Implementing Robotic Path Planning After Object Detection in Deterministic Environments Using Deep Learning Techniques

R. Gayathri, V. Uma, and Bettina O'Brien

1 Introduction

Robotic path planning [13] or motion planning is the most defining feature of a robot. It can be understood as the ability to navigate in an environment in which it is either previously used to or new to. Motion planning is used to define a number of successive configurations in an environment which leads the robot from a start position to a goal state. The problem of navigation [21] can be integrated into four categories such as perception, localization, motion control and path planning. Perception is one of the most significant capabilities of robots which visualizes the environment and are capable of capturing every activity along the way. Localization is the process of identifying the location of the robot in the environment. It can be done in terms of other obstacles present in the environment as well. One such localization algorithm is simultaneous localization and mapping (SLAM) [11]. Motion control is the problem of identifying the next action for a robot based on the present state of the environment. Path planning defines trajectories and provides a collision-free path for the robot to navigate.

Path planning algorithms can be categorized as deterministic or non-deterministic state algorithms, which can also be referred to as global and local path planning respectively. The robots in the deterministic environment takes advantage of knowing the environment before execution of a task, whereas in a non-deterministic type, the

R. Gayathri (✉)
Department of Computer Science, Rajiv Gandhi National Institute of Youth Development (RGNIYD), Ministry of Youth Affairs and Sports, Government of India, Sriperumbudur, Chennai, India
e-mail: gayathrir339@gmail.com

V. Uma · B. O'Brien
Department of Computer Science, Pondicherry University, Puducherry, India

R. Doriya et al. (eds.), *Machine Learning, Image Processing, Network Security and Data Sciences*, Lecture Notes in Electrical Engineering 946,
https://doi.org/10.1007/978-981-19-5868-7_12

robot explores and perceives a changing environment. Thus, the robot avoids collision with known obstacles and estimates feasible paths using various path planning algorithms [7, 13]. The traditional path planning algorithm builds a map or blueprint of the environment ahead of time, thereby generating the shortest path towards the goal. Later, neural networks [20] were used to train the robot to understand its environment and plan a path accordingly. Given a set of environmental images, a model could be built so as to train on images to perceive the environment [3, 14] and generate paths for an input image [20].

Object detection [15, 17, 22], an important aspect in neural networks, has been made possible using various models like convolutional neural network (CNN) [24], recurrent neural network (RNN), long short-term memory (LSTM) and so on. The metrics and hyperparameters can be tuned in such a way as to increase the efficiency of the model, so that a collision-free optimized path can be generated. In this paper, three sampling-based path planning algorithms, probabilistic RoadMap (PRM), rapidly exploring random trees (RRT) and bidirectional-RRT (Bi-RRT) are implemented after detecting obstacles in an environment.

The overview of the paper is defined as such. Section 2 discusses about the related work regarding the gradual improvement in the methods of implementing path planning techniques using neural network models. Section 3 deals with the methodologies used to process images, identify and detect objects as well as generation of paths. Our proposed work involves a CNN model [9] along with three sampling-based path planning algorithms [4, 18] to navigate the robot in a deterministic environment. In Sect. 4, experimental setup, the results obtained using sampling-based algorithms are defined. Section 5 depicts the implementation comparison of methods and its graphical analyses. Finally, in Sect. 6, the paper concludes the aim of the proposed technique and presents the future proceedings.

2 Related Study

In general, path planning algorithms can be branched into sampling-based, node-based, mathematical model, bio-inspired-based and multi-fusion-based methods [5]. Figure 1 depicts the branches of path planning algorithms. Node-based planning is a search mechanism which explores a set of nodes starting from an initial node and finds an optimal path based on decomposition process. Some of the traditional node-based algorithms are Dijkstra, $A*$, $D*$, etc. Mathematical model-based path planning is completely based on the kinematics and dynamics of the physical space where the robot navigates.

This paper aims at comparing PRM, RRT and Bi-RRT sampling-based algorithms [2] after differentiating the obstacle and free space using a CNN model. In sampling-based method, there are active and passive techniques [13]. In active method, given the start and goal points, the agent randomly finds ways by generating sampling points along the path towards the goal, whereas in passive method, the whole search space is given along with the goal and the agent creates a roadmap towards the goal.

