Platooning of Pioneer P3-DX robots using ROS and SICK laser sensor

Amol Vagad, Aniket Pendse, Debraj Bhattacharjee, Shishir Pagad
Team Yantra
University of Minnesota-Twin Cities

***Abstract*—It is aimed in this project to emulate a scaled version of real life connected vehicles which will move in a platoon formation to enhance safety, efficiency, and energy conservation. This project has two robots: one leader and one follower. Leader, as the name indicates, is the brain of the robot platoon and it makes driving decisions, thus leading the way. The follower on the other hand, follows the leader by imitating the actions of the leader.**

This is realized using Pioneer P3-DX robots having SICK LMS-200 laser sensors mounted on them. The interfacing of the robots and the sensors is done using Robot Operating System (ROS).

***Keywords*—Platooning; Mobile robots; Laser scanner; Pioneer P3-DX**

I. Introduction

The research on the development of autonomous vehicles in the last few years has created new horizons in Intelligent transportation systems. The potential changes due to the introduction of such systems have deep impacts on socio-economic life. These include enhanced safety and mobility of passengers and cargo, reduction in global emissions as well as increased savings in energy. Platooning of autonomous vehicles is one of the pertinent topics in this area of research.

The Platooning of mobile autonomous robots involves a formation which consist of one leader robot and one or more follower robots. Each of the follower robots must follow the preceding robot to hold the formation.

Several researchers have conducted studies on real-time platooning of mobile robots in the past few years. The authors present a novel control and decisional architecture to facilitate smooth and safe motion in the *Pratiele* autonomous vehicle project [1]. One of the issues encountered in this approach is that of weakly cut turns in the lateral control phase. In [2], a distributed robotics application is presented in which a group of mobile robots are coordinated to form a platoon. Obstacles are modeled as discs and navigation is done using the Gradient tracking algorithm. The focus of this work is on a real-time operating system that allows an assessment of real-time parameters on the performance of control application. In [3], a laser-based simultaneous localization and mapping (SLAM) method is presented for a leader-follower platooning of multiple robots. Follower robots develop occupancy grid map using leaders scan data and own readings, which are further used to generate target paths for each robot. Finally, a tracking controller, which maintains a constant gap between two successive robots is designed based on Ackerman geometry. A review on the platooning of trucks is presented in [4]. The author asserts that the benefits of Platooning of trucks are several including: Improvement in safety, reduction in personnel costs, energy savings due to a reduction in fuel consumption etc.

In this project, two mobile robots are arranged in a platoon formation. The leader robot has autonomous maneuvering capability. The follower bot maintains a constant gap from the leader bot and replicates the trajectory of the leader bot. The entire operation is accomplished using laser sensors.

Section II describes the Problem definition and the hardware/software used in this project. Section III describes the approach proposed to tackle the platooning problem. Section IV describes the results obtained while implementing the platooning system. Finally, conclusions are provided in section V.

Fig. 1. A Pioneer P3-DX robot.

II. Problem definition

A. Main Objective

This work considers the platooning of two Pioneer P3-DX mobile robots. An indoor navigation robot platoon system consisting of autonomous mobile robots. One of them acts as the formation leader while the other acts as a follower. The leader robot travels in the center of an alley maintaining a particular distance from a preferred wall. The follower maintains an appropriate distance from the leader robot throughout the course of motion. This is done to avoid crashes with the leader. The follower uses dynamic linear and angular velocities to achieve this desired gap between the leader and the follower.

Figure 1 shows a typical Pioneer P3-DX robot. The robots are programmed using ROS (Robot Operating System) [5] on a PC running LINUX operating system. Autonomous navigation on each robot is achieved using SICK LMS-200 Laser sensor that avoids obstacles to move the robot autonomously. The follower robot uses data from the laser sensor to identify the leader robot.

Both robots can move in a 2-D plane. The laser sensor is capable of producing 2D scan images. It is assumed that there is no external interference.

B. Assumptions

During the implementation of the platooning system the following things are assumed:

- The platooning system takes place inside an indoor hallway
- The hallway has uniform walls
- No doors are open in the hallway during platooning
- There are no obstacles having dimensions similar to the leader between the leader and the follower robot
- The side of the wall to be followed is given a preference
- The leader is always in the field of vision of the follower

C. Sensor Information

The field of vision of the SICK LMS-200 is 180 degrees. It has an angular resolution of 1 degree. The sensitivity of the LMS-200 to environmental factors during outdoor operation is reduced using Pixel-oriented evaluation [6]. Figure 2 shows a SICK LMS-200 laser sensor. The scanning range of the laser sensor is 80 m. It is a class 1 laser sensor which has a resolution of 1 mm. Detailed specifications of the SICK LMS-200 sensor are presented in Table I.

