

Path Planning in Heterogenous Environments

A Combined Approach

Ajinkya Krishnakumar

Master Thesis Report



Path Planning in Heterogenous Environments

A Combined Approach

by

Ajinkya Krishnakumar

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday January 29, 2019.

Student number: 4632974

Thesis committee:	Dr J. Alonso Mora,	TU Delft, supervisor
	Dr. Ir. Riender Hapee.	TU Delft
	L. Ferranti,	TU Delft

An electronic version of this thesis is available at <http://repository.tudelft.nl/>.

Acknowledgement

With this thesis i would be ending my studies at TU Delft. The past year has been very challenging as well as a learning experience for me. It wouldn't have been possible without the help and guidance from a lot of people.

I would first like to thank my thesis advisor Dr.J Alonso Mora from Cognitive robotics at TU Delft. The door to his office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this thesis to be my own work, but steered me in the right the direction whenever he thought I needed it. I would also like to thank Dr. Riender Happee and Laura Ferranti for taking time out of their schedule to be part of my thesis committee.

I would also like to acknowledge Jory Denny of the computer science department at University of Richmond and Ewing Kwang for all the help in regards to the algorithms. Without their valuable input, coding the dependencies in this thesis would have been impossible.

Finally, I must express my very profound gratitude to my parents and to my girlfriend Elaine for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

*Ajinkya Krishnakumar
Delft, Thursday 24th January, 2019*

Abstract

Autonomous vehicles are the inevitable future of the industry as theoretically they guarantee higher road throughput and a much safer means of transport compared to today's ground vehicle. This has attracted the industries and universities making it a very important topic of research. The basic function of the autonomous vehicle boils down to transporting its passenger safely from door to door. This requires planning of a path that is obstacle free. Currently, sampling-based methods are widely used for path planning. Although these methods are proving to be successful in open environments, they are inefficient in heterogeneous environments. Planning in urban environments would be successful if this obstacle is tackled. In the current literature, it was found that there are many methods which focus on solving the path planning problem while planning around the obstruction, while there are other methods that focus on converging to optimal solutions.

Hence, there is a need for methods that would plan optimal paths in heterogeneous environments. This thesis introduced a combined approach that shares elements of planning paths around obstacles and optimal path planning which are provided by the algorithms Adaptive RRT and Informed RRT* respectively. These methods are combined along with a Dubin's motion model to simulate basic vehicle's constraints.

The proposed approach was compared with state of the art methods like RRT* to evaluate its performance via simulation. The simulation was carried out in three different scenarios with variable complexity depending upon the available free configuration space. The ability to find a solution and converge it were evaluated, an improvement of 50% was noticed in finding the initial solution and around 25% improvement was seen in convergence. This concluded that such a hybrid approach could be an important contribution to urban path planning.

Contents

List of Algorithms	v
List of Figures	ix
List of Tables	xi
1 Introduction	1
1.1 Objectives.	2
1.2 Scope	2
1.3 Outline	3
2 Background	5
2.1 Preliminaries	5
2.1.1 State	5
2.1.2 Complexity	6
2.2 General Path Planning Problem.	6
2.3 Models used in path planning.	7
2.3.1 Geometric Model	8
2.3.2 Kinematic Model	8
2.3.3 Dynamic Model	8
2.4 Path Planning Algorithms	8
2.4.1 Variational Methods	10
2.4.2 Graph search Methods	10
2.4.3 Sampling Based Methods	11
2.5 State of the art	11
2.6 Discussion	13
3 Methodology	15
3.1 Introduction	15
3.2 Proposed Approach	15
3.2.1 Dubin's Motion Model	17
3.2.2 Adaptive RRT	19
3.2.3 Selection of Growth strategies	27
3.2.4 Informed RRT*.	27
3.3 Discussion	30
4 Simulations and Results	33
4.1 Simulation setup	33
4.1.1 Simulation Parameters.	33
4.1.2 Scenario selection.	33
4.1.3 Fine tuning for growth strategy selection	35

4.2	Results.	37
4.2.1	Initial solution performance	37
4.2.2	Convergence Performance	38
4.3	Discussion	40
5	Conclusion and Future work	43
5.1	Conclusions	43
5.2	Future work.	44
	Bibliography	47

List of Figures

1.1	Valid path	2
1.2	Invalid path	2
1.3	Path with sampled nodes	2
1.4	Path Planning for a vehicle like model [1]	3
2.1	Vehicle Kinematic Model	9
2.2	Dynamic Vehicle Model	9
3.1	Constituents of the proposed approach	17
3.2	Types of Dubin Curves	18
3.3	Possible Tangents	19
3.4	Collision check in the proposed approach	20
3.5	Distinct outcomes during expansion in RRT, fig 3.5a Null expansion, fig 3.5b Partial expansion, fig 3.5c Full expansion with C_{free} and C_{obs} being the free configuration space and the obstacle space respectively	22
3.6	Triangular mesh on a 3D hollow pipe	23
3.7	Triangular mesh with obstacle vectors of a 2D obstacle	23
3.8	Basic Extension	24
3.9	Random position with same orientation	24
3.10	Random obstacle vector with random orientation	25
3.11	Random obstacle vector with same orientation	25
3.12	Trace obstacle with random orientation	26
3.13	Trace obstacle with same orientation	26
3.14	Trace obstacle boundary	27
3.15	Progress of solution Informed RRT* adapted from [2]	28
3.16	Informed ellipsoid set	29
3.17	Combined Approached introduced in this Thesis	31
4.1	Scenario 1 map	34
4.2	Scenario 2 map	34
4.3	Scenario 3 map	35
4.4	Available configuration space in each scenario	36
4.5	Basic Visibility Map	36
4.6	Relative performance of the algorithms for the number of samples used to find solution	38
4.7	Relative performance of the algorithms for the time taken to find initial solution	38
4.8	Sample convergence data from scenario 1 simulations	39
4.9	Sample convergence data from scenario 2 simulations	39
4.10	Sample convergence data from scenario 3 simulations	40
4.11	Comparison of rate of convergence	41

List of Tables

2.1	Contributions towards urban path planning	14
4.1	Parameters used in the simulation	33
4.2	Growth Strategy distribution over the visibility range	37

List of Algorithms

1	RRT*	12
2	Proposed Approach	16
3	Adaptive RRT	20
4	Evaluation of visibility for nodes	21
5	Informed RRT*	28

Introduction

Over the past few years, the automobile industry and universities have had a growing affinity towards autonomous vehicles and Advanced Driver Assistance Systems (ADAS). Assisted by the introduction of Tesla Autopilot in 2014 and Audi's Traffic Jam Pilot in 2017 the industry is progressing towards fully autonomous vehicles. The lead analyst at Morgan Stanley, Ravi Shanker predicts that autonomous vehicles could save about 5.6 trillion dollars globally [3], all of this savings would be from accident prevention, congestion reduction and the driver's capability of working while being transported. Given the economic impact, it is evident that autonomous vehicles have had a long history as its research began in early 1920s, when Francis P Houdina professed a radio-controlled driver-less car that followed another car[4]. In 1980 EUREKA launched the Prometheus project collegial with six European countries in effort to work on vehicles with highest efficiency and unprecedented safety, but things really took a big turn during 2004-2005 DARPA grand challenge which encouraged universities to take part in the event where the autonomous vehicle had to complete a given course, During 2005 five of the teams were successful in completing the challenge [5]. By 2010 all the major automobile manufacturers began working on achieving a fully automated vehicle. With technological prerequisites becoming available and public admission to it, the vision of fully autonomous vehicles will come to realization.

As mentioned before such automation is only possible with on-board computers and electronics that will help the vehicle to act as an intelligent agent that can make sensible decisions in different scenarios. The research has also been extended to Vehicular Adhoc Networks(VANETs) [6] which is similar to a Mobile adhoc where the nodes are the autonomous vehicles. These help with more efficient route planning, traffic alert dissemination and mobile vehicular cloud services. For any automated vehicle with great technology and infrastructure the crucial problem that it needs to tackle is:

How to reach the goal safely?

In order to get to the destination safely , the vehicle must be able to plan a safe route while considering variables like the environment, current state of the vehicle, road geometry, traffic rules, other vehicles, vulnerable road users among others which is path planning. It plays an essential role in the automation of vehicles, and would be the topic of research in this thesis.

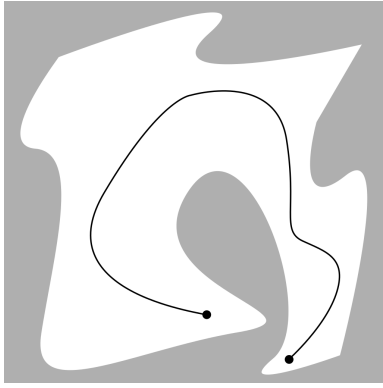


Figure 1.1: Valid path

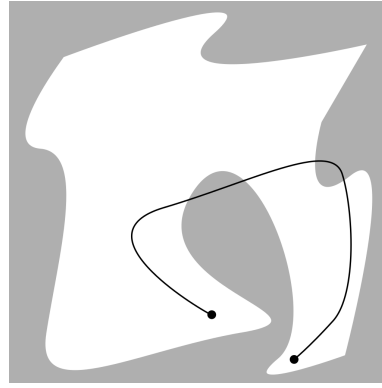


Figure 1.2: Invalid path

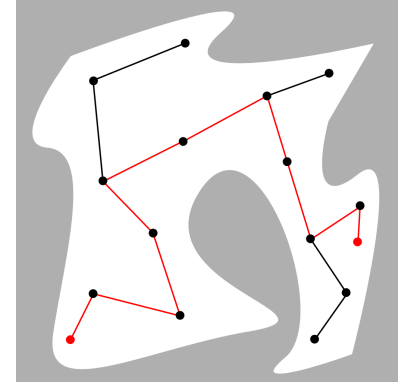


Figure 1.3: Path with sampled nodes

1.1. Objectives

For making fully autonomous vehicles a reality, it is necessary to tackle one of the most difficult problems in path planning i.e. planning in urban environments due to the presence of narrow passages. The presented work aims to study and evaluate the state of the art path planning methods and propose a combined approach that will outperform the conventional methods.

In an urban environment it is very important to find solutions quickly as well as effective. Most of the methods used currently focus on intelligent sampling around the obstacles or purely focus on the convergence of the initial solution and perform planning on holonomic agents. Consequently, the objectives of this thesis is to produce an algorithm with the following elements:

Sampling efficiency: Uniform random sampling in narrow passages reduces the probability of a node selected to grow, which causes the algorithm to generate more samples to generate a sample that can be successfully connected. Hence the research should provide an efficient sampling strategy that would require fewer samples to solve the path planning query presented.

Convergence: Finding the shortest and locally collision free path between the start and the goal via reiteration of the path planning algorithm.

Use of a realizable model: It is a challenge to find a path between two nodes depending on the agent as it increases the complexity, hence it would be important to implement a vehicle model along with incorporation of its pose

The end product would be a combination of the objectives mentioned above i.e., **a method that would focus on optimal path planning in narrow environments.**

1.2. Scope

In this thesis the agent is assumed to have almost ideal sensing with fixed tolerances to localize itself i.e the vehicle and obstacle boundary obtained is accurate. So, object detection isn't in the scope of this work. A path following controller is not considered and an ideal path follower is considered so the controlling criteria also wouldn't be in the scope of this work. The work circles around the software architecture that generates a path with the help

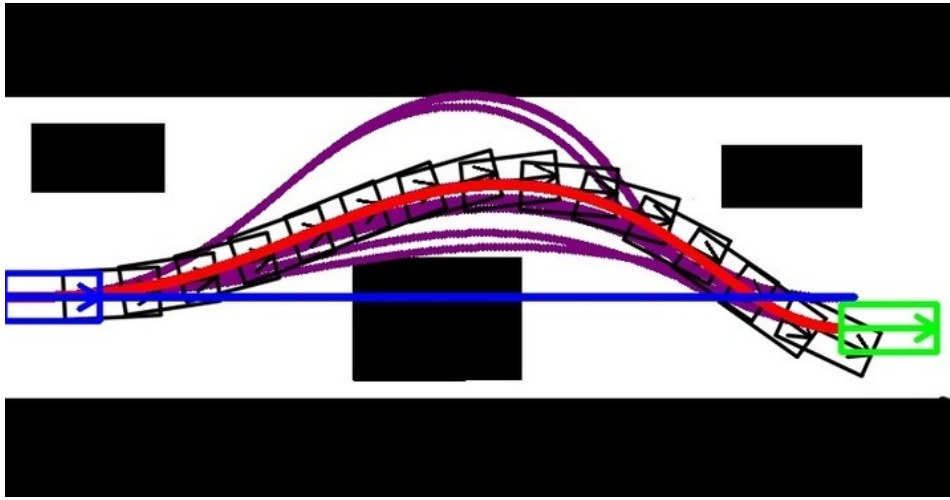


Figure 1.4: Path Planning for a vehicle like model [1]

of a path planning algorithm. The solution that is produced is and can be only used for simulation purposes in static environment conditions with an agent having known motion curves. The evaluation of the work in this thesis is carried out by relative comparison to compensate for unstable environment and non-optimized software.

1.3. Outline

To understand the essence of the work it is important for the readers to understand the basics of path planning which is given by chapter 2 . It consists of general terminology used in path planning along with the formulation of the path planning problem. It also discusses the literature for the different models used in path planning like geometric, kinematic and dynamic models and also discuss algorithms that have contributed towards urban path planning like RRT, RRT* etc.

From the contributions of the methods discussed, two methods are selected viz Adaptive RRT for its ability to adapt growth around narrow and heterogeneous obstacles and Informed RRT* for its capability to approach local optimal at a faster rate compared to other methods. Along with these, Dubins car model is selected as the motion model to generate a path between two nodes. Chapter 3 gives detailed description of the selected methods along with the way they are combined to arrive at the method proposed in this thesis. It starts with the discussion about the motion model that has been used to connect two states and moves into describing the constituent algorithms of the proposed combined approach. After which the combined approach is discussed and represented using a detailed flow chart.

In order to prove the identified concept there is a need for evaluating it against existing algorithms this is done in chapter 4 of this report. This chapter discusses the evaluation of the proposed method along with the baseline methods RRT* and Informed RRT*, it introduces the simulation setup and parameters, and analyses various scenarios in which the simulation was conducted. Then it discusses the tests that were conducted and presents the results relatively. Finally, chapter 5 summarizes the outcome of the result and conveys

the use of the combined approach that has been proposed in this thesis by highlighting its contributions. It then recommends readers about the future prospects in the same area of research.

2

Background

Firstly, this chapter introduces the path planning with its most general concepts. After which the general path planning problem is formulated mathematically. After this, the solutions to the path planning problem are described, highlighting the important differences between them and different motion models being used. After which the state of the art methods are explained along with methods that attempt to solve path planning in narrow environments. The chapter then concludes with some of the contribution the commonly used planning methods make and their favourability to be selected for the combined approach discussed later in this report.

2.1. Preliminaries

Definition 1 *Configuration space or C-space refers to all set of positions and orientations of the agent. The configuration space in this thesis considers the position x,y in 2D space along with the orientation in that plane which can be defined as:*

$$C = R^2 \times SO(1) \quad (2.1)$$

Definition 2 *A robot is holonomic if it can move in any direction in the configuration space. In classical mechanics a system may be defined as holonomic if all constraints of the system are holonomic. For a constraint to be holonomic it must be expressible as a function:*

$$f(q_1, q_2, \dots, q_n, t) = 0 \quad (2.2)$$

i.e. a holonomic constraint depends only on the coordinates $q(x,y)$ and time t . It does not depend on the velocities or any higher order derivative with respect to time. A system that does not satisfy these constraints is non holonomic.