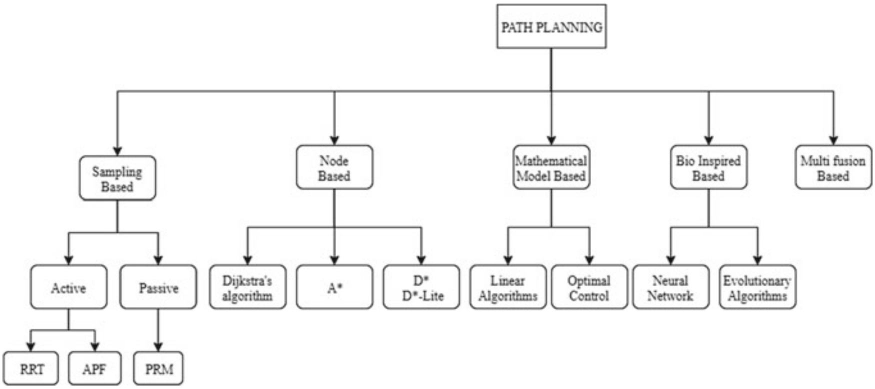


Fig. 1 Methods in path planning

Rapidly exploring random tree (RRT) and artificial potential fields (APF) are two notable active path planning algorithms. RRT algorithm generates random tree-like structural feasible paths from the start to goal points, among which the shortest path is selected [6]. Probabilistic RoadMap (PRM) [4] is a passive path planning method, where the outcome is based on high probability and sample points are generated to construct a graph. PRM has a local planner connects two points x and y in the Configuration-space (C-space) if there is no obstacles in that edge and a roadmap method to construct a map of the environment. A probabilistic planner is used and works in offline and online modes. In offline, the planner learns about the environment and constructs a roadmap, whereas in online mode, a graph is generated from the roadmap and the planner queries an optimal path for the robot. Bidirectional RRT (Bi-RRT) [18] algorithm works similar to RRT but generates random paths from both the start and goal positions.

Object detection [1], a core challenge in computer vision, detects objects from images which involves computer vision and deep learning strategies, particularly in facial detection [10] and image recognition [8] applications. Bio-inspired-based path planning is a method which is inspired by the natural behaviour of individuals. For instance, inspired by the human neural system, artificial neural network (ANN) [21] was invented. Robotic path planning which requires a deep understanding and learning about the environment is implemented using various architectures of CNN [23, 24]. For the past few years, CNN has proved to be a reliable method for object detection and classification [12] due to its speed and accuracy. Detection of objects has been possible with both deep learning and OpenCV [2]. You Only Look Once (YOLO) [14, 19] detects objects with higher accuracy but does not identify objects in a group. A deep learning single neural network, Single Shot MultiBox Detector (SSD) [16], is a straightforward and easy model for training smaller-sized data.

3 Proposed Methodology

The proposed model defines a method in which the obstacles in the given image are detected after which path planning is done by excluding the regions where detection is performed. In some instances, certain obstacles are not considered to be obstacles by the basic motion planning algorithms. Hence, paths were generated over such undetected obstacles. Thus, the proposed model is segmented into two modules, object detection and path generation phases. Figure 2 depicts the architecture of the proposed path planning strategy.

3.1 Object Detection

This phase identifies and localizes obstacles in digital images. The obstacle detection model used is a MobileNet deep CNN architecture with a Single Shot MultiBox Detector (SSD) framework. The SSD framework is similar to a VGG16 CNN model, used to localize objects, whereas the MobileNet architecture is made up of seventeen blocks used to classify images with labels. Together, object detection is done by replacing VGG16 with MobileNet. SSD is proved to precisely bound boxes for objects with high accuracy.

The model is trained with a sample environmental images, which includes both grayscale and coloured obstacles. Figure 3 shows some sample environmental images in which the obstacle shapes are emphasized, used for training and testing. This environment map provides the random position of fixed set of obstacles in varying shapes. The hurdles to overcome in the environment are that performing safely and robustly when negotiating tight spaces in map, like obstacles or edges. Further, a method is proposed to understand the real-time information about the obstacles of varying shapes and to detect the obstacle-free space. This is used in generating the collision-free waypoints and finding an optimal path. In the traditional path planning strategy, the obstacles that are sketched in Fig. 3a are not been identified or detected.

