Path planning algorithm for driver assistance during complex maneuvering of articulated vehicles

Master's Thesis

in Mechanical Engineering

Rhine-Waal University of Applied Sciences

Author:

Giridhar Vitta Bukka

Supervisors:

Company (HAN-AR): Dr. Karel Kural

University (HSRW): Dr. Ronny Hartanto

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Kleve, 11.09.2020 Giridhar Vitta Bukka

Preface

The following report is the outcome of study on path planning algorithms and focused attempt to simulate CL-RRT based path planner for the purpose of driver assistance during complex docking maneuvers in a known environment with static obstacles using MATLAB products like Simulink and Stateflow. Planner is initiated with an objective to reverse dock in a docking station with realistic dimensions. This project is carried out in order to fulfill the graduation requirement of Masters of Science, Mechanical Engineering at Rhine-Waal University of Applied Sciences and was carried out during an internship at HAN Automotive Research.

Acknowledgments

I would like to thank my supervisors Dr. Karel Kural and Dr. Ronny Hartanto for their support and guidance. Dr. Kural's constructive criticism on my work in regular intervals has motivated me to speculate upon and prove my propositions. His tips during my research work kept me focused and helped me achieve the results I aimed for. Dr. Hartanto has initiated me in the direction of controller design during my academics, with his experience in Robotics, he helped me peek through the current Robotic advancements. Working with such great personalities has been invaluable in my experience and surely helps me advance as a good Researcher.

I thank Mr. Jan benders, Program Manager Control Systems at HAN University of Applied Sciences for seeing potential in my application for this thesis project. My sincere thanks to HAN - Automotive research (VISTA PROJECT) and Rhine-Waal University of Applied Sciences (ERASMUS Program) for supporting me financially.

Last but not least, I would like to thank my family and friends for providing the necessary support for the completion of this Master thesis project.

Abstract

In the scope of this thesis work motion planning algorithms are studied briefly and one such algorithm called Closed-Loop Rapidly Exploring Random Tree (CL-RRT) is studied extensively in the context of performing docking maneuvers for articulated vehicles. This planner is experimented with and successfully implemented in MATLAB - Simulink and Stateflow environment. The implemented planner is specifically tuned for reverse docking maneuvers of Single articulated vehicle in predefined 2D map modelled in MATLAB with reference from a Distribution center in Netherlands. Only the Kinematics of the vehicle is considered as the work aims to aid the driver in achieving complex maneuvers at slow speed of 1m/s with realistic max steering rate at 15 degrees/sec , making the dynamics of the vehicle less relevant in influencing the actual path traversed by the driver while attempting to track the planned path.

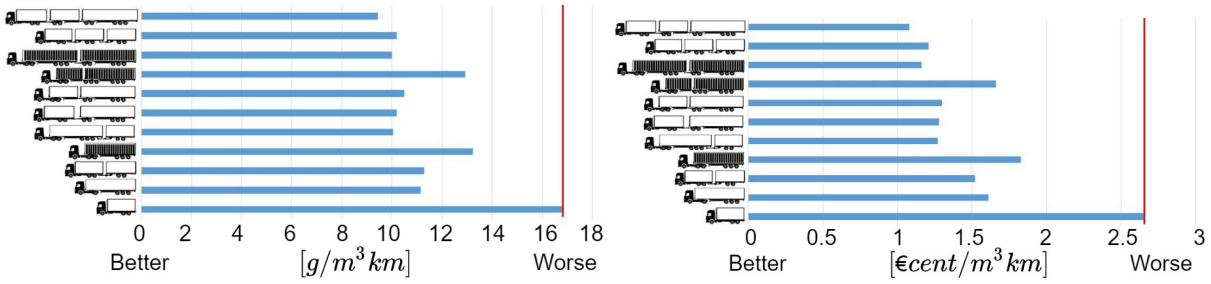
A Pure-Pursuit Bi-Directional kinematic controller based on the virtual tractor concept is used as the controller in the loop for the CL-RRT algorithm. State of the art CL-RRT components are presented followed by experimental results that discuss suitability of the controller to be included within the CL-RRT algorithm. Details on how the steering rate is controlled are provided. Flowcharts and Simulink Stateflow diagrams of the algorithm used for implementation are presented. Demonstration of the obtained solution paths is made followed by validation method using a path-following controller with maximum tracking error close to 0.1m. Finally conclusions are drawn and recommendations are made for further improvement.

Contents

1. Introduction

1.1. Background

Motion planning algorithms are not new to the automotive industry. For almost half a century they were employed in manufacturing units. Their scope of application has drastically increased in last decades with demand for efficient, safe, sustainable, reliable, autonomous and man-driven mobility solutions. Especially for freight mobility, HCV (High capacity vehicle) combinations are advocated for their contribution in reduction of CO_2 emission and Total costs of ownership in [1], see Figure 1.1. Total cost of ownership includes various operational costs specifically employee training costs, docking station damage costs and fuel costs which can be reduced by having a driver assistance system in place.



(a) Carbon dioxide emissions per volume distance

(b) Total cost of ownership

Figure 1.1.: Comparison between various vehicle types in terms of CO_2 emissions and Total cost of ownership [1].

To promote HCVs in European operational conditions without incurring any significant operational costs, an assistance system is proposed to aid the driver in performing the complex docking maneuvers in the scope of project VISTA, co-financed by the European Union via the INTERREG Deutschland-Nederland program.

VISTA stands for Vision Supported Truck docking Assistant. Once the Articulated vehicle is localized using computer vision techniques and its own position in the given distribution center map is known a path has to be planned between initial and final configurations of the vehicle see Figure 1.2. Various types of planning algorithms are published and discussed in the literature. A popular category of Planning algorithms, termed as motion planning algorithms in the space of mobile robotics and autonomous vehicles are well known for this application. Motion planning can be further classified into path planning and trajectory planning based on temporal dependence of the algorithm. In an attempt to find the best suitable planner or combination of planners HAN - AR research team is investigating motion planning algorithms suitable for Articulated vehicle docking maneuvers.

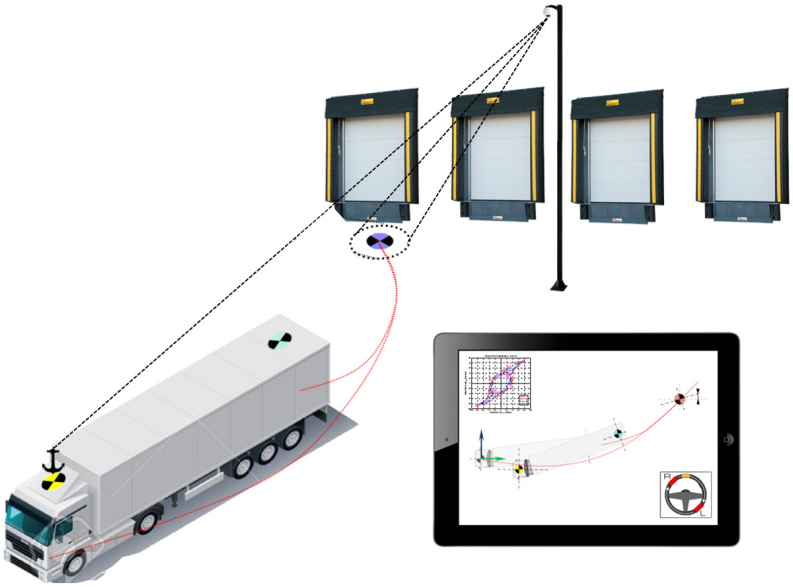


Figure 1.2.: Vision Supported Truck docking Assistant concept overview [2]

1.2. Motivation

There is noticeable rise in research and development that aims in bringing the autonomy level 3,4 and 5 to the roads, although it should be remembered that the shift will not be sudden across the industry. Research by McKinsey Center for Future Mobility (MCFM), points out that full autonomy in freight transport is long way off and estimates that the transformation will come in four waves see Figure1.3. In reality there exists a large fleet of articulated vehicles on ground which are man driven and cannot be dismissed or updated profitably.

High capacity articulated vehicles maneuvering in distribution centers is challenging and costlier. The problems faced by the driver as employee performing complex maneuvers and Logistic company as employer bearing OP-EX costs remain significant. In Figure1.3, it can be seen that the estimation shows significant improvement in Total cost of Ownership savings with increase in autonomy with time. In this context Driver assistance system backed by motion planning for Human - Driven HCVs will be significant for coming years in reducing damage costs, driver training costs and large unused open spaces in Distribution centers. And will continue to aid the driver within the city and distribution center when interstate complete autonomy for HCVs becomes functional. In [3] a framework is proposed where the human driver is being instructed by haptic or visual interface, on how the tractor should be controlled in order to bring the semi-trailer to the desired position and orientation at the docking gate. To achieve this, a suitable motion planner has to be integrated along with the localization and motion control components.

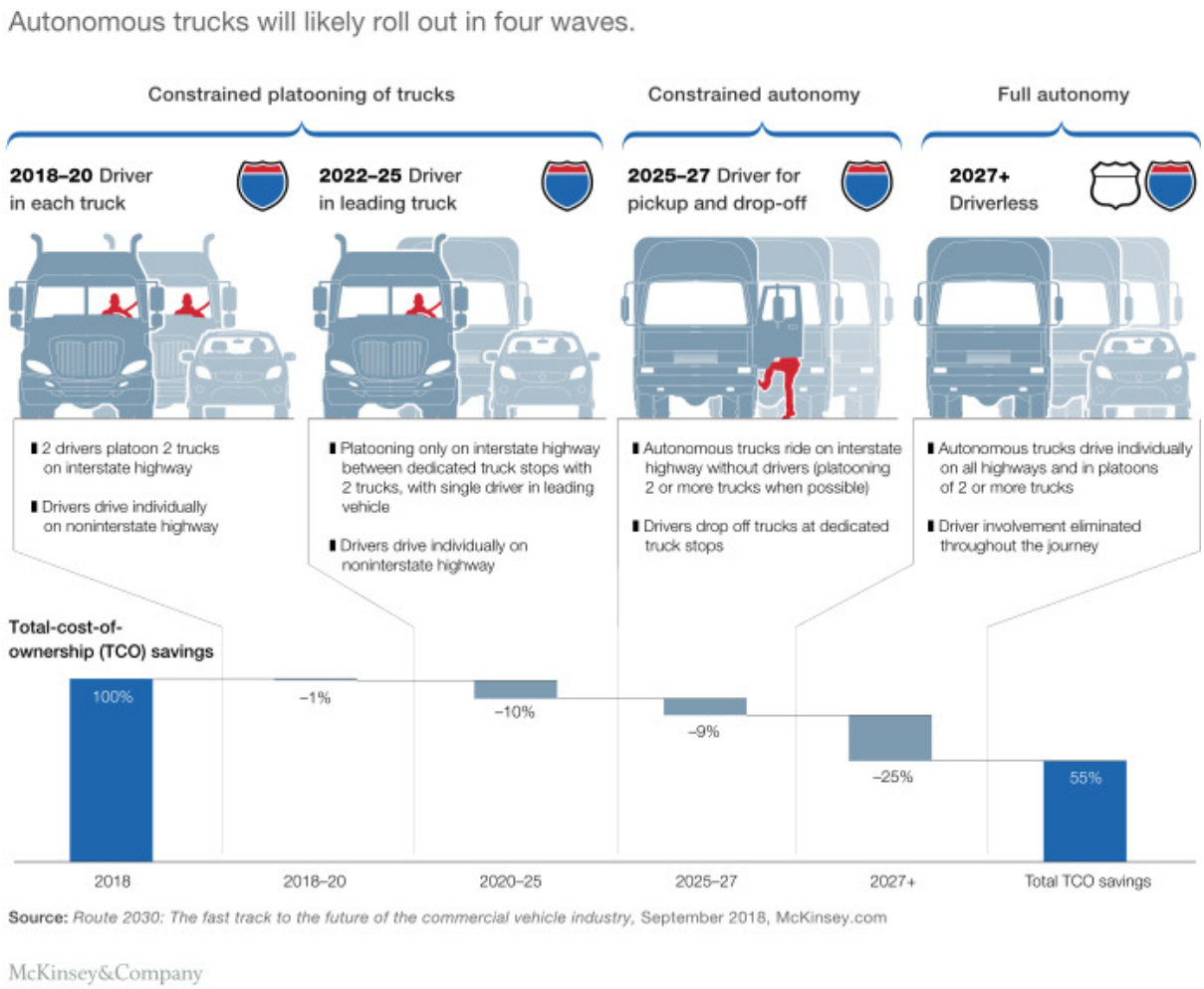


Figure 1.3.: Transformation in Four waves [4]

1.3. Problem definition

During the course of research work conducted for project VISTA, graph search methods like Dubins method and A* algorithm were studied and implemented at HAN Automotive research. These planners are limited to search only over the set of paths that are built from so called motion primitives and may fail to produce a feasible path or produce a noticeably suboptimal path [5]. In an attempt to find suitable motion planner with objective of performing reverse docking with quality path generation, CL-RRT, an incremental random sampling based search technique, an variant of RRT developed my MIT for DARPA challenge will be studied and experimented with in the context of articulated vehicle.

1.4. Project objectives

Conduct literature review on motion planning algorithms in general and extensively on CL-RRT based motion planner. To produce a MATLAB/Simulink executable of a motion planning solution based on CL-RRT while making use of available research resources from HAN - AR.

1.5. Thesis outline

This thesis is organized in chapters with the following contents:

- chapter 2, briefly studies the motion planning problem based on variety of factors and differences, the state of the art motion planning algorithms address. General motion planning components are introduced followed by discussion on state of the art CL-RRT planner. Then citation of the state of the art scientific publications related to the VISTA project, CL-RRT motion planner are presented with relevance to this thesis.
- chapter 3, elaborates the various components of CL-RRT algorithm used for this thesis in detail. Introduces the nomenclature specific to CL-RRT motion planner.
- chapter 4, presents details on type of the controller that can be included in the CL-RRT motion planner by presenting some experimental results. Then takes through the main steps involved in implementing CL-RRT algorithm by presenting flowcharts and Simulink Stateflow diagrams. A validation method is proposed and obtained paths are validated.
- chapter 5, demonstrates the results obtained and validates the solution paths based on demands by a driver assistance system.
- chapter 6, summarizes the main contributions of this thesis and recommendations for improvement are made.

2. Literature review

Computing a safe, comfortable, feasible and possibly optimal path from current to goal configuration while obeying the vehicle and surrounding constraints is the task of a motion planner. The popular perceive, plan and act cycle illustrated in the Figure 2.1 shows the place of a motion planner among the other important components of an autonomous vehicle or Driver assistance equipped vehicle. In recent years the need to aid man driven vehicles during complex maneuvering at low speeds has gained attention. Depending on the situational context of solving the planning problem various types of motion planning algorithms are published in the literature. In the Figure 2.1 the behavioral layer has a list of possible behaviours to be selected from as commanded by the the prior component in hierarchy within or from outside of the high level planner. Docking maneuver, lane change, lane following and overtaking are few behaviours that are decided based on the situational context. In this thesis a motion planner is developed aiming to address a specific behavior i.e, Docking maneuver, which can get challenging depending on the constraints from surroundings or the vehicle articulation.

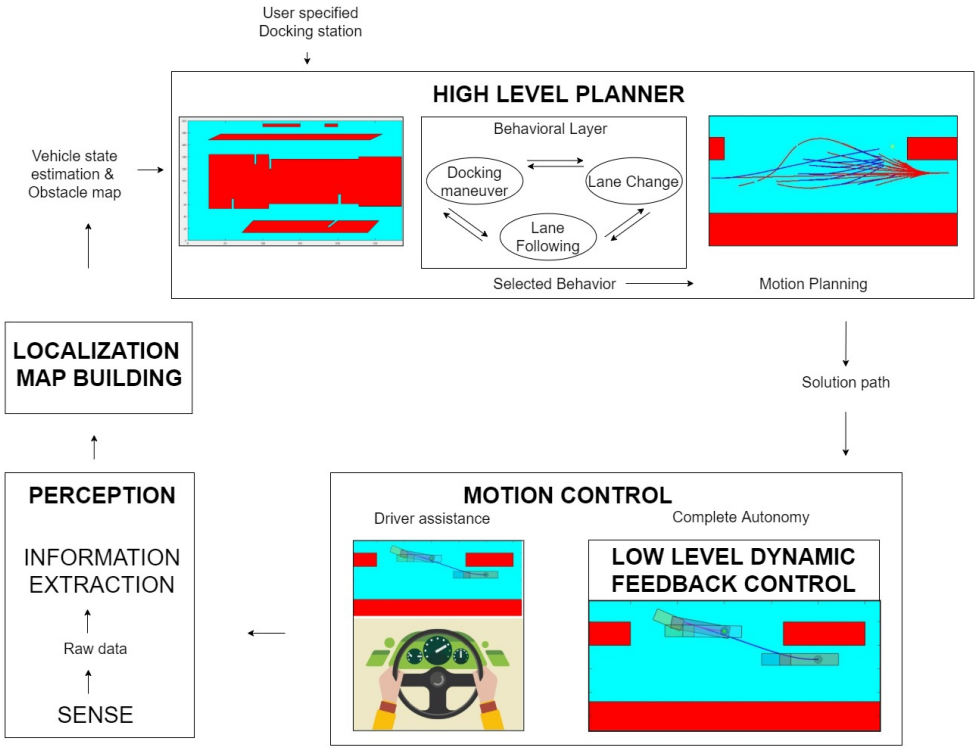


Figure 2.1.: Illustration of Motion planning and Driver assistance system's place in Perceive, Plan and Act control scheme

2.1. The motion planning problem

Variety of motion planning algorithms are found in the literature that solve the motion planning problem. In this section an attempt to highlight their differences is made. An overview of motion planning algorithm related classification found in the literature is illustrated in the Figure 2.2.

2.1.1. Based on time dependency ...

Depending on the requirement of temporal dependence of the vehicle state, motion planning algorithms can be a path planning or trajectory planning algorithm. A trajectory planning algorithm can be generalized to path planning algorithm by considering constant unit time step. Dynamics of the vehicle configuration comes into play while planning a trajectory. Where as path planning considers only the kinematics of the vehicle configuration. In literature applications with low speed maneuvering are solved with path planning by assuming minimal influence of vehicle dynamics on the solution path. In this thesis this is the aspect we explore.

2.1.2. Based on how the motion planning problem is addressed ...

Variational methods approach the planning problem by Solving classical optimization problem.They solve planning problem by performing global search in the discretized version of the path space. Based on how it is solved these methods are further categorized into direct and indirect methods.

Direct methods define the dynamics of the system using non-linear system of equations, the constraints using non-linear inequalities and solve the problem discretely at collocation points.

The State space description of a vehicle for which motion planning problem has to be solved can be represented as in the Equation 2.1.

$$\dot{x}(t) = f(x(t),\ u(t)), \quad x(0) = x_0 \tag{2.1}$$

where, x denotes the states of the system, u denotes the control inputs and x_0 the initial state configuration at time t = 0.

$u(t) \in U$ denotes the input constraints, $x(t) \in C_{free}$ denotes constraints on state configuration. While satisfying the demand that the final configuration $C(t_F) \in C_{goal}$ the following expressions 2.2, represent the motion planning problem when formulated as a typical optimal control problem where an objective function has to be minimized subject to certain constraints that dictate the system behaviour.

$$\begin{aligned} &\textit{minimize } \mathbf{u} && \int_{t_f}^{0} \Gamma(C_{free}(t),\ x(t))dt \ +\ \Phi(x(t_f),\ C_{goal}) \\ &\textit{subject to} && \begin{cases} \dot{x}(t) = f(x(t),\ u(t)),\ x(0) = x_0,\ x(t_f) \in C_{goal} \\ x(t) \in C_{free}(t),\ u(t) \in U \end{cases} \end{aligned} \tag{2.2}$$

where Γ is a penalty of the configuration during the motion and Φ is a penalty on final configuration. This problem is constrained optimization problem and is generally solved with complex numerical methods.

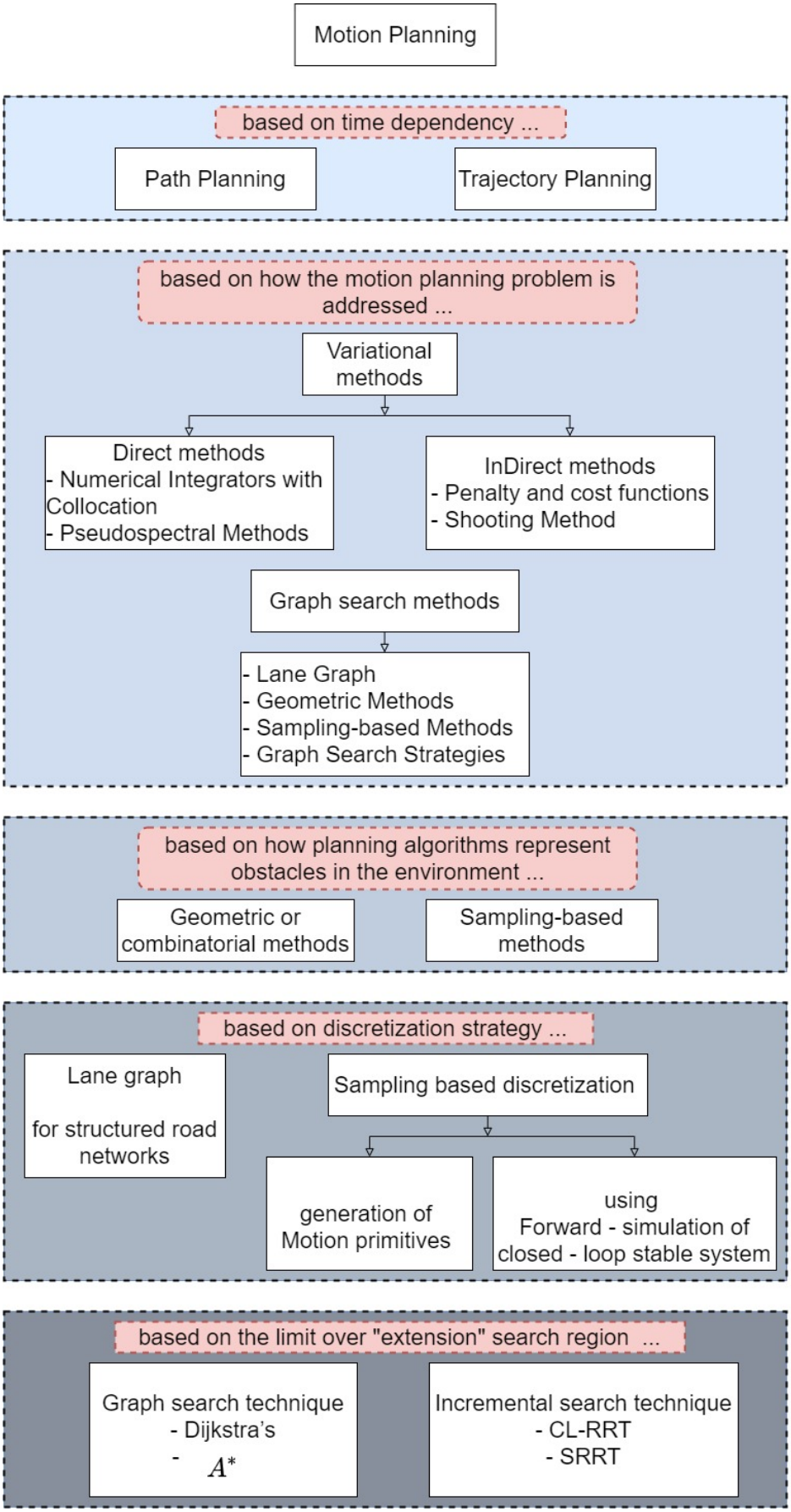


Figure 2.2.: Overview of the range of motion planning algorithm related classification found in the literature

.

In simple words solution of the optimization problem at discrete time steps is obtained. Then using basis functions interpolation between collocation points is made to obtain approximate trajectory. For example a simple piece wise linear basis together with collocation points are used to approximate trajectory using numerical integration methods like Euler or Runga-Kutta. Depending on the choice of more sophisticated basis functions for interpolation between collocation points another category within the direct variational methods exist under the name Pseudospectral methods.

This is an example of implementing variational method approach where the above problem can be addressed either using the *Direct method* or *indirect method*. Where Indirect method addresses the above problem by considering the solution first and then defines the constraints on the problem that converge to the solution. One popular example in literature is *shooting method*.

Due to the limitation of *Variational methods* by their convergence to only local minima these methods are applicable to certain contexts only. And in literature more practical approaches are available to solve the Docking maneuver problem of the articulated vehicles which are of interest for this thesis. The next section introduces these methods in detail.

2.1.3. Graph search methods

The graph search methods discretize the configuration space C of the vehicle and represent it in the form of a graph followed by searching for a path on the graph that has minimum cost. Within the graph search methods depending on how the planner represents the obstacles two broad categories of planners are available.

In the *first category*, Geometric or combinatorial methods descretization of C_{free} is performed. But C_{free} needs to be known before hand and usually in autonomous applications it is not always readily available. Example for such methods are vertical cell decomposition, generalized Voronoi diagrams and visibility graphs.

In the *second category*, sampling based methods on the other hand do not need to have the information of the C_{free}.

In real deployment scenarios specially for autonomous driving C_{free} is not available and is too costly to construct from raw sensoric data. Requirements on the resulting Path planned in the constructed C_{free} are beyond just a minimum curvature constraint. This explains the popularity of the sampling based methods. Instead of reasoning over a geometric representation, the sampling based methods explore the reachability of the free configuration space C_{free} using steering and collision checking routines [5].

In sampling based graph search methods two consecutive steps are to be followed to solve the planning problem.

The first step being creation of the discrete segments that are subject to kinematic constraints of the vehicle. This is achieved in few algorithms for example A^* by generating a motion primitive bank, where given two vehicle configurations a path is generated that is subjected to the kinematic constraints of the vehicle. Such paths for various configurations in the vicinity of the vehicle are pre-calculated and saved. Another approach which doesn't limit itself to the pre-calculated bank is by forward simulating the vehicle to any point in the vicinity of the vehicle. This method is applied in the planning algorithms like CL-RRT, and Spline-based Rapidly exploring Random Tree (SRRT).

The second step being searching for the proper sequence of the discrete path segments that will lead to the goal position starting from the initial position.

Search algorithms like A^* performs the search within the pre-calculated motion primitive bank to find such sequence. Therefore, these techniques may fail to return a feasible path

or return a noticeably sub-optimal one. Where as incremental motion planner like CL-RRT strive to build increasingly finer discretization of the configuration space while concurrently attempting to determine if a path from initial configuration to the goal region exists by discretization at each step [5].

2.1.4. Incremental search technique

Incremental search technique based planners like CL-RRT generate feasible path to any motion planning problem instance, if one exists, given enough computation time. If the scenario to solve is "easy", the solution is provided quickly, but in general the computation time can be unbounded [5]. Similarly, incremental optimal motion planning approaches on top of finding a feasible path fast attempt to provide a sequence of solutions of increasing quality that converges to an optimal path. The term probabilistically complete is used in the literature to describe the CL-RRT algorithm.

2.1.5. Method used for this thesis

Given the Driver assistance system frame work where a steering controller already available and generates steering angle inputs needed by the subsystems of the assistance system to aid the driver in docking maneuvering the incremental search technique based motion planner CL-RRT is a promising solution to generate solution paths that already includes the characteristics of the steering controller.

The research presented in in [6] is of great interest as the motion planning problem is discussed in the context of performing docking maneuvers of articulated vehicles. And is proposed to be solved using CL-RRT algorithm. As this technique gives a possibility of directly linking the steering controller characteristics into the loop while planning the solution paths this method is adopted in this thesis to solve the motion planning for docking maneuver problem.

Before getting into the details of motion planning ingredients, it is important to visualize its place in the complete system. The classic perceive, plan and act control scheme also applies to the current problem of aiding drivers in complex docking maneuvers. The Figure 2.1 illustrates the place of the motion planner and a driver assistance system in the framework of perceive, plan and act control scheme. The following sections introduce the important ingredients of a motion planner.

2.2. Motion Planning Components

Below listed are the components that generally constitute a motion planning algorithm in the context of articulated vehicles.

2.2.1. Vehicle model

An articulated vehicle has further options to be considered as discussed below,

SAV or Multi Articulated Vehicle (MAV)?

