Computing Surveys

A Systematic Literature Review on Trajectory Planning in Low-Structured Environments

Journal:	*Computing Surveys*
Manuscript ID	CSUR-2023-0722
Paper:	Long Survey Paper
Date Submitted by the Author:	31-Aug-2023
Complete List of Authors:	Souza de Oliveira, Cristiano; Federal University of Santa Catarina, epartment of Informatics and Statistics von Wangenheim, Aldo; Universidade Federal de Santa Catarina, Pós-Graduação em Ciência da Computação (PPGCC). Departamento de Informática e Estatística (INE).; Universidade Federal de Santa Catarina, Image Processing and Computer Graphics Lab. Instituto Nacional para Convergência Digital (INCoD). Departamento de Informática e Estatística (INE).
Computing Classification Systems:	

A Systematic Literature Review on Trajectory Planning in Low-Structured Environments

CRISTIANO SOUZA DE OLIVEIRA, Federal University of Santa Catarina, Brazil
ALDO VON WANGENHEIM, National Institute for Digital Convergence (INCoD), Brazil

Low-structured environments with poor driving conditions, such as dirt roads and suburban areas in developing countries, often present challenges for autonomous vehicles, especially in finding a feasible path while handling unreliable data. This paper offers a Systematic Review of the state-of-the-art in trajectory planning for car-like vehicles driving in low-structured and non-structured environments. The Review was performed on scientific literature databases, considering articles published in the last 12 years, resulting in 42 papers after filtering. We present a summary of recurrent problems and proposed approaches.

CCS Concepts: • **Computer systems organization** → **Robotic autonomy**; **Robotic control**; *Embedded systems*; Distributed architectures; *Sensors and actuators*; Real-time systems; • **Software and its engineering** → *Embedded software*; *Real-time systems software*; State systems.

Additional Key Words and Phrases: Trajectory Planning, Self-driving Cars, Obstacle Avoidance, Robot Control, Occupancy Grid, OctoMap, CostMap , Mesh, Gradient Grid, State Lattice, Local Planning, Global Planning

ACM Reference Format:
Cristiano Souza de Oliveira and Aldo von Wangenheim. 2023. A Systematic Literature Review on Trajectory Planning in Low-Structured Environments. 1, 1 (August 2023), 23 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The ability to reliably manage environments with a low level of road structuring is still an obstacle to be solved in the field of autonomous vehicular navigation[25]. These are not only off-road scenarios, deserts and heavily cluttered places such as narrow passages in forests, but also dirt roads or streets in the countryside and suburban areas in in-development countries, which often lack maintenance and signalization. Such *low-structured environments* with poor driving conditions often present both the challenge of finding a feasible path among defiant driving scenarios and problems with unreliable location systems and sensors, the absence of previous map information and even of driving rules, and an unstable or absent internet connection. In addition, navigable pathway detection itself in such a scenario is still a challenge to be overcome, due to many variations in surface type and pavement shape and maintenance [59]. At the same time, dirt roads can represent most countryside pathways in a country, especially in economically underdeveloped nations, where semi-autonomous driving could dramatically improve goods distribution and commodity transportation logistics.

Authors' addresses: Cristiano Souza de Oliveira, c.s.oliveira@posgrad.ufsc.br, Federal University of Santa Catarina, Eng. Agronômico Andrei Cristian Ferreira, Florianópolis, Santa Catarina, Brazil, 88040900; Aldo von Wangenheim, aldo.vw@ufsc.br, National Institute for Digital Convergence (INCoD), Eng. Agronômico Andrei Cristian Ferreira, Florianópolis, Santa Catarina, 88040900, Brazil.

Manuscript submitted to ACM

Figure 1 shows typical suburban scenarios from southern Brazil [59][58][47][48][57]: Traffic signs are nonexistent (a-i); road markings are either nonexistent (a, d-f) or extremely damaged (b, c); pedestrian areas either mix smoothly with driving pathways (d, h, i) or are nonexistent (a, e, g); speed bumps are either unmarked (f) or present damaged markings (c) and the pavement presents damages such as potholes and cracks (b). These are suburban scenarios from a relatively well developed region in southern Brazil, where autonomous individual transportation is very likely to be occurring in the near future. If long distance goods and commodities logistics through the countryside is also taken into consideration, autonomous vehicles will have to cope with path-interfering obstacles such as long deep ruts and large mud puddles. We understand that all these are scenarios that impose the need for differentiated vehicular perception and path planning [59][47][48].

Fig. 1. Unstructured scenarios in typical suburban areas in southern Brazil that, even not being off-road situations, afford special navigational capabilities [58][59][47][48]. Images from [57]. Source: the authors.

Self-driving cars are complex systems that have to account not only for internal constraints, such as the limited computing power of embedded hardware, real-time and fault-tolerant requirements, sensor limitations, and unreliability, but also for the vehicle's kinodynamics, environmental unpredictability, and driving rules.

The objective of the present work is to present the state-of-the-art in the field of path and trajectory planning for car-like autonomous navigation in *unstructured* and *low-structured* environments, showing and discussing the results of a systematic literature review (SLR). We focused on research published in the last 12 years and discuss and compare common approaches, issues, technology and possible solutions in this subject.

This paper is structured as follows: In Section 2, an overview of the adopted methodology for the systematic review is presented. Section 3 presents related work on reviewing trajectory planning for ground-based autonomous car-like vehicles. Section 4 presents an architecture classification and an overview of the selected sources. Section 5 presents a discussion of the results of this review. Section 6 concludes with highlights of the state of the art and discusses possible future work.

2 SYSTEMATIC ANALYSIS

In the context of this SLR, trajectory planning is understood as *the act of defining a set of motions that are to be followed by a vehicle to self-move from a starting point to an ending point, respecting constraints linked to the physical characteristics of the vehicle and the terrain or related to safety limitations.* As *non-structured* and *semi-structured* environments we consider driving outdoor pathways that present a low level of driving structure, such as off-road scenarios, deserts, roads without pavement or that often lack maintenance, and similar scenarios, excluding heavy, cluttered outdoor environments such as forest paths.

2.1 Methodology

We adopted a five-step approach in this SLR, following the protocols proposed by Kitchenham [37] and Biolchini [6] for SLRs in the Software Engineering field (see figure 2).

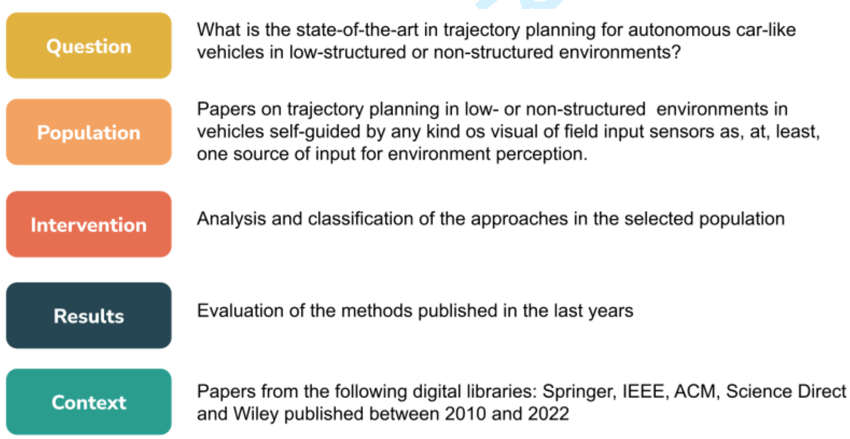


Fig. 2. General procedure used in the SLR

2.2 Inclusion and Exclusion Criteria

The *inclusion criteria* for this review are the following:

- Papers written in English, published between jan/2010 and dec/2022 on the areas of Computer Science or Robotics;
- Papers on self driving path planning, motion planning or trajectory planning in low or non-structured environments, on vehicles that are self-guided by any kind of visual or field input sensors as, at least, one source of input for environment perception;

The following *exclusion criteria* were employed to eliminate a candidate study from this review

- Short papers such as abstracts or expanded abstracts;
- Works on indoor environments, heavily cluttered non-drivable environments, such as forests, or micro-environments like medical applications;
- Papers unrelated to car-like vehicle trajectory planning, such as non-wheeled robots, bipedal or quadruped, aerial or aquatic vehicles and motion planning of robot arms or other robot parts not related to driving or collaborative planning for multi-agent or multi-vehicle scenarios;
- Papers that do not present any test case, neither in a Simulator nor on Unmanned Vehicles or that do not report the reaching of a feasible path as their solution.

2.3 Search execution

We employed three-step approach for the selection of the papers. Initially, a search was performed with the query strings defined in Appendix A, which resulted in a total of 1946 candidate papers. As a second step,we filtered the initial data, excluding papers that do not have at least one of the following words in its title or abstract: "unstructured, low-structured, semi-structured, off-road, offroad, non-urban and nonurban". This result is then filtered, eliminating repetitions where the same paper is published in more than one of the sources. This is done by comparing the title and DOI. Then, any paper that presents one of the following keywords in its title or abstract was removed:

- Underwater, UAV, indoor, SUAS, aerial, Humanoid, cellular networks, femtocell, robot arm, multi-arm, 6-DOF, human environment, UVMS, climbing robot, CLIBO, snake-like, homotopy, multi-material, construction, Editorial:, urban, exoskeleton, multi-robot, surgical, robotic construction, History of, healthcare, Industry 4.0, excavator, Logistics, jumping robots, Riverine, submerged, social, networks, unstructured meshes, governance, agile, scrum

Every paper was manually filtered by title, abstract, keywords, images, introduction, and conclusion, following the inclusion and exclusion criteria, resulting in *42 primary sources* selected for this review. The complete methodology and a detailed description of each paper can be found in [65].