The proposed work is based on transfer learning in which the model is not trained from the scratch, rather it uses the pre-trained weights of the SSD detection using coco dataset. The performance of SSD framework is directly proportional to object

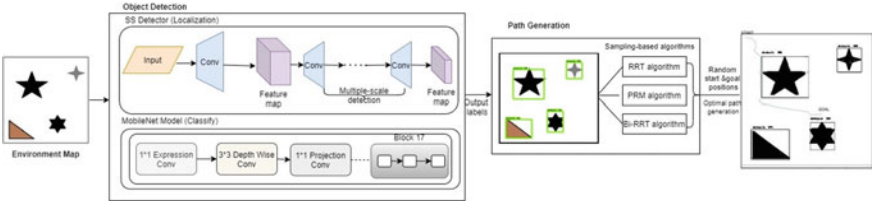


Fig. 2 Proposed architecture

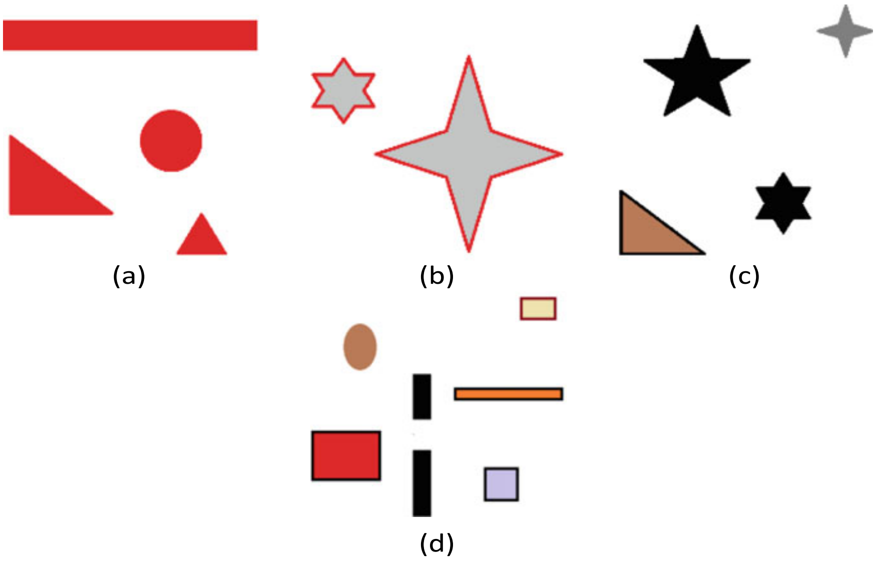


Fig. 3 **a**, **b** Sample train images and **c**, **d** Sample test images

sizes and does not fare too well on object categories with small sizes. Therefore, MobileNet architecture is used to resize certain parts of the image which helps the network to identify and learn features for small object categories.

The basic block of the MobileNet architecture has a 1 × 1 expansion layer, a 3 × 3 depthwise layer and a 1 × 1 projection layer. The expansion convolution augments the channel numbers of the input image. The depthwise convolution filters reduce the channels after which the filtered values are combined to give new learned features. The projection layer projects images with increased channel numbers and dimensions. The outcome of this phase is an image with bounding boxes around all the obstacles.

3.2 Path Generation

In this phase, the collision-free trajectory is generated by the sampling-based path planners such as RRT, PRM and Bi-RRT. The principle use of this method is to generate a random sample in the environment in the form of nodes, cells or in other forms in order to achieve a feasible path. This algorithms requires prior knowledge of the environment in finding a path.

During navigation, the path planner takes into account the geometric constraints of obstacles in order to reach the desired target point. The area explored by the algorithms is the area occupied by the free space. The obstacle detection-based sampling-based algorithms generate an optimal path towards the goal and improve the exploration efficiency, thereby reducing the computational over-head.

Initially, for each algorithm, for different start and goal positions, all possible feasible paths towards the goal are generated, out of which the optimal path is identified. Out of these, Bi-RRT and PRM have proved to produce the best paths in terms of time and distance, respectively.