A SAV has single articulated unit. In this thesis SAV configuration will be used for implementing motion planning algorithm. There are various module combinations that can make up a SAV as shown in Figure 2.3.

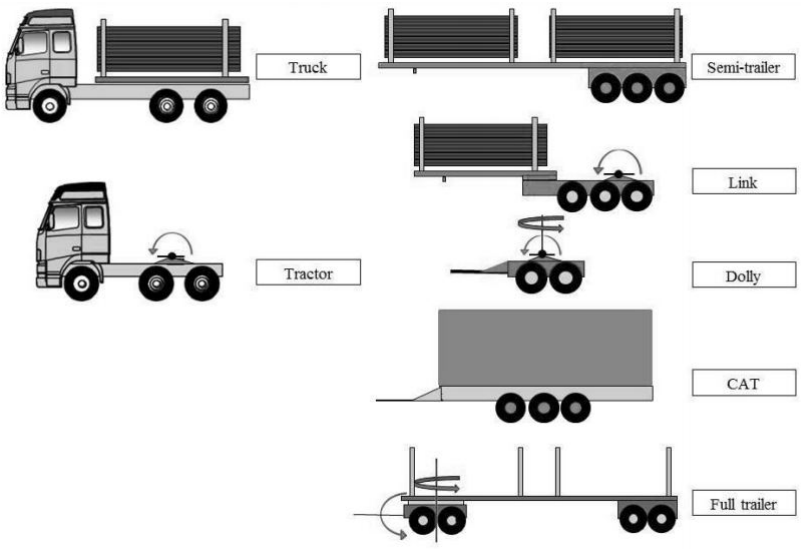


Figure 2.3.: Individual modules that constitute a SAV

.

Kinematic or Dynamic model?
Using a Mathematical model of the vehicle is a standard approach while building a motion planning algorithm. The mathematical representation of the SAV's can be classified into single track and double track model. vehicle is represented as a mathematical model based on its Kinematic constraints and dynamic properties. A suitable vehicle model is selected depending on factors like.,

- Speed
- Terrain
- Solution type

Table 2.1.: Kinematic Model (KM) or Dynamic Model (DM) ?

Low Speed	**High Speed**	**Flat Terrain**	**Rough Terrain**	**Path**	**Trajectory**
KM	DM	KM	DM	KM	DM

As shown in the table 2.1, as the operation speeds increase and terrain becomes uneven aspects of the vehicle design like drive-train and braking, suspension and steering, distribution of mass, aerodynamics and tires affect the vehicle dynamics and hence should be naturally taken into account for appropriate representation of the vehicle as a mathematical model for simulations.
It is proven in [7] that Wheeled Mobile Robot (WMR)'s on flat surfaces and under lateral acceleration of 0.2 can be reasonably represented with their kinematic models. This result can be applied to vehicles performing maneuvers at low speeds on flat surfaces similar to docking maneuvers in Distribution Center (DC)'s. Also in [8] comparison between kinematic and multi-body model of SAV shows the closeness of these representations at low speed maneuvering.

The thesis work deals with low speed vehicle maneuvering where influence of dynamics is not significant. Hence we use a Single-Track model that perfectly represents kinematics of the SAV as shown in Figure 2.5.

prime mover and trailer selection As shown in the Figure 2.3 various prime mover and trailer modules are available. Kinematic model varies with varying combination of modules.

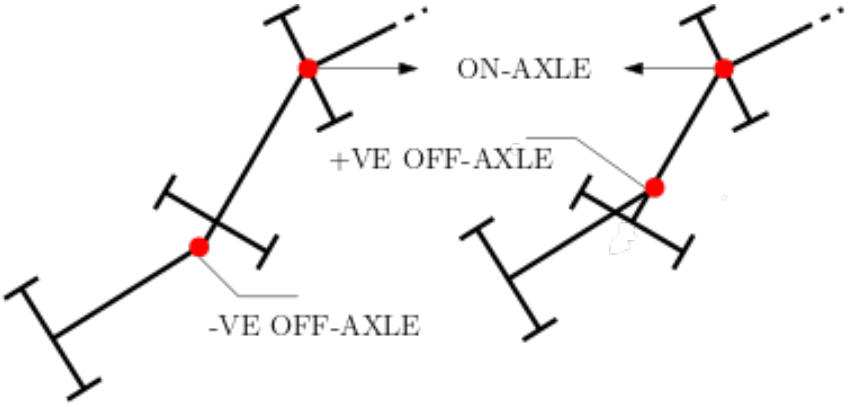


Figure 2.4.: Illustration of On-axle and Off-axle hitching

.

On-axle or Off axle? Depending on nature of hitching that connect various units of articulated vehicle, where the trailer is hitched on the tractor, two types of hitching possibilities are popular. On-axle hitching and Off-axle hitching.
Off-axle hitching can be positive off-axle or negative off-axle as shown in Figure 2.4.

This thesis work limits itself to positive off-axle, Kinematic model of SAV with dimensions as mentioned in Table 4.1.

The reference point taken for the kinematic modelling is the semi-trailer axle position (x_1, y_1). The steer axle position (x_{0f}, y_{0f}), The tractor drive axle position (x_0, y_0), king pin position (x_{1f}, y_{1f}) and tractor trailer velocities v_0 and v_1 can be seen in Figure 2.5. The sign conventions considered are ISO where forward x direction, left y direction and anti-clockwise angle of the vehicle is positive. Tractor wheel base, trailer wheel base and coupling overhang are denoted as L_{0f} , L_{1f} and L_{0bf} respectively. L_{0bf} is positive if the coupling point is in front of the tractor rear axle and negative if the coupling point is behind the tractor rear axle. The steering and articulation angles are denoted by δ and γ. The articulation angle is the difference between the tractor yaw angle θ_0 and the trailer yaw angle θ_1. Inputs to the model are tractor longitudinal velocity v_0 and steering angle δ.

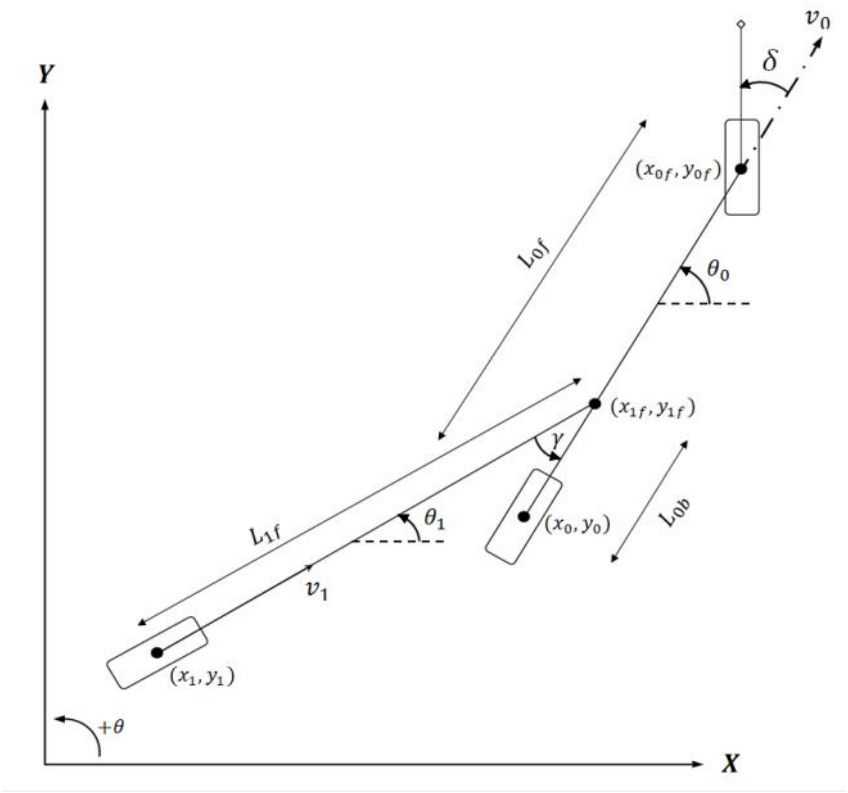


Figure 2.5.: SAV Single track Kinematic model

.

2.2.2. Configuration space (C_{space})

Set of possible transformations that could be applied to the vehicle model is called Configuration space (C_{space}) of the vehicle model. It is important to define C_{space} for the purpose of motion planning algorithms. Motion planning problem is solved by conducting search in this C_{space} and by finding a solution path eventually.

C_{space} = Obstacle space (C_{obs}) $\cup$ $Freespace(C_{free})$.

C_{obs} is the subset of the configuration space where the vehicle components (tractor or semi-trailer) collide with obstacle or they collide with each other (jackknifing).

C_{free} is the remaining space left when C_{obs} is removed from the C_{space}.

Solution path must traverse precisely through C_{free}. A motion planning algorithm must find a Solution path that traverses precisely from an initial configuration to a goal configuration in C_{free} . The Figure 2.6 illustrates motion planning problem with C_{space} ideas.

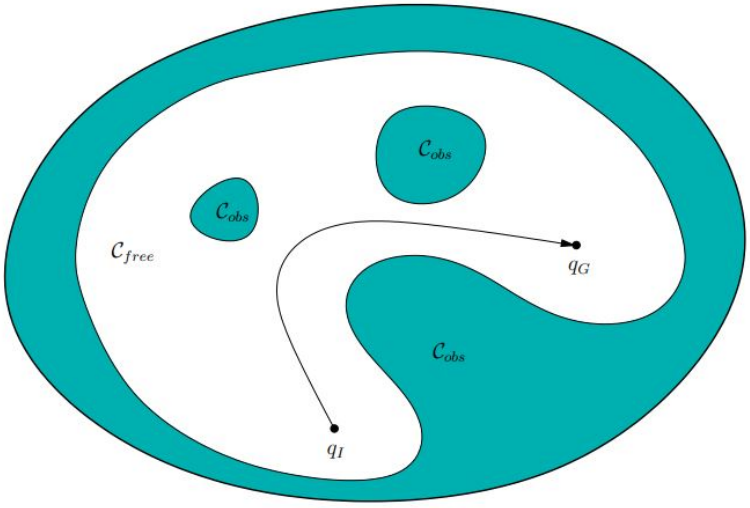


Figure 2.6.: Motion planning problem with Configuration space idea. where, q_I is initial configuration and q_G is goal configuration [9]

.

2.2.3. Test scenario Map

Test scenario Map is a geometric description of a test scenario, that defines the boundaries of the total working space, including definition of obstacles. This map forms the foundation for C_{space}. This map can be a static map with defined obstacles space or a dynamic map where the map is updated at a frequency, reflecting changes in the obstacle space. In this thesis a static map that depicts a real distribution center in size and shape is used.

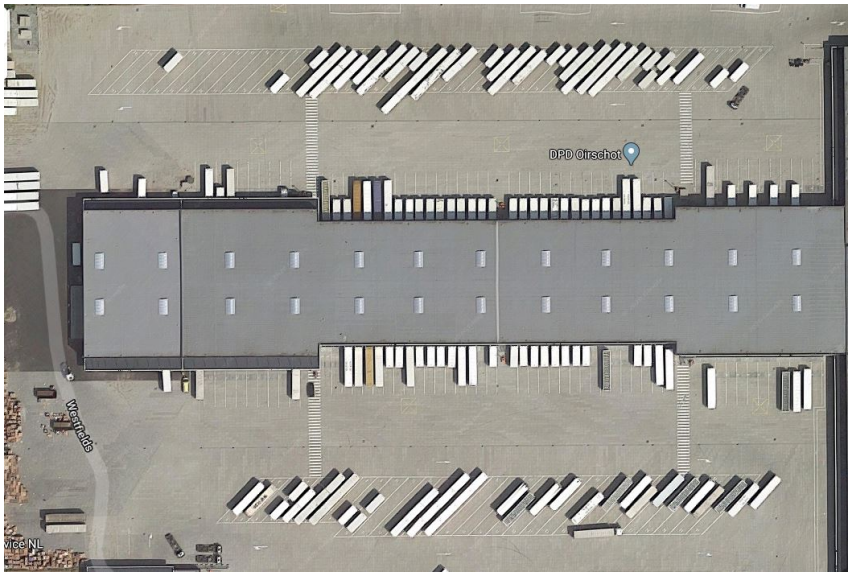


Figure 2.7.: DPD DC Oirschot, The Netherlands. Sattilite image

.

The map is used by researchers at HAN to test motion planners and hence can be a great medium for bench-marking motion planners in future research. The Figure 2.7 shows the satellite image of the DC. The model of the DC with obstacles defined with polygons is shown

in Figure 2.8.

When working with a motion planner an obstacle space plays a significant role in directing the vehicle in a desired position and orientation. The scenario was made in the same dimension as a real distribution.

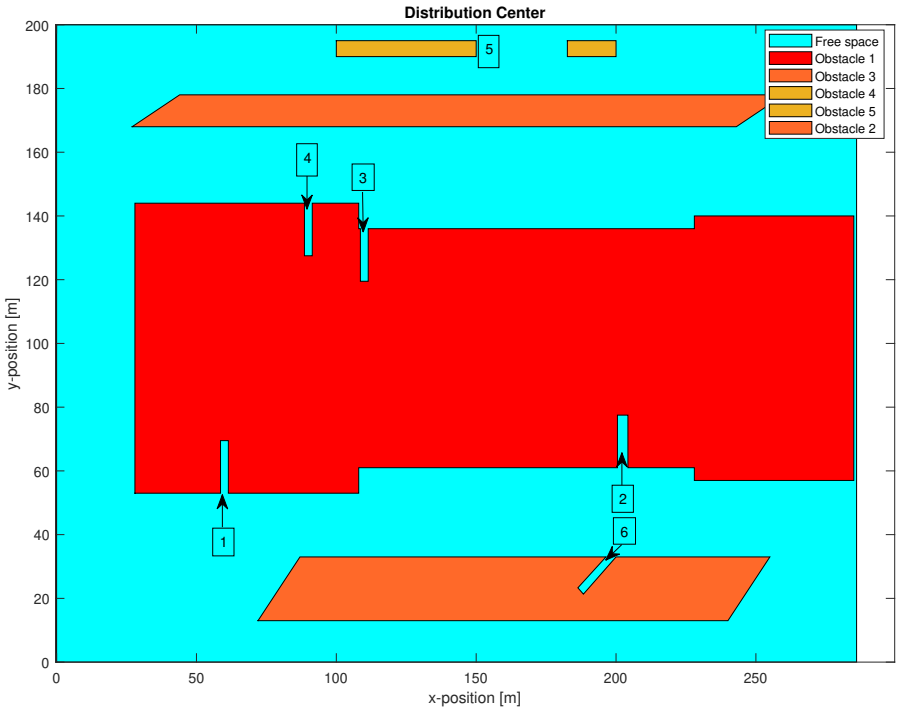


Figure 2.8.: Model of the DC with obstacles defined with polygons

.

2.2.4. Solution path quality

In this section we discuss very commonly used terms and their definitions which describe the quality of the solution path obtained by the motion planner.

The quality of the solution path is described in the literature with the terms listed as follows.

- Feasible
- Optimal
- Complete

Feasible path

The solution path that satisfies some predefined constraints without emphasizing on the quality is termed as a feasible path. Such path may or may not be realistic as the aspect of quality is not considered at all from the beginning.

Optimal path

The solution path that optimizes some quality criterion subject to given constraints is termed optimal or sub-optimal depending on the extent to which the optimal nature of the path is exhibited.

In [5] detailed explanation of these terms and the nature of the solution paths from different

motion planners is discussed.
Complete
An algorithm is considered complete if for any input it correctly reports whether there is a solution in a finite amount of time. If a solution exists, it must return one in finite time [9].

These three terms can be considered as a root words upon which depending on the extend to which they hold their nature of being feasible, optimal or compete, the following listed prefixes are added.

- Probabilistically
- Asymptotically
- Globally
- Monotonically

For example below listed are two qualities that describe the solution paths resulting from the CL-RRT motion planner ...
probabilistically complete is the term used in the literature to describe algorithms that find a solution, if one exists, with probability approaching one with increasing computation time. Note that probabilistically complete algorithm may not terminate if the solution does not exist [5].
asymptotically optimal is term used for algorithms that converge to optimal solution with probability one [5].

2.3. CLRRT state of art

CL-RRT motion planning algorithm works closely with a closed loop system and is intended to be used inside a closed loop stable system. The closed loop system in the context of autonomous vehicles generally consists of a reference path, plant or a vehicle, feedback from the sensors and observers, and a Controller. This constitutes a feedback control routine where the vehicle takes commands from a controller in order to pursue the reference paths. In real world disturbances cause the vehicle to drift away from pursuing the reference path. This is balanced by taking the input from sensors and observers into the controller design.

It is clear that the low level feedback control system which commands the vehicle actuators has its own characteristic behavior depending on which the autonomous vehicle fulfills its task like path tracking. When a reference path is generated without considering the controller characteristic behavior the probability of the controller failing to track the reference path always persists. Considering this, MIT team has developed a motion planner for the DARPA challenge that includes the characteristic behavior of the low level feedback control system of the autonomous vehicle.

In the recent years such motion planner is put to use for aiding articulated vehicles complex maneuvering. In [10] the motion planner was proposed as a driver assistance system with two options. First is a manual system where the driver can construct a kinematically feasible path which he can follow. The Second option lets the vehicle automatically perform the maneuver. For both the options a feedback controller based on cascaded control approach was implemented. Where a Linear Quadratic-based articulation angle stabilization at the low level and pure pursuit controller on higher level are used.

In literature several advantages of the CL-RRT planner over conventional RRT are discussed. Below few such advantages are listed ...

- The prediction error of the forward simulations for the closed-loop system in CL-RRT will be much lower than the prediction error of the forward simulations for the open-loop system based on conventional RRT.
- Robustness against model errors and other disturbances is observed to be better in CL-RRT [11]
- Due to simplicity in the reference paths the strategy of CL-RRT is suitable for real - time motion planning.
- Due to its general framework can be used as a motion planner for any type of system

2.4. Published work related to the Thesis topic

In the past few years Scientific papers are published on the topic, "Assistance Systems backed by motion planning for aiding drivers in complex maneuvering".

- A framework is proposed in [3], where vehicle localization, motion planning and motion control components simultaneously work together making a functional driver assistance system. where motion control of the vehicle can be autonomous or Human driven depending on the vehicle autonomy level.
  In [3] a drone is used to monitor the vehicle's interaction with docking station and the information is communicated to a device that is available for the driver to get assistance from.
- A kinematic path following controller [12] has also been developed at HAN - AR which successfully tracks reference paths. Where the controller can only track reference paths subject to certain constraints but with maximum tracking error close to 0.1m.
  The controller successfully produces vehicle traversed paths that are subjected to the complex kinematics of the articulated vehicle.
- Improvement to the existing bi-directional path following controller from [12] has been made by adding a Virtual Articulation Angle sub-system in [13].
- In [14] a cascaded controller approach which includes a Linear Quadratic (LQ)-based articulation angle stabilizing controller at low level and a pure pursuit steering controller for path tracking on high level is used to successfully perform reverse docking maneuvers in simulations and scaled Double articulated vehicle (DAV) and SAV configurations is presented.
- In [6] articulated vehicle docking was achieved with CL-RRT based motion planning algorithm. The cascaded controller from [14] was included in the loop of CL-RRT resulting in the solution paths that include characteristics of the low level cascaded controller.

  In general sampling based planner like RRT requires a method of joining the sampled points while building the tree ...

- In [6] the sampled points were joined by a simple line. Making the design of the reference path very simple and general, where as In [15] the sampled points were joined with splines.

Continuing the research flow for the project VISTA at HAN - AR, this thesis work utilizes the understanding from literature review and research resources available from above mentioned publications to develop a CL-RRT based path planner that contributes to the motion planning component of the proposed frame work in [3].

3. CL-RRT Framework and its Components

3.1. The Framework

In this chapter the Framework that holds all the components which constitute CL-RRT motion planner is presented and all the respective components are listed and explained in the context of the implemented algorithm for this thesis. The Figure 3.1 gives an overview of the framework while indicating the place of components involved and flow of information among the components in the total CL-RRT framework.

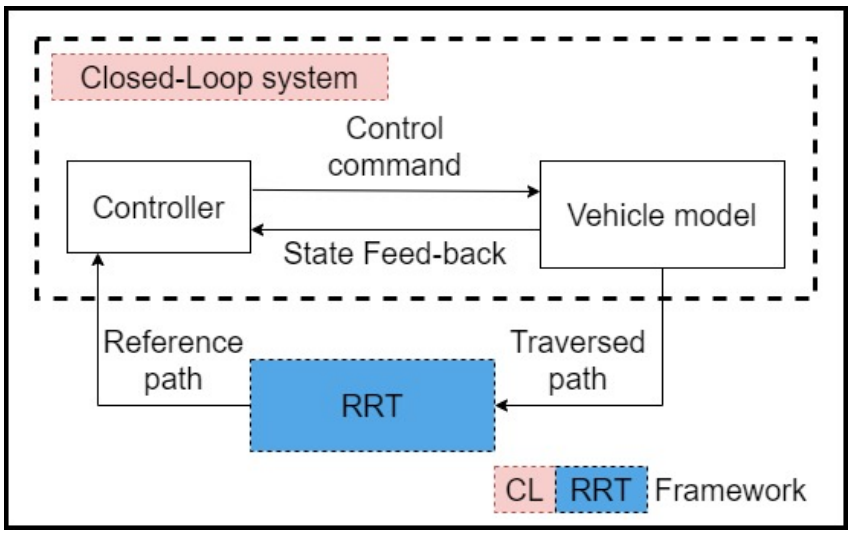


Figure 3.1.: Illustration of the CL-RRT algorithm working in the loop with controller and forward simulation model

Before getting into the details of each component within the CL-RRT framework, it is a good practice to visualize and know what system the motion planner will be part of and in what context will it be put to use.

To give an idea of the whole system an illustration in Figure 3.1 is provided. Where a driver assistance system setup and the subsystems involved are shown. The CL-RRT motion planner takes the obstacle map input and other necessary information like vehicle pose estimation to process and produces a solution path that is kinematically feasible and also close to optimal or Sub-optimal nature. The context in which this motion planner implementation is done is also of great importance. As there are various possible strategies achievable by a specific motion planner addressing the behaviour or the context, the planner is being used in. Few such behaviours are lane changing, overtaking, lane following and docking maneuvers. The CL-RRT motion planner is shown to be capable of achieving lane changing [11], path following [6] and docking maneuvering [6] behaviours in the literature. This thesis implements the docking maneuver behaviour strategy using the CL-RRT motion planner.

The obtained solution path can be taken as input by a variety of subsystems that can be used to aid a driver while performing complex maneuvering. Few such subsystems for driver support which are likely to enter the market in near future include Augumented

Reality - Heads up display (AR-HUD) [16] technology by a company called WayRay based in Switzerland and Haptic feedback system [17] developed by Biomimetics and Dexterous Manipulation Lab of Stanford University. Motion planners in combination with such driver support systems can readily contribute to the vision of project VISTA.

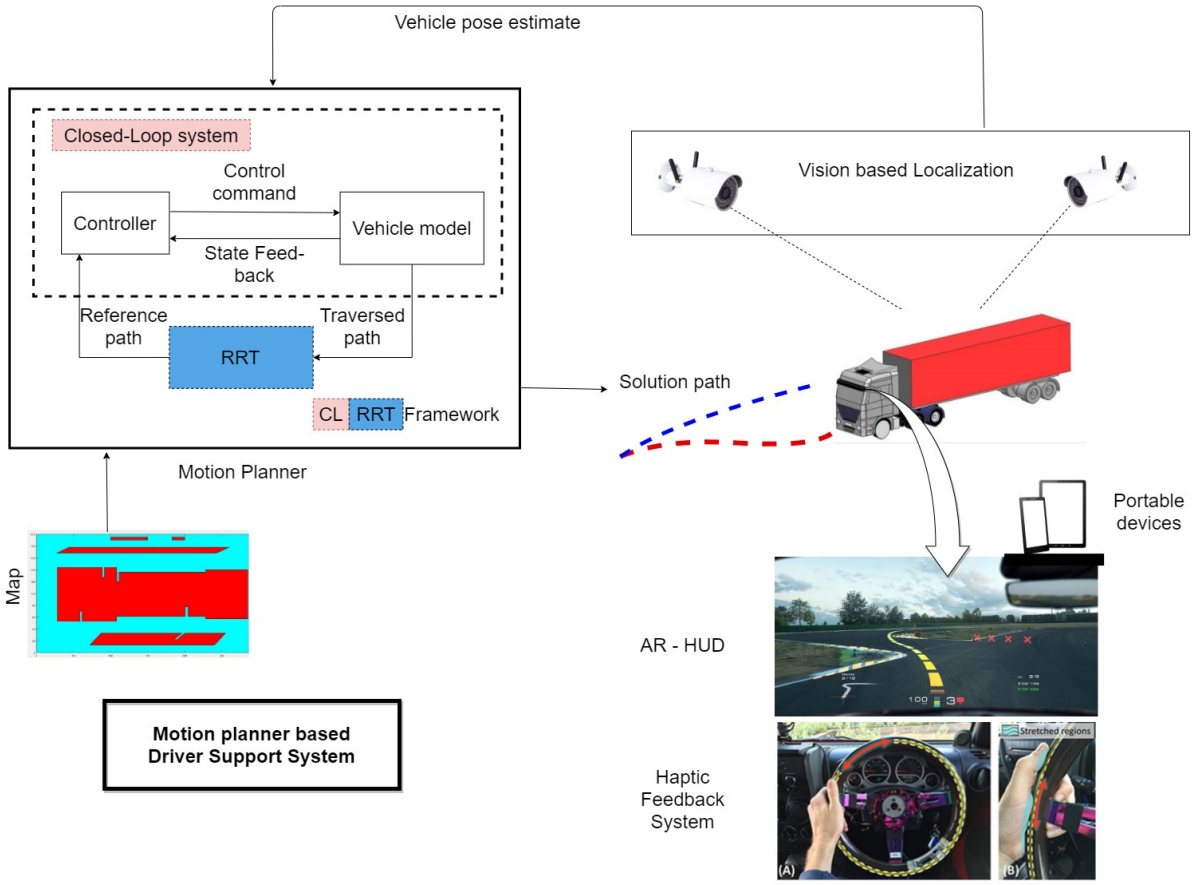


Figure 3.2.: Illustration of CL-RRT motion planner based driver support system

The following are the list of components that make up the CL-RRT framework as shown in Figure 3.1.,

- Vehicle model
- Controller
- RRT algorithm

3.2. Vehicle model

The major requirement of a CL-RRT motion planner is a vehicle model that can be forward simulated. A mathematical model of the vehicle for which motion planning problem has to be solved given the initial and the final configurations, can be considered as a vehicle model. As our main objective is to perform a docking maneuver, the vehicle model should be capable of performing forward, reverse motions alongside providing control over semitrailers positioning and orientation. In [12], a virtual tractor principle has been used because of the bidirectional maneuverability feature it offers for achieving docking maneuver. The same approach has been incorporated in the vehicle model used for the CL-RRT framework.

The vehicle model component in the Figure 3.3 shows structure of the vehicle model provided by the HAN. This Figure also illustrates the location of the controller and vehicle model within the CL-RRT framework. This vehicle model is based on virtual tractor principle. Where Inverse kinematic model and forward kinematic model of the SAV are coupled to perform two consecutive steps. The first step being kinematic inversion of the input virtual steering angle to obtain the steering angle of real tractor and the second step being, forward simulation of Kinematic model using previously generated steering angle for the real tractor that results in generation of the forward, reverse motion while positioning the semitrailer as commanded by the virtual steer angle. A steering controller is responsible for orienting the virtual tractor along a given reference path (further details of the controller are discussed in the following section). For this vehicle model input reference velocity sign decides the direction of motion.

