Multi Robot Environment Exploration Using Swarm

Hardik Gossain, Bhavya Sharma, Rachit Jain, and Jai Garg

Abstract This chapter describes the exploration of unknown environment using multiple robots. We designed this system in order to overcome the various disadvantages of exploration and mapping robots. When it comes to the mapping of large unknown environment, single robot would be inefficient and inaccurate. It leads to more exploration time and inaccurate mapping of the environment. There are multiple algorithms which already exist and are used but seeing their inefficiency we have worked on RRT that is random-exploration random tree. Multiple robots will be deployed in an unknown environment and based on which the robots will explore environment, to make robot communicate with each other we have implemented swarm algorithm so that robot communicate with each other in coordinated way. The Swarm algorithm will help robots explore the environment in a coordinated way because of which the two robots will never explore the same area repeatedly and hence will same time and will lead to more accurate mapping of the environment. To implement this solution, we used ROS as the middle ware, with the help of the ROS we can encounter and handle many real time parameters which will make our system more accurate and efficient. GAZEBO simulator is used in order to test and implement the system. This multi robot system will be very useful in the future smart city ecosystems.

Keywords Multiple robots · Exploration · RRT · Random-exploration · Swarm · Middle-ware, · ROS · GAZEBO

H. Gossain · J. Garg
Department of Electronics and Communication Engineering, Dr. Akhilesh Das Gupta Institute of Technology and Management, New Delhi, India

B. Sharma (✉) · R. Jain
Department of Electrical and Electronics Engineering, Dr. Akhilesh Das Gupta Institute of Technology and Management, New Delhi, India

V. Piuri et al. (eds.), *AI and IoT for Smart City Applications*, Studies in Computational Intelligence 1002, https://doi.org/10.1007/978-981-16-7498-3_11

1 Introduction

Robotics is the field of attraction for many researcher and hobbyist, in recent decade the robotics saw development, advancement and technical growth. From mobile robot to UAV i.e. Unmanned aerial vehicle has wide field of research opportunity. All humans want their work load to decrease. We are in search of everything automatic and robotic that can help humans make their lives easier. Currently, autonomous robots are capturing the market and are the centre of attraction for many hobbyists and researchers. Many advancement is going on in the field of autonomous robots. Autonomous mobile robot is wide application in the field of exploration of unknown environment such as mining, disaster management and many more [1–5].

Autonomous robots are useful in many ways, they are helping hand for humans, in various way they make themselves useful in various activities such as mapping and navigation of unknown environment, there are various programmes and algorithm for the autonomous robots which are currently used but they are not efficient when it comes to mapping of the unknown environment which has large area, they take more time and the final map is not accurate hence to overcome this we have implemented an algorithm to make multiple robot work in coordinated manner and increase the efficiency. Using ROS as middle ware to implement RRT and swarm algorithm on our robots we have designed this solution [6–10].

The RRT i.e. random-exploration random tree is implemented in this chapter. Rapid-exploration Random Tree also known as RRT is an algorithm used for exploring unknown map. It is used in non-convex searches where it randomly builds space filling trees. This algorithm is the base of our research. As explained in [1] it generates random maps and find the path closest to it, that is obstacle free and has been explored. Since we are also using the IR sesnor the output of data is in the form of 0 and 1 and -1 in our algorithm, 0 signifies that the path is free from obstacles, 1 signifies the obstacle in the path, and -1 signifies that the path is unexplored. To make our robot more efficient and faster, we have made multiple robots communicate to each other rather than just treating each other as obstacle. Considering each other as obstacle grows possibilities that another robot can revisit positions on the map that other robot has already visited and made its map in the master node.