4 Implementation and Analyses

The proposed model is implemented using a custom-made environmental image dataset with obstacles. All the images have a resolution of 500 × 500 pixels. We use the labeling annotator to label the images with ground truth boxes. The corresponding XMLs are converted into CSV format after splitting the training and test images. Then, the TFRecords are generated for both the train and test images.

The aim of this method is to perform obstacle detection on these images and generate all the possible obstacle-free paths. Amongst this, the optimal shortest path is found. The model has a single class called ‘obstacle’ with L2 regularizer, ReLU 6 activation, learning rate as 0.004 and RMSprop optimizer. Figure 4 depicts obstacle detection performed of three sample test images.

The obstacle detection model is implemented with a pre-trained SSD MobileNet model that uses a single GPU with hyperparameters 1500 epochs, 50 evaluation steps and batch size 12 set as such. This model is trained with the environment map with bounding boxes. As the number of training epochs increases, the test loss decreases. This is shown in Fig. 5.

The performance of obstacle detection model is measured in terms of mean average precision (mAP), recall, $F1$-score and frames per second (fps). The overall object labels from the area under precision and recall curve is calculated by mean average precision (mAP) which is the average of APs. This is shown in Eq. (4). Frames per second are the number of frames that can be processed per second.

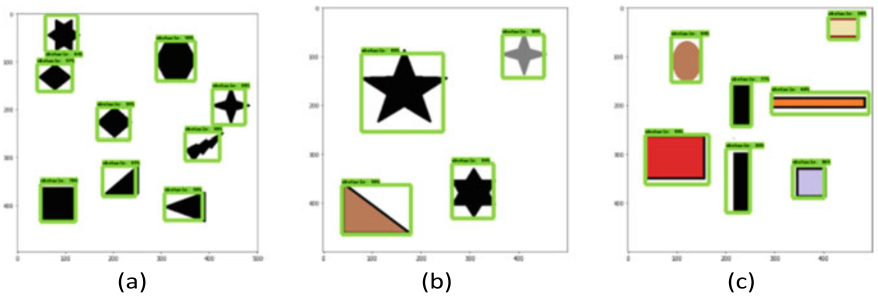

(a) (b) (c)

Fig. 4 Obstacle detection under different scenarios

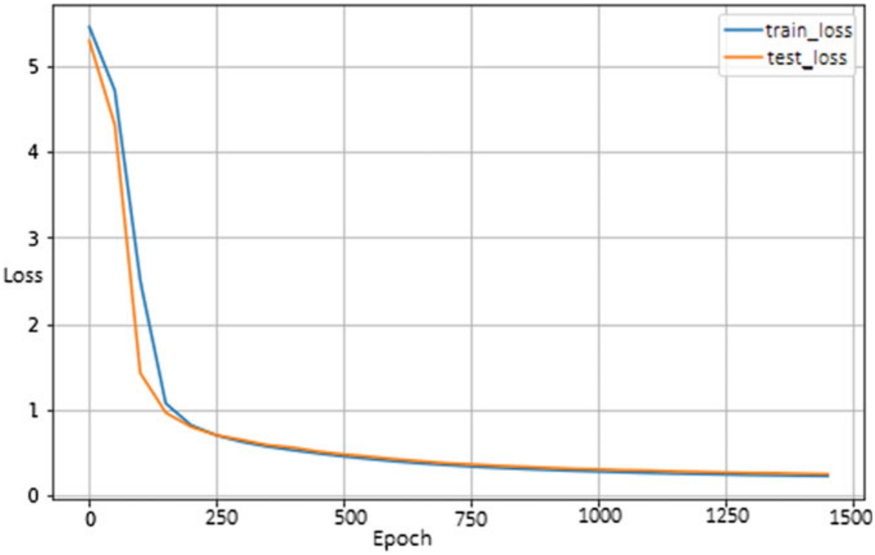


Fig. 5 Training curves of the loss function

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$F1 - \text{score} = \frac{2PR}{P + R} \tag{3}$$

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4}$$

Table 1 shows that the number of parameters, mean average precision (mAP), recall and $F1$-score as well as their per-frame inference speed. SSD MobileNet architecture obtains high $F1$-score, processing speed and mAP due to depthwise separable convolutions which drastically reduces the number of parameters in the network.