3 RELATED WORK

This review shares some common nomenclature with papers in the field of autonomous driving in structured environments, such as the division of responsibilities in a self-driving system. In [5] the architecture is divided into perception and decision-making layers. The perception system is generally subdivided into features such as localization, static obstacles mapping, moving obstacles detection and tracking, road mapping, traffic signalization detection and recognition. The decision-making system is commonly partitioned into route planning, path planning, behavior selection, motion planning, and control. We find some important definitions such as (1) the Route Planner as responsible for computing a route through a road network from the initial position to the final position, defined by a user operator; (2) the Path Planner, responsible for computing a set of paths, considering the current route, state, internal representation of the environment and traffic rules and a Behavior Selector, responsible for choosing the current driving behavior, such as lane keeping, intersection handling, traffic light handling; (3) the Motion Planner subsystem, responsible for computing a trajectory from the current state to the current goal, which must follow the path defined by the Behavior Selector as closely as possible while satisfying the car's kinematic and dynamic constraints; (4) the Obstacle Avoider, which receives the trajectory computed by the Motion Planner subsystem and changes it (typically reducing the velocity) if necessary to avoid collisions.

[51] decompose the decision-making system into four components: Route Planning, Behavioral Layer, Motion Planning and Local Feedback Control. At the highest level, a route is planned through the road network. This is followed by a behavioral layer, which decides on a local driving task that progresses the car towards the destination and abides by the rules of the road. A motion planning module then selects a continuous path through the environment to accomplish a local navigational task. A Control System then corrects errors in the execution of the planned motion. The motion planning problems adopting numerical solution are divided into three main categories: (1) Variational methods, which represent the path as a function parameterized by a finite-dimensional vector, and the optimal path is sought by optimizing over the vector parameter using non-linear continuous optimization techniques; (2) Graph-search methods, which seeks a path by performing a search for a minimum-cost path in a graph and (3) Incremental search methods, that sample the configuration space and incrementally build a reachability graph, maintaining a discrete set of reachable configurations and feasible transitions, finding a solution once the the graph is large enough so that at least one node is in the goal region.

More recently, [50] performed a survey particularly focused on deep learning methods. In perception, Convolutional Neural Networks (CNN) emerge as one of the most common approaches, both for monocular and binocular vision, regarding Object Detection, Lane Detection and Traffic Sign interpretation. Recurrent Neural Networks (RNN), Deep Stacked Auto Encoders (DAE) are also used, along with hybrid solutions. For scene classification and understanding, which refer to judging the current traffic scene and environmental information of the vehicle, the use of multi-resolution CNN is proposed, following the work of [69]. Other applications, particularly in path planning and motion control, are also pointed out. In path planning, Particle Swarm Optimization (PSO), Genetic Algorithms (GA) and methods based on deep reinforcement learning are highlighted. The work of [63] on stacking CNN and LTSM layers as a composite neural network for motion control is also present here. [2] presented a survey focusing on Deep Reinforcement Learning for motion planning, evaluating Deep-Q Learning Network, Deep Neural Networks, Deep Deterministic Policy Gradients (DDPG) and other techniques, classified by observation, action, and rewarding type.

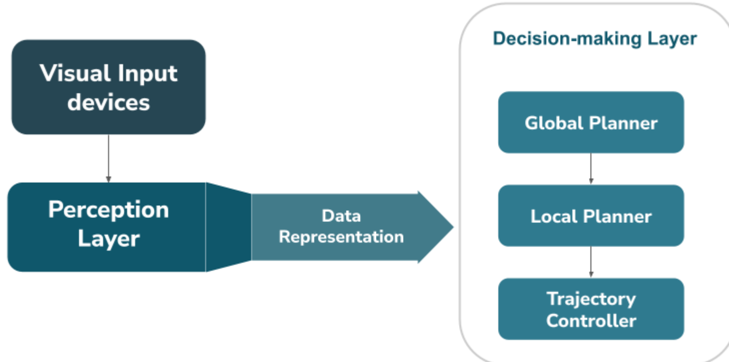


Fig. 3. Adopted architecture classification.

4 ARCHITECTURE CLASSIFICATION

In this review, a two-layer division is adopted, aligned to [5] (see figure 3), reflecting a division of tasks that is commonly present in most of the analyzed papers: Perception Layer and Decision-Making Layer.

4.1 Perception Layer

A Perception Layer, responsible for environment perception such as computer vision, and sensor data gathering. In this section we present both the visual input type for the papers selected in this review and the most common adopted data representation types.

4.1.1 Visual Input Type. In this section, the following visual input types included in this review are presented: Monocular cameras, Stereo cameras, LiDARs and Infrared sensors.

Monocular cameras can provide a relatively cheap passive solution for computer vision. 33% of the papers in this review adopted a camera-only solution for image perception. Most of them have adopted only monocular camera [12, 15, 16, 43, 49, 53, 62], comprising 60% of the camera-only papers.

Stereo cameras are devices that employ two or more cameras to estimate depth. Papers using stereo cameras adopted data representation types heterogeneously, such that in [60] traditional Occupancy Grid is used, whereas Traversability Map is used by [61] and [75], Elevation Map in [71], Cost Map in [46] and OctoMap in [38]. [44] combines Cost Map and Occupancy Grid.

LiDAR (Light detection and ranging) is an active sensor that typically emits pulsed light waves, which bounce off surrounding objects and return to the sensor. By computing the time they take to return to the sensor, it is possible to estimate the traveled distance, map the surroundings, and build a point cloud for object detection and depth calculation. Papers using LiDAR only are the most heterogeneous group in terms of data representation type, using Occupancy Grids [8, 28, 33, 70], Traversability Map [7], Gradient Grid [73], Velocity Grid [30], Elevation Maps [20, 72] and Triangular Mesh [22].

LiDARs can also be used in combination with cameras. They are much more frequently used with monocular cameras, as those devices usually fail to estimate depth. Works that combine monocular cameras and LiDAR adopt Occupancy Grid [34], Cost Map [54, 64, 74], Elevation Map and Cost Map [54] and OctoMap [4]. In [74] this combination splits them by feature, joining GPS and LiDAR data are used for obstacle detection, road edge detection, and moving object detection, while camera input is used to detect road edges, lanes, and traffic lights. In [24] we did not identify a data representation type specification. A combination of LiDAR and stereo-cameras is less common, as [75] is the only paper adopting both input devices. It uses the Traversability Map as a data representation.

Infrared sensors (IR) can be passive or active. They are made of pyroelectric materials and work in the infrared spectrum, which the human eye is unable to see. [21] is the only paper to adopt this sensor, relying on an array of IR to perform object detection for collision avoidance. The main issue with these devices is their very short range of operation.

4.1.2 Data Representation Type (DR). Computer vision in this review is often associated with a particular choice of data structure to hold environment perception. In this section, we discuss the most common ones: Occupancy Grid, Traversability Map, Elevation Map, Gradient Grid, Velocity Grid, OctoMap, Cost-Map, State Lattice and Mesh. Table 1 shows the classification of DS by Visual Input Type.

Occupancy Grid is a tessellated 2D grid that stores information regarding which areas of a robot's operating environment are occupied and which are empty on each cell by setting a certainty factor relating to the confidence that the particular cell is occupied. One of the advantages of this data type is its simplicity, especially in terms of memory management [11]. The most commonly adopted DR is the Occupancy Grid (OG), comprising 40% of the papers selected for this review. OG is used with all input sources listed in this review, with the exception of the combination of stereo-camera and LiDAR. In [15, 16, 43, 53], OG is adopted as DR for graph search local planning, using monocular cameras as input. [12] also combines it with a monocular camera but adopts RRT instead. In [44, 60], the local graph search is performed with OG data from stereo-cameras. LiDAR and OG are used to perform graph search in [8, 33], incremental search in [70] and Multi-path generation and selection in [28]. A variational approach was adopted for OG data from a combination of LiDAR and camera input in [45]. In [21], graph search is performed on infrared OG sensor data. Other non-classified types of visual input in this review adopting OG are [27], performing local planning using a variational approach, [31] implementing an end-to-end machine learning solution, a Dynamic Window Approach in [1] and graph search in [13].

Traversability Map holds information about a vehicle's traversability, which can be defined as the capability to reside over a terrain region under an admissible state wherein it is capable of entering given its current state, taking into account the terrain model and vehicle models, the kinematic constraints, and a set of optimality criteria. Geometry-based methods are the majority of the methodologies for analyzing traversability [52]. In [61] it is associated with swarm recognition path planning, using stereo cameras as the visual input source. In [7], LiDAR data is used in combination with a minimizing graph search algorithm, such as Dijsktra. [75] plans locally by combining LiDAR and stereo-camera images, using a Traversability Map combined with an Elevation Map, and using a Dynamic Window Approach.

