Stereo Image Partitioning Based Fuzzy Logic Controller for Real-time Obstacle Detection and Avoidance

Kirti Shankar Sharma and P. V. Manivannan
Department of Mechanical Engineering, Indian Institute of Technology Madras, Chennai, India
Email: sharmakirtishankar15@gmail.com, pvm@iitm.ac.in

***Abstract*— The research work in this paper deals with the development of a Fuzzy Logic based static and dynamic obstacle avoidance system for navigation of a mobile robot using vision. A low-cost stereo vision system is developed, calibrated and mounted on the differential drive robot, and an initial pair of images is captured. Kanade-Lucas-Tomasi tracker with stereo point cloud is used in this pair of images to initiate the robot's motion. Once the robot is in motion, the successive stereo pair of images is partitioned. These partitioning helps to detect forthcoming obstacle that appear from the center or left/right corners. The developed Mamdani-fuzzy logic controller then identifies the obstacles using Moore-Neighbor tracing algorithm based boundary detection and subsequently provides a direction of motion to avoid them. A simulation with a stereo camera mounted on mobile robot is carried out in a virtual environment is generated using V-rep and controlled by MATLAB command via API. The algorithm has also been experimentally validated on the Amigobot with Pioneer 3-DX as dynamic obstacle by VICON tracking system in an indoor laboratory environment, and the comparison between the simulation and experimental results are discussed.**

***Index Terms*— fuzzy logic, Kanade-Lucas-Tomasi, differential drive robot, Moore-Neighbor tracing, obstacle avoidance, stereo vision, image partition, boundary detection, VICON tracking system**

I. Introduction

In autonomous vehicles, the obstacle detection and avoidance play a vital role in navigation. Autonomous navigation is a difficult process that requires the environmental perception of the nature, size, and state of obstacles. This perception of the environment can be achieved by either the use of active or passive sensors. Active sensors require an external source of power to operate, whereas passive sensors detect and respond directly to some input from the physical environment. Active sensors like LiDAR, RADAR, etc. give more accurate results regarding point depth but consume huge amounts of energy. Passive sensors like cameras, on the other hand, uses the existing properties/quality of the environment to get the required information.

Since the invention of cameras, its use in navigation system has increased tremendously with the increased usage of image processing [1], machine learning and deep learning algorithms. Object recognition [2] and detection [3] is also being used as an application of image processing in industrial and autonomous systems. There are also a few difficulties with the use of deep learning and neural network in autonomous systems. Procedures such as data collection, handling, and processing increase the processing cost and processor specification [4]. These, in turn, increase the overall cost of development of an autonomous system.

A. Motivation

Active sensors like ultrasonic, infrared sensor, etc. are used for obstacle detection and depth calculation. An array of these sensors with the application of fuzzy logic is used to detect and avoid the dynamic and static obstacles [5]. LiDAR, on the other hand, gives the overall map of the environment in real time [6]. The major drawback being the difficulty in detecting shiny surfaces like mirrors and glasses, which requires it to be combined with other sensors. Kinect uses RGB camera and IR sensor data to detect the real-time depth calculation [7]. It is costly, heavy and best suited for the indoor environment [8].

The environmental perception using passive sensors is power efficient and cost-effective. With the development of CCD cameras, its use in the robotic application has been increased. This is because of a rich source of data available at low cost. On the one hand it is cost effective in terms of capturing the instants from the environment, while on the other hand availability of rich data makes it complicated to analyze the state and nature of obstacles.

Optical flow detection is a method to detect the movement of any obstacle using the pixel intensity values of images and the corresponding movement of feature points [9]. For a static case of observation point and environment, the displacement will be zero which results in zero optical flow vector. As soon as a dynamic obstacle comes into the environment, movement of feature points will be detected, and a corresponding optical flow vector is generated showing the direction of its motion. These feature points can also be used to track and avoid the dynamic based on optical flow vector [10].

Manuscript received August 3, 2019; revised July 9, 2020.

doi: 10.18178/ijmerr.9.8.1158-1169

G Monteiro, et al. used a monocular camera for obstacle detection and tracking using a laser pointer [11]. X Miao, et al. uses the image processing application with monocular camera and laser to detect the lanes on the road [12]. Although it is efficient in terms of accuracy, the combination of active and passive sensors increases the power requirements of the system. The stereo vision system which makes use of two identical cameras, works like a human eye, is used to map the point cloud for autonomous navigation [13]. The stereo vision system is calibrated to avoid the intrinsic and extrinsic effects like camera nonlinearity and relative camera positions. The development of a stereo vision system and algorithm for tracking dynamic obstacle using application of KLT tracker has been developed and verified using low-cost cameras [14].

Once the obstacle has been detected, the decision for its avoidance has to be carried out. Controllers like PID, PI, etc. are some of the generally used controllers for motor control of the vehicle. Fuzzy logic inference [15] is a method in which certain rules are defined during the development. The output probability is obtained using these predefined rules based on input parameters. Neural network [16] and machine learning [17] based approaches are also some methods to give controlled output, but they require higher processing power compared to fuzzy logic [18].

B. Problem Definition

For the navigation of an autonomous vehicle, it is very important to know the state and nature of obstacle i.e. static or dynamic. Table I shows the four possible relationships between the observer (vehicle) and the obstacle, and the corresponding method required for obstacle-free navigation.

TABLE I. METHODS TO DETECT AND AVOID THE OBSTACLES USING VISION SYSTEM

Obstacles	Observer	Methods used for obstacle avoidance
static	static	Only stereo vision system
dynamic	static	Stereo vision system + Object detection & tracking
static	dynamic	Optical Flow method
dynamic	dynamic	Optical flow + stereo vision + Object detection & tracking + Logic based navigation

The first two cases from Table I (Static-Static, Dynamic-Static) has been achieved using low-cost stereo vision systems and KLT tracking method [14]. In real life conditions, there is a movement of the observer, obstacles as well as the environment itself (last two cases from Table I), causing the above method to fail under these conditions. Hence this paper addresses Static-Dynamic and Dynamic-Dynamic obstacle-observer condition. The currently developed algorithm can also detect and avoid the obstacles which are not in the binocularly visible ranges.

C. Organization of Paper

The paper starts with an introduction and problem definition. The rest of the paper is organized as: Section II talks about the theoretical view of the methodology implemented and the specified review of the velocity relationship, while Section IV provides a flowchart and its explanation about the developed algorithm. Section V talks about the development of a simulation environment in V-rep and its integration with MATLAB and Section VI discusses the development of the experimental setup and its operation. Section VII consists of the comparison of simulation and experimental results for different static and dynamic obstacles case, and discussion about it. It also talks about the future scope and application of the research work while Section VIII consists of references that helped the development of the algorithm.

II. METHODOLOGY

Movement Profile: The motion of an autonomous vehicle/robot is affected by the movement of obstacles in the environment.