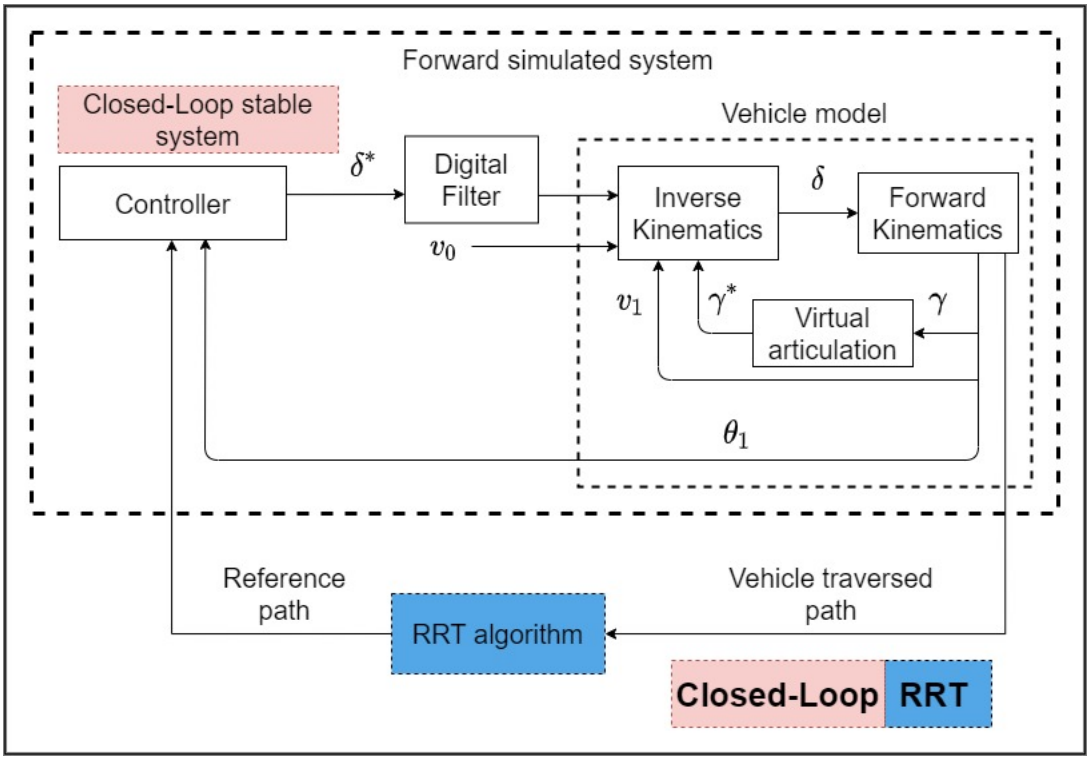


Figure 3.3.: Illustration of the RRT algorithm working in the loop with expected closed loop stable system, where δ^* is virtual steering angle, δ is actual steering angle, v_0 is reference longitudinal velocity for tractor, v_1 is reference longitudinal velocity for semitrailer(both are set to 1m/s), γ^* is virtual articulation angle, γ is actual articulation angle, θ_1 is heading angle of semitrailer.

3.3. Feedback-Controller

Feedback controller plays the role of stabilizing the vehicle along the reference path. For the purpose of the CL-RRT motion planner a controller that is capable of steering the SAV to achieve path following while reducing the lateral error is needed. As mentioned earlier this thesis aims to continue the research flow for the project VIsion Supported Truck docking Assistant (VISTA). Thereby previously developed Bi-directional path following controller [12], simulink model was provided to include within the CL-RRT algorithm. Hence initially Bi-directional path following controller was studied and experimented with attempts to include

the controller in CL-RRT framework presented in 3.1. In the following section the purpose and working principle are discussed. Followed by the discussion on the popular lateral control method, pure pursuit that has been used in the literature for the implementation of CL-RRT algorithm.

3.3.1. Bi-directional path following controller

This controller is developed with a motive of aiding a driver in performing complex reverse docking maneuvers. This was intended to be used inside a driver support system framework, where a solution path from a motion planner is taken as a input reference path and is followed closely by minimizing the lateral error. The generated steering angles obtained by forward simulating the controller-vehicle model are linked to other subsystems like AR-HUD, Haptic feedback or a portable device interface see Figure 3.2, from which the driver can seek commands to perform the complex maneuver. In [12] a path planned using a Dubins curve is tracked using the Bi-DPFC and the generated steering angle commands are sent wireless to a test vehicle which is capable of executing the steering command. The results from [12] show that the reference path was tracked with a minimal error, close to 0.1 meter. In Figure 3.4 the closed loop stable system of Bi-DPFC and vehicle model provided by HAN University of Applied Sciences (HAN) is shown

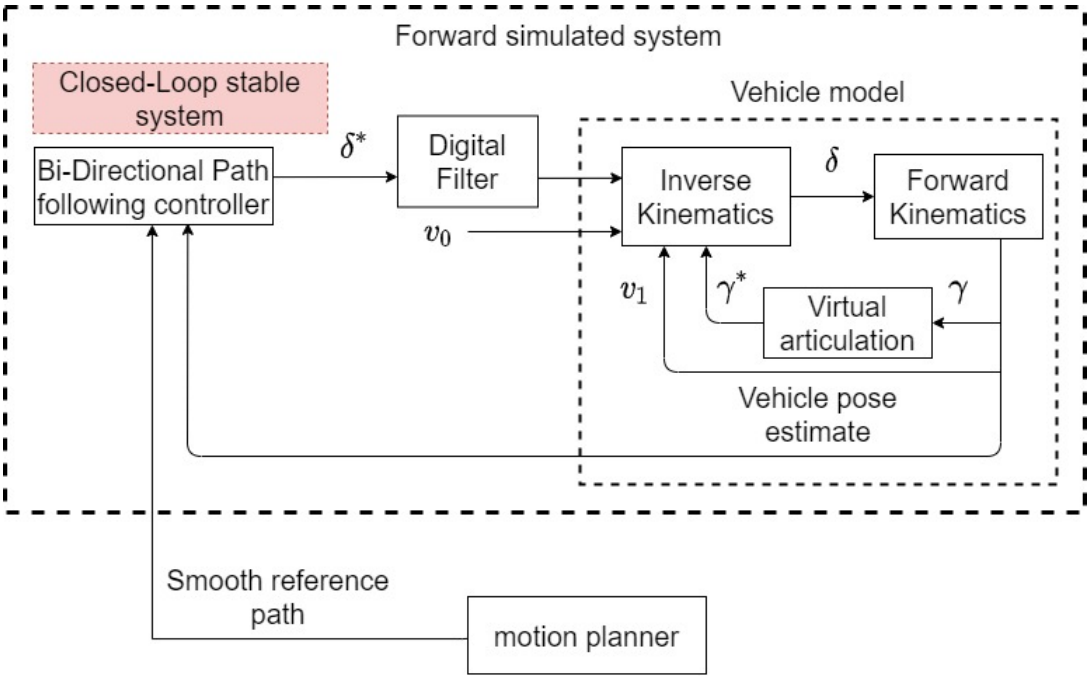


Figure 3.4.: Illustration of the Bi-DPFC and vehicle model together forming a closed loop stable system with reference paths being generated by a motion planner.

working principle The original Bi-DPFC is a PI controller with two look ahead distances where steering angle was calculated by gradually eliminating lateral cross-track error and vehicle heading error with respect to two look-ahead points. For this thesis a modification to the original Bi-DPFC has been made. Where the Integral term $K_I.\int ey1$ and the proportional term $K_\theta.e\theta$ from 3.1 have been omitted and only single look-ahead point was chosen. The resulting modified controller can be seen in Equation 3.2.

Based on two reasons the original control law 3.1 was modified to 3.2, the first being, the limited contributions of the other look-ahead term and integral term. And the second reason

being, due to increased simplicity without compromising the tracking capability.

$$\delta^* = K_s.\phi + K_I.\int ey1 + K_\theta.e_\theta \tag{3.1}$$

Where K_s, K_I and K_θ are controller gains, δ^* is virtual steer angle, ϕ is resulting correction angle, ey1 is the vehicle lateral error at look ahead point and e_θ is the heading error.

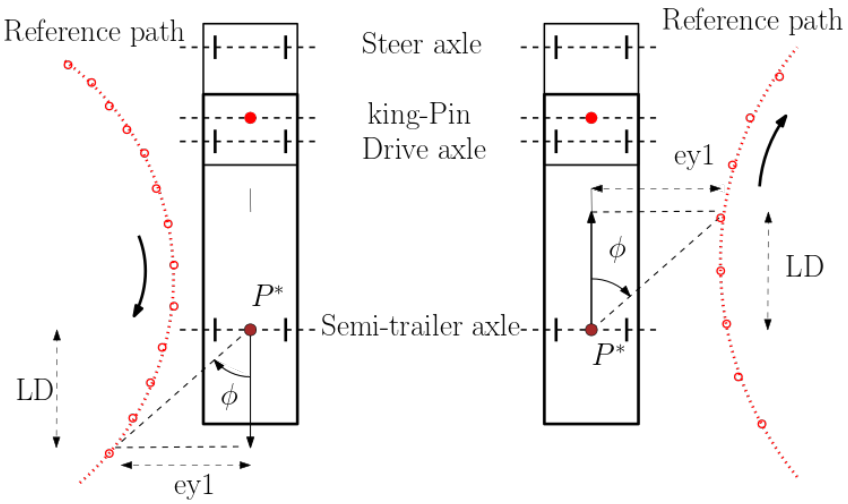


Figure 3.5.: Bi-DPFC geometry, reference paths in forward and reverse motion

This controller takes a series of reference points as input and a look-ahead point H is projected forward or reverse depending on the direction of motion with its projection root at the control point P^* of the vehicle. The control point P^* is chosen as the mid point of the semitrailer axle see Figure 3.5. The look-ahead point H iteratively measures the closest point on the reference path and calculates the cross-track error $ey1$ as shown the Figure 3.5. The equation 3.2 gives the required virtual steering angle which is translated via inverse kinematic relations 3.3 and 3.4 to the steering angle, 3.5 of the real tractor that drives the vehicle control point along a reference path during the forward simulation.

$$\begin{aligned} \phi &= \tan^{-1}(\frac{ey1}{LD}) \\ \delta^* &= K_s.\phi \end{aligned} \tag{3.2}$$

ϕ resulting correction angle is responsible for reduction of the lateral error based on profile of the reference path at given look ahead distance LD, and steering gain K_s.

$$\dot{\theta}_1 = \frac{|v_1|}{L_{1f}} \tan\delta^*, \tag{3.3}$$

$$\dot{\theta}_0 = sign(v_1).sign(L_{0b}).(-(\frac{v_1}{L_{0b}}.\sin(\gamma_1) + (\frac{L_{1f}}{L_{0b}}).(\dot{\theta}_1).\cos\gamma_1)), \tag{3.4}$$

$$\delta = \tan^{-1}(\frac{L_{0f}}{v_0}(\dot{\theta}_0)) \tag{3.5}$$

δ is the steering angle for the real tractor.

3.3.2. Pure Pursuit Controller

It is a simple vehicle lateral control method used in the literature. It is amongst the earliest path tracking strategy, dating back to 1980's. Its relevance still remains to-date because of its simplicity and satisfactory performance.Pure pursuit can be called as the geometric path tracking controller. A tracking controller that tracks a reference path using only the geometry of the vehicle kinematics and the reference path is called as geometric path tracking controller.

working principle The concept of pure pursuit is based on pursuing a look-ahead point P as shown in the Figure 3.6. This point is the point of intersection of a reference path and a Look-ahead circle with its center at vehicles control point P^*. The implementation of this task of calculating point of intersection in Matlab function called calcips( ) is presented in A.2. Once the look-ahead point P is known the controller defines an imaginary desired path between the control point P^* and the look-ahead point. And tries to drive the vehicle along the desired path by setting a control signal such that the vehicle will end up at P if the signal is hold constant. But as the vehicle moves the point of intersection P moves along, resulting in the smooth transition of the controller's desired signal aiming to pursue P with every step.This desired signal is the steering angle that lets the vehicle follow the reference path while generating a smooth vehicle traversed path as shown in the Figure 3.6.

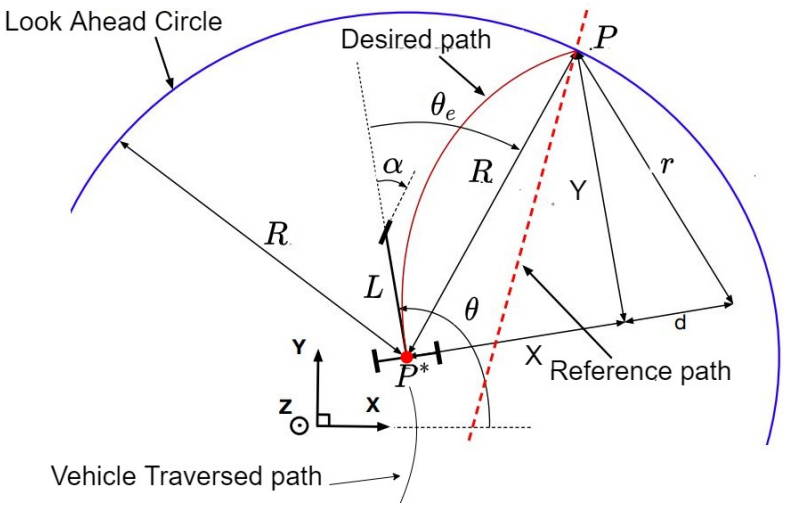


Figure 3.6.: The pure pursuit controller Geometry. A desired path is circular arc, fit between vehicle control point P^* and the look-ahead point P on the reference path such that the chord length P^*P is the look ahead distance R and the desired path is tangent to the heading direction of the vehicle unit whose control point P^* is considered. [6]

The following equations are derived based on the pure pursuit geometry. Where the orientation error θ_e is calculated first in order to obtain the desired control signal angle α. The implementation of this control law using a Matlab function called controlleroutput( ) is presented in A.2.

$$\begin{aligned} \theta_e &= \text{atan2}(X_P - X, Y_P - Y) - \theta \\ \alpha &= -\text{atan2}(\frac{2L\sin\theta_e}{R}) \end{aligned} \tag{3.6}$$

Types of paths in the context of Pure pursuit controller

As shown in Figure 3.6 there are three different types of paths involved as follows,

- Reference path
- Desired path
- Vehicle traversed path

Reference path
In general sampling based planner like RRT requires a method of joining the sampled points while building the tree. The resulting tree segments are fed as reference path to the controller. The controller tracks the reference paths and produces the vehicle traversed paths that respect the kinematic constraints of the vehicle. Various types of reference paths are used in the literature.

- piece-wise linear line segments
- spline segments

In literature a Pure pursuit controller is used in [11], [6] because of its great steering ability. Hence a pure pursuit controller has been developed as a Simulink subsystem for this thesis. This controller subsystem is introduced with the rest of the subsystems available like Inverse Kinematic Model (IK), KM, VAS and experimented for feasibility of forward simulation in chapter 4. The popularity for this steering controller is because of it ability to follow reference paths that are defined as piece-wise linear segments. More experimental work and results are discussed in chapter 4.

Desired path is specific to this controller and it is the path which constantly changes with every time step. This is the path the vehicle aims to pursue constantly. The control law drives the vehicle towards the reference path using this intermediate desired paths.

Vehicle traversed path is the path obtained after forward simulation of the model. This essentially includes vehicle control point history resulting from forward simulation. when combined with vehicle kinematics and outer body dimensions the traversed path of the whole SAV can be obtained. This information is used to detect collisions in the later sections.

3.4. Goal region constraints

Given the motion planning problem a goal region, C_{goal}, is used to define constraints on the final configuration of the states, $x(t_f)$. x_g is defined as an optimal goal position but all the solutions in the region close to x_g are accepted as final goal configurations. The final configuration of the SAV with control point at $P_* = [x_1(t_f)\ y_1(t_f)]$ (at the mid point of the drive axle) and heading of the semitrailer as $\theta_1(t_f)]$ are considered as $x(t_f)$ and the goal configuration $x_g = [x_{1,g}\ y_{1,g}\ \theta_{1,g}]$. The following conditions are used to decide if $x(t_f)$ is within the accepted goal region.

$$\begin{aligned}(x_1(t_f) - x_{1,g})^2 + (y_1(t_f) - y_{1,g})^2 &\leq (\Delta r)^2 \\ (\theta_1(t_f) - \theta_{1,g}) &\leq \Delta\theta \end{aligned} \tag{3.7}$$

where the constraints are set to the values $\Delta r = 0.6m$ and $\Delta\theta = 0.05rad$.

3.5. CL-RRT Nomenclature

Although CL-RRT has some terminology in common to other motion planners, it also has some unique terms specific to its components. Before getting into the details of each component of the CL-RRT in the coming chapters it is important to revise through the specific words used often in this motion planner. In this section we list such words followed by a short definition.

Node, is a data point which has information of vehicle configuration (i.e, co-ordinates and orientation), index number

Tree, is a data bank in which, incrementally generated data related to motion planner is stored. The current implementation of the algorithm for this thesis stores information of current node, parent node, sampling type choice, forward simulation results like vehicle traversed paths, cost of reaching the current node (arc length of traversed path), cumulative cost-total arc length from current node to the root node, direction of motion and number of time steps taken to reach the respective node. This information is reused during the extension of the tree itself and also is very helpful for post processing tasks like finding the most optimal path among the solution obtained. For initializing the tree a MATLAB array of the chosen size is created.

Forward simulation is a process of simulating an existing mathematical model by providing required inputs. In the context of the CL-RRT planner the Forward simulation process generates vehicle traversed paths and related information. The overview of the implemented forward simulation model for this thesis can be seen at A.2.

Input space is characteristic of a given controller. In the context of a vehicle steering controller that is intended to follow a reference paths, input-space of such controller is a subset of reference paths, that the controller is capable of following, while producing kinematically feasible vehicle traversed paths. During a planning cycle a sample, $s = [s_x, s_y]$, is drawn in the controller input space from a sample distribution [18]. The Heuristic maps which are a cost map of distance metric used with in the RRT algorithm see Figure 4.4 with linear reference paths and Bi-DPFC, 4.28 with linear reference paths and pure pursuit controller, give an idea of the Input space of the specific controller and vehicle model combination . The motion planning algorithm solves a optimal control problem by randomly exploring the input space of the chosen controller to the closed-loop system by performing forward simulations of the closed-loop system.

3.6. RRT components in the CL-RRT framework

The original CL-RRT algorithm builds a tree of kinodynamically feasible trajectories by randomly exploring the input space of the closed-loop stable system. As a motion planner for low speed maneuvering is intended we aim to build a tree of kinematically feasible paths. In [6] further research was conducted in the context of docking maneuver of articulated vehicles. Although the choice of forward simulation model and feedback controller is different from the implementation in [6], the components from the planning algorithm are borrowed.

3.6.1. Biasing the planning tree

Biasing is a process of selectively prioritizing tasks to achieve the purpose of the planning mission faster. Sampling strategies are used for biasing the planning tree. Sampling strategies play very important role and need to be tuned carefully to achieve faster results. Many variations of how a planning tree can be biased towards the goal configuration without spending time in exploring regions of less interest are proposed. In [6], sampling strategies are

used to direct the path planning tree in the direction of the goal configuration. In CL-RRT the tree of kinamatically feasible paths is grown incremantally by sampling in the input space of the controller. In other words sampling in inputspace of the controller leads to exploring of the vehicle configuration space. This is achieved by two such following sampling strategies.

- Exploring strategy
- Biasing towards goal strategy

Exploring strategy is a type of random sampling method where random sample is selected from a predefined geometric space. This strategy aids in exploring the configuration space of the vehicle within the predefined geometric space. The figure 3.7 give an idea of how this predefined geometric space is selected (dashed line black) and sampled depending on the region of the interest. Because a uniformly distributed random number is selected iteratively this method is called Uniform-Sampling in [6]. Standard deviation is also selectively applied to prioritize within the region of interest. In short the sample point obtained by employing exploring strategy is called as Random sample. The following equations show how this is achieved mathematically.

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X_0 + \sqrt{3}.\sigma_x.n_x \\ Y_0 + \sqrt{3}.\sigma_y.n_y \end{bmatrix} \tag{3.8}$$

Where σ_x and σ_y are the standard deviations in the respective coordinate axis. n_i is a uniformly distributed randomly generated number, i.e. $-1 \leq n_i \leq 1$.

Biasing towards goal strategy is a type of random sampling method where random sample is selected in the vicinity of the goal point. This strategy aids in drawing the planning tree towards it. Hence by introducing such bias the time required to obtain solution is also reduced. The figure 3.7 give an idea of how the vicinity of goal point is selectively sampled (yellow circle). Because a uniformly distributed random number is selected iteratively and samples that spans the radial and angular directions of a selected goal point are considered this type of sampling is called arc sampling in [6]. Standard deviation is also selectively applied to prioritize within the region of interest. In short the sample point obtained by employing biasing towards goal strategy is called as Goal sample. The following equations show how this is achieved mathematically.

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix} + r \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \textit{where} \begin{cases} r = \sqrt{3}.\sigma_r|n_r| + r_0 \\ \theta = \sqrt{3}.\sigma_\theta|n_\theta| + r_\theta \end{cases} \tag{3.9}$$

where σ_r and σ_θ is the standard deviation in radial and circumference direction. n_i is a uniformly distributed randomly generated number, i.e. $-1 \leq n_i \leq 1$.

By employing such strategies along side forward and reverse motion of the SAV the usual docking maneuver is achieved. In Figure 3.7 Random sampling and goal sampling with reference point $[X_0\ Y_0]$ at [180 187] and goal pose at [90 133.5 pi/2].

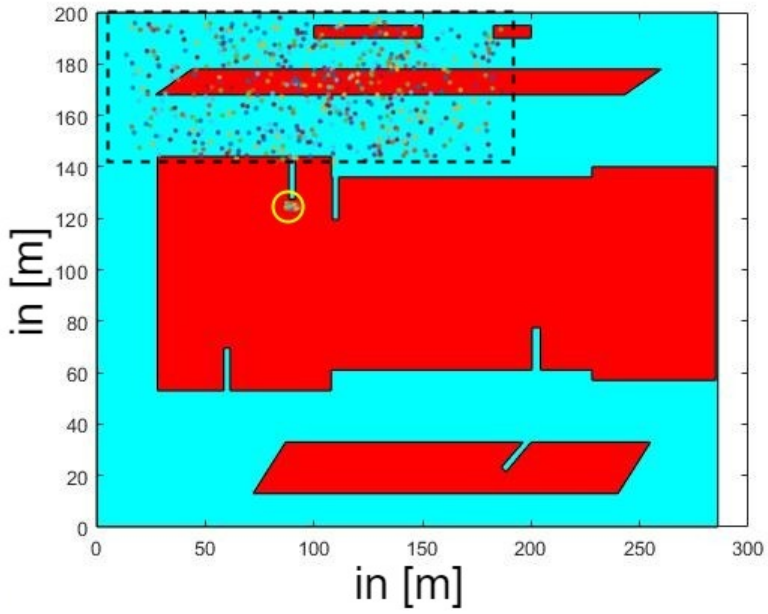
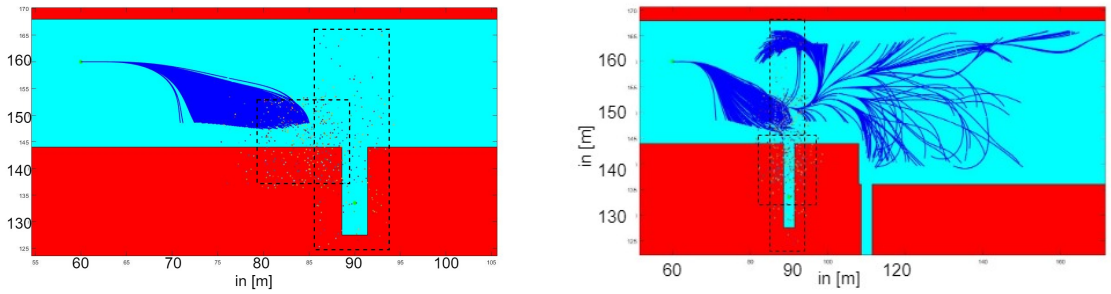


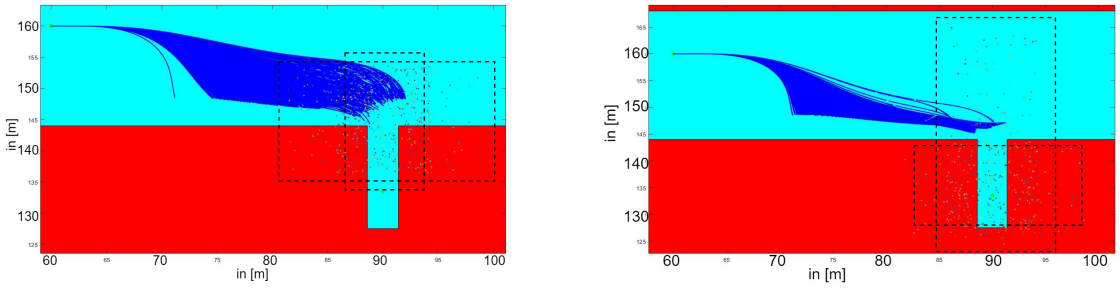
Figure 3.7.: Random sampling (dashed black line) of a specific region of interest within the given obstacle map and goal sampling (yellow circle)

The following Figures show how these sampling strategies draw the planning tree towards the region of interest.



(a) refer to Table 3.1 for the parameters chosen to obtain the shown sampling distribution

(b) refer to Table 3.2 for the parameters chosen to obtain the shown sampling distribution



(c) refer to Table 3.3 for the parameters chosen to obtain the shown sampling distribution

(d) refer to Table 3.4 for the parameters chosen to obtain the shown sampling distribution

Figure 3.8.: Comparison between the patterns of the planning tree development with variations in sampling strategy, dashed black lines highlight the region biased using the parameters like reference point, standard deviation and probability

Table 3.1.: exploring strategy parameters

Strategy	**Direction**	**Bias/**$[x_0, Y_0]$	**Standard deviation**	**Probability**
Exploring	Reverse	[85 145]	[5 5]	0.2
Exploring	Reverse	[85 145]	[2.5 5]	0.5
Exploring	Reverse	[90 150]	[2.5 12]	0.3

Table 3.2.: exploring strategy parameters

Strategy	**Direction**	**Bias/**$[x_0, Y_0]$	**Standard deviation**	**Probability**
Exploring	Reverse	[90 145]	[5 5]	0.2
Exploring	Reverse	[90 145]	[2.5 5]	0.5
Exploring	Reverse	[90 145]	[2.5 12]	0.3

Table 3.3.: exploring strategy parameters

Strategy	**Direction**	**Bias/**$[x_0, Y_0]$	**Standard deviation**	**Probability**
Exploring	Reverse	[90 135]	[5 5]	0.2
Exploring	Reverse	[90 135]	[2.5 5]	0.5
Exploring	Reverse	[90 145]	[2.5 5]	0.3

Table 3.4.: exploring strategy parameters

Strategy	**Direction**	**Bias/**$[x_0, Y_0]$	**Standard deviation**	**Probability**
Exploring	Reverse	[90 145]	[5 5]	0.2
Exploring	Reverse	[90 145]	[2.5 5]	0.5
Exploring	Reverse	[90 145]	[2.5 12]	0.3

3.6.2. Aiming for Optimal path

As mentioned earlier the motion planning algorithms should aim to generate a feasible path that is safely and comfortably trackable by the vehicle and if possible an optimal path that is the path when taken costs less resources and time. Although complete optimalness in the solution path is a very hard problem, many approaches that offer sub optimal solutions are being used in real deployments. Two such approaches commonly found in the literature of the RRT algorithms are,

- Nearest neighbour
- cheapest neighbour

Nearest neighbour method as the name sounds strives to find the nearest neighbour of the newly sampled point. The neighbour is selected based on some distance metric. Distance metric calculation should consider the vehicle kinematics for obtaining meaningful choice of the nearest neighbour. The simplest distance metric is the euclidean distance. But such distance metric fails to respect the kinematic nature of the vehicle and hence might even choose a worst neighbour that never leads to the solution path.

In order to capture the kinematic non-holonomic constraints of the articulated vehicle while calculating the distance metric a method is proposed in [6]. Where the vehicle model along with the controller combination is forward simulated over a predefined region of surroundings of the vehicle model. That is a vehicle model is initialized with its current position at origin and forward simulated over a grid of 50mx50m points starting from the origin and with resolution of 0.5m. This size of the grid is chosen selected as the given distribution center map is of size 270mx200m. After every such individual forward simulation the curve length of the vehicle traversed path is calculated and stored as a cost to reach that particular point of interest. Decision on choosing a type of reference path used to join initial point of the vehicle that is the origin and the selected point from the grid depends on the type of the controller being used.

