Real-time Fuzzy Image Processing Method for Visual feedback of a Follower Robot

MOHSEN DAVOUDI, AFSHIN HEIDARIAN
Department of Electrical Engineering
Imam Khomeini International University
University Blvd., Qazvin,
IRAN

Abstract: - In this paper, an adaptive visual feedback system and controller has been designed and implemented in real-time to control the movements of a line follower robot to be smoother and faster. The robot consists of a couple of motorized wheels, the real-time controller and a CMOS camera as the only sensor for detecting line and feedback. The measurement based on real-time image processing and motor drive feedback used in this robot makes it robust to the obstacles and surface disturbances that may deviate robot. The image processing algorithm is adaptive to the line's color and width too. Image processing techniques have been implemented in real-time to detect the line in the image frame and extract the necessary information (like line's edge, coordinates and angle). A NI myRIO module is used as a stand-alone hardware unit and RT (Real-Time) target for implementation of controllers and image processing in LabVIEW environment. Both results of real-time and non-real-time implementation of controllers have been compared. To show the performance of real-time image processing in the control of this robot, three types of controllers (i.e. P, PI and Fuzzy controllers) have been implemented for line following tests and the results have been compared. At the end, it was found that the fuzzy controller controls the robot movements smoother, faster, with less errors and quicker response time compare to the other controllers.

Key-Words: - Real-time, Image Processing, Fuzzy Logic, Visual Feedback, Angle measurement, Follower Robot

Received: October 18, 2019. Revised: February 13, 2020. Accepted: February 29, 2020. Published: March 15, 2020.

1 Introduction

In the past decades, as the robotics became more popular, the robots have been divided into the following categories in terms of application and movements: Manipulators, wheeled robots, flying robots and human-like robot [1]. Line follower robots are robots capable of tracing a colored line possibly different to the background color. Most of the line followers are using a number of light-sensitive sensors to detect the line [2]. Most of the time the accuracy of measurement by using light-sensitive sensors is low because the accuracy is affected by poor reflectivity of the ground plane and the noisy light and some nonlinearities. On the other hand, the high number of light-sensitive sensors with special arrangement causes the robot to be bigger in size and less ability to maneuver. Machine vision has been developed to solve this problem, but the need for large process capacity for real-time image processing has been the bottleneck of this approach. Improving the real-time image processing methods for visual feedback and visual servoing in robotics are now an essential in control of the robots The software for simulation and design of real-time visual feedback and motor control algorithms is LabVIEW that provides a graphical programming environment which is broadly used for different applications in industry, education, training and laboratory research [3].
Using image processing instead of light sensitive sensors to detect the line gives the robot this capability to choose and follow the lines with targeted colors and gives this possibly to change the robot's path by changing the desired line's color or by changing the target color in software. Also it would be possible to have a line with different colors in order to control the robot's speed using a simple controller. This would be useful in the applications that the line's color indicates the speed limit or any caution.
The real-time control algorithm runs on a NI myRIO board which has the capability to run the programs in real-time. The aim of writing the program in real time is to manage the tasks and hardware by giving the priority to each tasks in order to reduce the robot fails while following the targeted line by maximizing the frames to be taken and processed [4]. The experimental results show that among three types of

controller which all have been implemented in real-time, the fuzzy logic-based controller has smoother movements while the overshoot and oscillation of the robot angular degree has been minimized.