TABLE II. MOVEMENT PROFILE CATEGORIZATION BASED ON RELATIVE POSITION AND VELOCITY

Observation	subcategory	Main category
$V_O > V_R$ & $D_{OR} \rightarrow$ decreasing and $F_L = F_R$	Quickly approaching	Approaching
$V_O < V_R$ & $D_{OR} \rightarrow$ decreasing and $F_L = F_R$	Slowly approaching	
$V_O > V_R$ & $D_{OR} \rightarrow$ increasing and $F_L = F_R$	Quickly leaving	Leaving
$V_O < V_R$ & $D_{OR} \rightarrow$ increasing and $F_L = F_R$	Slowly leaving	
$V_O > V_R$ & $D_{OR} \rightarrow$ unchanged and $F_L = F_R$	Quickly translating	Translating
$V_O > V_R$ & $D_{OR} \rightarrow$ unchanged and $F_L = F_R$	Slowly translating	
$V_O > V_R$ & $D_{OR} \rightarrow$ unchanged and $F_L > F_R$	Rotating right	Rotating
$V_O > V_R$ & $D_{OR} \rightarrow$ unchanged and $F_L < F_R$	Rotating left	

The velocity and the movement profile of the autonomous system is defined by the relative distance and velocity of the robot & the obstacles [19]. Table II shows the different motion profiles between the obstacles and the autonomous system and the different optical flow (in the form of pixel movements) parameters that affect them. (Where V_O = Velocity of an obstacle, V_R = Velocity of Robot, D_{OR} = Distance between obstacle and robot, F_L = Optical flow vector in the left frame and F_R =

Optical flow vector in the right frame of the stereo vision system

To avoid these obstacles, next set of steps to be taken by the observer or an autonomous system will depend on the following environmental variables:

- Speed / Velocity of obstacle
- The direction of obstacle movement
- The velocity of an observer with respect to the environment
- Environmental visibility
- Prediction of forthcoming hindrances

Proposed Approach: The high sensor cost and complexity in sensor integration has led to an increase in vision-based environmental perception. Obstacle avoidance based purely on image data is a crucial area of research due to the amount of data that is available at any instance in time. Using only the stereo vision system, data handling and computation requirement are very high which leads to difficulty in onboard implementation of a stereo vision system for an unknown environment [20]. The above method uses continuous mapping and data compilation, but an alternative method, implemented in this paper, is to detect obstacles and corresponding movements in small intervals without storing much data.

From the camera sensor, rich data about the environment is captured thereby leading to high computation requirements while analyzing and interpreting the nature of obstacle and behavior of the environment. To achieve the above, highly complex algorithms like data analysis, machine learning and neural networks are utilized. Although these methods give good results at various conditions, low-cost processors like Raspberry pi [21] and Arduino cannot handle the computational requirements. Hence, we require some alternative low cost and low power algorithms which can satisfy the needs of the industrial and warehouse autonomous robot navigation. Kanade-Lucas-Tomashi method for dynamic obstacle detection based on feature tracking and property of stereo vision system [14] which was previously developed has certain limitations. This method cannot detect and trace obstacles while moving observer is in motion.

There are various methods to detect dynamic obstacle, but the fuzzy inference is a method that uses input vector and, based on some set of predefined rules, assigns the output vector. It is easy, fast and less complex because most input conditions have already been defined during the development of the fuzzy inference. As quoted by Lotfi Zadeh, "In almost every case you can build the same product without fuzzy logic, but fuzzy is faster and cheaper" [18]. Hence for the detection of various dynamic obstacles at a dynamically varying observation point, a fuzzy logic based intelligence has been developed. Fuzzy logic method for navigation is carried out using the following steps:

1. Fuzzification.
2. Control Rules.
3. Defuzzification.

A. *Fuzzification*

During this process, the inputs parameters to fuzzy logic inferences is defined. For the particular case of obstacle avoidance using low cost stereo vision system, the captured images are taken to be 160*120 pixels to maintain low data transfer time (Fig. 1). These captured images are partitioned namely: Left Most image (LMO), Left Image (LI), Left Mid Image (LMI), Left Top Image (LTI) and Left Bottom Image (LBI) for left camera image while Right Most image(RMO) Right Image (RI), Right Mid Image (RMI), Right Top Image (RTI) and Right Bottom Image (RBI) for right camera image as shown in Fig. 2. The number of dark pixels and the boundary around them in a particular subsection decides whether an obstacle is near / far. An obstacle is considered to be near/detected (D) in a subsection, when number of pixels inside it is more than 75% of the total pixels of that subsection.

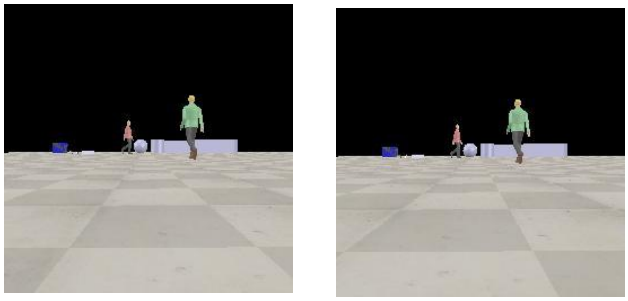


Figure 1. Original image (a) left image; (b) right image

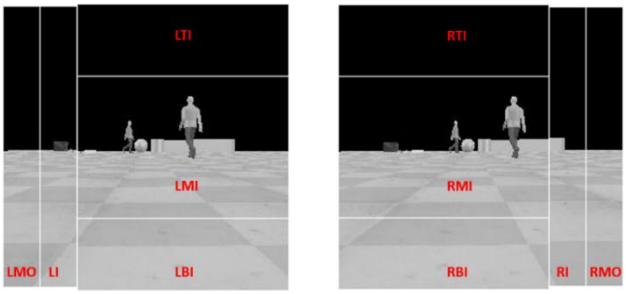


Figure 2. Partitioned image pairs (a) left image; (b) right image

Let U_{LMO}, U_{LI}, U_{LMI}, U_{LTI} and U_{LBI} denotes the undetected/far obstacle in left most, left, left mid, left top and left bottom part of image obtained from left camera respectively. While, U_{RMO}, U_{RI}, U_{RMI}, U_{RTI} and U_{RBI} denotes the undetected / far obstacle in right most, right, right mid, right top and right bottom part of image obtained from right camera respectively. Similarly, D_{LMO}, D_{LI}, D_{LMI}, D_{LTI}, D_{LBI} and D_{RMO}, D_{RI}, D_{RMI}, D_{RTI} and D_{RBI} denotes the detected/ near obstacle case. The membership function for each subsection of image plane is decided based on the nature of the input and its effect on the logical decision. The membership function for each segment of the input image is the same and is shown in Fig. 3(a) and 3(b).

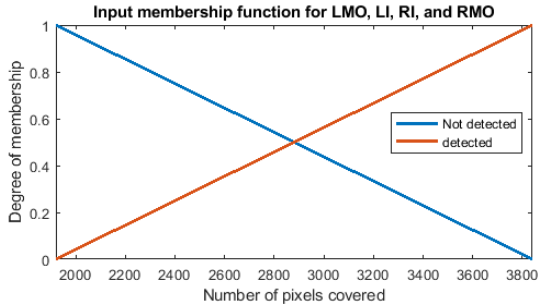


Figure 3(a). Membership function for LMO, LI, RI and RMO.

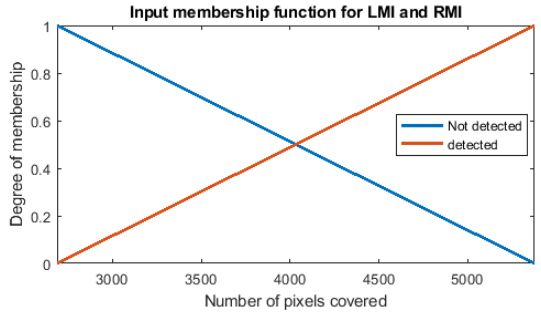


Figure 3(b). Membership function for LMI and RMI.

