Optimization and Mathematical Modelling for Path Planning of Co-operative Intra-logistics Automated Vehicles

Edward Derek Lambert

University of Leeds

Institute for Transport Studies

Submitted in accordance with the requirements for the degree of

Doctor of Philosophy

March, 2023

Intellectual Property Statement

Chapter 3 is based on a jointly authored publication [1] ED Lambert, R Romano, and D Watling. Optimal Smooth Paths based on Clothoids for Car-like Vehicles in the Presence of Obstacles. *International Journal of Control, Automation and Systems*, 2020. ISSN 1598-6446.

The candidate derived the equations, wrote the code, drew the figures and wrote the text. Richard Romano and David Watling suggested techniques and avenues of investigation, read the text, checked the equations and indicated corrections.

Acknowledgements

Thanks to Dr. Dave Barnett for introducing me to Automated Guided Vehicles, the clothoid curve, and teaching me how to write software. Thanks to Dr Martin Davies and Dr Paul Rivers and everyone at Guidance Automation Limited. I would like to express my great appreciation to Prof Rich Romano and Prof David Watling for keeping me on track, guarding my time and teaching me how to be a better academic. Special thanks to Derek and Olive Lambert. To Mum and Dad and Hazel and Rebecca for bailing me out every time I absent mindedly forgot to take care of something in the real-world.

This research project was supported by the Engineering and Physical Sciences Research Council (EPSRC) through a full-time Doctoral Training Partnership Studentship - Institute for Transport Studies in a Combined Award in Science and Engineering partnership with Guidance Automation Limited.

Abstract

Small indoor Autonomous Vehicles have revolutionized the operation of pick-pack-and-ship warehouses. The challenges for path planning and co-operation in this domain stem from uncontrolled environments including workspaces shared with humans and human-operated vehicles. Solutions are needed which scale up to the largest existing sites with thousands of vehicles and beyond. These challenges might be familiar to anyone modelling road traffic control with the introduction of Autonomous Vehicles, but key differences in the level of decision autonomy lead to different approaches to conflict-resolution. This thesis proposes a decomposition of site-wide conflict-free motion planning into individual shortest paths though a roadmap representing the free space across the site, zone-based speed optimization to resolve conflicts in the vicinity of one intersection and individual path optimization for local obstacles.

In numerical tests the individual path optimization based on clothoid basis functions created paths traversable by different vehicle configurations (steering rate limit, lateral acceleration limit and wheelbase) only by choosing an appropriate maximum longitudinal speed. Using two clothoid segments per convex region was sufficient to reach any goal, and the problem could be solved reliably and quickly with sequential quadratic programming due to the approximate graph method used to determine a good sequence of obstacle-free regions to the local goal.

A design for zone-based intersection management, obtained by minimizing a linear objective subject to quadratic constraints was refined by the addition of a messaging interface compatible with the path adaptations based on clothoids. A new approximation of the differential constraints was evaluated in a multi-agent simulation of an elementary intersection layout. The proposed FIFO ordering heuristic converted the problem into a linear program. Interior point methods either found a solution quickly or showed that the problem was infeasible, unlike a quadratic constraint formulation with ordering flexibility. Subsequent tests on more complex multi-lane intersection geometries showed the quadratic constraint formulation converged to significantly better solutions than FIFO at the cost of longer and unpredictable search time. Both effects were magnified as the number of vehicles increased.

To properly address site-wide conflict-free motion planning, it is essential that the local solutions are compatible with each other at the zone boundaries. The intersection management design was refined with new boundary constraints to ensure compatibility and smooth transitions without the need for a backup system. In numerical tests it was found that the additional boundary constraints were sufficient to ensure smooth transitions on an idealized map including two intersections.

Contents

List of Figures

List of Tables

Abbreviations

AGV	Automated Guided Vehicle
AVS/R	Automated Vehicle Storage and Retrieval
RMF	Robotic Mobile Fulfilment
AMR	Autonomous Mobile Robot
CARV	Connected and Autonomous Road Vehicle
ANN	Artificial Neural Network
GPS	Global Positioning System
LiDAR	Light Detection and Ranging Sensor
PRM	Probabilistic Roadmap
RRT	Rapidly-exploring Random Tree
AIM	Autonomous Intersection Management
CPLEX	Optimization software package from IBM
DCFVRP	Dispatch and Conflict-Free Routing Problem
PRT	Probabilistic Reservation table
ROS	Robot Operating System
DPMC	Distribute Model Predictive Control
MILP	Mixed-Integer Linear Program
I2I	Intersection-to-Intersection
V2I	Vehicle-to-Intersection
MFD	Macroscopic Fundamental Diagram
RPA	Rakha-Pasumarthy-Adjerid
DCC	Double Continuous Curvature
CC-Steer	Continuous Curvature Steer
G^0	Path continuity in position
G^1	Path continuity in position and heading
G^2	Path continuity in position, heading and curvature

Glossary

Convex Shape	A straight line between any two points within the shape remains entirely within the shape.
Convex Problem	The objective and the constraints both have a convex shape.
Elementary Intersection	Formed where two perpendicular unidirectional straight lanes meet.
Adaptive Planning	Path or trajectory generation from on-board range sensors.
Waypoint	A point in 2D space.
Path	A sequence of waypoints.
Trajectory	A path where each waypoint is labelled with a particular time.
Intra-logistics	The movement of goods within distribution warehouses. Here, homogeneous stock lines are combined and transported into heterogeneous customer boxes for delivery. Each customer order may be stored in a Warehouse Management Database, and new online orders may arrive at any time during operation.
State Space	The space with one dimension for each degree of freedom of the robot control point e.g [x,y,heading] for a ground vehicle.
Configuration Space	The same dimension as State Space with obstacles boundaries expanded by the dimension of the robot.
Holonomic Vehicle	Able to actuate in any dimension of state space independently.
Non-holonomic Vehicle	Actuators are arranged such that state space dimensions are coupled by differential constraints e.g. only able to move forwards or reverse at a given heading.
Complete algorithm	Guaranteed to find a solution if one exists or report failure if there is no solution.
Resolution complete	A sampling based algorithm that guarantees a solution will be found if one exists at the sampling resolution.
Asymptotically complete	A random-sampling based algorithm, that guarantees a solution will be found if one exists as the number of samples tends to infinity.

Optimal algorithm	Guarantees that the solution will minimize some objective function at least in a region of attraction.
Globally optimal algorithm	Guarantees that the solution will minimize some objective function over the entire parameter space.
Differential Constraints	Constraints on the time derivative of the configuration variables e.g. those arising from the kinematics or dynamics of a robot.
Potential Field Method	A planning approach where actions are scored with a smooth artificial potential function which is minimized at the goal.
Vector Force Field	A particular artificial potential function which attributes an attractive force to the goal and repulsive forces to the obstacles.
Decoupled Method	Decomposes a large problem into sub-problems with minimal coupling which can be solved in isolation.
Decentralized Method	Decomposes a large problem into sub-problems that are solved by different agents.
Centralized Method	A single agent solves the entire problem.
Roadmap	Nodes represent point locations. An edge is inserted where one location is reachable from another.
Probabilistic Roadmap	Nodes are random samples from a distribution.
Topology Graph	Nodes represent bounded regions. An edge is inserted where one region is reachable from another.
Search Tree	Data structure for graph search where nodes are labelled open until they have been checked. Open nodes connected by an edge to a closed node are referred to as child nodes.
Branching Factor	The number of children at each node in a search tree.
Topological Blindness	Failure to search discontinuous deformations such as passing the other side of obstacles.

Chapter 1

Introduction

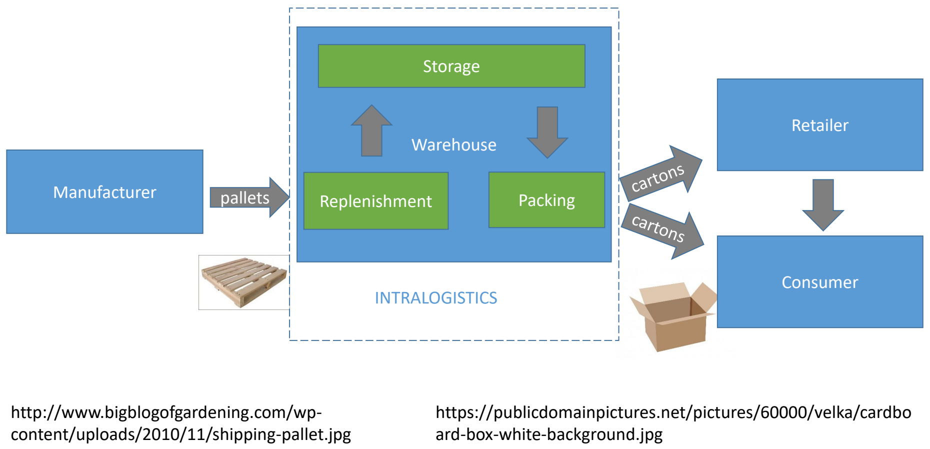

Storage
Warehouse
Manufacturer
pallets
Replenishment
Packing
cartons
cartons
Retailer
Consumer
INTRALOGISTICS
http://www.bigblogofgardening.com/wp-content/uploads/2010/11/shipping-pallet.jpg
https://publicdomainpictures.net/pictures/60000/velka/cardboard-box-white-background.jpg

Figure 1.1: Material flow in intra-logistics.

Collaborative mobile robots are being adopted widely in distribution centres and factories to increase efficiency and reduce operating costs [7]. Collaborative robots are so named because they are designed to operate in collaboration with human workers. This makes partial adoption possible and reduces the start-up costs with minimal modification to infrastructure. They are most beneficial to e-commerce pick-pack-and-ship warehouses where current pick density is low [8]. In this situation, adding a number of automated trolley robots to ferry items back to the packing station can reduce the time pickers spend walking backwards and forwards, freeing them up to pick more items.

The physical form of robots intended for horizontal movement of items from stock shelves to customer order boxes varies depending on the storage layout and the level of collaboration with humans. The main systems involve an array of inventory pods (containing items), some robotic drive units and a number of workstations [7]. With a minimal level of human input are Automated Vehicle Storage and Retrieval (AVS/R) systems. Here the pods are stacked vertically. In some cases the drive units move on rails above the stacks like the system from Ocado Robotics [9], which operates with no humans in the building. Other systems arrange the pods on pallets which can be retrieved by human or robotic fork-lift type vehicles. Robotic Mobile Fulfilment (RMF) such as the Kiva system (now Amazon Robotics [10]) involves more collaboration with humans and has been widely adopted. Here the pods take the form of movable shelves

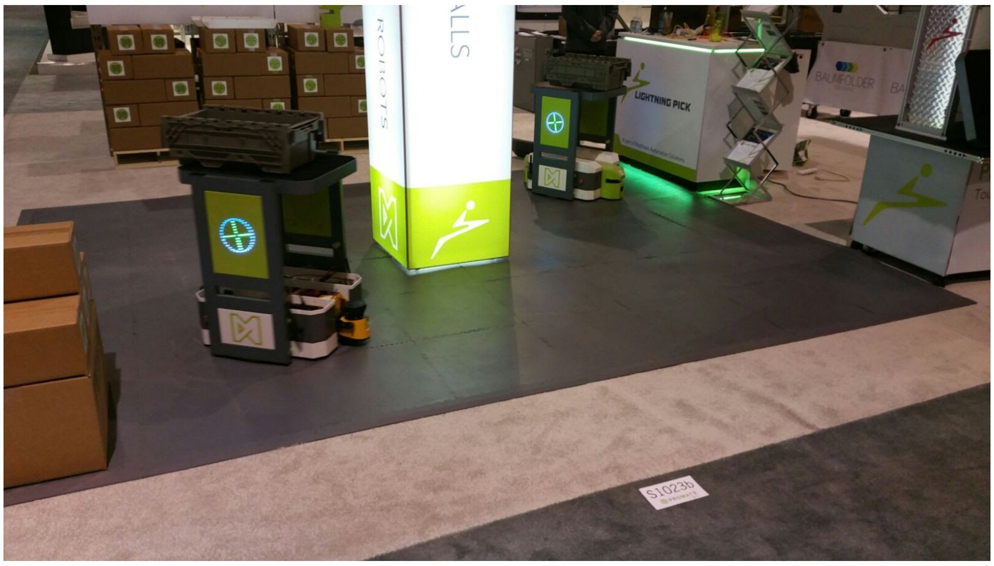


Figure 1.2: Collaborative mobile robots at the LightningPick (TM) stand at ProMAT material handling trade show Chicago 2017. Photograph taken by Edward Lambert.

with space for the drive units to travel beneath them [11]. Drive units which can operate alongside humans may need less sophisticated manipulators. Pick-support AGVs consist of a wheeled platform which either tow standard roll cages or have an integral item tray which is unloaded manually by human pickers. The workspace is locally flat and contains some permanent and some temporary obstacles. This has led to vehicles with differential drive configuration which can fit in the same width aisles as human pickers like the Guidance Automation 'AMRcore' [12][13]. These vehicles somewhat resemble a shopping trolley, having solid tyres and a basket on top. Two robots of this type circling a lit column at a trade show are shown in Figure 1.2. Single traction wheel designs which fit into the same footprint as a 'pump truck' or small fork lift truck for moving pallets have been reported in the literature [14]. From the perspective of control design for the powered base, either configuration is like a small electric car with simplified steering and suspension. As they primarily move forwards or backwards they approximate the well known single-track model for Ackermann steering [15].

Motion control of the drive units in an RMF system is sometimes referred to as Fleet Control [16]. The main components of this problem are shown in Figure 1.3. Execution of motion control activities is split between a fixed server and the embedded controllers

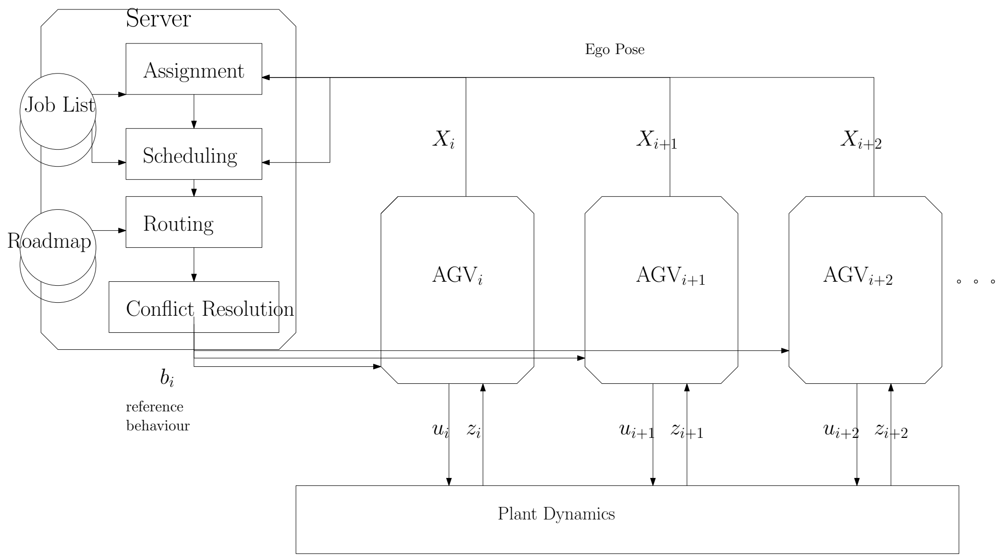


Figure 1.3: Overarching activity diagram.

on the drive units or AGVs. The server holds a list of the jobs to be completed and a roadmap representation of the connections between job locations. It receives frequent position estimates from each AGV by a wireless radio link, allowing it to send appropriate reference behaviours, such as assigning a nearby job. At a minimum, each job in the job list would contain movement instructions for the mobile base. Motion can be extracted from a more complex item fetch job using a database of stock locations. The fleet operator is likely to desire the most items transported in a set time, given a fixed amount of floor space. This leads to a key objective of maximum throughput for the whole site or equivalently minimum makespan [17]. The objective and many of the subtasks in Figure 1.3 are common to the Vehicle Routing Problem with Time Windows faced by operators of road vehicle fleets [18]. They too have a fixed number of drive units to complete a list of movement tasks in a known environment which can be encoded as a roadmap graph. They must assign every available drive unit to a job, schedule the remaining jobs for completion in a sensible way and find a route through the roadmap for each drive unit.

A key difference between fleet control for road traffic and the intra-logistics domain lies in the approach to conflict-resolution. The AGV literature has historically been more concerned with the prevention of collisions and deadlocks [19]. Right from the start, sensors were added to each vehicle to prevent collisions. Limited on-board computation meant that routing instructions were interpreted literally by feedback control. Even if the sensors were good enough to prevent a head-on collision, a deadlock would result as neither vehicle could proceed on its original path. To avoid this situation without affecting path layouts, additional constraints to ensure conflict-free motion were included in the centralized solution. Despite the improved capabilities of modern AMR, the 'Conflict-Free' variant of the Vehicle Routing Problem is still solved [20]. The extra constraints are still included and scale poorly with an increasing number of vehicles. For this reason low resolution approximations to the constraints are often accepted and still provide an upper limit on system size. In the road traffic domain conflict resolution is rarely considered for two reasons. One, the road network and highway code have been carefully designed and tweaked over many years to make sure conflicting traffic can be seen and priority is rule-based as in roundabouts or controlled by traffic lights. Two, because the level of autonomy of the 'drive units' is extremely high. Until fairly recently, the entire routing task was handled by individual drivers following signs or paper maps. Detailed fleet control for Connected and Autonomous Road Vehicles (CARVs) is not held back by conflict constraints. Other vehicles can be efficiently represented as 'congestion' or a reduction in link speed and solutions found for networks at city scale with thousands of taxis [21].

Armed with this analogy between fleet control solutions for taxi cabs on the road and fleet control in RMF systems, this research thesis will propose methods to adapt the best techniques for cross application and model their performance compared to the state of the art in the AGV domain.

The objectives of this thesis are:

- O1: Avoid static obstacles using parametrized alternative paths.
- O2: Resolve motion conflicts between communicating AGVs with independent schedules and routes which occur at isolated intersections.
- O3: Resolve motion conflicts between communicating AGVs across an entire site with multiple nearby intersections.

Chapter 2 reviews literature in relevant domains and specifies research gaps to be filled by the subsequent chapters. Chapter 3 presents a new approach to reactive planning for vehicles based on clothoid basis functions to address Objective O1. Chapter 4.1 proposes a method for autonomous intersection management with a messaging interface suitable for an isolated elementary intersection towards Objective O2. Chapter 5 goes on to test this approach on an intersection with multiple lanes to evaluate the approach to Objective O2. Chapter 6 examines the motion of vehicles on a layout with two nearby intersection managers and proposes a refinement of the messaging interface with new constraints to address Objective O3.

Chapter 2

Combined Literature Review

The literature covered in this chapter addresses different aspects of automated vehicle motion planning, and the models and control approaches which have been studied in different domains. Prior works have addressed off-highway mobile robots for material handling in smart factories and intra-logistics. Mobile robots in use today can be categorized into two types. They are known as Autonomous Mobile Robots (AMR) if they maintain an individual representation of the environment in which they have the authority to plan their own motion. Automated Guided Vehicles (AGV), on the other hand use feedback control to execute plans generated by a supervisory system. AGV are used for well-separated tasks in tightly controlled environments [20], but AMR are opening the door to collaborative multi-robot systems which can be reconfigured rapidly as explained in Chapter 1. A number of papers address air traffic control for collections of Unmanned Aerial Vehicles such as quadrotors [22]. Many works also address Connected Autonomous Road Vehicles (CARVs) intended for public highways and traffic signal controls directing the actions of human drivers through road intersections.

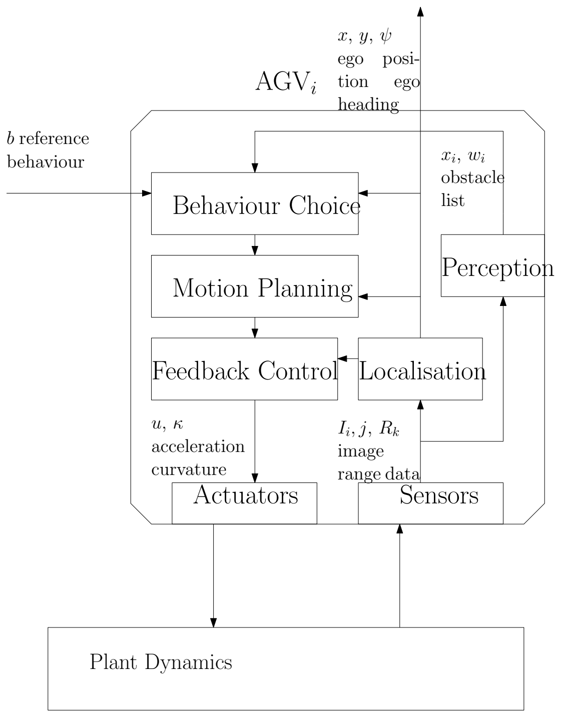


Figure 2.1: Logical flow of hierarchical planning

Planning architectures of automated vehicles across these application areas may be divided into three broad categories: Hierarchical, interactive behaviour aware, and end-to-end machine learning [23]. The hierarchical approach divides the overall problem into the logical blocks shown in Figure 2.1.

End-to-end machine learning would replace the discrete functional blocks in Figure 2.1 including perception, behaviour choice, motion planning and feedback control with multiple layers of Artificial Neural Network (ANN), to create a single block connecting Sensors to Actuators. This approach offers the tantalizing prospect of novel planning strategies inferred from past data, which to not need to be explicitly designed, and can adapt and improve as they acquire more data through experience [24]. In [25] the problem of acquiring rich enough training data to control rare/dangerous situations such as veering off the road is mitigated by perspective distortion of images from real driving.

Interactive behaviour aware planning is less extreme and closer to practical application by using machine learning only to replace certain blocks such as elements of perception or behaviour choice. Driving behaviours such as overtaking and tailgating have been generated using the Q-learning algorithm in a simulation environment [26]. The effectiveness of partial ANN integration has been shown in a service robot completing tasks in an office building [27]. The combined approach is more robust to model failure than pure hierarchical control but is less prone to finding extremely costly solutions in time. Certain high speed domains such as road traffic have a high risk of harm and require strong software validation in order to meet regulations [28]. It is difficult to validate an ANN planning approach to the same standard, especially if it is constantly adapting and learning from new data, although continuous integration and virtual testing can help with this.

Hierarchical planning continues to be useful to describe and understand planning approaches. It also underpins research across all domains, not least material handling AGV [29]. Even with strictly hierarchical approaches there is a fundamental connection between the representation of the environment and the complexity of plans which can be made within it. A two-dimensional polygonal obstacles can represent a range of environments [30]. The actual sensors involved may include cameras, LiDAR, radar or GPS [31].

A hierarchical structure is used to organise the rest of this review. Particular attention is paid to the motion planning block as this contains the main parallels

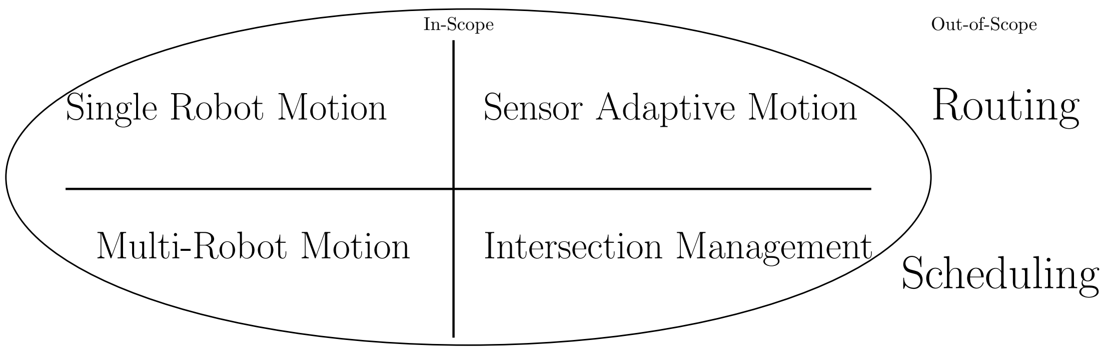


Figure 2.2: Scope of Literature Review.

identified in Chapter 1. Literature concerning motion planning can be divided into four sections, corresponding to one of the in-scope items shown in Figure 2.2.

The items specifically listed as out-of-scope are closely related to the topic of path planning optimization but are outside the direct contributions of this thesis. The presented algorithms could be combined with existing methods for task scheduling and global routing to create an entire fleet control system.

The (Global) Routing problem is excluded in terms of scale as the search for paths around static obstacles over the entire workspace. A number of the methods reviewed in Section 2.1 could be used e.g. probabilistic roadmap. In the road traffic domain this task is carried out by each agent independently. Other agents are only considered in aggregate as an increase in link traversal time if traffic guidance is used e.g. by a satellite navigation device.

The (Task) Scheduling problem concerns the assignment of tasks to agents, and the order in which they are undertaken. This problem has attracted a great deal or research interest in cases where a single decision maker has authority over a number of vehicles. For AGVs the authority is clear and scheduling is often solved in conjunction with routing in the methods reviewed in Section 2.3.

Section 2.1 Single Robot Path Planning looks at advanced techniques for motion planning of individual robots, starting wide and going into more detail about paths and obstacle avoidance in Section 2.2.

Next, Section 2.3 Multi-Robot Motion Planning discusses existing modelling approaches and research into control systems for fleets of identical mobile robots. Some of the ways the problem has been broken down are examined including the two-level logical decomposition used to divide the next two sections.

Rounding out the domains of interest is Section 2.4 Intersection Traffic Control, which looks at traffic control for multiple agents in a certain location, such as an intersection. General principles common to human operated vehicles are introduced and more detail is given on Automated Intersection Management approaches, which are typically intended for CARVs.

2.1 Single Robot Motion Planning

Geometric path planning or the piano mover's problem consists of finding a sequence of configurations between an origin and a destination within a field of obstacles for a single robot. The problem as defined in Figure 2.3 is known to be PSPACE hard, that is to require at least a polynomial amount of memory to solve [32]. It has been a topic of active research in robotics and animation for the last several decades and many practical solutions have been developed, especially for the case of a two dimensional workspace with polynomial obstacles [31].

Given:

1. A workspace $\mathbf{W}$ is either in R^2 or R^3
2. An obstacle region $\mathbf{O}$
3. A robot defined in $\mathbf{W}$ consisting of one or more rigid bodies
4. A configuration space $\mathbf{C}$ comprising $\mathbf{C_{obs}}$ and $\mathbf{C_{free}}$
5. An initial configuration $q_I \in \mathbf{C_{free}}$
6. A goal configuration $q_G \in \mathbf{C_{free}}$

Compute a continuous path $\tau : [0,1] \to C_{free}$ such that $\tau(0) = \mathbf{q_I}$ and $\tau(1) = \mathbf{q_G}$

Figure 2.3: The piano mover's problem - geometric path planning.

Typically some approximation is made to keep the problem in Figure 2.3 tractable, at the cost of some aspect of global optimality of the solution. Approaches may be divided roughly into Continuous and Sampling Based methods [32]. Solution approaches are grouped based on their treatment of state space and suitability for world propensity to change in Figure 2.4.

Different approaches have been used historically depending on the extent of the planning domain. Trajectory plans with a large spatial extent typically also cover long time periods so they can be called strategic plans by an analogy with business plans which cover longer time periods. At the other end of the spectrum small scale trajectory plans take place over shorter time periods so they might be called tactical plans.

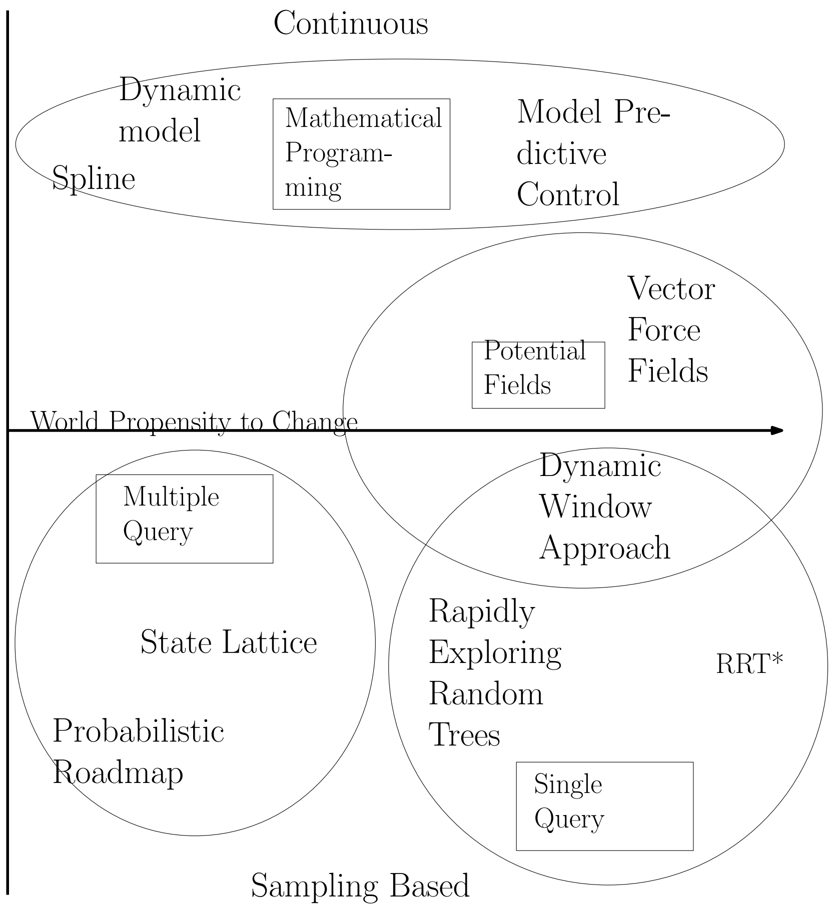


Figure 2.4: Motion Planning classification based on scale, discretization and suitability for changing environments.

Another fundamental distinction between approaches comes from the way they represent the finite state space in which they search for plans. This leads to the central dividing line in Figure 2.4, dividing continuous representations from those which sample at some resolution to create a discrete problem.

Within sampling based discrete methods the important temporal distinction between approaches is based on the reuse of a representation of the environment within a particular algorithm. This comes down to the world's propensity to change. If the environment is relatively static then a more efficient search can be performed by excluding configurations which have already been tested at the last iteration. This second branch is specific to discrete sampling based methods, dividing Single Query planners from those which make Multiple Queries on the same representation.

Dealing with the world's propensity to change for continuous methods also leads to a separate family bubble in Figure 2.4. Potential Field approaches such as the classic Vector Force Field [33] are purely reactive, choosing only the first action to explore the objective function as they travel. This leads to light computation and often robustness to environmental change, even without awareness it has changed. Due to this myopia there is a real danger of livelock traps where the same set of actions are executed repeatedly and the goal is never reached. Mathematical Programming methods such as [34] can be more long term as they incorporate an explicit model of the system evolution. This makes them less prone to getting stuck in local minima if the objective function is not convex.

2.1.1 Continuous Methods

Mathematical Programming describes a class of numerical procedures which search a parameter space to find the value $\hat{\boldsymbol{\phi}}$ at which the objective function $J(\boldsymbol{\phi})$ is minimized. Many techniques using this approach are also reviewed in [31]. These techniques have a wide range of applications, for path planning they are especially useful to overcome problems with local minima, or at least identify them before any wasteful actions are taken. Accordingly there are a wide variety of different approaches to the parametric representation of the trajectory between methods. At one end are parametric curves or splines with a certain degree of continuity such as the O(3) Bezier curves used by [35]. Conversely some methods embed a dynamic model of the vehicle in the objective function such as the linearised single track model in [36]. In [37] the parametric representations are classified as integral or differential form. Timed Elastic Bands Method is an example of differential form, the sequence of predicted positions alone form the parameter vector, and all differential constraints are applied to numerical derivatives of this vector making it extremely fast to solve with nonlinear least squares, with the constraints joined in to the cost. There is inevitably some trade off between the objective and constraint violation but it can be made tolerably small [38]. Approaches can also be distinguished by their representation of the obstacle constraints. In [30], a polygonal representation is transformed into a collection of possibly overlapping convex empty regions, by introducing integer variables to assign polynomial path pieces to regions. Methods which make use of a reference path can represent obstacles convexly in path relative coordinates as in [39] [40].

Potential field methods exist at the extremely short term end of the motion planning spectrum. Rather than produce a sequence of future states to traverse a region of the obstacle field, only the latest sensor readings are taken into account to decide the next move. They can also be referred to as Attractive/Repulsive Force methods or Vector Force Fields. They are popular for reactive planning as they are simple and effective. Issues with wasted motion searching the space using only immediate information may arise, oscillatory behaviour close to obstacles can be an issue if not filtered out and they are also vulnerable to local minima traps [41]. Differential constraints can be incorporated by searching feasible action space and choosing the action which leads to a state with the lowest potential. Numerous methods have been proposed to address issues with local minima traps such as randomized potential fields but it can still be extremely difficult to design a good potential function [42]. They have been surpassed by optimal methods in applications where sufficient computational power is available. This classical approach to reactive path planning is still attractive for its intuitive problem representation. It has been extended for use in platooning [43] where it improves recovery when the platoon needs to divert to avoid a static obstacle. Obstacle expansion by a convex ghost obstacle and intermediate goal points are useful to keep the platoon together while managing the problems with oscillation and local minima.

2.1.2 Sampling Based Methods

Sampling methods for motion planning can be divided into deterministic and randomized sampling. Deterministic sampling methods subdivide state space into a uniform lattice such as a grid or a more exotic structure designed to span the search space effectively. Randomized sampling methods fall into two main categories: Single query like Rapidly Exploring Random Trees (RRTs) and multiple query like Probabilistic

Roadmaps (PRMs).

Multiple query planners seek efficient computation in static environments. This is because it is time consuming to produce a new roadmap but quick to make additional queries on it. Single query planners are more useful when the environment is changing rapidly such as a robot exploring new areas. A search tree can constructed quickly based on the latest samples from obstacle-free space but there is little advantage to making subsequent queries. In theory, as the number of samples tends to infinity, truly random samples will provide unbiased and complete coverage of state space. In this way a PRM or RRT may be 'probabilistically complete' so that as the number of samples tends to infinity the probability of finding a solution if one exists tends to one.

PRMs are an example of a multiple-query sampling based planner. They are probabilistic in the sense they sample randomly from $\mathbf{W}$ to build up a connectivity graph - the roadmap. They do not inherently cope with uncertainty in the obstacle field [44]. One important component is the local planner, which is used to generate a path between two nearby configurations. The algorithm proceeds by generating a set of samples at random locations in the obstacle-free space. Each sample is tested by generating a local path (edge candidate) from the nearest configuration in the existing tree, and only adding the sample and the edge to the graph if the local path is obstacle free. Depending on the local planner it is possible to incorporate smooth paths respecting vehicle dynamic limits, but other methods are more appropriate in cases where the obstacle field is dynamic because the simplifying assumption making roadmaps so fast is that of a static environment.

Lattice planners are only distinguished from PRMs by the way samples are drawn from state space to construct the roadmap [32]. Instead of randomly selecting states to test, a lattice is overlaid. States in the lattice are pruned if they are not reachable according to the obstacle map and the local planner in the same way as before.

Sampling from the state space in a regular lattice to create a graph for efficient searching is one of the first practical approaches to path planning for mobile robots because the computational complexity is low enough for real-time operation with extremely limited resources [45]. The simplest lattice would be a regular grid, each connected to its neighbours by straight lines. Each node represents the x-y state of a holonomic robot, able to move in any workspace direction freely. Practical systems like cars or fork lift trucks are non-holonomic and primarily move forwards and backwards in the direction the wheels are pointing. In other words they are subject to differential constraints.

It is possible to carefully construct a lattice which captures differential constraints as detailed by Pivtoraiko et al [46].The trick is calculating the control inputs required to join a set of vertices which span state space (inverse kinematics) or sampling from the control space in order to generate a set of vertices (forward kinematics). The balance to be struck in the choice of resolution is that higher resolutions lead to a higher branching factor, increasing the memory required to represent a given area, but lower resolutions will give suboptimal paths.

Rapidly Exploring Random Trees are an example of a single query planner introduced by LaValle and Kuffner [47]. Most simply, a tree is constructed by sampling from state space close to an existing node in the tree. A local planner is used to generate a trajectory from the existing node to the new sample avoiding obstacle regions. If an obstacle free trajectory is found, a new node is added at the sample state with an edge from the existing node. Nearby nodes in the tree should also be checked and connected with edges if possible. Differential constraints can be incorporated by the local planner, so that only feasible extensions are considered. This can be done with splines with the desired degree of continuity as in [48].

Starting with the initial position and the goal position, this process is repeated until sufficient coverage of the state space is obtained. When a new sample is reachable from both trees, origin and destination must be connected and a graph method such as Dijkstra can be used to identify the vertices through the graph which make up the shortest path between them. The tree growing phase of the algorithm can be terminated at this point but it may be possible to improve the quality of the solution by adding more reachable states to the connected graph in case a better solution can be found.

A family of refinements to the basic RRT approach have been developed, which provide asymptotically optimal results. They continue to sample from the configuration space after finding an initial path. If enough time is available the globally optimal solution will eventually be found. In practise improvement over time can be very slow. Informed RRT* was developed to increase improvement rate in high dimensional spaces [49]. It is worth noting that 2D vehicle motion is an example of a low-dimensional space in which RRT can find high quality solutions much faster than real time, even with heading and curvature tacked on. The problems of artefacts due to limited sampling resolution have been mitigated by [50] who used waypoints to bias the search towards certain areas of the environment.

The Dynamic Window Approach is the default local (read Reactive/Tactical) planner implementation included in Robot Operating System [51]. Introduced in [52] it samples from reachable velocities to account for differential constraints leading to smooth motion around obstacles. It can be classified as a sampling based planner which integrates the vehicle dynamics into the local plans. It could also be classified as a potential field method as one way of selecting the best feasible velocity is based on the virtual potential field that would be reached at the next time step. The dynamic window is an example of searching in the control space, instead of configuration space. It is superseded by optimization approaches which search continuous control space to find feasible motions without sampling and also RRT methods which draw samples from control space rather than configuration space.

2.2 Tactical Motion Planning

Tactical motion planning addresses the rapid creation of a new short term plan. This is another way of adapting to latest sensor readings like the simple reactive methods reviewed in Section 2.1.2. To be called a tactical planner, an algorithm should plan a sequence of actions up to a planning horizon rather than just the next time step. One goal of sensor adaptive motion planning is computing the optimal alternative path, given an obstruction which invalidates the reference path. An approach based on precomputed clothoid curves arranged in a graph, centred at the vehicle location is described in [53]. It is important in an industrial environment where reference paths may be hand crafted and preferable to any seemingly convenient adaptive path for reasons of safety and predictability. The only reason to re-plan would be if the reference path is invalid, due to an unexpected obstacle. This problem is unlikely to result in a complex obstacle maze, and it is undesirable for a robot to explore it if it does.

Roadmap planners are still extremely useful at the global scale where local sensor updates are less relevant. One result is the split architecture described by [31] where a roadmap planner is used for strategic planning (planning over a longer time scale like a few minutes, extending from one part of the site to the other) and a different approach is used for tactical planning (creation of a detailed trajectory for the next few seconds). In [54] this approach, using different techniques for the two time-scales is described as integration planning.

There are other sampling based methods which can also be useful for path adaptation where local sensor updates are important such as Dense Random Trees as described in [55] [47] where the discretization is performed as the search proceeds. Sampling is frequently used to make problems of high dimensionality feasible, but can only offer resolution completeness. This is the guarantee that if a solution exists at the sampling resolution, then it will be found. Sampling from configuration space can result in paths which must be smoothed before they are traversable, so the most relevant techniques are based on sampling from the control space, or using parametrized curves so that every sample is feasible as in [48]. A frequent issue with dense random tree sampling methods is the introduction of artefacts in the solution which are difficult to remove by post-processing. Artefacts may include visiting locations on shortest path at the sampling resolution which could be bypassed at higher resolution.

By contrast, the family of solutions based on numerical optimization which operate directly on the continuous state space offer improved path quality and guarantees. These methods can be divided into parametric formulations which describe the path as some type of curve such as a polynomial [30] and those where the path is represented by a series of time samples which satisfy the differential constraints such as Timed Elastic Bands [34]. Compared to parametric methods, those optimising over a series of samples must search a much greater number of variables and also account for more constraints. This leads to additional computational burden, so they are often limited to a short time horizon and make use of a reference path to linearise obstacle constraints as the tactical planner in [39] which also uses output constraints to turn overtaking into a convex problem. For longer paths suitable for reuse parametric methods may be preferable, provided they are able to represent the dynamic limitations of the AGV.

Path representations which are suitable for the dynamic constraints of car-like vehicles can be based on different types of spline. Splines which are Cartesian can be calculated conveniently but they are only traversable if polynomial terms up to 5th order are included [56]. Polar splines have a smoothly varying curvature at first order and above but they are unable to represent a straight line they must be mixed with other curve types to form a complete path [57]. Other curve types such as Bezier curves exist but one representation which is particularly suited to industrial AGV roadmaps is the clothoid curve or Euler spiral [58]. Using this parametrization, and constraining peak curvature and sharpness, the resulting path will be feasible for a car-like vehicle at non-zero speed. The importance of sharpness limitation is sometimes overlooked but this is a real physical limit on the motion of a vehicle. This is because the sharpness is proportional to the angular acceleration at a constant traversal speed. Previous work on finding clothoid based paths around obstacles has mostly used spatial discretization to generate a series of points between the origin and destination, followed by curve fitting to find the clothoid segments which best fit to the points. This is a practical solution and variants of it are used by [59] who generates the key-points from a sequence of position samples from a manual drive, and [60] who fits to a series of predefined manoeuvres: u-turn, lane change and so on. As with other sampling based methods, the choice of sample points affects the final solution, leading to suboptimal solutions. This was made clear in [61] where sampling and curve fitting was compared to a direct optimization method.

An early method for creating continuous curvature paths based on clothoids, arcs and straight lines was the CC-Steer algorithm [62]. This local planner could be used to connect samples from configuration space to create a probabilistic road map. A similar algorithm from around the same time from [63] was also able to create CC-paths without using clothoids by considering curvature continuity while approximating a holonomic path with a heuristic exploiting the differential flatness of car pulling trailers. Differential flatness of a dynamical system indicates the prior states can be determined from the current state with no exogenous variables [64]. Fraichard et al [62] showed CC-Steer approximated Reeds-Shepp [3] paths which are provably the shortest for connecting points with heading continuity. Using the maximum sharpness and maximum curvature to produce the shortest path is fundamental to the operation of CC-Steer, but this is not the only important objective. Often it is preferable to minimize the sharpness of turns in order to reduce lateral forces and maintain high speed.

The contribution of Henrie and Wilde [60] was to describe an algorithm to join two configurations with the least maximum curvature and least sharpness to create comfortable paths similar to those a human driver would follow. This used symmetric clothoid pairs to assemble paths with the same structure as [62]. One limitation of this work is that the clothoid pairs are always arranged symmetrically which limits the range of manoeuvres. This was addressed by the bisection method proposed by [59] which performs a numerical search to find either two or four clothoid segments, which are not required to be symmetrical, only matched so that the curvature at the end is zero. Another contribution of [59] is to test one approach for creating smooth global plans by fitting arcs and lines and clothoids to a series of samples from a GPS trace of a test vehicle driven by remote control. [65] goes into further detail in an Appendix B regarding the reachability of clothoid pairs, but does not examine the geometric limits discussed in Section 3.3.2. The search procedure is fast at finding the parameters which meet the constraints but no algorithm is given for the correct choice of points to interpolate given a set of obstacles.

Recent works searching for paths with limited curvature rate such as [66] who used a superset of clothoids to find the best approximation to a holonomic path with limited curvature and provide a tuning parameter that could be tweaked to avoid obstacles. [67] also approximated holonomic paths but trained a neural network to speed up generation of the initial parameter guess. A double continuous curvature (DCC) path planner is a component in the path tracker described in [68]. The shortest DCC path from the current pose to the global reference path is found every control cycle by Nelder-Mead without considering obstacles. Silva et al [69] found a compact representation of a smooth road centre line consisting of arcs, lines and clothoids. Existing methods e.g.[70], could then be used to join certain key poses depending whether they would track a roundabout or a straight road. This approach is used to create entire sparse roadmaps based on clothoids in [71]. Others such as [72] used a smooth road centre line as a reference to linearise the obstacle constraints and then sequentially minimized maximum curvature and sharpness in addition to deviation from the reference path.

2.3 Multi-Robot Motion Planning

Coordinated conflict-free motion of a number of mobile robots in order to complete a material transfer task is important in the operation of fleets of AGV (Autonomous Guided Vehicles) used in flexible manufacturing and automated warehouses [19][73]. A crucial sub-problem is conflict resolution between the AGVs. Frequently motion conflicts are resolved as part of the Conflict-Free Routing Problem in the AGV domain, but not in the road traffic domain as explained in Chapter 1.

Reservation based approaches are a well known approximation for solving conflict-free routing as a centralized problem [74]. The roadmap is divided into discrete segments larger than the robot and each segment is assigned to at most one robot at a time. Robots must only occupy segments assigned to them to ensure they will not collide with each other. It is shown, in [2], that platooning provides superior throughput to the earlier reservation based systems, and that if a solution exists it is optimal, but not that a solution exists on all roadmaps. More importantly a set of conditions which must hold for a solution to exist, is not given [2]. The consensus algorithm in [75] also shows improved throughput in concert with a scheduling approach, but does not prove convergence.

AGV motion coordination can be posed as a variation of the Multiple Vehicle Routing Problem with the addition of challenging spatio-temporal constraints preventing collisions between each vehicle, as well as the usual timing and capacity constraints [77]. In [78], solutions are classified into centralized, decentralized and decoupled approaches. Each approach may be either optimal or heuristic based on whether or not they find the global minimum of some objective function. These features of the methods reviewed are shown by a check-mark ✓in Table 2.1.

Centralized methods which plan all of the vehicles' paths must be supported by effective numerical tools able to solve large combinatorial problems to optimality. Notably [79] found trajectories for aircraft using a linear approximation to the dynamics and obstacle constraints, creating a Mixed Integer Linear Program which could be solved with CPLEX. The contribution of [78] is to describe a new centralized, optimal method which scales well, finding a solution to the nonlinear-program for 10 vehicles in just a few seconds using an interior point method. However, size of the combined state space explodes as more vehicles are included so centralized methods are difficult to scale up to large numbers of vehicles.

Many types of decoupled methods have been developed because breaking the problem down into sub-problems is one way to reduce computation time for practical applications. Early methods of this type were based on timed petri nets [80] and agent based models [81]. Decoupled methods may sacrifice completeness (that a solution is found if one exists) in exchange for reduced average run-time. In [95], this was shown to cause practical difficulties in the spot-welding task studied. Spot welding requires close formation control of six vehicles, and the decoupled method frequently failed to find a solution. In [82], decoupled planning (incomplete) is compared with a new multi-phase heuristic, which is complete, for 50 robots on a tunnel map and 150 on an outdoor map.

Method	Optimal	Heuristic	Centralized	Decoupled	Roadmap	Adaptive	Complete
[74]		✓local	✓				
[2]	✓local		✓	✓	✓		
[75]		✓	✓		✓		
[76]		✓		✓		✓	✓
[77]	✓		✓				
[78]	✓		✓		✓		✓
[79]	✓local		✓			✓	✓
[80]		✓	✓		✓		✓
[81]		✓	✓		✓		
[82]		✓	✓		lattice		✓
[83]	✓		✓		lattice		
[84]	✓		✓		lattice		
[85]		✓	✓		lattice	✓	✓
[86]		✓	✓		✓	✓	
[87]	✓local			✓	✓	✓	
[88]	✓		✓		✓		✓
[89]		✓	✓	✓		✓	✓
[90]	✓local		✓		✓		✓
[91]		✓local		✓	✓		✓
[92]	✓local		✓		✓		✓
[93]		✓		✓	✓		
[94]	✓		✓		✓		✓

Table 2.1: Table comparing features of multi-robot co-ordination algorithms reviewed. The label 'local' with the ✓in the optimality column refers to the solution being local to an intersection.

Decoupled planning was consistently faster in execution and produced shorter paths for lower vehicle density however it failed to find any valid paths at all for high vehicle density (25 or more robots in tunnels and 75 or more outdoors). The multi-phase heuristic, being complete, found a solution in every test case.

More recently, [83] addressed the lack of optimality in decoupled methods operating on graphs. Optimal conflict-free motion is posed as a large Integer Linear Program. Resolution complete general purpose algorithms are used to solve it for 150 robots in just over 10 seconds demonstrating the tractability of integer methods if they can be linearised. The lattice/graph construction has recently been developed further to ensure kinematic constraints are met and improve coverage of state space around obstacles [84]. In [77], the combined problem of DCFVRP (Dispatch and Conflict-Free Vehicle Routing Problem) for flexible manufacturing is formulated as an integer program and two different decoupled algorithms are presented to solve it: local search and random search. Neither of the proposed algorithms is complete but local search found more valid solutions in the 10 random examples tested, all involving three vehicles.

Decentralized control is another option to decompose large scale problems which take too long to solve centrally [96]. Although limiting the information available to each decision maker can make reasoning about collective behaviour more difficult, various attempts to decentralize conflict-free routing have been made. In the field of conflict free routing for mobile robots, [85] presents a solution which generates a graph representation of the free space - effectively a roadmap - with the property of 'collision-avoidability.' This means that every node on the critical path must be at most one move away from a node that does not obstruct the critical path. The critical path is defined as the union of all the shortest paths between pick/drop locations in the roadmap. During decentralized planning, AGVs attempt to reserve 'private zones' consisting of the node on the critical path along with all adjacent collision avoidance nodes. Each AGV has an identical roadmap, plans the shortest path to its own goal and negotiates with those nearby based on a numeric priority to reserve the nodes in its own path. An AGV requests those in its path move to their collision avoidance node, and those with a lower priority will do so. Proof is given of correctness, that deadlocks are avoided, but throughput is sub-optimal with low priority vehicles frequently forced to stop and wait.

An alternative decentralized solution, based on a roadmap with two levels of detail is summarized in [86]. Conflict-free routing primarily takes place at the intersection-local level, based on prioritized roadmap reservation with local negotiation to guarantee correctness [87]. In [2], the speed of the approaching AGV is optimized at each intersection in a similar way to centralized intersection platooning. The result is higher throughput as time consuming negotiation is avoided in most cases. At the site-wide higher level congestion effects are represented by link performance functions. Intersections have a generalized cost which increases exponentially up to a fixed capacity which is identified by parameter tuning, An optimal task scheduling approach based on the Hungarian Algorithm is used to solve the full DCFVRP in [97].

Traffic delays are a type of emergent behaviour and modelling is challenging even in a completely automated system. This is the contribution of [89] which introduces the PRT (Probabilistic Reservation Table) to summarize the plans of robots including uncertainty so it can be in task scheduling. This approach is compared with a reservation based centralized planner as a baseline, in a simulation with 5 robots using a low level motion controller from ROS (Robot Operating System) which often fails at when two robots plan interfering paths. Congestion aware planning leads to fewer failures of the low level controller than independent planning but more than centralized planning. A more convincing comparison would be with a congestion aware planner using a deterministic congestion model instead of the PRT, but this is not reported.

Intersection control, based on platooning, is a concept developed for the operation of anticipated CARVs (Connected and Autonomous Road Vehicles). A recent review of approaches for intersection and merging coordination is given in [90]. Centralized optimization approaches improve on early ideas like First-Come-First-Served spatial reservation from [91] by minimizing fuel consumption, but the rapid increase in state space with larger numbers of vehicles is a challenge for large scale adoption. The communication channel connecting every vehicle with the central controller introduces a single point of failure, the reliability effect of which is difficult to evaluate in existing simulations. Moreover there are few CARVs available making a practical experiment unfeasible in most cases. Attempting to address these limitations are decentralized methods such as fuzzy-logic, virtual vehicle platooning and invariant set approaches. Notably the conditions for solutions to exist which minimize energy consumption are given in [92]. A feasibility enforcement zone precedes the conflict zone. If the vehicles enter the conflict zone in a feasible condition then the optimal control can be used. If the speed and arrival time are not feasible then a suboptimal safe control must be used instead.