Digital Elevation Map (DEM) are 3-D representations of rugged terrain, often implemented as grid-based representations, where the position of a point in the cartesian coordinate system can be derived from the measured range, obtained using passive or active sensing, and the direction of the bean at that point. Stereo cameras are typically employed for passive sensing. In active sensing, a direct measure is taken, suffering less interference from outside illumination, so it's preferred under this condition. Once the range image is obtained, the values θ and ϕ of the horizontal and vertical angles are computed using the step between two consecutive values, and the x,y,z coordinates are calculated using trigonometry [39]. DEM is the most frequently adopted when using machine learning [54, 68, 72]. In [72], an ant colony approach was applied, with a LiDAR as the visual input source. [54] also uses LiDAR, but in combination with a monocular camera, and adopts a learning algorithm based on rewards for the planning phase. In [68] data is obtained from a dataset. Classical approaches using DEM were [20, 71, 75]. Since Topological Maps and DEM are very similar, we have decided to classify them as DEM in this review.

Gradient Grid is a variation of an DEM that, for each cell, stores the angle between the tangent plane of a point and the horizontal plane. This angle is known as the gradient of the cell. This approach helps to account not only for the terrain elevations but also for possible inclinations a vehicle may face when navigating those variations in height. This DR was only adopted in [73], which uses LiDAR as an input source and a cost-field approach for local planning.

Velocity Grid is another variation of DEM that stores the maximum admissible velocity to cross a cell, accounting for both terrain and vehicle constraints. This DR was only adopted in [30], which uses LiDAR as an input source and a variational approach for local planning.

OctoMap is a framework for 3-D representation that seeks to perform probabilistic representation, model unmapped areas, and keep memory and computing efficiency [32]. It is based on Octrees, which are tree data structures where every node that is not a leaf has eight child nodes. The initialization of map volumes is delayed until measurements need to be integrated, so the map only contains volumes that have been measured. The probability of a leaf node n to be occupied given a measurement is estimated by an equation that takes a prior probability, the previous estimated value, and the current measurement as inputs. As measurements are taken, the probability changes towards a greater or lesser chance of being occupied. When a threshold is reached, it is assumed to be either occupied or free. This DR was used by [38] in combination with graph search and having a stereo camera as an input source, and in [9], which adopts a variational approach based on information theory and combines a LiDAR and a monocular camera as inputs.

Cost-Map is a grid where each cell contains one or many values that represent the average cost of traversing an area [41]. The total average cost of a path can be obtained by summing the values of each cell. Usually, the cost values vary from 0, for a fully traversable area, to ∞, for an obstacle or a non-traversable area. Any value in between corresponds to some degree of traversability [36]. [49] combines it with a monocular camera and graph search local planning. Stereo-cameras are used by [44], with a graph search approach, [46], with incremental search. and [38] with Dynamic Window Approach. A combination of Cost-Map, LiDAR and monocular camera as input was adopted by [54, 64, 74].

State Lattice is a search space built to satisfy motion constraints [55], representing a set of all reachable configurations, built by discretizing the C-space into a hyperdimensional grid, where for every node a feasible connection is attempted, resulting in a set of all possible feasible paths. It can be viewed as a generalization of a grid. It is used by [62], which adopts a monocular camera and a sampling based approach using RRT as local planner and [67], which uses graph search.

Mesh is a very common technique in computer graphics to represent 3-D data as a series of shapes. Triangular meshes, in particular, are composed of irregular triangles and can be used as a simplification for a 3-D raw point cloud. Raw data can be converted into meshes by applying a function, such as 2D Delaunay. In this case, the memory required to hold the data is lower than the initial raw data [23]. Compression techniques [42] can also be applied to reduce memory consumption . [22] uses Triangular Mesh to hold data from LiDAR and associates it with a multi-path generation and selection local planning approach.

Table 1. Classification of data representation by visual input type

	Monocular camera	Stereo camera	LiDAR	Monocular camera and LiDAR	Stereo-camera and LiDAR	Infrared	Other
Occupancy Grid	[12, 15, 16, 43]	[44, 60]	[8, 28, 33, 70]	[45]	-	[21]	[1, 13, 27, 31]
Traversability Map	-	[61]	[7]	-	[75]	-	
Gradient Grid	-	-	[73]	-	-	-	
Velocity Grid	-	-	[30]	-	-	-	
Elevation Map	-	[71]	[20, 72]	[54]	[75]	-	[3, 10, 68]
CostMap	[49]	[38, 44, 46]	-	[54, 64, 74]	-	-	
OctoMap	-	[38]	-	[4]	-	-	
State Lattice	[62]	-	-	-	-	-	[67]
Mesh	-	-	[22]	-	-	-	
Other	-	-	[66]	[24]	-	-	[26, 34]

4.2 Decision Making Layer

The Decision Making Layer is responsible for receiving a goal waypoint from a human operator and producing a series of control commands to the vehicle's actuators, such as steering angle definition, acceleration, brake activation, etc. In this section, different adopted solutions for low- or non-structured environments are discussed. We subdivided this layer into Global Planner, Local Planner and Trajectory Controller for classification.

4.2.1 The Global Planner (GP). It is responsible for interacting with offline maps in order to plan a path for the long run, which means that the path accounts mostly for a distance outside of the local perception range, so obstacles will most likely be unknown at this time. Most of the works in this review do not mention a GP. In [71] and in [1], the GP is given as a pre-computed mapping for a mission. In [27], the distant waypoints are stored as the vehicle learns them from a human driver. In [44], those waypoints are computed from a global topological map. Other works such as [8] D* Lite, [38] AD* and [64] Field D* use graph search, while in [12] RRT was adopted. [60] combines satellite-based GP to a local mapping to help future local planning by referencing the local map with the global coordinates.

4.2.2 The Local Planner (LP). It is responsible for planning for local reactions. It receives one or a a set of goal waypoints from the GP to perform a local planning decision, often producing feasible paths from the current location to achieve a local goal waypoint, but features such as performing emergency breaks or approaching a specific target can be present. Several variations were noticed in the approaches adopted in this review. They were grouped here as Graph search, Dynamic Window Approach, Multi-path generation and selection, Incremental search, Variational approach and Machine Learning. Table 2 shows the classification of LP techniques by Data Representation Type.

Graph Search is the most common approach, accounting for almost 50% of the analyzed papers. Solutions based on or derived from A* account for 21% of the total. In [49], [66], [53] and [14] a Hybrid A* version is used. The first one brings a combination with a Global Planner developed using Lanenet2. The second one replaces its search policy to make it harder to decide to go backwards, since it is also not a common decision for a human driver. In [53], some extensions to the planner were made to make it plan for the next two goal waypoints instead of one, avoiding dead-ends, among other features. In [14], post-processing is done to smooth the resulting path, and two opposing heuristics are used in combination when planning. [34] proposes a method of choosing vertices and edges that minimizes the dispersion of the vertices in the metric space induced by trajectory cost, optimally covering the space of feasible trajectories. Dispersion optimization is then used to select a tiled graph of motion primitives offline for later fast online search.

In [67], A* is combined with CNN for energy-cost estimation from a point cloud input. In [24], adaptive A* is used to improve speed when searching for a feasible path, and a root-mean-square is used for cost evaluation. In [43], the authors suggest an approach based on deep-first forward search (DF-FS) combined with a graph search algorithm such as A*, ARA* etc., where the first one is used when the vehicle is not in a certain range of waypoints with good positioning. [15] also employs a similar two-heuristic function approach, but with A*, making use of post-processing to prevent path discontinuity for the recently discovered ones. In [16], a modified A* admits many leaf goal-points. The least-cost path is returned once a timeout occurs, so there's no guarantee of an optimal result, but the processing time is clamped. [10] presents a modified version of A* as well, in which the topological connections of the local map serve as a navigation function for the computation of heuristic values.

Incremental versions of A* are also used. In [44], a multi-feature LP that uses Determined Finite Automata (DFA) to perform several tasks based on the vehicle state, adopted Field D* [17], an interpolation-based variation of D*, to compute paths when in a corner or intersection. A parabolic function and the centerline are adopted otherwise. [8] proposes to use Support Vector Machine (SVMLP) in conjunction with a feasible path computed by a Global Planner using D* Lite.

Solutions based on Dijkstra's algorithm are also present, generally combined with other approaches to improve performance or smoothness. In [7], trajectories are planned using a topological map G representing reachable locations (nodes) and possible transitions (edges). It relies on a behavior-based local piloting layer to avoid obstacles along the way using metrical short-term memory and uses Dijkstra's as the minimizing algorithm. [33] uses splines to build path segments, using Dijkstra to connect them to create an optimally smooth path. The smoothness of the path is also defined by the vehicle's speed plan: the faster it goes, the less curvy the path must be. In [3], three types of supervised learning schemes for building the traversability detection layer are used: multi-layer perceptrons, decision trees, and random forests, each outputing a traversable path. Then, Dijkstra is used to find an optimal path from the given result. In [73], a gradient cost field, where the cost is lower when the terrain is more flat, is built from the values of an Elevation Map, and later transformed into a network dataset. Dijkstra is then applied to obtain an optimal path. In [21], the local planning is performed by the wavefront propagation algorithm [40], which is a specialized version of Dijkstra's that avoids explicit construction of the priority queue. [20] also adopts wavefront and models the admissible free space as a set of nodes and a graph, in a concept called Admissible Space Trees (AST). A terrain map is performed by encoding ASTs in a Hierachical Topological Map, performing LP when building this map.