The output of the fuzzy logic controller is steering control and movement. These outputs are fed to the developed model in V-rep, wherein the vehicle response can be seen.

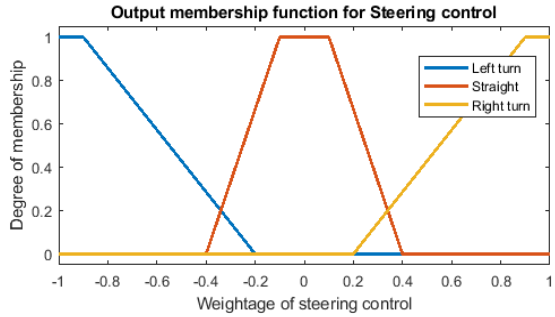


Figure 4(a). Membership function for Steering output.

The steering control decides left turn, straight or right turn of vehicle while movement decides the acceleration, constant velocity, and de-acceleration of vehicle-based on fuzzy logic controller output. The membership function for vehicle steering control and movement can be seen in Fig. 4(a) and 4(b) respectively.

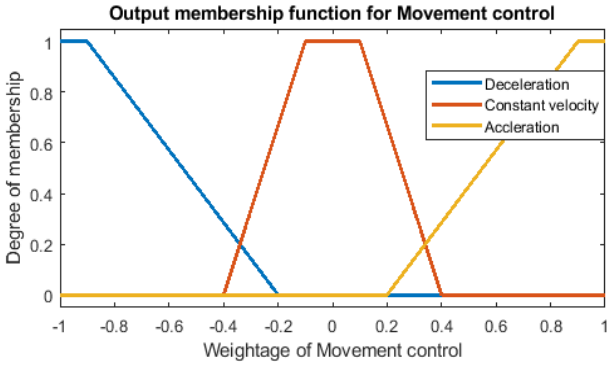


Figure 4(b). Membership function for movement output.

B. Control Rules

The control rules are decided based on the human intelligence and will guide robot at various input conditions. In the case of vision-based navigation, the control rules will be based on the pixel fill-up of the subsections of the stereo images as input. The control rules depends on the input images at subsequent time instances. For each sub-sectioned stereo image obtained and the corresponding logical output, there are 64 combinations that decide robot's motion. The fuzzy logic implementation makes use of the following subsections: LMO, LI, LMI, RMI, RI and RMO, while the remaining subsections: LTI, RTI, LBI and RBI are utilized for emergency conditions to avoid overhang and on-road hindrances.

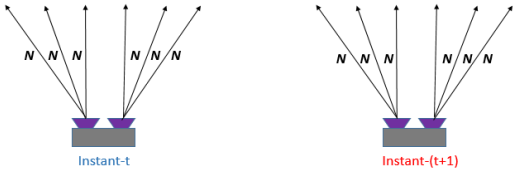
TABLE III. MOVEMENT PROFILE FOR DIFFERENT WHEEL VELOCITY OF VEHICLE

	Accelerating	Constant	Decelerating
Left	$(V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$ & $(V_{LFT} < V_{RGT})\|_{t+1} > (V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$	X	$(V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$ & $(V_{LFT} < V_{RGT})\|_{t+1} < (V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$
Straight	$(V_{LFT} = V_{RGT})\|_t$ & $(V_{LFT} = V_{RGT})\|_{t+1} > (V_{LFT} = V_{RGT})\|_t$	$(V_{LFT} = V_{RGT})\|_t$ & $(V_{LFT} = V_{RGT})\|_{t+1} = (V_{LFT} = V_{RGT})\|_t$	$(V_{LFT} = V_{RGT})\|_t$ & $(V_{LFT} = V_{RGT})\|_{t+1} < (V_{LFT} = V_{RGT})\|_t$
Right	$(V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$ & $(V_{LFT} > V_{RGT})\|_{t+1} > (V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$	X	$(V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$ & $(V_{LFT} > V_{RGT})\|_{t+1} < (V_{LFT} \le V_{RGT}\ or\ V_{LFT} \ge V_{RGT})\|_t$

There are three-speed control behaviors, namely: acceleration, constant velocity and deceleration. These are decided by the vehicle wheel speed where V_{LFT} and V_{RGT} denote the left and right wheel's velocity (Table III). The wheel velocity also depends on the condition of the detected obstacle. The higher the percentage of pixel area covered over and beyond 50 % (nearer the obstacle), lesser will be the wheel speed. The approaching and leaving of dynamic obstacle is decided by the direction of the fill up of the subsection for the captured stereo image.

The developed fuzzy logic considers multiple scenarios but certain critical cases which affect the robot motion are discussed below:

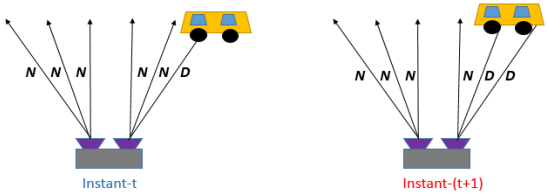
1. *No obstacle*: when the images sections are filled less than or equal to 50% of the image section.



If at time instant *t*:	At time instant (*t+1*):
LMO→U_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *accelerating, straight*	LMO→U_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → U_{RMO} then S_v = *accelerating, straight*

Figure 5(a). No obstacle detected.

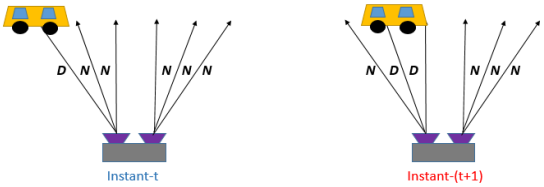
2. *Left translating obstacle*: when the obstacle is approaching to vehicle from right



If at time instant *t*:	And at time instant (*t+1*):
LMO→U_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → D_{RMO} and S_v = *accelerating, straight*	LMO→U_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → D_{RI} & RMO → D_{RMO} then S_v = *accelerating, left turn*

Figure 5(b). Translating left.

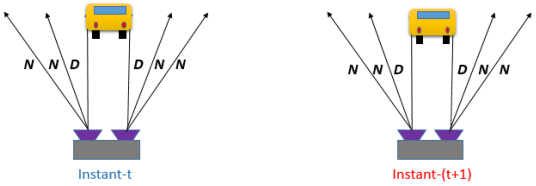
3. *Translation right obstacle*: when the obstacle is approaching to vehicle from left



If at time instant *t*:	And at time instant (*t+1*):
LMO→D_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *accelerating, straight*	LMO→U_{LMO}, LI → D_{LI}, LMI → D_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → U_{RMO} then S_v = *decelerating, straight*

Figure 5(d). Translating right.

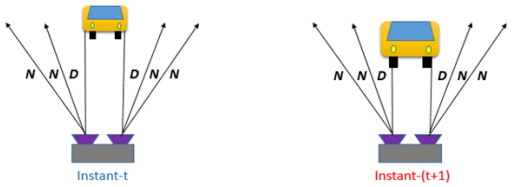
4. *Constant moving obstacle*: when the obstacle is moving at constant relative speed.