Recently [75] described an approach to the DCFVRP for flexible manufacturing based on dynamic platooning with vehicle-to-everything messaging and consensus speed control, resulting in a decentralized heuristic solution with some additional rules to ensure correct behaviour and avoid deadlocks by adding a reservation protocol for some parts of the roadmap. This was combined with feedback from the queue length at different workstations in a traffic management heuristic. Simulation results show an impressive improvement compared to the first-come-first-served scheduling approach meant to represent industry standard practice.

A promising approach for intersection control applied to AGV is given in [2]. As only the speeds are modified not the paths, the argument is that everyone must reach their destination eventually, proving liveness as speeds are always greater than zero. However, a backup system based on negotiation is still required because the problem is non-convex suggesting a feasible solution may not be found in time for certain roadmap and traffic combinations. The consensus based platooning method for local collision avoidance used in [75] is unusual in the AGV domain. That work makes no claims about completeness, but does consider the trivial consensus where all vehicles stop in a deadlock. In [93], a system for conflict avoidance based on time headway is shown to significantly reduce intersection crossing time and allow more vehicles to operate in the same floor-space compared to a traditional reservation based strategy. Of these different approaches to platooning for AGV coordination, the quadratic constraints of [2] are the closest to achieving the certainty required to build a distributed coordination scheme.

2.4 Automated Intersection Management

Autonomous Intersection Management (AIM) for near future Connected and Autonomous Road Vehicles (CARVs) offers an alternative vision for multi-vehicle collision avoidance, possibly suitable for recent AMR with a high level of autonomy. A recent review is given by Zhong et al [98]. Car-following conflicts are sometimes explicitly excluded such as [99]. Here, CARVs are assumed to maintain a safe headway using their own sensors, perhaps in a similar way to the Adaptive Cruise Control assistance feature available in many production vehicles. This assumption may be prove correct for CARVs on future public roads, but is an important issue for state-of-the-art AMR, where the impact of interactions between vehicles in the same lane must be considered to ensure safe motion across an intersection. For example, even a smart AMR able to overtake may reach precomputed conflict points at a different time, or conflict with different opposing paths. In some cases overtaking will be impossible due to opposing traffic or vary narrow aisles, so it is a constraint on the AIM problem which needs to be taken into account.

Zhong et al [98] includes a summary table of AIM approaches, which indicates that little attention has been given to the corridor coordination layer. This is the co-ordination of motion on lanes linking one intersection management zone and the next. In many cases, intersection management zones are assumed to be so far apart interaction effects can be neglected. One of the few works addressing interactions between intersection zones is [100] which uses a 3-layer hierarchy where the manager of each intersection sends the average traffic density to its neighbours and chooses speeds within its zone of influence to minimize deviation from average road velocity, subject to the constraint that conflicts are avoided in the single zone spanning the intersection.

Studies on the theoretical capacity of signalized intersections and roundabouts with an equivalent footprint indicate that in most cases, if there are few approach lanes small roundabouts will tend to have higher capacity. If there are many approach lanes signals tend to be more effective, unless the traffic on different approaches is extremely unequal [101]. A systematic procedure computing the conflict points in an intersection is given in [102]. Roundabouts tend to have a large number of merging and diverging conflicts, but fewer or none of the crossing and head-on conflicts which lead to the most serious collisions due to high relative speeds.

Intersection control often addresses crossing conflicts by separating vehicles in time, while they all take the shortest path straight through the intersection in the same way as if it were signal controlled. There are a wide range of optimal and heuristic approaches to solve for the speed profile, both decentralized and centralized, a good review is given in [90]. Many studies have looked at how to incorporate a proportion of human controlled vehicles which are not able to communicate their intention. One way of doing this is using traffic signals which only apply to human drivers [103]. The downside is that the nature of the intersection must remain similar to a traffic-light controlled one if non-communicating participants are going to be controlled by lights.

A number of studies have extended intersection coordination of Connected and Autonomous Road Vehicles (CARVs) to resolve the type of merging of diverging conflicts which occur at roundabouts. These are reviewed in [90]. A centralized solution with an intersection manager minimizing delay and energy consumption is described in [104]. This shows that a high proportion of vehicles need to be communicating for significant benefits to be realized.

A decentralized approach based on intent communication by way of virtual vehicles, can also be applied to roundabouts. In [105], reactive heuristics are shown to lead to poor performance compared to a model predictive control approach. The virtual vehicle concept allows common lane based heuristics such as car following to be extended to resolve conflicts in [106]. Another work investigating virtual lanes is [107]. Here a conflict graph is used to assign approaching vehicles to appropriate virtual lanes and a distributed controller is presented to stabilize the platoon.

Another approach presented in [108] is a decentralized solution to the global problem of minimizing the delay. The communication-enabled consensus based ordering heuristic is based on the concept of temporal advantage. One agent has temporal advantage over another if it is able leave its last conflict zone earlier, according to the motion plan they all submit on approach. The feasibility and stability of the multi-agent system when all vehicles were using the same strategy were proved theoretically. Collision constraints are based on conflict zones rather than conflict points as in [109]. The location of the conflict zones is fixed by the fixed paths between the entry and exit lanes of the junction. The space inefficiency of the zone representation for multiple lanes is addressed by using multiple zones, one for each pair of lanes. The ordering problem is resolved in a decentralized way based on game theory and the game 'Chicken.' Each agent computes the time it will arrive at a decision point approach an edge over the mixed integer optimization used in [109], in terms of how many vehicles they can control before running into execution time limits. It is a little surprising that the game would always produce the optimal ordering given the motion model used by each AGV. The consensus mechanism will be important here. Questions remain about the possibility of AGVs disagreeing about the order they calculate from the communicated position and speed data.

A similar method which solves the ordering problem sequentially, followed by individual optimization of the approach speed along fixed paths is described in [110]. This method claims only local (per-vehicle) optimality for the speed choice sub problem, and makes it clear the crossing order at convergence will be suboptimal, and depends strongly on the decision order. The sub problem is posed as a Linear Quadratic Regulator, commonly seen in optimal control problems. In general terms, those early in the decision order will deviate from the plans less. This is more of a problem when vehicles are not uniform, as to reduce energy consumption a late arriving lorry should deviate as little as possible. A heuristic is given for the decision order based on the time to conflict arrival.

The use of optimal control in [110] is shared with many earlier works regarding coordination of Unmanned Aerial Vehicles, many of which relax the assumption of static paths. In this way [111] addressed the full multi-vehicle motion planning problem for small numbers of aircraft with simple dynamics. The craft were assumed to be differentially flat: that is, able to actuate in any of the workspace degrees of freedom independently, like a quadrotor. They were represented using bounding rectangles, leading to a slightly conservative mixed integer problem. The integer variables are used to choose which constraints are active. This might seem excessive when representing static obstacles, however when the constraints arise from other moving vehicles, the integer variables are a natural way to represent the passing-order problem. The scaling to larger numbers of vehicles is a particular challenge, due to the combinatorial explosion of possibilities.

An alternative approach to the coordination of differentially flat aircraft which uses a sequential solution of per-vehicle receding horizon sub problems to approximate the global solution is given in [112]. An earlier theoretical treatment based on iterative bargaining with soft collision constraints is given by [113]. The parameters are real numbers, and the constraints linear while the cost is quadratic. It may converge to an infeasible solution given a particular minimum safety distance even from a valid set of starting positions and speeds, and the suggested solution is to reduce the threshold until it becomes feasible.

Solutions based on Distributed Model Predictive (DMPC) control have also been developed. In [114], per-vehicle optimizations runs simultaneously to reduce execution time. This ensures recursive feasibility and closed loop stability. Another DMPC approach is given by [115]. This scales up to 25 vehicles in real time. the quadrotors concerned are all identical and differentially flat. For an under-actuated system like an

AGV, some of the simplifications may no longer be appropriate.

Automated Intersection Management (AIM) techniques have been proposed to replace traffic light signals at intersections for Connected Autonomous Road Vehicles (CARV) in numerous studies reviewed in [116]. Approaches can be divided into different heuristics and those based on optimization. An early heuristic known as reservation based control put forward by Dresner and Stone [91] required each vehicle to submit a motion plan across the intersection a which could either be accepted or rejected. Reservation based control improved capacity in some traffic situations and is suitable for real time operation with many vehicles. However, certain traffic scenarios could suffer reduced throughput compared to appropriately prioritised traffic signals [117]. One way this could be addressed is by the use of pressure based policies [109]. Others have suggested extending distributed controllers for longitudinal platooning with virtual vehicles [107].

Another study looking at multiple intersections, in this case for on-road Connected Autonomous Vehicles (CARVs) is [100]. Here a scheduling top layer finds the right speeds to avoid crossing collisions, while a longitudinal Model Predictive Control (MPC) layer minimizes the speed tracking error at the entry to each segment. Nearby intersections exchange Intersection-to-intersection (I2I) messages to assist traffic distribution. Another approach using a similar hierarchy and Model Predictive Control for the lower layer but optimising the scheduling for minimum travel time as a mixed integer linear program (MILP) is given by [118]. The scheduling (upper) layer was modified to minimize energy consumption in [94].

The potential of forward guidance approaching intersections has been studied for human operated vehicles where it can lead to significant fuel economy improvements [119]. This is a result of the combination of vehicles travelling at lower speeds (reducing air resistance) and avoidance of the sharp acceleration after giving way at many junctions.

Another approach from Tettamanti et al combines Autonomous Intersection Management with network wide traffic control [120]. A macroscopic fundamental diagram (MFD) relating traffic density on a link to the average speed is approximated with a second order by repeated runs of the Krauss car-following microsimulation model used in the intersection layer. This differs from the MFD used in [100] which was a piecewise linear modified Greenshields model. In [121], a piecewise linear model is used for motorway traffic, this time a Godunov-discretized LWR (Lighthill and Whitham, 1955) model. Autonomous Intersection Management V2I comms and seems likely to substantially alter the relationship between density and speed, so estimating the link capacity and speed response by repeated runs of the intersection controls might be necessary to ensure the upper layer can make the appropriate routing decisions.

A messaging interface for AIM which permits speed optimization is introduced in [109] for an isolated intersection. The interface for quadratic programming formulation for industrial robots described by Digani et al [2] is similar. In this work multiple intersection zones are coupled to span the entire roadmap. Every place where two lanes intersect in the roadmap is identified and nearby elementary conflicts are collected into an conflict zone. The intersection manager for one zone only has authority over approaching vehicles, those within the intersection and departing can no longer be controlled, but their state is used as a constraint on the following vehicles.

A vehicle departing one intersection is by definition approaching the next one, and new speed instructions should be received as soon as communication with the next Intersection Manager can be established. As the speed of a departing vehicle is set by the next Intersection Manager, and the position of that vehicle forms a constraint on the solution to the speed of approaching vehicles computed by the previous intersection it could be argued that car-following behaviour is the fundamental mechanism for coupling intersection zones. Car-following in [2] is based on roadmap reservation, reflecting the state of practice in industrial robotics. This leads to constraints that one vehicle cannot enter a roadmap segment, until the preceding one has departed. Provided the segments chosen are long enough, this provides guaranteed safe behaviour. In many cases it will be overly conservative, especially when one vehicle is following another at a similar speed. As both are in motion, in the same direction they can follow each other much closer than the braking distance either vehicle would need to stop for a stationary obstacle.

Other works have investigated the car-following behaviour [122] using the Rakha-Pasumarthy-Adjerid (RPA) model, which accounts for vehicle dynamic limitations. Only an isolated intersection is modelled, but if it were used for Corridor Co-ordination it should be less conservative than roadmap reservation. The guarantees of correct behaviour need to be considered carefully as they could break down in some situations, such as those examined by [123].

2.5 Summary and Research Gaps

In the field of sensor adaptive motion or reactive planning reviewed in Section 2.2 numerous techniques have solved the problem for planar wheeled vehicles with dynamic limitations. Sampling based methods can create suitable paths for car-like vehicles by sampling from the control outputs rather than the configuration space directly. The asymptotically optimal methods are promising, but will not reach a global optimum in finite time. Methods based on convex optimization can improve on this. A gap is identified in investigating the use of clothoid basis functions for reactive planning using convex methods along with a convex obstacle representation of the free space.

Approaches to multi-robot planning methods span from per-agent decentralized to fully centralized as discussed in Section 2.3. Guarantees excluding deadlocks and livelocks are essential for real world operation. The latest decentralized methods are only able to achieve this by permitting suboptimal behaviour such as reversing and waiting. The approximations of zone occupancy needed to make the DCFVRP problem tractable have a detrimental effect on space utilization also. Introducing the concept of AIM has the potential to improve on this with car following behaviour. Further the division of control separating routing and scheduling from conflict resolution by speed adjustment has only received limited attention.

The methods for AIM reviewed in Section 2.4 take a variety of approaches, some locally optimal some heuristic. The relative importance of ordering optimization compared to model based speed adjustment alone is not clear for existing work on AIM for material handling. Another research gap identified is the study of the transition of control from one intersection zone to another nearby zone. The question of how close the intersection needs to be before one must communicate with the other to ensure correct behaviour. There has been limited work linking multiple intersection managers to resolve motion conflicts while scheduling and route planning are solved independently. Using AIM to guarantee intersection safety, there is an opportunity to improve throughput by following more closely. The remaining research questions of interest to improve decentralized control of mobile robot fleets are:

1. RQ1: How can reactive planning be added to simple mobile robots following clothoid based paths without reducing path quality?

2. RQ2: How can motion conflicts between vehicles at intersections be resolved

independently of scheduling and routing?

3. RQ3: How can independent intersection managers with average speed authority be modified for a safe handover of control between nearby intersections?

Chapter 3

Optimal Smooth Paths based on Clothoids for Car-like Vehicles in the Presence of Obstacles

3.1 Chapter Abstract

Motion planning for a single drive unit as shown in Figure 3.1 is the building block of the entire fleet control/RMF system and as such deserves particular attention. This chapter presents a new method for obstacle avoidance planning with clothoid curves and polygonal obstacle constraints. The clothoid splines produced have lower peak sharpness than polynomial splines, without artefacts. This is achieved by division of the problem into topology, solved with a graph method followed by curve fitting to regions rather than intermediate points.

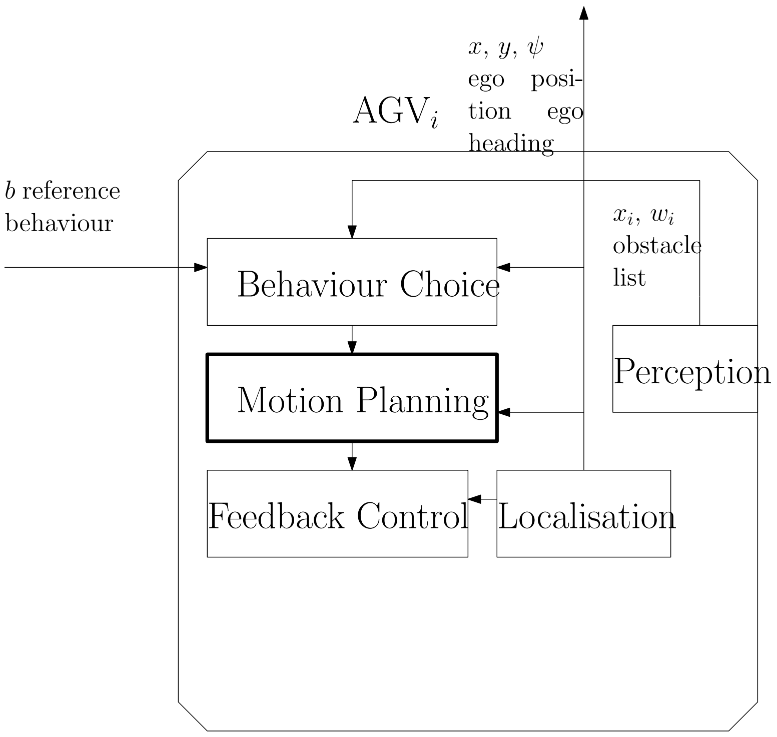


Figure 3.1: Motion planning and closely related activities for a single AGV.

3.2 Introduction

Consider a fleet of Autonomous Guided Vehicles (AGV) moving material in an automated manufacturing plant. A lattice roadmap made up of virtual guide paths is a widely used solution for planning the motion of each AGV [57]. This is designed by engineers installing the AGV system, and may remain in use for many years. In environments which are shared with humans and human operated vehicles there is a greater probability of unexpected obstructions blocking the guide paths. Automatic replanning to avoid these obstructions has the potential to increase performance and robustness of shared environment AGV systems. Although numerous techniques have been developed for planning paths of car-like vehicles around obstacles [31], none has achieved wide acceptance in industry [19] [124]. The problem addressed in this chapter is finding a smooth path around obstacles which can be followed exactly by a vehicle with car-like dynamics.

It is important to consider the variety of solutions which have been developed already in this area. A range of techniques for motion planning are well described with examples in [42] including graph search methods such as Probabilistic Roadmaps using Dijkstra's algorithm in addition to incremental search methods like Rapidly Exploring Random Trees. [125] goes into more detail for only those techniques suitable for on-road autonomous vehicles and [31] gives optimality and completeness results for many in a handy comparison table. Recent developments have used gradient descent to modify Bezier curves based on obstacle keypoints [35], evaluated alternative clothoid tentacles [126] and found the parameters of interpolating clothoids as an optimization [127]. The benefits of clothoids for controlling lateral acceleration identified in [128] can be exploited by unoccupied vehicles, which can travel faster on smooth paths without lateral instability [129][68].

3.3 Mathematical Representation and Problem Definition

Clothoid curves are widely used and appreciated for creating smooth drivable paths with limited angular acceleration. They are less frequently used within a numerical optimisation framework, instead many heuristic methods for calculating their parameters have been developed. In what follows the properties of clothoids which are relevant to numerical optimization are identified.

3.3.1 One Clothoid Segment

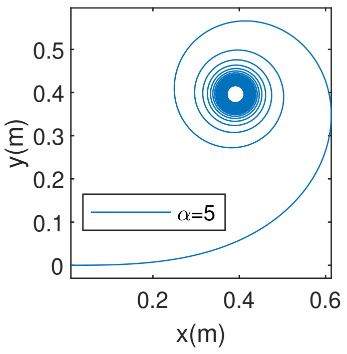


Figure 3.2: Clothoid spiral with $\alpha = 5$ for $s > 0$

The clothoid is defined as a curve whose curvature κ increases linearly along its length. The rate of increase is called the sharpness $\alpha = d\kappa/ds$. Points on such a curve are defined by two parameters; the arc length s and the sharpness α. These two parameters describe a curve spiralling out of the origin along the x axis towards one asymptote with infinite positive curvature (turning anti-clockwise) in the positive $x - y$ quadrant for $s > 0, \alpha > 0$, and towards a negative asymptote with infinite negative curvature in the negative $x - y$ quadrant where heading is decreasing (turning clockwise) for $s < 0$ or $\alpha < 0$. The negative distance curve is just a reflection of the positive one and all arcs can be joined by appropriate rotation and translation; so in what follows $s \geq 0$. The positive part of the curve with sharpness $\alpha = 5$ is shown in Figure 3.2. It can be evaluated in Cartesian coordinates with the Fresnel integrals which are reproduced in Equations 3.3-3.4. The change in angle over one segment is the deflection δ.

$$\kappa = \kappa_0 + \alpha s \tag{3.1}$$

$$\delta = \int_0^s (\kappa_0 + \alpha u) du = \kappa_0 s + \frac{\alpha s^2}{2} \tag{3.2}$$

$$x = C(\alpha, s, \kappa_0) = \int_0^s \cos(\kappa_0 + \alpha u) du \tag{3.3}$$

$$y = S(\alpha, s, \kappa_0) = \int_0^s \sin(\kappa_0 + \alpha u) du \tag{3.4}$$

The symbol ψ will be used for resultant heading angle after a number of segments. Whereas each δ increases unbounded, ψ is an angle measured clockwise from positive

x direction and may be wrapped in the range $[0, 2\pi]$ without consequence. The configuration of a rigid body in a 2D plane with components $[x, y, \psi]$ will be referred to as a pose. See Appendix C for more details. The range of poses reachable by varying the parameters of a single segment are limited. Any x and y position can be reached by choice of parameters α and s - this makes intuitive sense as the parameter space is two dimensional as are the constraints. In order to meet heading ψ and curvature κ constraints as well, a spline composed of multiple clothoid segments is needed. Note that a clothoid segment with $\alpha = 0$ will form either a straight line of length s if the initial curvature is zero or an arc of length s otherwise.

3.3.2 Required number of Segments for Interpolation

First, consider G^2 interpolation with clothoids. This involves fitting clothoid segments to a series of points with fixed $[x, y, \psi, \kappa]$ as addressed in [127][130]. It is helpful to understand the way the required number of clothoid segments varies depending on the constraints applied. To optimise some objective function of the curve it is required that the solution is under-determined, that numerous feasible solutions to the interpolation problem exist allowing us to search over them to find the best.

G^2 continuity, where position, heading and curvature match where one segment joins the next leads to a smooth path which is traversable for a car-like vehicle [59] [60]. Its importance for fork lift operation is detailed in [68]. As explained in Section 3.3.1 there are two additional degrees of freedom available for each clothoid segment included in a G^2-continuous spline. The requirement to end on a specified point with $[x, y, \psi, \kappa]$ provides four constraints. This implies that there is a unique solution for two segments per point as this gives rise to four parameters and four constraints.

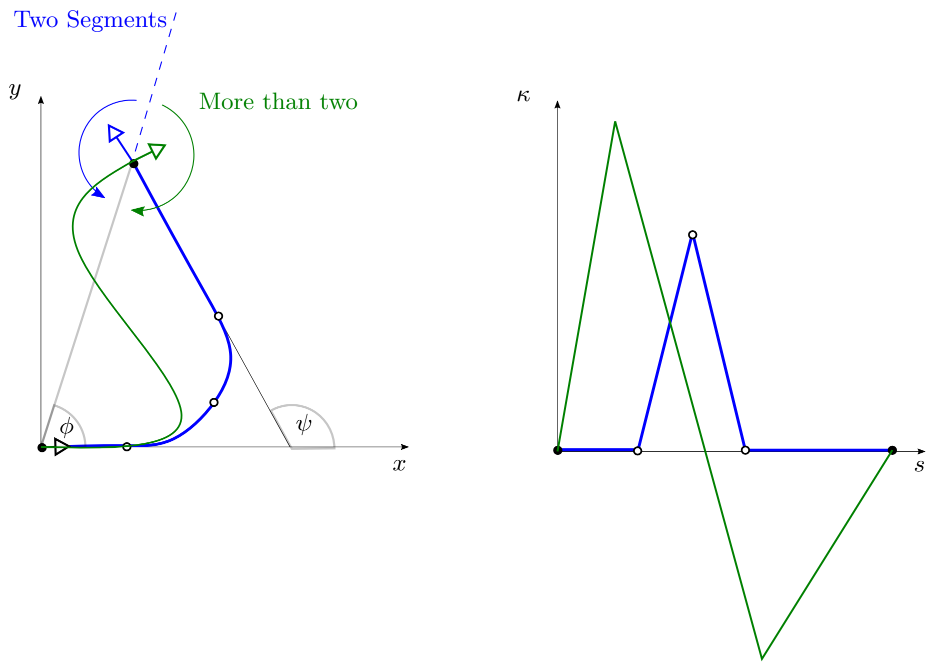
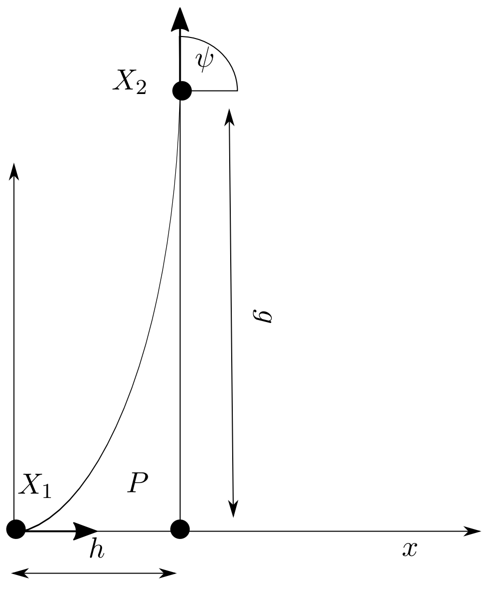


Figure 3.3: The angle ϕ of the line joining the origin with the target position (x, y) is the lower limit on heading ψ reachable with two clothoid segments (the blue curve). Smaller angles can be reached by adding a third clothoid segment to create an S shape as shown in green. Corresponding curvature κ with arc length s trace is plotted alongside.

In order to connect a clothoid segment with a straight line keeping G^2 continuity, the clothoid must have zero curvature at the point of connection. A clothoid pair with zero curvature at the end can be described as 'matched': The curvature change $\kappa = \alpha \cdot L$ over each segment is equal and opposite. Such a pair is shown in Figure 3.3 alongside a curvature-arc-length plot for illustration. Gim et al [59] show that the minimum reachable heading with two matched clothoids is given by Equation 3.5.

$$\psi > \phi = \arctan\left(\frac{y}{x}\right) \tag{3.5}$$

This is because an s-shaped curve cannot be formed with only two segments as illustrated in Figure 3.3. In [59] smaller angles are addressed with a second algorithm which computes two matched clothoid pairs (four segments total) instead. The advantage of the method is that the clothoids do not have to be symmetrical, so the final position can be reached with a lower sharpness and peak curvature as well as a shorter path length in some cases compared to using symmetric clothoid pairs.



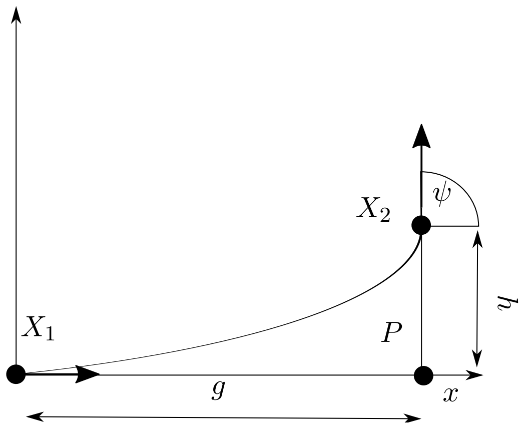


Figure 3.4: The two limiting positions for an unsymmetrical clothoid pair without line segments reaching an angle ψ according to the g/h condition in Equation 3.6. For $\psi = 90$, $\frac{I_C(\psi)}{I_S(\psi)} = 1.7749$

The condition for the existence of an unsymmetrical clothoid pair to fit three non-collinear points is given by Equation 3.6 reproduced from [131].

$$\frac{\frac{g}{h} + \cos(\psi)}{\sin(\psi)} < \frac{I_C(\psi)}{I_S(\psi)} \tag{3.6}$$

Equation 3.6 gives a maximum ratio of g/h in terms of the final angle ψ. The length g is always the distance to the furthest point from the intersection P and h is the distance to the closer point, as shown in Figure 3.4. As the condition is given in terms of points rather than poses, the requirement that the points not be collinear is equivalent to Equation 3.5 because $\psi = \phi$ would indicate collinearity and $\psi < \phi$ would result in the opposite initial heading given by vector $X_1 - P$. In order to avoid very short, high sharpness segments, straight line segments can be introduced before or after the clothoid pair as suggested in [131]. This allows unequal cases beyond the limits shown in Figure 3.4 to be fitted.

Using six parameters to describe two matched clothoids with zero curvature at the beginning and end permits the inclusion of a straight line at either end. The parameters would be two for each clothoid segment (total four) and two more for the length of the straight line at either end. The paths created for automated driving by [59] use a matched clothoid pair (two segments) for any corner. With this number of segments, there are certain unreachable poses as an s-shape cannot be formed within one region. The limited flexibility of these paths should be sufficient for executing a turn and returning to zero curvature in each region.

3.3.3 Generating a Convex Region Representation from LiDAR Data

A widely used representation for a 2D obstacle field is an occupancy grid [132]. Also see Probabilistic Robotics textbook [133] for details on creating a map from sensor data. Each cell represents an area of the floor, with a number $p \in [0,1]$ indicating the probability it contains an obstacle. Thresholding can be used to create a binary map of occupied and unoccupied cells. In order to connect adjacent unoccupied cells to form regions numerous approaches from image processing are available. A method suitable for simple environments is vertical cell decomposition [134]. This begins with piecewise linear polygons, which can be created by connecting the cell corners of a binary occupancy grid.

A single scan of LiDAR data from one sensor in a 2D workspace may be represented as list of range measurements r_i at a sequence of closely spaced bearings θ_i from a sensor pose $[x_i, y_i, \psi_i]$. This measurement indicates the presence of an obstacle at a distance r_i in the direction of the ray. The same measurement dually indicates the absence of an obstacle at any point along the ray from $[x_i, y_i]$ to $\boldsymbol{p}_i = [x_i + r_i \cos\psi + \theta_i, y_i + r_i \sin\psi + \theta_i]$. If the sensor is co-located with the AGV then connecting up each $\boldsymbol{p_i}$ where $i \in \{1, N\} \in \mathbb{Z}$ over one complete scan forms a piecewise-linear polygonal boundary of the free-space region surrounding the AGV. The observation is valid at sampling time t_i and is likely to be superseded by the following scan in the same direction. This arrives $t_i + T_s$ later, where T_s is the sampling rate of the LiDAR, as little as 25 ms with some models [135]. The sampling density is 1081 points over a 270 degrees (2.356 radians) field-of-view, providing 2.180 milliradian resolution. Downsampling the boundary can be useful as each line segment leads to an additional constraint on the path optimization. Algorithm 1 was used to create a downsampled convex empty region which excludes the entire high resolution boundary.

Algorithm 1 produces a lower bound convex region with a N boundary points which excludes every point in a LiDAR scan with $m > N$ points. A set of possibly overlapping convex regions similar to the test environments could be generated by placing a 2D

Algorithm 1 An algorithm for downsampling convex empty region boundaries.

```
Require: $0 < N < m$
 1: $step \leftarrow$round$(m/N)$;
 2: $j \leftarrow 1$;
 3: $X \in \mathbb{R}^N$
 4: $Y \in \mathbb{R}^N$
 5: $OuterX \in \mathbb{R}^N$;
 6: $OuterY \in \mathbb{R}^N$
 7: $OuterIdx \in \mathbb{Z}^N \in \{1, N\}$;
 8: for all $i \in \{1 : N\}$ do
 9:     $j \leftarrow j + step$
10:     $X(i) \leftarrow x(j)$
11:     $Y(i) \leftarrow y(j)$;
12:     $v_1 \leftarrow [x(j - step), y(j - step)]$
13:     $v_2 \leftarrow [x(j), y(j)]$
14:     $Delta \leftarrow v_2 - v_1$
15:     $\hat{n} \leftarrow Delta/\text{norm}(Delta)$;
16:     $scalar_b \leftarrow \hat{n} * v_1^T$                      ▷ % inner product
17:     $A \leftarrow \hat{n}^T \hat{n}$                              ▷ % outer product
18:     $b \leftarrow scalar\_b\hat{n}$                         ▷ % scalar times vector
19:                        ▷ % search high resolution points to find extremes
20:     $extreme\_outer\_vertex = v_1$;
21:     $j\_outer = j + 1$;
22:     for all $k \in \{1, step\}$ do
23:         $v\_k \leftarrow [x(j - step + k), y(j - step + k)]$
24:         $edge\_normal_vector \leftarrow A * v\_k^T$
25:         $edge\_distance \leftarrow edge\_normal\_vector^T \hat{n}^T$
26:         if $edge\_distance > A * extreme\_outer\_vertex^T \hat{n}$ then
27:             $extreme\_outer\_vertex \leftarrow v\_k$
28:             $j\_outer \leftarrow j - step + k$
29:         end if
30:     end for
31:     $OuterX(i) \leftarrow extreme\_outer\_vertex(1)$
32:     $OuterY(i) \leftarrow extreme\_outer\_vertex(2)$
33:     $OuterIdx \leftarrow j\_outer$
34: end for
```

LiDAR on a tripod at a fixed location close to the centre of every room-like area in the obstacle map. The location should ideally be the intersection of the longest corridors connected to the room. Collecting a number of scans from a stationary LiDAR allows the readings to be averaged and zero mean Gaussian noise in the range estimates suppressed somewhat. One scan location would produce one convex region capturing the shape of the room and some distance into the corridors. Lacking a local coordinates system to express the metric location of the fixed locations can be side-stepped in many instances with a bit of ad-hoc surveying. For example, mark a line between either end of the longest line-of-sight in the building with a chalked piece of string to create the x-axis. Then place a target (anything which gives a decent return to a hand held laser range finder) at interest locations along this line marked with chalk and record the x-values. Place a target at the intended fixed regions survey locations and measure perpendicular distance with a handheld laser range finder to get the y-value from each marked interest point. For a cuboidal warehouse with few corridors this works quite well, but may need to be adapted or even be completely impractical for non-square environments with diagonal corridors and multiple levels.

3.3.4 Obstacle Field Representation

Any field of polygonal obstacles can be equivalently represented as a set of possibly overlapping convex regions of free space [42]. Path planning within convex regions can be divided into the following steps:

1. Spanning. Convert obstacle representation into a small number of possibly overlapping convex regions which span the free space.
2. Assignment. Assign path segments to a sequence of connected regions between the region containing the start to the region containing the goal.
3. Curve Fitting. Solve for the best path from the start to the goal which remains within this sequence over its entire length.

The ‘Spanning’ problem involves calculating a minimum number of spanning regions and is an NP hard problem in itself. A common representation of an obstacle field constructed from range data is an occupancy grid [132]. This consists of a 2D array of cells and can be created from uncertain range measurements from vehicles which are

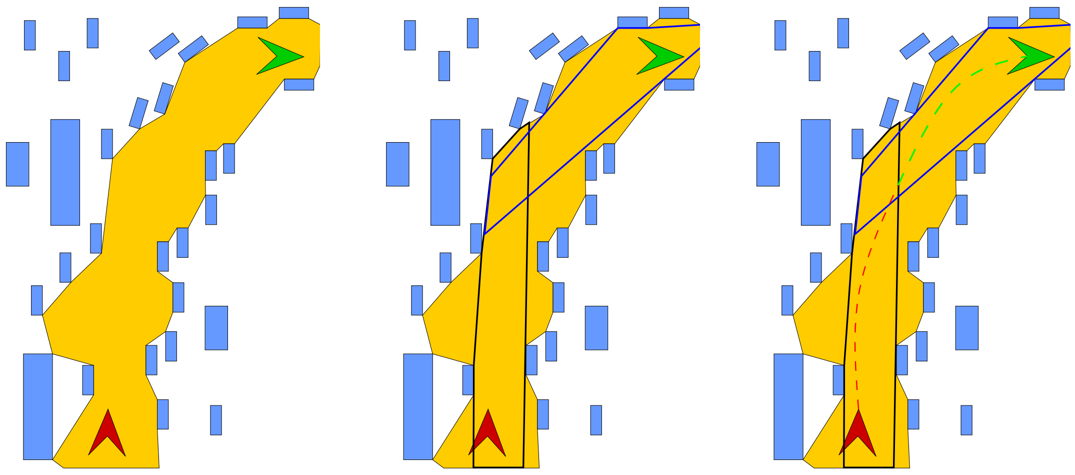

(a) Free Space Boundary (b) Convex regions spanning the free space (1. Spanning) (c) Polynomial segments fitted to regions (2. Assignment and 3. Curve Fitting)

Figure 3.5: Planning process

able to estimate their own position [133]. Each cell represents an area of the floor, with a number $p \in [0, 1]$ indicating the probability it contains an obstacle. A threshold can be used to create a binary map of occupied and unoccupied cells. The coarse obstacle-free region sets used in the numerical examples can be generated using the vertical decomposition method [134]. This begins with piecewise linear polygons, which can be created by connecting the cell corners of a binary occupancy grid. More complex environments and cluttered obstacle fields could be addressed using Iterative Region Inflation by Semi-definite Programming as described in [136].

The advantage of the obstacle regions being convex when they form one set of constraints on the path search, is that the resulting constrained optimization problem can retain the property of convexity, provided the objective function and the rest of the constraints are also convex. If this is so, the path search problem can be solved guaranteeing optimality (that no other parameter set can minimize the objective further) and completeness (that if a path exists it will certainly be found). Given that the polygonal representation is only an approximation to the real obstacle field, it can only offer completeness within the limits of the resolution used to represent the obstacles but this provides more freedom than a series of pose samples chosen heuristically and may permit lower cost solutions in some cases.

In this project 'Assignment' was addressed in isolation using Algorithm 2 in Appendix A, with a graph constructed from individual convex regions similar to [137]. Approximate shortest paths through the topology graph allow 'Curve fitting' to be solved separately with convex methods. Section 3.4 goes into detail describing a method to solve the 'Curve Fitting' problem with constant sharpness segments. Deits and Tedrake [30] use a mixed integer formulation which addresses 'Assignment' and 'Curve Fitting' simultaneously to achieve optimal results with cubic and quintic path segments. Extending this with exact clothoid integration for the segments leads to a non-linear mixed integer problem, which can be time consuming to solve. Instead, 'Assignment' is solved as a graph problem with a node representing each convex empty region. Edges are inserted between regions which are connected. Connected regions can be found easily in configuration space as the vehicle is reduced to a point and therefore if any corner of a region is contained within any other, it is certain one is reachable from the other and they should be connected in the graph. Using the straight line distance between region corners to give an edge weighting, results in an approximation of the shortest global path. In this way the topology can be solved separately, compensating for one weakness of numerical optimization, the 'topological blindness' identified in [37]. 'Topological blindness' can be described as a failure to search discontinuous deformations, such as a path which travels the other side of an obstacle from the initial guess. Paths found in this work will be suboptimal for the combined problem of 'Assignment' + 'Curve Fitting' because the assignment heuristic is based on the corner to corner distances used as edge weights in the topology graph, rather than the length of the fitted path segments.

A shortest path encodes a sequence of q empty regions, starting from the region containing the start and leading to the region containing the goal. Such a sequence can be encoded in a binary matrix $H \in \mathbb{Z}^{(p \times q)} \in \{0, 1\}$ which indicates the containment of p path segments in q regions.

3.3.5 Test Environment 1

The first test environment was created for a hypothetical item fetch AGV with a payload of 50kg and a top speed of 1m/s.

A simple region shape for exposition is an axis aligned rectangle extending from x_{min} to x_{max} and from y_{min} to y_{max}. Regions for the local avoidance problem are shown in Figure 3.6. The assignment is assumed between N regions and N clothoid pairs. Referring to Figure 3.6, the first pair will be assigned to the region on the left (blue rectagle), the second pair to the region across the top(red rectangle) and the last pair to the region on the right (amber rectangle).

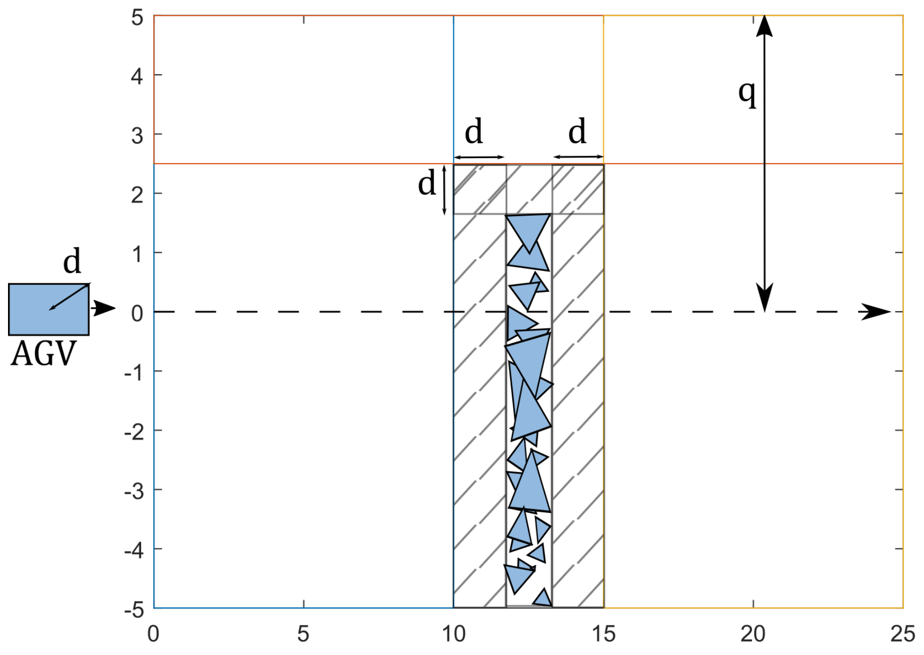


Figure 3.6: Example of an obstacle field represented as a set of possibly overlapping empty convex regions. The reference path is a straight dashed line along the x-axis (to the right). The largest dimension of the vehicle body used to expand the obstacles d and the maximum deviation from the path is q.

3.3.6 Test Environment 2

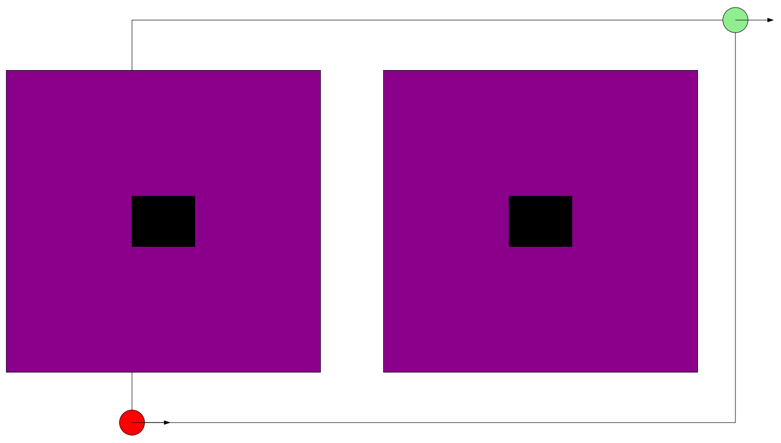


Figure 3.7: Pallet environment. Obstacles are black with expansion by the vehicle disk shown in purple.

The second test environment shown in Figure 3.7 was created for an automated fork lift AGV research platform [14]. The obstacles are based on (1.0×1.2)m pallets which are commonly used in the UK and Netherlands [138]. The datasheet for the manually operated vehicle on which the AGV is based, a Hyster E30-40HSD [4], gives the dimensions in Table 3.1. The plan of the vehicle is given in Figure 3.8.

Table 3.1: Dimensions from datasheet [4]. *Stopping distance R based on top speed 3.22m/s and hypothetical braking deceleration of 4m/s^2

Parameter	Dimension (mm)
W	1067
L	1583
r	1289
L_s	1001
S_0*	1200
R*	1300

The datasheet gives the maximum speed as 7.2 miles per hour (3.22m/s). Traction is provided by dual 4.8kW motors. The unloaded weight of the vehicle is 3059kg and the battery 1043kg for a total of 4201kg [4].

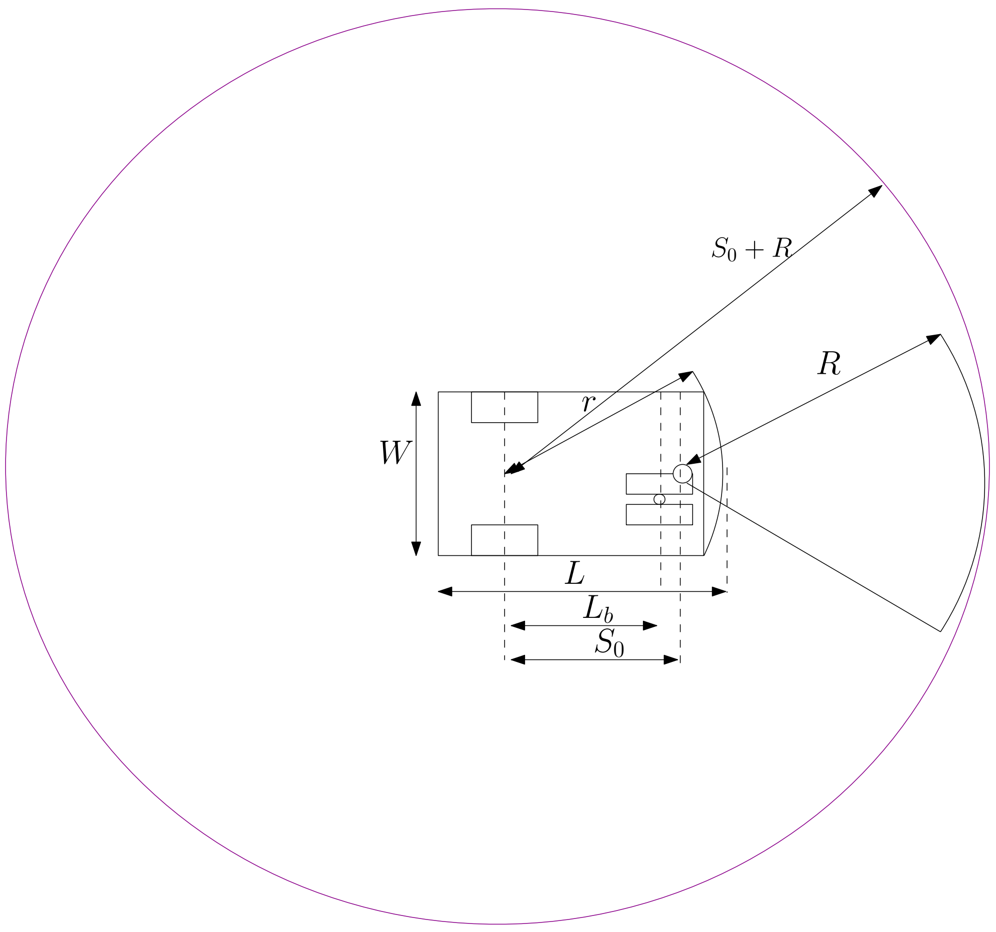


Figure 3.8: Dimensions used to expand the obstacles

For correct operation it is important to consider the exclusion zone of the safety rated range sensor fitted to the front of the vehicle. If an obstacle breaches the exclusion zone the AGV must perform an emergency stop, or in some cases slow down significantly. To avoid slowing down the path planning must account for not only the shape of the vehicle but also the shape of this zone. Often this is a cuboid slightly wider than the vehicle, sufficiently long that the AGV can come to a complete stop from full speed before the front makes contact with a static obstacle. More details are available in the NIST Safety Standards [139].

In the two simulated environments the obstacle field is represented in 2D. The bounding circle dimension is strongly influenced by the stopping distance R. Starting from the constant acceleration equation $v^2 = u^2 + 2as$ and setting $v = 0$ gives the stopping distance $s = \frac{u^2}{-2a}$.

3.3.7 Problem Definition: Clothoid Fitting to Convex Regions

The core problem is G^2 continuous path planning around obstacles for a point robot. Using a convex spanning region approach described in Section 3.4.1 this can be divided into a small number of sub-problems. This paper addresses the 'Curve Fitting' sub-problem in detail using the path representation set out in Section 3.3.2. The approach is described in Section 3.4.

The requirement for G^2 (curvature) continuity provides four constraints on each segment similar to the posture interpolation problem [127]. In each region one path piece consisting of a line segment length $s_{0,i}$, a matched pair of clothoid segments defined by $\alpha_{1,i}, \alpha_{2,i}, L_{1,i}, L_{2,i}$ and the other line segment length $s_{F,i}$, as explained in Section 3.3.2. This provides a total of six free parameters per path piece. As there are only four constraints per region the problem is under-determined and the length of the straight lines can be balanced against the length of the clothoids to find the ideal combination. Considering the start and end region to be separate even if they overlap entirely ensures there will be a minimum of four clothoid segments across the whole spline and s-shapes are feasible.

3.4 Method: Clothoid Fitting to Convex Regions

The core routine is a method to find the parameters of a sequence of G^2-continuous (continuous in curvature) clothoid segments which is contained entirely within a chain of obstacle-free regions. The path is divided into a sequence of N path pieces assigned to N convex regions. The objective function is designed to trade off smoothness with path length such that the best path is one that minimizes the weighted squared sum of sharpness and path length to reach the x, y, ψ and κ of the reference path. This is captured in Equation 3.7 below:

$$\begin{gathered}\min_{\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F}} J_T = b\cdot\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{L}^T\boldsymbol{L} + \boldsymbol{s_0}^T\boldsymbol{s_0} + \boldsymbol{s_F}^T\boldsymbol{s_F} \\ \text{such that}\begin{cases}\boldsymbol{c_{eq}}(\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F}) = \boldsymbol{0} \\ \boldsymbol{c}(\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F}) \le \boldsymbol{0}\end{cases} \\ \text{with bounds } \boldsymbol{L}\ge\boldsymbol{0}, \boldsymbol{s_0}\ge\boldsymbol{0}, \boldsymbol{s_F}\ge\boldsymbol{0}\end{gathered} \tag{3.7}$$

Bold variables indicate vectors of the parameters defined in Section 3.3.1 for every path piece. Scaling parameter b is discussed in Section 3.4.2. The equality constraints arise from enforcing continuity between each piece and of the first and last segments with the origin and destination respectively. The inequality constraints relate to the requirement to remain outside every obstacle. The first derivatives of the objective and constraints are given in Appendix B.

3.4.1 Existence of Paths and Completeness

The existence of a sequence of adjacent or overlapping obstacle-free regions from the start to the goal position is a necessary and sufficient condition for the existence of a path with G^0 continuity (position). This type of path is suitable for a holonomic robot which can move any direction with unlimited velocity and acceleration. If the obstacle-free regions are constructed in configuration space, where the obstacles have been expanded by the largest dimension of the robot body, any real robot can modelled as a point.

A differential drive configuration robot can follow a path with G^0-continuity exactly by moderating its longitudinal velocity.

A tricycle configuration robot is unable to exactly follow a path with any discontinuity in heading. Furthermore it has a fixed maximum radius of curvature $\bar{\kappa}$ which restricts the paths which can be followed exactly to a subset of those with G^1-continuity.

Both types of robot are only able to apply a finite amount of rotational torque, limiting the longitudinal speed at which they can exactly traverse a G^2-continuous path with a given curvature rate.

In the presented algorithm, a maximum curvature constraint is not applied, making it most applicable to differential-drive robots. A solution to the optimization is certain to exist if a set of connected regions containing the start and the goal can be found in step 2 ‘Assignment’ of Section . Paths with lower sharpness are advantageous to this type of robot because they can be traversed faster without risk off tipping over or exceeding the rotational torque which can be supplied by the motors and tyre contact patches. The dynamics models used for this reasoning are presented in Appendix D.

3.4.2 Objective Function

Two important objectives for an alternative path for an AGV are the total length and the peak sharpness. Both of these properties are desirable rather than mission critical so they are useful in resolving the many solutions which are available to reach a given pose smoothly. Other performance measures such as reaching the destination exactly, limiting the peak curvature and avoiding the obstacles are better interpreted as hard constraints, as it is not useful to compromise them in any way for the measures which are only desirable.

The degree to which smoothness is important compared to path length may depend on the application so it is left with a scaling parameter b. The units of b are m^2 per $\text{radian}^2/\text{m}^4 = \text{m}^6 \cdot \text{radian}^{-2}$. If b is set very high, smoother paths can be attained, at the cost of increasing path length. In general there is a trade off between path length and smoothness (measured by the least maximum rate of change of curvature as identified by [60]).

Results will be reported with three alternative parameters settings for b, with and without the straight lines, as described below.

- Equal Weighting with Lines, $b = 1$, $\min\limits_{\alpha_i, L_i, s_0, s_F} J_T = \sum\limits^{N} L_i^2 + \sum\limits^{N} \alpha_i^2$
- Equal Weighting, no Lines, $b = 1$, $s_0 = 0$, $s_F = 0$, $\min\limits_{\alpha_i, L_i} J_T = \sum\limits^{N} L_i^2 + \sum\limits^{N} \alpha_i^2$
- Minimum Sharpness, no Lines, $b = \inf$, $s_0 = 0$, $s_F = 0$, $\min\limits_{\alpha_i} J_T = \sum\limits^{N} \alpha_i^2$

3.4.3 Equality Constraints for Region i

For continuity, the spline must reach the goal pose and curvature given by $[\hat{x}, \hat{y}, \hat{\psi}, 0]^T$. A straight line s_0 before and s_F after the clothoid pair is included while maintaining curvature continuity by the constraint $\kappa = 0$ which forces the clothoid pair to be matched, smoothly returning the curvature to zero at the end.