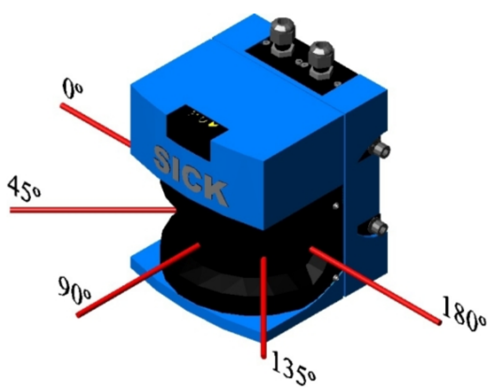

Fig. 2. A SICK LMS-200 sensor.

D. Software Packages

Robot Operating System (ROS) has been used for programming the platooning system. The interfacing between the robot and the computer is mainly carried out by the *RosAria* package. It helps to give USB permissions to the robot for hardware connections and provides passage for uploading the code on the robot. Python has been used to implement the code. ROS has special rospy package which can be used while programming in python. To visualize the sensor data the *Rviz* package has been used. The *Rviz* package belongs to ROS and is used for visualization of data from sensors in 2D and 3D environments. Screenshots of the scanning process are shown in Figures 3 and 4.

TABLE I
SPECIFICATIONS OF SICK LMS-200 LASER SENSOR[7]

Feature name	Feature value
Model Name	LMS200-301016
Part Number	1015850
Field of view	180°
Angular Resolution	1°
Response Time	13-53 ms
Resolution	10 mm
Systematic error	+/- 15 mm
Laser class	Class 1
Enclosure Rating	IP 65
Ambient Operating Temperature	0°C - 50°C
Scanning range	80 m
Data Interface	RS-232, RS-422
Supply voltage	24 V DC +/- 15%
Power consumption	20 W
Storage temperature	-30°C to 70°C
Weight	4.5 kg
Dimensions (L x W x H)	156 x 155 x 210 mm

