360 senzor koji gradi kartu okoline. Generirana karta se predaje reaktivnoj kontroli za izbjegavanje prepreka Deformiranoj Virutalnoj Zoni (DVZ). Prikazani su rezultati simulacija i eksperimenata u stvarnom svijetu kako bi se pokazala izvedivost i kvaliteta izvođenja predloženog sustava upravljanja.

Ključne riječi: invalidska kolica, auotonomna navigacija, FLC, izbjegavanje prepreka, 3D okruženje, DVZ, 3D senzor

1 INTRODUCTION

The latest statistics in Tunisia show a number of more than 150000 of disabled people. This number increases every year mainly because of traffic accidents [1]. People with mental or mobility handicaps make up approximately 42.5% of handicapped people. The wheelchair is one of the most popular apparatus that helps those people moving more easily. Traditional electric- powered wheelchairs (EPWs) are generally manual controlled by users via a standard joysticks. However, a significant number of individuals with severe motor impairments are unable to control their wheelchair using this device. In the attempt of assisting those people, many research projects of intelligent wheelchairs (IWs) have been carried over the last years [2]. It is assumed that IW may present at least some skills such as an intelligent interfacing with user and an autonomous navigation capability for good safety, flexibility, mobility, obstacle avoidance, etc. According to the general concept, an intelligent wheelchair can be defined as a robotic device built from an electric powered wheelchair, provided with a sensorial system, actuators and processing capabilities [3]. The successful development of the Intelligent Wheelchair (IWs) consists on its high performance and low cost.

The most advanced control allows an automatic movement of the wheelchair in known environments to a final destination first selected by the user [4]. This control mode is an efficient solution to assist severely handicapped people who are unable to provide low level orders, who get tired easily or who have visual impairment [5]. The essential feature of autonomous wheelchair navigation is obstacle detection. These techniques developed in robotics fields have the potential to improve user's safety and reduce the navigation complexity.

Obstacle avoidance consists basically on shaping the robot's path to overcome unexpected obstacles. A real-time collision avoidance method is composed essentially

of three parts: perception of the environment, collision detection and robot moving control. A significant number of algorithms described in previous paragraph have been developed to overcome obstacles. They differ basically in the sensorial data and the control strategies. Obstacles detection needs an actual map of the environment provided by sensors. The traditional two-dimensional (2D) obstacle detection techniques are often insufficient for safe navigation through complex environments. In this case, 3D sensors and 3D reactive detection algorithms are a promising solution to an accurate exploration for different shape obstacles. Figure. 1 illustrates an example of such capability. Using a 3D obstacle detection, the wheelchair will be able to go under the table if is sufficiently high.

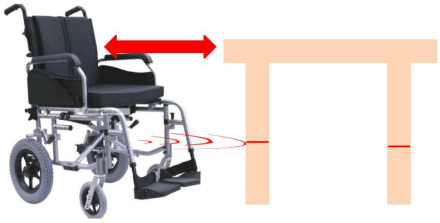


Fig. 1. Capabilities not performed with 2D planar sensors

In this paper, we propose a control system for enhancing wheelchair maneuverability for severe impairment users. The design of this system is based on the extension and the improvement of the works described in [6]. It consists of an autonomous driving module. The objective through this module is to develop a navigation system that is able to drive an electric wheelchair from a start position to a target position without human intervention. It provides the following functions: goal tracking for robot navigation coupled with obstacle detection and avoidance via 3D proximity information. In this driving module, the wheelchair motion model and the physical features are fixed by user through a graphical user interface (GUI). Also,we provide a wheelchair path simulation tool that permits a priori validation of the navigation process before sending the command to the real wheelchair for better security.

The paper is organized as follows. Sections 2 reviews related works for obstacle avoidance control and 3D sensors. Section 3 details our contribution in the various part of our module as FLC goal tracking , kinect Xbox generated 3D map and DVZ extension. Finally, section 5 reports on the obtained results.

2 RELATED WORKS

2.1 Obstacle avoidance

Obstacle avoidance can be defined as the ability to react when unscheduled events occur. Classical obstacle avoidance algorithms are based on behavioural approach, consisting in converting sensor reading to motion instructions through state machines (actions). Several techniques can be applied for obstacles avoidance such as fuzzy logic, neural networks and so on. Naturally these methods can be neither exhaustively tested or proved effective in all cases [7].

Alternatively, analogies in physical field have been used to derive methods for obstacle avoidance. This category of techniques uses sensory information in the robot control loop. The oldest and most famous methods are the Artificial Potential Field (APF) method developed by O. Khatib and The Vector Field Histogram (VFH) approach introduced by Borenstein. Using the APF method, every obstacle exerts a repulsing force on the robot while the goal exerts an attractive force. The main difficulty of this method is to design an artificial potential function without undesired local minimum. Elnagar and al, in [8], propose to model the potential field by Maxwell's equations that completely eliminate the local minima problem, with condition that a prior knowledge of the environment is available. The vector field histogram (VFH) is developed based on the concept of APF and the certainty grid. The VFH approach generates a polar histogram of the space occupancy in the vicinity of a robot. This polar histogram is then checked to select the most suitable sector out of all polar histogram sectors which have a low polar obstacle density. The steering of the robot is then aligned with that direction. The VFH approach is computationally expensive. It complicates the implementation and whereas the convergence is not ensured in some cases like a U-shaped corridor [9].

Another more recent approach, based on a reflex behavior reaction was developed by R.Zapata. It consists in using a Deformable Virtual Zone (DVZ). The main idea is to set the robot/environment interaction as a virtual zone surrounding the robot, whose configuration depends on the robots states. The deformation of this zone is due to the intrusion of proximity information. The system reaction is made in order to reform the risk zone to its nominal shape. The main characteristic of this approach is that there is no geometrical modelling of the obstacles, but rather a vector representation of the interaction between the robot and its environment [10]. On the other hand, experiments carried out on real robots with sensors have shown that there is never a real symmetry in the deformation of the DVZ and then it couldn't fall into local minima [11]. However the DVZ-based method is designed and detailed mainly for obstacles in 2D space. In the literature, Very few methods deal with 3D workspaces and existing algorithms are inherited from 2D to 3D workspaces [12, 13]. In this paper, we propose an extension of DVZ method to the case of 3D obstacles. Also, DVZ principle cannot be implemented alone

as it does not implement a planning procedure. These features make this reactive algorithm efficient for control hierarchies and used as a building block in path planning or goal tracking algorithms. Thus, in this paper we coupled DVZ algorithm with a goal tracking controller.

2.2 3D Sensors

As a common characteristic, collision avoidance algorithms require the measure of distances between the robot and the different obstacles. There is a wide variety of obstacle perception sensors. However, the most commonly used ones are laser range finders, sonars and cameras. Each of the above listed sensor types has its own strengths and weaknesses as presented in [14]. The laser is a heavy device, but provides long range, high precision and high rate measurements. The laser works by projecting infra-red light and measuring the time to return. The main problems with laser scanner are high energy consumption and cost. This cost is nearly 60% the cost of the wheelchair itself in some models. One of the lowest cost sensors is sonar. The main problem with this device is that depth readings are more coarsely spaced and less accurate. Common drawback for both approaches (laser and sonar), that they provide only readings for a horizontal plane parallel to the floor.