2.1.1. State

Path planning involves a state space which informs the situation of the robot. For example, the state of a vehicle could represent the orientation and position. States could be described continuously or discretely. A path planning algorithm would always imply the state space. In most cases the state is infinitely large and cannot be represented explicitly, it would mostly be represented by the desired bounds of the application. Hence, state is an

important component in the formulation of a path planning problem and the algorithms used to solve it. In the case of this thesis the state space would involve the position and the orientation of the vehicle alone.

2.1.2. Complexity

The speed at which any problem's solution can be found is dependent on how complex the problem is. The theory of computational complexity have classified problems dependent on their probabilistic complexity, some of the important classes are:

P - This class stands for the problems whose solutions are efficiently computable in polynomial time.

NP - Problems that are not deterministic in polynomial time and could be P type problems if any new characteristic is acknowledged.

BPP - Problems that can be solved in polynomial time with probabilistic errors.

NP Complete - Problems that are not deterministic but whose solution can be verified in polynomial time.

PSPACE - This class includes the problems that can be solved given infinite time but consuming polynomial memory.

EXPTIME - Problems that can be solved in exponential time.

NP Hard - Problems that cannot be solved in polynomial time and are as hard as the most difficult NP problems.

There are many other classes apart from these, for a detailed theory the reader is referred to the book: Computation complexity by Sanjeev arora and Boaz barak [7]. Reif [8] found a solution to a vehicle moving through a euclidean space while avoiding polygonal obstacles, but the differential constraints in this case are not considered. The differential constraints considered is a curvature bound path, hence it can be seen as a path planning problem from a point to point such that the path is the shortest path given an upper bound on the radius of curvature. Lazard, Reif and Wang [9] show that such a problem is NP-hard, which tells that there are no exact algorithms for this problem, hence the use of methods based on numerical approximation and discretization would be more appropriate to find a solution which can be nearly optimal.

2.2. General Path Planning Problem

Any path planning problem involves starting in some initial configuration and trying to reach a specific goal configuration. The actions are selected such that the goal configuration is achieved. There are differential constraints enforced on the path like the bound on path curvature, rate of path curvature and others so that smoothness is ensured for the vehicle to easily traverse it. Apart from the constraints in a path planning problem there would be criteria that would define the cost of the solution. **Based on the cost of the solution we can classify the solutions to be feasible or optimal.**

Say a vehicle has to go from point A to B without collision, a **feasible solution** would be any route taken from A to B without collision, whereas an **optimal solution** would be the one which consumes the least time. Here the quality of the solution is determined by the time spent by the vehicle to go from A to B. The majority of the current literature still focuses on finding a feasible solution or local optima, as it is already a hard problem to solve. In many cases even formulating the correct optimizing criteria is difficult, and even if it is formu-

lated it would be impossible to find a solution to it. Many methods used presently, provide feasible solutions that are closer to the optimal solution.

If we have to put this problem in mathematical terms, say the configuration space is defined as C , the initial configuration is $q_{start} \in C$ and the goal is represented as $q_{goal} \in C$. The region where the vehicle is freely allowed to move is C_{free} and the region with obstacles is represented by C_{obs} . The differential constraints mentioned above can be represented as $g(q)$ where q represents the state.

Definition 3 *Path* A function $\sigma : [0,1] \rightarrow R^2 \times SO(1)$ is called a,

- *Path, if it is continuous*
- *Collision free path, if it is a path and $\sigma(\alpha) \in C_{free}$, for all $\alpha \in [0,1]$*
- *Feasible path, if its a collision free path, $\sigma(0) = q_{start}$, and $\sigma(1) = q_{goal}$*

The quality of the solution is measured by a cost function $J(\sigma)$ (length of the path etc.), then an optimization problem can be constructed as follows

Problem 1

$$\begin{aligned}
 & \text{minimize } J(\sigma) \\
 & \text{subject to} \\
 & \sigma(0) = q_{start} \\
 & \sigma(\alpha) \in C_{free} \\
 & \sigma(1) = q_{goal} \\
 & \text{given constraints } g(q)
 \end{aligned} \tag{2.3}$$

2.3. Models used in path planning

In order to plan the route it is important to approximate the behaviour of the autonomous vehicle in response to the input given to it, such approximation is done by a vehicle model. The more complex the model is higher the accuracy it will achieve with the actual vehicle. But higher complexity comes at the cost of computational time and power. Hence one needs to meticulously select the model depending on the available computational power and path planning algorithm. The ground vehicle can be classified as a non-holonomic system, as the controllable DOF (Degrees of Freedom) are fewer than total DOF of the system. The normal a car is supposed to have 4 DOF in total where two are coordinate axis x and y direction and the other two are orientation and heading. Only the first two are controllable (Lateral and longitudinal control). Basically a vehicle model is a rigid body over springs i.e the suspension system and the tire. A definite vehicle dynamics model should be built to represent the vehicle dynamics as it could be used in simulations to validate any new instrumental additions to the vehicle. The simulation of the vehicle can be carried out computationally and hence reduces the cost of testing on a real vehicle. Generally the vehicle model can be classified into two viz. the ride model and the handling model. The ride model describes the performance of the vehicles damper displacement, body acceleration and wheel acceleration vertically. The ride model has 3 DOF which is vertical roll and pitch motion [10]. The handling model has 7 DOF, 3 of which corresponds to longitudinal, lateral and yaw motions, the other four represent the spinning motion of each wheel. The model that is relevant to path planning is the handling model.

2.3.1. Geometric Model

This model basically considers the vehicle dimensions, position and its radius of curvature without any consideration of velocity and acceleration. This model has seen its application in the pure pursuit path tracking method. This model is not used for simulation of vehicle motion as its components do not regard velocity and acceleration, but geometric models like Dubins Model which is based on Dubins curves[11] can be used for simulation of path planning as it would account for the non-holonomic constraints and would be an inexpensive alternative to simulate difficulty of planning paths for a vehicle like agent over holonomic agents that do not account for this nature.

2.3.2. Kinematic Model

Kinematics defines the motion of a body or group bodies without the consideration of mass, internal forces and inertia. Hence the motion of the body would be describes in terms of position, velocity and acceleration. None of the body's internal dynamics would be considered and only the geometry would only be considered. Kinematic model is commonly found due to its simplicity and the important data that it gives about the motion of the vehicle. This model describes the lateral, longitudinal and yaw dynamics in terms of vehicle velocity and acceleration. A simple kinematic model as seen in figure 2.1 is a vehicle with a local coordinates (x-y) and with world coordinates (X-Y). The local coordinates are fixed on the vehicle while the world coordinates are fixed to the ground with respect to which the model has to be simulated. If the vehicle is traveling with a velocity v with a heading angle θ and orientation ψ with respect to the world reference frame . The following equation gives the velocity of the vehicle in world coordinates

$$\begin{bmatrix} v_{worldx} \\ v_{worldy} \end{bmatrix} = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \begin{bmatrix} v_{localx} \\ v_{localy} \end{bmatrix} \quad (2.4)$$

2.3.3. Dynamic Model

Dynamic model describes the motion of the vehicle while considering forces and energy in terms of position, velocity and acceleration. The model includes the forces on the tire and the body, it has characteristics from both geometric and kinematic model. The modelling equations 2.5 are constructed using the equation of motions on the model in fig 2.2.

$$\begin{aligned} F_{xfl} + F_{xfr} + F_{xrl} + F_{xrr} &= m_b a_x \\ F_{yfl} + F_{yfr} + F_{trl} + F_{trr} &= m_b a_y \\ \sum M_z + [-F_{yrl} - F_{yrr}]l_r + [F_{yfl} + F_{yfr}]l_f &= I_{zz} r \end{aligned} \quad (2.5)$$

2.4. Path Planning Algorithms

Over the years there have been many types of algorithms developed to tackle the path planning problem. As discussed in the earlier that via numerical methods finding an optimal solution with even one differential constraint is a **NP-hard** problem. Over the course of time a variety of methods have been developed which can be separated into three categories viz. Variational methods, graph-search methods and sampling based methods.

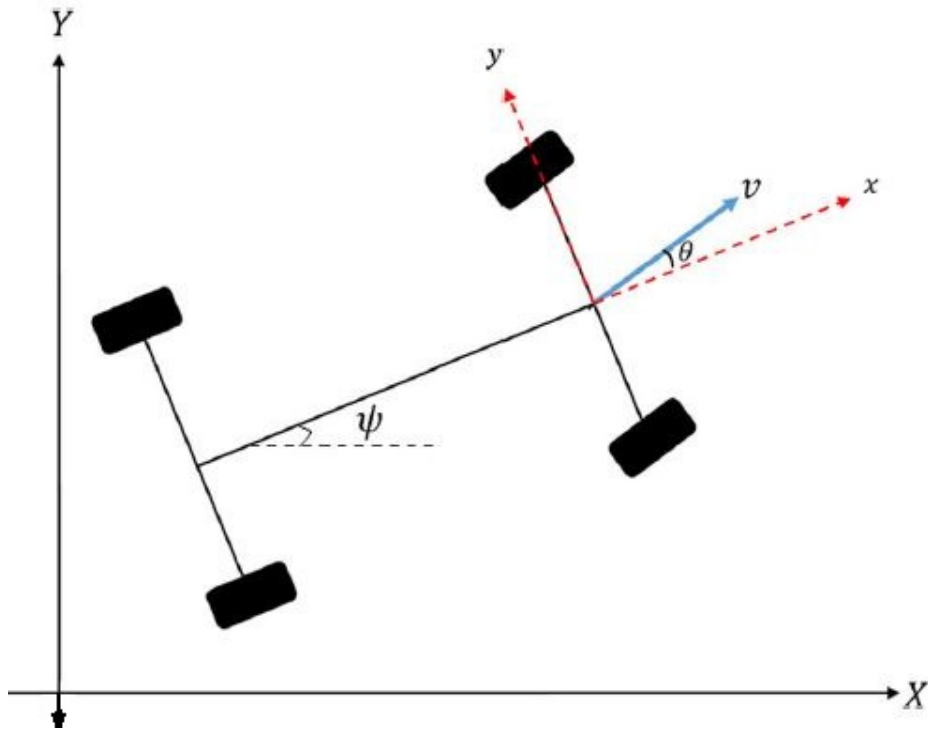


Figure 2.1: Vehicle Kinematic Model

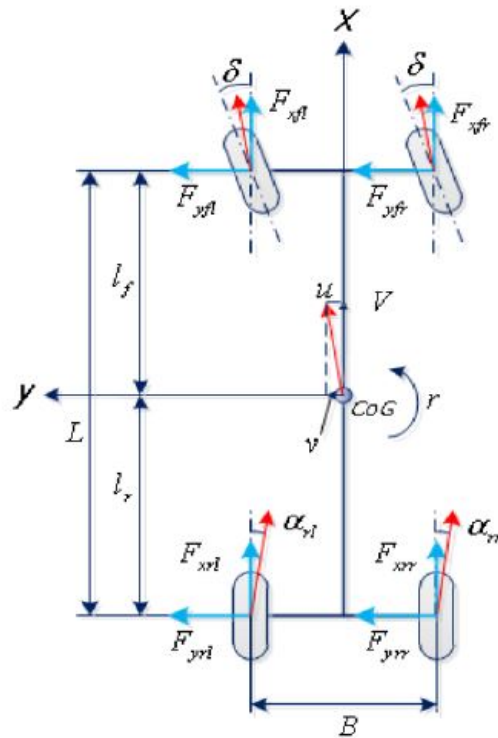


Figure 2.2: Dynamic Vehicle Model

2.4.1. Variational Methods

Variational methods are a general class of approximation techniques with wide application throughout applied mathematics. These methods involve representing the path as a function of certain parameters, these parameters are found by construction of a nonlinear optimization problem with the cost function defining the quality of the solution. These methods are known for the high rate of convergence, they find an optima but lack the resources to find a global optima unless an initial solution is provided. Variational methods are broadly classified into direct methods and indirect methods.

Direct methods are based on an iterative strategy for optimization based on gradient computation of the function space. Advantage of this method is that the convergence of the solution is independent of the initial solution. But rate of convergence decreases as the solution is approached and the need for termination is not known. In [12] the implementation of Euler integration for trajectory planning of a ground vehicle is seen.

Indirect methods are based on an iterative strategy for optimization based on the solution of a point BVP (Boundary Value Problem) with a system of ODE (Ordinary Differential Equation). The necessary conditions are provided by Pontryagin's minimum principle [13], yet it doesn't provide the sufficient conditions. The sufficient conditions are provided by HJB (Hamilton-Jacobi-Bellman)[13] equation. So the solution must satisfy these conditions. Solving a 2 point BVP is very difficult as the solution that may satisfy the first boundary conditions may not satisfy the other. One way to solve this is to let initial conditions be floating so that integration in search of initial conditions would lead to finding the solution, this method is known as shooting method [13]. Trent et.al [14] introduce a modified simple shooting method is used to trajectory optimization problem which solves a two point BVP for a aircraft re-entry, the study showed it to be faster and more accurate than conventional shooting method. But indirect methods have a lot of memory issues and need an initial solution to begin with.

2.4.2. Graph search Methods

These methods involve discretization of the configuration space into a graph G with vertices V and edges E . The vertices represent the various configurations which belong to the configuration space C and edges is constituent of q_{start} , q_{goal} , σ where q_{start} is the origin, the configuration that the vehicle starts from, q_{goal} is the desired end configuration and σ is the path that connects the origin to the desired configuration. The edges are constructed in such a way that they lie in C_{free} and satisfy the constraints $g(q)$. The graph constructions can be done by lane graph, geometric graphs or sampling based methods.

Lane graphs are partially human created and partially created via an algorithm, that refers to a digital map. The paths generated are good for vehicles to traverse through, but these maps do not consider the presence of obstacles in the environment causing it to lose one of the main aspects of path planning.

Geometric graphs represents the obstacles with a geometric form, polygonal when 2-Dimensional. Any point in this graph is reachable via a vertex on the graph. Cell decomposition is used to discretize the configuration space into cells, while using voronoi diagrams to construct a continuous collision free path. This method needs the exact obstacle dimensions to plan paths which might not be available all the time.

It would be computationally expensive to rely on the sensors to manage and collect the data dynamically. Also there are many other constraints to be considered for path plan-

ning apart from differential constraints of the curve. Sampling based techniques do not require the presence of geometric data of the environment and the constraints. Instead it checks reachability with steering and collision checks. The steering is checked with a steer function that constructs a path satisfying the differential constraints and collision checking function to ensure the path is collision free.

After the discretization a search would be performed over these graphs to get a path with best solution. The most known and basic algorithm to find the shortest path is the Dijkstra's algorithm [15] which builds a tree to find the shortest path. The next known algorithm is based on heuristic search, which is called the A* algorithm developed by Hart, Nilsson et.al [16]. It returns an optimal solution. For lesser computational effort A* was modified so that heuristics are multiplied by constant weights. The problem with these algorithms is that they need to be ran everytime as the sensors update the environment data. For such cases the D* (Dynamic A*) algorithm was developed which was able to do calculation in real time.