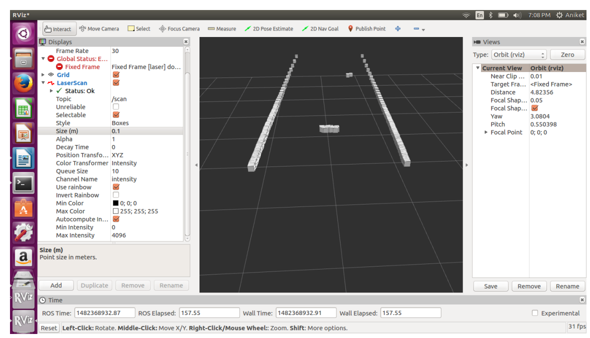

Fig. 3. Visualization of walls using Rviz package

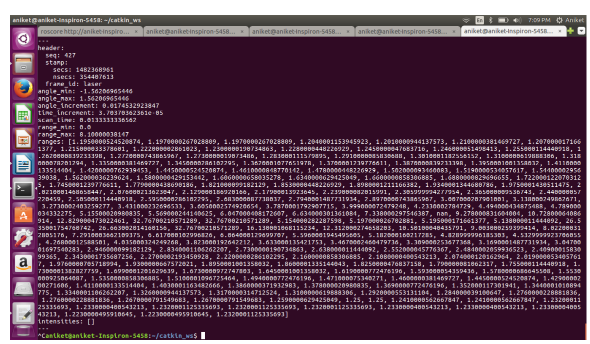

Fig. 4. Raw laser data obtained by using */scan* topic

Algorithm 1 Algorithm for wall following in leader bot

Input: Laser data 30°(Right wall) or 150°(Left wall)
Output: Motor actuation

Give preference to a particular wall

Determine zone in which leader robot is presently in
if (In zone 1) **then**
  Move forward with linear velocity = 0.2 m/s
  Move right with angular velocity = -0.15*direct rad/s
else if (In zone 2) **then**
  Move forward with linear velocity = 0.2 m/s
  Move right with angular velocity = -0.10*direct rad/s
else if (In zone 3) **then**
  Move forward with linear velocity = 0.2 m/s
  Move right with angular velocity = -0.05*direct rad/s
else if (In center zone) **then**
  Move forward with linear velocity = 0.2 m/s
  Don't rotate on any side
else if (In zone 5) **then**
  Move forward with linear velocity = 0.2 m/s
  Move left with angular velocity = 0.05*direct rad/s
else if (In zone 6) **then**
  Move forward with linear velocity = 0.2 m/s
  Move left with angular velocity = 0.10*direct rad/s
else if (In zone 7) **then**
  Move forward with linear velocity = 0.2 m/s
  Move left with angular velocity = 0.15*direct rad/s
end if

III. Proposed Approach

A. Leader Robot

The leader robot has two main features: wall-following and obstacle avoidance. The block diagram of the leader system is shown in Figure 5. Each of these topics are described below:

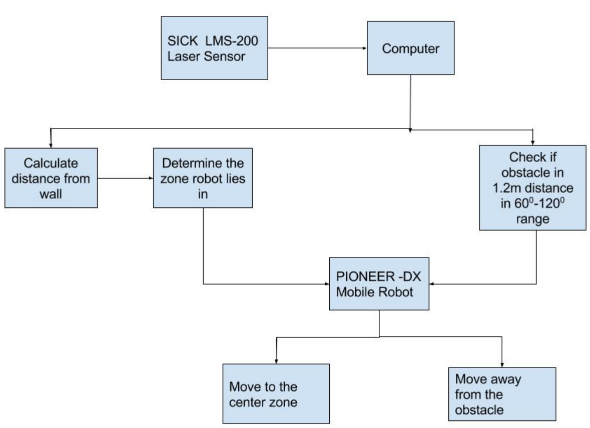

Fig. 5. Block Diagram of leader robot

1) Wall following: The leader robot uses wall-follower algorithm to autonomously navigate through hallways. The leader robot is set to follow left or right wall based on uniformity of wall and direction of turns. The hallway has been divided into 7 zones from the left to the right wall; the zones are based on distance from the left wall. If the robot is following the left wall, it uses 150° laser ray to measure its distance from the left wall, else it uses 30° to measure its distance from the right wall. Laser rays at angle 150° and 30° have been chosen to prevent the robot from misjudging distance from the wall when making a turn towards the wall.

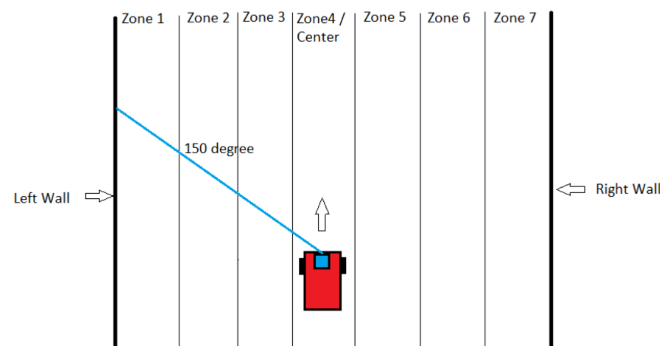

Fig. 6. Scanning process of leader robot

The leader robot tries to stay in the center zone i.e. zone 4, and as long as it is in the center zone, it moves straight. If the robot is in any zone other than the center, the angular velocity of the robot is calculated proportional to the distance of the robot from the center zone. For example: if the robot is following the left wall and it is in zone 1 i.e. very close to the left wall, the robot takes a steep right turn to maneuver itself away from the left wall and move closer to the center zone. The angular velocity gradually reduces as the robot gets closer to the center zone. This prevents the robot from oscillating about the center zone and helps it make a smooth turn into the center zone. The description of the wall following procedure is shown in Algorithm 1. Figures 5 to 8 describe the motion of the leader.

