International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 11 | Nov - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Object Detection and Path Planning Algorithms for Autonomous Vehicles

[1]Avanti Savji, [2]Ved Thorat, [3]Vedashri Patil and [4]Dr. Jayashree Tamkhade
Electronics and Telecommunication Vishwakarma Institute of Information Technology
Pune, India.
[1]avanti.22210023@viit.ac.in, [2]ved.22211044@viit.ac.in,
[3]vedashri.22210642@viit.ac.in, [4]jayashree.tamkhade@viit.ac.in

***Abstract*—With the increasing development in Artificial Tech- nology (Ai), the area of application increases as well. One such application of AI is Computer Vision, where the computer learns through images and videos. In this paper, we will explore various path planning and object detection algorithms used in autonomous vehicles. Autonomous vehicles drive without a driver, they are equipped with RADAR, LIDAR sensors and many cameras which acquire the data from the environment. The path planning algorithm uses the data to predict the optimal path for the vehicle to move.**

***Index Terms*—Autonomous Vehicles, Object Detection, Path Planning, RADAR/LIDAR**

I. Introduction

1) Autonomous Vehicles (AVs) are vehicles which drive themselves without the assistance of a human driver. [1] The Autonomous Vehicles are able to navigate through the roads, read signs and detect obstacles while avoiding them. [3]
2) Challenges faced by AVs:
   - Real time processing is crucial in AVs. The ability to make decisions on the road while driving is an important factor. [1]
   - Reaction to dynamic environment: The environment is not static, there are all these things moving in the outside world. [2] The AVs have to be able to make decisions based on the movements of vehicles, pedestrians around it. [3]
3) This paper focuses on the various Object Detection and path Planning algorithms that are used in autonomous vehicles to guide them and navigate them in the real world. Object detection algorithms are used to detect objects in the real world, an example of an object would be a car, a sidewalk, road signs, pedestrians, etc. By detecting objects the path Planning algorithms are able to make a suitable path for the vehicle to navigate on the road. [4]

II. Background and Literature Review

"A Review on Autonomous Vehicles: Progress, Methods and Challenges" by Parekh, Darsh, Nishi Poddar, Aakash Rajpurkar, Manisha Chahal, Neeraj Kumar, Gyanendra Prasad Joshi, and Woong Cho analysis in this work, they looked into the current state of research and development in environment detection, pedestrian detection, path planning, motion control, and vehicle cybersecurity for autonomous vehicles and aimed to study the different proposed technologies and compare their approaches. [1]

"Autonomous vehicles: theoretical and practical challenges" by Margarita Mart´ınez-D´ıaz, Francesc Soriguera this paper provides an overview of current state of the art in the key aspects of autonomous driving . general V2V corporation and ptooning are options being discussed, both with multiple variants. [2]

"Autonomous vehicles: A study of implementation and security" by Khan, Firoz and Ramasamy, Lakshmana and Kadry, Seifedine and Meqdad, Maytham N. and Nam, Yunyoung this paper will look at the pros and cons of implementation of autonomous vehicles. The vehicles depend highly on the sensors present on the vehicles and any tampering or manipulation of the data generated and transmitted by these can have disastrous consequences, as human lives are at stake here. Various attacks against the different type of sensors on-board an autonomous vehicle are covered. [3]

"Obstacle Avoidance and Path Planning Methods for Autonomous Navigation of Mobile Robot" by Katona, Korne´l, Husam A. Neamah, and Pe´ter Korondi This article provides an overview of key obstacle avoidance algorithms, including classic techniques such as the Bug algorithm and Dijkstra's algorithm, and newer developments like genetic algorithms and approaches based on neural networks. It analyzes in detail the advantages, limitations, and application areas of these algorithms and highlights current research directions in obstacle avoidance robotics. [4]

"A Systematic Literature Review of A* Pathfinding" by Daniel Foead, Alifio Ghifari, Marchel Budi Kusuma, Novita Hanafiah, Eric Gunawan this paper examines A-Star's current usage in the field of pathfinding, comparing A* to other search algorithms. It also analyzes potential future developments for A-Star's development this paper transforms the path planning problem into a multi-constraint optimization problem by considering three costs: path length, turning angle, and collision avoidance. A multi-strategy improved POA algorithm (IPOA)

is proposed to address this. [5]

"Path Planning of an Unmanned Aerial Vehicle Based on a Multi-Strategy Improved Pelican Optimization Algorithm" by Qiu, Shaoming, Jikun Dai, and Dongsheng Zhao autonomously. In this paper, a systematic review of mobile robot path planning techniques is presented. Firstly, path planning is classified into global path planning and local path planning according to the mastery of environmental information. In the global path planning, environment modeling methods and path evaluation method are introduced. [6]

"Path planning techniques for mobile robots: Review and prospect" by Lixing Liu, Xu Wang, Xin Yang, Hongjie Liu, Jianping Li, Pengfei Wang The presented approach has the advantage of a simple straight forward implementation, which enables a coordinated movement with comparatively small programming effort. [7]

