ARTÍCULO DE INVESTIGACIÓN
Vol. 35, No. 2, Agosto 2014, pp. 125-142

Revista Mexicana de Ingeniería Biomédica

SOMIB

Novel Fuzzy Logic Controller based on Time Delay Inputs for a Conventional Electric Wheelchair

M. Rojas
P. Ponce
A. Molina

Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Ciudad de México.

ABSTRACT

This work proposes a Dynamic *fuzzy logic* Controller for the navigation problem of an electric wheelchair. The controller uses present data from three ultrasonic sensors as the main source of information from the environment. However other inputs, named as "dynamic time delay", are obtained from past samples of those static data and are used to design the rule base. Although *fuzzy logic* controllers with static inputs could solve basic navigation problems, the proposed structure with dynamic inputs gets an excellent performance for more complex navigation problems. There were designed static and dynamic navigation strategies, which were first deployed in software just to evaluate their behavior. They were tested in a maze and their trajectories were compared to select the best. For improving its response, the dynamic *fuzzy logic* strategy was deployed in hardware. The paper presents a comparison between the software and hardware applications to illustrate the possibility of implementing the proposed methodology in different platforms. The dynamic *fuzzy logic* controller led the electric wheelchair without colliding against walls, and is a high performance navigation system. Moreover, this controller could solve the sensor limitations.

Keywords: fuzzy logic, dynamic, controller, wheelchair, ultrasonic sensors.

Correspondencia:
Mario Rojas
EGIA, Calle del Puente #222
Col. Ejidos de Huipulco,
Tlalpan C.P. 14380, México
D.F
Correo electrónico:
mario.rojas@itesm.mx

Fecha de recepción:
12 de Octubre de 2013.
Fecha de aceptación:
19 de Junio de 2014

RESUMEN

En este trabajo se presenta un controlador dinámico con lógica difusa para el problema de navegación de una silla de ruedas. El controlador usa datos presentes de tres sensores ultrasónicos como la principal fuente de información del entorno. Sin embargo, a partir de valores pasados se obtienen otras entradas designadas como "retrasos dinámicos´´ para la base de reglas. A pesar de que los controladores de lógica difusa con entradas estáticas pueden resolver problemas básicos de navegación, la estructura propuesta con entradas dinámicas tiene un excelente desempeño para problemas de navegación más complejos. Se diseñaron estrategias de navegación estáticas y dinámicas, las cuales fueron implementadas primero en software para evaluar su desempeño. Se usó un laberinto y sus trayectorias fueron comparadas para seleccionar el mejor. Para mejorar su respuesta, la estrategia dinámica fue implementada en hardware. Este artículo presenta una comparación entre las aplicaciones de hardware y software para ilustrar la posibilidad de implementar la metodología en diferentes plataformas. El controlador dinámico de lógica difusa dirigió la silla eléctrica sin colisionar contra los muros, y es un sistema de navegación de alto desempeño. Así mismo, este controlador podría resolver las limitaciones del sensor.

Palabras clave: lógica difusa, controlador, dinámico, silla de ruedas, sensores ultrasónicos.

INTRODUCTION

Word report on disability recommends the use of electric wheelchairs (EW) as assistive technology for handicapped persons [1]. Furthermore, smart electric wheelchairs could solve the mobility problem when the patients suffer strong mobility limits and cannot control the joystick. They could be assisted by a smart wheelchair which includes sensors, controllers, user interfaces and navigation modules as presented in [2] and [3].

The smart wheelchairs are classified as autonomous, semi-autonomous and hybrid systems [4]. In an autonomous wheelchair, the operator indicates it where to go, then the system plans the route and moves there without assistance. In those prototypes, the user only waits for the system to reach the specified objective without being able to choose the speed or the trajectory. Besides, those systems are limited to local and well-known environments, and they are planned to be used by patients who cannot control any device because of their disability. In the semi-autonomous prototypes, the operator and the system work together by means of some user interface, sensors and smart navigation techniques. With this approach, the patient partially needs help in certain navigation tasks but he is able to control the mobility. Tasks like obstacle avoidance, wall following, parking and door passage are typically used in this kind of smart EW as mentioned in [5], [6] and [7].

Conventional controllers are classified in two: linear and nonlinear. The linear controllers are constructed from analyzing a set of equations, which model the dynamic of the system with precision, or at least approximately. On the other hand, for nonlinear controllers the mathematical model contains uncertainties or is totally unknown because of its complex behavior (all control systems are actually nonlinear) [8].

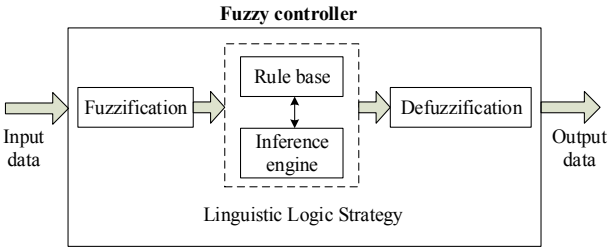


Figure 1. The fuzzy controller.

The *fuzzy logic*, introduced by Zadeh in [9], is used as a control technique for systems in which no mathematical model is known. It is complicated to find a mathematical model when the user takes navigation decisions based on vague and imprecise information, but it is possible to approximate their actions with a controller based in *fuzzy logic*. The basic topology of a Fuzzy Logic Controller (FCL) is shown in Figure 1. The inputs are crisp values, which are changed into degrees of membership between 0 and 1. The membership functions are described with linguistic labels (like Close or Far), which are very useful for constructing the rule base built with if-then structures. The input fuzzy sets are used as antecedents and the output fuzzy sets as consequents, both are connected with a fuzzy operator to determine the rule membership value. This value is used in the defuzzification process to determine the crisp output.

There are several works of FLC applied to autonomous mobile robots, for instance [10], [11] and [12]. More precisely, a review of *fuzzy logic* applications in electric wheelchairs is presented in Table 1. This technique can be used to implement obstacle avoidance or wall following algorithms for navigation of the EW in some unknown environments. For instance, in references [13], [14], [15], [16] and [17] are presented prototypes which use distance sensors as inputs for the *fuzzy logic* controller. Another cases of application are those with *fuzzy logic* as part of the user interface to get instructions from the user (i.e. the flex sensor in [18], the voice commands in [15] or the joystick operation mode described in [16]). Finally, other works use *fuzzy logic* for complementary tasks of the complete system as pairing location patterns in a map or controlling the speed of the wheelchair motors ([19] and [20], respectively).

According to the review presented in Table 1, the wheelchair performance is related with three main aspects: the sensors employed, the processing core and the implemented methodology. For the case of sensors, they are used to get distance measurements from obstacles. Ultrasonic sensors have been widely used in prototypes because of their low cost and fast responses, however they have some noise problems, which can be improved with software. Another alternative is the infrared sensors (IR), but they are limited in distance and visible light affects their performance as presented in [15]. Other systems use laser range finders, but they are not as cheap as ultrasonic sensors [5]. It is certainly true that a big number of sensors comprises more environment information for the controller, but this outcomes in a more complex control [16]. In those cases, if the processor is limited in resources the response will be slow.

The other aspect to consider is the processing core. In Table 1, all projects were implemented using microcontrollers or computers, and just one of them used real-time hardware to control the motors rotation speed [21]. In contrast, there are parallel architectures which allow data to access the resources at the same moment, and they guarantee the execution in a time period. The real-time hardware, like Field Gate Programmable Array (FPGA) has proved to be very efficient and reliable, and there are a lot of advantages about configuring a controller in the FPGA instead of using a computer or any microcontroller [22].

Finally, in relation with the implemented algorithm to avoid obstacles, the Mamdami methodology is used in [14], [18], [15], [16], [17] and [23]. Mamdani is a predominant inference technique for *fuzzy logic* controllers based on human experience. In all those prototypes, static inputs are used to compute the output of the controller (static inputs mean current time data), however more information can be obtained from the past samples. That information is known as dynamic, and can be used as extra inputs for the controller to improve the whole system performance.