For multiple regions, the continuity constraints are applied implicitly by integrating each segment starting from the final pose of the last. As the curvature is zero at the end of each pair, this can be done by integrating pairwise from the origin as in Equation 3.8 followed by a rotation and translation to the final pose of the last segment using the $\oplus$ operator detailed in Appendix C. This is an example of single shooting trajectory optimization as described in [140]. Each region gives rise to one additional $\kappa = 0$ constraint, to ensure the straight lines $s_{0,i}$ and $s_{F,i}$ can be added with curvature continuity.

$$\left[\begin{array}{c|r} x_i & \begin{array}{r} s_{0,i} + C(\alpha_{1,i}, L_{1,i}, 0) \\ + \cos(\delta_{1,i}) \cdot C(\alpha_{2,i}, L_{2,i}, k_{m,i}) \\ - \sin(\delta_{1,i}) \cdot S(\alpha_{2,i}, L_{2,i}, k_{m,i}) \\ + s_{F,i} \cdot \cos\hat{\psi}_i \end{array} \\ \hline y_i & \begin{array}{r} \sin(\delta_{1,i}) \cdot C(\alpha_{2,i}, L_{2,i}, k_{m,i}) \\ + \cos(\delta_{1,i}) \cdot S(\alpha_{2,i}, L_{2,i}, k_{m,i}) \\ + s_{F,i} \cdot \sin\hat{\psi}_i \end{array} \\ \hline \psi_i & \alpha_{1,i} L_{1,i}^2/2 + \alpha_{2,i} L_{2,i}^2/2 \\ \hline \kappa_i & \alpha_{1,i} L_{1,i} + \alpha_{2,i} L_{2,i} \end{array}\right] \tag{3.8}$$

The path in local coordinates for one piece, given by Equation 3.8, can be composed using the $\oplus$ operator to find the final position in global coordinates.

$$[X_i, Y_i, \Psi_i]^T = [X_{i-1}, Y_{i-1}, \Psi_{i-1}]^T \oplus [x_i, y_i, \psi_i]^T, \ \forall\, i \ \in [1, N]$$

where capital letters indicate global coordinates. The first region is given the index $i = 1$. $[X_0, Y_0, \Psi_0]$ indicates the starting position. This is taken to be the origin of the coordinate system.

3.4.4 Equality Constraints Vector

Subtracting the goal pose from the Nth pose in global coordinates gives the equality constraints, where N is the total number of convex regions under consideration.

$$\boldsymbol{c_{eq}} = \begin{bmatrix} X_N - \hat{X} \\ Y_N - \hat{Y} \\ \Psi_N - \hat{\Psi} \\ \kappa_1 - 0 \\ \vdots \\ \kappa_N - 0 \end{bmatrix} = \mathbf{0} \tag{3.9}$$

Note that the $\boldsymbol{c_{eq}}$ array is length $N + 3$ as there is a constraint on curvature at the end of each clothoid pair κ_i. The other quantities are scalar and refer to the pose of the final segment at the end of the spline.

3.4.5 Inequality Constraints

In order to operate on the obstacle field directly and avoid spatial sampling, inequality constraints are used to fix the segments inside their assigned regions. The assignment of curve sections to regions will not be discussed here, but can be approached as an integer problem to find the boolean assignment matrix $\boldsymbol{H}^{(N\times R)}$ which results in the lowest cost solution. R is the number of regions in the map, and N is the number of pieces of the spline between the start and the goal. A simple region shape for exposition is an axis aligned rectangle extending from X_{min} to X_{max} and from Y_{min} to Y_{max}. This leads to the following eight inequality constraints on every region, ensuring that both the start and end of each piece are contained. Based on the assignment $\boldsymbol{H}(i,j)$, indicating path piece i must be entirely contained in region subscript j. The constraints for one region are given by

$$\boldsymbol{H}(i,j)=1 \iff \boldsymbol{d_i}=\begin{bmatrix} X_i - X_{max,j} \\ -X_i + X_{min,j} \\ Y_i - Y_{max,j} \\ -Y_i + Y_{min,j} \\ X_{i-1} - X_{max,j} \\ -X_{i-1} + X_{min,j} \\ Y_{i-1} - Y_{max,j} \\ -Y_{i-1} + Y_{min,j} \end{bmatrix} \leq \mathbf{0} \tag{3.10}$$

These inequality constraints ensure the start and end of the curve piece i assigned to region j remain inside the region. The position at the start of curve i is identical to the position at the end of curve $i-1$ by the application of the continuity constraints. For the first curve the start is fixed at the origin by the choice of coordinate frame. The inequality confirms that the initial position must be in a region of free space for a solution to exist.

The constraints for the entire problem can be constructed by stacking $\mathbf{d_i}$ for each region into a partition vector as follows

$$\boldsymbol{c_{ineq}}=\begin{bmatrix} \boldsymbol{d_1} \\ \vdots \\ \boldsymbol{d_N} \end{bmatrix} \leq \mathbf{0} \tag{3.11}$$

Constraint Equation 3.10 only applies to the start and end of each clothoid pair, not the entire curve. This works quite well when the curvature remains low but will cause a problem for certain combinations of region shape and path curvature as discussed in Section 3.6.2.

3.4.6 Multiple Shooting Formulation

Alternatively the problem can be posed as a multiple shooting trajectory optimization by the terminology of [140]. The suggestion is that although there are more parameters and more constraints on the multiple-shooting problem, it may counter-intuitively be easier for the solver because each parameter-constraint pair is more independent and closer to linear. This involves extra parameters $X_{0,i}$, $Y_{0,i}$, $\Psi_{0,i}$ which provide a pose offset for each clothoid pair and explicit continuity constraints between each clothoid pair and the last. The problem would then be described as

$$\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F},\boldsymbol{X_0},\boldsymbol{Y_0},\boldsymbol{\Psi_0}} \quad & J_T = b\cdot\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{L}^T\boldsymbol{L} + \boldsymbol{s_0}^T\boldsymbol{s_0} + \boldsymbol{s_F}^T\boldsymbol{s_F} \\
\text{subject to} \quad & \begin{cases} \tilde{\boldsymbol{c}}_{\boldsymbol{eq}}(\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F},\boldsymbol{X_0},\boldsymbol{Y_0},\boldsymbol{\Psi_0}) = \boldsymbol{0} \\ \tilde{\boldsymbol{c}}_{\boldsymbol{ineq}}(\boldsymbol{\alpha},\boldsymbol{L},\boldsymbol{s_0},\boldsymbol{s_F},\boldsymbol{X_0},\boldsymbol{Y_0},\boldsymbol{\Psi_0}) \le \boldsymbol{0} \end{cases} \\
& \text{with bounds } \boldsymbol{L}\ge\boldsymbol{0}, \boldsymbol{s_0}\ge\boldsymbol{0}, \boldsymbol{s_F}\ge\boldsymbol{0}
\end{aligned} \tag{3.12}$$

Now the coordinates are calculated slightly differently, using the new offset parameters rather than a recurrence relation.

$$[X_i, Y_i, \Psi_i]^T = [X_{0,i-1}, Y_{0,i-1}, \Psi_{0,i-1}]^T \oplus [x_i, y_i, \psi_i]^T,\ \forall\, i\ \in [1, N]$$

where capital letters indicate global coordinates.

G^2 Continuity is ensured between pair i and pair $i-1$ in the adjacent region by enforcing constraint $\tilde{\boldsymbol{q}}_{\boldsymbol{i}} = \boldsymbol{0}$. The curvature is fixed to zero between each pair.

$$\tilde{\boldsymbol{q}}_{\boldsymbol{i}} = \begin{bmatrix} X_{0,i} - X_{i-1} \\ Y_{0,i} - Y_{i-1} \\ \Psi_{0,i} - \Psi_{i-1} \\ \kappa_{0,i} - 0 \end{bmatrix} = \boldsymbol{0} \tag{3.13}$$

There is also a constraint on the goal pose similar to the single shooting form, which must be included in the stack.

$$\tilde{\boldsymbol{q}}_{\boldsymbol{goal}} = \begin{bmatrix} X_N - \hat{X} \\ Y_N - \hat{Y} \\ \Psi_N - \hat{\Psi} \\ \kappa_N - 0 \end{bmatrix} = \boldsymbol{0} \tag{3.14}$$

Here the first subscript of $X_{0,i}$ indicates it is the start pose for the region given by the second subscript. The start poses for each region are new search parameters for this formulation. For symbols with a single subscript, the subscript identifies the region so X_i refers to the end of the curve assigned to that region. Again the first region is given the index $i = 1$. The start pose is $[X_0, Y_0, \Psi_0]$. This is taken to be the origin of the coordinate system.

The inequality constraints for the entire problem can be constructed by stacking $\tilde{\boldsymbol{q}}_{\boldsymbol{i}}$ for each region into a partition vector as follows

$$\tilde{\boldsymbol{c}}_{\boldsymbol{ineq}} = \begin{bmatrix} \tilde{\boldsymbol{q}}_{\boldsymbol{1}} \\ \vdots \\ \tilde{\boldsymbol{q}}_{\boldsymbol{N}} \\ \tilde{\boldsymbol{q}}_{\boldsymbol{goal}} \end{bmatrix} = \boldsymbol{0} \tag{3.15}$$

The inequality constraints for region i are given by

$$\boldsymbol{H}(i,j) = 1 \iff \tilde{\boldsymbol{d}}_{\boldsymbol{i}} = \begin{bmatrix} X_i - X_{max,j} \\ -X_i + X_{min,j} \\ Y_i - Y_{max,j} \\ -Y_i + Y_{min,j} \\ X_{i-1} - X_{max,j} \\ -X_{i-1} + X_{min,j} \\ Y_{i-1} - Y_{max,j} \\ -Y_{i-1} + Y_{min,j} \end{bmatrix} \leq \boldsymbol{0} \tag{3.16}$$

The constraints for the entire problem can be constructed by stacking $\tilde{d}_i$ for each region into a partition vector as follows

$$\tilde{\boldsymbol{c}}_{\boldsymbol{ineq}} = \begin{bmatrix} \tilde{\boldsymbol{d}}_{\boldsymbol{1}} \\ \vdots \\ \tilde{\boldsymbol{d}}_{\boldsymbol{N}} \end{bmatrix} \leq \boldsymbol{0} \tag{3.17}$$

3.5 Polynomial Method

To understand the advantages of using clothoid pieces to join two poses in a 2D obstacle field, it is informative to compare the output paths with an alternative curve type. One well known option is to use polynomial curves of degree 3.

The paths are compared on two simulated environments. The first, shown in Figure 3.6 is dimensioned for a small AGV with turning radius 0.5m, as might be used to deliver small items in a flexible manufacturing environment. The unexpected obstacle completely blocks the path to the right. The second is dimensioned for a fork lift type AGV with a larger turning radius of 2m, which is collecting standard size (1m x1.2m) pallets from a storage area. See Figure 3.7. The pallets may be placed by human operated vehicles so a reliable system for detecting their position and orientation is assumed to be in place. The AGV must manoeuvre to the pose of a target pallet, without colliding with the others. Paths with lower peak curvature κ and peak sharpness $\dot{\kappa}$ allow the AGV to traverse the path faster without compromising load stability. For more details on the relationship between longitudinal speed and path derivatives see Appendix D.

Both methods decompose the problem into topology followed by curve fitting. The topology problem is posed as a directed graph with weighted edges. There is a node for every intersection of one region's boundary with another. Nodes within the same region are fully connected. The weight of each edge corresponds to the euclidean distance between the two nodes. The A* Algorithm is used to search for the set of edges which give the minimum sum of weights between any start and end pose.

The sequence of edges is then used to populate the matrix $\boldsymbol{H}^{(R\times P)} \in [0,1]$. This is a binary matrix containing R columns, one for each of the R polygonal regions which comprise the accessible space. Each row corresponds to one of the P path pieces and contains a single non-zero element indicating the region to which is assigned. A path piece may be present in more than one region as the regions may be overlapping, but it must always remain completely inside its assigned region.

The problem specification calls for a path which changes smoothly in x, y, heading and curvature. This should start at a specified x_s, y_s, ψ_s with zero curvature and end at x_g, y_g, ψ_g with zero curvature.

$$\begin{aligned} x(t) &= a + bt + ct^2 + dt^3 \\ y(t) &= e + ft + gt^2 + ht^3 \end{aligned} \tag{3.18}$$

There is a unique solution for a cubic spline with fixed (x, y, heading) at the start and goal, passing through fixed x, y positions numbering n. A cubic spline defined by Equation 3.18 has eight free parameters per segment. To give a unique solution, eight constraints must be found for each segment. Passing through the n waypoints at the end of each segment gives two, one for the x coordinate and one for y. Enforcing continuity of position between the end of each segment and the next leads to two more. Four more can be determined from continuity in the first and second derivative of position for a total of eight.

However, only four constraints are needed at the start and end of the spline. Fixing the final position and heading, the acceleration must be left free. Stacking the parameters into a vector

$$\boldsymbol{p}_i = [a_i, b_i, c_i, d_i, e_i, f_i, g_i]^T \tag{3.19}$$

and

$$\boldsymbol{p} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_i, \cdots, \boldsymbol{p}_{n-1}]^T \tag{3.20}$$

leads to the system of linear equations is given in Equation 3.21.

$$[\boldsymbol{A}|\boldsymbol{b}_x] = \left[\begin{array}{cccccc|c} \boldsymbol{A}_0 & & & & \cdots & 0 & \boldsymbol{b}_{x0} \\ \boldsymbol{0} & \boldsymbol{A}_1 & & & \cdots & 0 & \boldsymbol{b}_{x1} \\ \vdots & & \ddots & & & \vdots & \vdots \\ 0 & & \cdots & \boldsymbol{A}_i & \cdots & 0 & \boldsymbol{b}_{xi} \\ \vdots & & & & \ddots & & \vdots \\ 0 & \cdots & & & & \boldsymbol{A}_n & \boldsymbol{b}_{xn} \end{array}\right] \tag{3.21}$$

Where

$$[\boldsymbol{A}_0|\boldsymbol{b}_{x0}] = \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & x_0 \\ 0 & 1 & 0 & 0 & \cos\psi_0 \end{array}\right] \tag{3.22}$$

$$[\boldsymbol{A}_i|\boldsymbol{b}_{xi}] = \left[\begin{array}{cccccccc|c} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & x_i \\ 1 & 1 & 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 3 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 6 & 0 & 0 & -2 & 0 & 0 \end{array}\right] \tag{3.23}$$

$$[\boldsymbol{A}_n|\boldsymbol{b}_{xn}] = \left[\begin{array}{cccccccc|c} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & x_n \\ 0 & 1 & 0 & 0 & 0 & 1 & 2 & 3 & \cos\psi_n \end{array}\right] \tag{3.24}$$

The $\boldsymbol{p}_x$ parameters to fit a set of n waypoints can be found by computing $\boldsymbol{p}_x = \boldsymbol{A}^{-1}\boldsymbol{b}_x$. The $\boldsymbol{p}_y$ parameters can be found almost identically, by computing $\boldsymbol{p}_y = \boldsymbol{A}^{-1}\boldsymbol{b}_y$. Constraint vector $\boldsymbol{b}_y$ is instead constructed using the y coordinates of the waypoints in b_i as in Equation 3.26 and the $\sin\psi$ component of the start and end heading in $\boldsymbol{b}_{y0}$ and $\boldsymbol{b}_{yn}$ as shown in Equation 3.25 and Equation 3.27.

$$\boldsymbol{b}_{y0} = \begin{bmatrix} y_0 \\ \sin\psi_0 \end{bmatrix} \tag{3.25}$$

$$\boldsymbol{b}_{yi} = \begin{bmatrix} y_i \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.26}$$

$$\boldsymbol{b}_{yn} = \begin{bmatrix} y_n \\ \sin\psi_n \end{bmatrix} \tag{3.27}$$

3.5.1 Optimization

The point-to-point solution through the region intersection points is quite a good solution. Continuous in $\dot{X}$ and $\ddot{X}$, it reaches the goal position and heading. However, objectives of path length and smoothness will not be minimized because it is excessively constrained by the requirement to meet the intermediate waypoints. This can be improved by relaxing the equality constraint of meeting the waypoints and searching over the entire region. Using the optimization in Equation 3.28

$$\begin{gathered} \min \textstyle\sum_{j=1}^{N} ||\frac{dP_j^3(t)}{dt^3}||^2 \\ \text{subject to} \\ P_j(1) = P_{j+1}(0) \\ \dot{P}_j(1) = \dot{P}_j(0) \\ \ddot{P}_j(1) = \ddot{P}_j(0) \\ P_1(0) = X_S \\ P_N(1) = X_G \\ \dot{P}_1(0) = \dot{X}_S \\ \dot{P}_N(1) = \dot{X}_G \\ \text{also subject to} \\ \boldsymbol{H}_{j,r} \Rightarrow X_r^{min} \leq P_j(t) \leq X_r^{max} \end{gathered} \tag{3.28}$$

The implementation of the region constraint in Equation 3.28 must ensure the entire arc remains inside the assigned region according to $\boldsymbol{H}$. For piecewise linear arcs this can be done by checking the containment of the start $P_0(0)$ and end $P_N(1)$. For cubic splines it is more challenging. Deits and Tedrake [30] describe an approach based on

Sum of Squares, where a small Semidefinite Program is solved in the parameters of P_j, to avoid sampling at different t values, which can lead to the path cutting corners and even passing through thin obstacles.

First notice that the constraints on one segment from its axis aligned containing region described by $x_{min}, x_{max}, y_{min}, y_{max}$ can be written as vector inequality.

$$q(t) = \begin{bmatrix} x_{max} - a - bt - ct^2 - dt^3 \\ a + bt + ct^2 + dt^3 - x_{min} \\ y_{max} - e - ft - gt^2 - ht^3 \\ e + ft + gt^2 + ht^3 - y_{min} \end{bmatrix} \geq \mathbf{0} \forall t \in [0,1] \tag{3.29}$$

The condition in Equation 3.29 can only hold if and only if it can be rewritten in the form

$$t\sigma_1(t) + (1-t)\sigma_2(t) \tag{3.30}$$

This leads to

$$\sigma_1(t) = \begin{bmatrix} x_{max} - a - b - ct - dt^2 \\ a + b + ct + dt^2 - x_{min} \\ y_{max} - e - f - gt - ht^2 \\ e + f + gt + ht^2 + -y_{min} \end{bmatrix} \tag{3.31}$$

and

$$\sigma_2 = \begin{bmatrix} x_{max} - a \\ a - x_{min} \\ y_{max} - e \\ e - y_{min} \end{bmatrix} \tag{3.32}$$

The standard Sum of Squares approach [141] calls for collecting the parameters of $\sigma(t)$ by the order of t. Matching coefficients against

$$\sigma_i = \beta_1 + \beta_2 t + \beta_3 t^2 \tag{3.33}$$

gives

$$\beta_1 = \begin{bmatrix} x_{max} - a - b \\ a + b - x_{min} \\ y_{max} - e - f \\ e + f + y_{min} \\ x_{max} - a \\ a - x_{min} \\ y_{max} - e \\ e + y_{min} \end{bmatrix} \tag{3.34}$$

and

$$\beta_2 = \begin{bmatrix} -c \\ +c \\ -c \\ +c \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.35}$$

and

$$\beta_3 = \begin{bmatrix} -d \\ +d \\ -d \\ +d \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.36}$$

The parameters for σ_2 have been stacked after those from σ_1.

In order for σ_i to be a sum of squares, there are the following conditions on β_1, β_2, β_3:

$$\begin{gathered} 4\beta_1\beta_3 - \beta_2^2 \geq 0 \\ \beta_1, \beta_3 \geq 0 \end{gathered} \tag{3.37}$$

In the simple case where the regions are axis aligned, this immediately creates a problem as the first two equations have $\beta_3 = d$ and $\beta_3 = -d$. The only value of d which will satisfy the sum of squares condition is zero. However the initial guess which joins the corner points, satisfies the region occupancy constraints with a non-zero d.

For this reason the constraints are enforced using 50 samples for each path piece. This leads to a small degree of corner cutting which must be included in the safety factor used on the vehicle body width used to inflate the obstacle. It also results in a large number of constraints as there are four for each sample. A fixed sampling length is used rather than a fixed number of samples per piece as path pieces can vary in length substantially changing the degree of corner cutting.

3.6 Numerical Results

First, in Section 3.6.1, the suitability for finding the smoothest path subject to obstacle constraints is tested on an environment similar to Figure 3.6. Subsequently, an alternative formulation, the effect of analytical gradients, tuning parameter b and the objective function are evaluated for their potential in speeding up the solution. A number of tests without obstacles are included in Section 3.6.5 and 3.6.6 so path parameters can be compared with an existing heuristic method.

3.6.1 Motivating Problem

The tested problem concerns the avoidance of a small obstacle blocking a straight path while remaining within a set tolerance from it. The allowable distance from the original path is used to generate the outer boundary enclosing [0, 5] and [11, -5]. With the obstacle information available from sensors this is broken up into three convex regions. As set out in Section 3.3.2 this necessitates a path with six clothoid segments, two at the start, two at the goal and two at an intermediate region needed to connect the two.

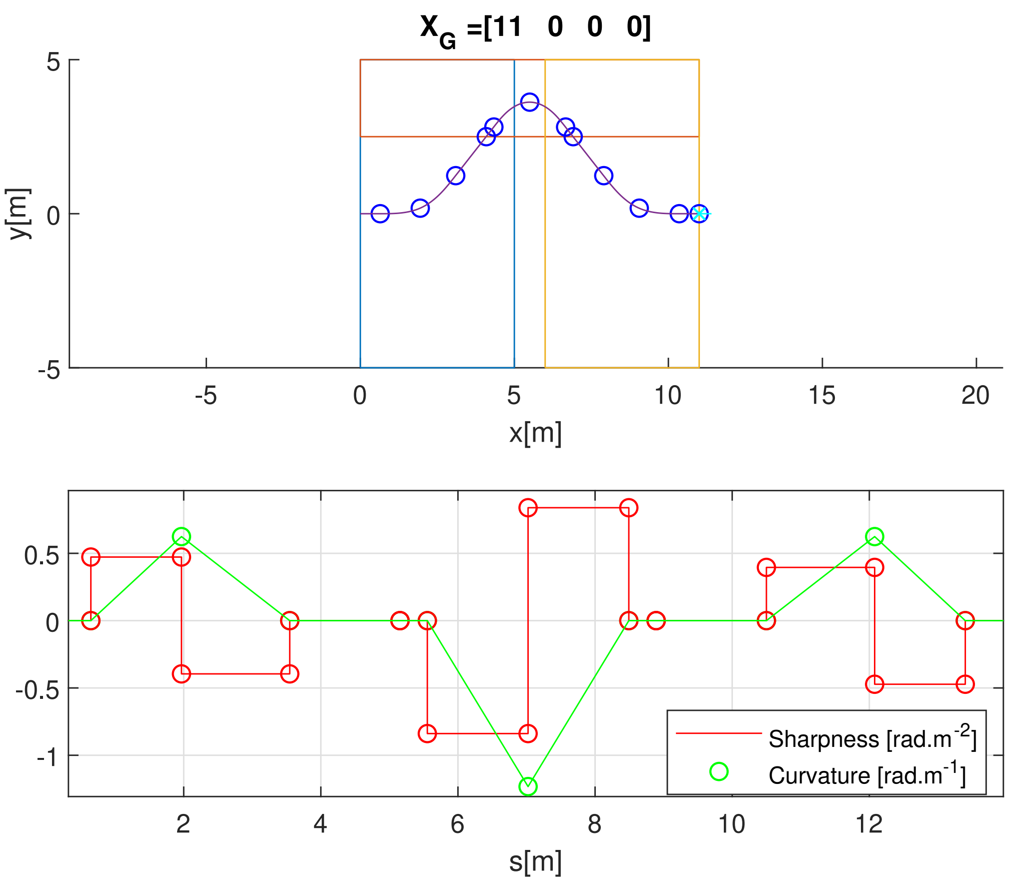


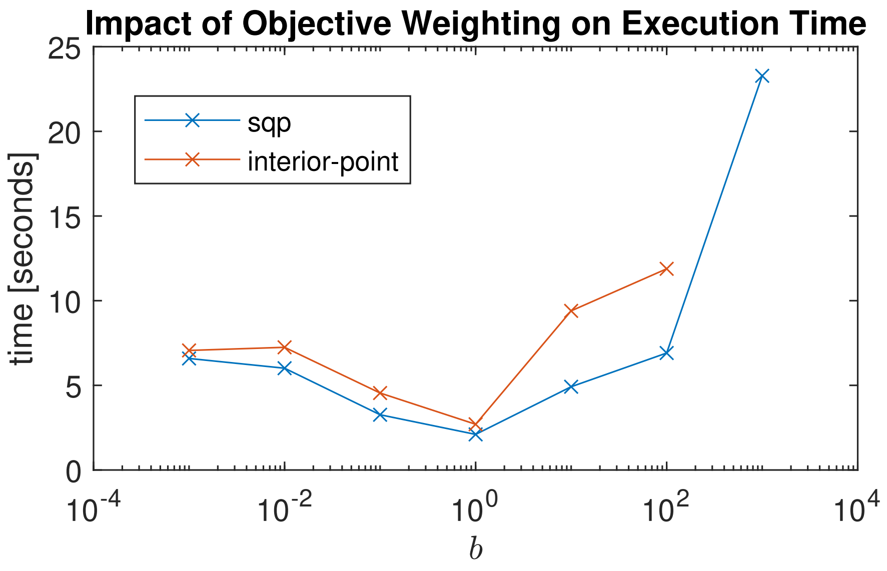
Figure 3.9: Single shooting optimal avoidance path for a vehicle traveling along the x-axis which encounters a rectangular obstacle at [5,-2.5], extending to [6,2.5], rejoining the reference path at [11, 0]. Weighting parameter $b = 1$

The single shooting formulation was solved using the *interior-point*[1]method in 36 iterations to produce the path and curvature profile shown in Figure 3.9. The region constraints are satisfied. This can be seen as each of the four segments is marked with an open circle. The same points are marked with open circles in the curvature and sharpness profile shown below the x-y plot in Figure 3.9. These can be used to evaluate the quality of the path without reference to a specific vehicle model. The largest magnitude curvature is found at the midpoint with a value of -1.233 [m^{-1}], corresponding to a turning radius of 811mm, suitable for a small vehicle with Ackermann steering, having a wheelbase less than 811mm. The largest magnitude sharpness of 0.8388 [radm^{-2}] is seen on the segments before and after the peak curvature. This path can be tracked with high accuracy by different AGVs by reducing forward speed to control the lateral acceleration based on the path curvature, and the angular acceleration based on the sharpness.

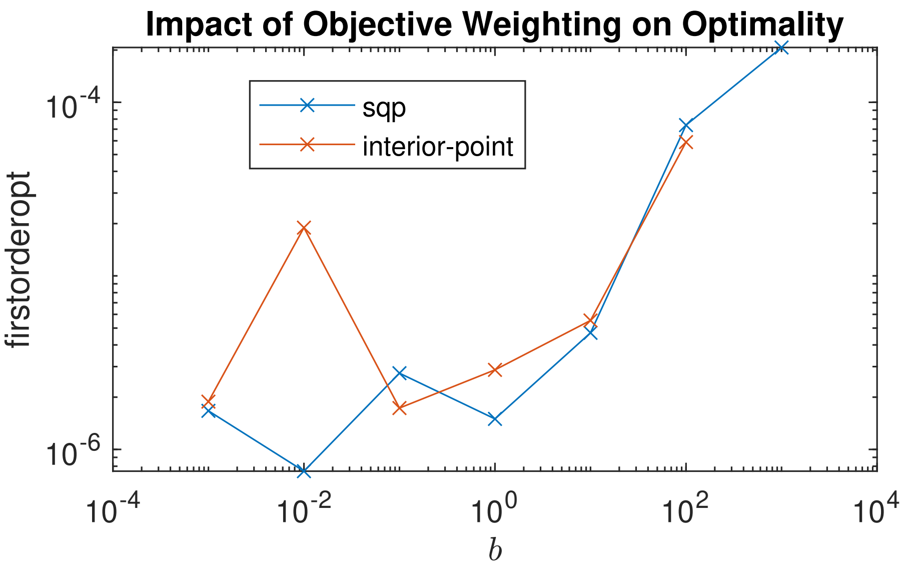
The single shooting formulation reduces the number of parameters which might be expected to reduce total execution time. In fact, it took multiple seconds to reach convergence on the small 20m×10m environment made up of the three regions tested.

3.6.2 Weighting Parameter b Effects

A weighting parameter b was included in the objective function in Section 3.4.2 to allow the trade-off between minimum path length and minimum peak sharpness to be adjusted for different applications. The weighting parameter b has a significant effect on convergence time as shown in Figure 3.10a. The number of iterations and function evaluations increased for larger weights, in the same way as time to convergence.

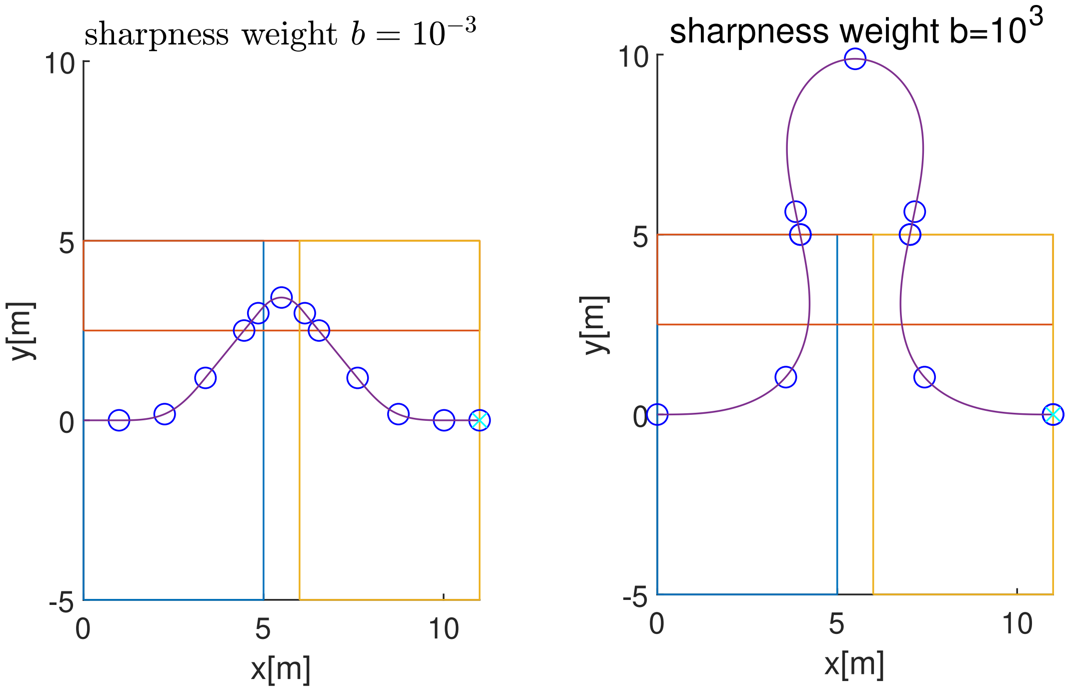


(a) Time to Convergence



(b) First Order Optimality

Figure 3.10: Variation as weighting b between sharpness and length is varied



(a) $b = 10^{-3}$ (b) $b = 10^{3}$

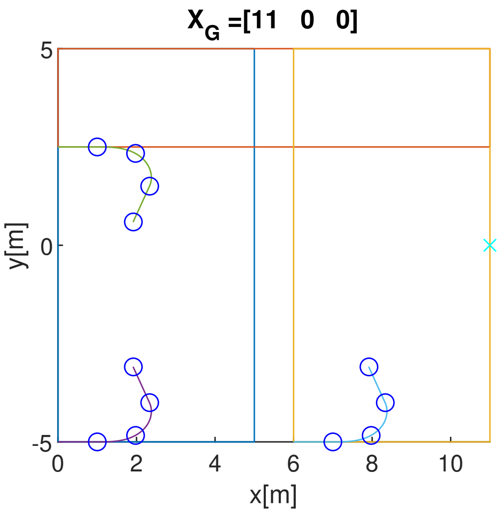
Figure 3.11: Path comparison for extremes of b

The two path plots in Figure 3.11 cover both extremes of weighting tested for the avoidance problem. Figure 3.11a shows the effect of a small weighting on sharpness. The path is shorter, the peak sharpness is higher, but it is not the shortest continuous path which would pass through the corner formed by region one and two (the top one). The segments are close to equal length because, for a set of numbers with a fixed sum, the sum of squares will be minimized if the numbers are equal. This provides a bias toward equal length segments and may contribute to the success of an equal weighting of the two components.

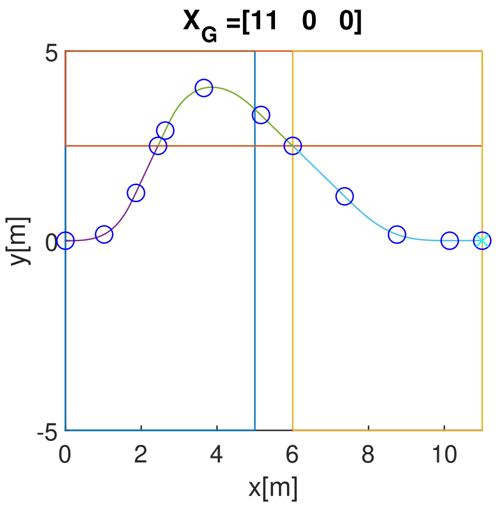
Figure 3.11b shows the effect of a large weighting on sharpness. The path is longer and more meandering but has lower peak sharpness. The constraints are met as the start and end are contained but the curve leaves the convex region at the top. Because the region constraint is not applied to the point of peak curvature, there are a large number of feasible solutions with similar sharpness. This makes an objective based on sharpness ($b >> 1$) very flat close to the minimum with this set of obstacles. That is to say that the objective changes very little over a wide range of parameters. Looking through the text output of *fmincon*, a feasible solution according to the threshold of 1e-6 is found earlier but the search continues until the gradient threshold is reached. This corresponds to searching for slightly less sharp curves which extend outside of the boundaries while the start and end meet the constraints. To solve this problem, the region constraints need to be applied to additional samples along the path.

3.6.3 Obstacle Avoidance Multiple Shooting Formulation

In the multiple shooting formulation by contrast, the curve positions in one region are independent of the parameters of the earlier segments due to the introduction of a new $[X_0, Y_0, \Psi_0]$ offset parameter for each region and a new constraint that this offset matches the final pose of the curve in the preceding region.



(a) Initial parameter guess



(b) After convergence

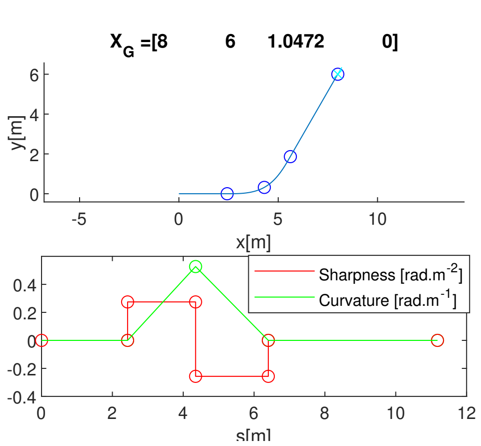
Figure 3.12: Multiple shooting initial guess ($\alpha_i = 1.0$, $L_i = 1.0$) showing path continuity constraints are not met until the optimization has converged

With the initial guess shown in Figure 3.12a the solver must find offsets which satisfy both continuity and region containment. This can give better performance than single shooting in some cases involving longer splines [140]. In the problem shown in Figure

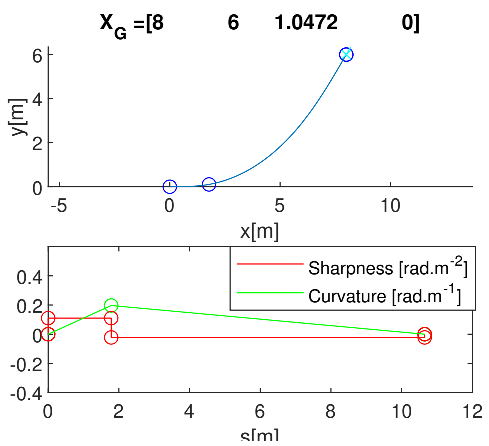
3.12b with three regions the discontinuities have been resolved after convergence; the execution time is comparable with single shooting but slightly improved, with around 700 function calls using the interior point method.

The execution time is improved despite increasing the number of parameters from six per region to nine per region (18 to 27 with $N = 3$), and the number of constraints from $4 + N$ equality and $8N$ inequality (7 and 24 with $N = 3$) to $4 + 4N$ equality and $8N$ inequality (16 and 24 with $N = 3$). The number of inequality constraints could be reduced slightly for the multiple shooting method by fixing the start position at the world origin, to match the features of the single shooting setup.

3.6.4 Curve fitting With Two Clothoids With Different Objectives



(a) Equal Weighting with Lines



(b) Minimum Sharpness, no Lines

Figure 3.13: Side by side comparison of a path from the origin to [8,6, 60], curvature profile is shown below.

Table 3.2: Parameters identified for a path from [0,0,0] to [8,6,60]. The objectives are explained in Section 3.4.2

	Equal Weighting, with Lines	Minimum Sharpness, no Lines
α_1 [m^{-2}]	0.1472	0.1094
α_2 [m^{-2}]	-0.1135	-0.0222
L_1 [m]	2.4893	1.7981
L_2[m]	3.2281	8.8535
s_0 [m]	1.5966	0
s_F [m]	3.7390	0
Total Length [m]	11.0529	10.6516
κ_m [m^{-1}]	0.3633	0.1966
interior-point iterations	12	2000+
interior-point funcCount	101	16220+
sqp iterations	15	12
sqp funcCount	130	89

Cartesian path and curvature profile results for a single region containing two clothoid segments and two straight lines are shown in Figure 3.13a for comparison to those reported by [59] for two matched clothoids. For illustration the parameters identified for the same final pose by our method with two different cost functions are shown in Table 3.2.

There are two remarks to make here. Firstly, the [8,6,60] point meets the condition in Equation 3.6, Section 3.3.2 and can be reached exactly with two clothoids and no line segments. Many nearby points require a line to be included at the end of the clothoid pair for convergence, particularly those with lower final angles (closer to the limit given by Equation 3.5). Secondly, the initial guess had to be very close to the minimum in order to reach convergence at all with *interior-point* method for this problem. Using the 'Equal Weighting with Lines' objective an initial guess of [1, -1, 1, 1, 1, 1] converged in 12 iterations with *interior-point* method. With the same initialisation vector and the 'Minimum Sharpness, no Lines' objective, *interior-point* did not converge within 2000 iterations. If the solver was changed to *sqp* convergence took 12 iterations and 89 function evaluations. The *interior-point* method is shown to be slightly faster on some problems but less stable than *sqp* on the limited numerical tests performed. These tests only involved three regions, leading to 18 parameters and 16 constraints. It is a notable advantage of the convex region representation that the number of regions can be strictly limited even in environments with lots of clutter. The documentation lists *sqp* as a medium-scale algorithm, which needs to store and operate on matrices with the dimension of the parameters [142]. If problems are encountered in larger tests, *interior-point* is a large scale-algorithm which does not rely on dense matrix operations, and should perform better. Due to the general constrained form of the problem other highly optimized methods for specific performance needs could be used depending on the needs of the application (e.g. limited processing time, limited memory, embedded platformâ€¦).

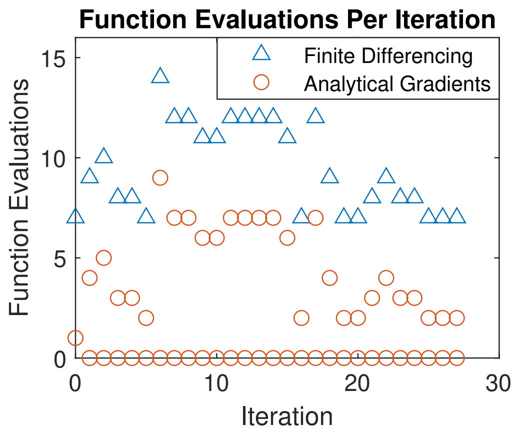
The 'Minimum Sharpness, no Lines' solution would be expected to be longer with lower peak sharpness than the 'Equal Weighting, with Lines' solution, as only reducing sharpness contributes to the objective. As expected Table 3.2 shows the peak sharpness is reduced from 0.1472 to 0.1094, around 30%. Surprisingly the total length of the 'Minimum Sharpness, no Lines' solution is also less than the 'Equal Weighting, with Lines'. In this case *sqp* is stuck in a local minimum when additional straight lines are included. Line segments must be permitted for convergence on cases outside the boundary of the existence criteria in Section 3.3.2, but it seems that well within the boundary such as the end point in Figure 3.13a, considering line segments may lead to the introduction of local minima.

3.6.5 Curve fitting With Two Clothoids with Analytical Gradient

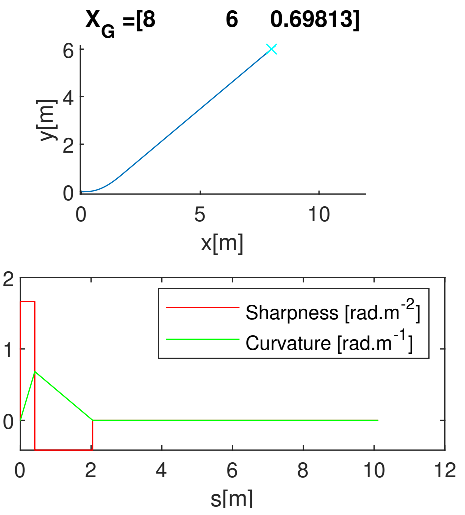
The impact of analytical gradients was tested on a single region containing two segments to the point [8, 6, 40], close to the lower angle limit, as shown in Figure 3.14b. With an initial guess which did not meet the regions constraints $\boldsymbol{p} = [0.1, -0.1, 1, 1, 1, 1]$, *interior-point* method[1] failed to converge. The alternative algorithm *sqp*[1] converged in 27 iterations with or without derivatives as shown in Figure 3.14a. When using analytical derivatives the number of function evaluations was reduced from 264 total to 123 total but the total execution time increased from 1.33 seconds to 2.28 seconds[2]. Therefore the mean execution time per function evaluation increased from $1.33/264 = 5$ms to $2.28/123 = 18$ms with the additional of numerical gradients.

[1] *interior-point* and *sqp* are available as options for the *fmincon* function of MATLAB

[2] Numerical tests ran on a consumer laptop with 16GB RAM and an Intel(R) Core(TM) i5-8250U CPU @1.60GHz



(a) Both took 27 iterations so are shown on the same figure



(b) Both produced exactly the same path and curvature profile

Figure 3.14: Comparison of performance with and without analytical gradients. The number of iterations is high because the heading angle at the end is close to the lower limit given by Equation 3.5 for this position.

On a similar problem with a different goal heading, providing analytical derivatives reduced the function count reported by *fmincon* from 213 to 55 and the number of iterations to reach the same optimality threshold from 25 to 24. However the computation time increased by about 50% from 2.35 seconds to 3.00 seconds. If the solver was changed to *sqp*[1], the effect was similar. Slightly reduced count of function evaluations but an increased computation time. This may be due to the numerous integral terms in the expressions for the gradients taking more time to evaluate than the function at multiple locations to allow differencing. There are 22 integrals to evaluate in the Jacobian. Only two integrals are required to evaluate the objective function. Equations 3.3 - 3.4 were evaluated with Vectorized Adaptive Quadrature method [143] with a relative tolerance[3] of 10^{-6} . No attempt was made to store and reuse repeated terms in the Jacobian, although there are several.

3.6.6 Curve fitting With Four Clothoids With Different Objectives

In order to enable further comparison with the bisection method of Gim et al [59], another test was reproduced involving an s-shaped path comprising four clothoids to the pose $[12, 10, \theta]$, where θ varied in increments of 10 degrees. This shows clearly the trade off between path length and peak sharpness.

[3]Vectorized Adaptive Quadrature is available in MATLAB as a built in function *q=integral(fun, xmin, xmax)*, the default upper bound on error is $10^{-6} \times q$

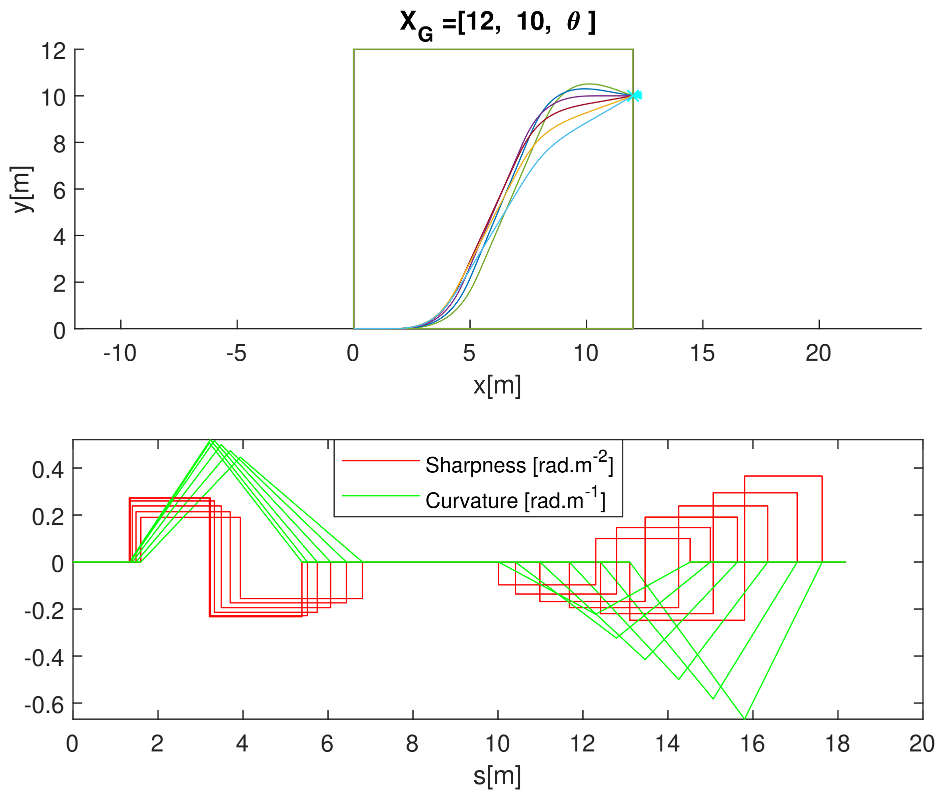


Figure 3.15: Six lane change paths with objective ‘Equal weighting, with lines’, each ending at the same point [12,10] with a heading θ separated by 10 degrees.

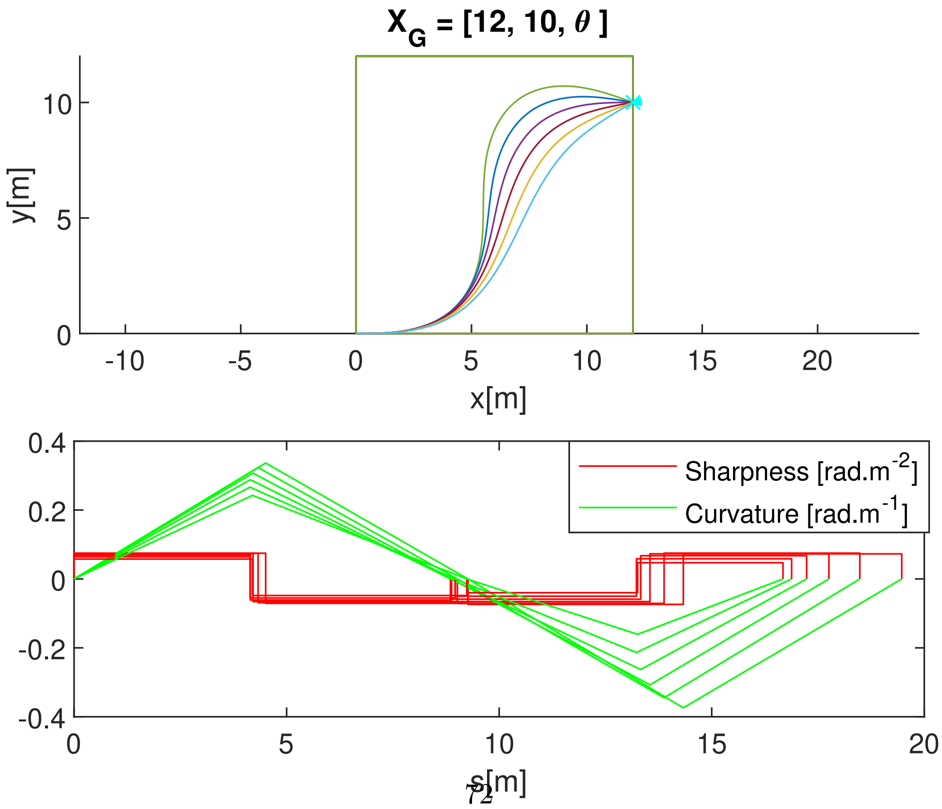


Figure 3.16: Six lane change paths with objective ‘Minimum sharpness, no lines’, each ending at the same point [12,10] with a heading θ separated by 10 degrees

The ‘Equal Weighting, with Lines’ set of paths is shown in Figure 3.15. For the final angle of -30 degrees the key parameters of the solution are shown in Table 3.3. The cost is equivalent to penalizing path length with $b = 1[m^6 rad^{-2}]$ but because each segment is squared individually, the cost is lower if each segment is of similar length. The ‘Minimum Sharpness, no Lines’ set of paths for comparison is shown in Figure 3.16. For every tested heading the path is significantly longer than the ‘Equal Weighting, with Lines’ solution, while the peak sharpness is much lower. This makes intuitive sense because over longer distances the shortest path is a straight line. As the sharpness is increased the path comes to resemble the point to point line more closely.

Table 3.3: Parameters identified for a path from [0,0,0] to [12,10,-30]. This is one of the curves plotted in figure 3.15. The objectives are explained in Section 3.4.2 Units α [m^{-2}], κ [m^{-1}], L [m]

	Equal Weighting with Lines	Equal Weighting, no Lines	Minimum Sharpness, no Lines
α_1	0.1707	0.0552	0.0727
α_2	-0.1404	-0.0443	-0.0709
α_3	-0.2808	-0.0894	-0.0745
α_4	0.4582	0.1716	0.0665
Peak Sharpness $\max \lvert\alpha_i\rvert$	0.4582	0.1716	0.0745
L_1	2.4569	4.8616	4.7639
L_2	2.9860	6.0572	4.8878
L_3	2.7117	5.4077	5.2704
L_4	1.6615	2.8180	5.967
s_{01}	1.7170	0	0
s_{F1}	3.2565	0	0
s_{02}	3.2565	0	0
s_{F2}	0.0893	0	0
Total Length	18.14	19.1445	20.8288
κ_{m1}	0.4142	0.2685	0.3464
κ_{m2}	-0.7557	-0.4837	-0.3928

Convergence with the ‘Equal Weighting, with Lines’ cost was particularly strong in the experiments attempted compared to either the more natural absolute sum of the segment lengths, all squared $J = (\sum_j |\alpha_j|)^2 + (\sum_j |L_j|)^2 + (\sum_j |s_{0,j}|)^2 + (\sum_j |s_{F,j}|)^2$ or the minimum sharpness with the line segments fixed at zero so $J = (\sum_j |\alpha_j|)^2$. Again the minimum sharpness approach produced paths almost identical to the bisection method proposed by [59] on the examples they reported.

The final position was taken from [59] so it can be used to compare the curvature profile and path trace to the one produced by the bisection method presented in that paper. This is a root finding approach which is suitable for meeting the constraint on final position. No objective function is defined so the first solution which meets the constraint tolerance will be accepted. The 'Minimum sharpness, no Lines' path trace in Figure 3.16 looks very similar to the paths produced by the bisection method. The total length of both is around 20.8m. The coordinate system differences make side by side comparison a little challenging, as they used a start position of $[0, 0, 90]$ ending at $[10, 12, 120]$ which is the same path subject to some affine transformations. The curvature profile is comparable side-by-side and shows the sharpness of each section found by bisection to be close to 0.07125, very close to the average of the 'Minimum Sharpness, no Lines' column in Table 3.3.