This forward simulation for generating the heuristic map should included exact components that will be used later for building the CL-RRT tree incrementally. Such map is suggested to be calculated offline. As this will reduce an overhead on the planner loop. Once the cost map otherwise called as the heuristic map is obtained it is stored and used as a distance metric for finding the nearest neighbour.

Cheapest neighbour as it sounds strives to aid the planner in selecting a node whose past traversed cost (i.e, the cost obtained when calculated from the current node to the root node at the origin) is minimal. This type of node selection aims to find a close to optimal path which cannot be assure to be completely optimal but proved to be sub optimal and kinematically feasible.

Each of the above mentioned node selection method is randomly selected with a probability of 0.5 and thereby benefits of both methods are included while finding the solution path. Once the node selection is made the newly sampled point connects itself to the selected node and creates and extension. Further details of implementation and the results obtained are presented in next chapter in a flow along with other components.

3.7. Obstacle Avoidance

The current implementation of the CL-RRT motion planer aims to plan a path within an environment with predefined static obstacles.

The predefined obstacles are defined using simple polygons as shown in the Figure 2.8. The obstacle space is taken as an input by the motion planner and using a simple collision detection method the obstacles are avoided during the motion planning. Figures in 3.8 show the planning tree by the SAV successfully avoiding the predefined obstacles.

In [19], Obstacle avoidance is achieved by defining the boundary vertices of the SAV tractor and semitrailer based on its real dimensions and moving control point of the vehicle in the configuration space. Dimensions for the vehicle model used for the current path planner are listed in the Table 4.1. The Figures 3.9, 3.10 show the geometry of the boundary model defining vertices of the vehicle tractor and semitrailer. starting from the co-ordinates of the vehicle control point and orientation of the tractor, all the outer vertices of the tractor body and the semitrailer body are obtained in the following sections.

Finding vertices of the trailer

The coordinates of all four vertices of the trailer are calculated by defining four vectors $C1_1$, $C2_1$, $C3_1$, $C4_1$ to all four vertices of the trailer as shown in figure 3.9. The front overhang and rear overhang are denoted by oh_{1f} and oh_{1b}. The width of the trailer is denoted by w. The angles $av1_1$, $av2_1$, $av3_1$ and $av4_1$ are the global angles of the vectors respectively. The orientation angle of the trailer is denoted by θ_1 and its wheelbase is denoted by L_{1f}.

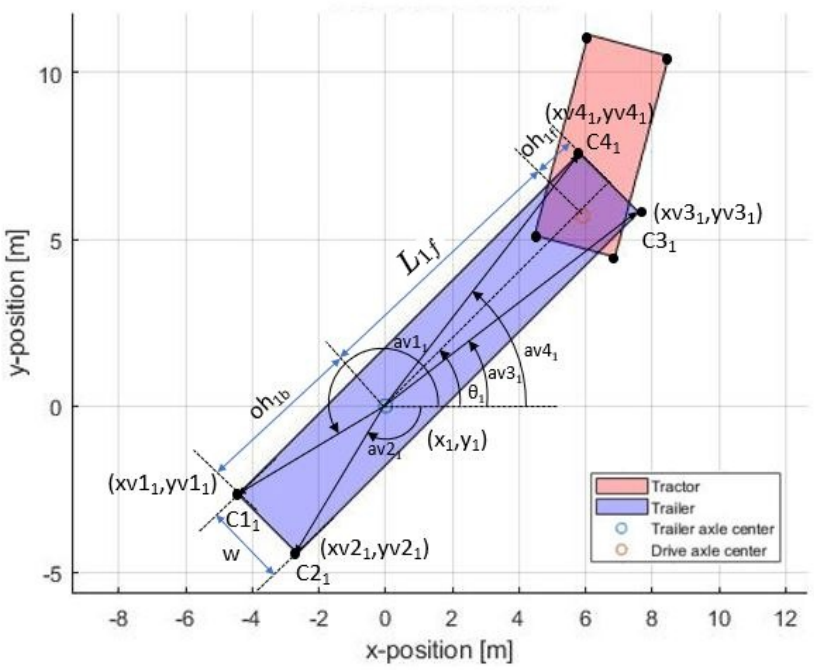


Figure 3.9.: Coordinates of the vertices of the trailer [19]

For the determination of these vertices the length of these vectors and their global orientation are to be obtained. First the lengths of the vectors are calculated. It can be observed that the length of $C1_1$ and $C2_1$ are equal and the lengths of $C3_1$ and $C4_1$ are equal. These lengths are found by the equation 3.10.

$$\begin{aligned} |C1_1| &= |C2_1| = \sqrt{(oh_{1b})^2 + (w_1/2)^2} \\ |C3_1| &= |C4_1| = \sqrt{(w_1/2)^2 + (oh_{1f} + L_1)^2} \end{aligned} \tag{3.10}$$

Then, for the determination of the angles of these vectors, equation 3.11 is used.

$$\begin{aligned} av1_1 &= \theta_1 + (\pi/2) + \arctan\left(\frac{oh_{1b}}{0.5w_1}\right) \\ av2_1 &= \theta_1 - (\pi/2) - \arctan\left(\frac{oh_{1b}}{0.5w_1}\right) \\ av3_1 &= \theta_1 - \arctan\left(\frac{0.5w_1}{L_1 + oh_{1f}}\right) \\ av4_1 &= \theta_1 + \arctan\left(\frac{0.5w_1}{L_1 + oh_{1f}}\right) \end{aligned} \tag{3.11}$$

Finally, the coordinates of the vertices are found by the equation 3.12

$$\begin{aligned} xv1_1 &= x_1 + |C1_1|\cos(av1_1), \quad yv1_1 = y_1 + |C1_1|\sin(av1_1) \\ xv2_1 &= x_1 + |C2_1|\cos(av2_1), \quad yv2_1 = y_1 + |C2_1|\sin(av2_1) \\ xv3_1 &= x_1 + |C3_1|\cos(av3_1), \quad yv3_1 = y_1 + |C3_1|\sin(av3_1) \\ xv4_1 &= x_1 + |C4_1|\cos(av4_1), \quad yv4_1 = y_1 + |C4_1|\sin(av4_1) \end{aligned} \tag{3.12}$$

Finding vertices of the tractor

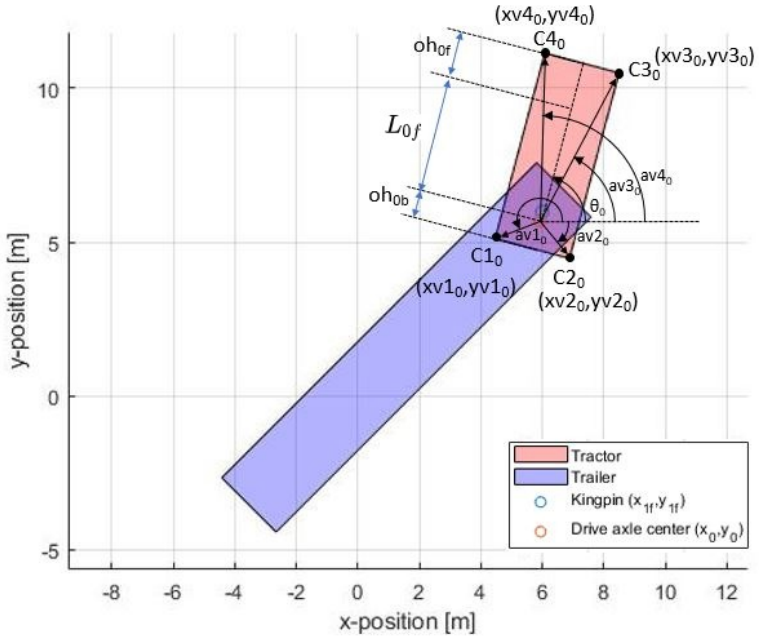


Figure 3.10.: Coordinates of the vertices of the tractor [19]

In order to find the vertices of the tractor, first the coordinates of the king pin (x_{1f}, y_{1f}) have to be found. This is done by the equation 3.13

$$\begin{aligned} x_{1f} &= x_1 + L_1\cos(\theta_1) \\ y_{1f} &= y_1 + L_1\sin(\theta_1) \end{aligned} \tag{3.13}$$

θ_0 tractor heading angle is obtained using the relation ship between the tractor heading angle θ_0, trailer heading angle θ_1 and the articulation angle γ as shown in the equation 3.14.

$$\theta_0 = \theta_1 + \gamma \tag{3.14}$$

In 3.15 Coordinates of the center of the trailer axle are obtained using equations 3.13 and equation 3.14 .

$$\begin{aligned} x_0 &= x_{1f} - L_b \cos(\theta_0) \\ y_0 &= y_{1f} - L_b \sin(\theta_0); \end{aligned} \tag{3.15}$$

Similar steps are taken as in the case of trailer to obtain the co-ordinates of the tractor. Firstly the lengths of the vectors to the vertices of the tractor are calculated. It can be observed that the length of $C1_0$ and $C2_0$ are equal and the lengths of $C3_0$ and $C4_0$ are equal. These lengths are found using the equation 3.16.

$$\begin{aligned} |C1_0|, |C2_0| &= \sqrt{(oh_{0b})^2 + (w_1/2)^2} \\ |C3_0|, |C4_0| &= \sqrt{(w_1/2)^2 + (oh_{0f} + L_0)^2} \end{aligned} \tag{3.16}$$

Then, for the determination of the angles of these vectors, equation 3.17 is used.

$$\begin{aligned} av1_0 &= \theta_0 + (\pi/2) + \arctan\left(\frac{oh_{0b}}{0.5w_1}\right) \\ av2_0 &= \theta_0 - (\pi/2) - \arctan\left(\frac{oh_{0b}}{0.5w_1}\right) \\ av3_0 &= \theta_0 - \arctan\left(\frac{0.5w_1}{L_0 + oh_{0f}}\right) \\ av4_0 &= \theta_0 + \arctan\left(\frac{0.5w_1}{L_0 + oh_{0f}}\right) \end{aligned} \tag{3.17}$$

Finally, the coordinates of the vertices are found by the equation 3.18

$$\begin{aligned} xv1_0 &= x_0 + |C1_0| \cos(av1_0), \quad yv1_0 = y_0 + |C1_0| \sin(av1_0) \\ xv2_0 &= x_0 + |C2_0| \cos(av2_0), \quad yv2_0 = y_0 + |C2_0| \sin(av2_0) \\ xv3_0 &= x_0 + |C3_0| \cos(av3_0), \quad yv3_0 = y_0 + |C3_0| \sin(av3_0) \\ xv4_0 &= x_0 + |C4_0| \cos(av4_0), \quad yv4_0 = y_0 + |C4_0| \sin(av4_0) \end{aligned} \tag{3.18}$$

Collision Detection method

Now by forward simulating the vehicle boundary model and the controller combination, the path tracked by all the corners of the boundary model is obtained as shown in the Figure3.11. Each point in this path is taken and checked if it falls inside the space defined by the polygonal obstacles. During every forward simulation of the vehicle model the path tracked by the boundary model is checked at every time step for any possible collision.

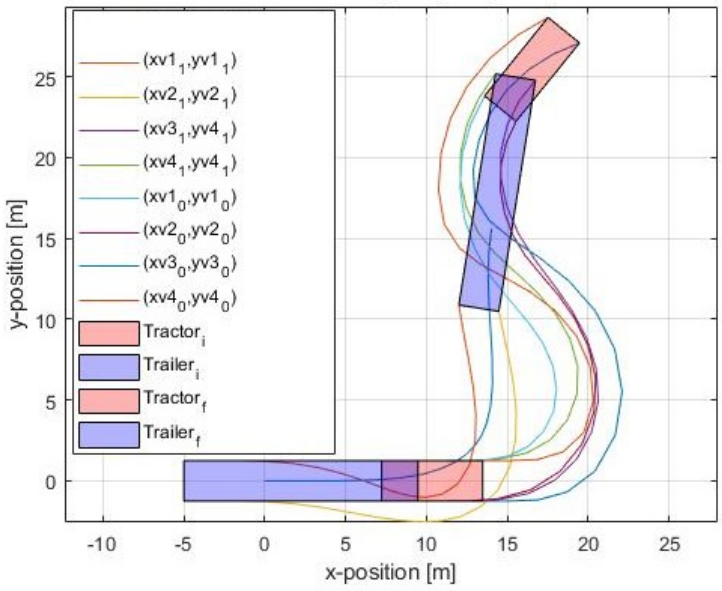


Figure 3.11.: Paths traced by vertices [19]

In this thesis the obtained boundary model is forward simulated with every extension of the planning tree and a collision check method is performed as discussed in [19]. Thereby ensuring the feasibility for the real vehicle which tries to follow the obtained solution path.

4. Experimental results and Implementation Method

In this chapter, continuing the research flow of project VISTA, the modified generic framework of HAN Auto-docking control or Bi-DPFC explained in chapter 3, sub section 3.3.1 for a single articulated vehicle will be reviewed and experimented with in the context of including it in the CL-RRT framework. In section 4.1.1 the Bi-DPFC will be tuned to a new set of vehicle dimensions as shown in the Table 4.1. Followed by sections 4.1.2, 4.1.6, 4.1.7, 4.1.3 which elaborate the Experiments performed, results obtained and conclusions made with the aim of building components that complete the loop of CL-RRT framework discussed in the chapter 3 in Figure 3.1. In the section 4.1.5, the Simulink Subsystem *Virtual Articulation Angle* added to the generic framework of HAN Auto-docking control in [13] to improve path tracking will be reviewed and studied for its suitability in the CL-RRT framework.

Finally in sections 4.2 implementation method for the CL-RRT motion planner used to obtain the solution paths as shown in chapter 5 is elaborated with the help of flowcharts, Simulink Stateflow diagrams and specific details.

4.1. Experimental results

4.1.1. Tuning the Bi-DPFC

After discussing the working principle of the Bi-DPFC we can see that the control parameters Look ahead Distance (LD) and proportional control gain (K_s) are variables and a tuning strategy has to be employed to find the appropriate values.

Tuning strategy for the Bi-DPFC is similar to the tuning strategy employed in [1]. As the controller is modified, only two gains are to be tuned to find the best optimal pair $[LD, k_s]$ that has minimum $MeansquaredError(MSE)_{ey1}$ of the lateral tracking error. The mean squared error is calculated using the equation 4.1. The lateral error $ey1$ is the component used for which the MSE is determined. The motive for using this method is to eliminate the polarity changes in the error values.

$$MSE_{ey1} = \frac{1}{n}\sum_{i=1}^{n}(ey1^2); \tag{4.1}$$

A looped simulation is created where the Permutations of these two gain values i.e, LD and K_s are selected over a predefined range and are tested for a given vehicle combination for the test case reference path. To forward simulate the vehicle model one has to decide on the reference path the vehicle will be following. For the vehicle dimensions listed in 4.1, the circular reference path 4.1 is chosen finding the optimal pair. The radius of the circle is selected based on the stable radius over which the vehicle can be forward simulated without jack-knifing. The mean squared values of Lateral tracking error at look ahead distance LD, (ey1) are stored for all simulations. The combination of gain value for which the mean squared error is least is selected as the optimum gain values. The optimum value map for LD and K_s is given in the Figure 4.2. It is important that the vehicle traversed path obtained by forward

Dimension/Parameter	Symbol	Value
Wheelbase of tractor	L_0f	3.8m
Wheelbase of semitrailer	L_1f	7.21m
Distance of king pin from the driven axle	L_0b	0.48m
Frontal overhang of semitrailer	oh_{1f}	1m
Frontal overhang of tractor	oh_{0f}	1.5m
Rear overhang of semitrailer	oh_{1b}	5m
Rear overhang of tractor	oh_{0b}	0.94m
Width of the tractor and semitrailer	w	2.5m

Table 4.1.: Dimensions of the vehicle model

simulating the Bi-DPFC has a very minimal lateral error w.r.t the reference path as this directly influences the steering angle commands that are communicated to the user using the driver support system.

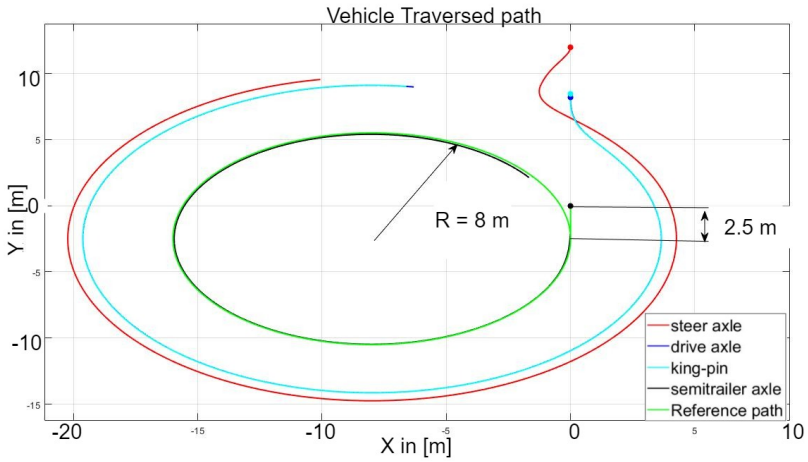


Figure 4.1.: Circular reference path chosen for forward simulations to find the right optimal combination of LD and K_s

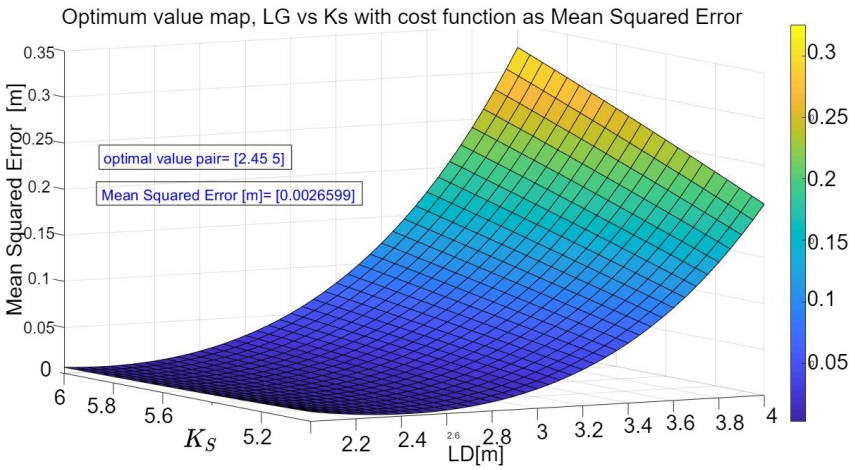


Figure 4.2.: Optimal value pair with minimal lateral error

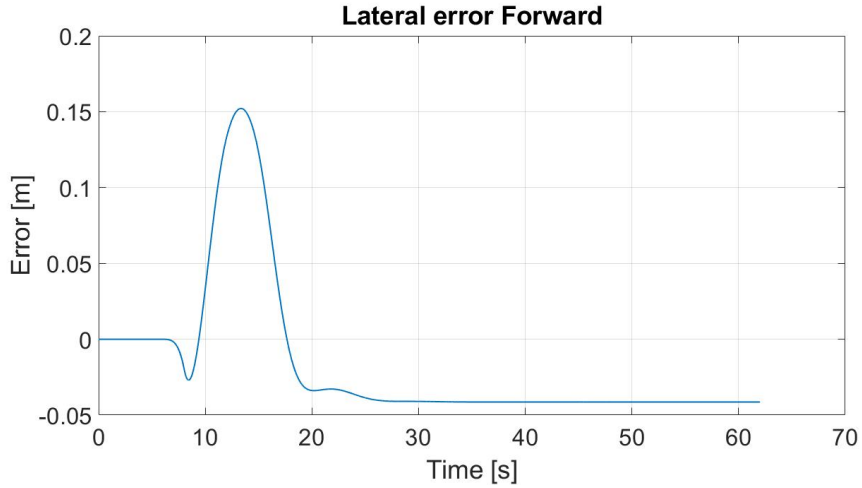


Figure 4.3.: lateral error plot

4.1.2. Selection of controller

Heuristic map which is a cost map based on the distance metric, is generated offline and the data is used to make a decision on choosing nearest neighbour. This map generation and its quality is directly linked to the type of controller and quality of the solution path. Hence it is important to select a controller that aids in proper heuristic map generation. In [6] a pure pursuit controller is forward simulated over a region and the corresponding cost map is generated. Following is the discussion on selection of the controller based on their Heuristic map generation ability for CL-RRT implementation.

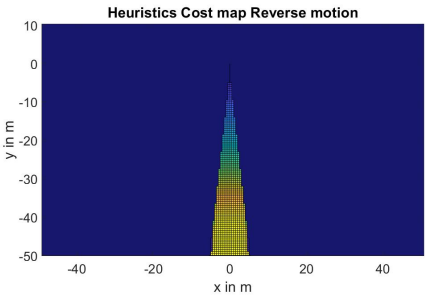
can the Bi-DPFC be included in CL-RRT?

Initial attempts were made to include Bi-DPFC in the standard CL-RRT planner with piece wise linear segments as reference paths. CL-RRT algorithm has components that work together and make decisions with the goal of obtaining a feasible and optimal path (refer 2.2.4 for the definition of feasible and optimal). One such component is "finding nearest neighbor". This component needs a distance metric to decide which node is the best option to connect in terms of building path that is kinematically feasible by the vehicle. For this purpose a Heuristic map generation is needed based on a meaningful distance metric. To achieve this, the vehicle

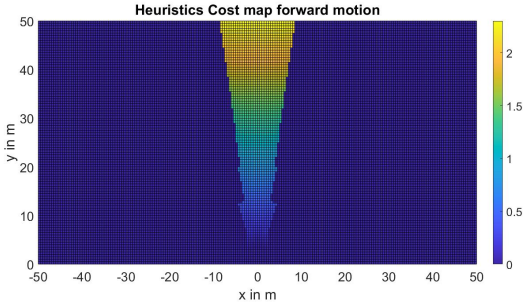
is forward simulated over a set of predefined coordinate points with a linear reference line joining from vehicle control point to the point chosen from the predefined set. The vehicle control point (mid point of the semitrailer drive axle) traversed path is stored as a cost which is considered as the distance metric to make decisions. When attempts were made to obtain such map using the Bi-DPFC the following observation was made.

- Bi-DPFC is not capable or rather not designed to track piece wise linear line segments in both forward and reverse directions.

when attempted to find a heuristic map with Bi-DPFC the results were obtained as in 4.4. These maps clearly show that the controller cannot track a linear reference path beyond certain angle. The Figures 4.5, 4.6 show the behavior and poor performance of the controller when forward simulated over a single linear line segment as a reference path.



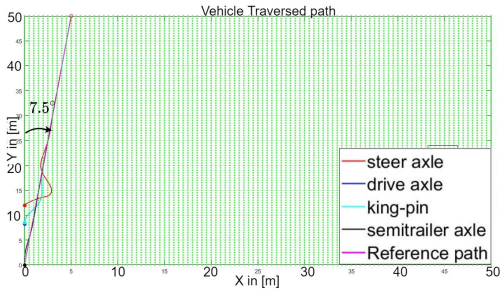
(a) Heuristic map for Bi-DPFC reverse direction



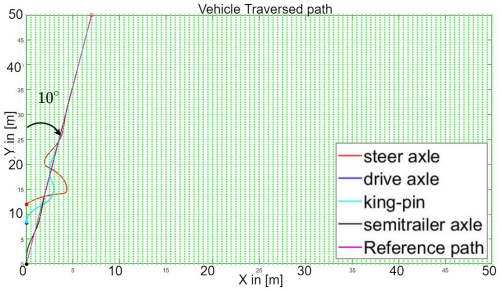
(b) Heuristic map for Bi-DPFC Forward direction

Figure 4.4.: Heuristic maps with Bi-DPFC

(a) Reference linear segment at an angle of 5° w.r.t vehicle heading direction

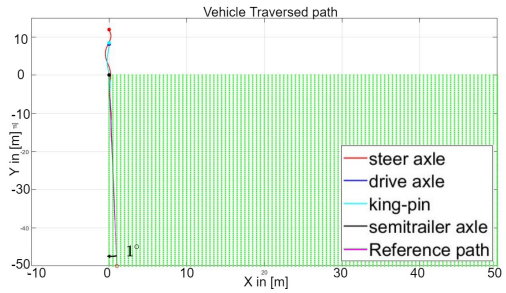


(b) Reference linear segment at an angle of 7.5° w.r.t vehicle heading direction

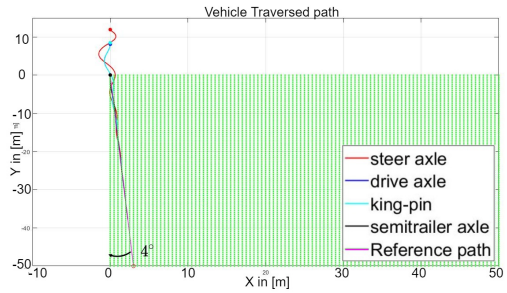


(c) Reference linear segment at an angle of 10° w.r.t vehicle heading direction

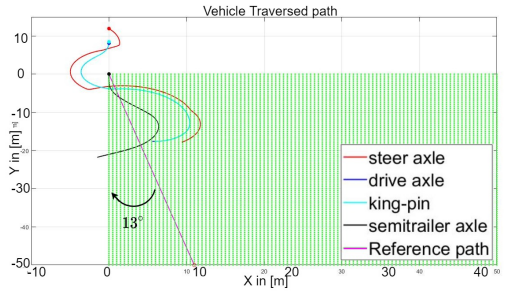
Figure 4.5.: poor performance of the Bi-DPFC over linear line segments as reference paths in Forward direction



(a) Reference linear segment at an angle of 1° w.r.t vehicle heading direction



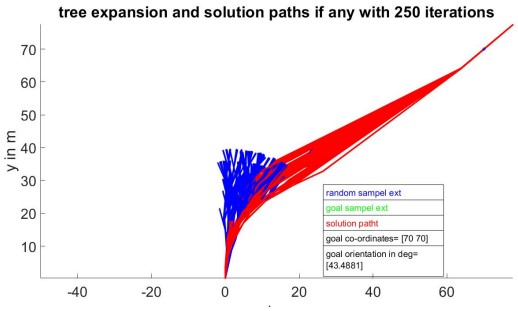
(b) Reference linear segment at an angle of 4° w.r.t vehicle heading direction



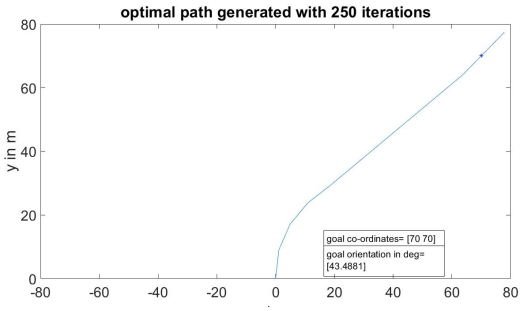
(c) Reference linear segment at an angle of 13° w.r.t vehicle heading direction

Figure 4.6.: poor performance of the Bi-DPFC over linear line segments as reference paths in Reverse direction

Noticing this limitation of the Bi-DPFC, a piece wise linear line segment based reference path that is within the limits of the Bi-DPFC capability to track has been generated using the RRT algorithm with the hope of obtaining a solution path. The Figure 4.7 shows the RRT expansion and obtained solution pice wise linear reference path meant to be fed as reference path to Bi-DPFC. This RRT nodes are sampled within the input space of the Bi-DPFC.



(a) RRT tree expansion to obtain a piece wise linear reference path that directs the SAV towards goal pose



(b) Obtained optimal (minimum length) reference path among all solutions

Figure 4.7.: RRT sampled within the input space of the Bi-DPFC

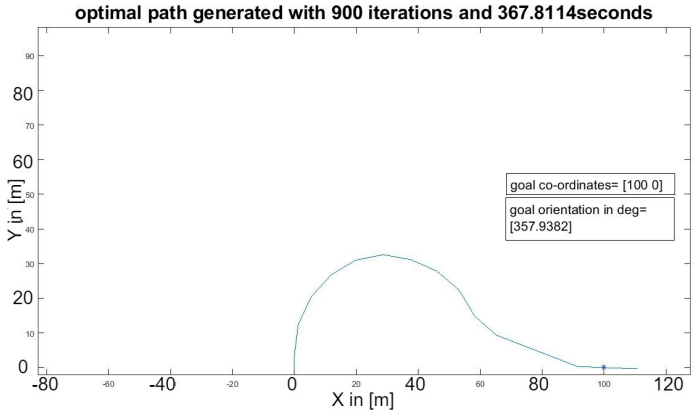


Figure 4.8.: piece wise linear reference path from initial pose to goal pose

But given the very small steering window which can be analogously seen from the heuristic map Figure 4.4 the obtained piece wise linear reference path to goal position was not at all realistic due to its large turning radius see Figures 4.7b and 4.8.