Table 1. Main works developed using *fuzzy logic* for an Electric Wheelchair (EW)

Ref.	Description
[13]	Two fuzzy controllers are used: one for joining a target specified by specifying an (x, y) coordinate and the other for avoiding obstacles.
[14]	The fuzzy controller considers distance, presence and direction from the objects to decide if it necessary to change the trajectory.
[18]	It uses two fuzzy controllers, one to determine actions from the flex sensors and the other for obstacle avoidance based in ultrasonic sensors. Preference is given to the fingertip control if obstacles are far and to the obstacle avoidance system if objects are close.
[15]	It includes an obstacle avoidance control which uses IR sensors, as well as a contour following control. Both are *fuzzy logic* controllers.
[20]	Utilizes FPGA technology in a wheelchair combined with a *fuzzy logic* control designed to manipulate the rotation speed of the driving motors. It is not a navigation control.
[16]	The fuzzy controller is based in the information given by eight sonar sensors and the joystick. Inference system is based in that information to control direction and speed of the wheelchair.
[19]	The fuzzy control is focused on matching the position of a wheelchair in a sidewalk network map of an urban area, by using a GPS.
[17]	The *fuzzy logic* controller is designed to alternate between manual and automatic navigation depending of near obstacles. This assures the switching to be gradual. The automatic controller is also based in *fuzzy logic* to avoid obstacles.
[23]	The controller is used to determine the operator orders by using a seat pressure sensor and body movements as the interface. The inputs for the inference system are the x and y velocity and acceleration of human gravity center. The prototype includes omnidirectional wheels for moving in every direction.

This work proposes a dynamic *fuzzy logic* controller for an EW navigation system, which utilizes only three ultrasonic sensors. Extra information is computed from the distance measurements as additional inputs for the controller in order to get better results. Implementation was done first in software running in Windows 7, and then in the FPGA chip embedded in a cRIO 9014 to guarantee data processing in real time.

METHODOLOGY

The electric wheelchair structural design

The system described in this section is named as "The software implementation". A commercial electric wheelchair by Quickie, model P222-SE, was adapted with the hardware shown in Figure 2. The NI USB-6211 is a data acquisition module used to generate the voltages that move the EW motors. Two analog channels are used: one for forward-backward movement and another for steering left-right actions.

In addition, three Parallax PING))) ultrasonic sensors were installed in the wheelchair at different positions: front left (S1), front right (S2) and back (S3). The general information regarding the ultrasonic sensors is presented below (more information in [24]). They detect objects by emitting a short ultrasonic burst and then "listening" the echo. The sensors normally emits a short 40 kHz burst under the operation of a host digital system (trigger pulse), for example a microcontroller. This burst travels through the air, hits an object and then bounces back to the sensor. The PING))) sensor provides an output pulse to the host that will terminate when the echo is detected, hence the width of this pulse corresponds to the distance to the target. The principle of operation of these sensors is shown in Figure 3.

The microcontroller block is a Basic Stamp 2 (BS2-IC), a 20 MHz speed processor made by Parallax. This microcontroller acquires data

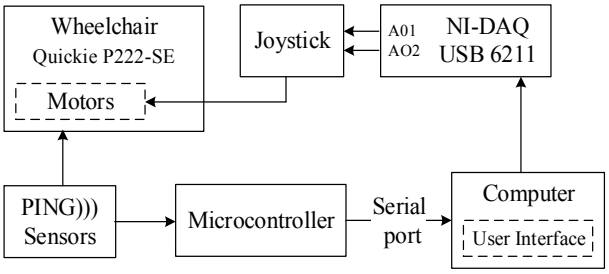


Figure 2. Components of the wheelchair system implemented in LabVIEW

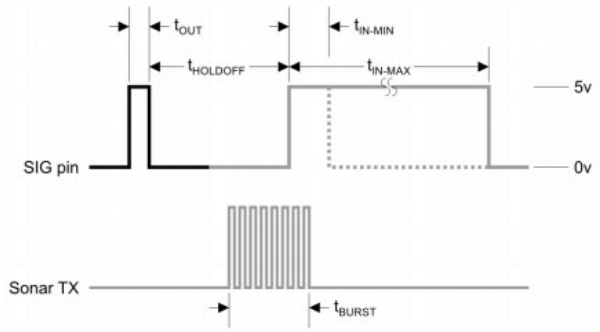


Figure 3. Figure 3. Parallax PING))) ultrasonic sensors principle of operation described in their manual [33].

from the ultrasonic sensors, converts it into distance measurements and sends that information via serial communication to the interface hosted in a laptop. The microcontroller is configured to receive distance samples every 100 ms from the sensors. Finally, the interface to operate the electric wheelchair was programed in LabVIEW 2013. It receives distance data from the BS2-IC and sends voltage operation values to the acquisition module 6211. This interface allows the user to move the wheelchair with virtual controls and to observe the measured distances to objects (in centimeters). Furthermore, the *fuzzy logic* controller (FLC) was integrated to execute automatic actions based in those measurements.

Navigation scenarios analysis

The wheelchair must move in any environment with static objects like walls, doors, hallways; and dynamic objects, which suddenly appear like a person walking. When an object is detected in the path, the controller computes which movement or steer action is going to

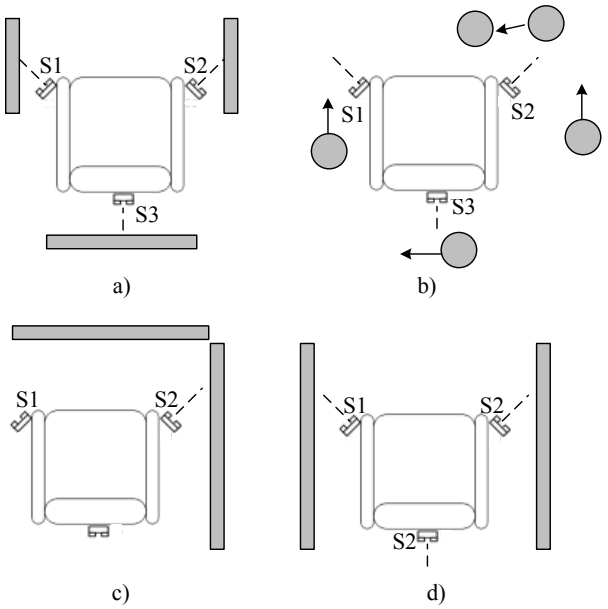


Figure 4. Navigation scenarios a) static obstacles, b) dynamic obstacles, c) turning corners, d) Straight navigation.

be performed. It is desirable that in every configuration, the system should go forward in a straight route avoiding obstacles. In Figure 4 are presented four configurations to analyze how the system behaves, which inputs are considered and what actions are needed to do in every case. With this analyses is determined the rule base for the fuzzy controller.

The configuration indicated in Figure 4.a. shows the sensors S1, S2 and S3 blocked by objects at a distance considered "close". Consequently, the action is to steer left or right to avoid the blocking object. The second scenario presented in Figure 4.b. shows dynamic and static obstacles moving either around the sensors S1 or S2. When an object appears suddenly, the EW must avoid crashing with it. The third configuration presented in Figure 4.c. shows if there is a steering action that must be carried out for a long time to turn over a corner (the blocked sensor stills in that same state until the corner is over). Finally, the last navigation case is a straightforward trajectory observed in Figure 4.d. It is desirable that the wheelchair moves in the middle of a hallway, and maintain same distance between left and right walls.

Fuzzy logic navigation strategies

Three *fuzzy logic* controllers were designed after the analysis of the configurations. Figure 5 shows the strategies proposed for navigation.

Strategy-A. It uses as inputs the distance measurements from left, right and back sensors. The idea is simple, when a sensor is blocked the controller calculates a direction to steer. Observe that these inputs are static because they are the current data taken from sensor.

Strategy-B. It uses the same logic as in Strategy-A, but additionally it considers past samples from sensors S1 and S2 as inputs to detect dynamic objects. The inputs labeled as dS1/dt and dS2/dt are defined as delayed data, thus s1 is the current distance and dS1/dt is the last past value obtained. In addition, this strategy uses as an input the arithmetic mean of 16 samples collected from steering past actions (D output) performed by the controller.

Strategy-C. This controller is based in the previous strategies, but it includes another input to make straighter trajectories. This input is obtained by subtracting S2 from S1. If this input is included, the EW tries to stay at the center of the path. 12 rules were proposed for this strategy, the next cases are described next:

- Rule 1, 2, 3. Left, right and back sensors are completely blocked.
- Rules 4, 5. Chair is blocked in one side, left or right.
- Rules 6, 7, 8. Chair is blocked in both sides simultaneously
- Rules 9, 10, 11. All sensors are in the "far" set.
- Rule 12. An object appears suddenly.

The complete rule set is shown in Table 2.