The optimization method has a clear advantage as it searches intermediate angles by continuously varying the length and sharpness of the first two clothoid segments. Bisection with four clothoids by contrast only searches very coarsely over intermediate angles (every 10 degrees). Searching more effectively should be a big advantage of using optimization, but in this particular example the improvement is very small. The main benefit of the new method is the ability to take into account obstacle constraints at the start and the end and find a path purely from a polygonal representation of the obstacles.

A range of curves with θ varying from -30 degrees to 20 degrees is shown in Figure 3.15. The parameters can be compared to the same range of angles with a cost function which only penalizes the sum of squared sharpness of each segment and forces the straight lines at the start and end of each segment to zero in Table 3.3. The sharpness is reduced but the total path length is increased until it is almost identical to the curve plotted in [59]. The added value of using four segments over three is questionable as the middle two take almost the same value.

The expected trade off between peak sharpness and total path length can be seen in Table 3.3.

3.6.7 Effect of Small Changes to Region Boundaries

When the convex regions are constructed based on an occupancy grid as suggested in Section 3.4.1 measurement errors may lead to sudden changes in the size of the free space as the probability of one cell being occupied crosses the threshold. In this situation the size of the position shift would be determined by the cell size of the binary occupancy grid. The cell size is typically made larger than the sensor noise by some constant factor. A cell size of 100mm was selected to provide an ample boundary for a common sensor with zero mean error, 10mm standard deviation [135].

The offset error was varied in increments of 100mm, and the resulting change in objective function is shown in Figure 3.17. Both curve types increase linearly with the position error at first. The cubic objective is the second derivative of position, while the clothoid minimized 'Equal Weighting With Lines' from Section 3.4.2 The clothoid curve objective increases more slowly until the final offset of 0.5m which causes a step change in the objective. This seems to be related to the lack of a constraint on the midpoint of the curve piece. As the obstacle is shifted the path must deflect more and in this case the midpoint leaves its assigned region. The problem of sensor errors pushing the path planner into a suboptimal local minimum could be significant. An additional constraint the midpoint should resolve it in this case, but for real world implementation some kind of safety checks will be needed before attempting to drive a new path for situations like this.

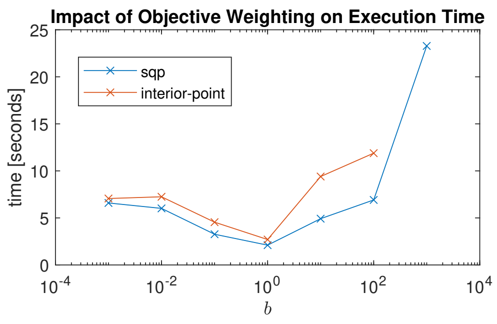


Figure 3.18: Convergence time variation with weighting on item fetch environment

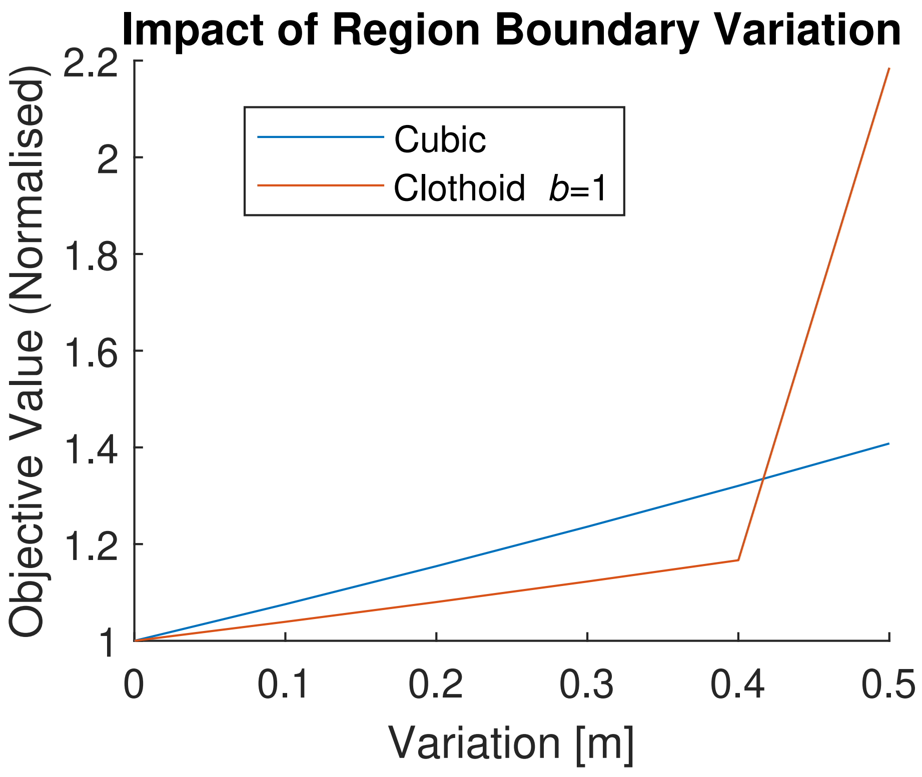


Figure 3.17: Normalised objective function against a linear shift of the boundary in x and y

3.6.8 Convergence dependence on b

In environment 1 both large and small values of w led to poor convergence. One possible explanation is numerical conditioning being damaged by the weighting. A large w

results in a solution with small α, while the length remains on the same order. Some matrix operations may return inaccurate results with different magnitude parameters. To test the hypothesis that only method using dense matrices such as sqp would be

b	1×10^{-3}	1	100	1000*
fevals	7929	10873	11986	200,000*
execution time (s)	55	72	80	1345*
path length (m)	7.071	11.694	36.038	5.695*

*Convergence Failure

Table 3.4: Convergence for different b values for the multiple shooting approach, in pallet environment

affected, the effect on the interior point method (which does not depend on dense matrix operations) is reported in Table 3.4. This shows that changing the weighting damages convergence significantly to the extent that no valid result is found even after 200000 function evaluations. There were no warnings printed by fmincon during this run. The value of feasibility was positive of the order 1e-6 and decreasing very slowly, while the objective function was large of the order 1e6 and also decreasing very slowly (relative to its magnitude).

Contrary to expectation, reducing b to emphasize the path length reduced the number of iterations to convergence. Table 3.4 shows an inverse relationship between b and the number of iterations. Sufficiently large b makes the problem more difficult to solve as the path segments deflect further, making a linear approximation worse. This effect is indirect because the constraints use the precise nonlinear dynamics and the interior-point method does not solve a series of linear approximations to the constraints like sqp. Interior point should always converge if given an initial guess in the feasible reason, unless the problem is non-convex. The initial graph step intended to remove non-convex elements of the problem may lead to suboptimal paths as the segments deviate from straight lines. But this does not explain the increasing number of iterations.

Curvature is more weakly linked to the parameters than the total length. The curvature objective is the squared sum of $2r$ parameters while the length objective is the squared sum of $4r$ parameters. Both of them leave numerous parameters to be coupled through the constraints as in total there are $9r$ parameters for the multiple shooting formulation.

3.7 Fitting Smooth Arcs to Polygon Regions - Comparative Results

Two poses in a convex region can be joined with a cubic spline. The next test compares the curvature profile of a cubic spline found in the same way: optimization within region constraints. The objective for the cubic optimization is to minimize the second derivative of position, which is analogous to curvature. The objective for the clothoid spline is to minimize the weighted squared sum of sharpness and path length as before.

3.7.1 Curvature Comparison with Cubic Spline on Environment 2

Side-by-side comparison of curvature against path length (lower) plot of the clothoid in Figure 3.19 and the cubic curve in Figure 3.20 reveals the clothoid curve has a higher peak curvature, but lower maximum sharpness of 1.5 [m^{-}2]. The cubic curve only has enough degrees of freedom to meet the heading constraint at the start and end by relaxing the constraint fixing the second derivative to zero at the start and the end. The curvature plot reveals spikes at the transition between one piece and the next. The sharpness peaks at the start and end and is too large to be shown on the figure. This will result in the steer-drive wheel actuating rapidly and likely greater tracking error. Using a higher order polynomial could improve on this, but not to the extent of the clothoid curve which has piecewise constant sharpness and the magnitude is always less than 2[m^{-2}] so it can be tracked by a real AGV such as the one described in Section 3.3.6.

Focusing on the x-y (upper) plot in 3.19 shows a different serious problem with the clothoid path. The middle section actually departs from its assigned convex region. This solution is found because the region constraints are only applied at the start and end of each segment currently. The degree of corner cutting can be reduced by increasing the number of samples, at the cost of increased execution time. Corner cutting is also likely with cubic splines if constraints are only applied at discrete intervals. The simpler cubic spline however permit a sum-of-squares constraint to ensure full containment in polygon regions. A similar result was not identified for clothoids.

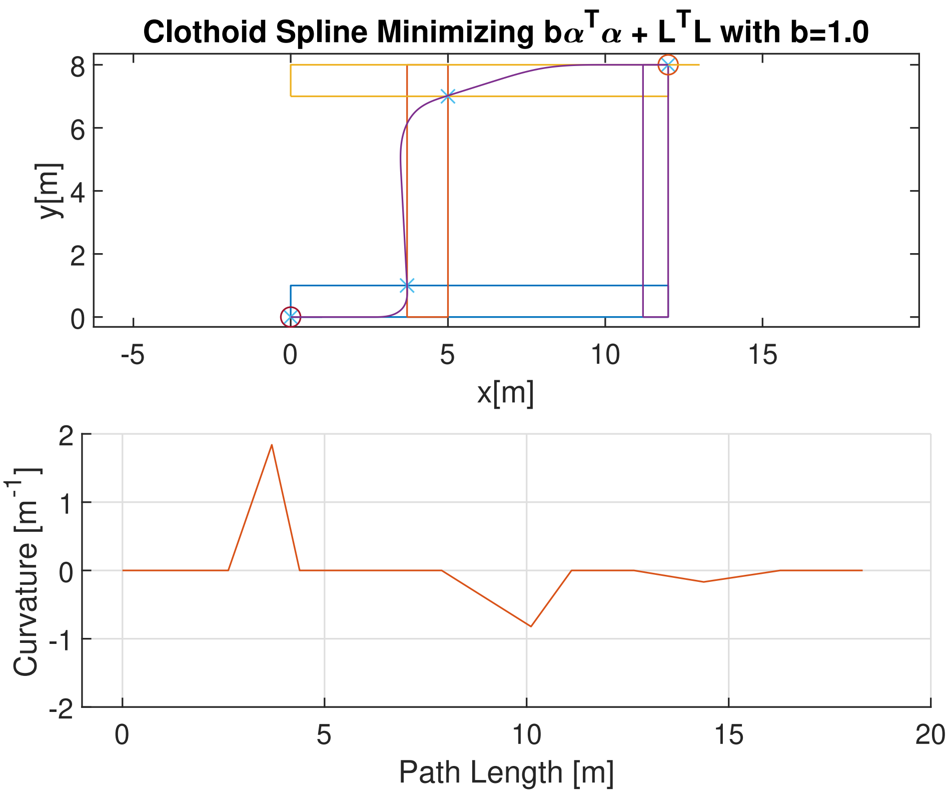


Figure 3.19: Path Trace and Curvature for Clothoid Spline

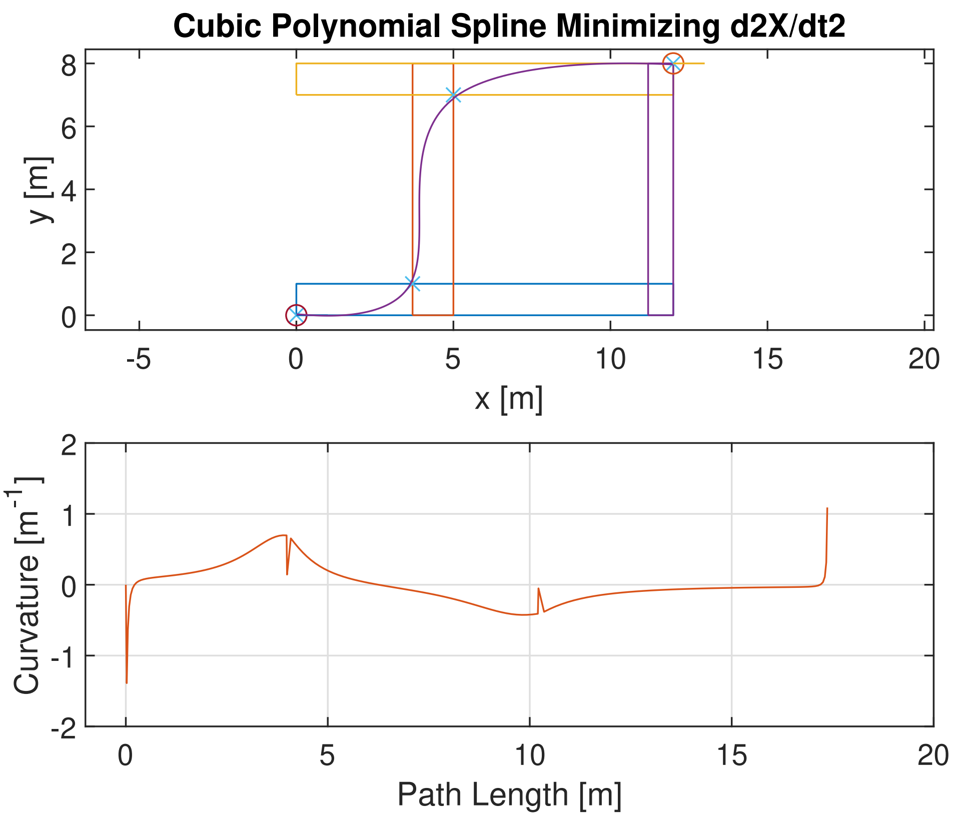


Figure 3.20: Path Trace and Curvature for Cubic Polynomial Spline

3.8 Summary and Links to Next Chapter

The formulation presented in Section 3.4 is sufficient to solve the path planning with obstacle constraints using clothoid curve segments with G^2 continuity, using a generic constrained non-linear solver. The path with three regions converges reliably and it can be extended to more complex scenes. Execution time of several seconds indicates a need for further optimization, but the path is equally smooth as the interpolation method of Gim et al [59], without the need for a sequence of points from manual driving or a heuristic for selection of waypoints based on the obstacle field, likely to be suboptimal.

Testing a wider range of environments is left to further work. This may motivate an increase in the number of obstacle constraint points on the path to prevent clipping problems seen in Section 3.6.2. A constraint on the point of peak curvature may be most helpful before sampling the curve at approximately linear intervals.

As discussed in Section 3.4.1, provided N connected regions can be found containing two poses, there always exists a path between them with G^0 continuity. It follows that a path with G^2 continuity comprised of line segments and clothoids in the order L-C-C-L can be found, as long as the peak curvature is unbounded. A wider range of test cases would be useful to identify cases where a path exists but the peak curvature is so high few tricycle vehicles would be able to follow it, or differential-drive vehicles would have to slow down unacceptably.

The method presented in this chapter addresses reactive planning by modifying the purely spatial guide paths. The question of longitudinal speed selection has been left open. The output paths shown in Figure 3.15 have G^2 continuity so the curvature varies smoothly. A speed limit can be calculated for a vehicle with Ackermann or differential drive steering geometry based on the limit to lateral acceleration and angular acceleration of that platform relative to the maximum rate of curvature of the path. For more details see Appendix D.

A simple solution to longitudinal speed would be to always proceed at the upper limit, and leave conflict resolution at intersections to the routing layer. A modification suitable for decentralized execution is to vary speeds within the limit to adjust the arrival time at guide path intersections. This enables the conflict resolution task identified in Figure 1.3 to be solved independently of the routing task. Shortest path routing could be combined with the traversal speed for multiple vehicles being jointly optimized to maximize throughput while avoiding inter-vehicle collisions. The problem of avoiding inter-vehicle collisions at an intersection without changing path geometry is addressed in the next chapter.

Chapter 4

Automated Intersection Management for an Isolated Elementary Intersection

4.1 Chapter Abstract

In this chapter a design for a Automated Intersection Manager (AIM) agent is introduced. As shown in Figure 4.1 this agent is responsible for resolving conflicts between a subset of agents whose route takes them through a particular intersection. The design is evaluated in simulation with random arrivals at discrete source locations, and multiple vehicles on each approach lane. The heuristic ordering algorithm presented matches the performance of a non-linear optimization from literature on the test intersection, while being a linear a program provides fast, guaranteed execution time.

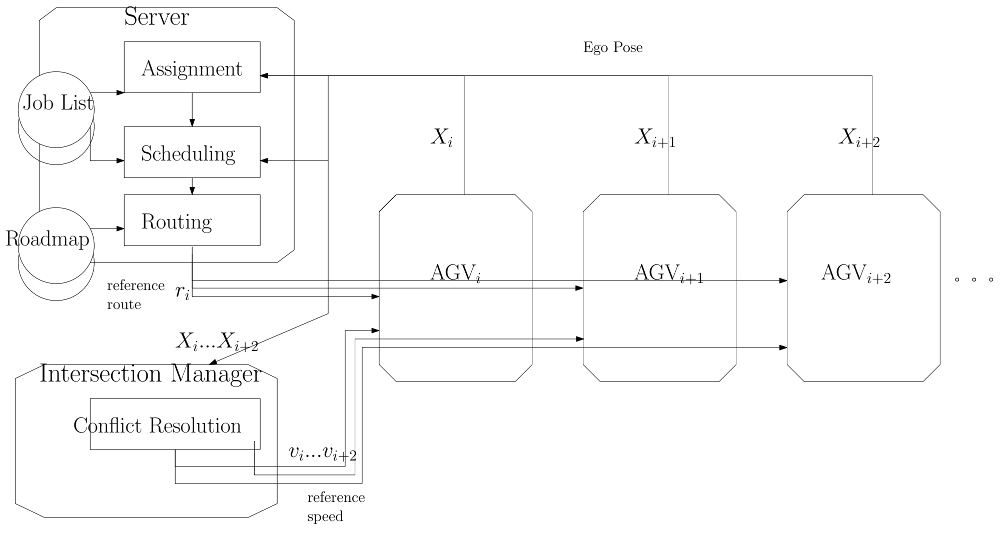


Figure 4.1: Overarching activity diagram showing the responsibility of the Intersection Manager.

4.2 Introduction

The design of fleet control software responsible for assigning movement tasks and resolving motion conflicts for large numbers of vehicles is a challenging problem. The solution presented by [76] guarantees correct behaviour and is decentralized and therefore scales well. It is currently based on time-reservation of path segments. The negotiation process takes some time, and the resulting manoeuvres may include complete stops and diversions including sometimes reversing.

In [2], a different decentralized negotiation scheme was improved by adding and

Autonomous Intersection Management (AIM) layer. The manager agent at each intersection collects speed and position information by Vehicle-To-Infrastructure (V2I) messaging before the vehicles get close to obstructing each other. In some cases an avoidance manoeuvre can be replaced by a speed variation at an earlier time. Unlike many systems based on negotiation, the study used a shared roadmap with full duplex fixed direction lanes. The key benefit of the AIM approach was higher average speeds without route changes.

The magnitude of speed improvement is likely to depend on the roadmap as car-following and AIM are more appropriate for full duplex lanes with fixed direction of travel. In automated warehouses it may be desirable to use half-duplex single track lanes with passing places to allow the stock shelves to be closer together. The replenishment of palletized goods with robotic fork lift vehicles where there is space for passing was studied in [76]. Another interesting scenario is collaborative warehouse robots with no manipulators. Each robot, carrying a tote box, can speed the return of items picked from shelves by acting as a dynamically placed conveyor belt. If human pickers are involved, the workspace is shared with unpredicatable dynamic obstacles and it is harder to prevent new static obstacles from accumulating. Some degree of local sensing and path adaptation may be needed as discussed in Chapter 3. The AIM design set out in this chapter improves the average speed crossing guide path intersections while being compatible with local avoidance manoeuvrers. It is also independent of global routing, so it can be used along with decentralized route planning as well as traditional centralized routing.

Research question: what is the impact of AIM and longitudinal platooning on the throughput and energy consumption at an isolated intersection? In particular, the performance in combination with decentralized fleet control, operating on dynamic paths with fixed direction of travel.

Hypothesis: The performance of a fleet of collaborative robotic warehouse vehicles can be improved by integrating Automated Intersection Management (AIM) and car-following behaviour/ longitudinal platooning.

4.3 Methodology

A numerical simulation of an isolated, elementary intersection is tested with random arrivals on two lanes at the marked sources shown in Figure 4.2. We propose a messaging interface where each new arrival transmits its motion plan on approach to extend the AIM concept to vehicles with dynamic paths and directions.

The delay and energy consumption sensitivity to different traffic scenarios is tested.

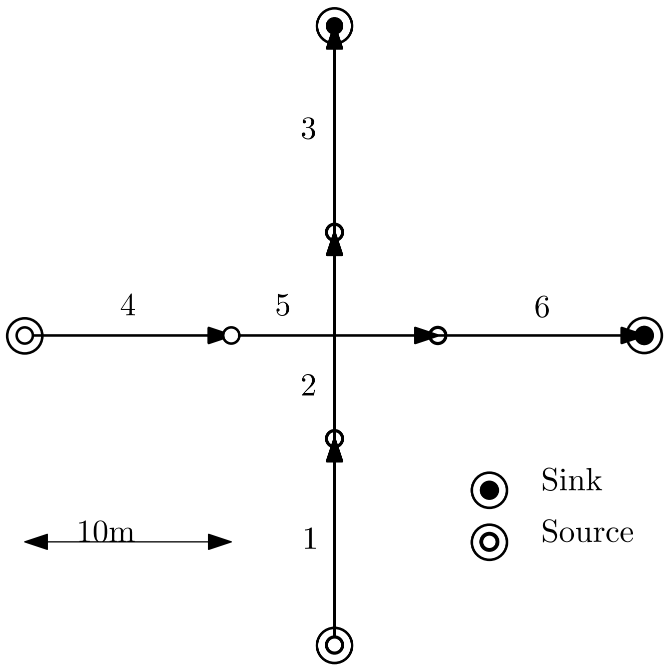


Figure 4.2: Elementary Intersection layout with two conflicting routes.

4.4 Application Context

4.4.1 Roadmap-based AGV System

Consider a demand responsive AGV system for intra-logistics [124] or a smart factory [73]. The system is concerned with completing a series of material transfer tasks. A well known solution to motion planning in a well known environment involves simplifying the free space into a (possibly irregular) lattice of reachable states, connected by arcs if there exists a feasible transition from one state to the other, to create a roadmap which can be encoded as a graph. A sequence of intermediate positions associated with each arc is sometimes stored alongside to avoid online re-computation. Using the roadmap graph, motion plans between any two states can be generated using a shortest path algorithm, which are detailed enough to be followed by the lateral position controller on board the vehicle.

In a centralized system the transfer tasks are assigned to available AGVs by a single scheduler which is aware of the status of every task and the position of every vehicle. The optimal assignment would minimize the makespan or total time for the completion of all tasks, but in practice this may be too time consuming, especially if new tasks are being generated all the time like in a fulfilment centre [7]. Conflict-free route planning depends on the task assignment and can be solved for jointly along with the assignment or performed sequentially based on a fixed assignment by searching the space time extended network to guarantee collisions are avoided.

Since 2010, a number of decentralized systems have been developed which offer advantages in the number of vehicles that can operate in one area, reduced downtime for reconfiguration and safe interaction with human operators [20] . In [144], a roadmap representation is still used, but the roadmap is shared between vehicles. The partially decentralized system described in [87] combines traffic routing with per-intersection control that is primarily roadmap based. In [145] the system is improved with the possibility for an AGV to deviate from the roadmap based on its own sensors and based on a shared sensor state called the global live view. In such a decentralized system, an intersection controller cannot be assumed to know the motion plan of approaching vehicles, unless they communicate their intention as part of the protocol. To this end it is assumed a channel exists with sufficient bandwidth and a fixed latency T for the messages described in Section 4.5.1.

4.5 Modelling Plant and Interacting Digital Control Systems

To examine the approach to intersection control an agent based model has been utilized. Although the objective is common between every AGV and the intersection manager, each agent has access to different information with a certain latency. For example, AGV agents representing the embedded PC on-board each vehicle have low latency access to ego-motion from e.g. Inertial Navigation and range data to nearby obstacles from e.g. a forward facing LiDAR. All additional state information is exchanged according to the messaging interface defined in Section 4.5.1. It is an implementation of AIM* [109], with some adaptations to use the roadmap representation of Digani et al, which is typical in the AGV space [2]. AIM* was selected as it offers scope for the intersection controller to improve performance through optimization, compared to earlier interface descriptions such as [146].

4.5.1 Dual Waypoint Interface

The dual waypoint interface is designed to be decoupled from the algorithms for scheduling and routing as far as possible. In order to support decentralized routing with adaptive paths, each approaching vehicle must send an ApproachPlan message that contains a detailed plan for how it intends to cross the intersection. The ApproachPlan contains four parameters $d = [t_A, \boldsymbol{X}(s_A), v_A, \boldsymbol{X}(s)]$. The plan consists of a transmission timestamp t_A, a measured position $\boldsymbol{X}(s_A)$, and speed v_A at the given time and a sequence of feasible positions with no timing information, the path $\boldsymbol{X}(s)$.

Embedding the path in each request for guidance means the approaching AGV can use obstacle avoidance planning before they enter the approach lane, and still receive the correct speeds at the intersection. As a result the size and shape of the conflict zone is not fixed but depends on the current traffic situation and the approach plans received.

The conflict zone shape is calculated by discretizing $\boldsymbol{X}(s)$ into linear segments of length $L = 1$m and searching for points where the minimum distance between two segments exceeds the diameter of the AGV bounding circle, and the direction of the segment is different. This ensures there is no conflict point identified where one segment joins another, which arises when two AGV are following the same path one after the other.

The intersection controller is responsible for generating an optimal speed profile $v(s)$ for the path. The resultant trajectory $\boldsymbol{\xi}(t) = X(\int_0^t v(u)du)$ must satisfy the collision avoidance constraints formed from trajectories of all known approaching vehicles $\boldsymbol{\xi}_i(t)$ $\forall i \in N$.

The trajectory across the intersection $\boldsymbol{\xi}(t)$ is found from the path $\boldsymbol{X}(s)$, the start time t_A and start position $\boldsymbol{X}(s_A)$ using Equation 4.1.

$$\boldsymbol{\xi}(t) = \int_{t_A}^{t} \boldsymbol{X}\left(v(u)\right) du \tag{4.1}$$

The speed profile is always expressed as two average speeds for two segments. The first segment AB begins at the position of the AGV at transmission time $\boldsymbol{X}(s_A)$, and ends at the nearest edge of the intersection conflict zone $\boldsymbol{X}(s_B)$. The second segment BC begins at $\boldsymbol{X}(s_B)$ and ends at the far edge of the intersection conflict zone $\boldsymbol{X}(s_C)$.

To represent this level of detail, the DualWaypoint contains four parameters $d = [t_B, t_C, s_B, s_C]$. These are independent of the discretization in the ApproachPlan, and expressed in path coordinates. The flow of messages over time is shown in Figure 4.3.

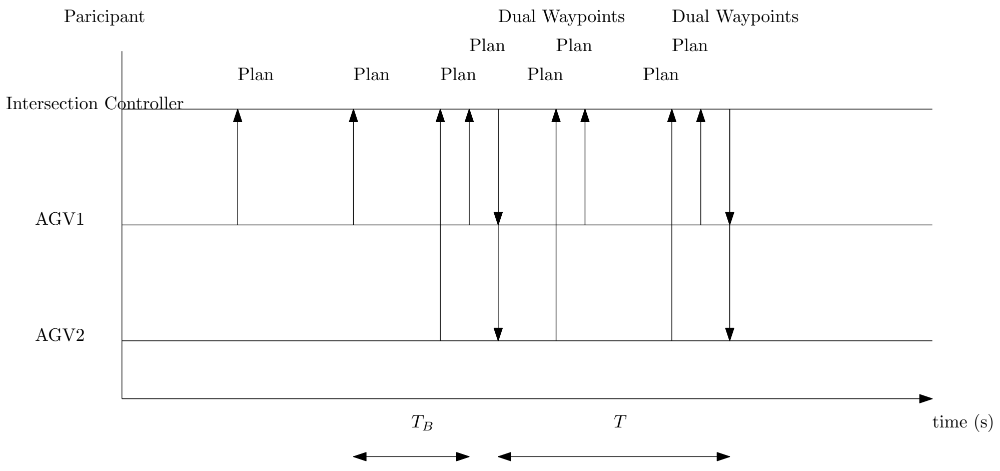


Figure 4.3: Sequence diagram for two AGVs communicating with the Intersection Controller which sends a DualWaypoint message to every known AGV every T seconds, considering the latest ApproachPlans it has received to date.

4.5.2 Longitudinal Speed Control

Longitudinal Speed Control for each Individual AGV is based on two main behaviours. The first one determines the speed on unconflicted links. The second one is required to meet the timing specification contained in the Dual Waypoint message, subject to disturbances and uncertainty in the plant. This can be done using position feedback.

Previous authors have modelled the speed on unconflicted links using car-following behaviour models. Automated traffic is assumed to follow an Adaptive Cruise Control Model with set headway, while human operated vehicles follow the Intelligent Driver Model in [147]. In the AGV space it is common to simplify car-following with mutual exclusion of discretized roadmap segments [87]. This means each may only be occupied by one vehicle at a time, even in the same direction. One vehicle follows another separated by one entire segment.The update period of individual controllers $T_L = 0.1$s must be shorter than that of the intersection controller T.

The Dual Waypoint Timing Specification is met with a constant acceleration model based on the collision-free operation modes in [148]. The agent-based simulation incorporates two modes, depending on whether the vehicle's position feedback $\boldsymbol{X}(\hat{s})$ at time $\hat{t}$ indicates it is approaching the conflict zone so $\hat{s} < s_B$ or already inside it so $s_B \leq s_A < s_C$. If the AGV has passed the conflict $\hat{s} > s_C$ then its speed is unconstrained from the perspective of this intersection controller. In the simulation, exiting vehicles always accelerate to maximum speed, at maximum acceleration α_{max}.

On approach to the conflict zone, where $\hat{s} < s_B$, the approach acceleration α_{AB} is given by Equation 4.2.

$$\alpha_{AB} = \frac{(s_B - \hat{s}) - \hat{u}(t_B - \hat{t})}{0.5(t_B - \hat{t})^2} \tag{4.2}$$

Within the conflict zone $s_B \leq s_A < s_C$ the acceleration α_{BC} is given by Equation 4.3.

$$\alpha_{BC} = \frac{(s_C - \hat{s}) - \hat{u}(t_C - \hat{t})}{0.5(t_C - \hat{t})^2} \tag{4.3}$$

4.5.3 AGV Motor Dynamic and Electrical Model

For the dynamics, every AGV was assumed to have the same mass $M = 100$kg whether loaded or unloaded, reflecting a negligible cargo mass, for example spare parts for mobile phone repair. An AGV may be propelled by brushless DC motors, which provide high torque and efficiency. Even so, a major source of power loss is internal resistance of the windings and magnetic losses in the core. The field strength of the magnets, the number of poles and the number turns of the armature coils can be captured in the motor constant k_T relating torque τ [Nm] to armature current.

$$\tau = k_T I_a \tag{4.4}$$

Similarly, the rotational speed ω [rpm] is related to the back emf ϵ [V] by Equation 4.5.

$$\omega = k_e \epsilon_D \tag{4.5}$$

These can be combined to give the plant model for one AGV in Equation 4.6

$$\ddot{x} = \frac{u \cdot k_T(V_{CC} - \epsilon_D)}{M R_a d_W / 2} \tag{4.6}$$

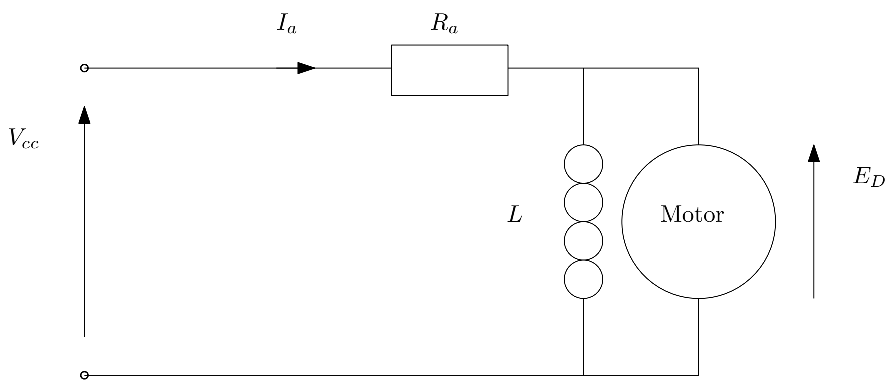


Figure 4.4: Steady state equivalent circuit for a DC motor.

There are numerous loss sources in an electric motor such as winding resistance, flux leakage, eddy currents in the core and so on [149]. By using real-world measured mechanical power output and electrical input, an equivalent winding resistance R_a for the simple model shown in Figure 4.4 can be found. The parameters are shown in Table 4.1.

Table 4.1: Motor parameters used in simulation. Electric fork lift mass and speed [5]. Motor and Electrical parameter from [6]. *Computed for equivalent circuit in Equation 4.6 to match τ_{max} at i_{max}

a_{max}	2.5	m/s^2
v_{max}	5.0	m/s
k_v	6	rpm/V
k_T	1.53	Nm/A
P_{mech}@375rpm	3.6	kW
P_{elec}@375rpm	6.37	kW
τ_{max}	127.2	Nm
$*R_a$	0.5	Ohms
V_{CC}	72	V
i_{max}	80	A
M	400	kg
d_W	0.256	m

As the top speed $v = 5$ m/s is quite low, and the vehicles stop and start frequently, air resistance which varies according to Equation 4.7 was found to be an order of magnitude smaller than the electrical losses, based on a frontal area $A = 1$ m^2 and the Drag coefficient C=1 for a cuboid shape was taken from [150]. Air density is taken to be $\rho = 1.224$ kg/m^2.

$$F_a = C\rho A v^2 \tag{4.7}$$

A brushless DC motor for an industrial vehicle typically has a constant voltage from a battery pack [5]. In this case we set V_a =72V, within the range tested in [6]. Torque can be varied from zero to maximum by changing the slip angle between the magnetic field generated digitally by the three phase coils and the magnetic field generated by the hgh strength magnets fixed on the rotor. The output of the vehicle's longitudinal speed controller must therefore be a duty cycle $-1.0 < d < 1.0$. A value of zero corresponds to zero torque, where the slip angle is zero, and a value of +/-1.0 to a slip angle of +/-90 degrees where torque is at a maximum in forward or reverse respectively.

4.5.4 Arrival Distribution

Previous work in road traffic modelling has used a variety of point distributions to model arrivals. A good summary is given in [151] or the chapter on Microsimulation in [152].

Learning from Road Transport Models

One option is to assume arrivals at a point are completely independent of each other, but occur at an average rate for the time of day which has been measured by inductive loop placed in the road. These assumptions lead to a Poisson distribution such as that shown in Figure 4.5. This can be generated computationally by drawing a sequence of numbers from a uniform distribution and applying Equation 4.8 to compute the time delta until the next arrival.

$$\Delta T = -\ln(1-p)/\lambda \tag{4.8}$$

Where λ is the average arrival rate in vehicles per second and $p{\sim}U[0,1]$ is drawn from a uniform distribution.

The assumption that arrivals are independent does not hold if traffic density is high. This is because vehicles slow down to keep a safe distance from the one in front of them.

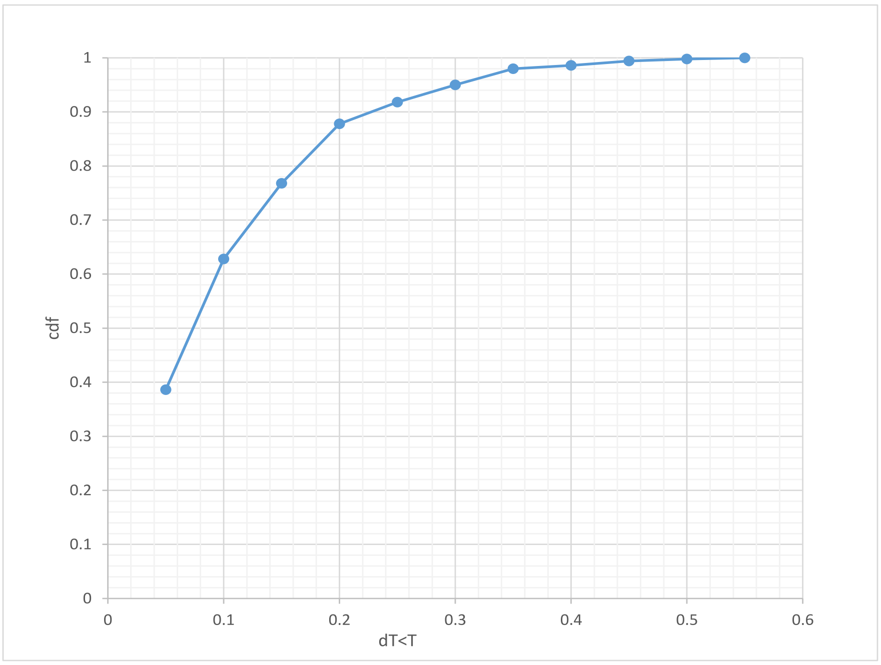


Figure 4.5: Cumulative Distribution Function of arrivals following a Poisson distribution.

The simplest modification to the poisson distribution to account for this is to discard time deltas which violate the specified safe headway, leading to the shifted exponential distribution.

$$h_T = \frac{v_0}{L} + h_L \tag{4.9}$$

To ensure non-colliding arrivals the modification in Equation 4.10 increases the minimum separation to allow space for the vehicle body L at the arrival speed v_0 and ensure a safe headway h_L between each arrival while still matching the expected inter-arrival time period $\tau = 1/\lambda$.

$$\Delta T = h_T - (\tau - h_T)\ln(1 - p) \tag{4.10}$$

To validate the arrivals drawn in this way, we check the log plot of the frequency against time delta is linear and the median is equal to the specified rate as shown in Figure 4.6. A scatter plot of times from this distribution is shown in Figure 4.7. Now the arrivals will always be realistic in the sense vehicle will not overlap/arrive unsafely but the variance will be smaller. The two parameters the average arrival rate λ and the minimum safe headway h_T together control the variance. As τ approaches the limit h_T

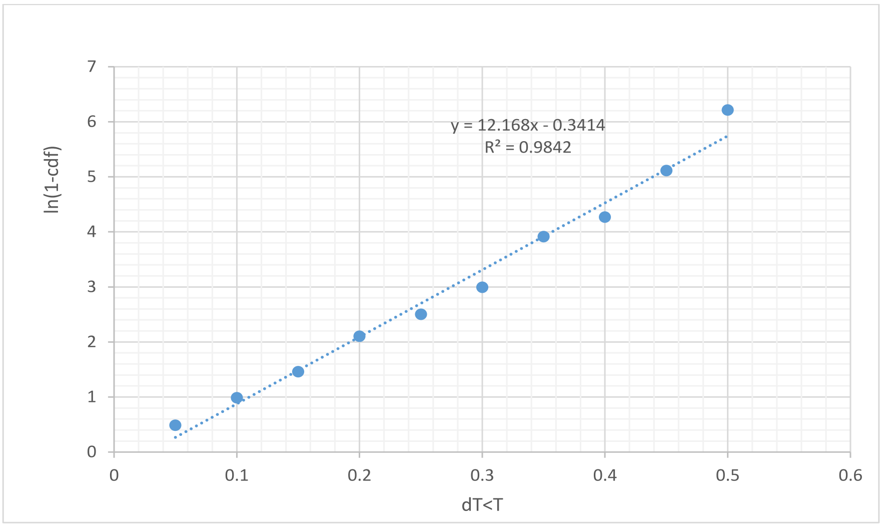


Figure 4.6: Log Cumulative Distribution Function of arrivals with linear fit.

the variance is reduced.

Unique Features of AGV Traffic

The arrival distribution corresponds to the properties of AGV traffic satisfying the following assumptions:

- More than one AGV is permitted to enter the same link if they are travelling in the same direction.
- Entry to links in the roadmap is denied if the number already present on the link reaches a fixed capacity
- Car-following behaviour within the link is governed by the frontal safety system on board each AGV. This provides an outer (warning) zone which, if tripped, the AGV will slow to w m/s at a_w m/s^2 and an inner zone (emergency)), which if tripped the vehicle will come to a complete stop at a_e m/s^2, where $|a_e| > |a_w|$
- The capacity of a link is the number of AGVs which would fit without the inner front safety zone of each being tripped by the one in front. This can be calculated for link of length L m and safety zone length d_e for each AGV as $n_c = \frac{L}{n \cdot d_w}$
- Every AGV is informed whether it is able to proceed before it reaches the end of its current link. The AGV will keep slowing down to stop at the end, until it is

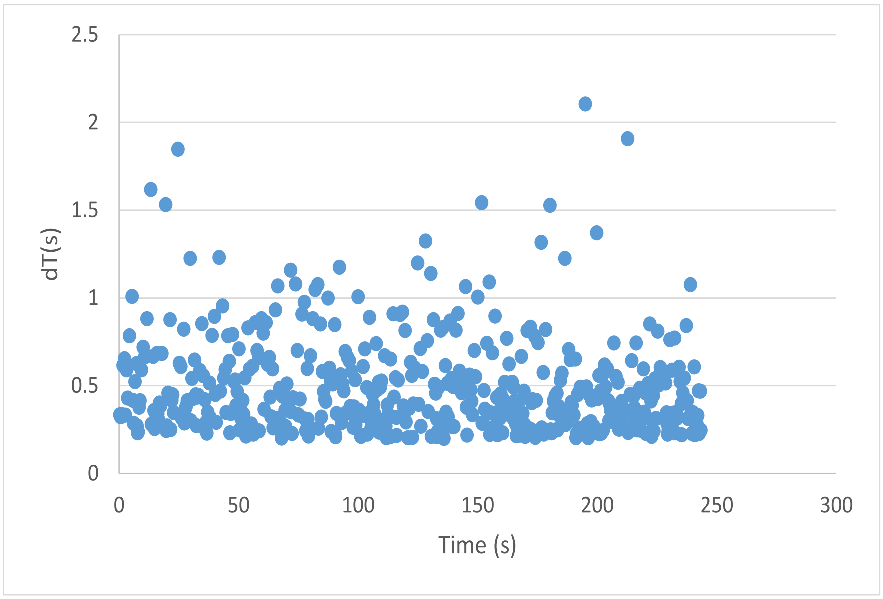


Figure 4.7: Scatter plot of 500 arrival time deltas drawn from a shifted exponential distribution with a minimum headway of 0.2 seconds.

informed there is space on the next link, at which point it will accelerate to full speed.

According to these assumptions, a suitable model is the shifted exponential, with a minimum headway based on the size of the AGV safety zone.

Arrival Variation Based on Approach Lane Occupancy

Another consideration is how the traffic on the approach lanes should feed back into the arrival times. Based on the counting semaphore for each link assumption, this can be modelled with two queues at the entryway. The lane queue and the busy queue. If the occupancy of the lane is less the n_c, new arrivals are added to the lane queue at full speed, in the simulation interval exceeding their point arrival time, at a position they would have reached by the simulation time. If the occupancy of the lane is more than n_c new arrivals are added to the back of the busy queue. At the next time step where $n < n_c$, the AGV at the front of the busy queue is moved onto the lane. The arrival speed is reduced according to acceleration rate a_W based on the time between the arrival and the time step where is can proceed. For longer time periods the new arrival will have zero speed. All arrivals which were not at the front of the busy queue will arrive at the warning speed v_w=0.3 m/s.

4.6 Method

To examine control of variable numbers of vehicles, and comment on safety effects as well as performance of experimental algorithms, utilizing a dynamic simulation test environment seemed prudent to start with. The goal here is to examine edge cases which are safety critical (may lead to a collision between AGVs). The different algorithms will be compared based on execution time, total travel time, total energy consumption and mean throughput delay with a given traffic pattern.

The symbol ϕ is used for the reciprocal speed, equal to $1/v\_ave$ over a distance segment. It is convenient to parametrize the problem in ϕ as it linearizes the travel time objective and time-based constraints. It also rules out any deadlock solutions as ϕ would need to tend towards infinity to actually stop the vehicles.

4.6.1 Conflict Zone Approximation

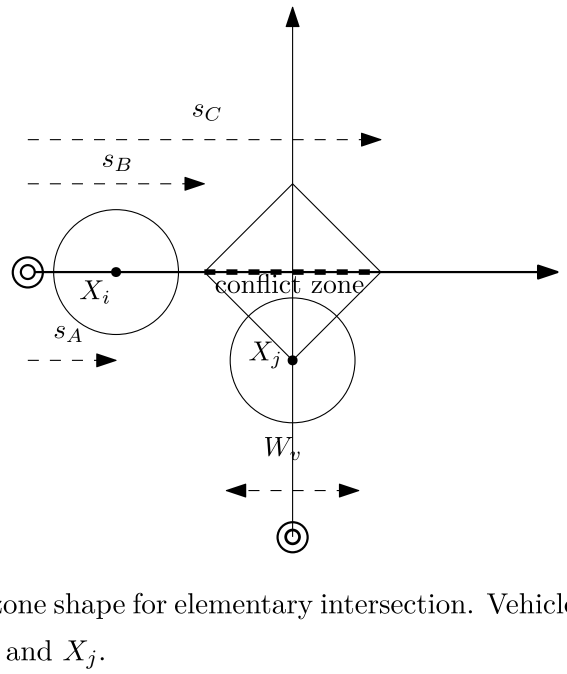


Figure 4.8: Conflict zone shape for elementary intersection. Vehicles shown as bounding circles centred at X_i and X_j.

The collision avoidance constraints are simplified by merging all the conflicts on each path, to keep only the smallest s_B and the largest s_C for that path. The extent of the conflict zone for vehicle i is given by Equation 4.11. The conflicted segment of each vehicle's path lies between s_B the smallest value of s which satisfies Equation 4.11 and s_C the largest value. The union of these conflicted segments form the total conflict zone, which is an irregular non-convex, connected compound shape, as shown in Figure 4.8.

In some cases it may be advantageous to limit mutual exclusion by only considering path segments which have different orientations to be in conflict, even if they satisfy Equation 4.11. This could allow closer spacing of AGVs travelling in the same direction by following according to the distance measured to leader by on-board sensors.

$$||X_i(s) - X_j(t)|| < W_v \quad \forall j \in N, \quad j > i \tag{4.11}$$

4.6.2 Objective

The total travel time (TTT) of all vehicles until all tasks are completed is a suitable objective to minimize in the pursuit of efficient transport. It is related but not identical to the minimum clock time to completion. If adding more vehicles to the network reduce average speeds then the TTT could increase while clock time goes down if tasks can be completed in parallel. It is chosen to highlight the congestion effect.

$$TTT = \sum_{i=1}^{N} \frac{\delta_i}{v_i} \tag{4.12}$$

In terms of the average speed on each segment v_i and the length of each segment δ_i the total travel time can be expressed using Equation 4.12.

By introducing the reciprocal speed $\phi_i = 1/v_i$ the total travel time objective can be linearized as in Equation 4.13. It is linear terms of the reciprocal speed vector $\boldsymbol{\phi} \in \mathbb{R}^{2n}$, which has up to two elements per AGV: One for the approach if it has not yet been passed and one for the conflict so $\boldsymbol{\phi}_i = [\phi_{AB}, \phi_{BC}]$. The segment lengths for the approach and the conflict are contained in distance vector $\boldsymbol{\delta}$ so $\boldsymbol{\delta}_i = [\delta_{AB}, \delta_{BC}]$.

$$\min_{\boldsymbol{\phi}} \boldsymbol{J}_T = \boldsymbol{\delta}^T \boldsymbol{\phi} \tag{4.13}$$

4.6.3 Differential Constraints

Vehicle acceleration limits are dealt with implicitly, by the maximum speed which can be expressed as a lower bound on $\boldsymbol{\phi} < \phi_{min}$. The simulated value ϕ_{min}=5m/s, is reachable within a certain distance δ_{min} from any feasible starting speed, assuming a constant limited acceleration A_{max} according to $v_{max} = \sqrt{2A_{max}\delta_{min}}$. Using the parameters from Table 4.1, the acceptable distance is δ_{min} =5m.

4.6.4 Online Feedback Considerations

In order to guarantee feasibility we need only to ensure the conflict zone length δ_{BC} and the approach length δ_{AB} are both greater than δ_{min} when vehicles receive their instructions. Owing to the fixed spatial position of the constraints at s_A and s_B, a vehicle proceeding across the intersection will eventually pass the point of no return where $\delta_{AB} = \delta_{min}$. This could also be referred to as the controllability limit distance, beyond which differential constraints (e.g. limited acceleration) make certain speeds unreachable. It can not be guaranteed that any instructions sent after this point can be satisfied by the on-board longitudinal control. The reciprocal speeds relating to any vehicle past the “point of no return” are removed from $\boldsymbol{\phi}$ and only appear in the optimization as a constraint on the speeds of subsequent vehicles. The constraint uses the latest reported speed and position for real-time feedback, so if a vehicle past the point of no return fails to meet its deadline, the later vehicles can be safely delayed until it leaves the conflict zone.

4.6.5 Collision Avoidance

The collision avoidance constraints can be expressed in terms of the arrival time a_i of approaching AGV i and the departure time d_j of approaching AGV j. For safe crossing between AGV i and j we require the condition in Equation 4.14 holds.

$$a_i > d_j \text{ OR } a_j > d_i \tag{4.14}$$

The only difference between the two AIM* approaches is the way they transform this condition into constraints on a standard form optimization which can be solved with convex methods. One uses a fixed order leading to linear constraints and the other permits any ordering through quadratic constraints.

Both optimize over the same parameter vector $\boldsymbol{\phi}$ as defined in Equation 4.15. This contains a stack of pairs of reciprocal average speeds, one for section AB and one across section BC.

$$\boldsymbol{\phi} = \begin{bmatrix} 1/v_{AB,1} \\ 1/v_{BC,1} \\ \vdots \\ 1/v_{AB,N} \\ 1/v_{BC,N} \end{bmatrix} \tag{4.15}$$

The following three subsections set out three alternative ways of expressing the collision avoidance constraints which have been evaluated. The arrival time is given by Equation 4.16. Considering average speeds, the departure time d_i is also linear, this is given by Equation 4.18.

$$a_i = \delta_{AB}\phi_{AB} = \boldsymbol{e}^T\boldsymbol{\phi}_i \tag{4.16}$$

The lengths of the segments are given by $\delta_{BC} = s_C - s_B$.

Where

$$\boldsymbol{e}^T = [\delta_{AB}, 0] \tag{4.17}$$

and

$$d_i = [\delta_{AB}, \delta_{BC}] \begin{bmatrix} \phi_{AB} \\ \phi_{BC} \end{bmatrix} = \boldsymbol{f}^T\boldsymbol{\phi}_i \tag{4.18}$$

Where

$$\boldsymbol{f}^T = \begin{cases} [\delta_{AB}, \delta_{BC}], & \text{if } \delta_{AB} > 0 \\ [0, \delta_{BC}], & \text{otherwise} \end{cases} \tag{4.19}$$

Following [2], the time window between a_i and d_i may be expressed in terms of the midpoint α and the extent β. In this way the collision avoidance constraints in Equation 4.20 are independent of the order in which AGV i and AGV j arrive.

$$|\alpha_i - \alpha_j| > \beta_i + \beta_j \tag{4.20}$$

In Equation 4.20, $\alpha_i/2$ is the midpoint of the time vehicle i occupies the conflicted segment.

$$\frac{\alpha_i}{2} = \frac{(d_i + a_i)}{2} \tag{4.21}$$

In Equation 4.20, β_i is the extent of the time vehicle i occupies the conflicted segment.

$$\beta_i = d_i - a_i \tag{4.22}$$

.