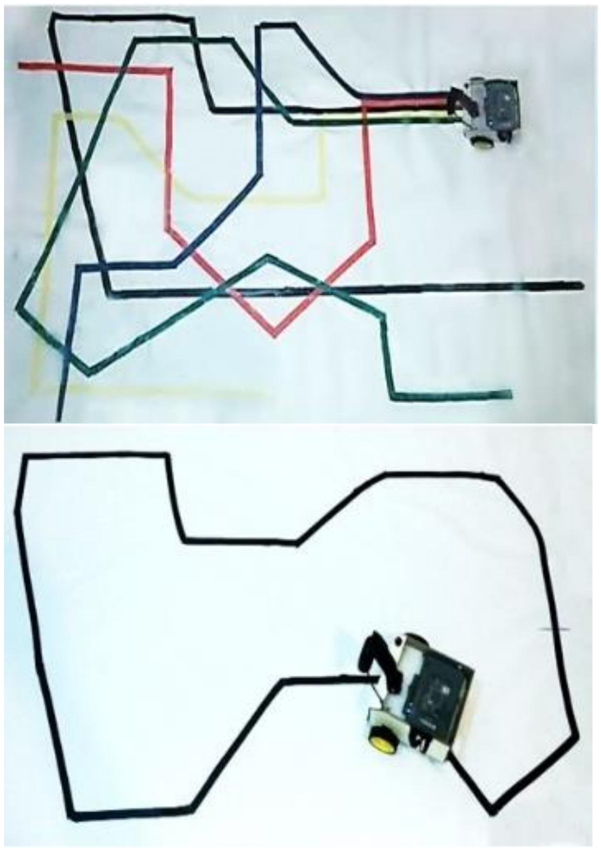


Fig.1: Top)The different paths which have been distinguished by their color. Five colors green, blue, yellow, red and black have been used in this sketch. Bottom) Robot's plane for testing.

2 Real-time Image Measurements

To detect the targeted colored line to be followed by the controller of the robot, the image processing techniques has been used. The first step is to get and store the image (one frame of the video) from the robot's CMOS camera with the resolution of 640*480. Higher resolution makes the angle measurement more accurate but it makes the time longer for image processing and as a result the number of frames per second reduces .The camera is connected via USB port to myRIO board. There are some functions for image processing including color detection, function, various mathematical filters for, noise reduction, angle detection, diagnosis, etc. Although there are some ready functions for abovementioned tasks, due to their long delay and uncertainty have not been used in real-time tasks. So those tasks have been all programmed using basic functions.

2.1 Image Processing

As previously mentioned, the line to be detected by the robot has a distinct color compared to its background. The difference of RGB color levels is used for detecting the line. The function used for detection of the colored line in LabVIEW image processing is "IMAQ Color Threshold" function [5]. In this function by determining the amount of red, blue and green color which are numbers between 0 and 255, the targeted color of the line can be determined.

In the next step, the extracted image consisting of the targeted color is changed from RGB mode to binary or two-colored mode. Meaning that the areas that consist of our desired color must be specified by a distinguished color that we can choose, like red, and the other areas are displayed in black, which means that these areas don't have our desired color and they have been removed from the image.

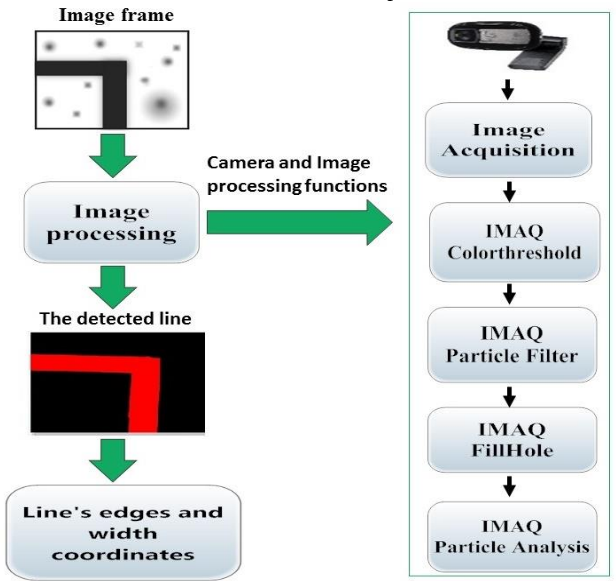


Fig. 2: This Fig. shows a black line located in a dirty plane, by using image processing all the noises is removed and it changed to the red line and the necessary information like line's edges and width coordinates is extracted from it as outputs.

Using some filter functions improve the two-colored image cleanness causing the extracted data be more accurate. The next function is the *IMAQ Particle Filter* function which can remove the values identified in the previous step based on the size, shape, length, width and other characteristics. Using this function, removes the potential points and spots on the ground floor, which have the same color as the line (see Fig. 2). The next step consists of extracting the desirable information that is needed for controlling the robot. The *IMAQ Particle Analysis* function has important capabilities such as detecting the angle, the center of the shape, shape's area, etc. This step is one of the main steps of the image processing algorithm in which only the necessary information should be extracted from the image; otherwise, it increases the quantity of calculation and cause's some delays in running the algorithm [5].

