Socially Aware Path Planning For Autonomous Road Vehicles

Victor Rasidescu

A Thesis
In the Department of
Mechanical, Industrial, and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Master of Applied Science (Mechanical Engineering)

at Concordia University
Montréal, Québec, Canada

July 2024

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Victor Rasidescu

Entitled: Socially Aware Path Planning For Autonomous Road Vehicles

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Mechanical Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________ Chair
Dr. Ramin Sedaghati

______________________________ External Examiner
Dr. Youmin Zhang

______________________________ Examiner
Dr. Ramin Sedaghatir

______________________________ Supervisor
Dr. Hamid Taghavifar

Approved by

____
Dr. Sivakumar Narayanswamy, Graduate Program Director

July 23, 2024

____
Dr. Mourad Debbabi, Dean of Faculty

Abstract

Socially Aware Path Planning For Autonomous Road Vehicles

Victor Rasidescu

This research addresses the critical challenge of path planning for autonomous vehicles by integrating Social Value Orientation (SVO) into path-planning algorithms for autonomous vehicles. The framework utilizes a fuzzy logic-based system to evaluate and categorize the social values of pedestrians and vehicles in real time by considering their observed behaviors and social cues. This approach enables autonomous vehicles to make more informed and socially aware decisions, thereby enhancing their ability to interact safely with human road users. In addition, the thesis introduces an adaptive artificial potential field (APF) method that dynamically assesses the dangers presented by different road users, taking into account factors such as type, size, speed, and societal importance. By integrating road layout and traffic signal potential fields, the APF method guarantees that the autonomous vehicle can navigate intricate road conditions while prioritizing safety.

The effectiveness of the proposed path planning framework is rigorously validated using CARLA driving simulator. These simulations create a realistic and dynamic traffic environment, allowing for the thorough testing of custom behavioral profiles for various actors in different traffic situations. Results demonstrate the framework's practical applicability and effectiveness in enhancing the interaction between autonomous vehicles and human road users.

The outcomes of this research contribute to the development of safe and human-centric path-planning algorithms for highly automated vehicles, particularly in dense or mixed-traffic environments. This work represents a significant step towards creating autonomous systems that can coexist harmoniously with human drivers and pedestrians, ultimately leading to safer and more efficient roadways.

Contents

List of Figures

List of Tables

List of Nomenclature

α	Social value gain
α_{best}	Best gradient descent step size
α_l	Rear axle to look-ahead point angle
α_s	Gradient descent step size
β	Vehicle slip angle at C.G.
β_f	Front tire slip angle
β_r	Rear tire slip angle
δ_D	Derivative steering controller output
δ_i	Integral steering controller output
δ_p	Proportional steering controller output
ϵ	Error margin
κ_x	Longitudinal actor potential scaling factor
κ_y	Lateral actor potential scaling factor
μ_A	Fuzzy membership function
μ_{gauss}	Gaussian fuzzy membership function
μ_{peak}	Peak tire friction coefficient
μ_{smf}	S-shaped fuzzy membership function

μ_{zmf} Z-shaped fuzzy membership function

Ω Social value

ω_l Local frame vehicle yaw rate

$\overline{f}^L$ UMF rule firing

$\overrightarrow{e}_l^x$ Local frame x-component unit vector

$\overrightarrow{e}_l^y$ Local frame y-component unit vector

$\overrightarrow{v}_l$ Local frame vehicle velocity at C.G.

$\overrightarrow{v}_{tf}$ Front tire tangential velocity vector

$\overrightarrow{v}_{tr}$ Rear tire tangential velocity vector

ϕ Gradient descent search angle

ϕ_{max} Maximum grafeint descent search angle

ψ_l Local frame yaw angle

σ_x Actor potential longitudinal standard deviation

σ_y Actor potential lateral standard deviation

$\underline{f}^L$ LMF rule firing

$\widetilde{F}^L$ Degree of fuzzy rule firing

A Accelerator controller dead zone

a_d Accelerator controller derivative gain

A_G Acceleration vector global frame

a_i Accelerator controller integral gain

a_p Accelerator controller proportional gain

a_{RMS} Root mean square acceleration

B Braking controller dead zone

$C_{\beta f}$ Front tire cornering stiffness

$C_{\beta r}$ Rear tire cornering stiffness

C_{free} Free Configuration space

C_{obs} Obstacle configuration space

$COG(u)$ C.O.G. of fuzzy memberships

$e_G^{x/y}$ Global frame unit vector

e_G^X X-component of global frame unit vector

e_G^Y Y-component of global frame unit vector

e_G^Z Z-component of global frame unit vector

e_l Lateral path error

e_v velocity controller error

$f(n)$ A* evaluation function on node set n

F_G Stationary global road coordinate frame

F_{xf} Front tire longitudinal force

F_{xr} Rear tire longitudinal force

F_{yf} Front tire lateral force

F_{yr} Rear tire lateral force

F_y Lateral force at vehicle C.G.

g Gravitational constant

$g(n)$ Calculated cost of path from start to node n

$h(n)$ Heuristic estimate of node n to goal node

i Local frame actor longitudinal position

i Longitudinal slip %

i_c Critical longitudinal slip %

I_z Moment of inertia about z-axis

j	Local frame actor lateral position
K_d	Derivative controller gain
K_i	Integral controller gain
K_p	Proportional controller gain
K_{rep}	Repulsive potential gain
K_u	ZN ultimate gain
K_x	Longitudinal acceleration weight
K_y	Lateral acceleration weight
K_z	Vertical acceleration weight
L	Actor length
L_{AD}	Look-ahead distance
L_A	Actor length
L_f	Vehicle C.G. to front axle length
L_{LPB}	Local planner boundry distance
L_{min}	Vehicle longitudinal distance to local minima
L_r	Vehicle C.G. to rear axle length
L_{stop}	Stopping distance safety margin
m	Mass
n_{c+f}	Local planner f^{th} node in front of ego-vehicle
n_{c-r}	Local planner r^{th} node behind ego-vehicle
n_c	Global planner closest node to ego-vehicle
n_{goal}	Goal node
n_G	Global planner nodes
n_i	Global planner i^{th} node

n_L Local planner nodes

O_{xy} Local frame origin

P_x Longitudinal Gaussian order

P_y Lateral Gaussian order

q Configuration point

R Turning radius

$R_{\psi l}$ Global to local frame rotation matrix

R_M Mamdani rule base

RMS_{total} Total RMS acceleration

$S(A, B)$ Fuzzy s-norm

$T(A, B)$ Fuzzy t-norm

T_u ZN ultimate oscillation period

$U(q)$ Potential at configuration point q

$U_{att}(q)$ Attractive potential at configuration point q

$U_{rep}(q)$ Repulsive potential at configuration point q

V Actor velocity

v_{free} Obstacle-free vehicle target velocity

V_G Vehicle velocity global frame

v_{min} Minimum vehicle velocity

v_{ref} Velocity controller Reference velocity

v_x Longitudinal velocity at vehicle C.G.

v_y Lateral velocity at vehicle C.G.

W Actor width

W_A Actor width

W_{lane} Width of traffic lane

$WRMS$ Weighted root mean square acceleration

X_l X-component of local frame origin

$X_{O_{xy}}$ X-component of reference frame O

y_c Vehicle lateral position

y_{LA} Look-ahead point lateral position

Y_l Y-component of local frame origin

$Y_{O_{xy}}$ Y-component of reference frame O

The following publications have been accepted based on the research presented in this thesis:

1. Victor Rasidescu, Hamid Taghavifar. *Socially Intelligent Path-planning for Autonomous Vehicles Using Type-2 Fuzzy Estimated Social Psychology Models, IEEE Access, In Press.*

2. Victor Rasidescu, Hamid Taghavifar. *Artificial Potential Fields-Enhanced Socially Intelligent Path-Planning for Autonomous Vehicles Using Type 2 Fuzzy Systems, IEEE Intelligent Transportation Systems Conference 2024, 24-27th September 2024, Edmonton, Canada.*

Chapter 1

Introduction

1.1 Background

Autonomous vehicles (AVs) promise transformative opportunities in future transportation systems, particularly in enhancing efficiency and safety and aligning with environmental sustainability. The integration of AVs could address key issues highlighted in the 2021 report by the National Highway Traffic Safety Administration (NHTSA) [1], which showed that 18.5% of fatalities involved driving over the posted speed limit, 7% from failure to yield the right of way, and 6.6% from failing to keep in the proper lane [2]. These statistics are further summarized in Table 1.1. These identified behaviors can be used to assess the risk associated with drivers or pedestrians, thereby improving AVs' safety and autonomy.

According to the Society of Automotive Engineers (SAE), there are 5 levels of autonomy that perform part or all of the dynamic driving task [3]. Level 0 performs no automation, and level 1 provides driver assistance such as cruise control. Level 2 performs partial driving automation, where the vehicle can perform steering and acceleration tasks. Level 3 implements conditional driving automation, where the vehicle can perform most driving tasks, but human override is still required. Level 4 is described as high automation. The vehicle can perform all driving tasks under specific circumstances, but geo-fencing is still required. Finally, level 5 systems achieve full automation without requiring human attention. This automation-level system has become the standard and is summarized in Table 1.2. Currently, autonomous vehicle systems from automotive manufacturers offer up to level 3 capability as defined by the SAE standard. Mercedes-Benz DRIVE PILOT [4] is an example of such a system. The ability to incorporate the driving behavior of other road users, be it drivers, pedestrians, cyclists, or mobility scooters, represents a potential avenue towards fully autonomous level 5 systems.

Table 1.1: Driving Behaviors Reported For Drivers And Motorcycle Operators Involved In Fatal Crashes, 2021 [2]

Behavior	**Number**	**Percent**
Driving too fast for conditions or in excess of posted limit or racing	11254	18.5
Under the influence of alcohol, drugs, or medication	6835	11.2
Operating vehicle in a careless manner	4601	7.6
Failure to yield right of way	4239	7
Failure to keep in proper lane	4042	6.6
Distracted (phone, talking, eating, object, etc.)	3346	5.5
Operating vehicle in erratic, reckless or negligent manner	2615	4.3
Failure to obey traffic signs, signals, or officer	2450	4
Overcorrecting/oversteering	1845	3
Vision obscured (rain, snow, glare, lights, building, trees, etc.)	1584	2.6
Drowsy, asleep, fatigued, ill, or blacked out	1310	2.2
Swerving or avoiding due to wind, slippery surface, etc.	1278	2.1
Driving wrong way on one-way traffic or wrong side of road	1179	1.9
Making improper turn	445	0.7
Other factors	5825	9.6
None reported	9576	15.7
Unknown	19636	32.2
Total	**60904**	**100**

Advanced path planning frameworks are a cornerstone for integrating AVs into public roadways, and they necessitate a deep understanding of vehicle dynamics and environmental interaction. AVs must navigate complex environments, balancing safety and efficiency, thus requiring sophisticated algorithms capable of real-time decision-making and adaptation to unpredictable scenarios. Moreover, integrating vehicle dynamics into the path-planning process is essential for generating feasible and safe trajectories. This integration allows AVs to make more informed and interpretive decisions, considering factors like vehicle speed, turning radius, and braking capabilities, leading to smoother and safer maneuvers. Despite the progress in path planning for AVs, the development of socially aware path-planning algorithms is vital to ensure their safe integration into public roadways. Transitioning from partially to fully automated driving requires socially conscious path planning based on interpreting social cues from pedestrians and other vehicles. These cues, coupled with an understanding of vehicle dynamics and environmental interactions, allow for sophisticated, real-time decision-making. The incorporation of human-like thinking models into path planning algorithms is vital in this context, enabling the assessment of risks and anticipation of intents. This blend of technical and social awareness allows AVs to navigate complex situations safely and efficiently.

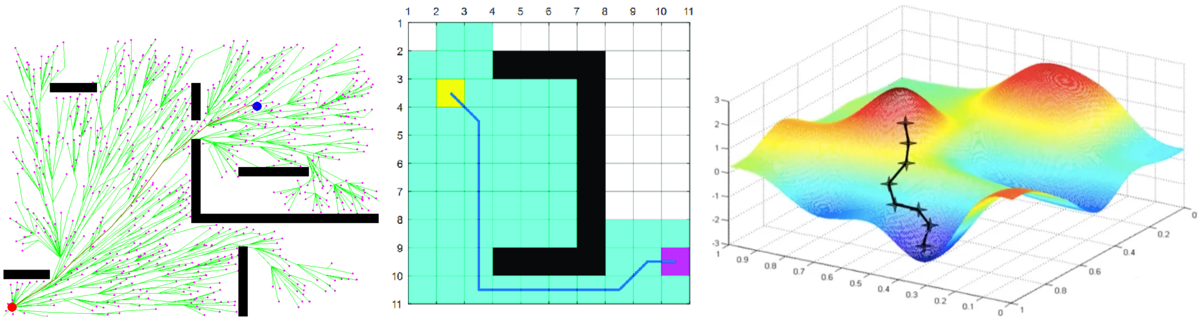
Table 1.2: Levels of Autonomy as Defined by SAE [3]

Level	**Description**
0	**No Automation:** The driver performs all driving tasks.
1	**Driver Assistance**: Vehicle is controlled by the driver, but some driving assist features may be included.
2	**Partial Automation:** The vehicle has combined automated functions like acceleration and steering, but the driver must remain engaged with the driving task and monitor the environment at all times.
3	**Conditional Automation:** The driver is a necessity, but is not required to monitor the environment. The driver must be ready to take control of the vehicle at all times with notice.
4	**High Automation:** The vehicle is capable of performing all driving functions under certain conditions. The driver may have the option to control the vehicle.
5	**Full Automation:** The vehicle is capable of performing all driving functions under all conditions. The driver is not required to be present.

High-level path planning, also known as global planning or mission-level planning, is the act of planning paths from the ego-vehicle's location to the destination, which is usually far and out of sight. The idea of distinguishing between high-level path planning and local path planning arises from high-precision path planners' limitations. Usually, incomplete or unknown map parameters would limit the usefulness of applying local planning techniques to navigation areas beyond sensor or data acquisition means. Furthermore, because of the computational complexity required for the precision needed in local planning, these would present an undue burden if applied to a very large workspace map. Methods such as Dijkstra's search algorithm were formulated to solve the question: *Given a network of roads connecting cities, what is the shortest route between two designated cities?* By posing the entire road network as a series of interconnected nodes, a series of navigational sub-objectives can be extracted by finding the shortest path through a weighted graph. Another graph-based method, `A*`, can also be adapted to high-level path planning. The issue of its significant computational complexity is mitigated by limiting the search space to only the road network and placing nodes that describe only the geometry of the road (such as the lane-center line) and not the entire road surface. The series of nodes found by graph-based approaches can then serve as sub-destinations in the local path-planning stage.

By separating macroscopic navigational tasks to the high-level planner, we can delegate more computational resources for the precision and accuracy needed for the local path-planner stage where the local path planner can acquire data through various sensor inputs (camera, ultrasonic, LiDAR, etc.) to create a high precision map of the ego-vehicle's immediate surroundings with information that would have been otherwise unavailable to a high-level planner. Furthermore, the high-level path planner can create a sequence of sub-destinations to reach the global destination, akin to leaving breadcrumbs to follow a path. This allows the local path planner to further reduce the range in which it needs to operate and can help eliminate local minima by providing closely spaced sub-destinations/goals to the local planner.

Path planning algorithms for autonomous vehicles encompass diverse state-of-the-art techniques, each with unique strengths. Sampling-based methods like Rapidly-exploring Random Trees (RRT) excel in high-dimensional spaces by efficiently exploring random paths, making them suitable for dynamic and unpredictable environments. Heuristic-based methods, such as the `A*` algorithm, offer precision and simplicity, ideal for structured environments, but can be computationally intensive in complex scenarios and easily incorporate many metrics for optimality with the appropriate heuristics [5]. Additionally, intelligent methods like reinforcement learning (RL) represent a dynamic approach, continuously learning and adapting from environmental interactions [6]. Artificial potential fields (APF) offer a unique perspective, simulating physical forces to navigate the obstacles. Khatib [7] proposed using attractive and repulsive forces to create a potential field. Multi-dimensional equations are used to represent the forces (potentials) created by the obstacles or the objective of the robot in the configuration space, where the configuration space $\mathcal{C}$ for a ground robot is usually constrained to $\mathbb{R}^2$ because the configuration space is assumed to be locally flat. Because of the problem of local minima, APFs are usually only used for local path-planning, as the problem of mitigating local minima between the robot's position and goal can become difficult for large configuration spaces [8]. This necessitates using a separate high-level path planner or minima-reducing potential field equations such as harmonic potential fields (HPF). Figure 1.1 shows path planning visualisations for common methods discussed above.



(a) RRT [9] (b) A* [10] (c) Gradient Descent [11]

Figure 1.1: Common Robotic Navigation Algorithm Visualizations

However, a critical aspect often overlooked in these algorithms is the human-centric element. Most of these methods are objective in nature and primarily focus on the technical aspect of path-finding, sometimes missing the human-centric aspect of decision-making and social interactions with pedestrians and other human-driven vehicles. This gap highlights the need for incorporating social psychological models and human behavior modeling in path planning algorithms, ensuring that autonomous vehicles can navigate efficiently and harmoniously coexist with human road users, respecting social norms and behaviors.

Social value orientation (SVO) is a social psychology concept introduced to describe a person's preference for allocating resources between themselves and another person [12]. This framework was proposed for explaining social preferences after observing the cooperation of participants [13] in social predicaments such as the Prisoner's Dilemma [14] which more simplistic *self-serving* behavioral models such as classical Game Theory did not predict. Griesinger and Livingston [12] proposed a geometric framework for SVO where a circle is used to show the spectrum of behavioral preferences.

Honjo and Kubo [15] proposed using SVO for social dilemmas in the context of nature-based tourism (NBT) tour providers. It was suggested that one can imagine the tourism dilemma as follows: Self-interested tourism firms increase tour supply to maximize their benefits. Over-exploitation of natural resources happens, undermining the basis of NBT. The economic value of NBT decreases, and the payoff allocation to tourism firms becomes inefficient. SVO is employed as a model to capture the diverse personality traits of individuals engaged in wildlife viewing tours. SVO is expressed mathematically as a function that calculates each person's utility by considering their own payoffs and the payoffs to others. The study assumes three types of players with distinct SVOs (individualistic, competitive, and prosocial). Honjo and Kubo investigated how these varying social preferences influence decision-making in the context of wildlife viewing tours, identifying which conditions lead to

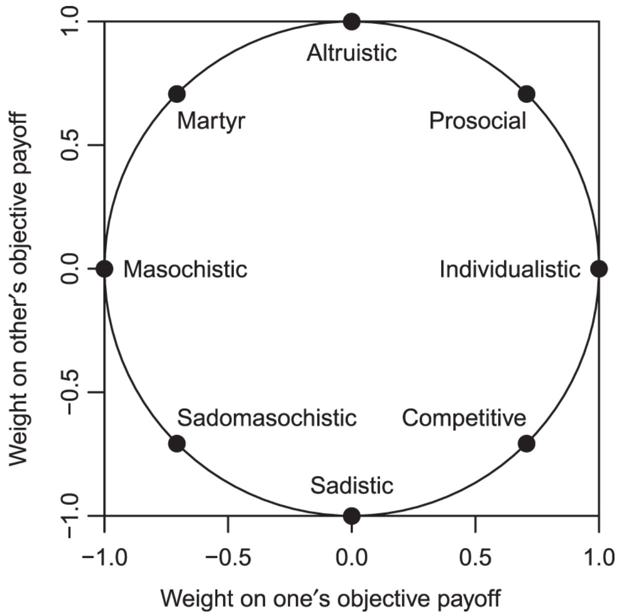


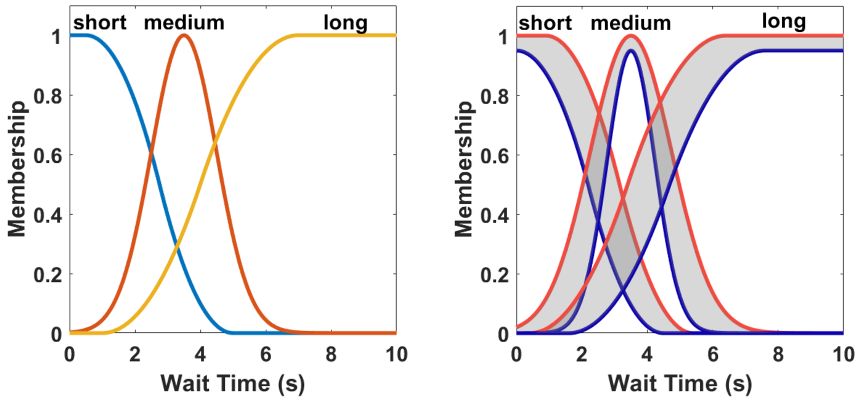
Figure 1.2: Social Value Orientation Ring [12]

.

Pareto-inefficient Nash equilibrium (PINE). Unlike traditional game-theoretic studies that often assume individualistic players maximize their own payoffs without considering others, the incorporation of SVO allows for a more realistic representation of human behavior.

As with most fluid definitions of human behavior, all actions exhibit some level of egotism (outcome to self) and cooperativeness (outcome for others). The accurate classification of the behavioral type of road users without explicit communication of their intent is a difficult process and will rely on non-verbal actions inherent in driving situations. The identification, processing, and estimation of these behaviors can be used to influence the path-planning protocol used by autonomous vehicles. Different social cues, such as a pedestrian's distance to a crosswalk, time waiting at a crosswalk, or the direction they are looking at, can serve as an input vector to an estimation process. Likewise, a vehicle's speed relative to general traffic or the speed limit, its driving smoothness, its following distance (in seconds) to the proceeding vehicle, its lane centering, and its number of lane changes per minute (zigzagging) can all serve as an input vector to obtain a social value to classify their cooperativeness.

Given the complexities of sociality and subjective human behavior interpretation, a systematic framework is essential for approximating the SVO of the ego vehicle and other road users. The fuzzy inference system is an ideal candidate due to its adeptness at capturing behavioral nuances and uncertainties. In this context, the fuzzy process is used as an esti-



(a) Type-1 Fuzzy set (b) Type-2 Fuzzy Set

Figure 1.3: Fuzzy Membership Functions With *Type-2* Visualization

mator to receive outputs from a behavioral input vector to estimate an actor's social value from $[-1, 1]$ as seen in Figure 1.2. Fuzzy estimation processes serve as a linguistic model to determine the *degree of truth* of someone's social value. Contrary to classical logical expressions, which just allow for Boolean outcomes, fuzzy logic allows for degrees of truth using separate membership functions for each input and output variable, providing a means for representing the uncertainty of a system. In this context, the input vector to a fuzzy estimator can define a set of observed quantified behaviors, with each membership function describing that behavior through a set of linguistic variables such as *low*, *medium*, and *high*. Those variables are then fuzzified and defuzzified via the output membership functions and fuzzy ruleset to obtain a value from $[-1, 1]$ on the SVO circle. Criticism of basic *type-1* fuzzy sets usually revolves around the fact that no uncertainty is incorporated in the membership functions themselves. *Type-2* fuzzy sets introduce a second membership function to each *type-1* set to represent uncertainty. This can be extended to *type-n* fuzzy sets as proposed by Zadeh [16]. All behaviors exhibit some level of egotism and cooperativeness, which seems particularly well suited to fuzzy logic, which allows for *degrees of truth* as a means to represent the uncertainty of a system as opposed to a more rigid or Boolean labeling process. Figure 1.3 depicts the input membership functions for type-1 and type-2 fuzzy sets.

1.2 Literature Review

1.2.1 High-Level Planning

Because of the computational complexity involved in high-accuracy local path planners, the local minima problem, and incomplete obstacle knowledge of distant areas, the path planner operation is divided into high-level and local-level planners. When using local path planners such as gradient descent, the issue of getting stuck in local minima instead of finding the global minima of the potential field is a major concern. By carefully choosing the shape and size of the potential field equations, it is possible to reduce the likelihood of getting stuck in local minima. Harmonic potential fields (HPF) were first proposed by C. Connolly [17] because they do not exhibit spurious local minima. The work was expanded upon by Daily *et al.* [18] where they apply HPF to high-speed road vehicles for obstacle avoidance. Other works applying HPF to the constraints of autonomous vehicle navigation are found in [19, 20]. Additional methods that build on artificial potential fields, such as the advanced fuzzy potential field method (AFPFM), were proposed by Park *et al.* [21], which used a Takagi-Sugeno fuzzy system to avoid local minima.

While harmonic potential field methods can increase the size of the potential field without exhibiting spurious local minima for a gradient descent path planner for a more unified solution, having two completely separate planning protocols is also possible. For the autonomous vehicle problem, information such as road geometry, intersection locations, lane configurations, and even traffic congestion is usually known. It can serve as sufficient information for a high-level objective planner. Graph-based search methods such as Dijkstra's search algorithm pose the entire road network as interconnected nodes. A series of navigational sub-objectives can be extracted by finding the shortest path through a weighted graph. Another graph-based method, `A*`, can also be adapted to high-level path planning. The issue of its significant computational complexity is mitigated by limiting the search space to only the road network and placing nodes that describe the geometry of the road (such as the lane-center line) and not the entire road surface. The series of nodes found by graph-based approaches can then serve as local objective points in the local path-planning stage.

Kavraki *et al.* [22] proposed probabilistic roadmaps (PRMs) to create a roadmap by randomly sampling the configuration space. Because of its random sampling, PRMs are probabilistically complete as the number of sampled points increases. By limiting the high-level workspace to the known road geometry ahead, PRM can sample random points, convert them to a weighted connected graph, and solve them using graph traversal techniques such as Dijkstra's algorithm. Qiao *et al.* [23] proposed a PRM approach optimized for narrow features such as roadways when viewed on a macroscopic level. PRMs can also be useful when the roadway map is known but does not necessarily contain precise navigational waypoints to input directly into a graph-based search algorithm.

1.2.2 Local Path Planning

Many methodologies have been developed to generate a suitable path from an artificial potential field. One of the most straight forward is the gradient descent method [24], where the gradient $\nabla = \frac{\partial}{\partial x}\hat{\imath} + \frac{\partial}{\partial y}\hat{\jmath} + \frac{\partial}{\partial z}\hat{k}$ of a potential field U(q) is performed iteratively until a minima is reached such that $\nabla U(q) = 0$. The advantage of this approach is its relatively fast computation time and is well suited for continuous real-time optimization. It is also possible to handle some nonholonomic properties by constraining the angle at which the gradient descent can be performed. However, gradient descent comes at the expense of its sensitivity to initial conditions, making it prone to converge to local minima. Harmonic potential fields (HPF) [17] and advanced fuzzy potential field method (AFPFM) [21] discussed previously for high-level planning can also be applied to the local planning level to minimize the risk of local minima.

`A*` is one of the earliest proposed path-planning algorithms and is useful for its completeness, optimality, and efficiency [25]. The configuration space is simplified into a mesh of nodes to form a weighted graph, which the `A*` algorithm traverses. Conditions for optimality can be specified to the node weightings, such as shortest path, shortest time, etc. A major drawback of the `A*` algorithm is its extensive computational time and memory complexity. The memory complexity is expressed as $O(b^d)$, where b represents the branching factor (average number of successors per node state). This challenges real-time high-speed applications, such as autonomous road vehicle navigation. Improvements have been introduced to the `A*` algorithm to address its shortcomings. Wang *et al.* [26] proposed the EBS-`A*` that introduces expansion distance, bidirectional search, and smoothing into path planning. Xiang *et al.* [27] has also shown improvements to memory efficiency and convergence speed by removing unnecessary nodes and retaining those at path inflection points.

Rapidly exploring random trees (RRT) is also a common algorithm for robotic navigation proposed by LaValle [28]. RRT grows a node tree with its root node at the starting configuration. It attempts random node connections within the configuration space $\mathcal{C}$ and completes the connection if it is feasible (no obstacle collisions, obeys constraint rule-set and distance to node respects growth factor). Due to RRT's random search, it can only achieve probabilistic completeness, where the accuracy of a solution is dependent on the computation time. This also causes RRT to be non-deterministic, where solutions can vary between two identical configuration spaces, and it does not guarantee optimality. However, RRT is well-suited for environments with complex geometries or system constraints. Improvements to the RRT algorithm have been proposed, such as the popular RRT* [29] method, where nodes are assigned a cost function denoting the shortest path from the start. Exploring nearby nodes within radius r and optimizing connections minimizes the cost function, yielding the shortest path. RRT*-Smart [30] uses path optimization and intelligent sampling to accelerate the convergence rate of RRT*.

1.2.3 Behavioral Labelling

The identification of a road user's behavioral profile to determine their degree of cooperativeness or egotism presents opportunities for research as there are many proposed approaches to the problem.

Stackelberg game theory is often proposed in the labeling process of driver aggressiveness. Schwarting *et al.* [31] proposed "driving as a game" in conjunction with SVO to label the sociality of drivers where the driving agents maximize their accumulated reward over time. At each point in time, the agent receives a multi-factorial reward involving an agent's progress towards their goal, comfort, delay, the distance between cars, and other driver priorities. To incorporate SVO into their decision-making process, it was proposed to consider the angle ϕ formed between the horizontal axis and the social label in the right-hand plane seen in Figure 1.2. On this basis, an altruistic actor has an angle $\phi \approx \pi/2$, an individualistic agent $\phi \approx 0$, and a competitive agent $\phi \approx -\pi/4$. To solve for the angle ϕ, using a utility function $g_1 = r_1 cos(\phi_1) + r_2 sin(\phi_1)$ was proposed, where the values of r_1 and r_2 represent the "reward to self" and the "reward to other" from the game's output, respectively.

Zadeh proposed a fuzzy set theory based on previously established logical sets by Lukasiewicz and Tarski. Unlike classical logic with Boolean outcomes, fuzzy logic employs separate membership functions for input and output variables, enabling the representation of uncertainty in systems, assessing the degree of truth regarding social value by analyzing observed quantified behaviors characterized by linguistic variables such as *low*, *medium*, and *high*. Deb *et al.* [32] conducted a survey where respondents rate the frequency with which they engage in different types of road-using behaviors as pedestrians and differentiated pedestrian behaviors into five-factor categories: violations, errors, lapses, aggressive behaviors, and positive behaviors. M. Lanzer *et al.* [33] evaluated pedestrian crossing, gaze, and gesture behavior. Substantial agreement was found between self-reported and video-observed data for crossing action. The identification and quantification of these behaviors can serve as an input vector and ruleset to a fuzzy estimation process to obtain a value from [-1,1] on the SVO circle. Criticism of basic *type-1* fuzzy sets usually revolves around the fact that no uncertainty is incorporated in the membership functions. Zadeh improved on his original fuzzy logic work by formulating *type-2* sets and generalizing these formulations as *type-n* sets. To obtain a crisp output, *type-2* fuzzy sets are type-reduced to *type-1* sets. Type-reduction methods such as the Karnik-Mendel method (KM-method) [34] is commonly used. The proposed improvements on the KM-methods can be found in works [35], [36] as well as the enhanced KM-method [37]. Chen *et al.* compare the computational efficiency of various type-reduction methods using big `O` notation [38]. Fuzzy estimators present the advantage of not needing large datasets to learn behaviors at the expense of their ability to notice micro-behaviors or spatiotemporal behavioral sequences.

Deep neural network (DNN) approaches have become prominent in mimicking human-like processes. The topic of spatiotemporal prediction of action sequences can be made possible through data acquisition from camera feeds or other sensors. The work of Pihrt *et al.* [39] used the large dataset provided by the Lyft Motion Prediction for Autonomous Vehicles where ride-sharing company Lyft provided a large dataset for a competition to predict the trajectories of other traffic participants. Their approach involved using a deep convolutional neural network (CNN) and recurrent networks (RNN). Pihrt *et al.* explored sequences of driving data using multiple architectures such as ConvLSTM, PsyDNet, and PredRNN, amongst others, to compare their ability to predict drivers' behavior. Detecting behavioral anomalies that stem from human susceptibilities, such as fatigue or aggression, could be used to proactively quantify AV safety relative to human drivers, such as explored in the works of Ryan *et al.* [40]. This work proposed an end-to-end model using convolutional neural networks (CNN) to compare human and AV driving behaviors. Contextual driving anomalies were detected using Gaussian processes (GP), and their frequency and severity were used to derive a risk score.

While the trends and predictions that DNN approaches have the potential to achieve are a very promising avenue of exploration, the difficulty in acquiring and managing the large datasets needed for these methods proves to be a large hurdle with these methods. Furthermore, it can be difficult to provide justification to traffic regulators with non-observable methods.

1.3 Problem Statement

The inability to classify the risk or cooperativeness of individual road users can lead to overly cautious driving behavior, paralysis of the autonomous vehicle, or dangerous interactions in uncertain situations such as right-of-way merges. Incorporating social awareness into path-planning algorithms can help address these issues. This work will focus on identifying and estimating behavioral types through social value orientation (SVO) and its incorporation into a path-planning algorithm for autonomous vehicles.

Additionally, a reliable framework for identifying and labeling behaviors must be established. This label should be utilized effectively to influence the ego vehicle's path planner. Furthermore, applying the behavior label must also consider the vehicle's nonholonomic characteristics when influencing its path.

The traffic data [1] serves as a motivator for developing socially aware path-planning algorithms. The ability to predict the likelihood of a vehicle or pedestrian performing a dangerous maneuver based on their social cues enables AVs to attribute the correct level of risk to them. This classification not only aids in avoiding potential hazards but also reduces overly cautious planned paths and maneuvers. Consequently, AVs can adapt their behavior based on other road users' perceived cooperativeness and safety consciousness, promoting smoother traffic flow and minimizing unnecessary disruptions. This approach to path planning addresses specific risk factors identified by the NHTSA and fosters a more nuanced and responsive interaction between AVs and the diverse behaviors exhibited by human drivers on the road.

Currently, no reliable framework for identifying road users' behavior profiles is in effect. This work seeks to implement interval type-2 fuzzy logic estimators (IT2FL) to estimate road users' social value (SVO). By identifying key behaviors and actions, an IT2FL estimator can estimate the social value with a human interpretable input, output, and inference system, allowing the system to be tuned for many demographics without the need for expansive datasets and allowing transportation regulators to validate the safety of the system.

1.4 Research Scope and Objectives

This work aims to develop a framework that can identify and label the cooperativeness of individual road users, as well as apply this label to the autonomous vehicle's path planner. This will aid the path-planning algorithm in attributing the appropriate amount of risk to the analyzed behavior to reduce overly cautious driving around actors who demonstrate a willingness to cooperate. Likewise, it will take more cautious routes around those who behave anti-socially.

Additionally, this work seeks to demonstrate the effectiveness of this framework by creating a simulation environment within the CARLA driving simulator, where custom behavioral profiles are created around pedestrians and vehicles in the simulation environment and then analyzed by the autonomous vehicle to determine their sociality in real-time. This implementation allows for extensive simulations, replicating complex traffic situations, evaluating the performance of our model, and laying the foundation for better interactions between autonomous and non-autonomous road users. A complete autonomous navigation framework will be developed, incorporating high-level route planning from start point to destination, local-level planning that analyzes the perceived environment of the AV, and a control system to follow the proposed planned path.

The path planner is separated into high-level and local-level path planners to achieve the navigation goal. This separation is created to utilize the best information available at the perception and global levels. The high-level path planner performs an `A*` search with Euclidean distance heuristic on all the possible navigation paths from the starting position to the desired destination. The roadway is meshed into a series of nodes, distinguishing between lanes and intersections to provide a continuous list of sub-destination points to reach the global destination using `A*`. This list of sub-destinations is utilized by the local path planner to apply appropriate repulsive potentials representing the local lane curvature and correct attractive potential to each sub-destination.

Because the high-level path planner provides the optimal route, the local planner can take this information as a series of sub-goals to achieve the global goal. Artificial potential fields are created from the perceived environment at the local level. The potential field for the local planner is created by the summation of the individual potential functions of each actor, as well as road geometry and traffic signals. The summation of potential functions represents the *spatial risk*, where the potential functions are multi-factorial. Moreover, an actor's type, size, relative speed, and heading are used to construct their potential function. Furthermore, the sub-goals obtained from the high-level planner serve as an attractive potential, whereas actors and obstacles act as repulsive ones. The size and granularity of the field can be dynamically adjusted to adapt to computational limitations.

The gradient descent method is used to generate a path from the potential field of the local planner for efficiency. The local-level planner is already deemed feasible and optimal by limiting the potential field to only the local planning and providing an optimal high-level path using `A*`. As such, any local minima remaining in the local planner arise from a set of potentials that require the vehicle to stop. These desirable local minima ensure the vehicle does not hit any obstacles when the road constraints and traffic configuration have created the local minima. Furthermore, the gradient descent path planner incorporates the nonholonomy of the vehicle as obtained from the vehicle kinematics model. This ensures that the proposed path is feasible and can be used directly as an input to a velocity and steering control system.

The social value of actors (vehicles and pedestrians) is used to modify the planned path with a robust navigation framework created. To accomplish this goal, the social value orientation circle, as shown in Figure 1.2, is used as a geometric template for the estimation process. The estimated social value $\Omega = [-1, 1]$ is used to modify the repulsive potential force of actors where an actor labeled as cooperative will have a decreased repulsive force, whereas an egoistic actor will have theirs increased.

To determine the social value of road users, a set of behaviors is identified for pedestrians and vehicles. These identified behaviors are explicitly observable, quantifiable, and linguistically definable. The identified behaviors are summarized in Table 1.3. Furthermore, the quantity or duration observed for each identified behavior is decomposed into the social value circle.