Simulation result obtained from the above proposed solution is compared with the other standard algorithms already available. Since the main aim of our research was to minimize the exploration time we have compared exploration time with the standard algorithm and also with exploration time of the multiple robot in comparison to single robot [11–15]. We wanted to give reference of some of the research work done in field of swarm [16–20]. We also took into consideration the exploration area of the robot as when exploration using a single robot was not accurate and sometimes do not some area, while testing our solution we tried to maximize the exploration area. Various algorithms were tried for merging of the map and the best algorithm is presented in this chapter.

This research chapter is organized as follow. Section 2 challenges and solution explain the challenges faced during designing of the algorithm and its solution.

Section 3 describes the methods adopted while designing this solution which includes explanation of "RRT', "SLAM" and "SWARM". Section 4 gives the brief about how the algorithm is implemented in simulation and how practically we can approach it further. Section 5 gives the overall result of the research chapter followed by Sect. 6 Conclusion.

2 Challenges

2.1 Task

The system involved in [1] is improved for improving efficiency and reducing time in map generation. Multiple robots working independently based on SLAM algorithm are used for mapping unknown locations.

2.2 Problem

The main problem caused by the algorithm given in [1, 2, 5] is that robots doesn't communicate with each other and treat each other as a dynamic obstacle. Main element of the algorithm could be improved by reducing the runtime and letting a single robot cover a particular part of the unknown locations.

2.3 Solution

Our system focuses on the communication of robots along with navigation and mapping of an unknown environment. Along with SLAM algorithm and RRT exploration, our system involves SWARM algorithm. This allows robots to know which robot is near to them and in which direction. The information of direction gives a probability that region on that particular is already been mapped. Apart from this, a single ROS node runs on the main processing unit that subscribes to all other nodes (robots) and generates map by taking multiple inputs. Independent robots are based on hector slam.

3 Method

3.1 RRT Exploration

- Unknown Region is the space which is not explored by robot or it's sensors
- Known Region is the space which is explored by robot—Open-Space is a known region where there are no obstacles.
- Occupied-Space is a known region which contains an obstacle.
- Occupancy Grid is a grid representation of the whole space. Its cell holds a value that defines the probability of it being occupied.
- Frontier is the segment that separates known (explored) regions from unknown regions. Generally, a frontier is a set of unknown points with at least one point in neighbour with Open-space.

Algorithm 1: RRT Exploration

```
Qgoal
Counter=0
lim=n
G(V<E)
While counter<Lim
Xnew=RandomPosition()
If IsInObstacle(Xnew)=True:
Link=Chain(Xnew,Xnearest)
G_append(Link)
If Xnew in Qgoal:
Return G
Return G
```

3.2 Frontier Detection

There are two modules for successful detection of a frontier. The above Algorithm is an independent process that resets after detection of each new node in unknown region. It is a sub-loop that is a part of the main loop that stores the detected frontiers and form the whole tree. In the inner loop, the code starts with an initial vertex v and the edge is set $E \leftarrow \varphi$, at each iteration a random point x_{new} is sampled that is the part of unknown region. The first vertex nearest to the random point is found called the $x_{nearest}$. Then it is checked that whether the new point or any point on the line segment x_{new} and $x_{nearest}$ lies in the unknown region using the Grid Check function. If either condition is true, frontier point is assigned. One more condition for obstacle detection is checked. If the line segment is obstacle free, then the point x_{new} is set as the new frontier and everything is reset, setting the current point as V ($V \leftarrow x_{current}$)

for the new iteration. This method will run as the robot traverses through the map and finding new frontiers.

This loop stores the current path of the robot to the new point while the main loop stores the whole map that has already been traversed including the current. This approach increases the overall speed of the map exploring algorithm. With multiple robots, each running this individually, the map is recorded and sent to the main ROS node which by filtering methods stores the whole map in a grid form. This algorithm plus multiple robot's implementation makes the mapping process really fast, efficient and time saving. Here we have implemented this on ground robots but it can be even implemented on drones.