3D obstacles collision avoidance has been successfully solved with the use of cameras. For instance, cameras are easily available but do not directly provide geometrical measurements of an environment, which must be alternatively inferred from the pixel data. It requires a significant delay to acquire a full 3D representation of the environment. Nevertheless it enables an accurate depth readings in real time. A recent 3D camera sensor, such as the Xbox Kinect structured light camera, can also provide a set of dense depth measures via a fulcrum-base. Moreover, the cost of kinect is cheap, which makes them practical for robotic systems. In our system, navigation is performed using a Microsoft Xbox Kinect sensor which provides RGB color and 3D depth imaging data. The main purpose of using Kinect device in this module is to provide 3D obstacles position from depth data which is necessary for obstacle detection algorithm and robot motion control.

3 AUTONOMOUS MODULE

Figure. 2 illustrates our control module for autonomous driving wheelchair. The proposed control architecture presents a solution to the problem of wheelchair navigation in an 3D cluttered environment without human intervention. We start at the beginning by ensuring a first constraint which is the wheelchair navigation. To command the wheelchair navigation wheels autonomously, we use one of the most common strategies for the mobile robot navigation which are Fuzzy Logic Controllers (FLCs) [15]. FLC generates velocities to be applied on the mobile base wheels during the transition from the initial position toward another desired position, taking into account wheelchair kinematic constraints. Next, we explore the kinect sensors to provide a real time map of the scene in front. Thanks to its RGB camera and depth sensor, we can extract obstacles position. After, we feed the DVZ obstacle avoidance algorithm with obstacles coordinates.

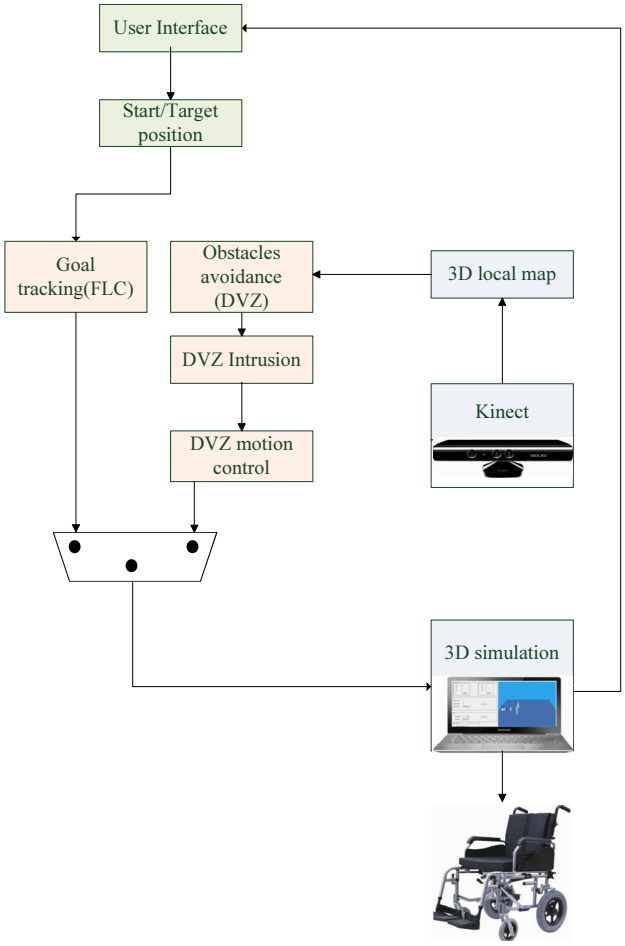


Fig. 2. Control module for autonomous driving wheelchair

It is the amount of DVZ intrusion that allows the system to make decision to keep FLC control or switch to DVZ control.Thus, we fixed threshold values $I_{FLCtoDVZ}$ and $I_{DVZtoFLC}$ for allowed intrusion. $I_{FLCtoDVZ}$ is the minimum allowed intrusion before proceeding to DVZ control and $I_{DVZtoFLC}$ is the allowed intrusion that permit to reverse to FLC control. Two cases may occur. For the first case, the wheelchair is intended to reach a target while the distance to an obstacle decreases, i.e. the intrusion increases from $I_{FLCtoDVZ}$ to I_{max} where I_{max} is the maximum reached intrusion. Consequently, the controller switches to DVZ algorithm and persists in this sit-

uation as the intrusion $I \succ I_{DVZtoFLC}$. In the second case, the robot is trying to avoid an obstacle, i.e. the deformation decreases from I_{max} to $I_{DVZtoFLC}$. In this case, the controller switches to fuzzy navigation only when $I \prec I_{DVZtoFLC}$ in order to eliminate the transition between the two controllers. Finally, we suggest a personalized interaction interface with 3D simulation of the performed path for each step in order to show the feasibility and the effectiveness of the proposed control system before real tests. In section 3.1, 3.2 and 3.3, we give a more detailed description of these steps.

3.1 FLC goal tracking for wheelchair:

Goal tracking or path following control methods are based on reducing the errors between the current and the desired robot position. Many methods are based on a PID controller or non-linear controller to decrease these errors. But, using just the current angle errors data, makes these methods unable to improve the control performance by considering the robot's dynamic and kinematic constraints. For example, when a robot makes a sharp turn with high translation and rotation velocities, motors will easily be saturated. This problem can degrade the robot's performance and in some cases causes robot instability [16]. In other works, path following control set up the vehicle's forward velocity tracks a desired speed profile, while the controller acts on the vehicle orientation to drive it to the path [10, 17]. These methods deal with the kinematic, dynamic and parameter uncertainty of the mobile robot. The fuzzy logic controller used in this paper for goal tracking problem handles the kinematic constraints. The controller acts only on the mobile robot velocity to drive it to the goal.

3.1.1 Kinematic model of wheelchair

A wheelchair is a unicycle robot with two steering wheels and two independent driving wheels. The wheelchair can be oriented and commanded by acting on the speed of each wheel. The kinematic model is given by Eq. 1:

$$\begin{cases} \frac{dx_R}{dt} = \frac{V_R+V_L}{2}\cos\alpha_R \\ \frac{dy_R}{dt} = \frac{V_R+V_L}{2}\sin\alpha_R \\ \frac{d\alpha_R}{dt} = \frac{V_R-V_L}{L} \end{cases} \quad (1)$$

where V_R and V_L are the wheelchair's right and left wheels' velocities, respectively; $\dot{\alpha}_R$ is the robot's angular velocity, L is the distance between the two wheels and α_R is the angle between the wheelchair's direction and the X-axis as shown in the schematic model of the Fig. 3.