Table 1.3: Identified Actor Behavior Vectors

Actor Type	Identified Behaviors
Pedestrian	**Distance to crosswalk:** The Euclidean distance a crosswalk beginning point (Where the crosswalk meets the sidewalk) in meters. **Wait time:** The time in seconds that the pedestrian has been waiting stationary at a crosswalk before crossing. **Look time:** The time in seconds that the pedestrian has been looking at oncoming traffic before crossing. Pedestrians can look at traffic before they reach the crosswalk.
Vehicle	**Speed limit**: Travel speed expressed as a percentage of the posted road speed limit. **Follow time**: Time in seconds to the proceeding vehicle (also known as time to impact) **Lane Changes**: Lane changes per minute performed by the driver. **Lane Centering**: Rolling average of the a vehicle's distance from the center of the lane.

A fuzzy estimation process is used to translate the observed behavior into a crisp value linguistically to obtain a social value from the input vector of observed behaviors. A commonsense approach was used to linguistically define the sociality of behavior through the use of tuned input membership functions and the appropriate fuzzy ruleset. The output memberships represent the social value orientation circle, where the crisp output value represents the social value of that actor. To account for the uncertainty of modeling through linguistic variables and any noise or uncertainty during the measurement of the behavior, an interval type-2 fuzzy (IT2FS) estimation process was used. The IT2FS's footprint of uncertainty (FOU) is adjusted by the uncertainty in linguistically defining and measuring each behavior action.

This work provides a simulation model through CARLA Simulator [41] software to observe the efficacy and applicability of the proposed framework. Custom behavioral profiles for actors are created to obtain a realistic simulation environment that also allows the autonomous vehicle to observe the behavior of road users for their social value estimation process, in addition to creating realistic navigation environments.

The contributions of this work can be summarized as follows:

- **Integration of a Social Psychological Model into Path Planning:** The thesis introduces a new approach to incorporating Social Value Orientation (SVO) into the path-planning algorithms of autonomous vehicles. This integration allows the autonomous vehicle to account for other road users' social behavior and intentions, enhancing the vehicle's ability to make socially aware decisions and improve safety and harmony on the road.

- **Development of a Fuzzy Logic Estimator for Real-Time Social Value Assessment:** The research develops a fuzzy logic-based system for the real-time assessment and labeling of the social value of pedestrians and vehicles. This estimator uses observed behaviors and social cues to assign a social value, providing a dynamic and responsive method to gauge the cooperativeness or aggressiveness of other road users. In particular, Type-2 fuzzy estimators are used to model the uncertainty of the social value estimators for pedestrians and vehicles. The type-2 fuzzy estimators are type-reduced using the Enhanced Karnik-Mendel.

- **Simulation and Validation in a Realistic Driving Environment:** The research utilizes the CARLA Simulator to create a comprehensive simulation environment that includes custom behavioral profiles for various actors. This setup allows for extensive testing and validation of the proposed path planning framework in realistic and dynamic traffic scenarios, demonstrating the approach's practical applicability and effectiveness.

- **Advanced Potential Field for Dynamic Risk Representation:** The research introduces an advanced artificial potential field (APF) method that dynamically represents the risks posed by various road actors based on their type, size, speed, and social value. This method incorporates road geometry and traffic signal potential fields, enabling the autonomous vehicle to navigate complex road environments while maintaining safety. Furthermore, the nonholonomic constraints of the autonomous vehicle are used to obtain an executable path for the AV to follow.

Chapter 2

Simulation Environment

2.1 CARLA Driving Simulator

CARLA [41] is an open-source driving simulator, that encourages collaboration and innovation within the research community. Researchers can access the source code, modify existing functionalities, or develop new features to suit their specific research requirements. Additionally, CARLA supports the integration of external modules and plugins, enabling researchers to extend its capabilities further. CARLA offers a highly realistic and flexible environment for simulating various urban driving scenarios. CARLA is also used by industry partners such as Intel for the development of off-road autonomous vehicle simulators [42] as well as Toyota to aid in autonomous vehicle and computer vision development [43].

2.1.1 Physics Engine

CARLA Simulator uses Unreal Engine 4 (UE4) [44] for its 3D computer graphics engine, providing an environment with high visual fidelity. Furthermore, motion and physics simulation in CARLA is managed by Nvidia PhysX[45]. PhysX allows for the simulation of multiple aspects of autonomous vehicles, such as wheel and tire dynamics. The simulation capabilities of PhysX for autonomous vehicles and robotics applications are summarized in Table 2.1.

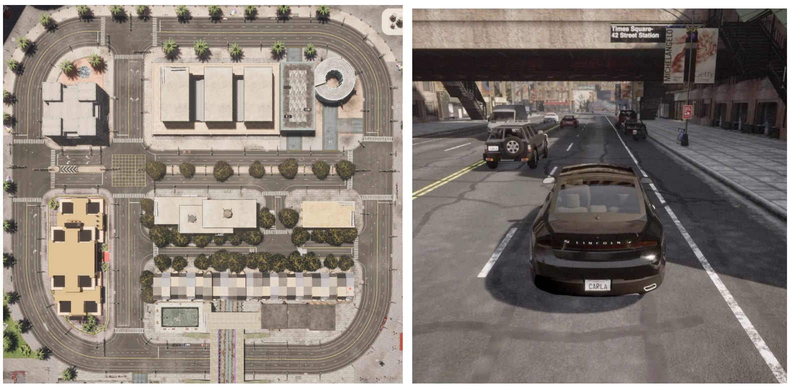
Table 2.1: Nvidia PhysX Solvers [45]

Solver	AVs	Game Dev.	Industrial App.	Robotics
Rigid Body Dynamics		✓	✓	✓
Scene Query	✓	✓	✓	✓
Joints	✓	✓	✓	✓
Articulations	✓	✓	✓	✓
Vehicle Dynamics	✓	✓	✓	✓
Character Control		✓	✓	✓
Soft Body Dynamics			✓	✓
Position Based Dynamics			✓	✓
Flow		✓	✓	✓

2.2 Roadway and Navigation Information

2.2.1 Road Layout

CARLA Simulator offers many driving maps. For this work, the urban driving map *Town 10* was selected for its many pedestrian and vehicle interaction points, its varied intersections and relatively slow travel speed. The compactness of this map also permits rapid testing, fast loading and constraining environmental variables. Figure 2.1 shows the road layout of CARLA's *Town 10* environment and the third person view of the ego-vehicle.



(a) Aerial View (b) Third Person View

Figure 2.1: CARLA Simulator Town 10

2.2.2 ASAM OpenDRIVE

The ASAM OpenDRIVE [46] format provides a common base for describing road networks with extensible markup language (XML) syntax. The data that is stored in an OpenDRIVE file describes road geometry, lane configurations, and objects, such as road markings and traffic signals.

OpenDRIVE provides a road network description that can be fed into simulators to develop and validate advanced driver assistance systems (ADAS) and AV features. Most importantly, OpenDRIVE provides a standardized format for road descriptions, which enables the industry to reduce the cost of creating and converting these files for their development and testing purposes and researchers to develop new autonomous vehicle features. [46]

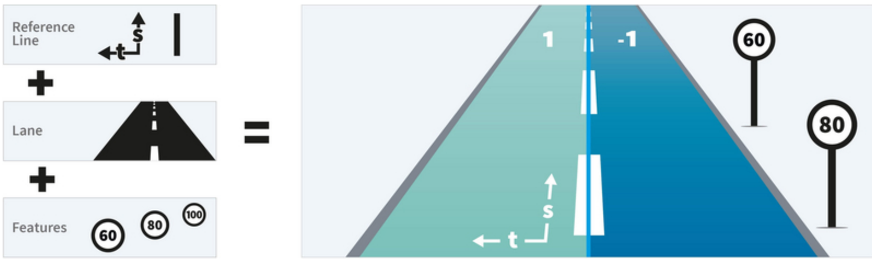
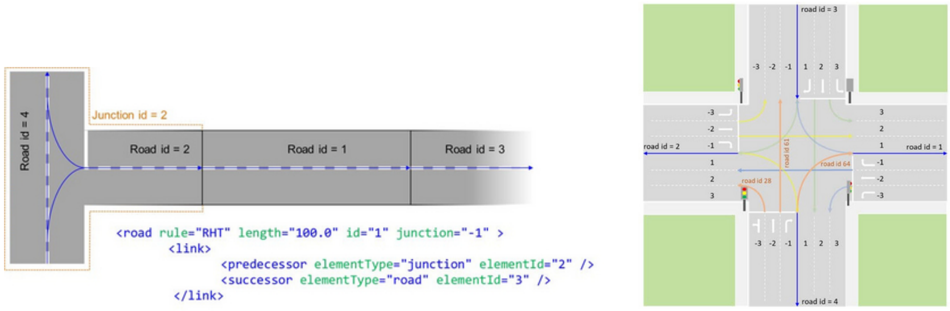


Figure 2.2: OpenDRIVE Roadway Features [46]

The ASAM OpenDRIVE road network is modeled along the reference line, which is the core piece of every road. Roads, lanes, and their elevation profiles are all attached to the reference line. Objects representing features, such as signals, can be placed by using either the reference line or the global coordinate system [46] as shown in Figure 2.2.

In an OpenDRIVE file, the overall road network is composed of individual sections interconnected with each other as shown in Figure 2.3. These links can support the driving logic of simulated traffic [46], especially for routing purposes, and provide a framework in which high-level path planners, such as A*, can operate quickly and accurately. Additionally, the individual lanes on roads are also distinguished, allowing for high-precision turn-by-turn directions when applying route planning algorithms, which is of particular importance in this work.

The information provided by OpenDRIVE can be used to plan paths for autonomous vehicles. It includes the correct lane, lane center line, road speed limit, and other road signals.



(a) Road Segments (b) Lane Segments

Figure 2.3: OpenDRIVE Roadway Geometry [46]

2.3 Sensors

2.3.1 Collision and Invasion Detector

The collision detection sensor registers an event each time its parent actor collides with another item by verifying if the parent actor's bounding box has entered the item's bounding box. Every collision sensor generates a collision event for each collision occurring in a frame. Collisions with multiple other actors can generate multiple collision events within a single frame.

Similarly to the collision sensor, the lane invasion sensor registers an event each time its parent crosses a lane marking. The sensor uses road data provided by the OpenDRIVE description of the map to determine whether the parent vehicle is invading another lane by considering the space between wheels.

2.3.2 Cameras

CARLA offers multiple camera types and performs some of the data post-processing. The following camera types are available naively through CARLA's API [41]:

RGB Camera acts as a regular camera capturing images from the scene. For added realism, it has the following disturbance/noise parameters:

- Vignette: Darkens the border of the screen.
- Grain jitter: Adds some noise to the render.
- Bloom: Intense lights burn the area around them.
- Auto exposure: Modifies the image gamma to simulate the eye adaptation to darker or brighter areas.

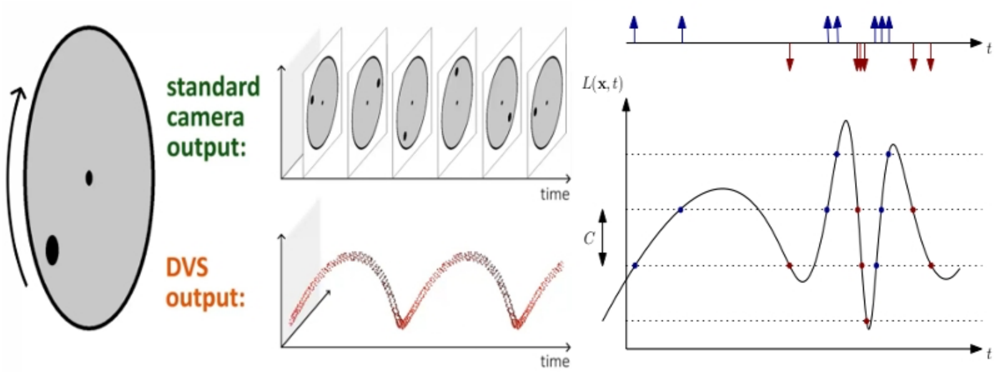
- Lens flares: Simulates the reflection of bright objects on the lens.
- Depth of field: Blurs objects near or very far away from the camera.

Semantic Segmentation Camera This camera classifies every object in sight by displaying it in a different color according to its tags (e.g., pedestrians in a different color than vehicles). When the simulation starts, every element in the scene is created with a tag. So, it happens when an actor is spawned. The objects are classified by their relative file path in the project.

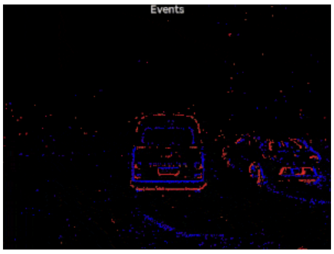
Instance Segmentation Camera Similar in effect and output to the semantic segmentation camera, this camera classifies every object in the field of view by class and instance ID. When the simulation starts, every element in the scene is created with a tag. Thus, it happens when an actor is spawned. The objects are classified by their relative file path in the project.

DVS Camera A Dynamic Vision Sensor (DVS) or Event camera is a sensor that works radically differently from a conventional camera. Instead of capturing intensity images at a fixed rate, event cameras measure changes in intensity asynchronously in the form of a stream of events, which encode per-pixel brightness changes. Event cameras possess distinct properties when compared to standard cameras. They have a very high dynamic range (140 dB versus 60 dB), no motion blur, and high temporal resolution (in the order of microseconds). Event cameras are thus sensors that can provide high-quality visual information even in challenging high-speed scenarios and high dynamic range environments, enabling new application domains for vision-based algorithms.

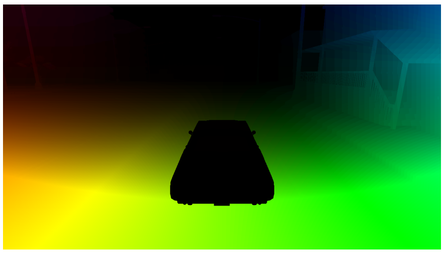
Optical Flow Camera The Optical Flow camera captures the motion perceived from the point of view of the camera. Every pixel recorded by this sensor encodes the velocity of that point projected to the image plane.



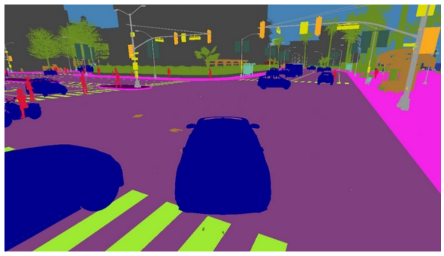
(a) DVS Camera Diagram



(b) DVS Output



(c) Optical Flow



(d) Semantic Segmentation

Figure 2.4: CARLA Camera Output & Diagrams [41]

2.3.3 Distance and Ranging

Distance and ranging sensors do not necessarily create a human interpretable output but return a depth map of the obstacles surrounding the sensor. Visualizations of distance and ranging methods are shown in Figures 2.4 and 2.5. CARLA offers the following depth and ranging sensors [41]:

LiDAR This sensor simulates a rotating LiDAR implemented using ray-casting. The points are computed by adding a laser for each channel distributed in the vertical FOV. The rotation is simulated by computing the horizontal angle at which the LiDAR rotates in a frame. The point cloud is calculated using a ray-cast for each laser in every step. For added realism, it has the following disturbance/noise parameters:

- General drop-off: Proportion of points that are dropped off randomly.
- Intensity-based drop-off: For each point detected, an extra drop-off is performed with a probability based on the computed intensity. Ray intensity is as:

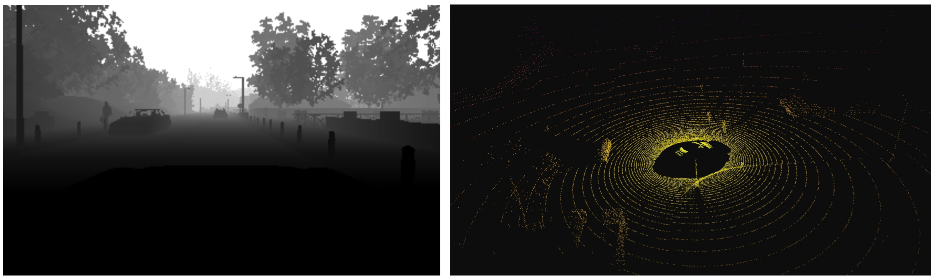
$$\frac{I}{I_0} = e^{-a \cdot d}$$

where a is the attenuation coefficient, which quantifies how much the intensity of the laser ray decreases per unit distance, and d is the distance between the LiDAR sensor and the detected point.

– Noise model to simulate unexpected deviations that appear in real-life sensors. For positive values, each point is randomly perturbed along the vector of the laser ray.

Radar The sensor creates a conic view translated to a 2D point map of the elements in sight and their speed regarding the sensor. This can be used to shape elements and evaluate their movement and direction. Due to the use of polar coordinates, the points will concentrate around the center of the view.

Depth Camera The camera provides raw data of the scene, codifying the distance of each pixel to the camera (also known as depth buffer or z-buffer) to create a depth map of the elements. Operates like an ideal stereoscopic camera image.



(a) Depth Camera (b) LiDAR

Figure 2.5: CARLA Ranging Sensors [41]

2.3.4 Position and Inertia

Position and inertial sensors are crucial for determining the vehicle's spatial orientation and dynamic motion. These sensors return the absolute position of the vehicle and the relative motion of the sensor, respectively, providing essential data for navigation and control systems. CARLA provides the following position and inertial sensors [41]:

GPS Reports the current global position of its parent object. The GPS sensor provides latitude, longitude, and altitude data, essential for geolocation and navigation tasks. This data allows for accurate mapping and localization within a global coordinate system.

Inertial Measurement Unit (IMU) Provides measurements that an accelerometer, gyroscope, and compass would retrieve for the parent object. The IMU collects data on linear acceleration, angular velocity, and orientation, which are critical for understanding the vehicle's motion dynamics. This information is used to infer the vehicle's velocity, orientation, and changes in motion, aiding in stability control and navigation.

2.4 Traffic Management

CARLA offers realistic vehicle and pedestrian behaviors. Vehicles follow traffic rules, such as lane-keeping, traffic lights, and yielding right-of-way, making the simulation environment conducive to studying interactions between AVs and other road users. Pedestrians exhibit natural movement patterns, including walking, crossing streets, and reacting to vehicle movements.

2.4.1 Traffic Management Architecture

The traffic management module of CARLA occurs on the client side. It is organized into stages for independent operations and goals, enhancing computational efficiency. Each stage operates on a separate thread, communicating synchronously in a one-way flow. Users can customize traffic flow by adjusting parameters to dictate or encourage online and offline behaviors. This flexibility is crucial for simulating real-world scenarios and training driving systems under diverse conditions [41].

The traffic manager is divided into multiple tasks and multiple stages to provide adequate traffic flow. Table 2.2 and 2.3 summarize the different stages of traffic state storage and the stages of traffic management and support the system block diagram shown in Figure 2.6.

Table 2.2: CARLA Simulation Storage [41]

Component	**Description**
Actor Life-cycle & State Management (ALSM)	Scans the world to keep track of all the vehicles and walkers present and to clean up entries for those that no longer exist. All the data is retrieved from the server and is passed through several stages.
Vehicle Registry	Contains an array of vehicles on autopilot (controlled by the TM) and a list of pedestrians and vehicles not on autopilot (not controlled by the TM).
Simulation State	cache store of the position, velocity, and additional information of all the vehicles and pedestrians in the simulation.

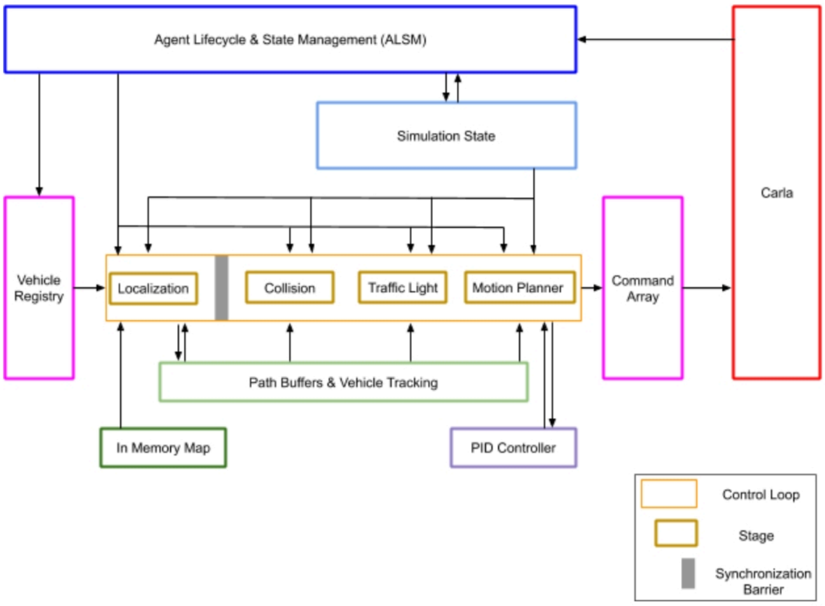


Figure 2.6: CARLA Traffic Manager Architecture [41]

.

Table 2.3: CARLA Actor Management Stages [41]

Stage	Description
Localization	Paths are created dynamically using a list of nearby waypoints collected from the In-Memory Map, a simplification of the simulation map as a grid of waypoints. Directions at junctions are chosen randomly.
Collisions	Bounding boxes are extended over each vehicle's path to identify and navigate potential collision hazards.
Traffic Lights	Potential hazards that affect each vehicle's path due to traffic light influence, stop signs, and junction priority are identified.
Motion Planner	Vehicle movement is computed based on the defined path. A PID controller determines how to reach the target waypoints.
Vehicle Lights	The vehicle lights switch on/off dynamically based on environmental factors (e.g. sunlight and the presence of fog or rain) and vehicle behavior (e.g. turning on direction indicators if the vehicle will turn left/right at the next junction, or turn on the stop lights if braking).

2.4.2 Pedestrian and Vehicle Behavior Simulation

CARLA offers realistic vehicle and pedestrian behaviors. Vehicles follow traffic rules, such as lane-keeping, traffic lights, and yielding right-of-way, making the simulation environment conducive to studying interactions between AVs and other road users. Pedestrians exhibit natural movement patterns, including walking, crossing streets, and reacting to vehicle movements.

To create custom behavior profiles for this research, a wrapper class was created around the different modifiable parameters for each actor type. Where the vehicle speeds, follow distance to proceeding vehicles, lane centering and lane changes are controlled to fit custom behavior types. Additionally pedestrian behaviors around crosswalks is also controlled to fit behavioral types, such as their walking speed, time spent before crossing and time looking at traffic. Table 2.4 summarizes the available parameters that can be modified to create behavioral driving profiles and Figure 2.7 illustrates how the behavior parameters are used to create custom behavioral profiles.

Table 2.4: CARLA Behavior Parameters [41]

Parameter	**Description**
Safety	• Set a minimum distance between stopped vehicles (for a single vehicle or for all vehicles). This will affect the minimum moving distance. • Set the desired speed as a percentage of the current speed limit (for a single vehicle or for all vehicles). • Reset traffic lights.
Collisions	• Enable/Disable collisions between a vehicle and a specific actor. • Define collision radius bounding box around vehicles, walkers, and traffic lights.
Lane Changes	• Force a lane change, ignoring possible collisions. • Enable/Disable lane changes for a vehicle.

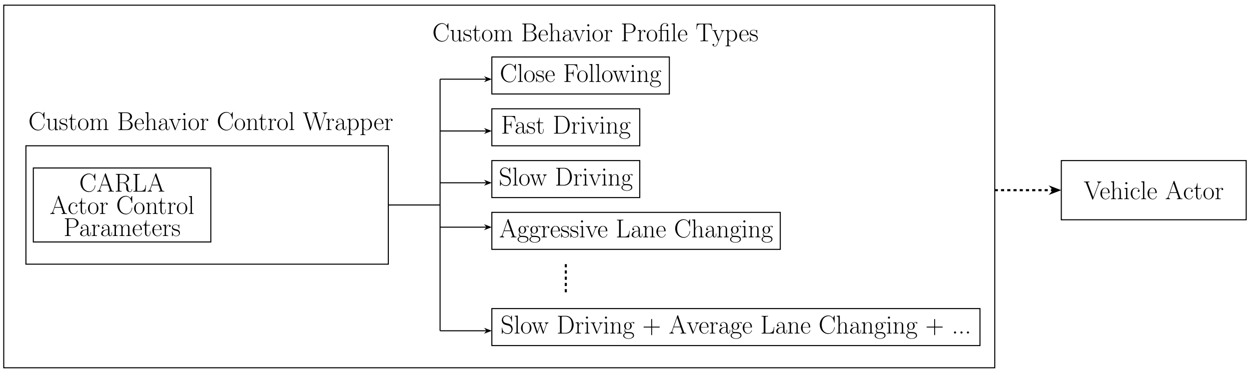


Figure 2.7: Custom Behavior Control Diagram

2.4.3 Collision Detection

To detect whether a vehicle has entered into a collision, CARLA using a bounding box system to check if the area is clear between each time-step. This bounding box applies to traffic lights, pedestrians, vehicles, road markings such as lane lines, and other road obstacles. For traffic lights, a bounding box is present for the area affected by the traffic light when it is red. This bounding box acts as a collision indicator. Figure 2.8 illustrates the bounding box created around vehicles. A collision is detected when the ego-vehicle's bounding box interferes with an obstacle's bounding box.

2.4.4 Driving Scenario Creation

To validate the proposed path planning framework, the creation of driving scenarios is necessary. From the traffic management module, it is possible to create a wrapper class over the tools it offers to extend its functionality. Unfortunately, CARLA's autopilot for driver and pedestrian actors possesses no intelligence, as vehicles and pedestrians are not goal oriented. They follow a dynamically produced trajectory and choose a path randomly when approaching a junction and their path is endless [41].

However, their initial spawning positions and their initial destinations (once reached they randomly generate a new one) can be set manually. This, in conjunction to a custom behavior profile that can be attributed to them allows us to have an adequate level of control.

2.5 Data Acquisition

To analyse and process a simulated scene, acquiring and saving data generated by the simulator is an important aspect of the development cycle. Data can be acquired in two ways:

Figure 2.8: Vehicle Bounding Boxes [41]

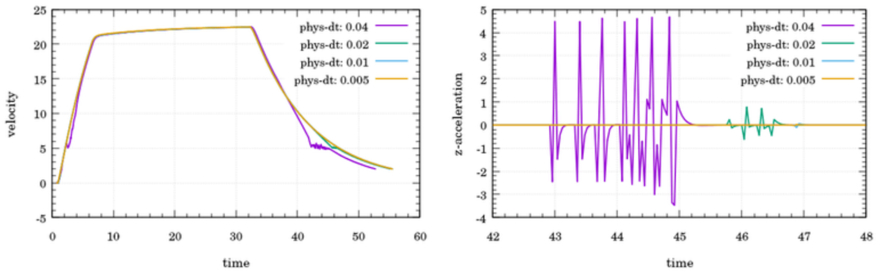
By directly querying the simulation software for actor positions, poses, states, etc., or by using the sensor suite available above and estimating the state of the world based on the sensor outputs. This work focuses on path planning, and thus generally directly queries the CARLA server for information about the surroundings of the ego-vehicle.

2.5.1 Synchronism and Determinism

CARLA offers two operation modes when communicating with the server, synchronous and asynchronous. While CARLA is a multithreaded task, the computations performed in each thread can be sent and received to the server in a synchronous or asynchronous manner. In synchronous mode, all computations are completed for each event (collisions, movements, inputs, physics calculations, etc) and their results are joined together before the next timestep. In asynchronous mode, calculations are performed and returned as they are completed, but their time to completion, order, or synchronicity is not guaranteed. While this can provide a smoother experience in certain contexts such as video games or web-browsing, it is at the direct expense of the determinism of the simulation. As such, this work only operates synchronously.

Physics Substepping

When computing the physics of the simulation step, achieving precision often requires computation within very low time steps. This poses a challenge when determining a suitable delta time for simulations, particularly those involving multiple computations per frame, such as sensor rendering. The limitation primarily arises from the physics simulation itself, prompting the application of substeps exclusively to physical computations. Using the appropriate physics substep is crucial to accurate simulations. However, an excessively small substep can needlessly increase computation time and prevent real-time simulation. Figure 2.9 shows the effect of substep size on convergence.



(a) Velocity Computation (b) Z-Acceleration Computation

Figure 2.9: Effect of Timestep on Physics Convergence [41]

2.5.2 Data Recording

Actors are updated on every frame according to the data contained in the recorded file. Actors in the current simulation that appear in the recording will be either moved or re-spawned to emulate it. Those that do not appear in the recording will continue their way as if nothing happened. The recorder file includes information regarding many different elements.

- **Actors** creation and destruction, bounding and trigger boxes.
- **Traffic Lights** state changes and time settings.
- **Vehicles** position, orientation, linear and angular velocity, light state, physics control.
- **Pedestrians** position orientation, linear and angular velocity.
- **Lights** Light states from buildings, streets, and vehicles.

From the recorded information generated by the simulation, it is possible to begin playback from any point in time within the recorded data. Additionally, the time factor of the simulation can be adjusted, where a time factor of $t_f = 1$ is real-time, $t_f < 1$ is slow motion, and $t_f > 1$ is fast motion. In addition to saving data to a file output, it is also possible to query the server for information in real time.

2.5.3 Performance Metrics and Evaluation

From the queryable information that can be obtained from the simulation server, and the collision sensors provided through the CARLA API, it is possible to evaluate and record the performance of the autonomous system. From this information, it is possible to determine the distance from other actors and elements, states of all the actors as well as any collisions. It is also possible to evaluate the respect of certain traffic laws, such as red lights and lane markings. Through these elements, it is possible to make comparisons between path planning and control protocols for the ego-vehicle and evaluate performance metrics related to safety and path efficiency.

2.5.4 Coordinate System

The CARLA Simulator operates using a standard 6-degree-of-freedom coordinate system (x, y, z, pitch, roll, yaw). This system provides a common reference for all actors and objects in the simulation, ensuring that their positions and orientations are accurately represented. The global origin of this coordinate system is located at the bottom left (South-West) of the map file. This means that a simple coordinate transformation matrix can be applied to any actor or object to convert them from the global frame of reference to an alternate reference, if needed. This standard coordinate system is a fundamental aspect of the CARLA Simulator, providing a consistent and reliable environment for autonomous vehicle system development.

Chapter 3

Autonomous Vehicle System

3.1 Bicycle Model Derivation

The dynamic equations of the bicycle model of a vehicle in the plane of motion are derived in this section. The kinematic model is first derived to express the vehicle motion in body-attached and ground-fixed coordinate frames. Figure 3.1 shows the bicycle model coordinate frames and Figure 3.2 shows the free body diagram of the forces applied to the vehicle. The bicycle model approximation assumes geometric symmetry of the vehicle, as well as a locally flat surface with no pitch or roll vertical dynamics. Because the path planner provides a 2-dimensional path, only horizontal dynamics were considered. For low to moderate speeds and reasonable maneuvers, this is an adequate simplification of the system. In the event where full vehicle dynamics need to be considered, a separate suspension controller can be implemented to control for vertical phenomena.

3.1.1 Coordinate Frames

In vehicle dynamics, a clear distinction between coordinate frames is crucial. The stationary global road coordinate frame $\mathcal{F}_G = \left\{O_{XY}, \mathbf{e}_G^X, \mathbf{e}_G^Y\right\}$ is fixed to the ground with vector $\mathbf{e}_G^X$ oriented in the initial driving direction. The vehicle body-attached local coordinate frame $\mathcal{F}_L = \{O_{xy}, \mathbf{e}_l^x, \mathbf{e}_l^y\}$ is located at the ego-vehicle's center of gravity $C.G.$ with vector $\mathbf{e}_l^x$ directed forward along the longitudinal axis of the vehicle. The global and local coordinates are utilized to express the vehicle's position, velocity, and acceleration vectors. In this context, the position vector $\mathbf{r}_{O_{xy}/O_{XY}}$ describes the location of the origin of the local vehicle frame O_{xy} relative to the global road frame O_{XY}. This vector is expressed as a linear combination of the global frame unit vectors $\mathbf{e}_G^X$ and $\mathbf{e}_G^Y$:

$$\mathbf{r}_{O_{xy}/O_{XY}} = X_{O_{xy}}\mathbf{e}_G^X + Y_{O_{xy}}\mathbf{e}_G^Y = \begin{bmatrix} X_l \\ Y_l \end{bmatrix} \mathbf{e}_G^{x/y}. \tag{3.1}$$

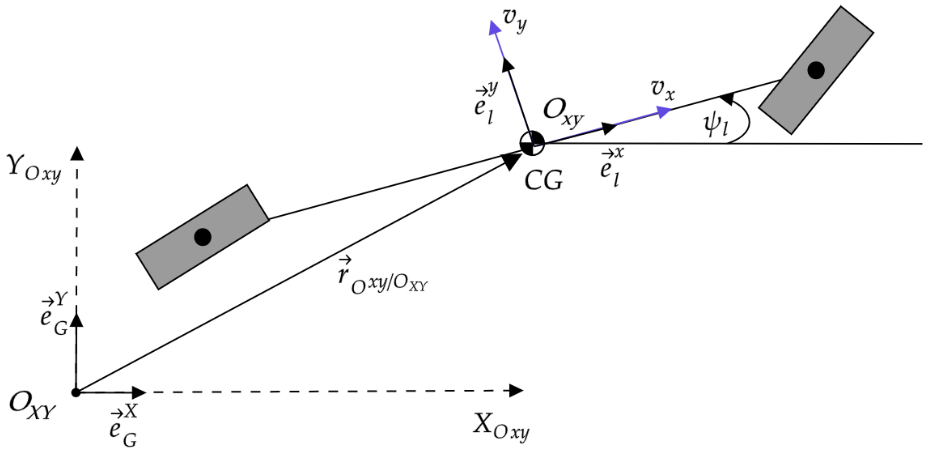


Figure 3.1: Bicycle Model Configuration With Global and Local Coordinate Frames

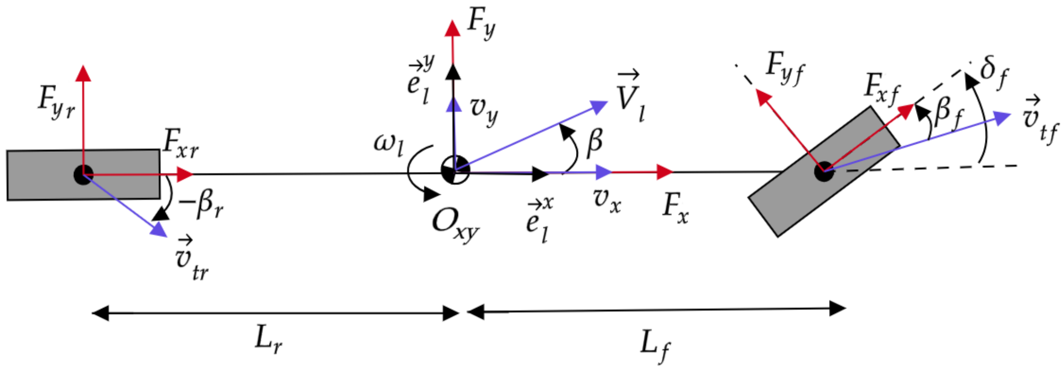


Figure 3.2: Bicycle Model Free-Body Diagram

where $\mathbf{e}_G^{x/y}$ represents the global frame unit vectors $\mathbf{e}_G^X$ and $\mathbf{e}_G^Y$ $\begin{bmatrix} X_l \\ Y_l \end{bmatrix}$ is a column vector representing the coordinates X_l and Y_l of the local frame origin O_{xy} in the global coordinate system, respectively. This position vector is crucial in vehicle dynamics as it defines the vehicle's position globally, allowing for navigation and trajectory planning calculations.

In this research, the rotation matrix R_{ψ_l} is used to transform position vectors from the global coordinate system to the local coordinate system. It rotates a vector around the $\mathbf{e}_G^Z$ axis, which is perpendicular to the ground plane defined by $\mathbf{e}_G^X$ and $\mathbf{e}_G^Y$ (*i.e.* $\mathbf{e}_G^Z = \mathbf{e}_G^X \times \mathbf{e}_G^Y$):

$$\mathbf{e}_G = R_{\psi_l}\mathbf{e}_l = \begin{bmatrix} \cos(\psi_l) & -\sin(\psi_l) \\ \sin(\psi_l) & \cos(\psi_l) \end{bmatrix} \mathbf{e}_l. \tag{3.2}$$

where ψ_l is the heading angle of the vehicle defined as the angle between $\mathbf{e}_G^X$ and $\mathbf{e}_l^x$. The angle ψ_l represents the heading angle of the vehicle, which is essential for understanding how the vehicle is oriented with respect to the road. This orientation affects maneuvering and stability. The velocity vector $\mathbf{V}_G$ of the ego-vehicle in the global frame is given by the derivative of the position vector $\mathbf{r}_{O_{xy}/O_{XY}}$:

$$\mathbf{V}_G = \dot{\mathbf{r}}_{O_{xy}/O_{XY}} = \dot{X}_l \mathbf{e}_G^X + \dot{Y}_l \mathbf{e}_G^Y = \begin{bmatrix} \dot{X}_l \\ \dot{Y}_l \end{bmatrix} \mathbf{e}_G. \tag{3.3}$$

where $\dot{X}_l$ and $\dot{Y}_l$ are the time derivatives of X_l and Y_l, respectively. Furthermore, the kinematic relations of the vehicle are derived considering the velocities at different points on the vehicle. The linear velocity $\mathbf{V}_l$ at the center of gravity $C.G.$ is related to the angular velocity $\boldsymbol{\omega} = \dot{\psi}_l$ as:

$$\mathbf{V}_l = \mathbf{V}_G + \boldsymbol{\omega} \times \mathbf{r}_{O_{xy}/O_{XY}}. \tag{3.4}$$

Expressing velocity in different frames (global and local) is fundamental for dynamic analysis and control algorithms in vehicle dynamics, such as trajectory tracking or evasive maneuvers. Acceleration is a key factor in vehicle dynamics, affecting everything from comfort to stability. It is particularly important in analyzing transient behaviors and designing control systems. The acceleration vector $\mathbf{A}_G$ of the ego-vehicle in the global frame is the derivative of the velocity vector $\mathbf{V}_G$:

$$\mathbf{A}_G = \dot{\mathbf{V}}_G = \ddot{X}_l \mathbf{e}_G^X + \ddot{Y}_l \mathbf{e}_G^Y = \begin{bmatrix} \ddot{X}_l \\ \ddot{Y}_l \end{bmatrix} \mathbf{e}_G. \tag{3.5}$$

where $\ddot{X}_l$ and $\ddot{Y}_l$ represent the second derivatives of the vehicle's position coordinates in the global frame. Additionally, $\mathbf{e}_G = \begin{bmatrix} \mathbf{e}_G^X & \mathbf{e}_G^Y \end{bmatrix}^T$ denotes the unit vectors in the global coordinate system.