In matrix form this can be written

$$\alpha_i = \boldsymbol{f}^T \boldsymbol{\phi}_i + \boldsymbol{e}^T \boldsymbol{\phi}_i = \mathbf{1}_i^T \boldsymbol{A} \boldsymbol{\phi}_i \tag{4.23}$$

with $\boldsymbol{A} = \mathrm{diag}(\boldsymbol{f} + \boldsymbol{e})$

$$\beta_i = \boldsymbol{f}^T \boldsymbol{\phi}_i - \boldsymbol{e}^T \boldsymbol{\phi}_i = \mathbf{1}_i^T \boldsymbol{B} \boldsymbol{\phi}_i \tag{4.24}$$

with $\boldsymbol{B} = \mathrm{diag}(\boldsymbol{f} - \boldsymbol{e})$

4.6.6 Linear FIFO Constraints

FIFO is a simple ordering heuristic based on the one described by [100]. In a world where all the vehicles have the same mass and acceleration and two arrive at full speed on different approach lanes, the farther vehicle will have to slow down less to allow the closer one to pass. To use a different ordering in this simple case always wastes time, and our objective is to minimize travel time. For this method, the pairs in $\boldsymbol{\phi}$ must be sorted by the remaining distance along path AB before the start of the conflict $s - s_B$. After sorting, index $i+1$ will be further from the decision point than index i.

With a fixed ordering such as First-Come-First-Served, the reciprocal speed vector $\boldsymbol{\phi}$ is arranged in arrival order.

The constraint in Equation 4.14 only needs to be applied between adjacent vehicles and it will hold for all vehicles. This reduces the number of constraints between n vehicles to $n-1$.

The timing constraint that the leader exits the conflict zone before the follower enters is

$$d_i > a_{i+1} \tag{4.25}$$

This can be expressed as

$$\boldsymbol{e}_i^T \boldsymbol{\phi}_i > \boldsymbol{f}_{i+1}^T \boldsymbol{\phi}_{i+1} \tag{4.26}$$

leading to a pairwise matrix $Q_i \in \mathbb{R}^{(n-1)\times n}$

$$\boldsymbol{Q}_i \boldsymbol{\phi} = \begin{bmatrix} 0 & \dots & & \\ \dots & \boldsymbol{e}_i^T & -\boldsymbol{f}_{i+1}^T & \dots \\ & & \dots & 0 \end{bmatrix} \begin{bmatrix} \vdots \\ \boldsymbol{\phi}_i \\ \boldsymbol{\phi}_{i+1} \\ \vdots \end{bmatrix} \tag{4.27}$$

The pairwise Q_i matrices are added together to get A_{ub} in Equation 4.28.

$$A_{ub}\boldsymbol{\phi} > 0 \tag{4.28}$$

Care should be taken to exclude from $\boldsymbol{\phi}$ those AGV which have already passed the decision point defined as $s_B - min\_conflict\_length$. The motion of these vehicles can no longer be altered because the onboard longitudinal controller would likely be unable to meet the new waypoint times due to physical limits on acceleration.

The resulting optimization problem has linear constraints and a linear objective

$$\begin{gathered}\min_{\boldsymbol{\phi}} \boldsymbol{J}_T = \boldsymbol{\delta}^T\boldsymbol{\phi} \\ \text{subject to} \\ \boldsymbol{\phi} > \phi_{min} \\ \boldsymbol{Q_i}\boldsymbol{\phi} > \mathbf{0} \quad \forall i \in [1, 2, ..., n-1]\end{gathered} \tag{4.29}$$

4.6.7 Quadratic Constraints

The constraints can be expressed without imposing a fixed order by transforming the reserved time blocks into centroid α and half-width β representation. This leads to Equation 4.30 for the constraint between any pair of AGVs (i, j).

$$|\alpha_i - \alpha_j| > \beta_i + \beta_j \tag{4.30}$$

And equivalently Equation 4.31.

$$(\alpha_i - \alpha_j)^2 - (\beta_i + \beta_j)^2 > 0 \tag{4.31}$$

This makes it possible to express the constraint that vehicles do not collide in terms of time. Vehicle i arrives at the first conflicted segment at time a_i and departs from the last at time d_i .

In Equation 4.30, $\alpha_i/2$ is the midpoint of the time vehicle i occupies the conflicted segment.

$$\frac{\alpha_i}{2} = \frac{(a_i + d_i)}{2} \tag{4.32}$$

In Equation 4.30, $\beta_i/2$ is the extent before or after the midpoint vehicle i occupies the conflicted segment.

$$\frac{\beta_i}{2} = \frac{(d_i - a_i)}{2} \tag{4.33}$$

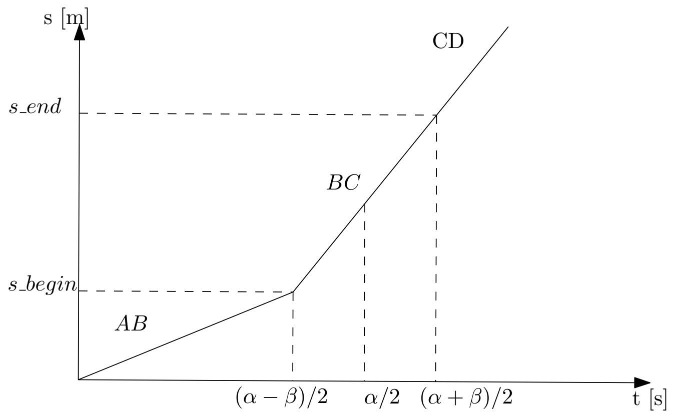


Figure 4.9: Diagram of one vehicle in path coordinates.

Equation 4.30 can be converted to standard form by squaring both sides and substituting the matrix expressions for α_i and β_i. This gives the matrix inequality for each pair of vehicles shown in Equation 4.34.

$$\begin{bmatrix} \boldsymbol{\phi}_i^T & \boldsymbol{\phi}_j^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_{ij}^{ii} & \boldsymbol{\Lambda}_{ij}^{ij} \\ \boldsymbol{\Lambda}_{ij}^{ji} & \boldsymbol{\Lambda}_{ij}^{jj} \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}_i \\ \boldsymbol{\phi}_j \end{bmatrix} > 0 \tag{4.34}$$

The four pairwise sub-matrices can be expressed in terms of the diagonalized distance $\boldsymbol{A}$ and $\boldsymbol{B}$ as follows:

$$\boldsymbol{\Lambda}_{ij}^{ii} = (\boldsymbol{A}_i - \boldsymbol{B}_i)\mathbf{1}_i\mathbf{1}_i^T(\boldsymbol{A}_i + \boldsymbol{B}_i) \tag{4.35}$$

$$\boldsymbol{\Lambda}_{ij}^{jj} = -(\boldsymbol{A}_j + \boldsymbol{B}_j)\mathbf{1}_j\mathbf{1}_j^T(\boldsymbol{A}_j + \boldsymbol{B}_j) \tag{4.36}$$

$$\boldsymbol{\Lambda}_{ij}^{ij} = \boldsymbol{\Lambda}_{ij}^{jiT} = -(\boldsymbol{A}_j + \boldsymbol{B}_j)\mathbf{1}_j\mathbf{1}_i^T(\boldsymbol{A}_i + \boldsymbol{B}_i) \tag{4.37}$$

For more than two vehicles this can be arranged into a block diagonal matrix $\boldsymbol{H}_{ij} \in R^{(n\times n)}$ which is compatible with the input parameters, but still only represents the constraints between a pair with zeros for the other elements. The full constraint matrix $\boldsymbol{H}$ is the sum of these pairwise matrices, for every pair with $j < i$.

$$\begin{gathered} \min_{\boldsymbol{\phi}} \boldsymbol{J}_T = \boldsymbol{\delta}^T\boldsymbol{\phi} \\ \text{subject to} \\ \boldsymbol{\phi} > \phi_{min} \\ \boldsymbol{\phi}^T\boldsymbol{H_{ij}}\boldsymbol{\phi} > \mathbf{0} \quad \forall i,j \in [1,2,...,n] \quad \text{with } j > i \end{gathered} \tag{4.38}$$

The condition $j > i$ in Equation 5.6 indicates that the number of constraints varies with the number of vehicles n as $\frac{n(n-1)}{2}$. This corresponds to one constraint between each pair of approaching AGVs.

4.6.8 Semaphore Based Collision Avoidance

The constraints can be enforced without any optimization using a common synchronization object: the binary semaphore. This is also based on first come-first-served ordering and the intersection controller gives the semaphore to the closest vehicle who provides an Approach Plan. This requires special messages in the dual waypoint interface, as the semaphore method provides no timing information. It issues a "full speed ahead" command to the vehicle with the semaphore and a stopping distance to all other vehicles. This consists of a distance along the AGVs submitted plan at which it must stop. The semaphore is released by the crossing AGV's position indicating is clear of the conflict zone. The semaphore is then issued to the next closest, by sending it a "full speed ahead" command

A semaphore-based system is expected to produce solutions with sub optimal throughput but be fast to calculate and guarantee safe operation. Similar schemes have been described in the literature so it is included in the comparison to give an idea of the benefits of departure time modelling and approach speed synchronization.

4.7 Numerical Results

The different approaches to intersection control were evaluated on a simulation of a so-called elementary intersection, comprised of two 30 m lanes which cross in the middle as shown in Figure 4.2. There are two entrances to the map, one at the start of each lane.

In all simulation runs, an assertion statement at every iteration ensured each pair of AGVs were separated by more than $2W_v$.

4.7.1 Quadratic Constraints Non-Convexity

The simulated setup is shown in Figure 4.2, with two AGVs approaching the crossroads one from each source node. Each vehicle is stationary at the start of its respective lane at t=0. Both vehicles request speed guidance for three segments, taking them directly across the intersection. First the Hessian is examined, then the results of a 10 second simulation of vehicles with limited acceleration are reported.

800.	400.	-800.	-400.
400.	0.	-400.	-400.
-800.	-400.	800.	400.
-400.	-400.	400.	0.

Table 4.2: Block diagonal Hessian matrix H_{01} for crossroads with two equidistant approaching vehicles

The Hessian $\boldsymbol{H}_{ij}$ was evaluated for the simple crossroads shown in Figure 4.2 with two approaching AGVs. In this case, only $\boldsymbol{H}_{01}$ is included and this is identical to pairwise $\Lambda_{01} \in R^{(4\times4)}$ given by Equation 4.34 as there are only two vehicles in total. This has eigenvalues $\boldsymbol{e_1} = [0, 2000, -400, 0]$, which have different signs so the quadratic form of the constraints is not convex. The linear objective is convex by definition but cannot be strictly convex, as strict convexity precludes linear regions. Convexity of both objective and constraints would permit multiple minima but ensure that every local minimum is a global minimum. Without convexity the search algorithms employed are vulnerable to becoming trapped in suboptimal local minima, depending on where they are initialized.

Using the 'trust-constr' solver in scipy.optimize and providing the derived Jacobian and Hessian of both cost and constraints, the execution time to find the optimal speeds was 0.22 seconds. The minimum was found to be $J_T = 5$s with parameters $\boldsymbol{\phi}_1 = [0.20000006, 0.10000002]$ and $\boldsymbol{\phi}_2 = [0.10000001, 0.10000001]$. This is close to the true value of $\boldsymbol{\phi}_1 = [0.2, 0.1]$ and $\boldsymbol{\phi}_2 = [0.1, 0.1]$ and more precision can be achieved by tuning the value of $'gtol'$. Smaller tolerance values get closer to the true min. A value of $'gtol' = 1e - 14$ was used, leading to constraint error of 2.4707×10^{-06}. There is another equally valid minimum with $J_T = 5$s which is not found with the initial guess in which all parameters were set to ϕ_{min}.

If the scenario is modified so the second vehicle starts 1m closer to the intersection, both minima are no longer equally costly. Now the global minimum where the vehicle at the start slows down to 5.26m/s to allow the closer vehicle to pass in front of it leads to $J_T = 4.8$s. The alternative order where the AGV 9m away slows to 4.5m/s leads to $J_T = 5.22$s. The Hessian was evaluated and the eigenvalues found to be

$\boldsymbol{e_2} = [0, 1848, -400, 0]$. They do not all have the same sign, so the non-convexity of the constraints is proven by counter-example. The 'trust-constr' solver, provided with analytical Jacobian and Hessian converged to either minimum depending on the initial guess. As a result the speed choice for larger numbers of vehicles could be sub-optimal if no steps are taken to explore the cost surface such as trying multiple initial guesses for each problem.

Another test involved vehicles approaching in the same lane. Only one AGV may occupy the conflict zone at a time according to the constraints so each additional AGV should slow down enough for the preceding one to have left the conflict zone by the time it arrives. At the global minimum for the two vehicle example, traffic from conflicting directions should be interleaved on a FIFO basis. However, the sub-optimal local minima lead to a more serious problem here because vehicles in a queue will collide if those further back are given higher speeds. A workaround based on 'car-following' behaviour might be implemented at an individual vehicle level, based on the distance to the vehicle in front. However individual behaviour contradicting the speed instructions from the intersection manager may lead to collisions with conflicting traffic.

4.7.2 Arrival Rate Test Scenarios

By varying the arrival rate λ and the reciprocal of the intersection manager update period $f = 1/T$, six scenarios were created with the parameters shown in Table 4.3.

	λ_1	λ_2	f
HLHT	0.5	0.5	2
HLMT	0.1	0.5	2
HLLT	0.1	0.1	2
LLHT	0.5	0.5	10
LLMT	0.1	0.5	10
LLLT	0.1	0.1	10

Table 4.3: Parameters for test scenarios. All units s$^-$1.

Each scenario is identified with the first two characters relating to the latency between periodic messages from the intersection controller where High Latency is 500ms and Low Latency is 100ms and the second two relating to the arrival rate, where High

Traffic has λ=0.5 arrivals per second on both approaches, Low Traffic has λ=0.1 arrivals per second on both, and Mixed Traffic has one lane with λ_1=0.5 and the other with λ_2=0.1. For example High Latency, High Traffic becomes HLHT

	T[s]	TTT[s]	t[s]	Δ[s]	Ee[MJ]	Em[MJ]	Ex T[s]
FIFO	45.7	181.0	6.033	0.033	43.906	30.323	0.0036
Quad	44.8	181.6	6.0533	0.053	44.832	30.964	0.5252
Sema	83.9	326.4	10.88	4.88	159.1	68.140	¡0.001

Table 4.4: Intersection performance over 30 crossings with three different controllers for the HLHT scenario.

The effects of the different controllers can be seen in the position time trace for 30 simulated crossings. The conflict point is located at the intersection between the two lanes at s = 15 m. This is expanded by the largest dimension of the vehicle to ensure that if one vehicle stops outside the zone, the next has room to cross in front of it. With 1 m bounding circle diameter, the conflict zone extends from s = 14 m to s = 16 m. Both lanes are collapsed onto one diagram, with × markers for vehicles travelling along the x axis and △ markers for vehicles travelling along the y-axis. The controller is successful provided only one type of marker is present in the conflict zone (start and end shown by horizontal lines) at one time. All controllers are safe, so the main comparison is how much the vehicles must slow down, shown by the gradient of the lines.

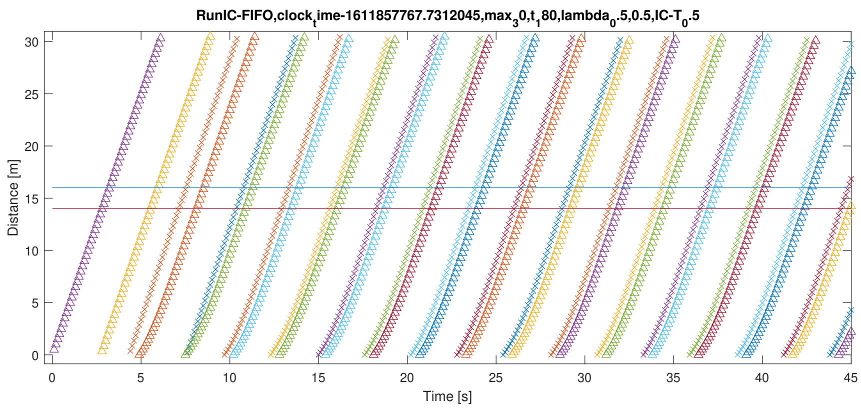


Figure 4.10: Position-Time trace for HLHT Scenario under FIFO optimal controller.

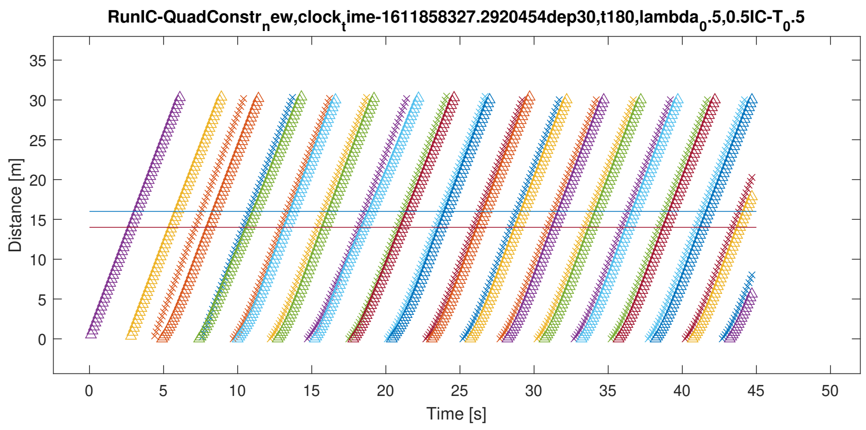


Figure 4.11: Position-Time trace for HLHT Scenario under Quadratic Constraints optimal controller.

The efficiency of modelling the departure time and adjusting speeds in advance is clear from comparing either of the optimal controllers in Figure 4.10 and Figure 4.11. Both optimal methods change the vehicle speed very slightly as soon as they come into communication range. This is seen as an increase in the gradient of alternate distance-time trajectories, as they accelerate from the queue exit speed of 2.5m/s to a constant speed which ensures they arrive at the conflict when it is free. The speed is maintained until the conflict zone (shown by two horizontal lines) is passed. The speed increase after the increase is barely noticeable as the conflict-free speeds are close to the maximum. both methods assume as similar pattern where pairs of vehicles approach t the same time, one along the x-axis lane and one along the y-axis lane. The gap before another vehicle arrives on the same lane is longer because the source is blocked until the lead vehicle completes the first 10m section.

By comparison the trajectories for the Semaphore method shown in Figure 4.12 are very different. Crossing still happens roughly in pairs but this time the gradient of those on the y-axis shown by triangle markers (crossing second) is reduced much more. They are executing constant deceleration ensuring they come to a complete stop before the conflict zone, until the lead vehicle is confirmed to have passed. This leads to lower average speeds and corresponds to an increase in delay of 4.85 seconds per vehicle with HLHT according to Table 4.4.

The two optimal methods are very close, with FIFO achieving a slight improvement in total travel time of 0.6 seconds, but a lower completion time by 0.9 seconds. This

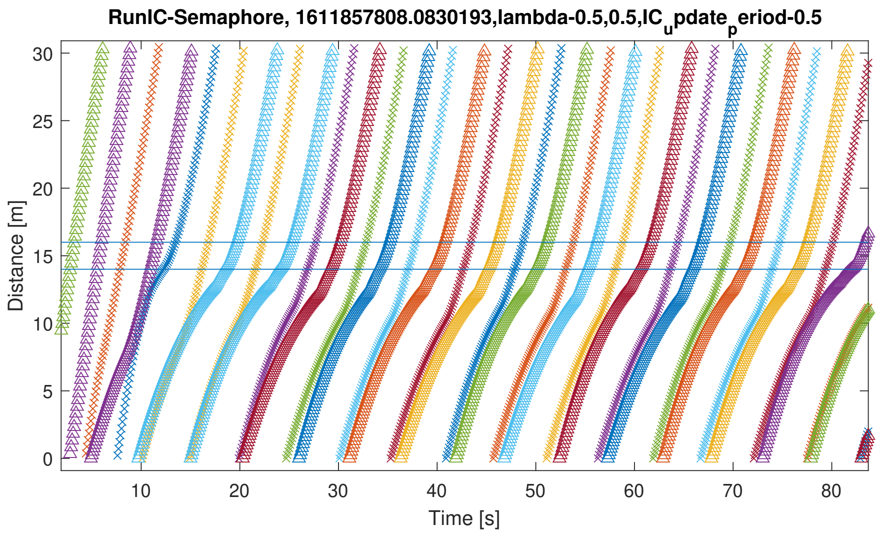


Figure 4.12: Position-Time trace for HLHT Scenario under Semaphore controller.

discrepancy may occur because the waiting time in the arrival queue is not counted in the total travel time, which should be addressed in further testing. It is more likely the Quadratic constraints achieved a slight improvement in throughput because of the freedom to vary the departure order. However, the optimal crossing order in Figure 4.11 turns out to be close to FIFO.

Another avenue of comparison is the energy usage. The semaphore method uses much more energy as the vehicles have to slow down more. Energy usage is not included in the objective for the optimal methods, so the question depends on whether higher average speeds or more acceleration lead to higher losses with our simple motor model.

The total power consumption of every vehicle in the intersection for the High latency High traffic scenario under FIFO optimal control is shown in Figure 4.13. The spikes are caused by acceleration and are higher or lower depending on how many vehicles are changing their speed at one time. The bang-bang commanded longitudinal speed control policy means that any speed change will lead to maximum acceleration for some time. Smaller speed changes mean less time so the chance of random arrivals accelerating at the same time is reduced.

The power consumption increase due to acceleration clearly dominates in Figure 4.14, as the mechanical power is around 50 percent greater for the semaphore method than in either of the optimal runs. This difference is compounded by the reduction

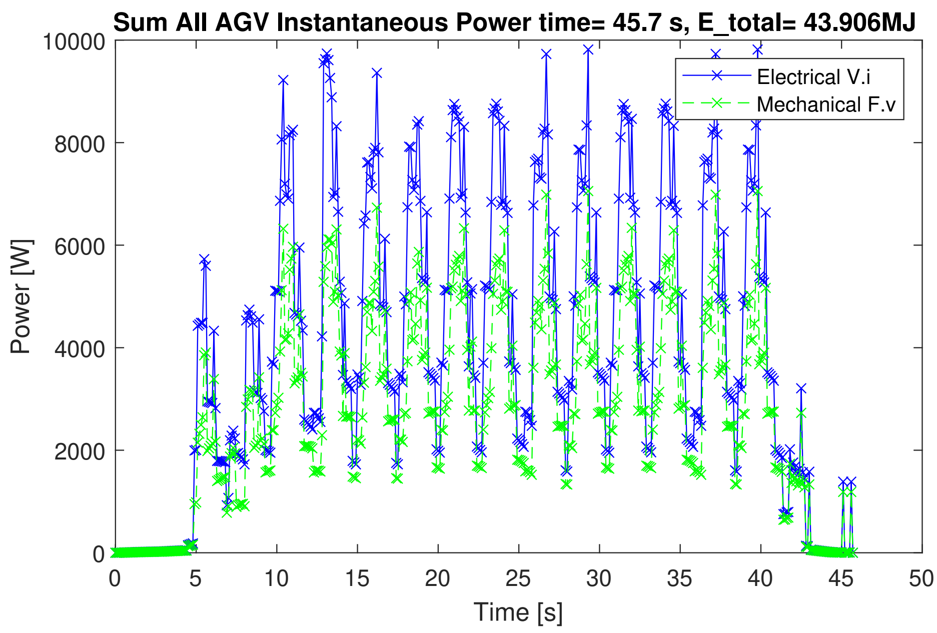


Figure 4.13: Power Dissipation-Time trace for HLHT Scenario under FIFO optimal controller.

in motor efficiency in high acceleration so the resultant increase in electrical poser dissipation is much greater, closer to 200 percent.

Power consumption is similar between the two optimal approaches on the elementary intersection. Unlike the delay, in this case maintaining FIFO order leads to a slight improvement: 43.9 MJ total energy compared to 44.8MJ. A spike in usage at around 18 seconds can be seen in Figure 4.15, possibly this corresponds to a change in order which leads to lower delay but uses some extra energy.

4.7.3 Impact of Analytical Hessian on Execution Time of Trust Region Method

The optimization problem with quadratic constraints described in Section **??** was implemented in Python and solved periodically based on the latest position information at the specified control frequency f. The method chosen was 'trust-constr' from the Scipy.Optimize library [153]. Trust region methods make use of the exact Semi-Definite Program relaxation for the Trust Region Sub-problem (TRS), of optimizing a non-convex quadratic objective subject to a Euclidean ball constraint. This relaxation is

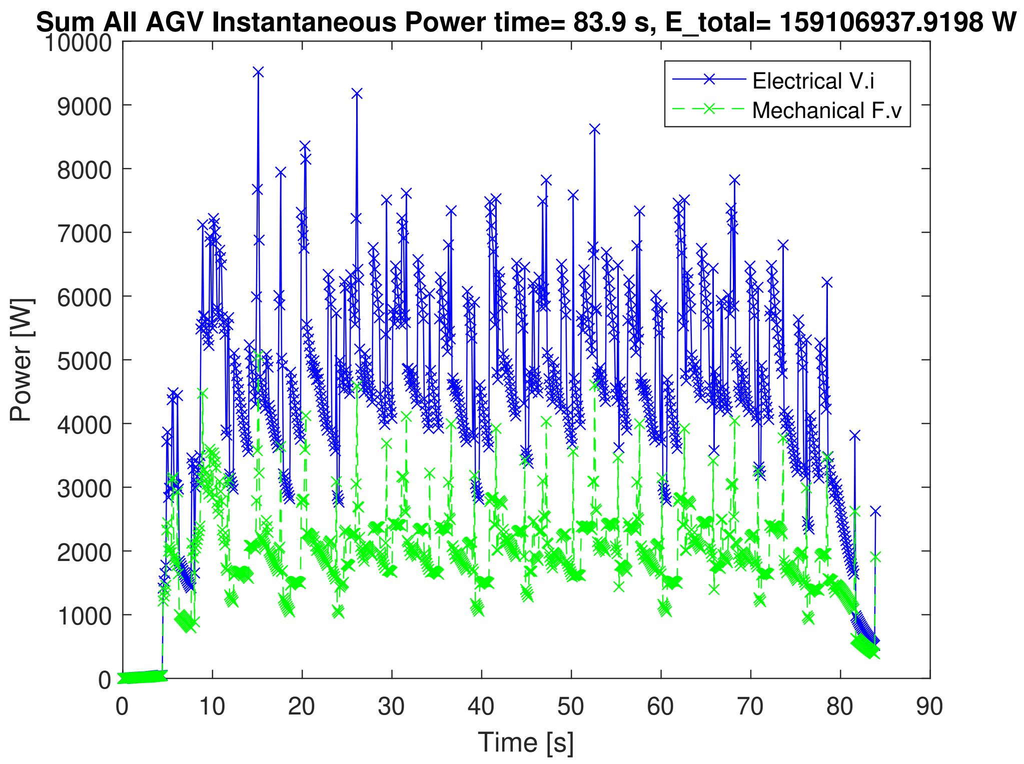


Figure 4.14: Power Dissipation-Time trace for HLHT Scenario under Semaphore controller.

used to iteratively solve the general non-convex function with non-convex constraints by successive approximation [154]. Trust region methods are more effective when the general problem has more in common with the TRS and recent methods have been proven to solve variants of that problem in linear time in terms of the input [155]. Unlike some other general constrained optimization methods in Scipy.Optimize such as SLSQP, 'trust-constr' can make use of the analytical Hessian for the objective and constraints which may be important to exploit the linear objective and quadratic constraints.

The Hessian must be provided to SciPy.Optimize in the form of a linear combination rather than a stacked matrix. This is to avoid forming the complete Hessian $H \in R^{(n \times np)}$ which may use a significant amount of memory for large problems. Instead, the analytical Hessian function must accept an additional parameter $v \in R^{(1 \times p)}$. This is a vector the same length as the constraints $c_{ineq} \in R^{(1 \times p)}$. The Hessian is returned

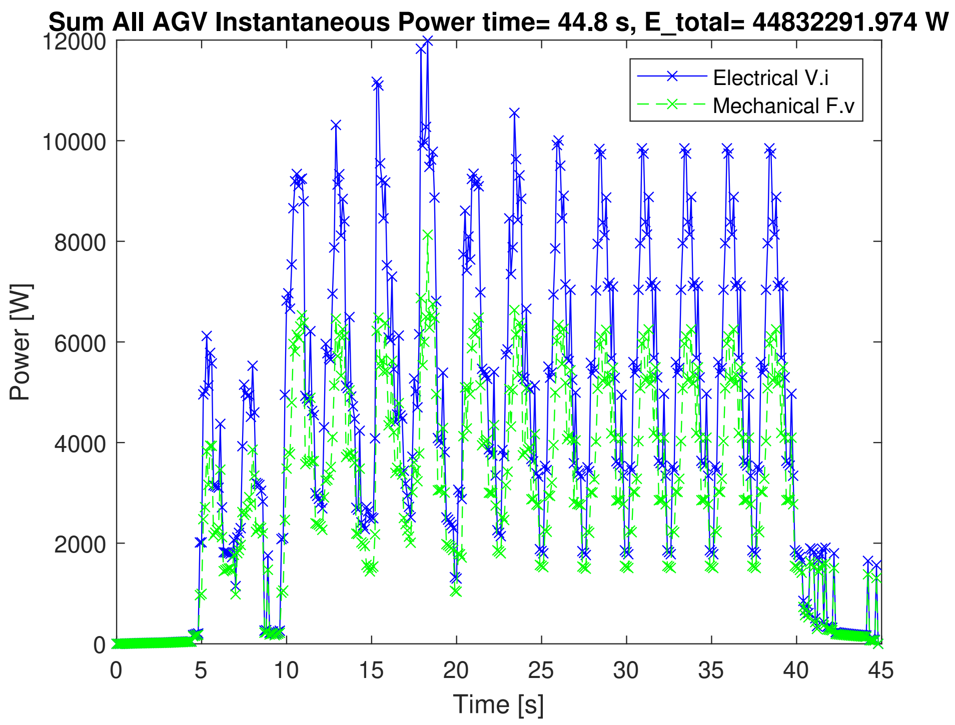


Figure 4.15: Power Dissipation-Time trace for HLHT Scenario under Quadratic Constraints controller.

as a $R^{(n \times n)}$, the weighted sum of pairwise blocks scaled according to $\sum_{i=1}^{p} v_i H_i j$.

With the analytical Hessian the average execution time for the Quadratic Constraints method over the HLHT run in which 30 vehicles passed through the intersection was 0.5251 seconds, varying between 0.0512 seconds to 1.215 seconds as the number of constraints varied from 1 to 6. Without the analytical Hessian of the constraints the mean time taken over the same run was 383 milliseconds, varying between 46.8 milliseconds to 7.696 seconds. It is surprising that the worst case time is so much worse and yet the mean time is better. This suggests that in the test data there are more cases with few constraints. It also motivates investigation into the cause of the outlier time.

The execution time with the FIFO controller never exceeds 15.6 milliseconds on the same set of problems, with the average being 3.6 milliseconds.

4.8 Summary and links to next chapter

The advantages of centralized intersection optimization over a binary semaphore heuristic are significant on the elementary intersection. The total travel time is reduced by close to 50%. The energy consumption according to a simple model with no air resistance was reduced by more than 50% by avoiding the need to come to a stop.

Concerning the optimization methods only, solving the variable ordering problem by enforcing quadratic constraints on a linear objective as reported in literature is a promising approach. Convex methods such as trust region constrained optimization made use of the analytical Hessian and converged reliably although the execution time would need to be improved to ensure real-time operation. For example, the simulation ran with a manager update period of 0.5 seconds when feedback was applied from the latest update of the actual vehicle positions. This closed-loop control corrects for errors in the execution of the commanded plan from the previous time-step. With the quadratic constraints, the average execution time was less than the simulated update period but exceeded it by 7 seconds in the worst case, with only 6 participants. This variability shows the downside of including a possibly non-convex constraint set optimization into a control system with hard time deadlines. The formulation of the intersection management problem as a linear objective with linear constraints by applying a fixed FIFO order limited the maximum execution time to 15.6 milliseconds.

Lack of ordering flexibility will lead to higher crossing times in situations where FIFO order is suboptimal. Numerical results for the elementary intersection show very similar performance in both delay and energy consumption, suggesting it is optimal in this situation. In environments similar to the one tested with identical vehicles the FIFO controller is a promising choice for real world implementation, as it can be solved orders of magnitude faster than the variable ordering problem and captures almost all of the throughput advantage.

The next step is to investigate if any performance difference emerges on more complex intersections, where exploring alternative orderings may be more significant to the objective. Another avenue to explore is the introduction of a (binary) integer variable for each pair to create a mixed integer linear program, which might be competitive with the convex formulation in terms of execution time and objective function.

Chapter 5

Multi-Lane Automated Intersection Management

5.1 Chapter Abstract

This chapter compares two algorithms used to solve the Quadratic constraints formulation of intersection management *slsqp* and *interior_point*. The first-come-first served FIFO method is used as the performance baseline, rather than the semaphore crossing heuristic. The methods introduced in Section 4.1 are compared by total travel time and execution time on multi-lane intersections with a wider variety of turning movements, where the conflict zone shape depends on the occupied lanes. The arrival pattern has been simplified to exclude car following effects. Over ten runs, one AGV was initialized at each entrance, and assigned a random exit. No further arrivals take place at any sources over the run. The individual vehicle dynamics have been simplified to first order to focus in the effects resulting from conflict resolution.

The placement within the wider picture of this thesis is captured by Figure 4.1. The messaging interface is developed further and two different algorithms for optimizing subject to the Quadratic constraints, proved to be non-convex in the previous chapter, are compared. On complex intersections the reordering capability of the Quadratic constraints formulation is shown to offer significant benefits. The trust-region constrained algorithm always finds superior solutions but the convergence time increases rapidly for larger numbers of vehicles.

5.2 Introduction

Real-world environments are rarely limited to pairs of perpendicular lanes. To establish the performance of the approaches to intersection management introduced in the previous chapter, it may be important to examine larger multi-lane intersection geometries. These often permit certain simultaneous flows such as opposing left turns (for those driving on the left-hand side). As a result, the difference between optimization of the crossing order compared to a simple ordering heuristic like Nearest-First (called FIFO in the test) might be shown more clearly.

The motion coordination problem faced by fleets of identical material transfer AGVs is described in Problem 1. This part of the conflict-free scheduling and routing problem is isolated in Figure 1.3 in Chapter 1. A convenient representation of the workspace W is a connected graph of waypoint paths that avoid the static obstacles in the environment. The starting position $\tau_i(0)$ of AGV i (for all $i = 1, 2, ..., N$), and the mission locations

M_j.

Problem 1 Given a number N of AGVs operating in a workspace $W \in \mathbb{R}^2$, what trajectory $\tau_i(t) \in W$ should each one follow to complete all missions in the minimum total AGV-Time. A mission takes place at position $M_j \in W$. A collision-free distance $||\tau_i(t) - \tau_k(t)|| > d_{min}$ must be maintained between every pair (i, k) of AGVs at all times $t \geq 0$.

Numerous ways of dividing and conquering this problem have been developed. One of the most common is the conflict-free routing approach, where the path is discretized and segments are assigned to one vehicle at a time [156]. This is also known as prioritized planning and the priority order has a strong influence on the solution [157]. The global routing and crossing conflict resolution are solved simultaneously, using the same network representation for both.

An alternative where global routing is solved individually, and conflict resolution is performed at the intersection level is described in the breakdown in Chapter 3. In this scheme, Fleet Control assigns a mission j to an available AGV i and finds a suitable path through the network for only one AGV i using Dijkstra or a similar graph shortest path technique. Now each AGV has a path π_i with total length L_i leading to its destination, which can be represented as a list of waypoints. A point in the Cartesian workspace for any distance $s < L_i$ along the path can be found by interpolation of the waypoint list $[\ x \quad y\ ] = \pi_i(s)$. The resolution of the speed profile used for conflict avoidance $v_i(s)$ can be higher than the resolution used to divide the roadmap into a graph to solve global routing. A set of conflict-avoiding speed profiles can be found by solving the local coordination sub-problem Problem 2.

Problem 2 Given a number $N < \bar{N}$ of AGV operating on a set of assigned paths $[\pi_0, ...\pi_N]$ which intersect between s_begin_i and s_end_i in path coordinates, at what speed profile $(v_i(s);\ 0 \leq s \leq L_i)$ should each path be traversed so all paths are completed in minimum AGV-Time and the collision-free distance is maintained at the intersection points.

The 2-layer architecture introduced in Section 4.5 is used to solve Problem 1. This incorporates top-layer zone capacitated routing problem with a lower layer coordination algorithm within a zone around each intersection to solve Subproblem 2. A negotiation-based priority scheme had been previously used to address Subproblem 2. The negotiation itself was shown to be time consuming so the system was improved by the addition of Autonomous Intersection Management (AIM) to avoid the need for negotiation in many cases by Digani et al [158]. The original negotiation scheme remained active at all times as a backup. This led to higher average speeds, crucially reducing the average crossing time by more than the execution time of the optimization algorithm used to solve AIM. The backup system ensured correct behaviour in edge cases: preventing arrivals in the same lane until the intersection was clear. Having a backup gave sufficient certainty for a number of tests with hardware. This motivates the improved AIM methods described in the methods section which are compared to find those which offer the lowest cost optimal solutions along with real-time performance so the backup system is no longer required

The collision avoidance constraints can be restated so they are convex if the crossing order is fixed (FIFO order was tested). Questions remain about the importance of searching alternative crossing orders, compared to the additional computational cost. Better solution time guarantees due to a fixed order could enable AIM to operate without a backup system in more situations.

Preliminary work reported in Chapter 4.8 demonstrated the FIFO heuristic could produce equally good solutions with reduced execution time on an elementary intersection.

In the present chapter, the aim is to compare the performance of an intersection manager with fixed First-In-First-Out (FIFO) crossing order with a published method based on a non-convex optimization with Linear objective and Quadratic constraints ($Quad\_constr$), in which the crossing order can be varied to improve the objective, which is to minimize the sum of crossing time for a set number of participants. The numerical performance of the intersection manager is also important for its use as a real-time safety-critical system, including the scaling with increasing numbers of approaching vehicles.

5.2.1 Hypothesis

The $Quad\_constr$ approach is expected to find solutions with a lower total travel time than $FIFO$. If there are opportunities for allowing sets of vehicles whose paths do not intersect to cross the intersection together, $Quad\_constr$ can change the crossing order to exploit this while $FIFO$ cannot.

The $Quad\_constr$ approach should match $FIFO$ performance if that is the best ordering, or in other cases improve upon it because it is able to search over different orderings. Both approaches make the same simplifying assumptions about the constraints and maximize the same objective by varying the arrival time at the same control waypoints.

The $Quad\_constr$ may take longer to solve. The search space created by the quadratic constraints is non-convex as shown in Section 4.7.1. If this leads to an unpredictable solution time, it may limit the scale at which is can be applied, and possibly rule out its use as a stand-alone collision avoidance method at any scale. The execution time is expected to be shorter and more consistent for the $FIFO$ approach, as the linear program can be solved efficiently with interior point methods.

5.3 Method

5.3.1 Assumptions

The objective is to minimize the total travel time for all AGVs to complete their missions. At this stage, the missions just consist of reaching the end of their assigned path π_i.

The path π_i can be evaluated at a longitudinal distance s to find the position $\xi_i(s) = \pi_i(s)$. Using the variable s it is helpful to divide the path into three parts as shown in Figure 4.9: approach, crossing and departing. The first part AB approaching the conflict where $s < s\_begin$, a part traversing the conflict zone BC where $s\_begin < s < s\_end$, and a part leaving the intersection CD where $s > s\_end$.

A1 No external obstacles can be found on the roadmap.

A2 All vehicles whose paths intersect are known to the manager at execution time and no additional vehicles arrive during the execution of the planned trajectories.

A3 There is a maximum of one vehicle per arrival location, each with a fixed path π_i through the intersection.

A4 The full set of paths through the intersection is known at initialization time.

Assumption A1 ensures that collisions to be avoided by the motion coordination system are those between different AGVs. AGVs are assumed to communicate their longitudinal position s with bounded uncertainty P and modify their trajectory within some tolerance ϵ in response to instructions. This is reasonable as the bounds can be identified by experiment for specific sensors and longitudinal speed control algorithm. External obstacles comprise those which are static and have not been detected and those which are dynamic but do not respond to instructions such as pedestrians and human-operated vehicles. For more on unexpected obstacles please refer to Chapter 3.

Assumption A2 ensures that car-following behaviour does not factor in the following analysis. In the test system arrivals were limited at the source, which counted AGVs already present on any of the associated lane alternatives, and if the lane capacity was exceeded further arrivals were stacked in a vertical queue. In Chapter 6 assumption A3 is relaxed in order to study traffic density on a link between two independently managed intersections.

Similarly, A3 rules out unavoidable collisions caused by two vehicles starting in the same place. In the decomposition described in Chapter 1, shown in Figure 1.3 this function would be the responsibility of the task scheduling module.

5.3.2 Conflict Zone Shape

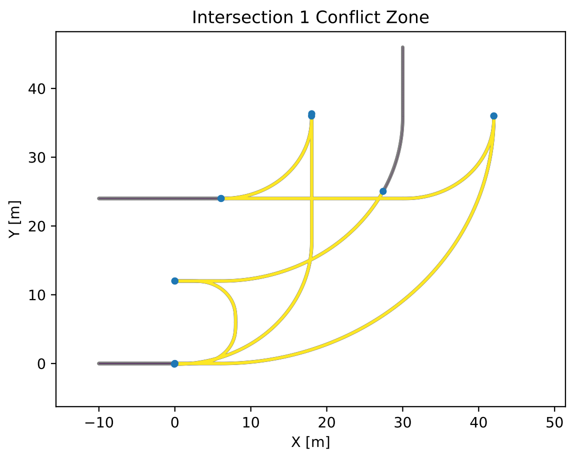


Figure 5.1: Start and end points of each conflict zone component for Intersection 1.

Based on Assumption A4 the shape of the conflict zone can be calculated. This

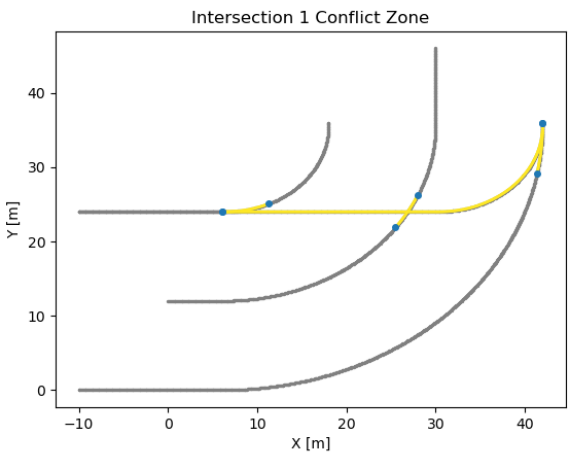


Figure 5.2: Conflict zone on Intersection 1 with arrivals only on four lanes.

is defined in path coordinates by the earliest intersection point with any other path *s_begin* and the latest intersection point *s_end*. As AGVs have some bodywork extending around their control point, the conflict must be expanded by the diameter of the bounding circle. The conflict can be further expanded by $P + \epsilon$ the sum of longitudinal position measurement and control execution uncertainty. In this way, an AGV waiting outside the conflict zone can never collide with one inside.

The full conflict zone for Intersection 1 is shown in Figure 5.1. It is defined by a *s_begin* and *s_end* in path coordinates. If one path intersects with multiple others, the largest *s_end* and smallest *s_begin* are used. This means nearby elementary intersections are merged to form a single conflict zone to be managed by each intersection manager. The division of a site into intersection zones is explained along with an automatic approach for generating the path layout given in [57].

The conflict zone can be smaller depending on the approaching vehicles' intended paths as shown in Figure 5.2. The *end_s* is fixed at the largest value to ensure that the collision constraint is active until a crossing vehicle is out of the way of a potential new arrival on any lane.

5.3.3 Problem Representation

In general, there are differential constraints on the motion of an AGV along a path. For any path and vehicle combination, a maximum forward speed can be selected which keeps the lateral and angular acceleration within the vehicle constraints at two points; the point of peak curvature and the point of peak sharpness. If the lateral acceleration is acceptable at the point of peak curvature it must be acceptable at every other point. Likewise, if the steering rate is acceptable at a given speed at the point of peak sharpness it must be acceptable along the entire path. Details on this transformation are given in Appendix D. For simplicity of exposition in this work, the lower of the two speed limits is taken as the maximum $\bar{v}$ over the entire length so the lateral dynamics can be neglected without loss of generality.

Additional assumptions B1, B2, and B3 allow the optimal speed profile to be specified by the time of arrival at two waypoints. These are located at the beginning s_begin and end s_end of the conflict zone.

B1 Waypoint timing instructions are only sent with sufficient approach distance remaining to adjust speed before reaching the first waypoint $s\_begin - s > min\_conflict\_dist$

B2 The conflict zone must be long enough to reach the second waypoint at the right time without violating acceleration limits $s\_end - s\_begin > min\_conflict\_dist$

B3 All waypoints can be reached with an average speed $v < \bar{v}$

The maximum speed $\bar{v}$ is taken from the specification sheet. The $min\_conflict\_dist$ parameter is calculated based on the maximum acceleration parameter $\bar{a}$ of the AGV model. The acceleration limit is assumed to be symmetrical, so peak deceleration is $-\bar{a}$. The distance required to decelerate at $\bar{a}$ from any valid speed $v < \bar{v}$ is given by Equation 5.1.

$$min\_conflict\_dist = \frac{v^2}{2\bar{a}} \tag{5.1}$$

5.3.4 Timed Waypoint Messages

The speeds were converted into timed waypoints which the AGV controllers must meet as closely as possible. Timed waypoints are expressed as a $[s, t] \in \mathbb{R}^2$ tuple. The discrete waypoint list which specifies the path is used to convert the path distance s into a workspace position $\begin{bmatrix} x & y \end{bmatrix}$.

The timed waypoint conversion takes the constant speed solution for the intersection problem and retains only the safety-critical part at the entry and exit of the conflict zone. As there are two constant speed sections, two timed waypoints for fixed position $[s\_begin, s\_end]$ are calculated according to Equation 5.2.

Provided vehicles reach point s at time t, the average speed constraint is satisfied over that section and there is no risk of a collision. The actual speed profile over the segment can vary as long as it always remains positive.

$$\begin{bmatrix} a \\ d \end{bmatrix} = \begin{bmatrix} \phi_{AB}(s\_begin - s) \\ \phi_{AB}(s\_begin - s) + \phi_{BC}(s\_end - s\_begin) \end{bmatrix} \tag{5.2}$$

Collision avoidance given any monotonic speed profile was confirmed in Chapter 4.1 as long as speed profiles were followed. A second-order model of the motor dynamics followed the instructions from a manager based on average segment speed given in Section 4.5.1. In this chapter, the simulated AGVs follow a simple first-order model. The closed loop individual longitudinal speed controller only compensates for communication delay and is specified in Section 5.3.5.

5.3.5 Individual Longitudinal Control

The individual controllers are simplified a great deal by using a first-order dynamic model. This means that the controller can set the speed directly. The only source of error arises from the communication latency between the intersection manager and the vehicle, taken to be 200ms round trip. This is equal to two control cycles on the vehicle. The individual controller uses the latest position s and time t to meet the next Timed Waypoint $[\hat{s}, \hat{t}]$. The waypoint to target is chosen base on current position, the closest one with both $\hat{s} > s$ and $\hat{t} > t$ is selected. Then the target speed is calculated according to Equation 5.3.

$$u = \frac{\hat{s} - s}{\hat{t} - t} \tag{5.3}$$

This will ensure arrival at the right time, whatever the speed was over the communication delay. The actual speed of the simulated vehicle will be constant (assuming no safety sensor activation) but slightly different from that calculated by the intersection manager.

If the individual controller has not received a Timed waypoint yet, or it has passed the last one it will set the speed to the maximum allowed for this path.

5.3.6 Objective

The objective to minimize the total travel time given by Equation 5.4 is identical to that in chapter 4.1. It is linear in terms of the reciprocal speed vector $\boldsymbol{\phi} \in \mathbb{R}^{2N}$, which has up to two elements per AGV. One for the approach if it has not yet been passed and one for crossing conflict zone so $\boldsymbol{\phi}_i = [\phi_{AB}, \phi_{BC}]$. The segment lengths for the approach and the conflict are contained in distance vector $\boldsymbol{L}$ so $\boldsymbol{\delta}_i = [\delta_{AB}, \delta_{BC}]$.

$$\min_{\phi} \boldsymbol{J}_T = \boldsymbol{\delta}^T \boldsymbol{\phi} \tag{5.4}$$

The condition $j > i$ in Equation 5.4 indicates that the number of constraints varies with the number of vehicles n as $\frac{n(n-1)}{2}$. This corresponds to one constraint between each pair of approaching AGVs.

5.3.7 Collision Avoidance

5.3.8 Linear FIFO constraints

$$\begin{gathered} \min_{\phi} \boldsymbol{J}_T = \boldsymbol{\delta}^T \boldsymbol{\phi} \\ \text{subject to} \\ \boldsymbol{\phi} > \phi_{min} \\ \boldsymbol{Q_i}\boldsymbol{\phi} > \mathbf{0} \quad \forall i \in [1, n-1] \end{gathered} \tag{5.5}$$

The FIFO constraints are linear in the reciprocal speed parameters and there are $n-1$ constraints. The constraints are introduced in Chapter 4.6.6. The parameters must be sorted by distance to the decision point, so one constraint is with the leading vehicle and one is with the following vehicle. The first vehicle to arrive is always unconstrained and will travel at the bounded speed.

5.3.9 Non-convex Quadratic Constraints

$$\begin{gathered} \min_{\phi} \boldsymbol{J}_T = \boldsymbol{\delta}^T \boldsymbol{\phi} \\ \text{subject to} \\ \boldsymbol{\phi} > \phi_{min} \\ \boldsymbol{\phi}^T \boldsymbol{H_{ij}} \boldsymbol{\phi} > \mathbf{0} \quad \forall i, j \in [1, 2, ..., n] \quad \text{with } j > i \end{gathered} \tag{5.6}$$

The quadratic constraints are introduced in Section 4.6.7 and allow ordering flexibilty. As shown in Chapter 4.1 the constraints are non-convex, so convex solvers may be vulnerable to suboptimal local minima.

5.3.10 Solver Details

The $FIFO$ speeds were found with scipy.optimize.linprog and the vehicles ordered with the Python 3.7.6 function sorted.

The *Quad_constr* speeds were found with scipy.optimize.minimize given the analytical Jacobian and Hessian. Two algorithms were tested:$SLSQP$ stands for the Sequential Least Squares Quadratic Programming algorithm and *trust_constr* indicates the trust-region constrained algorithm [154][159].

The default settings for each algorithm are shown in Table 5.1. The value of $gtol$ is the threshold the gradient of the objective must be below for termination with the *trust_constr* algorithm. The equivalent parameter for the $slsqp$ algorithm is $ftol$. The threshold for termination is that the change in function value must be less than this.

Only the *trustr_constr* algorithm makes use of the analytical Hessian of the objective and constraints. Both methods were provided with an analytical Jacobian of the objective and constraints.

As inequality constraints are present *trust_constr* will introduce slack variables and solve a sequence of equality-constrained barrier problems [160].

constraint form	$FIFO$	$Quad$	$Quad$
algorithm	$linprog$	$slsqp$	$trust_constr$
init	-	$\boldsymbol{\phi\_min}$	$\boldsymbol{\phi\_min}$
max_iter	1000	120	240
ftol	-	10^{-9}	-
gtol	-	-	10^{-9}

Table 5.1: Solver Settings.

For execution time comparisons all tests ran on a 1.6Ghz Intel(R) Core(TM) i5-8250U Quad Core laptop with 16 GB main memory.

5.4 Numerical Experiment

The two intersection managers were implemented in Python with an identical messaging interface to communicate with a collection of AGV controllers with simplified dynamics.

The first test is inspired by the published results in [2], showing the improvement of intersection management similar to Quad_constr over decentralized negotiation. The tests took place on a close duplicate of three realistic intersection layouts with fixed direction lanes. The reproduction of intersection 4 is shown in Figure 5.3.