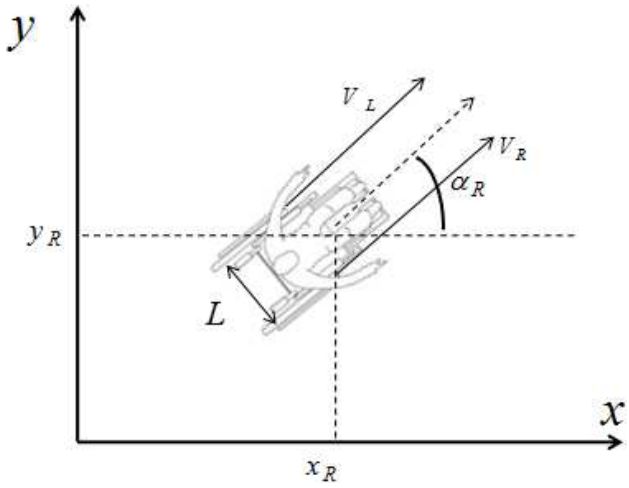


Fig. 3. The schematic model of wheelchair

To progressively generate displacement increments allowing the robot to reach the target, we are based on the process of dead reckoning. Dead-reckoning method provides the new position and orientation of the robot using the previous position, orientation and known or estimated speeds over elapsed time. Our dead-reckoning process is described by Eq. 2 , where T is the sampling time.

$$\begin{cases} x_R^{new} = x_R^{old} + T\frac{V_R^{old}+V_L^{old}}{2}\cos\alpha_R^{old} \\ y_R^{new} = y_R^{old} + T\frac{V_R^{old}+V_L^{old}}{2}\sin\alpha_R^{old} \\ \alpha_R^{new} = \alpha_R^{old} + T\frac{V_R^{old}-V_L^{old}}{L} \end{cases} \quad (2)$$

3.1.2 Fuzzy Logic Controller

The controller proposed in this paper adapts the work presented in [17] for wheelchair. It allows wheelchair to reach the target position (x_T, y_T, θ_T) from a current position (x_R, y_R, α_R) as presented in Fig. 4. Figure. 5 shows that our controller has two inputs: the distance d and the angle φ. d is the distance between the center of the robot and its target and φ is the difference between the angle of the robot's direction and the angle θ_T of the connecting line between the center of the robot and its target.

The distance d and the angle φ are expressed according to Eq. 3 and Eq. 4. The controller outputs are velocities V_R and V_L for the right and left wheels of the wheelchair.

$$d = \sqrt{(x_T - x_R)^2 - (y_T - y_R)^2} \quad (3)$$

$$\varphi = \theta_T - \alpha_R \quad (4)$$

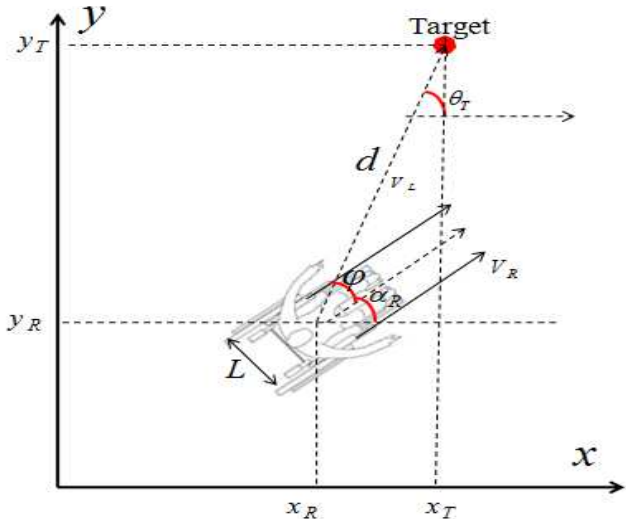


Fig. 4. Wheelchair configuration according to the objective

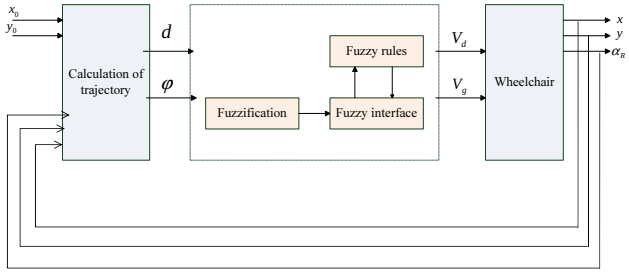


Fig. 5. Schematic diagram of wheelchair fuzzy control for goal tracking

With

$$\theta_T = \tan^{-1} \frac{y_T - y_R}{x_T - x_R} \tag{5}$$

The distance d ranges between zero and d_{max} with d_{max} is the maximum distance in the area of navigation of the wheelchair. Five fuzzy subsets are assigned to the variable distance (d): VS: Very Small; S: Small; M: Medium; L: Large and VL: Very Large. Membership functions for distance (d) are outlined in Fig. 6.

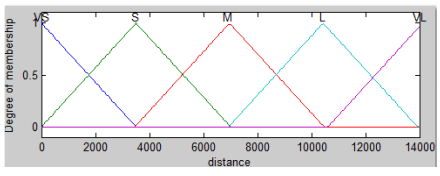


Fig. 6. Membership functions for the distance d (mm)

The angle φ is constrained between -180 to $+180$. Seven fuzzy subsets have been associated to the angle (φ) : NL: Negative Large; NM: Negative Medium; NS: Negative Small; Z: Zero; PS: Positive Small; PM: Positive Medium; and PL: Positive Large. Membership functions for the angle (φ) are reported in Fig. 7.

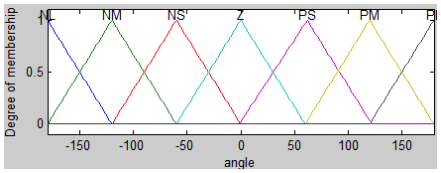


Fig. 7. Membership functions for the angle φ

With each variable input value combination, an action of the output variables is associated with it. Fuzzy rules (situation/action) are proposed in Table 1 for the left wheel and Table 2 for the right wheel. We use 35 rules manually built after several simulations.

Table 1. Linguistic inference table for V_L

V_L		φ						
		NL	NM	NS	Z	PS	PM	PL
d	VS	B	M	S	Z	Z	S	S
	S	VB	B	M	S	S	S	M
	M	VB	VB	B	M	S	M	B
	L	VB	VB	VB	B	M	B	B
	VL	VB	VB	VB	VB	B	B	B

Table 2. Linguistic inference table for V_R

V_R		φ						
		NL	NM	NS	Z	PS	PM	PL
d	VS	S	S	Z	Z	S	M	B
	S	M	S	S	S	M	B	VB
	M	B	M	S	M	B	VB	VB
	L	B	B	M	B	VB	B	VB
	VL	B	B	B	VB	VB	VB	VB

With the notations: V_L: Left Velocity; V_R: Right Velocity; Z: Zero; S: Small; M: medium; B: Big; VB: Very Big.

Examples of fuzzy rules:

- Rule 1:
  IF [the distance d] is Very Small AND
  IF [the angle φ] is Negative Large THEN [Left Velocity] is Big
  AND [Right Velocity] is Small

According to this rule, the wheelchair turn to right with large difference between left and right speed as the distance to the target is small

- Rule 35:
  IF [the distance d] is Very Large AND
  IF [the angle φ] is Positive Large THEN [Left Velocity] is Big
  AND [Right Velocity] is VB
  According to this rule, the wheelchair turn to left with large speed as the distance to the target is large

In this work, we used a zero-order Sugeno model. Simulation tests have been carried out in order to attribute numerical values to the wheels' velocities V_R and V_L (see Fig. 8).