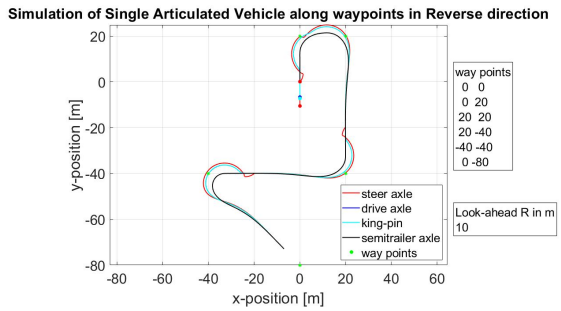
The result shows that the controller is incapable of tracking a linear line segment beyond the scope of its Input space (refer to section 3.5 for definition of Input space). It should be noted that this inability of the controller doesn't undermine its potential for performing in the application it is developed for and intended to be used in. The conclusion here is that this controller cannot be applied directly to the CL-RRT framework withe piece wise linear line segments as reference paths.

Performance of simple Pure Pursuit controller

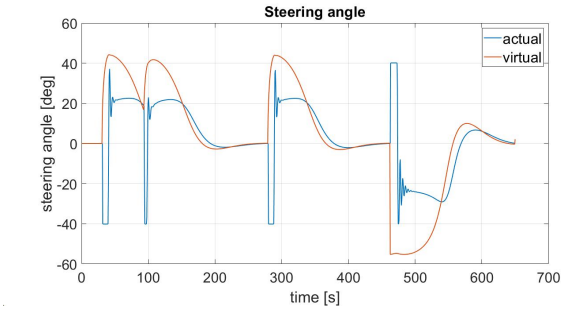
Based on the working principle explained in the chapter 3, Section3.3.2 a subsystem is created in Simulink as shown in the Figure A.3. This controller subsystem when combined with the vehicle model without articulation subsystem has shown promising results in following piece wise linear reference paths. The results obtained are presented in the Figures 4.9.

It can be seen that although the steering ability of simple Pure Pursuit Controller (PPC) is great it demands instantaneous changes in the steer angle which can be seen in the Figures 4.9c and 4.9d. This is the known drawback of the PPC in the literature [5]. Several modifications are available in the literature that modify the simple PPC that make it possible to deploy in real scenarios. in this thesis on such method is employed to overcome this drawback. Further explanation regarding this is provided in the section 4.1.3.

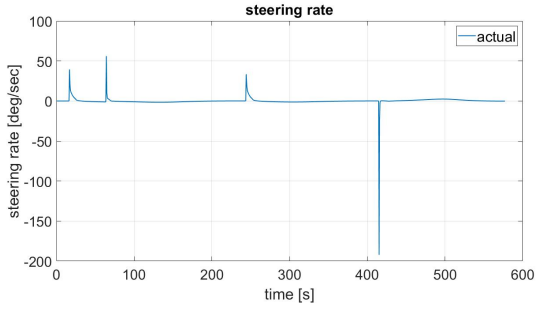
Conclusion Based on the below presented results the pure pursuit controller has been opted given its great steering ability over piece wise linear reference paths.



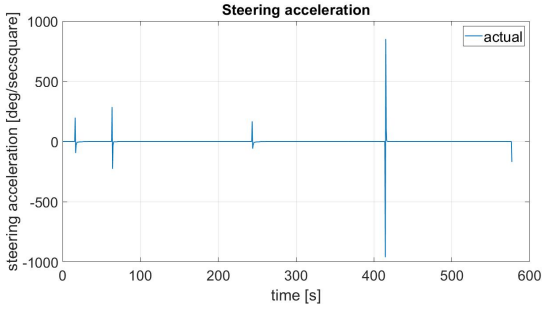
(a) Vehicle traversed path



(b) Steering angle demanded by PPC to follow the reference path



(c) Steering rate demanded by PPC to follow the reference path



(d) Steering acceleration demanded by PPC to follow the reference path

Figure 4.9.: Simple PPC and vehicle model forward simulated over linear line segments as reference paths in Reverse direction

4.1.3. Control on Steering rate

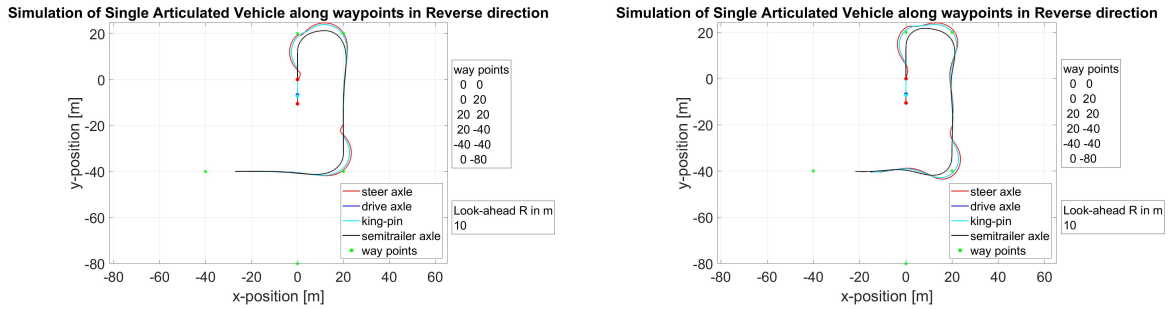
The paths generated by the simple pure pursuit controller depends on the reference path geometry. Although it is a very good steering controller that is simple and yet reliable it has a drawback of demanding unrealistic steering rates hence cannot be directly used in realistic scenarios. The unrealistic steering rates are demanded due to sudden change in the angle of the reference path. As a piece wise linear reference path is used as a reference path that is built by the RRT algorithm by randomly selecting the next point, the sudden change in the angle of the reference path is expected. In [5] limitations of this simple controller is highlighted. Additional improvements over the simple pure pursuit algorithm are made and integrated into the controller frame work that is actually deployed on the real vehicles. [14] and [11], for example use modified pure pursuit concept. In [14] a pure pursuit on high level is used and in

low level a LQ based articulation angle stabilization controller is used in forward simulating the vehicle model while motion planning. In the [11] they introduced an anchor point with reference to vehicle control point from which the look ahead circle is initiated. By doing so they proved improved stability of the controller during tracking.

In this thesis a low pass filter/differentiateor or also called a low pass digital filter is used to attenuate the unwanted noise from the out put of the controller. The incoming virtual steering angle is fit to a second order differential equation and eventually discretely integrated to obtain the steering rate and steering acceleration. By doing so the aggressive demand from the pure pursuit controller is delayed resulting in the control of steering acceleration and steering rate.

An important aspect of the Bi-DPFC is to deliver steering angle information to the subsystems of the driver assistance system. Therefore it is important to set the steering rate limit which is realistically achievable by the driver. 12 to 15 degrees/sec is the realistic value a human driver can achieve, hence this aspect has been included while tuning the pure pursuit controller and the digital filter combination. A look-ahead distance LD and digital filter parameter, Response time are set to 15 m and 13.5 respectively as it can be observed from Figures 4.12, 4.13 that the steering rate and the acceleration are under control confirming to the realistic needs. The digital filter parameter respose time was taken as 13.5 and LD of 10 m for following results.

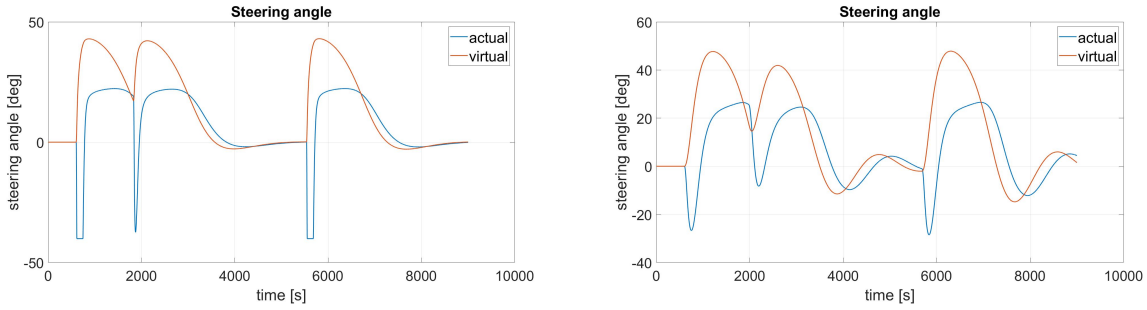
The Figures



(a) vehicle traversed path obtained without digital filter

(b) vehicle traversed path obtained with digital filter

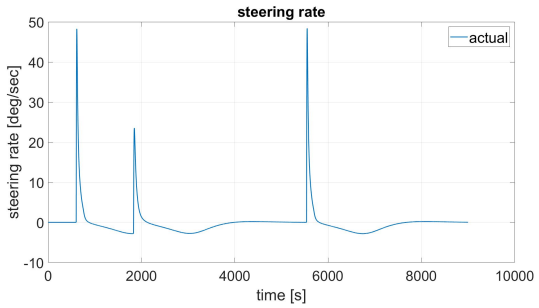
Figure 4.10.: Comparing vehicle traversed path of pure pursuit controller and vehicle model combination over forward simulation without and with digital filter



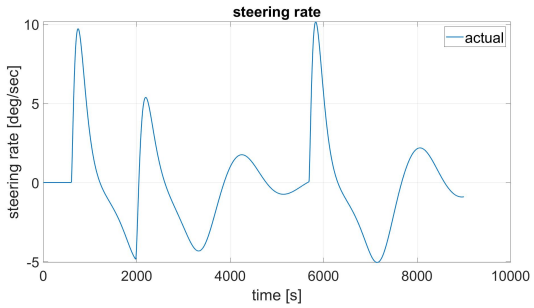
(a) steering angle performance obtained without digital filter

(b) steering angle performance obtained with digital filter

Figure 4.11.: Comparing steering angle performance of pure pursuit controller and vehicle model combination over forward simulation without and with digital filter

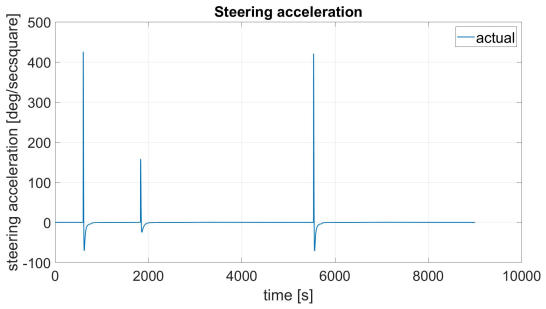


(a) steering rate performance obtained without digital filter

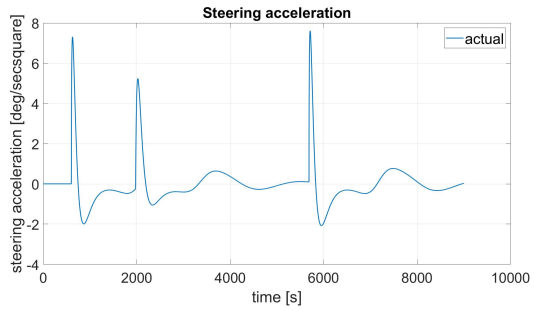


(b) steering rate performance obtained with digital filter

Figure 4.12.: Comparing steering rate performance of pure pursuit controller and vehicle model combination over forward simulation without and with digital filter

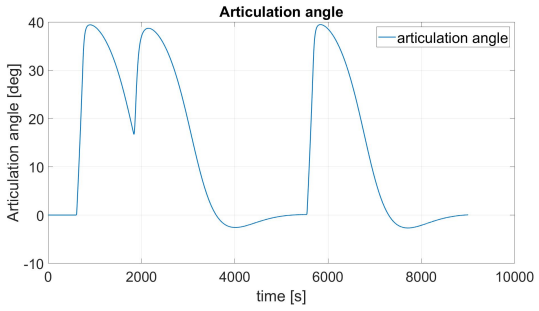


(a) Steering acceleration performance obtained without digital filter

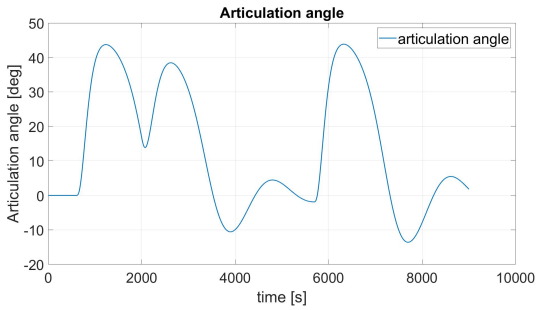


(b) Steering acceleration performance obtained with digital filter

Figure 4.13.: Comparing steering acceleration performance of pure pursuit controller and vehicle model combination over forward simulation without and with digital filter



(a) Articulation angle performance obtained without digital filter



(b) Articulation angle performance obtained with digital filter

Figure 4.14.: Comparing Articulation angle performance of pure pursuit controller and vehicle model combination over forward simulation without and with digital filter

Using such subsystem creates paths that are kinematically feasible but also adhere to the realistic steering rates that can be achieved by the driver who is using the motion planner which is part of a driver support system. The Figures 4.10, 4.11, 4.14 and 4.13 show performance

difference of the simple pure pursuit controller tracking piece wise linear reference paths with and with out the steering filter.

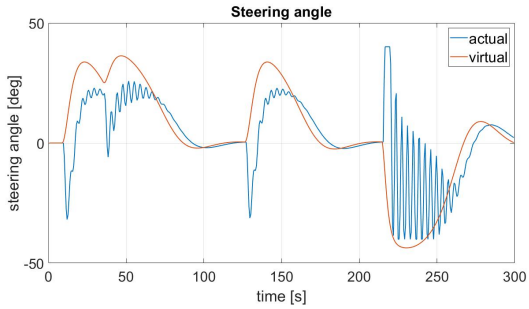
conclusion Two important aspects are to be addressed by a controller to be successfully included in the loop for CL-RRT framework.

- ability to steer towards and track piece wise linear reference paths with realistic steering rates
- contribute to the closed loop system stability either directly as in [14] or indirectly by not introducing any instability.

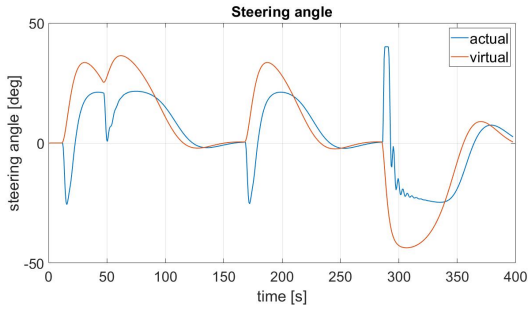
Based on the above presented results the pure pursuit controller in combination with the digital filter satisfies both the requirements and hence will be used for the Forward simulation.

4.1.4. Stepsize selection

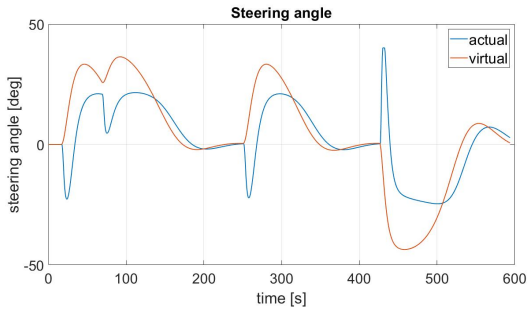
For Bi-DPFC and the pure pursuit controller the step size has a great influence on the end result of the steering angle which will be taken as the input for the portable interface of the driver support system. The Figures 4.15 shows the quality of the results at different step sizes. Higher the step size lower the quality of the results obtained.



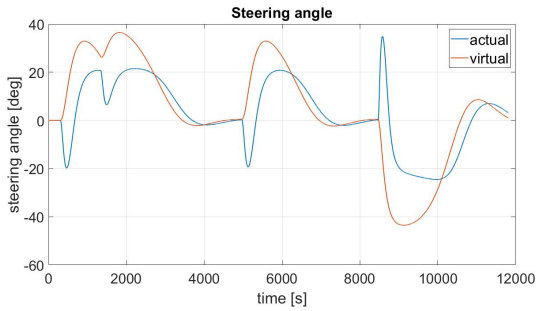
(a) with step size of 0.4



(b) with step size of 0.3



(c) with step size of 0.2



(d) with step size of 0.01

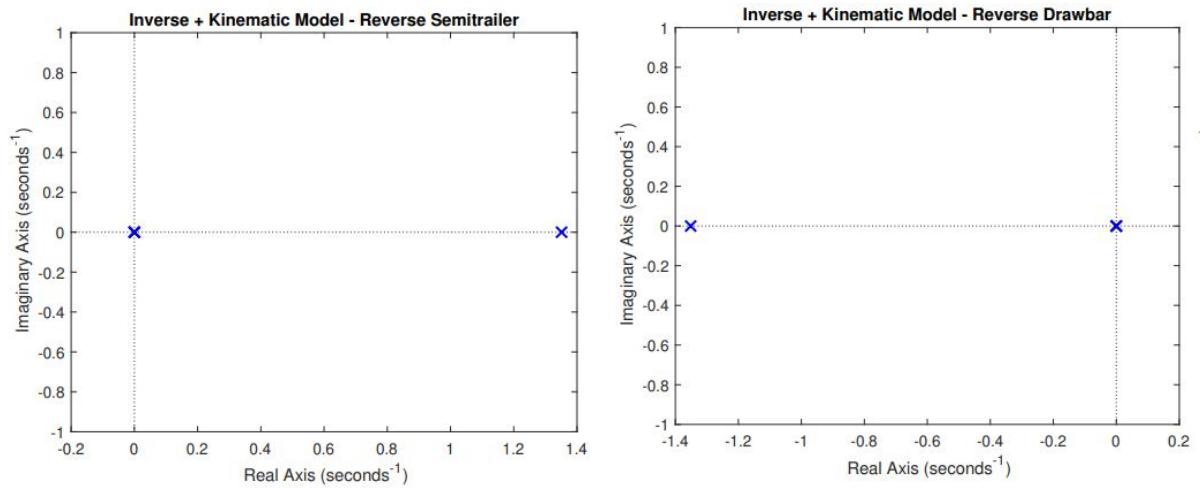
Figure 4.15.: Comparing steering angles obtained with different step sizes for the reference path from Figure 4.9a

Although having a lower step size produces good quality results 4.15d, having such setting will require more computation time. Hence it is very important to choose a step size that adequately small like in 4.15c and yet not too large like in 4.15a which may compromise the quality.

4.1.5. Virtual Articulation angle Subsystem VAS

In this section the special scenarios which are prone to instability are explained and improvements made to handle such scenarios are reviewed followed by the study of the

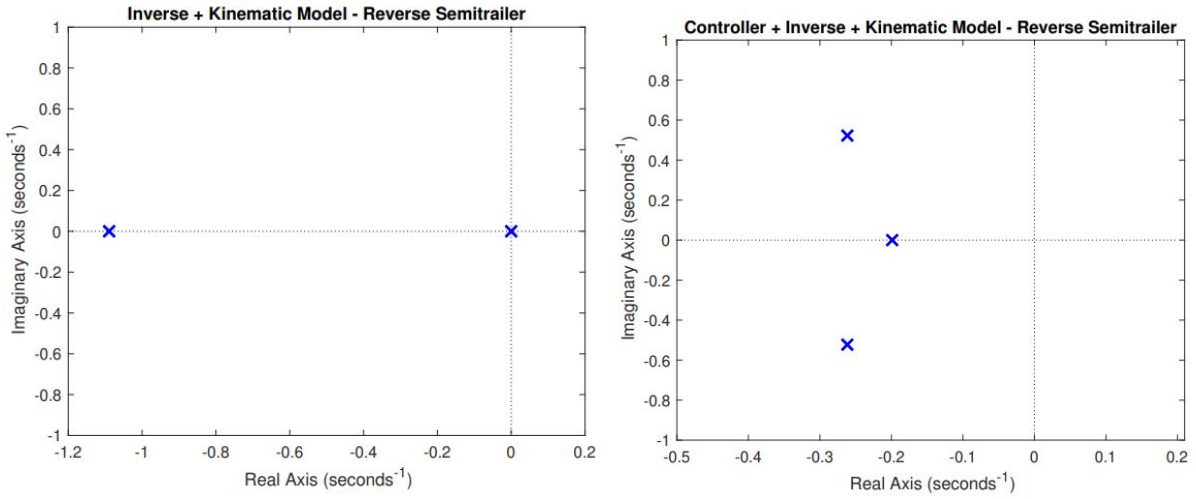
PPC+IK+KM+VAS combination for inclusion in the loop of CL-RRT. In [13] it is shown that in certain maneuvering scenarios for docking of SAV lead to non minimum phase scenarios. Among the four possible scenarios namely Forward/reverse for a semi-trailer and forward/reverse for a drawbar trailer two scenarios i.e, forward-drawbar, reverse-semitrailer, poles are shown to be on right half plane of pole - zero map for IK+KM and Bi-DPFC+IK+KM combination of the SAV configuration.



(a) Pole-zero map of IK+KM model for Reverse motion of Semi-trailer a Non-minimum phase scenario [13]

(b) Pole-zero map of IK+KM model for Reverse motion of drawbar trailer a stable scenario [13]

Figure 4.16.: Pole zero maps obtained from the transfer function of the IK, KM



(a) IK+KM pole zero map for reverse-semitrailer NMP scenario with signum [13]

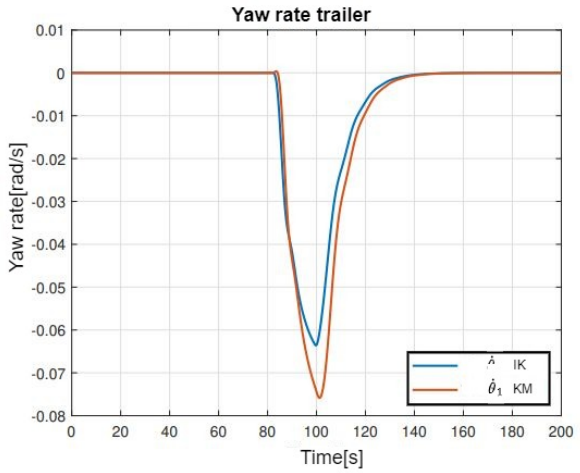
(b) Bi-DPFC+IK+KM pole zero map for reverse-semitrailer NMP scenario with signum [13]

Figure 4.17.: Pole zero maps obtained from the transfer function of the IK, KM and Bi-DPFC

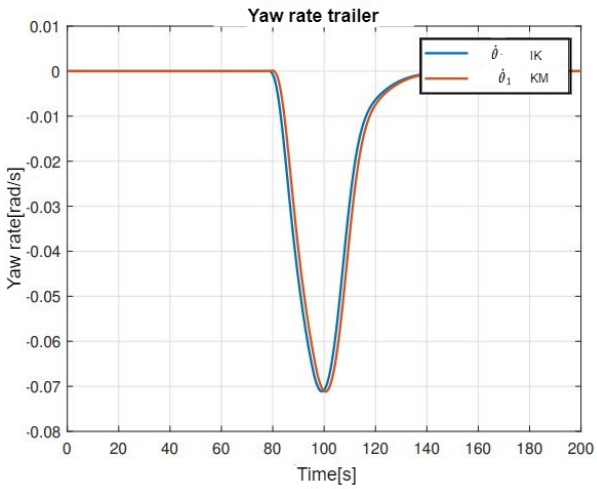
The Figure 4.16a shows pole-zero map of the reverse-semi trailer scenario and the Figure 4.16b shows the stable scenario of reverse-drawbar trailer combination. This instability in the reverse-semitrailer scenario has been resolved by using signum functions. For details on how the signum functions are introduced refer A.1.1. The Figure 4.17 shows the pole-zero map with poles on left half plane indicating the stability of the IK+KM and Bi-DPFC+IK+KM combinations after including signum function in the kineamtic relations. To compensate for the

inaccurate kinematic inversion as shown in Figure 4.18a resulted due to addition of the signum an improvement that offers correction to the kinematic inversion during these scenarios is proposed by introducing a virtual articulation angle subsystem in [13].

For a non-minimum phase scenario like the tractor and semitrailer combination in reverse direction the subsystem takes real articulation angle γ_r as input and converts it into virtual articulation angle γ_v with the reference to the stable drawbar trailer motion in reverse direction 4.16b, A.1.



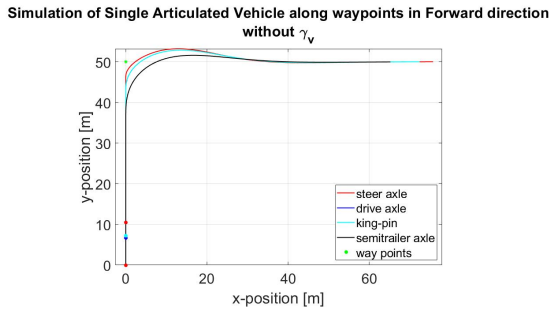
(a) Reverse-semitrailer without articulation angle subsystem [13].



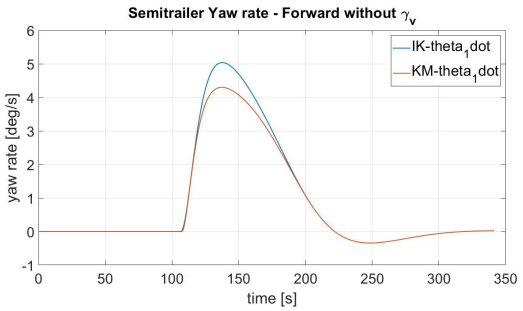
(b) Reverse-semitrailer with articulation angle [13].

Figure 4.18.: Difference in yaw rates caused during a 90deg turn for NMP scenario.

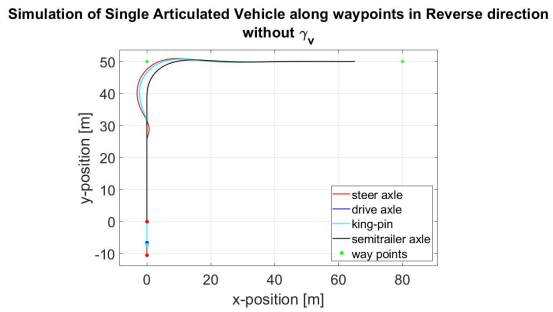
Finally the observed difference yaw rates due to inaccurate kinematic inversion has been minimized by including the virtual articulation angle correction based on the equation A.7 in [13]. The Figure 4.18b shows close to accurate kinematic inversion after introducing virtual articulation angle subsystem in the nmp Reverse-semitrailer scenario. The results presented in [13] state that stability is obtained in the IK+KM and Bi-DPFC+IK+KM combination and the kinematic inversion has been improvised by using the signum function and the Virtual articulation angle concept respectively. Ideally these improvements should also be achievable for any vehicle model dimensions. As the derivations in [13] were of general form. Therefore VAS, IK and KM combinations satisfy the need for forming a closed loop stable system required for the CL-RRT implementation.



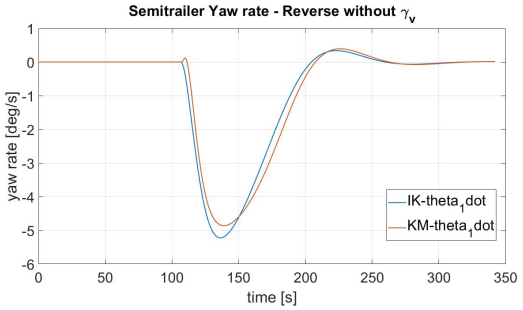
(a) Forward-semitrailer with articulation angle subsystem.



(b) Kinematic Inversion difference Normal scenario.

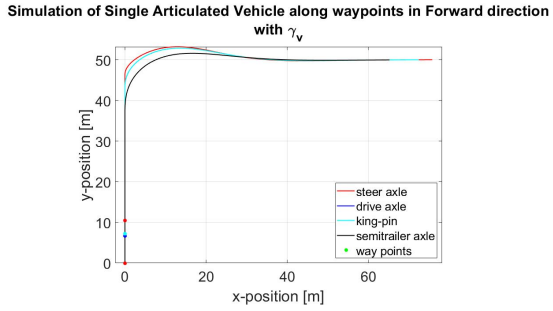


(c) Reverse-semitrailer with articulation angle Subsystem.