If at time instant *t*:	And at time instant (*t+1*):
LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *constant, straight*	LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *constant, straight*

Figure 5(e). Moving at constant speed.

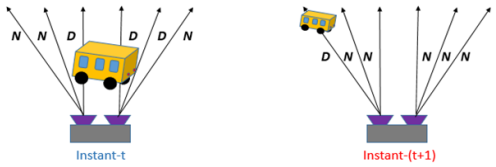
5. *Obstacle approaching*: when the obstacle is moving at greater speed towards vehicle (observer) i.e. pixel fill-up is increased in each subsection as compare to previous frames.



If at time instant *t*:	And at time instant (*t+1*):
LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *constant, straight*	LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *decelerate, straight approaching*

Figure 5(f). Vehicle approaching.

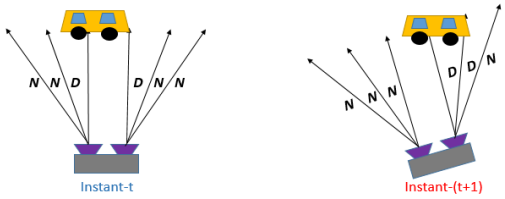
6. *Obstacle leaving*: when the obstacle is moving at greater speed away from the vehicle (observer) i.e. pixel fill-up is decreased in each subsection as compared to previous frames.



If at time instant *t*:	And at time instant (*t+1)*:
LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → D_{RI} & RMO → U_{RMO} and S_v = *decelerate, right turn*	LMO→D_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → U_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *accelerate, straight*

Figure 5(g). Vehicle approaching.

7. *Vehicle (observer) rotating*: when the obstacle is moving or stationary and the vehicle (observer) is rotating i.e. pixel fill-up is transferred from one subsection to other in each subsection as compared to previous frames.



If at time instant *t*:	And at time instant (*t+1)*:
LMO→U_{LMO}, LI → U_{LI}, LMI → D_{LMI}, RMI → D_{RMI}, RI → U_{RI} & RMO → U_{RMO} and S_v = *decelerate, straight*	LMO→U_{LMO}, LI → U_{LI}, LMI → U_{LMI}, RMI → D_{RMI}, RI → D_{RI} & RMO → U_{RMO} and S_v = *accelerate, left turn*

Figure 5(h). Vehicle rotating.

C. Defuzzification

Defuzzification is a process by which the control outputs are changed for its implementation to the physical system. The control outputs from the fuzzy logic design are:

- Steering control:
  - Left-turn
  - Straight
  - Right-turn
- Movement:
  - Deceleration
  - Constant speed
  - Acceleration

The output of control logic is the input for the physical system: P3-DX (Virtual or experimental). For left-turn, the left wheel speed must be less than right wheel speed and for right turn of vehicle, the right wheel speed must be less than left wheel speed. Straight motion (front or reverse) requires both wheel speed to be same. The combination of steering control along with the movement gives the final motion behavior of the vehicle. For the obstacle avoidance in virtual environment, the left and right wheel velocity is given. Algorithm 1 shows the formulation of those velocities which is fed to the virtually developed environment in V-rep.

Algorithm 1:

if absolute (Steering)>=0.01
Vl = Vl(1-Steering);*
Vr = Vr(1+Steering);*
else
*Vl = Vl+ Movement*T;*
*Vr = Vr+ Movement*T;*
end
if absolute (Vl)> Vl_{max}
*Vl = sign (Vl)*5;*
end
if absolute (Vr)> Vr_{max}
*Vr = sign (Vr)*5;*
end

where,
Vl = Left wheel speed for robot
Vr = right wheel speed for robot
Vl_{max} = maximum left wheel speed for robot
Vr_{max} = maximum right wheel speed for robot
Steering = Value of steering output from fuzzy logic
Movement = Value of movement output from fuzzy logic
T = constant (defined by user)

For the actual testing the mobile robot takes input in the form of linear and angular velocity. Since the maximum allowable linear velocity and angular velocity for Amigobot is 740 mm/sec and 300 degree/sec. So, for verifying the developed algorithm in controlled indoor environment the maximum linear and angular velocity is considered as 100 mm/sec and 20 degree/sec. The wheel speed at any instant depends on the initial speed of wheel and pixel fill-up in the consecutive frames. Algorithm 2 provides the wheel's linear velocity and angular velocity at a particular instant.

Algorithm 2:

V_L = R(Vl + Vr) / 2;*
ω = R(Vr - Vl)*SF / L;*
if absolute (V_L)> VL_{max}
V_L = Vl_{max};
end
if absolute(ω)> ω_{max}
ω = ω_{max};
end

where,
V_L = Linear velocity for robot
R = Wheel radius for Amigobot
ω = Angular velocity for robot
SF= Scaling factor (decided by user)
L = Axle track
Vl_{max} = Maximum linear velocity
ω_{max} = Maximum angular velocity

III. Algorithm Development

The developed algorithm uses the left and right images as a source of information for autonomous navigation. Before starting the motion of robot/vehicle, it is important to get the perception of environment and the obstacle position. The left and right camera of stereo vision system

is initiated and an initial frame is captured. These frames are used by the KLT based tracking system[14] to detect and track the dynamic and static obstacles.

Once the path is clear / obstacle free, the consecutive frames are fed for grayscale conversion. The grayscale images are then converted to binary image based on the threshold value for a particular color detection. Partitioning of the binary images is done based on the image size, the predefined area of coverage and with the knowledge of possible hindrance from left / right and top/bottom corners. Once the partitioning of images is complete (10 subsections for each stereo image), the calculation of the total number of dark pixels and boundary detection in those subsections is obtained. These values are used to decide the proximity of the obstacle, which is then fed to the developed fuzzy logic algorithm in the form of membership functions.

Depending on this membership function value and the obtained fuzzy logic rule, the two outputs are generated in the form of movement and steering control. A positive value, zero and negative value of steering output represents right turn, straight and the left turn for the vehicle respectively. Similarly, a positive value, zero and negative value of movement represents the acceleration, constant speed and deceleration respectively.

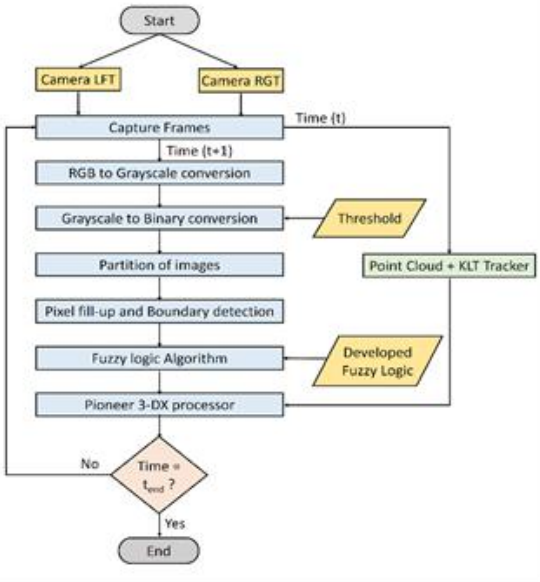


Figure 6. Flowchart of the developed Algorithm

The fuzzy logic outputs are then fed to P3-DX controller board via Aria installed in the processor. Aria is an API which takes linear and angular velocity as input and feeds them to the motor controller board which then decides the further course of action of the robot. The above process is repeated for next consecutive frames until the user stops the experiment (Fig. 6). With an interval of 900 frames, the hybrid vision system algorithm developed previously [14] is integrated into the logic and is used to get the depth of each obstacle.