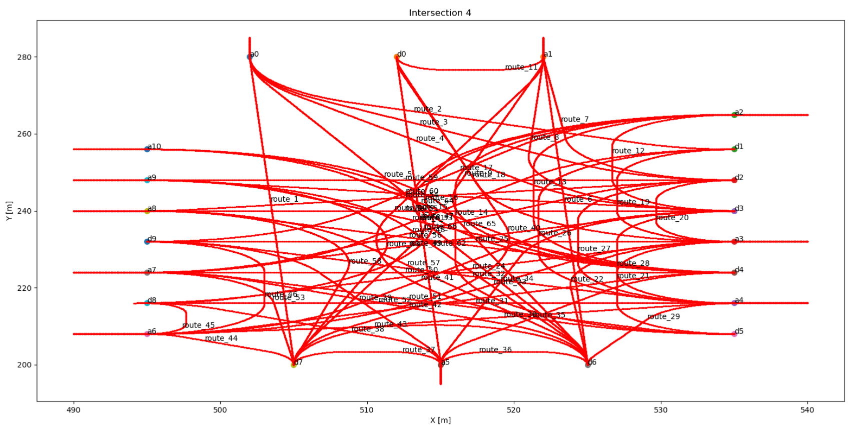


Figure 5.3: Intersection 4 detailed waypoints using Reeds-Shepp paths [3] to join the 66 $a - d$ pairs.

Over ten runs, one AGV was initialized at each entrance, and assigned a random exit. The clearing time $Tclearing$ for the last vehicle to leave the intersection is recorded in each run. Depending on the number of paths included in the intersection this gave a different level of coverage of the total number of combinations.

The execution time to calculate the optimal waypoint times for every vehicle was calculated at the start $Texec$. The worst delay for a single AGV compared to its free flow time to cross the intersection $Twait$ is also recorded. The parameter $Twait$ is defined somewhat differently to Digani 2019 [2]. In that paper, the metric captures the time spent stationary so the optimal method has zero waiting time in every case. As both the tested methods in the present study are optimal in that sense, it makes sense to use the delay-based definition given in Equation 5.7 to compare them. The delay-based $Twait$ is still averaged over different AGVs in each run, and the worst run average is recorded for that layout.

$$Twait = crossing\_time - \frac{PathLength}{max\_speed} \tag{5.7}$$

In the table below $Twait$ is calculated according to Equation 5.7 so it is likely to be non-zero, even for optimal methods. It represents the delay caused by the intersection compared to the free-flow speed. In some cases, it is slightly negative because vehicles can exceed the maximum speed to meet the waypoints given by the intersection manager.

In all simulation runs, an assertion statement at every iteration ensured each pair of AGVs were separated by more than $2W_v$.

5.4.1 Note on Combinations

The 66 $a-d$ pairs with 11 arrival nodes are partially explored over 10 runs, with 2 - 11 arrivals assigned random routes. The binomial formula for combinations without ordering is appropriate as all arrivals appear at the same time in the experiment. For a subset p of the total number of routes r the number of combinations ${}^rC_p = \frac{r!}{p!(r-p)!}$. For a small layout like Intersection 1 with 6 $a-d$ pairs and 3 entrances, there are 20 combinations of 3 routes and 15 combinations of two routes, so 10 runs should provide over 50% coverage.

For a large layout like Intersection 4 ${}^rC_p = \frac{66!}{11!(55)!} = 1,074,082,795,968$ or over 1 trillion combinations! It must be noted that some of these combinations are invalid according to the assumption of one AGV per arrival node, but as an order of magnitude estimate the calculation is valid. For a test on Intersection 4 including only two vehicles at a time leads to ${}^rC_p = \frac{66!}{2!(64)!} = 2145$ combinations. There are inevitably many traffic scenarios which were not captured in the ten runs tested. The validity of the comparative results hinges on the fact that the same random seed was used to generate the exact same set of scenarios for each intersection control algorithm.

5.5 Numerical Results

To examine the success of the FIFO heuristic on a complex intersection we begin with the results for Intersection 4 with 11 entrances. The waypoints were reproduced by

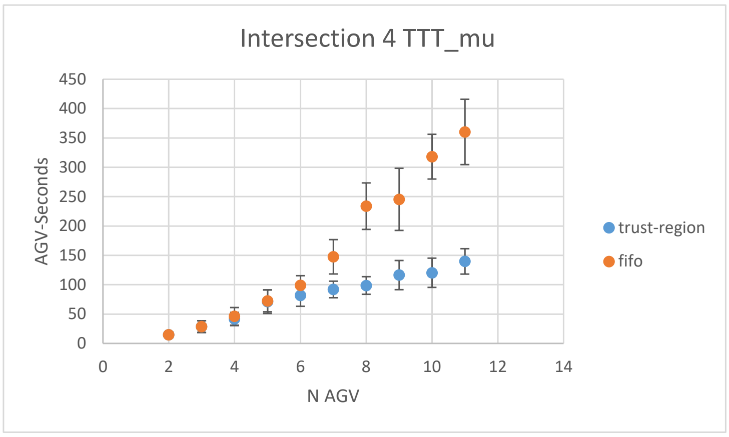


Figure 5.4: Total Travel Time for 2-11 Vehicles on Intersection 4. Trust Region Algorithm. 1 standard deviation error bars.

estimating the origin and destination points and finding new Reeds-Shepp paths [3] between them. The resulting layout is shown in Figure 5.3.

In figure 5.4 the total travel time produced by $FIFO$ ordering is similar to the solution to the quadratic constraints for 2 - 5 AGV. For 6 - 11 AGV the quadratic constraints permit much lower travel times. The greater the number of AGVs the greater the advantage, with travel times for 11 vehicles reduced from over 350 AGV-Seconds to less than 150 AGV-Seconds. This corresponds to a doubling of average crossing speed. The error bars show +/- 1 standard deviation over the 10 trials with random entraces and routes drawn from a uniform distribution.

Execution time results in Figure 5.5 are for the $trust-region$ algorithm. The FIFO method is much faster to compute, as expected. The computation time increases roughly linearly with the number of vehicles, reaching 65 seconds for 11 AGV. This may be too slow for a responsive real-time controller, but the search can be terminated earlier and the constraints will be satisfied, so no risk of collision only a slightly reduced crossing speed.

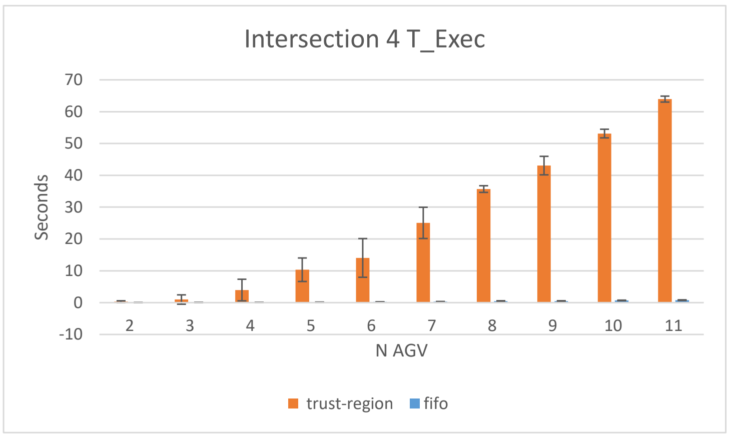


Figure 5.5: Execution time for 2-11 vehicles compared on Intersection 4. Trust region Algorithm. 1 standard deviation error bars.

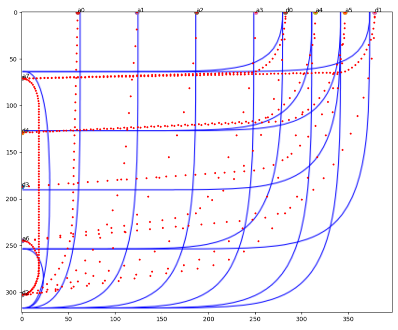


Figure 5.6: Intersection 3 Layout including eight arrival nodes. Reproduced waypoints as tested shown as red dots.

5.5.1 Intersection 3

To establish how performance varies with intersection geometry, Intersection 3 in Figure 5.6 has up to 8 entrances, and Intersection 4 shown in Figure 5.3 has up to 12. Lower numbers of vehicles were tested on the full-size intersection by leaving a fixed number of arrival points empty. The empty start locations were selected from a uniform distribution over 10 repeated runs.

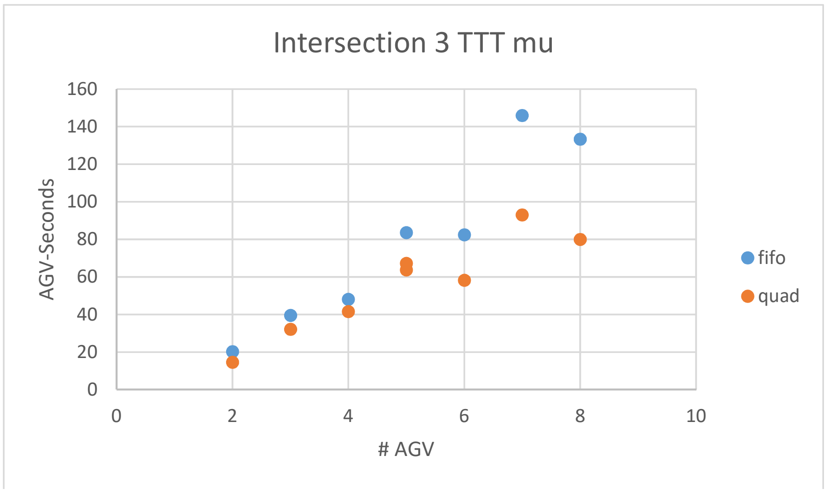


Figure 5.7: Intersection 3 Total Travel Time. Quadratic constraints solved with SLSQP algorithm.

The total travel time TTT in Figure 5.7 is consistently improved by the $Quad\_constr$ method, and the degree of improvement is greater the busier the intersection becomes.

The travel time is improved further by switching to the trust-region algorithm, as shown in Figure 5.8. On the same layout, with the same arrivals, the trust-region algorithm takes much longer to converge but finds a superior ordering which almost halves the travel time objective.

Comparing the $Tworst\_wait$ should reveal the degree of individual sacrifice required to reach the social optimum. As before, $Quad\_constr$ is consistently able to improve on the $FIFO$ solution by choosing a different order with the benefits increasing as the intersection gets busier shown in figure 5.9. This shows that the wrong ordering leads to individual delays in higher traffic, even with a completely fair policy. Missed opportunities for non-conflicting flows to progress at the same time, lead to a longer

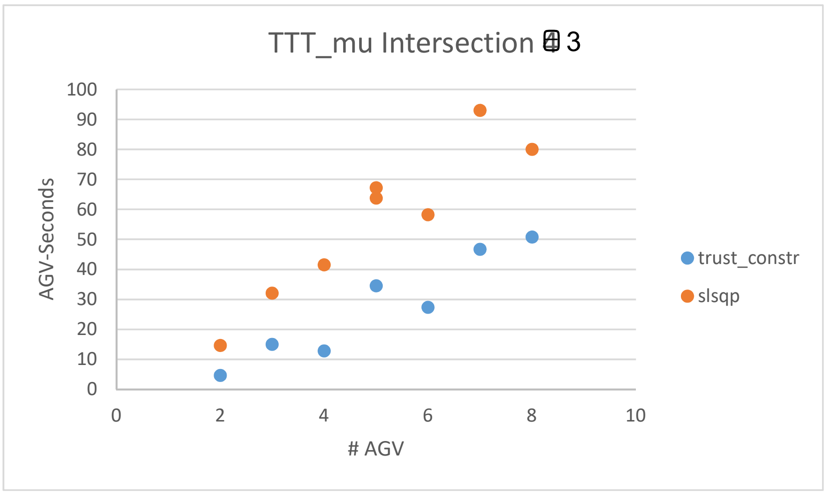


Figure 5.8: Solution quality with the two algorithms can be compared based on the TTT objective.

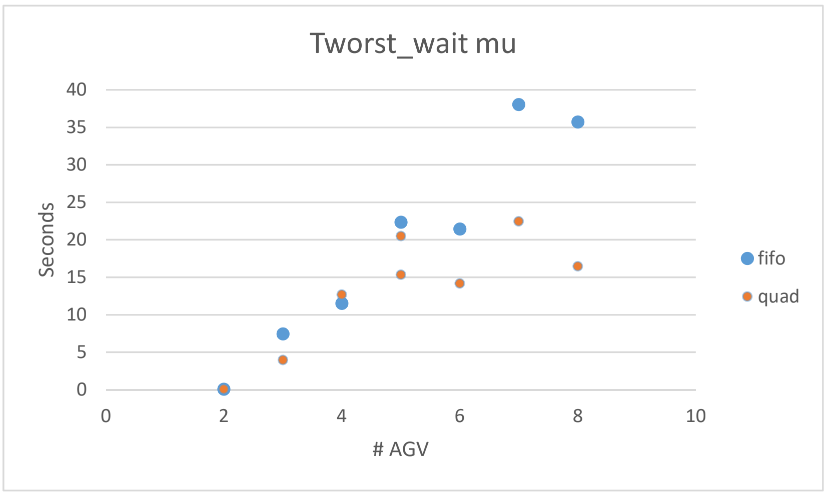


Figure 5.9: Worst waiting time experienced by a single AGV, averaged over 10 runs across Intersection 3.

wait for whoever crosses last.

5.5.2 Intersection 1 Results in Detail

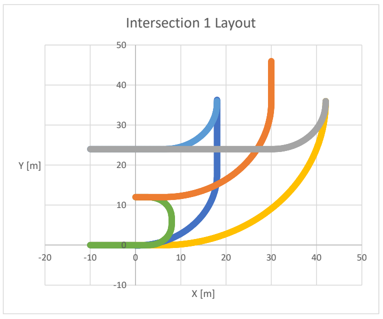


Figure 5.10: Waypoints with 10cm spacing for each of the six paths through Intersection 1.

Intersection 1 only has three arrival nodes as shown in Figure 5.10, so only two and three arrivals can be tested while keeping one vehicle per lane and no car following (Assumption A3).

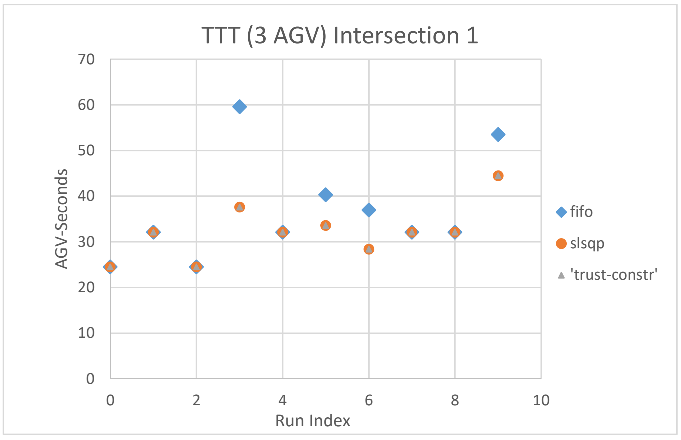


Figure 5.11: Total Simulated Travel Time in AGV-Time units over 10 runs with 3 AGVs on random paths.

Results for three vehicles in Figure 5.11 suggest *Quad_constr* is able to find a better solution at the cost of a higher execution time. With few vehicles interacting the difference is minimal. Only run index 3, 5, 6 and 9 could be improved with a change in ordering, the others were the same with either method.

The execution time over the 10 runs with different path choices shown in Figure 5.12 increases markedly with the switch to quadratic constraints. The choice of algorithm makes little difference in this case, but trust-constr takes slightly longer in every run except for run 0 and run 3.

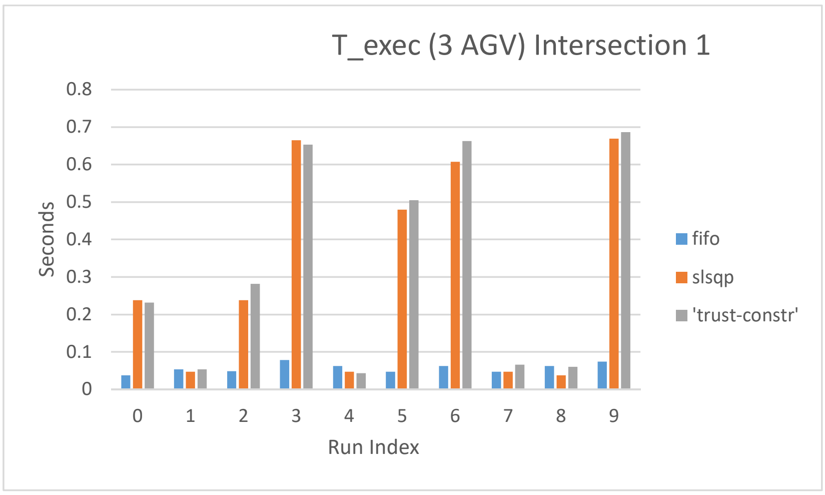


Figure 5.12: Execution Time over 10 runs with 3 AGVs on random paths.

Detail of Improved Ordering with 3 AGVs

To understand better the mechanism for improving travel time by reordering, it is instructive to examine three vehicles in Intersection 1. Here, FIFO was consistently beaten by Quadratic Constraints even with a small number of vehicles. The reordering benefit is detectable with 3 AGVs, although it became more significant for higher numbers of crossings on larger intersections.

The distance-time trace for each vehicle can be plotted together, where distance is measured as chainage along the path, starting at the arrival point. This means the conflict start distance is different for each AGV. The location is marked by a circle

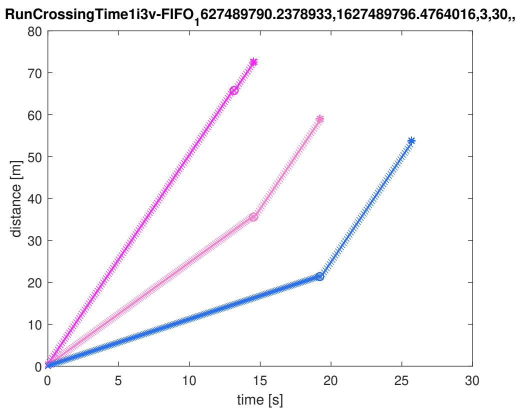


Figure 5.13: Distance along path over time for each AGV on Intersection 1, Run 3 with FIFO method.

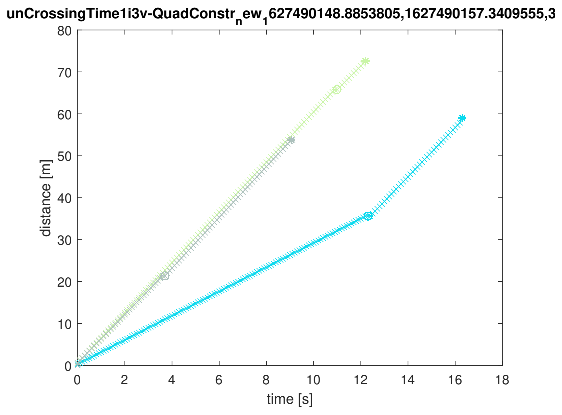


Figure 5.14: Distance along path over time for each AGV on Intersection 1, Run 3 with Quad (trust-constr) method.

plotted at the start and the star plotted at the end in Figure 5.13. Position at a given time step is represented by a cross of one colour for each AGV. The timing instruction from the intersection manager is indicated by the position of the circle and start on the time axis. The use of longitudinal speed control described in Section 5.3.5 ensures the deadlines are met exactly in this simulation which did not include any disturbances.

The distance-time plot of the same scenario with three vehicles under quad control in Figure 5.14 looks very different. The fact that two routes are not in direct conflict is exploited to allow two vehicles to proceed at full speed. The third vehicle needs to slow down a little less as less time is used for the second one to get clear.

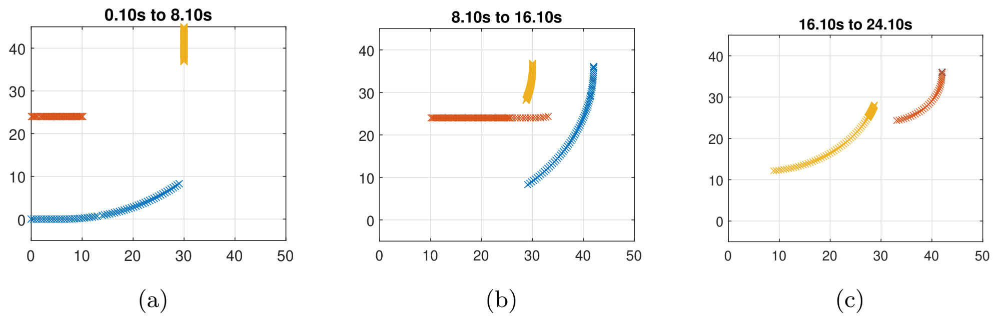


Figure 5.15: Positions of three AGV over the specified time window under FIFO ordering.

Shown on a plan view in Figure 5.15 the red and yellow traces are seen to travel very slowly to give the blue trace (closest to the decision point) time to get clear.

By contrast, the traces in Figure 5.16 both blue and yellow proceed at full speed as they don't conflict. Once they are out of the way the red trace can proceed, saving 10 seconds overall. With larger numbers of vehicles on larger intersection such opportunities to batch non-conflicting movements are more numerous. This explains the increasing advantage of the Quadratic constraints for larger numbers of participants.

5.5.3 Alternative Algorithm for Minimizing Linear Objective with Quadratic Constraints

With the settings in Table 5.1 the *slsqp* algorithm terminated faster than the *trust_constr* method with $gtol = 1e^{-6}$ in Figure 5.4. The value of $gtol$ is the threshold the gradient of the objective must be below for termination.

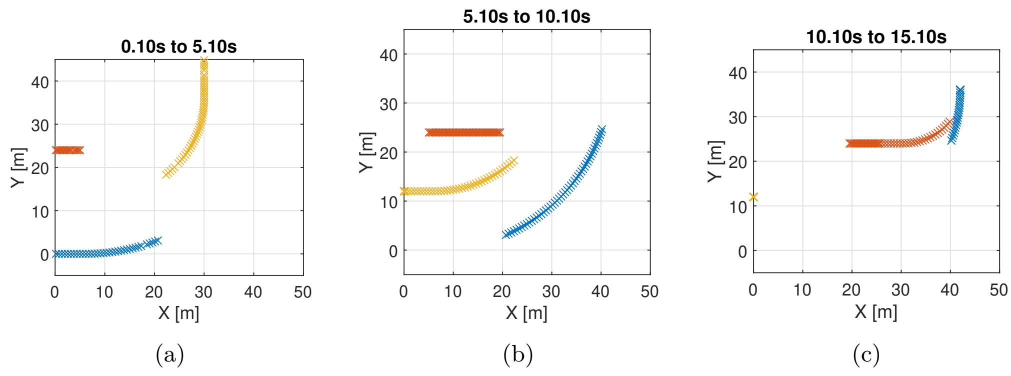


Figure 5.16: Positions of three AGV over the specified time window under Quad management with flexible ordering.

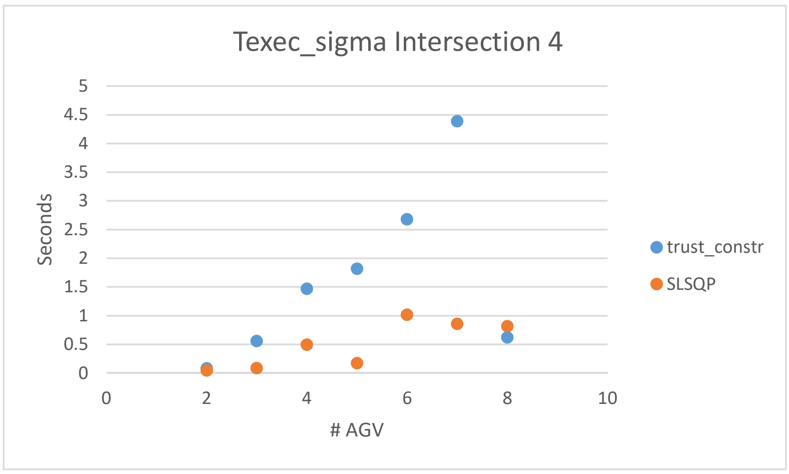


Figure 5.17: Standard Deviation of execution time for the Quadratic constraints.

There is a problem when n_agv=8, which leads to the number of iterations exceeding the limit, which was set to n_iter=600. This explains the very low standard deviation over the 10 runs with different paths seen in Figure 5.17. The time taken to complete n_iter iterations is very consistent. Despite early termination, the solution meets the constraints.

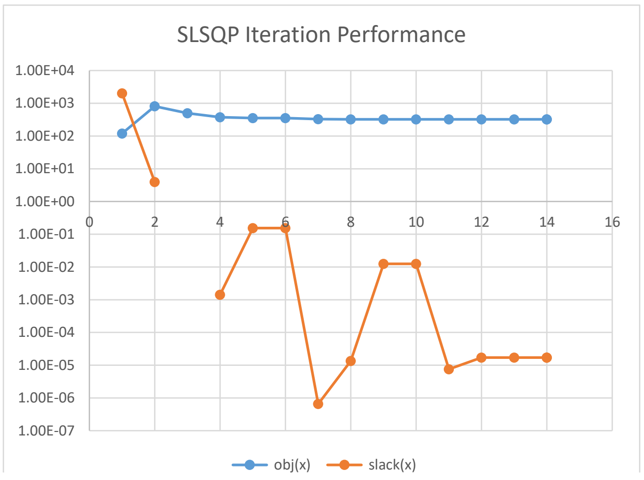


Figure 5.18: SLSQP slack (sum of 2-norms of constraint violations)

Comparing the iteration performance on a particular problem, intersection 3 with 8 arrivals in Figure 5.18 shows that *slsqp* did not reach convergence. The iteration terminated due to “positive derivative in direction of linesearch.” This indicates the initial guess was infeasible, and the unconstrained minimum is also infeasible. In this instance, the solution is quite close to the minimum and the constraint violation is small so it just seems like the method has solved the problem very quickly.

Many more iterations are needed before termination with the $trust-constr$ method shown in Figure 5.19. The constraints are satisfied to the specified tolerance in this case, making the execution time comparison unfair.

5.6 Summary and Links to next chapter

Across all three intersection layouts tested, the quadratic constraints formulation consistently found a crossing solution with higher average speeds. The negative impact of suboptimal $FIFO$ ordering on the travel time is large, outweighing the benefit of a

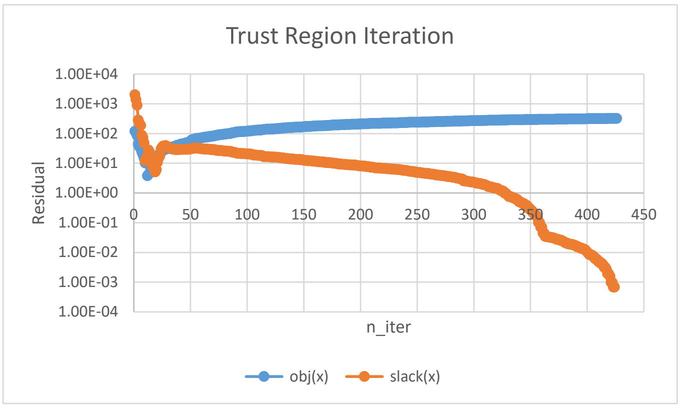


Figure 5.19: Trust-constr slack on Intersection 3 with 8 arrivals

faster execution time.

The two different algorithms for minimizing the travel time subject to quadratic constraints performed very differently. The *trust_constr* algorithm found much lower cost solutions, but the execution time ran to multiple seconds and even minutes in some cases with 11 vehicles. The other algorithm *slsqp* did not converge in many cases. The return value indicated the unconstrained minimum lay outside the feasible region, and because the initial guess did not meet the constraints the algorithm could not proceed. In future work initializing the algorithm with the $FIFO$ solution, so it has a feasible start point could improve the results. If the *slsqp* algorithm was able to determine it had converged, it would be the superior choice as the constraint error drops in fewer iterations.

A more realistic intersection comprising curved paths with numerous elementary path crossings has demonstrated the advantage of the quadratic constraints. In order to utilize multiple intersection managers to avoid conflict in a decentralized way as envisaged in Section 1, questions remain about the downstream effects of traffic at one intersection on the next. In the next chapter, multiple nearby intersections will be solved independently and the need for I2I messaging tested.

Chapter 6

Car-following Constraints to Link Multiple Independent Intersection Managers

6.1 Chapter Abstract

In this chapter the boundary between two independently managed intersection zones is investigated in simulation. Latency between vehicles reporting their position and the reference speed commands reaching them lead to potential collisions even with guaranteed safety in isolation. Additional constraints on the problem for each zone are presented and solved to find a set of speeds which is optimal within each zone and collision free across multiple zones with a given latency.

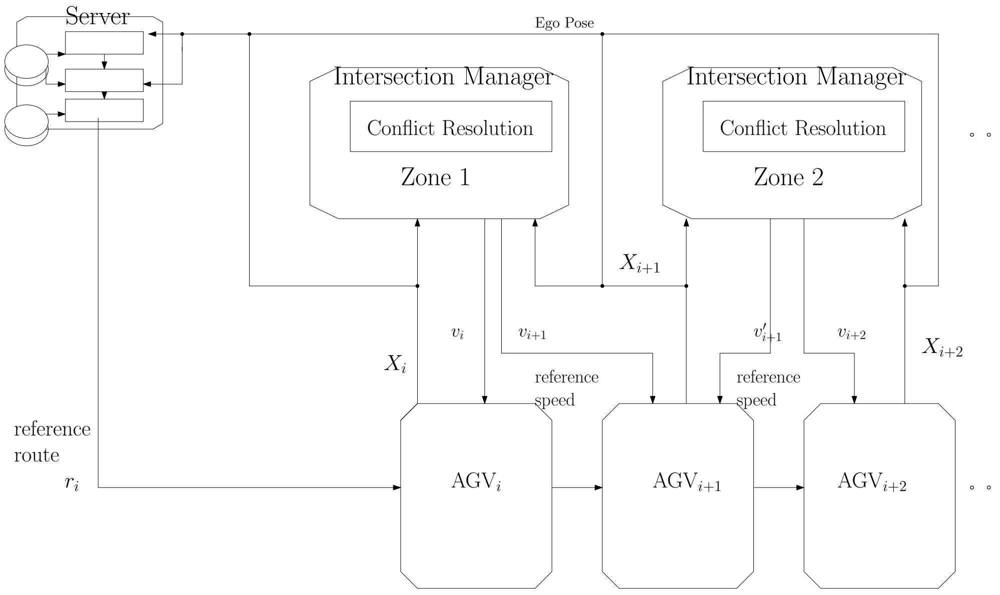


Figure 6.1: Overarching activity diagram showing the shared responsibility of multiple Intersection Managers.

6.2 Introduction

Literature reviewed in Section 2.4 reveals that most AIM designs have been modelled in isolation [98]. This means the optimal methods do not consider departing traffic, which may need to slow for the next intersection. This may be important in urban roads and also indoor material handling sites, where the intersections are quite close together. The two-layer control scheme proposed by Digani et al gives the zone around each intersection a fixed capacity determined by experiment [2]. The intersection management layer finds maximal conflict-free crossing speeds for each AGV within each zone. The estimated zone capacity is used in the routing layer to route AGV away from zones closer to their capacity and thereby prevent queue spillback becoming important. In this scheme routes across the intersection are reserved for one vehicle, so another will not follow in the same lane until it is clear. This reservation system is common in industrial automation controls but may lead to lower capacity than car-following.

In [100], car-following is performed independently by approaching agents alongside AIM sending average speed commands. In [108], Liu et al formulate the centralized problem and introduce a consensus method in which agents repeatedly compute their arrival time at each conflict. Both works select an approach speed limit for each AGV - leaving local controllers free to travel slower. This differs from the formulation in Section 6 where the manager selects the average speed at which an AGV should approach and cross. This has a big impact on the speed autonomy of the AGVs, and the complexity of their individual speed control. A speed limit allows the individual AGV to choose a lower speed without danger of collision. With a specified average speed, if the local controller slows down for any reason, it must speed up before the end of the timed segment to remain inside its collision-free window. In a case of congestion this will not be possible, so the intersection level solution must incorporate some type of car-following constraints.

The Intersection Management Algorithms FIFO and Quad_constr used to resolve crossing conflicts in a locally optimal way in Chapter 5 can be used to control multiple nearby intersections with a simple distance based message filter. These methods both assume intersections are spaced far apart so they neglect downstream effects. At what spacing and traffic level does this assumption break down? A messaging scheme to break the problem down is presented, and the coupling effects studied.

Next, a modification to the AIM control problem representation is described to resolve the safety issue. Results are included for two intersections at varying distances, and the limitations of the specified average speed control breakdown are evaluated.

6.3 Aim/Objective

- Find the intersection spacing and traffic level at which intersection coupling effects can no longer be neglected.
- Describe an approach for independent managers to ensure a safe handover of

control between nearby intersections.

6.4 Problem Setup

The first problem to specify is finding the centralized, site-optimal collision-free longitudinal speed profiles $v_i(s)$ for N AGVs with fixed paths $\begin{bmatrix} x & y \end{bmatrix} = \pi_i(s)$. The site-wide minimum time conflict-free problem is defined in Equation 6.1. This is the same as Problem 1 in Section 5.2 with discrete conflict zones.

$$\begin{gathered} \min_{v_i} \textstyle\sum_{i=1}^{N} \int_{s=0}^{L_i} v_i(s) ds \\ \text{subject to} \\ \text{outside kth discrete conflict zone at the same time} \\ \pi_i(s_i(t)) \notin Z_k \text{ if } \pi_j(s_j(t)) \in Z_k \\ \forall i, j \in \{1, 2, ..., N\}] \ \forall t > 0 \\ \forall k \in \{1, 2, ..., P\} \end{gathered} \tag{6.1}$$

Here s is the longitudinal distance along the path from the start at $[x_S \quad y_S] = \pi_i(0)$ towards the goal at $\begin{bmatrix} x_G & y_G \end{bmatrix} = \pi_i(L_i)$. The path $\pi_i(s)$ may be a list of connected edges from a roadmap graph. This means the position can be expressed relative to the start of the current lane rather than from the absolute start position of the AGV.

In addition to the discrete conflict zones Z_k, each speed profile $\pi_i(s_i)$ can be discretized to a sequence of average speeds ν_k over M discrete segments.

V2I messaging with multiple nearby intersections is modelled by message passing of plan messages from every AGV to every AIM. A plan message must contain the path of travel $\pi_i(s)$, the current position $s(t)$ and the longitudinal speed $\dot{s}$ to be used to compensate for the transmission delay.

The AIM filters by approach lane and ignores plans which do not intersect the zone Z_k under its control. Plans which do intersect Z_k at some point are only included in the local problem if there are no intervening intersections. This means the approach distance a for one lane is determined by the distance between the end of the nearest other conflict zone on an 'IN' lane. Based on the latest plan message from every vehicle which is not excluded, the intersection local optimization problem can be constructed as defined in Equation 6.1 including only a subset N_k of vehicles which passed the filter. This is the same as the global problem only with a single conflict zone Z_k.

The average speed command from the next intersection is forwarded by departing vehicles on receipt. This implementation detail might be quite important, it is used to avoid direct communication between notionally independent managers. The manager only has access to the downstream average speed after a delay of time delayed $2T_{aim}$. The downstream speed is set freely by the downstream intersection manager and used as a constraint by the one upstream. This is the point of control handover which motivates the modification.

A third waypoint has been added that changes the problem setup slightly. It is needed in our simple "average speed at three sections" control model so constraints on the handover point can affect the approach speed parameter, which are set by AIM.

6.4.1 Messaging Partition

The V2I messaging scheme for multiple intersections is a slight generalisation of that for an isolated intersection described in Chapter 4.5.1. Each AGV sends its individual crossing plan for the intersection it is approaching to every manager which is in range.

The plan messages are filtered by the receiving manager, based on the lane and the current position along the lane. With reference to Figure 6.2, the plan is either approaching a, crossing c or departing section d. Every AGV in the approaching section can be commanded to meet any feasible average speed, for a sequence of distance segments across the intersection.

Existing managers from Chapter 5 specified the average speed on a minimum number of segments to reduce the computational burden, which was two segments for an isolated intersection.

6.4.2 Multi Agent First Order Dynamics

Each AGV within the simulation is represented as a particle with state $\boldsymbol{x}(t) = [x(t), v(t)]$. The on-board controller can set the value of scalar control input u. The dynamics are shown in Equation 6.2.

$$\dot{x} = u \tag{6.2}$$

6.4.3 Delay Compensating Longitudinal Speed Control

The on-board speed control uses a constant acceleration model to choose a feed-forward speed at each time step to match the specified average speed before the end of the segment, even if the waypoint timing was delayed in transmission. This was introduced in Section 5.3.5 and defined by Equation 5.3.

6.5 Method: Transition Constraint with Car-following

Focusing on the transition point, I propose a Triple Waypoint interface to smooth the handover and avoid using the safety system in normal operation. The essence of the system is a transition region where a vehicle departing intersection 1 on the lane toward intersection 2 sends plan messages to both. The handover of control takes place according to predefined distances configured in the intersection managers shown in Figure 6.2. A vehicle will attempt to meet any constraints which it receives so adjacent managers must both be configured to avoid conflicting instructions. The cut off at which an intersection will ignore a vehicle on an incoming lane is s^i_{inc}. The conflict zone starts at s^i_{begin} and ends at s^i_{end} marked by a cross. The three speeds sent in the Triple Waypoint constrain motion until s^i_{exc}. But the next intersection will take control as soon as the vehicle is out of the conflict zone.

The third section runs until s_c after the conflict is over so vehicles with limited acceleration can reach the specified time. Limited acceleration is not enforced in the first-order dynamic model described in Section 6.4.2 but the constraint is included to give more realistic crossing speeds. Departing vehicles along the entire transition lane send the same motion plan to both intersection managers.

As soon as the vehicle is past the conflict zone (second waypoint) of this intersection, the manager of the next intersection has full control authority. The waypoint location is fixed by intersection geometry and known to both managers. The speed sent by the manager with control authority will also be passed back to the previous intersection to be used as a constraint on the speed of approaching vehicles.

6.5.1 Transition Lane Constraint

The timing instruction for arrival at the next intersection conflict zone $[s_n, t_n]$ is used to generate a constraint at the previous intersection using the average speed. $v_d = (s(t) - s_n)/(t - t_n)$

The average speed is referred to a fixed point at the minimum distance from the end of the conflict zone $s_E + min\_distance$ at time $(s_E + min\_distance - s(t))/(t - t_n)$.

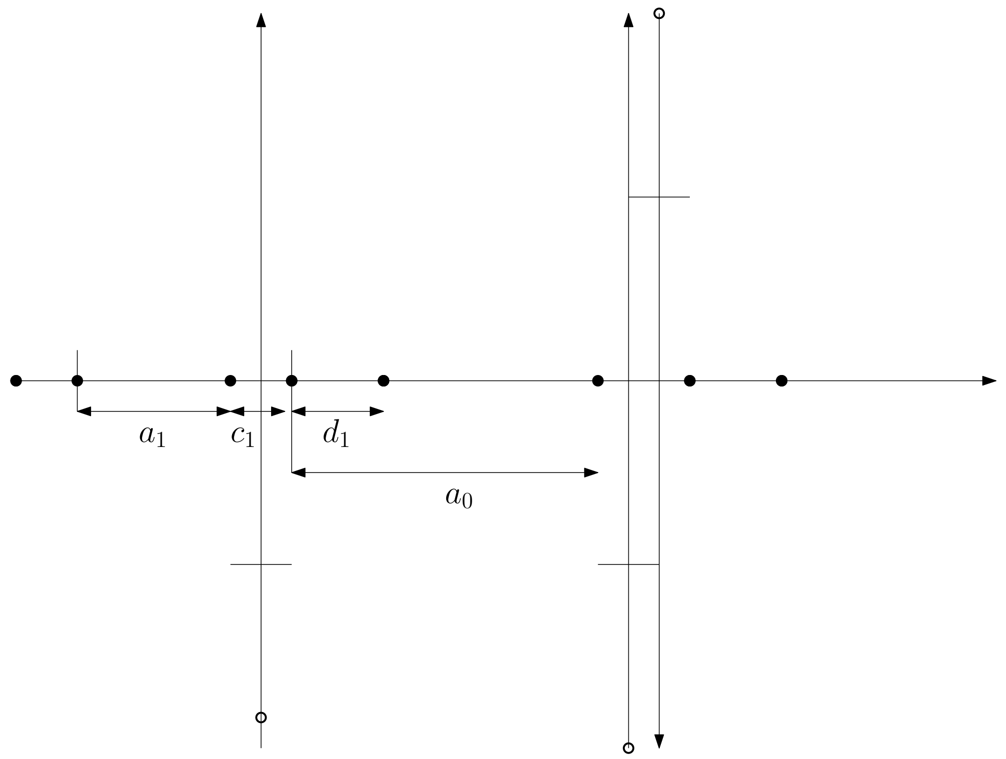


Figure 6.2: Cut off points where control switches from one intersection to the next.

The constraint is applied to the first vehicle in each lane.

$$-\boldsymbol{K\phi} \leq -\boldsymbol{l} == \boldsymbol{K\phi} > \boldsymbol{l} \tag{6.3}$$

where $\boldsymbol{K} \in \mathbb{R}^{(n \times 3n)}$ is a block diagonal matrix assembled with blocks compatible with $\boldsymbol{\phi_i} \in \mathbb{R}^{(3\times1)}$ so $\boldsymbol{K_i\phi_i} > \boldsymbol{l_i}$. The number of vehicles crossing is m, each being sent three timed waypoints.

$$\boldsymbol{l_i} = t_n - \frac{s_n - (s_E + min\_distance)}{v_d} = t_n - \frac{a_0}{v_d} \tag{6.4}$$

and

$$\boldsymbol{K_i} = \begin{bmatrix} s_n - s(t) \\ s_E - s_S \\ min\_distance \end{bmatrix}^T = \begin{bmatrix} s_n - s(t) \\ c_1 \\ d_1 \end{bmatrix}^T \tag{6.5}$$

6.5.2 Car Following Constraints

There are three reasons for applying additional constraints to the intersection level solution to take account of congestion downstream.

- To avoid jerky motion on the departure lane.
- To ensure speed commands always exclude a car-following collision, even with vehicles which are receiving instructions from another intersection manager.
- To ensure flow balancing, so if a transition lane has high traffic, those approaching this intersection intending to depart on that lane will be slowed.

There are different ways to add I2I comms between intersections. It is particularly neat to use the vehicles in motion as proxies to forward the constraints to avoid an additional messaging channel.

The constraints should be enforced at every sample point with an associated free variable. They ensure the front bumper (control point $s_F + L_B/2$) of the follower vehicle arrives at each sample point τ_h seconds later than the rear bumper of the leader (control point s_L - $L_B/2$).

Considering only the first point on arrival at the conflict s_{AB} the constraint can be written as follows:

$$\begin{bmatrix} s^F_{AB} - L_B/2 & 0 & -s^L_{AB} - L_B/2 & 0 \end{bmatrix} \begin{bmatrix} \phi^F_{AB} \\ \phi^F_{BC} \\ \phi^L_{AB} \\ \phi^F_{BC} \end{bmatrix} > \tau_h \tag{6.6}$$

Here s^F_{AB} is the distance to the start of the conflict zone for the follower and s^L_{AB} is the distance to the start of the conflict zone for the leader. ϕ^L_{AB} is the reciprocal speed along this distance for the leader. Once the instructions are committed to the leader, so it is no longer controllable but will arrive at time T_L Equation 6.6 can be rewritten

$$\begin{bmatrix} s^F_{AB} - L_B/2 & 0 \end{bmatrix} \begin{bmatrix} \phi^F_{AB} \\ \phi^F_{BC} \end{bmatrix} > T_L + \tau_h \tag{6.7}$$

6.6 Numerical Experiment

This test aims to characterise the performance of the nearby intersections based on the total occupancy at the arrival time of each additional AGV. The exit time is recorded, and used to calculate the average speed across the intersection. The hypothesis is that the more vehicles are present, the slower a new vehicle will be able to cross.

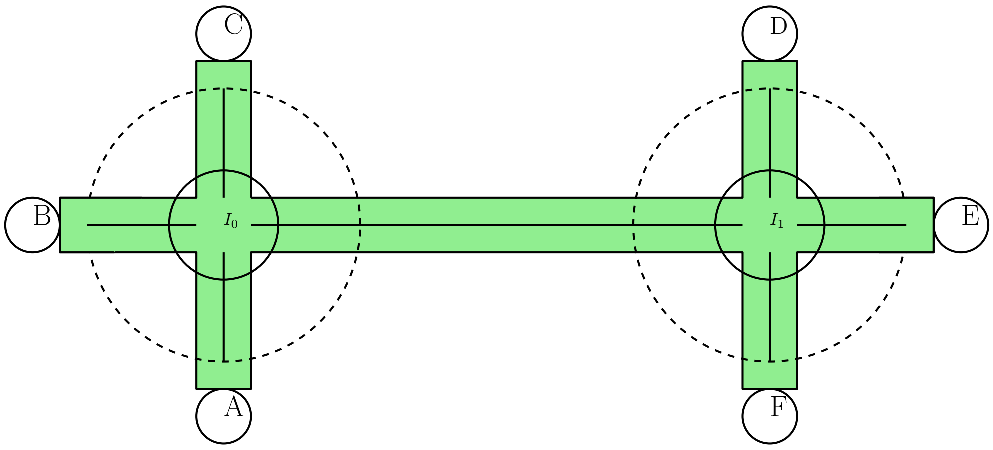


Figure 6.3: Path layout with two nearby intersections.

The test takes place on the synthetic path layout shown in Figure 6.3. This comprises six location nodes labelled A-F which a mobile robot may need to visit. This could represent a small pick-pack-and-ship warehouse, where nodes A-C are in the picking aisles and nodes D-F are packing stations where customer orders are assembled.

I believe this path layout is the minimum number of intersection connections which can be generalized to comment about queue spillback on larger scale problems. Some tests were performed with a single directed transition lane from I_0 towards I_1. The solid circle around the intersection represents the conflict zone which the manager is protecting access to and the larger dashed circle represents the communication range.

Both methods from Chapter 4 *Quad* and *FIFO* have been tested running independently on two nearby intersections, with the transition handled as described in 6.4.1. Additional car-following constraints detailed in Section 6.5.2 have been applied to the FIFO method to create a $FIFO\_carf$ controller which should smooth traffic flow when the intersections are close together.

6.6.1 Illustration of the coupled intersection interaction problem

The trajectories of every vehicle which crossed the intersection are shown with one dot per time interval (0.1s) in a unique colour in the composite distance-time graph in Figure 6.4. The conflict zone is shown by the command from the intersection manager which is a solid line in the same colour. For vehicles travelling along the x-axis the entry and exit timed waypoints are marked with an 'x'. For vehicle travelling on the cross lanes the entry and exit timed waypoints are marked with a triangle. The first conflict in any lane starts at 12.5m and ends at 17.5m. The trajectories are seen to meet the waypoint commands in their colour and the waypoint commands never overlap, so one vehicle must depart before the next arrives.

In Figure 6.4, all the cross-lane trajectories travelling in the y-axis end at 30m, where they reach the sink at the end of the cross lane. Those on the x-axis proceed toward a second conflict zone from 36 - 41m. Cross traffic on the second intersection has not been offset from cross traffic on the first in this diagram so the interactions at the second conflict zone are more difficult to make out.

The intersection manager runs every 0.5s in the simulation, using only historic data reported by the simulated agents over the preceding 0.1 seconds. This leads to a delay between local measurement, centralized demand calculation and local demand execution by on-board control of 0.7 seconds which is not fully compensated. Despite the extremely simplified first order dynamic model, position extrapolation by one time step and delay compensating local speed control, the lag results in consistent failure of every vehicle to meet its time specification by the next manager update. This can be seen in the distance-time plot for all vehicles in Figure 6.5. The cross marks joined by a solid diagonal lines connecting the entry $\{s_a, t_a\}$ to the exit $\{s_d, t_d\}$ with the same colour as the dashed vehicle trajectory are seen shifting rightwards as the centralized manager calculates new later crossing times because the originals were missed. Extrapolating the reported position by three time steps using the reported speed before solving the optimization could help with this. It requires a change to the messaging interface to add the current speed as well as the position along the current curve.

The circle in Figure 6.4 and Figure 6.5 highlights where the two vehicle trajectories are separated by less than the minimum separation W_v. The red trace slows down to meet the timed waypoint sent by the intersection manager. The grey trace represents the midpoint of a vehicle following in the same lane. The gradients of the two dotted

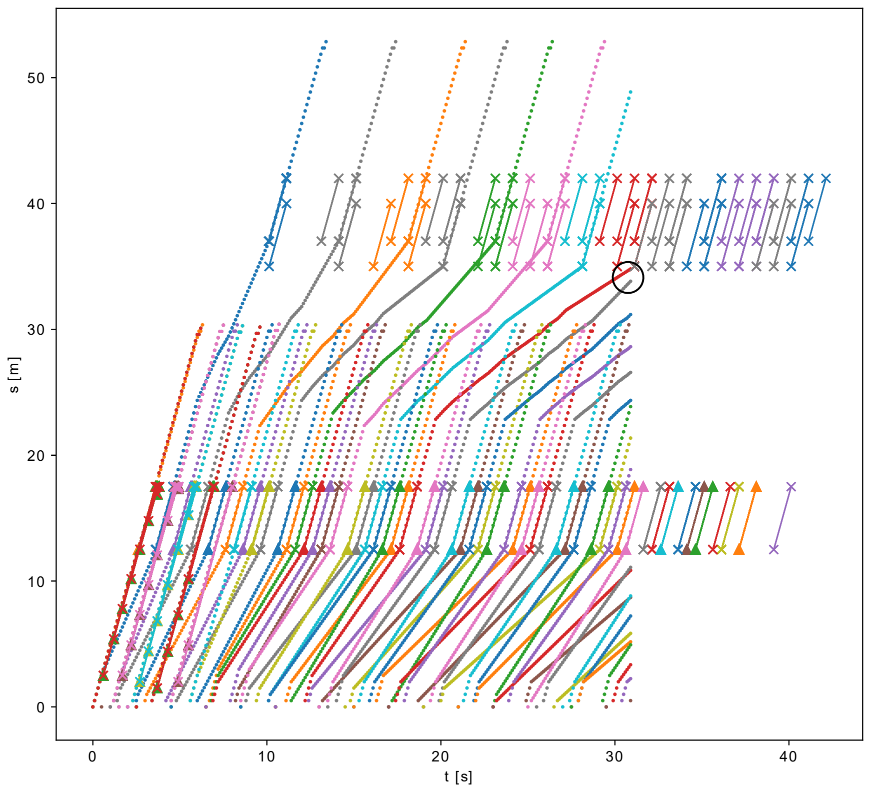


Figure 6.4: Overview of the transition effect distance-time plot for all vehicles with unmodified $FIFO$ management. Trajectories shown by one dot per time step with a colour for each vehicle. Timed waypoint instructions entry and exit times marked by an 'x' connected by a solid line for lane BE, by a triangle for lane AC and FD. Two vehicle traces which come closer than W_v leading to test termination circled.