2.4.3. Sampling Based Methods

The disadvantage of graph search methods is that the search is only over the fixed graph discretized by certain motion primitives, which may cause the algorithms to return a path that is not feasible. These methods try to mitigate this problem. The grid of discretization gets increasingly finer, while trying to find a feasible path at the same time. Such a task requires an exploring behaviour from the algorithm. The process is selection of a random vertex from the tree which begins at the vehicle initial position, which will be expanded by applying the steer function. Once the goal is found the path is traced back to the initial vertex. This is where modern path planning is currently researching on, the next section gives a detailed literature of the various path planning methods used before and the state of the art methods used for urban path planning.

2.5. State of the art

As mentioned earlier path planning is an important topic of research with the development of autonomous vehicles, going over the major focus of the research which is path planning in urban environments this section would present and discuss the state of the art methods that have made contributions to the aforementioned.

Rapidly Exploring Random Trees (RRT) proposed by LaValle [17] is a method that uses randomized data structure for path planning. These are specifically designed to handle non-holonomic constraints with high dimension. The rapid exploration is achieved by taking a random sample in the configuration space and extending a the tree towards this sample via the nearest node available on that tree. The path between the node and the random sample is constructed using a steering function. Kunz et.al in [18] that RRT is not probabilistically complete. It was also proven to provide sub-optimal solutions in [19].

An extension to RRT was introduced by Karman et.al [19] called RRT* which converged to an optimal solution asymptotically. RRT and RRT* are pivotal to the algorithms that plan paths in heterogeneous environments. RRT* involved nearest neighbour search and

rewiring operations that helped in providing optimality.

Algorithm 1 RRT*

Require: Problem P

while Iteration < Max Iterations **do**

$q_{rand} \leftarrow \text{RandomState}()$

$q_{near} \leftarrow \text{SearchNear}(\text{Tree}, q_{rand})$

if Collision free **then**

 Extend from tree

end if

$nn \leftarrow \text{nearestNeighbour}(q_{near}, \text{radius})$ {Obtaining the set of nearest neighbours in a radius}

 ObtainbestParent(q_{near}, nn) {Obtaining the best parent for near node}

 Rewire(nn) {Rewiring the nearest neighbours}

 UpdateTree

end while

As shown in Algorithm 1 it initially samples a random sample from the configuration space, after which it searches for node pre-existing on the tree that is closer to the random sample. If the path extending from the nearest node is collision free an extension is made towards the random sample. Then the nearest neighbours are selected around the new sample in a radius dependent on the number of vertices in the tree [19] present the best radius function as well. Then amongst the nearest neighbours a new parent is chosen ensuring that the cost to arrive to the new sample is the least from the root of the tree and the nearest neighbours are rewired with each other to have least cost, this ensures the nodes in tree around a new sample continuously be update to improve the quality of the solution and achieve an optimal solution.

Whenever path is planned in urban environments with narrow passages as mentioned in chapter 1, It is a challenge to plan with narrow passages and complex environments. Retraction Based RRT [20] is method that tries planning in presence of narrow environments. The basic idea of this method is to retract a randomly generated sample to a more desirable region in the free space. A desirable location for retraction would be the closest boundary point in the free space. Intuitively, the new sample which is close to the boundary, is retracted into the narrow passage. This increases the number of samples in this narrow passage and is almost independent of the volume of the narrow passage.

Environment Guided RRT introduced in [21] is a combined approach that combines two methods Reachability Guided RRT [22] and Resolute Complete RRT [23] to produce a hybrid method that can handle kinodynamic constraints and environment uncertainties at the same time while planning a path through it smoothly and efficiently by using forward dynamics. Although this method has worked on bettering sampling in cluttered environment along with uncertainty it does not focus on solution convergence.

Adaptive RRT [24] is one of the modern methods which selects growth strategies based on a two level selection one of which involves the visibility of the nodes, i.e how easily a node can be connected from an adjacent node and its cost based learning approach. The

next chapter gives a more detailed outlook on this method.

RRT*-Smart introduced in [25] aims to improve the convergence of RRT* via intelligent sampling around the initial solution and path optimizing. On obtaining the initial solution via RRT* optimizes the path via pruning repeatable nodes and setting up sampling regions around the nodes present in the solution. Although this produced convergence, it was very much dependent on the quality and speed of initial solution and the pruning of nodes during path optimization caused loss of information making the algorithm less optimal.

Informed RRT* [2] implements a similar technique but rather than using the nodes themselves it uses the theoretical minimal cost and the current cost to construct a hyperellipsoid around the initial solution in which uniform sampling is performed to improve the solution. This produced better solutions than RRT*-smart as there was a much more optimized sampling set. But the dependence on initial solution always remained. Homotopy aware RRT* [26] uses human informed to tackle the clutter environments and advocated human-robot interactive path planning and remains to be theoretical without any analysis with existing methods. GART [27] plans in clutter environment with the obstacles producing artificial potential such that it can lead the agent to safety.

2.6. Discussion

Out of the different methods to plan paths, sampling based methods are the ones that led to modern path planning. From the discussion of the methods that focus on urban path planning or planning path with narrow environment present there are not many approaches that focus on obtaining optimal solution in narrow environments. Over the literature discussed it was found that methods either focused completely on tackling environment constraints or tackling the convergence of solutions. Hence there is need for a method that can give good performance in both these aspects and inter-utilize its advantages for both purposes. The next chapter introduces the proposed algorithm that tries to achieve the aforementioned.

Table 2.1: Contributions towards urban path planning

S.no	Research Article	Author	Algorithm	Contribution
1	Rapidly-exploring random trees: A new tool for path planning [17].	LaValle, Steven M	RRT	-Pioneered modern path planning algorithms. -Sampling based method that could handle non-holonomic constraints
2	Optimal kinodynamic motion planning using incremental sampling-based methods [19].	Sertac Karaman and Emilio Frazzoli	RRT*	-Proved to be asymptotically optimal -Introduced features like nearest neighbours -Rewiring operation to improve efficiency
3	An efficient retraction-based RRT planner [20].	Zhang, Liangjun and Manocha, Dinesh	Retraction Based RRT	-Computes samples near the obstacle space thereby identifying the bounds of the obstacle (contact space). -Manages to plan paths in narrow passages
4	Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles [21].	Léonard Jaillet, Judy Hoffman, Jur Van den Berg, Pieter Abbeel, Josep M Porta, and Ken Goldberg	EGRRT	-Manages kindodynamic and environment constraints using a combined approach using forward dynamics. -Handles environment uncertainty
5	Adapting RRT growth for heterogeneous environments [24].	Jory Denny, Marco Morales, Samuel Rodriguez, and Nancy M Amato	Adaptive RRT	-Introduces the concept of visibility -Uses a cost based learning approach to adapt growth
6	RRT*-smart: A rapid convergence implementation of RRT [25].	Jauwairia Nasir, Fahad Islam, Usman Malik, Yasar Ayaz, Osman Hasan, Mushtaq Khan, and Mannan Saeed Muhammad	RRT* Smart	-Improvises initial solution by intelligent sampling around the nodes.
7	Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic [2].	Jonathan D Gam-mell, Siddhartha S Srinivasa, and Timothy D Barfoot	Informed RRT*	-Optimizes the initial solution by the construction of a hyperellipsoid around it and directly sampling from it
8	Homotopy-aware RRT*: Toward human-robot topological path-planning [26].	Daqing Yi, Michael A Goodrich, and Kevin D Seppi.	Homotopy aware RRT*	-Presented a theory for human-robot interactive path planning algorithm
9	GART: An environment-guided path planner for robots in crowded environments under kinodynamic constraints [27].	Liang Yang, Jizhong Xiao, Juntong Qi, Liying Yang, Lei Wang, and Jianda Han.	GART	-Introduces bidirectional potential field to redistribute each newly sampled state, such that the in-collision samples can be redistributed for extension.

3

Methodology

3.1. Introduction

It has been discussed in chapter 2 how RRT* [19] has provided solution for optimal path planning, guaranteeing asymptotic optimality and probabilistic completeness. Although this algorithm is able to find an optimal solution, it comes at the cost of incalculable time. It was found that Informed RRT* [2] addressed this problem and drastically improved the convergence rate by employing a heuristic that set up the sampling set to reach local optimality using the initial solution. But the sampling still remains in question as the probability of sampling a reachable sample in a narrow or cluttered environment is lower with random sampling when compared to sampling in open spaces. In [24] it is seen that Adaptive RRT efficiently samples around obstacles using growth strategies which uses implicit obstacle data to adapt growth of the tree. This chapter introduces the combined approach proposed in this thesis, it leverages the advantages of Informed RRT* and Adaptive RRT to converge to available optimal solution in finite time.

3.2. Proposed Approach

The algorithm introduced in this thesis is a combined approach that absorbs the exploratory strengths of Informed RRT* and adaptability of Adaptive RRT in order to solve the path planning problem in narrow environments while using the Dubins motion model as it steer function. The overview of the algorithm can be seen in Algorithm 2.

Algorithm 2 Proposed Approach**Require:** Problem P, Scenario 1/2/3, growth distance Δd **while** Iter < Max Iteration **do** **if** q_{goal} is reached **then** $q_{rand} \leftarrow \text{InformedSample}(q_{start}, q_{goal}, \text{current solution})$ {Random sample selected from the ellipsoidal informed subset} **else** $q_{rand} \leftarrow \text{RandomState}(\text{Configuration space bounds})$ **end if** $q_{near} = \text{SearchNear}(\text{Tree}, q_{rand})$ {Search for the nearest node} $vis \leftarrow \text{visibility}(q_{near})$ {Obtaining visibility of the near node} $GSS \leftarrow \text{SelectGrowthStrategySet}(vis)$ {Growth strategy set based on the visibility} $GS \leftarrow \text{SelectGrowtStrategy}(GSS)$ {Update probabilities and obtain growth strategy} $q_{new} \leftarrow \text{Grow}(GS)$ {Grow the tree towards the new node using Dubins Motion Model} $\text{UpdateVisibility}(q_{near}, q_{new}, q_{rand})$ {Update visibility of the new node based on the growth} $\text{Reward}(GS)$ {Reward the growth strategy based on its performance} $nn \leftarrow \text{nearestneighbour}(q_{near}, \text{radius})$ {Obtaining nearest neighbours around q_{near} in a radius} $\text{ObtainbestParent}(q_{near}, nn)$ {Obtaining the best parent for near node} $\text{Rewire}(nn)$ {Rewiring the nearest neighbours} UpdateTree Iter $\leftarrow$ Iter+1**end while****if** Solution Exists **then** $\text{Display}()$ {Displays Solution}**else**

No solution

end if

The proposed algorithm addresses the research objectives as seen in Chapter 1. The Dubins model helps in incorporating a realizable motion model with vehicular constraints. Adaptive RRT works on efficiently sampling in the complex environment. Informed RRT* helps in converging the solution to local optima in finite time. These 3 algorithms are the major constituents of the proposed method as seen in figure 3.1. The base structure of algorithm pertains to a standard RRT* algorithm. The algorithm initially obtains the start and goal configurations along with model, configuration space and obstacle specifications. The tree is initialized with starting node, while maximum iterations is reached the algorithm samples a configuration randomly either from the configuration space or the informed subset depending on whether a path is not found or not. The construction of informed subset and sampling from it can be seen in section 3.2.4. After sampling a configuration, the algorithm finds a nearest node on the current tree. The visibility of this particular node is obtained, visibility is seen in detail in the section 3.2.2. Using this visibility a growth strategy set is selected which contains multiple growth strategies that may be suitable for growing towards the random sample. The selection of growth strategy and rewarding is as explained in sections 3.2.2 and 3.2.3. Once the new configuration is obtained, the best par-

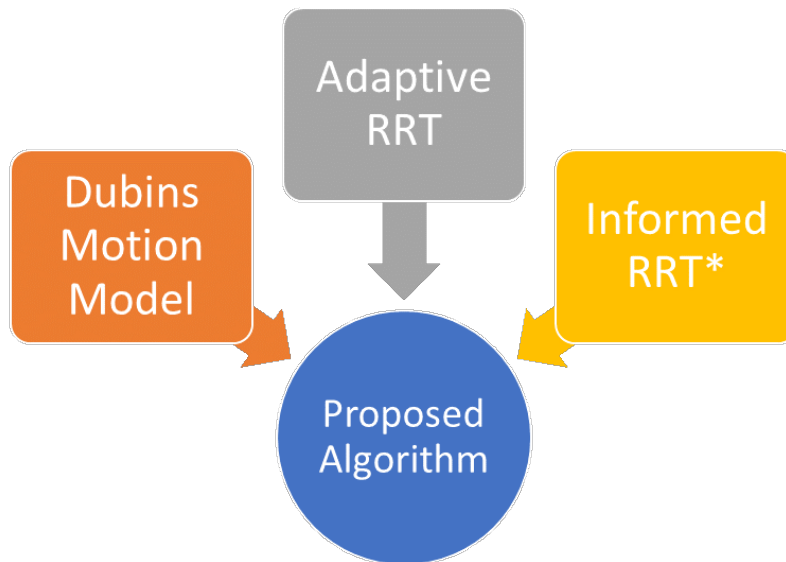


Figure 3.1: Constituents of the proposed approach

ent is selected from the nearest neighbor set and these neighbors are rewired as explained before. This algorithm is expected to improve performance in finding initial solution and convergence. The subsections ahead would explain all of the constituent algorithms of the proposed approach.

3.2.1. Dubin's Motion Model

Imagine having an open field with two points 100 meters apart, the shortest path between these two points would be a straight line connected between them. As a human walking that straight line would break no sweat, with a car one can get to that point even faster. But the case with car isn't always true, say the orientation of the vehicle was such that it was facing away from that point, to add more say we had the requirement of reaching the goal point at a particular orientation. The human can still do the task easily, but the car cannot follow the same shortest path. Cars are different they don't just slide horizontally or turn about their center of mass. Planning for cars can be made easier by using a Dubin's car. In robotics based on how controllable an agent is it can be broadly classified as holonomic or Non-holonomic agents. The human is an agent that can walk forward, backward, sideways and can turn about the same point, such an agent is said to have control over its coordinates and can be called a holonomic agent. A car on the other hand cannot move side ways and cannot turn about its current position, so an agent that does not have control over its coordinates all the time is called a non-holonomic agent. Cars are capable of turning about a point is at distance from its center of mass, the distance is often called as turning radius. Hence planning the shortest path for a non-holonomic agent is difficult compared to a holonomic agent.

The Dubins car was seen in the literature after it was introduced by Lester Dubins in 1957 [11]. In this model the car has 3 control moves it can make, turn left at maximum angle, turn right at maximum and go straight, it cannot move backwards. The paths that will be planned using the Dubins car model will return a combination of these three control

moves. For optimal planning the control moves can be used in six combinations LSL, RSR, RSL, LSR, RLR, LRL, where L stands for left max, R stands for right max and S stands for straight. The next section will give an idea about the modelling involved in the Dubins car.

