Mobility21

A USDOT NATIONAL UNIVERSITY TRANSPORTATION CENTER

Carnegie Mellon University

Penn
University of Pennsylvania

The Ohio State University

CCAC

Effect of Pedestrian and Crowds on Vehicle Motion and Traffic Flow

Bilal Hejase (https://orcid.org/0000-0001-5125-0892)
Muhammad Saim (https://orcid.org/0009-0004-7228-285X)
Dongfang Yang (https://orcid.org/0000-0001-9212-6804)
Mert Koç (https://orcid.org/0000-0001-5777-4072)
Fatema T Johora (https://orcid.org/0000-0001-8857-9958)
Keith Redmill (https://orcid.org/0000-0003-1332-1332)
Ümit Özgüner, PI (https://orcid.org/0000-0003-2241-7547)

FINAL RESEARCH REPORT - July 25, 2023

Contract # 69A3551747111

Contents

1 Introduction **4**

2 Multi-State Social Force Based Framework for Vehicle-Pedestrian Interaction in Uncontrolled Pedestrian Crossing Scenarios **6**
2.1 Introduction . . . 6
2.2 Framework . . . 8
2.3 Interactive Pedestrian Motion . . . 9
2.3.1 Gap Estimation . . . 9
2.3.2 Pedestrian Dynamics . . . 9
2.3.3 Multi-State Social Force Interaction Model . . . 9
2.4 Vehicle Motion . . . 11
2.4.1 Pedestrian Motion Prediction . . . 11
2.4.2 Vehicle Dynamics . . . 11
2.4.3 Control Strategies . . . 12
2.5 Experiments . . . 13
2.6 Results . . . 14
2.6.1 Qualitative Evaluation . . . 14
2.6.2 Quantitative Evaluation . . . 16
2.7 Conclusion . . . 17

3 On the Generalizability of Motion Models for Road Users in Heterogeneous Shared Traffic Spaces **18**
3.1 Introduction . . . 18
3.2 Related Works . . . 20
3.3 Modeling Process . . . 21
3.4 Data Sets and Interaction Classifications . . . 23
3.4.1 Data Sets . . . 23
3.4.2 Interaction Classification . . . 24
3.5 Agent-Based Simulation Model . . . 24
3.6 Calibration Methodology . . . 29
3.6.1 Clustering . . . 30
3.6.2 Calibration . . . 32
3.7 Evaluation . . . 34
3.7.1 Evaluation Metrics . . . 34
3.7.2 Results . . . 35
3.8 Conclusion . . . 37

4 Social Force Model with Model Predictive Control for Longitudinal Vehicle Control in Pedestrian-Dense Scenarios 38
4.1 Background . . . 38
4.2 Pedestrian Motion Prediction . . . 39
4.2.1 Vehicle-Crowd Interaction Model . . . 39
4.2.2 Vehicle Influence . . . 40
4.2.3 Motion Prediction . . . 40
4.3 Longitudinal Speed Regulation . . . 40
4.3.1 Vehicle Dynamics . . . 40
4.3.2 Model Predictive Controller (MPC) Synthesis . . . 42
4.3.3 Cost Function Design and Quadratic Programming (QP) Problem Formulation . . . 43
4.3.4 MPC Feasibility and Supplementary Proportional-integral-derivative (PID) Control . . . 44
4.3.5 Overall Algorithm . . . 46
4.4 Evaluation . . . 46
4.5 Result . . . 47
4.5.1 Comparison Between MPC and PID . . . 47
4.5.2 Different Pedestrian Density . . . 50
4.6 Conclusion . . . 50

5 Predicting Pedestrian Crossing Intention 52
5.1 Background . . . 52
5.1.1 Related Work . . . 54
5.2 Proposed Method . . . 56
5.2.1 Problem formulation . . . 56
5.2.2 Input acquisition . . . 57
5.2.3 Model architecture . . . 58
5.3 Experiments . . . 59
5.3.1 Dataset and Benchmark . . . 59
5.3.2 Implementation . . . 60
5.3.3 Ablation study . . . 60
5.4 Results . . . 61
5.4.1 Quantitative Results . . . 62
5.4.2 Qualitative Results . . . 63
5.5 Conclusion . . . 64

6 Pedestrian Emergence Estimation and Occlusion-Aware Risk Assessment 65
6.1 Background . . . 65
6.2 Methodology . . . 67
6.2.1 Estimating Pedestrian Emergence from Occlusions . . . 67
6.2.2 Risk Assessment . . . 67
6.2.3 Driving Policy . . . 68
6.3 Experiments . . . 71
6.3.1 Metrics . . . 72
6.4 Results . . . 72
6.5 Conclusion . . . 73

7 Stability Regulation of Learning-Based Continuous Control **77**
7.1 Background . . . 77
7.1.1 Reinforcement Learning . . . 78
7.1.2 Lyapunov Stability . . . 79
7.2 Lyapunov Stability-Regulated DDPG Framework . . . 80
7.2.1 Joint Learning of Dynamics and Control-Lyapunov Function . . . 80
7.2.2 Lyapunov Stability Regulated DDPG Control . . . 81
7.2.3 Framework Overview and Remarks . . . 82
7.3 Simulation Experiments . . . 83
7.3.1 Simulation Setup . . . 83
7.3.2 Reward Function and Termination . . . 84
7.3.3 Network Parameters . . . 84
7.3.4 Results . . . 84
7.4 Conclusion . . . 87

8 Conclusion **88**
8.1 Conclusion . . . 88
8.2 Future Work . . . 88

A Research Products for this Project **90**
A.1 Journal Publications . . . 90
A.2 Conference Publications . . . 90
A.3 Dissertation and Thesis . . . 90

Bibliography **92**

Chapter 1

Introduction

Vehicle-pedestrian interactions in shared spaces represents a complex safety problem. Ideally, the vehicle must react safely to any pedestrian behavior, while the pedestrian behavior itself can be very complex and unpredictable. To emphasize this safety problem, in a 2019 Traffic Safety Facts report by the National Highway Traffic Safety Administration (NHTSA) [1], it was shown that the percentage of pedestrian fatalities was increasing from 2008 to 2017, even as advanced driving assist systems (ADASs) were being developed and deployed. Pedestrian safety has become an increasingly important problem in the development of state-of-the-art autonomous driving systems. These systems must be able to handle both common interactions, e.g., a pedestrian crossing a street at a crosswalk, and risky or rare scenarios, e.g., an occluded pedestrian emerging between parked cars.

Handling vehicle-pedestrian interactions, requires the utilization of accurate models of pedestrian behavior. One such commonly used paradigm within the research community, is the use of the Social Force Model for modeling pedestrian behavior. This model is useful as data-driven methods lack to deal with risky or rare behavior, due to the lack of edge case samples. In addition to the explicit modeling of pedestrian behavior, the vehicle must possess the predictive ability to determine the pedestrian intention. For example, does the pedestrian intend to cross the road? If so, in what direction will they move? Similar to the popularity of model predictive methods for nonlinear control tasks, pedestrian intention prediction can increase the safety of vehicle-pedestrian interactions and allow the vehicle to react proactively to the pedestrian's behaviors.

With the introduction of deep neural networks (DNNs) to sequential control tasks, such as end-to-end driving tasks, the problem of safety remains a hindrance to the real-world application of this technology. Unlike rule-based methods that represent white-box models with safety or stability certificates, DNN-based control lack these certificates due to the black-box nature. When considering pedestrian safety, such safety certificates that give insight into the performance of the controller become critical. A rising topic has been the application of control-Lyapunov functions and learner-verifier frameworks to certify black-box DNNs, however its application to continuous control remains an open problem. Continuous control is emphasized as the problem of vehicle-pedestrian safety, or any general driving task in the real-world, requires continuous reasoning.

The rest of the report focuses on different cases of safety for vehicle-pedestrian interactions. Chapter 2 describes a multi-state social force model for modeling pedestrian crossing behavior in uncontrolled pedestrian crossing scenarios for vehicle-pedestrian interactions. The effects of vehicle-pedestrian interactions on motion planning is considered in Chapter 3, such as stepping off a curb for individual and crowds of pedestrians. Chapter 4

describes the utilization of social force models within model predictive control schemes for longitudinal vehicle control. This chapter specifically focuses on pedestrian-dense traffic scenarios and aims to provide enhanced safety by including the prediction of pedestrian or crowd movements in the control formulation. Chapter 5 considers the problem of pedestrian crossing intention prediction utilizing data-driven approaches from several sensing sources. In Chapter 6 the occluded pedestrian case is considered, which represents a complex problem in perception, prediction, and control. A probabilistic risk assessment approach using pedestrian estimation and occlusion is described. Chapter 7 considers the stability of deep neural network control in continuous control tasks to enhance the safety of both vehicles and vulnerable road users.

Chapter 2

Multi-State Social Force Based Framework for Vehicle-Pedestrian Interaction in Uncontrolled Pedestrian Crossing Scenarios

This chapter is derived from the published work in [2].

2.1 Introduction

Pedestrian safety has been an important issue in transportation for a long time. According to NHTSA [1], the percentage of pedestrian fatalities in total traffic fatalities has increased from 12% in 2008 to 16% in 2017. This implies that pedestrian safety is still a big concern. Developing advanced driver-assistance systems (ADAS) is a promising route to improve pedestrian safety. Although research, applications, and products are continuously updating, the approaches for the vehicle to handle the vehicle-pedestrian interaction (VPI) still need to be improved, especially in uncontrolled scenarios (no crosswalk and no traffic signals). For example, a recent report by NTSB [3] about a pedestrian jaywalking fatality involved with an autonomously driving vehicle demonstrated the inadequacy of the pedestrian handling functionality in the automated driving system. Therefore, this Chapter focuses on the VPI scenario in which an autonomous vehicle interacts with a crossing pedestrian at uncontrolled road segments, as illustrated in Fig.2.1. This scenario is very common and is closely linked to pedestrian safety.

It is generally challenging to model and evaluate such VPI because various VPI patterns can not be usually observed in real-world situations. Having an effective VPI framework for simulation would benefit the testing of newly designed algorithms before moving to the next step. To this end, this Chapter proposed an improved VPI framework that can produce more realistic pedestrian behaviors by extending the social force pedestrian motion model [4]. The framework is suitable for evaluating different automated driving strategies in different situations.

Several works have studied the VPI in crossing scenarios. A recent comprehensive review [5] identified factors such as pedestrian demographics, traffic dynamics, and environmental conditions by surveying both the classical driver-pedestrian interaction and the VPI that involves automated vehicles. Gap acceptance is a major factor that affects

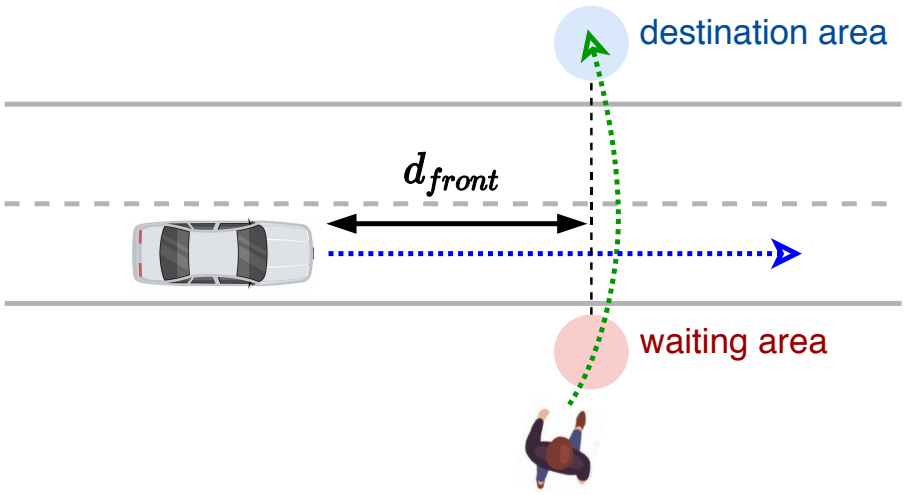

Figure 2.1: Scenario illustration. The pedestrian first goes to the waiting area (red circle), judges the situation to decide crossing or yielding, and walks to the destination area (blue circle). The vehicle regulates it longitudinal speed to balance between the safety and the efficiency of finishing the interaction.

the pedestrian's behavior in crossing scenarios. Existing works like [6, 7] studied the gap acceptance in different conditions, i.e., at mid-blocks and in front of a platooning of low-speed autonomous pods. A stochastic interaction model using the multivariate Gaussian mixture model [8] was also proposed to simulate the mutual interaction by simultaneously considering the behavior of both the vehicle and the pedestrian. The above works addressed the VPI in a statistic way, however, if we focus on the precise motion of the interacting agents, it is expected to have a more detailed VPI framework that models both agents' dynamics.

Regulating longitudinal speed is the most direct approach for the vehicle to handle the VPI in crossing scenarios. Partially observable Markov decision process (POMDP) [9, 10, 11] is one of the popular methods for discretized longitudinal control incorporating the uncertainty of the pedestrian behavior. Model-based control methods like hybrid feedback control [12] and model predictive control [13] are also suitable for this type of problem. Although model-based control does not inherently consider the pedestrian's behavior, a prediction module [14] can be applied to provide the predicted pedestrian motion hence utilized by the model-based controllers.

Pedestrian behavior is usually described by a motion model that can execute decisions like when and how to cross the road. A common assumption is that the pedestrian simply makes the decision on when to start crossing but while the pedestrian is crossing, a constant crossing velocity is maintained [8, 12]. The assumption is good for analyzing decision-making, but not for generating precise motion. This Chapter introduced a new model that combines the microscopic social force pedestrian model with a state machine to describe the explicit motion of the pedestrian in crossing scenarios. Social force model [15] was originally designed for simulating the crowd dynamics of multiple interacting pedestrians. Recently, the effect of the vehicle on pedestrians has been added into the social force model [16, 17, 4], which makes it possible to be used in the VPI framework. In terms of decision making, we designed a state machine to handle different phases of a complete

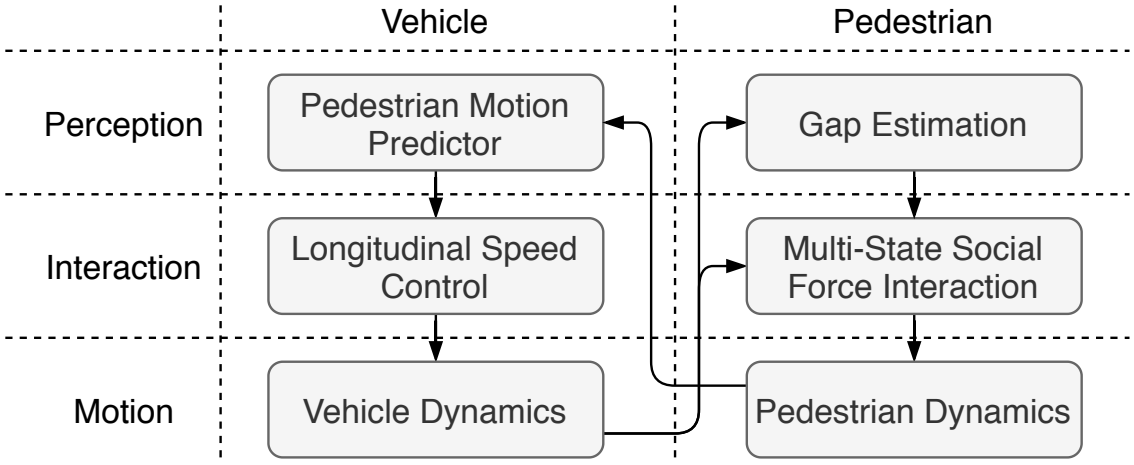

Figure 2.2: Framework for vehicle-pedestrian interaction. Both the vehicle and the pedestrian follows a 3-level hierarchy.

crossing behavior, which includes approaching the road, judging the situation, and crossing the road.

The contribution of this Chapter is summarized as follows: (a) A general VPI framework for uncontrolled crossing scenarios was proposed and can simulate various VPI patterns. Both the vehicle and the pedestrian have a hierarchical pipeline of perception, interaction, and motion that interact with each other. (b) A newly designed multi-state social force model was proposed to describe more realistic pedestrian motion under the effect of the interacting vehicle, i.e., the pedestrian can change velocity and direction in the process of crossing. The proposed pedestrian model also introduces the uncertainty in both the gap acceptance and the desired crossing speed. (c) Different vehicle control strategies (pure velocity keeping, obstacle avoidance, and model predictive) were implemented to verify the effectiveness and the capability of the VPI framework. The simulation can successfully generate various VPI patterns in the uncontrolled crossing scenarios.

2.2 Framework

The process of VPI can be interpreted as a process of two agents mutually recognizing and affecting each other. A general framework consisting of the same hierarchy for either the pedestrian or the vehicle can be conceptually divided as layers of perception, interaction, and motion, as illustrated in Fig. 7.1. All of these layers should ideally interact with any of the others.

In this Chapter, we are dealing with the VPI in a specific and representative scenario as shown in Fig. 2.1. The vehicle moves along the lane adjacent to the side where the pedestrian appears, predicts whether the pedestrian is going to cross the road, and adjusts its speed accordingly to avoid the collision while keeping its desired speed as much as possible. The pedestrian emerges from the sidewalk, goes to the waiting area, and judges the situation to see if it is safe to cross the road. If safe the pedestrian will proceed to cross, otherwise, the pedestrian will wait until the situation becomes safe.

To address this specific scenario, the interaction among the aforementioned layers was streamlined as described by the arrows in Fig. 7.1. Both the vehicle and the pedestrian are

assumed to know the exact past and current states of each other. Therefore, the vehicle's perception layer employs a predictor to predict the pedestrian's future motion. And the pedestrian's perception layer applies an estimator for the time gap of the vehicle going from its real-time position to the pedestrian's crossing position. The interaction is not controlled by either traffic signals or crosswalk markings, which represents the common situations in residential areas or when the pedestrian jaywalks. The vehicle's interaction layer adopts a longitudinal speed control that can somehow take advantage of the trajectory predicted by the pedestrian motion predictor. The pedestrian's interaction layer uses the newly-designed multi-state social force model that allows the pedestrian to change the speed and the direction to avoid uncomfortable movement due to the approaching vehicle. Both bottom layers apply the dynamics to obey the physics of the motion. All the above layers are explained in detail in the following sections.