Algorithm 2: Frontier Detection

$V \leftarrow x_{init}$; $E \leftarrow \varphi$;
while True do
$x_{rand} \leftarrow$ Sample Free;
$x_{nearest} \leftarrow$ Nearest $(G(V, E), x_{rand})$;
$x_{new} \leftarrow$ Steer $(x_{nearest}, x_{rand}, \eta)$;
if GridCheck(map, $x_{nearest}$, x_{new})=−1 then
PublishPoint(x_{new});
$V \leftarrow x_{current}$; $E \leftarrow \varphi$; ◁ reset the tree
else if GridCheck(map, $x_{nearest}$, x_{new})=1 then
$V \leftarrow V \cup \{x_{new}\}$; $E \leftarrow E \cup (x_{nearest}, x_{new})\}$;
end if
end while

3.3 *SLAM*

The whole project revolves around Simultaneous Localization and Mapping and the improvement made in the traditional method to increase efficiency and speed of map generation using a greater number of robots. RRT based algorithm has been applied for finding unknown regions in the environment. The map generation is totally based on ROS hector slam. Hector SLAM is a very efficient tool to produce a graphic map for the user to further command and make the robot perform particular tasks. The generated map can have only three possibilities, Unknown, Occupied-space, free-space represented by −1, 1, 0, respectively [10, 11].

The exploration of unknown area uses RRT algorithm as explained in Sect. 2.1. After map generation, there were many path planning algorithms to move from one point of the known location to other. We had the option of using A* but we instead used Dijkstra algorithm for making our work simple as our main goal was to implement multiple robots. Dijkstra finds the smallest path between the robot's current position and the goal position. Even though our occupancy grip has the data of obstacles, still we implemented bug 2 algorithm that is a greedy algorithm that follows a path nearer to the goal location when there is an obstacle in between. This

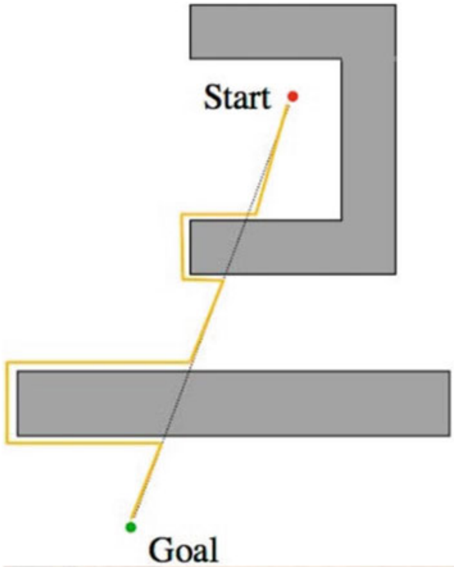

Fig. 1 Robot objective to reach goal while avoiding obstacle

was included mainly for avoiding dynamic obstacles or newly added obstacles which were not included while the generation of the map (Fig. 1).

3.4 Swarm

Swarm algorithm is responsible for establishing communication between multiple robots and make them work together to increase efficiency and decrease time to complete any task [3, 6]. In its usual implementation, generally one of the robots from the swarm is treated as the leader and other units become the slave and follow the master's command. In our implementation, we have made the main processing unit (computer) as the master node which gives command to other mobile robots to move around the arena. The master sends and receives data from other robots using Wi-Fi but other robots communicate to each other using infrared sensors. Four infrared transmitters and receivers are placed at each quarter of each unit. They constantly produce data containing a unique address assigned to each individual robot. By this method whenever there will be two robots' side to side or front back, they would be able to know that which robot is on which of their sides. This information will make the robot to avoid the path that is already been traversed by other robots. The algorithm also makes the robot able to distinguish between an obstacle and other moving robots [8].

3.5 Odometry Position

After the frontier detection, the main task revolves around detecting the current position of the robot. One cannot rely on image processing and detecting common points to find the position of the robot. Odometry data is very important for building an error-less map for the robot traversal. There are various ways to do so, we considered using Accelerometer and Encoders. One of them is enough by in cases where slipping of the robot can cause an error in encoder readings, we used accelerometer along with it. We avoided individual use of the Accelerometer as the probability of error in it is more as compared to digital encoders.