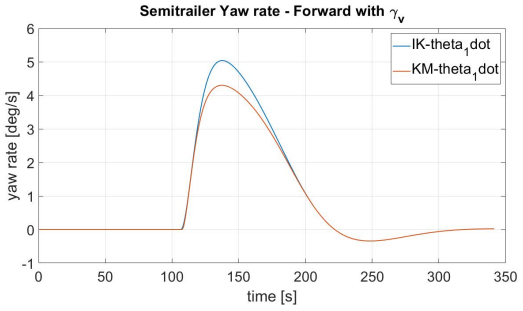


(d) Kinematic Inversion difference Non Minimum Phase (NMP) scenario.

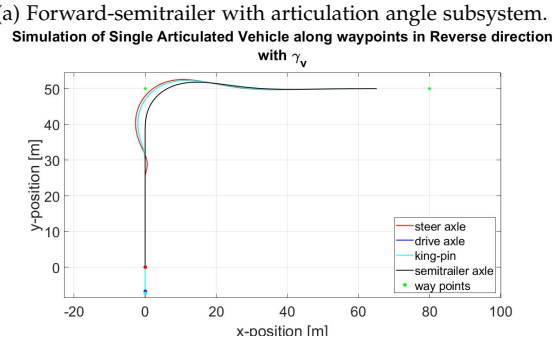
Figure 4.19.: Forward simulation of PPC+IK+KM over a 90deg turn for normal and NMP scenario with-out Virtual articulation angle subsystem.



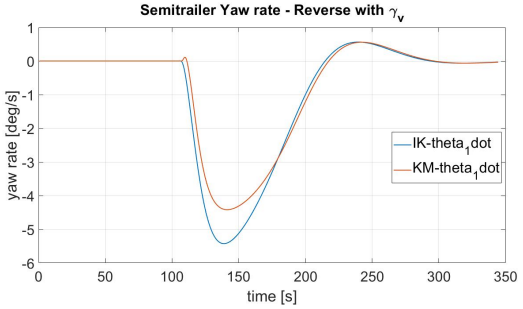
(a) Forward-semitrailer with articulation angle subsystem.



(b) Kinematic Inversion difference Normal scenario.



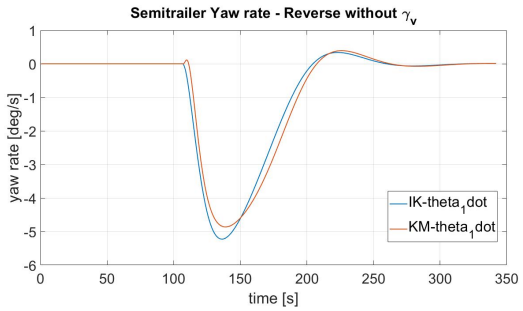
(c) Reverse-semitrailer with articulation angle Subsystem.



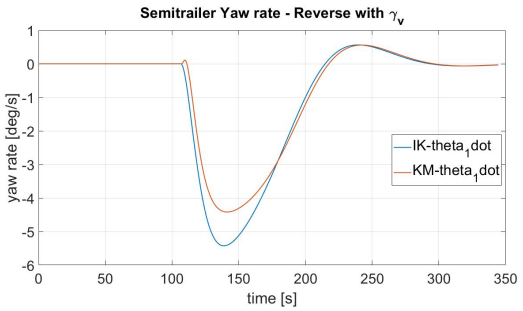
(d) Kinematic Inversion difference NMP scenario.

Figure 4.20.: Forward simulation of PPC+IK+KM over a 90deg turn for normal and NMP scenario with Virtual articulation angle subsystem.

But when the PPC is implemented instead of the Bi-DPFC, the trailer yaw rate plot similar to the one used in [13] Figure 4.18 for the normal and NMP scenarios, the results obtained were not as expected and are presented in the Figures 4.19 and 4.20.



(a) Kinematic Inversion difference NMP scenario.



(b) Kinematic Inversion difference NMP scenario.

Figure 4.21.: Forward simulation of PPC+IK+KM over a 90deg turn for NMP scenario with-out and with Virtual articulation angle subsystem.

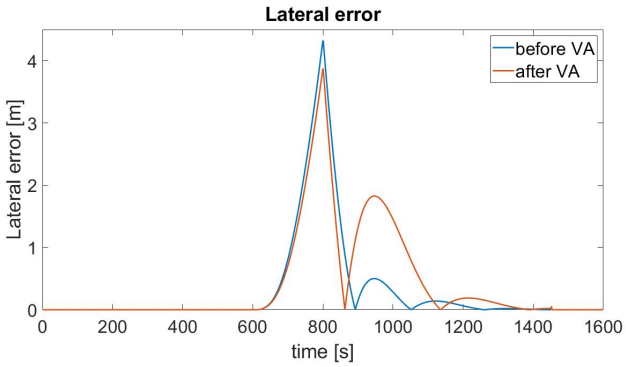


Figure 4.22.: path tracking error before and after virtual articulation

Table 4.2.: Comparison of lateral errors with and without Virtual articulation Subsystem

Error Type	**PPC without γ_v**	**PPC with γ_v**
Mean Squared error	0.7228	0.7639
Maximum error [m]	4.323	3.874

The Figures 4.19 Shows the results obtained when the PPC+IK+KM has been forward simulated over a 90° turn in Forward and reverse motion *without* VAS. Followed by the results in Figures 4.20 where PPC+IK+KM has been forward simulated over a 90° turn in Forward and reverse motion *with* the VAS.

For the normal scenario of forward-semitrailer the VAS does not influence the system. It is interesting to make the comparison between the results obtained without and with the VAS in Figure 4.21 for NMP scenario. Although, from the Table 4.2 it can be seen that inclusion of the VAS has reduced maximum error by $\approx$ 0.5 - 0.7 meters, the mean squared error which is a measure of overall deviation of traversed path from the reference path was obtained less for the case without vas). From the PPC perspective MSE is of more importance as the steering effort by the controller always aims to track the reference path as closely as possible and the observed maximum error is acceptable as the CL-RRT planner stitches the discrete paths produced by the PPC. Also from the Figure 4.21 it can be seen that the accuracy of the kinematic inversion is compromised after including the VAS. To confirm the results obtained

above, the combination is forward simulated over a series of way points that form linear line segments representing the reference paths generated by the RRT motion planner as in the Figure 4.23. The results obtained in Table 4.3 and Figures 4.24, 4.25 show similar pattern as in the in Table 4.2 and Figures 4.21, 4.22

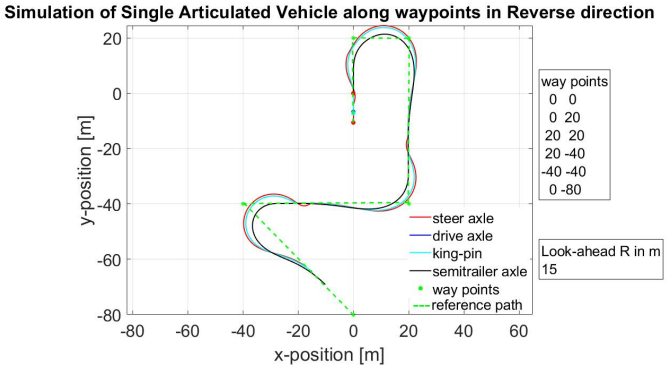
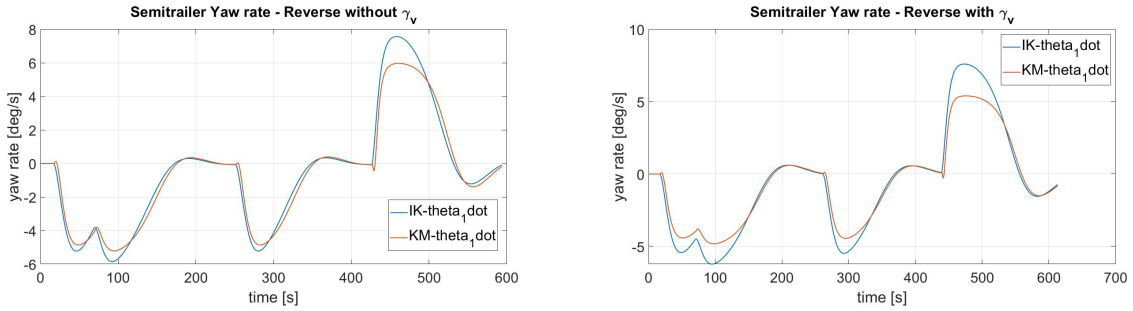


Figure 4.23.: vehicle traversed path with virtual articulation



(a) Kinematic Inversion difference NMP scenario.

(b) Kinematic Inversion difference NMP scenario.

Figure 4.24.: Forward simulation of PPC+IK+KM over a a series of line segments for NMP scenario with-out and with Virtual articulation angle subsystem.

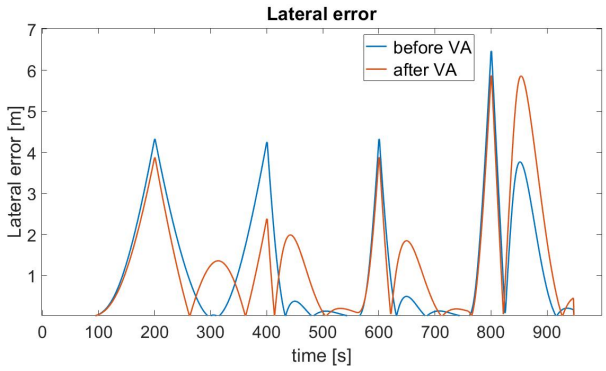


Figure 4.25.: path tracking error before and after virtual articulation

The PPC in combination with digital filter discussed in 4.1.3 offers very good steering but has poor tracking performance as shown in the Tables 4.2 and 4.3 when compared to the Bi-DPFC which has a minimal tracking error of close to 0.1 m. This drawback will not be of concern for CL-RRT implementation as piece wise extensions are joined together to obtain the

Table 4.3.: Comparison of lateral errors with and without Virtual articulation Subsystem over piece wise linear reference paths

Error Type	PPC without γ_v	PPC with γ_v
Mean Squared error	3.0324	3.3390
Maximum error [m]	6.461	5.866

final solution path. Considering the MSE as deciding factor from Tables 4.2 and 4.3 the virtual articulation angle subsystem is omitted from the vehicle model that will be forward simulated in the loop for CL-RRT framework.

Finally the following are the finalized subsystems that, are used for forward simulation.

- Inverse Kinematic model of SAV
- Forward Kinematic model of SAV
- PPC
- Digital Filter

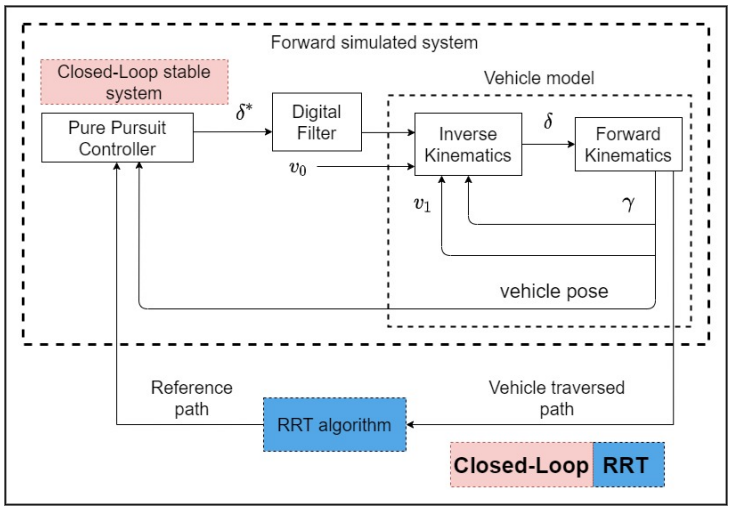


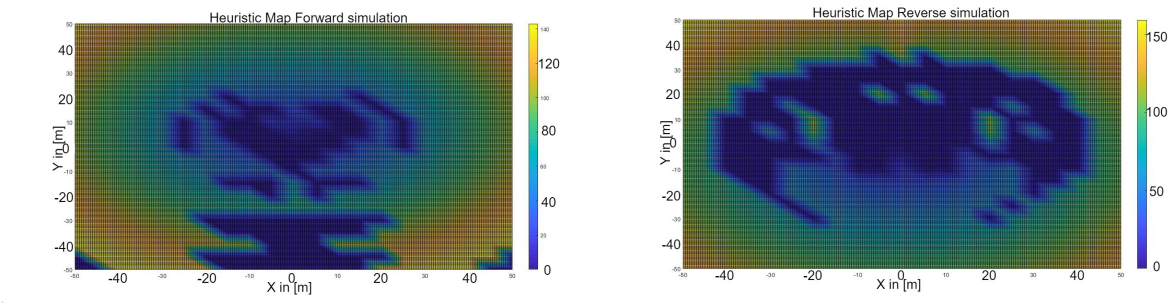
Figure 4.26.: Finalized components that will be forward simulated in the loop of CL-RRT frame work

The Figure 4.26 shows the flow of data between these subsystems along with the rest of the CL-RRT components.

4.1.6. Heuristic Map Uniformity

The pure pursuit controller subsystem and Vehicle model as shown in Figure 4.26 will be forward simulated over the predefined region of points over a grid of 50m x 50m with resolution of 0.5m. Starting from origin the Heuristic map generation algorithm iteratively forward simulates the vehicle model and controller combination to each individual point in the grid. A decision should be made here on deciding when the forward simulation should be stopped with respect to each point in the grid. Intuitively one would think the individual forward simulation should stop when the vehicle control point reaches the selected goal

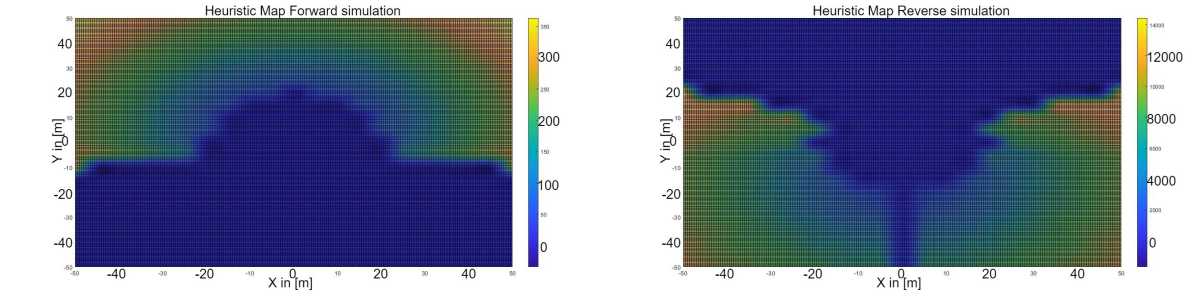
point from the grid. Such approach has been implemented and the obtained results were not satisfactory, see Figure 4.27.



(a) Heuristic map Forward motion with pure pursuit

(b) Heuristic map Reverse motion with pure pursuit,

Figure 4.27.: Distance metric map or Heuristic map for Forward and reverse motion when propagated until control point P^* 3.6 reaches each grid point in 50m X 50m



(a) Heuristic map Forward motion with pure pursuit

(b) Heuristic map Reverse motion with pure pursuit

Figure 4.28.: Distance metric map or Heuristic map for Forward and reverse motion when propagated until Intersection point P 3.6 reaches each grid point in 50m X 50m

The decision of when to stop the individual forward simulation plays a important role in the quality of the map obtained. It is observed that the type of the controller and the decision strategy of when to stop the forward simulation are linked.

The earlier choice of the decision strategy i.e, propagating the forward simulation until the control point P^* 4.31b reaches the grid point, will fail as the pure pursuit controller has not been tuned for the bounded lateral error. During the propagation there is lateral error which has to be accepted at a cost of the steering performance it offers. The trick here is to stop the individual forward simulation when the look-ahead point P 4.31b comes close to the goal point selected from the grid, which is on the linear reference path. By doing so we do not consider the positional error or heading error of the vehicle w.r.t to the selected goal point. Quite meaningful when looked from the controllers perspective of tracking reference paths.

If piece wise spline segments were chosen in combination with controller, then the strategy of forward simulating the vehicle until the vehicle control point P^* reaches the goal point would make sense as the controller is capable of reaching the goal point with some bounded error.

This observation that the type of controller, reference paths chosen and the decision of when to stop the forward simulation is of great importance to obtain a meaningful and uniform Heuristic map. In [6], the source which is taken as reference for CL-RRT algorithm implementation does not elaborate on this decision making aspect. During this thesis this step was not obvious and was actually identified by observing the demonstration videos published

by the author of the [6],[14]. Where the simulation videos clearly show a look ahead circle, piece wise linear reference path, the pure pursuit geometry and the point of intersection all in a play while tracking the piece wise linear reference path.

The Figure 4.28 shows the results obtained by following the decision strategy based on the look ahead circle and has given satisfactory results which helped later on while building the planning tree and obtaining solution paths.

4.1.7. Integration of the controller with vehicle model and CL-RRT algorithm

Once the controller, Vehicle model components are ready they have to be integrated into the loop of algorithm that generates heuristic map and later on into the loop of CL-RRT algorithm.

Initially the controller and the vehicle model which are built in Simulink are iteratively called by a Matlab script. Two Matlab scripts are written one for heuristic map generation and the second for CL-RRT algorithm. The observation here is that, every time a Matlab script calls a Simulink model, the Simulink model re-compiles to included the recent changes made by the Matlab script. This method of calling the Simulink model iteratively leads to repeated compilation of the Simulink model which is time consuming process. As the heuristic map has to be generated offline it is not a problem to use such script to calculate the heuristic map. But when it comes to the CL-RRT algorithm implementation such method cannot be used as the time required to obtain the solution path is of great concern.

Following methods looked like possible solutions to address the problem.

- MATLAB Fast restart
- CL-RRT algorithm in simulink
- Simulink model to c language using c coder

The *first option* although works for the heuristic map generation, cannot be used to for CL-RRT algorithm implementation as the Fast restart method works by saving the Simulink model operating point and during the next call the Simulink model will be re-initialized at the same operating point. The Simulink model built is a combination of a vehicle model and controller which needs to be completely reset before initializing the next iteration. Hence MATLAB Fast restart cannot be used based on the current requirement.

The *second option* is to built the whole CL-RRT algorithm with in the Simulink model and hence avoiding the problem of recalling and repeated compilation. This is achieved by using the Stateflow environment of the simulink. Where individual components of the CL-RRT algorithm are defined as states and by defining the transition conditions the flow of the control with in the algorithm can be controlled, Refer to the Simulink Stateflow diagram 4.2.3. This approach worked and was fast than the initial approach of calling the Simulink model iteratively from a Matlab script file.

The *Third option* definitely will reduce the time required to obtain the solution path drastically as there are no high level languages like MATLAB or Simulink involved. The simulink components i.e, Pure pursuit controller, vehicle model including the IK and KM subsystems can be converted to c language using Matlab product, c-coder and the RRT algorithm can be implemented directly in c language. Due to the limited time frame available for the thesis, method three could not be implemented but definitely is a possibility to obtain a solution path in a real time scenario implementation.

4.2. Implementation method

In this section flowcharts for two major algorithms that are needed for complete implementation of the CL-RRT motion planner are presented.

4.2.1. Heuristic map generation

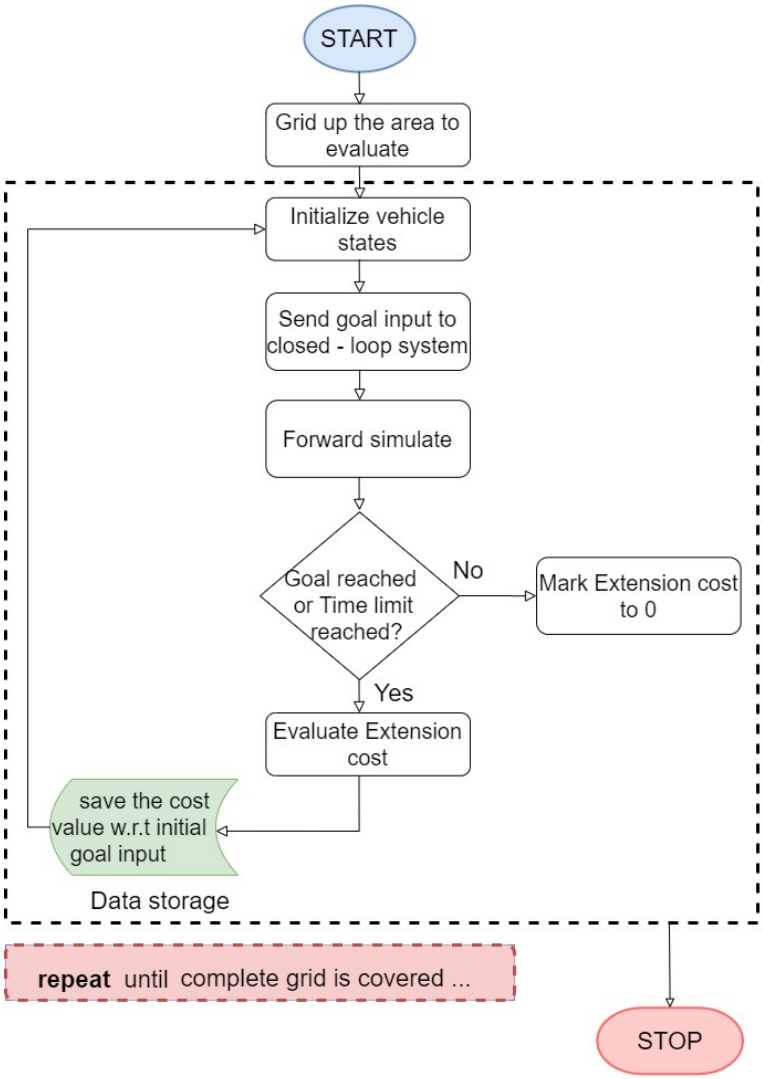


Figure 4.29.: Flowchart for heuristic map generation

A quick look at the flowchart of the CL-RRT motion planner in Figure 4.32 shows that a heuristic map generated off line has to be fed as input in the beginning of the motion planning. In this section the options available for heuristic map generation and approach taken to obtain one that contributed in generation of solution paths for this thesis is discussed in detail.

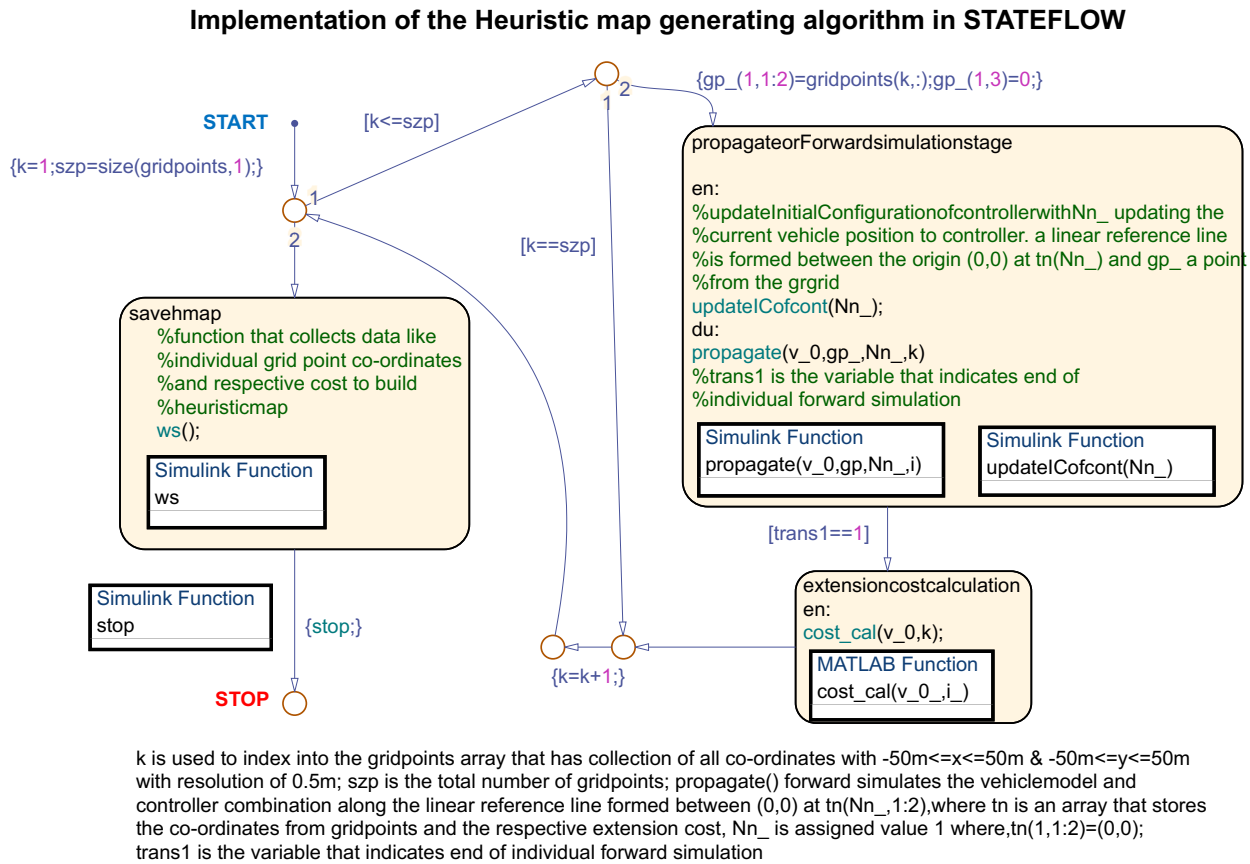


Figure 4.30.: Simulink Stateflow implementation

In [15] a RRT framework where reference paths were built by connecting splines. With such reference paths a possibility to include Bi-DPFC exists as the controller is very good at tracking the reference path stably that is smooth and subject to maximum curvature 1/R, where R is found 9 meters for the current vehicle dimensions.

In literature a Pure pursuit controller is used in implementation of the CL-RRT. At this point two possible options were available to achieve a path planner based on CL-RRT. Accordingly Heuristic map had to be generated. The option are listed below,

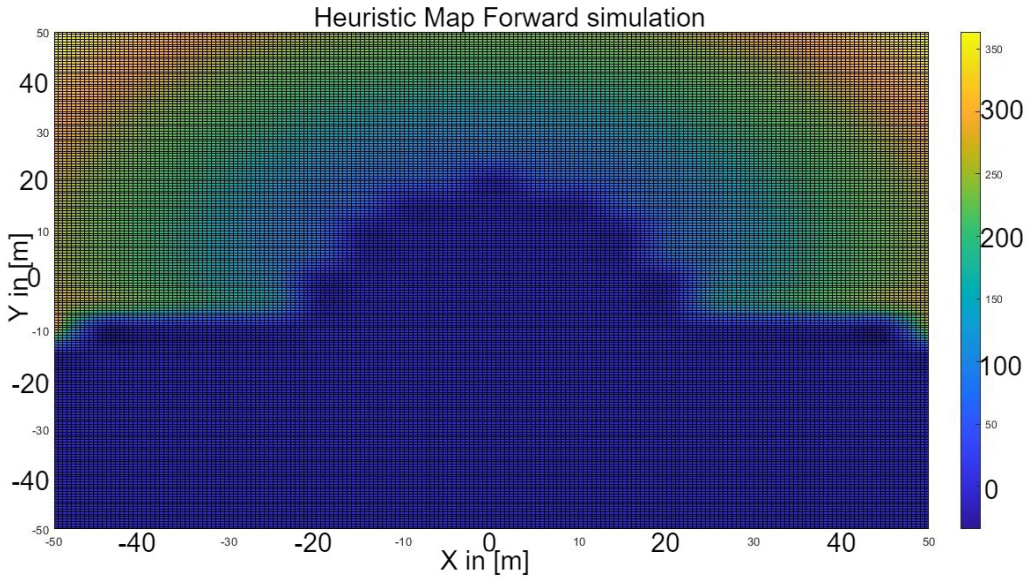
- use the combination of spline based reference paths and Bi-DPFC
- use the combination of linear reference paths and a pure pursuit controller

Given the limited time frame for the thesis quick decision had to be made. Although option one looked simple and direct, option two was chosen as it seemed to be more convincing in a way of achieving solution paths that are not bound by any curvature limitations and yet kinematically feasible to achieve by the SAV.