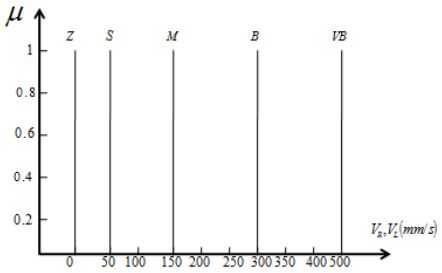


Fig. 8. Output variables

3.2 Kinect real time and local map generation

The Kinect for Xbox 360 is a low-cost vision device equipped with two cameras and a laser-based infrared (IR) projector. One of the cameras is a standard RGB camera, while the other is an IR camera which looks for a specific pattern projected onto a scene by the laser-based IR projector. The Kinect provides horizontal field of view of $57°$and a vertical field of view of $43°$. Its range varies between 0.5 and 4 meters. The kinect sensor is dedicated mainly to control video games where the user becomes the controller using gestures and body movements. In this section, we will show how to use the Kinect sensor for obstacle detection and generation of the local map of the environment based on the OpenNI driver for the acquisition and PCL library for displaying and processing of the depth image. This map will be supplied directly to the obstacle avoidance controller.

3.2.1 From depth image to a point cloud

Thanks to its depth camera, the kinect can acquire distance measurements and generate a depth image. The OpenNI framework allows to establish a points cloud from the depth image as presented in Fig. 9.

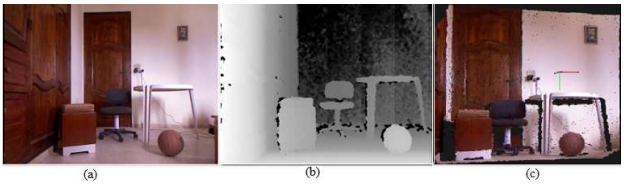


Fig. 9. (a) Kinect is RGB image, (b) depth image, (c) point cloud

The cloud consists of $307,200 \approx (680 * 460)$ points. Each point is described by its coordinates (x, y, z), RGB color, intensity, etc. Black areas observed in the Fig. 9.(c) are areas where the Kinect was not able to calculate their depth. Indeed, shiny surfaces, direct sunlight and objects too close to the sensor can results these black areas in the cloud. In order to reduce the number of items processed by the autonomous navigation module, we chose to eliminate these points. Indeed, discarding these items does not cause a problem because usually these regions are small, weak, dispersed and do not constitute an entire obstacle. So for each wrong pixel, another very close pixel will always be present with a real and proper depth value. Other methods to reduce the number of points in the cloud are introduced in the next sections.

3.2.2 VoxelGrid Filter

With a $7.8\mu m$ pixel size, the adjacent points should be very similar in depth values. So we could achieve similar accuracy while reducing the number of analyzed points. This reduces the processing time while maintaining useful pixels data. The filter VoxelGrid of the library PCL is responsible for carrying out this feature. The VoxelGrid class creates a 3D voxel grid (3D boxes in space). Then, in each voxel, all the present points will be approximated with their centroid. Figure. 10 shows the point cloud after applying the filter VoxelGrid with the dimension of the voxel is $(1cm * 1cm * 1cm)$

3.2.3 PassThrough Filter

The idea is to fix the range of the Kinect sensor. In fact, providing only the point cloud of obstacles at a distance of $2m$ from the sensor is sufficient to implement the control architecture. The filter PassThrough of the PCL library can be used to fix the range of this sensor. This filter eliminates depth points that are above or below a given threshold. Figure. 11 shows the point cloud with a threshold that fix the maximum range of the kinect to $2m$.

Fig. 10. Point cloud after applying the filter VoxelGrid

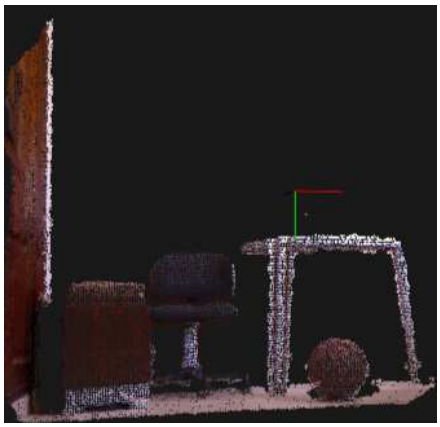


Fig. 11. Point cloud with a maximum range of $2m$

3.2.4 RANSAC

RANSAC algorithm allows to remove plans captured by the Kinect but which are not considered as obstacles like the floor. There are many approaches for determining the existing plans in a scene. We adopt, the RANSAC method (RANdom SAmple Consensus) to find plans. This method can provide an accurate estimation of plans with small amount of calculation. The RANSAC algorithm takes three randomly points from the cloud to construct a geometric plan. Then each point in the cloud is evaluated to see if it's belongs (inliers) or not (outliers) of the plan. This method can detect important bodies that have a significant size in the entire cloud.

3.2.5 Euclidean Cluster Extraction

This algorithm is used for a better representation of obstacles where each object is placed in a separate table. This method consists of dividing an unorganized cloud in different parts by subdivision the 3D space in different boxes or

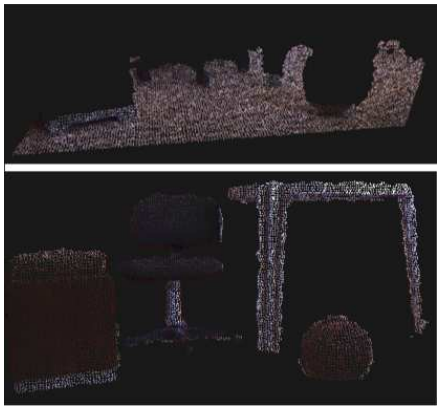


Fig. 12. Above the detected plan and below the cloud points that are not part of this plan

different type of data structure as tree. After applying this method to our scene, we obtained as shown in Fig. 13, distinguish objects that can be manipulated separately.

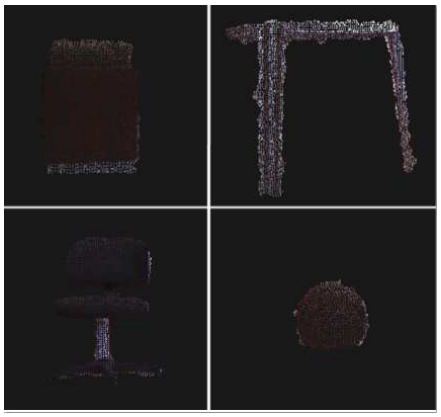


Fig. 13. Objects after extraction

3.3 DVZ formulation

In current DVZ literature, the risk zone is an ellipse surrounding the robot. To address 3D obstacles, the risk zone is defined as an ellipsoid that protects the robot.