2.2 Angle Measurement Algorithm

After detecting the line and its edges it is time to use these data to measure the line's angle. Fig. 3 shows an image frame of the camera (the pink area) which the part of the line that located in the frame is determined by different color. This part of the targeted line at least has four edges because the camera frame has cut the line while it could have more edges in facing with broken targeted lines too. In order to choose the right edges, it is necessary to define some rules. In fact, the four blue spots seen in the pink area of the Fig. 3, i.e. (x_1, y_1) to (x_4, y_4), should be chosen correctly in any condition of the targeted line like straight line, turn left and turn right.

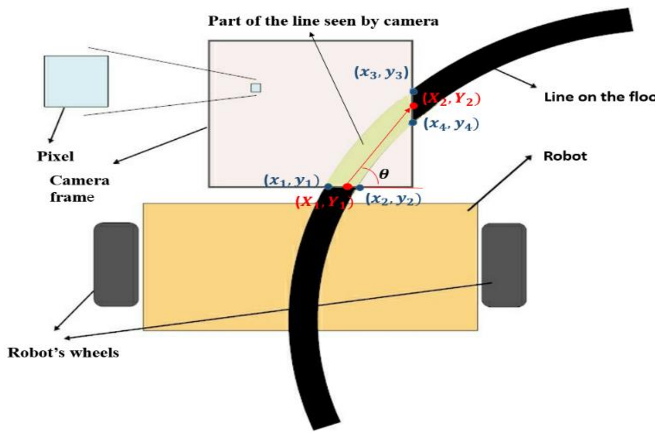


Fig. 3: A view of the part of the line seen in the frame and the edges needed for angle measurement.

There is the possibility that the $(x_1 . y_1)$, $(x_2 . y_2)$ spots could be chosen instead of each other but in the angle measurement process it will be clear that it does not make any problem.

Table 1: The spots which is needed for calculating the angle and the rules for choosing them among the other edges

Point	Rules
$(x_1 . y_1)$	The edge with the zero y
$(x_2 . y_2)$	The edge with the zero y and has not been chosen
$(x_3 . y_3)$	The point with the highest y **and** with the lowest or the highest x quantity which there is another point with same x quantity among edges
$(x_4 . y_4)$	The second point with the highest y **and** with the lowest or the highest x quantity which there is another point with same x quantity among edges

The next step in measuring the targeted line's angle is to calculate the $(X_1 . Y_1)$ and $(X_2 . Y_2)$ (the red points in Fig. 3). Here are the equations for calculation of those two points.

$$X_1 = x_1 - x_2 \\ Y_1 = y_1 = y_2 \tag{1}$$

$$X_2 = x_3 = x_4 \\ Y_2 = y_3 - y_4 \tag{2}$$

The absolute value of X_1 has been used for the next steps. These two points are used in the final step to measure the targeted line angle. A line between $(X_1 . Y_1)$ and $(X_2 . Y_2)$ is drawn to create the θ angle with the horizontal line, which is considered as the angle of the targeted line seen in the frame of the camera to be measured using equation 3.

$$\theta = \tan^{-1} \frac{Y_2 - Y_1}{X_2 - X_1} \tag{3}$$

This algorithm calculates the angle from 0 to 180 degree, but it is not symmetric and also it calculates the straight line's angle 90 degree which it is better to be zero to make the line follower robot to be controlled easier so 90 degree is decrease from the angle that calculated from the equation 3.

3 Line Follower Design

The sample robot, chosen to implement the proposed algorithm, is a two-wheeled robot. The Motors used in this robot are couple of DC motors with a maximum speed of 20 rpm, whose velocities are controlled using PWM (pulse width modulation) amplitude signals generated directly by myRIO. For supplying the myRIO board and bridge drivers a DC power supply of 12 volts or a 12-volt battery can be used.

MyRIO board has the capability to setup a webcam (connected via USB port) [6] and also has the capability to connect to a computer via Wi-Fi. The PWM outputs of the myRIO board is used for controlling the robot motors.