3.1.2 Dynamics Equations

The dynamic equations for the ego-vehicle, considered a rigid body in the host vehicle frame, consist of the fundamental interactions between forces and motion. This model assumes no compliance in the vehicle chassis or suspension systems, together with a smooth road and geometric symmetry between the left and right sides of the vehicle. This simplification allows for a clearer analysis of the fundamental handling dynamics without the additional complexity of compliance effects. These equations are crucial for understanding vehicle dynamics and are given by:

$$m\ddot{X}_l = F_x - m\dot{Y}_l\dot{\psi}_l, \tag{3.6}$$

$$m\ddot{Y}_l = F_y + m\dot{X}_l\dot{\psi}_l, \tag{3.7}$$

$$I_z\ddot{\psi}_l = M_z, \tag{3.8}$$

where m represents the vehicle's mass, and I_z denotes the moment of inertia about the vertical axis, reflecting the vehicle's resistance to rotational acceleration. The forces F_x and F_y are the net forces acting along the local frame axes, and M_z is the net moment about the vertical axis. These equations are central to modeling the vehicle's linear and angular accelerations.

The role of tire forces is pivotal in vehicle dynamics, especially in modeling and simulation. Pneumatic tires primarily generate the forces necessary for both propulsion and maneuvering. The lateral forces the tires produce are particularly significant for vehicle handling stability and performance. They arise primarily due to slip angles, a function of the vehicle speed and the tire characteristics. According to prevalent tire models, the lateral forces are proportional to the magnitude of these slip angles.

In other words, the lateral and longitudinal forces at the front and rear tires, denoted as F_{yf}, F_{yr}, F_{xf}, and F_{xr}, respectively, depend on the slip angle, normal load, and friction coefficient of the tires. For the cornering forces, which contribute to the maneuverability of the vehicle and motion in the lateral direction, the tire force for the front and rear tires can be expressed as:

$$F_{yf} = 2C_{\beta f}\beta_f = \frac{mgV_l^2 L_r}{gR(L_f + L_r)} \tag{3.9}$$

$$F_{yr} = 2C_{\beta r}\beta_r = \frac{mgV_l^2 L_f}{gR(L_f + L_r)} \tag{3.10}$$

Where R is the steady-state turning radius of the vehicle, $C_{\beta f}$, $C_{\beta r}$ is the cornering stiffness and β_f and β_r is the slip angle of the front and rear tires respectively. L_f and L_r are the distances from the front and rear axle to the CG. Finally, V_l is the longitudinal velocity of the vehicle and g is the gravitational constant.

The general form of the Magic Tire formula stated by Pacejka is:

$$F_y = D\sin(C\arctan[Bx - E(Bx - \arctan(Bx))]) \tag{3.11}$$

where F_y is the lateral tire force resulting from a slip angle, and the parameters B, C, D, and E are the fitting constants for a given tire curve.

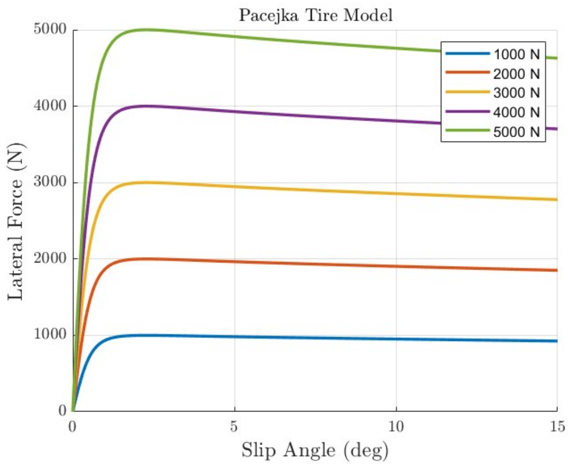


Figure 3.3: Pacejka Tire Model Lateral Force vs. Slip Angle [47]

Pacejka developed this tire model through extensive empirical testing of different tire constructions and conditions. The name *Magic Tire Formula* is used because there is no physical basis for this model, but instead relies on the empirical data to obtain the best fit between experimental data and the tire model. These coefficients are then used to generate equations showing how much force is generated for a given vertical load on the tire, camber angle, and slip angle [48]. The Pacejka tire model is widely used in professional vehicle dynamics simulations, and racing car games, as they are reasonably accurate, easy to program, and solve quickly [47]. An example curve for the Pacejka tire model is shown in Figure 3.3.

Longitudinal Forces

When using Simplified Julian's Elastic Band traction model, the longitudinal traction force F_x is generated by the area of the tire in full adhesion and the area experiencing slip. The resultant longitudinal force depends on which portion of the curve the amount of slip is situated and is shown in Figure 3.4.

For slip $i < i_c$:

$$F_x = C_i i \tag{3.12}$$

For slip $i = i_c$:

$$F_x = C_i i_c = \frac{\mu_{peak} mg}{2} \tag{3.13}$$

For slip $i > i_c$:

$$F_x = \mu_{peak} mg \left(1 - \frac{\mu_{peak} mg}{4C_i i}\right) \tag{3.14}$$

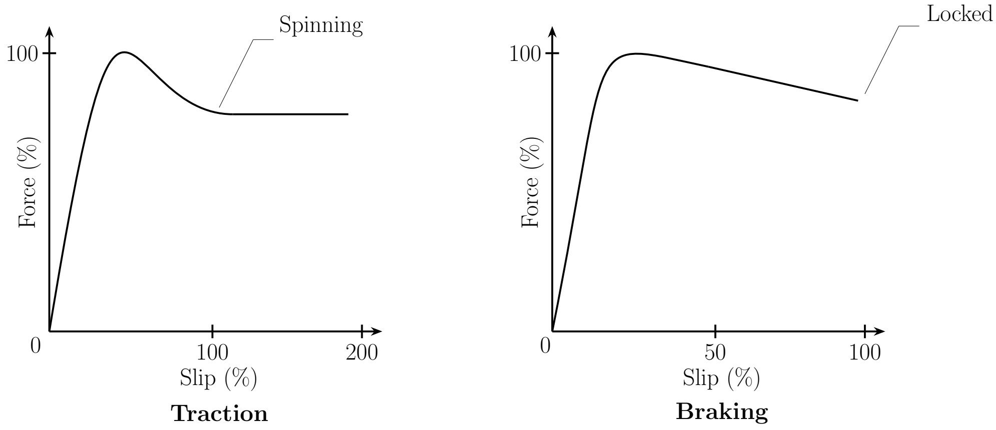


Figure 3.4: Longitudinal Tire Force-Slip Diagram

Where C_i is the longitudinal tire stiffness, i is the longitudinal slip %, i_c is the critical slip percent, determined by the linear region of the traction force-slip diagram shown in Figure 3.4. μ_{peak} is the peak coefficient of friction generated by the tire along the force-slip curve. Different tire models can also be used, such as the Brush Tire model.

The friction force ellipse describes the maximum cornering forces available when compound longitudinal and cornering forces are present as shown in Figure 3.5. The friction ellipse can be represented as:

$$\left(\frac{F_x}{F_{xmax}}\right)^2 + \left(\frac{F_y}{F_{ymax}}\right)^2 = 1 \tag{3.15}$$

Where F_{xmax}, F_{ymax} are the maximum available friction forces when purely longitudinal or lateral tire force is applied. The values of F_{xmax}, F_{ymax} are determined by the tire material and manufacturing properties.

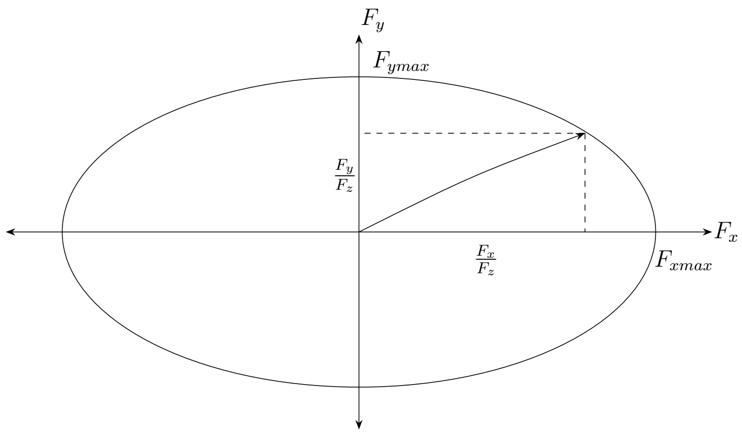


Figure 3.5: Tire Friction Ellipse

3.1.3 Slip Angle

The slip angle $\beta_{f,r}$ at the front and rear tires is defined as:

$$\beta_f = \arctan\left(\frac{\dot{Y}_l + L_f\dot{\psi}_l}{\dot{X}_l}\right) - \delta_f \tag{3.16}$$

$$\beta_r = \arctan\left(\frac{\dot{Y}_h - L_r\dot{\psi}_l}{\dot{X}_l}\right) \tag{3.17}$$

In the simplified bicycle model of the vehicle, the yaw rate $\left(\dot{\psi}_l\right)$ and the lateral velocity (v_y) of the CG are the two states of the system. The two states, together with the longitudinal position, form the three degrees of freedom for the vehicle, where the longitudinal velocity is denoted by (v_x). The model's input is the steering angle (δ). From the vehicle's free-body diagram, nonlinear equations are derived, relating the $C.G.$'s accelerations in longitudinal, lateral, and yaw directions $\left(\dot{v}_x, \dot{v}_y, \ddot{\psi}_l\right)$ to tire forces. These equations are:

$$\begin{aligned} m\dot{v}_x &= F_{xf}\cos(\delta_f) + F_{xf} - F_{yf}\sin(\delta_f) + m\dot{\psi}_l v_y, \\ m\dot{v}_y &= F_{xf}\sin(\delta_f) + F_{yf}\cos(\delta_f) + F_{yr} - m\dot{\psi}_l v_x \\ I_z\dot{\omega}_l &= L_f\left(F_{xf}\sin(\delta_f) + F_{yf}\cos(\delta_f)\right) - L_r F_{y_r} \end{aligned} \tag{3.18}$$

Tire forces play a crucial role, especially in high-speed highway driving with low road curvature and minimal steering angle. Under these conditions, small-angle approximations are used for the vehicle body slip angle (β), the front and rear tire slip angles (β_f and β_r), and the steering angle (δ_f). At high speeds, significant lateral forces from the tires counteract lateral acceleration. The tire slip angle, defined as the angle between the vehicle's heading and travel direction, increases the lateral forces. In the simplified linearized bicycle model, the lateral forces are linear functions of the slip angles:

$$\begin{aligned} F_{y_f} &= C_f \beta_f, \\ F_{y_r} &= C_r \beta_r. \end{aligned} \tag{3.19}$$

For small longitudinal slip ratios:

$$\begin{aligned} F_{y_f} &= 2C_f \left(\frac{\dot{Y}_l + L_f \dot{\psi}_l}{\dot{X}_l} \right) - \delta_f \\ F_{y_r} &= 2C_r \left(\frac{\dot{Y}_h - L_r \dot{\psi}_l}{\dot{X}_l} \right) \end{aligned} \tag{3.20}$$

$$\begin{aligned} \ddot{y} &= \frac{2C_{\beta f}\delta_f}{m} - \frac{2(C_{\beta f} - C_{\beta r})}{m v_x}\dot{y} - \left(v_x + \frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{m v_x} \right)\dot{\psi}, \\ \ddot{\psi} &= \frac{2L_f C_{\beta f}\delta_f}{I_z} - \frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{I_z v_x}\dot{y} - \frac{2(C_{\beta f}L_f^2 + C_{\beta r}L_r^2)}{I_z v_x}\dot{\psi} \end{aligned} \tag{3.21}$$

Presented in state-space form, we obtain:

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{2(C_{\beta f}-C_{\beta r})}{m v_x} & 0 & -v_x - \frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{m v_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{I_z v_x} & 0 & -\frac{2(C_{\beta f}L_f^2 + C_{\beta r}L_r^2)}{I_z v_x} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{2C_{\beta f}}{m} \\ 0 \\ \frac{2L_f C_{\beta f}}{I_z} \end{bmatrix} \delta \tag{3.22}$$

3.1.4 Nonholonomic Constraints

Kinematic Equations

The nonholonomy of a robot can be shown more simplistically through the kinematic equations of motion of a system, as the nonholonomy is usually determined by the motion constraints and limitations of the system. From Figure 3.1, we state the kinematic model as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ \frac{\tan(\delta)}{L} & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \tag{3.23}$$

Where the length L represents the wheelbase of the vehicle ($L_f + L_r$). We can further generalize the kinematic model by assuming that the steering control is applied as a steering angle, not a steering rate. The relation between the steering angle δ and the vehicle angular velocity ω can be written as $\omega = \frac{v}{L}\tan(\delta)$. The kinematic model in Equation 3.23 then simplifies to:

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = G(q)u = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \tag{3.24}$$

The implicit Pfaffian constraint on the velocities is written as:

$$A(q)\dot{q} = \begin{bmatrix} 0 & \sin\psi & -\cos\psi \end{bmatrix} \dot{q} = \dot{x}\sin\psi - \dot{y}\cos\psi = 0 \tag{3.25}$$

Because this velocity constraint can not be integrated into a constraint on the system's configuration, the system is thus nonholonomic. The real system is further constrained by the limitation imposed on the maximum steering angle δ caused by the vehicle's steering geometry does not allowing it to rotate to $\delta = \pm\,\frac{\pi}{2}$. The relation between the steering angle δ and the vehicle angular velocity ω is rewritten as: $\omega_{max} = \frac{v}{L}\tan\left(\delta_{max}\right)$. Because of this limitation, the vehicle can not rotate in place. The system has the following control transformation that takes the virtual controls V and ω and expresses them in terms of the actual controls.

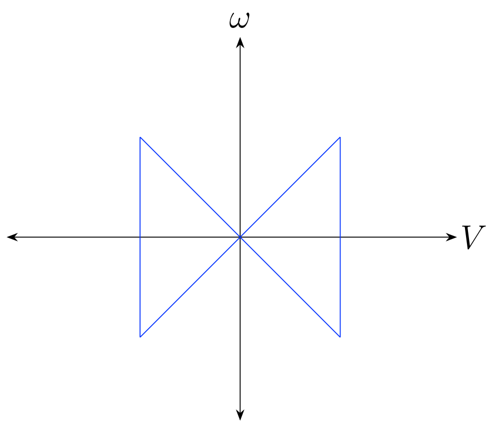


Figure 3.6: Car-Like Vehicle Control Bounds [49]

Figure 3.6 represents the control bounds of a car-like robot with Ackermann steering where ω and V represent the vehicle's yaw rate and longitudinal velocity, respectively. The bow-tie shape arises from the limit on the vehicle's turning radius, which affects its maximum yaw rate ω, and the vehicle's speed limit, which limits its forward-backward velocity V. In the case of a forward-only car, Figure 3.6 reduces to only the right-hand plane [49].

3.2 Vehicle Control

3.2.1 System Response

A steering controller is to be implemented to steer the vehicle along a path generated by the local path planner. For simplicity, only the lateral deviation from the generated path is used for path tracking. At low speeds and moderate steering angles, this showed to be an adequate means of controlling the ego-vehicle. From the bicycle model equations of motion developed in equation 3.22, the Laplace transformation of the state-space model takes the form:

$$\frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D \tag{3.26}$$

From the state-space model in Equation 3.22, where the observed state is y:

$$\begin{aligned} \dot{q}(t) &= \mathbf{A}q(t) + \mathbf{B}u(t) \\ y(t) &= \mathbf{C}q(t) + \mathbf{D}u(t) \end{aligned} \tag{3.27}$$

$$\begin{aligned} \dot{q}(t) &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{2(C_{\beta f}-C_{\beta r})}{mv_x} & 0 & -v_x - \frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{mv_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2(C_{\beta f}L_f - C_{\beta r}L_r)}{I_z v_x} & 0 & -\frac{2(C_{\beta f}L_f^2 + C_{\beta r}L_r^2)}{I_z v_x} \end{bmatrix} q(t) + \begin{bmatrix} 0 \\ \frac{2C_{\beta f}}{m} \\ 0 \\ \frac{2L_f C_{\beta f}}{I_z} \end{bmatrix} u(t), \\ y(t) &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} q(t) \end{aligned} \tag{3.28}$$

Using the Laplace transformation presented in Equation 3.26, the transfer function for each state is obtained by setting the feed-through matrix $D = [0]$ and setting the observability matrix $C = [1\ 0\ 0\ 0]$, $C = [0\ 1\ 0\ 0]$, $C = [0\ 0\ 1\ 0]$, $C = [0\ 0\ 0\ 1]$ to obtain the transfer functions for the lateral position, lateral velocity, yaw, and yaw rate respectively. For a constant velocity $v_x = 7m/s$, moment of inertia $I_z = 1500\,kg.m^2$, mass $m = 1750kg$, tire cornering stiffness $C_{\beta r} = C_{\beta f} = 100\,kN/rad$, front and rear lengths are $L_f = 1.8\,m$ and $L_r = 1.9m$:

$$G_1(s) = \frac{171.4s^2 + 20440s + 136.2}{s^4 + 163.1s^3 + 3882s^2} \tag{3.29}$$

$$G_2(s) = \frac{171.4s^2 + 20440s + 136.2}{s^3 + 163.1s^2 + 3882s} \tag{3.30}$$

$$G_3(s) = \frac{360s^2 - 324.1s - 2.177}{s^4 + 163.1s^3 + 3882s^2} \tag{3.31}$$

$$G_4(s) = \frac{360s^2 - 324.1s - 2.177}{s^3 + 163.1s^2 + 3882s} \tag{3.32}$$

3.2.2 Steering Control

Any automatic transmission road-going vehicle has three control inputs: the steering wheel, accelerator, and brake pedal. Two of these inputs can be simplified into one, where the accelerator and brake pedal can simply be combined as a velocity controller because it is reasonable to assume that road vehicles do not require the brake pedal and accelerator to be pressed simultaneously. While not the main focus of this work, two controllers were implemented to aid in simulating various traffic situations within the CARLA Simulator.

Geometric Look-Ahead Controller

Geometric steering controllers are based on a pure pursuit control method, with a look-ahead parameter. The output of the computed steering command is used to drive the ego-vehicle from its current position to the look-ahead point. A precondition to this controller is that the path generated by the path planner is designed reasonably and requests acceptable steering maneuvers. This also has the added benefit of reducing oscillations, improving stability, and as a consequence, improving passenger comfort.

The look-ahead distance acts as a proportional steering controller, where large look-ahead distances usually perform smoother inputs, at the cost of missing fine changes in the path. A look-ahead distance that is too large can result in significantly cut corners, which in the context of autonomous vehicles could mean a collision with the sidewalk or an obstacle [50]. The effect the look-ahead distance has on steering control is demonstrated in Figure 3.7.

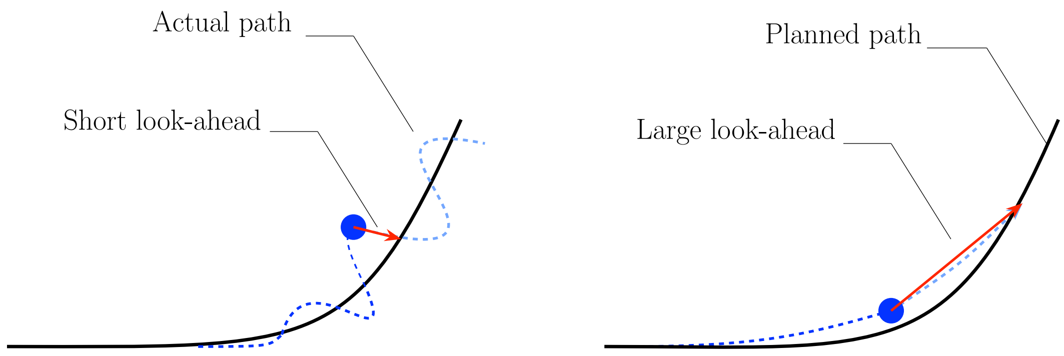


Figure 3.7: Effect of Look-Ahead Distance

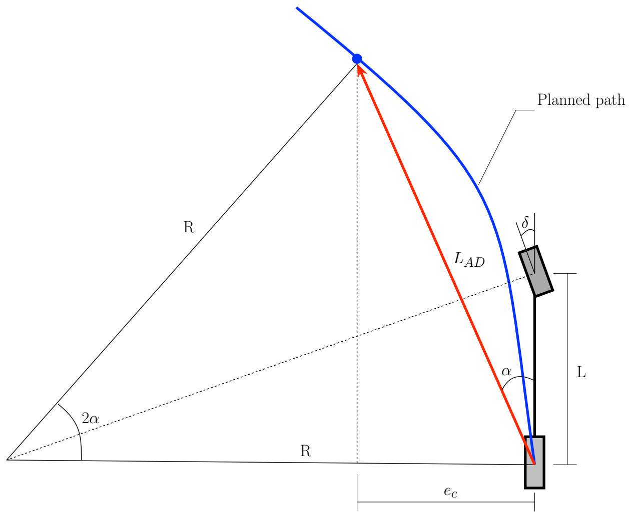


Figure 3.8: Look-Ahead Distance Geometry

From the kinematic equations of motion described in equation 3.23, the geometric relationship between the turning radius and steering angle δ is expressed as:

$$\tan(\delta) = \frac{L}{R} \tag{3.33}$$

According to the geometric relationship shown in Figure 3.8, the turning radius of the vehicle is:

$$R = \frac{2L\sin(\alpha(t))}{L_{AD}} \tag{3.34}$$

Where α is the angle formed between the rear wheel and the look-ahead point, and L_{AD} is the look-ahead distance. From equation 3.33 and 3.34 the steering angle applied to the vehicle for a look-ahead distance L_{AD} can be expressed as:

$$\delta(t) = \arctan\left(\frac{2L\sin(\alpha(t))}{L_{AD}}\right) \tag{3.35}$$

The lateral error between the ego vehicle's rear wheel and the look-ahead point is defined as $e_l = L_{AD}\sin(\alpha)$. By small angle approximation, we use equation 3.35 to express the lateral error as:

$$e_l = \frac{L_{AD}^2\delta(t)}{2L} \tag{3.36}$$

PID Controller

In an attempt to achieve more comfortable path tracking for the ego vehicle's occupants, a PID controller was implemented on the lateral position error. The introduction of a PID controller allows for more tuning possibilities over the path tracking of the ego-vehicle. This added control can give us more flexibility over various design parameters such as tracking overshoot, responsiveness, or other criteria such as perceived passenger comfort. The equation for a P-controller is given as:

$$\delta_p = K_p e_l \tag{3.37}$$

where K_p is the proportional gain constant. The I-controller reduces the steady-state error in lateral error tracking by integrating the lateral error from the planned path over time. The equation for the I-controller is:

$$\delta_i = K_i \int_0^t e_l(\tau)\ d\tau \tag{3.38}$$

where K_i is the integral gain constant. Finally, the derivative component dampens large velocities in the controller output by differentiating the lateral error from the planned path over time. The equation for the D-controller is:

$$\delta_d = K_d \frac{d}{dt}\left(e_l\right) \tag{3.39}$$

where K_d is the differential gain constant. However, it is important to note that it is sometimes difficult to incorporate D-controllers in systems that have noise present in the input, as the differential controller attempts to compensate for the large changes in the derivative of the input and must usually be coupled with a signal processing method to clean the signal. Because of this, this work focused on using a PID controller to compensate the system. Figure 3.9 shows the simplified block diagram of the steering controller, and y_{LA} is the lateral position of the planned path at the look-ahead point, and δ_p, δ_i, δ_d are proportional, integral and derivative controllers described by equations 3.37, 3.38 and 3.39 respectively.

To find an appropriate set of controller gains for the PID controller, the Ziegler-Nichols (ZN) method was used to obtain a baseline for performance. The ZN-method is a heuristic turning procedure for PID controllers that offer adequate but not necessarily optimal gain parameters for a controller [51]. The ZN method suggests first stimulating the system via an increase of the proportional gain until the ultimate gain K_u is reached by observing when the system output is stable with consistent oscillations. T_u is the oscillation period when the system is acted on by gain K_u. Based on the desired controller and performance, Ziegler et

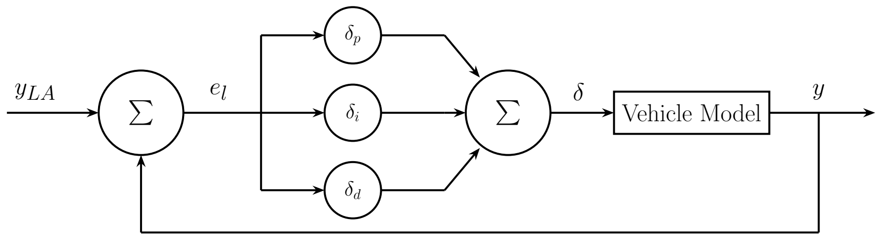


Figure 3.9: PID Steering Controller Block Diagram

Table 3.1: Ziegler-Nichols Method Parameters

Controller	K_p	K_i	K_d	T_i	T_d
P	$0.5K_u$	—	—	—	—
PI	$0.45K_u$	$0.54K_p/T_u$	—	$0.83T_u$	—
PD	$0.8K_u$	—	$0.1K_uT_u$	—	$0.125T_u$
Classic PID	$0.6K_u$	$1.2K_p/T_u$	$0.075K_uT_u$	$0.5T_u$	$0.125T_u$
Some Overshoot	$0.33K_u$	$0.66K_p/T_u$	$0.11K_uT_u$	$0.5T_u$	$0.33T_u$
No Overshoot	$0.2K_u$	$0.4K_p/T_u$	$0.066K_uT_u$	$0.5T_u$	$0.33T_u$

al. summarize the values for proportional, integral, and derivative gains K_p, K_i, K_d as seen in Table 3.1.

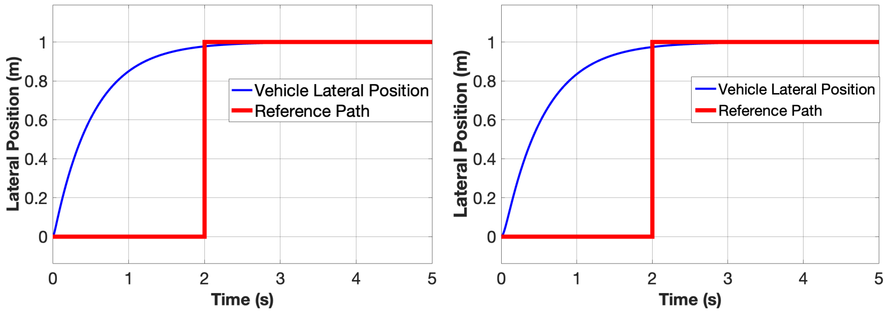
Using Table 3.1 to compute the parameters K_p, K_i, K_d, T_i and T_d, yields the following transfer function for compensator $u(s)$ from error $e(s)$

$$u(s) = K_p \left(\frac{T_d T_i s^2 + T_i s + 1}{t_i s} \right) e(s) \tag{3.40}$$

Controller Results

To ensure passenger comfort and reduce unease from steering overshoot, the performance requirement of a maximum of 0% overshoot was imposed for the steering controller. The best results were obtained by setting the look-ahead distance to 2 seconds of travel in front of the vehicle (i.e., $v_x = 3m/s \rightarrow L_{AD} = 6m$). Using the ZN method as a starting point, for a vehicle traveling 10km/h, the controller gains were $K_p = 1.25$, $K_i = 0.001$. At 25km/h, $K_p = 0.5$, $K_i = 0.001$. The response to a step input representing a reference path of 1 m is shown in Figure 3.10.

Remark: The controller response shown in Figure 3.10 assumes that the ego-vehicle travels at a constant velocity v_x of 10 km/h and the controller will be inadequate for velocities that are significantly different. Multiple velocity regions can be defined and each tuned using the same method as above.



(a) 10 km/h

(b) 25 km/h

Figure 3.10: Steering Controller Response to Step Input

A response to a step input represents the worst-case scenario for the steering controller, as a step input usually does not represent a reasonably planned path. The gain parameters were further refined within the CARLA simulation to balance the system's responsiveness with the road geometry within the simulator.

3.2.3 Velocity Control

The longitudinal velocity control in a conventional road vehicle is controlled by the accelerator and brake pedals. Because it is reasonable to assume that the accelerator and brake pedal do not need to be applied simultaneously, these two inputs can be grouped into a unified velocity controller. The sign of the output dictates whether the accelerator or brake pedal must be actuated.

Remark: The state-space vehicle dynamics model presented in equation 3.22 assumes constant longitudinal velocity v_x. The assumption of constant velocity renders the state-space equations a linear time-invariant (LTI) system. The system becomes a time-variant (LTV) if a velocity controller is introduced. To overcome this, regions of constant velocity are created, and the controllers are tuned according to the system's behavior for each region. This creates multiple LTI systems to describe the vehicle dynamics when velocity is not constant.

Reference Velocity

The reference velocity v_{ref} is the length of the planned path generated by the local planner relative to the length of the local planner boundary. Intuitively, it is reasonable to assume that if the local path planner can generate a continuous, acceptable path from the ego vehicle to the edge of the local planner boundary, there are no significant obstacles that would require a reduction of speed from the cruising velocity. Conversely, if the local path planner does not reach its boundary because of a local minima, an obstacle is hindering the navigation path. The distance to this local minima represents the proportional reduction of speed required, where a local minima at the ego-vehicle would require a full stop.

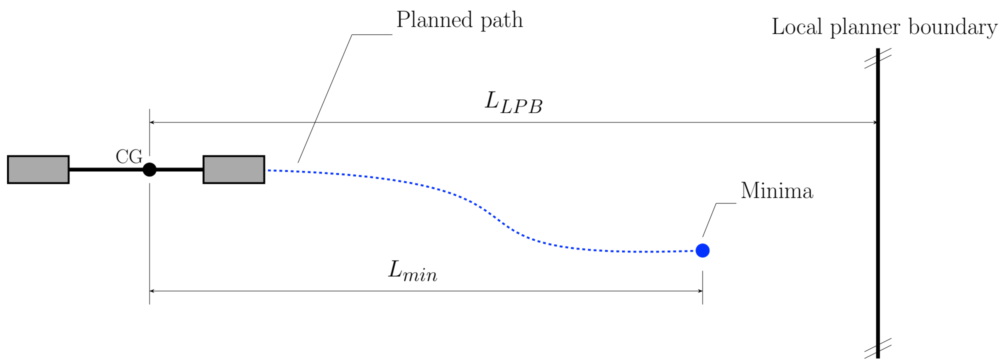


Figure 3.11: Free Distance Diagram

Figure 3.11 shows the relationship between the local planner boundary distance relative to the planned path length, where L_{LPB} is the local planner boundary distance from the ego vehicle's center of gravity and L_{min} is the planned path length in the longitudinal direction. Both L_{LPB} and L_{min} are in the ego-vehicle's longitudinal direction of travel.

The control input uses a value mapping formula to achieve a proportional control output relative to the length of the planned path. The standard mapping formula is given as:

$$m_{new} = \frac{(m - a_{min})(b_{max} - b_{min})}{(a_{max} - a_{min})} + b_{min} \tag{3.41}$$

where $a_{min} \leq m \leq a_{max}$ is to be mapped to new range $b_{min} \leq m_{new} \leq b_{max}$. Adapting the mapping formula to obtain a target velocity for the velocity controller, we have:

$$v_{ref} = \frac{(L_{min} - L_{stop})(v_{free} - v_{min})}{(L_{LPB} - L_{stop})} + v_{min} \tag{3.42}$$

where v_{ref} is the reference velocity, L_{stop} is the distance from the end of the planned path at which the ego-vehicle should be stationary, v_{free} is the free path speed (i.e. the path reaches the local planner boundary). v_{free} could represent the speed limit of the road or that of general traffic flow. v_{min} is the minimum velocity that the vehicle should drive at.

PID Controller

As with the steering controller, a PID controller was implemented for velocity control in order to have more flexibility in its performance. Equations 3.37, 3.38 and 3.39 described previously describe the P, I and D-controller respectively. The input to the controller uses the computed geometric velocity target as a reference value, and the vehicle's current position as feedback. The velocity error is stated as:

$$e_v = v_{ref} - v \tag{3.43}$$

where e_v is the velocity error, v_{ref} is the mapped reference velocity from equation 3.42 and v is the current ego-vehicle velocity. To achieve both braking and accelerator control, the signal is split based on the sign of the error term e_v.

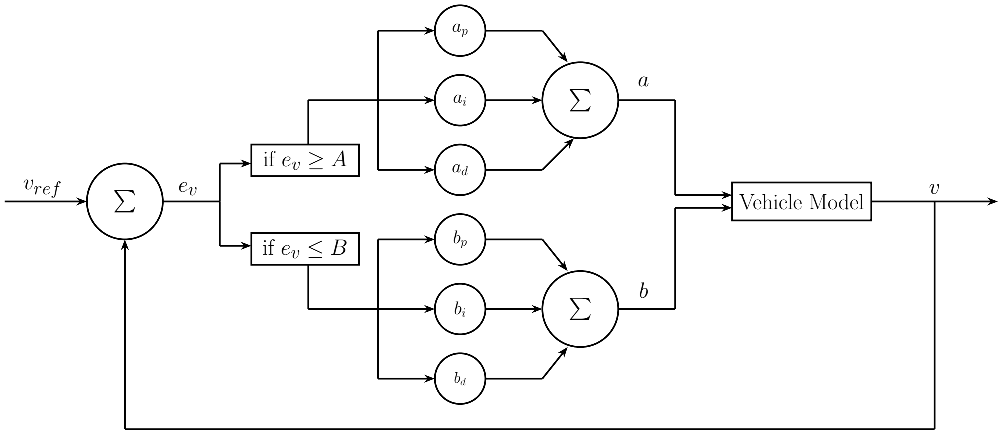


Figure 3.12: PID Switching Velocity Controller Block Diagram

Figure 3.12 shows the simplified block diagram for a switching velocity controller. Here, a is the accelerator output and b is the brake pedal output. a_p, a_i, a_d and b_p, b_i, b_d, are the accelerator and brake proportional, integral and derivative controllers respectively. Finally, the values of A and B are the velocity error dead-zone constants for the accelerator and brake pedal, which are set to prevent oscillatory switching between each controller.

The selection of controller gains K_p, K_i and K_d for the braking and accelerator switching controllers is done using the Ziegler-Nichols method which is summarized in Table 3.1 as a starting point.

Controller Performance

To prevent passenger unease, a control requirement of zero overshoot is necessary, and a maximum acceleration rate of $1m/s^2$ was imposed. For breaking, a zero percent overshoot was required to prevent jerking when coming to a stop.

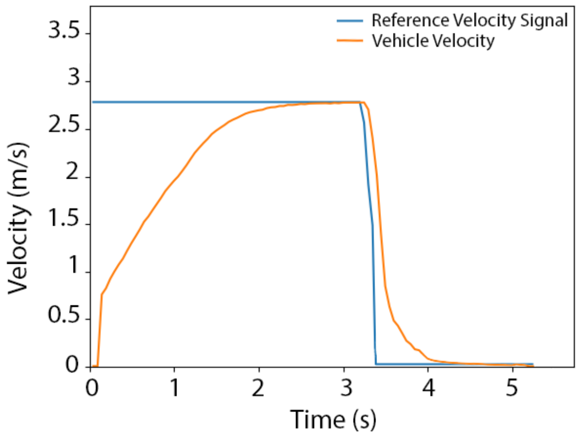


Figure 3.13: Acceleration and Braking Response

Experimentally, for a constant cruising speed of 10km/h, the acceleration controller proportional and integral gains were $a_p = 0.1$, $a_i = 0.05$ respectively. The braking controller gains were $b_p = 0.12$, $b_i = 0.05$. Furthermore, the dead-zone values for the controller switching were set at $A = B = 0.5km/h$. Figure 3.13 shows the acceleration an braking response from a 10km/h to 0km/h reference velocity signal.