3.3.1 3D undeformed DVZ $d_h(\theta, \phi)$:

In this section, we suggest an analytical expression for the 3D undeformed DVZ. Mathematically, an ellipsoid shape is defined by three axes c_x, c_y and c_z. We assume that the proper reference frame of the DVZ is translated from the center of the robot frame (R) by a vector $[a_x a_y a_z]$.Thus, the Cartesian equation of the ellipsoid is presented by Eq. 6:

$$\frac{(x-a_x)^2}{c_x^2} + \frac{(y-a_y)^2}{c_y^2} + \frac{(z-a_z)^2}{c_z^2} = 1 \qquad (6)$$

The 3D undeformed DVZ moves with the robot. Therefore, the coefficients a_x, a_y, a_z, c_x, c_y and c_z are chosen in a way that they depend on the translational velocity of the robot.

$$\begin{cases} c_x = \lambda_{cx} V^2 + c_{xmin} \\ c_y = \frac{\sqrt{5}}{3} c_x \\ c_z = c_y \\ a_x = -\frac{2}{3} c_x \\ a_y = a_z = 0 \end{cases} \tag{7}$$

with c_{xmin} is the minimum length that can have the axis of the ellipsoid. The choice of the value of this variable is related to the width and length of the robot and λ_{cx} is a constant which determines the degree of dependency of this zone on the linear speed of the robot. Further-more, Considering that the DVZ is rigidly attached to the robot, oriented in the main direction of the robot movement, the resulting DVZ needs to be rotated around the z-axis with the angle of the robot.

$$\begin{cases} \acute{x} = x cos\alpha_R + y sin\alpha_R \\ \acute{y} = y cos\alpha_R - x sin\alpha_R \\ \acute{z} = z \end{cases} \tag{8}$$

Considering the spherical coordinates $(d_h(\theta,\phi), \theta, \phi)$ of a point P, the Cartesian coordinate system (x, y, z), are defined by Eq. 9

$$\begin{cases} x = d_h(\theta,\phi) cos\theta cos\phi \\ y = d_h(\theta,\phi) sin\theta cos\phi \\ z = d_h(\theta,\phi) sin\phi \end{cases} \tag{9}$$

with $d_h(\theta,\phi)$ is the distance between the center of the robot and the undeformed DVZ, θ refers to the longitude direction in the DVZ and ϕ refers to the latitude direction as shown in Fig. 14.

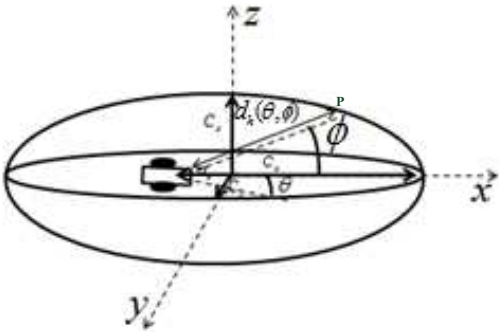


Fig. 14. Undeformed DVZ parametres in 3D

Substituting Eq. 8 and Eq. 9 in Eq. 6 leads to the quadratic equation:

$$A d_h(\theta,\phi)^2 + B d_h(\theta,\phi) + C = 0 \tag{10}$$

with:

$$\begin{cases} A = c_y^2 c_z^2 cos\alpha_R^2 cos\theta^2 cos\phi^2 + 2 * cos\alpha_R sin\alpha_R \\ \quad cos\phi^2 cos\theta sin\theta * (c_y^2 c_z^2 - c_x^2 c_z^2) + c_y^2 c_z^2 sin\alpha_R^2 \\ \quad cos\phi^2 sin\theta^2 + c_x^2 c_z^2 cos\alpha_R^2 sin\theta^2 cos\phi^2 + c_x^2 c_z^2 \\ \quad sin\alpha_R^2 cos\theta^2 cos\phi^2 + c_x^2 c_y^2 sin\phi^2 \\ B = -2 c_y^2 c_z^2 a_x cos\alpha_R cos\theta cos\phi - 2 c_y^2 c_z^2 a_x sin\alpha_R \\ \quad sin\theta cos\phi \\ C = a_x^2 c_y^2 c_z^2 - c_x^2 c_y^2 c_z^2 \end{cases} \tag{11}$$

the solution of Eq. 10 is Eq. 12

$$d_h(\theta,\phi) = \frac{-B + \sqrt{B^2 - 4AC}}{2A} \tag{12}$$

3.3.2 3D deformed DVZ $d(\theta,\phi)$:

The 3D deformed DVZ is acquired from Kinect information. But the meaningful information is restricted to that inside the 3D undeformed DVZ. So, a preliminary test on the Kinect information is necessary. If we consider that $c(\theta,\phi)$ is the kinect output and that $d(\theta,\phi)$ is the distance between the Kinect and the deformed DVZ (see Fig. 15), then $d(\theta,\phi)$ is obtained by saturation of $c(\theta,\phi)$ according to Eq. 13:

$$d(\theta,\phi) = \begin{cases} c(\theta,\phi) & if c(\theta,\phi) \prec d_h(\theta,\phi) \\ d_h(\theta,\phi) & elsewhere \end{cases} \tag{13}$$

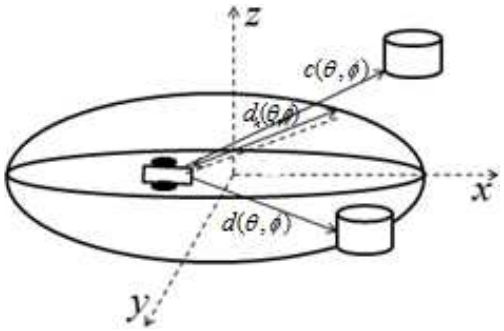


Fig. 15. 3D Deformed DVZ

3.3.3 Reactive term (Intrusion):

The choice of the intrusion information expression is instrumental in the control of the robot reactivity. The intrusion ratio is introduced to quantify the DVZ deformation. We introduce the intrusion ratio by Eq. 14:

$$I = \int_{\theta=0}^{\pi} \int_{\phi=0}^{\pi} \frac{d_h(\theta,\phi) - d(\theta,\phi)}{d(\theta,\phi)} d\theta d\phi \tag{14}$$

The Eq. 14 quantifies the global amount of intrusion, normalized by the factor $d(\theta,\phi)$. Note that a null distance

robot/obstacle yields an infinite value of I. Thus, a control strategy that guarantees a bounded intrusion ratio at any time insures that the system avoids any obstacle.

3.3.4 Control design for 3D static and/or dynamic environments:

We design this controller in order to minimize the DVZ deformation. Thus, to conceive the speed applied to the wheels of the mobile robot, we choose to satisfy the Lyapunov theorem of stability according to Eq. 15.