"Simple Path Planning Algorithm for Two-Wheeled Differentially Driven (2WDD) Soccer Robots." by Novak, Gregor and Seyr, Martin. This paper describes a novel simple approach based on kinematics. The testing bed is a tiny two-wheeled robot. The robot's movement is specified by its translatoric velocity vR and its angular velocity . The kinematic approach generates piecewise circular arcs based on a number of possible different sets of boundary conditions in the target positions. [8]

"Enhancing Real-time Object Detection with YOLO Algorithm" by Lavanya, Gudala and Pande, Sagar This paper introduces YOLO, the best approach to object detection. Real-time detection plays a significant role in various domains like video surveillance, computer vision, autonomous driving and the operation of robots. YOLO algorithm has emerged as a well-liked and structured solution for real-time object detection due to its ability to detect items in one operation through the neural network. [11]

"Object Detection for Autonomous Driving using YOLO [You Only Look Once] algorithm" by A. Sarda, S. Dixit and A. Bhan This paper introduces YOLO, the best approach to object detection. Real-time detection plays a significant role in various domains like video surveillance, computer vision, autonomous driving and the operation of robots. YOLO algorithm has emerged as a well-liked and structured solution for real-time object detection due to its ability to detect items in one operation through the neural network. [14]

III. Object Detection Algorithms

Object detection plays a vital role in autonomous vehicles. Object detection algorithms help in detection of various objects in the environment with the help of cameras and sensors such as RADAR/LIDAR. [4] Object detection algorithms are able to detect obstacles such as cars, sidewalks, pedestrians, and they are able to read signs on board on the road, read driving instructions, marking on the roads.

On a comparative of all the sensors that exist, camera sensors tend to be the best performers at an efficient comprehension of the surrounding. Comprehension of images containing multiple objects make it an extremely difficult task to incur especially when objects belonging to multiple classes are captured in the same image. [6] 90% of the accidents on the road happens due to human error and autonomous vehicles can help us cut through the human error and reduce risk of life significantly. [7]

Various types of object detection algorithms which are used in autonomous vehicles are:

1) **YOLO (You only look once)**: The YOLO algorithm detects objects with the help of bounding boxes around the objects that it detects so we want to convert the labels of images in the training and testing data set into the YOLO format where it has details of bounding boxes which are known and annotations of the images. After the annotations have been generated and converted into the YOLO format, we incorporate all the data which includes the annotations and the images for training and testing into two separate folders respectively [14]. **YOLO Architecture**: The backbone architecture that is used in YOLO v4 is CSPDarknet53[14]. It also consists of a NECK architecture of SPP along with PAN. The head architecture that is used is the YOLO v3. [13] YOLO v4 makes use of data augmentation architectures which are also termed as 'bag of freebies' because a change in the bof does create an extra cost on the training of the algorithm. The augmentation used for backbones are the drop block regularization, Mosaic data augmentation. When the accuracy can be increased significantly by a very small increase in cost is termed as the 'bag of specials'. [12] The BOS used in backbone are Mish activation, CSP, Mi WPRC. [14] The BOF used for detector is- SAT, CmBN The BOS used for detector is-Mish activation, SAM, PAN. [11]
2) **Histogram of Object Oriented Gradients (HOG)**:Algorithm searches all possible areas and directly decides whether vehicles are included in the areas, which avoids much trouble encountered in the HG step. HOG descriptor is used to obtain the representation of vehicles. Some calculation skills could be easily adopted in this procedure, for example, Integral Image, and therefore this appearance-based method is in some sense more efficient than others, especially Gabor filters. [15]
   The system consists of three main components: HOG feature extraction, linear SVM classifier training and vehicle detection. The input image is firstly represented as a HOG feature which is later used as the input of the training step or linear SVM classifier. The learning phase creates a binary classifier that provides cars/non-cars decisions for fixed sized image regions (windows); while the detection phase uses the classifier to perform a dense multi-scale scan reporting preliminary object decisions at each location of the test image. These preliminary decisions are then fused to obtain the final object detections. [16]
3) **Region Based Convolutional Neural Networks (R-**

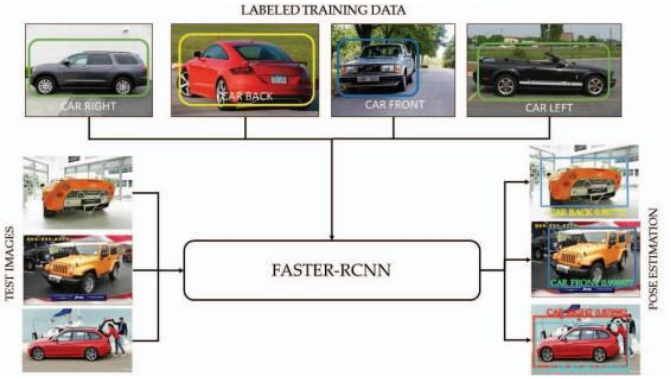

Fig. 1. Faster-RCNN [16]

CNN): Estimating the pose of a vehicle in an application to an intelligent transportation system is beneficial not only for autonomous navigation systems but also in tracking the direction of the vehicle. [19]

FASTER-RCNN is composed of two modules; the first module is the composition of a fully convolutional network which produces the region and the second module uses these regions with a FASTER-RCNN detector. The whole system is a single, consolidate network for detecting objects. [20]