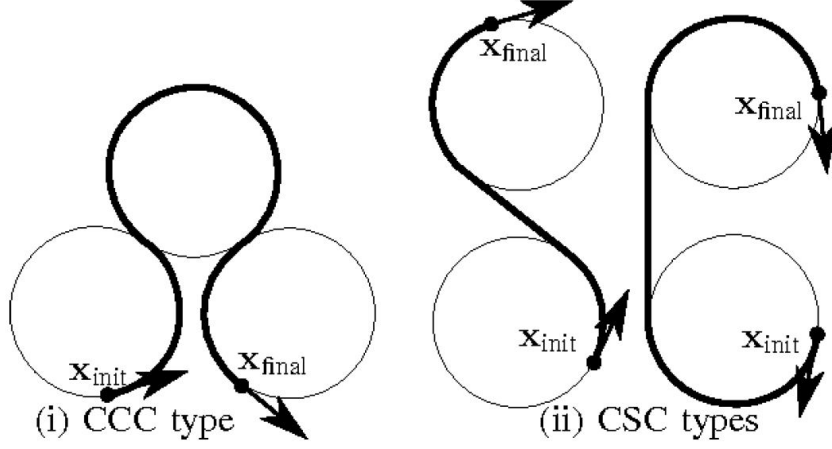


Figure 3.2: Types of Dubin Curves

Modelling

The vehicle is assumed to move at unit velocity, hence the shortest path can be solved using simple geometry. To describe the configuration of the car we use position and orientation that is x, y, θ . When the car turns max left and max right its turning radius will be smallest in value. To construct the equations that will explain the dynamics of the vehicle, let say the vehicle is at point $P1(x1, y1, \theta1)$ and it moved straight for unit time, the new point configuration is $P2(x1+v\cos\theta1, x1+v\cos\theta1, \theta1)$. If it was turning then it would turn with an angular velocity $\omega=(v/r_{min})$, where r_{min} is the minimum turning radius. The entire system can be represented by the following set of equations (considering unit velocity).

$$\begin{aligned}\dot{x} &= \cos\theta \\ \dot{y} &= \sin\theta \\ \dot{\theta} &= \omega = \frac{1}{r_{min}}\end{aligned}\tag{3.1}$$

The system above is linear, but an actual car is a nonlinear system. To approximate the vehicle well it is advisable to trace paths at smaller steps hence the velocity is set to unity.

Path Construction

Generally the path that can be constructed using Dubins car can be classified into two types CSC (Curve-Straight-Curve) and CCC(Curve-Curve-Curve).

- **CSC Path**

The CSC trajectories that are possible from the combinations discussed above are (LSL,RSR,LSR and RSL), that is a max turn followed by a straight line followed by another max turn. Consider the start and end position of the vehicle as two points with arrows pointing in the direction of the orientation. Construct circles to the left and

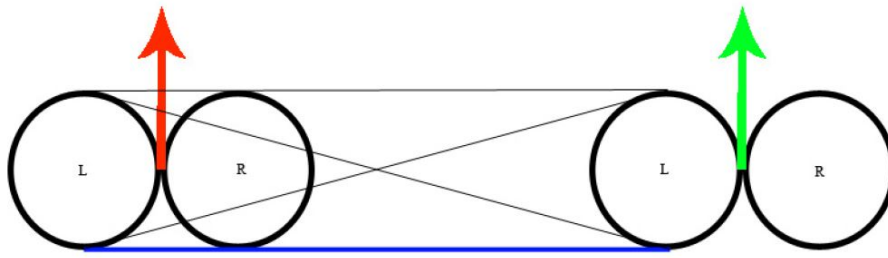


Figure 3.3: Possible Tangents

right of the start and goal positions and take a pair of circle and construct tangents (one circle from start and one from the goal), there are four tangents possible for each pair (can be seen in figure 3.3), but only one of which can be a valid solution as a path. The computation of the valid tangent is crucial to finding a path in this combination of control moves.

- **CCC Path**

This path is different from that of CSC path, it consists of a turn followed by a turn in the opposite direction which is followed again by the turn in the original direction. It has a similar setup to that of CSC path, but instead of finding the tangent to join two circles a third circle tangent to both circles is found that will help the car reach its goal.

Dubins Car is one of the simplest ways to accommodate the non-holonomic constraints and is easy to compute. But in the real world all kinds of errors can come into play. Its implementation in RRT can be performed by replacing the moving function with the possible control moves and the cost measurement can be done by using Dubins length, which is the total path length.

Collision check

The approach presented in thesis considers the agent to be a rigid rectangular mass instead of a point mass. Hence, the collision check for any path would be performed by checking collision for the rectangle at every point in the path seen in figure 3.4. Tolerances are included apart from this.

3.2.2. Adaptive RRT

Adaptive RRT [24] is an approach that adapts the growth of the tree based on visibility of the nodes and then selects a suitable growth strategy randomly based on a cost sensitive approach. Algorithm 3 summarizes the working of Adaptive RRT. The subsections below explain the contributing elements of Adaptive RRT.

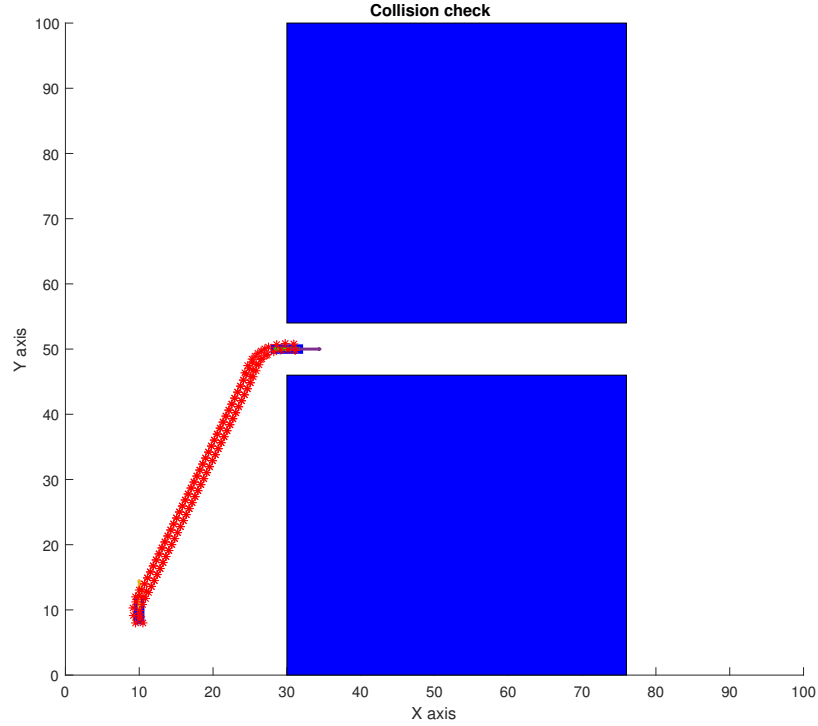


Figure 3.4: Collision check in the proposed approach

Algorithm 3 Adaptive RRT

Require: Problem P , growth distance Δd

while P is not solved **do**

$q_{rand} \leftarrow \text{RandomState}()$

$q_{near} \leftarrow \text{SearchNear}(\text{Tree}, q_{rand})$

$\text{vis} \leftarrow \text{visibility}(q_{near})$ {Obtaining visibility of the near node}

$\text{GSS} \leftarrow \text{SelectGrowthStrategySet}(\text{vis})$ {Growth strategy set based on the visibility }

$\text{GS} \leftarrow \text{SelectGrowtStrategy}(\text{GSS})$

$q_{new} \leftarrow \text{Grow}(\text{GS})$

$\text{UpdateVisibility}(q_{near}, q_{new}, q_{rand})$

$\text{Reward}(\text{GS})$

end while

Visibility

In [24] the concept of visibility is introduced. Visibility is the estimation of how good the connectivity of a node is with its local environment. A node x is said to be visible from another configuration $\bar{x}$ when a path $\tau = \{x, x_1, x_2, \dots, \bar{x}\}$ can be produced with the intermediate configurations being valid. Algorithm 4 updates visibility of nodes on the tree which takes $q_{near}, q_{rand}, q_{new}$ as the input along with a constant penalty and the distance function that calculates distance between two configurations based on the vehicle model. The total distance (TotalDist) between the nearest node and the random sample is calculated using this distance metric. ExpandRatio is the ratio of the distance expanded from the nearest con-

figuration to the total distance between q_{near} and q_{rand} .

Algorithm 4 Evaluation of visibility for nodes

Require: $q_{near}, q_{new}, q_{rand}, \text{Penalty}, \text{distance}$
 TotalDist $\leftarrow \text{distance}(q_{near}, q_{rand})$
 expandedDist $\leftarrow \text{distance}(q_{near}, q_{new})$
 expandRatio $\leftarrow \text{expandedDist} / \text{TotalDist}$
 nearVisibility $\leftarrow q_{near}.\text{visibility}$
 $q_{near}.\text{visibility} \leftarrow \text{Average}(\text{nearVisibility}, \text{expandRatio})$
if $0 < \text{expandRatio} < 1$ **then**
 $q_{new}.\text{visibility} \leftarrow \text{Penalty}$ { Partial expansion}
else if $\text{expandRatio} = 1$ **then**
 $q_{new}.\text{visibility} \leftarrow \text{Average}(\text{nearVisibility}, \text{expandRatio})$ { Full Expansion}
else
 $q_{new}.\text{visibility} \leftarrow 0$ { Null Expansion}
end if

There are three distinct outcomes when it comes to node expansion in RRT- null, partial and full expansion. When there is null expansion the nearest node on the tree q_{near} may be near the obstacle and cannot expand in the direction of the random sample q_{rand} or there is a failure to grow this awarded a visibility 0. Partial expansion is when q_{new} is between q_{near} and q_{rand} due to an obstacle this is awarded a visibility penalty which is a user defined constant and influences the visibility mapping. Full expansion is where expansion is successful all the way to q_{rand} for which the visibility is the average of expandRatio and the visibility of q_{near} . Every time an expansion occurs the visibility of the nearest node is averaged with the expansion ratio which ensure a relative increase or decrease of the visibility based on its ability to expand to a random sample. This visibility value helps in selecting growth strategies which is explained in further subsections.

Growth strategies

To efficiently grow trees in an environment with obstacles it may be beneficial to use growth methods which would rely on obstacle information rather than just being expanded to random samples. Such growth strategies are introduced in [28] where some rely on obstacle information while others preserve the random nature of sampling based methods. The growth strategies dependent on the obstacle information are dependent on elements known as obstacle vectors. The subsections further explain the obstacle vectors and growth methods seen in [28].

Obstacle Vectors

These are vectors obtained from the geometry of the obstacles and assist the growth strategies in selecting a direction in which they can grow. These vectors are generally constructed using a 3D model mesh. In the presented work 2D environments are considered and the workspace is meshed to obtain obstacle vectors which are implicitly used to guide the extension in some of the growth strategies. In fig 3.6, we can see a triangular mesh on a 3-dimensional hollow pipe.

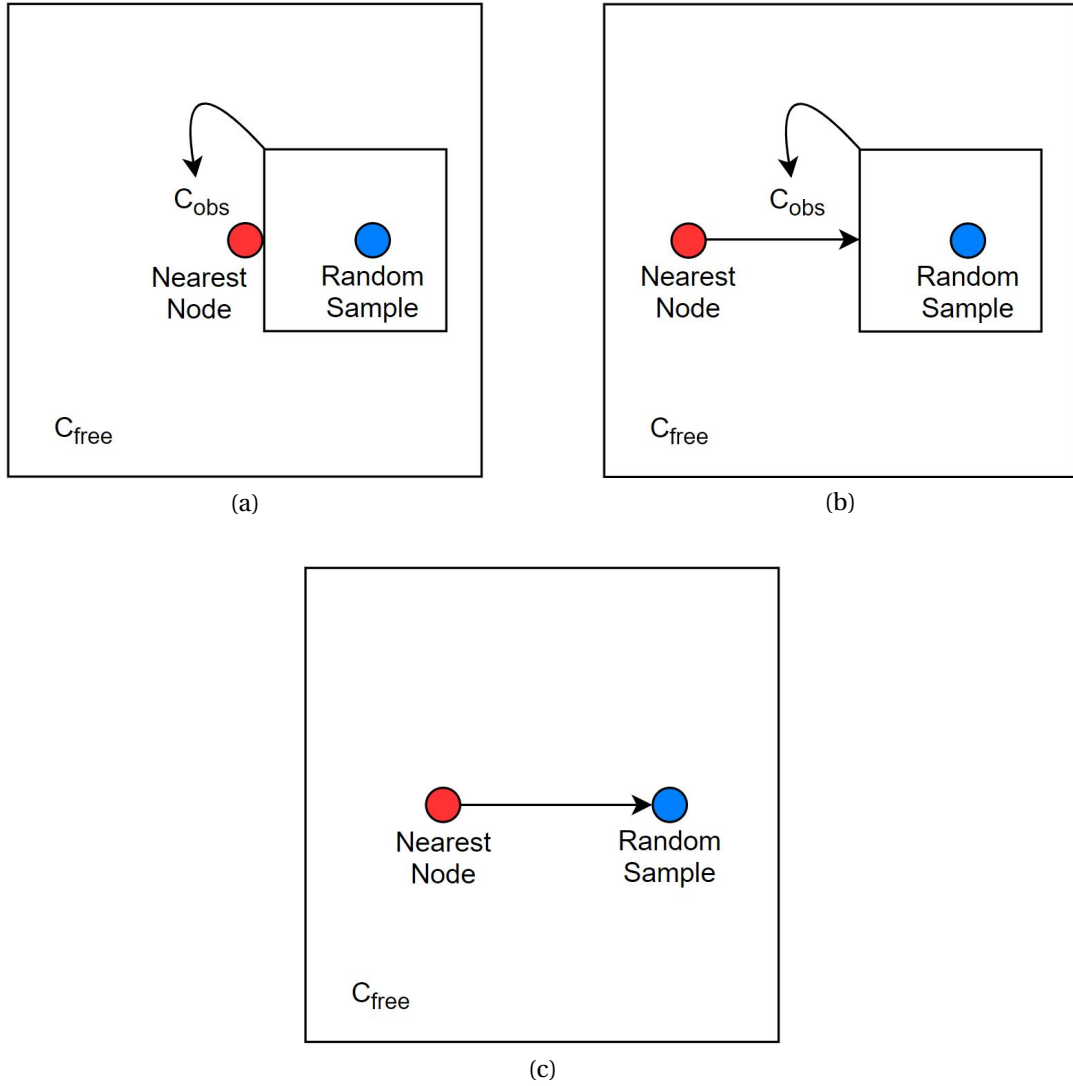


Figure 3.5: Distinct outcomes during expansion in RRT, fig 3.5a Null expansion, fig 3.5b Partial expansion, fig 3.5c Full expansion with C_{free} and C_{obs} being the free configuration space and the obstacle space respectively

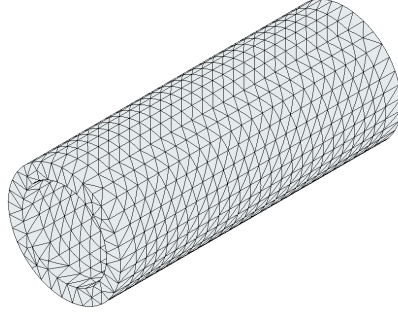


Figure 3.6: Triangular mesh on a 3D hollow pipe

The research in this report focuses on 2D obstacles, hence there is a need for simplification of the mesh in order to obtain obstacle vectors. The obstacles being dealt with are mostly rectangles or a combination of rectangles. A rectangle can be simply be meshed as a combination of two triangles as seen in fig 3.7, the double sided arrows indicate the obstacle vectors. So it can be concluded that obstacle vectors are more like directional vectors which help some of the growth methods which can be seen in the following subsection.