Dynamic Window Approach (DWA) is a technique that reduces the search space to the dynamic window, which consists of the velocities reachable within a short time interval and also allows the vehicle to stop safely. In [38] it is used for local planning, combined with AD*, for global planning. In [1], a method similar to DWA is applied [56]. Since this work is focused on motion planning, the idea here is to make the robot avoid obstacle collisions and then hand over the control back to the MP. Finally, in [75], Deep Neural Networks (DNN) fuse data of semantic segmentation from stereo cameras and LiDAR to build a traversability map and plan locally using DWA.

Multi-path generation and selection is a solution that rely on the projection or generation of several parallel paths from a reference path or waypoint, following motion constraints, which are then subject of a selection process to choose the best one. In [28], a set of tentacles corresponding to possible movements for the vehicle are generated. A decision value is computed from clearance, flatness, and trajectory values, and this value is analyzed along with a weight modifier that changes based on the current state of the vehicle. In [74], a series of lines with equal intervals were generated on a map, fusing lanes and obstacles that are all parallel to a line generated by the geographic information system (GIS). In [22], the algorithm computes a number of streamlines, which are the paths followed by a fluid molecule along its spatial displacement and are tangent to the velocity vectors for the steady-state case. It builds a list of roadmap candidates, which are the navigable paths, and the optimal one is selected by total length and energy consumption.

Incremental search: consists of techniques which sample the configuration space to build a reachability graph. Once this graph is large enough to touch the goal point or a goal region, a feasible path can be obtained. In [30], a Tree of Motion Patterns is built. The motion pattern is used to simplify the motion planning in two parts: a series of static commands that do not change and an array of oriented positions that represent the trajectory on which the vehicle would probably move when the command series is sent to it. [62] proposes a hierarchical planner capable of anytime planning, which uses a simplified vehicle model and a variant of RRT*, that works by sampling a pose in the continuum within some radius and yaw tolerance of a random waypoint. When a feasible path is found, it spends the rest of the free time optimizing it. In [70], a reference path is given and must be correctly followed. At first, a general regression neural network (GRNN) is used to find the centerline, which is checked for reliability. If not reliable, the LP is switched to sample a set of terminal states in the control space. A cubic Bezier curve is used to generate trajectories for every terminal state. An obstacle cost function evaluates the path candidates, and the optimal path with the lowest cost can be extracted from them. [46] performs path planning using another sample-based algorithm called Rapid Semi-Optimal Motion Planning (RASMO). It expands search trees according to the physical model under control input, where each node in the tree represents a state of the robot model, including pose and velocity. The optimal path is obtained by finding a tree that has the nearest node to the target state.

Variational approach is a technique that obtains optimal paths by performing optimizations over a function representation of a path. [26] treats the trajectory planning as an optimal control problem (OCP) and transforms it into a nonlinear programming (NLP) problem. As the OCP is highly non-convex and nonlinear, it is dependent on a proper initial guess. An initialization strategy decoupling the original scheme into two sub-tasks is proposed, reducing the equation violation by mainly optimizing vehicle positions and orientations, and trying to find a solution that strictly satisfies all inequalities by only optimizing the total travel time. [45] suggests the usage of Model Predictive Control (MPC) for achieving a more complex and predictable behavior. As it relies heavily on an accurate representation of the system dynamics model, which is often not possible, the learning-based Gaussian Process version (GP-MPC) was adopted. In [71], a combination of stereo-camera and polarization navigation sensors tries to mimic the way some animals guide themselves with vision and skylight polarization. The planner works by solving an optimization problem over the constructed graph of poses. In [27] a cost-valley approach to optimization is used. Initially, waypoints are given by another system layer, such as the Global Planner. The waypoints are optimized by a line simplification algorithm such as Douglas-Peucker (DP) or a local approach that considers the difference in the first-order derivative of the current path element and of the consecutive path elements when simplifying. The path is chosen by following a cost-based version similar to a potential field approach using Breshenham's algorithm. [4] proposes a planning solution based on Information Theory, where the Mutual Information (MI) and motion costs are computed over the values from a probability density function (PDF), allowing integrating the measurements into a probabilistic map representation using Bayesian updates. The plan bases on computing a control sequence for the vehicle's model that maximizes the ratio of mutual information between the map and future sensor observations, and the motion cost of the planned robot trajectory.

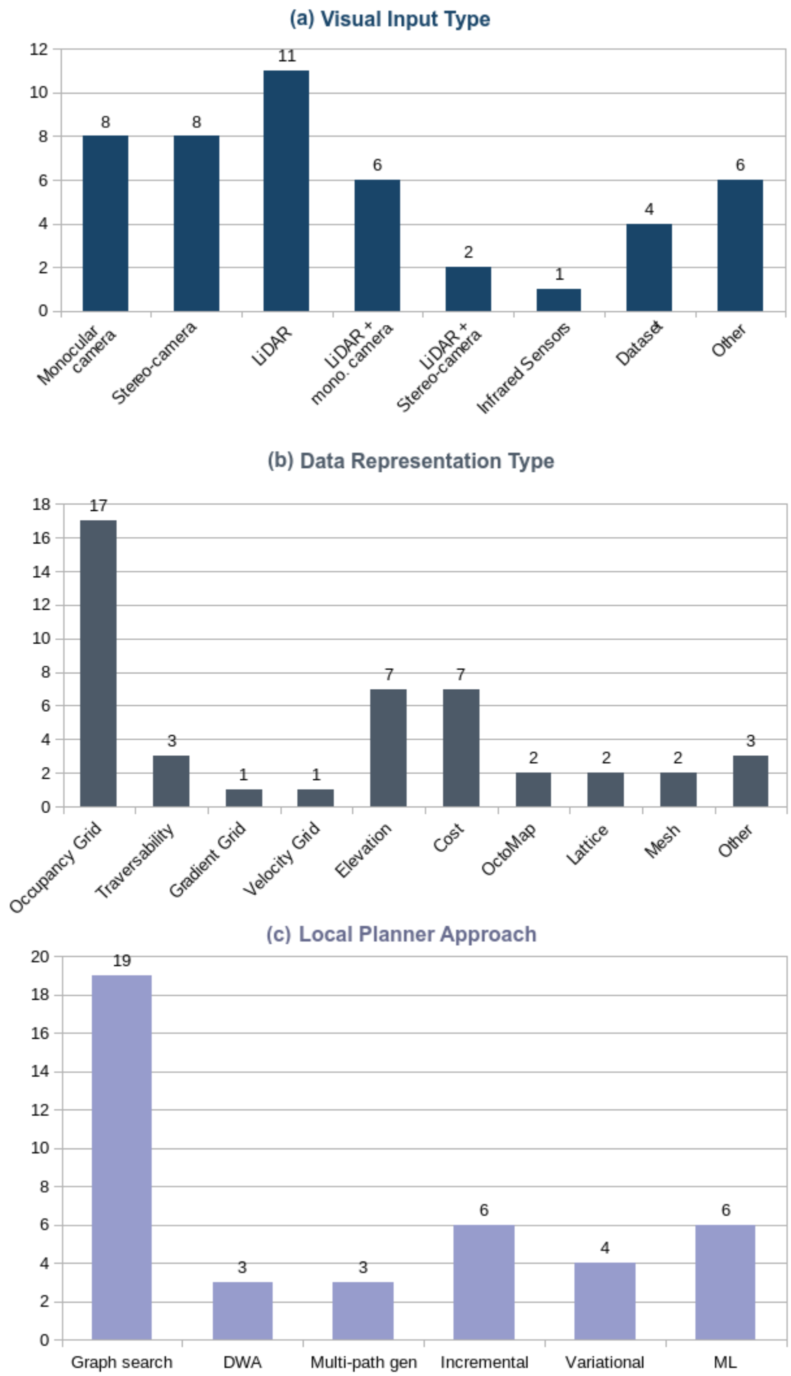
Machine Learning (ML) comprises techniques that use neural networks or evolutionary algorithms to learn how to plan. [31] proposes a training-based approach using a CNN to visually predict the future path that a vehicle will take, as an end-to-end solution. In [61] and [72], a solution based on Ant Colony bio-inspired behavior is presented. Virtual ants perform search tasks and mark a proposed path with pheromones. Optimal paths get stronger, for the value of added pheromones increases when ants choose the path, and more ants are attracted to it because of this. [54] proposes a motion planning approach for planetary rovers whereby the outcome of control actions is learned from experience using the Markov Decision Process (MDP). The model is trained using sample executions of motion primitives on the representative terrain, to predict the future outcome of control actions on similar terrain. [68] examines the utilization of prior information for speeding up the process of path planning. Using MOEA/D with random initialization and with graph-based approximation, computed with NAMOA*, showed that the latter option is more efficient. MOEA/D initialized with a sequence of previously solved problems stored in memory was found to be even more effective, but it is only appropriate when the planning problem is not evolving rapidly and the previous information has some bearing on the current path being computed, such as similar road shape or distribution. In [64], a global planner learns from a demonstration of human driver behavior to build a cost map for planning. The RANGER algorithm [35] is used as a local planner in combination with Field D* as a global planner to perform motion.

4.2.3 The Trajectory Controller (TC). The TC is responsible for planning or executing the next vehicle's state based on the Local Planner output and the current vehicle's state. Most works in this review do not present a Trajectory Controller. In that case, motion commands are issued by other modules, such as the LP itself.