Overall Structure of our Pose Estimation based on Region Based Convolutional Neural Network. [21]

4) **Region Based Fully Convolutional Neural Networks (R-FCN)**:Road detection is a crucial task in autonomous navigation systems. It is responsible for delimiting the road area and hence the free and valid space for maneuvers.The main component of the system is a Convolutional Neural Network (CNN). [23] Convolutional layers tend to be followed by polling layers. Such layers reduce the dimensionality of the input by performing non-parametric operations (e.g. average, max) in local regions of the input. This layer tends to aggregate similar regions, enforce translation invariance and avoid overfitting. A typical CNN architecture consists of a repeated sequence of convolutional and polling layers, referred by some authors as the features extraction phase, followed by a number of fully connected layers (as in an MLP) where the final layer implements the loss function (e.g. Softmax). After each convolutional and fully connected layer a pointwise nonlinearity, such as the rectified linear unit (ReLU) function

$$\sigma(x) = \max(0, x)$$

is usually applied. The training is performed using stochastic gradient descent and the gradients are computed according to the chain rule, exactly as in a standard MLP. [24]

CNN architectures, as the one proposed, can be converted into a Fully Convolutional Network (FCNs) by converting the fully connected layers into convolutional

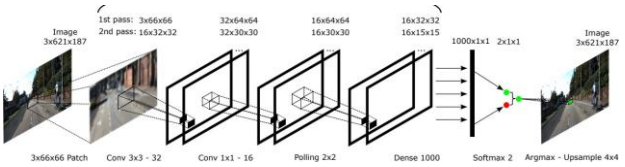

Fig. 2. R-FCN [6]

layers. The benefit of doing so is that, instead of inputting only a patch, the whole image could be used as input and the network would output the classes for every image region. The use of FCNs for inference is especially important in cases where the contextual window or patch is large and there is a lot of overlap between subsequent patches, which is the case of the proposed architecture. [25]

Network Architecture: The input to the network is a three-channel image patch and its output is a class (road or non-road) that is attributed to the 4×4 region in the center of the patch. To classify a whole image, a patch should be extracted and classified for every 4×4 region of the original image (i.e. patches are extracted 3175 with a 4×4 stride). The network itself starts with a conv. 3×3 - 32 (32 filters sized 3×3 each) layer, followed by a conv. 1×1 - 16 and a max-polling layer of 2×2. These three layers are repeated in sequence with the same parameters. Finally, there is a fully-connected layer with 1000 neurons and a final layer with 2 neurons (one for each class). All convolutional layers have a stride of 1×1, are followed by the *ReLU* activation function and do not employ padding. The first fully connected layer is followed by the *ReLU* function while the final layer implements the *Softmax* loss function. [26]

Detailed view of the network architecture. The input to the model is a patch extracted from the image, its output is the class (road or non-road) which is attributed to a 4×4 region in the center of the input patch. [26]

5) **Atrous Convolution & Spatial Pyramid Pooling**: Lane detection is a fundamental capability for autonomous driving. Many effective lane detection algorithms based on traditional computer vision and recent deep learning technologies have been proposed. However, the current state-of-the-art lane detection accuracy is still not satisfactory for realizing fully autonomous driving. [27] The network is built on LaneNet. It mainly consists of one encoder and two decoders. The two decoders are respectively named as the Embedding Decoder and Binary Decoder. They output the embedding feature map and binary segmentation map, respectively. The embedding feature map and binary segmentation map are element-wisely multiplied, and then sent to a post-processing module to get the instance segmentation for the lane detection. Different lanes are labeled with different numbers, which are depicted with different colors. [28] The structure of instance segmentation network for lane

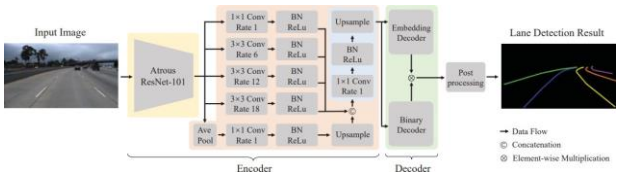

Fig. 3. Atrous Convolution and Pyramid Pooling

detection. The yellow box represents the Atrous CNN module.The pink and blue boxes represent the Atrous SPP module. The decoder module is shown in the green box. Conv, BN, Ave Pool represent convolutional, batch normalization and average pooling layers, respectively. Rate means the dilation rates for the Conv layers. In the detection result, different lanes are denoted with different colors. The figure is best viewed in color. [29]

IV. Path Planning Algorithms

Path Planning algorithms are used to calculate the best possible path for a robot, vehicle or a device to travel in so that it can reach its destination from a certain starting point. Path Planning algorithms often use the help of Object detection algorithms to find the best possible path. They used Object detection algorithms to detect objects in the path and navigate their way around the object in such a way that the device does not collide with the object.