IV. Simulation of Virtual Environment

The developed algorithm is based on human perspective on how to behave when finding an obstacle. As humans, our continuous learning facilitates us to react to a sudden change in environmental conditions. The machine/robots on the other hand can handle only situations that have been programmed and cannot handle unexpected or undefined environmental variables. For the validation of the developed fuzzy logic algorithm, it is important first to test it in a virtual environment rather than going for experimental process. The virtual environment is developed in V-rep (Virtual Robot Experimental Platform) software considering actual world physics.

Development of Virtual Environment: The environment consists of mainly two types of obstacles namely: static and dynamic. In vision-based detection, the texture color and background also play a vital role. Following are the main considerations taken for the development of a virtual environment in V-rep.

- A stereo vision system has been developed using the vision sensors available in V-rep which exactly matches with the available RGB camera used in the experimental setup. The resolution used is 160x120 pixels, while field of view used is 60^o .
- The developed stereo vision system is mounted in already available model of P3-DX (Pioneer 3-DX) differential drive mobile robot facing the front.
- The V-rep platform is surrounded by detectable boundary wall in order for the robot to navigate within the given range.
- Spherical, cuboid and cylindrical obstacle are put in the environment as static obstacles within the boundary and are made detectable.
- For dynamic obstacle, a randomly moving P3-DX is used in the virtual environment.

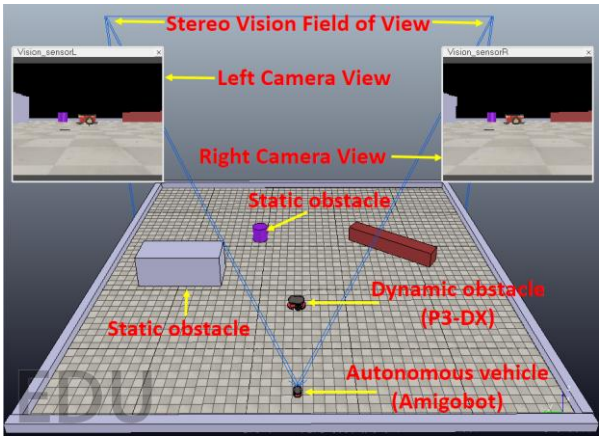


Figure 7. Developed virtual environment in V-rep

The Fig. 7 shows the developed virtual environment.

Integration of V-rep and MATLAB: V-rep is a virtual environment creator for the simulation of robot algorithm. The availability of remote API has made it easy to fetch data from V-rep and give control command from MATLAB. There are two possible way of communication between V-Rep and MATLAB: synchronous and asynchronous mode. In synchronous mode, communication between the two will happen in fixed time intervals, and the clock frequency between V-rep and

MATLAB must match each other for effective communication. While in asynchronous mode communication, the data transfer is done for different clock frequency and which in turn depends on the availability and requirement of communication. For the validation of developed algorithm, asynchronous mode of communication was chosen for reducing the data transmission time and load in the communication channel.

V. Experimental Setup and Implementation

Development of Stereo Vision System: The stereo vision system has been developed using a pair of low-cost cameras (Logitech c310). Both the cameras have identical specifications and are connected via USB 2.0 port with the processor (laptop). The cameras are mounted in a fixed frame printed using 3d printing (Fig. 8) having a fixed distance between them. This is done to capture the disparity between the frames, which in turn gives the depth mapping and point cloud mapping.

Figure 8. Low-cost stereo camera system

In order to avoid the high processing requirement for images, low resolution with 160x120 pixels are captured at the rate of 30 frames per second. After the initiation of camera, a small delay is taken before capturing the first frame. This is done to avoid/compensate the loss of information during the initiation of both the cameras simultaneously.

Pioneer 3-DX and Amigobot Mobile Robot: Pioneer 3-DX and Amigobot are differential drive mobile robot with an added support of a caster wheel (Fig. 10 (b)). It comes with a complete aluminum body, reversible DC motors, a balanced drive system, high-resolution motion encoders, and battery power, all managed by an onboard microcontroller and mobile-robot server software. It has its own onboard processor which supports Advanced Robot Interface for Applications (ARIA) which is used to interact with another processor (i.e. laptop, single board processor, etc.) to communicate in synchronous and asynchronous modes. Its onboard processor and sensors make it easy to travel in complex environments like a classroom, etc. It owns a serial port for wired communication or connecting wireless modules, ON/OFF switch, front and rear sonar panel rigs and indicator LED for battery and various operating modes like Wander or manual modes.

Preparation of input to fuzzy algorithm: Before implementation of the proposed algorithm the captured stereo pair of images are divided into subsections. The subdivisions are obtained as follows (Fig. 1 and 2):

- The left most and the following subsection of left stereo image is used to detect the possible movement of obstacle towards robot from left.
- The right most and the following subsection of right stereo image is used to detect the possible movement of obstacle towards robot from the right.
- The center subsection of both left and right stereo pair of image represents the safe movement of robot if the pixel fill-up increases proportional to the robot movement.
- Upper subsection off in left and right stereo image pair is used to detect the overhanging obstacles.
- The lower subsection in left and right stereo image pair decides the safe distance of obstacle from robot.

Development of Fuzzy Logic System: The computer/processor works on binary system i.e. either 0 or 1 (yes or no). Hence the decision taken by processor also behaves in the similar way. When it comes to real life decision making, it is not always a 0 or 1 decision and there are multiple factors affecting the possibility and probability of a decision. These probability cases cannot be addressed by normal 0 or 1 decision in processors. Thus, Fuzzy logic is a method, which takes the input data values and according to each input value, the output probability is decided. The fuzzy logic can be developed by mainly three methods:

- Mamdani
- Sugeno
- Tsukamoto

The fuzzy logic algorithm is developed using Fuzzy Logic Developer Toolbox in MATLAB (Fig. 9) which uses the minimum of all the input to get the probability of an event. There are six input values for the fuzzy logic system namely: LMO, LI, LMI, RMI, RI and RMO given in the form of linear membership function. So the total number of possible set of rules will be 2^6, which gives us 64 rule sets.

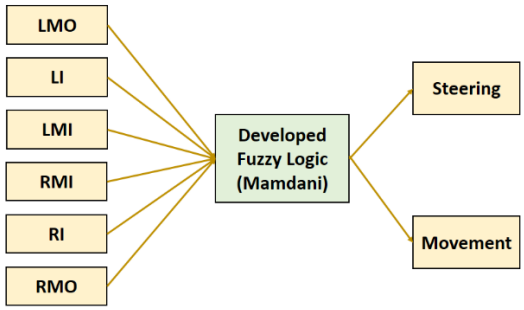


Figure 9. Fuzzy logic controller

The logics are decided based on normal human behavior for obstacle avoidance. The output will be in the form of Movement of vehicle and steering control. Usage of Mamdani fuzzy inference system requires defuzzification of the output to get the desired output. This is done by using outputs from fuzzy logic inference system. (Refer algorithm 1 and 2)

Integration of P3-DX Processor with MATLAB: MATLAB, Python or C can communicate with the Pioneer 3-DX onboard processor via serial port (Fig.