Variables are defined in terms of fuzzy sets termed as:

S1, S2, S3 → C (Close), F (Far)

ds1, ds2 → GF (Getting far)

S → P (Positive), Z (Zero), N (Negative)

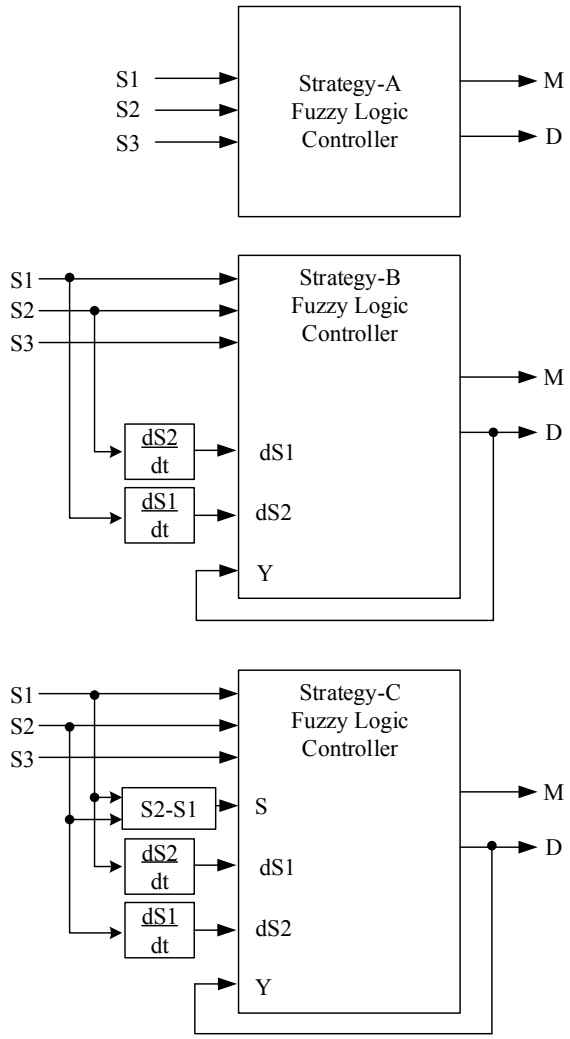


Figure 5. Fuzzy controller structures designed for approaches A, B and C

M → N (Negative), MF (Medium Fast), B (Backward), F (Forward), MF (Middle Forward)

D → L (Left), ML (Medium Left), N (Negative), MR (Medium Right), R (Right)

Y → TR (Turning Right), TN (Turning Null), TL (Turning Left)

Input variables description

The fuzzy sets used for every variable are described next:

Table 2. The software implementation rule set (Strategy-C)

1	$s : N \cap s_1 : C \cap s_2 : C \cap s_3 : C \Rightarrow M : N \cap D : N$
2	$s : Z \cap s_1 : C \cap s_2 : C \cap s_3 : C \Rightarrow M : N \cap D : N$
3	$s : P \cap s_1 : C \cap s_2 : C \cap s_3 : C \Rightarrow M : N \cap D : N$
4	$s : N \cap s_1 : F \cap s_2 : C \Rightarrow M : MF \cap D : L$
5	$s : P \cap s_1 : C \cap s_2 : F \Rightarrow M : MF \cap D : R$
6	$s : N \cap s_1 : C \cap s_2 : C \cap Y : TR \Rightarrow M : B \cap D : R$
7	$s : Z \cap s_1 : C \cap s_2 : C \cap Y : TN \Rightarrow M : B \cap D : N$
8	$s : P \cap s_1 : C \cap s_2 : C \cap Y : TL \Rightarrow M : B \cap D : L$
9	$s : N \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : ML$
10	$s : Z \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : N$
11	$s : P \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : MR$
12	$ds_1 : GF \cup ds_2 : GF \Rightarrow M : MF \cap D : N$

Where $\cap = T_m = \min(x, y)$.

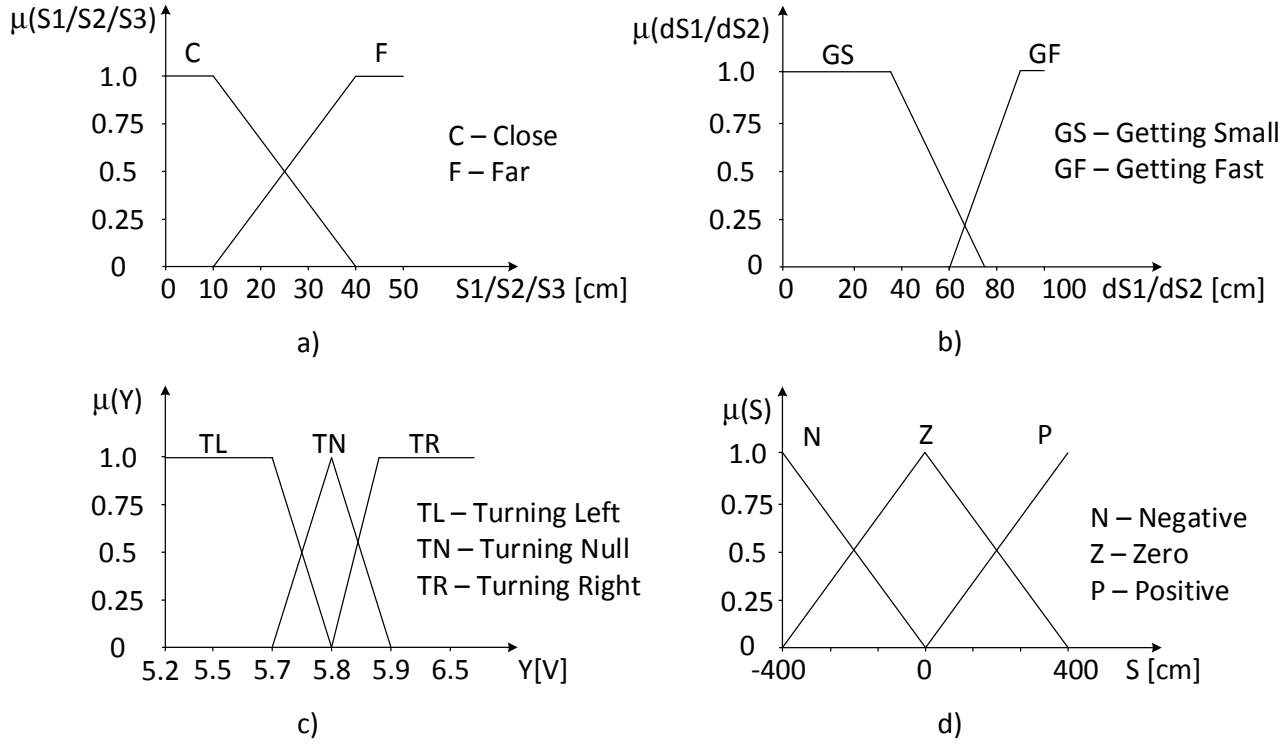


Figure 6. Fuzzy Input definitions and memberships functions a) distance, b) distance differential, c) past steering action and d) sensor difference.

Distance. This variable is defined with two fuzzy sets: close ("C") and far ("F") and is specified for S1, S2 and S3 sensors. The distance range of these inputs was considered as much as necessary to avoid collisions as shown in Figure 6.a.

Distance differential. These inputs were calculated from S1 and S2. The "dS1" and "dS2" inputs are defined by two fuzzy sets: getting fast ("GF") and getting slow ("GS"). They are useful for the system to take decisions by considering the approaching of dynamic objects to the wheelchair. Membership functions of these inputs are presented in Figure 6.b.

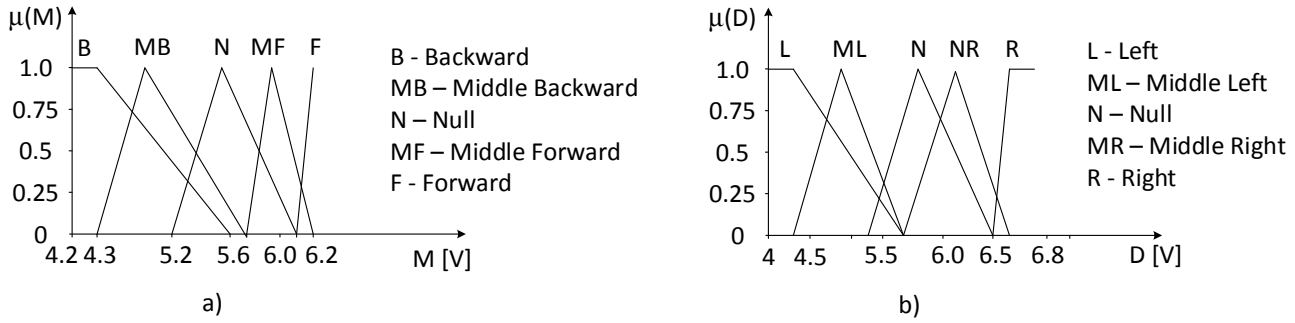


Figure 7. Fuzzy outputs definition a) Movement, b) Direction outputs

Past steering action. It is defined from the collected information about the past steering values that indicate an action performed for a long time. This input named "Y" is obtained from the "D" output and is defined with 3 membership functions as shown in Figure 6.c: turning left, turning null and turning right ("TL", "TN" and "TR", respectively).