We cannot directly use the data from these sensors to be used as the input for ROS. We have to find the position of the robot using both the reading and then give it as an input for ROS. To do so we used a microcontroller (Namely STM32F103C8) for the collection of data and further calculation of the position of the robot.

As shown on [11], we used the motion model to calculate the position of the robot from the encoder's input. As told earlier, the probability of errors in the accelerometer is more so while applying the Kalman filter, the weightage of the accelerometer was 30% as compared to 70% of encoders approximately. Internally, the accelerometer is only considered when there is a large dissimilarity in the encoder's predicted reading along with the difference in accelerometers readings.

The overall process proceeds as followed, the microcontroller records the data, then calculates the position from both the sensors, and finally sends the position data to ROS (Fig. 2).

4 Implementation

Above stated solution was implemented on GAZEBO software with the help of which we were able to demonstrate the working of the RRT and were able to handle robot in unknown environment [9].

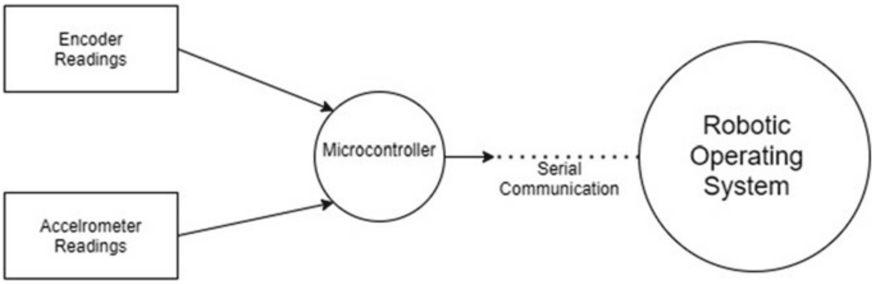

Fig. 2 Calculation of robot position

4.1 Simulation

Gazebo stimulator was used along with ROS as the middle ware. Software setup required for the stimulation is as follow:

1. Ubuntu 18.04 O.S
2. ROS melodic
3. GAZEBO
4. RVIZ.

Initially for the testing purpose Sample world was imported in GAZEBO and with the help of the ROS two robots were simultaneously spawned into the sample world (Fig. 3).

After the robots were spawned in the world, with the help of the LIDAR scanner installed on the robots started mapping and navigating through the environment autonomously (Fig. 4).

The final map of the environment which was mapped by the robots is shown below. With the help of the multiple robots we were able to map the unknown environment. The technique used while mapping the environment was hector slam.

With the help of the hector slam mapping we got more accurate result because hector slam mapping the mapping also consider robots localization which make it the accurate mapping technique.