Remark: Due to the non-linear force of air resistance acting on the vehicle, engine breaking and rolling resistance on the tires, multiple regions of velocity can be tuned to achieve adequate performance for many target speeds. This work mainly operates at lower velocities.

Chapter 4

Path Planning

As discussed previously, there is an important distinction in the methodologies used for high-level path planning (global or mission-level planning) and local planning. By employing a hybrid methodology, we can find total route optimality via an appropriate route planning algorithm, reduce the possibility of local minima, and allocate the appropriate computational resources for high-precision tasks.

4.1 High-Level Route Planning

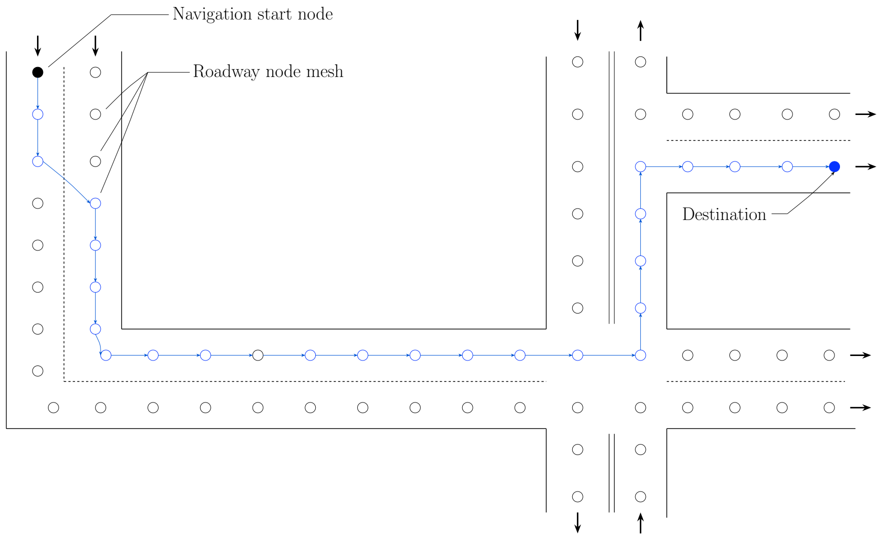


Figure 4.1: Roadway Planned Path

To generate a series of sub-goals, the `A*` search algorithm finds the shortest path connecting the origin and destination with a distance heuristic. To accomplish this, the entire roadway map is meshed into a node structure where each node is placed on the center line of all lanes, and the connection direction replicates the lane's traffic direction. Figure 4.1 demonstrates how a road network meshed on each lane's center line can be solved to obtain a path from the origin to the destination if one exists.

Remark: To ensure the feasibility and existence of a path from the initial ego-vehicle node to a requested destination node, a depth-first search (DFS) is performed before searching for the optimal path using `A*`.

4.1.1 `A*` Route Search

The `A*` search belongs to the *best-first* class of algorithms, which explores a graph by expanding the most promising node chosen according to a specified rule in a weighted-connected graph structure. In this case, the specified heuristic for optimality is only the Euclidean distance between nodes. However, it can be expanded to many parameters, such as time (with real-time traffic data), number of left-hand turns (to minimize intersection wait times), etc. The scope of this high-level path planner is simply to obtain an optimal distance path from origin to destination.

When Hart et al. [52] formulated the `A*` search, they stipulated that a graph G is defined to be a set $\{n_i\}$ of nodes and a set $\{e_{ij}\}$ of directed connections called arcs. If e_{pq} is an element of the set $\{e_{ij}\}$, it is said that there is an arc from node n_p to node n_q and that n_q is a successor node of n_p. These arcs have a cost, be it distance, time, etc. The cost of arc e_{ij} is defined as c_{ij}. Due to the directed nature of the graph, a connection between node n_i, n_j does not imply the reverse connection of node n_j, n_i exists. If such a connection did exist, their respective costs $c_{ij} \neq c_{ji}$. Only graphs G where $\delta > 0$ such that the cost of every arc of G $c_{ij} \geq \delta$ are considered.

The search problem is specified implicitly as a set of source nodes $S \subset \{n_i\}$ and a successor operator Γ on $\{n_i\}$ whose value for each n_i yields all the successors n_i of n_j and their associated arc costs c_{ij}. Applying the Γ operator to the source node and all its subsequent successors, as long as new nodes can be generated, results in an explicit graph specification. It is assumed that graph G is always given in implicit form. The sub-graph G_n from any node n in set $\{n_i\}$ is the graph defined implicitly by the single source node n and some operator Γ defined on $\{n_i\}$. It is said that each node G_n is accessible from n. A path from n_1 to n_k is an ordered set of nodes $(n_1, n_2, n_3, ...n_k)$ with each node n_{n+1} a successor of n_i. There exists a path from n_i to n_j if and only if n_j is accessible from n_i. The summation of all arc provides every path costs $\sum_{i=0}^{j} c_{i,i+1}$ and a path from n_i to n_j is said to optimal by having the smallest cost over all other sets of paths n_i to n_j. The cost of a path n_i to n_j

is represented by $h(n_i, n_j)$. When searching for the optimal path, we have the start node s and a nonempty set of potential goal nodes T in G. For any node n in graph G, an element $t \in T$ is said to be a preferred goal node of n if and only if the path's cost from n to t is less than any other path. The unique cost of an optimal path from n to t is described as: $h(n) = \min_{t \in T}(h(n,t))$.

The algorithm is performed by maintaining a tree of paths in memory from the origin to the destination. The algorithm has the evaluation function $f(n)$, which is defined as:

$$F(n) = g(n) + h(n) \tag{4.1}$$

where the function $g(n)$ is the calculated cost of the path from the start to node n, and $h(n)$ is the heuristic estimate of the cost from node n to the goal node. The `A*` algorithm is an iterative process. The algorithm maintains an open list of nodes yet to be evaluated, and at each iteration, it selects the node n, which has the lowest value evaluated at $f(n)$. At each iteration, the neighboring nodes in the directed graph are evaluated, which updates their respective $g(n)$ and $f(n)$ values and adds them to the open list when appropriate. For `A*` to guarantee an optimal solution, the heuristic function $h(n)$ must satisfy the following properties:

Admissibility: The heuristic function can not overestimate the true cost to reach the goal from node n such that: $h(n) \leq Cost_{Actual}(n, n_{goal})$

Consistency: For every node n and its successor $n+1$, the estimated cost from the origin to node n in addition to the cost of reaching node $n+1$ must be less or equal to the estimated cost from the origin to node $n+1$. Where: $h(n) \leq Cost_{Actual}(n, n+1) + h(n+1)$

Because the configuration space $\mathcal{C}$ is assumed to be locally flat between nodes, the heuristic Euclidean distance function $h(n) = \sqrt{(x_n - x_{goal})^2 + (y_n - y_{goal})^2}$. Procedurally, we can summarize the `A*` algorithm as follows [52]:

Step 1: Start at origin node s.

Step 2: Note s as open, calculate $f(s)$.

Step 3: Select neighbouring connected node n whose cost $f(n)$ is lowest. Tie-breakers are resolved arbitrarily unless node $n \in T$.

Step 4: If $n \in T$:

- **True:** Terminate search.
- **False:** Mark n closed. Apply successor operator Γ to node n. Calculate $f(n)$ for each successor and mark all successor nodes as open that are not already closed. Remark as open any closed node n_i which is a successor of node n if $f_{new}(n_i) < f_{old}(n_i)$. **Repeat step 2.**

Application

The application of the A* search to the high-level path planner involves using pre-processed graphs G, which provided a source node s as the ego-vehicle starting point and a destination node t. The pre-processed graph ensures that the directed graph G only has directed nodes that are legal driving maneuvers. For example, the directed connections ensure that the A* search can not perform illegal lane changes, U-turns, or drive the wrong way. Figure 4.2 illustrates the subset of graph G for a small subsection of the road network in which if a path exists between origin and destination, the optimal path can be found using A* search.

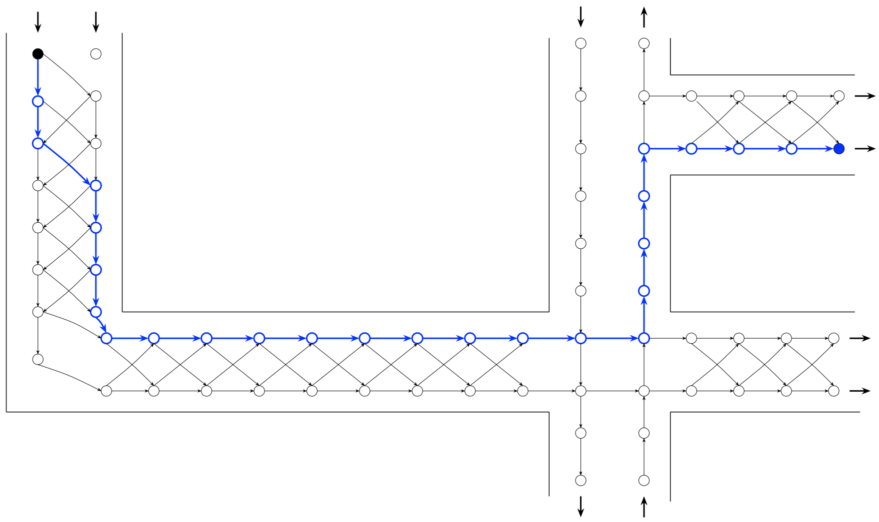


Figure 4.2: Roadway Node Mesh

By providing a directed graph G with a curated set of nodes placed on the lane center and providing only legal maneuvers through the arc direction, we can perform the A* search to find a valid, legal, optimal path from the source node to the destination. Furthermore, the A* search returns an ordered list N of nodes $(n_1, n_2, n_3, ...n_k)$ where $N \in G$. This ordered list N acts as a list of sub-goals that the local-level planner uses to navigate its local surroundings.

The high-level planner executes a node search once at the beginning of the navigation task and ideally never again, where the set of ordered nodes N would be sufficient for the local path planner to navigate the sequentially series of sub-goals. Road hazards or traffic blockages can trigger a re-calculation if real-time traffic information is known or the local path planner cannot execute the path after a set attempt time.

4.2 Local-Level Path Planning

4.2.1 Configuration Space

The configuration space $\mathcal{C}$ represents the set of all possible configurations that the autonomous vehicle can be in. While the autonomous vehicle can rotate in the pitch, roll, and yaw directions, the configuration space in this work is considered locally flat with negligible pitch and roll modes of motion. For a vehicle that can travel and rotate in 2-dimensional space, we have a configuration space $\mathcal{C} = \mathbb{R}^2$ and the special Euclidean orthogonal group of 2-D rotations $SE(2) = \mathbb{R}^2 \times SO(2)$ is applied. The $\mathcal{C}$ is represented with parameters (x, y, θ). The obstacle space $\mathcal{C}_{obs}$ represents the configuration space that the robot is not allowed to occupy. In this context, this represents the configuration space occupied by pedestrians, other vehicles, environmental obstacles, opposite lanes of traffic, etc. It should be noted that $\mathcal{C}_{obs}$ is not essentially limited to the physical space taken by the obstacles. This work provides a framework for incorporating the obstacle's dimensions, speed, direction, and social value (when applicable) into a complete risk distribution representing its obstacle configuration space.

Let there be some obstacle region O in the workspace W with $O \subset W$. The vehicle rigid body $A \subset W$ is defined. If $q \in \mathcal{C}$ represents the configuration of the rigid body A, the obstacle configuration space is defined as $C_{obs} = q \in \mathcal{C} | A(q) \cap \emptyset \neq 0$. This configuration space is the set of all possible configurations at which the vehicle intersects the obstacle region, O. Since sets $A(q)$ and O are closed sets, the obstacle region is a closed set in $\mathcal{C}$. The rest of the configurations make up the free space, denoted as $\mathcal{C}_{free} = \mathcal{C} \backslash \mathcal{C}_{obs}$. The free space $\mathcal{C}_{free}$ is an open set such that the rigid body can come arbitrarily close to the obstacle region O and still be in $\mathcal{C}_{free}$ [53] [54].

The free space $\mathcal{C}_{free}$ is the set of configurations without obstacles or possible collisions. Finally, the target space $\mathcal{C}_{target}$ represents the target destination or goal of the robot. In the context of path-planning and autonomous vehicles, usually, the final destination or goal of the vehicle is not directly observable but can be reached by a sequence of sub-targets, akin to turn-by-turn driving directions given to humans to reach a destination.

4.2.2 Coordinate System

CARLA Simulator's environment provides all positions of actors and objects in a global coordinate reference frame. To adequately create a potential field from the ego vehicle's perspective, a translation and rotation transformation matrix is needed to convert all information from the global reference frame to the local ego-centric frame of reference. The transformation matrix M matrix is represented as:

$$M = \begin{bmatrix} T & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} T^{-1} & x \\ 0 & 1 \end{bmatrix} \tag{4.2}$$

The translation matrix T and the rotation matrix R are expressed as follows:

$$R = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

$$T = \begin{bmatrix} v_1 & v_2 & v_p \end{bmatrix} \tag{4.4}$$

where θ is the angle of rotation between frames, v_p is the vector along the axis of rotation, and $[v_1, v_2]$ are the plane of rotation. The coordinate system used for the local planner in subsequent sections is the local ego-centric frame of reference unless otherwise stated.

4.2.3 Artificial Potential Fields

Potential functions offer a path-planning protocol in which a robot or autonomous vehicle follows the gradient such that $\nabla U(q) = 0$. In its simplest form, the ideal robot is represented by a particle that can rotate and translate freely. Due to the nonholonomy of the autonomous vehicle, constraints can be placed on the gradient descent that limits the angle at which the potential gradient can be followed. The potential force acting on a particle robot for a configuration space $\mathcal{C} = \mathbb{R}^2$:

$$F(q) = -\vec{\nabla} U(q) \tag{4.5}$$

The force F acting on the robot at a given configuration point q is equal to the gradient $\vec{\nabla} U(q) = [\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}]$.

The artificial potential field (APF) is created by adding together repulsive and attractive potentials. As mentioned earlier, attractive potentials indicate the local sub-target defined by the path planner, while repulsive potentials are generated by elements around the obstacle configuration space $\mathcal{C}_{obs}$. The total potential field is:

$$U(q) = U_{att}(q) + \sum U_{rep,i}(q) \tag{4.6}$$

The sub-goal of the overall navigation problem is represented by the potential function $U_{att}(q)$. Each obstacle has its own repulsive potential $U_{rep,i}(q)$ formed by a Gaussian distribution equation representing the spatial risk created by the obstacle. The equations for each potential should be differentiable $[\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}]$ for $\mathbb{R}^2$.

4.2.4 Gradient Descent

The general approach for obtaining a navigation path from a potential field using the gradient descent method involves performing the gradient operation on an iterative basis from an initial condition (the ego vehicle's initial configuration) until a minimum is found. This iterative search method is performed on a potential field $U(q)$ with the following equation [54]:

$$q_{k+1} = q_k - \alpha \nabla U(q_k) \tag{4.7}$$

where q_k is the current configuration of the ego-vehicle, $\nabla U(q_k)$ is the gradient of the potential field at the current configuration point, and scalar α is the iterative step size that the gradient descent is using.

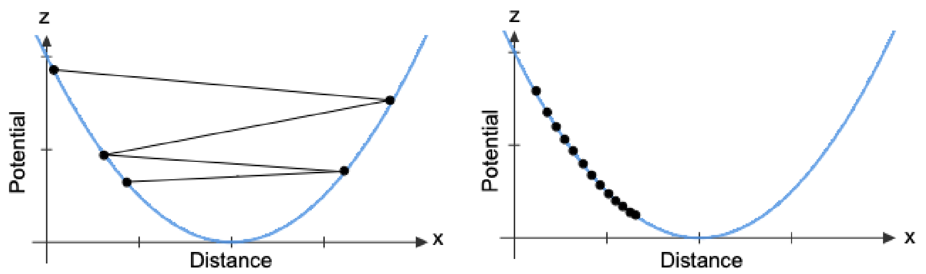


Figure 4.3: Effect of Gradient Step Size α

The tuning of the adequate step size α is highly important due to its effect on the convergence rate and accuracy of the gradient descent. A step size α that is too large may result in significant oscillations in the path or its inability to converge to a minimum. One that is too small can result in needless computational complexity and converging time, which is important in real-time applications. The gradient descent is iteratively performed until the end condition is met, *i.e.* a minimum is found such that: $\nabla U(q_k) = 0$. However, this punctual point on the graph is usually infinitesimally small and unattainable with a step size α. Thus, a more relaxed end condition is set:

$$\|\nabla U(q_k)\| \leq \epsilon, \quad \epsilon > 0 \tag{4.8}$$

where ϵ is a minimum gradient value at which the iterative process is stopped. Figure 4.3 demonstrates the effect of step size when performing the gradient descent of a function.

4.2.5 Adaptive Gradient Descent

The step size α was assumed to be a constant value. However, as discussed previously, sub-optimal values of α can lead to path oscillations or convergence issues. The concept of adaptive step size α is utilized [54].

$$\alpha_{best} = \arg\min_{\alpha}[U(q - \alpha \nabla U(q))] \tag{4.9}$$

Newton's method or other numerical methods can be used to determine the optimal step size. However, this can quickly become inefficient as the optimal step size must be calculated for each gradient step. A more effective method is finding an acceptable step size at each iteration rather than spending time minimizing the value. A backtracking line search is a common method for optimizing the step size method. Given a starting configuration q and a search direction ∇U, a line search determines a step size $\alpha \geq 0$ that reduces the objective function $U : \mathbb{R}^2 \to \mathbb{R}$ which is assumed to be continuously differentiable.

For a backtracking line search, an initial step size α is guessed from the previous step size by setting the initial guess larger than the previous step size α. This guess is then tested with the Armijo-Goldstein [55] condition:

$$U(q_{k+1}) \leq U(q_k + \beta \alpha_k \nabla U(q_k)^T \nabla U(q_k) \tag{4.10}$$

If the condition is satisfied, the step size is accepted and the gradient descent continues. If not, Armijo proposes reducing the step size by half and checking again. More formally, the step size α is reduced by a decay factor β, where $0 \leq \beta \leq 1$ and tested again for the Armijo-Goldstein condition.

Another approach is to incorporate the second-order information of the function by applying the Hessian of the objective function. Where the i^{th} row and j_{th} column of a Hessian are $(\mathbf{H}_U)_{i,j} = \frac{\partial^2 U}{\partial x_i \partial x_j}$. The general form Hessian matrix of a function U is given as:

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 U}{\partial x_1^2} & \frac{\partial^2 U}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 U}{\partial x_1 \partial x_n} \\ \frac{\partial^2 U}{\partial x_2 \partial x_1} & \frac{\partial^2 U}{\partial x_2^2} & \cdots & \frac{\partial^2 U}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 U}{\partial x_n \partial x_1} & \frac{\partial^2 U}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 U}{\partial x_n^2} \end{bmatrix} \tag{4.11}$$

The backtracking line search Armijo-Goldstein end condition for a step size α becomes:

$$U(q_{k+1}) \leq U(q_k) + \beta \alpha_k \nabla U(q_k)^T (\mathbf{H}_k)^{-1} \nabla U(q_k) \tag{4.12}$$

Where H_k and $\nabla U(q_k)$ are the Hessian and gradient at the current configuration point, respectively. The Hessian method considers the curvature of the potential function U to make a more informed step size adjustment and usually converges to a solution with fewer iterations. However, it adds significant computational complexity by requiring the Hessian and its inverse computation at each iteration [56].

4.2.6 Path Optimization

Because of the separation between the high- and local-level path planners, the local-level planner has already been tested for feasibility and optimality by the high-level path planner using the `A*` search algorithm. As such, any local minima remaining in the local planner arise from a set of potentials that *require* the vehicle to stop. These *desirable* local minima ensure the vehicle does not hit any obstacles when the road constraints and traffic configuration have created the local minima. This aspect allows for safe navigation within the confines of the road and traffic and differentiates this work from a simple obstacle avoidance model. Red lights, pedestrians, and vehicles can all create a minimum that prevents the ego-vehicle from moving forward, and the potential field of the lane prevents the ego-vehicle from illegally or unsafely trying to avoid the obstacle.

4.2.7 Nonholonomic Constraints

The straightforward gradient descent of a potential field generates a path that follows the direction of the steepest descent to a minimum. However, the path generated does not incorporate the nonholonomic constraints of the vehicle attempting to follow it. The gradient might require a motion or state that is not attainable. In the previous section, the Pfaffian constraint of a bicycle model vehicle was shown in Equation 3.25, and the control bounds of the model were shown in Figure 3.6.

In the context of path planning and providing a steering control mechanism for the autonomous vehicle, providing a safe and attainable path is important, without which the proposed path is pointless. To adapt the gradient descent method to the nonholonomic constraints of the vehicle, the angle at which the gradient descent can be performed is limited to a factor ϕ_{max} which is obtained from the maximum yaw rate that ego-vehicle can generate.

From the kinematic model presented in Equation 3.24, we have the yaw rate $\dot{\phi} = \left(\frac{\tan(\delta)}{L}\right) V$. Where δ is the steering angle of the front tires, and δ_{max} will denote the maximum steering angle that can be generated by the vehicle's steering geometry. The maximum angle ϕ_{max} at which the gradient descent can search between iterations is thus:

$$\phi_{max} = \frac{\alpha \tan(\delta_{max})}{L} \tag{4.13}$$

Where α is the gradient descent step size, and L is the length of the vehicle's wheelbase. This condition is implemented by limiting the maximum angle at which the gradient can be performed. A limiting function $\mathbf{C}$ is implemented, where $\mathbf{C}$ is defined as:

$$\mathbf{C}(x, \phi_{max}) = \begin{cases} \phi_{max}, & \text{if } x > \phi_{max} \\ x, & \text{if } -\phi_{max} \leq x \leq \phi_{max} \\ -\phi_{max} & \text{if } x < -\phi_{max} \end{cases} \tag{4.14}$$

Programmatically, the limiting function can also be written as $\max(\min(x, \phi_{max}), -\phi_{max})$. Applying the limiting function $\mathbf{C}$ to the iterative gradient descent of function U, we obtain:

$$q_{k+1} = q_k - \alpha \mathbf{C}(\nabla U(q_k), \phi_{max}) \tag{4.15}$$

Providing a gradient path that incorporates the maximum steering angle of the ego-vehicle helps generate accurate and relevant paths around obstacles and the roadway.

4.3 Potential Functions

4.3.1 Repulsive Potentials

The repulsive potentials represent all objects or elements that the autonomous vehicle must avoid or stay within. In the CARLA Driving Simulator, these represent other vehicles, pedestrians, traffic lights, and the lane in which the ego-vehicle must navigate. For simplicity, obstacles such as lamp posts, sidewalks, or other obstacles not pertaining to the navigable road surface are accounted for by the driving lane's potential field, which includes these hazards.

Pedestrians and Vehicles

All pedestrian and vehicle actors are represented by a 3-dimensional Gaussian distribution. The general form of the potential force equation used by social actors (pedestrians and vehicles) is:

$$U_{rep} = K_{rep} e^{-\left(\left(\frac{(x-i)^2}{2\sigma_x^2}\right)^{P_x} + \left(\frac{(y-j)^2}{2\sigma_y^2}\right)^{P_y}\right)} \tag{4.16}$$

where K_{rep} is the gain parameter for each repulsive equation, (i, j) are the centroids of the actor's location (m), which equates to the mean of the distribution in x and y directions, respectively. (σ_x, σ_x) represent the standard deviations of the potential force distribution (m). Parameters P_x and P_y are the order of the Gaussian distribution.

We can extend the definition of the standard deviations (σ_x, σ_x) to include not only the physical dimensions of the actor but also a dynamic growth factor that incorporates the reaction time of that actor to create a safe perimeter around them. To incorporate this parameter, we define the reaction distance as the relative incoming velocity V between the actor and the ego-vehicle (m/s), multiplied by the reaction time parameter t (s). We include the actor's physical length (m) and width (m) dimensions as L, W, and the actor's social value Ω multiplied by a scaling factor α.

$$\sigma_x = (L\kappa_x + t|V|)(1 + \alpha\Omega) \tag{4.17}$$

$$\sigma_y = (W\kappa_y + t|V|)(1 + \alpha\Omega) \tag{4.18}$$

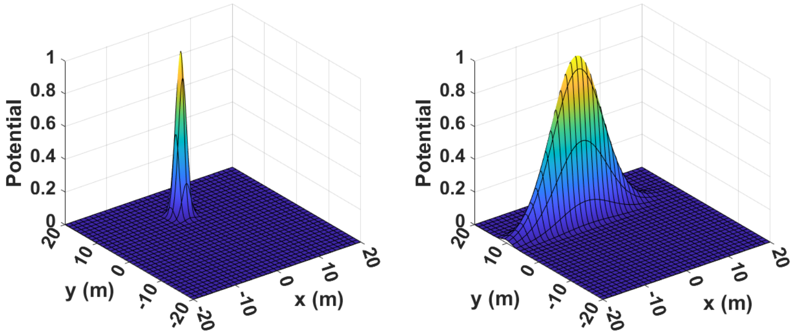
We also introduce scaling factors κ_x and κ_y to scale the growth due to relative incoming velocity. For example, if the ego-vehicle and a vehicle traveling in the opposite direction on the highway have a relative closing velocity of 200km/h (55.5m/s), we would expect a large increase in that actor potential longitudinally (in its direction of travel), but not as much laterally. In Equations (4.17) and (4.18), we apply the actor's social value $\Omega = [-1, 1]$ as a multiplicative element to its potential force, without removing potential caused by its physical dimensions and reaction distance if its social value is equal to 0 (individualistic). The scaling factor α is constrained between [0,1].

Extending Equation (4.16) we obtain:

$$U_{rep} = K_{rep} e^{-\left(\left(\frac{(x-i)^2}{2[(L\kappa_x+t|V|)(1+\alpha\Omega)]^2}\right)^{P_x} + \left(\frac{(y-j)^2}{2[(W\kappa_y+t|V|)(1+\alpha\Omega)]^2}\right)^{P_y}\right)} \tag{4.19}$$

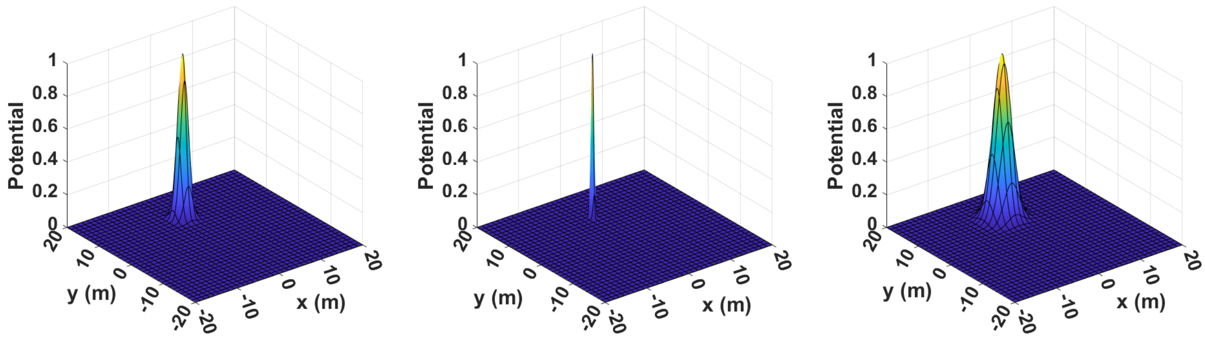
Pedestrians

To demonstrate the potential field effect created by pedestrians and their various input variables, we plot the discretized versions of their potential equations in Figures 4.4 and 4.5. Because pedestrians can move in any direction with little notice, we propose a wider risk distribution and set their Gaussian order in Equation (4.19) to one. In the following example, we demonstrate the effect of relative velocity and social value on their APF. We set the SVO scaling factor $\alpha = 1$, longitudinal and horizontal reaction time scaling factors $\kappa_x = 1$, $\kappa_y = 0.1$ with a reaction time $t = 2s$, location centroid $(i, j) = (0, 10)$, pedestrian dimensions $L, W = 1m$ and the Gaussian order $P_x, P_y = 1$.



(a) $V = 0m/s, \Omega = 0$ (b) $V = 5m/s, \Omega = 0$

Figure 4.4: Effect of Relative Velocity on Pedestrian APF



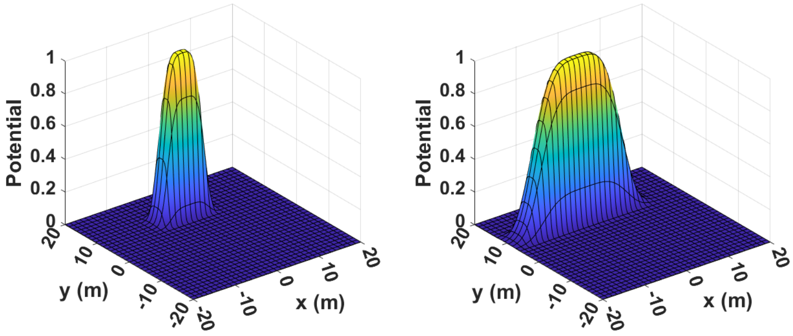
(a) $\Omega = 0$ (individualist) (b) $\Omega = -0.75$ (Cooperative) (c) $\Omega = 0.75$ (Competitive)

Figure 4.5: Effect of Social Value on Pedestrian APF

Vehicles

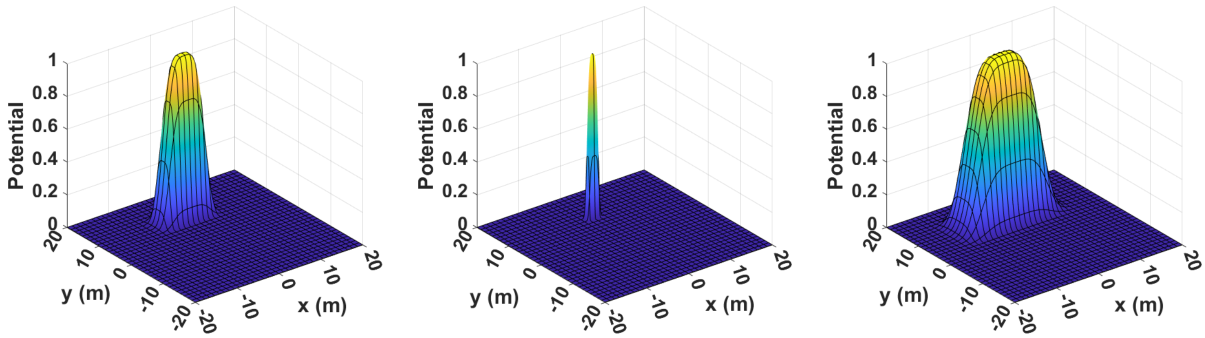
A Gaussian order of 2 is used to better represent vehicles' geometry. This creates a more rectangular distribution that better encapsulates a vehicle as an obstacle. We also propose that vehicles are less sensitive to lateral reaction time (because of their nonholonomic nature) than pedestrians, who can change direction at any time and with little notice.

The discretized versions of their potential equations are plotted in Figures 4.6 and 4.7 to demonstrate the potential field effect created by vehicles and their various input variables. A Gaussian order $P_x, P_y = 2$ is set, the SVO scaling factor $\alpha = 1$, longitudinal and horizontal reaction time scaling factors $\kappa_x = 1$, $\kappa_y = 0.05$ with a reaction time $t = 2s$, location centroid $(i, j) = (0, 10)$, and vehicle dimensions $L = 4m, W = 2m$.



(a) $V = 0m/s, \Omega = 0$ (b) $V = 5m/s, \Omega = 0$

Figure 4.6: Effect of Relative Velocity on Vehicle APF



(a) $\Omega = 0$ (individualist) (b) $\Omega = -0.75$ (Cooperative) (c) $\Omega = 0.75$ (Egotist)

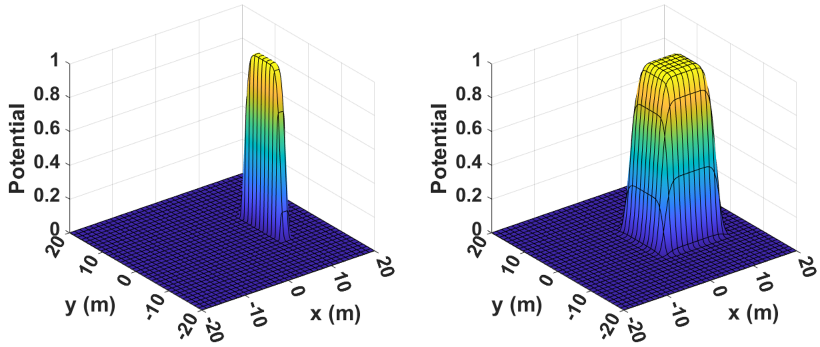
Figure 4.7: Effect of Social Value on Vehicle APF

Traffic Lights

To navigate intersections and come to a stop when a traffic signal turns red, a traffic light APF is implemented, which is conditional to its light state (red, yellow, green). As with previous equations, the traffic light APF uses a super-Gaussian distribution, which the general form is presented in Equation (4.16). However, traffic lights do not possess a social component, so the standard deviations of the equation are modified accordingly. Additionally, the σ_x and σ_y parameters presented previously in Equations (4.17) and (4.18) become:

$$\sigma_x = L\kappa_x + t|V| \tag{4.20}$$

$$\sigma_y = \frac{1}{2}W_{lane} \tag{4.21}$$



(a) $V = 0m/s$ (b) $V = 5m/s$

Figure 4.8: Effect of Relative Velocity on Traffic Light APF

Extending Equation (4.16), we can obtain the traffic light artificial potential:

$$U_{rep} = K_{rep} e^{-\left(\left(\frac{(x-i)^2}{2(L\kappa_x + t|V|)^2} \right)^{P_x} + \left(\frac{(y-j)^2}{\frac{1}{2} W_{lane}^2} \right)^{P_y} \right)} \tag{4.22}$$

where the Gaussian order $P_x, P_y = 4$. It is noted that the distribution does not expand horizontally from the reaction time of incoming velocity (see Equation (4.21)). In conjunction with a Gaussian order of 4, this prevents the traffic light APF from *bleeding* into adjacent lanes and serves as a wall to the navigation path the ego-vehicle takes during red lights. However, the potential field still includes expanding the distribution in the ego-vehicle's travel direction. This is maintained to slow the vehicle down appropriately the faster it travels toward the intersection.

The discretized versions of their potential equations are plotted in Figure 4.8 to demonstrate the potential field effect created by traffic lights and their various input variables. A Gaussian order of $P_x, P_y = 4$ is set, longitudinal reaction time scaling factor $\kappa_x = 1$, with a reaction time $t = 2s$, location centroid $(i, j) = (10, 0)$ (m), and traffic light dimensions $L = 1m$, $W_{lane} = 4m$.

Lane of Navigation

A repulsion force is created around its left and right bounds to keep the vehicle within the confines of the lane it needs to navigate in. A 2^{nd}-order equation creates the potential force as:

$$U_{rep} = K_{rep} \left(\frac{(y-j)^2}{W_{lane}} \right) \tag{4.23}$$

where U_{rep} is the potential at any point, K_{rep} is the repulsive force gain factor, j is the lateral distance of the lane center to the ego-vehicle (m), and W_{lane} is the width of the ego-vehicle's lane. As shown, the lane's potential force is not dependent on x and solely from the lateral distance to the ego-vehicle. This will be further explained in the following subsection. The discretized version of its potential equation is plotted in Figure 4.9 with lane width $W_{lane} = 4m$ to demonstrate the potential field effect created by the basic navigation lane.

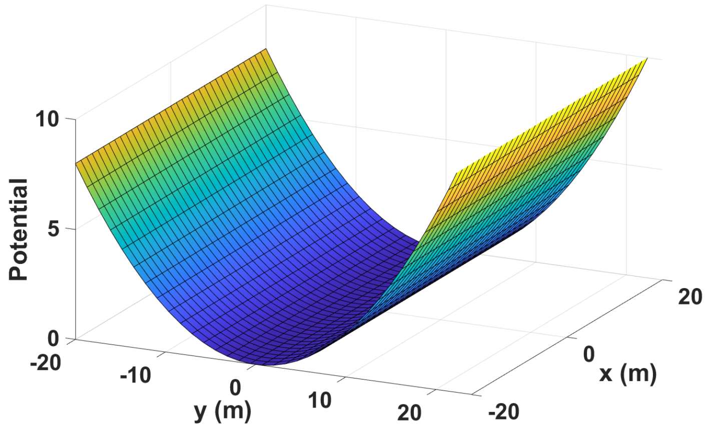


Figure 4.9: Straight Lane of Navigation APF

Lane Curvature

It is immediately evident that the basic lane of navigation presented is not a complete solution to describe the repulsive potential created by the lane. To correctly plan for the road geometry ahead, the lane potential field must also represent the road's curvature. A polynomial regression is performed on the road geometry from a series of sampled points of the lane center. The discussion of these sampling points from the OpenDRIVE road geometry file is the topic of the high-level planner in Chapter 4.

To summarize those key points, the high-level path planner performs an `A*` search of all roadway nodes, where the roadway is a directed graph G. All directed nodes are located on all lane centers, and arcs only directionally connect legally reachable nodes (i.e., no illegal lane changes, U-turns, or wrong-way). The `A*` search returns an ordered list N of nodes $(n_1, n_2, n_3, ...n_k)$ where $N \in G$. This ordered list N acts as a list of sub-goals that the local-level planner uses to navigate its local surroundings.