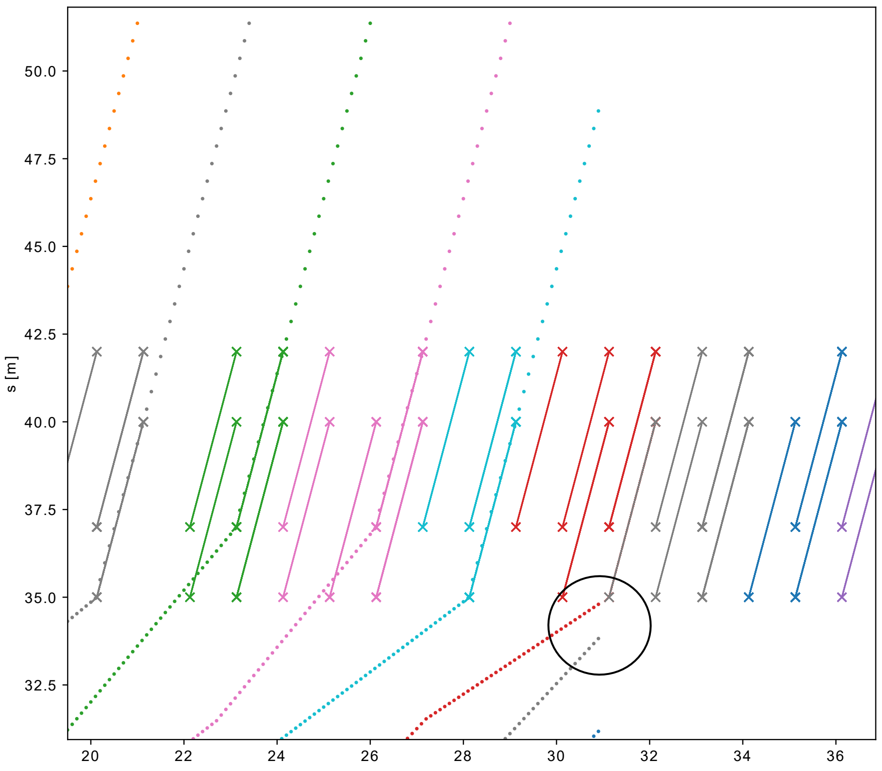


Figure 6.5: Detail of the transition effect distance-time plot for all vehicles with unmodified $FIFO$ management. Trajectories shown by one dot per time step with a colour for each vehicle. Timed waypoint instructions entry and exit times marked by an 'x' connected by a solid line. Two vehicle traces which come closer than W_v leading to test termination circled.

lines show that the grey vehicle is travelling faster and catching up. At t=31.0 the grey vehicle catches up with the red vehicle; the distance between the two falls below W_v bringing the test to an end.

Inspection of the solid lines connected by 'x's of the same colour which represent the timed waypoint instructions sent to the vehicles shows that changing successive instructions were sent to both of these vehicles. A full set of timed waypoints for all vehicles is recalculated by the manager every 0.5 seconds. Ideally, the instructions would be constant over time, and only account for new arrivals in the management zone. If vehicles are not on track to meet their assigned waypoints for any reason, the new instructions calculated will compensate by delaying subsequent vehicles. The waypoints at the second intersection between $35 - 42.5$m are seen to change not only in time but also in position. This is an implementation problem related to two lanes crossing at the second intersection. When a vehicle is approaching on the first cross lane the conflict is closer, when the conflicting vehicle is on the other lane the conflict shifts further away by the 2m distance between the lanes. When the conflict location changes, the timings are adjusted accordingly.

What this illustrates with regard to the interaction problem is that a manager which can maintain safe spacing of all vehicles at a single intersection, may break down when extended to multiple intersections as described in section Section 6.4.1. The critical added value of tackling the interaction problem is to maintain safe spacing across an unlimited number of safe intersection managers.

6.6.2 All-lanes Average Speed

The impact of the car-following constraints can be compared based on the average speed across the first two segments of each lane: the segment approaching the conflict and the segment crossing the conflict. In this test a single direction transition is included. Arrivals appear at node B heading to node E along the transition lane and crossing both intersection conflict zones. At I_0 vehicles arrive at node A heading to C. At I_1 arrivals appear at node D heading to F and also arrive at F heading to D, giving I_1 twice the length of approach lane to I_0 in total. No turning movements are included. Each arrival node has a burst pattern which spawns five vehicles at 1 second spacing, followed by five at 2 second spacing.

The crossing speed shown in Figure 6.6 relates to the left intersection, labelled I_0 in

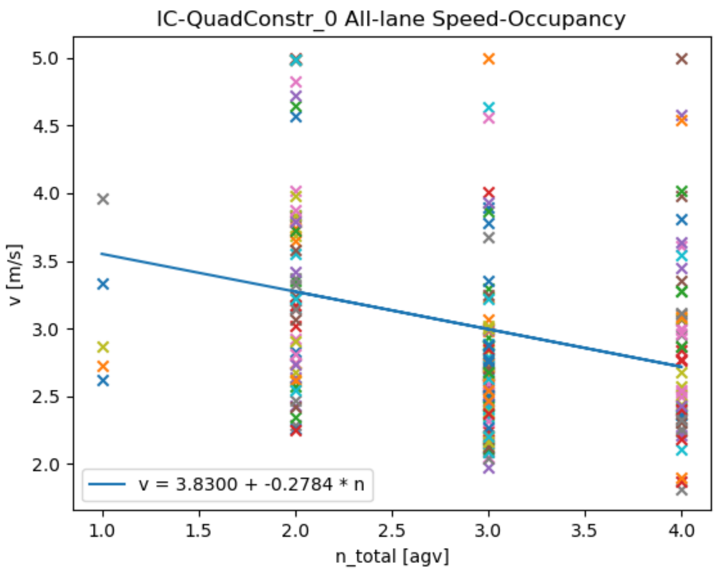


Figure 6.6: Average speed for each vehicle against the total number of vehicles across all lanes approaching I_0 for the *Quad_Constr* Method.

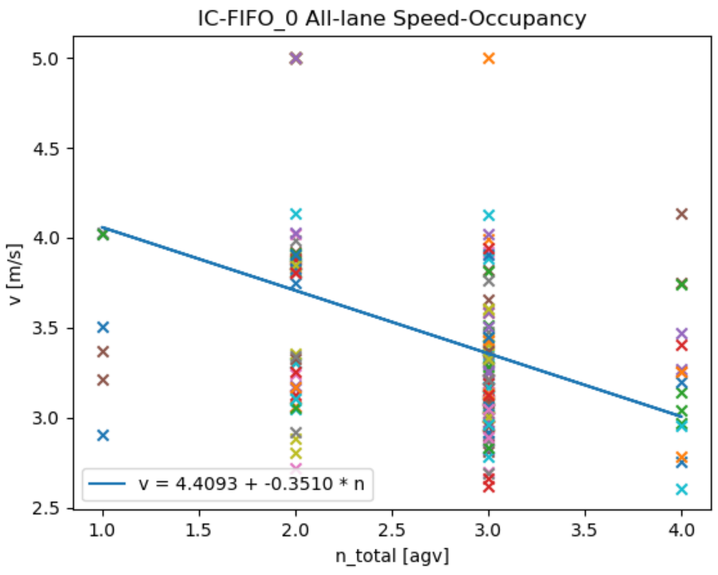


Figure 6.7: Average speed for each vehicle against the total number of vehicles across all lanes approaching I_0 for the *FIFO* Method.

Figure 6.3. With Quadratic Constraints the order is flexible which can improve speed if a better ordering can be found than FIFO but the search time is increased. There is no increase in average speed between Figure 6.6 and Figure 6.7 because the conflict zone shape is fixed, the vehicles are identical and I_0 is an elementary intersection with only single lanes. On more complex geometries it is likely the improvement would be substantial as they were for the isolated multi-lane intersections tested in Chapter 5. Figure 6.6 sows a weak linear trend suggesting a reduction in crossing speed of 0.2784 m/s for each additional vehicle in the communication zone at arrival. The highest number approaching vehicles seen in simulation at I_0 is four. The length of lane included in the problem is $a = 12.5$ m for the horizontal approach lane, $c = 5.0$ m to cross the horizontal conflict zone and then $a = 12.5$ m for the vertical cross lane plus $c = 5.0$m for the vertical conflict crossing. Maximum capacity would be reached if one arrived every second on all lanes. With a 5 m/s arrival speed, this would pack them each 5 m apart so the communication area has space for six approaching plus one crossing at most. The trend line $3.83 - 0.2784n$ suggests the resulting speed would be $v = 1.8812$ m/s for $n = 7$.

The relationship between crossing speed and intersection occupancy for the $FIFO$ constraints is shown in Figure 6.7. With the same arrival pattern the largest recorded occupancy is the same at four vehicles. The linear fit indicates a higher average crossing speed for the first vehicle, although this may be an artefact as there are few solutions with only a single occupant. The average speed is reduced faster under $FIFO$ management losing 0.3510 m/s for each additional vehicle to reach about the same average speed for four vehicles.

A noisy negative linear trend is also quite a good fit to the intersection on the right I_1 as shown in Figure 6.8. As both cross lanes were active and the transition distance is set to 20m the total approach length is $20 + 15 + 15 = 50$ metres. With the same burst arrival pattern on each lane, this time up to 8 vehicles approached at the same time, compared to a theoretical maximum of 10 vehicles. The capacity is higher but the speed reduction due to one additional vehicle is steeper for this geometry so the expected speed for the last vehicle based on $n = 9$ from the trend line $v = 4.7512-0.4859n$ would be $v = 0.3457$ m/s.

The speed-occupancy curve for the right-hand intersection I_1 with the $FIFO$ method is shown in Figure 6.9. Again a maximum of $n = 8$ was seen in simulation and the

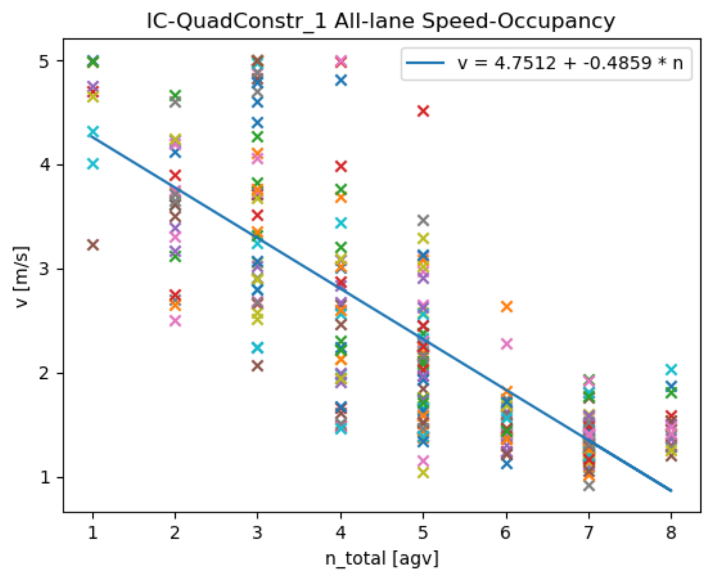


Figure 6.8: Average speed for each vehicle against the total number of vehicles across all lanes approaching I_1 for the *Quad_Constr* Method.

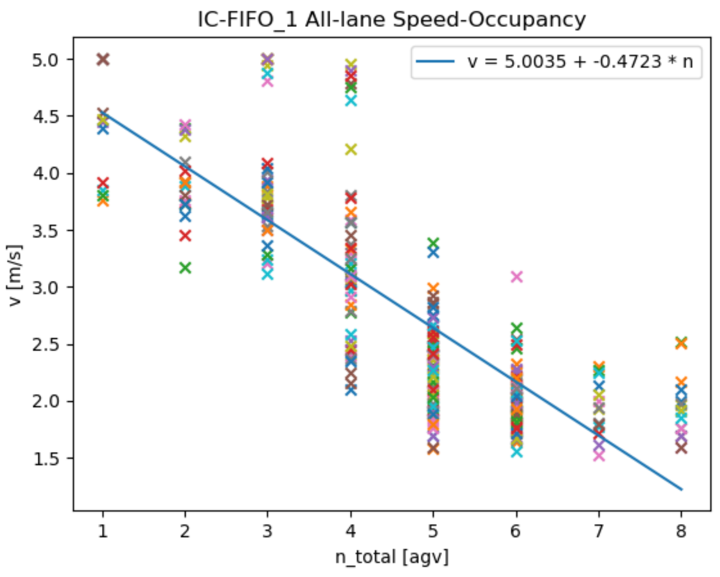


Figure 6.9: Average speed for each vehicle against the total number of vehicles across all lanes approaching I_1 for the *FIFO* Method.

trend is very similar, leading to a projected speed of $v = 0.7528$ for the tenth vehicle, arriving when occupancy $n = 9$. The fact that speeds on the busy intersection are slightly higher in I_1 for $FIFO$ is difficult to explain, but consistent with the results for I_0. The overall trend is similar crossing time performance between the isolated methods $FIFO$ and $Quad$, but much longer execution time for $Quad$.

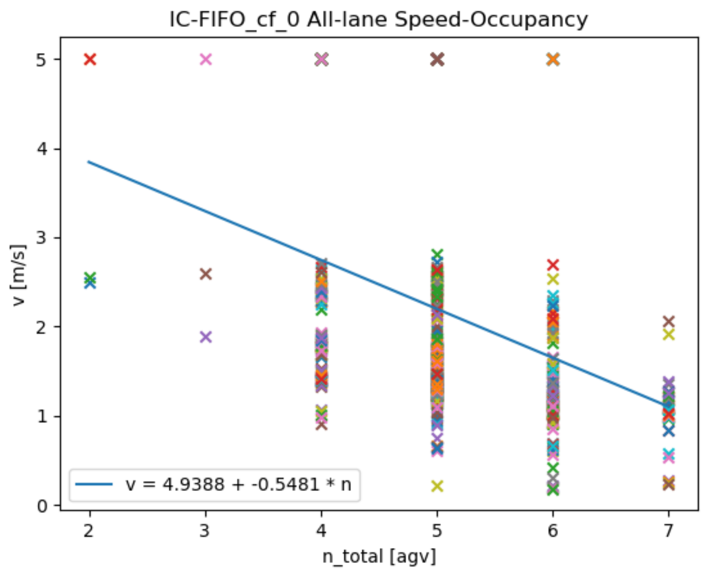


Figure 6.10: Average speed for each vehicle against the total number of vehicles in the intersection zone around I_0 at arrival time for the $FIFO\_carf$ method.

The relationship between crossing speed and occupancy on I_0 shown in Figure 6.10 with $FIFO\_carf$ management where additional departure constraints and car-following constraints are applied to ensure safe handover from one intersection to the next. This method is successful at enabling operation at higher traffic densities as up to eight vehicles approaching could be simulated without a minimum distance violation. The theoretical capacity of I_0 is actually reached in simulation before the minimum distance is violated. This is double the capacity that either of the isolated methods could safely direct on the same intersection I_0.

On the right-hand intersection I_1 shown in Figure 6.11 with more lanes inside the communication range (total length 65m: capacity 13), the highest capacity observed was 13. This is much improved over the 8 vehicles recorded by either of the isolated methods. Looking at the linear fit in Figure 6.11 compared to that in Figure 6.9, the

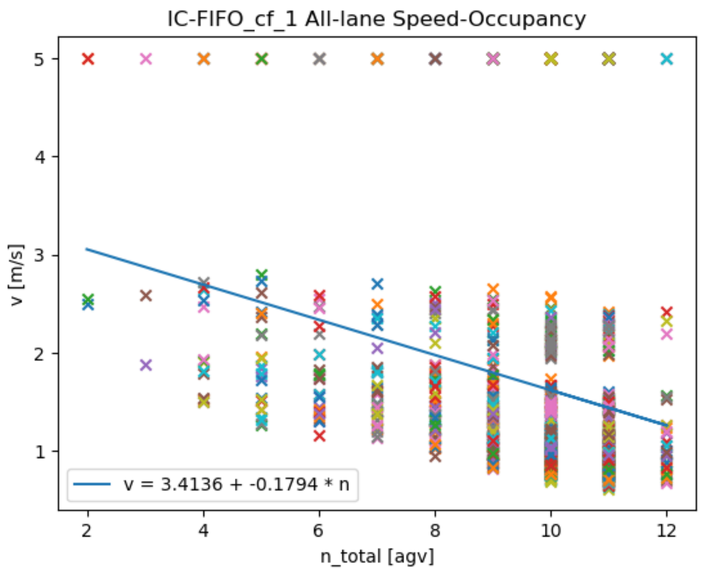


Figure 6.11: Average speed for each vehicle against the total number of vehicles in the intersection zone around I_1 at arrival time for the $FIFO\_carf$ method.

lower intercept implies that due to the constraints, speeds are a bit lower when traffic is light. The less steep gradient means that only 0.1794 m/s average speed is lost for each additional participant, so the speeds at high density are comparable with either unmodified method $FIFO$ or $Quad$. The linear fit $v = 3.4136 - 0.1794n$ indicates that a 13th vehicle would be able to cross at 1.0814 m/s.

Most importantly, higher densities could cross safely in simulation.

Comparing the execution time, the difference was very noticeable on repeated runs at two orders of magnitude. $Quad\_constr$ averaged 0.3950 seconds on I_0 with 1vps burst arrival rate, compared to 0.0051 seconds for $FIFO\_carf$. Trust region constrained method was used to solve $Quad\_constr$ to a gradient tolerance $gtol : 1e - 06$. Interior point method was used to solve $FIFO\_carf$. Both tests were performed on a Quad Core Intel (R) Core(TM) i5-8250U 1.8GHz with 16GB of RAM. Only $FIFO$ or $FIFO\_carf$ are suitable for real-time operation with a manager update rate of 0.5 seconds without further optimization, as the worst case execution time for $Quad\_constr$ on $I_0$was 0.8268 seconds, exceeding the deadline with 8 vehicles included in the problem.

6.6.3 Effect of Intersection Spacing

The next test varies the distance between the two intersections. As in the previous section, new arrivals appear at node A, B, D and F so the transition lane is traversed only in one direction from I_0 to I_1. Arrivals at B will cross both intersections via the transition lane and depart at E. Arrivals crossing from both D and F reduce the capacity available at I_1 and cause traffic to build up on the transition lane as the simulation goes on. Beyond a certain intersection spacing, the intersections should be effectively independent. One deterministic arrival pattern was tested. This had a burst length of 5 vehicles, and a lull length of 5 vehicles with a burst rate of 1 vehicle per second and a lull of 0.25 vehicles per second. These rates were chosen to match the rate limiter on the source which ensures 5 metres of clearance before a new vehicle can arrive.

The first intersection manager is responsible for the intersection I_0 of lane $\overrightarrow{BE}$ with a single lane $\overrightarrow{AC}$. The second manager is responsible for the intersection I_1 at a distance $d + 25$, which is crossed by two lanes $\overrightarrow{DF}$ and $\overrightarrow{FD}$ with the same arrival pattern. The second intersection with more traffic would be expected to permit reduced crossing speed. As the approach speed along lane $s1$ decreases vehicles departing from $I1$ will catch up to them, leading to a congestion spillback effect. The growth of the moving jam will depend on the capacity difference between the two intersections, which is kept the same in all runs. As a result the time to control failure should be linear in the length of the approach lane. The shortest lane which allowed 30 vehicles to cross is 15 metres. 10 runs were performed with the same traffic pattern, starting with $a_1 = 30$ m and decreasing a_1 by 3m each time to cover the range $a_1 = [30, 27, ..., 6, 3, 0]$ m.

Figure 6.12 shows the minimum separation over each run. The run ended prematurely if the separation fell below 1 m at any point, as this indicates a collision which the constraints are designed to avoid. $FIFO\_carf$ maintains a car following distance of 2 m on most runs, dropping to a still safe 1.1 m for $a_1 = 18$ m and failing to find a solution for $a_1 = 6$m. The minimum separation for the $Quad$ method was less than $FIFO\_carf$ on every run, except for $a_1 = 18$ m. The minimum separation dropped below 1 m for $a_0 = 6$ m and $a_0 = 30$m leading to early run termination. This is surprising as it does not reveal a cut-off distance beyond which the vehicles can operate in isolation. This is because the uncontrolled handover from one intersection manager to the next is a collision risk at all transition distances. The minimum distance vari-

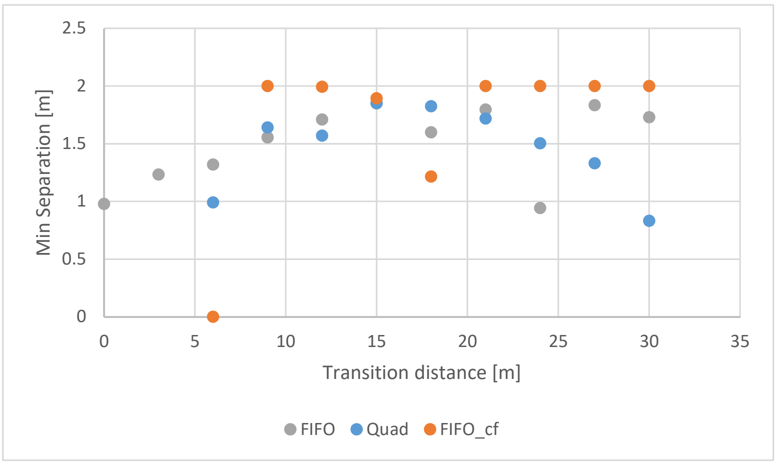


Figure 6.12: Variation of number of smallest gap between two AGVs over the entire run with transition lane length.

ation in Figure 6.12 as the transition distance increases in steps of 3 m does not show a clear linear trend. It shows the best (most well separated) transition distance for *Quad_constr* is 18 m, with which the minimum separation remained under 2 m. Under *Quad_constr* management, both greater and smaller transition distances led to reduced minimum separation. Critically, minimum spacing fell below the safety distance of 1 m for both a_1 =30 m and a_1 =6 m. The minimum separation over transition distance a_1 for isolated $FIFO$ management shows a different trend, this time with the minimum separation increasing as a_1 increases until $a_1 = 15$ m before levelling off. An isolated drop below the safety threshold is seen when $a_1 = 24$ m. The lack of distance violation at $a_1 = 3$ m is surprising. The $FIFO_carf$ method could not always maintain a safe spacing with a transition lane of this length.

Figure 6.13 shows the number of successful crossings which took place out of a possible 30 at each transition distance with each manager. Neither of the methods operating in isolation ($FIFO$ and *Quad_constr*) consistently maintained the target minimum distances for $a_1 > \underline{a}$. This shows independence between distant intersections does not hold over the distances tested for this geometry.

The partial extension of the messaging interface (to filter out vehicles which are

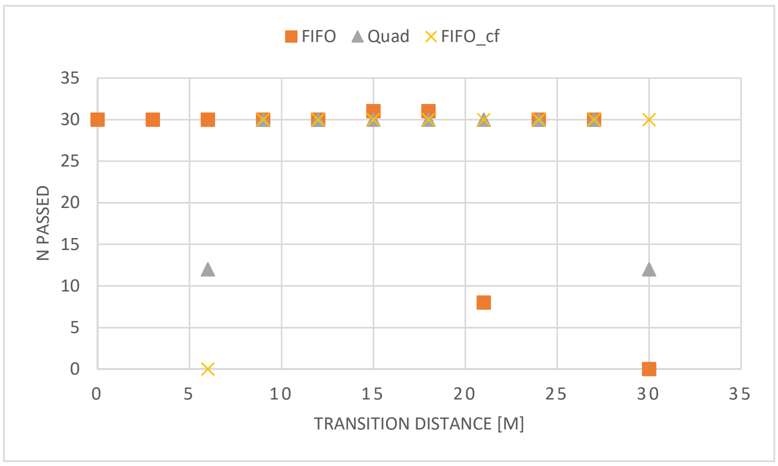


Figure 6.13: Variation of number of safe crossings with the length of the transition lane. The target was 30.

departing) works well for many transition distances, but the unmanaged car-following on the transition lane can lead to rear-end type collisions even with increasing a_1. The studies which assume distant intersection may be treated independently rely on a further assumption of car-following speed choice by AGV. Only the $FIFO\_carf$ approach is collision free at all separations above 6 m, due to the car-following being explicitly managed by the intersection manager.

On layouts with short transition distances 6 metres and below, $FIFO$ method is more successful than $Quad\_constr$. All 30 vehicles crossed without incident with only 3 metres and even a zero-length transition lane. The flexibility to change ordering leads to extra collisions with high occupancy. This is due ot the inherently $FIFO$ nature of car-following, meaning it can be handled implicitly by controlling the time vehicles in the same lane cross the conflict zone. The $Quad\_constr$ method would benefit from additional constraints for car-following so vehicle in the same lane are never reordered, only the order of cross traffic platoons.

6.7 Summary and Further Work

This chapter has investigated issues where collision avoidance by independent managers at nearby intersection breaks down without I2I messaging. Three constraint forms were tested, two independent: *Quad_constr* and $FIFO$; and one aware of neighbouring intersections $FIFO\_carf$.

The *Quad_constr* method attained similar average speeds $FIFO$ as seen in earlier chapters due to its ordering flexibility. This allows vehicles arriving on less busy lanes to skip ahead of those which are held upon a busy cross lane. In these tests the conflict zone shape was fixed at the largest size (including all paths) to ensure late arrivals did not make the optimization infeasible. The long execution time solving the *Quad_constr* problem presents a problem for real-time operation as it exceeded the simulated updated rate of 0.5s on a number of occasions coordinating only four vehicles. $FIFO\_carf$ produced solutions with lower average speeds but being based on a linear program, they were always found in under 17ms for up to seven vehicles.

$FIFO\_carf$ utilized an extended scheme of V2I messaging where departing vehicles report their position to the intersection they have crossed while they follow the instruction of the intersection they are approaching. The average crossing speed of the I2I augmented $FIFO\_carf$ controller was lower than the unmodified $FIFO$ and *Quad_constr* algorithm from the last chapter. This was to be expected as the car following constraints were added to ensure safety on the transition lane, not to increase average speeds. Accordingly $FIFO\_carf$ showed lower average speeds than unmodified $FIFO$, but the transition lane collision provoked by dense traffic at reasonable transition distances $a_1 > 6$m was resolved.

The two methods without I2I communication or car following constraints (FIFO and Quad from the previous chapter) performed quite well at a range of intersection spacings. They were prone to intermittent failures (that is, collisions caught by the assertion) in particular traffic scenarios, which were never exposed in isolation. The feedback effect of shifting timed waypoints from deadlines being missed only appears when multiple intersections are tested due to the additional latency introduced by indirect message passing. The fragility of the system to latency makes a string case for direct I2I messaging to reduce latency. Another way to improve performance is effective delay compensation. This would require the speed to be included in the vehicle state report so it can be used to extrapolate vehicle positions to the time the commands will be received. This wasn't possible with the position only state report interface tested.

Further research is needed into explicit I2I messaging combined with the car-following constraints. Another topic for further research is the transition point between a large merged conflict zone such as those in Chapter 4 and multiple intersections very close-by. At some transition distance the frequency and bandwidth of I2I messaging required to ensure safe operation may require a single agent to manage them.

Chapter 7

Conclusion

The preceding chapters divide the automated vehicle fleet trajectory planning problem between spatial path and longitudinal speed profile, by scale and by distance from a particular intersection. Methods to solve certain sub-problems with limited execution time are introduced in each chapter and numerical results provided to illustrate their performance. The two sub-problems addressed in detail are spline fitting subject to static obstacles constraints and intersection approach speed choice subject to inter-vehicle collision constraints. Both can be expressed as 'mixed' optimization problems because they involve a continuous parameter vector $\boldsymbol{x} \in \mathbb{R}^n$, continuous constraints of the form $\boldsymbol{C}(\boldsymbol{x}) \leq \mathbf{0} \in \mathbb{R}^m$ with combinatorial aspects which could be expressed as additional binary integer constraints $\boldsymbol{C_z}(\boldsymbol{x}) \leq \mathbf{0} \in \mathbb{Z}^p \in \{0,1\}^p$. Different heuristics were employed to handle the combinatorial aspect of each problem. In Chapter 3, a weighted graph construction solves region assignment. In Chapter 4.1, a simple sort based on distance to the conflict-zone start solves crossing order in the $FIFO$ method.

The heuristic part of each solution is essential to generate algorithms which can solve mixed combinatorial problems with limited execution time. This characteristic is important for use in real-time control systems. Due to the use of heuristics for part of the solution, the minimum found by convex optimization will be suboptimal for the original mixed problem. Non-linear convex methods such as Sequential Quadratic Programming can be applied directly to the mixed problem, but in both sub-problems this can lead to long and unpredictable search times. The full continuous constraints for both sub-problems appear to be non-convex. The integer region assignment in Chapter 3 resolves one source of non-convexity, but the application of the constraints at insufficient discrete samples can reintroduce non-convexity. For the ordering problem in Chapter 5 convex methods are used by the *Quad_constr* method to solve for the ordering. The order found leads to much reduced delay than $FIFO$ but the execution time currently exceeds the update period. A better ordering heuristic, based on predicted arrival time at the intersection, instead of distance might be a fruitful direction to explore in future research.

The division of site control into zone local multi-vehicle speed adaptation combined with vehicle-level tactical path adaptation has shown promise under the assumptions made in numerical simulation of three different components. This chapter aims to synthesize the findings across the preceding chapters, and use them to answer the research questions. Next, a discussion of contributions to knowledge from the combination of the studies. Finally, some limitations of the completed studies in answering the research questions are introduced, along with the implications for future research.

7.1 RQ1: How can reactive planning be added to simple mobile robots following clothoid based paths without reducing path quality?

Reactive planning can be included based on clothoid splines with continuous curvature profiles with limited rate of change to match the quality of predefined paths. With the detected obstacle field represented as a collection of possibly overlapping convex obstacle-free regions containing the start and goal pose, the parameters for the spline can be found by sequential quadratic programming.

Parameters for two constant-curvature sections should be included per region, plus a starting segment and an ending segment to reach a specified $[x, y, heading]$ with zero curvature. The multiple-shooting formulation showed a small improvement in execution time on a small environment with three regions. The insertion of straight lines at the beginning and end pose led to the solver becoming trapped in a local minimum, which met the constraints but had worse characteristics in terms of increased overall length and increased peak sharpness. The provision of analytical derivatives reduced the number of iterations but not the execution time, as it took longer to evaluate the exact clothoid integrals within each derivative than performing a cheap finite difference approximation.

These findings show general constrained non-linear optimization algorithms can be effective at finding the best parametrized path through detected obstacles. The execution time and variability mean that this approach is more suited to centralized path replanning. Used in this way, the continuous curvature characteristic allows the paths identified to be used by different vehicles by varying the longitudinal speed.

7.2 RQ2: How can motion conflicts between vehicles at an intersection be resolved independently of scheduling and routing?

Motion conflict between vehicles at an intersection can be resolved by co-operative choice of longitudinal speed. A simple way to do this is for all vehicles to hand over longitudinal speed authority to a single agent, in this case an intersection manager. A messaging interface, where every approaching vehicle reports the path they will follow and their current position, leads to a constrained optimization problem for the safe speeds.

In Chapter 4.1 baseline performance is provided by a binary-semaphore heuristic method which combines First-Come-First-Served crossing order with instructions to stop at the give way line unless holding the semaphore. This method is simple to implement, practically useful and guaranteed to avoid collisions. The ordering is reasonable for identical vehicles at low traffic density, but can lead to long delays.

For an elementary intersection, both optimal methods achieved a major reduction in both crossing time and total energy usage compared to a binary semaphore approach. The semaphore method did not utilize the plans of all approaching vehicles, only their reported position. Between the plan-based methods, there was no advantage to optimizing the order of arrival using a Quadratic constraints formulation. On an elementary intersection, the $FIFO$ method with a fixed First-In-First-Out crossing order was able to find the same safe speeds with a reduced execution time.

In Chapter 5 the baseline performance is provided by the $FIFO$ method. Numerical tests on real-world intersection geometries showed that optimal ordering reduced total travel time by as much as 50%. On an intersection with more complex geometry, the importance of optimizing the order of arrival was clear. In all three realistic maps tested, the increase in execution time from solving the Quadratic constraints was much less than the reduction in crossing time from the improved solution.

The findings show that decentralizing longitudinal speed optimization to individual intersection managers limited the number of participants in each problem and made it possible to solve the passing order problem with convex methods. Numerical tests on an elementary intersection showed a reduction in travel time and a reduction in energy consumption compared to a binary semaphore stop/go approach.

7.3 RQ3: How can independent intersection managers with average speed authority be modified for a safe handover of control between nearby intersections?

Independent intersection managers can be modified for safe handover of control using a predefined cut-off distance to create a transition lane where vehicles follow the speeds commanded by the intersection they are approaching while also reporting their position to the one they are departing. Without any modification, this can lead to sharp accelerations and a dangerous handover at a spacing of 30m. With the introduction of a transition constraint, handovers are smooth for transitions greater than 4 metres.

Across two intersections, new transition constraints used to create $FIFO\_carf$ were effective at preventing car-following collisions on the transition lane at lengths of 6 m and above. The extra constraints led to lower average speeds, but achieved higher intersection occupancies close to the theoretical maximum capacity. The transition lane constraints presented are expressed in the same fixed distance along path coordinates as the crossing constraints from previous chapters, allowing the problem to remain a linear program, suitable for interior point methods.

The findings in this section take the first step to proving the proposed division of control between intersection managers and individual vehicle computers is a valuable middle ground between per-vehicle decentralization and fully centralized control with roadmap reservations. The next step would be a test on the two intersection map with a bidirectional transition and traffic in both directions. Three or more intersections making up a small site would permit a meaningful comparison of the task completion rate with a centralized method based on conflict-free routing.

7.4 Discussion

Chapter 3 presented a method to address objective O1 based on fitting clothoid splines of minimum length and sharpness to polygon regions to create paths with the property of curvature continuity. Up to a maximum speed limit, such a path is dynamically feasible for an Ackermann steered vehicle with non-zero mass and inertia. Real vehicles will always be subject to disturbances and require closed loop path-tracking feedback control. Even so, tracking error can be reduced substantially if the target trajectory is dynamically feasible. Freedom to choose any longitudinal speed below this limit and maintain dynamic feasibility, presents an opportunity to share control authority. Low resolution data with a low update rate like the discrete waypoint path to be followed for the next few seconds can be broadcast over a wireless link to synchronise the motion of co-operating vehicles.

One approach to synchronisation is detailed in Chapter 4.1. This addresses Objective O2 with the introduction of an intersection manager agent, which exchanges messages with vehicles approaching an intersection. The low-resolution data describing the motion plan of each vehicle feeds into a centralized per-intersection speed optimization bounded by the speed limit. Solving for just two average speeds per vehicle (one to approach and one to cross) makes the problem tractable for larger numbers of vehicles. The complexity of the ordering problem as the number of vehicles increases can still be unacceptable for real-time execution. The presented method with a fixed ordering cut the execution time by an order of magnitude, but on a realistic intersection layout in Chapter 5 the average crossing speeds were much lower. The time wasted by waiting vehicles always exceeded the extra computation time. A promising direction for further work would be a study of ordering heuristics other than First-Come-First-Served. These chapters show that the optimal speed choice for any fixed ordering can be expressed as a linear program on multi-lane intersection with merged conflict zones. This type of problem can be solved with interior-point methods with bounded execution time, making them good candidates for the sole conflict resolution algorithm. On isolated intersections, this could lead to improved crossing times for automated traffic compared to the semaphore approach tested, roadmap reservation methods and potentially traffic light signals too. The next question for guaranteed collision avoidance on deployed systems is the tolerance to an isolated mechanical failure. All the crossing times reported in this thesis assumed perfect mechanical reliability. For real-world deployment, the spacing between vehicles in perpendicular directions should be sufficient to ensure that if a crossing vehicle fails in such a way as to block the intersection, there is time for those approaching to stop safely. This property might be called isolated-failure-safety. The provision of this safety buffer between each crossing pair may temper the throughput improvements offered by optimal management over isolated-failure-safe methods like the binary semaphore, but should not affect the reduction in energy usage. Further studies may be warranted focusing on the energy usage objective which was obtained in the present research as an unintended side effect of throughput optimization.

The natural extension of intersection management by assigning one manager per intersection zone to resolve all conflicts across an entire roadmap (Objective O3) has only been partially explored in Chapter 6. It was shown that even with a locally managed crossing order and approach speed, collisions can still occur on the transition lane between two closely spaced intersections. The proposed combination of departure constraints along with car following constraints was effective at preventing collisions if the transition lane was longer than 6 m on the particular numerical example tested. This had two elementary intersections with a transition lane in one direction only. The next step is to get a bidirectional transition between two intersections working. To help with this, it may be important to relax the requirement for AGVs to exactly match the arrival and departure times specified by the manager. Instead, the earliest possible safe arrival time (or equivalently the maximum average speed across the approach segment) could be specified. Then car-following behaviour could be implemented on the individual AGV controllers without invalidating the plans made at the intersection level. Individual vehicles could then ensure safety along the transition lane. This may also be beneficial for achieving isolated-failure-safety.

7.5 Future Work

The results presented here suggest further investigation into fleet control decomposition with locally optimal intersection speed management is worthwhile to attain the crossing time and energy usage benefits. To make a head-to-head comparison with other approaches to fleet control such as roadmap reservation the intersection management algorithm presented in Chapter 6 would need to be combined with a routing algorithm incorporating an approximate model of intersection zone capacity.

Future research is needed to characterize the speed-density curve resulting from optimal intersection management. Chapter 6 suggested linear fits to results for a small example. An accurate capacity model for an intersection communication zone would make it possible to augment the method for conflict resolution with a routing algorithm which directed vehicles away from zones close to capacity.

The assumption of fixed direction lanes does limit the applicability of the presented constraints in Chapter 4.1 slightly. Many logistics applications involve low speed vehicles which move backward and forward on the same lanes to save floor space. The presented method applied to such a site would need to define every link as a conflict zone to ensure only one vehicle at a time entered it. In these situations it may be more appropriate to use a roadmap reservation method, as the benefits of platooning are less significant.

All the methods tested in this work used a merged conflict zone, which may reduce capacity if there are multiple lanes like in Chapter 5. Further work could examine breaking the merged conflict zone down into sub-conflicts, each involving a subset of lanes. An intersection manager which solved this problem would be capable of managing every elementary intersection across the site. Based on the review in Chapter 2, such an approach to centralised fleet management would be quite novel. The more pressing question is can the presented site control decomposition reduce energy consumption and reduce the makespan for a given set of tasks compared to roadmap reservations across a whole site?

Appendix A

Constructing Topology Graph From Convex Region List

Algorithm 2 Algorithm for constructing an extremal point graph from a collection of possibly overlapping convex regions.

```
1: $graph \leftarrow CheckStartAndGoalInFreeSpaceandSetGroups(polygon\_region\_list, start, goal)$
2: $graph \leftarrow AddNodesAndSetGroups(polygon\_region\_list, start, goal, start\_groups, goal\_groups)$
3: $graph \leftarrow ConnectUpTheGroups(graph)$
4: $shortest\_path \leftarrow Dijkstra(graph)$
5: $H = \leftarrow BuildHMatrix(shortest\_path)$
```

Algorithm 3 List the group ID of empty regions occupied by the start position and the empty regions containing the goal position.

```
1: function $CheckStartAndGoalInFreeSpaceandSetGroups(polygon\_region\_list, start, goal)$
2:     $graph \leftarrow DirectedGraph()$
3:     $start\_groups \leftarrow [];$
4:     $goal\_groups \leftarrow [];$
5:     for all $k \in \{1 : length(polygon\_region\_list)\}$ do
6:         if $polygon\_region\_list(k).GetOnlyInteriorPoints(start) \neq \emptyset$ then
7:             $start\_groups \leftarrow [start\_groups, k];$
8:         end if
9:         if $polygon\_region\_list(k).GetOnlyInteriorPoints(goal) \neq \emptyset$ then
10:            $goal\_groups \leftarrow [goal\_groups, k];$
11:        end if
12:    end for
13:    return $start\_groups, goal\_groups$
14: end function
```

Algorithm 4 Construct a graph with a node for every point on the boundary of each empty region and every intersection between edges of two empty regions. Each node is labelled with one or more group IDs indicating the empty regions which it occupies.

```
1: function $AddNodesAndSetGroups$ ( $polygon\_region\_list, start, goal, start_groups, goal_groups$
   )
2:     $graph \leftarrow graph.AddNode(start, start\_groups)$;
3:     $n\_regions \leftarrow size(obj.polygon\_region\_list, 2)$;
4:     for all $i \in \{1 : n\_regions\}$ do
5:         for all $j \in \{1 : n\_regions\}$ do
6:             if $i \neq j$ then
7:                 $vertex\_list\_i \leftarrow polygon\_region\_list(i).vertex\_list$;
8:                 $vertex\_list\_within\_j \leftarrow polygon\_region\_list(j).GetOnlyInteriorPoints(vertex\_list$
9:                 $n\_within\_j \leftarrow size(vertex\_list\_within\_j, 1)$;
10:                for all $m \in 1 : n\_within\_j$ do
11:                    $graph \leftarrow graph.AddNode(vertex\_list\_within\_j(m, :), [i, j])$;
12:                end for
13:                if $j > i$ then
14:                    $intercept\_points \leftarrow GetInterceptPoints(polygon\_region\_list(i), polygon\_region$
15:                    $n\_intercept\_points = size(intercept\_points, 1)$;
16:                    for all $p \in \{1 : n\_intercept\_points\}$ do
17:                        $graph \leftarrow graph.AddNode(intercept\_points(p, :), [i, j])$;
18:                    end for
19:                end if
20:            end if
21:        end for
22:    end for
23:    $graph = graph.AddNode(goal, goal_groups)$;
24:    return $graph$
25: end function
```

Algorithm 5 Add an undirected link between each pair of nodes labelled with same group ID.

```
1: function $ConnectUpTheGroups$ ( $polygon\_region\_list, graph$ )
2:     for all $k \in \{1 : graph.node\_map.Count\}$ do
3:         $from\_node \leftarrow graph.node\_map(k)$
4:         $groups = from\_node.groups$
5:         $group\_count = length(groups)$
6:         for $q = 1 : group\_count$ do
7:             $group\_id \leftarrow groups(q)$
8:             for all $r \in \{1 : graph.node\_map.Count\}$ do
9:                 if $r \neq k \wedge graph.node\_map(r).IsMember(group\_id)$ then $graph \leftarrow graph.AddLink(from\_node.id, graph.node\_map(r).id)$
10:                end if
11:            end for
12:        end for
13:    end for
14:    return $graph$
15: end function
```

Appendix B

Analytical Gradients of the Single Shooting Clothoid Formulation Within One Region

Stacking the parameters for the optimization described in Section 3.4 into a single vector $\boldsymbol{p}^{1\times 6} = [\alpha_1, \alpha_2, L_1, L_2, s_0, s_F]^T$ allows the Jacobian of the objective for one region to be expressed as

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{p}} = 2 \cdot [b\alpha_1, b\alpha_2, L_1, L_2, s_0, s_F]^T \tag{B.1}$$

. The pose at the end of the path piece in the region of interest is $[X_F, Y_F, \Psi_F]$. For the derivatives we only consider one path piece, and drop the i subscript used in Equation 3.8. The equality constraint vector with goal pose $[\hat{X}, \hat{Y}, \hat{\Psi}]$ is given by

$$\boldsymbol{c_{eq}} = \begin{bmatrix} X_F - \hat{X} \\ Y_F - \hat{Y} \\ \Psi_F - \hat{\Psi} \\ \kappa_F - 0 \end{bmatrix} = \boldsymbol{0} \tag{B.2}$$

and the inequality constraints to remain within a square region bounded by $[X_{max}, X_{min}, Y_{max}, Y_{min}]$ is given by

$$\boldsymbol{c_{ineq}} = \begin{bmatrix} X_F - X_{max} \\ -X_F + X_{min} \\ Y_F - Y_{max} \\ -Y_F + Y_{min} \end{bmatrix} \leq \boldsymbol{0} \tag{B.3}$$

. The matrix derivatives $\frac{\partial \boldsymbol{c}_{eq}}{\partial \boldsymbol{p}}^{(6\times 4)}$ and $\frac{\partial \boldsymbol{c}_{ineq}}{\partial \boldsymbol{p}}^{(6\times 4)}$ were constructed from scalar partial derivatives listed below. Each element is defined in terms of the parameters $\boldsymbol{p}$ in the following sections. The symbols are introduced in Section 3.3.1, the starting pose is taken to be the origin, so the end pose in global coordinates is given by Equation 3.8. Functions $C(\alpha, s, \kappa)$ and $S(\alpha, s, \kappa)$ are defined by Equation 3.3 and Equation 3.4 respectively. The peak curvature where the two clothoids meet is given the symbol $\kappa_m = \alpha_1 L_1$ and the final heading at the end of the second clothoid is given by $\Psi_F = \frac{\alpha_1 L_1^2}{2} + \frac{\alpha_2 L_2^2}{2}$.

B.1 Components with respect to α_1

$$\begin{aligned}
\frac{\partial X_F}{\partial \alpha_1} = \int_0^{L_1} -\frac{u^2}{2}\sin\left(\frac{\alpha_1 u^2}{2}\right) du \\
+ \cos(\delta_1)\int_0^{L_2} (-L_1 \cdot u)\sin\left(\alpha_1 L_1 \cdot u + \frac{\alpha_2 u^2}{2}\right) du \\
+ (-\sin(\delta_1)L_1^2/2)\int_0^{L_2} \cos(\alpha_1 L_1 + \alpha_2 u) du \\
- \sin(\delta_1)\int_0^{L_2} (L_1 \cdot u)\cos\left(\alpha_1 L_1 \cdot u + \frac{\alpha_2 u^2}{2}\right) du \\
- \left(\cos(\delta_1)L_1^2/2\right)\int_0^{L_2} \sin(\alpha_1 L_1 + \alpha_2 u) du \\
+ s_F(L_1 L_2 + L_1^2/2)(-\sin\Psi_F)
\end{aligned} \tag{B.4}$$

$$\begin{aligned}
\frac{\partial Y_F}{\partial \alpha_1} = \int_0^{L_1} \frac{u^2}{2}\cos\left(\frac{\alpha_1 u^2}{2}\right) du \\
+ \sin(\delta_1)\int_0^{L_2} (-L_1 \cdot u)\sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\
+ (\cos(\delta_1)\frac{L_1^2}{2})\int_0^{L_2} \cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\
+ \cos(\delta_1)\int_0^{L_2} (L_1 \cdot u)\cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\
+ (-\sin(\delta_1)\frac{L_1^2}{2})\int_0^{L_2} \sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\
+ s_F(L_1 L_2 + L_1^2/2)\cos(\Psi_F)
\end{aligned} \tag{B.5}$$

$$\partial\Psi_F/\partial\alpha_1 = \frac{1}{2}L_1^2 + L_1 L_2 \tag{B.6}$$

$$\partial\kappa_F/\partial\alpha_1 = L_1 \tag{B.7}$$

B.2 Components with respect to α_2

$$\begin{aligned}\frac{\partial X_F}{\partial \alpha_2} = \cos(\delta_1) \int_0^{L_2} \left(\frac{-u^2}{2}\right) \sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ - \sin(\delta_1) \int_0^{L_2} \left(\frac{u^2}{2}\right) \cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ + s_F(-\sin\Psi_F)\left(\frac{L_2^2}{2}\right)\end{aligned} \quad \text{(B.8)}$$

$$\begin{aligned}\frac{\partial Y_F}{\partial \alpha_2} = \\ sin(\delta_1) \int_0^{L_2} -\frac{u^2}{2} \sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ + \cos(\delta_1) \int_0^{L_2} \frac{u^2}{2} \cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ + s_F(\cos\Psi_F) \cdot \left(\frac{1}{2} L_2^2\right)\end{aligned} \quad \text{(B.9)}$$

$$\frac{\partial \Psi_F}{\partial \alpha_2} = \frac{1}{2} L_2^2 \quad \text{(B.10)}$$

$$\frac{\partial \kappa_F}{\partial \alpha_2} = L_2 \quad \text{(B.11)}$$

B.3 Components with respect to L_1

$$\begin{aligned}\frac{\partial X_F}{\partial L_1} = \cos(\alpha_1 L_1^2/2) \\ + \cos(\delta_1) \int_0^{L_2} (-\alpha_1 \cdot u) \sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ + (-\sin(\delta_1)\alpha_1 L_1) \int_0^{L_2} \cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ - \sin(\delta_1) \int_0^{L_2} (\alpha_1 \cdot u) \cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ - (\cos(\delta_1)\alpha_1 L_1) \int_0^{L_2} \sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right) du \\ - s_F(\alpha_1 L_1 + \alpha_1 L_2)\sin(\Psi_F)\end{aligned} \quad \text{(B.12)}$$

$$\begin{aligned}\frac{\partial Y_F}{\partial L_1} = \sin(\alpha_1 L_1^2/2)\\ + \sin(\delta_1)\int_0^{L_2}(-\alpha_1 \cdot u)\sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right)du\\ + (\cos(\delta_1)\alpha_1 L_1)\int_0^{L_2}\cos\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right)du\\ + \cos(\delta_1)\int_0^{L_2}(\alpha_1 \cdot u)\cos\left(\alpha_1 L_1 u + \frac{\alpha u^2}{2}\right)du\\ + (\sin(\delta_1)\alpha_1 L_1)\int_0^{L_2}\sin\left(\alpha_1 L_1 u + \frac{\alpha_2 u^2}{2}\right)du\\ + s_F(\alpha_1 L_1 + \alpha_1 L_2)\cos(\Psi_F)\end{aligned} \tag{B.13}$$

$$\frac{\partial \Psi_F}{\partial L_1} = \alpha_1 L_1 + \alpha_1 L_2 \tag{B.14}$$

$$\frac{\kappa_F}{\partial L_1} = \alpha_1 \tag{B.15}$$

B.4 Components with respect to L_2

$$\begin{aligned}\frac{\partial X_F}{\partial L_2} = \cos\left(\alpha_1 L_1 L_2 + \frac{\alpha_2 L_2^2}{2}\right)\cos(\delta_1)\\ - \sin\left(\alpha_1 L_1 L_2 + \frac{\alpha_2 L_2^2}{2}\right)\sin(\delta_1)\\ + s_F \cdot (-\sin\Psi_F)(\alpha_1 L_1 + \alpha_2 L_2)\end{aligned} \tag{B.16}$$

$$\begin{aligned}\frac{\partial Y_F}{\partial L_2} = \cos\left(\alpha_1 L_1 L_2 + \frac{\alpha_2 L_2^2}{2}\right)\sin(\delta_1)\\ + \sin\left(\alpha_1 L_1 L_2 + \frac{\alpha_2 L_2^2}{2}\right)\cos(\delta_1)\\ + s_F \cdot (\cos\Psi_F)(\alpha_1 L_1 + \alpha_2 L_2)\end{aligned} \tag{B.17}$$

$$\frac{\partial \Psi_F}{\partial L_2} = \alpha_1 L_1 + \alpha_2 L_2 \tag{B.18}$$

$$\frac{\kappa_F}{\partial L_2} = \alpha_2 \tag{B.19}$$

B.5 Components with respect to s_0

$$\partial X_F/\partial s_0 = 1 \tag{B.20}$$

$$\partial Y_F/\partial s_0 = 0 \tag{B.21}$$

$$\partial \Psi_F/\partial s_0 = 0 \tag{B.22}$$

$$\partial \kappa_F/\partial s_0 = 0 \tag{B.23}$$

B.6 Components with respect to s_F

$$\partial X_F/\partial s_F = \cos(\Psi_F) \tag{B.24}$$

$$\partial Y_F/\partial s_F = \sin(\Psi_F) \tag{B.25}$$

$$\partial \Psi_F/\partial s_F = 0 \tag{B.26}$$

$$\partial \kappa_F/\partial s_F = 0 \tag{B.27}$$

Appendix C

Pose Operators

These operators are defined to simplify working with rigid bodies in two dimensions. A rigid body in a plane has $2+1$ parameters, two for translation in the plane and one for heading angle. The position in metres and the heading in radians can be assembled into a three vector, called the pose.

$$\boldsymbol{p_1} = [x_1, y_1, \psi_1]^T \tag{C.1}$$

$$\boldsymbol{p_2} = [x_2, y_2, \psi_2]^T \tag{C.2}$$

Working with higher dimensional spaces it is common to use homogeneous coordinates to simplify operations on rigid bodies. The two operators defined here are convenient for working in 2D without the complexity of homogeneous notation. They are inverse operators in the sense that $\boldsymbol{p_3} = \boldsymbol{p_1} \oplus \boldsymbol{p_2} \rightarrow \boldsymbol{p_3} \ominus \boldsymbol{p_2} = \boldsymbol{p_1}$. The decompose operator $\ominus$ is order dependent like ordinary subtraction. They are defined below using the algebra of homogeneous coordinates given in [161].