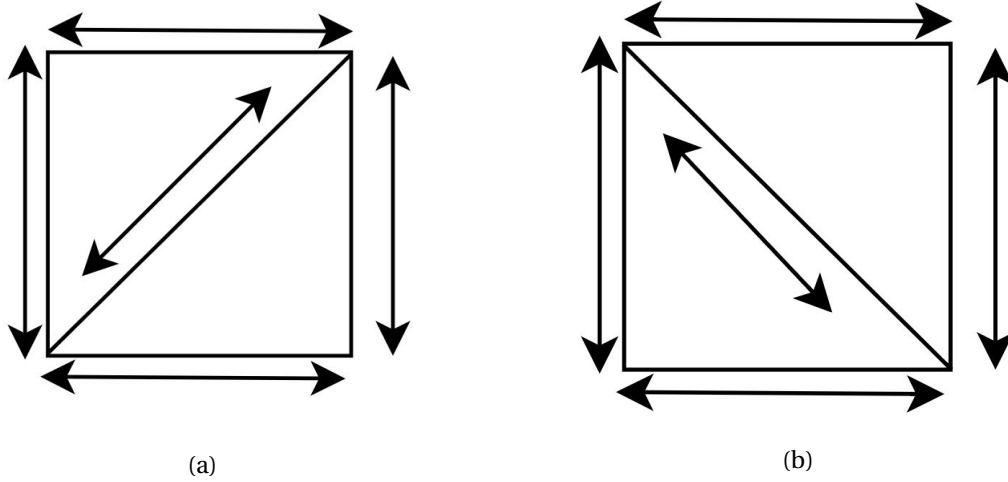


Figure 3.7: Triangular mesh with obstacle vectors of a 2D obstacle

Growth Methods

The growth methods in [28] are specifically designed for a robotic arm, hence its states consist of position and orientation. This is helpful in the case of this project as the focus is on a Dubin's Vehicle Model being planned in a $S^1 \times R^2$ space, but some of the growth methods didn't seem to satisfy the constraints of a vehicle. The growth methods that seemed to be good to use involved randomness in terms of orientation and methods which traced obstacle vectors while keeping the orientation same. Out of the nine methods discussed in the paper, seven methods were selected which are as follows,

- G1: Basic extension

In this method, the extension the new random sample is set as same as the random sample generated initially which can be seen in the figure 3.8, the red and blue arrows indicate the orientation.

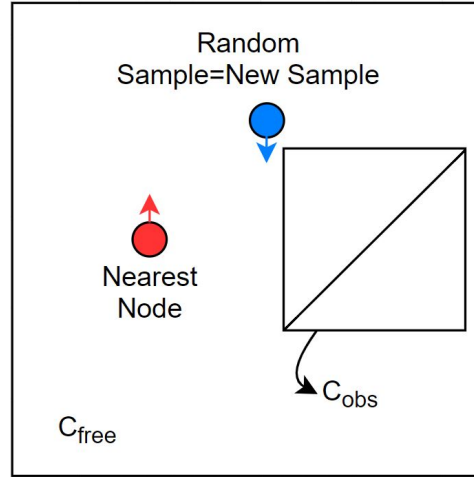


Figure 3.8: Basic Extension

- G2:Random position with same orientation

In this method of extension the new random sample has random positional vectors (x and y coordinates) but maintains the same orientation as that of the nearest node. The growth can be visualized as seen in figure 3.9

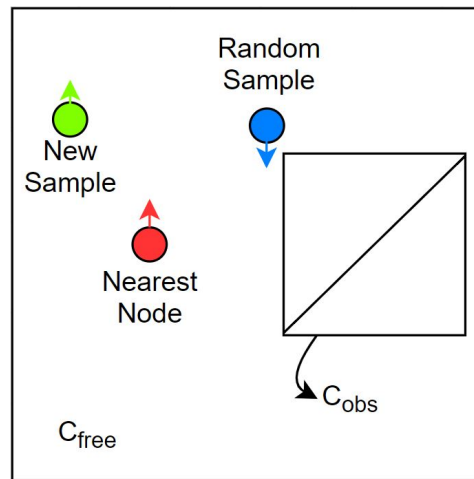


Figure 3.9: Random position with same orientation

- G3:Random obstacle vector with random orientation

In this method of growth the new random sample has positional vector grown in the direction of one of the obstacle vectors (Equation 3.2) that is randomly chosen while being assigned a random orientation. In the figure 3.10, it is observed the diagonal obstacle vector (highlighted in magenta) being chose while the new sample having a completely random direction. The magnitude of the obstacle vector chosen would be constant and is given by the user.

$$q_{new}.pos = q_{near}.pos + Obs\vec{Vect} \quad (3.2)$$

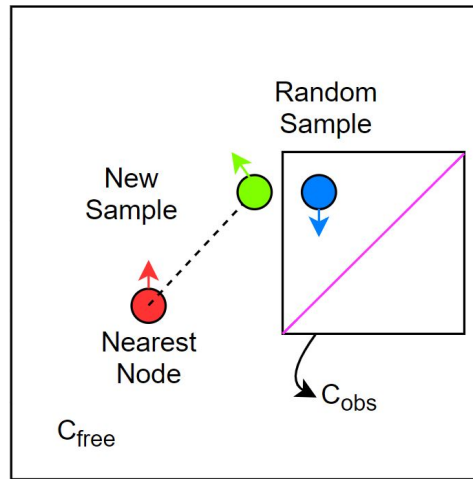


Figure 3.10: Random obstacle vector with random orientation

- G4: Random Obstacle Vector with same orientation

This method is same as G3, but the only difference is that the orientation is same as that of the nearest node.

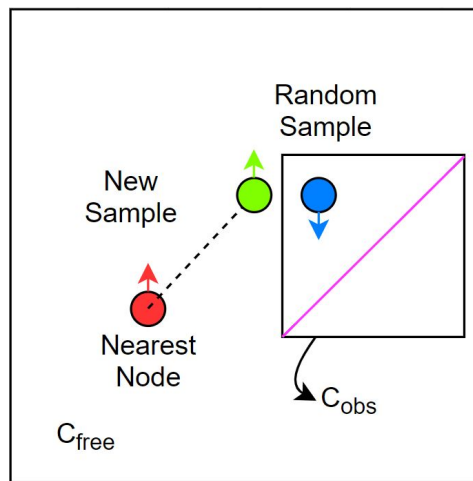


Figure 3.11: Random obstacle vector with same orientation

- G5: Trace Obstacle with Random Orientation

In this method of growth, the first colliding triangle in the mesh of an obstacle is found by growing via G1, on finding the triangle an obstacle vector is randomly chosen and the new sample is constructed in the direction of this obstacle vector using the equation 3.2 and random orientation is assigned. In figure 3.12, it is seen that the selected triangle is highlighted in red and the randomly selected obstacle vector is highlighted in magenta.

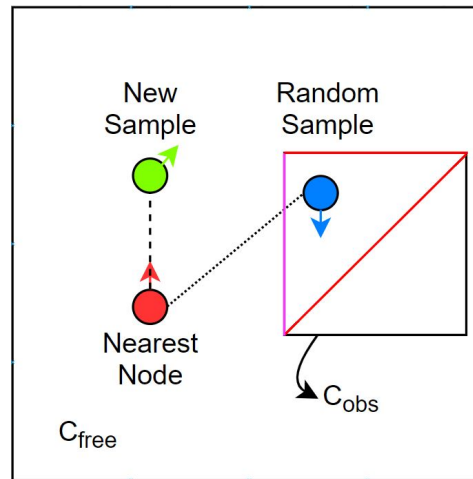


Figure 3.12: Trace obstacle with random orientation

- G6: Trace Obstacle with same Orientation

This method is same as G5 with the exception of the orientation being same as that of the nearest node.

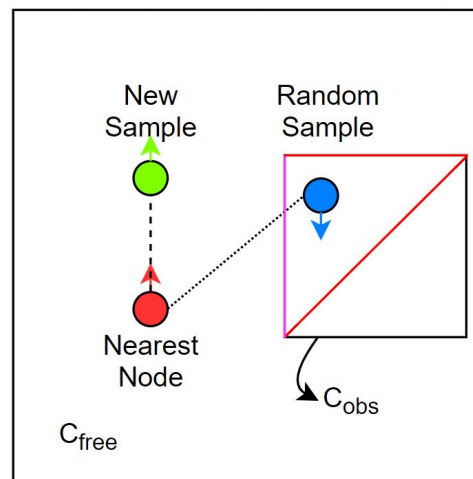


Figure 3.13: Trace obstacle with same orientation

- G7: Trace Obstacle Boundary

In this method the objective is to find a vector parallel to the obstacle space using an circle search method where a circle with a fixed radius and the center being the nearest node is constructed, whose points of intersection are recorded to establish a vector parallel to the obstacle boundary between the random sample and the nearest node. From figure 3.14, it can be seen that node is moved in the same direction as that of the vector found via the intersection of the circle with the obstacle and the magnitude is equivalent to that of the vector. This growth method is very suitable for scenarios with narrow passages.

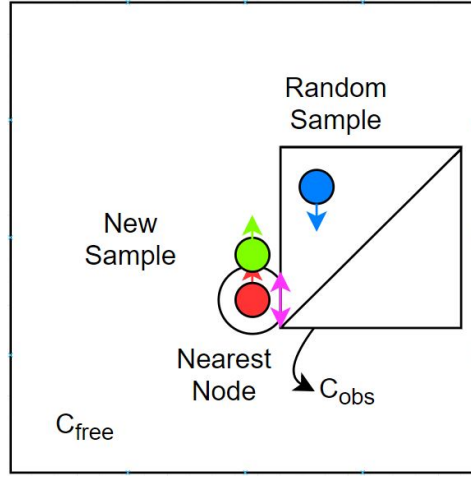


Figure 3.14: Trace obstacle boundary

3.2.3. Selection of Growth strategies

Despite the availability of several growth strategies it can be seen that not all of these will work successfully all of the time. There is a need for a systematic approach to use the growth methods. In the paper [24] Jory Denny et.al construct growth strategy sets which contain multiple growth strategies that can be used depending on the visibility of the nearest node. From the growth strategy sets these methods are chosen randomly via cost sensitive probabilities that are evaluated every iteration. Initially all growth strategies are given a constant weight w and cost c , then a cost insensitive probability is calculated using equation 3.3.

$$cip* = (1 - \gamma) * \frac{\log(w + 1)}{\sum_{i=0}^n \log(w_i + 1)} + \gamma \frac{1}{n} \quad (3.3)$$

After which cost is taken into account to calculate final probability seen in equation 3.4

$$p = \frac{\frac{cip*}{c}}{\sum_{i=0}^n \frac{cip*_i}{c_i}} \quad (3.4)$$

Using this probability a method is randomly selected. After a method is selected a reward is given based on its expansion ratio, if there is failure to expand (i.e expansion ratio = 0) the reward r is set to -0.001 , the methods that are not selected are rewarded with 0, after which weights are updated for all the strategies using 3.5. The reader is directed to [29] for more details on the probability based selection.

$$w = w * \exp(\gamma \frac{r}{(cip*)(n)}) \quad (3.5)$$

3.2.4. Informed RRT*

Informed RRT* introduced in [2] by Gammell et.al is an algorithm that is built over basic RRT* in order to improve the rate of convergence with the help of an admissible ellipsoidal heuristic. It has been shown to outperform RRT* in terms of convergence in a highly cluttered environment. On obtaining the initial solution, the algorithm calculates an ellipsoidal

subset in the configuration space with the help of the cost of the current solution and the theoretical minimum cost. As the algorithm is based on RRT* it maintains the property of being probabilistically complete. (i.e the probability of finding an optimal solution is 1 as time tends to ∞). The basic structure Informed RRT* is given in Algorithm 5.

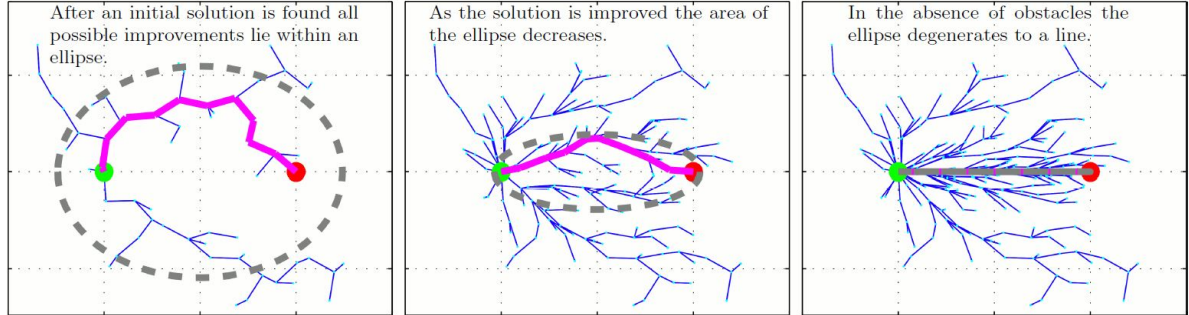


Figure 3.15: Progress of solution Informed RRT* adapted from [2]

Algorithm 5 Informed RRT*

Require: Problem P

while Iteration < Max Iterations **do**

if P is not solved **then**

$q_{rand} \leftarrow \text{RandomState}()$

else

$q_{rand} \leftarrow \text{InformedSample}(q_{start}, q_{goal}, \text{current solution})$ {Random sample selected from the ellipsoidal informed subset}

end if

$q_{near} \leftarrow \text{SearchNear}(\text{Tree}, q_{rand})$

$nn \leftarrow \text{nearestNeighbour}(q_{near}, \text{radius})$ {Obtaining the set of nearest neighbours in a radius}

 ObtainbestParent(q_{near}, nn) {Obtaining the best parent for near node}

 Rewire(nn) {Rewiring the nearest neighbours}

 UpdateTree

end while

Ellipsoidal informed subset

The purpose of using an Ellipsoidal informed subset can be understood as the increased chances in improving a solution by obtaining useful samples in the configuration space by reducing the sampling set. Given a positive cost between the q_{start} and q_{goal} configuration with a constraint to pass through only C (Configuration space) apart from the motion constraints on the agent with c_{best} as the current best cost. Euclidean distance can be considered as an admissible distance metric to obtain global minima even with a vehicle like agent. The new informed subset that will be able improve the solution further must consists of sample which are satisfy the conditions seen in 3.6, which is the equation of a n-dimensional prolate hyper-ellipsoid. For the details on admissibility of the hyper ellipsoid the user is referred to [2]. As the solution keeps improving the ellipsoid keeps shrinking to global minima which can be seen in 3.15.

$$C_{informed} = \{q \in C \mid \|q_{start} - q\|_2 + \|q - q_{goal}\|_2 \leq c_{best}\} \quad (3.6)$$

The focal points are q_{start} and q_{goal} with the traverse diameter being c_{best} and the conjugate diameter being $\sqrt{c_{best}^2 - c_{min}^2}$ which can be seen in figure 3.16