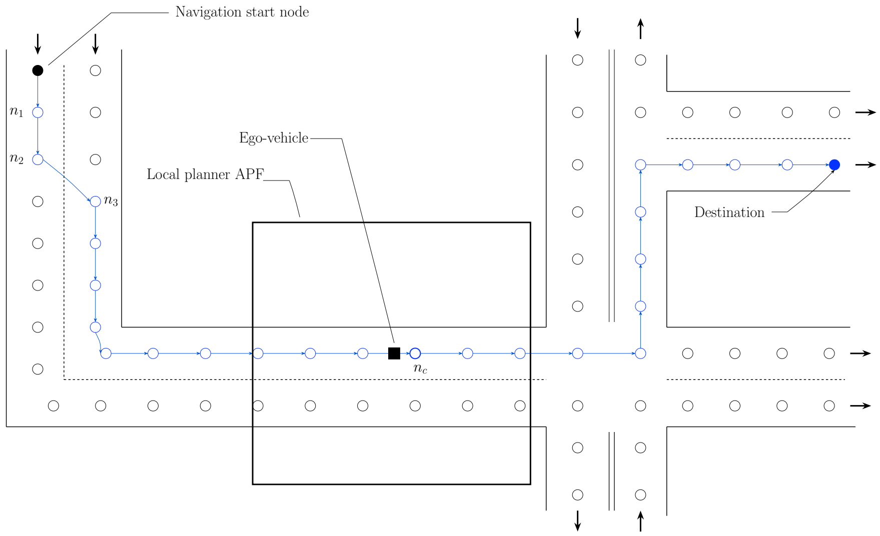


Figure 4.10: Local Planner Node Subset

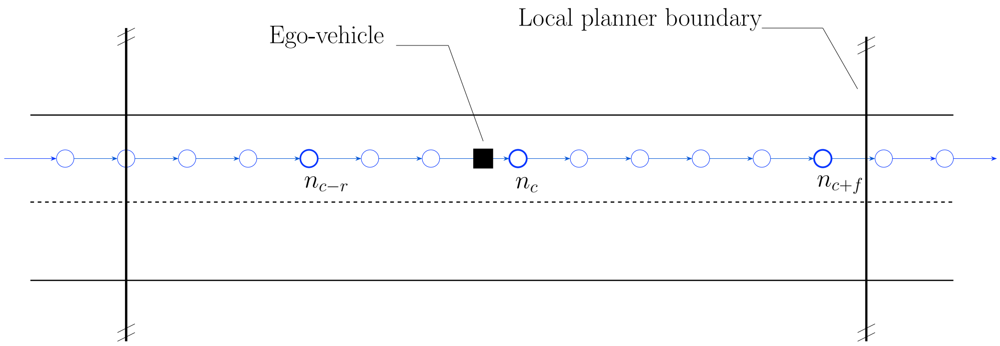


Figure 4.11: Local Planner Node Subset $N_L = \{n_{c-r}, n_{c+f}\}$

From the ordered list N of sub-goals, the local planner determines the closest node n_c to the ego-vehicle by Euclidean distance. Then, the local set of sub-goals N_L is extracted from the ordered list N such that $N_L \in N$ and $N_L = \{n_{c-r}, n_{c+f}\}$ where r and f are the number of nodes behind the ego-vehicle and in front of the ego-vehicle in the ordered list N. Experimentation showed that the best coefficients for r and f were obtained at $r = 8$ and $f = 20$. N_L acts as a sliding window on the ordered node set N determined by the high-level planner as shown in Figure 4.10 and is shown in isolation in Figure 4.11.

By replacing mean j in Equation (4.23) by a polynomially regressed equation describing the position of the set of sub-goals N_L, it causes the mean of the distribution to follow the curvature of the road ahead. The subset N_L is expressed in local ego-centric coordinates by applying the transform matrix M_L^G and the polynomial regression formula applied to the transformed nodes:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + ... + \beta_n x^n + \epsilon \tag{4.24}$$

The least squares method is used to minimize the residual vertical error between sampled road geometry points and the polynomial regression as follows:

$$R^2 = \sum [y_i - f(x_i, \beta_0, \beta_1, \beta_2, \beta_3 + ... + \beta_n)]^2 \tag{4.25}$$

Experimentation showed that the best balance between over-fitting and accuracy for the polynomial lane equation was obtained by performing a 4^{th}-order regression:

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 \tag{4.26}$$

Combining Equation (4.26) to Equation (4.23) we obtain:

$$U_{rep} = K_{rep} \left(\frac{(y - f(x))^2}{W_{lane}} \right) \tag{4.27}$$

To demonstrate the potential field effect created by the complete navigation lane given by Equation (4.27), we plot the discretized version of its potential equation in Figure 4.12. The lane width is set to $W_{lane} = 4m$, and a 4^{th}-order regression from the local sampled road section is used:

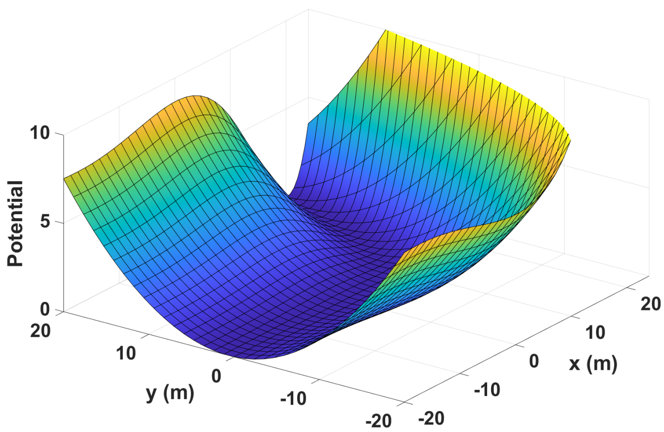


Figure 4.12: Lane of Navigation APF

4.3.2 Attractive Potential

The purpose of the attractive potential is to guide the ego-vehicle toward the navigation goal. Within the context of a local planner, the attractive potential location usually implies a sub-goal. Navigating a sequence of correct sub-goals will amount to completing the global navigation task. As mentioned earlier, the global goal is usually out of sight, and a large configuration space would increase the probability of encountering undesirable local minima.

The high-level path planner shows that the sub-goal locations (navigation points) are determined by the `A*` search algorithm and packaged as a list of sequential locations.

Parabolic Attraction to Sub-Goal

The most common method of creating an attractive potential field is to use a parabolic or paraboloid equation centered at the objective or local sub-goal. The equation of the paraboloid takes the form:

$$U_{att} = \frac{(x-i)^2}{a^2} + \frac{(y-j)^2}{b^2} \tag{4.28}$$

Where i, j are the centroid of the parabola and a, b are scaling factors in the x, y directions, respectively. The sub-goal paraboloid attractive potential is plotted in Figure 4.13.

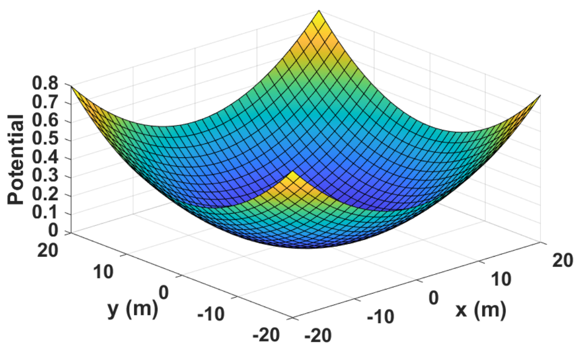


Figure 4.13: Paraboloid Sub-Goal Attractive Potential

From the ordered list N of sub-goals obtained from the global planner, the local goal-node subset $N_L \in N$ contains all the of the local planner nodes used for the lane regression as shown in Figure 4.11. $N_L = \{n_{c-r}, n_{c+f}\}$ where r and f are the number of nodes behind the ego-vehicle and in front of the ego-vehicle in the ordered list N as explained in the previous section. The $(c+f)^{th}$ node in list N_L is used as the centroid for the attractive paraboloid. As the ego-vehicle moves forward, the new $(c+f)^{th}$ node is used as the attractive paraboloid center.

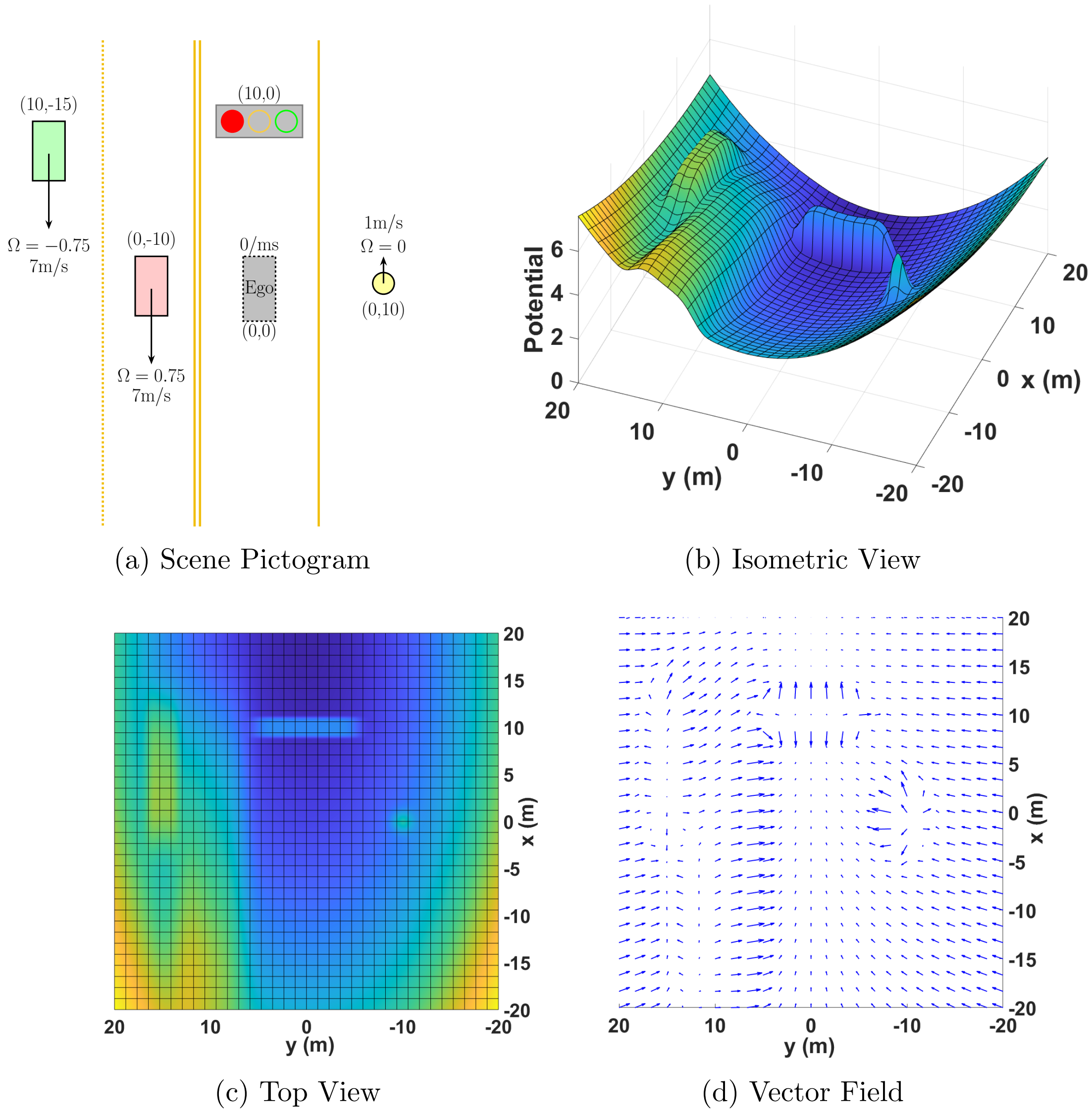


Figure 4.14: Total Potential Field Example Scene

4.3.3 Total Potential

The total potential is created by adding all individual repulsive forces created by actors and the environment and the attractive potential of the local sub-goal. The total repulsive potential is expressed as:

$$
\begin{aligned}
U_{tot} &= U_{att} + U_{rep} \\
U_{tot} &= U_{att} + U_{rep,_{lane}} + \sum U_{rep,_{vehicle,i}} + \sum U_{rep,_{pedestrian,i}} + \sum U_{rep,_{light,i}}
\end{aligned}
\tag{4.29}
$$

where i represents the i^{th} actor of that type and its state in the local navigation scene. To demonstrate the total potential of a navigation scene, we plot the following discretized version of the repulsive potential field. A fictitious road scene is created where the ego-vehicle navigates along a northbound straight lane and is stopped at a red traffic light. An individualistic ($\Omega = 0$) pedestrian is walking at 1m/s northbound on the sidewalk parallel to the ego-vehicle, and two incoming vehicles are driving in the southbound lanes. The leading vehicle has an egoistic social value ($\Omega = 0.75$), and the following vehicle has a cooperative social value ($\Omega = -0.75$). Both are traveling southbound at 7m/s. The total potential for this scene is shown in Figure 4.14.

Chapter 5

Social Value Orientation Modeling

This work limits the use of the social value orientation ring to the right-hand plane of possible behavioral labels. The difficulty of using the SVO circle to determine the sociality of a road user comes from the labeling process, where the non-verbal communication of individual actors is used to estimate their behavioral profile. All behaviors exhibit some level of egotism and cooperativeness, which is particularly well suited to fuzzy logic. Membership functions for each input and output variable allow for *degrees of truth* as a means to represent the uncertainty of a system as opposed to a more rigid or Boolean labeling process. Each actor has a different set of input variables to determine their SVO which observes behaviors that are unique to that actor type. The use of a fuzzy estimator allows for a human observable system that can be tuned to various demographic or geographic tendencies. Additionally, it presents the advantage of not needing large datasets to learn behaviors, at the expense of its ability to notice behavioral sequences. The fuzzy sets were created by identifying common actions observed during different driving scenarios. From these identified actions, their behavioral spectrum was subjectively quantified from altruistic to sadistic using the right-hand plane of the SVO circle shown in Figure 5.1.

The original *type-1* fuzzy set proposal by Zadeh was criticized for having membership functions with no uncertainty associated with it. Introducing *type-2* fuzzy sets allows for uncertainty to be attributed to the membership functions defining the measured behaviors, which is of particular importance in the context of commonsense approaches to fuzzy estimator tuning, as well as significant variation in demographic or geographic tendencies in the measurement of behavioral cues.

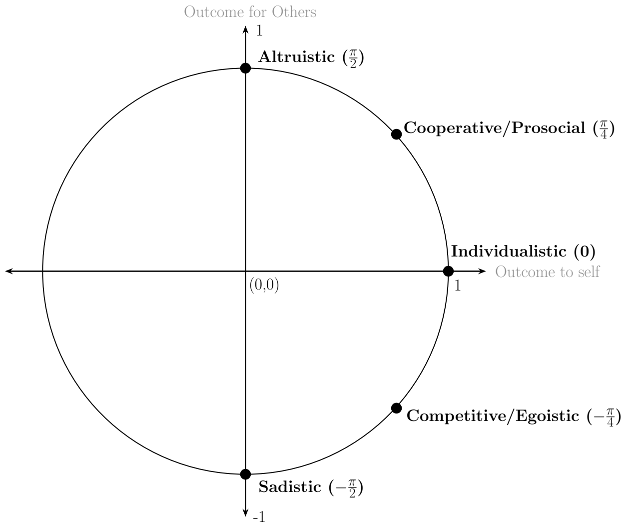


Figure 5.1: SVO Half-Ring

5.1 Fuzzy Systems

5.1.1 Type-1 Fuzzy Systems

Zadeh [16] provided the following definition for fuzzy sets: A fuzzy set is a class with a continuum of membership grades. A fuzzy set A in a referential (universe of discourse) X is characterized by a membership function A which associates with each element $x \in X$ a real number $A(x) \in [0, 1]$, having the interpretation $A(x)$ is the membership grade of x in the fuzzy set A.

$$A : X \longrightarrow [0, 1] \tag{5.1}$$

Where $A(x)$ is the membership degree of x to the fuzzy set A and the collection of all fuzzy subsets of X is denoted by $\mathcal{F}(X)$. The case of $A(x) = 1$ represents full membership of x in A, and $A(x) = 0$ represents no membership. We can define the membership function of a classical set $A \subseteq X$ as its characteristic function:

$$\mu_A(x) = \begin{cases} 1, & \text{if } x \in X \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

This work uses 3 types of membership functions: the Z-shaped membership (zmf), S-shaped membership (smf), and the Gaussian membership (gaussmf), which are described as follows:

$$\mu_{zmf}(x;a,b) = \begin{cases} 1, & x \leq a \\ 1 - 2\left(\frac{x-a}{b-a}\right)^2, & a \leq x \leq \frac{a+b}{2} \\ 2\left(\frac{x-b}{b-a}\right)^2, & \frac{a+b}{2} \leq x \leq b \\ 0, & x \geq b \end{cases} \tag{5.3}$$

$$\mu_{smf}(x;a,b) = \begin{cases} 0, & x \leq a \\ 2\left(\frac{x-a}{b-a}\right)^2, & a \leq x \leq \frac{a+b}{2} \\ 1 - 2\left(\frac{x-b}{b-a}\right)^2, & \frac{a+b}{2} \leq x \leq b \\ 1, & x \geq b \end{cases} \tag{5.4}$$

$$\mu_{gauss}(x;\sigma,c) = e^{\frac{-(x-c)^2}{2\sigma^2}} \tag{5.5}$$

Fuzzy Inference

Zadeh proposed the use of linguistic variables to define degrees of truth, with linguistic variables working as a dictionary that translates linguistic terms into fuzzy sets. A linguistic variable is a quintuple (X, T, U, G, M) in which X is the name of the linguistic variable, and T is the set of linguistic terms that represent the values of the variable. Additionally, U is the *universe* of discourse, G is a collection of syntax that produces the correct expressions in T, and M is a set of semantic rules that map T to a fuzzy set in the universe U [57]. Figure 5.2 demonstrates a set of membership functions for an arbitrary linguistic variable over its universe of discourse.

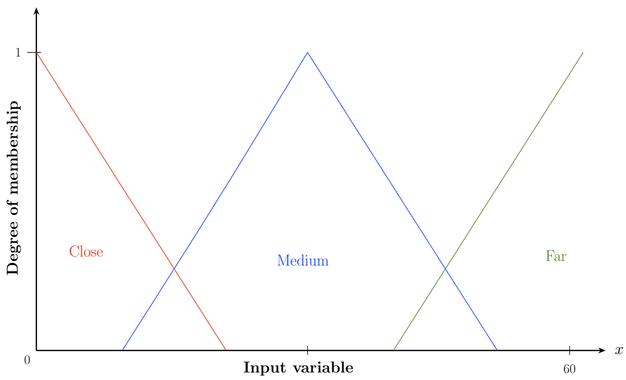


Figure 5.2: Fuzzy Membership Functions Example

Following the definition of the linguistic variable (X, T, U, G, M), a fuzzy ruleset is created. A fuzzy rule is used to either map empirical information, use expert opinion, or even commonsense knowledge to express these rules that would otherwise be difficult to model mathematically. In cases when expert opinion is highly subjective, this can be expressed linguistically through a range of consensus. Uncertainty of this nature is also addressed by type-2 fuzzy systems, which are presented in the following section. It is often intuitive to map a value to an outcome, and fuzzy rules are a method of modeling this. A fuzzy rule is a triplet (A, B, R) that consists of an antecedent $A \in \mathcal{F}(X)$, a consequent $B \in \mathcal{F}(X)$ that are linguistic variables linked through a fuzzy relation $R \in \mathcal{F}(X \times Y)$. Using fuzzy sets a fuzzy rule is written in a *if-then* format as follows:

$$\text{IF } x \text{ is } A_i \text{ THEN } y \text{ is } B_i, \quad i = 1, 2, ...n \tag{5.6}$$

This work uses the Mamdani rule base:

$$R_M(x, y) = \bigvee_{i=1}^{n} A_i x \wedge B_i y, \quad i = 1, 2, ...n \tag{5.7}$$

As individual rules may have more antecedents linked through conjunctions we can have the same situation for a fuzzy rule base [57]:

$$\text{IF } x \text{ is } A_i \text{ AND } y \text{ is } B_i \text{ THEN } z \text{ is } C_i \quad i = 1, 2, ...n \tag{5.8}$$

The Mamdani rule base becomes:

$$R_M(x, y, z) = \bigvee_{i=1}^{n} A_i x \wedge B_i y \wedge C_i z, \quad i = 1, 2, ...n \tag{5.9}$$

With the linguistic variables defined and the Mamdani rule sets created, the process of fuzzy inference is the process of obtaining an output for a given input that was possibly never encountered previously [57]. The compositional rule of inference is based on the classical rule of Modus Ponens $P \to Q, P \vdash Q$. Given a fuzzy rule base $R \in \mathcal{F}(X \times Y)$ the compositional rule of inference is a function $F : \mathcal{F}(X) \to \mathcal{F}(Y)$ determined though the composition $B' = F(A') = A' * R$ with $* : \mathcal{F}(x) \times \mathcal{F}(X \times Y) \to \mathcal{F}(Y)$ being a composition of fuzzy relations [57] such that: x is $A_i \to y$ is B_i, x is $A' \vdash y$ is B'. For a Mamdani inference system, the composition law is written as follows:

$$B'(y) = A' \circ R(x, y) = \bigvee_{x \in X} A'(x) \wedge R(x, y) \tag{5.10}$$

Finally, the output is *defuzzified* to provide us once again with a crisp output. There are different defuzzification methods, such as the centroid method (or center of gravity), center of area, mean of maxima, and maxima criterion, amongst others. This work utilizes the centroid defuzzification method, where the output value selected is the center of gravity of the fuzzy set $u \in \mathcal{F}(X)$ expressed as:

$$COG(u) = \frac{\int_W x \cdot u(x)dx}{\int_W u(x)dx} \tag{5.11}$$

5.1.2 Type-2 Fuzzy Systems

There are different sources of uncertainty that fuzzy systems handle in practical applications and real-world environments. Some of these sources of uncertainty are as follows [58]:

- Uncertainty in inputs due to noise and conditions of observers and sensors.
- Uncertainty caused by changes in the conditions of operation controllers.
- Use of noise data for training parameters.
- ⋆ Uncertainty in modeling through verbal variables.

The last of which is of particular relevance to this work. Where the uncertainty of modeling the behavioral variables of road users is particularly sensitive to geographical norms and tendencies, subjectivity, and lack of empirical data. Where type-1 systems use crisp inputs and utilize membership functions which are applied in an antecedent and consequent manner, type-2 systems pose the membership degree as a fuzzy number. This degree of freedom helps model further uncertainties.

A type-2 fuzzy set $\widetilde{A}$ which is defined by the membership function $0 \leq \mu_{\widetilde{A}}(x,u) \leq 1$ such that $x \in X$ and $u \in J_x \subseteq [0,1]$:

$$\widetilde{A} = \{(x,u), \mu_{\widetilde{A}}(x,u) | \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \tag{5.12}$$

Where J_x is the type-1 primary membership function when $J_x \subseteq [0,1]$ for $\forall x \in X$, and $\mu_{\widetilde{A}}(x,u)$ is a type-2 secondary membership function. If the degree of the secondary membership is 1, then there will be an interval type-2 membership function. The upper-bound membership function (UMF) and the lower-bound membership function (LMF) indicate the highest and lowest degrees of primary membership, respectively. In general, a type-2 membership function can be considered the combination of type-1 membership functions, each of which is called an embedded membership function. If the type-2 membership function is sliced horizontally, the lowest level is called the footprint of uncertainty (FOU) [59]. Figure

5.3 illustrates the relationship between the UMF, LMF and FOU of a membership function for an arbitrary linguistic variable over its universe of discourse.

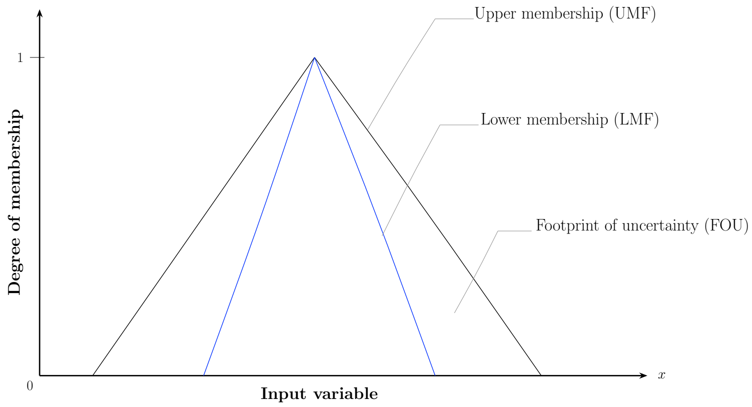


Figure 5.3: Type-2 Upper and Lower Membership Functions

Fuzzification and Ruleset

Similar to type-1 systems, an input vector $x = (x_1, x_2, ..., x_p)^T$ is mapped onto a membership function $\widetilde{A}$ described in Equation 5.12. The ruleset structure is also identical to type-1 systems in which antecedent and consequent membership functions are utilized with the sole difference being that type-2 $\mu_{\widetilde{A}}(x, u)$ functions are used. To describe the ruleset of a type-2 system, we consider the following ruleset R with p inputs $x = (x_1, x_2, ..., x_p)^T$, and output y where the L^{th} rule R is defined as:

$$R^L : \text{IF } x_1 \text{ is } \widetilde{F}_1^L \text{ AND } x_2 \text{ is } \widetilde{F}_2^L, ..., \text{AND } x_p \text{ is } \widetilde{F}_p^L \text{ THEN } y \text{ is } \widetilde{G}^L,\ L = 1, 2, ..., M \quad (5.13)$$

The t-norm and s-nor, also known as the t-conorm, represent the mathematical operators $\wedge$ (logical AND) and $\vee$ (logical OR), respectively. They are used to generalize the degree of truth between [1, 0] and determine the correct degree of rule firing in a fuzzy system. The t-norm and s-norm for a type-2 fuzzy system can be defined as follows:

$$\begin{aligned} T(A, B) &= [\min(\underline{A}(x), \underline{B}(x)), \max(\overline{A}(x), \overline{B}(x))] \\ S(A, B) &= [\max(\underline{A}(x), \underline{B}(x)), \min(\overline{A}(x), \overline{B}(x))] \end{aligned} \quad (5.14)$$

The LMF of the type-2 set is the minimum of LMFs A, B, and the UMF is the maximum of UMFs A, B for the t-norm and vice-versa for the s-norm. This definition captures the intersection of the resulting uncertainty. In most applications, the interval type-2 fuzzy set is used with t-norm and the degree of rule firing becomes [59]:

$$\widetilde{F}^L(X') = [\underline{f}^L(X'), \overline{f}^L(X')] = [\underline{f}^L, \overline{f}^L] \tag{5.15}$$

$$\begin{aligned} \underline{f}^L(X') &= \underline{\mu}_{\widetilde{F}_1^L}(x_1') \cdot ... \cdot \underline{\mu}_{\widetilde{F}_p^L}(x_p') \\ \overline{f}^L(X') &= \overline{\mu}_{\widetilde{F}_1^L}(x_1') \cdot ... \cdot \overline{\mu}_{\widetilde{F}_p^L}(x_p') \end{aligned} \tag{5.16}$$

Type Reduction

To obtain a crisp output from a type-2 system, it must be type-reduced. This work uses the Karnik-Mendel (KM)[34] and Enhanced Karnik-Mendel (EKM)[37] type-reduction methods, which are iterative approaches to obtain crisp type-2 outputs. KM and EKM methods are the most commonly used methods for type reduction, as their relative simplicity and computational efficiency make them possible to use in real-time applications. For an interval type-2 fuzzy system (IT2 FS) we have:

$$y(Z_1, ..., Z_M, W_1, ..., W_M) = \frac{\int_{z_1} \cdots \int_{z_M} \int_{w_1} \cdots \int_{w_M} T_{L=1}^{M} \mu_{z_l}(z_l) * T_{L=1}^{M} \mu_{w_l}(w_l)}{\frac{\sum_{i=1}^{M} w_l z_l}{\sum_{i=1}^{M} w_l}} \tag{5.17}$$

where T and $*$ denote t-norm with $w_l \in W_l, z_l \in Z_l$. Because the secondary membership functions $\mu_{z_l}(z_l) = 1$ and $\mu_{w_l}(w_l) = 1$ as a property of IT2FSs. Thus, Equation 5.17 simplifies to:

$$y(Z_1, ..., Z_M, W_1, ..., W_M) = \frac{\int_{z_1} \cdots \int_{z_M} \int_{w_1} \cdots \int_{w_M}}{\frac{\sum_{i=1}^{M} w_l z_l}{\sum_{i=1}^{M} w_l}} \tag{5.18}$$

In interval type-1 fuzzy sets, memberships are crisp numbers with upper and lower bounds. For general interval type-2 fuzzy systems, each z_l in Equation 5.18 is an interval type-1 set with centroid c_l, expanding on both sides by s_l. Additionally, each w_l is an interval type-1 set with centroid h_l and expansion Δ_l. As y is an interval type-1 set, determining only the first and last points y_l and y_R is required [59]. It is well known that y_l and y_R can be written as:

$$\begin{aligned} y_l &= \min_{w_i \in [\underline{w}_i, \overline{w}_i]} \frac{\sum_{i=1}^{M} \underline{z}_i w_i}{\sum_{i=1}^{M} w_i} = \frac{\sum_{i=1}^{L} \underline{z}_i \overline{w}_i + \sum_{i=L+1}^{M} \underline{z}_i \underline{w}_i}{\sum_{i=1}^{L} \overline{w}_i + \sum_{i=L+1}^{M} \underline{w}_i} \\ y_r &= \max_{w_i \in [\underline{w}_i, \overline{w}_i]} \frac{\sum_{i=1}^{M} \overline{z}_i w_i}{\sum_{i=1}^{M} w_i} = \frac{\sum_{i=1}^{R} \overline{z}_i \underline{w}_i + \sum_{i=R+1}^{M} \overline{z}_i \overline{w}_i}{\sum_{i=1}^{R} \underline{w}_i + \sum_{i=R+1}^{R} \overline{w}_i} \end{aligned} \tag{5.19}$$

The final output y is obtained by the average of y_r and y_l:

$$y = \frac{y_r + y_l}{2} \tag{5.20}$$

where L and R are switch points. There is no closed-form solution for L and R, and hence for y_l and y_r. KM algorithms are used to compute them iteratively, which converges to a solution in at most M iterations [34, 37]. From the following definition:

$$S(w_1, w_2, ..., w_M) = \frac{\sum_{i=1}^{M} w_l z_B}{\sum_{i=1}^{M} w_l} \tag{5.21}$$

where $w_l \in [h_l - \Delta_l, h_l - \Delta_l]$, $h_l \geq \Delta_l$ and $z_l \in [c_l - s_l, c_l + s_l]$. the maximum values of S and y_l are determine through the following procedure [59] by considering $z_l = c_l + s_l$ and $z_1 \leq z_2 \leq ... \leq z_M$:

Step 1: Using Equation 5.21, compute $S' = S(h_1, h_2, ..., h_M)$ by considering $w_l = h_l$

Step 2: Find switch point $k(1 \leq k \leq M - 1)$ when $z_k \leq S' \leq z_{k+1}$.

Step 3: Compute $S'' = S(h_1 - \Delta_l, ..., h_M - \Delta_k)$ for $L \leq k$ and $w_l = h_l + \Delta_l$

Step 4: if $S'' = S'$:

  True Terminate, $S'' = \max(S)$

  False Repeat **Step 2**, set $S'' = S$

The Enhanced Karnik-Mendel (EKM) algorithm [37] brings computational improvements to the original KM method. Although still an iterative process, by providing a better initial guess to the switch points R and L, as well as providing a termination condition. Similar to the KM method, y_r and y_l are found iteratively, and the initial conditions for y_r and y_l are shown in Equation 5.19. For the EKM method, the optimal initial conditions are shown as:

$$\begin{aligned} y_l &= \frac{\sum_{i=1}^{M} \underline{z}_i \frac{\overline{w}_i + \underline{w}_i}{2}}{\sum_{i=1}^{M} \frac{\overline{w}_i + \underline{w}_i}{2}} \\ y_l &= \frac{\sum_{i=1}^{M} \overline{z}_i \frac{\underline{w}_i + \overline{w}_i}{2}}{\sum_{i=1}^{M} \frac{\underline{w}_i + \overline{w}_i}{2}} \end{aligned} \tag{5.22}$$

Observing the original KM-method initialization of y_r, y_l in Equation 5.19, we notice that when $i \leq L$, $\overline{w}_i$ is used to compute y_l, and when $i > L$, $\underline{w}_i$ is used to compute y_l. This implies that a better initialization of the parameters is possible by finding appropriate guesses for L and R, L_0, R_0. Because y_l is the smallest value of Y , we conjecture that very probably it is smaller than [34] $\underline{z}_{[M/2]}$, the center element of zi; consequently, L_0, should also be smaller than $[M/2]$, and so forth for y_r [37]. Wu and Mendel found through simulation that initializing $L_0 = [M/2.4]$ and $R_0 = [M/1.7]$ provided the fewest number of iterations to converge to a solution.

The original KM-method termination condition is $S'' = S'$. Since S'' is obtained in the current iteration and S' from the previous iteration, $S'' = S'$ implies that the current iteration does not contribute to minimizing y_l, allowing its deletion without altering y_l. The switch points for S'' and S' are denoted as k' and k, respectively. By changing the termination condition from $S'' = S'$ to $k' = k$, we achieve the same y_l while saving one iteration [37].

The computational cost can further be reduced by removing redundant iterations. The KM-method computes $\sum_{M}^{i=1} w_i$ and $\sum_{i=1}^{M} \underline{z}_i w_i$ to compute S''. This does not retain or utilize the results from previous iterations. During the j^{th} iteration, the switch point k , $\sum_{i=1}^{M} w_i$ and $\sum_{i=1}^{M} \underline{z}_i w_i$ is denoted as $\left(\sum_{i=1}^{M} w_i\right)_j$ and $\left(\sum_{i=1}^{M} \underline{z}_i w_i\right)_j$. Wu and Mendel propose that k_j and k_{j+1} are close in value, and as a consequence, the value of w_i value of the j^{th} iteration share many common terms with the $j + 1^{th}$, such that only the differences between $\left(\sum_{i=1}^{M} w_i\right)_j$, $\left(\sum_{i=1}^{M} w_i\right)_{j+1}$ and $\left(\sum_{i=1}^{M} \underline{z}_i w_i\right)_j$, $\left(\sum_{i=1}^{M} \underline{z}_i w_i\right)_{j+1}$ must be computed. As with the procedure shown for the KM type reduction method, the EKM procedure is fully described in [37].

5.2 Social Value Operationalization

The operationalization process involves translating the fuzzy concept of social value that is not directly measurable into a crisp and distinguishable metric. By providing a framework in which well-defined measurable data points can be translated into a social value, we aim to create a robust foundation for risk attribution for non-autonomous road users.

5.2.1 Social Cues

To measure the non-verbal communication between road users and the autonomous ego-vehicle, different social behaviors and actions were identified for pedestrians and drivers. These behaviors were selected based on the following factors:

- Explicitly observable and measurable.
- Implies an intent or a social contract.

- Has a spectrum of implied intent, where the measured amount (duration, frequency, etc.) of said action implies a different intent.
- The spectrum can be mapped between an outcome for self (egoistic behavior) and an outcome for others (altruistic behavior).

Because the social contract of non-verbal communication between vehicle-to-vehicle and vehicle-to-pedestrian differ vastly, different social cues between each were identified for each. These actions and their definitions are summarized in Table 5.1. Furthermore, to correctly attribute a social value to an observed behavior, each action must represent a cooperative or egoistical intent and the quantity/frequency of the observed action determines the intensity of the intent. A summary of each social cue's sociability spectrum is given in Table 5.3.

Table 5.1: Road-users Social Cues and Type-2 Uncertainty

Cue	**Pedestrian Social Cue Description**	**Uncertainty**
1	**Distance to Crosswalk (m):** The pedestrian's Euclidean distance to the closest crosswalk beginning point is measured.	*Low*
2	**Time Waiting (s):** Time spent waiting at a crosswalk point before crossing.	*Medium*
3	**Time Looking (s):** Time spent looking at oncoming traffic before crossing.	*High*
Cue	**Vehicle Social Cue Description**	**Uncertainty**
1	**Speed Limit (%):** The vehicle's maximum travel speed within a time window T_w as a percentage of the road's posted speed limit.	*Low*
2	**Lane Centering (m):** Distance from the lane center within time window T_w.	*Medium*
3	**Following Time (s):** Following distance in seconds of travel to proceeding vehicle (Time-to-impact).	*Medium*
4	**Lane Changes ($\frac{changes}{min}$):** Lane change frequency of vehicle.	*Medium*

Table 5.2: Type-2 Uncertainty Operationalization

Uncertainty	**UMF Lead**	**LMF Lag**	**LMF Scale**
Low	0.05	0.05	1
Medium	0.1	0.1	0.95
High	0.15	0.15	0.90

The uncertainty associated with the measurement of social cues is expressed through the interval between the UMFs and LMFs. Through logical reasoning, the determination of a pedestrian's distance to a crosswalk is relatively straightforward. A pedestrian's position is simply referenced to a map of known crosswalk positions from the OpenDRIVE information. We can thus make the assumption that the uncertainty in this measurement is *low*. However, the social cue of a pedestrian looking at oncoming traffic represents a less trivial measurement. Not only is the pose of the head relative to a pedestrian's body much more difficult to determine than their general spatial position, but a small difference in their gaze can make the difference between looking at oncoming traffic or the scenery. It is thus assumed that the uncertainty in this process is *high*. Table 5.1 summarizes the estimated uncertainty surrounding each social cue.