2.3 Interactive Pedestrian Motion

2.3.1 Gap Estimation

As discussed in [18, 12], *gap acceptance* is a major factor that determines whether the pedestrian decides to cross or yield to the vehicle. It is defined as a time:

$$t_{gap} = \frac{d_{front}}{v_{veh}} \tag{2.1}$$

where v_{veh} is the current vehicle velocity, and d_{front} is the the distance to the interaction shown in Fig. 2.1. t_{gap} is updated as time evolves and is compared with the threshold of the gap acceptance τ_{gap}, which is drawn from a normal distribution $N(\mu_{gap}, \sigma_{gap})$. The statistics follows the results in [19].

2.3.2 Pedestrian Dynamics

Instead of assuming a straight motion with constant crossing speed [8, 12], the social force model applies a 2D point-mass Newtonian dynamics [4]:

$$\ddot{x}_p = \frac{1}{m_p} f_{total}, \tag{2.2}$$

where $x_p \in \mathbb{R}^4$ is the pedestrian state vector that represents positions and velocities in x, y axes, respectively, m_p is the mass of the pedestrian, and $f_{total} \in \mathbb{R}^2$ is the total applied force, which is detailed in the following subsection. In the simulation, the point-mass dynamics was discretized with the time step Δt. The dynamics also imposes constraints on the velocity $v_{p,max}$ and the acceleration $a_{p,max}$.

2.3.3 Multi-State Social Force Interaction Model

Social Force

Social force model describes each type of the interaction as a virtual force that applies on the pedestrian dynamics. The cumulative interaction effect is the summation of all individual virtual forces. In the proposed model, the total force was designed as:

$$f_{total} = f_{des} + f_{veh}, \tag{2.3}$$

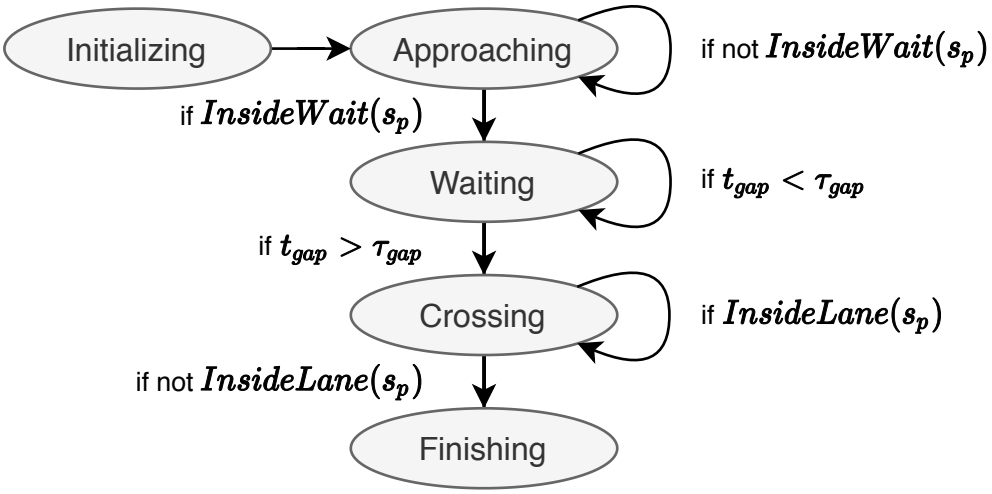

Figure 2.3: State transition of the social force model

where $f_{des} \in \mathbb{R}^2$ is the destination force and $f_{veh} \in \mathbb{R}^2$ is the vehicle effect force.

The destination force concretizes the pedestrian's desire to reach a particular location. Given a destination position, this force adjusts the pedestrian's velocity to walk toward the given destination with the desired speed. The destination force f_{des} is defined as:

$$f_{des} = k_{des}(v_p - v_d), \tag{2.4}$$

where $v_p \in \mathbb{R}^2$ is the current pedestrian velocity vector, $v_d \in \mathbb{R}^2$ is the desired velocity vector that points to the destination, and k_{des} is a scalar parameter that magnifies the difference between v_p and v_d. The desired velocity vector is defined as $v_d := v_0 \cdot \frac{s_{des}-s_p}{\sqrt{|s_{des}-s_p|^2+(\sigma_{des})^2}}$. In the definition, $s_{des} \in \mathbb{R}^2$ is the destination, $s_p \in \mathbb{R}^2$ is the current pedestrian position, and σ_{des} is a scalar parameter that decreases the desired speed when the pedestrian is getting close to the destination. v_0 is the desired speed magnitude that represents the most comfortable walking speed. v_0 is drawn from a normal distribution $N(\mu_{v_0}, \sigma_{v_0})$. The distribution follows the statistic results in [20].

The vehicle effect is defined as a repulsive force:

$$f_{veh} = A_{veh} \cdot \exp(-b \cdot d_{v2p}) \cdot \vec{n}_{v2p}. \tag{2.5}$$

d_{v2p} is the distance from the influential point of the vehicle s_{inf} to the current pedestrian position s_p. The influential point s_{inf} is selected as the point on the vehicle contour that is closest to s_p. A pedestrian radius R_p and an extension length l_e for the contour are considered as the buffer. Therefore, $d_{v2p} := |s_p - s_{inf}| - R_p - l_e$. $\vec{n}_{v2p}$ is a unit direction vector, pointing from s_{inf} to s_p. The force magnitude applies an exponential relationship, with parameters A_{veh} and b_{veh}. More details about the f_{veh} can be found in [4].

State Transition

A crossing behavior is decomposed into several phases, as shown in Fig. 2.3, during which the status of the social force model is slightly different:

- *Initializing*: The desired speed $v_0 \sim N(\mu_{v_0}, \sigma_{v_0})$ and the gap threshold $\tau_{gap} \sim N(\mu_{gap}, \sigma_{gap})$ are obtained. The destination s_{des} is set to be the waiting area in Fig. 2.1.

- *Approaching*: Only the destination force f_{des} is effective in equation (2.3). If the waiting area, noted as *InsideWait()* in Fig. 2.3, is reached, switch to the next state.
- *Waiting*: At this state, the pedestrian judges the gap t_{gap}. If $t_{gap} > \tau_{gap}$, the pedestrian starts to cross (by switching the destination s_{des} to be the destination area in Fig. 2.1) and switches to the next state. Otherwise, the pedestrian just waits in the waiting area until $t_{gap} > \tau_{gap}$, which could be either the vehicle has passed the crossing position or the vehicle slows down and yields to the pedestrian. Still, only f_{des} is effective.
- *Crossing*: Pedestrian is within the lane where the vehicle is driving. Both the destination force f_{des} and the vehicle effect force f_{veh} are effective. The desired speed v_0 will be temporarily changed if the pedestrian needs to avoid aggressive vehicle maneuvers (e.g., not yielding but accelerating). This is achieved by comparing the time to collision (TTC) for the vehicle $t_{TTC} := \frac{d_{front}}{v_{veh}}$ with the time to finishing the crossing (TTF) for the pedestrian $t_{TTF} := \frac{d_{rem}}{v_0}$, where d_{rem} is the remaining distance from current pedestrian position to the other edge of vehicle driving lane. If $t_{TTC} < t_{TTF}$, then the updated desired speed $v_0' = d_{rem}/t_{TTC}$. Once the pedestrian leaves the vehicle driving lane, noted as $InsideLane()$ in Fig. 2.3, switch to the next state.
- *Finishing*: The pedestrian continues to the destination area. The vehicle effect force f_{veh} is no longer effective.

2.4 Vehicle Motion

2.4.1 Pedestrian Motion Prediction

Pedestrian motion prediction usually applies a system $T_{pred} = f_{pred}(T_{obs})$ that inputs an observed trajectory $T_{obs} = \{x_p(k-M+1), x_p(k-M+2), \cdots, x_p(k)\}$ of length M and outputs a predicted trajectory $T_{pred} = \{x_p(k+1), x_p(k+2), \cdots, x_p(k+N)\}$ of length N. This Chapter applies a linear pedestrian motion predictor, in which only the last observed pedestrian state $x_p(k)$ is used as input, and the output states are propagated by assuming the same velocity last observed.

2.4.2 Vehicle Dynamics

A longitudinal point-mass model with drag effect is applied as the vehicle dynamics: $M\ddot{s}(t) + \alpha\dot{s}(t) = u(t)$, where s is the longitudinal position, M is the vehicle mass, α is a drag coefficient, and u is the control action (throttle/brake). Rewriting the dynamics using a state vector $x = [s, \dot{s}]^T$ (position and velocity) into discretized time of Δt, we have:

$$x(k+1) = Ax(k) + Bu(k) \tag{2.6}$$

where $A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 - \frac{\alpha \Delta t}{M} \end{bmatrix}, B = \begin{bmatrix} 0 \\ \Delta t \end{bmatrix}$, and $u(k)$ is the discretized control action. A constraint $[v_{min}, v_{max}]$ on speed is imposed. Also, the control action is bounded by $[u_{min}, u_{max}]$ and the control action rate is bounded by $[\Delta u_{min}, \Delta u_{max}]$.

2.4.3 Control Strategies

The speed controller has two objectives: (a) keeping a safe distance to the pedestrian; (b) maintaining the desired speed as much as possible. This Chapter analyzed 3 different control strategies. A vanilla pure velocity keeping control (VKC) was implemented as a baseline and to test the pedestrian behavior under extreme conditions as well. It is compared with an obstacle avoidance control (OAC) and a model predictive control (MPC) that use the predicted pedestrian motion.

Velocity Keeping Control (VKC)

It simply applies a PI controller to keep the vehicle's desired velocity $v_{0,veh}$:

$$u = K_P \cdot (v_{0,veh} - v_{veh}) + K_I \cdot I, \tag{2.7}$$

where $I(k) = \sum_{i=0}^{k} \big(v_{0,veh}(i) - v_{veh}(i)\big)$ is the cumulative error of the speed deviation until current time step k and K_P, K_I are the proportional gain and the integral gain, respectively.

Obstacle Avoidance Control (OAC)

It extends the VKC by adding a strategy to decelerate if the predicted pedestrian trajectory at any time obstructs the vehicle driving lane. In that case, the control action is obtained by:

$$u = -\frac{v_{veh}^2}{2 \cdot (d_{front} - d_{safe})}, \tag{2.8}$$

where d_{safe} is the safe distance, which equals to the d_{front} when the vehicle decelerates and stops right in front of the pedestrian with the minimum deceleration.

Model Predictive Control (MPC)

MPC predicts N_p steps of the vehicle motion. Two safety criteria were designed and imposed on the constraints of the MPC problem. First, for any predicted pedestrian state $x_p(k+m), m \in 1, 2, \cdots, M$ that lies in the vehicle driving lane, its longitudinal position along the road $s_{obs}(k+n), n \in \mathbb{N}$ is used as a longitudinal displacement constraint for the MPC. $\mathbb{N} \subseteq \{1, 2, \cdots, N_p\}$ is the index set of the future time steps when the predicted $x_p(k+m)$ lies in the vehicle driving lane. It must satisfy

$$|s_{obj}(k+n) - s(k+n)| > d_{safe}, \forall n \in \mathbb{N}, \tag{2.9}$$

where d_{safe} is the safe distance that should be always kept between the vehicle and the pedestrian and $s(k+n)$ is the predicted vehicle longitudinal position at time step $k+n$. These constraints guarantee that the vehicle never collides with the pedestrian within the prediction horizon. Second, if the predicted pedestrian position at final step N_p also lies in the vehicle driving lane, a constraint on the vehicle's last predicted speed should be added such that the vehicle is expected to stop in front of the pedestrian at least a safe distance of d_{safe}. Therefore, a deceleration distance $d_{dec} := \frac{v(k+N_p)}{2} \cdot \frac{v(k+N_p)}{|u_{min}|}$ that allows the vehicle to decelerate from the terminal speed to zero speed within maximum deceleration (i.e., minimum control action u_{min}, which is negative) is added to d_{safe}. However, the square of $v(k+N_p)$ is not supported as constraints in most available MPC solvers. We relax

the constraint in a way such that $d_{dec} = \frac{v(k+N_p)}{2} \cdot \frac{v(k+N_p)}{|u_{min}|} \leq \frac{v(k+N_p)}{2} \cdot \frac{v_{max}}{|u_{min}|}$ hence the constraint becoming linear, where v_{max} is the maximum allowed speed. So, the terminal constraint is:

$$|s_{obj}(k+N_p) - s(k+N_p)| > \frac{v_{max}}{2|u_{min}|} \cdot v(k+N_p) + d_{safe}. \tag{2.10}$$

Naturally, constraints on the velocity, control action, and control action rate should be added, they are:

$$v_{min} < v(k+n) < v_{max}, \forall n \in \{1, 2, \cdots, N_p\} \tag{2.11}$$
$$u_{min} < u(k+n) < u_{max}, \forall n \in \{0, 1, \cdots, N_p - 1\} \tag{2.12}$$
$$\Delta u_{min} < \Delta u(k+n) < \Delta u_{max}, \forall n \in \{0, 1, \cdots, N_p - 1\} \tag{2.13}$$

Finally, the MPC problem is formulated as:

$$\boldsymbol{U}^* = \arg\min_{\boldsymbol{U}} \Big(\sum_{n=1}^{N_p} w_v \big(v(k+n) - v_{0,veh}\big)^2 + \sum_{n=0}^{N_p-1} w_u \big(u(k+n)\big)^2 \Big) \tag{2.14}$$
$$\text{s.t. } [s(k), v(k)]^T = x(k), \text{and } (2.6)(2.9)(2.10)(2.11)(2.12)(2.13),$$

where $x(k)$ is the current vehicle state, and w_v, w_u are the weights for the cost of velocity and control, respectively. In extreme cases if solving the MPC fails, maximum deceleration u_{min} is applied, with the constraint (2.13) still valid.

Table 2.1: Parameters in the Simulation

Symbol	Value	Units	Symbol	Value	Units
$v_{p,max}$	2.5	m/s	m_p	80.0	kg
$a_{p,max}$	5.0	m/s^2	R_p	0.27	m
μ_{v_0}	1.4	m/s	μ_{gap}	2.5	sec
σ_{v_0}	0.2	m/s	σ_{gap}	4.0	sec
A_{veh}	200.0	-	σ_{des}	1.0	-
b_{veh}	2.6	-	k_{des}	300.0	-
M	2000.0	kg	α	100	-
u_{min}	-7.0	m/s^2	v_{min}	0.0	m/s
u_{max}	7.0	m/s^2	v_{max}	22.5	m/s
Δu_{min}	-5.0	m/s^3	d_{safe}	3.0	m
Δu_{max}	5.0	m/s^3	N_{pred}	15	-
K_P	1.0	-	w_v	1.0	-
K_I	0.1	-	w_u	1.0	-

2.5 Experiments

A two-lane road was created for simulation experiments, with lane width $W_{lane} = 3.2m$. For each episode, the pedestrian was initialized at 2 meters from the right to the right edge of the road. The destination was set at 3.6 meters from the left to the other edge of the road. The waiting area was centered at 0.5 meters to the right road edge. These 3 points have the same longitudinal position. The vehicle was initialized with 30 different

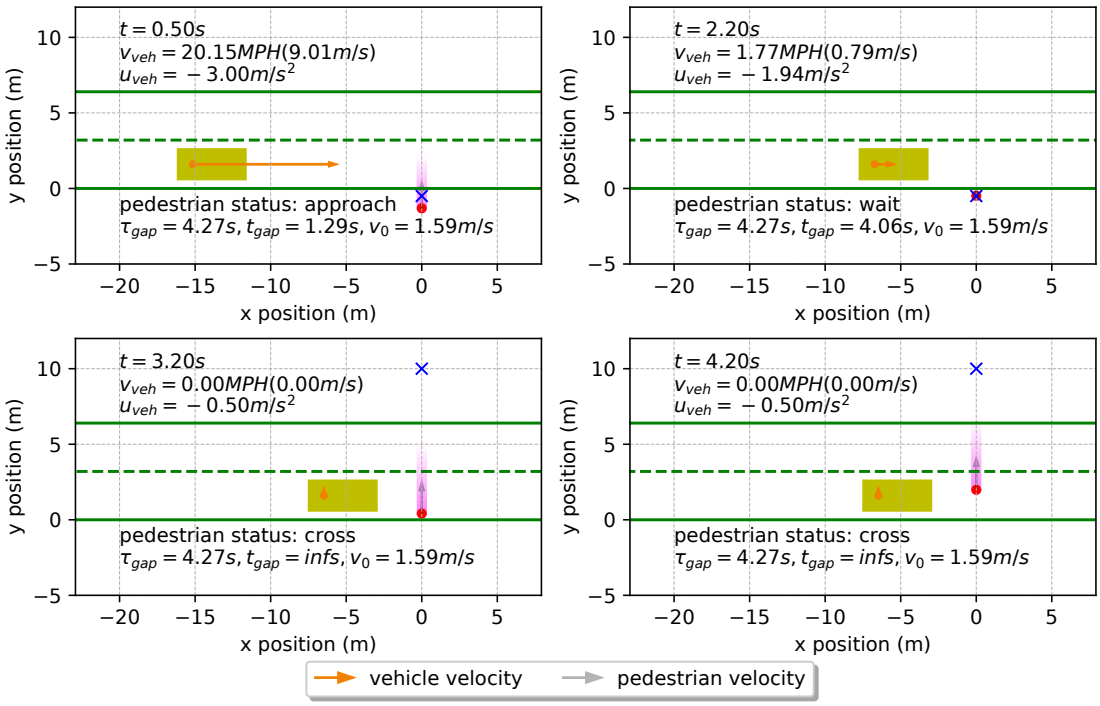

Figure 2.4: Screenshots of a simulation that applies MPC. The initial state of the vehicle (yellow box) is $d_{front,0} = 16.5m$ and initial/desired speed $\dot{s}_0 = 10m/s$. The pedestrian (red circle) was initialized with a gap acceptance threshold of $\tau_{gap} = 4.27s$ and a desired walking velocity $v_0 = 1.59m/s$. The blue 'x' is the pedestrian's destination. Purple shadows indicate the predicted pedestrian positions (lighter color means longer predicted time).

combinations of initial longitudinal position $d_{front,0} \in \{11.5, 16.5, 21.5, 26.5, 31.5, 36.5\}$ and speed $\dot{s}_0 \in \{2, 4, 6, 8, 10\}$ so that the majority of scenarios of different time gaps were covered. The desired speed was set as $v_{0,veh} = \dot{s}_0$. For each combination, 3 different control strategies were simulated for 200 times, respectively. CVXPY [21] was applied as the MPC solver. Table 2.1 shows the parameter values, which were manually tuned according to the parameter values in previous works [4, 13, 12].