The Path Planning algorithms are categorized into Grid Based Searching algorithms, Sampling Based Algorithms, Optimization Based Algorithms, Potential Field Methods, Hybrid Approaches

The most popular path planning algorithms used into modern technologies are:

1) **A* (A-Star)**:A* algorithm, also known as the A-star algorithm, is the pathfinding and graph traversal algorithm widely used because of its efficiency and flexibility. [31] It is a search algorithm in finding the cheapest path from a start node to a goal node. A star algorithm uses a heuristic to guide search between the breadth first search and the greedy best-first search toward the optimal solutions.
It is utilized in numerous applications like robotics and computer games as well as artificial intelligence to be applied to problems like finding the shortest path between two points.
A* algorithm is defined as best-first algorithm, because each cell in the configuration space is evaluated by the value

$$f(v) = h(v) + g(v)$$

Where h(v) is the heuristic distance (Manhattan, Euclidean, or Chebyshev) of the cell to the finishing point, and *g(v)* is the total length of stretch from the initial position to the intended position going through the given set of cells. It is clear that this set terminates in the currently processed node. A corresponding value *f(v)* is assigned to the nature of the evaluated cell's surrounding area. The one that follows in the order is the one who has the minimum value of *f(v)*. [32]
Basic Model of A* Algorithm:

- **Raster method**: This technique, in effect, breaks down the entire environment that is to be explored into a grid made of small identical squares. This method focuses on the small squares technique in the representation of the obstacles' environment and its visualization. In this particular study, the whole in the experiment is represented with the aid of a two dimensional array where by 1 stands for the walls while 0 stands for normal cells. [31]
- **Manhattan distance formula**: Manhattan distance can be used as a heuristic if direction of movement is restricted to vertical or horizontal. The active nodes that are set to be searched next are referred to as the open list. The nodes that have been already searched are collectively referred to as the close list. The first step that the direction finder takes is to superimpose square grids over the area to be searched. A grid will have a 2D array digestion. Each element in the array corresponds to a grid and its state is either walkable (without obstruction) or un-walkable (with obstruction). Every obstacles position is centered at the grid therefore the grid cells will not be occupying the space of the nodes. [33]

2) **Dijkstra's Algorithm**: Dijkstra Algorithm is a graph-based algorithm that is used to find shortest distance between two points or nodes in a weighted graph. [36] In autonomous driving systems such as Autonomous Vehicles or Autonomous driving robots Dijkstra;s algorithm helps in: [35]

- **Nodes/Point**: Represent the location like starting point, destination of the vehicle. [39]
- **Edges**: Edges represent the route or way from one node to another or the network from source to destination. [39]
- **Edge Weights**: Edge Weights represent the cost required to travel from one node to another. It may be the distance between the two nodes, a traffic constraint that is used to measure traffic density, traveling time, etc. [39]

Dijkstra's algorithm works in the following way:
Initialization:

- Assign an initial cost of 0 to the starting node and infinity to all other nodes.
- Mark all nodes as unvisited. [36], [37]

Selection:

- Select the unvisited node with the smallest current cost. [36]

Relaxation:

- For each neighbor of the selected node:

– Calculate the cost to reach it from the start.
– If the calculated cost is smaller than the currently recorded cost, update it.

Repetition:

- Repeat steps 2 and 3 until the goal node is visited or all reachable nodes are processed.

Path Reconstruction:

- Backtrack from the goal node to the start node using the recorded shortest paths. [36]

Dijkstra's algorithm forms the foundation for many advanced path planning methods in autonomous vehicles. Its deterministic nature ensures reliability in static or semi-dynamic environments, making it a valuable tool in urban and controlled scenarios. [38]

3) **Breadth First Search Algorithm**:Breadth-first search (BFS) is one of the oldest and most fundamental graph traversal algorithms, influencing many other graph algorithms.
The breadth-first search algorithm (BFS) is utilized for the purpose of exploring a single connected graph component. It specifies A systematic way of traversing the vertices of the graph component. The basic outline of BFS states that there exists a loop that is executed while there are unvisited nodes in a queue. [40]

- **Distributed BFS with 1D Partitioning**:In a 1D partitioning of a graph, the vertices of the graph are partitioned in such a way that every vertex along with its incident edges, is the responsibility of one processor 1. The vertices contained within a processor are referred to as also belonging to that processor. Below is 1D P-way partitioning for an A, the adjacency matrix of the corresponding graph, neatly re-ordered so that processors own the vertices in a contiguous manner. [41]

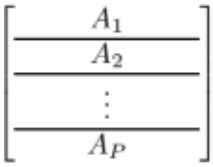

Fig. 4. 1D Partitioning [41]

- **Distributed BFS with 2D partitioning**:When we talk about the 2D partitioning of a graph, it refers to partitioning of edges of a graph in which an edge belongs to one and only one processor. In addition to this the vertices also have to be partitioned in such a way that each vertex belongs to one and only one. Each process holds a limited number of edges attached to the vertices that belong to the same process and a few that do not. This type of partitioning may be expressed by way of an adjacency matrix A which is reorganized such that

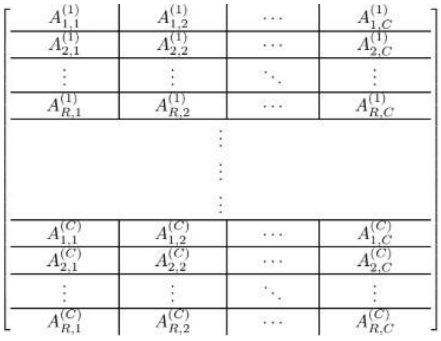

Fig. 5. 2D Partitioning [41]

vertices belonging to the same processor appear in a rectangular block. [41]