Using the myRIO board makes it possible to monitor the robot wirelessly at any time, for example displaying the PWM signal of each motor, the image

of the detected line, the measured angle, the frame of the image, etc. on the desktop version of the written program on PC [7], [8]. There is also the possibility to modify the controller method, run and stop the robot and choose the targeted line color that the robot should follow at any moment by the computer via the wireless connection while the robot is running and also there is a capability to improve the program like controller parameters or image processing algorithm wirelessly while the program is not running. This feature makes the process of changing and testing the program easier.
Since the controller can't supply the power needed by the motors, a PWM signal amplifier has been designed using a L298N bridge driver to switch the required power to the motors.

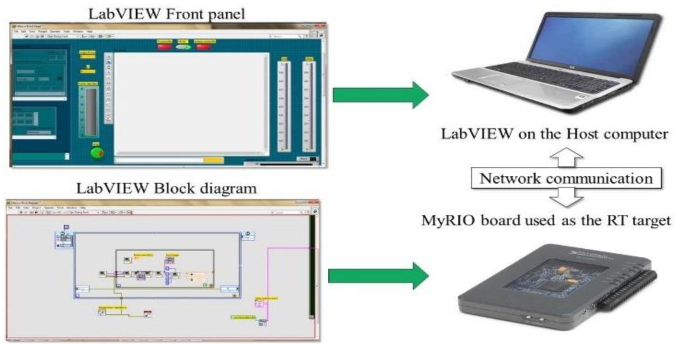
4 Real-Time Controller

One of the important features of the controllers and visual feedback algorithms, which is running on this robot, is the real-time principles applied to its tasks. This means that the tasks are prioritized and run according to their priority in a fixed time period. In fact, tasks are run before passing the deadline so that no disruptions or fails would occur while the robot is being controlled using the proposed algorithm. The idea of designing the controller to run in real-time is due to the fact that a certain number of image frames to be processed to extract the information needed for the fuzzy controller which makes the control more reliable and ensures the movements of the robot to be smooth especially in the line turns.
To implement the control algorithm in real-time, the different parts of the graphical code in LabVIEW is considered as a task that executes in a certain time period, then by creating a subVI for each of these tasks, different parts of the control algorithm such as commending the robot, image processing part and control algorithm (such as Fuzzy, P, PI) are all separated and prioritized depending on the importance of its role they play in controlling the robot and the time needed to execution. Table 2 shows all of the subVIs and their dedicated priorities. In this table the smaller the number the higher the priority [9].

Table 2: The SubVIs of the real-time control algorithm and the priorities assigned to each of them

Tasks	h
Image Acquisition subVIs	0
Image processing subVIs	0
Proportional Controller subVIs	1
PI Controller subVIs	1
Fuzzy controller subVIs	1
Sending PWM signals to myRIO outputs subVIs	2
Front Panel data monitor on PC subVIs	3
Receiving the controller parameters, start/stop and other subVIs	3

Another important issue to be considered while coding the visual feedback and controller in real-time is the data flow among the prioritized tasks. For correct implementing the data flow among the prioritized subVIs instead of using "wire" in LabVIEW, specific functions for sending and receiving the data should be used called "global variables" and "Real-time FIFO" functions. In myRIO, for making a set of sub VIs to run in real-time four hardware/software components are needed: 1) *Host computer* that runs the monitoring and front panel programs, 2) *Real-time module* that is a software module to arrange the sub VIs to run in real-time, 3) *RT target* that runs the visual feedback and controller algorithms in real-time and 4) *RT engine* that is a real-time LabVIEW version [9].
To describe more in detail the structure of abovementioned hardware and software set, it can be said that the host computer is a computer that designs VI which can have operating systems such as Windows, Linux, and IOS. RT targets are a collection of hardware devices which have the capability to run an RT engine. In fact, these hardware devices are suitable for running software programs in real-time while they don't have the hardware limits and delays seen in normal computers with operating system running a program. RT targets are divided into plug-in devices and RT network series [9].


LabVIEW Front panel
LabVIEW on the Host computer
Network communication
LabVIEW Block diagram
MyRIO board used as the RT target
Fig. 4: A view of communication of RT target and Host computer and the part of the LabVIEW they run

In the designed robot, the RT target is myRIO board, which is a plug-in device. The RT engine is a LabVIEW version which has been installed on the RT target and basically, its job is to run VIs which are designed by a user through a computer. Since it runs on the RT target, it is able to run the program in real-time [9].