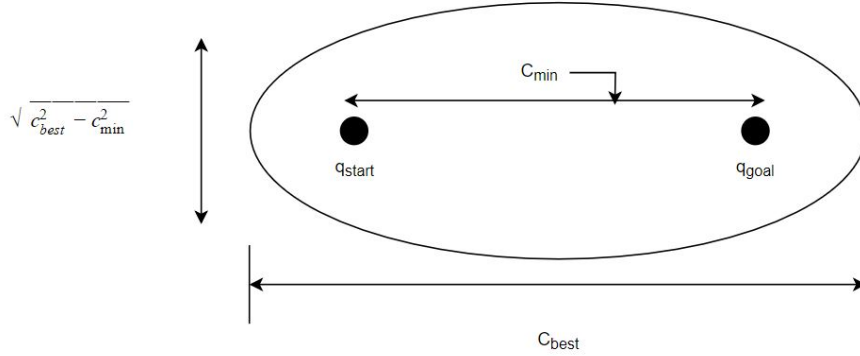


Figure 3.16: Informed ellipsoid set

Sampling from the Ellipsoid

Uniformly distributed samples in the Ellipsoidal Informed set can be obtained by firstly uniformly sampling in a unit n-ball. Geometrically a ball can be defined as a region or space comprising of all points at a fixed distance from a given point. An n-ball is a ball in a n-dimensional euclidean space, so unit n ball will be a ball with unit radius in an n dimensional euclidean space. Since the research here is fixed to a 2-dimensional space, the uniform sampling occurs in a unit 2-ball which is also known as a circle. Once a particular point is sampled in the unit n-ball it can be transformed to the ellipse using equation 3.7 where $q_{centre} = (q_{start} + q_{goal})/2$.

$$q_{ellipse} = L * q_{ball} + q_{centre} \quad (3.7)$$

L is the transformation matrix which can be obtained through the cholskey decomposition of the hyperellipsoid matrix, $S \in \mathbb{R}$

$$L^T L = S, \text{ where} \quad (3.8)$$

$$(q - q_{centre})^T S (q - q_{centre}) = 1 \quad (3.9)$$

We obtain the hyperellipsoid matrix with eigen values corresponding to the radii of the informed ellipsoid, hence S can be written in the form of a diagonal matrix as

$$S = \text{diag}\left\{\frac{c_{best}^2}{4}, \frac{c_{best}^2 - c_{min}^2}{4}\right\} \quad (3.10)$$

with L resulting as,

$$L = \text{diag}\left\{\frac{c_{best}}{2}, \frac{\sqrt{c_{best}^2 - c_{min}^2}}{2}\right\} \quad (3.11)$$

Rotation from hyper-ellipsoid to the world coordinates can be done by solving the Wahba problem [30]. The rotation matrix is given by

$$C = U \text{diag}\{1, \det(U)\det(V)\} V^T \quad (3.12)$$

Where $\det()$ is the determinant of the matrix. Where $U \in \mathbb{R}^{2 \times 2}$ and $V \in \mathbb{R}^{2 \times 2}$ are unitary matrices such that $U \Sigma V^T = M$ due to singular value decomposition. Where M is the product of the outer transverse axis a_1 and the first column of the identity matrix. $M = a_1 1_1$ where,

$$a_1 = \frac{q_{goal} - q_{start}}{\|q_{goal} - q_{start}\|}, \text{Hence} \quad (3.13)$$

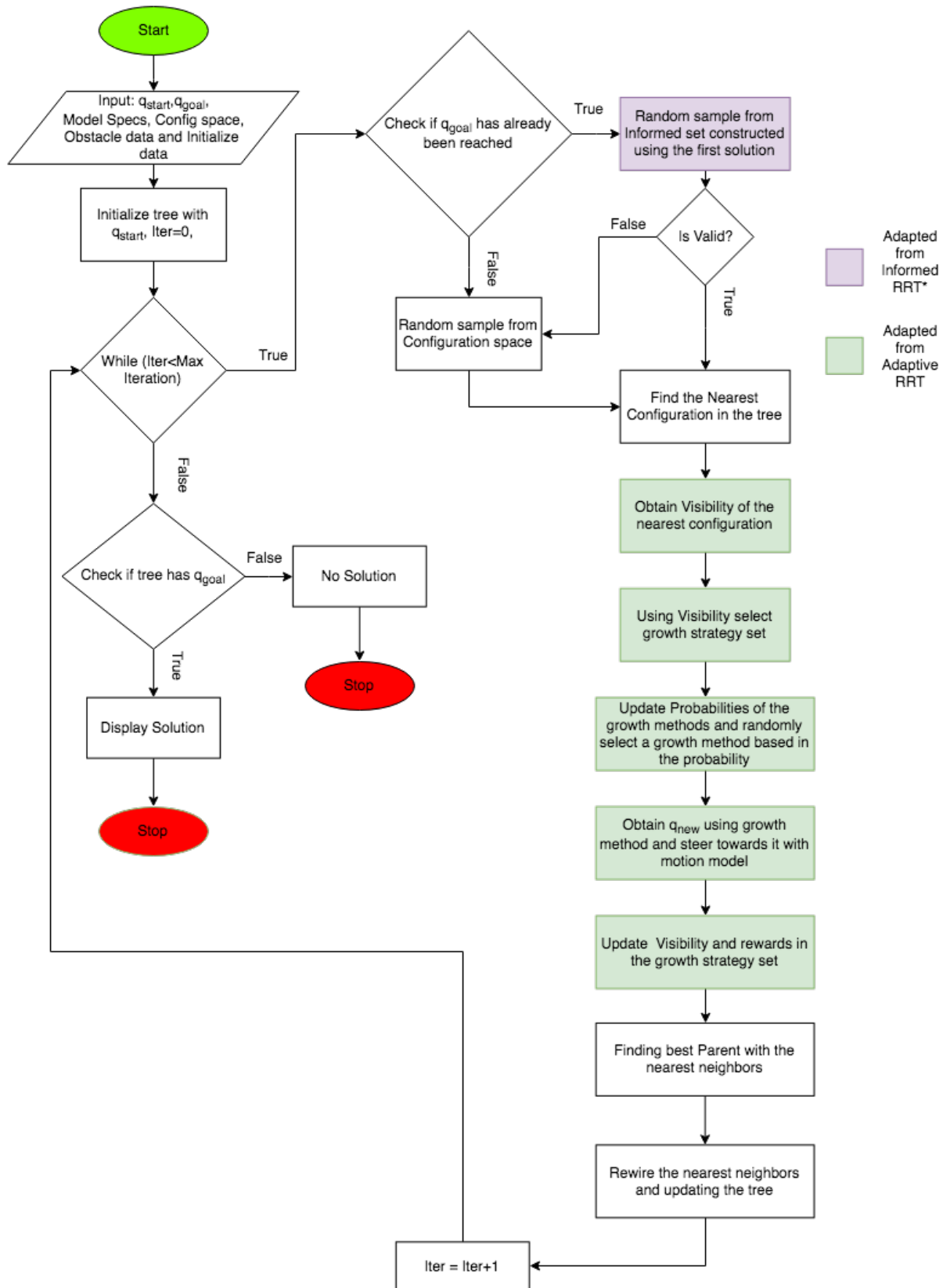
the final transformation to world q_{world} is given by:

$$q_{world} = CLq_{ball} + q_{centre} \quad (3.14)$$

if the transformed sample does not lie in the configuration space, a point is normally sampled to replace it. As the research also deals with orientation it is just sampled the standard way by assigning a random orientation between $(0 - 2\pi)$

3.3. Discussion

This chapter explains in detail the combined approach proposed in this thesis, the entire process can be inferred from the figure 3.17. The proposed approach borrows the shrinking ellipsoidal sampling set from Informed RRT* to help its convergence. Uses the two step sampling strategy of Adaptive RRT to sample efficiently in an obstacle filled environment. The motion model used is Dubins model which lets the agent have non holonomic constraints similar to a vehicle. Furthermore the collision check implemented is also explained as it accounts for vehicle and obstacle boundary interaction. Despite the heuristics the algorithm functions similar to RRT*. Hence, its **probabilistic completeness** and **asymptotic optimality** is theoretically is guaranteed. For the mathematical proofs, readers are referred to the paper by karman et.al [19]. In order to prove it is better than the existing state of art, it is important to perform validation. The validation of the proposed method is seen in the next chapter. Its initial solution and convergence performance is evaluated in three different scenarios with varying obstacle complexity.

Figure 3.17: **Combined Approach** introduced in this Thesis

4

Simulations and Results

4.1. Simulation setup

This chapter presents the testing and simulation of the algorithms mentioned in Chapters 2 and 3, the algorithms are scripted and simulated in MatLab version- 2017a. The simulation package, road-maps along with the vehicle motion model have been entirely coded from scratch without any external libraries or dependencies. The tests that were carried out to assess the performance of finding the initial solution and convergence for a fixed number of nodes.

4.1.1. Simulation Parameters

Table 4.1: Parameters used in the simulation

Parameter	Annotation	Value
Vehicle Model	-	Dubins Motion Model
Map size	-	100*100 Units
Vehicle Size	-	4*2 Units
Vehicle Localization Tolerances	-	8*4 Units
Vehicle Turning Radius	r_{min}	5 Units
Hit Penalty	Penalty	0.2
Initial Visibility	visibility	1
Initial Weights for growth strategies	w	1
Initial Cost for growth strategies	c	1

4.1.2. Scenario selection

For the simulation carried out, three scenarios were selected which can be seen in figures 4.1, 4.2, 4.3, the first scenario comprises of a passage that the vehicle needs to pass through. The initial state is at (10,10) with the vehicle orientation at 90 degrees with respect to the x-axis and the goal state is at (90,90) with the vehicle orientation at 0 degrees with respect to the x-axis.

Scenario 2 represents roads around a market centre square, simple narrow road that outlines a square. The size of the map is 100 by 100 units. The initial state is at (15,15) with

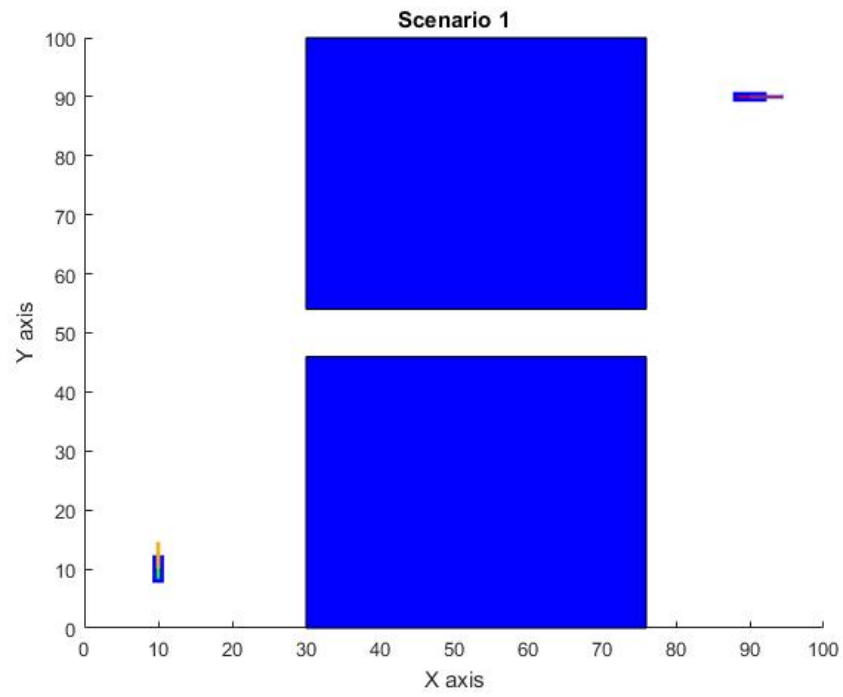


Figure 4.1: Scenario 1 map

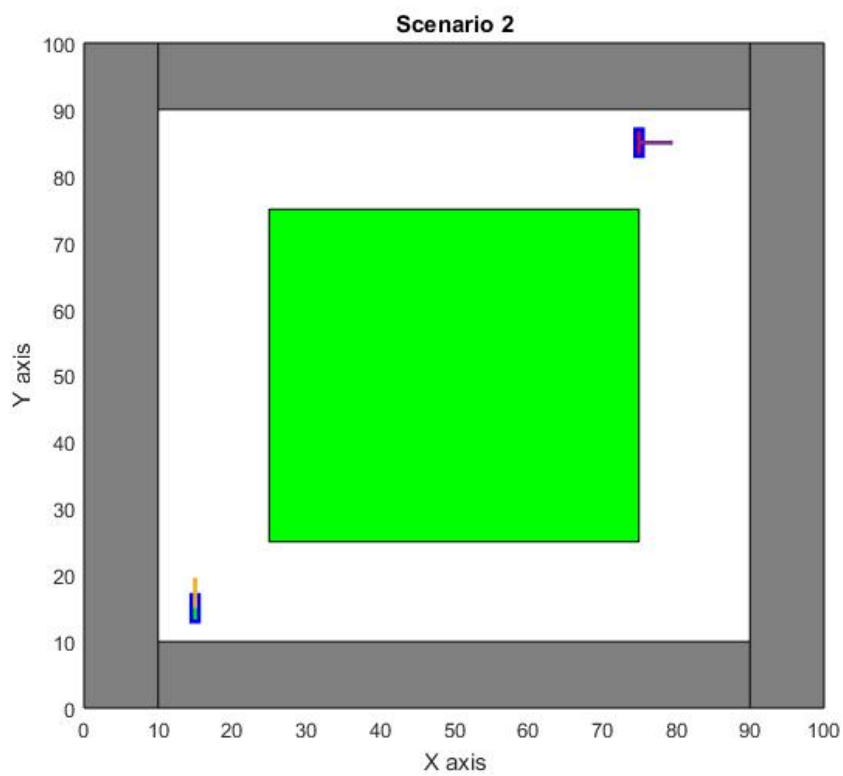


Figure 4.2: Scenario 2 map

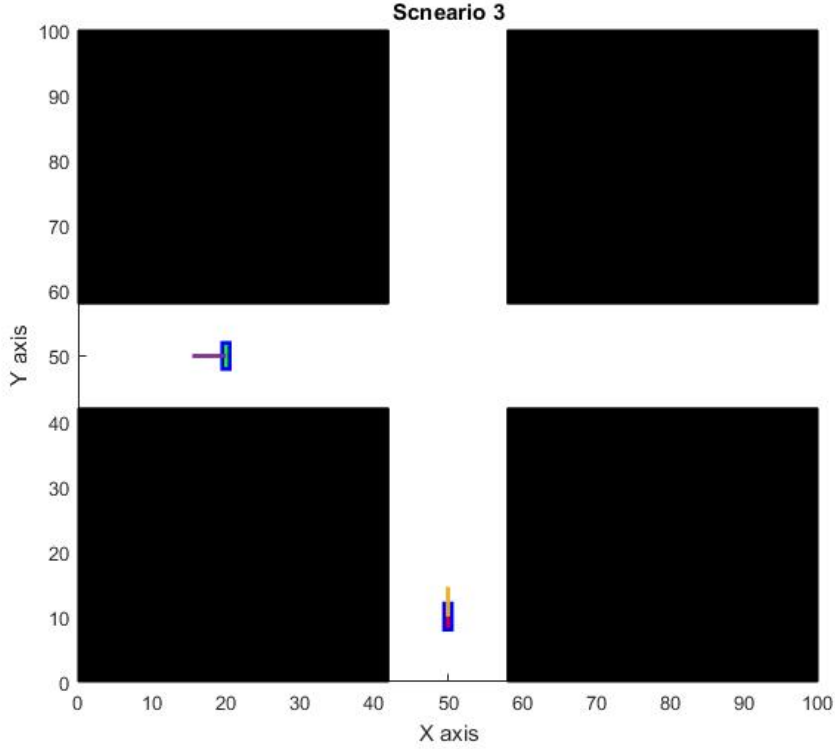


Figure 4.3: Scenario 3 map

the vehicle orientation at 90 degrees with respect to x-axis, and the goal state is (75,85) with the vehicle orientation at 0 degrees with respect to x axis.