2.6 Results

In general, there are 4 hyper-parameters in the simulation configuration. They are the threshold of the accepted gap τ_{gap}, the desired pedestrian walking speed v_0, the initial vehicle longitudinal position d_{front}, and the initial/desired vehicle speed $\dot{s} = v_{0,veh}$. The results were evaluated based on selected combinations of the above hyper-parameters.

2.6.1 Qualitative Evaluation

Fig. 2.4 shows the screenshots of a simulation using MPC. The evolution of the corresponding pedestrian state and vehicle state are plotted in Fig. 2.5. At the beginning stage, the vehicle recognized the pedestrian who was approaching the edge of the road. The pedestrian motion predictor provided a sequence of predicted future positions that lie in the

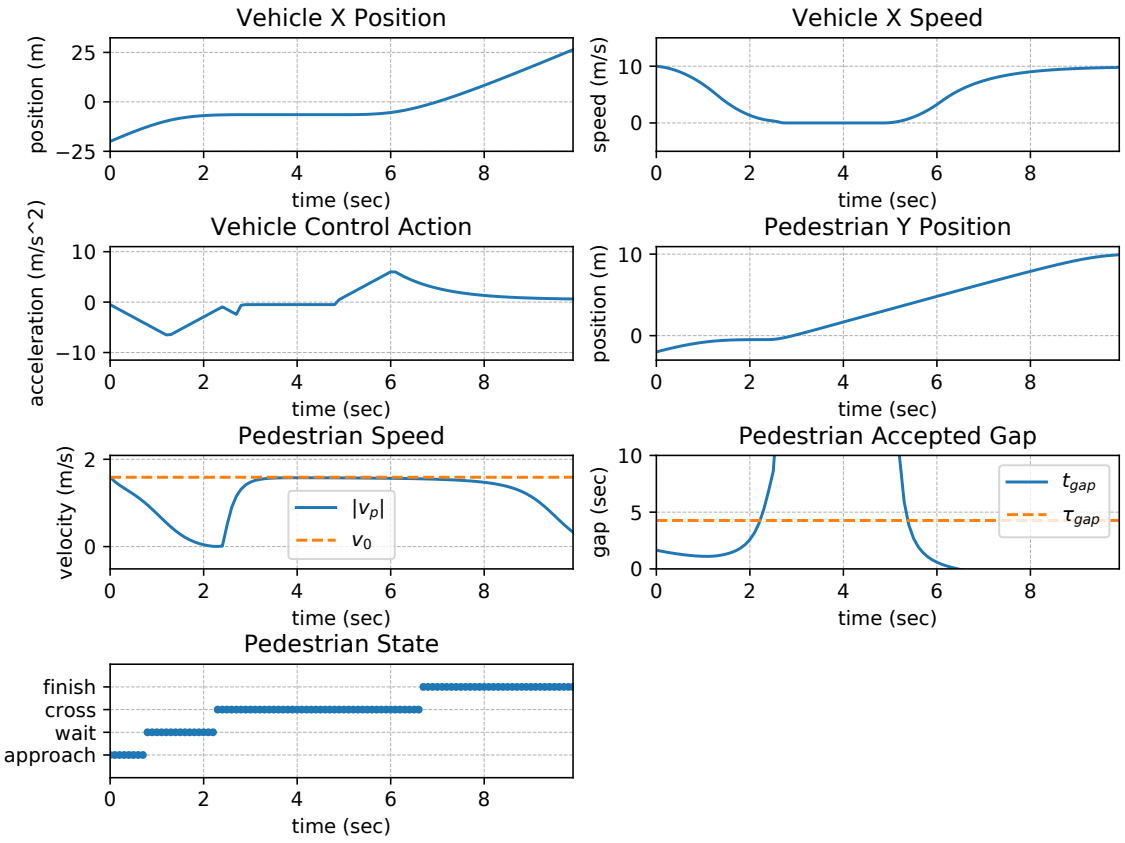

Figure 2.5: Evolution of some states that corresponds to the simulation in Fig. 2.4: vehicle's longitudinal (x) position, velocity, and control action; pedestrian's crossing distance (y position), speed, state, and the estimated gap.

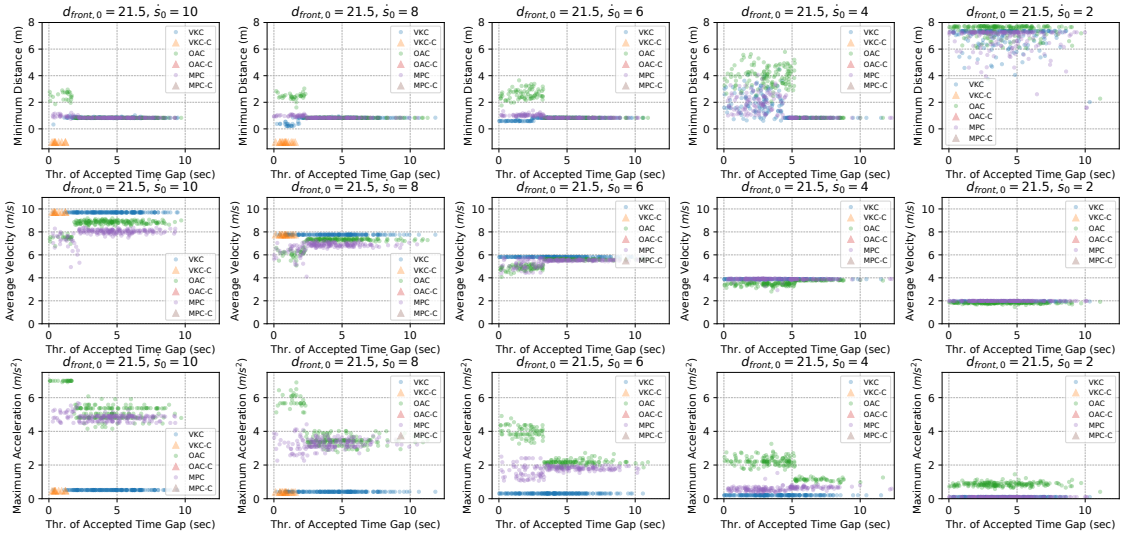

Figure 2.6: Quantitative analysis of the simulation results of $d_{front,0} = 21.5$ and $\dot{s}_0 \in \{2, 4, 6, 8, 10\}$. The 1st row shows the minimum distance from the pedestrian to any point of the vehicle contour during the entire simulation, the 2nd row shows the average velocity, and the 3rd row shows the maximum value of the absolute acceleration/deceleration. Each plot shows the results of running each control strategy for 200 times. In each plot, VKC indicates velocity keeping control, OAC indicates obstacle avoidance control, and MPC indicates model predictive control. The suffix -C indicates collision results.

vehicle driving lane. Therefore, the MPC was generating the control action to slow down the vehicle. In the meantime, the pedestrian was also slowing down because $t_{gap} < \tau_{gap}$. As the vehicle was continuing to slow down, the estimated gap t_{gap} was increasing (see the gap value in Fig. 2.5), and around $t = 2.2s$, $t_{gap} > \tau_{gap}$, which made the pedestrian switch from *waiting* state to *crossing* state. After that, the pedestrian was crossing the road while the vehicle was fully stopped and waiting for the pedestrian to complete the crossing. This example demonstrated a successful interactive VPI, which validated the effectiveness of the proposed framework.

2.6.2 Quantitative Evaluation

The quantitative evaluation of the proposed framework was conducted to inspect the safety, efficiency, and smoothness of the vehicle, which follows the evaluation process in the work [12]. Fig. 2.6 shows the results of combinations of $d_{front,0} = 21.5$ and all vehicle initial/desired speeds $\dot{s}_0$. The variance of the results corresponding to particular τ_{gap}s was due to the variance of the pedestrian's desired speed v_0.

Safety

The minimum distance between the pedestrian and the vehicle during an entire episode is used to evaluate pedestrian safety. According to all the plots in the 1st row, if the threshold of τ_{gap} was larger than the gap estimated at the time when the vehicle was initialized, most likely the pedestrian decided to wait and yield to the vehicle. In these cases, the minimum distances were almost the same, which is approximately equal to the distance from the road edge to the vehicle's right side. For the cases of pedestrian not

yielding, the OAC was comparatively safer than the other two (but at the expense of smoothness). Note that for the VKC, collision happened in some cases when the vehicle's initial/desired speed $v_{0,veh}$ is relatively high.

Efficiency

The average vehicle velocity is used to evaluate the driving efficiency. The VKC had the highest efficiency because it didn't react to the pedestrian at all, but the collision is unacceptable. Comparing the OAC with the MPC, the OAC outperformed the MPC at high speed $v_{0,veh}$, but was outperformed by the MPC as the $v_{0,veh}$ decreases.

Smoothness

The minimum value of the absolute acceleration/deceleration is used to evaluate the smoothness. Similarly to the efficiency, the VKC maintained the same minimum acceleration but collision happened. The MPC was always smoother than the OAC in general.

In sum, considering safety, efficiency, and smoothness simultaneously, when the vehicle speed $v_{0,veh}$ is relatively low, the MPC outperformed the OAC and the VKC. This is because the MPC predicts the future vehicle motion so that the control action is optimized to consider both the safety, efficiency, and smoothness. And when $v_{0,veh}$ is low, the MPC has more allowed time to optimize its control action. The results associated with other $d_{front,0}$ also follow a similar pattern as illustrated in Fig. 2.6.

2.7 Conclusion

This Chapter proposed a framework to address the VPI in uncontrolled crossing scenarios. A novel multi-state social force pedestrian motion model was integrated into the framework. The behavior of both the vehicle and the pedestrian in the experiments of 3 different vehicle control strategies demonstrated the effectiveness of the proposed framework.

Major further work would be improving the pedestrian model by considering demographics and by leveraging VPI datasets. Framework components such as pedestrian motion predictor could also be replaced with a more accurate one. In terms of vehicle control, the next step would be better integrating pedestrian uncertainty into the controllers. And lastly, of course, the framework should be extended to cover more VPI scenarios.

Chapter 3

On the Generalizability of Motion Models for Road Users in Heterogeneous Shared Traffic Spaces

This chapter is derived from the published work in [22].

3.1 Introduction

Shared space design principles [23] have been drawing significant attention in recent years, as an alternative to traditional regulated traffic designs. In shared spaces, heterogeneous road users such as pedestrians, cars, bicycles share the same space. Unlike traditional traffic environments, in shared spaces, there are no or very few road signs, signals, and markings; this causes frequent direct interactions among road users to coordinate their trajectories. There is an ongoing debate on the safeness of shared spaces; while some studies state that the lack of explicit traffic regulations makes road users more safety-conscious and may lead to fewer road accidents [24, 23, 25], others ([26, 27]) argue the lack of acceptance and understanding of the concept can compromise safety in shared spaces. Notwithstanding this debate, traditional road designs have been replaced by shared spaces in a growing number of urban areas; some examples are the Laweiplein intersection in Drachten, Skvallertorget in Norrköping, and Kensington High Street in London [24].

Yet, the lack of explicit rules makes it essential to investigate the safety issues in shared spaces. Modeling and simulation shared spaces by analyzing and reproducing the motion behaviors of road users including their interactions is crucial to assess and optimize such spaces during the planning phase. Realistic simulation models can also form a safe basis for autonomous cars to learn how to interact with other road users.

Interpreting and modeling mixed-traffic interactions pose challenging problems; an interaction can be a simple reaction or a result of complex human decision-making processes, i.e., modifying speed or direction by predicting other road users' behavior, or communicating with them [5]. Moreover, how one interacts with others is dependent on many factors like their transport mode, current situation, road structures and conditions, social norms (culture), and many individual factors (e.g. age, gender, or time pressure [25]).

To the best of our knowledge, so far, there are not many works on modeling and

simulation of shared spaces. We observe mostly two different state-of-the-art approaches: (1) physics-based models, mainly the social force model (SFM) of pedestrian dynamics [15] including numerous extensions adding, e.g., new forces, decision-theoretic concepts, or rule-based constraints, to describe different types of actors such as cars [28, 29] or bicycles [30]; and (2) cellular Automata (CA) models [31, 32, 33], which are mainly used for modeling mixed-traffic flows in settings with explicit traffic regulations, unlike most shared spaces.

Although these approaches perform well for single bilateral conflicts (i.e., for any point in time, a road user can only handle a single explicit conflict with one other user), they fail in representing multiple conflicts among heterogeneous road users and groups, which are very common in shared spaces. Hence, in our previous works, we integrated SFM with a game-theoretic model to address both bilateral and multiple conflicts among pedestrians and cars [34, 35]. In this chapter, we describe *conflict* as "an observable situation in which two or more road users approach each other in time and space to such an extent that there is a risk of collision if their movements remain unchanged" as specified in [36]; here, we use the terms *conflict* and *interaction* interchangeably.

In the literature, motion models do not adequately consider the differences in road users' behaviors induced by differing environmental settings. These models are usually calibrated and validated using scenarios from a single shared space environment. In [37], we took a first step to address this gap by proposing the concept of zone-specific motion behaviors for pedestrians and cars, considering road and intersection zones. In [38], we evaluated the transferability[1] of our existing model by modeling scenarios that differ from the one used in [37] in terms of traffic conditions, spatial layout and social norms. Subsequent results show that our model can suitably replicate the motion of pedestrians and cars from the new scenarios.

In this chapter, we delve further into this direction by proposing a conceptually systematic and simple process of modeling general motion models and output a moderate version of a general motion model for pedestrians and cars, by following our proposed modeling process. A general model should be able to reproduce a large variety of motion behaviors of heterogeneous road users ranging from simple free-flow motions to resulted-motions from complex interactions and transferable to new environments with minimal time and effort. The differences between the current work and our previous work ([38]) in terms of model transferability are: (1) In this chapter, we build a general model to capture motion behaviors from three data sets with incremental integration of new motion behaviors, and a well-defined and largely automated calibration process to adapt model parameters to the target environment. Whereas in [38], as we did not have any specific process to generate a general motion model, to adapt to the new environment, we had to analyze, consider and explicitly change our model parameters and methods based on the social norms of that new environment, which resulted in different versions of our model, i.e. each version for each different environment. (2) In the current work, the transferability of our model is evaluated using the DUT and HBS data sets as in [38] and also by a new data set (CITR) that contains unique conflict scenarios than the other two data sets (see Section 3.4).

We further introduce heterogeneity in pedestrian motion by recognizing different motion patterns, by calibrating individual motion characteristics (e.g., sensitivity when interacting with others) and clustering them into different groups [2] (see Section 3.6). The

[1] We use the terms *transferability* and *generalizability* interchangeably

[2] In this chapter, the keyword **group** is used to represent a set of pedestrians with a similar motion pattern, not the social group, e.g., family members.

contributions of this chapter are:

- We propose a systematic process to formulate a general motion model.
- We propose a motion model for pedestrians and cars, which can simulate a large variety of conflict scenarios among road users and evaluate the generalizability of our model by using three different shared space data sets. The results of our evaluation process indicate that our model achieves satisfactory performance for each data set.
- We present a methodology to recognize and model different motion patterns of pedestrians from real-world data sets. To do so, we investigate several approaches to cluster pedestrians with similar motion patterns into groups. Our evaluation results show that the heterogeneity in pedestrians motion improves the model performance.

Following a review of previous research in Section 3.2, we propose the formulation of a general model for movement modeling of heterogeneous road users in Section 3.3. We illustrate the examined data sets and the architecture of our Game-Theoretic Social Force Model (GSFM) in Section 3.4 and Section 3.5, respectively. Section 3.6 explains the calibration methodology and recognition of different walking styles of pedestrians. In Section 3.7, we describe how we evaluate model performance and discuss the results. We conclude by outlining future research venues.

3.2 Related Works

Existing mixed-traffic motion models are mostly built based on rule-based models (e.g. Cellular Automata (CA) [32]), or physics-based models, most preeminently the Social Force Model (SFM) [15].

CA models describe road users motion behavior by a set of state transforming rules in a discrete environment. They have been used to model motion behaviors of a set of homogeneous road users, e.g., pedestrians [39, 40], cars [41, 42] and there are also few works describing mixed-traffic motion, e.g., [32] who study interactions among pedestrians and cars at crosswalks, [31] who model car-following and lane-changing actions of cars and motorcycles, or [43] who study bicycle-to-vehicle interactions and its impact on traffic delay.

In the classical SFM, introduced in [15], the movement of a pedestrian is represented by differential equations comprising a set of simple attractive, and repulsive forces from other pedestrians and static obstacles that he/she experiences at a specific place and time. Even though SFM was initially modeled for pedestrian dynamics [44, 45, 46], many studies extended it for modeling other types of road users. For example, [17, 47] who include vehicles, considering their impact on pedestrians as separate forces; in [29], Anvari et al. add new forces and rule-based constraints to handle short-range and long-range conflicts among pedestrians and cars. In [30], SFM is combined with long-range collision avoidance mechanisms to model motion behaviors of pedestrians, vehicles and bicycles.