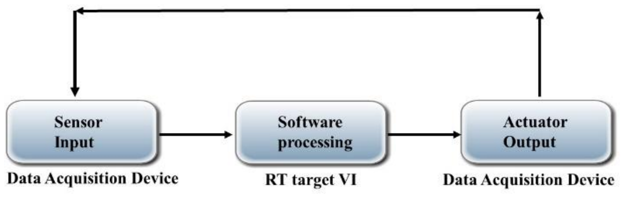


Fig. 5: An overall view of the real-time algorithm

Also, experimental results show that when the program, run in a non-real-time framework, the number of video frames can be processed are among 18 to 23 frames per second. But the real-time framework has the advantage of increasing the number of video frames to be processed higher (23-26) frames per second.
In the simulation and experimental part of the work, three types of controlling methods include proportional (P), proportional integration (PI) and Fuzzy logic controller (FLC) have been implemented and as the final experimental results, these three methods have been compared with each other too.

4.1 P and PI controller

Since the design of these two controllers is nearly same, the designing of the PI controller is just explained but the results of both Proportional and PI controllers have been compared by each other at the end to show the effects of integral coefficient, K_I.
In order to design PI controller, the range of the angles that the robot makes with the targeted line has been extracted from some experiments. The characteristics of the robot used in this study is generally not linear, but in some determined ranges, it can be piecewise linearized. Since the control method is not suitable for nonlinear systems, a controller has been designed for each range of angles that the robot shows nearly linear behavior. Table 3 shows the different ranges.

Table 3: The ranges that the Proportional and PI controllers shows nearly linear behavior.

Mode	Angle
Turn right	-90 to -5
Straight line	-5 to 5
Turn left	5 to 90

The implemented controller is a PI controller, a feedback from the angle is applied to the control system. One of the advantages of using visual feedback is that there is no requirement to use other types of the feedback like motor shaft encoder pulses, light sensors, etc.
The desired set point angle is 0 degree which is applied to the input of the controller (i.e. *R(s)*). Thus, the output angle is reduced from zero at every moment and the error (i.e. the difference between set point and the angle of the robot direction) is multiplied to the proportion coefficient, Kp, and is applied to the system [10]. The Fig. 12 shows the block diagram of the PI controller designed for the robot in which *R(s)* is the input or set point angle, *C(s)* is the output angle, Kp is the proportional coefficient, K_I is the integral coefficient, *H(s)* is the feedback ratio and *G(s)* is the transfer function of the robot system. The priorities assigned to each part of the real-time control loop have been included in this Fig. too. Table 5 shows the coefficients obtained for the PI controller.

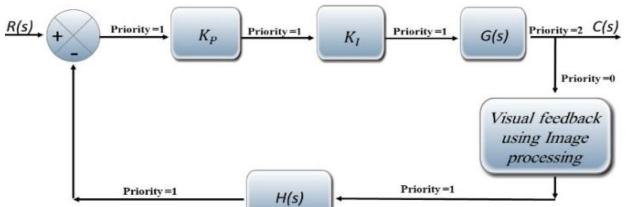


Fig. 6: The block diagram of the PI controller designed for the robot including visual feedback and the priorities assigned to each part of the real-time control loop

Table 4: K_P and K_I coefficients of the PI controller

Mode	Left motor's K_P	Left motor's K_I	Right motor's K_P	Right motor's K_I
Turn left	1	0.001	0.05	0.001
Straigh t line	0.1	0.01	0. 1	0.01
Turn right	0.05	0.001	1	0.001

4.2 Controlling the targeted line's location

In addition to necessity for controlling the angle of the robot direction, it must be noted that the targeted line should always be placed in a desired range between two wheels otherwise it would be outside the visibility range of the camera and it couldn't be seen and eventually the robot wouldn't be able to follow the line. So, the location of the line on the image or in fact the line width coordinates also should be measured in image processing steps to be used in controlling the robot. A proportional closed-loop controller is used for the line's coordinate control.