4) **Rapidly Exploring Random Tree (RRT)**: Rapidly-Exploring Random Tree (*RRT*) is a sampling-based path planning algorithm. It is especially useful for navigating high-dimensional, complex, and dynamic environments where traditional graph-based methods like *Dijkstra* or *A** might struggle. [44]
RRT is designed to efficiently explore large spaces by growing a tree structure from the start point toward the goal. Its key strength lies in its ability to handle non-holonomic constraints (e.g., those imposed by vehicle kinematics). [46]
The *RRT* algorithm works by sampling method. It first darts with a single node representing the vehicle's current position in the configuration. [44] That space includes parameters like the vehicle's position, orientation, and velocity, etc. Then the algorithm randomly selects a point in the space and finds a path or a way point towards that node. The node is added to the tree if no collisions are found with the obstacle in the way. This process of adding a new node to the tree is repeated until the destination node is reached. [47] If the algorithm detects an obstacle in its path to a node, then it backtracks in the tree and starts from the parent node to find a new way from the current position to the new node. [48] The *RRT* algorithm works in the best way for a dynamic environment with moving obstacles as it builds the path incrementally. [46] It respects vehicle dynamics such as minimum turning radius to adapt to real-world scenarios. However the paths produced by the *RRT* algorithm may be jagged or non-optimal and require post-processing most of the time. Can be slow in environments with dense obstacles or high-resolutions sampling requirements. [45]

5) **Ant Colony Algorithm**: The Ant Colony Optimization (ACO) algorithm is a nature-inspired, probabilistic approach to finding optimal paths, modeled after the foraging behavior of ants. In the context of autonomous vehicles, ACO is used to identify optimal routes by simulating the behavior of ants laying and following

pheromone trails, which represents the quality of a path. [49]

The Ant colony algorithm works in the following way:

a) **Initialization**: The road network is represented as a graph where intersections or waypoints are nodes and roads are edges.Each edge is initialized with a small amount of pheromone. [50]
b) **Ant Simulation**: Multiple artificial *ants* are placed at the starting node.
Each ant explores the graph by probabilistically choosing edges based on:
- **Pheromone Levels**: Higher pheromone levels increase the likelihood of selection.
- **Heuristic Information**: Factors like distance, traffic conditions, or road safety. [50]

c) **Pheromone Update**: After all ants complete their paths:
- Edges that are part of shorter, more efficient paths receive more pheromone (reinforcement).
- Pheromone on unused or less efficient edges evaporates over time. [50]

d) **Path Selection**: he path with the highest pheromone concentration is selected as the best route.
e) **Iteration**: The process is repeated for several iterations to refine the solution. [50]

The Ant Colony Optimization algorithm or *ACO* is often used together with other path planning algorithms like *A** or *RRT* to improve its performance and dynamic refinement. [51]
The *ACO* is used in autonomous vehicles due its dynamic adaptability. The pheromone levels can adapt to real time changes in the environment such as traffic updates or newly detected obstacles. [52] It can also be used for multiple objectives such as distance, safety, and energy efficiency. However the *ACO* is computationally intensive, simulating multiple ants and updating pheromones for large scale environments. Due to its higher convergence time, the algorithm takes many iterations to converge, making it slower for real-time tasks. [52]

V. Integration of Object Detection and Path Planning Algorithms

The above explored algorithms focus on Object Detection and Path Planning. These algorithms play a vital role in Autonomous Vehicles. AV's depend crucially on these algorithms for navigating in the real world where the obstacles are dynamic. Using these algorithms interchangeably can improve the performances of AV's making them more reliable, safe, easier to control and navigate.
Some of the Object Detection and Path Planning algorithms that go hand-in-hand with each other to provide the best combination of Integration for Autonomous Vehicles are:

1) **YOLO and Hybrid A***: *YOLO* (You only look once) is an object detection algorithm that detects the objects in the environment like pedestrians, cars, sidewalks, etc. [12]
The algorithm works by bounding box methods and classifying objects. Meanwhile *Hybrid A** algorithm is used for efficient path finding and path planning from destination to source with a dynamic environment . [33]
The integration of these algorithms include the input taken from the *YOLO* algorithm. The output of the algorithm is given to the *Hybrid A** algorithm which then searches for the shortest and efficient path for the vehicle to go from one destination to another. [32] The *A** algorithm takes care of the dynamic environment from *YOLO* algorithm and navigates a way around the obstacles while avoiding them.
2) **R-CNN and RRT**: Region based convolutional neural networks and Rapidly exploring Random trees works in the same flow as *YOLO* and *A**. The *R-CNN* algorithm is responsible for detecting objects in the environments such as pedestrians, cars, etc. It classifies the objects and sends it to the planning module. [19] The detected objects from the *R-CNN* are mapped to the world coordinate of the car. *RRT* begins from the vehicles current position and begins to find a possible path for the vehicle to navigate while detecting the obstacles zones and avoiding it. [44]
The advantages of this type of system is that both of the systems operate independently but integrate seamlessly. The *R-CNN* efficiently detects obstacles while *RRT* handles dynamic and cluttered environments by rapidly generating paths that adapt to changes in the environment.