Both CA-based and SFM-based models can represent simple situations well. However, game-theoretic or probabilistic models are more suitable for complex scenarios where road users must choose an action among many alternatives to handle a given situation [15]. In [48], in case of complex interactions, road users' choice of action is modeled by a logit model, based on available data but without considering what other users might do. In

[49], Fujii et al. used a discrete choice model to illustrate decision making while in pedestrian interactions. Game-theoretic models have often been applied to interpret human decision-making processes, also in traffic situations. Some examples are the application of non-cooperative games to illustrate merging-give way interaction among vehicles ([50]), pedestrian-to-car interaction in shared spaces ([28]), bicyclist-to-car interaction at zebra crossings ([51]), or analyze the difference of cyclist/pedestrian interaction with human-driven or autonomous vehicles in [52]. In [53], lane-changing behaviors of cars are modeled using a cooperative game where cars cooperate with each other for collective reward. Whereas, in a non-cooperative game, each player makes decisions by predicting others' decisions, which is very similar to what real-world road users often do [51].

Although there are several works on modeling motion behavior of road users, only a very few studies consider different motion patterns for individual road user types [54, 55, 56]. Kabtoul et al. [54] manually annotates several predefined pedestrian types based on willingness to give way to a car. Alahi et al. [55] obtain different movement styles for pedestrians by learning collision avoidance parameters of individual pedestrians and clustering them into groups using the k-means clustering. Their model is restricted to pedestrian-only scenarios. In [56], the authors classified pedestrians into groups based on their age range and gender and assigned individual speed profiles to each group. These speed profiles are collected from the literature instead of real-world data sets.

Existing closed-source commercial (e.g., AIMSUN [57] or VISSIM [58]) and open-source (SUMO [59]) simulators are somewhat capable of modeling and simulating mixed-traffic at a microscopic level. However, open-source simulators like SUMO have limited means for modeling interaction between heterogeneous road users. To address this issue, some studies combined SUMO with agent frameworks such as JADE ([60]) or JASON ([61]); however, adding new environmental features or define new modalities in such models is difficult. Also, SUMO lacks flexibility regarding lane and vehicle geometries, which is restrictive for shared spaces.

3.3 Modeling Process

A general motion model should be able to reproduce realistic motion behaviors of road users in different environmental settings in terms of road structures, culture or norm, types of road users, and types of interactions and to adapt to new environments with less time and effort, which make generating such models very challenging.

We propose a systematic process to construct a general motion model in Figure 3.1. Here, D, A, and M represents the decision, action and merge nodes respectively. The process starts with modeling the free-flow movements of road users (**A1**) with their type and origin, destination, and speed profiles as *input*. The next step is to analyze and model interactions among road users. To do so, one can collect and explore a real-world traffic data set (**A2**) to identify and extract conflict scenarios between two or more road users (**A3**) to recognize and classify the interactions among the road users (**A4**) and then model these interactions (**A5**). Finally, the model needs to be calibrated (**A6**) and evaluated (**A7**) both quantitatively (minimize the difference between real and generated trajectories) and qualitatively (reproduce realistic behaviors) by using these extracted conflict scenarios. However, generating a general motion model is a continuous process which requires testing the model with new data sets, i.e., new environments and also adding new modalities. As shown in Figure 3.1, to evaluate the model performance on a new (**D1**)

data set, it is necessary to check (**D2**) if there are any new kind of interaction(s), if yes, then this interaction(s) needs to be integrated (**A5**) into the model. Next, the calibration of all parameters (including the new ones) and the model evaluation on each data set is required. To add a new user type (**M1**) e.g., integrating vehicle in the pedestrian-only motion model, one needs to go through all the steps in Figure 3.1. This iterative process of modeling continues until a stopping criterion, such as a certain level of accuracy in realistic trajectory modeling, has been reached. The stopping criterion is application dependent.

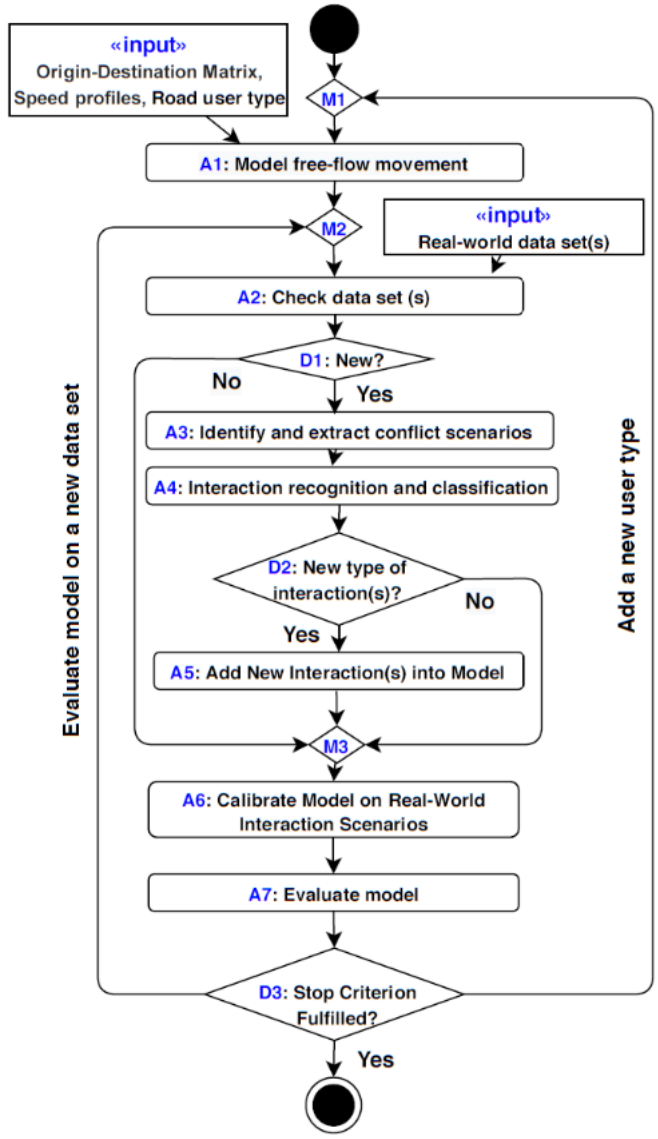

Figure 3.1: Formulation of a general motion model for mixed-traffic environments.

In this chapter, we use this process to *output* a moderate version of a general model for generating realistic trajectories of pedestrians and cars in different shared spaces, using the HBS, DUT and CITR data sets. Our way of recognizing and classification of interactions (**A4**), modeling these interactions (**A5**), the calibration (**A6**) and evaluation (**A7**) of the model are discussed in Section 3.4.2, 3.5, 3.6 and 3.7, respectively.

3.4 Data Sets and Interaction Classifications

3.4.1 Data Sets

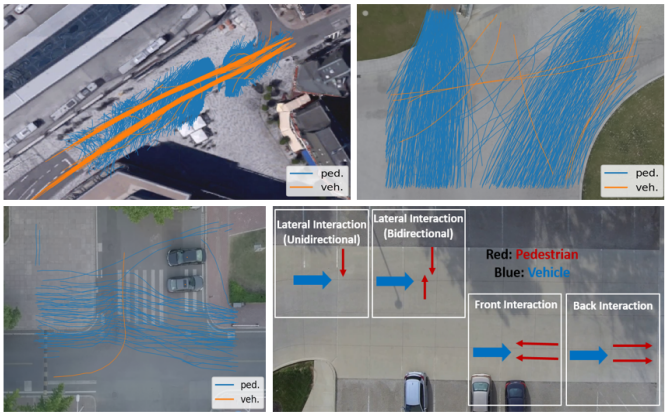

Figure 3.2: The spatial layout of three shared space environments; the top-left sub-figure visualizes the shared street from HBS, the top-right sub-plot shows the roundabout from DUT, the bottom-left sub-plot depicts the intersection from DUT and the bottom-right sub-figure shows interactions from CITR.

We have been developing a motion model of pedestrians and cars, named Game-Theoretic Social Force Model (GSFM) [34, 37, 62], mainly based on the scenarios manually extracted from a street-like shared space environment in Hamburg, Germany (HBS). In this chapter, to move towards a general model, we evaluate our model on two other data sets which are different from the HBS data set in terms of spatial structures, types of interactions, and the number of road users. These data sets are the DUT data set from Dalian University of Technology campus in China and the CITR data set from the Ohio State University campus in the USA. All three data sets are visualized in Figure 3.2, and their details are given below:

- **HBS** [30]: The HBS data set collected from a street with pedestrian crossing from both sides. It contains both bilateral and multilateral interactions among pedestrians and cars, with or without car following interactions. We extracted 103 such scenarios from HBS.
- **DUT** [63]: The DUT data set contains trajectories of pedestrians and cars from a roundabout and an intersection. It comprises of car-to-crowd lateral interactions; most scenarios extracted from DUT have a large number of pedestrians compared to the HBS and CITR scenarios.
- **CITR** [63]: CITR is an experimentally designed data set, collected from a university parking lot. It contains several lateral, front, and back interactions among pedestrians and cars.

Here as shown in the bottom-right sub-figure of Figure 3.2, lateral interaction indicates the situation where pedestrian(s) cross from in front or behind the car. Front interaction is the face-to-face interaction, and in back interaction scenario, car drives behind the pedestrian(s). There are also observable differences in these data sets which can be interpreted

as cultural differences. For example, in the DUT data set, road users maintain less inter-distance (i.e., safety distance) compared to the HBS and CITR data sets (see Section 3.6). In all three data sets, an agent's position at each time step (i.e., 0.5 s) is given as a 2D vector in the pixel coordinate system, and they also contain the pixel-to-meter conversion scales. Table 3.1 summarizes the number of scenarios and individuals involved.

Table 3.1: Statistics of Datasets

Data set	# of Scenarios	# of Pedestrians	# of Cars	Time step
HBS	103	206	126	0.5s
CITR	26	208	26	0.5s
DUT	30	607	39	0.5s

3.4.2 Interaction Classification

In our previous works [34, 62], we classified road users interactions broadly into two categories based on Helbing's classification of road agents' behavior [15] and the observation of the shared space video data (mostly HBS): *simple interaction* (percept → act) and *complex interaction* (percept → choose an action from different alternatives → act). These interactions can also be sub-categorized based on the number and types of road users involved: *simple interaction* contains car-following, pedestrian-to-pedestrian, and pedestrian(s)-to-car reactive interactions and *complex interaction* includes pedestrian(s)-to-cars, pedestrians-to-car and car-to-car interactions. We note that complex car-to-car interaction is not included in this chapter.

As mentioned earlier, in this chapter, we are still focusing on pedestrians and cars, but we aim to evaluate the performance of our model on the DUT and CITR data sets. According to the process proposed in Figure 3.1, we analyze these two data sets and detect the following new types of interactions:

- Unlike HBS, in the DUT data set, sometimes, cars somewhat deviate from their trajectory as a result of reactive interaction with pedestrians. Mostly because of the environment structure in DUT, i.e., more free space for motion of cars.

- As already discussed in Section 3.4.1, the CITR data set [63] contains front and back interactions among pedestrians and cars, which are not observed in the HBS or DUT data sets [30].

How we model these interactions, including integration of new interaction types, is described in Section 3.5.

3.5 Agent-Based Simulation Model

We pursue an agent-based model, GSFM, to represent the motion behaviors of pedestrians and cars, initially described in [34]. Here, we give an overview of the architecture of GSFM, visualized in Figure 3.3. In GSFM, each road users is modeled as an individual *agent* and their movements are conducted in three interacting modules, namely, *trajectory planning*, *force-based modeling*, and *game-theoretic decision-making*. Each of this module has individual roles. GSFM is implemented on a BDI (**B**elief, **D**esire, **I**ntention) platform, LightJason [64], which permits flexible design and explanation of the control flow of GSFM

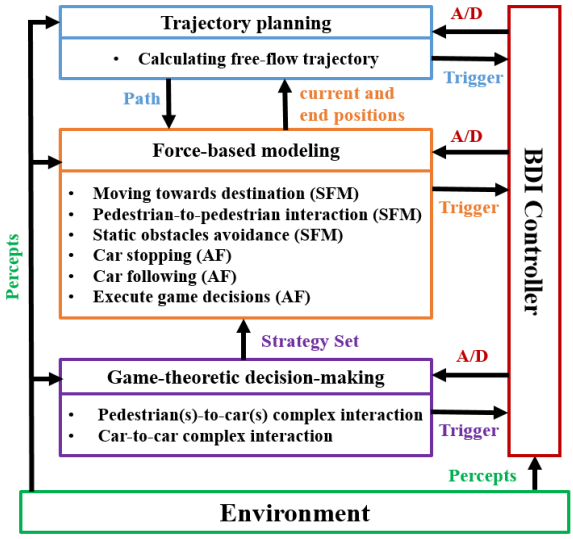

Figure 3.3: Conceptual model of pedestrians and cars motion behaviors. Here, **AF** denotes the added force to classical SFM and **A/D** signifies activation/deactivation of a module.

through its three modules. Based on current situation, the *BDI controller* activates the relevant module, which then informs the controller on completion of its task.

The trajectory planning module computes the free-flow trajectories for all agents by only considering static obstacles (e.g. boundaries, or trees) in the environment. For individuals trajectory planning, we transform the simulation environment into a visibility graph [65], add their origin and destination positions into the graph and perform the A* algorithm [66].

The force-based module governs the actual execution of an agent's physical movement and also captures the simple interactions between agents by using and extending the SFM. To model the driving force of agents towards their destination ($\vec{D}_i^o$), the repulsive force from the static obstacles ($\vec{I}_{iW}$) and other agents ($\vec{I}_{ij}$), we use the classical SFM. Here, $\vec{D}_i^o = \frac{\vec{v^*}_i(t) - \vec{v}_i(t)}{\tau}$ for a relaxation time τ and $\vec{v^*}_i(t)$ and $\vec{v}_i(t)$ denote the desired and current velocities of i, $\vec{I}_{ij} = V_{ij}^o \exp\left[\frac{r_i + r_j - \vec{d}_{ij}(t)}{\sigma}\right] \hat{n}_{ij} F_{ij}$ and $\vec{I}_{iW} = U_{iW}^o \exp\left[\frac{r_i - \vec{d}_{iW}(t)}{\gamma}\right] \hat{n}_{iW}$, where V_{ij}^o and U_{iW}^o symbolize the interaction strengths, and σ and γ are the range of these repulsive interactions, $\vec{d}_{ij}(t)$ and $\vec{d}_{iW}(t)$ are the distances from i to j, or i to W at a specific time, $\hat{n}_{ij}$ and $\hat{n}_{iW}$ indicate the normalized vectors. $F_{ij} = \lambda + (1-\lambda)\frac{1+\cos {}_{ij}}{2}$ describes the fact that human are mostly affected by the objects which are within their field of view [67]. Here, λ stands for the strength of interactions from behind and ${}_{ij}$ symbolizes the angle between i and j. Additionally, we extend SFM to represent car following interaction ($\vec{I}_{\text{follow}}$) and pedestrian-to-car reactive interaction ($\vec{I}_{\text{stop}}$). If $\vec{d}_{ij}(t) \geq D_{min}$, $\vec{I}_{\text{follow}} = \hat{n}_{p_i x_i(t)}$, i.e., i continues moving towards $p_i = x_i(t) + \hat{v}_j(t) * D_{min}$, otherwise, i decelerates. Here, D_{min} is the minimum vehicle distance, $\hat{v}_j(t)$ is the normalized velocity of j, and $\vec{d}_{ij}(t)$ denotes the distance between i and j (the leader car). $\vec{I}_{\text{stop}}$ emerges only if pedestrian(s) have already begun walking in-front of the car. Then the car decelerates to let the pedestrian(s) proceed. This module also executes the decisions computed in the game module $\vec{I}_{\text{game}}$.

As discussed in Section 3.4.2, the CITR data set contains two new types of interaction, namely, the front and back interaction between pedestrian (i) and vehicle (j). We incor-

porate these two interactions to our model as a single type, i.e., longitudinal interaction, $\vec{I}_{long}$ and following:

If $\vec{d}_{ij}(t) < D_{min}^{long}$ and (C_1 or (C_2 and C_3)), we add a temporary goal $p_i = \vec{x}_i(t) + R_f$ for the respective pedestrian, where C_1, C_2, and C_3 are symbolized in Eq. (3.1) with $g = \vec{e}_i \cdot \vec{e}_j$, i.e., the dot product of the direction vectors of i and j, and R_f is the rotation of $f = \vec{e}_j * c$ using rotation theory in Eq. (3.2) [68] and the calculation of c and θ are given in Eq. (3.4) and Eq. (3.3) respectively. Thus, $\vec{I}_{long} = \hat{n}_{p_i} x_i(t)$, i.e., i continues moving towards p_i to avoid conflict.

$$\begin{aligned} C_1 &= \theta_{\vec{e}_j \hat{n}_{ji}} < 2^\circ \; or \; \theta_{\vec{e}_j \hat{n}_{ji}} > 358^\circ \\ C_2 &= g \geq 0.99 \; or \; g \leq -0.99 \\ C_3 &= \theta_{\vec{e}_j \hat{n}_{ji}} \geq 348^\circ \; or \; \theta_{\vec{e}_j \hat{n}_{ji}} \leq 12^\circ \end{aligned} \tag{3.1}$$

$$\begin{aligned} f_{x_2} &= \cos\theta f_x - \sin\theta f_y \\ f_{y_2} &= \sin\theta f_x + \cos\theta f_y \end{aligned} \tag{3.2}$$

$$\theta = \begin{cases} 90^\circ, & if\, \theta_{\vec{e}_j \hat{n}_{ji}} \geq 348^\circ \\ 180^\circ, & \text{otherwise} \end{cases} \tag{3.3}$$

$$b = \begin{cases} 1, & if\, g \leq -0.99 \\ 1.5, & \text{otherwise} \end{cases} \quad c = \begin{cases} 3 * b, & if\, \theta_{\vec{e}_j \hat{n}_{ji}} \geq 348^\circ \\ 2.2 * b, & \text{otherwise} \end{cases} \tag{3.4}$$

In this chapter, D_{min}^{long} is set to 10 m. Deviation of cars due to reactive interaction with pedestrian in the DUT scenarios is addressed by $\vec{I}_{ij}$, i.e., the SFM repulsive force.