Table 1 Number of parameters, mAP, recall, $F1$-score and processing speed of object detection approach

Model	Backbone	Parameters	mAP	Recall	$F1$-score	Speed (fps)
SSD	MobileNet	7.5M	84.6	80.11	90.38	95

Path generation for sampling-based algorithms RRT, PRM and Bi-RRT are done using octave at different start and goal positions in the environment. Negative coordinates would not be accepted and for such positions, an exception 'coordinates lies on obstacles' is shown. For each random sampling method, multiple paths are found by exploring the environment, out of which the optimal path is generated. In addition to the shortest path, the execution time in milliseconds and the distance in terms of the number of nodes along the path are calculated. The distance of the path between the two coordinate points (x_1, y_1) and (x_2, y_2) is calculated using the Euclidean distance. This formula is given in Eq. (5).

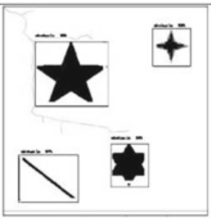
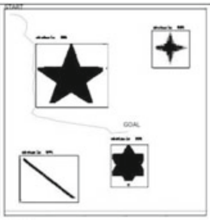
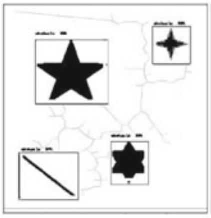
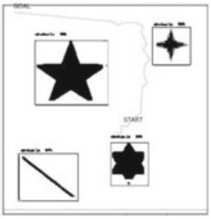
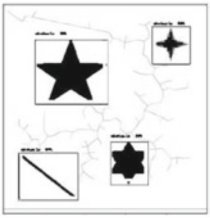
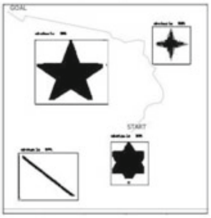
$$\text{Distance}(d) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{5}$$

The execution time is estimated the time taken to compute the solution from an initial position to goal position. For each method, paths are generated for many sample environments, out of which one is depicted.

5 Comparative Analysis

Tables 2, 3 and 4 show a comparison of RRT, PRM and Bi-RRT path generation algorithms for a sample environment, after obstacle detection. Clearly, from these table, we capture that all methods provide an optimal path in a certain measure, but the bidirectional-RRT algorithm provides an optimal path in a much lesser time.

Table 2 Path generated by RRT in sample environment

Start Goal	Exploration	Optimal path	Runtime (sec) Distance(m)
S[20 20] G[390 390]			69.46, 684.96
S[400 350] G[25 35]			84.09, 831.14
S[400 400] G[25 15]			99.22, 726.00

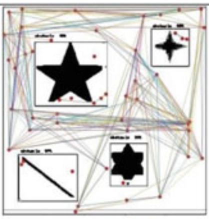
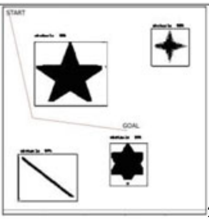
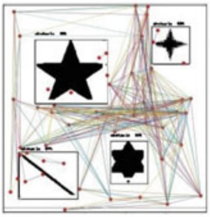
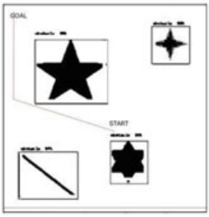
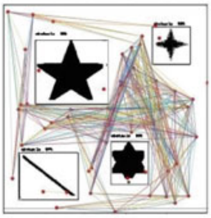
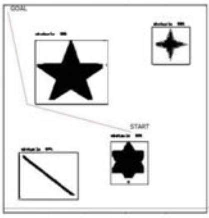
We computed the path length and run time of each environment. The results are shown in Table 5. From the obtained results, we found that Bi-RRT yields minimum time when compared with the other two approaches. PRM results in minimum distance due to more obstacle-free space in the environment but it has taken maximum time to search the optimal path due to the number of vertices in the roadmap parameter.

Figure 6a and b depicts the time and distance metrics for three different source *S* and goal *G* given by the three path generation algorithms.