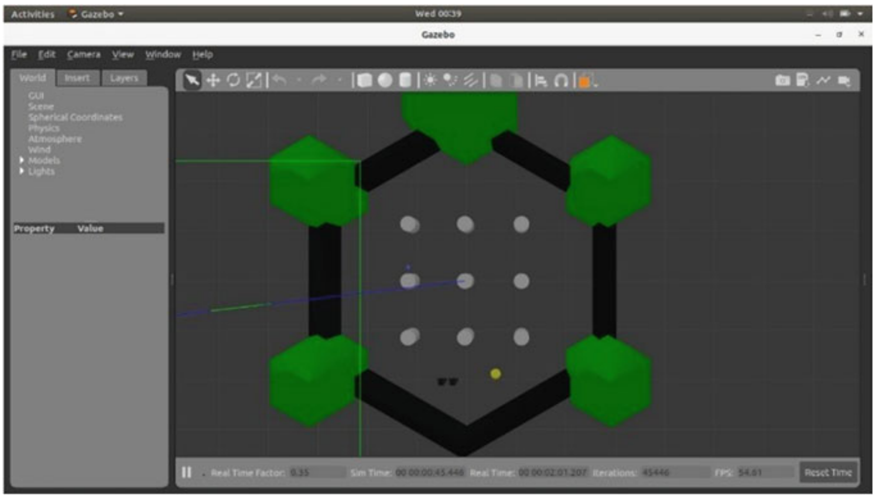

Fig. 3 Two robots spawned in simulation world

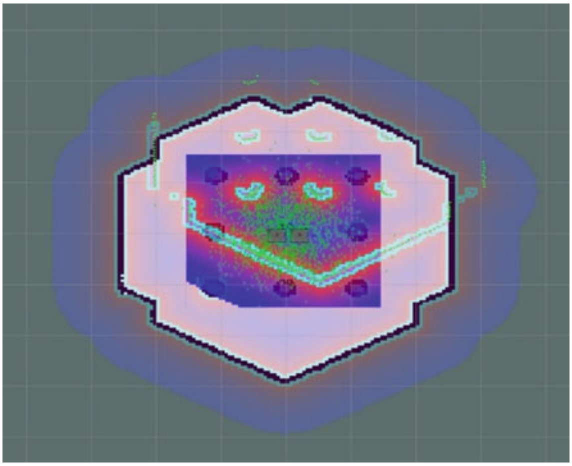

Fig. 4 Robot mapping unknown environment

4.2 Practical Approach

The RRT algorithm which was used in this chapter was helpful in designing the whole system [4]. With the help of the RRT and frontier detection we avoided the obstacle along with avoiding obstacle and reaching goal destination.

In Fig. 6, we can see robot initialized from co-ordinate (0, 0) and the goal destination is at (5, 10) with the help of the algorithm the robot was able to reach the goal by traversing shortest path and also avoiding obstacle along the path [7] (Fig. 6).

Fig. 5 Final map created by robots

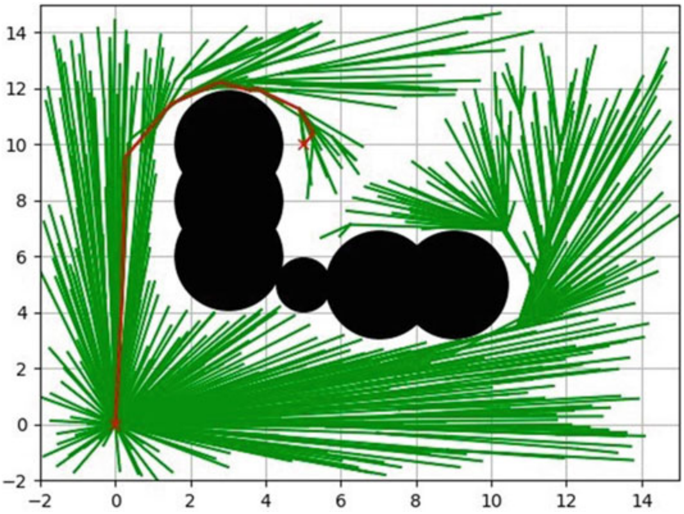

Fig. 6 Working representation of RRT algorithm

5 Result

From the simulation we can observe that we were able to reach the goal in less time, we compared our algorithm with the existing algorithm available and found that our algorithm is 98% accurate and take less time in mapping and navigating through the environment.

6 Application

SLAM is central to a range of indoor, outdoor, in-air and underwater applications for both manned and autonomous vehicles.

Examples:

- At home: vacuum cleaner, lawn mower
- Air: surveillance with unmanned air vehicles
- Underwater: reef monitoring
- Underground: exploration of mines
- Space: terrain mapping for localization.