This is just a higher version of how these algorithms work together and almost all the Object Detection and path planning algorithms work in this way. The basic sequence is to take an input from the environment, this is done by some object detection algorithm and this algorithm generates an output which is fed to some path planning algorithm which then finds the best possible path for the vehicle to navigate around the obstacles.

VI. Comparison of Methods and Performance Analysis

The performance analysis of Object Detection Algorithms is done by the metrics accuracy, precision and mean average precision. The performance analysis of path finding algorithms is done by path smoothness, computational time and safety. Comparison Of Object Detection Algorithms:

From the above comparison we find that the YOLO algorithm proves to be the best algorithm out of the five algorithms. It provides the fastest speed and accuracy in detecting objects. The YOLO algorithm is also the most suitable for real time tasks where the latency requirement is very low.

TABLE I
COMPARISON OF OBJECT DETECTION ALGORITHMS

Algorithm	**Speed**	**Accuracy / Optimality**	**Suitable For**	**Limitations**
YOLO	Very Fast	High for real-time detection	Real-time Systems	Small over-lapping Objects
HOG	Fast	Good in simple en-vironments	Lightweight Applica-tions	Scale Rotation Variance
R-CNN	Moderate	High	Offline Processing	Computation- ally Expensive
R-FCN	Fast	High for multi-scale objects	Multiscale detection	Complex to implement
Atrous Convolu- tion	Fast	Excellent for small objects	Dense ob- ject detec-tion	High resource re-quire-ment

Comparison of Path Planning Algorithms:
The comparison tells us that *A** algorithm is the best suitable path planning algorithm however the *A** algorithm is mainly used for static environments. [34] For Autonomous Vehicles the environment is dynamic, for AV's the best path planning algorithm is RRT [48] or Rapidly exploring Random Trees, this algorithm is fast and the most suitable for dynamic environments, the only limitation of this algorithm is that it requires path smoothing which increases the post processing time.

TABLE II
COMPARISON OF PATHFINDING ALGORITHMS

Algorithm	**Speed**	**Accuracy / Optimality**	**Suitable For**	**Limitations**
A*	Moderate	Optimal (with Heuristic)	Static envi-ronments	Slower in larger graphs
Dijkstra	Slow	Always Optimal	Grid Based Environ-ments	Inefficient for large graphs
BFS	Slow	Shortest Path	Small Scale Problems	Memory Intensive
RRT	Fast	Feasible (not optimal)	Dynamic, Complex environ- ments	Requires path smoothing
Ant Colony	Slow	Near-optimal Paths	Dynamic, multi-path	Slow Con-vergence

REFERENCES

[1] Parekh, Darsh, Nishi Poddar, Aakash Rajpurkar, Manisha Chahal, Neeraj Kumar, Gyanendra Prasad Joshi, and Woong Cho. 2022. "A Review on Autonomous Vehicles: Progress, Methods and Challenges" Electronics 11, no. 14: 2162. https://doi.org/10.3390/electronics11142162

[2] Margarita Mart´ınez-D´ıaz, Francesc Soriguera,Autonomous vehicles: theoretical and practical challenges,Transportation Research Proce- dia,Volume 33,2018,Pages 275-282,ISSN 2352-1465, Khan, Firoz & Ramasamy, Lakshmana & Kadry, Seifedine & Meqdad, Maytham N. & Nam, Yunyoung. (2021). Autonomous vehicles: A study of implementation and security. International Journal of Electrical and Computer Engineering. 11. 3013-3021. 10.11591/ijece.v11i4.pp3013-3021.

[3] Katona, Korne´l, Husam A. Neamah, and Pe´ter Korondi. 2024. "Obstacle Avoidance and Path Planning Methods for Autonomous Navigation of Mobile Robot" Sensors 24, no. 11: 3573. https://doi.org/10.3390/s24113573

[4] Daniel Foead, Alifio Ghifari, Marchel Budi Kusuma, Novita Hanafiah, Eric Gunawan,A Systematic Literature Review of A* Pathfind- ing,Procedia Computer Science,Volume 179,2021,Pages 507-514,ISSN 1877-0509,

[5] Qiu, Shaoming, Jikun Dai, and Dongsheng Zhao. 2024. "Path Plan- ning of an Unmanned Aerial Vehicle Based on a Multi-Strategy Im- proved Pelican Optimization Algorithm" Biomimetics 9, no. 10: 647. https://doi.org/10.3390/biomimetics9100647

[6] Lixing Liu, Xu Wang, Xin Yang, Hongjie Liu,

Jianping Li, Pengfei Wang,Path planning techniques for mobile robots: Review and prospect,Expert Systems with Applications,Volume 227,2023,120254,ISSN 0957-4174,

[7] Novak, Gregor & Seyr, Martin. (2004). Simple Path Planning Algorithm for Two-Wheeled Differentially Driven (2WDD) Soccer Robots.. 91- 102.