C.1 Compose $\oplus$

This operator takes the second pose and rotates it into the frame of the first before vector addition of all three components.

$$\boldsymbol{p_3} = \boldsymbol{p_1} \oplus \boldsymbol{p_2} = \begin{bmatrix} x_1 + x_2\cos(\psi_2) - y_2\sin(\psi_2) \\ y_1 + x_2\sin(\psi_2) + y_2\cos(\psi_2) \\ \psi_1 + \psi_2 \end{bmatrix} \tag{C.3}$$

. If each pose is expressed in homogeneous 3x3 form,

$$\boldsymbol{T_1} = \begin{bmatrix} \cos(\psi_1) & -\sin(\psi_1) & x_1 \\ \sin(\psi_1) & \cos(\psi_1) & y_1 \\ 0 & 0 & 1 \end{bmatrix} \tag{C.4}$$

the same operation is a matrix multiplication

$$\boldsymbol{T_3} = \boldsymbol{T_1}\boldsymbol{T_2} \tag{C.5}$$

C.2 Decompose $\ominus$

This operator finds the pose of $\boldsymbol{p_3}$ expressed relative to pose $\boldsymbol{p_1}$

$$\boldsymbol{p_1} = \boldsymbol{p_3} \ominus \boldsymbol{p_2} \tag{C.6}$$

If each pose is expressed in homogeneous 3x3 form, the same operation is a matrix multiplication by the the inverse

$$\boldsymbol{T_1} = \boldsymbol{T_3}\boldsymbol{T_2}^{-1} \tag{C.7}$$

Appendix D

Vehicle Dynamics to Path Constraints

The motion of a vehicle with Differential Drive configuration can be represented as a rigid body with forward velocity v, rotational velocity ω and zero lateral motion centred at the control point shown in Figure D.1. An exact form incorporating the turning radius (which has a singularity for travelling in a straight line) is given in textbooks [162].

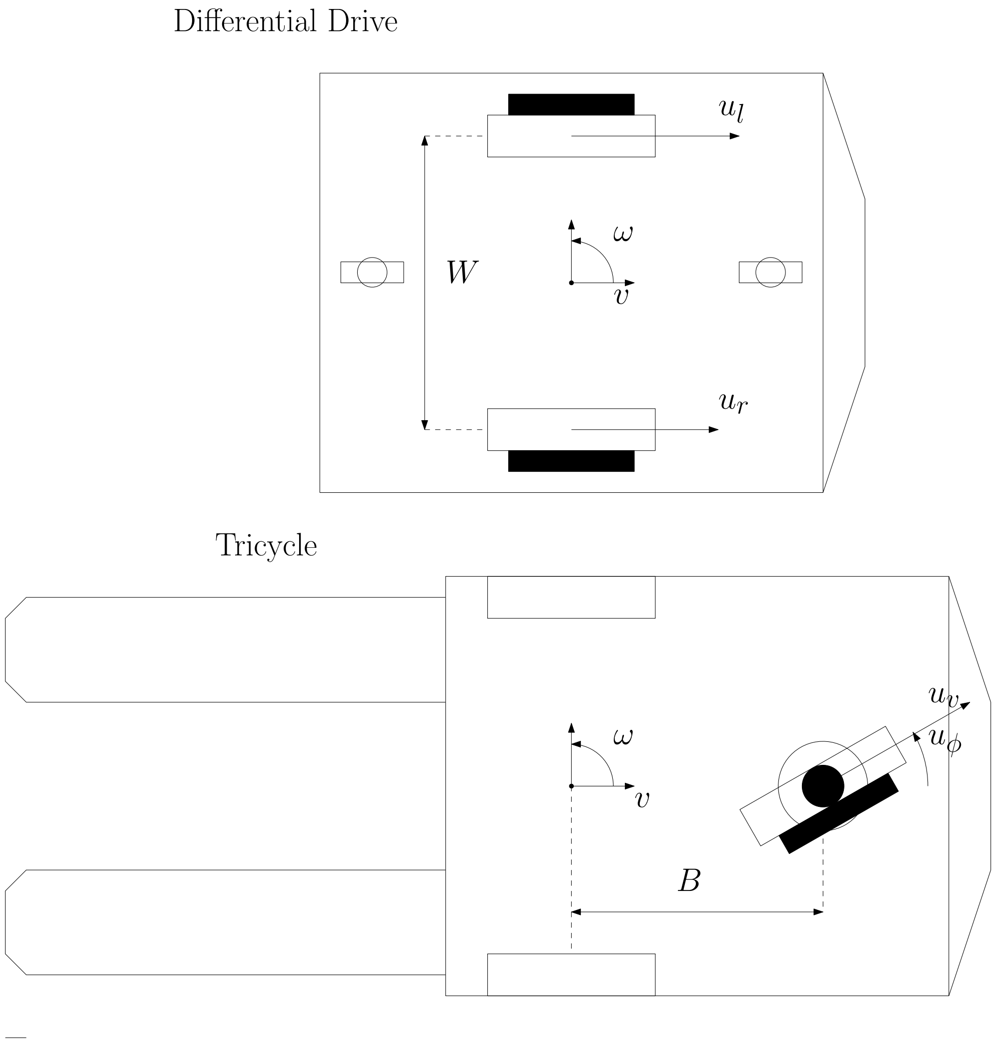


Figure D.1: Plan view of Differential Drive and Tricycle vehicle configurations.

The forward velocity is given by Equation D.1 and the angular velocity by Equation D.2. The control vector contains $[u_l, u_r]$ the speed command for the left and right wheels.

$$v = \frac{u_l + u_r}{2} \tag{D.1}$$

$$\omega = \frac{u_r - u_l}{W} \tag{D.2}$$

This model implies the Differential Drive configuration is capable of following any continuous path in R_2 [162], as long as the velocity commands can be followed exactly. It is unrealistic to expect low level controllers to follow their set point exactly due to inertial effects so a more realistic control vector would be the acceleration on each wheel.

A vehicle with a Tricycle configuration can be represented as a rigid body with forward velocity v, rotational velocity ω and zero lateral motion centred at the control point shown in Figure D.1.

$$v = u_v \cos u_\phi \tag{D.3}$$

$$\omega = \frac{u_v \tan u_\phi}{B} \tag{D.4}$$

The forward velocity is approximated by Equation D.3 and the angular velocity by Equation D.4. The control vector contains $[u_v, u_\phi]$ the speed command for single drive wheel and the angle command for the rotating steering column.

Both configurations have two actuated degrees of freedom and can move the control point to any point $[x, y, \psi]$ in the workspace. They can also be inverted to derive the impact of limited actuator response and motor power on achievable path shapes. Some actuator limits are purely kinematic: arising from the geometry of wheel placement without regard to the forces that cause motion [163]. Examples include the maximum turning radius and its relation to the steer angle limit. Others are clearly dynamic, like the maximum linear acceleration depending on mass of the vehicle and the peak torque provided by the motor. Both kinematic and dynamic actuator constraints will change the trajectory of the physical vehicle relative to an idealized one.

Equations D.1 to D.4 can be used to transform actuator limits in the control space to find their effect on the path. When the trajectory is decomposed into a path and a speed profile, the kinematic constraints mainly affect the path while the dynamic constraints mainly affect the speed profile. However, dynamic constraints on the actuators can also lead to geometric constraints on the path. Particularly on the spatial derivatives, curvature κ and sharpness $\dot{\kappa}$. These constraints are still coupled through the resulting trajectory. It can be useful to think of the limits in terms of trajectory properties like centripetal acceleration a_l. If this is too high it can lead to a skid where the platform moves laterally, or a reduction in down force on the inside wheels and eventually tipping. The curvature $\kappa = 1/r$ is a geometric property of the path, but the constraint $a_l < \bar{a}_l$ can always be satisfied by choosing a sufficiently low longitudinal speed v.

Starting with the expression for centrifugal force needed for curved motion in term of instantaneous curvature at a constant radial speed in Equation D.5.

$$a_l = M\kappa v^2 \tag{D.5}$$

$$\bar{v} = \sqrt{\frac{\bar{a}_l}{M\kappa}} \tag{D.6}$$

The relation in D.6 holds for segments over which the longitudinal speed v is constant. It can be identified for each segment by calculating $\bar{v}$ at a series of sample locations and setting the segment maximum to be the lowest limit across the samples. Optimal profiles which do not neglect the impact of longitudinal acceleration have also been developed [164], including for more exotic vehicle configurations like towing trailers [15].

Bibliography

[1] ED Lambert, R Romano, and D Watling. Optimal Smooth Paths based on Clothoids for Car-like Vehicles in the Presence of Obstacles. *International Journal of Control, Automation and Systems*, 2020. ISSN 1598-6446.

[2] Valerio Digani, M. Ani Hsieh, Lorenzo Sabattini, and Cristian Secchi. Coordination of multiple AGVs: a quadratic optimization method. *Autonomous Robots*, 43(3):539–555, 2019. ISSN 15737527. doi: 10.1007/s10514-018-9730-9. URL https://doi.org/10.1007/s10514-018-9730-9.

[3] J A Reeds and L A Shepp. Optimal paths for a car that goes forwards and backwards. 145(2):367–393, 1990.

[4] Hyster. E30-40HSD Technical guide. www.hyster.com, August 2020.

[5] Hyster. J30-40XNT/XN Series Technical Guide. http://www.hyster.com/north-america/en-us/products/3-wheel-electric-trucks/E30-40HSD3/, August 2020.

[6] Szymon Racewicz, Paweł Kazimierczuk, Bronisław Kolator, and Andrzej Olszewski. Use of 3 kW BLDC motor for light two-wheeled electric vehicle construction. *IOP Conference Series: Materials Science and Engineering*, 421(4), 2018. ISSN 1757899X. doi: 10.1088/1757-899X/421/4/042067.

[7] Kaveh Azadeh, René De Koster, and Debjit Roy. Robotized and automated warehouse systems: Review and recent developments. *Transportation Science*, 53 (4):917–945, 2019. ISSN 15265447. doi: 10.1287/trsc.2018.0873.

[8] Russel Meller, Dima Nazzal, and Lisa M. Thomas. Collaborative Bots in Distribution Centers. *Savannah, Imhrc Proceedings*, 2018.

[9] Daniel Sykes and Gavin Keighren. Industrial-scale environments with bounded uncertainty. *2018 IEEE/ACM 1st International Workshop on Robotics Software Engineering (RoSE)*, pages 29–32, 2018. doi: 10.1145/3196558.3196563.

[10] Robert Bogue. Growth in e-commerce boosts innovation in the warehouse robot market. *Industrial Robot*, 43(6):583–587, 2016. ISSN 0143991X. doi: 10.1108/IR-07-2016-0194.

[11] Peter R Wurman, Raffaello D'Andrea, and Mick Mountz. Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses. *AI Magazine*, 29(1):9–20, 2008. ISSN 07384602. doi: 10.1609/aimag.v29i1.2082. URL http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/2082.

[12] AMRCore, 2022. URL https://www.guidanceautomation.com/wp-content/uploads/2021/10/94-5080-Rev-2-AMRCore-Datasheet-1.pdf.

[13] Martin Raymond Davies. Deskilling Robots in Logistics Environments. *KI - Kunstliche Intelligenz*, 33(4):407–410, 2019. ISSN 16101987. doi: 10.1007/s13218-019-00624-y. URL https://doi.org/10.1007/s13218-019-00624-y.

[14] Renardo Baird. An autonomous forklift research platform for warehouse operations. Master's thesis, Massachusetts Institute of Technology, 2018. URL https://dspace.mit.edu/handle/1721.1/121621.

[15] Mirko Kokot, Damjan Miklic, and Tamara Petrovic. Path continuity for multi-wheeled AGVs. *IEEE Robotics and Automation Letters*, 3766(c):1–1, 2021. doi: 10.1109/lra.2021.3099086.

[16] M. De Ryck, M. Versteyhe, and F. Debrouwere. Automated guided vehicle systems, state-of-the-art control algorithms and techniques. *Journal of Manufacturing Systems*, 54:152–173, 2020. ISSN 02786125. doi: 10.1016/j.jmsy.2019.12.002.

[17] T. Lamballais, D. Roy, and M. B.M. De Koster. Estimating performance in a Robotic Mobile Fulfillment System. *European Journal of Operational Research*, 256(3):976–990, 2017. ISSN 03772217. doi: 10.1016/j.ejor.2016.06.063. URL http://dx.doi.org/10.1016/j.ejor.2016.06.063.

[18] Olli Bräysy and Michel Gendreau. Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation Science*, 39(1):104–118, 2005. ISSN 00411655. doi: 10.1287/trsc.1030.0056.

[19] Iris F.A. Vis. Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research*, 170(3):677–709, 2006. ISSN 03772217. doi: 10.1016/j.ejor.2004.09.020.

[20] Giuseppe Fragapane, René de Koster, Fabio Sgarbossa, and Jan Ola Strandhagen. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. *European Journal of Operational Research*, 2021. ISSN 03772217. doi: 10.1016/j.ejor.2021.01.019. URL https://doi.org/10.1016/j.ejor.2021.01.019.

[21] Dimitris Bertsimas, Patrick Jaillet, and Sébastien Martin. Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research*, 67(1): 143–162, 2019. ISSN 15265463. doi: 10.1287/opre.2018.1763.

[22] Liang Yang, Juntong Qi, Jizhong Xiao, and Xia Yong. A literature review of UAV 3D path planning. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, 2015-March(March):2376–2381, 2015. doi: 10.1109/WCICA.2014.7053093.

[23] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and Decision-Making for Autonomous Vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):annurev—-control—-060117—-105157, 2018. ISSN 2573-5144. doi: 10.1146/annurev-control-060117-105157. URL http://www.annualreviews.org/doi/10.1146/annurev-control-060117-105157.

[24] J Andrew (Drew) Bagnell. An Invitation to Imitation. Technical Report CMU-RI-TR-15-08, Carnegie Mellon University, Pittsburgh, PA, mar 2015. URL https://www.ri.cmu.edu/publications/an-invitation-to-imitation/.

[25] Lu Chi and Yadong Mu. Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues. *CoRR*, abs/1708.0, 2017. URL http://arxiv.org/abs/1708.03798.

[26] Changxi You, Jianbo Lu, Dimitar Filev, and Panagiotis Tsiotras. Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning. *Robotics and Autonomous Systems*, 114:1–18, 2019. ISSN 09218890. doi: 10.1016/j.robot.2019.01.003. URL https://doi.org/10.1016/j.robot.2019.01.003.

[27] Matteo Leonetti, Luca Iocchi, and Peter Stone. A synthesis of automated planning and reinforcement learning for efficient, robust decision-making. *Artificial Intelligence*, 241:103–130, 2016. ISSN 00043702. doi: 10.1016/j.artint.2016.07.004. URL http://dx.doi.org/10.1016/j.artint.2016.07.004.

[28] Thomas Levermore and Alan Peters. Test framework and key challenges for virtual verification of automated vehicles : the VeriCAV project. In *39th International Conference on Computer Safety, reliability and Security (SAFECOMP)*, Lisbon, Portugal, 2020. HAL. URL https://hal.archives-ouvertes.fr/hal-02931723.

[29] Yunlong Zhao, Xiaoping Liu, Shaobo Wu, and Gang Wang. Spare zone based hierarchical motion coordination for multi-AGV systems. *Simulation Modelling Practice and Theory*, 109(December 2020):102294, 2021. ISSN 1569190X. doi: 10.1016/j.simpat.2021.102294. URL https://doi.org/10.1016/j.simpat.2021.102294.

[30] R Deits and R Tedrake. Efficient mixed-integer planning for UAVs in cluttered environments. *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 42–49, 2015. ISSN 10504729. doi: 10.1109/ICRA.2015.7138978.

[31] Brian Paden, Michal Cap, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles. *arXiv preprint arXiv:*, pages 1–27, 2016. ISSN 2379-8904. doi: 10.1109/TIV.2016.2578706. URL http://arxiv.org/abs/1604.07446.

[32] Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer Handbooks. Springer, Berlin, 2 edition, 2016. ISBN 978-3-319-32550-7. doi: 10.1007/978-3-540-30301-5.

[33] Johann Borenstein and Yoram Koren. The vector field histogram-fast obstacle

avoidance for mobile robots - Robotics and Automation, IEEE Transactions on. *IEEE Transactions on Robotics*, 7(3):278–288, 1991.

[34] Christoph Rosmann, Frank Hoffmann, and Torsten Bertram. Kinodynamic trajectory optimization and control for car-like robots. *IEEE International Conference on Intelligent Robots and Systems*, 2017-Septe:5681–5686, 2017. ISSN 21530866. doi: 10.1109/IROS.2017.8206458.

[35] Julien Moreau, Pierre Melchior, Stéphane Victor, Louis Cassany, Mathieu Moze, François Aioun, and Franck Guillemard. Reactive path planning in intersection for autonomous vehicle. *IFAC-PapersOnLine*, 52(5):109–114, 2019. ISSN 24058963. doi: 10.1016/j.ifacol.2019.09.018.

[36] Yongsoon Yoon, Jongho Shin, H. Jin Kim, Yongwoon Park, and Shankar Sastry. Model-predictive active steering and obstacle avoidance for autonomous ground vehicles. *Control Engineering Practice*, 17(7):741–750, 2009. ISSN 09670661. doi: 10.1016/j.conengprac.2008.12.001.

[37] Tianyu Gu. *Improved Trajectory Planning for On-Road Self-Driving Vehicles Via Combined Graph Search, Optimization & Topology Analysis.* Doctor of philiosophy, Carnegie Mellon University, 2017.

[38] Martin Keller, Frank Hoffmann, Torsten Bertram, Carsten Hass, and Alois Seewald. *Planning of optimal collision avoidance trajectories with timed elastic bands*, volume 19. IFAC, 2014. ISBN 9783902823625. doi: 10.3182/20140824-6-ZA-1003.01143. URL http://dx.doi.org/10.3182/20140824-6-ZA-1003.01143.

[39] Ji Chang Kim, Dong Sung Pae, and Myo Taeg Lim. Obstacle Avoidance Path Planning based on Output Constrained Model Predictive Control. *International Journal of Control, Automation and Systems*, 17(11):2850–2861, 2019. ISSN 20054092. doi: 10.1007/s12555-019-9091-y.

[40] Pedro F. Lima, Mattias Nilsson, Marco Trincavelli, Jonas Martensson, and Bo Wahlberg. Spatial Model Predictive Control for Smooth and Accurate Steering of an Autonomous Truck. *IEEE Transactions on Intelligent Vehicles*, 2(4): 238–250, 2017. ISSN 2379-8858. doi: 10.1109/tiv.2017.2767279.

[41] Laurene Claussmann, Marc Revilloud, Dominique Gruyer, and Sebastien Glaser. A Review of Motion Planning for Highway Autonomous Driving. *IEEE Transactions on Intelligent Transportation Systems*, (June):1–23, 2019. ISSN 1524-9050. doi: 10.1109/tits.2019.2913998.

[42] Steven M. LaValle and Daniel Leidner. Chapter 6: Combinatorial Motion Planning. In *Planning Algorithms*, chapter 6, pages 249–310. Cambridge University Press, 2006. ISBN 9780511546877. doi: 10.1017/CBO9780511546877. URL http://ebooks.cambridge.org/ref/id/CBO9780511546877.

[43] Marco L. Della Vedova, Matteo Rubagotti, Tullio Facchinetti, and Antonella Ferrara. Platooning control of autonomous nonholonomic mobile robots in a human-robot coexisting environment. *Proceedings of the American Control Conference*, 1(1):6569–6574, 2012. ISSN 07431619. doi: 10.1109/acc.2012.6314938.

[44] Lydia E. Kavraki and Steven M. LaValle. Handbook of Robotics Chapter 7 : Motion planning. In Bruno Siciliano and O Khatib, editors, *Springer Handbook of Robotics*, chapter 7, pages 139–161. Springer, 2016. ISBN 9783319325521. doi: 10.1007/978-3-319-32552-1_7.

[45] Ross A. Knepper and Alonzo Kelly. High performance state lattice planning using heuristic look-up tables. *IEEE International Conference on Intelligent Robots and Systems*, page 3375, 2006. doi: 10.1109/IROS.2006.282515.

[46] Mikhail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3): 308–333, 2009.

[47] Steven M. LaValle and James J. Kuffner. Rapidly-exploring random trees: Progress and prospects. *4th Workshop on Algorithmic and Computational Robotics: New Directions*, pages 293–308, 2000. ISSN 16130073. doi: 10.1017/CBO9781107415324.004. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.38.1387.

[48] Sangyol Yoon, Dasol Lee, Jiwon Jung, and David Hyunchul Shim. Spline-based RRT* Using Piecewise Continuous Collision-checking Algorithm for Car-like Vehicles. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 90(3-4):537–549, 2018. ISSN 15730409. doi: 10.1007/s10846-017-0693-4.

[49] Marios Xanthidis, Joel M. Esposito, Ioannis Rekleitis, and Jason M. O'Kane. Motion Planning by Sampling in Subspaces of Progressively Increasing Dimension. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 100 (3-4):777–789, 2020. ISSN 15730409. doi: 10.1007/s10846-020-01217-w.

[50] Katrina Samperi and Nick Hawes. Improving the Generation of Rapidly Exploring Randomised Trees (RRTs) in Large Scale Virtual Environments Using Trails. In Michael Mistry, Aleš Leonardis, Mark Witkowski, and Chris Melhuish, editors, *Advances in Autonomous Robotics Systems*, pages 231–242, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10401-0.

[51] Isira Naotunna and Theeraphong Wongratanaphisan. Comparison of ROS Local Planners with Differential Drive Heavy Robotic System. *International Conference on Advanced Mechatronic Systems, ICAMechS*, 2020-December:1–6, 2020. ISSN 23250690. doi: 10.1109/ICAMechS49982.2020.9310123.

[52] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamc window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, pages 23–33, 1997. doi: 1070-9932/97.

[53] D. Coombs, K. Murphy, A. Lacaze, and S. Legowik. Driving autonomously offroad up to 35 km/h. *IEEE Intelligent Vehicles Symposium, Proceedings*, (Mi):186–191, 2000. doi: 10.1109/ivs.2000.898339.

[54] Javier Minguez, Florent Lamiraux, and Jean-paul Laumond. Motion Planning and Obstacle Avoidance. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, number 2, chapter 47, pages 1177–1202. Springer, 2 edition, 2016. ISBN 9783540303015. doi: 10.1007/978-3-540-30301-5.

[55] Yoshiaki Kuwata, Justin Teo, Gaston Fiore, Sertac Karaman, Emilio Frazzoli, and Jonathan P How. Real-Time Motion Planning With Applications to Autonomous Urban Driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, 2009. ISSN 1063-6536, 1558-0865. doi: 10.1109/TCST.2008.2012116.

[56] Ming Yue, Xiangmin Wu, Lie Guo, and Junjie Gao. Quintic Polynomial-based Obstacle Avoidance Trajectory Planning and Tracking Control Framework for Tractor-trailer System. *International Journal of Control, Automation and Systems*, 17(10):2634–2646, 2019. ISSN 20054092. doi: 10.1007/s12555-018-0889-9.

[57] Valerio Digani, Fabrizio Caramaschi, Lorenzo Sabattini, Cristian Secchi, and Cesare Fantuzzi. Obstacle avoidance for industrial AGVs. In *Proceedings - 2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing, ICCP 2014*, pages 227–232. IEEE, 2014. ISBN 9781479965687. doi: 10.1109/ICCP.2014.6937001.

[58] Raph Levien. The Elastica: A Mathematical History. 2008.

[59] Suhyeon Gim, Lounis Adouane, Sukhan Lee, and Jean Pierre Dérutin. Clothoids Composition Method for Smooth Path Generation of Car-Like Vehicle Navigation. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 88 (1):129–146, 2017. ISSN 15730409. doi: 10.1007/s10846-017-0531-8.

[60] Joshua Henrie and Doran Wilde. Planning Continuous Curvature Paths Using Constructive Polylines. *Journal of Aerospace Computing, Information, and Communication*, 4(12):1143–1157, 2007. doi: 10.2514/1.32776.

[61] Marcus Lundberg. Path planning for autonomous vehicles using clothoid based smoothing of A * generated paths and optimal control. 2017.

[62] Thierry Fraichard and Alexis Scheuer. From Reeds and Shepp's to Continuous-Curvature Paths. *IEEE Transactions on Robotics*, 20(6):1025–1035, 2004.

[63] F. Lamiraux and J. P. Laumond. Smooth motion planning for car-like vehicles. *IEEE Transactions on Robotics and Automation*, 17(4):498–502, 2001. ISSN 1042296X. doi: 10.1109/70.954762.

[64] Michel Fliess, Jean Levine, Philippe Martin, and Pierre Rouchon. Flatness and defect of non-linear systems: Introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, 1995. ISSN 13665820. doi: 10.1080/00207179508921959.

[65] Suhyeon Gim. *Flexible and Smooth Trajectory Generation based on Parametric Clothoids for Nonholonomic Car-like Vehicles.* PhD thesis, Université Clermont Auvergne, 2017. URL https://tel.archives-ouvertes.fr/tel-01751530.

[66] R. Gobithaasan, Yip Wei, Kenjiro Miura, and Madhavan Shanmugavel. Optimal Path Smoothing with Log-Aesthetic Curves based on Shortest Distance,

Minimum Bending Energy and Curvature Variation Energy. In *Proceedings of CAD'19,*, pages 397–402, Singapore, 2019. doi: 10.14733/cadconfp.2019.397-402.

[67] J E Solanes, L Armesto, J Tornero, P Muñoz-Benavent, and V Girbés. Mobile robot obstacle avoidance based on quasi-holonomic smooth paths. In *Joint of the 13th Annual Conference on Towards Autonomous Robotic Systems, TAROS 2012 and the 15th Annual FIRA RoboWorld Congress*, volume 7429, pages 152–163, 2012. ISBN 978-3-642-32526-7. doi: 10.1007/978-3-642-32527-4. URL http://www.mendeley.com/research/dualrate-nonlinear-high-order-holds-visual-servoing-applications/.

[68] Vicent Girbés, Leopoldo Armesto, and Josep Tornero. Path following hybrid control for vehicle stability applied to industrial forklifts. *Robotics and Autonomous Systems*, 62(6):910–922, 2014. ISSN 09218890. doi: 10.1016/j.robot.2014.01.004. URL http://dx.doi.org/10.1016/j.robot.2014.01.004.

[69] Junior A.R. Silva and Valdir Grassi. Clothoid-based global path planning for autonomous vehicles in urban scenarios. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4312–4318, 2018. ISSN 10504729. doi: 10.1109/ICRA.2018.8461201.

[70] Pedro F. Lima, Marco Trincavelli, Jonas Martensson, and Bo Wahlberg. Clothoid-based model predictive control for autonomous driving. *2015 European Control Conference, ECC 2015*, pages 2983–2990, 2015. doi: 10.1109/ECC.2015.7330991.

[71] Junior Anderson Rodrigues Da Silva, Iago Pacheco Gomes, Denis Fernando Wolf, and Valdir Grassi. Sparse Road Network Model for Autonomous Navigation Using Clothoids. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):885–898, 2022. ISSN 15580016. doi: 10.1109/TITS.2020.3016620.

[72] Mogens Graf Plessen, Pedro F. Lima, Jonas Martensson, Alberto Bemporad, and Bo Wahlberg. Trajectory planning under vehicle dimension constraints using sequential linear programming. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-March:1–6, 2018. doi: 10.1109/ITSC.2017.8317665.

[73] Mariagrazia Dotoli, Alexander Fay, Marek Miśkowicz, and Carla Seatzu. An

overview of current technologies and emerging trends in factory automation. *International Journal of Production Research*, 57(15-16):5047–5067, 2019. ISSN 1366588X. doi: 10.1080/00207543.2018.1510558. URL https://doi.org/10.1080/00207543.2018.1510558.

[74] Roberto Olmi, Cristian Secchi, and Cesare Fantuzzi. An efficient control strategy for the traffic coordination of AGVs. *IEEE International Conference on Intelligent Robots and Systems*, (1):4615–4620, 2011. doi: 10.1109/IROS.2011.6048083.

[75] Kumiko Tadano and Yoshiharu Maeno. Scalable control system for dense transfer vehicles in flexible manufacturing. *2019 IEEE Conference on Control Technology and Applications (CCTA)*, (3):325–331, 2019.

[76] Ivica Draganjac, Tamara Petrović, Damjan Miklić, Zdenko Kovačić, and Juraj Oršulić. Highly-scalable traffic management of autonomous industrial transportation systems. *Robotics and Computer-Integrated Manufacturing*, 63(July 2018):101915, 2020. ISSN 07365845. doi: 10.1016/j.rcim.2019.101915. URL https://doi.org/10.1016/j.rcim.2019.101915.

[77] Toshiyuki Miyamoto and Kensuke Inoue. Local and random searches for dispatch and conflict-free routing problem of capacitated AGV systems. *Computers and Industrial Engineering*, 91:1–9, 2016. ISSN 03608352. doi: 10.1016/j.cie.2015.10.017. URL http://dx.doi.org/10.1016/j.cie.2015.10.017.

[78] Bai Li, Hong Liu, Duo Xiao, Guizhen Yu, and Youmin Zhang. Centralized and optimal motion planning for large-scale AGV systems: A generic approach. *Advances in Engineering Software*, 106:33–46, 2017. ISSN 18735339. doi: 10.1016/j.advengsoft.2017.01.002. URL http://dx.doi.org/10.1016/j.advengsoft.2017.01.002.

[79] Arthur Richards and Jonathan P. How. Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming. In *Proceedings of the American Control Conference*, pages 1936–1941, Anchorage, AK, USA, 2002. ISBN 0000000000.

[80] M. Dotoli and M. P. Fanti. Coloured timed Petri net model for real-time control of automated guided vehicle systems. *International Journal of Production Research*, 42(9):1787–1814, 2004. ISSN 00207543. doi: 10.1080/00207540410001661364.

[81] S. P. Singh and M. K. Tiwari. Intelligent agent framework to determine the optimal conflict-free path for an automated guided vehicles system. *International Journal of Production Research*, 40(16):4195–4223, 2002. ISSN 00207543. doi: 10.1080/00207540210155783.

[82] Mike Peasgood, Christopher M. Clark, and John McPhee. A complete and scalable strategy for coordinating multiple robots within roadmaps. *IEEE Transactions on Robotics*, 24(2):283–292, 2008. ISSN 15523098. doi: 10.1109/TRO.2008.918056.

[83] Jingjin Yu and Steven M. Lavalle. Planning optimal paths for multiple robots on graphs. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3612–3617, 2013. ISSN 10504729. doi: 10.1109/ICRA.2013.6631084.

[84] Jingjin Yu and Daniela Rus. An Effective Algorithmic Framework for Near Optimal Multi-robot Path Planning. pages 495–511, 2018. doi: 10.1007/978-3-319-51532-8_30.

[85] Ivica Draganjac, Damjan Miklic, Zdenko Kovacic, Goran Vasiljevic, and Stjepan Bogdan. Decentralized Control of Multi-AGV Systems in Autonomous Warehousing Applications. *IEEE Transactions on Automation Science and Engineering*, 13(4):1433–1447, 2016. ISSN 15455955. doi: 10.1109/TASE.2016.2603781.

[86] Lorenzo Sabattini, Mika Aikio, Patric Beinschob, Markus Boehning, Elena Cardarelli, Valerio Digani, Annette Krengel, Massimiliano Magnani, Szilard Mandici, Fabio Oleari, Christoph Reinke, Davide Ronzoni, Christian Stimming, Robert Varga, Andrei Vatavu, Sergi Castells Lopez, Cesare Fantuzzi, Aki Mayra, Sergiu Nedevschi, Cristian Secchi, and Kay Fuerstenberg. The PAN-robots project: Advanced automated guided vehicle systems for industrial logistics. *IEEE Robotics and Automation Magazine*, 25(1):55–64, 2018. ISSN 10709932. doi: 10.1109/MRA.2017.2700325.

[87] Valerio Digani, Lorenzo Sabattini, Cristian Secchi, and Cesare Fantuzzi. Hierarchical traffic control for partially decentralized coordination of multi AGV systems in industrial environments. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 6144–6149, 2014. ISSN 10504729. doi: 10.1109/ICRA.2014.6907764.

[88] Lorenzo Sabattini, Elena Cardarelli, Valerio Digani, Cristian Secchi, Cesare Fantuzzi, and Kay Fuerstenberg. Advanced sensing and control techniques for multi AGV systems in shared industrial environments. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2015-Octob, 2015. ISSN 19460759. doi: 10.1109/ETFA.2015.7301488.

[89] Charlie Street, Bruno Lacerda, Manuel Mühlig, and Nick Hawes. Multi-Robot Planning Under Uncertainty with Congestion-Aware Models. In N. B. An, A. Yorke-Smith, El Fallah Seghrouchni, and G. Sukthankar, editors, *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems*, page 9, Auckland, New Zealand, 2020.

[90] Jackeline Rios-Torres and Andreas A. Malikopoulos. A Survey on the Coordination of Connected and Automated Vehicles at Intersections and Merging at Highway On-Ramps. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1066–1077, 2017. ISSN 15249050. doi: 10.1109/TITS.2016.2600504.

[91] Kurt Dresner. A Multiagent Approach to Autonomous Intersection Management. *Journal of Artificial Intelligence Research*, 31:591–656, 2008.

[92] Andreas A. Malikopoulos, Christos G. Cassandras, and Yue J. Zhang. A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections. *Automatica*, 93:244–256, 2018. ISSN 00051098. doi: 10.1016/j.automatica.2018.03.056. URL https://doi.org/10.1016/j.automatica.2018.03.056.

[93] Yingfeng Zhang, Zhengfei Zhu, and Jingxiang Lv. CPS-Based Smart Control Model for Shopfloor Material Handling. *IEEE Transactions on Industrial Informatics*, 14(4):1764–1775, 2018. ISSN 15513203. doi: 10.1109/TII.2017.2759319.

[94] Behdad Chalaki and Andreas A. Malikopoulos. Optimal Control of Connected and Automated Vehicles at Multiple Adjacent Intersections. *IEEE Transactions on Control Systems Technology*, pages 1–13, 2021. ISSN 1063-6536. doi: 10.1109/TCST.2021.3082306. URL https://ieeexplore.ieee.org/document/9444611/.

[95] G Sanchez and J Latombe. Using a PRM Planner to Compare centralized and decoupled planning for multi-robot systems. In *IEEE International Conference on Robotics and Automation*, pages 2112–2119, 2002.

[96] Lubomír Bakule. Decentralized control: An overview. *Annual Reviews in Control*, 32(1):87–98, 2008. ISSN 13675788. doi: 10.1016/j.arcontrol.2008.03.004.

[97] Lorenzo Sabattini, Valerio Digani, Matteo Lucchi, Cristian Secchi, and Cesare Fantuzzi. Mission Assignment for Multi-Vehicle Systems in Industrial Environments. *IFAC-PapersOnLine*, 48(19):268–273, 2015. ISSN 24058963. doi: 10.1016/j.ifacol.2015.12.044. URL http://dx.doi.org/10.1016/j.ifacol.2015.12.044.

[98] Zijia Zhong, Mark Nejad, and Earl E. Lee. Autonomous and Semi-Autonomous Intersection Management: A Survey. *IEEE Intelligent Transportation Systems Magazine*, 3014074:1–16, 2020. ISSN 19411197. doi: 10.1109/MITS.2020.3014074.

[99] Handong Yao and Xiaopeng Li. Decentralized control of connected automated vehicle trajectories in mixed traffic at an isolated signalized intersection. *Transportation Research Part C: Emerging Technologies*, 121(March):102846, 2020. ISSN 0968090X. doi: 10.1016/j.trc.2020.102846. URL https://doi.org/10.1016/j.trc.2020.102846.

[100] Zhiyuan Du, Baisravan HomChaudhuri, and Pierluigi Pisu. Hierarchical distributed coordination strategy of connected and automated vehicles at multiple intersections. *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, 22(2):144–158, 2018. ISSN 15472442. doi: 10.1080/15472450.2017.1407930. URL https://doi.org/10.1080/15472450.2017.1407930.

[101] Tan Jian-an. Comparison of capacity between roundabout design and signalised junction design. *Swiss Transport Research Conference*, page 18, 2001. URL http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Comparison+of+capacity+between+roundabout+design+and+signalised+junction+design{#}0.

[102] Jian John Lu, Shengdi Chen, Xing Ge, and Fuquan Pan. A programmable calculation procedure for number of traffic conflict points at highway intersections. *Journal of Advanced Transportation*, 47(8):692–703, dec 2013. ISSN 01976729. doi: 10.1002/atr.190. URL http://doi.wiley.com/10.1002/atr.190.

[103] Weiming Zhao, Ronghui Liu, and Dong Ngoduy. A bilevel programming model for autonomous intersection control and trajectory planning. *Transportmetrica*

A: Transport Science, 0(0):1–25, 2019. ISSN 23249943. doi: 10.1080/23249935.2018.1563921. URL https://doi.org/23249935.2018.1563921.

[104] Liuhui Zhao, Andreas Malikopoulos, and Jackeline Rios-Torres. Optimal Control of Connected and Automated Vehicles at Roundabouts: An Investigation in a Mixed-Traffic Environment. *IFAC-PapersOnLine*, 51(9):73–78, 2018. ISSN 24058963. doi: 10.1016/j.ifacol.2018.07.013. URL https://doi.org/10.1016/j.ifacol.2018.07.013.

[105] Ezequiel Debada, Laleh Makarem, and Denis Gillet. Autonomous coordination of heterogeneous vehicles at roundabouts. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pages 1489–1495, 2016. doi: 10.1109/ITSC.2016.7795754.

[106] Ezequiel Gonzale Debada and Denis Gillet. Virtual Vehicle-Based Cooperative Maneuver Planning for Connected Automated Vehicles at Single-Lane Roundabouts. *IEEE Intelligent Transportation Systems Magazine*, 10(4):35–46, 2018. ISSN 19411197. doi: 10.1109/MITS.2018.2867529.

[107] Biao Xu, Shengbo Eben Li, Yougang Bian, Shen Li, Xuegang Jeff Ban, Jianqiang Wang, and Keqiang Li. Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections. *Transportation Research Part C: Emerging Technologies*, 93(December 2017):322–334, 2018. ISSN 0968090X. doi: 10.1016/j.trc.2018.06.004.

[108] Changliu Liu, Chung Wei Lin, Shinichi Shiraishi, and Masayoshi Tomizuka. Distributed Conflict Resolution for Connected Autonomous Vehicles. *IEEE Transactions on Intelligent Vehicles*, 3(1):18–29, 2018. ISSN 23798858. doi: 10.1109/TIV.2017.2788209.

[109] Michael W. Levin and David Rey. Conflict-point formulation of intersection control for autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, 85(January):528–547, 2017. ISSN 0968090X. doi: 10.1016/j.trc.2017.09.025. URL http://dx.doi.org/10.1016/j.trc.2017.09.025.

[110] Gabriel Rodrigues De Campos, Paolo Falcone, Robert Hult, Henk Wymeersch, and Jonas Sjöberg. Traffic Coordination at Road Intersections: Autonomous

Decision-Making Algorithms Using Model-Based Heuristics. *IEEE Intelligent Transportation Systems Magazine*, 9(1):8–21, 2017. ISSN 19391390. doi: 10.1109/MITS.2016.2630585.

[111] Tom Schouwenaars, Jonathan How, and Eric Feron. Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 5141, 2004.

[112] Tamás Keviczky, Francesco Borrelli, Kingsley Fregene, Datta Godbole, and Gary J. Balas. Decentralized receding horizon control and coordination of autonomous vehicle formations. *IEEE Transactions on Control Systems Technology*, 16(1):19–33, 2008. ISSN 10636536. doi: 10.1109/TCST.2007.903066.

[113] G. Inalhan, D.M. Stipanovic, and C.J. Tomlin. Decentralized optimization, with application to multiple aircraft coordination. *Proceedings of the 41st IEEE Conference on Decision and Control*, 1:1147–1155, 2002. ISSN 0191-2216. doi: 10.1109/CDC.2002.1184667. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1184667.

[114] Penglin Dai, Kai Liu, Qingfeng Zhuge, Edwin H.M. Sha, Victor Chung Sing Lee, and Sang Hyuk Son. A Convex Optimization Based Autonomous Intersection Control Strategy in Vehicular Cyber-Physical Systems. *Proceedings - 13th IEEE International Conference on Ubiquitous Intelligence and Computing, 13th IEEE International Conference on Advanced and Trusted Computing, 16th IEEE International Conference on Scalable Computing and Communications, IEEE Internationa*, pages 203–210, 2017. doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0050.

[115] Carlos E. Luis and Angela P. Schoellig. Trajectory generation for multiagent point-to-point transitions via distributed model predictive control. *arXiv*, 4(2):375–382, 2018. ISSN 2377-3766. doi: 10.1109/lra.2018.2890572.

[116] Lei Chen and Cristofer Englund. Cooperative Intersection Management: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):570–586, 2016. ISSN 15249050. doi: 10.1109/TITS.2015.2471812.

[117] Michael W. Levin, Stephen D. Boyles, and Rahul Patel. Paradoxes of reservation-based intersection controls in traffic networks. *Transportation Research Part A:*

Policy and Practice, 90:14–25, 2016. ISSN 09658564. doi: 10.1016/j.tra.2016.05.013. URL http://dx.doi.org/10.1016/j.tra.2016.05.013.

[118] Behdad Chalaki and Andreas A. Malikopoulos. Time-Optimal Coordination for Connected and Automated Vehicles at Adjacent Intersections. *IEEE Transactions on Intelligent Transportation Systems*, 2019. URL http://arxiv.org/abs/1911.04082.

[119] Peng Chen, Cong Yan, Jian Sun, Yunpeng Wang, Shenyang Chen, and Keping Li. Dynamic Eco-Driving Speed Guidance at Signalized Intersections: Multivehicle Driving Simulator Based Experimental Study. *Journal of Advanced Transportation*, 2018, 2018. ISSN 20423195. doi: 10.1155/2018/6031764.

[120] Tamás Tettamanti, Arash Mohammadi, Houshyar Asadi, and István Varga. A two-level urban traffic control for autonomous vehicles to improve network-wide performance. *Transportation Research Procedia*, 27(January):913–920, 2017. ISSN 23521465. doi: 10.1016/j.trpro.2017.12.160. URL https://doi.org/10.1016/j.trpro.2017.12.160.

[121] Georgia Perraki, Claudio Roncoli, Ioannis Papamichail, and Markos Papageorgiou. Evaluation of a model predictive control framework for motorway traffic involving conventional and automated vehicles. *Transportation Research Part C: Emerging Technologies*, 92(April):456–471, 2018. ISSN 0968090X. doi: 10.1016/j.trc.2018.05.002. URL https://doi.org/10.1016/j.trc.2018.05.002.

[122] Youssef Bichiou and Hesham A. Rakha. Developing an Optimal Intersection Control System for Automated Connected Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 20(5):1908–1916, 2019. ISSN 15249050. doi: 10.1109/TITS.2018.2850335.

[123] Wendan Du, Abdeljalil Abbas-turki, Abderrafiaa Koukam, and Franck Gechter. On the Safety Constraints Between Intersecting Movements of Autonomous and Connected Robots. In *International Conference on Control, Automation and Diagnosis (ICCAD 2020)*, number October, Paris, 2020.

[124] Nils Boysen, René de Koster, and Felix Weidinger. Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, 277(2):

396–411, 2019. ISSN 03772217. doi: 10.1016/j.ejor.2018.08.023. URL https://doi.org/10.1016/j.ejor.2018.08.023.

[125] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2015.09.011. URL http://www.sciencedirect.com/science/article/pii/S0968090X15003447.

[126] Alia Chebly, Reine Talj, Ali Charara, Alia Chebly, Reine Talj, Ali Charara, Maneuver Planning, Chebly Alia, Talj Reine, and Charara Ali. Maneuver Planning for Autonomous Vehicles , with Clothoid Tentacles for Local Trajectory Planning. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017. ISBN 9781538615263.

[127] Enrico Bertolazzi and Marco Frego. Interpolating clothoid splines with curvature continuity. *Mathematical Methods in the Applied Sciences*, 41(4):1723–1737, 2018. ISSN 10991476. doi: 10.1002/mma.4700.

[128] Edward Lambert, Richard Romano, and David Watling. Optimal Path Planning with Clothoid Curves for Passenger Comfort. In Oleg Gusikhin and Markus Helfert, editors, *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems*, pages 609–615, Heraklion, Crete, 2019. SCITEPRESS. ISBN 0000000221. doi: 10.5220/0007801806090615.

[129] Marco Frego, Enrico Bertolazzi, Francesco Biral, Daniele Fontanelli, and Luigi Palopoli. Semi-analytical minimum time solutions for a vehicle following clothoid-based trajectory subject to velocity constraints. *2016 European Control Conference, ECC 2016*, pages 2221–2227, 2017. doi: 10.1109/ECC.2016.7810621.

[130] D.H. Shin, S. Singh, and W. Whittaker. Path Generation for a Robot Vehicle Using Composite Clothoid Segments. *IFAC Proceedings Volumes*, 25(6):443–448, 1992. ISSN 14746670. doi: 10.1016/s1474-6670(17)50946-5.

[131] D. J. Walton and D. S. Meek. A controlled clothoid spline. *Computers and Graphics (Pergamon)*, 29(3):353–363, 2005. ISSN 00978493. doi: 10.1016/j.cag.2005.03.008.

[132] Sebastian Thrun and Arno Buecken. Integrating grid-based and topological maps for mobile robot navigation. *Proceedings of the National Conference on Artificial Intelligence*, 2(August):944–950, 1996.

[133] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT Press, Cambridge, Mass., 2005. ISBN 0262201623 9780262201629. URL http://www.amazon.de/gp/product/0262201623/102-8479661-9831324?v=glance{&}n=283155{&}n=507846{&}s=books{&}v=glance.

[134] Steven M. LaValle and Daniel Leidner. Chapter 6: Combinatorial Motion Planning. In *Planning Algorithms*, chapter 6, pages 249–310. Cambridge University Press, 2006. ISBN 9780511546877. doi: 10.1017/CBO9780511546877. URL http://ebooks.cambridge.org/ref/id/CBO9780511546877.

[135] Matthew A. Cooper, John F. Raquet, and Rick Patton. Range information characterization of the Hokuyo UST-20LX LIDAR sensor. *Photonics*, 5(2), 2018. ISSN 23046732. doi: 10.3390/photonics5020012.

[136] Robin Deits and Russ Tedrake. Computing large convex regions of obstacle-free space through semidefinite programming. *Springer Tracts in Advanced Robotics*, 107:109–124, 2015. ISSN 1610742X. doi: 10.1007/978-3-319-16595-0.

[137] Youngsang Suh, Jiseock Kang, and Dongjun Lee. A fast and safe motion planning algorithm in cluttered environment using maximally occupying convex space. *International Conference on Control, Automation and Systems*, 2020-October (Iccas):173–178, 2020. ISSN 15987833. doi: 10.23919/ICCAS50221.2020.9268421.

[138] Gaël Raballand. How do differing standards increase trade costs? THE CASE OF PALLETS Gaël. *World Bank Policy Research Working Paper 3519*, pages 1–20, 2005.

[139] Roger Bostelman, Richard Norcross, Joe Falco, and Jeremy Marvel. Development of standard test methods for unmanned and manned industrial vehicles used near humans. *Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2013*, 8756:87560P, 2013. ISSN 0277786X. doi: 10.1117/12.2019063.

[140] Matthew P. Kelly. Transcription Methods for Trajectory Optimization: a beginners tutorial. Technical report, 2017. URL http://arxiv.org/abs/1707.00284.

[141] Lecturer Pablo a Parrilo. Lecture 5 Univariate polynomials Root bounds and Sturm sequences Counting real roots. pages 1–7, 2006.

[142] MATLAB Help Center - Choosing the Algorithm. https://uk.mathworks.com/help/optim/ug/choosing-the-algorithm.html, August 2020.

[143] L. F. Shampine. Vectorized adaptive quadrature in MATLAB. *Journal of Computational and Applied Mathematics*, 211(2):131–140, 2008. ISSN 03770427. doi: 10.1016/j.cam.2006.11.021.

[144] Robert Walenta, Twan Schellekens, Alexander Ferrein, and Stefan Schiffer. A decentralised system approach for controlling AGVs with ROS. *2017 IEEE AFRICON: Science, Technology and Innovation for Africa, AFRICON 2017*, pages 1436–1441, 2017. doi: 10.1109/AFRCON.2017.8095693.

[145] Elena Cardarelli, Valerio Digani, Lorenzo Sabattini, Cristian Secchi, and Cesare Fantuzzi. Cooperative cloud robotics architecture for the coordination of multi-AGV systems in industrial warehouses. *Mechatronics*, 45:1–13, 2017. ISSN 09574158. doi: 10.1016/j.mechatronics.2017.04.005.

[146] Kurt Dresner and Peter Stone. Multiagent traffic management: A reservation-based intersection control mechanism. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2004*, 2:530–537, 2004.

[147] Abdullah Baz, Ping Yi, and Ahmad Qurashi. Intersection Control and Delay Optimization for Autonomous Vehicles Flows Only as Well as Mixed Flows with Ordinary Vehicles. *Vehicles*, 2(3):523–541, 2020. doi: 10.3390/vehicles2030029.

[148] Yuchu He, Zhijuan Jia, Yage Cheng, Zhe Li, Lipeng Wang, and Junjun Fu. Modeling and Simulation of Heterogeneous Traffic Flow in the Vicinity of Intersections Considering Communication Delay. *Chinese Control Conference, CCC*, 2020-July:5665–5670, 2020. ISSN 21612927. doi: 10.23919/CCC50068.2020.9189519.

[149] Bulent Sarlioglu. Understanding Electric Motors and Loss Mechanisms. 2016. URL https://www.irc.wisc.edu/export.php?ID=421.

[150] Drag Coefficient, 2004. URL https://www.engineeringtoolbox.com/drag-coefficient-d{_}627.html.

[151] VL Knoop. Headway models, 2020. URL https://ocw.tudelft.nl/courses/traffic-flow-theory-simulation/?view=info.

[152] Yosef Sheffi. *Urban Transportation Networks*, volume id. 1985. ISBN 0139397299.

[153] Optimization and Root Finding (scipy.optimize) minimize(method='trust-constr'), 2019. URL https://docs.scipy.org/doc/scipy/reference/optimize.minimize-trustconstr.html#optimize-minimize-trustconstr.

[154] Andrew R Conn, Nicholas IM Gould, and Ph L Toint. *Trust region methods*, volume 1. Siam, 2000.

[155] Kaizheng Wang, Yafei Wang, Lin Wang, and Jingkai Wu. Distributed Model Predictive Control for Automated Intersection with Multiple Maneuvers. In *2019 IEEE Vehicle Power and Propulsion Conference (VPPC)*, pages 1–5. IEEE, oct 2019. ISBN 978-1-7281-1249-7. doi: 10.1109/VPPC46532.2019.8952571. URL https://ieeexplore.ieee.org/document/8952571/.

[156] Jur P. Van Den Berg and Mark H. Overmars. Prioritized motion planning for multiple robots. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 2217–2222, 2005. doi: 10.1109/IROS.2005.1545306.

[157] Maren Bennewitz, Wolfram Burgard, and Sebastian Thrun. Constraint-based optimization of priority schemes for decoupled path planning techniques. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2174(July):78–93, 2001. ISSN 16113349. doi: 10.1007/3-540-45422-5_7.

[158] Valerio Digani, M. Ani Hsieh, Lorenzo Sabattini, and Cristian Secchi. A Quadratic Programming approach for coordinating multi-AGV systems. *IEEE International Conference on Automation Science and Engineering*, 2015-Octob:600–605, 2015. ISSN 21618089. doi: 10.1109/CoASE.2015.7294144.

[159] Optimization and Root Finding (scipy.optimize) minimize(method='trust-constr'), 2019. URL https://docs.scipy.org/doc/scipy/reference/optimize.minimize-trustconstr.html#optimize-minimize-trustconstr.

[160] Richard H. Byrd, Mary E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999. ISSN 10526234. doi: 10.1137/S1052623497325107.

[161] Kenneth J Waldron and James Schmiedeler. Kinematics. In *Springer handbook of robotics*, chapter 2, pages 11–36. Springer, 2016.

[162] Steven M LaValle and Daniel Leidner. Chapter 13: Differential Constraints. In *Planning Algorithms*, chapter 13, pages 715–786. Cambridge University Press, 2006. ISBN 9780511546877. doi: 10.1017/CBO9780511546877. URL http://ebooks.cambridge.org/ref/id/CBO9780511546877.

[163] Joseph Stiles Beggs. *Kinematics*. CRC Press, 1983.

[164] Pedro F. Lima, Marco Trincavelli, Jonas Martensson, and Bo Wahlberg. Clothoid-Based Speed Profiler and Control for Autonomous Driving. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2015-Octob:2194–2199, 2015. doi: 10.1109/ITSC.2015.354.