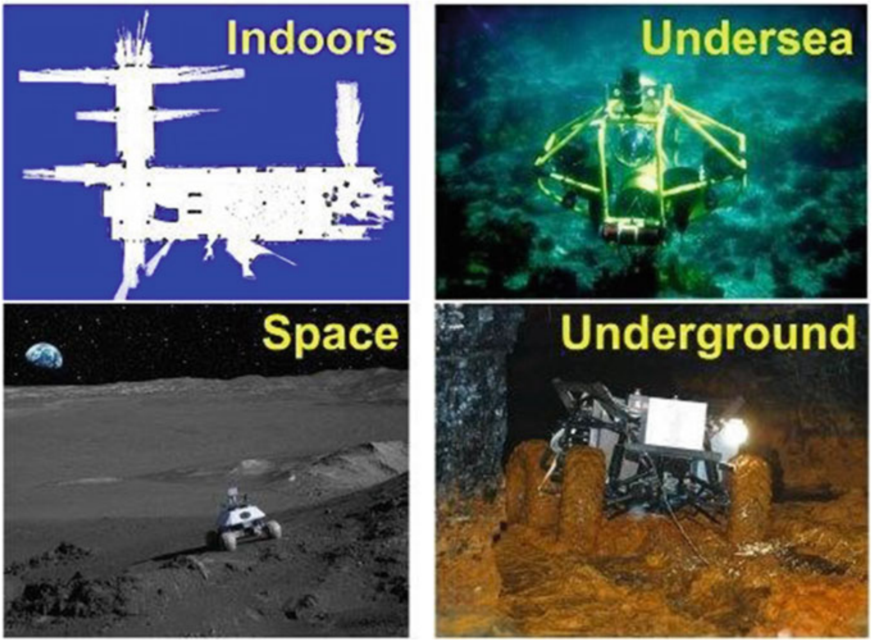

Fig. 7 Application of slam robot

7 Conclusion

In this chapter we have successfully developed and implemented an algorithm for multiple robot exploration and mapping. With the help of the developed algorithm, we were able to minimize the time consumed by robot to navigate and map the unknown environment. The mapped environment is observed in form of g-mapping which helps us in clear understanding of the environment. Further, swarm algorithm was successfully implemented with the help of which multiple robots were able to communicate with each other. This helped us with increased efficiency and reduced exploration time with 98% of efficiency in environment exploration. This multi robot system will be applicable in the smart city environment in near future (Fig. 7).

References

1. Umari H, Mukhopadhyay S (2017) Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In: 2017 IEEE/RSJ International conference on intelligent robots and systems (IROS), Vancouver, BC, 2017, pp 1396–1402. https://doi.org/10.1109/IROS.2017.8202319