Table 5: K_P Coefficients of the line width coordinate proportional controllers

Mode	Right Motors K_P	Left Motors K_P
Straight line toward right	0.1	2.1

Straight line toward left	3.3	1.6

For simplicity, this control can be done when the robot follows a straight path. The measuring range of the line width coordinate is in the range of 0cm to 19.5cm and the desired range is 1.8cm to 16.5cm. To control the line's width coordinate. The variable is specified in two ranges: 1) Straight lines toward the left, and 2) Straight lines toward the right. A couple of Proportional closed-loop controllers are designed for both ranges.

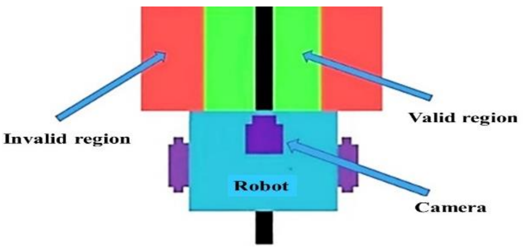


Fig. 7: A view of the line width coordinates controlling method

4.3 Fuzzy controller

To implement the fuzzy logic-based controller for the line follower robot a *Fuzzy System Designer* function has been used in the LabVIEW environment which is similar to the *Fuzzy logic tool* in MATLAB[11].
The first step in designing the Fuzzy controller is to introduce the input and output membership functions. For simplification, by specifying different states of the robot angle direction to the targeted line, three triangular-trapezoidal input membership functions have been introduced for the line angle: 1) Left, 2) Straight and 3) Right. Fig. 8 (a). Shows the line angle input membership functions.

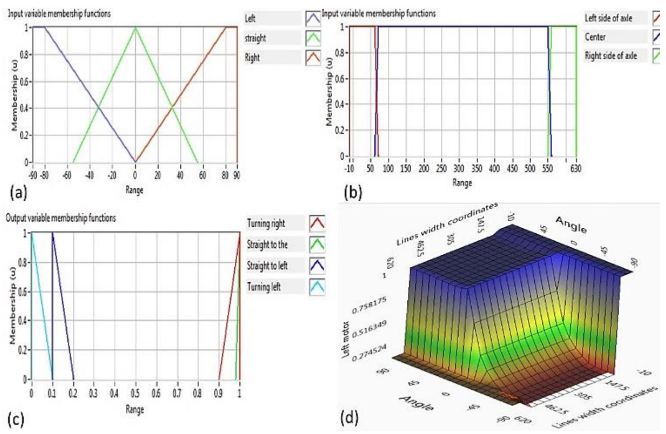


Fig. 8: Fuzzy membership functions and 3D view of the fuzzy controller results.

As noted in the previous section, in addition to controlling the angle of the robot, line width coordinates is also necessary to be measured and used in controlling the robot. Thus, line width coordinate membership functions of the line must be introduced for different states. Fig. 8(b). Shows the line width coordination input membership functions. Red membership function represents the state that the line is located in the left side of axle between the wheels, blue membership function represents the state that the line is in the desired region (center) between the wheels and the green membership function is related to the state in which the line is located in the right side of axle of wheels.
The outputs are normalized in order to get the right PWM signal for right and left motors using the standard functions to be able to use the full range of the DC motors operation.
Fig. 8(c). Shows the output membership function of the left motor, the green and yellow membership functions represent the cases that the line width coordinates are in the left and right side of the wheels' axis respectively, and blue and red colors are for the states which the line leans to the left and right respectively. The output membership functions of the right motor is nearly same as the left one.
One of the objects of designing the different controllers (i.e. P, PI and Fuzzy) in this project is to make the robot to work at its maximum speed and minimum oscillation/vibration. One of the reasons that a gap exists in the Fuzzy output membership functions between 0.2 to 0.9 is this. In fact, it is desired to increase the PWM averaged amplitude signal as much as possible in order to improve the robot's operation. Another reason of the gap existence is that the speed changes of the motors is not linearly proportional to the changes of the PWM averaged amplitude signal, while it is more sensitive in lower amplitudes rather than the medium amplitudes.
After introducing the input and output membership functions, it is time to determine Fuzzy Rules. This step actually relates the input and output states. Table 6 shows the fuzzy rules of the left motor. The fuzzy rules cover all possible conditions of the inputs.