Sensor difference. It is obtained by subtracting S1 from S2 and determines if the wheelchair is deviating negatively, positively or zero ("N", "P", and "Z"). If the difference is negative, the wheelchair steers to the left side; if positive, steers to the right side of the reference. Membership functions are shown in Figure 6.d.

Output variables description

Output variables indicate the movement or steering action of the wheelchair: forward, backward, left or right. The obtained values are defuzzified into analog voltages. Figure 7 shows the sets definition for these outputs. Their ranges are adjusted to the functional voltages for moving the motors and they are not symmetrical.

Movement. The "M" output corresponds to analog voltage channel 1, and it is defined by five fuzzy sets named Backward, Middle Backward, Null, Middle Forward and Forward ("B", "MB", "N", "MF", "F"). These five membership functions allow the system to go backward or forward in different speeds.

Direction. This output (labelled as "D") activates analog channel 2 and is defined by five sets named Left, Middle Left, Null, Middle Right, Right ("L", "ML", "N", "MR", "R").

Static and dynamic fuzzy controllers

Navigation circumstances presented above could be used to define static and dynamic *fuzzy logic* controllers. A static FLC operates with the current sensor data to obtain the outputs (Strategy-A), but a dynamic FLC considers current and past values from sensors to obtain the outputs (Strategies B and C). A study of a static controller behavior is presented below in order to see how the information from the past is not affecting the firing rules. It was used the proposed Strategy-C in this analysis.

The study case has the following conditions: there are objects blocking the sensors S1 and S2, approaching at different speeds from a distance considered far. This is illustrated in Figure 8.

If the controller has a set of fixed linguistic rules (as those in Table 2) and it is assumed that the rules (10, 9, 7 and 4) are affected for specific inputs. The firing strength graphs obtained in this case study are shown in Figure 9.

The firing strength shows how the rules change according to the movement of the EW. The Speed response is presented in Figure 10.a. which shows actions executed. In the first configuration, distance registered in sensors S1 and S2 decrease at the same rate. In the velocity graph as the distance becomes small, forward speed is needed to slow down to avoid collision up to the moment it changes direction to backward. Meanwhile, in the angular velocity response no change in direction is registered. However, for the second response presented in Figure

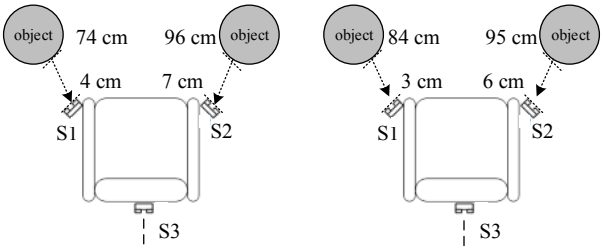
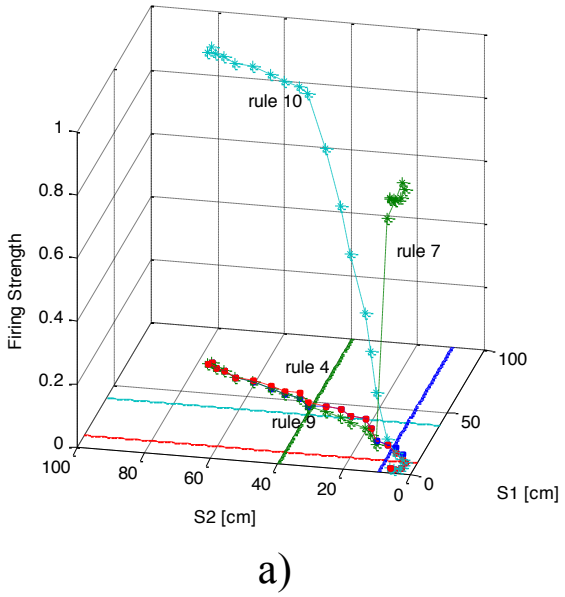
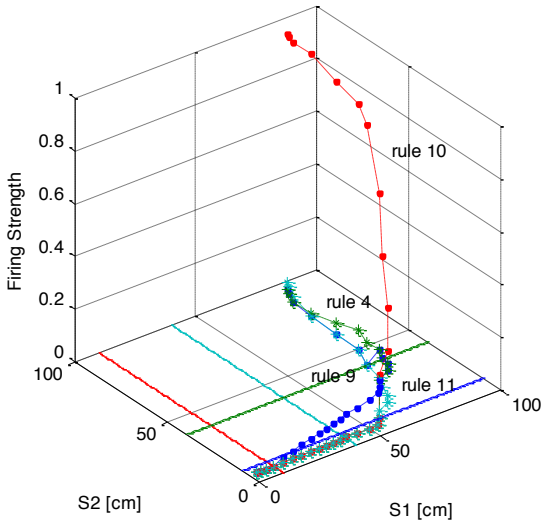


Figure 8. Case Study regarding the configuration when both sensors change values (a) at same speed and (b) S1 changes faster than S2.

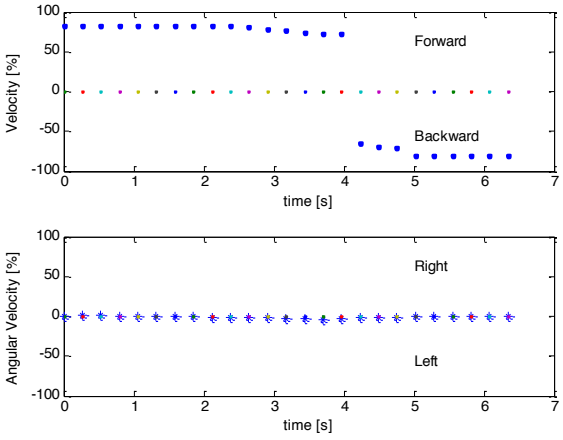


a)

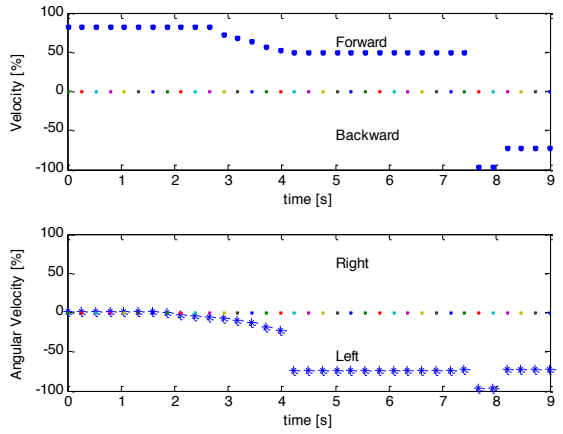


b)

Figure 9. Firing strength graphs for conditions (a) obstacles moving at the same speed (b) obstacles moving at the different speed.



a)



b)

Figure 10. The velocity graphs for configurations (a) obstacles moving at the same speed (b) obstacles moving at different speed.

10.b. corresponding to the other configuration, forward speed decreases slowly until it changes to backward when both sensors are completely blocked. Because S2 arrives first at the close region, a left angular velocity is registered.

The FPGA controller implementation

In fact, the controllers implemented in software platform cannot operate under deterministic processing time [25]; hence, the processing cycles running on LabVIEW cannot be greater than milliseconds and the real time applications which need deterministic time do not use a software platform.