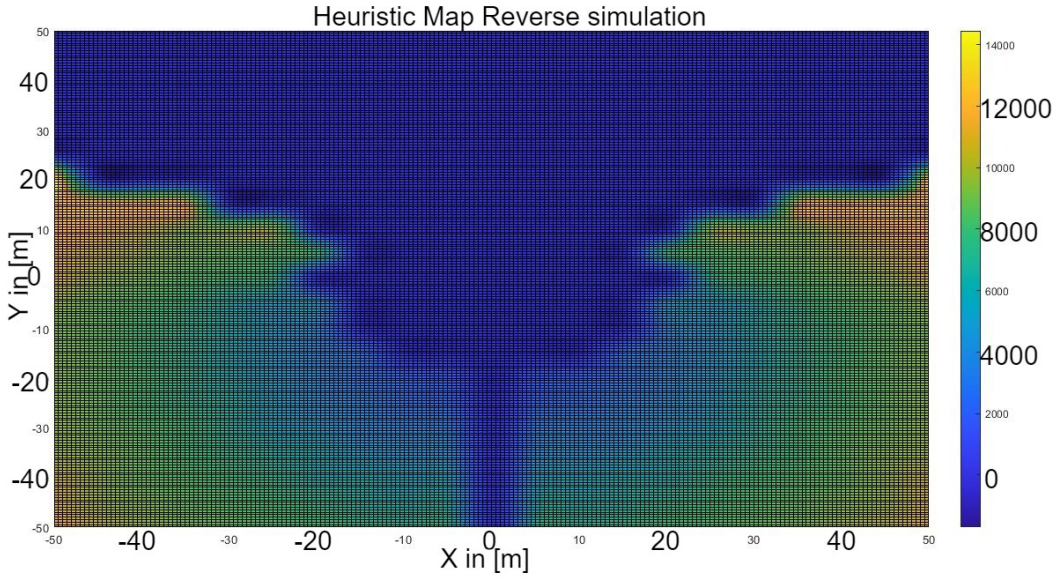
The procedure to generate the Heuristic map is elaborated below. The Figure 4.29 shows the flow chart for heuristic map generation. And the Simulink Stateflow implementation can be seen in the Figure above 4.30.

The planning tree based on RRT builds its extensions by performing forward simulations, starting from a suitable nearest neighbour to the new sample point. A method has to be

defined to decide on the suitable nearest neighbour. In this thesis a heuristic map based on the distance metric is used to find the suitable nearest neighbour. The distance metric has to be decided based on the vehicle kinematic constraints. For example distance metric for an omni-directional vehicle with only holonomic constraints can be as simple as euclidean distance. But if such distance metric is applied to complex vehicle kinematics like SAV with non-holonomic constraints the task of finding a suitable nearest neighbour with such distance metric will be flawed. And obtaining a solution path will be remote and mostly impossible.



(a) Heuristic map Forward motion with pure pursuit



(b) Heuristic map Reverse motion with pure pursuit

Figure 4.31.: Heuristic map for Forward and reverse motion

In this thesis a distance metric that considers the non-holonomic constraints and vehicle kinematics of a SAV is evaluated. The closed loop stable system which includes vehicle model of SAV, PPC along with the steering rate controlling digital filter is forward simulated from an initial start position at origin to each and every point in the predefined grid. The traversed

path of the control point P which is at the mid point of the semitrailer drive axle is recorded for every individual simulation along side the respective grid point. The arc length of the path traversed is calculated and recorded as a heuristic cost. When costs for each and individual grid point is obtained and plotted against the co-ordinates of the grid region the required Heuristic maps for forward and reverse direction are obtained as shown in the Figure 4.31.

As a human driver naturally prefers forward motion compared to reverse motion due to the ease of vehicle control the motion in reverse direction is punished by adding more weights resulting in higher costs. This can be seen by comparing the difference in the cost values in the Forward and reverse Heuristic maps. Doing so the motion planner chooses the solution paths that have forward motion maneuvers wherever possible.

Also it is important to have paths that are smooth and comfortable for the driver to track in the reverse motion. To achieve this the change in angles and long distances are punished by adding more weights. This results in the motion planner producing solution paths that are possibly shorter and smooth.

All grid points which were not reachable during the forward simulation are marked with an extension cost of 0.

The equations used for calculating the cost are listed below. As mentioned earlier the costs are evaluated differently based on the direction of the motion.

$$\begin{aligned} E_{fwd} &= \sum_{i=1}^{N}(10(x_1[i]-x_1[i-1])^2+10(y_1[i]-y_1[i-1])^2) \\ E_{rev} &= \sum_{i=1}^{N}(10(x_1[i]-x_1[i-1])^2+10(y_1[i]-y_1[i-1])^2) \\ &\quad +10^4(\gamma[i]-\gamma[i-1])^2+10^4(\theta_1[i]-\theta_1[i-1])^2) \end{aligned} \tag{4.2}$$

Where $(x_1[i]\ y_1[i])$ are the control point P co-ordinates at the mid point of the semitrailer drive axle. $\gamma[i]$ is the articulation angle of the SAV, $\theta_1[i]$ is the heading of the semitrailer.

4.2.2. CL-RRT implementation

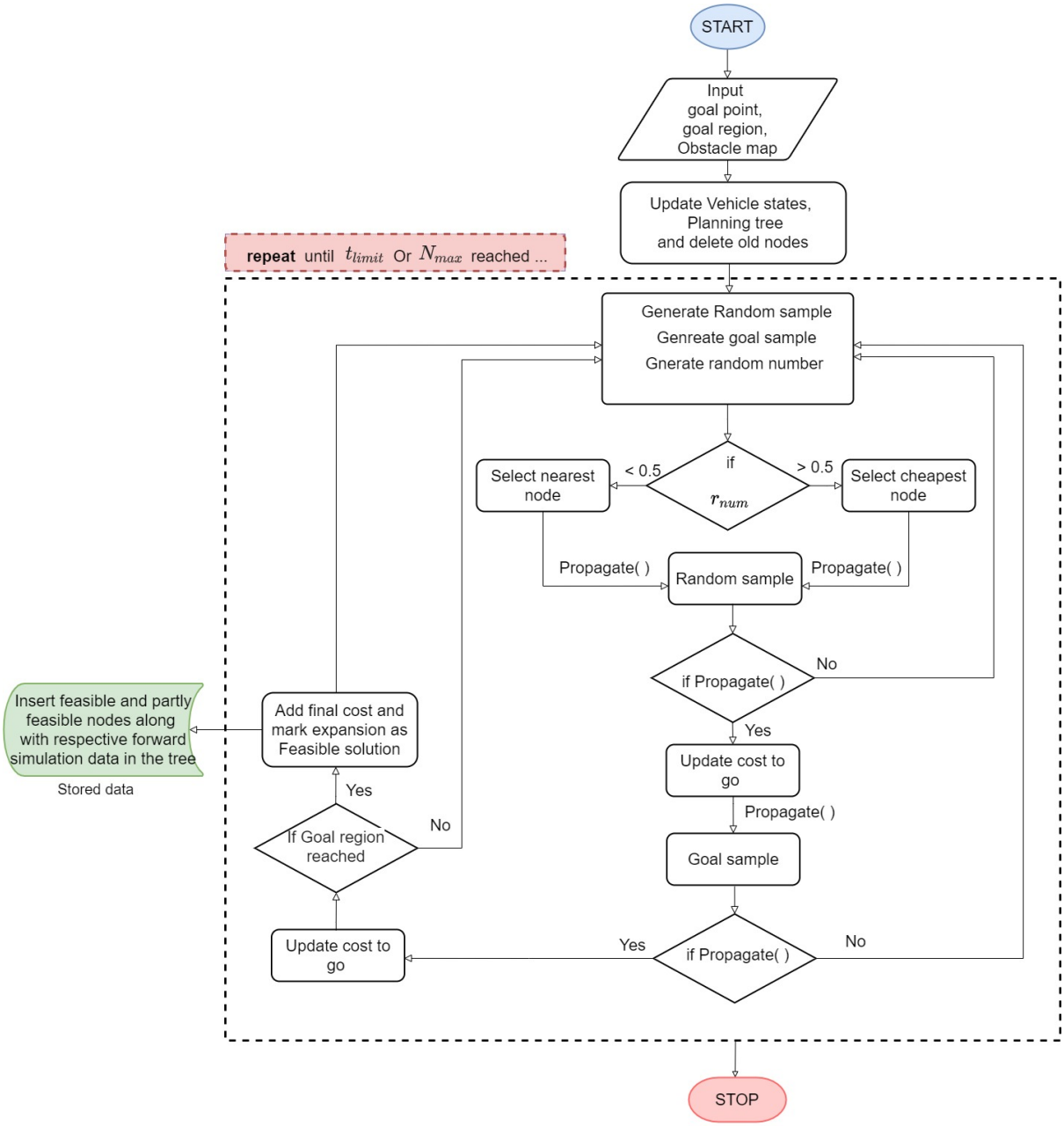


Figure 4.32.: Flowchart for CL-RRT motion planner

4.2.3. CL-RRT implementation

The Figure 4.32 shows the flow chart that elaborates the CL-RRT algorithm. Following steps are performed before entering into the planning loop.

- deleting any collected old nodes from the previous simulations
- initiated with inputs like Initial and the goal configurations
- define Maximum number of nodes N
- setting Goal region constraints 3.4

- input Obstacle map 2.2.3

The starting configuration is stored as a root node in the planning tree. Then the planner is set to enter a loop shown in box with dashed line in Figure 4.32, where it remains until maximum number of nodes N_{max} is reached. All the solution paths obtained during this loop are collected and evaluated to find an optimal path whose optimal nature is evaluated based on the arc length of the solution path.

Once the loop is entered, the planner generates two sample points based on the biasing strategy discussed in 3.6.1.

- First sample point called Random sample is generated with an intent of exploring the region of interest. The Figure 3.7 shows a collection of sampled points (inside the box with dashed line), based on the exploring strategy.
- Second sample point called Goal sample is generated with an intent of biasing the planning tree towards the goal region. The Figure 3.7 shows a collection of sampled points (inside the yellow circle), based on the biasing towards goal strategy.

The vehicle is forward simulated (or propagated with the function propagate( ) in flow chart, refer to A.2 for the actual implementation of the vehicle and controller model in Simulink) over the linear reference path between the root node and the Random sample and the obtained vehicle traversed path co-ordinates and respective extension cost 4.2 are stored. The Function Propagate( ), (refer 4.32 and also the Simulink Stateflow diagram presented in this section) also implements a collision check method based on the strategy discussed in 3.7.

If this extension was successful i.e, kinematically feasible and collision free, then the vehicle is forward simulated over the linear reference path starting from the Random sample to the Goal sample and the obtained vehicle traversed path co-ordinates and respective extension cost and the cumulative extension cost are stored.

After every attempt to reach the Goal sample a goal region reach-ability check is performed where the final configuration of the vehicle is checked for its presence within the goal region constraints defined earlier in 3.4. If the the reach-ability test is passed all the respective nodes starting from the root node to the final goal sample node are collected and stored as solution path reference nodes inside the planning tree. The Figure 5.2 shows these solution path reference nodes in yellow dots, when joined form a piece wise linear solution reference path shown in yellow lines between initial and goal sample node.

In case of any extension failure i.e, no kinematically feasible path available or collison check method failed, the function propagate( ) returns 0 and the flow within the flowchart is directed to the block of generating the two sample points discussed previously.

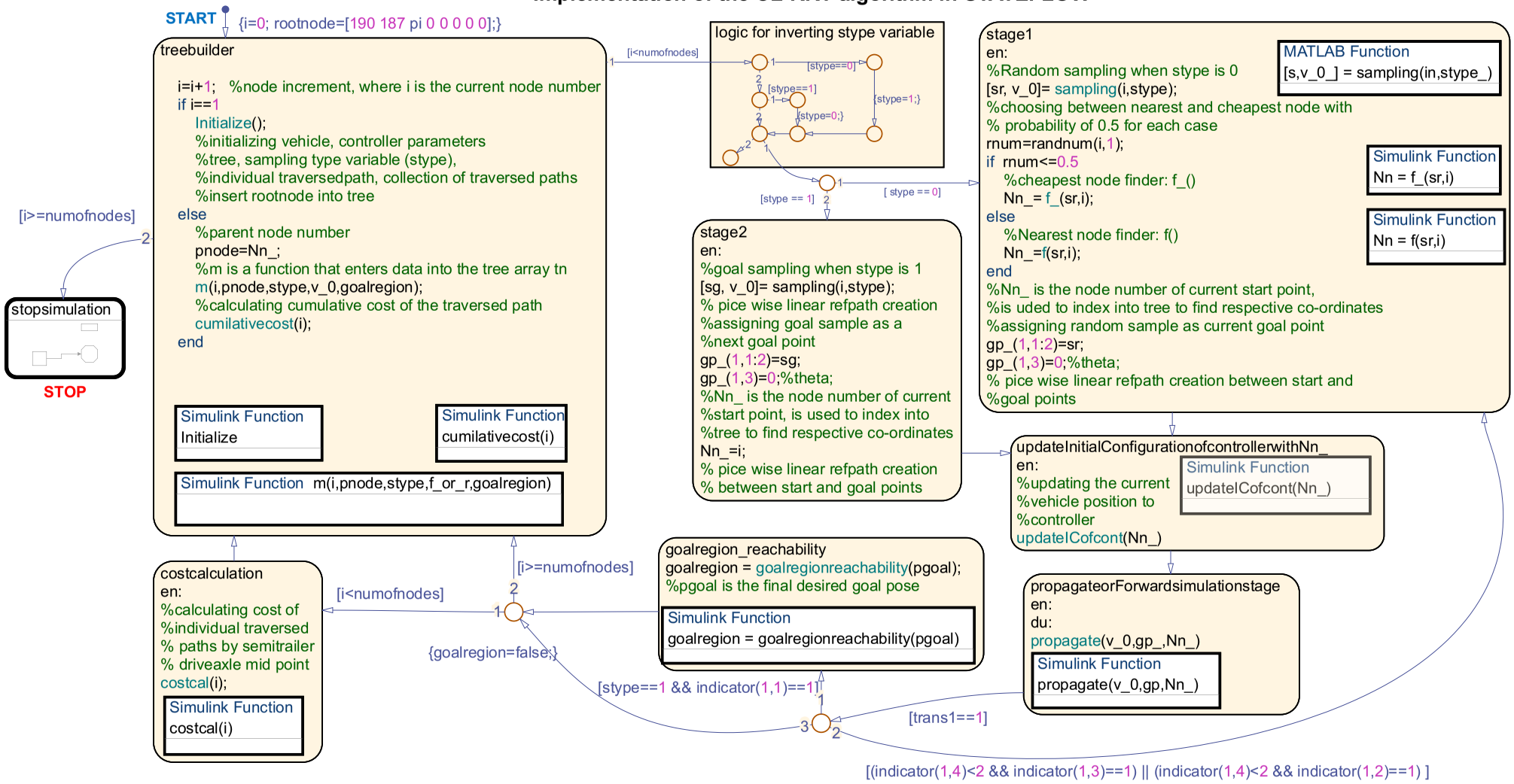
From the second iteration, a strategy to obtain a optimal path as discussed in 3.6.2 comes into action. Two methods are implemented with a probability of 0.5 assigned to each method.

- Nearest neighbor 3.6.2
- Cheapest neighbour 3.6.2

They strive to produce the optimal paths while building the tree. The Heuristic map 4.31 presented in the previous section will be put to use while finding the nearest neighbour. 2D-Lookup tables A.4 are used to retrieve the data from the heuristic map stored offline inside an 2D array.

The cheapest neighbour is selected from the existing nodes of the tree based on the cheapest cumulative cost. In this thesis the planning tree is run until maximum number of nodes N_{max} are reached. Another possibility is to run the loop until desired number of solution paths are obtained or maximum time is reached.

Implementation of the CL-RRT algorithm in STATEFLOW



v_0 & v_1are the logotudinal velocity of tractor and semitrailer set to 1m/s; stype is a variable used to select random sample, (sr) or goal sample, (sg) within the sampling(); trans1 is the variable that indicates end of individual forward simulation; Indicator size 1x4 contains boolean values representing true false for sample point reachability, nan values that might result due kinematic infeasibility,obstacle collision and maxtimelimit reach set for each individual forward simulation. Any of these four factors can stop the individual simulation and continue to next; numofnodes is maxlimit on number of nodes

5. Validation and Results

5.1. Demonstration of the SAV performing complex maneuvers in static obstacle space

In this section various docking maneuvers solution paths for SAV configuration, generated by the CL-RRT algorithm are presented. Initially One such instance of the docking maneuver is explained by highlighting crucial steps that lead to such solution path. The Figure 5.1 shows the exploring strategy implemented. The table 5.1 gives the parameters set to achieve this exploring strategy. The Figure 5.2 shows the piece wise linear reference path that is generated by the RRT algorithm working in the loop with the closed loop stable system that represents SAV. This piece wise linear reference path is plotted from the information collected in the planning tree variable during the planning simulation in Simulink. The set of nodes that form this piece wise linear reference path can be used to reproduce the collision free vehicle traversed path by forward simulating the previously used close - loop stable system for solution path generation, that constitutes the vehicle model, pure pursuit controller and the digital filter.

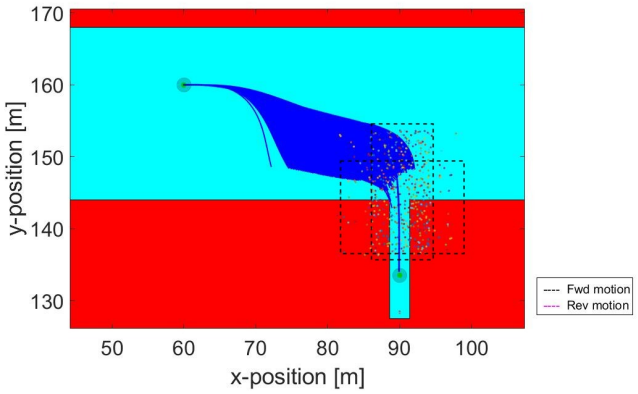


Figure 5.1.: planning tree expansion using the exploring strategy. Refer to the table 5.1 for the parameters set to obtain this exploring strategy.

Table 5.1.: exploring strategy parameters

Strategy	**Direction**	**Bias/**$[x_0, Y_0]$	**Standard deviation**	**Probability**
Exploring	Reverse	[90 145]	[5 5]	0.2
Exploring	Reverse	[90 145]	[2.5 5]	0.5
Exploring	Reverse	[90 145]	[2.5 12]	0.3

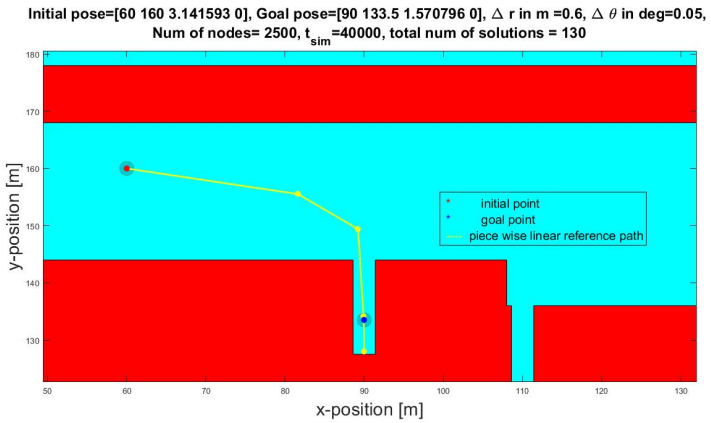


Figure 5.2.: piece wise linear reference path only

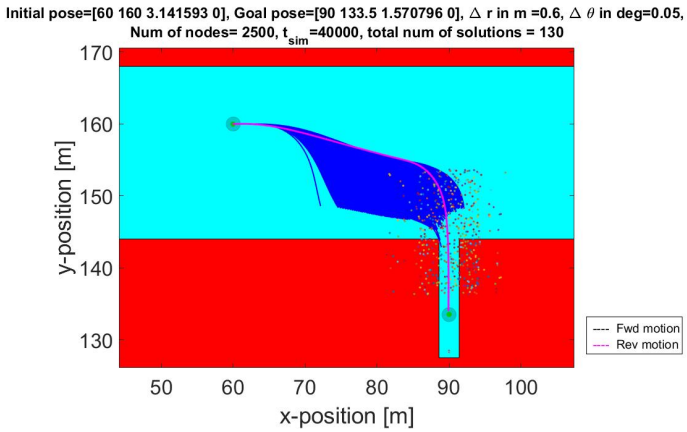


Figure 5.3.: planning tree and the sampling strategy

The Figure 5.3 shows the solution path obtained and the data related to that particular information like maximum number of nodes the tree was extended to, time to expand these number of nodes, total number of solution obtained during this particular planning simulation, initial pose and the goal pose of the vehicle, the goal region constraints set to achieve this solution paths. The Figure 5.4 shows the demonstration of the vehicle boundary model successfully traversing the C_{free} by avoiding the $C_{obstacle}$ using the collision detection method 3.7 employed in the algorithm.

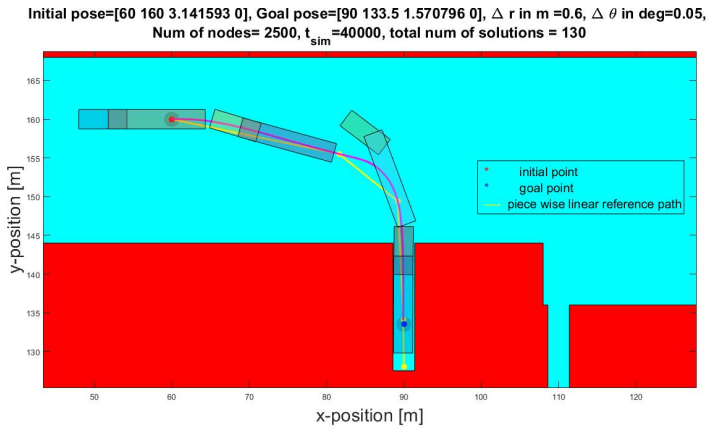


Figure 5.4.: Piece wise linear Reference path and respective solution path combination can be seen in this figure

The Figures 5.5 and 5.6 demonstrate an important aspect of the CL-RRT planner. The optimal nature is defined quantitatively based on the curve length of the solution path in this thesis. The Figures demonstrates the probabilistically optimal nature of the CL-RRT planner where it can be clearly seen that the planner has attempted to achieve the optimal path given enough computational time. The influence of the goal region constraints on the time taken to obtain an optimal solution path is also observed during the simulations. It is observed that having adequately larger goal region constraints has produced optimal solution paths in shorter time or(lower maximum number of nodes limit) than having very strict goal region constraints. For the solution path in Figure 5.5 the goal region constraint parameter, r is 0.5 m and θ_1 is 2°, where as for solution path in Figure 5.6 the goal region constraint parameters r is 0.3 m and θ_1 is 5°. Given the same initial and final goal configurations this difference in the solution paths highlights the possibility of selectively choosing the best solution path among the obtained solution paths. In this particular parallel parking scenario one can easily put less weight on the r and choose the shortest path solution with minimum changes in the heading of the vehicle.

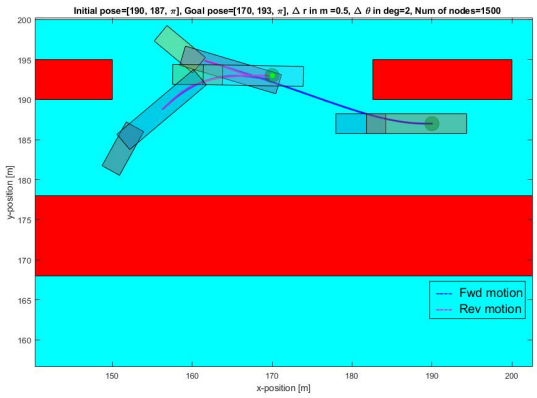


Figure 5.5.: Parallel parking solution one

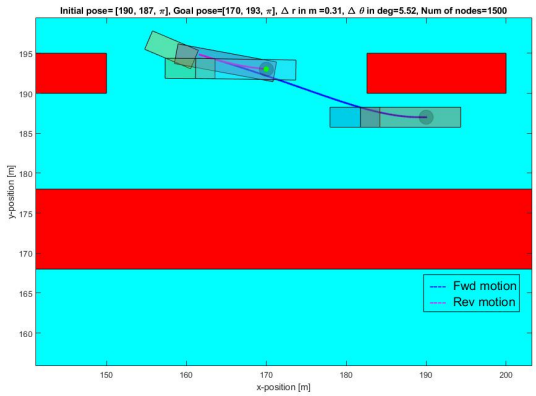


Figure 5.6.: Parallel parking solution two

The width of the docking station used in the obstacle map is smaller than the real width of the docking station in real distribution center. It is important to observe here that such arrangement also aids in finding the solution path with minimal heading error faster. This is because the forwards simulation model does not waste time building the tree extensions that partially enter the docking lane and cannot proceed further. Instead it strives to enter the docking station lane with minimal error and is sure that it will reach the goal region in the forthcoming extensions.

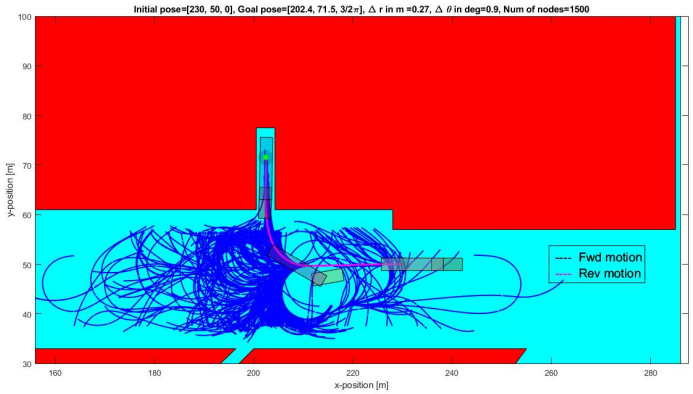


Figure 5.7.: SAV boundary model tracking solution path by avoiding obstacles

The Figure 5.7 shows another example of docking maneuver in a different docking lane of the distribution center. In this Figure 5.7 the expansion of the planning tree shows that a more generalized exploring strategy has been set, leading to exploring of the regions which are not of interest.

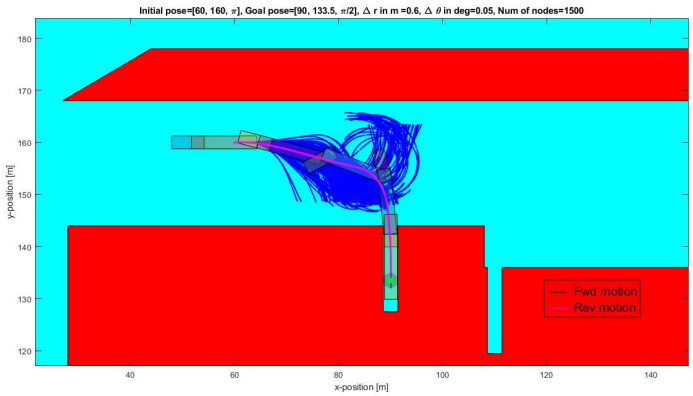


Figure 5.8.: SAV boundary model tracking solution path by avoiding obstacles

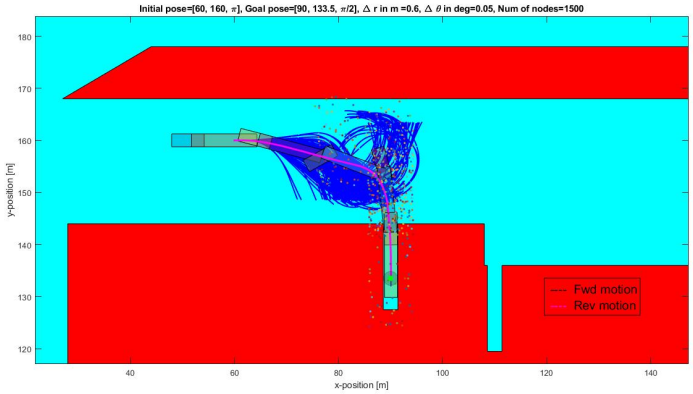


Figure 5.9.: SAV boundary model tracking solution path by avoiding obstacles

By comparing the exploring strategy and solution path in Figure 5.3 to solution path and exploring strategy in Figures 5.8 and 5.9 the probabilistically complete nature of the CL-RRT can be observed. Where the goal configuration has been reached in both cases with variations in the exploring strategy. Probabilistically complete means given enough computation time the goal region is assured to be reached with probability reaching 1. In both Figures although the initial and final configuration are same different exploring strategies are employed and this can be clearly depicted by the way the planning tree has expanded.