The *low*, *medium* and *high* uncertainties dictate how large the FOU will be for the membership functions associated with that behavior. Using the type-1 membership functions as a baseline, the lead and lag parameters determine the positions of the UMF and LMFs relative to the type-1 distribution respectively. The scale represents the gain applied to the LMF relative to the UMF. Table 5.2 summarizes the lead, lag and scale values used for each uncertainty profile.

The choice of membership function shapes is problem-specific and there are many ways to characterize fuzziness, such as looking at the distribution of the data. However, the trial and error method is often used for MF shape because there is no exact method for choosing the MFs. The shape of MFs depends on how one believes in a given linguistic variable. The basic problem with modeling a situation is breaking the 0–1 modeling. Generally speaking, triangular MF is one of the most frequently encountered practices. Of highly applied MFs, the triangular MFs are formed using straight lines. These straight-line membership functions have the advantage of simplicity. Gaussian MFs are popular methods for specifying fuzzy sets because of their smoothness and concise notation. These curves have the advantage of being smooth, nonzero at all points [60, 61], with guaranteed continuity, and are faster to compute for smaller rule bases [62]. Gaussian MFs were used in this work, as the supposition is that the noise and statistical distribution present at the sensor and detection level of social cues would present a Gaussian noise distribution [63].

Table 5.3: Cue Spectrum of Sociability

Pedestrian
Distance to Crosswalk
↗ **Far:** Represents no intent, or willing to let others pass (Cooperative)
↘ **Close:** Represents an intent to cross (Competitive)
Time Waiting
↗ **Long:** Represents a sense of self-preservation (Cooperative)
↘ **Short:** Represents no sense of self-preservation (Competitive)
Time Looking
↗ **Long:** Represents a willingness to engage in a social contract (Cooperative)
↘ **Short:** Represents no acknowledgement of others (Competitive)

Vehicles
Speed Limit
↗ **Low:** Allows others to get ahead at their expense (Cooperative)
↘ **High:** Gets themselves ahead at the expense of others (Competitive)
Lane Centering
↗ **Good:** Does not show intent for passing (Cooperative)
↘ **Poor:** Shows intent for passing (Competitive)
Following Time
↗ **Short:** Does not pressure others and allows for merging space (Cooperative)
↘ **Long:** Applies pressure to others for personal benefit (Competitive)
Lane Changes
↗ **Few:** Is patient for better collective outcome (Cooperative)
↘ **Many:** favours personal outcome over collective outcome (Competitive)

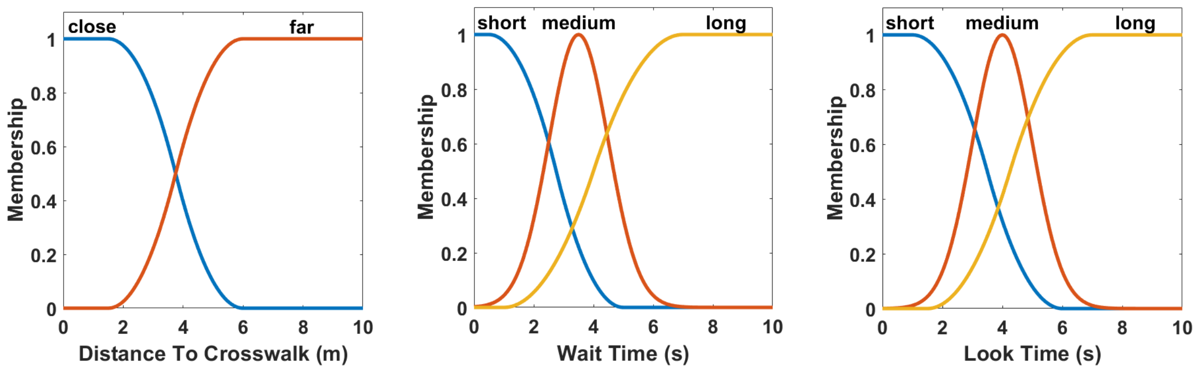
5.3 Pedestrian Social Value Estimator

5.3.1 Pedestrian Type-1 Fuzzy Estimator

From the actions identified in Table 5.1 and their spectrum of sociability summarized in Table 5.3, a fuzzy input vector is created to define the linguistic variables $L_M = (X, T, U, G, M),\ M = 3$ for the identified social cues.

Inputs

Figure 5.4 illustrates the input membership functions for each input variable obtained from the observed pedestrian behavior. The characteristics of each function are further discussed below.



(a) L_1 (b) L_2 (c) L_3

Figure 5.4: Type-1 Pedestrian Input Membership Functions

For L_1, X is *Distance to Crosswalk* with linguistic values T: $\{close, far\}$, over a universe of discourse $U = [0, 10]$ (m). The syntax rules G *close* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M are represented by ZMF $\mu_{close} = (1.5, 6)$, and SMF $\mu_{far} = (1.5, 6)$.

For L_2, X is *Wait Time* with linguistic values T: $\{short, medium, long\}$, over a universe of discourse $U = [0, 10]$ (s). The syntax rules G *short* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M is represented by ZMF $\mu_{short} = (0.5, 5)$, Gaussian $\mu_{medium} = (1, 3.5)$ and SMF $\mu_{long} = (1, 7)$.

For L_3, X is *Look Time* with linguistic values T: $\{short, medium, long\}$, over a universe of discourse $U = [0, 10]$ (s). The syntax rules G *short* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M is represented by ZMF $\mu_{short} = (1, 6)$, Gaussian $\mu_{medium} = (1, 4)$ and SMF $\mu_{long} = (1.5, 7)$.

Ruleset

A commonsense ruleset was created, such that the triggering of more *competitive* membership functions would result in more anti-social labeling and vice versa. A set of 11 rules was created to cover all the combinations of membership activation without redundancy. Table 5.4 summarizes the pedestrian ruleset.

Table 5.4: Pedestrian Fuzzy Ruleset

1. (D. Crosswalk is Far) $\wedge$ (Wait Time is Long) $\wedge$ (Look Time is Long) $\rightarrow$ SVO is Altruist
2. (D. Crosswalk is Far) $\wedge$ (Wait Time is Long) $\wedge$ (Look Time is Med.) $\rightarrow$ SVO is Altruist
3. (D. Crosswalk is Far) $\wedge$ (Wait Time is Med.) $\wedge$ (Look Time is Med.) $\rightarrow$ SVO is Cooperative
4. (D. Crosswalk is Far) $\wedge$ (Wait Time is Med.) $\wedge$ (Look Time is Short) $\rightarrow$ SVO is Cooperative
5. (D. Crosswalk is Close) $\wedge$ (Wait Time is Long) $\wedge$ (Look Time is Long) $\rightarrow$ SVO is Cooperative
6. (D. Crosswalk is Close) $\wedge$ (Wait Time is Long) $\wedge$ (Look Time is Med.) $\rightarrow$ SVO is Cooperative
7. (D. Crosswalk is Far) $\wedge$ (Wait Time is Short) $\wedge$ (Look Time is Short) $\rightarrow$ SVO is Individualist
8. (D. Crosswalk is Close) $\wedge$ (Wait Time is Long) $\wedge$ (Look Time is Short) $\rightarrow$ SVO is Individualist
9. (D. Crosswalk is Close) $\wedge$ (Wait Time is Short) $\wedge$ (Look Time is Med.) $\rightarrow$ SVO is Competitive
10. (D. Crosswalk is Close) $\wedge$ (Wait Time is Med.) $\wedge$ (Look Time is Short) $\rightarrow$ SVO is Competitive
11. (D. Crosswalk is Close) $\wedge$ (Wait Time is Short) $\wedge$ (Look Time is Short) $\rightarrow$ SVO is Sadistic

Outputs

The output membership functions were also created based on a commonsense approach, such that each label seen in the right-hand-plane of the social value orientation circle in Figure 5.1 represents one-fifth of the spectrum as a Gaussian distribution function. Because the SVO output is used as an additive feature to that actor's potential field, altruistic behavior is a negative SVO, so its effect will decrease that actor's potential. Inversely, the positive sign on competitive and sadistic SVO will increase their effect.

It is given that all output memberships follow the Gaussian distribution presented in Equation 5.5, with $\mu_{alt} = (0.25, -1)$, $\mu_{coop} = (0.25, -0.5)$, $\mu_{ind} = (0.25, 0)$, $\mu_{comp} = (0.25, 0.5)$, $\mu_{sad} = (0.25, 1)$ and is shown in Figure 5.5.

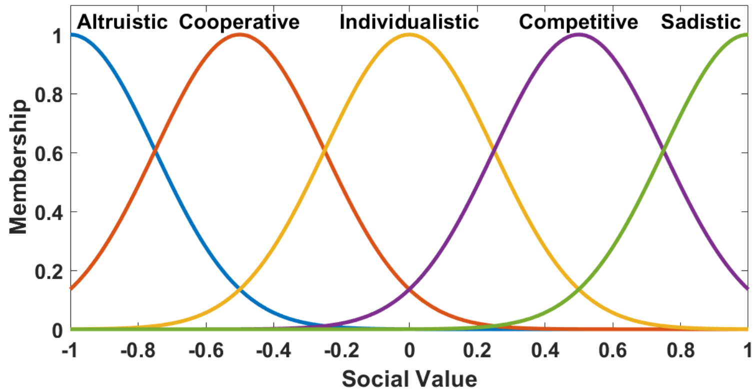


Figure 5.5: Type-1 Pedestrian Output Membership Functions

Pedestrian Type-1 Social Value Estimation Example

An example of the type-1 fuzzy estimator for pedestrians is shown in Figure 5.6. The input vector to the fuzzy estimation process is given as $x_{crosswalk} = 3$, $x_{wait} = 5$, $x_{look} = 5$.

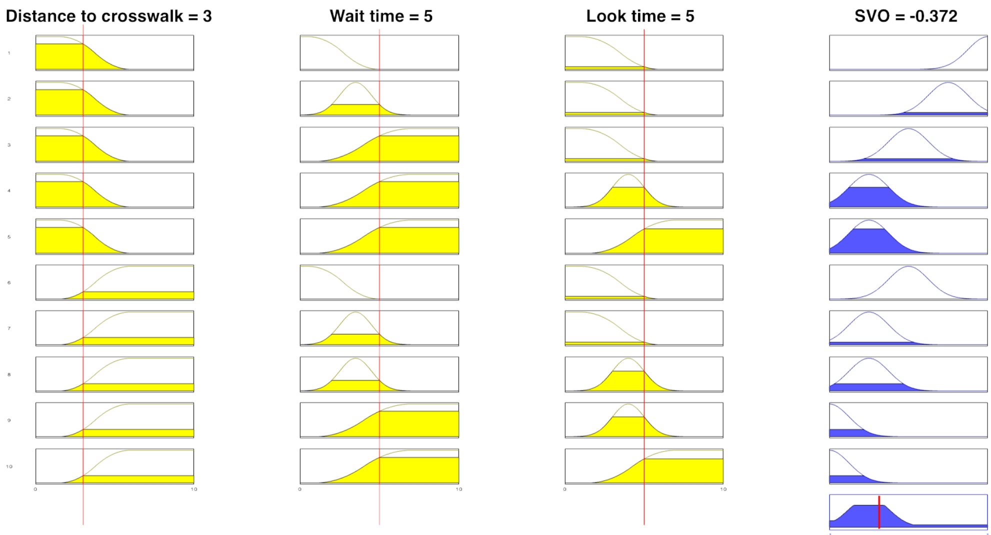


Figure 5.6: Type-1 Pedestrian Estimator Membership Activation

Observing the activated membership functions in Figure 5.6, with the fuzzy ruleset described above, we demonstrate that this provides an SVO output of 0.372, which is a mostly *Cooperative* behavior when plotted on the SVO circle. Figure 5.7 displays the estimator output on the RHP of the social value orientation circle.

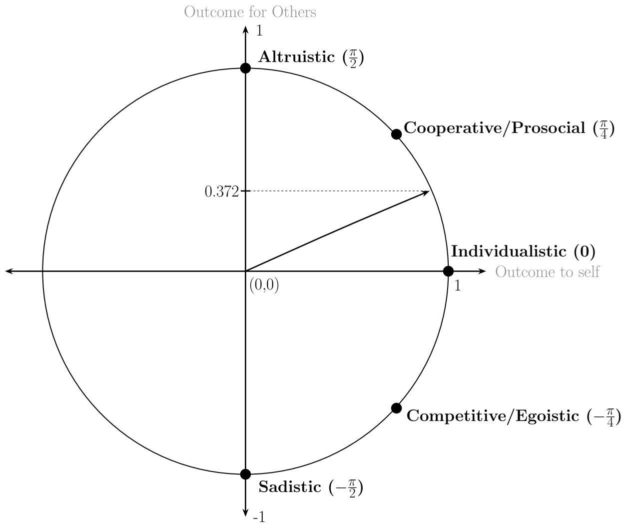


Figure 5.7: Type-1 Pedestrian Estimator Example Social Value

Discussing the results of the example, it seems plausible that a pedestrian who has been waiting for 5 seconds at a crosswalk ($x_{wait} = 5$), has been looking at oncoming traffic for 5 seconds ($x_{look} = 5$) and is 3 meters away from a crosswalk ($x_{crosswalk} = 3$) exhibits a good willingness to cooperate with drivers, but is still showing their intent to cross by being close to a crosswalk beginning point. A social value of 0.372 seems like an adequate estimate for the behavior exhibited.

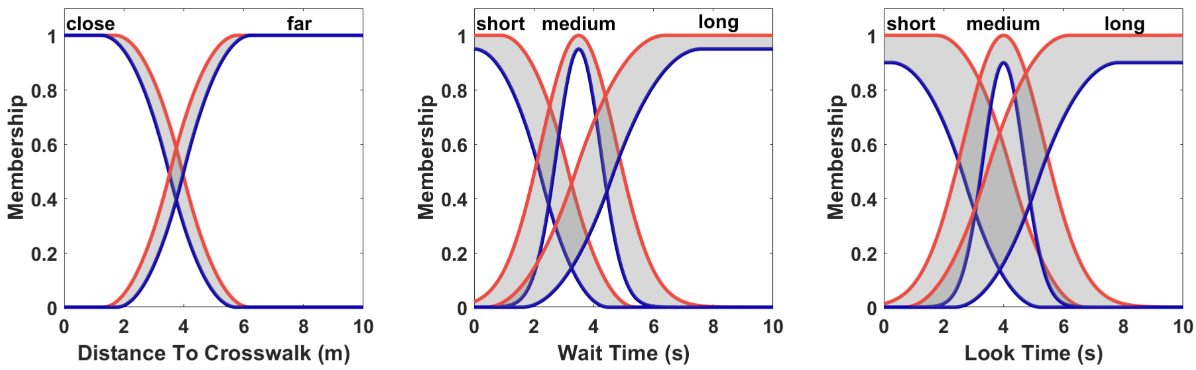
5.3.2 Pedestrian Type-2 Fuzzy Estimator

A type-2 fuzzy estimator is introduced to better represent the uncertainty of the estimation process, which is of particular concern when using a commonsense approach for the linguistic variables introduced for the type-1 fuzzy estimator. As mentioned earlier, the uncertainty of modelling with linguistic variables, as well as uncertainty and noise when measuring social cues, prompt the use of type-2 fuzzy estimators to model these phenomena.

Similar to the type-1 pedestrian estimator defined above, the same linguistic variables $L_M = (X, T, U, G, M),\ M = 3$ were identified for the type-2 system. The upper membership functions (UMFs) are set equal to the type-1 memberships and the lower membership functions (LMFs) are implemented with identical distributions to their respective UMF but with a lag of 0.2 and a scale of 1.

Inputs

Figure 5.8 illustrates the type-2 input membership functions for each input variable obtained from the observed pedestrian behavior including the FOUs created between the UMFs and LMFs. The parameters pertaining to the lead and lag of the UMFs and LMFs relative to the type-1 memberships are shown in Table 5.2.



(a) L_1 (b) L_2 (c) L_3

Figure 5.8: Type-2 Pedestrian Input Membership Functions

Ruleset

The ruleset for the type-2 system is identical to the ruleset presented for the type-1 system. This is congruent with the fact that the logical assertion of an actor's social value does not change between type-1 and type-2 systems but rather only the uncertainty surrounding the measurement and labeling of their behavior. The pedestrian ruleset is summarized in Table 5.4.

Outputs

The output membership functions follow the same commonsense approach as presented in the type-1 estimator in Figure 5.5, with each of the 5 labels representing one-fifth of the spectrum. The LMF distributions follow the same distributions as the type-1 estimator but have a lag parameter of 0.2 and a scale of 1 with their respective type-2 MFs shown in Figure 5.9.

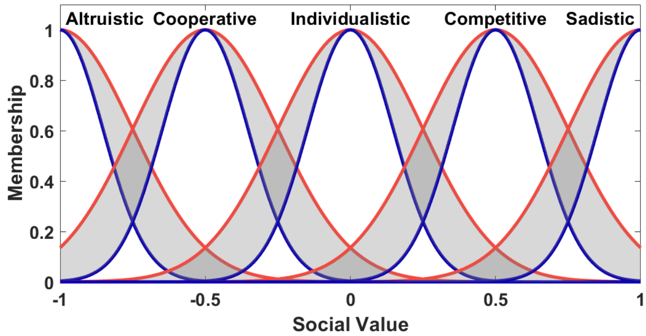


Figure 5.9: Type-2 Pedestrian Output Membership Functions

Pedestrian Type-1 and Type-2 Comparison

As a matter of comparison, the same input vector is used as shown in Figure 5.6. The input vector to the type-2 fuzzy estimation process is given as: $x_{crosswalk} = 3$, $x_{wait} = 5$, $x_{look} = 5$. Observing the activated type-2 membership functions in Figure 5.10, we obtain an SVO output of 0.316 which is more *individualist* than the type-1 estimator from Figure 5.6. In both cases, the label attributed to the actor is congruent with the observed behavior.

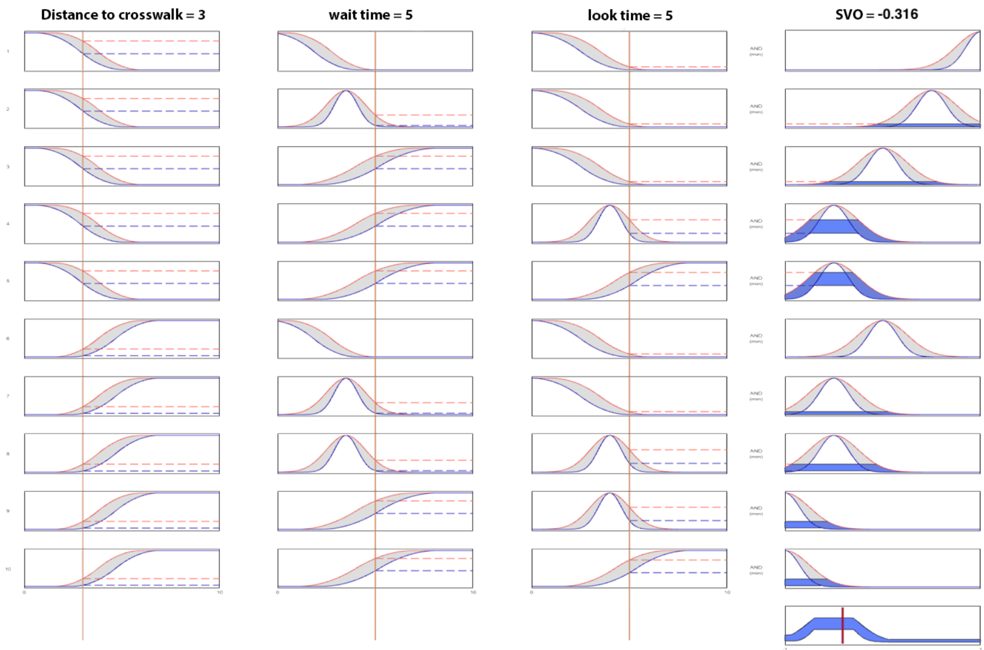


Figure 5.10: Type-2 Pedestrian Estimator Membership Activation

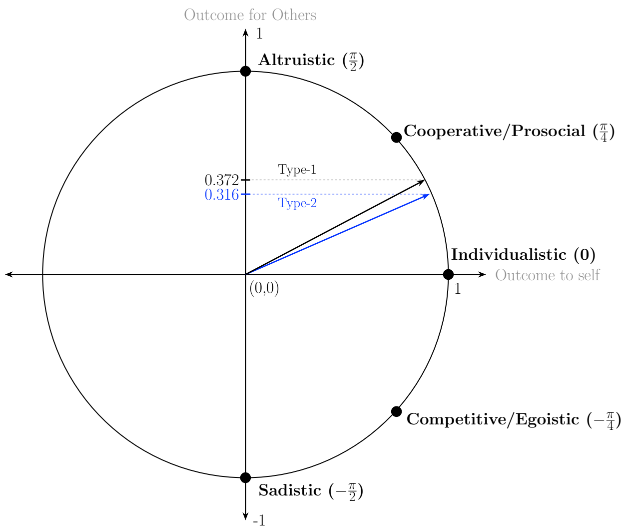
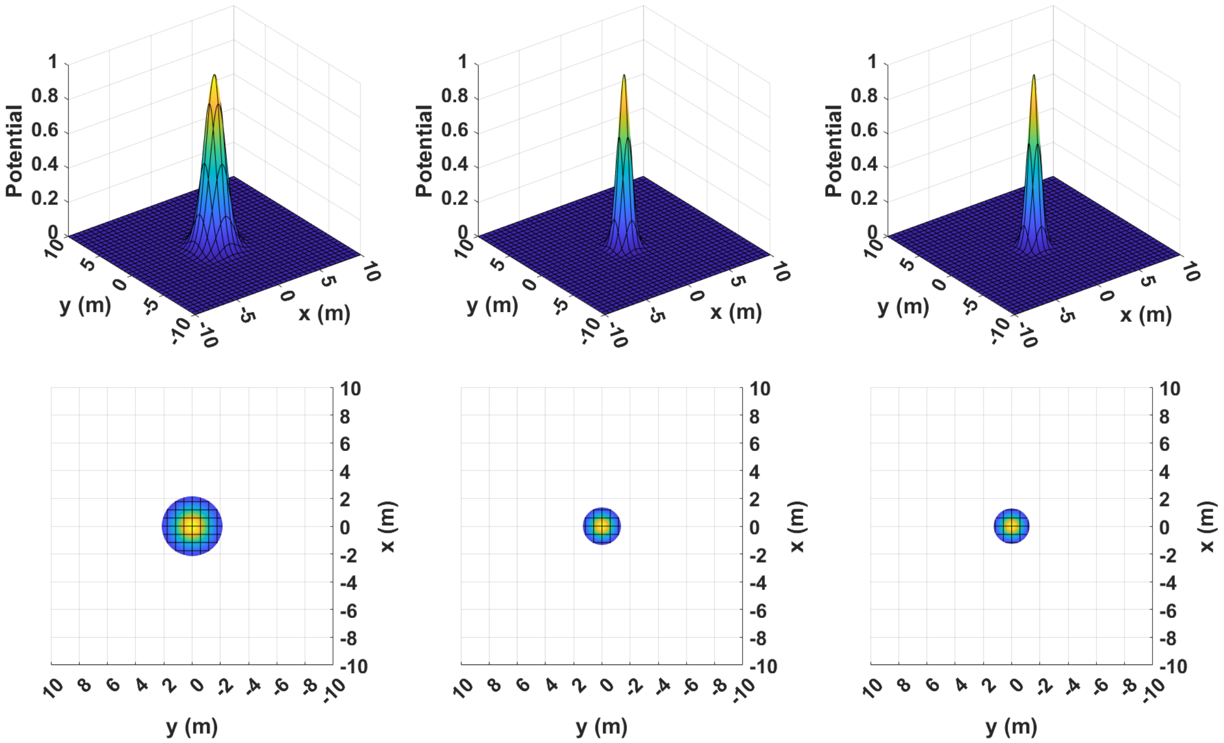


Figure 5.11: Type-2 and Type-1 Pedestrian Estimator Comparison

In both cases, the label attributed to the actor is congruent with the observed behavior. Showing the potential field created by a static pedestrian with no SVO, type-1 SVO, and type-2 SVO, we have the potential field distributions shown in Figure 5.12.

Observing the potential field distributions shown in Figure 5.12, we note that smaller SVO values (cooperative) cause smaller potentials. If the safety perimeter around a potential field is delimited at a potential value of 0.1 (10% of maximum), the decrease in potential from the effect of SVO represents a decrease in lateral safety distance from 2.146m with no SVO component ($\Omega = 0$) to 1.352m with type-1 estimation ($\Omega = 0.372$) and 1.423m with type-2 estimation $\Omega = 0.831$. This represents an decrease of safety margin decrease of 36.99% and 33.83% for type-1 and type-2 respectively. The gain parameter α shown in equations 4.17 and 4.18 can help modulate this decrease in safety margin, where $0 \leq \alpha \leq 1$. In other words, the type-2 estimator was more conservative in its estimation. The type-1 and type-2 output on the social value orientation circle is shown in Figure 5.11.

In both cases, the SVO estimators allow for less risk to be attributed to individuals who are perceived to be cooperative, without totally neglecting their effect.



(a) None (b) Type-1 (c) Type-2

Figure 5.12: Type-1 & Type-2 Estimation for Pedestrian Potential Field, 10% Safety Contour

5.4 Vehicle Social Value Estimator

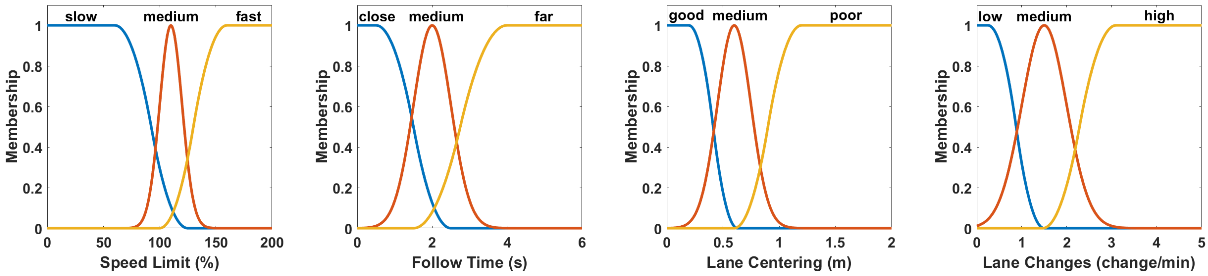
5.4.1 Vehicle Type-1 Fuzzy Estimator

From the actions identified in Table 5.1 and their spectrum of sociability summarized in Table 5.3, a fuzzy input vector is created to define the linguistic variables $L_M = (X, T, U, G, M),\ M = 4$ for the identified social cues.

Inputs

Figure 5.13 illustrates the input membership functions for each input variable obtained from the observed vehicle actor behavior. The specifications of each linguistic variable is further discussed below.

For L_1, X is *Speed Limit* with linguistic values T: {*slow, medium, fast*}, over a universe of discourse $U = [0, 200]$ (%). The syntax rules G *slow* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth.



(a) L_1 (b) L_2 (c) L_3 (d) L_4

Figure 5.13: Type-1 Vehicle Input Membership Functions

The set of membership functions M are represented by ZMF $\mu_{slow} = (60, 125)$, and Gaussian $\mu_{medium} = (10, 110)$ and SMF $\mu_{fast} = (100, 160)$.

For L_2, X is *Follow Time* with linguistic values T: $\{$*close, medium, far*$\}$, over a universe of discourse $U = [0, 6]$ (s). The syntax rules G *close* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M is represented by ZMF $\mu_{close} = (0.5, 2.5)$, Gaussian $\mu_{medium} = (0.5, 2)$ and SMF $\mu_{far} = (1.5, 4)$.

For L_3, X is *Lane Centering* with linguistic values T: $\{$*good, medium, poor*$\}$, over a universe of discourse $U = [0, 2]$ (m). The syntax rules G *good* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M is represented by ZMF $\mu_{good} = (0.2, 0.6)$, Gaussian $\mu_{medium} = (0.15, 0.6)$ and SMF $\mu_{poor} = (0.6, 1.2)$.

For L_4, X is *Lane Changes* with linguistic values T: $\{$*low, medium, high*$\}$, over a universe of discourse $U = [0, 5]$ (changes/min). The syntax rules G *low* $\in G$, if $x \in G$ then *very* $x \in G$ and so forth. The set of membership functions M is represented by ZMF $\mu_{low} = (0.25, 1.5)$, Gaussian $\mu_{medium} = (0.5, 1.5)$ and SMF $\mu_{high} = (1.5, 3)$.

Ruleset

The ruleset for vehicle social value follows the same commonsense approach as the pedestrian estimator, such that the triggering of more competitive membership functions would result in more anti-social labeling and vice versa. A set of 15 rules was created to cover all the combinations of membership activation without redundancy. Table 5.5 summarizes the vehicle ruleset.

Table 5.5: Vehicle Fuzzy Ruleset

1. (Follow T. is Far) $\wedge$ (L. Changes is Low) $\wedge$ (L. Centering is good) $\wedge$ (Speed Lim. is Slow) $\rightarrow$ SVO is Altruist
2. (Follow T. is Far) $\wedge$ (L. Changes is Low) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Cooperative
3. (Follow T. is Far) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is good) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Cooperative
4. (Follow T. is Far) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Slow) $\rightarrow$ SVO is Cooperative
5. (Follow T. is Med.) $\wedge$ (L. Changes is Low) $\wedge$ (L. Centering is good) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Cooperative
6. (Follow T. is Med.) $\wedge$ (L. Changes is Low) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Slow) $\rightarrow$ SVO is Cooperative
7. (Follow T. is Med.) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is good) $\wedge$ (Speed Lim. is Slow) $\rightarrow$ SVO is Cooperative
8. (Follow T. is Med.) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Individualist
9. (Follow T. is Close) $\wedge$ (L. Changes is High) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Competitive
10. (Follow T. is Close) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is Poor) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Competitive
11. (Follow T. is Close) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Fast) $\rightarrow$ SVO is Competitive
12. (Follow T. is Med.) $\wedge$ (L. Changes is High) $\wedge$ (L. Centering is Poor) $\wedge$ (Speed Lim. is Med.) $\rightarrow$ SVO is Competitive
13. (Follow T. is Med.) $\wedge$ (L. Changes is High) $\wedge$ (L. Centering is Med.) $\wedge$ (Speed Lim. is Fast) $\rightarrow$ SVO is Competitive
14. (Follow T. is Med.) $\wedge$ (L. Changes is Med.) $\wedge$ (L. Centering is Poor) $\wedge$ (Speed Lim. is Fast) $\rightarrow$ SVO is Competitive
15. (Follow T. is Close) $\wedge$ (L. Changes is High) $\wedge$ (L. Centering is Poor) $\wedge$ (Speed Lim. is Fast) $\rightarrow$ SVO is Sadistic

Outputs

The output membership functions are identical to the pedestrian output memberships shown in Figure 5.5, with each label seen in the right-hand-plane of the social value orientation circle in Figure 5.1 representing one-fifth of the spectrum.

Vehicle Type-1 Social Value Estimation Example

An example of the type-1 fuzzy estimator for pedestrians is shown in Figure 5.14. The input vector to the fuzzy estimation process is given as $x_{speed} = 160$, $x_{follow} = 1$, $x_{centering} = 2$, $x_{changes} = 5$.

Observing the activated membership functions in Figure 5.14, with the fuzzy ruleset described above, we demonstrate that this provides an SVO output of -0.801, which is a very *competitive/sadistic* behavior when plotted on the SVO circle. Figure 5.7 displays the estimator output on the RHP of the social value orientation circle.

Analyzing the example's outcomes, it appears plausible that the driver exhibits highly competitive behavior with an SVO value of 0.801. The result is also shown on the social value orientation circle in Figure 5.15. This behavior is evident in a vehicle traveling at 160% over the speed limit ($x_{speed} = 160$), following the next vehicle (i.e. *tailgating*) with only a 1-second time-to-impact distance ($x_{follow} = 1$), maintaining a distance of 2 meters from the center of their lane ($x_{centering} = 2$),

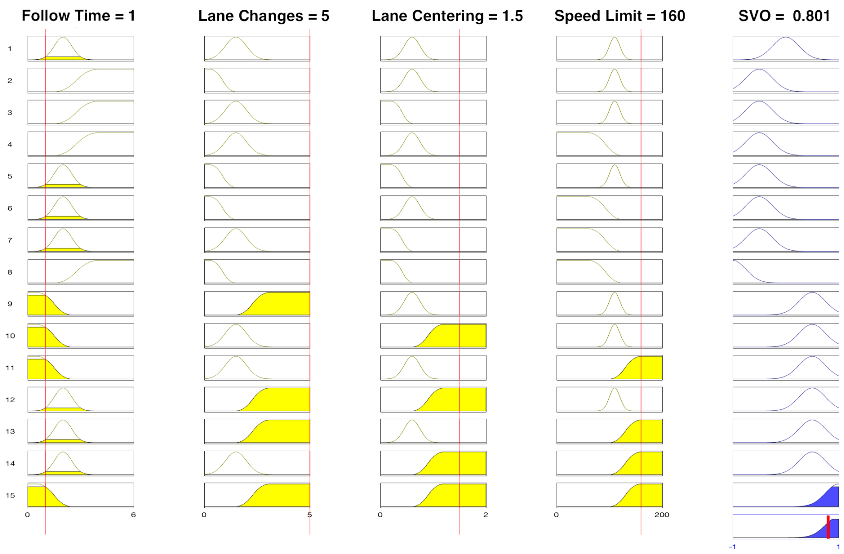


Figure 5.14: Type-1 Vehicle Estimator Membership Activation

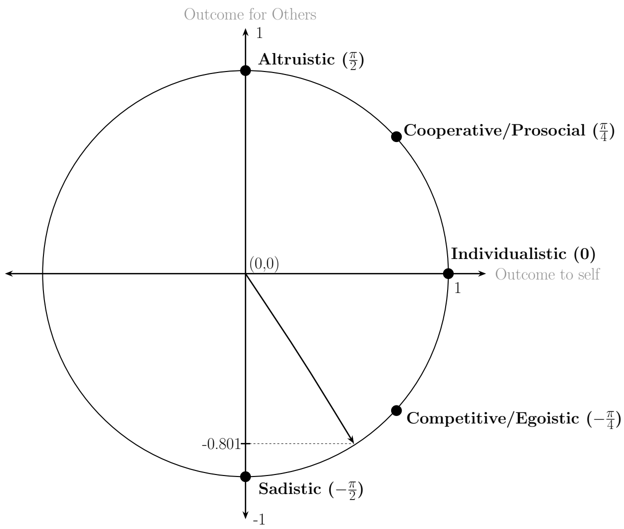


Figure 5.15: Type-1 Vehicle Estimator Example Social Value

and executing 5 lane changes per minute ($x_{changes} = 5$). Given these parameters, the label of *very competitive* is fitting for this driver's behavior.

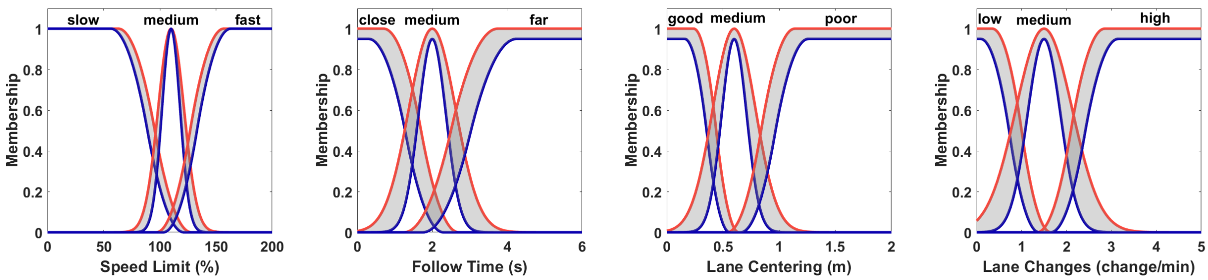
5.4.2 Vehicle Type-2 Vehicle Fuzzy Estimator

As previously mentioned with the type-2 pedestrian SVO estimator, a type-2 fuzzy process models the uncertainty of a commonsense approach to linguistic variables, as well as uncertainty and noise when measuring social cues.

The same linguistic variables $L_M = (X, T, U, G, M)$, $M = 4$ were identified for the type-2 system. The upper membership functions (UMFs) are set equal to the type-1 memberships and the lower membership functions (LMFs) are implemented with identical distributions to their respective UMF but with a lag of 0.2 and scale of 1.

Inputs

Figure 5.16 illustrates the type-2 input membership functions for each input variable obtained from the observed vehicle actor behavior including the FOUs created between the UMFs and LMFs. The parameters pertaining to the lead and lag of the UMFs and LMFs relative to the type-1 memberships are shown in Table 5.2.



(a) L_1 (b) L_2 (c) L_3 (d) L_4

Figure 5.16: Type-2 Vehicle Input Membership Functions

Ruleset

The ruleset for the type-2 system is identical to the ruleset presented for the type-1 system. This is congruent with the fact that the logical assertion of an actor's social value does not change between type-1 and type-2 systems but rather only the uncertainty surrounding the measurement and labeling of their behavior. Table 5.5 summarizes the vehicle ruleset.