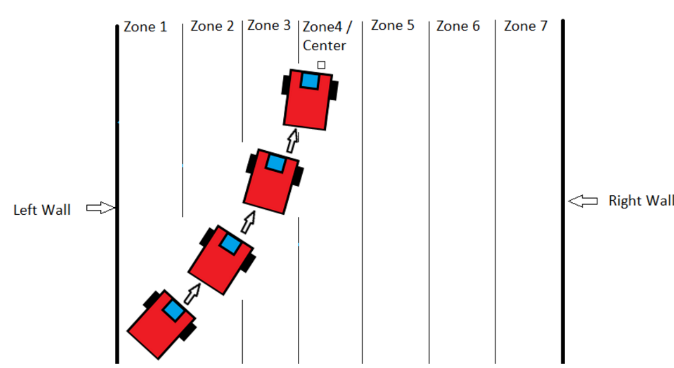

Fig. 7. Turning of leader robot

2) Obstacle detection and avoidance: The capability to maneuver around obstacles is of prime importance in autonomous navigation. In this study, the leader robot detects obstacles and maneuvers around them using segmentation. To implement the obstacle avoidance system, it is determined first whether there is any obstacle in the 60° to 120° laser data range. If something does exist in this zone, the zone is, further, subdivided into three categories: center, left and right zone, each having 20°. The subsequent step is the determination of the zone in which the object lies closest to the leader robot. Once determined, the leader moves in a direction which takes it away from the obstacle. A paradoxical situation occurs if the center zone is the one in which the object is closest to the leader. The leader, at this moment does not know the direction it has to move to get away from the obstacle. In this case, the zone having more free space is given preference and the leader is made to move in that zone. The procedure is shown in Figure 8. A description of the obstacle avoidance algorithm is shown in Algorithm 2. A picture of the Leader-Follower assembly is shown in Figure 10.

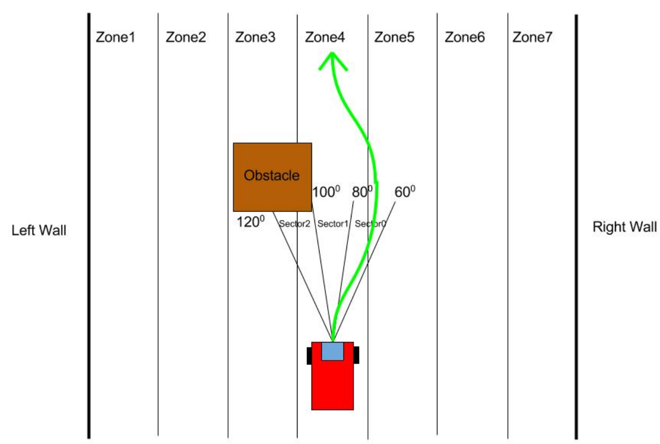

Fig. 8. Process of obstacle avoidance

Algorithm 2 Algorithm for obstacle detection and avoidance in leader bot

Input: Laser data in range 60° to 120°
Output: Motor actuation

Determine if an obstacle present in 60° to 120° range

$Obstacle = False$
for $i = 60$ to 120 **do**
  if ($laserRange[i] < 1.2$) **then**
    $Obstacle = True$
  end if
end for
if ($Obstacle = True$) **then**
  Determine sub-zone(60°-80°, 80°-100°, 100°-120°) in which obstacle is closest
  if ($Zone = Left$) **then**
    Move right
  end if
  if ($Zone = Right$) **then**
    Move left
  end if
  if ($Zone = Center$) **then**
    Determine whether left or right zone has more space
    Move to the corresponding zone
  end for
end if

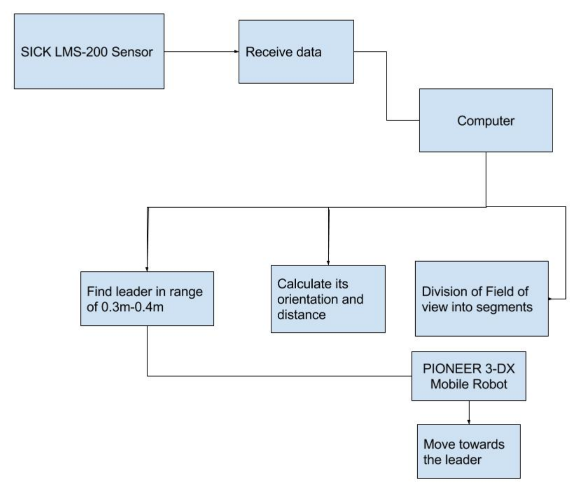

Fig. 9. Block diagram of follower robot

B. Follower

The follower knows the orientation of the Leader through the polar coordinates of the points of the Leader which is a part of the laser data. The follower then calculates the distance of the Leader with respect to itself and the angle between the Leader axis and its own axis. Now the objective of the follower is to orient behind the Leader at a certain distance with its axis aligned with that of the Leader. The follower has to move a distance of L and orient by an angle theta. The follower therefore, moves with an angular velocity proportional to theta and a linear velocity that is proportional to the distance L. The block diagram of the follower system is shown in Figure 9.