10(a).) with the use of wireless module and fixed IP address. This is done by the use of a remote client API Advanced Robot Interface for Applications (ARIA). ARIA supports direct interface between P3-DX processor and MATLAB (in laptop) by using serial to USB port converter. Before installation of stereo vision system, the P3-DX is tested in wander mode and manual mode by running the demo code available while installing ARIA. For the navigation of the vehicle, it takes linear and angular velocity as input. These velocities must be in a proper ratio and should not change drastically. To maintain these ratios, small changes in the value of the velocities is maintained to obtain a smooth transition.

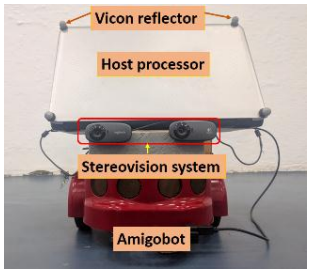


Figure 10(a). Autonomous vehicle (Amigobot).

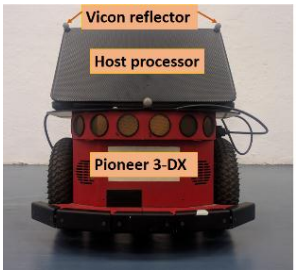


Figure 10(b). Dynamic obstacle (P3-DX).

Stereo vision system has been mounted and connected to the Laptop via USB. Laptop is also connected to serial port of pioneer 3-DX. Camera images are taken, partitioned and fed to the fuzzy logic input. The fuzzy logic output is then defuzzified for getting the angular and linear velocities. These velocities are then fed to Pioneer 3-DX processor, which then provides the velocity to each motor of the vehicle.

Tracking of Position: Validation of the algorithm requires us to ensure both the tracking of vehicular movement and the obstacles. For this purpose, Vicon IR camera based motion tracking system has been used (Fig. 11(b)). It uses IR sensors as a source and depending upon the reflection of infrared rays, the distance of the reflectors is calculated. Before tracking an object, cameras are placed so as to ensure the reflectors on the T rod are within its field of view. Once the rod is visible to all cameras, calibration is carried out. The cameras are calibrated by moving the calibration T rod at various positions within the range of Vicon camera system. The calibration parameters are saved and is used for further tracking of obstacles/vehicle keeping the same camera positions. The developed physical environment for testing of algorithm is shown (Fig. 11(a)).

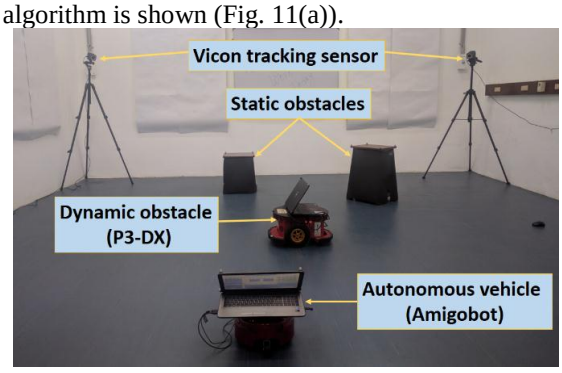


Figure 11(a). Actual experimental environment.

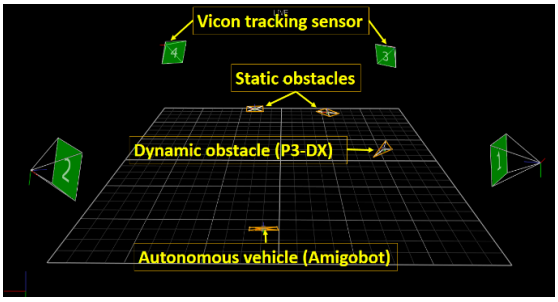


Figure 11(b). Experimental environment (virtual view using Vicon).

VI. Results and Discussion

A. Simulation Results

Static obstacles: The simulation environment has been set in V-rep by putting static (with sizes 38x38x47 mm^3 and 43x38x60 mm^3) and dynamic obstacles (P3-DX) along with the stereovision system mounted in mobile robot. Fig. 12 shows the static obstacle avoidance via developed algorithm in simulation environment developed in V-rep.

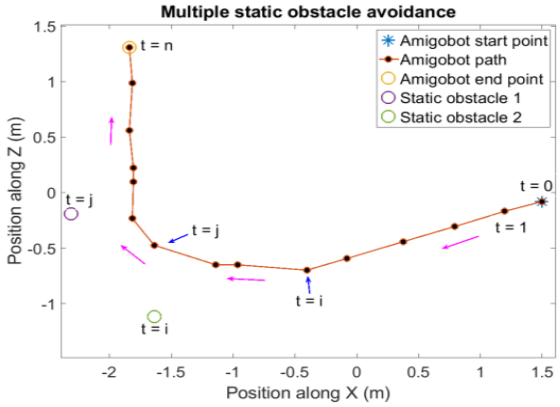


Figure 12. Simulation result for multiple static obstacle avoidance

Dynamic obstacle: For dynamic obstacle avoidance, P3-DX has taken to move from left towards right and vice versa w.r.t autonomous robot. Fig. 13(a) to 13(d) shows the avoidance of dynamic obstacle by using the developed algorithm.

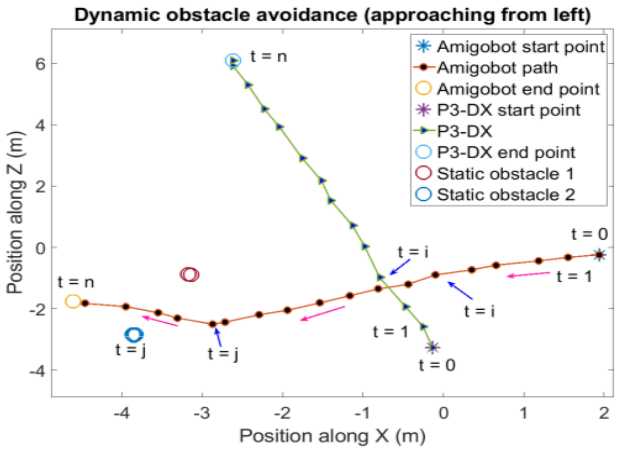


Figure 13(a). Simulation result for obstacle avoidance approaching from left

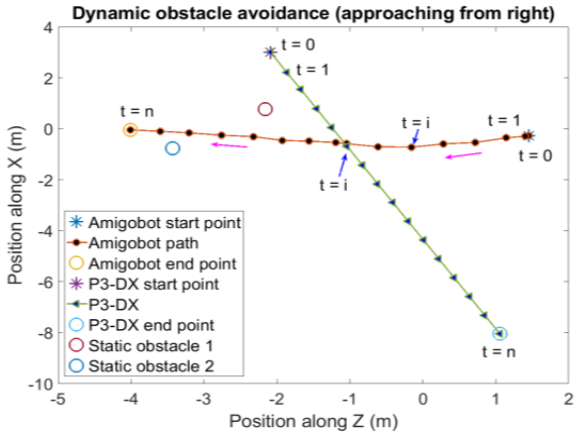


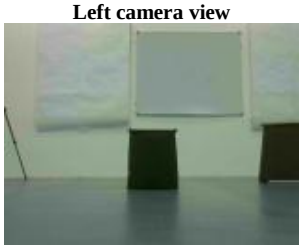
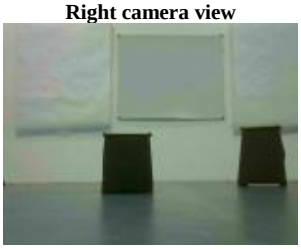
Figure 13(b). Simulation result for obstacle avoidance approaching from right

B. *Experimental Results*

The physical environment was developed and testing has been done in proper lighting condition and within the Vicon tracking region.