[8] Karur, Karthik, Nitin Sharma, Chinmay Dharmatti, and Joshua E. Siegel. 2021. "A Survey of Path Planning Algorithms for Mobile Robots" Vehicles 3, no. 3: 448-468. https://doi.org/10.3390/vehicles3030027

[9] https://doi.org/10.48550/arXiv.1506.02640

[10] Lavanya, Gudala & Pande, Sagar. (2023). Enhancing Real-time Object Detection with YOLO Algorithm. EAI Endorsed Transactions on Inter- net of Things. 10. 10.4108/eetiot.4541.

[11] Terven, Juan, Diana-Margarita Co´rdova-Esparza, and Julio-Alejandro Romero-Gonza´lez. 2023. "A Comprehensive Review of YOLO Architec- tures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS" Machine Learning and Knowledge Extraction 5, no. 4: 1680-1716.

[12] N. M. Krishna, R. Y. Reddy, M. S. C. Reddy, K. P. Madhav and G. Sudham, "Object Detection and Tracking Using Yolo," 2021 Third International Conference on Inventive Research in Comput- ing Applications (ICIRCA), Coimbatore, India, 2021, pp. 1-7, doi: 10.1109/ICIRCA51532.2021.9544598.

[13] A. Sarda, S. Dixit and A. Bhan, "Object Detection for Autonomous Driving using YOLO [You Only Look Once] algorithm," 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Tirunelveli, India, 2021, pp. 1370-1374, doi: 10.1109/ICICV50876.2021.9388577

[14] L. Mao, M. Xie, Y. Huang and Y. Zhang, "Preceding vehicle detection using Histograms of Oriented Gradients," 2010 International Conference on Communications, Circuits and Systems (ICCCAS), Chengdu, China, 2010, pp. 354-358, doi: 10.1109/ICCCAS.2010.5581983

[15] Dalal, N., & Triggs, B. (n.d.). Histograms of Oriented Gradients for Hu- man Detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). doi:10.1109/cvpr.2005.177

[16] Terayama, Masahiro & Shin, Jungpil & Chang, Won-Du. (2009). Object Detection using Histogram of Oriented Gradients.

[17] Kaur, Ravpreet, and Sarbjeet Singh. "A comprehensive review of object detection with deep learning." Digital Signal Processing 132 (2023): 103812.

[18] S. Azam, A. Rafique and M. Jeon, "Vehicle pose detection using region based convolutional neural network," 2016 International Conference on Control, Automation and Information Sciences (ICCAIS), Ansan, Korea (South), 2016, pp. 194-198, doi: 10.1109/ICCAIS.2016.7822459

[19] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Region-Based Con- volutional Networks for Accurate Object Detection and Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 1, pp. 142-158, 1 Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[20] Kishore Anthuvan Sahayaraj K., Balamurugan G.,Faster region based convolution neural network with context iterative refinement for object detection,Measurement: Sensors,Volume 31,2024,101025,ISSN 2665- 9174,

[21] Girshick, Ross & Donahue, Jeff & Darrell, Trevor & Malik, Jitendra. (2015). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 38. 1-1. 10.1109/TPAMI.2015.2437384.

[22] C. C. T. Mendes, V. Fre´mont and D. F. Wolf, "Exploiting fully convolu- tional neural networks for fast road detection," 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 2016, pp. 3174-3179, doi: 10.1109/ICRA.2016.7487486.

[23] Wang, Yitong, et al. "Detecting faces using region-based fully convolu- tional networks." arXiv preprint arXiv:1709.05256 (2017).

[24] Kim, Hongjo, et al. "Detecting construction equipment using a region- based fully convolutional network and transfer learning." Journal of computing in Civil Engineering 32.2 (2018): 04017082.

[25] Kang, Kai, and Xiaogang Wang. "Fully convolutional neural networks for crowd segmentation." arXiv preprint arXiv:1411.4464 (2014).

[26] "Y. Sun, L. Wang, Y. Chen and M. Liu, ""Accurate Lane Detec- tion with Atrous Convolution and Spatial Pyramid Pooling for Au- tonomous Driving,"" 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), Dali, China, 2019, pp. 642-647, doi: 10.1109/ROBIO49542.2019.8961705."

[27] Mohamed, Nur Ayuni, Mohd Asyraf Zulkifley, and Siti Raihanah Ab- dani. "Spatial pyramid pooling with atrous convolutional for mobilenet." 2020 IEEE student conference on research and development (SCOReD). IEEE, 2020.

[28] Ma, J., Dai, Y., & Tan, Y.-P. (2019). Atrous convolutions spatial pyramid network for crowd counting and density estimation. Neurocomputing.

[29] Men, Kuo, et al. "Cascaded atrous convolution and spatial pyramid pooling for more accurate tumor target segmentation for rectal cancer radiotherapy." Physics in Medicine & Biology 63.18 (2018): 185016.

[30] Yan, Yumeng. (2023). Research on the A Star Algorithm for Finding Shortest Path. Highlights in Science, Engineering and Technology. 46. 154-161. 10.54097/hset.v46i.7697.

[31] Duchonˇ, Frantisˇek, et al. "Path planning with modified a star algorithm for a mobile robot." Procedia engineering

96 (2014): 59-69.