Outputs

The output membership functions follow the same format as shown in the pedestrian Type-2 output memberships in Figure 5.9.

Vehicle Type-1 and Type-2 Comparison

As a matter of comparison, the same vehicle behavior input vector is used as shown in Figure 5.14. The input vector to the fuzzy estimation process is given as $x_{speed} = 160$, $x_{follow} = 1$, $x_{centering} = 2$, $x_{changes} = 5$.

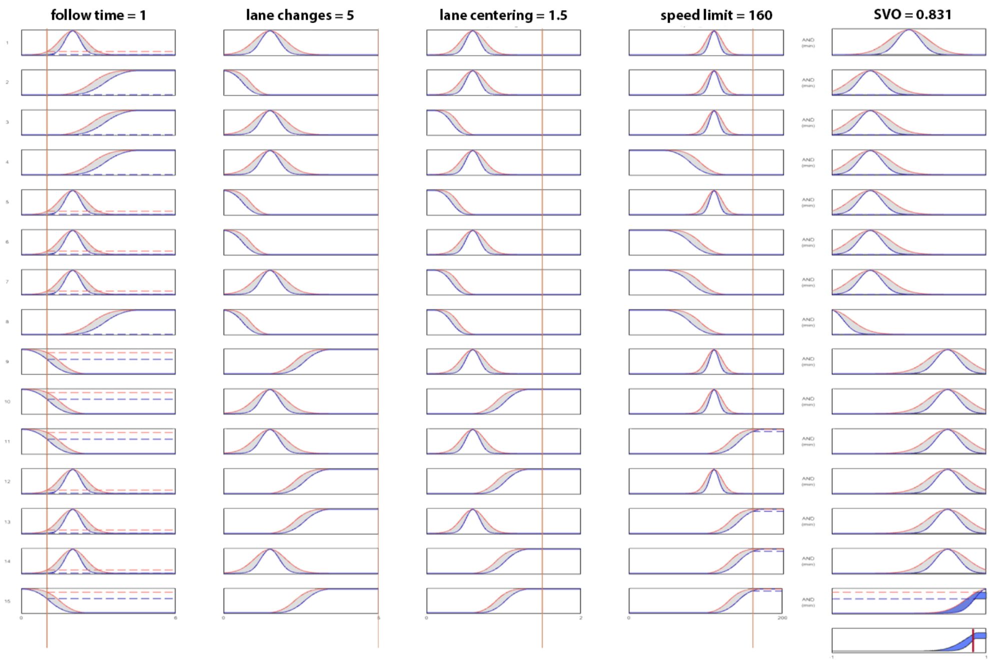


Figure 5.17: Type-2 Vehicle Estimator Membership Activation

Observing the activated type-2 membership functions in Figure 5.17, we obtain an SVO output of -0.831 which is more *sadistic* than the type-1 estimator from Figure 5.14. Type-1 and type-2 results are compared on the social value orientation circle in Figure 5.18. In both cases, the label attributed to the actor is congruent with the observed behavior. Showing the potential field created by a static vehicle with no SVO, type-1 SVO, and type-2 SVO, we have the potential field distributions shown in Figure 5.19.

Observing the potential field distributions shown in Figure 5.19, we note that larger SVO values (egotism) cause larger potentials. If the safety perimeter around a potential field is delimited at a potential value of 0.1 (10% of maximum), the increase in potential field from the effect of SVO represents an increase in lateral safety distance from 2.92m with no SVO component ($\Omega = 0$) to 5.27m with type-1 estimation ($\Omega = 0.801$) and 5.36m with type-2 estimation $\Omega = 0.831$. This represents a safety margin increase of 80.4% and 83.5% for type-1 and type-2, respectively. The gain parameter α shown in equations 4.17 and 4.18 can help modulate this increase in safety margin, where $0 \leq \alpha \leq 1$.

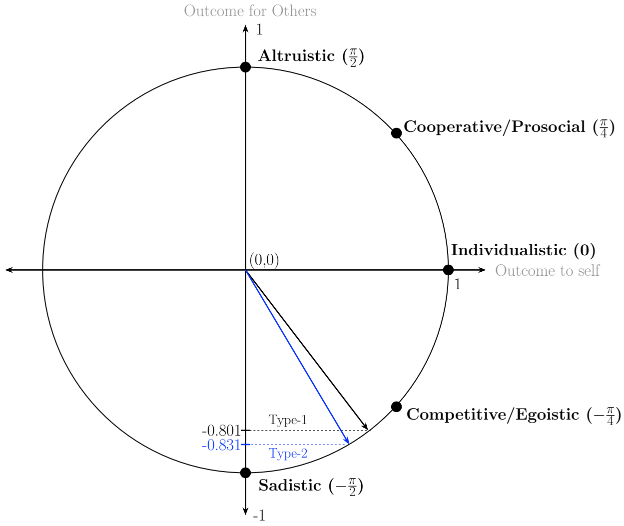


Figure 5.18: Type-2 and Type-1 Vehicle Estimator Comparison

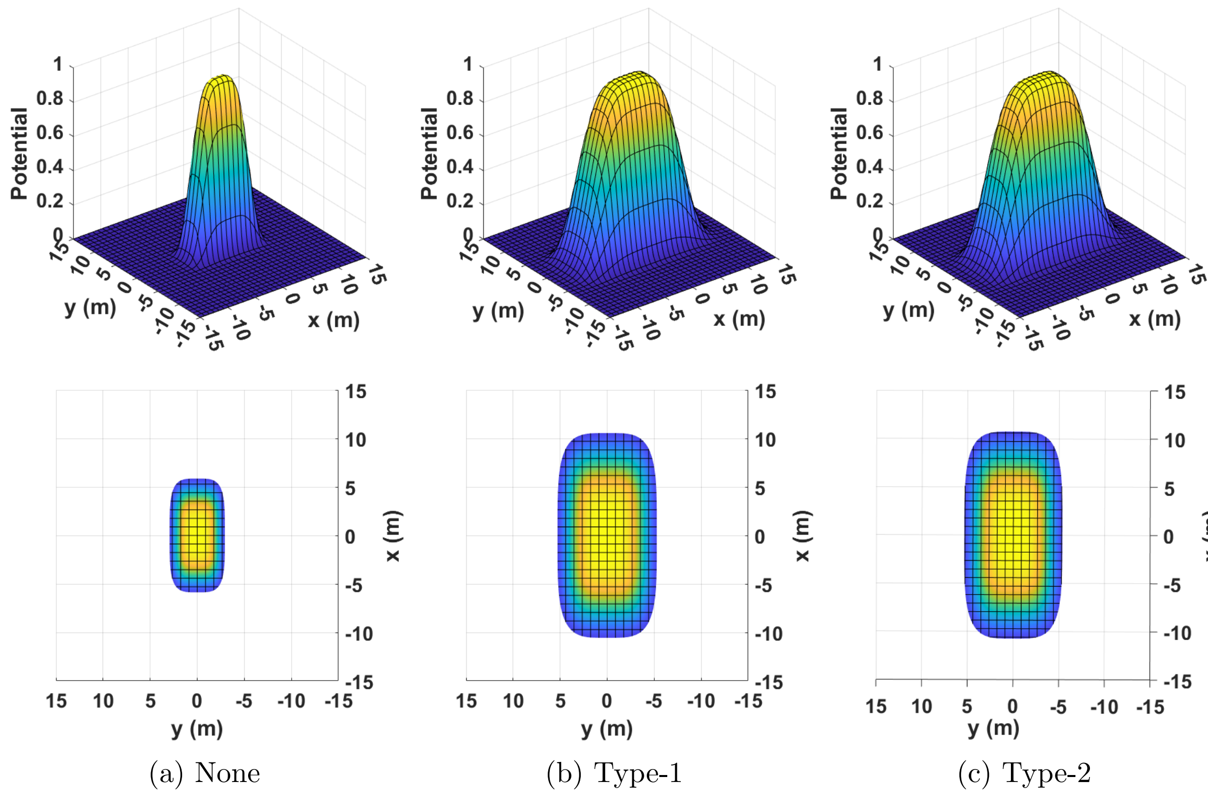


Figure 5.19: Type-1 & Type-2 Estimation for Vehicle Potential Field, 10% Safety Contour

Chapter 6

Simulation Experiment Results

6.1 System Overview

The various subsystems of navigation are illustrated in Figure 6.1, which presents a comprehensive block diagram of the entire system. This diagram also summarizes the simulation framework, where a driving scenario is defined and its parameters are applied to the simulator. Additionally, the navigation and control of the ego-vehicle are recorded and displayed, enabling a thorough analysis of the results.

6.2 Estimation Process Performance

Various comparison metrics were observed to measure the efficiency and potential of each estimation process. These metrics will help determine if each estimation process is suitable for real-time navigation and if any added computational complexity demonstrates a higher ability for nuance or performance.

6.2.1 Computation Time

Because the type-2 fuzzy estimation process has the added complexity of type-reduction to obtain a crisp output, there is additional computational overhead over type-1 estimation. Figure 6.2 demonstrates the computation time for each fuzzy estimation process to quantify the additional overhead.

The execution time for type-1 and type-2 using the EKM type-reduction method was meticulously compared by generating a random input vector to obtain its crisp output 50,000 times. Each iteration was timed and stored to obtain a mean and standard deviation of the computation for each SVO estimation process.

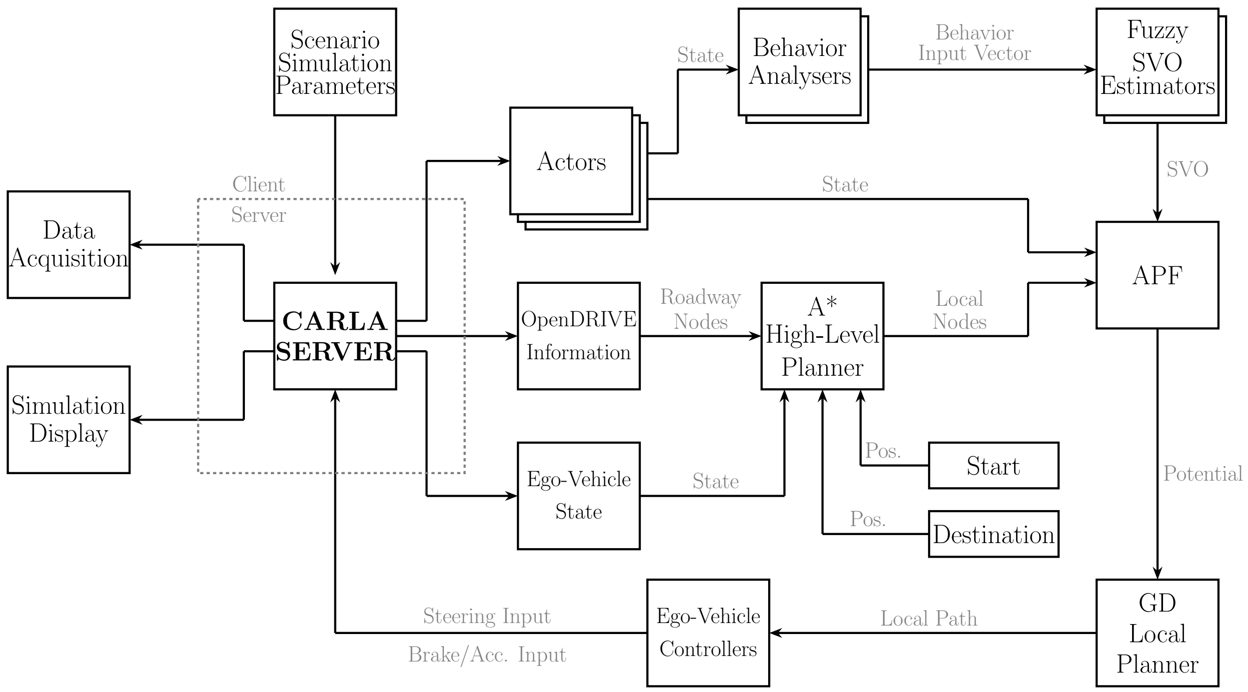
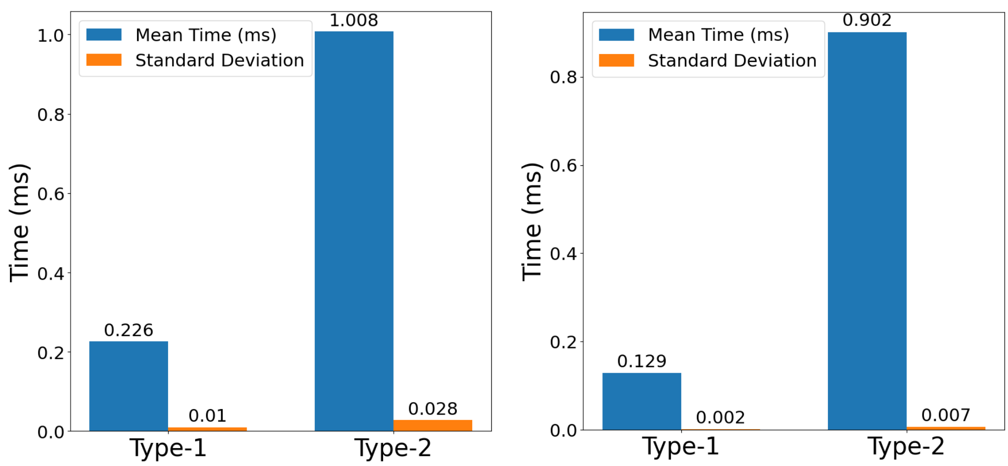


Figure 6.1: System Block Diagram



(a) Pedestrian Estimator (b) Vehicle Estimator

Figure 6.2: Fuzzy Estimator Computation Time Comparison

Figure 6.2 demonstrates that the type-1 fuzzy estimation is 84.1% and 75.5% faster than the type-2 estimation for pedestrians and vehicles, respectively. However, while type-2 estimation is significantly slower than type-1, each estimation requires only 0.833 ms and 1.02 ms to complete, which does not pose a significant burden to the real-time response of the estimation framework. Conversely, the processing time is additive. Each actor in a local scene must have their SVO computed, and the total execution time is the sum of all computations. This can be mitigated by intelligently computing SVO on a distance basis or any other heuristic to reduce the amount of computations. Furthermore, a scheduling method that assesses the freshness of an actor's SVO before committing resources to updating it can also be used.

6.2.2 Estimation Range

Because the footprints of uncertainty (FOUs) for type-2 membership functions cover a larger span of each input's domain, it is hypothesized that the type-2 estimators should have a greater output range and more nuance in their crisp output

When measuring the input vector for both pedestrians and vehicles, their behavior is measured in real-time. As such, an actor's social value continuously varies as the duration and quantity of each exhibited behavior change. For each linguistic variable, we can observe the system response over time when the input vector changes in Figures 6.3 and 6.4.

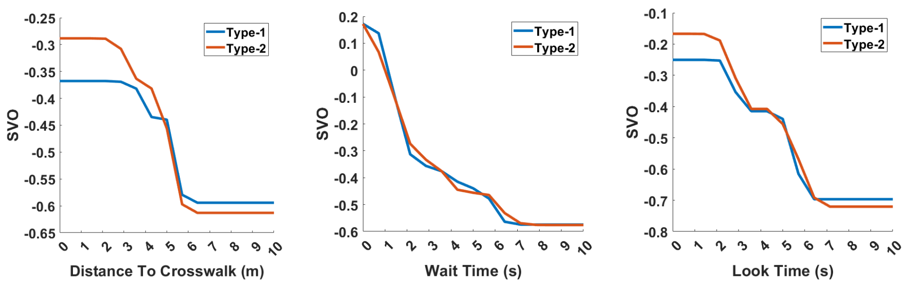


Figure 6.3: Pedestrian Estimator Crisp Output Range

Observing Figures 6.3 and 6.4, we note that there is a significant increase in the output range for type-2 systems. Furthermore, every input with the exception of *Lane Changes* changes earlier in the measurement, as noted by the lag in the initial change point for type-2 systems. This results in type-2 systems responding more rapidly to initial changes in an actor's behavior. Type-2 systems show a clear improvement in range, nuance, and responsiveness over type-1 systems for the same ruleset. Table 6.1 summarizes the initial response time difference and difference in dynamic range for each input variable between type-1 and type-2 systems. Negative response deltas represent a system that responds to less input, and a greater dynamic range delta indicates a wider range of possible SVO values and, thus, greater nuance.

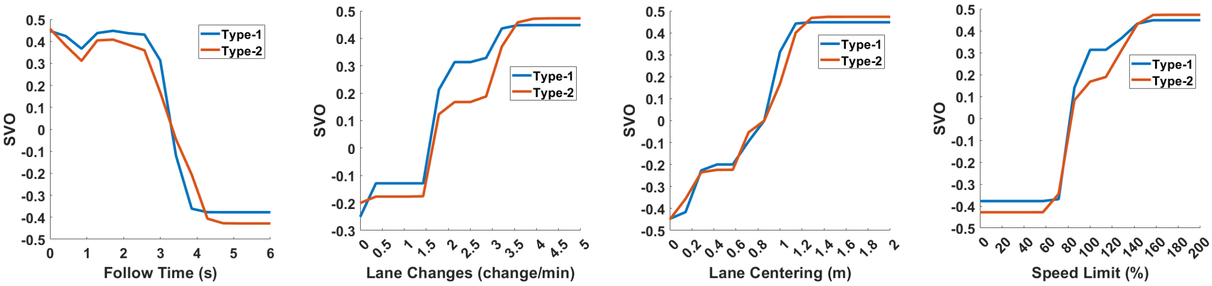


Figure 6.4: Vehicle Estimator Crisp Output Range

Table 6.1: Type-1 vs Type-2 Estimator Performance

Input	Initial Response Type-1	Initial Response Type-2	Delta	Range Type-1 (Ω)	Range Type-2 (Ω)	Delta
Dist. to Cross.	6.43m	6.43m	**0m**	0.22	0.32	**0.1**
Wait Time	0s	0s	**0s**	0.74	0.74	**0**
Look Time	2.14s	1.42s	**-0.72s**	0.44	0.55	**0.11**
Follow Time	4.28s	4.71s	**-0.43s**	0.82	0.89	**0.07**
Lane Changes	0 chg/min	0 chg/min	**0 chg/min**	0.69	0.67	**0.02**
Lane Centering	0m	0m	**0m**	0.89	0.92	**0.03**
Speed Limit	71.42%	57.13%	**-15.29%**	0.82	0.90	**0.08**

Table 6.1 indicates that the type-2 estimator is better or equal on both measured performance indicators.

6.3 Behavioral Simulation Parameters

6.3.1 Parameters and Definitions

To create a uniform and repeatable simulation environment for social value estimation, standard behavioral profiles were created in CARLA Simulator. These behavioral profiles do not have an effect on the navigation path of the vehicle but rather on how their respective paths will be followed. 5 Standard profiles were created, but many more can be implemented through the flexible framework that was designed.

Vehicles

The traffic controller discussed in Chapter 2 determines how CARLA will control vehicle inputs for vehicle actors. A wrapper around this framework was created to easily and effectively create repeatable behavioral patterns for vehicle actors and apply user-defined parameters that dictate *how* an actor will follow their planned path. The parameters affecting vehicle actor control and their descriptions are summarized in Table 6.2. Furthermore, the values that were set for each behavioral profile are defined in Table 6.3.

Table 6.2: Vehicle Behavior Parameter Descriptions

Parameters	**Description**
Speed Limit Distance	Value is the percentage of a speed limit that defines how far the vehicle's target speed will be from the current speed limit.
Speed Decrease	How quickly in km/h the vehicle will slow down when approaching a slower vehicle ahead.
Safety Time	Time-to-Collision: Time it will take for the vehicle to collide with a proceeding object that suddenly stops.
Min. Proximity	The minimum distance in meters from another vehicle or pedestrian before the vehicle performs a maneuver such as avoidance, or tailgating.
E-Stop Distance	The distance from an object at which the vehicle will perform an emergency stop.
Tailgate Count	A counter to avoid tailgating too quickly after the last tailgate.

Table 6.3: Vehicle Behavior Parameter Values

Parameters	**Altruistic**	**Cooperative**	**Individualistic**	**Competitive**	**Sadistic**
Speed Limit Distance (%)	-60	-30	0	30	60
Speed Decrease (km/h)	14	12	8	4	1
Safety Time (s)	5	3	2	1	0.25
Min. Proximity (m)	12	10	8	4	0
E-Stop Distance (m)	8	6	4	2	0
Tailgate Count	0	0	-1	-2	$-\infty$

Pedestrians

As with vehicle actors, a wrapper for pedestrian actor controllers was also created. Because pedestrian controls, paths, and desired simulated behaviors differ vastly from those of vehicles, a completely different set of parameters is used to modify the pedestrian manager. These parameters and their descriptions are summarized in Table 6.4 and the values set for each behavioral profile are defined in Table 6.5.

Table 6.4: Pedestrian Behavior Parameter Descriptions

Parameters	**Description**
Walking Speed	Steady-state speed at which the pedestrian travels
Speed Change	Percent speed change around crosswalks
Wait Time	Time a pedestrian waits at a crosswalk before crossing
Look Time	Time spent looking at traffic before crossing
Effect Distance	Distance from the crosswalk at which pedestrians will begin to exhibit behaviors
Safe to Cross Dist.	Pedestrian will begin to cross if all vehicles are *further* than this distance

Table 6.5: Pedestrian Behavior Parameter Values

Parameters	**Altruistic**	**Cooperative**	**Individualistic**	**Competitive**	**Sadistic**
Walking Speed (km/h)	2	4	5	6	7
Speed Change (%)	60	80	100	120	140
Wait Time (s)	4	2	1	0.5	0
Look Time (s)	4	2	1	0.5	0
Effect Distance (m)	5	4	3	2	1
Safe to Cross Dist. (m)	20	15	10	5	0

6.3.2 Observation, Measurement, and Quantification

The ego vehicle observes each behavior directly from actors and is unknown before they exhibit that action. Table 6.6 summarize the method in which each input behavior for the fuzzy estimation process is measured and quantified. A *peak* measurement is the maximum value observed over time window t_w, an *average* measurement is the average value observed over t_w, and *live* is the measured in-progress value. The choice of which method is used is based on whatever is most representative of that behavior. For example, someone's respect for the speed limit might not be correctly observable in heavy traffic if it is not possible to exceed the traffic flow. Thus, observing the peak value over t_w would be more representative of someone's *willingness* to speed.

Table 6.6: Actor Behavior Quantification

Input	**Quantification**
Speed Limit (%)	Peak
Lane Centering (m)	Live
Follow Time (s)	Peak
Lane Changes ($\frac{chg}{min}$)	Peak
Dist. to Crosswalk (m)	Live
Time Waiting (s)	Live
Time Looking (s)	Live

6.4 Simulation Results

6.4.1 Merging Vehicle Avoidance

Overview

The simulation event involves the ego-vehicle being maliciously *cut-off* by a vehicle in the adjacent lane, where the vehicle begins performing the merging maneuver before they've cleared the ego-vehicle. This malicious behavior is indicative of a sadistic profile, such that the driver is not only trying to get ahead of the ego vehicle in a competitive manner but is trying to inflict harm. Similar situations can present themselves in the real world during road rage incidents. Figure 6.5 represents the intended vehicle path during the scenario, and Figure 6.6 demonstrates the progression of the event and the corresponding artificial potential field in a film-strip style representation.

Simulation

The road speed limit is set to 20km/h and the actors are left to reach their steady-state velocity for the respective profile. The profile chosen for the merging vehicle is *sadistic* and its associated parameters are summarized in Table 6.3. Prior To the event, the merging vehicle's observed behavior was also left to exhibit itself and reach a steady state.

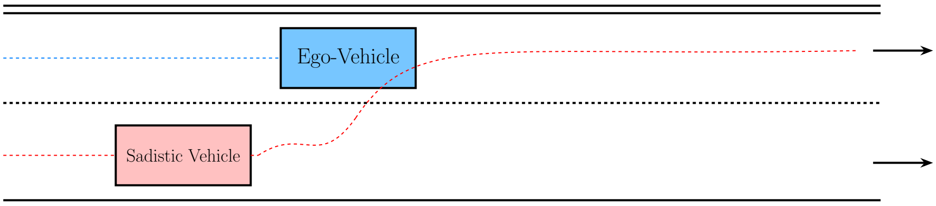


Figure 6.5: Merge Emergency Avoidance

Figure 6.6: Merge Scenario & Potential Field Progression

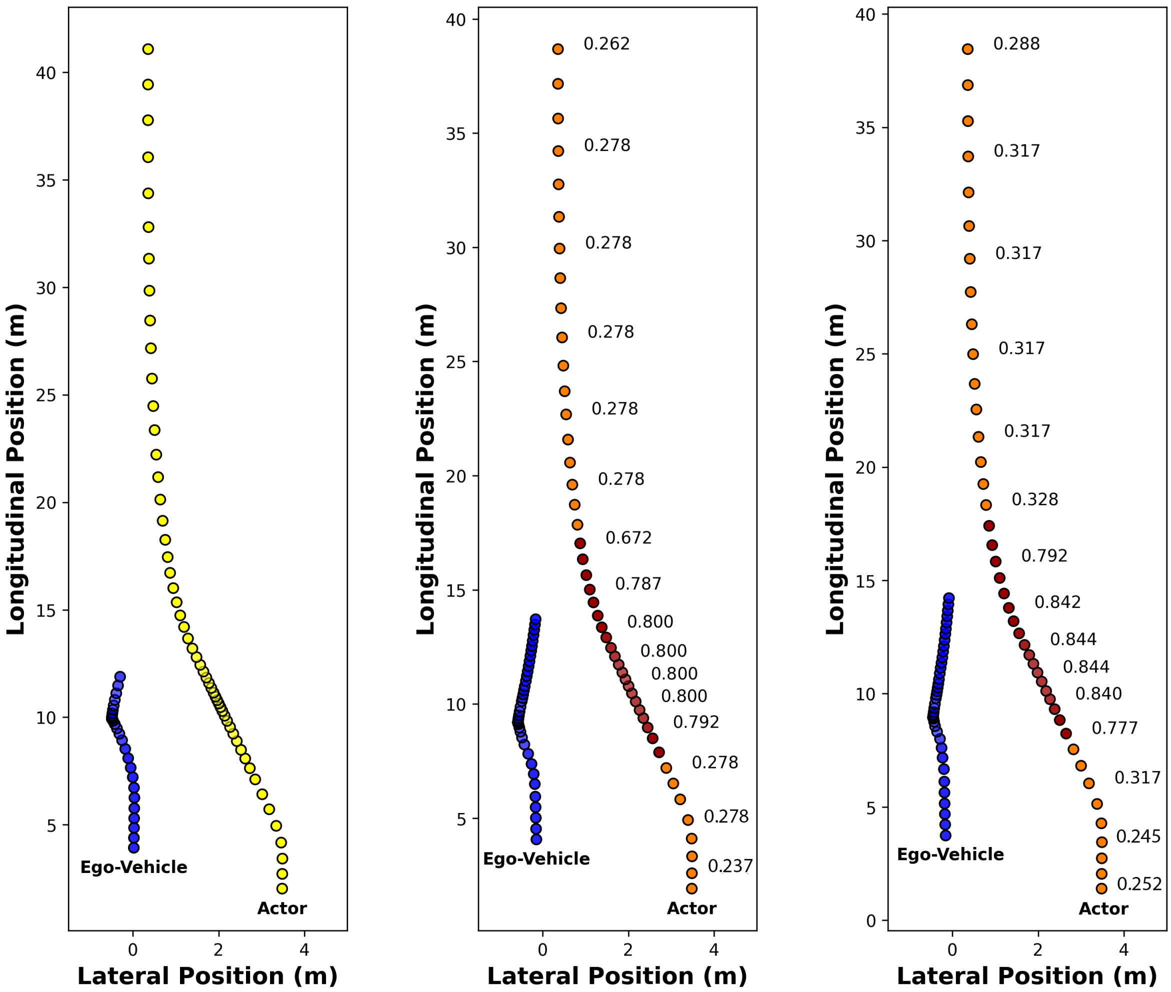
Table 6.7 shows the RMS, weighted RMS (WRMS), and peak longitudinal and lateral accelerations from the emergency merge situation for each type of SVO estimation process. The RMS and WRMS values are calculated as follows:

$$a_{RMS} = \sqrt{\frac{1}{T}\int a_w^2 dt} = \sqrt{\frac{1}{n}(a_1^2 + a_2^2 + ... + a_n^2)} \tag{6.1}$$

$$RMS_{Total} = \sqrt{RMS_x^2 + RMS_y^2 + RMS_z^2} \tag{6.2}$$

Table 6.7: Emergency Merge Acceleration Metrics

Estimation Type	**RMS** (m/s^2)	**WRMS** (m/s^2)	**Peak Long.** (m/s^2)	**Peak Lat.** (m/s^2)
None	0.291	0.407	14.536	3.816
Type-1	0.135	0.190	8.054	1.419
Type-2	0.123	0.172	7.819	1.059



(a) No Estimation ($\Omega = 0$) (b) Type-1 Estimation (c) Type-2 Estimation

Figure 6.7: Emergency Merge Trajectory and SVO Progression

Table 6.8: Emergency Merge Position Metrics

Estimation Type	**Closest Dist.** (m)	**Total Dist.** (m)
None	2.580	8.01
Type-1	2.91	9.70
Type-2	2.88	10.51

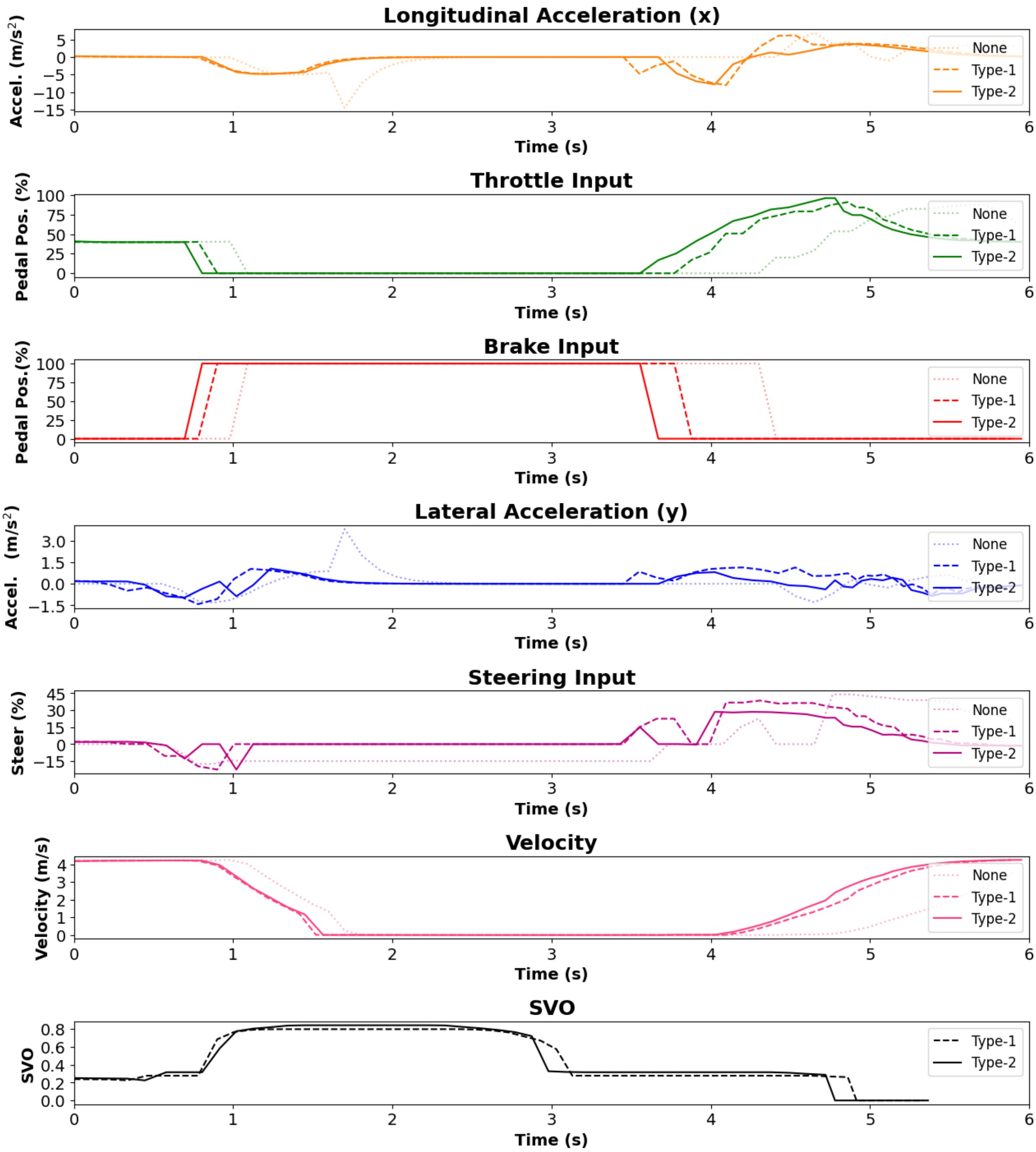


Figure 6.8: Emergency Merge Accelerations and Vehicle Inputs

$$WRMS = \sqrt{K_x RMS_x^2 + K_y RMS_y^2 + K_z RMS_z^2} \tag{6.3}$$

Where the weighted RMS coefficients K_x, K_y, K_z are 1.4, 1.4, and 1, respectively, and represent the human's sensitivity to accelerations in those directions. [65]

Discussion

From Figure 6.7, we can observe that SVO estimation provided a stronger, and earlier response to the maliciously merging vehicle. This is also corroborated in Figure 6.8 which shows that both type-2 and type-1 fuzzy estimation responded significantly earlier than no estimation shown in the earlier change in brake and throttle applications, as well as an earlier steering response. This trade-off resulted in lower RMS and WRMS values shown in Table 6.7 and significantly smoother vehicle inputs during the initial response to the merging vehicle, as well as quicker and smoother re-acceleration once the ego-vehicle has been passed.

Comparing type-1 and type-2 estimation in Figure 6.8, we notice that type-2 estimation had a more gradual response than type-1 estimation. This resulted in the ego-vehicle applying less initial steering input than type-1 and instead relying on an earlier braking input to avoid the merging car. This resulted in less lateral movement into the adjacent lane and a more direct path. The total distance traveled by the type-2 estimator was 8.35 % further than the type-1 estimation during the simulated scenario (9.70 m vs. 10.51 m) at the expense of 1.03% less safety margin (2.91 m vs. 2.88 m). The path efficiency and safety metrics are summarized in Table 6.8. The safety distance was measured as the closest distance between vehicle centroids during the scenario.

6.4.2 Immobile Vehicle Avoidance

Overview

The simulation event involves a parked vehicle blocking the ego vehicle's lane. There are two incoming vehicles in the opposite lane: a leading altruistic profile vehicle and a following individualistic vehicle. The ego-vehicle can either perform a lane change maneuver or wait for the two vehicles to pass, then perform a lane change. The correct estimation of the altruistic vehicle's SVO will allow the ego vehicle to aggressively perform a lane change instead of being paralyzed while waiting for both to pass. Correctly assessing the vehicle's SVO will also allow the total time lost by all vehicles to be minimized. Figure 6.9 illustrates the actor positions and intended ego-vehicle path. Figure 6.10 demonstrates the progression of the event and the corresponding artificial potential field in a film-strip style representation.

Simulation

The road speed limit is set to 20 km/h, and the actors are left to reach their steady-state velocity for the respective profile. The profile chosen for the right-lane leading vehicle is *altruistic*, and the following vehicle is set to *individualistic*. Its associated parameters are summarized in Table 6.3. The actors observed behavior was also left to exhibit itself and reach a steady state.