The game-theoretic module controls the complex interactions among agents, e.g. pedestrians-to-cars interaction, using Stackelberg game, i.e., a sequential leader-follower game. In a Stackelberg game, first, the leader decides on a strategy that maximizes its utility by predicting all possible reactions of followers and then, the follower reacts by choosing its best response [28]. The game is solved by finding the sub-game perfect Nash equilibrium (SPNE) i.e., the optimal strategy pair. The Eq. (3.5) and Eq. (3.6) depict the SPNE and the best response of the follower, respectively. Here, s_l, s_f, u_l, u_f and S_l, S_f are the leader's and follower's strategies, utilities of the corresponding strategies and their strategy sets, respectively.

$$\text{SPNE} = \{s_l \in S_l | max(u_l(s_l, Bs_f(s_l)))\}, \forall s_l \in S_l. \tag{3.5}$$

$$Bs_f(s_l) = \{s_f \in S_f | max(u_f(s_f | s_l))\}. \tag{3.6}$$

An individual game manages each complex interaction, and the games are independent on each other. In each game, the number of leaders is fixed to one but the followers can be more. We perform separate experiments with randomly chosen leader, the faster agent as leader (i.e., the car), and pedestrian as a leader. The result suggests that and the faster agent as leader is the best choice. However, if the scenario includes more than one car (e.g., pedestrian-to-cars interaction), then the one who recognizes the conflict first is considered as the leader. To calculate the payoff matrix of the game, as shown in Figure 3.4, first, all actions of the players are ordinally valued, assuming that they prefer to reach their destination safely and promptly. Then, to express situation dynamics, we select several features by analyzing real-world situations and perform a backward elimination process on the selected features to get the most relevant ones. Let, i be an agent which interacts with another agent j; then the relevant features are the following:

- **NOAI**: the number of active interactions of i as a car.
- **CarStopped**: has value 1 if i (as a car) already stopping to give way to another agent j', otherwise 0.
- **MinDist**: has value G_{dis}^{min}- distance(i, j), if distance(i, j) $< G_{dis}^{min}$; its difficult to stop for car i, otherwise 0.
- **CompetitorSpeed**: has value 1, if current speed of j, $S_{current} < S_{normal}$, otherwise 0.
- **OwnSpeed**:

$$\begin{cases} S_{current}, & if\ i\ is\ a\ car \\ 1, & if\ i\ is\ a\ pedestrian\ and\ S_{current} > S_{high} \\ 0, & \text{otherwise} \end{cases}$$

- **Angle**:

$$\begin{cases} 8, & if(\theta_{\vec{e}_j\hat{n}_{ij}} < 16^\circ\ and\ \geq 0^\circ)\, \theta_{\vec{e}_j\hat{n}_{ij}} > 344 \\ 7, & \text{if}(\theta_{\vec{e}_j\hat{n}_{ij}} \leq 42^\circ\ and\ \geq 16^\circ)\ (\theta_{\vec{e}_j\hat{n}_{ij}} \leq 344^\circ\ and\ \geq 318^\circ) \\ 6, & \text{if}(\theta_{\vec{e}_j\hat{n}_{ij}} \leq 65^\circ\ and\ > 42^\circ)\ (\theta_{\vec{e}_j\hat{n}_{ij}} < 318^\circ\ and\ \geq 295^\circ) \\ 5, & \text{if}\,(\theta_{\vec{e}_j\hat{n}_{ij}} \leq 90^\circ\ and\ > 65^\circ)\,(\theta_{\vec{e}_j\hat{n}_{ij}} < 295^\circ\ and\ \geq 270^\circ) \\ 1, & \text{otherwise} \end{cases}$$

During game playing, *Continue*, *Decelerate* and *Deviate* (only for pedestrian) are the viable actions for road users. Execution of these actions are performed in the force-based module.

- Continue: Any pedestrian i crosses a car j from the point $p_i = x_j(t) + S_A * \overrightarrow{e}_j$ if $line(x_i(t), x_i^{des})$ intersects $line(x_j(t) + S_A * \overrightarrow{e}_j, x_j(t) - \frac{S_A}{2} * \overrightarrow{e}_j)$, otherwise continues her free-flow motion. Here, $\overrightarrow{e}$ is the direction vector, S_A is a scaling factor, $x(t)$ and x_i^{des} are the current and final positions respectively. Cars continue by following their free-flow motion.
- Decelerate: Agents decelerate and in the end stop, if required. For pedestrians, $\text{newSpeed}_i = \frac{\text{Speed}_i(t)}{2}$, unless the car is very near (i.e., distance(i, j) $\leq r_i + r_j + 1$ m), in that case pedestrian will stop and in case of cars, $\text{newSpeed}_j = \text{Speed}_j(t) - \text{decRate}$.

  Here, $\text{decRate} = \begin{cases} \frac{\text{Speed}_j(t)}{2}, \text{if distance}(i,j) \leq D_{min}, \\ \frac{\text{Speed}_j^2}{\text{distance}(i,j) - D_{min}}, \text{otherwise.} \end{cases}$

  D_{min} is the critical spatial distance.
- Deviate: A pedestrian i passes a car j from behind from a position $p_i = x_j(t) - S_D * \overrightarrow{e}_j$ (up till j stays within the range of view of i) and afterwards i proceeds moving towards her original goal position.

Car \ Ped	Continue	Decelerate	Deviate
Continue	-100, -100	C_c , P_d	C_C , Pc_{dev}
Decelerate	C_D , 4	-50, -50	C_D , Pd_{dev}

(a) Pedestrian-to-Car Interaction

C_c = - $G_{speed}^{competitor}$ * CompetitorSpeed + G_{speed}^{C} * OwnSpeed + MinDist - G_{angle}^{F} * (Angle >= G_{angle}^{Ace} ? Angle : 0)

C_d = $G_{stopped}$ * CarStopped + G_{NOAI} * NOAI + G_{angle}^{F} * (Angle >= G_{angle}^{Dec} ? Angle : 0)

P_d = 3 - G_{speed}^{P} * OwnSpeed

Pc_{dev} = 2 + G_{speed}^{P} * OwnSpeed + (Angle <= 6 ? G_{angle}^{Dev} - Angle : 0)

Pd_{dev} = Angle <= 6 ? G_{angle}^{Dev} - Angle : 0

(b) Impacts of Situation Dynamics

Figure 3.4: The complete payoff matrices for pedestrian-to-car interactions.

Although these modules do not obey any sequence and take control alternatively, at the start of the simulation, GSFM keeps a hierarchy among them. It starts with trajectory planning, assuming that agents plan their trajectories before they begin moving. When trajectories are planned, the BDI controller actives the force-based module to model the physical movement of agents. Conflict recognition and classification are performed at regular intervals (the algorithm is given in [37]), and if it detects any complex conflict, then the controller activates the game-based module. As soon as the strategies are decided, the controller activates the force-based module again to execute the chosen strategies. The BDI controller also prioritizes different interactions based on their seriousness, for example, for cars, $\vec{I}_{\text{stop}}$ takes precedence over $\vec{I}_{\text{game}}$ and $\vec{I}_{\text{game}}$ obtains priority over car following. The following code fragment depicts the basic elements of a BDI program consisting of beliefs (in pink), plans (in blue), and actions (in black).

: Sample BDI Code Fragment

```
module(3.0).
!main.
+!main <-
    generic/print("Name", MyName);
    !!calculate/route; !walk.

+!calculate/route <-
    route/calculate.

+!walk: >>(module(S), generic/type/isnumeric(S)
    && S==3.0) <-
    calculate/next/position; !walk.

+!update/belief(G) <-
    >>module(S); -module(S); +module(G).

+!game/decelerate: >> (module(S), general/type/
    isnumeric(S) && S==1.0) <-
    stop/moving; !game/decelerate; !walk.
```

Here, '+', '-', '>>' signify add (plan or belief), remove (belief) and unification (belief), respectively. The double exclamation mark before *calculate/route* plan indicates that this plan should run in the current time-step and one exclamation mark before *walk* says that the plan will execute in the next time-step. An agent can also trigger a plan from the environment. As an example, when the game module decides on the strategies for the road users involved in a conflict situation, it triggers the plan *update/belief*, and the plan related to the decision, i.e., *game/decelerate* in this sample (not complete) code fragment.

$$\text{Pedestrian: } \frac{d\overrightarrow{v^t}_i}{dt} = \left(\overrightarrow{D}_i^o + \Sigma\overrightarrow{I}_{iW} + \Sigma\overrightarrow{I}_{ij} + w_p \cdot \overrightarrow{I}_{long}\right) or \overrightarrow{I}_{\text{game}} \tag{3.7}$$

$$\text{Car: } \frac{d\overrightarrow{v^t}_i}{dt} = \left(\overrightarrow{D}_i^o + w_c \cdot \sum_{j \neq car} \overrightarrow{I}_{ij}\right) or \overrightarrow{I}_{\text{follow}} or \overrightarrow{I}_{\text{game}} or \overrightarrow{I}_{\text{stop}}, \tag{3.8}$$

$$\hat{Y}_i^{t+\Delta t} = f\{Z_i, (\frac{d\overrightarrow{v^t}_i}{dt} + x_i(t))\}. \tag{3.9}$$

The process of modeling the movements of any agent i at any time step t in GSFM is summarized in Eq. (3.7)–(3.9). Here, i, j, W, Z_i, $x_i(t)$, and $\hat{Y}_i^{t+\Delta t}$ denote the target agent, competitive agent, static obstacle, model inputs, the position of i in current and next time step respectively. The input profile Z_i contains start (x_i^{st}), goal (x_i^{des}), and speed profile of i. The goal of i is estimated by extending its last observed position (x_i^{gt}) in real trajectory using Eq. (3.10) with the extended length $l_{\text{des}} = 5$ m. The weight $w_p = 1$ for the CITR scenarios, otherwise 0 and $w_c = 1$ for the DUT scenarios, otherwise 0.

$$x_i^{des} = x_i^{st} + l_{\text{des}} \cdot (x_i^{gt} - x_i^{st}), \tag{3.10}$$

We calculate the desired speed v_{d} of a pedestrian by identifying the walking portion of his/her trajectory, i.e., where the pedestrian's speed is larger than a threshold v_{walk} and then, we average all the speed values to obtain v_{d}. We set $v_{\text{walk}} = 0.8m/s$. A car's desired speed is set to: $mean(v_{\text{i}}) + std(v_{\text{i}}) * 0.5$, where v_{i} is the set of all the speed values of car i.

3.6 Calibration Methodology

In this chapter, we calibrate our model parameters in several steps as visualized in Figure 3.5 and the calibration is performed using a genetic algorithm (see section 3.6.2). To recognize different motion patterns of pedestrians from real-world scenarios, we investigate two clustering approaches, namely Principal Component Analysis (**PCA**) with the k-means algorithm (step **S3**), and k-means with step-wise forward selection (**FS**) method (steps **S4** and **S6**), see section 3.6.1. The steps in Figure 3.5 are as follows: We start by performing universal calibration to get one unique set of parameter values for all pedestrians by assuming that in the same situation, they all act similarly.

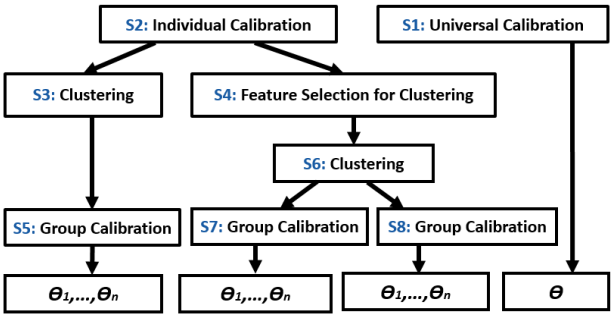

Figure 3.5: The workflow of model calibration.

At the next step, we calibrate the parameters individually for each pedestrian, then cluster individual parameters using the above-mentioned clustering approaches which give us two different sets of pedestrian groups. Next, we perform group calibration (steps: **S5**, **S7** and **S8**) so that each group has a unique set of parameters values. For the groups (i.e., clusters) that are obtained in step **S3**, we perform group calibration directly. However, for the groups obtained by completing **S4** and **S6**, we perform group calibration in two different phases i.e., **S7** and **S8**. In **S7**, we individually calibrate the selected parameters by the FS method for each group, while keeping the rest of the parameters' values (obtained in **S1**) same for all groups. Whereas in **S8**, we calibrate all parameters separately for each group. Each of these approaches above generates a different version of the GSFM model (see section 3.6.2).

GSFM contains a large set of parameters, which can be broadly classified into parameters for SFM interaction, safety measurements, and payoff matrix calculation for game playing. The SFM and safety-related parameters are listed in Table 3.4 and Table 3.3 shows the game parameters. Among these parameters, for grouping pedestrians, we select the sets of parameters given in Table 3.2 based on sensitivity analysis. The rest of the parameters are calibrated universally as step **S1**.

Table 3.2: The list of parameters calibrated for clustering

Interaction strength: V_{ij}^{o} (PP), V_{ij}^{o} (PC), V_{ij}^{o} (CP),
Repulsive interaction range: σ (PP), σ (PC),
Anisotropic parameter: λ, Scaling factor for deviate action: S_D

3.6.1 Clustering

K-means with Principal Component Analysis **K-means** is a simple, fast and widely used clustering algorithm for classifying data based on euclidean distance between the data points, with a predefined number of clusters [69]. In this chapter, we decide on the number of clusters using the elbow method [69], and each data point represents the calibrated parameters' values of an individual pedestrian.

Principal Component Analysis [69] is a technique that reduces a larger number of parameters to a smaller set of parameters which are linear combinations of the original parameters and contains most of their information. As stated in [70], reducing the dimension of data using PCA is beneficial for k-means. Thus, we use PCA to reduce the number of parameters given in Table 3.2, and then perform k-means on the reduced parameters

set to cluster pedestrians into groups.

K-means with Forward Selection **Forward selection** is a simple but commonly used feature (or parameter) selection method. It starts with a empty model which contains no parameters, then continue adding the most significant parameter one after another until a predefined stopping criteria has reached or if all present parameters are already in the model [71].

Algorithm 1 Forward Selection with k-means

Number of clusters K, Set of parameters A_p, Predefined score C_s Set of selected parameters for clustering S_p $S_p \leftarrow \{\}$ $Cs_{int} = 0$*initialize clustering score to 0 $temp_s = 0$ $temp_p \leftarrow \{\}$

while $Cs_{int} < C_s$ **do**

for each $p \in A_p$ **do**

if $S_p == \emptyset$ **then** perform k-means clustering for p

else $t_p = S_p \cup \{p\}$ perform k-means clustering for t_p $score$ = silhouette score of clusters

if $temp\_s < score$ **then** $temp_s = score$ $temp_p = p$ $S_p = S_p \cup \{temp_p\}$ $A_p = A_p \setminus \{temp_p\}$ $Cs_{int} = temp_p$

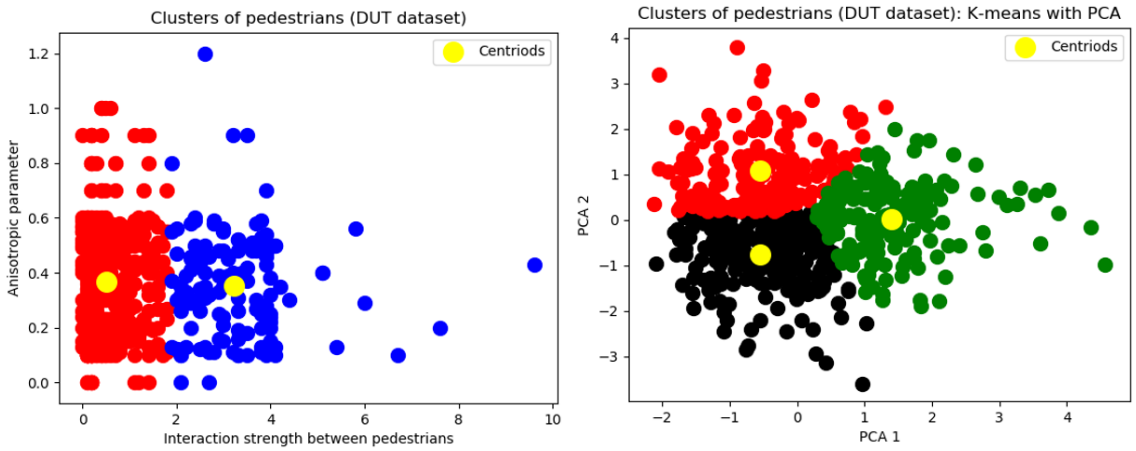

Figure 3.6: Different pedestrians groups of the DUT data set with different motion patterns.

We calculate the significance of the parameter(s) by executing k-means for some k (i.e., number of clusters) and measure the clustering performance using the silhouette score. This method terminates if a preset value of silhouette score has been reached. The silhouette value is a measure to see if a data point is similar to its own cluster than to others [72]. Algorithm 1 shows the steps of the forward selection method with k-means. After performing feature selection using Algorithm 1, we perform k-means on the reduced set of parameters to cluster pedestrians into groups with different motion patterns.

Figure 3.6 shows different clusters of pedestrians from the DUT data set obtained by performing k-means with forward selection and k-means with PCA, from left to right. We conduct these approaches separately on each data set.

3.6.2 Calibration

Genetic algorithms (**GA**) [73] are evolutionary algorithms, largely applied to tackle optimization problems such as calibration of model parameters [74, 75].

As stated earlier, we calibrate our model parameters using a GA. It begins with feeding a random initial set of chromosomes i.e., the set of parameters that need to be calibrated into the simulation model to acquire and compare outputs with real-world data to compute and assign a fitness score to the respective chromosome. Next, an offspring population is generated by performing the selection (of the fittest members), crossover, and mutation processes and fed into the model again unless a specific stopping criterion has reached.