$$V_I = \frac{I^2}{2} \tag{15}$$

The derivation generates:

$$\dot{V}_I = I(J_I^V \dot{V} + J_I^{\alpha_R}\omega + F^{Rob.vel}V + F^{Obs.vel}) \tag{16}$$

with

$$\begin{cases} J_I^V = \int_{\theta=0}^{\pi}\int_{\phi=0}^{\pi} \frac{1}{d(\theta,\phi)}(\frac{1}{2A}[-J_V^B + \frac{BJ_V^B - 2CJ_V^A - 2AJ_V^C}{\sqrt{(B^2-4AC)}}] \\ \quad -J_V^A\frac{-B+\sqrt{(B^2-4AC)}}{2A^2})d\theta d\phi \\ J_I^{\alpha_R} = \int_{\theta=0}^{\pi}\int_{\phi=0}^{\pi} -\frac{1}{d(\theta,\phi=)}(\frac{1}{2A}[-J_{\alpha_R}^B + \frac{BJ_{\alpha_R}^B - 2CJ_{\alpha_R}^A}{\sqrt{(B^2-4AC)}}] \\ \quad -J_{\alpha_R}^A\frac{-B+\sqrt{(B^2-4AC)}}{2A^2})d\theta d\phi \\ F^{Rob.vel} = \int_{\theta=0}^{\pi}\int_{\phi=0}^{\pi}\frac{d_h(\theta,\phi)}{d^2(\theta,\phi)}\cos(\theta)\cos(\phi)d\theta d\phi \\ F^{Obs.vel} = \int_{\theta=0}^{\pi}\int_{\beta=0}^{\pi}\frac{d_h(\theta,\phi)}{d^2(\theta,\phi)}[\dot{X}_o\cos(\theta+\alpha_R)\cos(\phi) \\ \quad +\dot{Y}_o\sin(\theta+\alpha_R)\cos(\phi) + \dot{Z}_o\sin(\phi)]d\theta d\phi \end{cases} \tag{17}$$

Where $\dot{X}_o$,$\dot{Y}_o$, $\dot{Z}_o$ define the obstacle absolute velocity. The assumption of static obstacles yields $F^{Obs.vel} = 0$. and

$$\begin{cases} J_V^A = 2*(c_y\frac{dc_y}{dV}c_z^2 + c_z\frac{dc_z}{dV}c_y^2)cos\alpha_R^2 cos\theta^2 cos\phi^2 + \\ \quad 4*cos\alpha_R sin\alpha_R cos\phi^2 cos\theta sin\theta*(c_y\frac{dc_y}{dV}c_z^2 + \\ \quad c_z\frac{dc_z}{dV}c_y^2 - c_x\frac{dc_x}{dV}c_z^2 - c_z\frac{dc_z}{dV}c_x^2) + \\ \quad 2*(c_y\frac{dc_y}{dV}c_z^2 + c_z\frac{dc_z}{dV}c_y^2)sin\alpha_R^2 cos\phi^2 sin\theta^2 + \\ \quad 2*(c_x\frac{dc_x}{dV}c_z^2 + c_z\frac{dc_z}{dV}c_x^2)cos\alpha_R^2 sin\theta^2 cos\phi^2 + \\ \quad 2*(c_x\frac{dc_x}{dV}c_z^2 + c_z\frac{dc_z}{dV}c_x^2)sin\alpha_R^2 cos\theta^2 cos\phi^2 + \\ \quad 2*(c_x\frac{dc_x}{dV}c_y^2 + c_y\frac{dc_y}{dV}c_x^2)sin\phi^2 \\ J_{\alpha_R}^A = -2*c_y^2c_z^2 cos\alpha_R sin\alpha_R cos\theta^2 cos\phi^2 + \\ \quad 2*cos\phi^2 cos\theta sin\theta*(c_y^2c_z^2 - c_x^2c_z^2)* \\ \quad ({cos\alpha_R}^2 - {sin\alpha_R}^2) + \\ \quad 2*c_y^2c_z^2 cos\alpha_R sin\alpha_R sin\theta^2 cos\phi^2 - \\ \quad 2*c_x^2c_z^2 cos\alpha_R sin\alpha_R sin\theta^2 cos\phi^2 + \\ \quad 2*c_x^2c_z^2 cos\alpha_R sin\alpha_R cos\theta^2 cos\phi^2 \end{cases} \tag{18}$$

$$\begin{cases} J_V^B = -2(4*c_y^3\frac{dc_y}{dV}a_x + c_y^4 a_x\frac{da_x}{dV})(cos\alpha_R \\ \quad cos\theta cos\phi + sin\alpha_R sin\theta cos\phi) \\ J_{\alpha_R}^B = -2*c_y^2c_z^2a_x(cos\alpha_R sin\theta cos\phi - sin\alpha_R \\ \quad sin\theta cos\phi) \end{cases} \tag{19}$$

$$\begin{cases} J_V^C = 2*a_x\frac{da_x}{dV}c_y^4 + 4*a_x^2c_y^3\frac{dc_y}{dV} - 2*c_x\frac{dc_x}{dV}c_y^4 \\ \quad -4*c_x^2c_y^3\frac{dc_y}{dV} \end{cases} \tag{20}$$

Hence, we come out with the robot control system which is expressed through the Eq. 21 and Eq. 22, where $K_V and K_{r_1}$ are arbitrary positive gains yielding $\dot{V}_I \leq 0 \forall t$. In the case of static obstacles:

$$\begin{cases} \dot{V} = -K_V J_I^V I - \frac{F^{Rob.vel}}{J_I^V}V \\ \omega = -K_{r_1}J_I^{\alpha_R}I \end{cases} \tag{21}$$

In the case of dynamic obstacles, where the speed of the obstacle is known in advance:

$$\begin{cases} \dot{V} = -K_V J_I^V I - \frac{F^{Rob.vel}}{J_I^V}V - \frac{F^{Obs.vel}}{J_I^V} \\ \omega = -K_{r_1}J_I^{\alpha_R}I \end{cases} \tag{22}$$

4 SIMULATION AND REAL WORLD EXPERIMENT RESULTS

To test the robustness of our control approach, we start with the simulation of the autonomous control in a full virtual environment. The first environment includes 3 fixed obstacles placed in different positions. The robot starting position is $(x = -2.5m, y = 2.5m)$ and the target position is $(x = 2.5m, y = -2.5m)$. Figure. 16 shows the path followed by the wheelchair without (entirely blue) and with obstacle avoidance control (blue and green) to reach the desired target. The Wheelchair reaches the target safely.

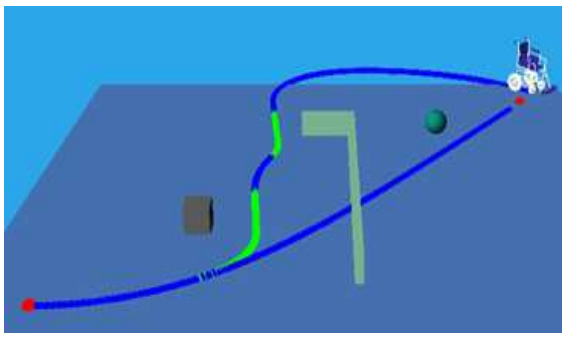


Fig. 16. Wheelchair's path without and with DVZ control

Figure. 17 shows the simulation of the autonomous navigation module for a virtual environment with suspended obstacle. In Fig. 17, the box number 1 is not sufficiently high to allow the robot to navigate under it (case

when $c(\theta,\phi) \prec d_h(\theta,\phi)$) while the box number 2 is sufficiently high to permit the robot to navigate under it. (case when $c(\theta,\phi) \succ d_h(\theta,\phi)$).