Scenario 3 represents a narrow street intersection like scenario. The initial state is at (50,10) with the vehicle orientation at 90 degrees with respect to x-axis and the goal state is at (20,50) with the vehicle orientation at 180 degrees with respect to the x-axis.

On comparison of the three scenarios, it is clear that scenario 1 is has more open space while with a single narrow column, the second scenario only consist of narrow columns and the third scenario is a fusion of the first and second scenarios with moderately narrow columns. The whole idea behind selecting these scenarios is to have a wide spectrum of test environments in which the algorithms can be tested and compared, the free space percentages can be seen in the figure 4.4

4.1.3. Fine tuning for growth strategy selection

As discussed in Section 3.2.2 since the motion model is different from that followed in [24] it is necessary to fine tune the visibility range to growth strategies in order for the algorithm to find solutions smoothly. Hence, the simulation of visibility was carried out according to Algorithm 4 on all scenarios with classification of low visibility with weight below 0.3, medium with weight in between 0.3 and 0.6 and high with weight above 0.6. A sample visibility map on a point mass with Dubin's motion model can be seen in the figure 4.5. The blue points indicate the ones that have relatively high visibility, the green ones indicate the one with medium visibility and the red ones indicate samples with low visibility. Due to lack of samples not many low visibility points are discovered, increased samples would

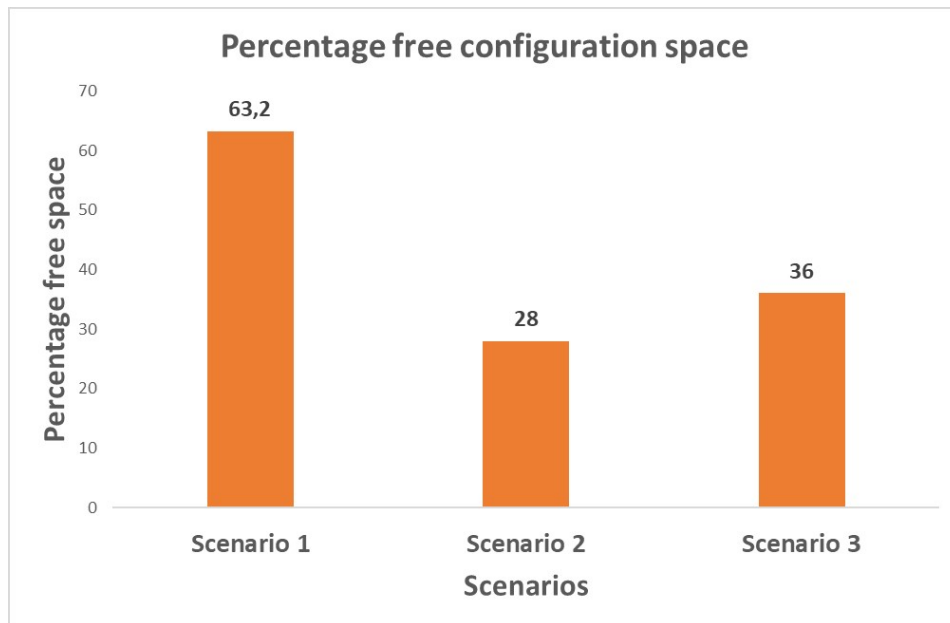


Figure 4.4: Available configuration space in each scenario

make the map cluttered with paths making it difficult to visualize the visibility of the nodes discretely.

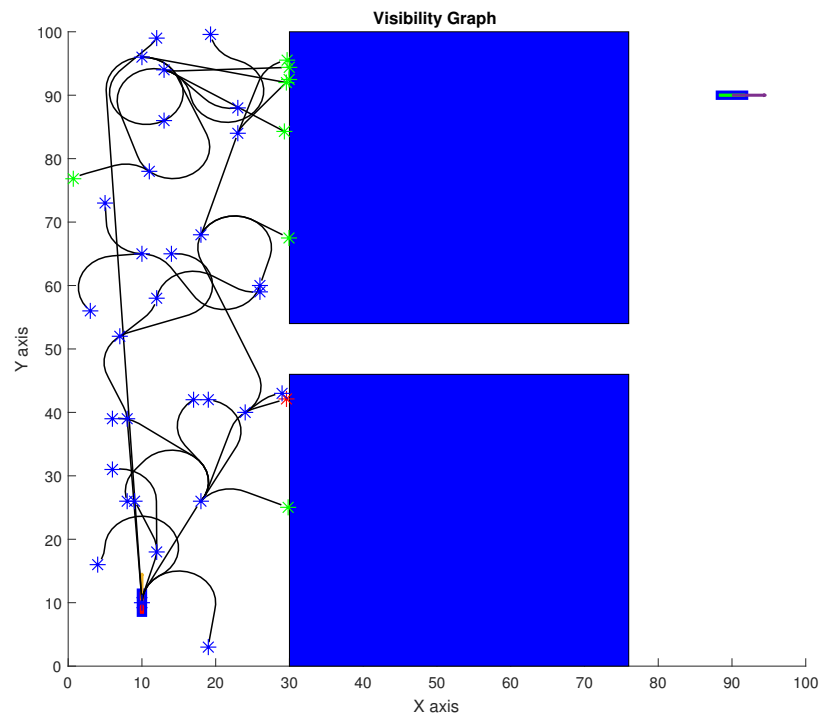


Figure 4.5: Basic Visibility Map

After tuning the growth strategies to the visibility range for different scenarios, the final

growth strategy distribution can be seen in Table 4.2. If the visibility of the node is higher, upper hand is given to methods with randomness. In the case of medium visibility, methods that trace obstacle vectors found more success. Finally in the case of nodes with very low visibility the best performing strategy is tracing the obstacle boundary. As low visibility nodes generally would lie very close to the obstacle boundary.

Table 4.2: Growth Strategy distribution over the visibility range

Visibility	Low	Medium	High
Strategies	{G7}	{G3,G4,G5,G6}	{G1,G2}

4.2. Results

The simulations mainly focus on two performance factors to evaluate its ability to plan in narrow environment, first is in the effectiveness to find an initial solution , second is ability to converge to a better solution for a fixed number of iterations. All the results are expressed on relative performance for easier comparison.

4.2.1. Initial solution performance

This result is presented by performing 100 independent runs until the algorithm finds a solution. As the performance is only limited to the initial solution the algorithms which are compared are RRT* and the proposed algorithm as **informed RRT* functions the same way as RRT* in the initial run**. For comparison, two criteria are looked at. The number of samples used to find the initial solution, which would indicate the efficiency of finding a solution and the time required to find the solution to indicate the computational efficiency. Both the criteria are recorded as the median of the 100 runs to account for randomness. When comparing the number of samples used to find the initial solution seen in figure 4.6, it is observed that the proposed method generally needed fewer nodes to find the solution compared to RRT*. This implies that the proposed method has not only improved effective exploration of the configuration space but also has an increase in the efficiency of the nearest neighbour search which is dependent on the size of the tree which is smaller in the case of proposed method. In terms of planning time an even higher gain in performance is seen compared to RRT* which can be observed in the figure 4.7. There is a clear cut increase in the planning time in scenario 2 which discussed earlier is one of the more challenging environments. Due to the presence of narrow columns alone it causes the algorithm to invoke the expensive growth method G7 increasing the overall computation time.

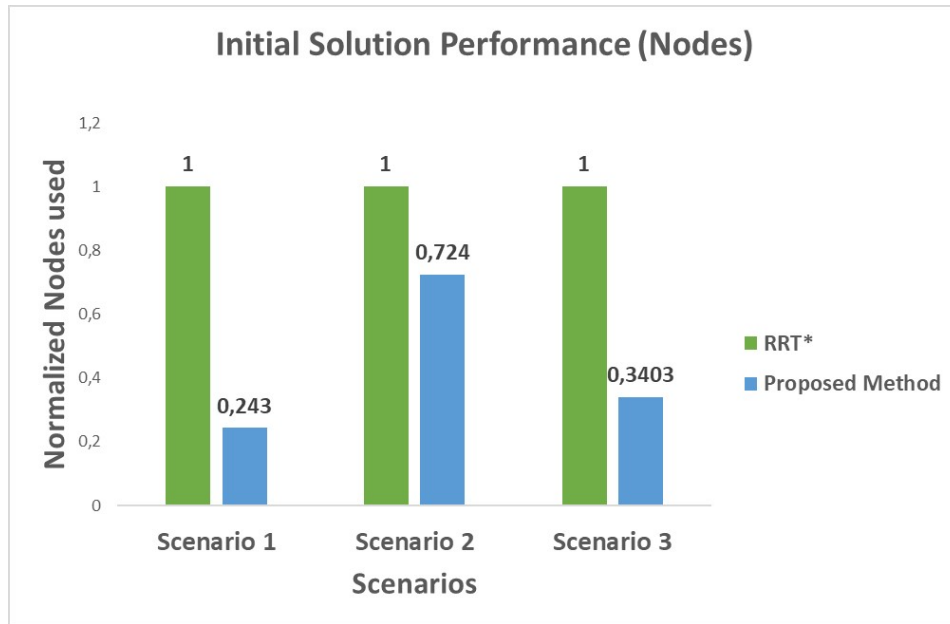


Figure 4.6: Relative performance of the algorithms for the number of samples used to find solution

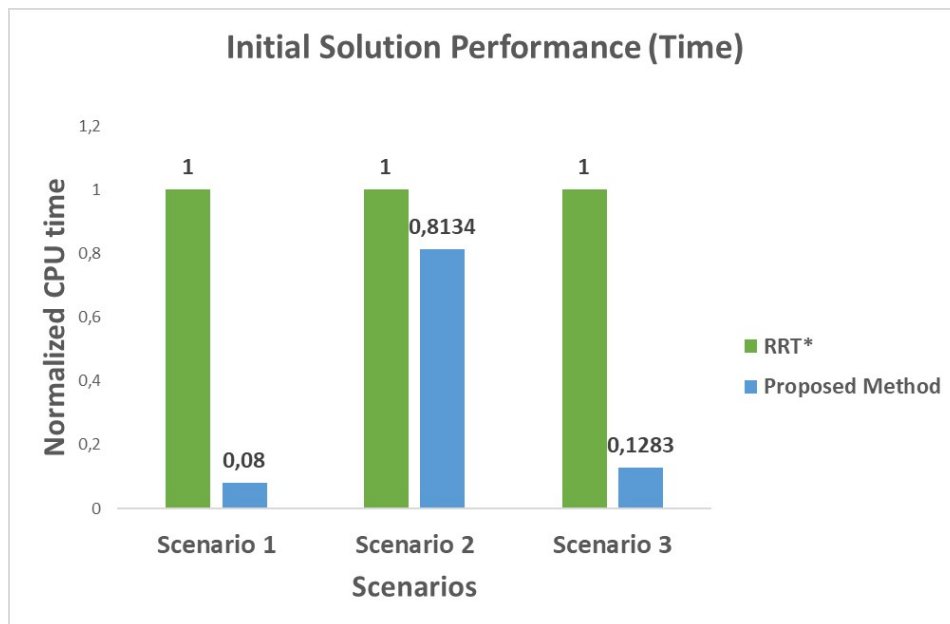


Figure 4.7: Relative performance of the algorithms for the time taken to find initial solution

4.2.2. Convergence Performance

The second result indicates the convergence of the algorithms in order to find a local optimal solution with common pseudo samples until the first solution is found. The convergence comparison is done between RRT*, Informed RRT* and the proposed method. The comparison is done through recording their rate of convergence over 100 runs and taking the median rate of convergence to compare and compensate the randomness of the random sampling that is involved. Sample convergence of the different algorithms can be seen in figures 4.8 to 4.10.