6 Conclusion and Future Directions

The aim of this paper is to provide a path planning approach using deep learning, in which the obstacles are identified in the environment and detected using SSD framework. Then, path planning is performed using three sampling-based algorithms

Table 3 Path generated by PRM in sample environment

Start Goal	Exploration	Optimal path	Runtime (sec) Distance(m)
S[20 20] G[390 390]			214.12, 637.72
S[400 350] G[25 35]			190.24, 606.70
S[400 400] G[25 15]			284.88, 630.62

and comparative analyses is performed for each method's time and distance metrics. The results are tabulated, and Bi-RRT proves to be the best out of the three methods. The future work is this paper is to include semantic features in obstacles and to identify obstacles in terms of the semantic labels.

Table 4 Path generated by Bi-RRT in sample environment

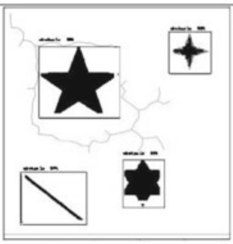
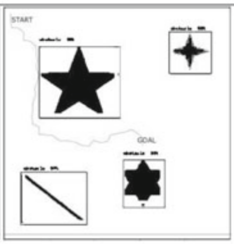
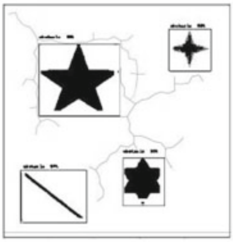
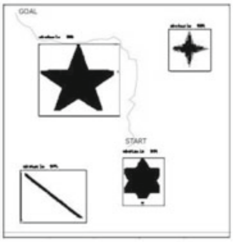
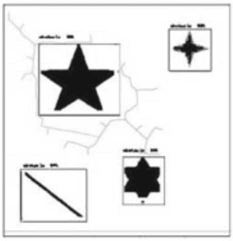
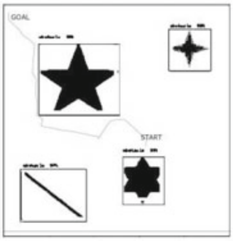
Start Goal	Exploration	Optimal path	Runtime (sec) Distance(m)
S[20 20] G[390 390]			67.48, 755.56
S[400 350] G[25 35]			48.49, 705.28
S[400 400] G[25 15]			34.36, 734.85

Table 5 Comparison of each method with respect to execution time and path length

Performance metrics	RRT	PRM	Bi-RRT
Avg. execution time (s)	84.25	229.74	50.11
Avg. distance (m)	747.36	625.01	731.89