In [12] the TC transforms a planned path into a set of line segments, representing the input for moving a virtual robot, which is treated as a path reference. The TC compares the real state of the vehicle with this reference to issue commands and collect new states, resulting in a feedback loop. Final movement is controlled by a model-reference adaptive controller (MRAC) and a proportional controller. The error between the ideal virtual path and the real path is measured and minimized it as much as possible. In [74], a reference path is also followed. A speed controller defines the velocity based on a perceived distance from a front obstacle, so the controller works with both a reference path from the path planner and a reference speed from the speed planner. [1] focus on reference path correction using a Distributed Control System (DCS) to achieve precision in navigation. Several different techniques are compared, and the one which has the best concordance rate of sensor readings and the ideal path database is selected as high priority to control the vehicle.

Table 2. Classification of local planning techniques by data representation type

	Graph search	Dynamic Window Approach	Multi-path generation and selection	Incremental search	Variational	Machine Learning
Occupancy Grid	[8, 15, 16, 21, 22, 33, 43, 44, 53, 60]	[1]	[28]	[12, 70]	[27, 45]	[31]
Traversability Map	[7]	[75]	-	-	-	[61]
Gradient Grid	[73]	-	-	-	-	-
Velocity Grid	-	-	-	[30]	-	-
Topological and Elevation Maps	[3, 10, 20]	[75]	-	[71]	-	[54, 68, 72]
CostMap	[44, 49]	[38]	[74]	[46]	-	[54, 64]
OctoMap	-	[38]	-	-	[4]	-
State Lattice	[67]	-	-	[62]	-	-
Mesh	[24]	-	[22]	-	-	-
Other	[34, 66]	-	-	-	[26]	-



[H]

Fig. 4. Distribution of the techniques by layer.

5 DISCUSSION

Following the description of the contributions, we now focus on the discussion of the observations that we were able to make during this review.

Regarding Visual input type, LiDAR was found to be the most commonly adopted device for self-driving cars, followed by monocular cameras and stereo cameras. 45% of the papers use LiDAR, as we can see in Figure 4(a), either alone or combined with cameras. A combination of LiDAR and monocular cameras is also more frequent than a combination of LiDAR and stereo cameras.

In accordance with the proposed architecture, visual input data is stored in Data Representation structures, which are used by the local planner to compute a feasible local path. As we can see in Figure 4(b), the most common data representation type is the Occupancy Grid, accounting for 40% of the papers. Grid-based solutions such as Occupancy Grid, Elevation Maps, Traversability Maps and Cost-Maps, represent together 34 papers, or 80% of the selected sources, which indicates that grid-based approaches are the most common choice for self-driving vehicles in non-structured environments. This choice poses some advantages such as simplicity and the possibility of making them probabilistic and therefore able to account for sensor failures. Another advantage is that, due to their vectorial nature, they can be accelerated using GPUs, such as in [18]. A disadvantage of adopting this approach is high memory consumption, although some works, such as [19] propose new methods for reducing memory usage.

Local planning in this review is mostly performed using a graph search approach. 19 papers have adopted it, accounting for 45% of the works reviewed, as we can see in Figure 4(c). It is most frequently combined with OG as data representation, accounting for 10 papers, or 52% of them. Particularly, 7 papers combined a monocular or stereo-camera as an input source and an OG as a data representation. From those, 6 performed a graph search approach as local planning, accounting for 85%, which seems to indicate a preference.

When we cross-compare data representation, visual input type, and local planning choices, we hardly find preferences for association. The association between monocular cameras and OG is most common, but the difference from the others is narrow. 5 papers associate monocular cameras with OG, whereas OG and LiDAR are found in 4 papers, stereo-cameras and Cost-Map in 3 papers, and LiDAR + monocular cameras and Cost-Map also in 3 papers.

When grouping together Occupancy Grid and its variations, we find that papers adopting monocular cameras prefer using a graph search approach as the local planner, comprising 6 out of 8 papers. Papers using stereo-cameras show no preference, varying from graph search, incremental search, machine learning and DWA. The most chosen approach for papers using LiDAR is graph search, but only 4 out of 11 have adopted it. The most popular approach for papers combining monocular cameras and LiDAR was Machine Learning, but only 2 out of 6 have adopted it. The results are in table 3.

Table 3. The most adopted local planning approach for visual input type

	Local planner approach	% clear preference?
Monocular camera	Graph search	yes, 75%
Stereo camera	no preference	-
LiDAR	Graph search	no, 36%
Monocular camera and LiDAR	Machine Learning	no, 33%

Another aspect worth investigating is a possible evolution in approach selection over the years. Thus, the papers were also divided into four categories:

- Symbolic Artificial Intelligence and Discrete Mathematics, comprising papers that use discrete mathematics or classic symbolic artificial intelligence, such as graph search, optimizing control problems etc.
- Evolutionary Computing, including papers that are based on evolutionary algorithms for learning or adaptation.
- Neural Networks and Deep Learning, with papers which the planning is based on any form of neural networks, such as imitation learning.
- Hybrid, comprising any work that combines more than one solution approach to plan the trajectory.

A total of 76% of the works use Symbolic Artificial Intelligence and Discrete Mathematics (Symbolic AI) as search approaches for planning algorithms, while 7% use Evolutionary Computing, 5% use Neural Networks and Deep Learning and 12% use a Hybrid form of planning. Thus, only 19% of the solutions use some sort of machine learning approach. However, when we look at the last 5 years, this rate increases to 37% for Neural Networks or Hybrid approaches, a growth of about 95%. The results are in table 4.

Table 4. Classification of local planning by approach category

	Papers	% adoption (all time)	% adoption (last 5 years)
Symbolic Artificial Intelligence and Discrete Mathematics	[1, 7–10, 12, 13, 15, 16, 20–22, 26–28, 30, 33, 34, 38, 43, 44, 46, 49, 53, 54, 60, 62, 64, 66, 71, 73, 74]	76%	63%
Evolutionary Computing	[61, 68, 72]	7%	-
Neural Networks and Deep Learning	[3, 31]	5%	11%
Hybrid	[24, 45, 67, 70, 75]	12%	26%

Global planning approaches were much less commonly mentioned in this review, as most of the papers focused on solving the local planning task. Nine papers described any form of long-term planning. In [8, 38, 64], a graph search approach was adopted based on variations of D*, while [12] has chosen RRT. In those cases, the global path is constructed by the vehicle, which uses some of its computing power to perform the planning. It is unclear how these solutions would perform in a low-structured scenario where an unpaved road exists and should be followed.

5.1 Identified Risks and Solutions

Planning and correctly following a local trajectory in off-road scenarios often present many problems due to terrain variations or sensor faults. Those problems can vary according to the path planning approach, the chosen vision type, or special terrain characteristics. In this section, we highlight some interesting risks and problems, along with their proposed solutions, aiming to present practical ways of dealing with them.

5.1.1 *The planner chooses valid but less traversable waypoints.*

This problem can appear when the vehicle chooses a feasible path, which could even be seen as optimal from the point of view of the local planner, but in practical terms, a better choice should be made to increase riding comfort or safety.

More data is usually required to support a better planning decision, such as class of segmentation, height, detected terrain type, or a better cost function that takes many sensed environment variables as input and the use of DR that takes terrain cost into account, such as Cost-Map or Traversability Map. Accounting for terrain heights when computing better paths can also improve local planning. [75] use Traversabily Map and Neural Networks to learn how to estimate risks from past driving missions. [9] seeks to mitigate segmentation uncertainty by deriving a closed-form efficiently computable lower bound for the Shannon mutual information between a multi-class occupancy map and a set of range-category measurements, allowing planning to re-evaluate a path with a better trade-off between uncertainty reduction and efficient exploration. [38] uses OctoMap, where each cell contains the probability of being occupied or free. A 50% chance means unknown state, while anything equals or lower than 12% represents a known free space, whereas anything equal to or above 97% represents a known occupied space. No subsequent measurement can abruptly change those values, so faulty image segmentation will most likely not result in a sudden path change. [54] use a Gaussian Process regression to map previous driving missions. The model is trained using sample executions of motion primitives on representative terrain, and it predicts the future outcome of control actions on similar terrain, thus helping the vehicle make better decisions over time. [53] incorporates terrain characteristics to get a more realistic cost function result for its Hybrid A* graph search local planner, such as needed energy for climbing hills, needed steering energy, and surface quality along with the driven distance. [73] takes terrain height into account by using a Gradient Grid and computing optimal paths using cost field.

5.1.2 *Mismatch between the path a vehicle should be driving on and the one it is currently following.*

This issue can happen due to loose terrain, sensor faults, and other instabilities. [70], [74] and [12] adopt a form of error correction to try to ensure that the vehicle is following the planned path. A reference path is used in [70], which is considered a virtual ideal path, often the result of the planning process. It can then be compared with the current position to possibly feed a trajectory correction back to the planner.

An inability to model the vehicle's response when translating issued movement commands into motion due to terrain variations or high model complexity can also be an issue. In this case, the vehicle's behavior is modeled, but the model can't predict every interaction with the terrain. [64] adopts a model simplification, which expects the motion to follow well defined patterns, and a self-learning mechanism to perform behavior correction based on imitation learning.