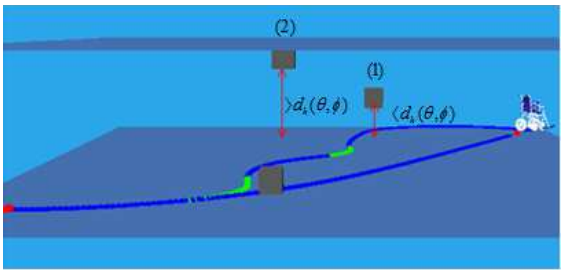


Fig. 17. DVZ configuration for suspended obstacles

To test the robustness of our control approach for real obstacles, the robot control software communicates with the kinect sensor for receiving obstacle coordinate.Thus, 3D points from the Kinect frame are transformed to the wheelchair frame.

As a first scenario, we placed three static obstacles in the field of view of the Kinect. The first obstacle is a table placed at a distance of $1.25m$, the second is a ball placed at a distance of $1.75m$ and the third is a chair placed at a distance of $2.75m$. Figure. 18 shows the actual scene detected by the Kinect.

Fig. 18. Real environment with static obstacles

For this scenario the wheelchair must reach the target position on $(x = 0m, y = 2.2m)$ from the starting position $(x = 0m, y = -2.2m)$ and with the starting angle $(\alpha_R = 90°)$. Figure. 19 describes the path made by the wheelchair to avoid these obstacles at different times. The trajectory shows clearly the efficiency of the control module in different levels such as obstacle detection, reactivity of the system in front of these obstacles, command generated for avoidance and the switch between the two controllers goal tracking and obstacles avoidance.

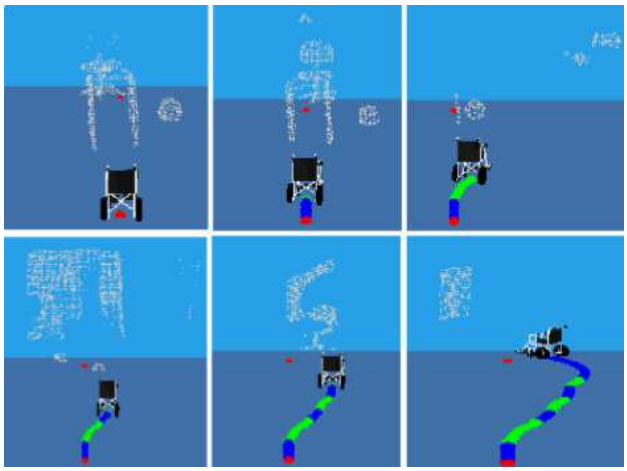


Fig. 19. Wheelchair path for static obstacles

Figure. 20 shows the speed of the right wheel V_R and left wheel V_L generated by the overall control module. Initially, $V_R = V_L$ because robot navigation is along a straight line towards the target. Then there is a rise of V_L for three moments. These moments present the cases when the wheelchair tries to avoid obstacles by turning right. For moments when V_R is larger than V_L, the wheelchair looks after each obstacle avoidance control to recover its direction towards the target by turning left.

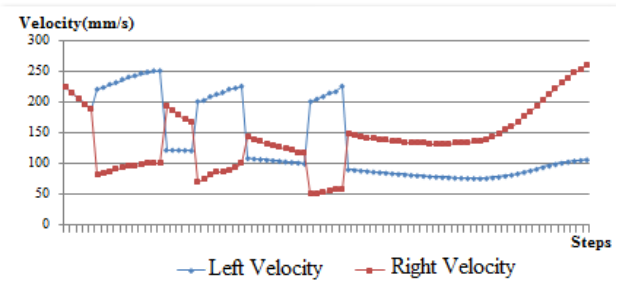


Fig. 20. Speeds of the right and left wheel in the case of static obstacles

The second scenario consists to present an empty room with a person who moves in. The person is asked to approach gradually to the kinect as shown in Fig. 21.

This scenario allows to test the system's ability to detect and avoid dynamic obstacles. Figure. 22 shows that the wheelchair deviates from the normal trajectory for 6 times. The Kinect detects this person (considered as a moving obstacle) several times. But this person, in his

Fig. 21. Real environment with dynamic obstacles

movements, causes the deformation of the safety zone 6 times.

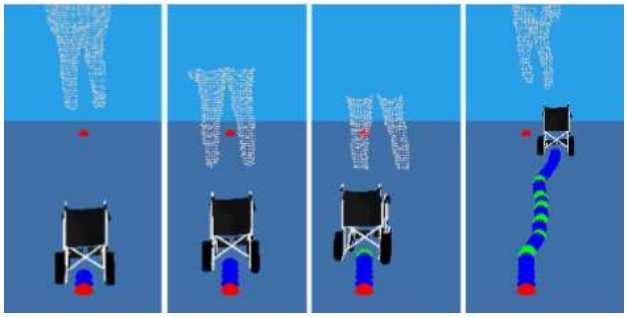


Fig. 22. Wheelchair path for dynamic obstacles

Figure. 23 shows the speed of the right and the left wheel generated by the overall control architecture with dynamic obstacles. We note that for the first variation, the wheelchair has tried to turn to the right to avoid the person, while for the other variations, the wheelchair has tried to turn to the left to avoid the person.

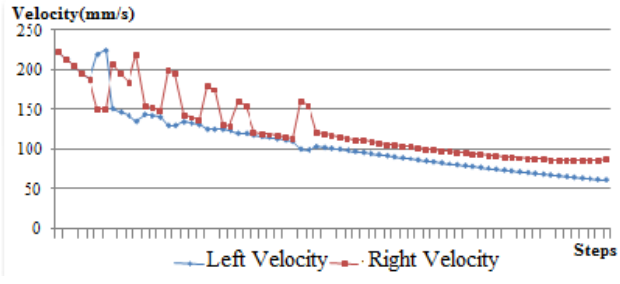


Fig. 23. Speeds of the right and left wheel in the case of dynamic obstacles

The third scenario consists on placing 3 suspended objects in the room. This scenario will test the system's ability to distinguish between obstacles by which the robot can pass safely, and those the robot will collide. Figure. 24 presents a picture of our test environment with 3 suspended obstacles. The first obstacle is placed at a height of 0.75m, the second at a height of 1.2m and the third at a height of 0.5m.

Fig. 24. Real environment with suspended obstacles

Figure. 25 shows that the wheelchair has avoided the first and third obstacles and he passed safely under the second obstacle. Indeed, for a 1m high security zone, the first and third obstacle will necessarily cause the deformation of the area, which is not the case for the second object.

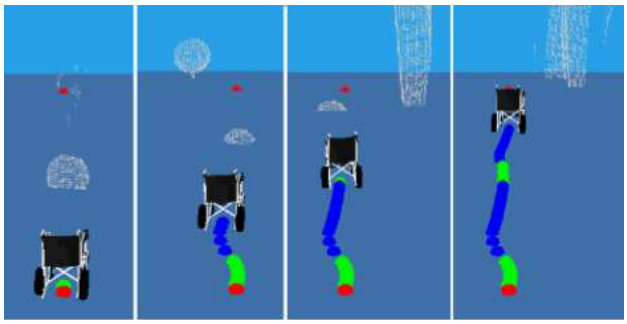


Fig. 25. Wheelchair path for suspended obstacles

Figure. 26 shows the speed of the right and the left wheel generated by the overall control architecture with suspended obstacles. The simulation starts from the beginning with an obstacle avoidance behavior by turning left to avoid the first obstacle. Then the wheelchair turns right to recover the direction of the target. This variation is repeated to avoid the third obstacle. During the movement of the wheelchair side of the second object no change is detected.