Table 6: Fuzzy rule-base for the left motor

Angle / **Width coordinate**	**Left**	**Straight**	**Right**
Left side of axle	Straight to the right	Straight to the right	Straight to the right
Center	Turning left	Straight to the right	Turning right
Right side of axle	Straight to the left	Straight to the left	Straight to the left

5 Experimental Results

To compare the experimental results of the three control methods implemented (i.e. P, PI and Fuzzy), the performance of the robot controllers for two angles of 90 degrees and 50 degrees have been analyzed, and the data samples have been acquired directly from myRIO to the host computer. The acquired data are saved in .bin files and later opened in MATLAB to draw the plots. It is noted that in the angle control of the robot using PI controller there is a big undershoot and oscillation when the 50-degree turn is performed (see Fig. 9). As shown in Fig. 9, the undershoot became bigger when the turn got sharper (90 degree turn).
Instead, by using the fuzzy controller, the results recorded for both 50 degree turn and 90 degree turn (shown in Fig. 9 and Fig. 10 respectively) show that the undershoot and oscillations are totally cleared. This means that in the turns the robot mimics smooth movements which bring some advantages that comes as follows: 1) The need for smooth movements in the industrial factories that the robot carry liquids are satisfied using fuzzy controller. 2) The mechanical vibrations and shocks reduced very well causing long life operation of the robot. 3) The energy consumption of the robot reduced.

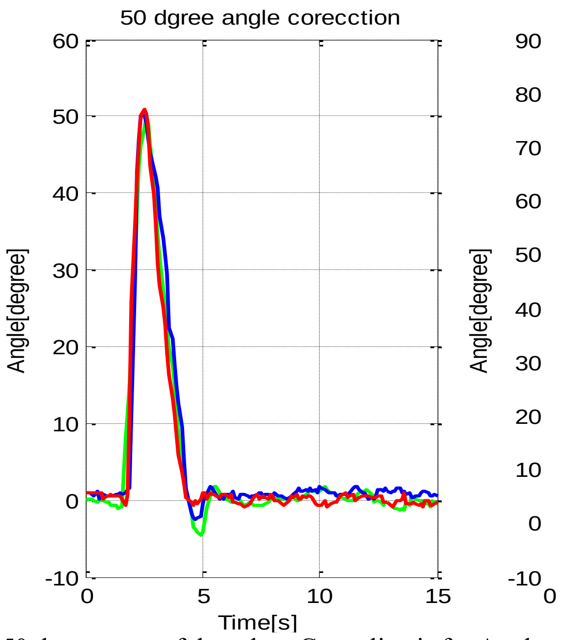


Fig. 9: 50 degree turn of the robot. Green line is for Angle correction using Proportional controller, Blue line is for Angle correction using PI controller and Red line is for Angle correction using Fuzzy controller.

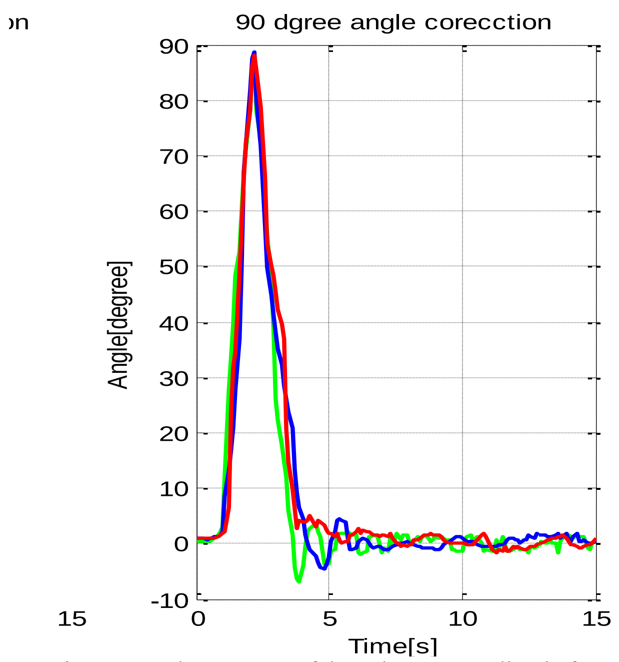


Fig. 10: 90 degree turn of the robot. Green line is for Angle correction using Proportional controller, Blue line is for Angle correction using PI controller and Red line is for Angle correction using Fuzzy controller .