5.1.3 *Unreliability of visual data.*

Visual input data is one of the most important environment data used in the local planning decision process. Depending on the source of visual input, problems such as faulty image segmentation, sensor failure, or wrong distance estimation can happen unexpectedly. In order to avoid a faulty image frame ruining the vehicle's perception of the environment, it is possible to use probability to mark some regions as having obstacles. [45] uses an occupancy grid implementing an OG risk map divided into two layers: static risk, to hold terrain's cost, and dynamic risk, to hold uncertainty's cost, such as moving objects. [3] uses a traversability map to compute the cost of moving while employing machine learning to learn from past driving missions.

Works that use stereo-cameras are susceptible to errors under strong light. To compensate this, [46] uses a Semi-Global Matching 3D representation [29], which uses a pixelwise, Mutual Information based matching cost for compensating radiometric differences in input images.

5.1.4 *The planned set of waypoints is feasible, but not smooth enough.*

We identified post-processing solutions adopted after a selection for an optimal path is performed. In [27], the path is optimized using Douglas-Peucker DP to decimate the path into smaller points by a point distance factor ϵ. In [33] splines are used for path planning, while in [70] they are applied to fit the reference path to the target points. [66] applies post-processing to smooth the path generated by the Hybrid A* planner using Catmull-Rom interpolation. [15] also uses post-processing, applying a non-linear conjugated gradient descent optimization to eliminate path discontinuity. [33] use splines to obtain a smooth Dijkstra-computed trajectory. The splines vary according to their class, which is chosen based on the current speed. The faster the robot runs, the less curvy the resulting path from the local planner must be. [13] adopts post-processing for path smoothing via Conjugate Gradient.

5.1.5 *Local planning is not fit for terrain type variations in low-structured environments.*

A vehicle self-driving under low-structured environments needs to adapt to different terrain environment, such as country road, woodland road, slope road, wading road and so on. [44] proposes a deterministic finite automaton to change the planning, based on environment changes, such as detection of uphill or downhill, detection of road or slope road. Each mode has a special planner to adapt to the terrain's particular characteristics. In [1] a Distributed Control System performs the path tracking decision using several parallel modules that follow a hierarchy of priority based on the concordance rate of the sensor's readings and the database. The module with the highest concordance rate is the one selected to control navigation over the next stretch of the planned path.

To cope with obstacles' and road damage's heights, a 3D representation of the environment is generally used, which usually requires a LiDAR or a stereo camera. Traversability is used to simplify the cost decision in [7] and [61]. [75] fuses stereo camera data and LiDAR to obtain a better terrain height, then a probabilistic transformation using Bayes is applied to the output of the semantic probabilistic traversable grid map, which accounts for height variations. [10] uses wave peak of polar histogram to build a Topological Map to account for terrain height while computing the path using A*.

6 CONCLUSION

This review presents the state-of-the-art in path or trajectory planning for car-like vehicles driving in low-structured or non-structured environments over the last 12 years using a systematic methodology. We have adopted a three-layer division to classify self-driving features and project decisions by type of visual input, data representation and trajectory planning. From a computer vision perspective, we've identified that LiDAR is the most common visual input choice, closely followed by monocular cameras and stereo-cameras. LiDARs are also associated with cameras to improve terrain topology sensing. Data from visual input is most commonly stored and processed using Occupancy Grid (OG) and variations of OG such as Elevation Maps and Traversability Maps.

From a trajectory planning perspective, we've found that the majority of the papers use a form of graph search. Since we've opted to look at an extensive time frame, from 2010 to 2022, it was possible to notice a change in approach type, from a Symbolic Artificial Intelligence and Discrete Mathematics to a more prominent use of Deep Learning in trajectory computing. Nonetheless, future systematic reviews should be performed to confirm this.

Lastly, we've highlighted a series of risks and proposed solutions found in this review that we have judged interesting, aiming to present practical ways of dealing with uncertainties associated with the challenging scenario that a self-driven vehicle must face when performing a mission in a low- or non-structured environment.

7 DECLARATIONS

7.1 Compliance with Ethical Standards

This manuscript is original and was not submitted or published elsewhere in any form or language. The study was conducted impartially without any financial, personal, or professional relationships that could have biased the findings. The authors also affirm no conflicts of interest that could have influenced the publication, interpretation of results, or conclusions of this research.

7.2 Consent for publication

All authors have approved the manuscript and agree with its submission.

7.3 Funding

The authors declare that no funding was received for conducting this review.

7.4 Availability of data and material

The authors state that all pictures and images included in this review either belong to the authors and were previously made available in the RTK Dataset, cited in the manuscript and publicly published on our site and also on Mendeley, or were produced specifically for this review. There are no copyright conflicts.

7.5 Acknowledgments

The authors would like to acknowledge Nathalie Ferreira for helping with some of the illustrations present in this work.

7.6 Author contribution

Cristiano Souza de Oliveira wrote this paper. Aldo von Wangenheim helped conceptualize the idea, design the methodology, supervised and edited the writing and provided the RTK Dataset. All authors reviewed the results and approved the final version of the manuscript.

7.7 Conflict of interest

The authors have no conflicts of interest to declare.

REFERENCES

[1] Naoki Akai, Sam Ann Rahok, Kazumichi Inoue, and Koichi Ozaki. 2014. Development of magnetic navigation method based on distributed control system using magnetic and geometric landmarks. *ROBOMECH Journal* 1, 1 (21 11 2014), 21. https://doi.org/10.1186/s40648-014-0021-8

[2] Szilárd Aradi. 2020. Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles. (01 2020).

[3] Paolo Arena, Luca Patanè, and Salvatore Taffara. 2021. Learning risk-mediated traversability maps in unstructured terrains navigation through robot-oriented models. *Information Sciences* 576 (2021), 1–23. https://doi.org/10.1016/j.ins.2021.06.007

[4] Arash Asgharivaskasi and Nikolay Atanasov. 2021. Active Bayesian Multi-class Mapping from Range and Semantic Segmentation Observations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 1–7. https://doi.org/10.1109/ICRA48506.2021.9561711

[5] Claudine Badue, Ranik Guidolini, Raphael Carneiro, Pedro Azevedo, Vinicius Cardoso, Avelino Forechi, Luan Ferreira Reis de Jesus, Rodrigo Berriel, Thiago Paixão, Filipe Mutz, Thiago Oliveira-Santos, and Alberto De Souza. 2019. Self-Driving Cars: A Survey. (01 2019).

[6] Jorge Biolchini, Paula Mian, Ana Candida, and Cruz Natali. 2005. Systematic Review in Software Engineering. (01 2005).

[7] T. Braun. 2011. Cost-Efficient Global Robot Navigation in Rugged Off-Road Terrain. *KI - Künstliche Intelligenz* 25, 2 (01 05 2011), 173–177. https://doi.org/10.1007/s13218-011-0088-9

[8] Konstantinos Charalampous, Ioannis Kostavelis, and Antonios Gasteratos. 2015. Thorough robot navigation based on SVM local planning. *Robotics and Autonomous Systems* 70 (2015), 166–180. https://doi.org/10.1016/j.robot.2015.02.010

[9] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Kenneth Goldberg, Pieter Abbeel, Nathan Michael, and Vijaya Kondepogu. 2015. Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. https://doi.org/10.15607/RSS.2015.XI.003

[10] Qingyang Chen, Zhenping Sun, Daxue Liu, Liu Cao, and Tongtong Chen. 2011. Integrating topological map for autonomous navigation in unstructured environments. In *2011 2nd International Conference on Intelligent Control and Information Processing*, Vol. 1. 185–189. https://doi.org/10.1109/ICICIP.2011.6008228

[11] Thomas Collins, J.J. Collins, and Donor Ryan. 2007. Occupancy grid mapping: An empirical evaluation. In *2007 Mediterranean Conference on Control & Automation*. 1–6. https://doi.org/10.1109/MED.2007.4433772

[12] Rodrigo W. S. M. de Oliveira, Ricardo Bauchspiess, Letícia H. S. Porto, Camila G. de Brito, Luis F. C. Figueredo, Geovany A. Borges, and Guilherme N. Ramos. 2020. A Robot Architecture for Outdoor Competitions. *Journal of Intelligent & Robotic Systems* 99, 3 (01 09 2020), 629–646. https://doi.org/10.1007/s10846-019-01140-9

[13] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. 2010. Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments. *The International Journal of Robotics Research* 29, 5 (2010), 485–501. https://doi.org/10.1177/0278364909359210 arXiv:https://doi.org/10.1177/0278364909359210

[14] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. 2010. Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments. *The International Journal of Robotics Research* 29, 5 (2010), 485–501. https://doi.org/10.1177/0278364909359210 arXiv:https://doi.org/10.1177/0278364909359210

[15] Xingxing Du, Xiaohui Li, Daxue Liu, and Bin Dai. 2016. Path planning for autonomous vehicles in complicated environments. In *2016 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. 1–7. https://doi.org/10.1109/ICVES.2016.7548170

[16] Dennis Fassbender, André Mueller, and Hans-Joachim Wuensche. 2014. Trajectory planning for car-like robots in unknown, unstructured environments. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3630–3635. https://doi.org/10.1109/IROS.2014.6943071

[17] Dave Ferguson and Anthony Stentz. 2007. Field D*: An Interpolation-Based Path Planner and Replanner. In *Robotics Research*, Sebastian Thrun, Rodney Brooks, and Hugh Durrant-Whyte (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 239–253.