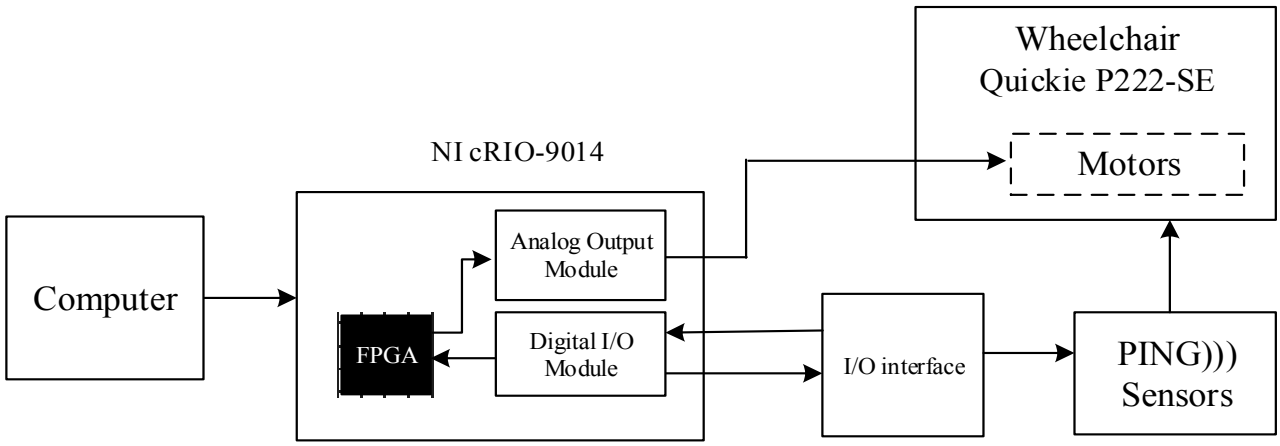


Figure 11. Components of the wheelchair system implemented in the FPGA.

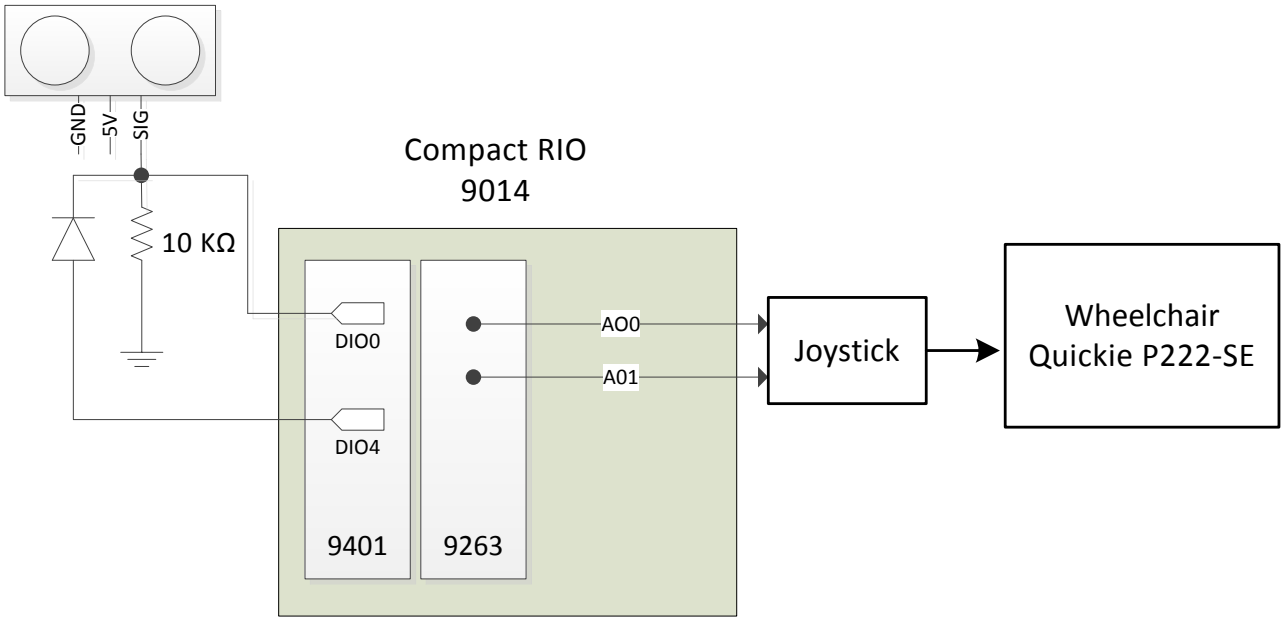


Figure 12. Digital I/O and analog output modules configuration.

For the EW application is very important to ensure that the system will execute without interruptions or possible operating system failures. In addition, it is necessary to have a very fast response because a person's integrity depends on it. Hardware designed controllers can solve the mentioned drawbacks of the software implemented ones. Frequently, FPGAs are used because they are accessible in different locations as embedded systems, and because of their processing characteristics the speed range of nanoseconds can be reached for the operating cycles. If the FPGA is used, the information is processed inside the chip and the computer is required only for setting the initial conditions of the FCL, thus no operating system interruptions appear. Based in those advantages, it was proposed an alternative version of the system named as "The hardware implementation" which components are shown in Figure 11.

It was used a NI Compact-RIO (c-RIO) 9014 to implement a deterministic real-time system. The c-RIO combines the real-time approach and reconfigurable FPGA technologies in the same device for embedded control, data acquisition and analysis. This device supports interchangeable modules for I/O to access data to the Spartan-3 Xilinx chip with 3

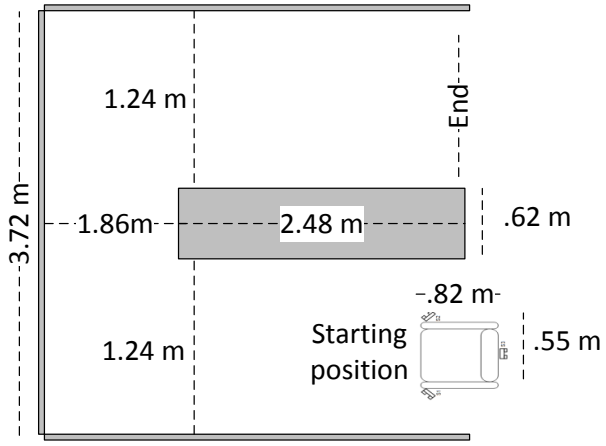


Figure 13. Maze test scenario.

million equivalent gates, besides it integrates a 40MHz clock. In this hardware implementation the ultrasonic sensors are connected directly to the device, thus the processing time is reduced because it is not necessary a serial communication port as in the software system. For all these reasons, the hardware implementation is expected to provide better results.

Only 2/4 analog output channels from the NI C-Series 9263 module and 6/8 high speed digital I/O from the NI 9401 C-Series module were used. DIO0-DIO3 were configured as digital inputs and DIO4-DIO7 as outputs. The interface uses a diode and a resistance to implement a bidirectional ultrasonic line in the NI 9401 module as shown in Figure 12. As explained with the microcontroller, the FPGA implementation sends a pulse to the ultrasonic sensor and waits to receive the response. It is used the same sampling time as in the software implementation: 100 ms. The 9263 analog output module is used for sending control voltage (channels AO0 and A01) to the wheelchair's joystick, in the same way the NI-DAQ9611 does in the software implementation.

Control strategies test and validation

A maze was designed for validating the proposed controllers under different navigation conditions; all the dimensions of the maze are presented in Figure 13. The target of the electric wheelchair is to navigate from the initial point to the final one without colliding against the walls.

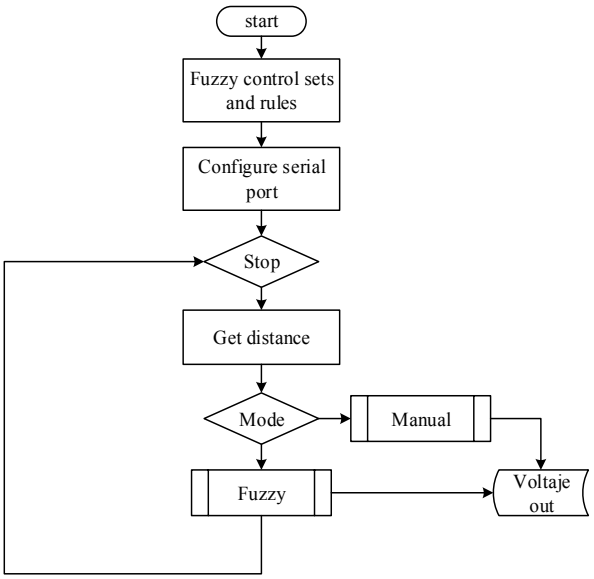


Figure 14. Flux diagram for software controller implementation.