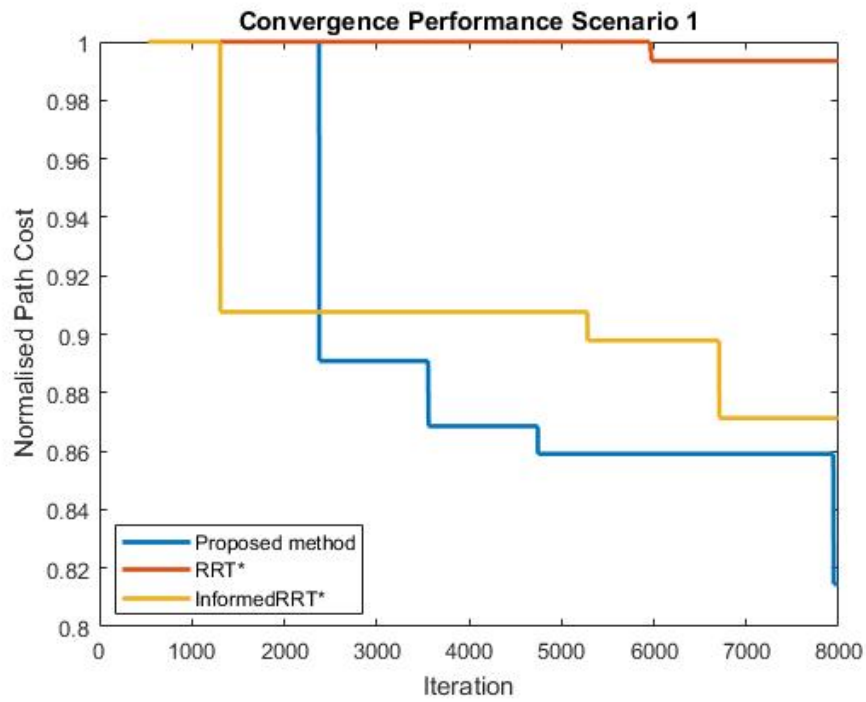


Figure 4.8: Sample convergence data from scenario 1 simulations

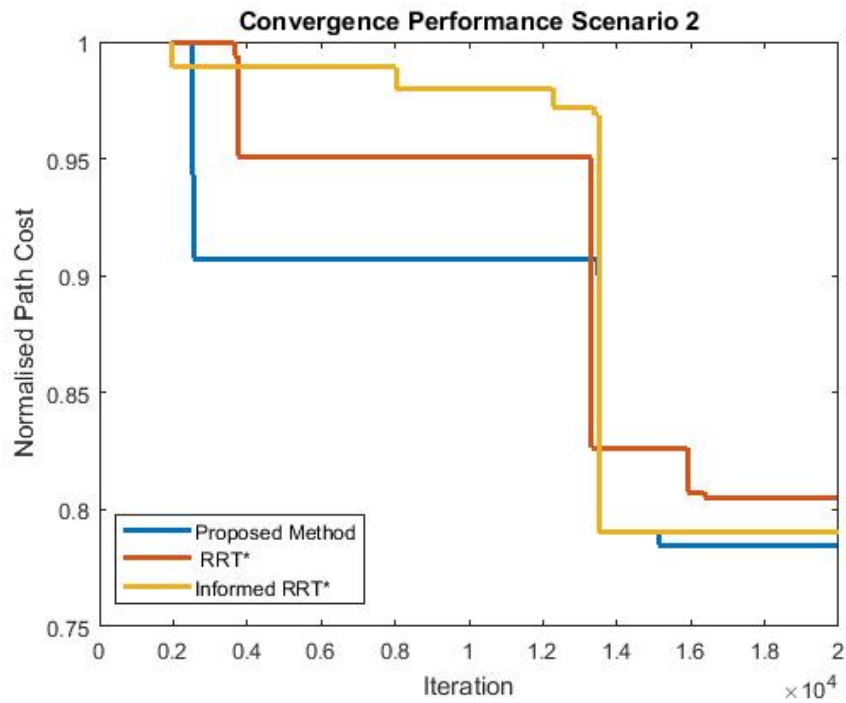


Figure 4.9: Sample convergence data from scenario 2 simulations

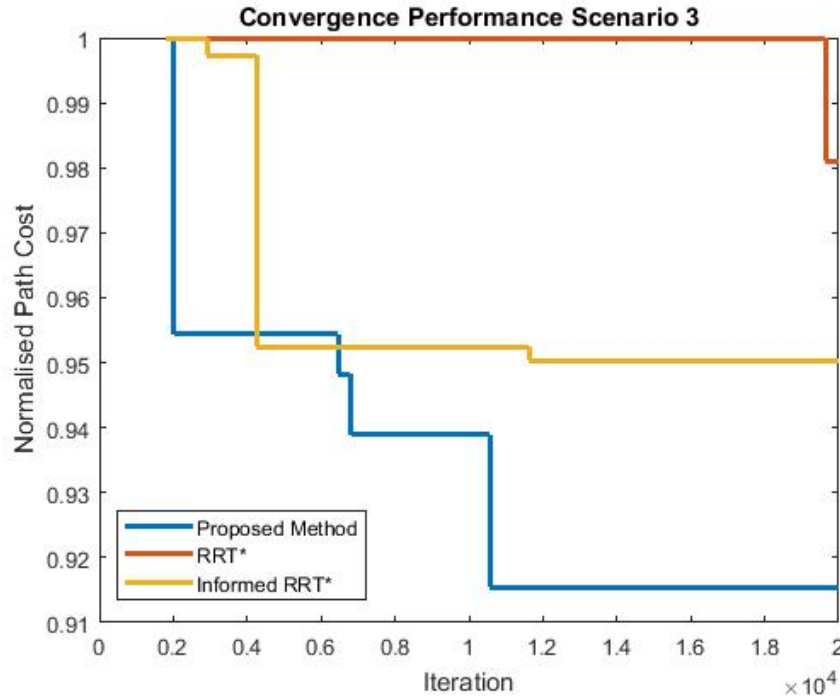


Figure 4.10: Sample convergence data from scenario 3 simulations

Comparing the rate of convergence obtained for all the algorithms in three scenarios which can be observed in figure 4.11, the proposed method comes on top of the other convergence algorithms in all the three scenarios, but it is observed that increasing complexity and linearity of the environment (Attaining an optima which is closer to global optima) shows improvement but much lesser.

4.3. Discussion

The entire chapter goes through the simulation setup, the parameters involved in the simulation and the fine tuning required for the adoption of adaptive growth strategies along with the analysis of selection of the scenarios. The simulation is performed on 3 different scenarios with different environmental constraints while keeping the parameters the same. The results showed that the proposed method outperformed the state of the art algorithms. The sampling efficiency of the proposed combined approach was 50% better than RRT* and Informed RRT* causing lower times to compute initial solution. This is due to the ability to sample in cluttered region using different growth strategies. Its convergence performance was significantly improved due to the same reason despite sampling in the same method as informed RRT*. It was observed in scenario 2, performance variation was much lesser due to the linearity of the environment. Yet the proposed method showed slight outperforming convergence characteristics. The initial solution performance remained competitive in scenario 2 due to the proposed algorithm using the expensive growth methods to overcome the complexity of the environment. After observing all the results it can be said that the combined approach that indulges in the nature of intelligent sampling while adaptive to the environment proved to be superior to the state of the art methods.

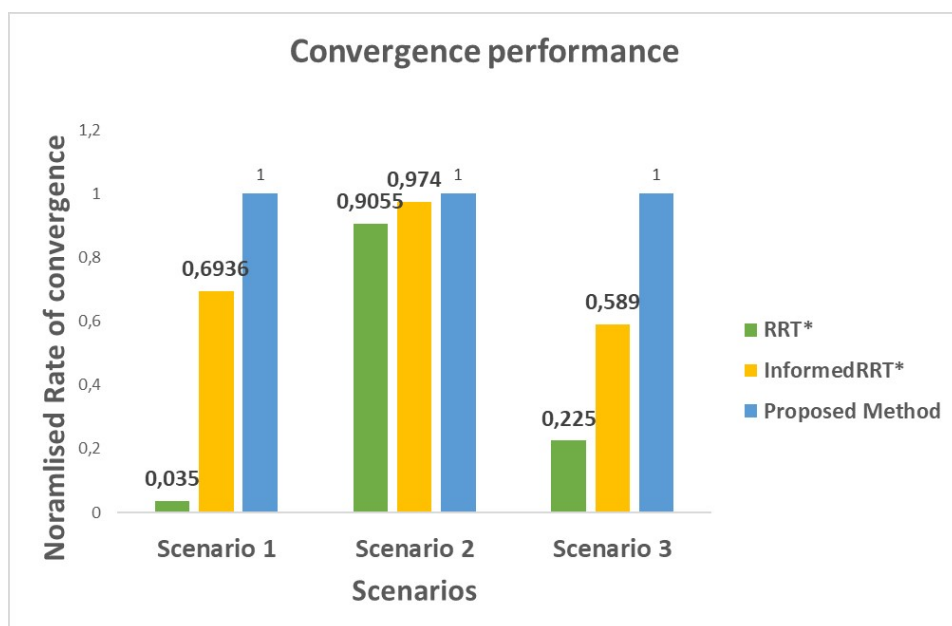


Figure 4.11: Comparison of rate of convergence

Conclusion and Future work

This chapter concludes the thesis. Section 1 recalls the objectives that had been set initially and goes over the important **contributions** of the work. Section 2 guides the readers through the identified prospective future study that can be conducted in light of the **limits in the current framework**.

5.1. Conclusions

Path planning of an agent involves generating a path from the agent's initial state to its goal state while not colliding with obstruction in the configuration space. Optimal path planning is finding a similar path while optimizing around a cost function like the shortest distance which is considered here. In Chapter 2 it was seen that there are many methods to approach this problem of which sampling based methods come out on top due to their flexibility and speed.

In sampling based methods, many methods focused on solving the path planning problem while planning around the obstruction, while other methods focused on converging to optimal solutions. Also most of the research was conducted on holonomic agents. Keeping this in mind objectives were set for this thesis in chapter 1 which would introduce an algorithm that have the following features:

- An algorithm that would efficiently sample in heterogenous environments.
- An algorithm that would also focus on the convergence of the solution.
- With a realizable model that would imitate basic vehicle motion.

This inspired the use of a hybrid approach that would combine state of the art methods. Adaptive RRT and informed RRT* were the methods that were selected to use for the hybrid approach. Informed RRT* provided exploratory benefits by converging sampling space around the initial solution with the construction of an ellipsoid. Adaptive RRT provided the adaptability around obstacles with the help of a two step selection process. It selected a growth strategy that depended on the accessibility of a state and a learning process that accounted for the performance of a growth strategy with respect to the cost of using it. The

motion model used was Dubins car model that considered vehicle turning radius to construct curves that would help the agent reach from one state to another.

This thesis provided a combined approach to optimal planning in heterogeneous environments. The method was evaluated via simulation carried out in Matlab with three different scenarios with varying complexities. The method was compared to the state of the art methods RRT* in terms of finding an initial solution, and with RRT* and Informed RRT* in terms of finding an optimal solution for a fixed number of samples. In the initial solution performance, the sampling efficiency along with the time to obtain an initial solution was compared and it was found that there was almost **50% improvement compared to RRT*** in both the cases on an average. For the convergence performance the combined approach was compared with both Informed RRT* and RRT* as both the methods work on converging the solution to a more optimal one. It was found that the combined approach was able to obtain higher rate of convergence compared to other methods. This clearly tells us that the use of such hybrid methods could be a key to solving path planning problems in heterogeneous environments. This combined approach is one of a kind method out there compared to the literature that is focusing on either one of the problems at hand and could be the key to urban path planning of autonomous vehicles. It encourages researchers to dive into the open problem of optimal path planning in heterogeneous environments which was identified in this thesis. The next section paves the path to research advances that could help solve this problem at a higher level of difficulty.

5.2. Future work

As seen in previous section, the combined approach proposed in thesis opens up a line of research and there are many ways to improve the path planning algorithm such that it would be more reliable and flexible. The path planning in this thesis is done offline with static obstacles, progressing towards online planning with dynamic obstacles is the next step. Such an implementation can be done via the same approach but, would require tweaks in the model, algorithms and the hardware.

The model currently used although easy to implement, only considers limited curves for the vehicle. In order to obtain shorter routes, a steering function would be required which could be obtained using a kinematic model that can perform relatively in terms of generating a path between two states. Including velocity and acceleration would make it more suitable for being implementable on real hardware. It should be noted that involving more parameters would increase the complexity. Hence a better computer would be required to compute those calculation to be able to generate paths real time for vehicle moving at a certain velocity.

Adaptive RRT used for planning around obstacles needs to be fine tuned depending on performance to obtain the best performance out of it. Modifying the learning approach with updated learning approaches can help in eliminating the process of offline tuning so it becomes viable for real time planning over offline planning.

Informed RRT* used for the convergence of the solution is dependent on the initial solution and focuses on finding a reliable solution in the shrinking subset around the optimal solution, this can lead to finding a solution that may be locally optimal. An extension to Informed RRT*, Batch Informed Trees (BIT*) is introduced in [31], focuses searching without requiring an initial solution by incrementally increasing the search subset.

Additionally, accounting for the perception layer which is ideal here would also make the simulation more realistic so that these methods can be evaluated and scaled to real time simulation. Involving higher level concepts like path following and path generation for human comfort would also be a reasonable research spectrum to bring urban path planning to a reality.

Bibliography

- [1] Motion planning for autonomous driving. URL http://wavelab.uwaterloo.ca/?weblizar_portfolio=motion-planning-for-autonomous-driving.
- [2] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. *arXiv preprint arXiv:1404.2334*, 2014.
- [3] Ravi Shanker. Autonomous cars: The future is now. URL <https://www.morganstanley.com/articles/autonomous-cars-the-future-is-now>.
- [4] The man who tested the first driverless car in 1925 had a bizarre feud with harry houdini. <https://jalopnik.com/the-man-who-tested-the-first-driverless-car-in-1925-had-1792312207>. (Accessed on 12/19/2018).
- [5] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The 2005 DARPA grand challenge: the great robot race*, volume 36. Springer, 2007.
- [6] Nguyen B Truong, Gyu Myoung Lee, and Yacine Ghamri-Doudane. Software defined networking-based vehicular adhoc network with fog computing. In *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, pages 1202–1207. IEEE, 2015.
- [7] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- [8] John H Reif. Complexity of the mover’s problem and generalizations. In *Foundations of Computer Science, 1979., 20th Annual Symposium on*, pages 421–427. IEEE, 1979.
- [9] Sylvain Lazard, John Reif, and Hongyan Wang. The complexity of the two dimensional curvatureconstrained shortest-path problem. In *Proceedings of the Third International Workshop on the Algorithmic Foundations of Robotics, (Houston, Texas, USA)*, pages 49–57. Citeseer, 1998.
- [10] Joga Dharma Setiawan, Mochamad Safarudin, and Amrik Singh. Modeling, simulation and validation of 14 dof full vehicle model. In *Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME), 2009 International Conference on*, pages 1–6. IEEE, 2009.
- [11] Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics*, 79(3):497–516, 1957.

- [12] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knoppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb. Making berth drive—an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6(2):8–20, Summer 2014. ISSN 1939-1390. doi: 10.1109/MITS.2014.2306552.
- [13] Steven Michael. LaValle. *Planning algorithms*. Cambridge University Press, 2014.
- [14] Ashley Trent, Ram Venkataraman, and David Doman. Trajectory generation using a modified simple shooting method. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 4, pages 2723–2729. IEEE, 2004.
- [15] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [16] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- [17] Steven M LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998.
- [18] Tobias Kunz and Mike Stilman. Kinodynamic rrts with fixed time step and best-input extension are not probabilistically complete. In *Algorithmic foundations of robotics XI*, pages 233–244. Springer, 2015.
- [19] Sertac Karaman and Emilio Frazzoli. Optimal kinodynamic motion planning using incremental sampling-based methods. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 7681–7687. IEEE, 2010.
- [20] Liangjun Zhang and Dinesh Manocha. An efficient retraction-based rrt planner. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3743–3750. IEEE, 2008.
- [21] Léonard Jaillet, Judy Hoffman, Jur Van den Berg, Pieter Abbeel, Josep M Porta, and Ken Goldberg. Eg-rrt: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2646–2652. IEEE, 2011.
- [22] Alexander Shkolnik, Matthew Walter, and Russ Tedrake. Reachability-guided sampling for planning under differential constraints. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2859–2865. IEEE, 2009.
- [23] Peng Cheng. Sampling-based motion planning with differential constraints. Technical report, 2005.
- [24] Jory Denny, Marco Morales, Samuel Rodriguez, and Nancy M Amato. Adapting rrt growth for heterogeneous environments. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1772–1778. IEEE, 2013.

- [25] Jauwairia Nasir, Fahad Islam, Usman Malik, Yasar Ayaz, Osman Hasan, Mushtaq Khan, and Mannan Saeed Muhammad. Rrt*-smart: A rapid convergence implementation of rrt. *International Journal of Advanced Robotic Systems*, 10(7):299, 2013.
- [26] Daqing Yi, Michael A Goodrich, and Kevin D Seppi. Homotopy-aware rrt*: Toward human-robot topological path-planning. In *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*, pages 279–286. IEEE Press, 2016.
- [27] Liang Yang, Jizhong Xiao, Juntong Qi, Liying Yang, Lei Wang, and Jianda Han. Gart: An environment-guided path planner for robots in crowded environments under kinodynamic constraints. *International Journal of Advanced Robotic Systems*, 13(6): 1729881416671111, 2016.
- [28] Samuel Rodriguez, Xinyu Tang, Jyh-Ming Lien, and Nancy M Amato. An obstacle-based rapidly-exploring random tree. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 895–900. IEEE, 2006.
- [29] David Hsu, Gildardo Sánchez-Ante, and Zheng Sun. Hybrid prm sampling with a cost-sensitive adaptive strategy. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3874–3880. IEEE, 2005.
- [30] Grace Wahba. A least squares estimate of satellite attitude. *SIAM review*, 7(3):409–409, 1965.
- [31] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Bit*: Batch informed trees for optimal sampling-based planning via dynamic programming on implicit random geometric graphs. *arXiv preprint*, 2014.