Static obstacle: Static obstacles (with sizes 38x38x47 mm^3 and 43x38x60 mm^3) were placed inside the tracking region of Vicon motion tracking system. Special IR reflectors supplied Vicon were placed on the both static obstacles and Amigobot vehicle; and this enables the developed software to detect & avoid obstacles and simultaneously the robot behavior while avoiding an obstacles. Table IV shows the detection and avoidance of static obstacle by the vehicle in a controlled laboratory environment.

TABLE IV. STATIC OBSTACLE AVOIDANCE BY AMIGOBOT

Left camera view	Right camera view
	
Two static obstacles seen.	
	
Amigobot vehicle approaching to obstacle 1 (decision made for avoidance).	
	
Amigobot approaching and avoiding obstacle 2.	

The Amigobot avoided the two static obstacles after detecting it at time t = i and time t = j and robot path is shown in the Fig 14. It is clear from the path taken by the robot that, it can detect the obstacles of various sizes & shapes from a distance ranging from: 1.5 to 2m and avoid them. The distance at which the robot should take the logical decision to turn depends on the initial pixels considered which depends on the size of the obstacle and the distance of it from the vehicle.

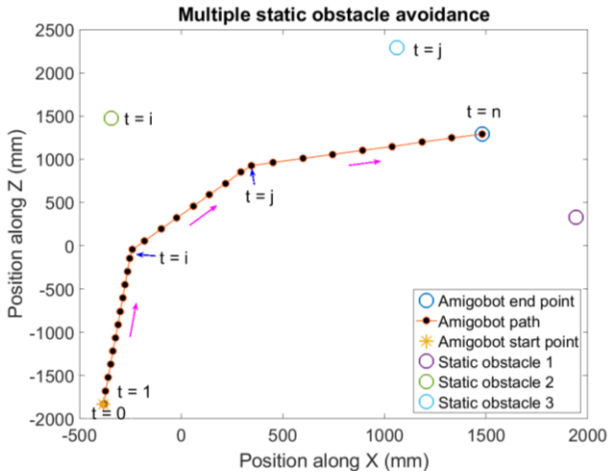


Figure 14. Path traced in Vicon tracker during multiple static obstacle avoidance.

Dynamic obstacle: The developed algorithm has also been tested for dynamic obstacle by mounting IR reflectors in it along with two static obstacles. Table V shows the movement of a dynamic obstacle approaching from left towards the vehicle and the logical decisions taken by Amigobot to avoid it.

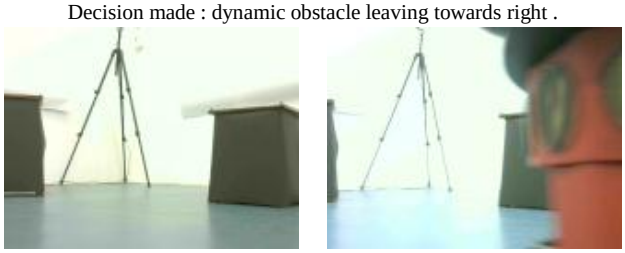
For the dynamic obstacle, Fig. 15(a) and 15(b) shows the path tracked by using Vicon tracking system for the right moving and left moving obstacle w.r.t. Vehicle position. It is clear from the plot that when the obstacle is moving right, the obvious decision of any vehicle is to turn left and vice versa, which is also validated by tracking the dynamic obstacle (P3-DX) and autonomous vehicle (Amigobot). The distance at which the Amigobot should make decision (time t = j) depends on the movement of P3-DX in the field of view of installed stereo vision system and the action of turning is based on the pixel fill-up and its proximity to autonomous vehicle. The dynamic obstacle was detected at an approximate distance of 1m and the control decision was taken to avoid it.

TABLE V. DYNAMIC OBSTACLE AVOIDANCE BY AMIGOBOT

Left camera view	Right camera view
	

Autonomous vehicle moving : dynamic obstacle approaching from left.

Decision made : dynamic obstacle leaving towards right .



Amigobot turned left and avoided obstacle.

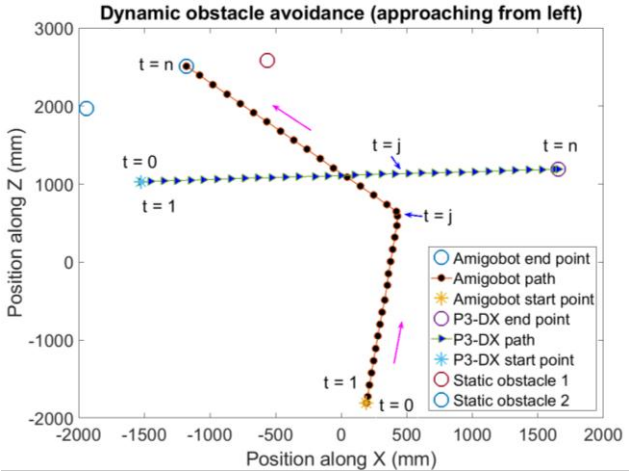


Figure 15(a). Path traced in Vicon tracker for dynamic obstacle avoidance (approaching from left).

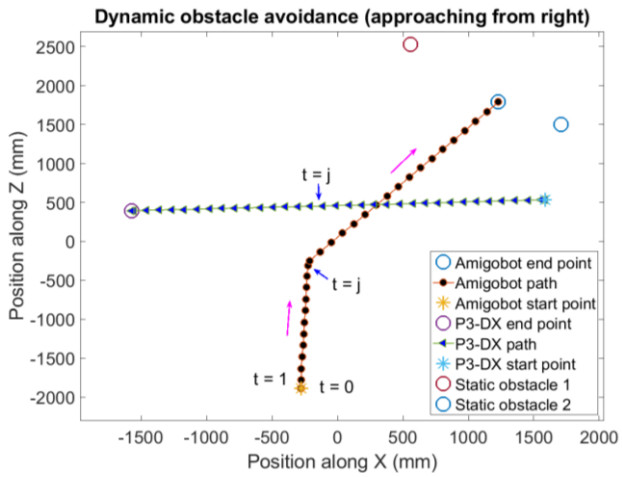


Figure 15(b). Path traced in Vicon tracker for dynamic obstacle avoidance (approaching from right).

Fig. 16 shows the other cases, where the robot is able to avoid the static and dynamic obstacle with a reasonable margin, when the avoiding of dynamic obstacle approached from right causes the visibility of a static obstacle. The control decision taken by Amigobot at both the instant (time t = i and t = j) depend on the speed of dynamic obstacle, nearness/far from the autonomous vehicle and the pixel fill-up corresponding to it.