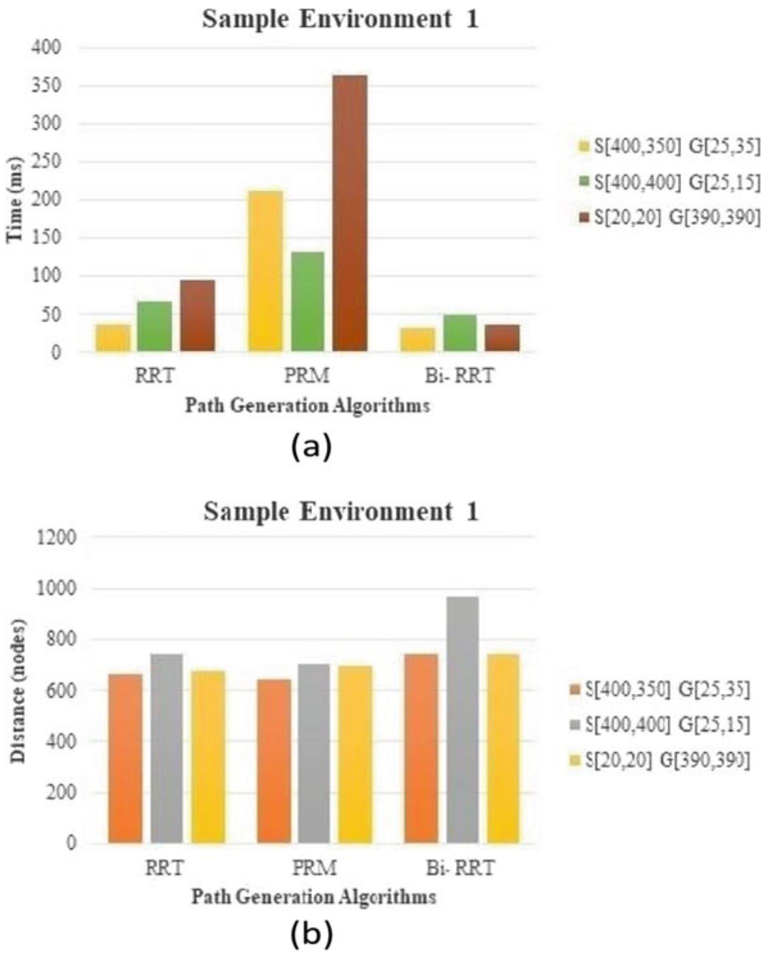


Fig. 6 Comparison of performances of various algorithms in difference start and goal positions with respect to **a** execution time **b** distance covered

References

1. Ahmed SM, Tan YZ, Lee GH, Chew CM, Pang CK (2016) Object detection and motion planning for automated welding of tubular joints. In: 2016 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 2610–2615
2. Chandan G, Jain A, Jain H et al (2018) Real time object detection and tracking using deep learning and openCV. In: 2018 international conference on inventive research in computing applications (ICIRCA). IEEE, pp 1305–1308
3. Duguleana M, Mogan G (2016) Neural networks based reinforcement learning for mobile robots obstacle avoidance. Expert Syst Appl 62:104–115
4. Faust A, Oslund K, Ramirez O, Francis A, Tapia L, Fiser M, Davidson J (2018) PRM-RL: long-range robotic navigation tasks by combining reinforcement learning and sampling-based

planning. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE, pp 5113–5120

5. Gayathri R, Uma V (2019) Performance analysis of robotic path planning algorithms in a deterministic environment. Int J Imaging Robot 19(4):83–108
6. Gayathri R, Uma V, Bettina O (2021) Unified robot task and motion planning with extended planner using ROS simulator. J King Saud Univ Comput Inf Sci
7. Geraerts RJ (2006) Sampling-based motion planning: analysis and path quality
8. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
9. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
10. Jiang H, Learned-Miller E (2017) Face detection with the faster R-CNN. In: 2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017). IEEE, pp 650–657
11. Khairuddin AR, Talib MS, Haron H (2015) Review on simultaneous localization and mapping (slam). In: 2015 IEEE international conference on control system, computing and engineering (ICCSCE). IEEE, pp 85–90
12. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60(6):84–90
13. LaValle SM (2006) Planning algorithms. Cambridge University Press
14. Li G, Ma Y (2018) A deep path planning algorithm based on CNNs for perception images. In: 2018 Chinese automation congress (CAC). IEEE, pp 2536–2541
15. Li X, Wang S (2017) Object detection using convolutional neural networks in a coarse-to-fine manner. IEEE Geosci Remote Sens Lett 14(11):2037–2041
16. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) SSD: single shot multibox detector. In: European conference on computer vision. Springer, pp 21–37
17. Orozco-Rosas U, Picos K, Montiel O, Sepu´lveda R, D´ıaz-Ram´ırez VH (2016) Obstacle recognition for path planning in autonomous mobile robots. In: Optics and photonics for information processing X, vol 9970. International Society for Optics and Photonics, p 99700X
18. Qureshi AH, Ayaz Y (2015) Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments. Robot Auton Syst 68:1–11
19. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
20. Szegedy C, Toshev A, Erhan D (2013) Deep neural networks for object detection. In: Advances in neural information processing systems, pp 2553–2561
21. Tai L, Li S, Liu M (2017) Autonomous exploration of mobile robots through deep neural networks. Int J Adv Rob Syst 14(4):1729881417703571
22. Tripathi S, Dane G, Kang B, Bhaskaran V, Nguyen T (2017) LCDet: Low-complexity fully-convolutional neural networks for object detection in embedded systems. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 94–103
23. Zhao ZQ, Zheng P, Xu ST, Wu X (2019) Object detection with deep learning: a review. IEEE Trans Neural Networks Learn Syst 30(11):3212–3232
24. Zhiqiang W, Jun L (2017) A review of object detection based on convolutional neural network. In: 2017 36th Chinese control conference (CCC). IEEE, pp 11104–11109