[18] Jörg Fickenscher, Jens Schlumberger, Frank Hannig, Jürgen Teich, and Mohamed Essayed Bouzouraa. 2018. Cell-based update algorithm for occupancy grid maps and hybrid map for ADAS on embedded GPUs. In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 443–448. https://doi.org/10.23919/DATE.2018.8342050

[19] Alex Fisher, Ricardo Cannizzaro, Madeleine Cochrane, Chatura Nagahawatte, and Jennifer L. Palmer. 2021. ColMap: A memory-efficient occupancy grid mapping framework. *Robotics and Autonomous Systems* 142 (2021), 103755. https://doi.org/10.1016/j.robot.2021.103755

[20] Shuzhi Sam Ge, Qun Zhang, Aswin Thomas Abraham, and Brice Rebsamen. 2011. Simultaneous Path Planning and Topological Mapping (SP2ATM) for environment exploration and goal oriented navigation. *Robotics and Autonomous Systems* 59, 3 (2011), 228–242. https://doi.org/10.1016/j.robot.2010.12.003

[21] Christopher M. Gifford, Russell Webb, James Bley, Daniel Leung, Mark Calnon, Joseph Makarewicz, Bryan Banz, and Arvin Agah. 2010. A novel low-cost, limited-resource approach to autonomous multi-robot exploration and mapping. *Robotics and Autonomous Systems* 58, 2 (2010), 186–202. https://doi.org/10.1016/j.robot.2009.09.014 Selected papers from the 2007 European Conference on Mobile Robots (ECMR '07).

[22] David Gingras, Erick Dupus, Guy Payre, and Jean de Lafontaine. 2010. Path planning based on fluid mechanics for mobile robots using unstructured terrain models. In *2010 IEEE International Conference on Robotics and Automation*. 1978–1984. https://doi.org/10.1109/ROBOT.2010.5509679

[23] David Gingras, Tom Lamarche, Jean-Luc Bedwani, and Erick Dupuis. 2010. Rough Terrain Reconstruction for Rover Motion Planning. *CRV 2010 - 7th Canadian Conference on Computer and Robot Vision*, 191 – 198. https://doi.org/10.1109/CRV.2010.32

[24] Ueli Graf, Paulo Borges, Emili Hernández, Roland Siegwart, and Renaud Dubé. 2019. Optimization-Based Terrain Analysis and Path Planning in Unstructured Environments. In *2019 International Conference on Robotics and Automation (ICRA)*. 5614–5620. https://doi.org/10.1109/ICRA.2019.8794331

[25] Dario Calogero Guastella and Giovanni Muscato. 2021. Learning-Based Methods of Perception and Navigation for Ground Vehicles in Unstructured Environments: A Review. *Sensors* 21, 1 (2021). https://doi.org/10.3390/s21010073

[26] Yuqing Guo, Danya Yao, Bai Li, Haichuan Gao, and Li Li. 2023. Down-Sized Initialization for Optimization-Based Unstructured Trajectory Planning by Only Optimizing Critical Variables. *IEEE Transactions on Intelligent Vehicles* 8, 1 (2023), 709–720. https://doi.org/10.1109/TIV.2022.3163335

[27] Nina Felicitas Heide, Alexander Albrecht, Thomas Emter, and Janko Petereit. 2019. Performance Optimization of Autonomous Platforms in Unstructured Outdoor Environments Using a Novel Constrained Planning Approach. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. 2359–2364. https://doi.org/10.1109/IVS.2019.8813805

[28] Michael Himmelsbach, Thorsten Luettel, Falk Hecker, Felix von Hundelshausen, and Hans-Joachim Wuensche. 2011. Autonomous Off-Road Navigation for MuCAR-3. *KI - Künstliche Intelligenz* 25, 2 (01 05 2011), 145–149. https://doi.org/10.1007/s13218-011-0091-1

[29] Heiko Hirschmuller. 2008. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 2 (2008), 328–341. https://doi.org/10.1109/TPAMI.2007.1166

[30] Frank Hoeller, Timo Röhling, and Dirk Schulz. 2011. Offroad Navigation Using Adaptable Motion Patterns. *KI - Künstliche Intelligenz* 25, 2 (01 05 2011), 151–154. https://doi.org/10.1007/s13218-011-0099-6

[31] Christopher J. Holder and Toby P. Breckon. 2018. Learning to Drive: Using Visual Odometry to Bootstrap Deep Learning for Off-Road Path Prediction. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. 2104–2110. https://doi.org/10.1109/IVS.2018.8500526

[32] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. 2013. OctoMap: an efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots* 34, 3 (01 04 2013), 189–206. https://doi.org/10.1007/s10514-012-9321-0

[33] Marcel Häselich, Nikolay Handzhiyski, Christian Winkens, and Dietrich Paulus. 2011. Spline templates for fast path planning in unstructured environments. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3545–3550. https://doi.org/10.1109/IROS.2011.6094756

[34] Laura Jarin-Lipschitz, James Paulos, Raymond Bjorkman, and Vijay Kumar. 2021. Dispersion-Minimizing Motion Primitives for Search-Based Motion Planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 12625–12631. https://doi.org/10.1109/ICRA48506.2021.9561840

[35] Alonzo Kelly and Anthony Stentz. 1998. Rough Terrain Autonomous Mobility—Part 2: An Active Vision, Predictive Control Approach. *Autonomous Robots* 5, 2 (01 05 1998), 163–198. https://doi.org/10.1023/A:1008822205706

[36] Jennifer King and Maxim Likhachev. 2009. Efficient cost computation in cost map planning for non-circular robots. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3924–3930. https://doi.org/10.1109/IROS.2009.5354074

[37] Barbara Kitchenham. 2004. Procedures for Performing Systematic Reviews. *Keele, UK, Keele Univ.* 33 (08 2004).

[38] Rafael Luiz Klaser, Fernando Santos Osório, and Denis Wolf. 2014. Vision-Based Autonomous Navigation with a Probabilistic Occupancy Map on Unstructured Scenarios. In *2014 Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol*. 146–150. https://doi.org/10.1109/SBR.LARS.Robocontrol.2014.13

[39] I.S. Kweon and T. Kanade. 1992. High-resolution terrain map from multiple sensor data. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14, 2 (1992), 278–292. https://doi.org/10.1109/34.121795

[40] S. M. LaValle. 2006. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K. Available at http://planning.cs.uiuc.edu/.

[41] Jinhan Lee, Charles Pippin, and Tucker Balch. 2008. Cost based planning with RRT in outdoor environments. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 684–689. https://doi.org/10.1109/IROS.2008.4651052

[42] Jiankun Li and C.-C.J. Kuo. 1998. A dual graph approach to 3D triangular mesh compression. In *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*, Vol. 2. 891–894 vol.2. https://doi.org/10.1109/ICIP.1998.723699

[43] Ning Li, Xijun Zhao, Ruina Dang, and Bo Su. 2018. DF -FS based path planning algorithm with sparse waypoints in unstructured terrain. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 1783–1788. https://doi.org/10.1109/ROBIO.2018.8664797

[44] Ning Li, Xijun Zhao, Jianfeng Gao, and Xing Cui. 2017. DFA based autonomous decision-making for UGV in unstructured terrain. In *2017 IEEE International Conference on Unmanned Systems (ICUS)*. 34–39. https://doi.org/10.1109/ICUS.2017.8278313

[45] Zhiyuan Li, Pan Zhao, Chunmao Jiang, Weixin Huang, and Huawei Liang. 2022. A Learning-Based Model Predictive Trajectory Planning Controller for Automated Driving in Unstructured Dynamic Environments. *IEEE Transactions on Vehicular Technology* 71, 6 (2022), 5944–5959. https://doi.org/10.1109/TVT.2022.3159994

[46] Sho Matsunaga, Yuichi Kobayashi, Kazuki Matsumaura, and Chyon Hae Kim. 2016. Motion planning of mobile robot considering velocity-dependent cost and time. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 619–624. https://doi.org/10.1109/ROBIO.2016.7866391

[47] Jefferson Menegazzo and Aldo von Wangenheim. 2021. Road surface type classification based on inertial sensors and machine learning. *Computing* 03 (2021), 2143–2170. https://doi.org/10.1007/s00607-021-00914-0

[48] Jefferson Menegazzo and Aldo von Wangenheim. 2023. Speed Bump Detection Through Inertial Sensors and Deep Learning in a Multi-contextual Analysis. *SN Comp.Sci.* 4, 18 (2023). https://doi.org/10.1007/s42979-022-01438-w

[49] Karan Narula, Stewart Worrall, and Eduardo Nebot. 2020. Two-Level Hierarchical Planning in a Known Semi-Structured Environment. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. 1–6. https://doi.org/10.1109/ITSC45102.2020.9294373

[50] Jianjun Ni, Yinan Chen, Yan Chen, Jinxiu Zhu, Deena Ali, and Weidong Cao. 2020. A Survey on Theories and Applications for Self-Driving Cars Based on Deep Learning Methods. *Applied Sciences* 10, 8 (2020). https://doi.org/10.3390/app10082749

[51] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. 2016. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles* 1 (04 2016). https://doi.org/10.1109/TIV.2016.2578706

[52] Panagiotis Papadakis. 2013. Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence* 26, 4 (2013), 1373–1385. https://doi.org/10.1016/j.engappai.2013.01.006

[53] Janko Petereit, Thomas Emter, Christian W. Frey, Thomas Kopfstedt, and Andreas Beutel. 2012. Application of Hybrid A* to an Autonomous Mobile Robot for Path Planning in Unstructured Outdoor Environments. In *ROBOTIK 2012; 7th German Conference on Robotics*. 1–6.