[32] G. Tang, C. Tang, C. Claramunt, X. Hu and P. Zhou, ”Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in a Port Environment,” in IEEE Access, vol. 9, pp. 59196-59210, 2021, doi: 10.1109/ACCESS.2021.3070054

[33] Liu, L.; Wang, B.; Xu, H. Research on Path-Planning Algorithm Integrating Optimization A-Star Algorithm and Artificial Potential Field Method. Electronics 2022, 11, 3660. https://doi.org/10.3390/electronics11223660

[34] Alshammrei, Shaher & Boubaker, Sahbi & Kolsi, Lioua. (2022). Im- proved Dijkstra Algorithm for Mobile Robot Path Planning and Ob- stacle Avoidance. Computers, Materials & Continua. 72. 5939-5954. 10.32604/cmc.2022.028165.

[35] Javaid, Adeel. (2013). Understanding Dijkstra Algorithm. SSRN Elec- tronic Journal. 10.2139/ssrn.2340905.

[36] Huijuan Wang, Yuan Yu, & Quanbo Yuan. (2011). Application of Dijkstra algorithm in robot path-planning. 2011 Second Interna- tional Conference on Mechanic Automation and Control Engineering. doi:10.1109/mace.2011.5987118

[37] Chen, Yi-zhou, et al. ”Path optimization study for vehicles evacuation based on Dijkstra algorithm.” Procedia Engineering 71 (2014): 159-165.

[38] D. Fan and P. Shi, ”Improvement of Dijkstra’s algorithm and its application in route planning,” 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, China, 2010, pp. 1901-1904, doi: 10.1109/FSKD.2010.5569452.

[39] Holdsworth, Jason. (1999). The Nature of Breadth-First Search.

[40] A. Yoo, E. Chow, K. Henderson, W. McLendon, B. Hendrickson and U. Catalyurek, ”A Scalable Distributed Parallel Breadth-First Search Algorithm on BlueGene/L,” SC ’05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing, Seattle, WA, USA, 2005, pp. 25-25, doi: 10.1109/SC.2005.4.

[41] J. Silvela and J. Portillo, ”Breadth-first search and its application to image processing problems,” in IEEE Transactions on Image Processing, vol. 10, no. 8, pp. 1194-1199, Aug. 2001, doi: 10.1109/83.935035.

[42] Bonifacius Vicky Indriyono, and Widyatmoko. “Optimization of Breadth-First Search Algorithm for Path Solutions in Mazyin Games”. International Journal of Artificial Intelligence & Robotics (IJAIR), Vol. 3, no. 2, Nov. 2021, pp. 58-66, doi:10.25139/ijair.v3i2.4256.

[43] Rodriguez, Xinyu Tang, Jyh-Ming Lien and N. M. Amato, ”An obstacle-based rapidly-exploring random tree,” Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., Orlando, FL, USA, 2006, pp. 895-900, doi: 10.1109/ROBOT.2006.1641823.

[44] Martinez, Fredy & Jacinto, Edwar & Montiel, Holman. (2023). Rapidly Exploring Random Trees for Autonomous Navigation in Observable and Uncertain Environments. International Journal of Advanced Computer Science and Applications. 14. 10.14569/IJACSA.2023.0140399.

[45] L. G. D. O. Ve´ras, F. L. L. Medeiros and L. N. F. Guimara´es, ”Sys- tematic Literature Review of Sampling Process in Rapidly-Exploring Random Trees,” in IEEE Access, vol. 7, pp. 50933-50953, 2019, doi: 10.1109/ACCESS.2019.2908100.

[46] C. E. Tuncali and G. Fainekos, ”Rapidly-exploring Random Trees for Testing Automated Vehicles,” 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 2019, pp. 661- 666, doi: 10.1109/ITSC.2019.8917375.

[47] H. Umari and S. Mukhopadhyay, ”Autonomous robotic explo- ration based on multiple rapidly-exploring randomized trees,” 2017 IEEE/RSJ International Conference on Intelligent Robots and Sys- tems (IROS), Vancouver, BC, Canada, 2017, pp. 1396-1402, doi: 10.1109/IROS.2017.8202319.

[48] M. Dorigo, M. Birattari and T. Stutzle, ”Ant colony optimization,” in IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28-39, Nov. 2006, doi: 10.1109/MCI.2006.329691

[49] Yang, Jingan, and Yanbin Zhuang. ”An improved ant colony opti- mization algorithm for solving a complex combinatorial optimization problem.” Applied soft computing 10.2 (2010): 653-660.

[50] M. Brand, M. Masuda, N. Wehner and Xiao-Hua Yu, ”Ant Colony Op- timization algorithm for robot path planning,” 2010 International Con- ference On Computer Design and Applications, Qinhuangdao, China, 2010, pp. V3-436-V3-440, doi: 10.1109/ICCDA.2010.5541300.

[51] Ying-Tung Hsiao, Cheng-Long Chuang and Cheng-Chih Chien, ”Ant colony optimization for best path planning,” IEEE Interna- tional Symposium on Communications and Information Technology, 2004. ISCIT 2004., Sapporo, Japan, 2004, pp. 109-113 vol.1, doi: 10.1109/ISCIT.2004.1412460.