To show the advantage of the fuzzy controller, in another test, the time that the robot needs to finish a constant loop is measured for Proportional, PI and Fuzzy controllers (see Fig. 1 (b). The experimental results show fuzzy controller makes the robot quicker than PI controller around 5% and quicker than P controller around 6.5%.
Also, experimental results show that when the program, run in a non-real-time framework, the number of video frames can be processed are among 18 to 23 frames per second. But the real-time framework has the advantage of increasing the number of video frames to be processed higher (23-26) frames per second.

6 Conclusions

In this paper, an image processing algorithm for targeted line angle detection and robot alignment has been designed and implemented in real-time to control the movements of a robot. Using visual feedback and fuzzy controller, the line follower robot designed in this experiment is being able to follow the line faster while showing smoother movements in turns. The camera is the only feedback device while there is no need to use other position and speed feedback from the wheels in which big errors and uncertainties are incorporated.
The visual feedback in this robot has an advantage that even if the robot was deviated due to clash of obstacles, it will correct its angle. In this paper the

image processing algorithm using P, PI and Fuzzy controllers have been implemented by LabVIEW programming to make the controllers run fully in real-time on a myRIO module. Using the real-time framework to implement the controllers results the control loop run faster. The experimental results show that the fuzzy controller with smart rule defining for such robots has some advantages including faster and smoother movements, less mechanical vibration and shocks, more reliability to carry the liquids, less energy consumption, etc.

References

[1] C.-P. Tsai, C.-T. Chuang, M.-C. Lu, W.-Y. Wang, S.-F. Su, and S.-L. Chang, "Machine-vision based obstacle avoidance system for robot system," in *System Science and Engineering (ICSSE), 2013 International Conference on*, 2013, pp. 273–277.
[2] N. Chakravarthy and J. Xiao, "FPGA-based control system for miniature robots," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 3399–3404.
[3] N. I. L. U. Manual, *National Instruments (2003)*. 2015.
[4] L. Cuzco, J. Arias, A. Abril, and D. Cabrera, "Acquisition, filtering and decomposition into real-time ECG signal with Ni — MyRio," in *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 2015, pp. 209–213.
[5] R. Posada-gómez, O. O. Sandoval-González, A. Martínez Sibaja, O. Portillo-Rodríguez, and G. Alor-Hernández, "Digital Image Processing Using LabView," *Intechopen*, 2001.
[6] National Instruments, "User guide and specification: NI myRIO-1900," p. 20, 2016.
[7] E. Doering, "NI myRIO-project essentials guide," *Natl. Instruments*, vol. 129, pp. 55–59, 2014.
[8] Y. Angal and A. Gade, "LabVIEW controlled robot for object handling using NI myRIO," *2016 IEEE Int. Conf. Adv. Electron. Commun. Comput. Technol. ICAECCT 2016*, pp. 167–171, 2017.
[9] R. M. U. Manual, "Real-Time Module User Manual," no. 322154, 2003.
[10] F. Logic and T. User, "PID and Fuzzy Logic Toolkit User Manual," *Program*, no. June, 2009.
[11] M. Davoudi and M. Davoudi, "Bayesian fuzzy logic-based estimation of electron cyclotron heating (ECH) power deposition in MHD control systems," *Fusion Eng. Des.*, vol. 119, pp. 35–41, 2017.

Authors

Mohsen Davoudi received his Ph.D. in Electrical Engineering from Polytechnic University of Milan (Politecnico di Milano), Milan, Italy, in 2011. Currently he has assistant professor position at Imam Khomeini International University (IKIU), Qazvin, Iran.

Afshin Heidarian received his BSc degree in Electrical Engineering from Shiraz University in 2015. He is presently pursuing Masters in Electrical Engineering at Imam Khomeini International University, Qazvin, Iran. His research interests are Image processing and enhancement, Hardware implementation of controllers (fuzzy, adaptive, PID…) and Instrumentation.