2. Reid R, Cann A, Meiklejohn C, Poli L, Boeing A, Braunl T (2013) Cooperative multi-robot navigation, exploration, mapping and object detection with ROS. In: 2013 IEEE Intelligent vehicles symposium (IV), gold coast, QLD, 2013, pp 1083–1088. https://doi.org/10.1109/IVS.2013.6629610
3. Mahdoui N, Fr´emont V, Natalizio E (2018) Cooperative frontier-based exploration strategy for multirobot system. In: 2018 13th Annual conference on system of systems engineering (SoSE), Paris, 2018, pp 203–210. https://doi.org/10.1109/SYSOSE.2018.8428787
4. Sim´eon DT, Cort´es J (2014) A multi-tree extension of the transition-based RRT: application to ordering-and-pathfinding problems in continuous cost spaces. In: 2014 IEEE/RSJ International conference on intelligent robots and systems, Chicago, IL, 2014, pp 2991–2996. https://doi.org/10.1109/IROS.2014.6942975
5. Zhang B, Shang W, Cong S (2018) Optimal RRT* planning and synchronous control of cable-driven parallel robots. In: 2018 3rd International conference on advanced robotics and mechatronics (ICARM), Singapore, 2018, pp 95–100. https://doi.org/10.1109/ICARM.2018.8610680
6. Dos Reis WPN, Bastos GS (2015) Multi-robot task allocation approach using ROS. In: 2015 12th Latin American robotics symposium and 2015 3rd Brazilian symposium on robotics (LARSSBR), Uberlandia, 2015, pp 163–168. https://doi.org/10.1109/LARS-SBR.2015.20
7. Andre T, Neuhold D, Bettstetter C (2014) Coordinated multi-robot exploration: out of the box packages for ROS. In: 2014 IEEE Globecom workshops (GC Wkshps), Austin, TX, 2014, pp 1457–1462. https://doi.org/10.1109/GLOCOMW.2014.7063639
8. Werger BB (2000) Ayllu: Distributed port-arbitrated behavior-based control. In: Parker LE, Bekey G, Barhen J (eds) Distributed autonomous robotic systems 4, pp 25–34. Springer-Verlag, Knoxville, Tennessee, October 2000
9. Wu F, Rdiger C, Yuce MR (2017) Real-time performance of a self-powered environmental IoT sensor network system. Sensors 17(2):282
10. Lie S et al. SLAM and navigation of a mobile robot for indoor environments. Extended Kalman filter (EKF)—Data Association
11. Gossain H, Sharma B (2020) Mobile robot navigation and mapping using optical encoders. Int J Creat Res Thoughts (IJCRT) 8(7):254–258. ISSN: 2320-2882
12. .Thrun S, Leonard JJ (2008) Simultaneous localization and mapping. In: Siciliano B. Khatib O (eds) Springer handbook of robotics. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-30301-5_38
13. Thrun S, Liu Y (2005) Multi-robot SLAM with sparse extended information filers. In: Dario P, Chatila R (eds) Robotics research. The eleventh international symposium. Springer tracts in advanced robotics, vol 15. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11008941_27
14. Eustice RM, Singh H, Leonard JJ (2005) Exactly sparse delayed-state filters. In: Proceedings of IEEE international conference on robotics and automation (ICRA), Barcelona, Spain, pp 2428–2435
15. .Wang Z, Huang S, Dissanayake G (2005) Implementation issues and experimental evaluation of D-SLAM. In: Proceedings of the 5th international conference on field and service robotics (FSR), Port Douglas, Australia, pp 153–164
16. Mandal S, Biswas S, Balas VE, Shaw RN, Ghosh A (2021) Lyft 3D object detection for autonomous vehicles. Artif Intell Fut Gener Robot 119–136. https://doi.org/10.1016/B978-0-323-85498-6.00003-4
17. Banerjee A et al (2022) Construction of effective wireless sensor network for smart communication using modified ant colony optimization technique. In: Bianchini M, Piuri V, Das S, Shaw RN (eds) Advanced computing and intelligent technologies. Lecture notes in networks and systems, vol 218. Springer, Singapore. https://doi.org/10.1007/978-981-16-2164-2_22
18. Biswas S, Bianchini M, Shaw RN, Ghosh A (2021) Prediction of traffic movement for autonomous vehicles. In: Bianchini M, Simic M, Ghosh A, Shaw RN (eds) Machine learning for robotics applications. Studies in computational intelligence, vol 960. Springer, Singapore. https://doi.org/10.1007/978-981-16-0598-7_12

19. Soni A. Dharmacharya D, Pal A, Srivastava VK, Shaw RN, Ghosh A (2021) Design of a machine learning-based self-driving car. In: Bianchini M, Simic M, Ghosh A, Shaw RN (eds) Machine learning for robotics applications. Studies in computational intelligence, vol 960. Springer, Singapore. https://doi.org/10.1007/978-981-16-0598-7_11
20. Mandal S, Sk Md Basharat Mones, Das A, Balas VE, Shaw RN, Ghosh A (2021) Single shot detection for detecting real-time flying objects for unmanned aerial vehicle. Artif Intell Fut Gener Robot 37–53. https://doi.org/10.1016/B978-0-323-85498-6.00005-8