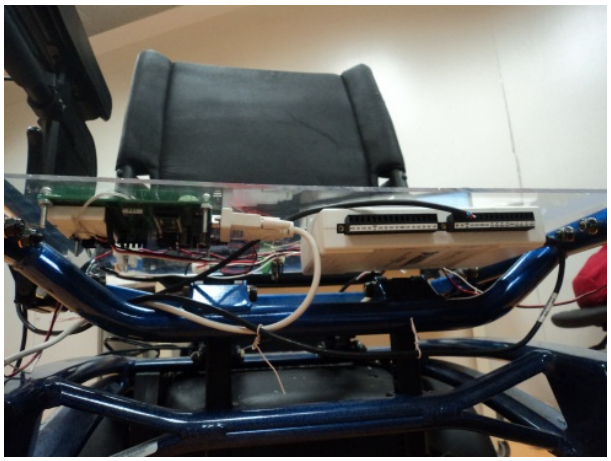


Figure 15. Installed components for the software version

Notice that the scenario has right angle corners, and for security matters flexible walls were used. All the experiments were performed with the same start position. The strategies A, B, C implemented in software and Strategy-C implemented in hardware were tested in this maze.

RESULTS

Software implementation

For the software implementation, the FCL strategies were realized with the "PID and fuzzy

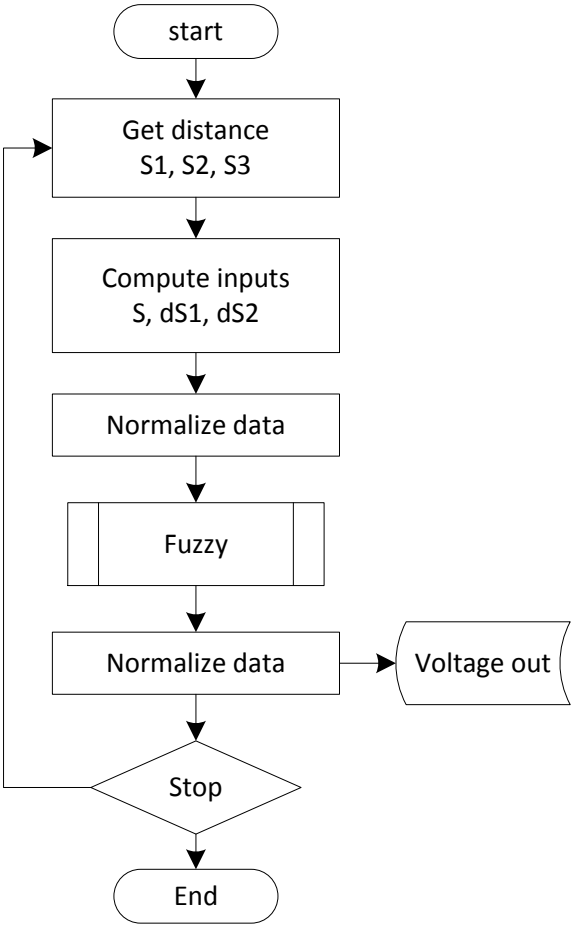


Figure 16. *fuzzy logic* controller block diagram

implemented in the FPGA.

logic Control toolkit" in LabVIEW 2013. The control was integrated to the LabVIEW interface as presented in the flux diagram shown in Figure 14.

In Figure 15 are presented the components assembly under the EW seat for the software implementation.

The hardware implementation

Apart from the software version, the hardware implementation is described. Tasks done by the real-time controller are indicated in Figure 16 and they were programmed in the LabVIEW FPGA toolkit. The sensors distance to objects are obtained and with those data other inputs are computed: dS1, dS2, S. Numerical values are normalized to fit the fixed point format used by the fuzzy controller for the decision making. Obtained outputs are de-normalized to fit useful voltages for the EW.

Variable Y is not considered because S variable helps the controller to approach the curves better. The rule set for the FPGA implementation is shown in Table 3.

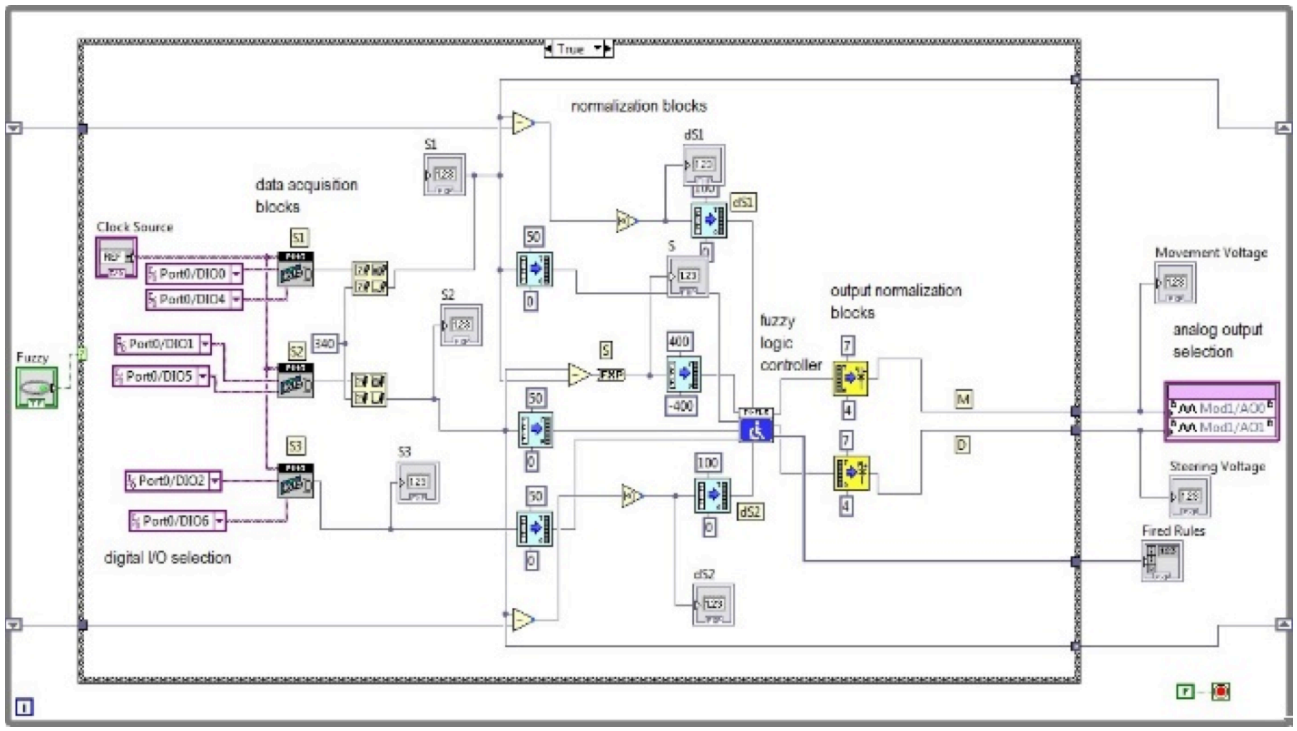


Figure 17. Fuzzy logic code programmed with LabVIEW FPGA toolkit.

Table 3. The FPGA implementation rule set

1	$s_1 : C \cap s_2 : C \cap s_3 : C \Rightarrow M : N \cap D : N$
2	$s : N \cap s_1 : F \cap s_2 : C \Rightarrow M : MF \cap D : L$
3	$s : P \cap s_1 : C \cap s_2 : F \Rightarrow M : MF \cap D : R$
4	$s : N \cap s_1 : C \cap s_2 : C \Rightarrow M : B \cap D : R$
5	$s : Z \cap s_1 : C \cap s_2 : C \Rightarrow M : B \cap D : N$
6	$s : P \cap s_1 : C \cap s_2 : C \Rightarrow M : B \cap D : L$
7	$s : N \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : ML$
8	$s : Z \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : N$
9	$s : P \cap s_1 : F \cap s_2 : F \Rightarrow M : F \cap D : MR$
10	$ds_1 : GF \cup ds_2 : GF \Rightarrow M : MF \cap D : N$

Table 4. Consumed resources with the system

Functional Block	Logo	Total slices	Slice registers	Slice LUTs
T1 Wheelchair Control	NA	9794	9562	14888

Figure 18. NI Compact-RIO installed for the real-time system.