We only consider the parameters in Table 3.2 for grouping pedestrians, and we calibrate these parameters as illustrated in Figure 3.5. Whereas, we calibrate the rest of the parameters of GSFM in beforehand, separately and in two steps: first, we calibrate the remaining SFM and safety parameters and then calibrate the game parameters. We conduct all these calibration steps using the above-described genetic algorithm. To be noted, during individual calibration of pedestrian, we simulate only the target pedestrian and update the states of surrounding agents as their real trajectories.

Selection of the fitness function and simulation output type depends on the types of parameters to calibrate. To calibrate the SFM and safety parameters, GSFM outputs the simulated positions of agent(s) ($\vec{P}_u^{sim}$) to compare with their real positions ($\vec{P}_u^{real}$) for calculating the fitness score of any respective chromosome. For the universal and group calibration, the fitness score is calculated by Eq.3.11 and the fitness function for the individual calibration is given in Eq.3.12.

$$f_{score} = \left(\sum_{e}^{E} \left(\sum_{u}^{U} \left(\sum_{t}^{T} \left| \vec{P}_u^{real}(t) - \vec{P}_u^{sim}(t) \right| \right) / T \right) \Big/ U \right) \Big/ E \tag{3.11}$$

$$f_{score} = \left(\sum_{t}^{T} \left| \vec{P}_u^{real}(t) - \vec{P}_u^{sim}(t) \right| \right) / T \tag{3.12}$$

$$f_{score} = \sum_{e}^{E} \left(\left(\sum_{u}^{U} \begin{cases} 1, & if\ A_u^{real} == A_u^{sim} \\ -1, & \text{otherwise} \end{cases} \right) \Big/ U \right) \Big/ E \tag{3.13}$$

Here, E, U, and T denote the number of scenarios, the number of agents, and the number of time steps, respectively. For Eq. 3.13, the simulated decisions (A_u^{sim}) are obtained by game playing and the real decisions (A_u^{real}) are manually extracted from the video data. To calibrate the game parameters, calculating the fitness score using Eq. 3.13 is preferable, as the game module is responsible for deciding on decisions/strategies for agents in any conflict situation, not their motion (see Section 3.5). We use Eq. 3.13 for calibrating the game parameters for the HBS data set but in case of the CITR and DUT data sets, Eq. 3.11 is used due to the difficulty on extracting the real decisions manually.

The values of the game parameters are given in Table 3.3. Table 3.4 shows the values of the SFM and safety-related parameters with their calibrated values, where, PP, PC, CP, and CC denote pedestrian-to-pedestrian, pedestrian-to-car, car-to-pedestrian, and car-to-car interactions, respectively.

After performing the clustering and calibration processes, we got several sets of parameters which results in different versions of our model. Specifically, GSFM-M1 which indicates the model with k-means and PCA, GSFM-M2 is the model that combines the forward selection method with k-means and calibrates all parameters given in Table 3.2

Table 3.3: List of game parameters with calibrated values

Symbol	**HBS Value**	**DUT Value**	**CITR Value**
G^{C}_{speed}	11	4	10.4
G^{P}_{speed}	1	0	1
$G^{competitor}_{speed}$	11	0	6.3
G_{noai}	3	0	0.3
$G_{stopped}$	2	0	1.1
G^{F}_{angle}	1	6.6	0.4
G^{min}_{dis}	7	5	6.1
G^{Ace}_{angle}	7	8	7
G^{Dec}_{angle}	5	8	5
G^{Dev}_{angle}	8	6	8

Table 3.4: The list of the SFM and safety parameters with their calibrated values. Here, G1, G2, G3 are the clustered groups.

Symbol	**Description**	**Unit**	**HBS**				**DUT**			**CITR**			
			G1	**G2**	**G3**	**U**	**G1**	**G2**	**U**	**G1**	**G2**	**G3**	**U**
$V^{o}_{ij}(PP)$	Interaction strength	m^2s^{-2}	0.1	0.1	1.9	0.1	0.01	0.1	0.1	0.2	0.1	0.4	0.1
$V^{o}_{ij}(PC)$	Interaction strength	m^2s^{-2}	15.1	17.3	11.9	11.7	1.6	3.4	4.5	0.2	2.6	0.07	1.5
$V^{o}_{ij}(CP)$	Interaction strength	m^2s^{-2}	—	—	—	—	0.76	1.7	2.27	—	—	—	—
σ (PP)	Repulsive interaction range	m	0.17	0.24	0.25	0.25	0.17	0.18	0.23	0.1	0.2	0.25	0.18
σ (PC)	Repulsive interaction range	m	0.1	0.7	0.2	0.91	0.11	0.14	0.27	1.5	0.39	1.1	0.69
λ	Anisotropic parameter	—	0.35	0.339	0.42	0.35	0.43	0.16	0.41	0.15	0.59	0.52	0.13
S_D	Scaling factor for deviate action	—	7.6	12	7.8	6	6	7	9.01	6.1	8.4	8.3	7.0
V_R	Range of view	m	18.4				10			12.3			
$D_{min}(PC)$	Critical spatial distance	m	7.8				8			7			
S_A	Scaling factor for accelerate action	—	6										
S_C	Scaling factor for conflict detection	—	9										
$D_{min}(CC)$	Critical spatial distance	m	8										
U^{o}_{iB}	Interaction strength for obstacle	m^2s^{-2}	10										
γ (obstacle)	Repulsive interaction range	m	0.2										

Table 3.5: Quantitative results i.e., **aADE**(m) / **aFDE**(m) / **SD**(ms^{-1}) / **CI** of the classical SFM and all versions of GSFM. Here, the **bold** number denotes the best score.

Model	**pedestrian**			**Vehicle**		
	HBS	**DUT**	**CITR**	**HBS**	**DUT**	**CITR**
GSFM-M1	**0.745**/**0.807**/0.338/0.0182	0.654/1.07/0.261/0.033	0.565/0.859/0.1742/0.0037	**1.26**/**3.33**/**1.083**	1.29/3.33/0.795	**2.41**/**5.17**/**1.153**
GSFM-M2	0.747/0.812/**0.333**/**0.0112**	**0.643**/**1.06**/0.263/0.036	**0.546**/**0.813**/0.1754/0.0037	1.33/3.46/1.107	**1.22**/**3.04**/**0.787**	2.46/5.19/1.166
GSFM-M3	0.766/0.854/0.338/0.0138	0.698/1.19/**0.260**/0.033	0.577/0.878/0.1742/**0.0035**	1.28/3.39/1.094	1.25/3.27/0.803	2.49/5.28/1.183
GSFM-U	0.754/0.829/0.335/0.0127	0.705/1.22/0.265/**0.030**	0.577/0.880/**0.1740**/0.0035	1.30/3.42/1.097	1.41/3.51/0.842	2.49/5.29/1.180
SFM	1.122/1.164/0.376/0.0305	1.499/2.26/0.263/0.036	1.185/1.791/0.2566/0.0123	—	—	—

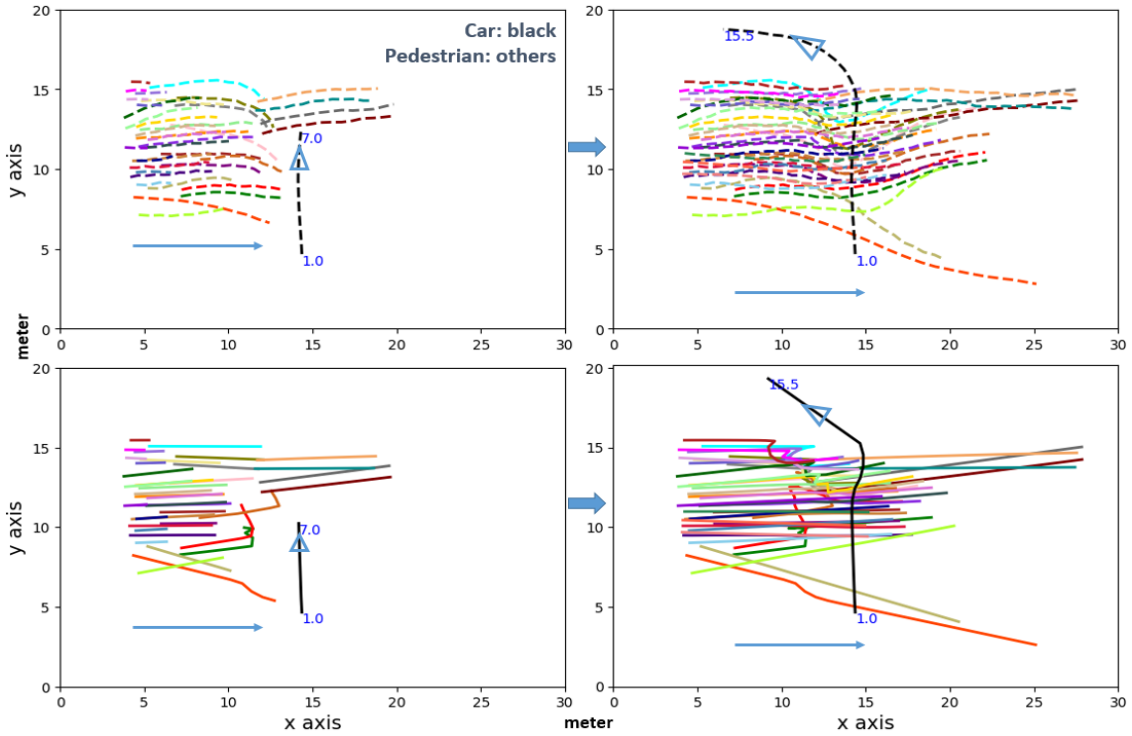

Figure 3.7: Crowd-to-car interaction from the DUT data set. The first row shows the real trajectories and the second row depicts the simulated trajectories, at two subsequent time steps. Car's trajectories are in black color.

during group calibration (**S8**), GSFM-M3 is the model with FS and k-means where only the selected parameters by FS are calibrated in group calibration (**S7**), and GSFM-U denotes the universal model, i.e., the model with one set of parameters. Due to space restrictions, Table 3.4 visualizes only the the values of parameters in GSFM-U (denoted as **U**) and GSFM-M2, for each data set. Here, G1, G2, G3 denote the clusters or groups.

3.7 Evaluation

As a quantitative evaluation, we compare all our models, namely GSFM-M1, GSFM-M2, GSFM-M3 and GSFM-U and the classical SFM proposed in [76]. We calibrate all parameters of the classical SFM for each data set using the GA in Section 3.6.2 and the fitness function in Eq. (3.11), for a fair comparison. The performances of these models are evaluated by the metrics given in Section 3.7.1 on the extracted interaction scenarios from the HBS, DUT and CITR data sets (summarized in Table 3.1). We select three example scenarios among all to evaluate the performance of our model qualitatively. We run all simulations on an Intel® Core™i5 processor with 16 GB RAM.

3.7.1 Evaluation Metrics

To evaluate the performance of the proposed models in terms of how realistic the resulting trajectories are, we consider two most commonly used metrics [77, 55], namely average displacement error (ADE) and final displacement error (FDE), together with two other metrics:

- **Adjusted Average Displacement Error (aADE)**: ADE computes the pairwise

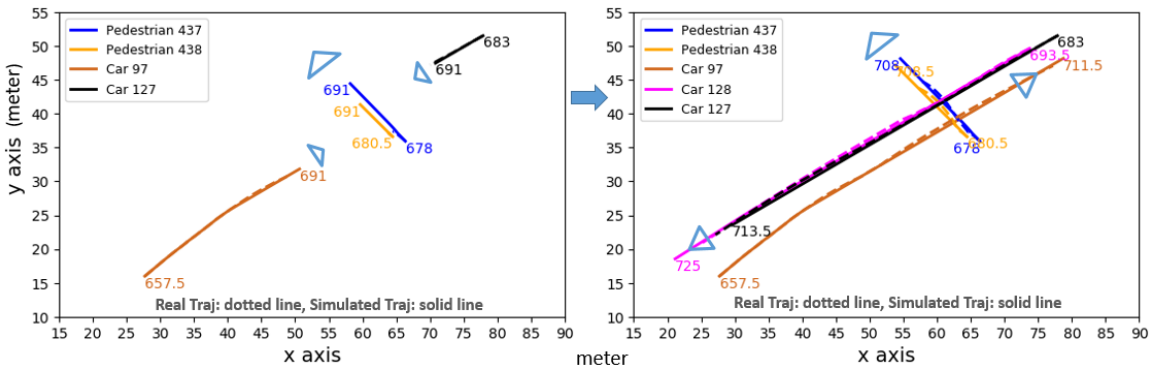

Figure 3.8: Pedestrians-to-cars crossing scenario from the HBS data set. The dotted lines represent the real trajectories and the solid lines are the simulated trajectories. Trajectories are visualized at two subsequent time steps.

mean square error (in meter m) between the simulated and real trajectories of each agent over all positions and averages the error over all agents. In our extracted scenarios, the trajectory length of agents k are different; thus, we choose an adjusted version of ADE to evaluate our models' performance more precisely: aADE $= \frac{k_0}{k}$ ADE, with k_0 as a predefined trajectory length (i.e., number of time steps), assuming that the error in trajectory modeling increases linearly.

- **Adjusted Final Displacement Error (aFDE)**: FDE calculates the average displacement error (in m) of the final point of all agents. We also adjust FDE like aADE.
- **Speed Deviation (SD):** the SD metric is for measuring the pairwise speed difference (in ms^{-1}) of simulated and real speed of each agent over all time steps and averaging these difference over all agents. SD is adjusted as aADE.
- **Collision Index (CI):** We choose the CI metric to penalize any collision of pedestrian(s) with the car(s). For each pedestrian i, CI $\in [0, 1]$ is described as the portion of the simulated trajectory of i that overlaps with any car's occupancy. CI $= 0$ means no collision. CI is averaged over all pedestrians and adjusted as other metrics.

3.7.2 Results

Table 6.2 visualizes the performances of the GSFM-M1, GSFM-M2, GSFM-M3, GSFM-U and the classical SFM models on the HBS, DUT and CITR data sets, evaluated using the above-described metrics. In column entries of Table 6.2, for pedestrians, we reported four scores that are aADE, aFDE, SD, and CI, respectively and for cars, three scores are shown as CI is only calculated from the perspective of pedestrians. The bold number indicates the best score. In all criterion, the GSFM-M1 and GSFM-M2 models perform similarly, and both these models outperform the universal model GSFM-U, but GSFM-M3 performs mostly similar to GSFM-U. All versions of GSFM model always perform better than the classical SFM. For all data sets, the average errors of our best-performed model in trajectory modeling, i.e. aADE and aFDE is range from 0.5 m to 1 m for pedestrian, which considers as a good result given the stochasticity in pedestrians behaviors and also similarities with the results presented in [78], a state-of-the-art trajectory prediction model of pedestrians that evaluated by pedestrian-only scenarios. However, the aADE/aFDE

scores of our model for vehicles is comparatively higher than pedestrians, i.e. bigger error, mainly for the CITR data set. One reason behind this is the significant difference in simulated and real speeds of vehicles. Thus, improving our vehicle motion modeling, e.g., by considering different motion patterns and speed profiles of vehicles, is part of our future work.

In all cases, the collision index CI is minimal, which indicates all models simulate collision-free trajectories for most of the time. Moreover, in terms of CI, our models perform much better than SFM for the CITR and HBS data sets, but due to higher pedestrian density in DUT, the performance of our models drop and become similar to SFM. For SFM, the entries for cars are empty because the classical SFM can only model pedestrian motions. Thus, in SFM, during the simulation of the extracted scenarios, the cars follow their real trajectories.

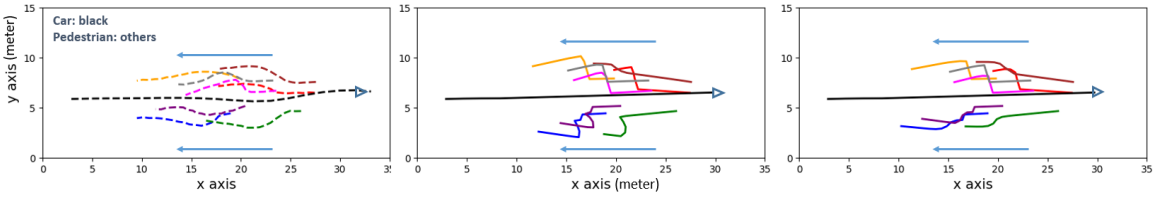

Figure 3.9: Pedestrians-to-car interaction from the CITR data set. The trajectories of road users: real, simulated in GSFM-U, and simulated in GSFM-M2 are visualized respectively, from left to right.

To show the differences in the DUT, HBS and CITR data sets and the capability of our model to address these differences, we choose one scenario from each data set and simulate each scenario in GSFM-M2. In all Figures 3.7, 3.8, and 3.9, the dotted lines indicate the real trajectory and the solid lines represent the simulated trajectories of road users. In Figure 3.7 and Figure 3.8, the real and simulated trajectories are visualized at two specific subsequent time steps. The black lines in Figure 3.7 and Figure 3.9 indicate the trajectories of car and the color-coded lines depict the trajectories of pedestrians.

Figure 3.7 visualizes a crowd-to-car interaction scenario from the DUT data set. Here, the first row shows the real trajectories of the involved road users, and the second row visualizes the simulated trajectories. Most of the DUT scenarios contain a large number of pedestrians, as shown in Figure 3.7.

Figure 3.8 depicts a complex pedestrians road crossing example with cars coming from two directions, extracted from the HBS data set. Both in simulation and reality, both cars stop to let the pedestrians cross first, which is a common phenomenon in HBS scenarios.

Figure 3.9 shows a pedestrians-to-car interaction scenario from CITR. As visualized in Figure 3.9, GSFM-U simulates all pedestrians in a similar style, while in GSFM-M2, pedestrians follow different motion patterns. Thus, the simulated trajectories of pedestrians in GSFM-M2 are more identical to their real trajectories than the trajectories generated by GSFM-U.

To sum up, in all example scenarios, our model realistically simulates complex interactions among pedestrians and car(s). Table 6.2 shows that our model performs satisfactorily for all data sets. Thus, our model was able to model scenarios from new data sets convincingly (i.e. CITR and DUT) with minimal effort compared to traditional approaches (i.e. starting modeling process from scratch for each new case), through the integration of new types of interactions into the model and largely automated calibration process. This evaluates the generalizability of our model. Plus, the results of our quantitative evaluation and

the visualization and discussion of the scenario in Figure 3.9 state that the performance of our model is improved due to heterogeneous motion patterns of pedestrians.