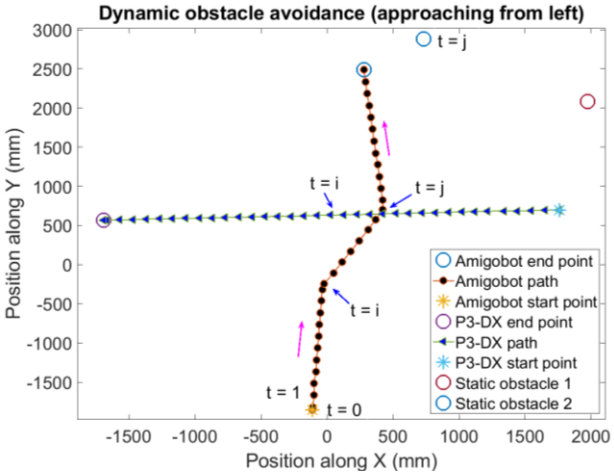


Figure 16. Path traced in Vicon tracker for static obstacle and dynamic obstacle avoidance approaching from right

VII. Conclusion and Future Scope

The developed fuzzy logic based obstacle avoidance system for autonomous navigation can avoid static as well as the dynamic obstacle. The system has been tested in the virtual environment using V-rep simulator. Similarly, it is validated experimentally in real time using Pioneer 3-DX robot keeping the same dimension for the static and dynamic obstacle. The path followed by the robot avoiding the static obstacle and dynamic obstacles has been traced using Vicon Motion Tracker system. The result shows that the performance of the algorithm in both the virtual and experimental environment under the various obstacle and environmental conditions is satisfactory.

Although the developed algorithm performed well in simulation and controlled indoor laboratory conditions, there are few limitations to it. Since the research presented in this paper depends purely on image processing and fuzzy logic, it highly depends on the lighting and the texture in the environment. The image processing capability requires further modifications and improvements for utility in outdoor navigation. The path planning and navigation have not been integrated as a part of this research, and this needs to be added to further improve the autonomous navigation.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

The research work is carried by Kirti Shankar Sharma under the guidance of P.V. Manivannan. This paper has contribution from both the authors. The research work including experimentation and paper writing is carried by Kirti Shankar Sharma, while the performance evaluation, analysis and paper review is done by P.V. Manivannan.

REFERENCES

[1] Y. K. Lee, J. M. Lim, K. S. Eu, Y. H. Goh, and Y. Q. Tew, "Real time image processing based obstacle avoidance and navigation system for autonomous wheelchair application," In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference* (APSIPA ASC), 2017, pp. 380-385. IEEE, 2017.

[2] T. Masahiro and S. Yuta, "Mobile robot navigation in indoor environments using object and character recognition," In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1, pp. 313-320. IEEE, 2000.

[3] Farag, A. Aly, and A. E. Abdel-Hakim, "Detection, categorization and recognition of road signs for autonomous navigation," in *Proc. ACIVS*, pp. 125-130. 2004.

[4] Z. Yuke, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3357-3364. IEEE, 2017.

[5] L. Yutian, D. J. Chen, and S. Z. Zhang, "Obstacle avoidance method based on the movement trend of dynamic obstacles," in *Proc. 2018 3rd International Conference on Control and Robotics Engineering (ICCRE)*, pp. 45-50. IEEE, 2018.

[6] K. Yeonsik, C. Roh, S. B. Suh, and B. Song, "A lidar-based decision-making method for road boundary detection using multiple kalman filters," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 11, 2012, pp. 4360-4368.

[7] I. Shahram, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton et al., "KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proc. the 24th Annual ACM Symposium on User Interface Software and Technology*, pp. 559-568. ACM, 2011.

[8] K. Kourosh, and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, 2012, pp. 1437-1454.

[9] S. Kahlouche and A. Karim, "Optical flow based robot obstacle avoidance," *International Journal of Advanced Robotic Systems*, vol. 4, no. 1, 2007, p. 2.

[10] S. R. Sahoo and P. V. Manivannan, "A hybrid approach for Dynamic Observer to Detect and Track Dynamic Obstacles," *International Journal of Machine Learning and Computing* vol. 10, no. 2, pp. 393-399, 2020.

[11] M. Gonçalo, C. Premebida, P. Peixoto, and U. Nunes, "Tracking and classification of dynamic obstacles using laser range finder and vision," In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1-7. 2006.

[12] M. Xiaodong, S. M. Li, and H. Shen, "On-board lane detection system for intelligent vehicle based on monocular vision," *International Journal on Smart Sensing & Intelligent Systems*, vol. 5, no. 4, 2012.

[13] S. Kohtaro, M. Fukuchi, J. S. Gutmann, T. Ohashi, K. Kawamoto, and T. Yoshigahara, "Obstacle avoidance and path planning for humanoid robots using stereo vision," In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 1, pp. 592-597. IEEE, 2004.

[14] Sharma, K. Shankar, S. R. Sahoo, and P. V. Manivannan, "A hybrid vision system for dynamic obstacle detection," *Procedia Computer Science*, vol. 133, 2018, pp. 153-160.

[15] S. Alessandro, "The uses of fuzzy logic in autonomous robot navigation," *Soft Computing*, vol. 1, no. 4, 1997, pp. 180-197.

[16] Beom, H. Rak, and H. Cho, "A sensor-based obstacle avoidance controller for a mobile robot using fuzzy logic and neural network," In *IEEE IROS*, pp. 1470-1475. IEEE, 1992.

[17] M. Jeff, A. Saxena, and A. Y. Ng, "High speed obstacle avoidance using monocular vision and reinforcement learning," In *Proc. the 22nd International Conference on Machine Learning*, pp. 593-600. ACM, 2005.

[18] Zadeh, A. Lotfi, "Fuzzy logic, neural networks, and soft computing," in *Fuzzy Sets, Fuzzy Logic, And Fuzzy Systems: Selected Papers by Lotfi A Zadeh*, pp. 775-782. 1996.

[19] S. Uluc, M. Buehler, and D. E. Koditschek, "RHex: A simple and highly mobile hexapod robot," *The International Journal of Robotics Research*, vol. 20, no. 7, 2001, pp. 616-631.

[20] M. Don, and J. J. Little, "Using real-time stereo vision for mobile robot navigation," *Autonomous Robots*, vol. 8, no. 2, 2000, pp. 161-171.

[21] Pannu, G. Singh, M. D. Ansari, and P. Gupta, "Design and implementation of autonomous car using raspberry Pi," *International Journal of Computer Applications*, vol. 113, no. 9, 2015.

Kirti Shankar Sharma was a MS research scholar in the department of mechanical engineering at IIT Madras, India. Presently, he is working as a Deputy Manager, Amara Raja Battery Limited, Tirupati, India. He did his B.E. in Electrical and Electronics Engineering from Government engineering college, Raipur, Chhattisgarh, India in the year 2014. In IIT Madras, his research work was on vision-based navigation of autonomous vehicles. His research interest includes: robotics and computer vision.

P.V.Manivannan received his Ph.D in Control System for SI Engines from IIT Madras, India and Master in Applied Electronics from College of Engineering, Anna University, Chennai, India. Currently, he is an Associate Professor at Department of Mechanical Engineering, Indian Institute of Technology Madras, Chennai, India. He has also taught summer term course "Mechatronic Systems" in University of Nebraska, Lincoln, USA and University of Kaiserslautern, Germany as a visiting faculty. He is also a recipient of prestigious DAAD fellowship and ERASMUS MUNDUS Teaching fellowship. His teaching / research interests includes: Mechatronics, Robotics, Automotive Control Systems, Embedded System Design, Sensor Network.