In Figure 17 is presented the configured fuzzy controller code created on LabVIEW FPGA toolkit, constructed with fixed point operations and configured as hardware into the FPGA chip. There have been labeled five different parts: digital I/O port and line selection for sending/receiving data from ultrasonic sensors, normalization blocks to scale signals in useful ranges for the fuzzy controller, the fuzzy controller block (which contains the membership functions, the inference engine and the rules base), the output normalization blocks for values computed, and finally, the analog

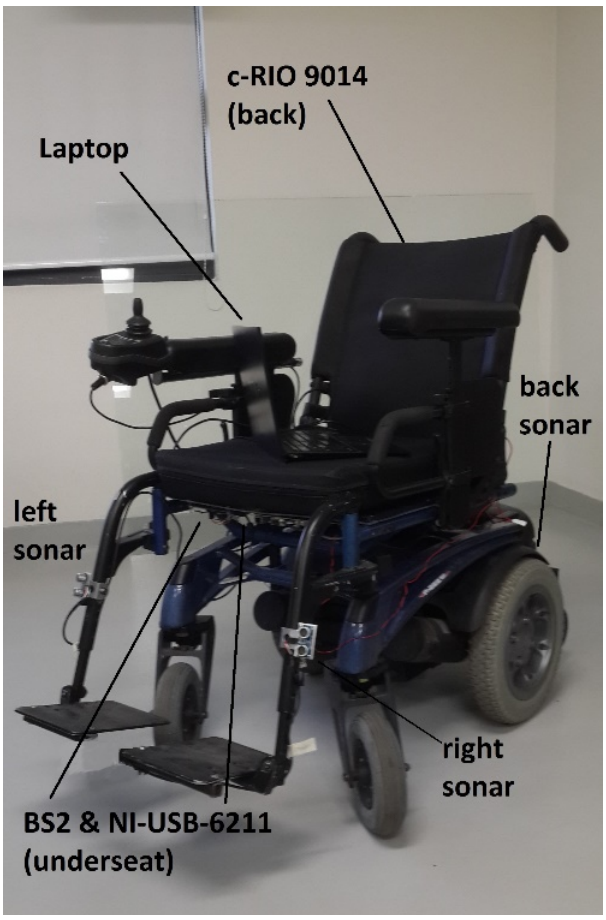


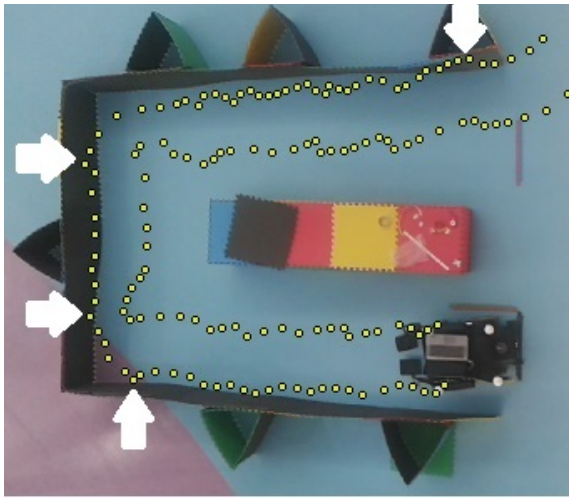
Figure 19. The complete system. For the hardware implementation, the laptop is only used for setting the controller initial conditions and to register data.

output channels selected to supply voltage for movement and steering actions between 4 and 7 volts. After the compilation into the FPGA, the consumed resources shown in the summary with this configuration is shown in Table 4.

Moreover, in Figure 18 is presented the c-RIO installation which is online with the PC in the hardware implementation. In figure Figure 19 is presented the complete wheelchair system.

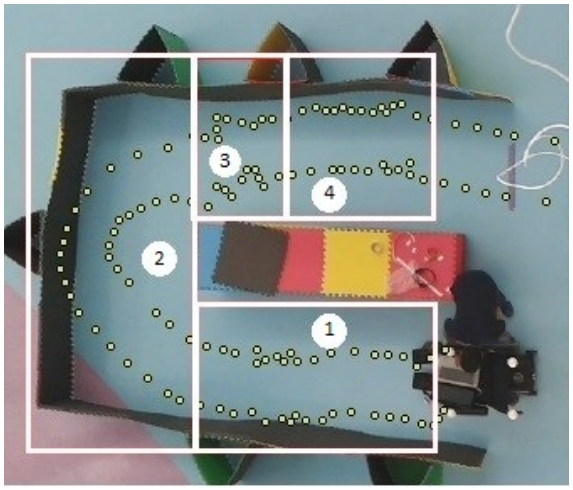
The maze test validation for the software version

The three strategies A, B, and C implemented in LabVIEW were tested, but only the third one was completely successful. Figure 20 presents the observed trajectories for the test. Images were taken from an upper view and because of that perspective some walls look wider.



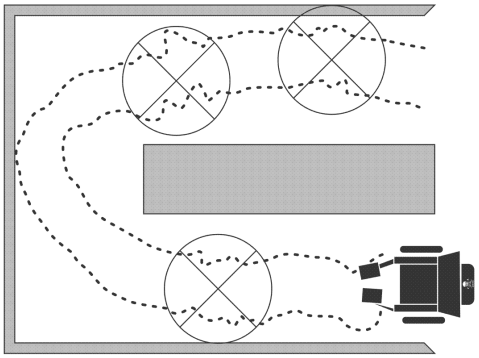
a)

b)

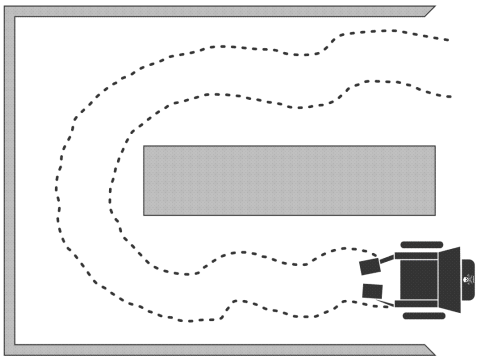


c)

Figure 20. Obtained trajectories in test scenario. Dots represent lateral sensor position during the route, a) Strategy A, b) Strategy B, c) Strategy C.



a



B

Figure 21. Hardware and software trajectory tracking comparison for the Strategy-C using, a) Computer device b) Compact-RIO.

By using strategy-A, the wheelchair crashed four times as indicated with arrows in Figure 20.a. and it was very close to the left wall, however it finished the maze in 1.26 minutes. Strategy-B did not finish the maze because wheelchair got stocked in the first corner as can be observed in Figure 20.b. The third trajectory corresponds to strategy-C, which was completed in 1.10 minutes without colliding. Four zones are labelled in Figure 20.c. as "1", "2", "3" and "4" to analyze them. It seems that in the middle of the curve (zone 2) there was a collision, but it is only a perspective effect.

Software and hardware implementations

Figure 21 shows the results trajectories observed in the hardware and software implementations. It was compared the Strategy-C implemented in software and the strategy designed for the hardware in the maze test.

DISCUSSION

Static and dynamic controllers comparative

As presented in Figure 20, the Strategy-C was the only one that completed the maze without colliding. The differences between navigation strategies implemented and the considerations done about dynamic and static controllers are remarkable. A comparison between Strategy-A and Strategy C shows that the last one results more efficient in time. Further, as presented in Figure 20.c., in the zone labeled as "1" there were oscillations caused by the wheelchair approaching to the left wall and the controller action trying to correct its trajectory. In zone "2", a continuous soft curve to turn is described contrasting the actions observed in strategy-A (Figure 20.a.), where it is notorious that for the same curve the steering actions are more complicated. In zone "3", there are more oscillations caused by the slow response of the system processing data in software. When the computer takes a specific action at some time instant, another obstacles is detected by sensors. In addition, when the computer sends data to the motors they react after some time. This phenomenon is repeated several times until stabilizes. Finally, in zone "4" the trajectory stabilizes and only one abrupt controller correction is noted.

Other differences between the observed trajectories are caused by the dynamic inputs considered in the Strategy-C, which are designed to help the controller in tasks as turning in a curve. For the static controller implemented with Strategy-A, it is distinguished a "squared" turn, but for the same zone the dynamic controller uses data collected from past actions to make decisions. This paper does not show all the possible devices in which the controller could be deployed, but it analyzed the performance of the proposed controller in order to validate it. Normally, micro-controllers are chipper than FPGAs, so it is a very attractive possibility to implement this controller using micro-controllers.

Comparative between real-time and software versions

The hardware version describes smoother trajectories and continuous movements, which are better in contrast with the abrupt movements obtained with the software implementation. The uncertainties exhibited in the marked regions of Figure 21.a. do not occur in Figure 21.b. and the described curve is smoother. In this test, the measured time to complete the maze was 20 seconds, which is really fast compared to that obtained in software Strategies A and C (1.26 and 1.19 seconds, respectively).