IV. Experimental Results

Using the above approaches, it is found that the platooning system works effectively and efficiently in several test situations. While moving through the hallways, the platoon system follows the path along the center of the hallways with very little oscillations about the center zone. Also, the platoon system takes effective turns without colliding walls while turning. The leader robot is capable of detecting obstacles in its path and maneuvering around them. However, there were a few instances where it was unable to maneuver around an obstacle which was very close to the wall.

The follower robot detects the leader robot correctly for

Fig. 10. The Leader-Follower assembly

Algorithm 3 Algorithm for the follower robot

Input: Laser data in range 0° to 180°
Output: Motor actuation

Determine the position of the Leader

$Leader = False$
for $i = 0$ to 180 **do**
  if $(laserRange[i+1] - laserRange[i] > 0.5)$ **then**
    Store the index i in diff
  end if
end for
for $i = 0$ to $len(diff)$ **do**
  $(x1Coor, y1Coor) = (laser(diff) * cos(diff), laser(diff) * sin(diff)$
  $(x2Coor, y2Coor) = (laser(diff+1) * cos(diff+1), laser(diff+1) * sin(diff+1))$
  $dist = d((x1, y1), (x2, y2))$
  if $0.3 < dist < 0.4$ **then**
    Store x1Coor,x2Coor,y1Coor,y2Coor in x1,x2,y1,y2
  end if
end for
for $i = 0$ to $len(x1)$ **do**
  $dist[i] = d((x1[i], y1[i]), (x2[i], y2[i]))$
  if $dist[i] > dist[i+1]$ **then**
    $distclose = dist[i+1]$
    $xclose = (x1[i+1] + x2[i+1])/2$
    $yclose = (y1[i+1] + y2[i+1])/2$
  end if
end for
$theta = atan2(yclose/xclose)$
$angvel = theta - 90^\circ$
$r = distclose/(2 * sin(theta/2))$
$linvel = r * theta$

most of the time while moving through the straight hallway. But there are a few instances where it detected a different object with the same width as the leader and did not follow the leader. Also, if the leader robot is very close to the wall, the segmentation algorithm does not work as expected and the follower robot stops following the leader.

V. Conclusion and Future work

The objective of robot platoon system was achieved along with an additional feature of obstacle avoidance. The follower robot only has information about its own orientation and distance, azimuth of the leader robot. By the use of visual sensors and image processing the results can be enhanced.

The system can also be implemented in outdoor environments by developing suitable algorithms. Better recognition of the leader can be implemented by using algorithms like Hough Transform.

Acknowledgement

We are highly indebted to Professor Roumeliotis and his Teaching Assistant Ahmed Medhat for their invaluable guidance and support throughout the course of the project.

References

1 Laugier, C., Paromtchik, I., and Parent, M., "Developing autonomous maneuvering capabilities for future cars," in *Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on*. IEEE, 1999, pp. 68–73.

2 Della Vedova, M. L., Facchinetti, T., Ferrara, A., and Martinelli, A., "Real-time platooning of mobile robots: design and implementation," in *2009 IEEE Conference on Emerging Technologies & Factory Automation*. IEEE, 2009, pp. 1–4.

3 Esaka, Y., Enomoto, H., Hashimoto, M., and Takahashi, K., "Platooning method for multiple mobile robots using laser-based slam," in *SICE Annual Conference (SICE), 2011 Proceedings of*. IEEE, 2011, pp. 251–254.

4 Tsugawa, S., Jeschke, S., and Shladover, S. E., "A review of truck platooning projects for energy savings," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 68–77, 2016.

5 O.S.R.F. Robot Operating System (ROS) wiki. [Online]. Available: http://www.ros.org

6 Waldkirch, S. A. (2006) Lms200/211/221/291 laser measurement systems: Technical desciption. [Online]. Available: http://sicktoolbox.sourceforge.net/docs/sick-lms-technical-description.pdf

7 Sevick, K. Technical Specifications of SICK LMS-200 sensor. [Online]. Available: http://www.pages.drexel.edu/~kws23/tutorials/sick/sick.html