3.8 Conclusion

In this chapter, we proposed a procedure to formulate general motion models and applied this process to extend our Game-Theoretic Social Force Model (GSFM) towards a general model for generating realistic trajectories of pedestrians and cars in different shared spaces. Secondly, we applied and examined two clustering approaches namely, Principal Component Analysis (PCA) with the k-means algorithm and k-means with the forward selection method, to recognize and model different motion patterns of pedestrians.

We calibrated, validated, and evaluated our model using three shared space data sets, namely the HBS, DUT and CITR data sets. These data sets differ from one another in terms of spatial layout, types of interactions, traffic culture and density. In both quantitative and qualitative evaluation process, our model performed satisfactorily for each data set, which evinces that by following a systematic procedure with a well-defined calibration methodology, a shared-space model can adapt to a new environment and model a large variety of interactions. The results also indicate that the heterogeneity in pedestrians motion improves the performance of our model.

Our future research will focus on improving the motion model for vehicles, adding new modalities (e.g., cyclists) into our model, calibrating the model parameters for a wider range of interactions (e.g., vehicle-to-vehicle complex interaction), recognizing different motion patterns of other user types such as vehicles, and calibrating and evaluating our model using more open-source data sets of shared spaces. Most significantly, we shall study large scenarios with a larger number of participants to investigate the scalability of different interaction types and also our simulation model.

Chapter 4

Social Force Model with Model Predictive Control for Longitudinal Vehicle Control in Pedestrian-Dense Scenarios

This chapter is derived from the published work in [79].

4.1 Background

Pedestrian safety has always been the main concern of the traffic system. In U.S., from year 2007 to 2016, the percentage of pedestrian fatalities in total fatalities has increased from 11% to 16%[80]. In the statistics of 2016, 72% of pedestrian fatalities does not happen at intersection, which means these fatalities happen at places where there is no traffic signal controlling the priorities of the traffic participants. Therefore, it is important to study such unsignalized scenarios, especially for autonomous vehicles that have the ability to automatically adjust their speed hence avoid errors caused by human drivers. In this study, a specific unsignalized scenario is considered, in which the autonomous vehicle needs to drive through a crowd of pedestrians. Pedestrian crowd with high density (e.g. larger than 10 pedestrians) is the main focus.

To cope with this scenario, this study proposed a model-based predictive control strategy that incorporates a pedestrian motion prediction model to achieve the longitudinal speed regulation. Model predictive control (MPC) [81] has been used for vehicle longitudinal speed regulation for a long time. Most studies about MPC longitudinal speed regulation focuses on problems such as keeping a desired distance to the front vehicle, e.g., adaptive cruise control [82], or dealing with big obstacles that appear in the front, e.g., yielding to a cut-in vehicle [13]. This study entered into a new area of using MPC for vehicle-pedestrian interaction scenario, which has not been properly addressed yet.

Since pedestrian motion is affected by various factors, it is necessary to find an appropriate model that can effectively describe the vehicle-pedestrian interaction. Most existing studies explored only the interaction between the vehicle and a couple of pedestrians, in which theories such as gap acceptance and time to collision have been used to determine the pedestrian's intention [8]. In [83], a social force model [15] appended with the vehicle influence was proposed to predict the pedestrian's motion for the vehicle speed regulation.

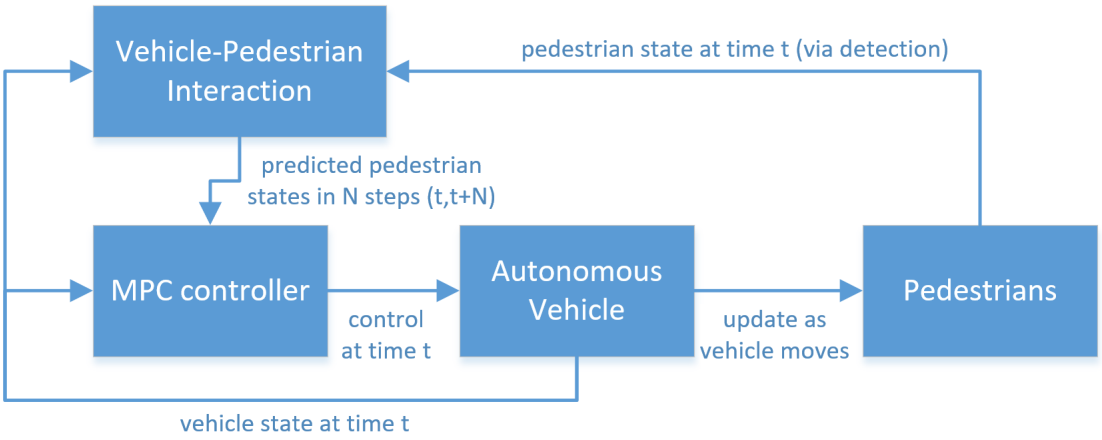

Figure 4.1: The structure of the proposed MPC-VCI longitudinal speed regulation strategy

However, this model doesn't consider the interaction with high-density pedestrians and the proposed vehicle speed regulation method has no predictive ability. In our previous work [17], a social force based vehicle-crowd interaction (VCI) model was proposed to predict the motion of pedestrian crowd of any density, in which each individual pedestrian motion is subject to surrounding pedestrians and incoming vehicles. This study combined our previous work with the long-established MPC by designing the customized state constraints and the cost function to address the longitudinal speed regulation problem. The proposed method can keep the vehicle from a safe distance to the closest pedestrian in front, and in the meantime, try to maintain its desired speed as much as possible.

The flowchart of the combined MPC-VCI method is illustrated in figure 4.1. Vehicle-pedestrian interaction is evaluated at each time step k. The future N-step pedestrian motion is then predicted and provided with MPC. MPC utilizes the predicted motion to formulate a standard quadratic programming (QP) problem. By solving this QP problem, the optimized control is obtained and consequently applied to the vehicle.

The rest of the chapter is organized as follows. Section 2 outlines the vehicle-crowd interaction model with emphasis of the slightly modified vehicle influence. Section 3 details the MPC based longitudinal speed regulation policy, which includes vehicle dynamics, MPC synthesis, QP generation, and complete algorithm. Section 4 presents the evaluation procedure, followed by simulation results in section 5. In the end, section 6 concludes the chapter.

4.2 Pedestrian Motion Prediction

4.2.1 Vehicle-Crowd Interaction Model

A social force based vehicle-crowd interaction (VCI) model [17] is used for the pedestrian motion prediction under the vehicle influence. In this model, each pedestrian motion $x_i \in \mathbb{R}^2$ is governed by 2D planar point-mass Newtonian dynamics subject to a total force $F_i \in \mathbb{R}^2$ consisting of several sub-forces:

$$\frac{d^2x_i}{dt^2} = \frac{v_i}{dt} = a_i = \frac{F_i}{m}, \tag{4.1}$$

where

$$F_i = \sum_{j \in \mathbb{Q}(i)} (f_r^{ij} + f_c^{ij} + f_n^{ij}) + f_v^i + \beta_i(f_v^i) \cdot f_d^i.$$

$j \in \mathbb{Q}(i)$ denotes the index of nearby pedestrians around pedestrian i. $f_r^{ij}, f_c^{ij}, f_n^{ij}$ are social forces, which denote the repulsive (attractive) force, the collision force, and the navigational force from pedestrian j to pedestrian i, respectively. f_v^i is the vehicle influence on pedestrian i, which is also a repulsive force with a specific direction. f_d^i is the destination force that drives the pedestrian to the temporary destination. $\beta_i(f_v^i) \in [0, 1]$ is a scalar that adjusts the magnitude of f_d^i. Details of modeling the above sub-forces can be found in [17].

4.2.2 Vehicle Influence

Based on the original vehicle influence proposed in [17], some parameters of the model was slightly modified to be more suitable for this specific interaction problem. When the vehicle longitudinal speed is less than $0.2m/s$, the vehicle is simply regarded as a rectangular static obstacle. Roughly speaking, the vehicle influence can be viewed as a potential field subject to the change of the vehicle speed. Figure 4.2 shows the magnitude and the direction of f_v^i in the surrounding area of the vehicle with different vehicle longitudinal speed.

4.2.3 Motion Prediction

To predict pedestrian motion, it's assumed that all pedestrian state at time $t = t'$ can be correctly obtained, which means the sensing capability of the vehicle is perfect. In the prediction process, it is also assumed that the vehicle moves at a constant longitudinal speed the same as the current speed. Pedestrian motion at $t > t'$ is then calculated by iteratively applying the pedestrian dynamics in equation (4.1).

4.3 Longitudinal Speed Regulation

4.3.1 Vehicle Dynamics

This study only considers longitudinal speed regulation, so a planner vehicle model with only longitudinal dynamics [13] is sufficient for this study:

$$M\ddot{s}(t) + \alpha\dot{s}(t) = F_t(t) - F_b(t) \tag{4.2}$$

where s is vehicle longitudinal position, M is the vehicle mass, α is a linearized friction coefficient, and F_t, F_b are traction force and brake force of the vehicle, respectively.

Let $x = [x_1, x_2]^{\mathrm{T}} = [s, \dot{s}]^{\mathrm{T}} \in \mathbb{R}^2$ be a state vector of the vehicle position and speed. Equation (4.2) can be written as matrix form. Furthermore, with discretization time Δt, the discretized vehicle dynamics can be obtained:

$$x(k+1) = Ax(k) + Bu(k) \tag{4.3}$$

where

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 - \frac{\alpha \Delta t}{M} \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{\Delta t}{M} \end{bmatrix}, u(k) = F_t(k) - F_b(k).$$

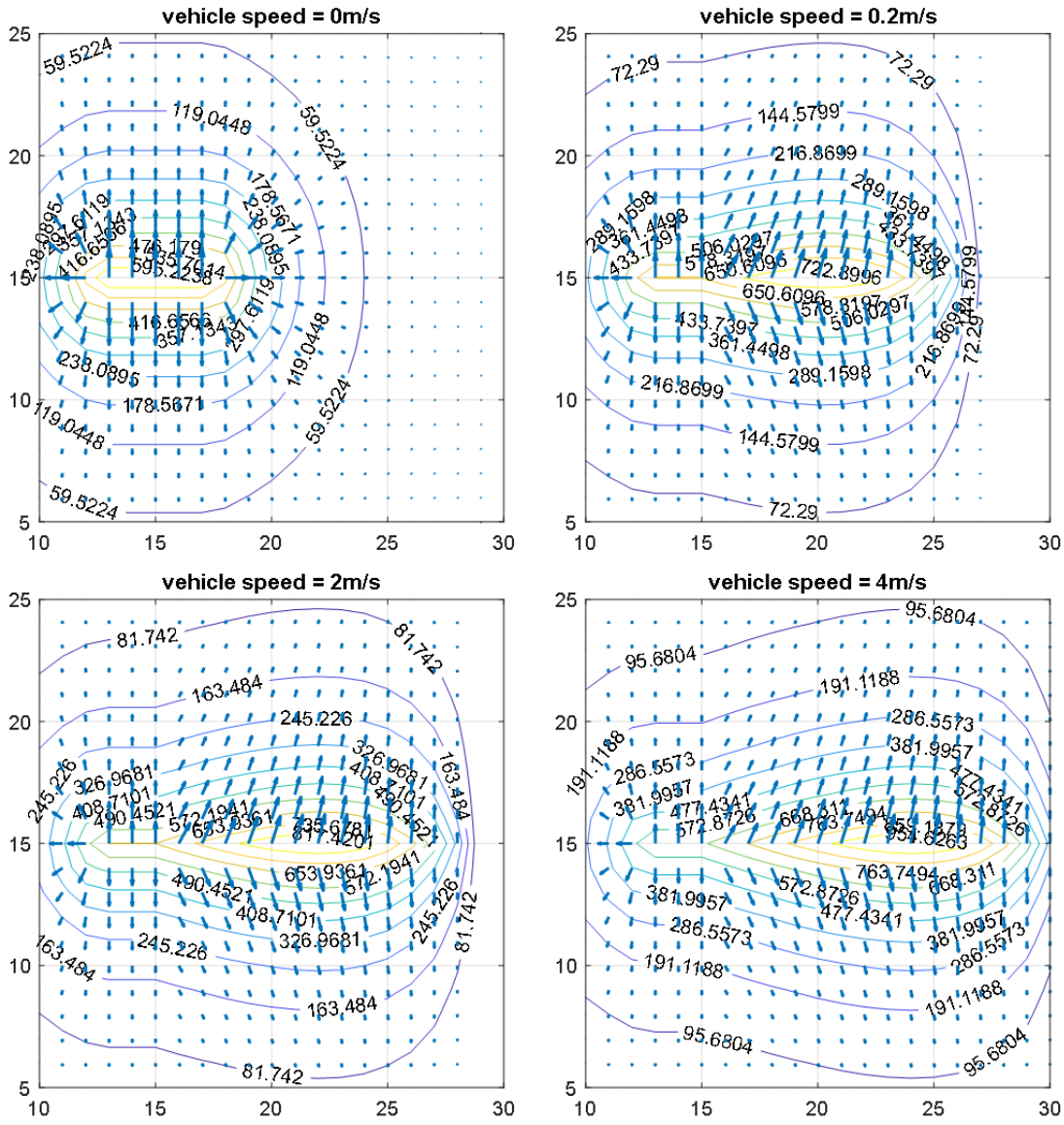

Figure 4.2: The contour plot of vehicle influence at different longitudinal speed. The vehicle is located at $(15, 15)^{\mathrm{T}}$, facing positive x-axis. The vehicle length is 5, and vehicle width is 2. Blue arrows indicate the direction and the magnitude (arrow length) of vehicle influence force on a pedestrian located at the arrow position. As the longitudinal speed increases, the influence area expands.

4.3.2 Model Predictive Controller (MPC) Synthesis

At time step k, with the vehicle dynamics (4.3) and current vehicle state $x(k)$, future vehicle state $x(k+n)$ can be obtained by iteratively applying the vehicle dynamics:

$$x(k+n|k) = A^n x(k) + A^{n-1}Bu(k|k) + A^{n-2}Bu(k+1|k) + \\ \cdots + ABu(k+n-2|k) + Bu(k+n-1|k). \quad (4.4)$$

For simplicity, $x(k+n)$ will be used instead of $x(k+n|k)$ for the rest of the paper.

Now, consider a MPC with N-step prediction horizon. The vehicle state from steps $k+1$ to $k+N$ can be combined and represented as following equation:

$$X = S_x x_k + S_u U \quad (4.5)$$

where

$$X = \begin{bmatrix} x(k+1) \\ x(k+2) \\ \vdots \\ x(k+N) \end{bmatrix} \in \mathbb{R}^{2N}, S_x = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \in \mathbb{R}^{2N\times 2},$$

$$S_u = \begin{bmatrix} B & 0 & \dots & 0 \\ AB & B & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B & \dots & AB & B \end{bmatrix} \in \mathbb{R}^{2N\times N},$$

$$U = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N-1) \end{bmatrix} \in \mathbb{R}^N, x_k = x(k) \in \mathbb{R}^2.$$

Due to the physical limitation of the vehicle, there are constraints on both the control action and the control action rate. Hence, $\forall i = k+1, \dots, k+N$, we have

$$|u(i)| \leq u_{max} \quad (4.6)$$

$$|\Delta u(i)| \leq \Delta u_{max}. \quad (4.7)$$

A speed constraint is also considered:

$$v_{min} \leq x_2(i) \leq v_{max}, \forall i = k+1, \dots, k+N. \quad (4.8)$$

To avoid collision between the vehicle and pedestrians, a safe distance d_{safe} to the closest pedestrian in front of the vehicle must be maintained all the time. Since the previous defined pedestrian model can predict the pedestrian motion under vehicle influence in future N steps, the predicted pedestrian information can be used by MPC to maintain the safe distance in N steps.

This pedestrian safety is formulated as a hard constraint on the vehicle state:

$$x_1(i) - x_p(i) \geq d_{safe}, \forall i = k+1, \dots, k+N \quad (4.9)$$

where $x_p(i)$ is position of the closest pedestrian in front of the vehicle in the axis of vehicle's longitudinal position.

4.3.3 Cost Function Design and Quadratic Programming (QP) Problem Formulation

The ultimate goal of the MPC is to find a control sequence $U = [u(k), u(k+1), \ldots, u(k+N-1)]^{\mathrm{T}}$ at every $x(k)$ such that the vehicle obeys both the state constraints and the pedestrian safety requirement, while in the meantime, tries to keep the desired speed v_d as much as possible. Therefore, the cost function is designed as how close the vehicle will keep its desired speed:

$$J(k) = (A_r X - V_r)^{\mathrm{T}} Q (A_r X - V_r) \tag{4.10}$$

where Q is the quadratic cost, $V_r = [v_r, v_r, \ldots, v_r]^{\mathrm{T}} \in \mathbb{R}^N$ represents the reference speed in N steps, and

$$A_r = \begin{bmatrix} 0 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 0 & 1 \end{bmatrix} \in \mathbb{R}^{N \times 2N}$$

which extracts the velocity states from X.