5.2. Validation of the obtained paths

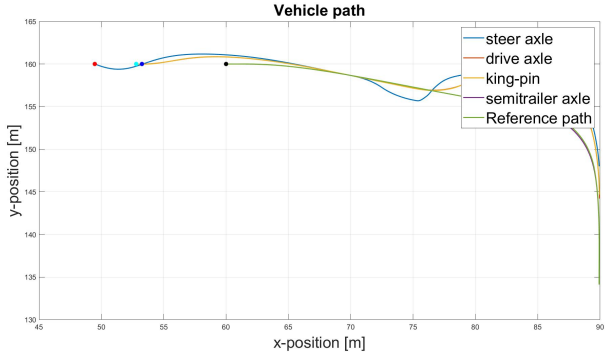
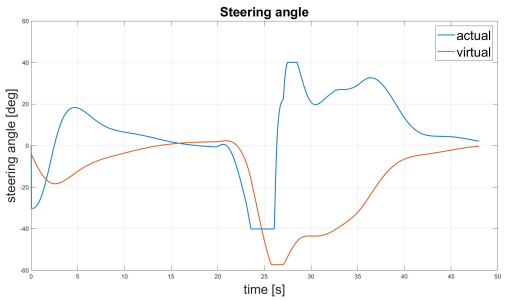
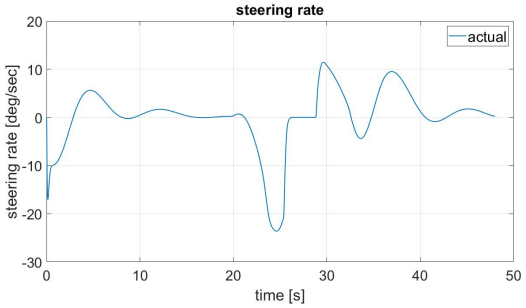


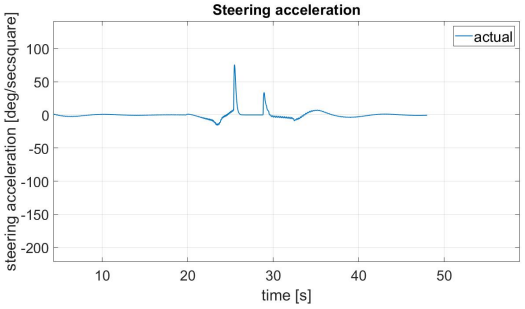
Figure 5.10.: vehicle traversed path obtained when Bi-DPFC is forward simulated of the solution path 5.3.



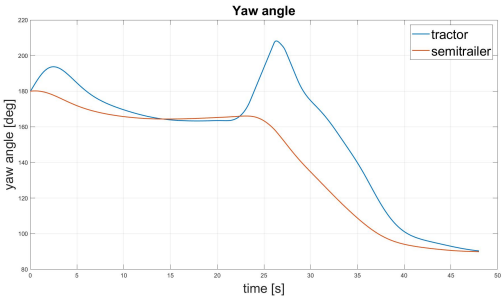
(a) Steering angle demanded by the Pure pursuit controller to follow the solution path



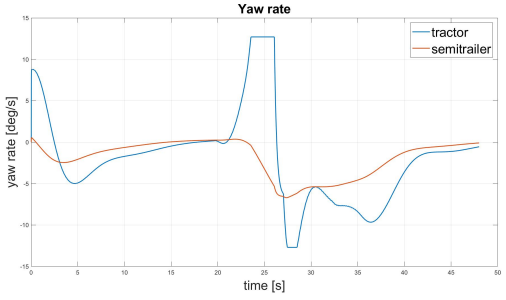
(b) Resulting steering rate when solution path 5.3 is followed by Bi-DPFC



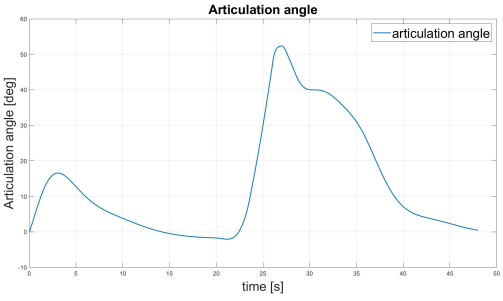
(c) Resulting steering acceleration when solution path 5.3 is followed by Bi-DPFC



(d) Resulting Yaw angle when solution path 5.3 is followed by Bi-DPFC



(e) Resulting Yaw rate when solution path 5.3 is followed by Bi-DPFC



(f) Resulting articulation angle when solution path 5.3 is followed by Bi-DPFC

Figure 5.11.: Results obtaine when solution path 5.3 is followed by Bi-DPFC

The Bi-DPFC as discussed earlier is developed with the intention of tracking a reference path provided by the motion planner with minimal lateral error. The output of this controller is the vehicle steering angles which are linked to a driver assistance system which is available with the driver to receive the support during the complex maneuvering of the vehicle.

Hence forward simulating the obtained reference path with this controller is a proper validation technique to examine the steering angle performance and kinematic feasibility of the solution paths generated by the motion planner.

The Figure 5.10 shows vehicle model traversed path obtained by forward simulating the Bi-DPFC and vehicle model combination over the solution path 5.3 obtained by the CL-RRT motion planner.

The results obtained show that the Bi-DPFC can successfully follow the solution path with lateral error as minimum as 0.1m as shown in the Figure 5.12. Max Steering rate close to 20 degrees/sec was achieved by introducing the digital filter.

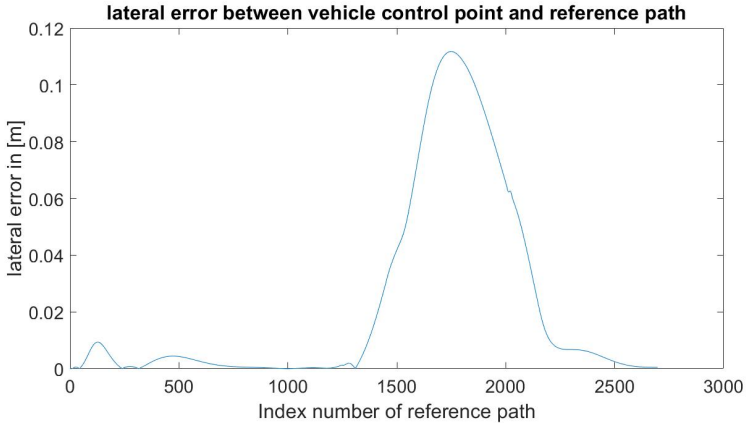


Figure 5.12.: Lateral error in [m] between vehicle traversed path and the reference solution path 5.3.

6. Conclusion and Recommendations

6.1. Conclusions

Bidirectional path planning for docking maneuvers has been achieved using the CL-RRT planner under the assumption of low speed maneuvering at constant speed of 1m/s and no-slip condition on flat surface. Inertial properties of the vehicle and tyre ground interaction forces are neglected. The solution paths generated using the CL-RRT planner are feasible and probabilistically optimal as discussed in section 5.1 w.r.t the results shown in the Figures 5.5, 5.6.

The vehicle model components like Inverse kinematics and the forward kinematics are based on general recursive formula that describes a general N-trailer system. Reasons for not including the Bi-DPFC [12] in the loop of the CL-RRT are discussed. A virtual tractor principle based vehicle model and pure pursuit controller, combination has been used to produce backward and forward motion of a SAV. Reasons for not including the VAS in the vehicle model are discussed. The controller's output is treated with a filter/differentator to acquire a steering output that is achievable by the driver.

The solution paths generated are inherently kinematic as the CL-RRT planner builds the extensions in the planning tree incrementally by forward simulating a mathematical vehicle model built on kinematic relation ships and constraints of the SAV and this is validated using the Bi-DPFC. The controller was able to track the generated solution paths with maximum path tracking error of 0.1m as shown in chapter 5. The planner has to be tuned on the exploring strategy parameters depending on the context of application. Maximum steering rate limit of 20 degrees/sec was achieved.

Forward and back ward motion or only backward motion or only forward motion can be chosen and accordingly solution paths are planned.

Obstacle avoidance is successfully implemented and a solution path in C_{free} is generated by avoiding $C_{obstacle}$ as can be seen in the results obtained. In this thesis the planning tree is run until predefined maximum number of nodes is reached.

6.2. Recommendations

As presented earlier although kinematic feasibility was achieved but the required steering rate range 12 - 15 degrees/sec was not achieved for the solution paths obtained. To achieve this a better combination of the Digital Filter parameter and the controller parameter LD have to be found. Achieving this pair of values and including it in the forward simulation will ensure the control on steering rate within the expected range for the solution paths obtained.

Further improvement in the kinematic feasibility of the solution paths can be made by using a heuristic map of higher resolution. This will also directly improve the time required to obtain the solutions.

In this thesis the criterion to stop the planning loop was based on maximum number of nodes limit N_{max}, other possibilities like running the loop until desired number of solution paths or maximum time limit can be implemented.

In this thesis effort was made to obtain the proper solution paths to figure out the correct set of steps to achieve solution paths. The actual implementation time can drastically be improved by implementing the algorithm steps in a lower level language like c using Matlab/Simulink c coder.

In [15] a RRT framework where reference paths were built by connecting splines. With such reference paths a possibility to include Bi-DPFC exists as the controller is very good at tracking the reference path with smooth and minimum curvature constraint. Solution paths obtained with this combination will include the characteristics of the Bi-DPFC which is used as the middle ware between the reference paths and the driver assistance system components like AR-HUD.

List of Figures

List of Tables

Acronyms

AR-HUD Augumented Reality - Heads up display. 18, 21, 67

Bi-DPFC Bi-Directional Path Following Controller. vi, 21, 22, 25, 34–38, 41, 43–46, 48, 53, 64–70

C_{free} Free space. 12

C_{obs} Obstacle space. 12

C_{space} Configuration space. 12, 13

CL-RRT Closed-Loop Rapidly Exploring Random Tree. v–vii, 3, 4, 8, 9, 15, 16, 18–21, 25, 26, 29, 34, 36, 39, 43–45, 47–53, 56, 59, 61, 63, 65, 66, 68–70

DAV Double articulated vehicle. 16

DC Distribution Center. 10, 13, 14, 68

DM Dynamic Model. 10, 71

ey1 Lateral tracking error at look ahead distance LD,. 34

HAN HAN University of Applied Sciences. 21, 34

IK Inverse Kinematic Model. 24, 44–48, 51, 69

K_s proportional control gain. 34, 35, 69

KM Kinematic Model. 10, 24, 44–48, 51, 69, 71

LD Look ahead Distance. 34, 35, 41, 66, 69, 79

LQ Linear Quadratic. 16, 41

MAV Multi Articulated Vehicle. 9

MSE Mean squared Error. 34, 47, 49

NMP Non Minimum Phase. 46–48, 76

PPC Pure Pursuit Controller. 40, 44, 46–49, 54, 69, 79

RRT Rapidly Exploring Random Tree. vi, 3, 15, 16, 19, 20, 24, 25, 29, 38, 40, 48, 51, 53, 59, 67–69

SAV Single articulated vehicle. vii, 9–12, 16, 20, 24, 26, 30, 38, 44, 49, 53–55, 59, 62, 63, 66, 68, 70

SRRT Spline-based Rapidly exploring Random Tree. 8

VAS Virtual Articulation angle Subsystem. vii, 24, 43–45, 47, 66, 76

VISTA VIsion Supported Truck docking Assistant. 20, 34

WMR Wheeled Mobile Robot. 10

Bibliography

[1] K. Kural. "Analysis of high capacity vehicles for Europe: application of performance based standards and improving manoeuvrability". In: (2019).

[2] *Vision Supported Truck docking Assistant concept*. https://vistaproject.eu/. Accessed: 2020-07-20.

[3] K. Kural, I. Besselink, Y. Xu, A. Tomar, and H. Nijmeijer. "A driver support system for improved maneuvering of articulated vehicles using an unmanned aerial vehicle". In: *HVTT14: International Symposium on Heavy Vehicle Transport Technology, Rotorua, New Zealand*. 2016.

[4] *Distraction or disruption? Autonomous trucks gain ground in US logistics*. https://www.mckinsey.com/industries/travel-logistics-and-transport-infrastructure/our-insights/distraction-or-disruption-autonomous-trucks-gain-ground-in-us-logistics#. Accessed: 2020-07-20.

[5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli. "A survey of motion planning and control techniques for self-driving urban vehicles". In: *IEEE Transactions on intelligent vehicles* 1.1 (2016), pp. 33–55.

[6] N. Evestedt, O. Ljungqvist, and D. Axehill. "Motion planning for a reversing general 2-trailer configuration using Closed-Loop RRT". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 3690–3697.

[7] F. D. Boyden and S. A. Velinsky. "Limitations of kinematic models for wheeled mobile robots". In: *Advances in robot kinematics and computational geometry*. Springer, 1994, pp. 151–160.

[8] R. KUSUMAKAR, K. KURAL, A. S. TOMAR, and B. PYMAN. "Autonomous Parking for Articulated Vehicles". In: *HAN University of Applied Science, Netherlands* (2017).

[9] S. M. LaValle. *Planning algorithms*. Cambridge university press, 2006.

[10] N. Evestedt. "Sampling Based Motion Planning for Heavy Duty Autonomous Vehicles". PhD thesis. Linköping University Electronic Press, 2016.

[11] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How. "Real-time motion planning with applications to autonomous urban driving". In: *IEEE Transactions on control systems technology* 17.5 (2009), pp. 1105–1118.

[12] A. TOMAR, R. KUSUMAKAR, and A. NAREN. "PATH FOLLOWING BI-DIRECTIONAL CONTROLLER FOR ARTICULATED VEHICLES". In: ().

[13] D. B. P. Dinakar et al. "lmplementation of Autodocking Controller onto an Articulated Vehicle". In: (2020).

[14] N. Evestedt, O. Ljungqvist, and D. Axehill. "Path tracking and stabilization for a reversing general 2-trailer configuration using a cascaded control approach". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2016, pp. 1156–1161.

[15] K. Yang, S. Moon, S. Yoo, J. Kang, N. L. Doh, H. B. Kim, and S. Joo. "Spline-based RRT path planner for non-holonomic robots". In: *Journal of Intelligent & Robotic Systems* 73.1-4 (2014), pp. 763–782.

[16] *The First Holographic Navigation system for cars*. `https://wayray.com/navion`. Accessed: 2020-08-1.

[17] *Haptic Feedback system*. `http://bdml.stanford.edu/Main/AutomotiveHaptics`. Accessed: 2020-08-1.

[18] N. Evestedt, D. Axehill, M. Trincavelli, and F. Gustafsson. "Sampling recovery for closed loop rapidly expanding random tree using brake profile regeneration". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 101–106.

[19] D. Devasia. *Motion planning with obstacle avoidance for autonomous docking of single articulated vehicle*. 2020.

A. General Addenda

A.1. Virtual Articulation Angle VAS

A.1.1. Signum Functions for stability

Following are excerpts from [13] and are helpful to understand the purpose of VAS. using inverse kinematic relations we get Equation A.1

$$\dot{\theta}_0 = \left[\frac{v_1 . \sin\gamma - L_{1f} . \cos(\gamma) . (\dot{\theta}_1)}{-L_{0b}} \right] \tag{A.1}$$

For positive value of L_{0b} and positive value of v_1 in equation A.1 the system of tractor/semi-trailer combination is found stable. But with the same combination, when v_1 is negative the system is found unstable.

When L_{0b} is negative in equation A.1, the system with tractor/draw bar trailer combination is found unstable. To make sure that the system is always stable during both the non-minimum phase scenarios (forward-draw bar, Reverse-semi trailer) in inverse kinematics the equation A.1 is modified to get equation A.2 where the $sign(v_1)$ and $sign(L_{0b})$ are used to make sure a stable form of the equation is obtained in forward and reverse directions.

$$\dot{\theta}_0 = sign(v_1) . sign(L_{0b}) . \left[\frac{v_1 . \sin\gamma - L_{1f} . \cos(\gamma) . (\dot{\theta}_1)}{-L_{0b}} \right] \tag{A.2}$$

The downside to using these functions are that the yaw rate $\dot{\theta}_0$ required for a tractor is always for a tractor/semi-trailer combination in forward direction and always for a tractor/draw bar trailer combination in reverse direction. This leads to inaccurate kinematic inversion and hence corrections are proposed to obtain proper kinematic inversion for both NMP scenarios after including signum functions. Following section mentions only the details related to the reverse-semitrailer NMP scenario applicable to this thesis.

A.1.2. Virtual articulation angle derivation

In reverse-semitrailer NMP scenario the real articulation angle γ_r for a tractor/semi-trailer combination is converted to a virtual articulation angle γ_v for a tractor/draw bar trailer combination with modified dimensions as seen in Figure A.1.

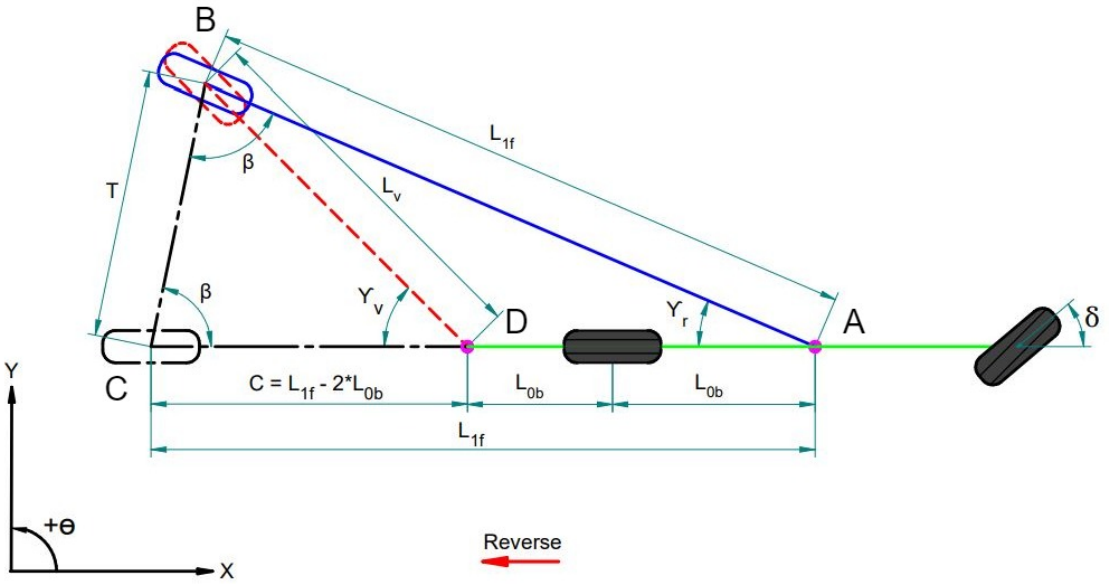


Figure A.1.: Geometry showing semitrailer real articulation angle γ_r and stable drawbar articulation inspired, virtual articulation angle for semitrailer γ_v [13]

The Figure A.1 shows the geometry and the following equations show the conversion principle used to obtain the virtual articulation angle.

The triangle ΔCBD formed by the virtual trailer of drawbar type is an isosceles triangle and the dimensions of the triangle change with γ_r, the equations A.3 and A.4 can be drawn from this arrangement.

$$2.\beta = (\pi - \gamma_r) \tag{A.3}$$

$$T = 2.\cos(\beta).L_{1f} \tag{A.4}$$

The dimensions of the triangle ΔABC formed by the real trailer remain constant. Using cosine rule the virtual angle calculation are obtained.

$$C = L_{1f} - 2.L_{0b} \tag{A.5}$$

$$L_v^2 = T^2 + C^2 - (2.T.\cos\beta) \tag{A.6}$$

$$\gamma_v = \frac{L_v^2 + C^2 - T^2}{2.L_v.C} \tag{A.7}$$

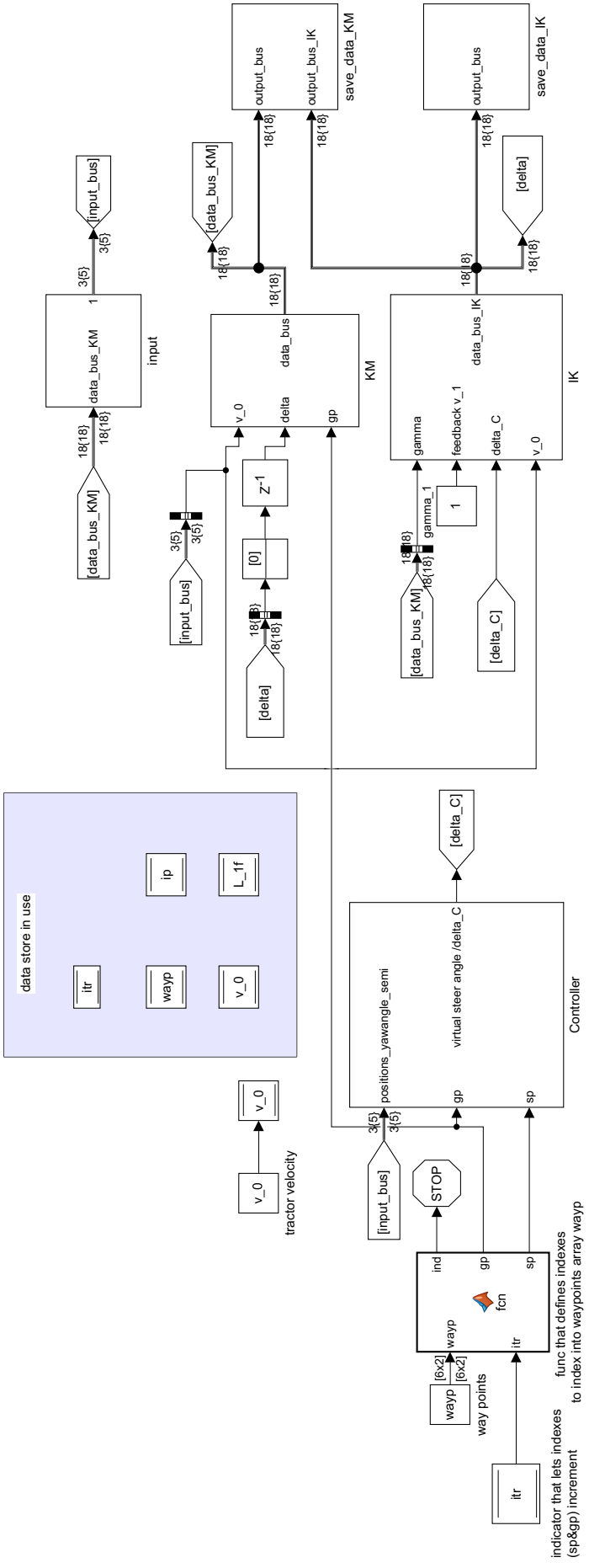


Figure A.2.: Closed loop stable system that includes the vehicle Inverse Kinematics(IK), Forwards kinematics(KM), controller. System used for Forward simulation

A.2. Simulink blocks and MATLAB script snippets

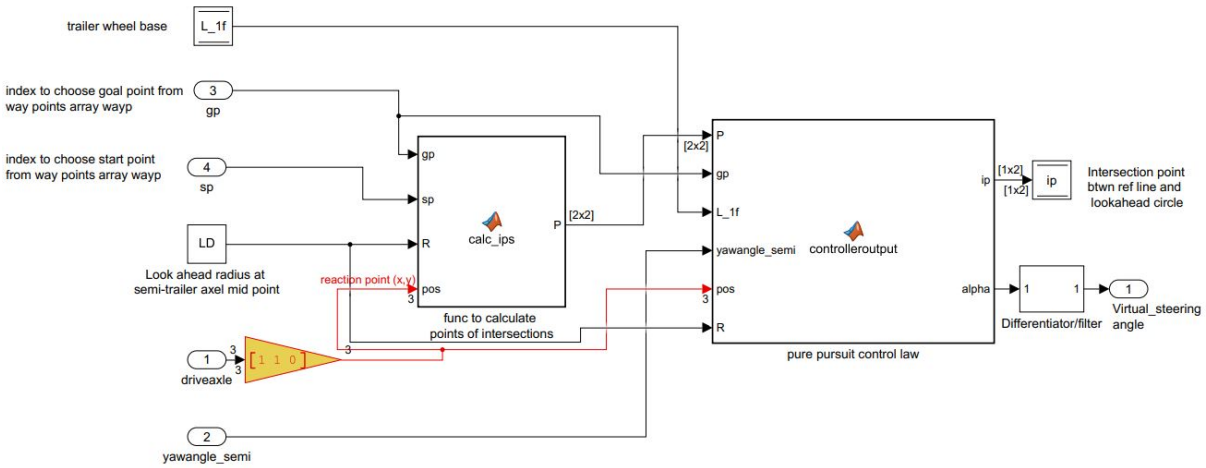


Figure A.3.: Flowchart for heuristic map generation

Matlab function to calculate point of intersection between the linear reference path and the look ahead circle with radius LD

```
function P = calc_ips(gp, sp, R,pos)
%this func calculates points of intersection between ref line (formed
%with goal point, start point) and look ahead circle with center at semitrailer axle midpoint
global wayp
P=zeros(2,2);
%indexing into wayp with indexes gp and sp
p1=[wayp(gp,1) wayp(gp,2)];
p2=[wayp(sp,1) wayp(sp,2)];
slope = (p1(1,2)-p2(1,2))/(p1(1,1)-p2(1,1));
if slope == -inf || slope ==inf
intercpt=p1(1,1);
else
intercpt = p2(1,2)-slope*p2(1,1);
end
%updating center of circle with the current pose of semitrailer axle
%position
centerx=pos(1,1);
centery=pos(2,1);
radius=R;
coder.extrinsic('linecirc')
[xout,yout] = linecirc(slope,intercpt,centerx,centery,radius);
P=[xout ; yout];
```

Matlab function that implements PPC control law

```
function [ip,alpha] = controlleroutput(P,gp,L_1f,yawangle_semi,pos,R)
%this func calculates steer angle that pursues the ref line formed between
%start point and goal point chosen from wayp
global wayp
```

```
Psz=size(P,2); %num of intersection points, with x_cords in 1st column y_cords in 2nd column
gp_cords=wayp(gp,:);
distarray=zeros(Psz,3); % intersection points distance to gp (goal point)
for k=1:Psz
distarray(k,1:2)=[P(1,k) P(2,k)];
distarray(k,3)=sqrt((P(1,k)-gp_cords(1,1))^2+(P(2,k)-gp_cords(1,2))^2);
end
%choosing the nearest intersection point to the goal point from the
%available 2 points of intersection between reference line and look ahead
%circle
sortP=sortrows(distarray,3);
ip=sortP(1,1:2);
alpha=zeros(1,1);
%Theta_e
num=ip(1,1)-pos(1,1);
den=ip(1,2)-pos(2,1);
theta_e = atan2(den,num)-yawangle_semi; %orientation error
alpha = -atan((2*L_1f*sin(theta_e))/R); %
```

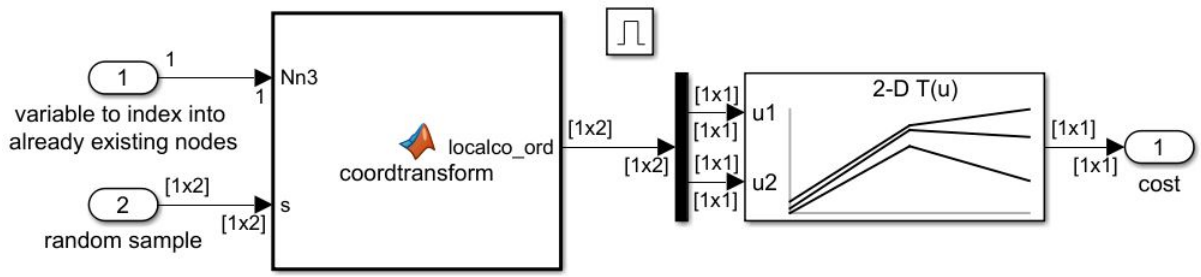


Figure A.4.: retrieving data from heuristic map using lookup tables