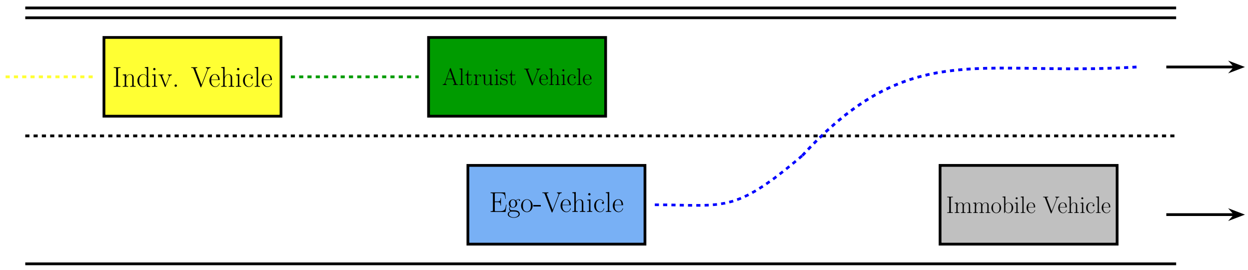


Figure 6.9: Merge Emergency Avoidance

Table 6.9: Immobile Vehicle Scenario Position Metrics

Estimation Type	**Closest Dist.** (m)	**Total Dist.** (m)
None	2.545	20.089
Type-1	3.116	39.565
Type-2	3.098	40.323

Table 6.10: Immobile Vehicle Scenario Acceleration Metrics

Estimation Type	**RMS** (m/s^2)	**WRMS** (m/s^2)	**Peak Long.** (m/s^2)	**Peak Lat.** (m/s^2)
None	0.193	0.270	4.181	0.688
Type-1	0.060	0.083	0.193	0.944
Type-2	0.053	0.074	0.196	1.124

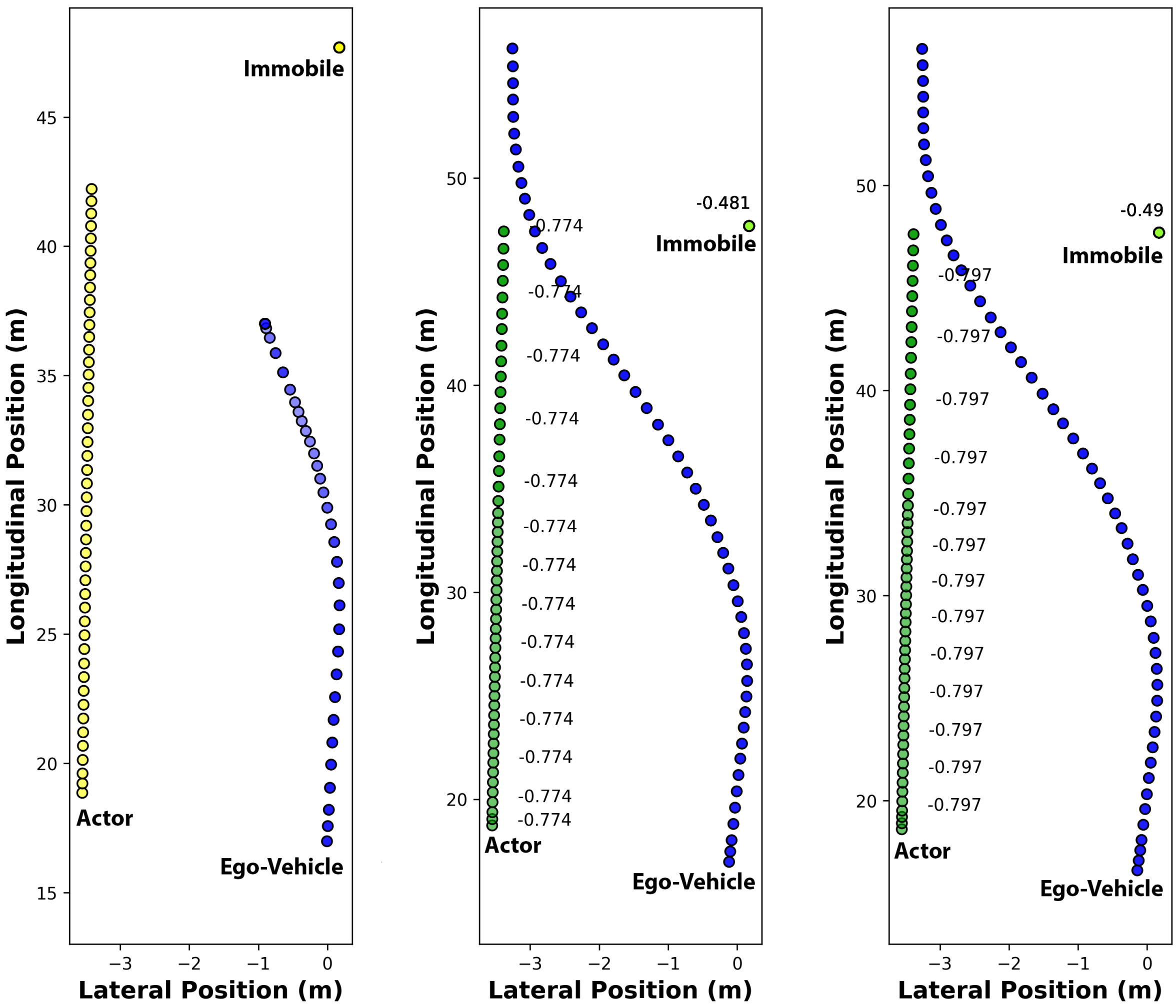
Discussion

The use of SVO had very observable outcomes in this scenario. The lack of SVO estimation resulted in significant hesitation by the ego-vehicle as seen in the velocity, steering, and braking profiles shown in Figure 6.12 as well as the actor positions in Figure 6.11. Without SVO, the ego-vehicle overestimated the risk posed by the *altruistic* actor in the right lane and over-estimated the risk posed by the immobile vehicle blocking its path. This initially resulted in hesitation when trying to pass the *altruistic* actor, which caused the ego-vehicle to overshoot its intended path and come closer to the immobile vehicle. Because the immobile vehicle's risk was also overestimated, the ego vehicle could not continue its path and was indefinitely paralyzed. Arguably, this paralysis could be overcome with a forward-reverse path planner but could result in additional distress and frustration to passengers.

(a) Ego-Vehicle Lane Change and Overtake

(b) Ego-Vehicle Hesitation and Paralysis

Figure 6.10: Immobile Vehicle Scenario & Potential Field Progression



(a) No Estimation ($\Omega = 0$) (b) Type-1 Estimation (c) Type-2 Estimation

Figure 6.11: Immobile Vehicle Scenario Trajectory and SVO Progression

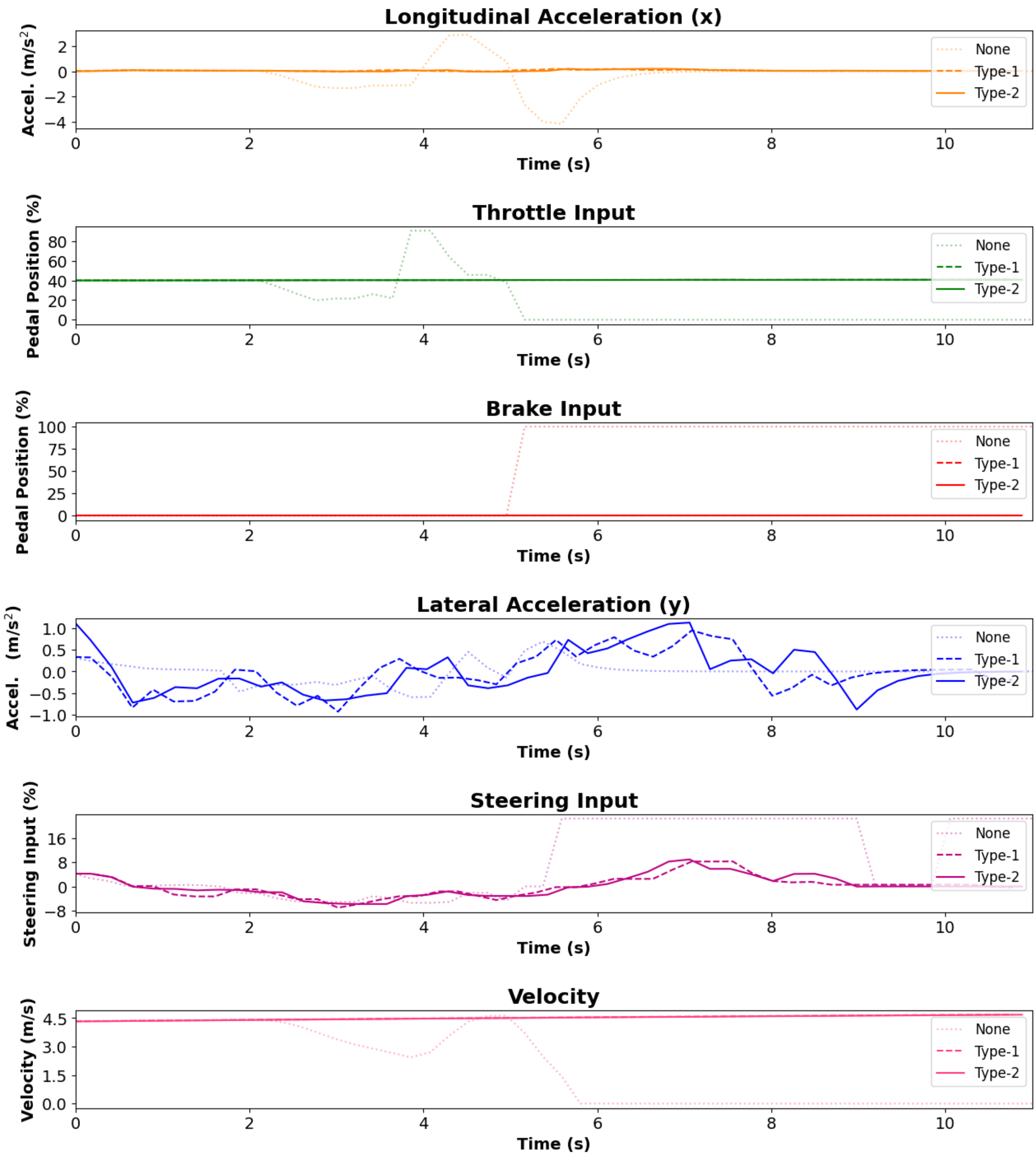


Figure 6.12: Immobile Vehicle Scenario Accelerations and Vehicle Inputs

In this scenario, the difference between type-1 and type-2 estimation methods was moderate but observable. However, due to the type-2 estimator attributing lower SVOs to the *altruistic* actor and immobile vehicle, the ego-vehicle was able to take a more direct path and cover a larger distance during the simulated event. The SVO estimation was 0.023, or 2.93% more *altruistic* for the vehicle in the right lane, and 0.009 or 1.85% more *altruistic* for the immobile vehicle. Table 6.10 summarizes the acceleration metrics of the scenario and Table 6.9 summarizes the position and distance metrics. Similar acceleration performance was obtained between type-1 and type-2 estimation but type-2 estimation covered 1.87% greater distance (39.565m vs. 40.323m) while having 12.38% better WRMS performance (0.060m/s^2 vs 0.053m/s^2).

6.4.3 Pedestrian Crossing

Overview

Pedestrian-vehicle interactions are an everyday occurrence in urban environments. In crosswalk scenarios, an unspoken contract is formed between a pedestrian and a vehicle, where a pedestrian non-verbally shows their intention to cross the road. Likewise, the driver must observe this pedestrian and determine the likelihood of their intention to cross.

The scenario created depicts a priority crosswalk situation, where a pedestrian has priority to cross, and a vehicle must yield if it is safe. Thus, the social contract between a competitive or altruistic pedestrian and a vehicle presents much more nuance than a signaled or non-priority crosswalk. Figure 6.13 illustrates the pedestrian-vehicle interaction, and Figure 6.14 depicts the progression of the APF during the scenario.

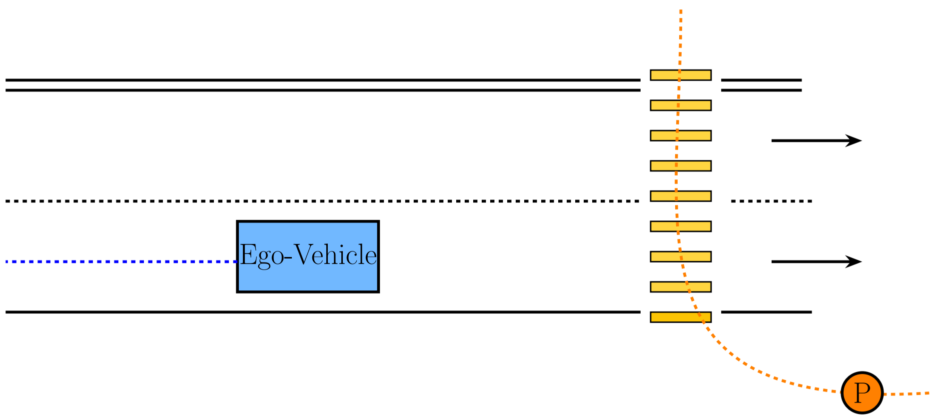


Figure 6.13: Pedestrian Crossing Scenario

Simulation

The road speed limit is set to $20km/h$ and the actors are left to reach their steady-state operation for their respective behavioral profiles. The crossing pedestrian was given a *competitive* profile, where the associated parameters are summarized in Table 6.4.

Figure 6.14: Pedestrian Crossing Scenario & Potential Field Progression

Table 6.11: Pedestrian Crossing Position Metrics

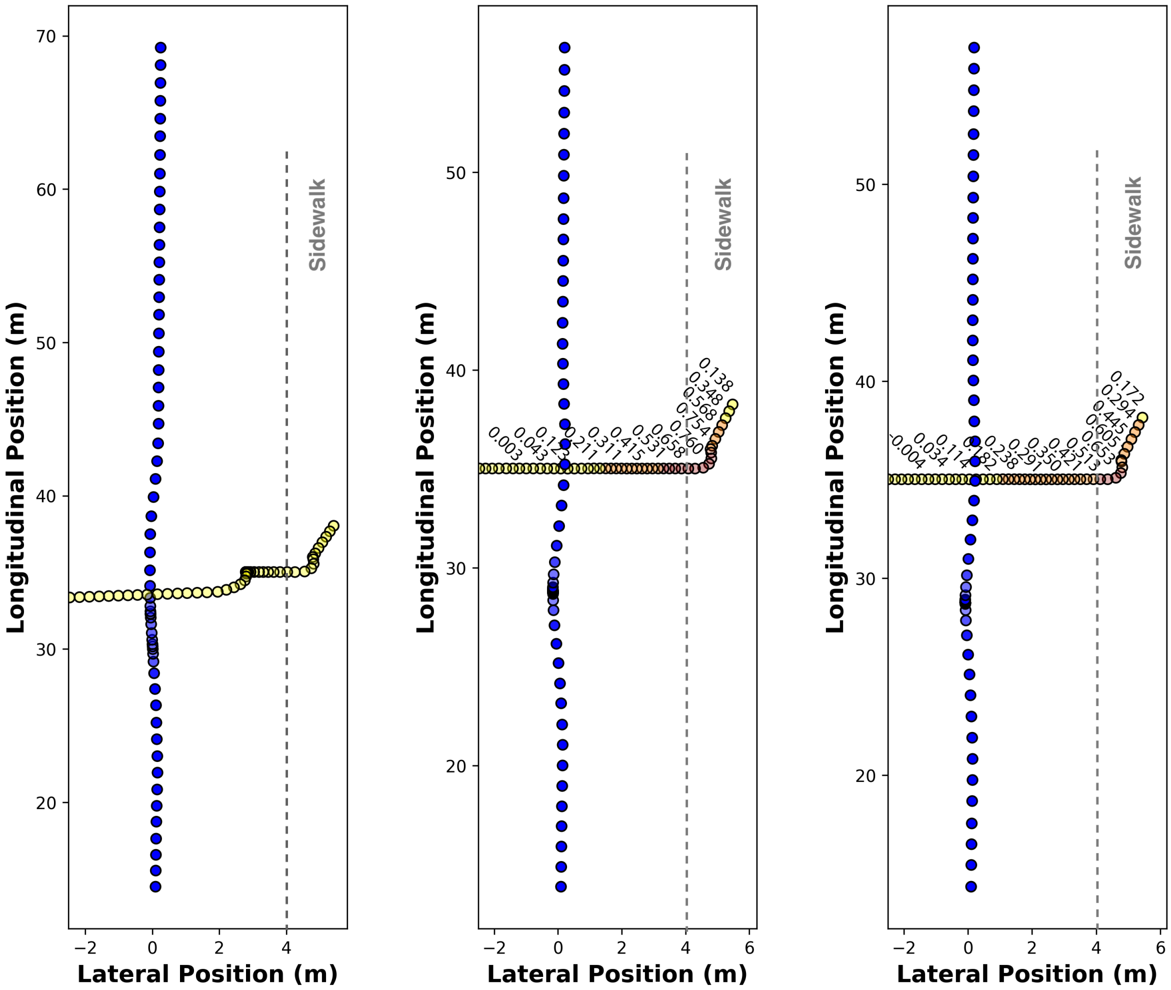
Estimation Type	**Closest Dist.** (m)	**Total Dist.** (m)
None	3.050	74.746*
Type-1	4.790	42.050
Type-2	4.810	42.620

Table 6.12: Pedestrian Crossing Scenario Acceleration Metrics

Estimation Type	**RMS** (m/s^2)	**WRMS** (m/s^2)	**Peak Long.** (m/s^2)	**Peak Lat.** (m/s^2)
None	0.364	0.510	20.657	0.502
Type-1	0.814	1.139	8.734	0.718
Type-2	0.285	0.399	8.376	0.489

Discussion

As with previous scenarios, significant improvements are made by the use of SVO estimation. In the case of no estimation, the ego-vehicle did not successfully stop for the pedestrian. While an attempt at stopping was made, as seen by the velocity profile in Figure 6.12 and in the actor positions in Figure 6.15, strong hesitation caused excessive longitudinal accelerations and eventual



(a) No Estimation ($\Omega = 0$) (b) Type-1 Estimation (c) Type-2 Estimation

Figure 6.15: Pedestrian Crossing Scenario Trajectory and SVO Progression

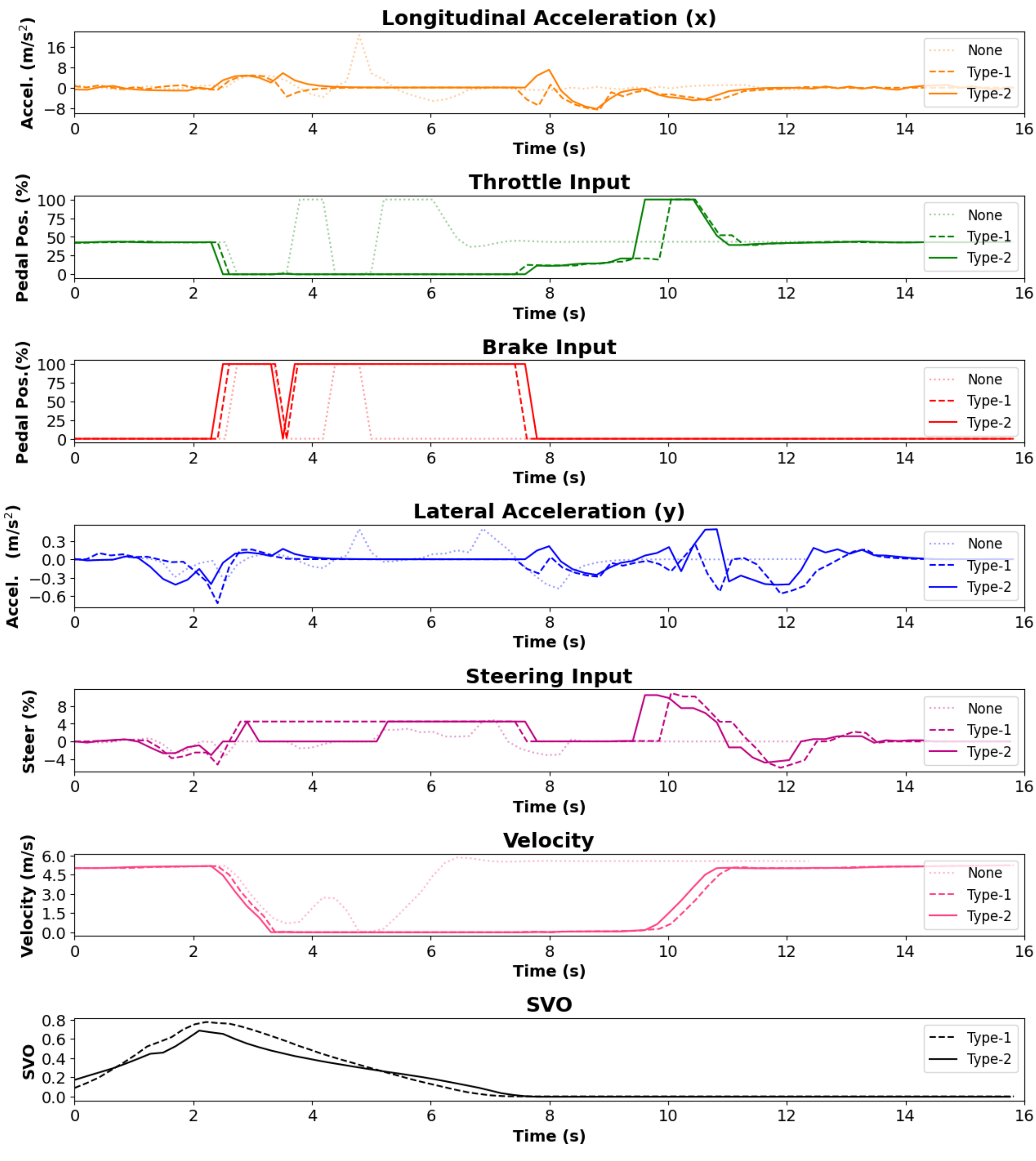


Figure 6.16: Pedestrian Crossing Scenario Accelerations and Vehicle Inputs

abandonment of the braking maneuver. When observing the distance metrics summarized in Table 6.11*, no estimation managed to drive significantly further during the simulation window because it did not wait for the pedestrian to cross. Figure 6.15 a) shows that the ego-vehicle significantly overshot the crosswalk, causing the pedestrian to stop and divert behind the ego-vehicle. This is an unacceptable outcome as the ego-vehicle did not yield for the pedestrian, nor did it continue, and instead hesitated significantly. Figure 6.17 depicts the hesitant pedestrian-vehicle interaction.

Figure 6.17: Pedestrian Crossing Scenario - No SVO Hesitation

The results were similar when using type-1 and type-2 estimation. However, because the type-2 estimator responded marginally stronger before the pedestrian started to cross, the vehicle began braking slightly earlier, as seen in Figure 6.16. While the type-1 estimator had a stronger overall response, it did not impact braking performance as the ego vehicle was already in a full-force braking maneuver. Therefore, it is noted that an earlier response was more effective than a stronger response. Furthermore, the type-2 estimator had slightly smoother steering inputs because the increase and decrease in the SVO when the pedestrian finished crossing was more gradual. Finally, the type-2 estimator permitted the ego-vehicle to begin re-accelerating sooner because the smaller SVO decreased the potential of the pedestrian once they had crossed sooner. Table 6.12 summarizes the acceleration metrics of the scenario and Table 6.11. Type-2 estimation provided 96.23% better WRMS performance (1.139m/s^2 vs. 0.399m/s^2 as well as 4.18% and 37.95% less peak longitudinal and lateral accelerations respectively (8.734m/s^2 vs. 8.376m/s^2 and 0.718m/s^2 vs. 0.489m/s^2). Furthermore, type-2 estimation achieved a 0.42% greater safety distance (4.79 m vs. 4.81 m) while covering a 1.35% greater distance during the scenario (42.62 m vs. 42.05 m).

Chapter 7

Conclusions and Future Research

7.1 Conclusion

The use of social value orientation with a gradient descent path planner showed significant improvements over classical gradient descent methods. Results showed that the use of a fuzzy estimation process significantly improved path planning by reducing hesitations, smoother control, increased safety margins for dangerous drivers, and improved traffic flow with those deemed cooperative.

While the largest leap forward was made through the introduction of fuzzy estimation of SVO over none, incremental improvements were shown when comparing type-2 fuzzy estimation to type-1. The system handles uncertainties in sensor data and environmental variations by leveraging the robustness of type-2 fuzzy logic, thus improving decision-making in complex scenarios. Simulation results demonstrate that the model can optimize path planning by predicting and adapting to potential hazards with greater accuracy, significantly reducing the risk of accidents and enhancing traffic flow.

It is important to note that type-2 fuzzy systems introduce significant computational complexity over type-1 systems. Because SVO estimation is a real-time process and must be computed for each actor per time-step, large computational overhead can occur in high-density traffic situations which could affect real-time decision-making. Possible alternatives include limiting the process to actors in the immediate proximity of the ego-vehicle, scheduling SVO estimation with a round-robin or priority-queue scheduler using each actor's SVO *freshness* or proximity as a priority metric. Furthermore, it could also be permissible to fall back to type-1 estimation as a *good enough* process during high traffic. A combination of all suggestions can be incorporated to create a computationally efficient and safe estimation process. OEM vehicle manufacturers usually rely on robust, lower-end processors as a cost-saving measure when producing vehicles at scale, and could exacerbate computation issues if these proposals are not incorporated.

In conclusion, this research advances the technical specifications of autonomous navigation systems and addresses the critical aspect of social acceptance by ensuring that AV operations align with human values and safety norms.

7.2 Future Research

This work created a robust artificial potential field navigation framework incorporating SVO as a gain parameter for each actor's potential. However, a shortcoming of this work is that the fuzzy estimation process used a common-sense approach to determine what behavioral parameters and their membership functions to estimate an actor's SVO.

Because this work was implemented in the CARLA simulator and a flexible custom actor behavior manager was implemented, it would be of great value to create many behavioral profiles and obtain participants to drive in a full vehicle simulator inside a CARLA environment. While driving, participants can either verbally communicate their perceived cooperativeness of a vehicle-vehicle or vehicle-pedestrian interaction or rely on the use of electroencephalograms (EEG) to automatically capture their sentiment of a simulated interaction. This data can then be referenced to that actor's behavioral profile parameters and used as input to a neuro-fuzzy tuning process. This can present an opportunity for cross-disciplinary research with medical and behavioral science academics to further expand this field of research. If a very significant dataset can be created, it could be possible to forego fuzzy estimation and implement a deep neural network framework. However, this would require industry partners and significant investment to achieve and would omit the many advantages fuzzy estimation presents for industry regulators.

Additionally, different path planning algorithms, such as `A*`, can be utilized to validate whether SVO's advantages also bring the same improvements as gradient descent. Documenting and contrasting SVO with different path planning protocols can increase its likelihood of adoption by the automotive industry or its likelihood of being applied to non-vehicular or other non-related robotic industries.

Bibliography

[1] Stewart T. *Overview of motor vehicle traffic crashes in 2021*. Tech. rep. National Highway Transportation Safety Board, 2023.

[2] Insurance Information Institute. *Driving Behaviors Reported For Drivers And Motorcycle Operators Involved In Fatal Crashes*. 2023. URL: https://www.iii.org/fact-statistic/facts-statistics-aggressive-driving.

[3] SAE. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016 202104*. Tech. rep. SAE International, 2021.

[4] Mercedes-Benz. *Certification for SAE Level 3 system for U.S. market*. 2023. URL: https://group.mercedes-benz.com/innovation/product-innovation/autonomous-driving/drive-pilot-nevada.html.

[5] Yuxuan He, Pingping Wang, and Jing Zhang. "A Comparison Between A* & RRT in Maze Solving Problem". In: *2021 3rd International Symposium on Robotics & Intelligent Manufacturing Technology (ISRIMT)*. 2021, pp. 333–338. DOI: 10.1109/ISRIMT53730.2021.9596830.

[6] Yinxin Long and Huajin He. "Robot path planning based on deep reinforcement learning". In: *2020 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS)*. 2020, pp. 151–154. DOI: 10.1109/TOCS50858.2020.9339752.

[7] Oussama Khatib. "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots". In: *The International Journal of Robotics Research* (1986).

[8] Savvas G. Loizou. "Closed form Navigation Functions based on harmonic potentials". In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*. 2011, pp. 6361–6366. DOI: 10.1109/CDC.2011.6161438.

[9] Gregory Antonovsky. *Bidirectional RRT* FND algorithm*. 2018. URL: https://github.com/onlinex/bidirectionalRRTStarFND.

[10] Hongwei Qin et al. "Review of Autonomous Path Planning Algorithms for Mobile Robots". In: *Drones* 7 (Mar. 2023), p. 211.

[11] Fahad Anwar. *Gradient Descent — Introduction and Implementation in Python*. 2019. URL: https://machinelearningmind.com.

[12] James W. Griesinger Donald W. Livingston. "Toward a model of interpersonal motivation in experimental games". In: *Behavioral Science, 18(3)* (1973), pp. 173–188.

[13] Urs Fehr Ernst; Fischbacher. "The Nature of Human Altruism". In: *Nature 425* (2003).

[14] Eric van Dijk. *Social Dilemmas, Psychology of*. Ed. by James D. Wright. Second Edition. Oxford: Elsevier, 2015, pp. 245–249. ISBN: 978-0-08-097087-5.

[15] Keita Honjo; takahiro Kubo. "Social Dilemmas in nature-Based tourism Depend on Social Value orientations". In: *Scientific Reports 10(1) 3730* (2020).

[16] L.A. Zadeh. "The Concept of a Linguistic Variable and Its Application to Approximate Reasoning". In: *Information Sciences, vol. 8, pp. 199–249* (1975).

[17] Christopher I. Connolly; Roderic A. Grupen. "The applications of harmonic functions to robotics". In: *Journal of Field Robotics (Vol. 10-7 p. 931-946)* (1993).

[18] Robert Daily and David M. Bevly. "Harmonic potential field path planning for high speed vehicles". In: *2008 American Control Conference*. 2008, pp. 4609–4614. DOI: 10.1109/ACC.2008.4587222.

[19] Panagiotis V; Constantinos V; Charalampos P. B.; Kostas J. K. "Orientation-Aware Motion Planning in Complex Workspaces using Adaptive Harmonic Potential Fields". In: *2019 International Conference on Robotics and Automation (ICRA) (pp. 8592-8598)* (2019).

[20] Ahmad A. Masoud. "A harmonic potential field approach for joint planning and control of a rigid separable nonholonomic mobile robot". In: *Robotics and Autonomous Systems* 61.6 (2013), pp. 593–615.

[21] Park J.W. Kwak H.J. Kang Y. and Dong W. K. "Advanced Fuzzy Potential Field Method for Mobile Robot Obstacle Avoidance". In: *Computational Intelligence and Neuroscience, vol. 2016* (2016).

[22] Kavraki L. E.; Svestka P.; Latombe J.-C.; Overmars M. H. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE Transactions on Robotics and Automation 12(4) p. 566–580* (1996).

[23] Lijun Qiao, Xiao Luo, and Qingsheng Luo. "An Optimized Probabilistic Roadmap Algorithm for Path Planning of Mobile Robots in Complex Environments with Narrow Channels". In: *Sensors* 22.22 (2022). ISSN: 1424-8220.

[24] Jana Kosecka. *Potential Field Methods*. 2018. URL: https://cs.gmu.edu/~kosecka/cs685/cs685-potential-fields.pdf.

[25] Stuart J. Russell. *Artificial intelligence a modern approach 4th ed.* Pearson, 2018.

[26] Jing J; Wang Y; Liu W; Liu T Wang H; Lou S. "An improved A* algorithm for path planning". In: *PLoS ONE 17(2)* (2022).

[27] J. Ouyang; D. Huang D. Xiang; H. Lin. “Combined improved A* and greedy algorithm for path planning of multi-objective mobile robot”. In: *Scientific Reports (Sci Rep 12, 13273)* (2022).

[28] Steven M. LaValle. *Rapidly-exploring random trees: A new tool for path planning*. Tech. rep. Iowa State University, 1998.

[29] E. Frazzoli S. Karaman. “Sampling-based Algorithms for Optimal Motion Planning”. In: *The International Journal of Robotics Research vol. 30(7) p.846-894* (2011).

[30] I. Fahad; N. Jauwairia; M. Usman; A. Yasar; H. Osman. “RRT*-Smart: Rapid convergence implementation of RRT* towards optimal solution”. In: *Proceedings of IEEE International Conference on Mechatronics and Automation (ICMA) p. 1651–1656* (2012).

[31] Wilko Schwartinga; Alyssa Piersona; Javier Alonso-Morab; Sertac Karamanc; Daniela Rusa. “Social Behavior for Autonomous Vehicles”. In: *PNAS* (2019).

[32] S. Deb; L.Strawderman; J. DuBien; B. Smith and D.W. Carruth; T.M. Garrison. “Evaluating pedestrian behavior at crosswalks: Validation of a pedestrian behavior questionnaire for the U.S. population”. In: *Accident Analysis and Prevention* 106 (2017), pp. 191–201. ISSN: 0001-4575.

[33] M. Lanzer; M. Gieselmann; K. Mühl; M. Baumann. “Assessing crossing and communication behavior of pedestrians at urban streets”. In: *Transportation Research Part F: Traffic Psychology and Behaviour* 80 (2021), pp. 341–358. ISSN: 1369-8478.

[34] N.N. Karnik; J.M. Mendel. “Centroid of a type-2 fuzzy set”. In: *Information Sciences 132 pp. 195–220* (2002).

[35] Vaibhav Saxena et al. “Type reduction techniques for two-dimensional interval type-2 fuzzy sets”. In: *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 2017, pp. 1–6.

[36] Li Wu et al. “An Improved Type-Reduction Algorithm for General Type-2 Fuzzy Sets”. In: *Inf. Sci.* 593.C (2022), pp. 99–120. ISSN: 0020-0255.

[37] D. Wu; J.M Mendel. “Enhanced Karnik–Mendel Algorithms”. In: *IEEE Transactions on Fuzzy Systems* 17.4 (2009), pp. 923–934.

[38] Chao Chen et al. “A Comprehensive Study of the Efficiency of Type-Reduction Algorithms”. In: *IEEE Transactions on Fuzzy Systems* 29.6 (2021), pp. 1556–1566.

[39] Jiří Pihrt; Petr Šimánek. “Spatiotemporal Prediction of Vehicle Movement Using Artificial Neural Networks”. In: *2022 IEEE Intelligent Vehicles Symposium (IV) p. 734-739* (2022).

[40] C. Ryan; F. Murphy; M. Mullins. “End-to-End Autonomous Driving Risk Analysis: A Behavioural Anomaly Detection Approach”. In: *IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 3, pp. 1650-1662* (2021).

[41] Alexey Dosovitskiy et al. *CARLA: An Open Urban Driving Simulator*. 2017.

[42] Intel German Ros. *CARLA and AVL Collaboration: Ushering in a New Era of Off-Road Autonomous Vehicle Simulation*. 2023. URL: `https://community.intel.com/t5/Blogs/Tech-Innovation/Artificial-Intelligence-AI/CARLA-and-AVL-Collaboration-Ushering-in-a-New-Era-of-Off-Road/post/1493433`.

[43] Toyota Research Institute. "Toyota Research Institute Supports Development of Open-Source Automated Driving Simulator". In: *News Release* (2018).

[44] Epic Games. *Unreal Engine*. Version 4.22.1. Apr. 25, 2019. URL: `https://www.unrealengine.com`.

[45] Nvidia. *PhysX*. Version 5.1. URL: `https://developer.nvidia.com/physx-sdk`.

[46] *ASAM OpenDRIVE*. Tech. rep. Association for Standardization of Automation and Measuring Systems, 2024.

[47] H.B. Pacejka and Society of Automotive Engineers. *Tire and Vehicle Dynamics*. SAE International, 2006. ISBN: 9780768017021.

[48] Brian Beckman. *The Physics of Racing*. 2001.

[49] K.M. Lynch and F.C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017. ISBN: 9781108509695.

[50] Yuanpeng Chen et al. "2018 21st International Conference on Intelligent Transportation Systems (ITSC)". In: *Optimization of Pure Pursuit Controller based on PID Controller and Low-pass Filter* (2018), pp. 3294–3299.

[51] J. M. S. Ribeiro et al. "Comparison of PID controller tuning methods: analytical/classical techniques versus optimization algorithms". In: *2017 18th International Carpathian Control Conference (ICCC)*. 2017, pp. 533–538.

[52] Hart P. E. Nilsson N.J. Raphael B. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". In: *IEEE Transactions on Systems Science and Cybernetics* (1968).

[53] Steven M. LaValle. "Planning Algorithms". In: *Cambridge University Press* (2006).

[54] M. Askari Hemmat. "Path Planning Algorithms for Autonomous Mobile Robots". In: *Concordia University* (2018).

[55] Armijo and Larry. "Minimization of functions having Lipschitz continuous first partial derivatives". In: *Pacific J. Math. 16 (1): 1–3* (1966).

[56] Ioannis Filippidis and Kostas J. Kyriakopoulos. "Roadmaps using gradient extremal paths". In: *2013 IEEE International Conference on Robotics and Automation*. 2013, pp. 370–375. DOI: `10.1109/ICRA.2013.6630602`.

[57] Barnabas Bede. *Mathematics of Fuzzy Sets and Fuzzy Logic*. Springer Berlin, Heidelberg, 2013.

[58] M. Park Y.W. Cho C.W. Park. "An indirect model reference adaptive fuzzy control for SISO Takagi-Sugeno model". In: *Fuzzy Sets Syst. 131, 197–215* (2002).

[59] A. Mohammadzadeh et al. *Modern Adaptive Fuzzy Control Systems*. Springer Cham, 2022, pp. 17–28.

[60] A. Sadollah. *Fuzzy Logic Based in Optimization Methods and Control Systems and its Applications*. InTech, 2018.

[61] T.J. Ross. *Fuzzy Logic with Engineering Applications. 3ed*. John Wiley & Sons, 2010.

[62] Dongrui Wu and Jerry M. Mendel. "Recommendations on designing practical interval type-2 fuzzy systems". In: *Engineering Applications of Artificial Intelligence* 85 (2019), pp. 182–193. ISSN: 0952-1976.

[63] Jean Jiang Lizhe Tan. *Digital Signal Processing: Fundamentals and Applications*. Academic Press, 2018.

[64] Amir Arslan Haghrah and Sehraneh Ghaemi. *PyIT2FLS: A New Python Toolkit for Interval Type 2 Fuzzy Logic Systems*. 2019. arXiv: `1909.10051 [eess.SY]`.

[65] Chen Su et al. "The Evaluation Standards and Methods of Human Vibration Comfort in Seated and Recumbent Position". In: *Proceedings of the 15th International Conference on Man–Machine–Environment System Engineering*. Ed. by Shengzhao Long and Balbir S. Dhillon. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 521–528.