Table 5. Comparison table between hardware and software implementations

Characteristic	Software implementation	Hardware implementation
Trajectories	Rough, abrupt	Smooth, clean
Operations cycle rate	500 ms	100 ms
Operative system	Windows 7	None
Sensors	3 ultrasonic Parallax PING)))	3 ultrasonic Parallax PING)))
Sensors sample time	100 ms	100 ms
Input acquisition device	Microcontroller BS2-IC @ 20MHz	9401 digital inputs module
Output acquisition device	NI USB 6211	9263 analog outputs module
Maze time consumed	1.19 sec	20 sec
Number of rules	12	10
Processor	Intel Core @ 2.4 GHz	Spartan-3 Xilinx @ 40 MHz
Data Communication to the computer	Serial	TCP/IP (just for data sharing)

Those differences between both implementations are remarkable. It is explained because the hardware version uses a dedicated processor to acquire and process data that do not depend on any operating system. The target processor is networked to a host PC only for the graphical interface and data logging. In Table 5 is presented a comparison.

Sensors

As reviewed in the datasheet, the T_{burst} is 200 μs and the maximum echo return pulse is 18.5 ms for the maximum distance, $t_{holdoff}$ is 740 μs and tout is 2 μs. Consequently, the fastest time in the process of measuring data is calculated as:

$$5\mu + 750\mu + 18.5ms = 19.255ms$$

This sample time is very slow even for the software version, and it limits the controller speed response. The acquisition cycle for the software and the hardware versions is fixed to 100ms. However, in the software version distance data is passed from the microcontroller by a serial communication to the computer and, after calculating the outputs, the numerical result goes to the 6211 module. This recurrent process (indicated in Figure 14.) consumes 500 ms. Meanwhile in the FPGA version, the analogous process indicated in Figure 16 consumes 101 ms. Since sampling time for acquiring distance is 100 ms, then only 1.7 ms are used by the fuzzy controller. Comparing consumed time in the hardware and software versions, it is remarkable that FPGA is superior. Besides, the FPGA implementation could process data faster but it is limited by the ultrasonic sensors response speed.

In order to work properly, the blocking obstacles must be in front of the sensors sight line to be detected because they are strictly directional. However, the use of the dynamic inputs increase their performance for avoiding static obstacles.

CONCLUSIONS

Novel dynamic fuzzy logic navigation strategies were proposed and evaluated using an electric wheelchair. Although the ultrasonic sensors provide limited information regarding the navigation environment, the fuzzy logic controllers work properly because the dynamic information (time delay inputs) about the navigation system was included in the linguistic rules. The dynamic controllers do not change the conventional structure of a fuzzy logic controller but they modified the quality of the information about the navigation environment by adding input with delays. The main goal of this controller is to extend the input information using time delay signals, hence the controller is able to find the correct solution using limited input information.

Initially, a study of the navigation performance on software of each controller was presented in order to implement in real time the best navigation controller. The implementation based on hardware reaches excellent results and the electric wheelchair movements are flatter than movements implemented on software. Since the FPGA implementation of the dynamic controller shows reduction in time response, good avoiding obstacles performance and less sever movements, this is the best option to implement a dynamic controller for an electric wheelchair.

One of the main limitations of the controller are the blind points, caused by the number of sonar sensors used (only two of them provide information about the forward navigation). Adding sensors could expand the information from the environment of the actual prototype. Besides, it is a good idea to extract dynamic inputs from the new sensors. Although the dynamic controller increases the navigation performance, the number of fuzzy rules and membership functions will be more and the tuning process will be more complex. It is recommended to use an optimization method, i.e. genetic algorithms. On the other hand, the electric wheelchair controller is not robust to noisy signals, so it is recommended to use an adaptive filter and sensor signal estimator.

In order to have more information about the quantitative performance of the prototype, other issues could be evaluated. For example: the consumed time to solve alternatively mazes, the necessary distances for detection between the mobile objects and the wheelchair, the response to materials and composition of different objects and the behavior of the dynamic navigation strategy in small space scenarios.

REFERENCES

1. World Health Organization, "World report on disability", The World Bank, 2011.
2. D. Ding and R. A. Cooper, "Electric-Powered Wheelchairs, A Review of Current Technology and Insight into Future Direction", IEEE Control System Magazine, vol. 25, no. 2, pp. 22-34, 2005.
3. C. Urdiales, Collaborative Assistive Robot for Mobility Enhancement (CARMEN), Malaga: Springer , 2013.
4. R. C. Simpson, "Smart Wheelchairs: A literature review", *Journal of Rehabilitation Research & Development*, vol. 42, no. 4, pp. 423-436, 2005.
5. P. Boucher, "Design and validation of an intelligent wheelchair towards a clinically-functional outcome", *Journal of Neuroengineering and Rehabilitation*, vol. 10, 2013.
6. S. P. Levine, "The NavChair Assistive Wheelchair", *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 4, 1999.
7. L. Conde, G. Pires and U. Nunes, "A behavior based fuzzy control architecture for path tracking and obstacle avoidance", in *Proceedings of the 5th Portuguese Conference on Automatic Control*, 2002.
8. J. J. Slotine and L. W., *Applied Nonlinear Control*, New Jersey: Prentice Hall, 1991.
9. L. Zadeh, "Fuzzy sets *," Information and Control, vol. 8, no. 3, p. 338-353, 1965.
10. H. A. Hagras, "A Hierarchical Type-2 *fuzzy logic* Control Architecture for Autonomous Mobile Robots", *IEEE Transactions on Fuzzy Systems*, vol. 12, nº 4, 2004.
11. S.-L. El-Teleity, "Fuzzy logic control of an autonomous mobile robot", de *Methods and Models in Automation and Robotics (MMAR)*, 2011 16th International Conference on, Miedzyzdroje, 2011.
12. K. D. a. S. T. G.P. Moustris, "Feedback Equivalence and Control of Mobile Robots Through a Scalable FPGA Architecture", de *Recent Advances in Mobile Robotics*, InTech, 2011, pp. 401-426.
13. M. Njah and M. Jallouli, "Wheelchair Obstacle Avoidance Based on Fuzzy Controller and Ultrasonic Sensors", in *International Conference on Computer Applications Technology* (ICCAT), Sousse, 2013.
14. G. Liu, "Fuzzy Controller for Obstacle Avoidance in Electric Wheelchair with Ultrasonic Sensors", in *International Symposium on Computer Science and Society*, Kota Kinabalu, 2011.
15. G. Pires and U. Nunes, "A Wheelchair Steered through Voice Commands and Assisted by a Reactive Fuzzy-Logic Controller. Journal of Intelligent and Robotic Systems", *Journal of Intelligent and Robotic Systems*, vol. 34, no. 3, pp. 301-314, 2002
16. H. R. Moslehi, "Design and Development of *fuzzy logic* Operated Smart Motorized Wheelchair", in 24th Canadian Conference on Electrical and Computer Engineering (CCECE), Niagara Falls, Canada., 2011.
17. I. Spacapan, J. Kocijan and T. Bajd, "Simulation of fuzzy-logic-based intelligent wheelchair control system". *Journal of Intelligent & Robotic Systems*, vol. 39, no. 2, pp. 227-241, 2004.

18. V. Tyagi, N. Gupta and P. Tyagi., "Smart wheelchair using fuzzy inference system", in *Global Humanitarian Technology Conference: South Asia Satellite* (GHTC-SAS), 2013.

19. M. Ren and K. H.A., "A fuzzy logic map matching for wheelchair navigation", *GPS solutions*, vol. 16, no. 3, pp. 273-282, 2012.

20. M. Poplawski and M. Bialko., "Implementation of parallel fuzzy logic controller in FPGA circuit for guiding electric wheelchair", in *Conference on Human System Interactions*, 2008.

21. P. Marek and M. Bialko, "Implementation of *fuzzy logic* Controller in FPGA Circuit for Guiding Electric Wheelchair", in *11th International Conference, ICAISC* 2012, Zakopane, Poland, 2012.

22. K. Parnell and R. Bryner, "Comparing and Contrasting FPGA and Microprocessor System Design and Development", 21 July 2004. [Online]. Available: http://www.xilinx.com/.

23. J. Songmin and e. al., "Multimodal intelligent wheelchair control based on fuzzy algorithm", in *International Conference on Information and Automation* (ICIA), 2012.

24. Parallax Inc., "Product documentation for the PING))) Ultrasonic Distance Sensor", 11 9 2009. [En línea]. Available: http://www.parallax.com/sites/default/files/downloads/28015-PING-Documentation-v1.6.pdf.