5 CONCLUSION

We have designed a control architecture for enhancing wheelchair manoeuvrability for severe impairment users.

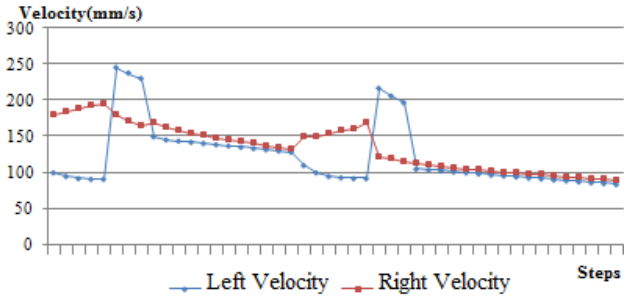


Fig. 26. Speeds of the right and left wheel in the case of suspended obstacles

It consists on a combined goal tracking using fuzzy logic and obstacle avoidance control based on the DVZ concept using the Kinect as 3D sensor. This new formulation allows a soft navigation of the wheelchair even with complex shaped and dynamic obstacles. In our system, we used the Kinect sensor, which facilitates real-time and accurate 3D measurements in an inexpensive manner. Our experiments demonstrate the feasibility of using a 3D sensor for obstacle detection and avoidance. The used sensor contains several advantages over previous sensors, such as low power, low cost, high resolution, high frame rate, and the ability to provide readings outside of a horizontal plane. This last capability is particularly important, as it enables the robot to detect obstacles based on their height.

REFERENCES

[1] M.Kassab, "Statistical data about disabled persons in tunisia," tech. rep., Physical Medicine Institute-Functional adaptation- Orthopedics Manouba, Tunis, 2010.

[2] B. M. Faria, S. Vasconcelos, L. P. Reis, and N. Lau, "Evaluation of distinct input methods of an intelligent wheelchair in simulated and real environments:a performance and usability study," *The Official Journal of RESNA*, pp. 88–98, 2013.

[3] M. Petry, A. P. Moreira, L. P. Reis, and R. Rossetti, "Intelligent wheelchair simulation: Requirements and architectural issues," *11th International Conference on Mobile Robots and Competition , Robotica*, 2011.

[4] R. Tomari, Y. Kobayashi, and Y. Kuno, "Enhancing wheelchair maneuverability for severe impairment users," *International Journal of Advanced Robotic Systems*, vol. 10, 2013.

[5] R. Simpson, E. LoPresti, and R. Cooper, "How many people would benefit from a smart wheelchair," *Journal of Rehabilitation Research and Development*, p. 53–72, 2008.

[6] E. Baklouti, M. Jallouli, N. B. Amor, S. Titi, and A. Nafti, "Autonomous mobile robot navigation coupling fuzzy logic and reactive dvz 3d obstacle avoidance control," *International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pp. 1–6, 2015.

[7] G. Alenya, A. Negre, and J. Crowley, "Time to contact for obstacle avoidance," *4th European Conference on Mobile Robots -ECMR*, pp. 19–24, 2009.

[8] A. Elnagar and A. Hussein, "Motion planning using maxwell's equations," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2002.

[9] M. Zohaib, M. Pasha, R. A. Riaz, N.Javaid, M. Ilahi, and R. D. Khan, "Control strategies for mobile robot with obstacle avoidance," *Journal of Basic and Applied Scientific Research*, vol. 3, pp. 1027–1036, 2013.

[10] L. Lapierre, R. Zapata, and P. Lepinay, "Simultaneous path following and obstacle avoidance control of a unicycle-type robot," *The International Journal of Robotics Research*, vol. 26, p. 261–375, 2008.

[11] R. Zapata and P. Lepinay, "Flying among obstacles," *IEEE*, 1999.

[12] D. Vikerimark and J. Minguez, "Reactive obstacle avoidance for mobile robots that operate in confined 3d workspaces," *Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean*, pp. 1246–1251, 2006.

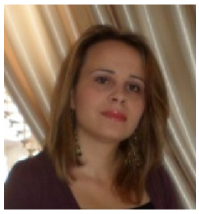
[13] A. Aalbers, *Obstacle avoidance using limit cycles*. PhD thesis, Faculty of Mechanical, Maritime and Materials Engineering, Department Delft Center for Systems and Control, 2013.

[14] B. Peasley and S. Birchfield, "Real-time obstacle detection and avoidance in the presence of specular surfaces using an active 3d sensor," *IEEE Workshop on Robot Vision (WORV)*, pp. 197–202, 2013.

[15] T. S. Hong, D. Nakhaeinia, and B. Karasfi, "Application of fuzzy logic in mobile robot navigation," *Journal of Fuzzy Logic - Controls, Concepts, Theories and Applications*, 2012.

[16] G. Indiveri, J. Paulus, and P. Plöger, "Motion control of swedish wheeled mobile robots in the presence of actuator saturation," *RoboCup 2006: Robot Soccer World Cup X*, vol. 4434, p. 35–46, 2007.

[17] D. Soetanto, L. Lapierre, and A. Pascoal, "Nonsingular path following controlof dynamic wheeled robot," *42nd IEEE Conference on Decision and Control*, 2003.



E. Baklouti was born in 1986 in Sfax, Tunisia. She graduated in 2010 from National Engineering School of Sfax, Tunisia. She received the Master degree in New Technologies of Dedicated Computer Systems discipline from National Engineering School of Sfax (ENIS), Tunisia in 2011. She received the PHD degree in Computer Science also from ENIS in 2016. Her research interests are in the area of robotics as autonomous navigation, robot security, motion control and sensors.

N. Ben Amor is an Assistant Professor at the National Engineering School of Sfax, Tunisia. He received his PhD in Electrical Engineering from both Sfax University (Tunisia) and Bretagne Sud University (France) in 2005. His research interests are Hard-ware-Software System on Chip, co-design methodology, self adaptive systems, embedded real time system, real time image processing on FPGA systems, robotics.

M. Jallouli received his DEA in Automatics from University of Valenciennes in 1986 and PhD in Robotics Engineering from University Paris XII, France, in 1991. In 1987, he then joined French University in education activities for his post- doctoral period. In April 1991, he joined the Tunisian University where he held different positions involved in both education and research activities. He is currently an Associate Professor at Higher Institute of Industrial Systems of Gabes and a Computer & Embedded System laboratory's member. His current interests include the implementation of intelligent methods (neural network, fuzzy logic and genetic algorithm) in robotic and vision system as well as in multi-sensory data fusion mobile bases.

AUTHORS' ADDRESSES

Dr. Emna Baklouti, Ph.D.
Prof. Nader Ben Amor, Ph.D.
Prof. Mohamed Jallouli, Ph.D.
Computer & Embedded System laboratory
National School of Engineers of Sfax,
Soukra Street, Sfax; 3072, Tunisie
email: baklouti.emna@gmail.com, nader.benamor@enis.tn, mohamed.jallouli@enis.rnu.tn

Received: 2015-07-23
Accepted: 2017-01-10