[54] Thierry Peynot, Sin-Ting Lui, Rowan McAllister, Robert Fitch, and Salah Sukkarieh. 2014. Learned Stochastic Mobility Prediction for Planning with Control Uncertainty on Unstructured Terrain. *Journal of Field Robotics* 31, 6 (2014), 969–995. https://doi.org/10.1002/rob.21536 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21536

[55] Mikhail Pivtoraiko and Alonzo Kelly. 2005. Efficient constrained path planning via search in state lattices. In *Proceedings of 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS '05)*.

[56] Sam Rahok, Kazumichi Inoue, and Koichi Ozaki. 2012. Development of a Mobile Robot to Run in Tsukuba Challenge 2010. *Advanced Robotics* 26 (09 2012), 1555–1575. https://doi.org/10.1080/01691864.2012.694631

[57] Thiago Rateke, Karla Justen, Aldo von Wangenheim, and Rafael Toledo. 2023. Road Traversing Knowledge for Surface Classification. https://doi.org/10.17632/fxy5khmhpb.1

[58] Thiago Rateke, Karla Aparecida Justen, and Aldo von Wangenheim. 2019. Road Surface Classification with Images Captured From Low-cost Camera - Road Traversing Knowledge (RTK) Dataset. *Revista de Informática Teórica e Aplicada* 26, 3 (Nov. 2019), 50–64. https://doi.org/10.22456/2175-2745.91522

[59] Thiago Rateke and Aldo von Wangenheim. 2021. Road surface detection and differentiation considering surface damages. *Autonomous Robots* 45, 2 (01 Feb 2021), 299–312. https://doi.org/10.1007/s10514-020-09964-3

[60] Francisco Rovira-Más, John Reid, and Quanyi Zhang. 2006. Stereovision Data Processing With 3 d Density Maps For Agricultural Vehicles. *Transactions of the ASABE* 49 (07 2006). https://doi.org/10.13031/2013.21721

[61] Pedro Santana and Luís Correia. 2011. Swarm cognition on off-road autonomous robots. *Swarm Intelligence* 5, 1 (01 03 2011), 45–72. https://doi.org/10.1007/s11721-010-0051-7

[62] Neal Seegmiller, Jason Gassaway, Elliot Johnson, and Jerry Towler. 2017. The Maverick planner: An efficient hierarchical planner for autonomous vehicles in unstructured environments. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018–2023. https://doi.org/10.1109/IROS.2017.8206021

[63] Chao Shen, Xiangmo Zhao, Xing Fan, Xinyu Lian, Fan Zhang, Abdul Rahman Kreidieh, and Zhanwen Liu. 2019. Multi-receptive field graph convolutional neural networks for pedestrian detection. *IET Intelligent Transport Systems* 13, 9 (2019), 1319–1328. https://doi.org/10.1049/iet-its.2018.5618 arXiv:https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-its.2018.5618

[64] David Silver, J. Andrew Bagnell, and Anthony Stentz. 2010. Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain. *The International Journal of Robotics Research* 29, 12 (2010), 1565–1592. https://doi.org/10.1177/0278364910369715 arXiv:https://doi.org/10.1177/0278364910369715

[65] Cristiano Souza de Oliveira and Aldo Von Wangenheim. 2023. *Systematic Literature Review on Trajectory Planning for Autonomous Vehicular Navigation in Non-Structured or Semi- Structured Environments.* Technical Report. Federal University of Santa Catarina. https://doi.org/10.13140/RG.2.2.36265.88161

[66] Kangbin Tu, Shuaishuai Yang, Hao Zhang, and Zhuping Wang. 2019. Hybrid A* Based Motion Planning for Autonomous Vehicles in Unstructured Environment. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–4. https://doi.org/10.1109/ISCAS.2019.8702779

[67] Marco Visca, Arthur Bouton, Roger Powell, Yang Gao, and Saber Fallah. 2021. Conv1D Energy-Aware Path Planner for Mobile Robots in Unstructured Environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2279–2285. https://doi.org/10.1109/ICRA48506.2021.9560771

[68] Antony Waldock and David W. Corne. 2012. Exploiting Prior Information in Multi-objective Route Planning. In *Parallel Problem Solving from Nature - PPSN XII*, Carlos A. Coello Coello, Vincenzo Cutello, Kalyanmoy Deb, Stephanie Forrest, Giuseppe Nicosia, and Mario Pavone (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 11–21.

[69] Limin Wang, Sheng Guo, Weilin Huang, Yuanjun Xiong, and Yu Qiao. 2017. Knowledge Guided Disambiguation for Large-Scale Scene Classification With Multi-Resolution CNNs. *IEEE Trans Image Process* 26, 4 (Feb. 2017), 2055–2068.

[70] Shaobo Wang, Pan Zhao, Yanghao Zheng, Biao Yu, Weixin Huang, Hui Zhu, and Huawei Liang. 2019. Reference Path Correction for Autonomous Ground Vehicles Driving Over Rough Terrain. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. 2405–2410. https://doi.org/10.1109/IVS.2019.8813812

[71] Zhiwen Xian, Xiaofeng He, Junxiang Lian, Xiaoping Hu, and Lilian Zhang. 2017. A bionic autonomous navigation system by using polarization navigation sensor and stereo camera. *Autonomous Robots* 41, 5 (01 06 2017), 1107–1118. https://doi.org/10.1007/s10514-016-9596-7

[72] Qinkun Xiao, Yi Wang, Song Gao, and Jiantao Wang. 2013. 3D Path Planning Based on Elevation Model and Ant Colony Algorithm. In *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Vol. 1. 74–77. https://doi.org/10.1109/IHMSC.2013.25

[73] Yong Yu and Meiling Wang. 2012. The Path Planning Algorithm Research Based on Cost Field for Autonomous Vehicles. In *2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Vol. 2. 38–41. https://doi.org/10.1109/IHMSC.2012.105

[74] Kai Zhang, Mengyin Fu, Yi Yang, Songtian Shang, and Meiling Wang. 2017. An efficient decision and planning method for high speed autonomous driving in dynamic environment. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. 806–811. https://doi.org/10.1109/IVS.2017.7995815

[75] Yimo Zhao, Peilin Liu, Wuyang Xue, Ruihang Miao, Zheng Gong, and Rendong Ying. 2019. Semantic Probabilistic Traversable Map Generation For Robot Path Planning. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 2576–2582. https://doi.org/10.1109/ROBIO49542.2019.8961533

A QUERY SEARCH STRINGS

Table 5. Query strings and the selection result for each location

Location	Search String	Result	Selected
ACM Digital Library	Title:("unstructured" OR "offroad" OR "semi-structured" OR "non-urban") AND ("path planning" OR "motion planning" OR "trajectory planning" OR "self-driving" OR "autonomous") NOT "UAV" NOT "indoor" NOT "UVMS" NOT "water" NOT "submerged" NOT "SOCIAL" OR Abstract:("unstructured" OR "offroad" OR "semi-structured" OR "non-urban") AND ("path planning" OR "motion planning" OR "trajectory planning" OR "self-driving" OR "autonomous") NOT "UAV" NOT "indoor" NOT "UVMS" NOT "water" NOT "submerged" NOT "SOCIAL" OR Keyword:("unstructured" OR "offroad" OR "semi-structured" OR "non-urban") AND ("path planning" OR "motion planning" OR "trajectory planning" OR "self-driving" OR "autonomous") NOT "UAV" NOT "indoor" NOT "UVMS" NOT "water" NOT "submerged" NOT "SOCIAL"	596	4
IEEE Xplore	(("Document Title":"unstructured" OR "Document Title":"offroad" OR "Document Title":"semi-structured" OR "Document Title":"non-urban") AND ("Document Title":"path planning" OR "Document Title":"motion planning" OR "Document Title":"trajectory planning" OR "Document Title":"self-driving" OR "Document Title":"autonomous") OR Abstract:("Abstract":"unstructured" OR "Abstract":"offroad" OR "Abstract":"semi-structured" OR "Abstract":"non-urban") AND ("Abstract":"path planning" OR "Abstract":"motion planning" OR "Abstract":"trajectory planning" OR "Abstract":"self-driving" OR "Abstract":"autonomous") OR AuthorKeywords:("Author Keywords":"unstructured" OR "Author Keywords":"offroad" OR "Author Keywords":"semi-structured" OR "Author Keywords":"non-urban") AND ("Author Keywords":"path planning" OR "Author Keywords":"motion planning" OR "Author Keywords":"trajectory planning" OR "Author Keywords":"self-driving" OR "Author Keywords":"autonomous"))	61	24
Science Direct	(("unstructured" OR "offroad" OR "semi-structured" OR "non-urban") AND ("path planning" OR "motion planning" OR "trajectory planning" OR "self-driving" OR "autonomous"))	467	5
Springer Link	(("unstructured" OR "offroad" OR "semi-structured" OR "non-urban") AND ("path planning" OR "motion planning" OR "trajectory planning" OR "self-driving" OR "autonomous"))	278	7
Wiley	("path planning" OR "motion planning" OR "trajectory planning")	544	1
Total		1946	42

Received 21 Jul 2023