Substituting equation (4.5), we can rewrite the cost function as:

$$J(k) = U^{\mathrm{T}} H U + 2FU + Y \tag{4.11}$$

where

$$\begin{aligned} H &= S_u^{\mathrm{T}} A_r^{\mathrm{T}} Q A_r S_u \\ F &= (A_r S_x x_k - V_r)^{\mathrm{T}} Q A_r S_u \\ Y &= (A_r S_x x_k - V_r)^{\mathrm{T}} Q (A_r S_x x_k - V_r) = const. \end{aligned}$$

Similarly, by substituting equation (4.5) and rearrange the constraint equations, the state constraints can be rewritten as:

$$A_u U \geq -U_{max} \tag{4.12}$$
$$-M_u U \geq -\Delta U_{max} - u_0 \tag{4.13}$$
$$M_u U \geq -\Delta U_{max} + u_0 \tag{4.14}$$
$$-M_v S_u U \geq -V_{max} + M_v S_x x_k \tag{4.15}$$
$$M_v S_u U \geq V_{max} - M_v S_x x_k \tag{4.16}$$
$$-M_x S_u U \geq D_{safe} - X_p + M_x S_x x_k \tag{4.17}$$

where

$$\begin{aligned} U_{max} &= [u_{max}, u_{max}, \ldots, u_{max}]^{\mathrm{T}} \in \mathbb{R}^{2N}, \\ \Delta U_{max} &= [\Delta u_{max}, \Delta u_{max}, \ldots, \Delta u_{max}]^{\mathrm{T}} \in \mathbb{R}^{N}, \\ u_0 &= [u(k-1), 0, \ldots, 0]^{\mathrm{T}} \in \mathbb{R}^{N}, \\ V_{max} &= [v_{max}, v_{max}, \ldots, v_{max}]^{\mathrm{T}} \in \mathbb{R}^{N}, \\ V_{min} &= [v_{min}, v_{min}, \ldots, v_{min}]^{\mathrm{T}} \in \mathbb{R}^{N}, \\ D_{safe} &= [d_{safe}, d_{safe}, \ldots, d_{safe}]^{\mathrm{T}} \in \mathbb{R}^{N}, \\ X_p &= [x_p(k+1), x_p(k+2), \ldots, x_p(k+N-1)]^{\mathrm{T}} \in \mathbb{R}^{N}, \end{aligned}$$

$$A_u = \begin{bmatrix} -1 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ 0 & -1 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 \\ 0 & 0 & \dots & 1 \end{bmatrix} \in \mathbb{R}^{2N \times N},$$

$$M_u = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ -1 & 1 & \dots & 0 & 0 \\ 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -1 & 1 \\ 0 & 0 & \dots & 0 & -1 \end{bmatrix} \in \mathbb{R}^{N \times N},$$

$$M_v = A_r, M_x = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix} \in \mathbb{R}^{N \times 2N}.$$

Finally, the optimal control sequence U^* can be obtained by solving the following standard QP problem:

$$U^* = \arg\min_U (U^{\mathrm{T}} H U + 2FU + Y) \tag{4.18}$$

subject to equations from (4.12) to (4.17).

This QP problem can be easily solved by standard QP toolbox. In this study, the 'mpcqpsolver' in Matlab is used to solve this QP problem. Once U^* is obtained, The first action of U^* is applied for the current step.

4.3.4 MPC Feasibility and Supplementary Proportional-integral-derivative (PID) Control

Due to the highly uncertainty of the pedestrian-dense traffic scenario, solving the above QP problem for MPC might be infeasible. In this study, a classical PID approach [84] is proposed to supplement the MPC. At any time step, when the MPC cannot find a feasible solution, the controller switches to PID approach. To maintain the safe distance d_{safe}, a reference longitudinal speed $v_r^{PID}(k)$ for PID is determined based on the current distance to the closest pedestrian in front $x_p(k) - x_1(k)$, as shown in figure 4.3. The discrete-time PID control action is obtained as follows:

$$u(k) = -[u_p(k) + u_i(k) + u_d(k)] \tag{4.19}$$

where

$$\begin{aligned} u_p(k) &= K_p e(k), \\ u_i(k) &= K_i e(k)\Delta t + u_i(k-1), \\ u_d(k) &= K_d[e(k) - e(k-1)]/\Delta t, \\ e(k) &= x_2(k) - v_r^{PID}(k), \end{aligned}$$

and K_p, K_i, K_d are PID parameters.

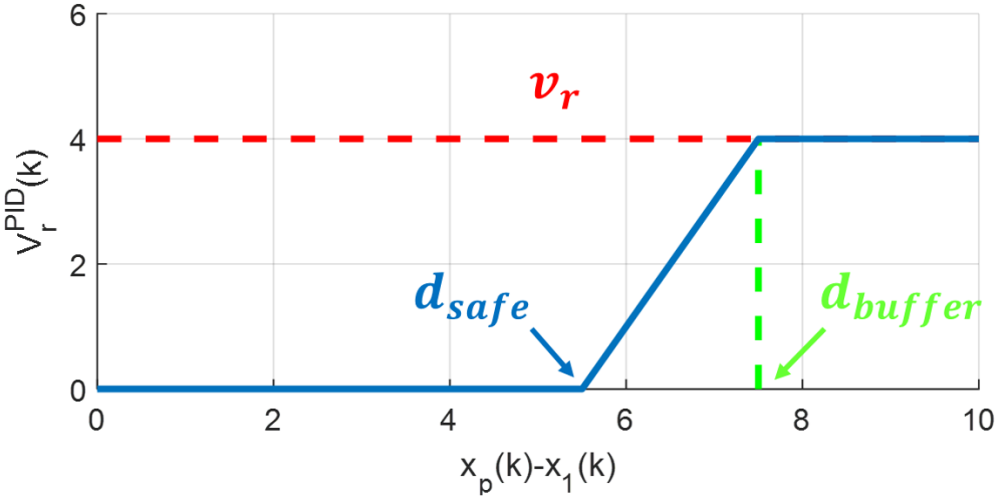

Figure 4.3: The longitudinal reference speed for the PID controller. A buffer distance is designed to gradually change the reference speed.

Table 4.1: Parameters for the longitudinal speed regulation

Parameter	Tuned Value	Unit
M	1000	kg
α	100	$N(m/s)^{-1}$
Δt	0.05	s
u_{max}	8000	N
v_r	4	m/s
N	15	
Δu_{max}	1000	N
K_p	300	
d_{safe}	8	m
v_{max}	20	m/s
K_i	10	
d_{buffer}	10	m
v_{min}	0	m/s
K_d	100	
Q	I_N	

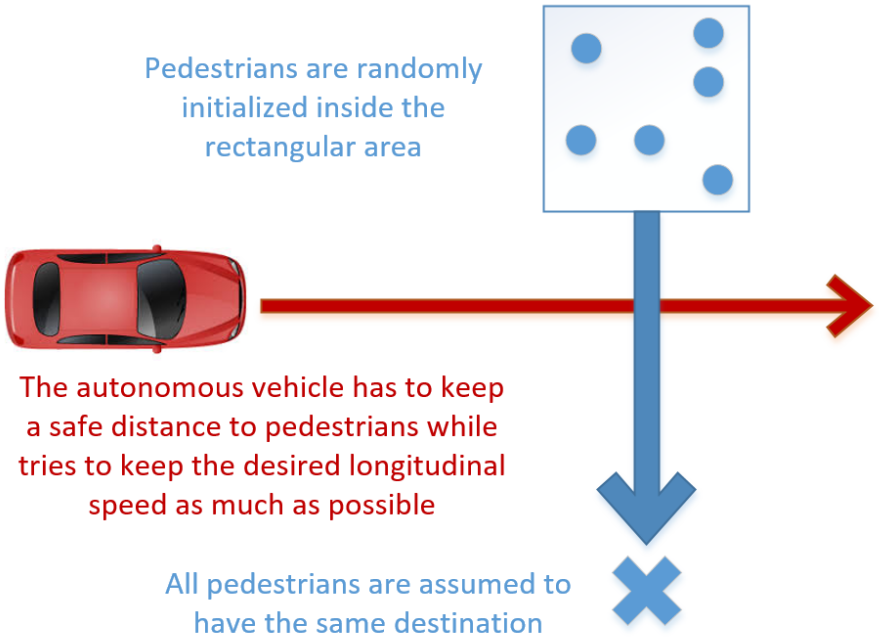

Figure 4.4: The scenario to be evaluated in the simulation. An autonomous vehicle interacts with a crowd of crossing pedestrians. There is no road layout, so that the vehicle and pedestrians have the same priority. The objective of the autonomous vehicle is to keep a safe distance to the closest pedestrian in front while tries to keep the desired longitudinal speed as much as possible. It is assumed that the vehicle can only move longitudinally, so this study doesn't consider steering action.

4.3.5 Overall Algorithm

Table 4.1 shows all the parameters for the vehicle dynamics, the MPC, and the PID, which are manually tuned in the simulation. The overall algorithm to regulate the longitudinal speed of the autonomous vehicle is summarized in Algorithm 2.

Algorithm 2 MPC+PID longitudinal speed regulation

control action $u(k)$ initialization

for each time step k **do** obtain $x(k)$ and all pedestrian states predict pedestrian motion and obtain X_p solve for $U^* = \arg\min_U (U^{\mathrm{T}} H U + 2FU + Y)$ MPC is feasible apply $u(k) = U^*(1)$ apply $u(k) = f_{PID}(x(k), x_p(k))$

4.4 Evaluation

A classical pedestrian crossing scenario was designed to evaluate the proposed MPC, as illustrated in figure 4.4. The actual (not predicted) pedestrian motion is also generated by aforementioned VCI model [17]. The simulation was repeatedly conducted for 2000 times. For each simulation, pedestrians were randomly initialized inside a rectangular area, so that situations of different pedestrian patterns can be covered.

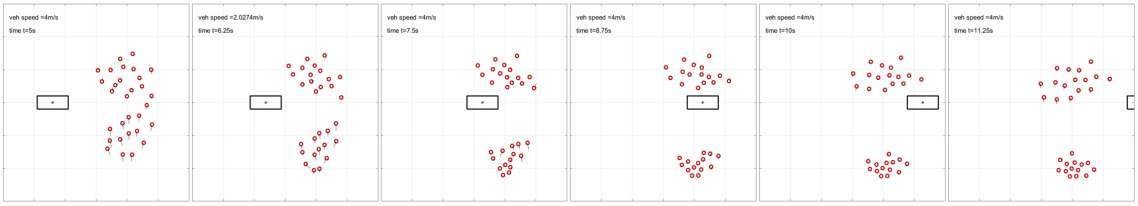

Figure 4.5: Screen-shots of an example of simulation at $t = 5, 6.25, 7.5, 8.75, 10, 11.25$ (s), respectively. Red circles indicate crossing pedestrians. Black rectangle indicates the autonomous vehicle using MPC approach.

The major evaluation criteria is the time spent for the autonomous vehicle to complete the vehicle-pedestrian interaction. The same number of simulations with pure PID approach on the vehicle was also conducted for the comparison purpose, because PID is regarded as the most efficient traditional approach for the longitudinal speed regulation.

Either using MPC or PID, different simulation might generate different interaction results, in which the vehicle might stop and wait for the pedestrian crossing, or directly drive through the pedestrian crowd without stopping and waiting. The reason is that when pedestrians interact with the autonomous vehicle, different pedestrian positions at any time $t = t'$ result in different vehicle speed regulation, which further increases the uncertainty of pedestrian motion at time $t > t'$. Therefore, the simulation was evaluated based on 3 different situations:

- General Situation: consider the entire simulation results.
- Stop-and-Wait Situation: consider situations when both MPC and PID approaches stop and wait for pedestrian crossing.
- Non-stop Situation: consider situations when both MPC and PID approaches do not stop and wait.

4.5 Result

4.5.1 Comparison Between MPC and PID

To visually illustrate the simulation result, figure 4.5 shows the screen-shots of one simulation example. The corresponding video is available online.[1] In this example, the autonomous vehicle slightly adjusted its longitudinal speed and successfully completed the vehicle-pedestrian interaction.

Figure 4.6 shows the change of the vehicle state and controller state. In this particular example, MPC approach generates smoother longitudinal speed than PID approach.

The 2000 simulation results in scenarios of a number of 30 pedestrians were used for further analysis, which is divided into following 3 situations:

- Difference of total time spent to complete the interaction in General Situation.
- Difference of longest time spent to wait for pedestrian crossing in Stop-and-Wait Situation.
- Difference of total time spent to complete the interaction in Non-stop Situation.

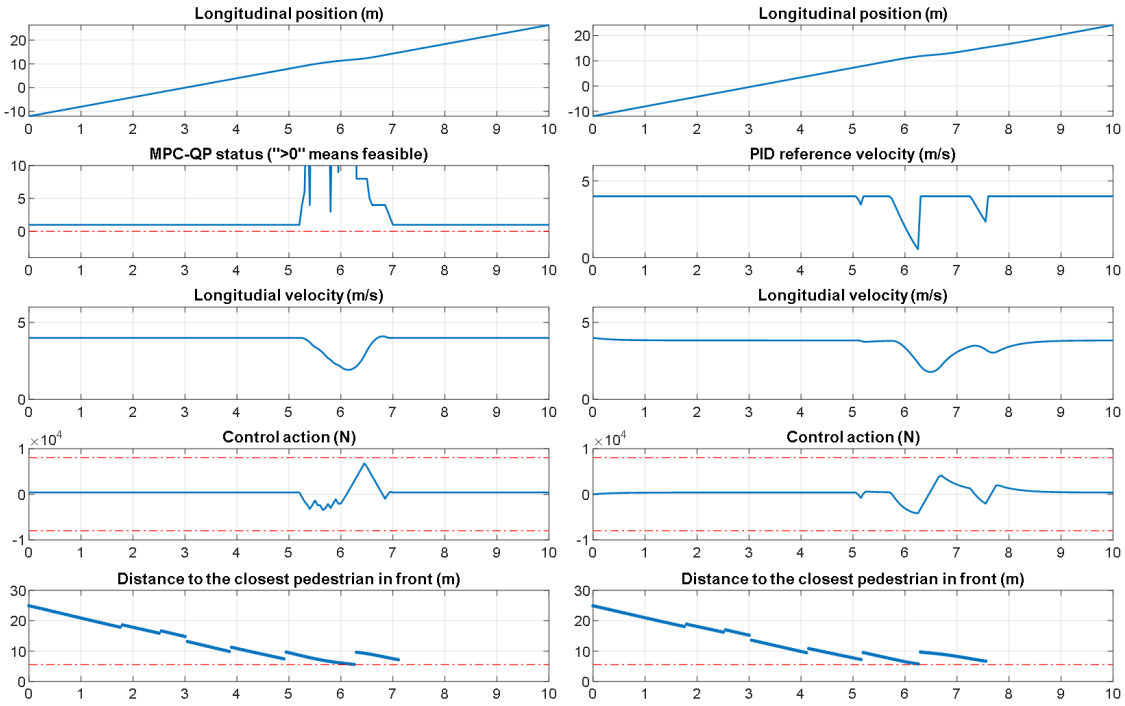

Figure 4.6: An example of performance comparison between MPC approach (left column) and PID approach (right column). In general, since MPC approach can predict the future trajectories of pedestrians, it generates smoother longitudinal velocity than PID approach. Note that half of the vehicle length (2.5m) is subtracted in the distance to the closest pedestrian in front.

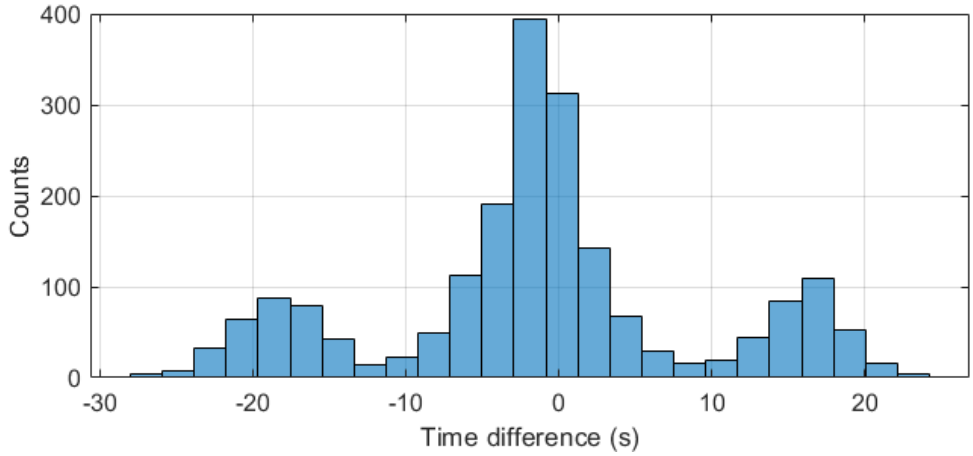

Figure 4.7: The difference of total time spent to complete the interaction between MPC approach and PID approach in all situations. The histogram is almost symmetric with a slight shift to the left (approximately 1s), which indicates the total time spent in MPC approach is generally shorter than PID approach.

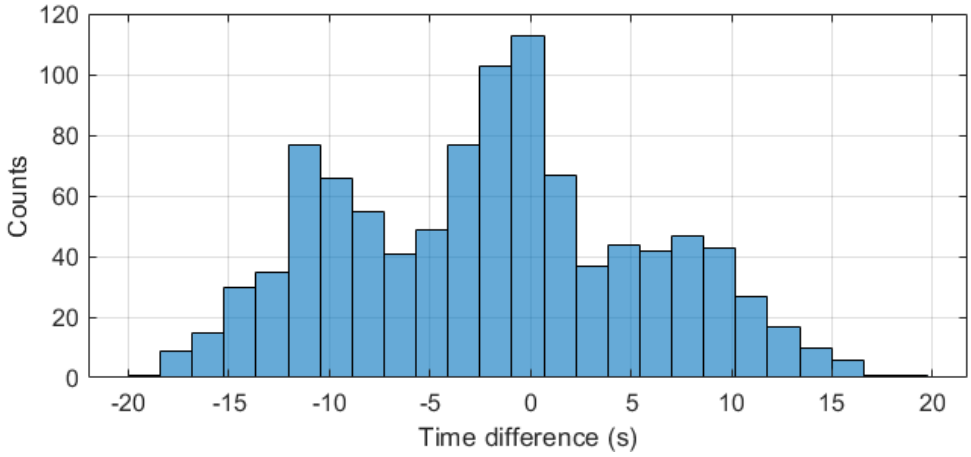

Figure 4.8: The difference of the longest time spent to wait for pedestrian crossing between MPC approach and PID approach in situations where both approaches stop and wait for pedestrian crossing. In the histogram, the slight shift to the left indicates the longest waiting time in MPC approach is generally shorter than PID approach.

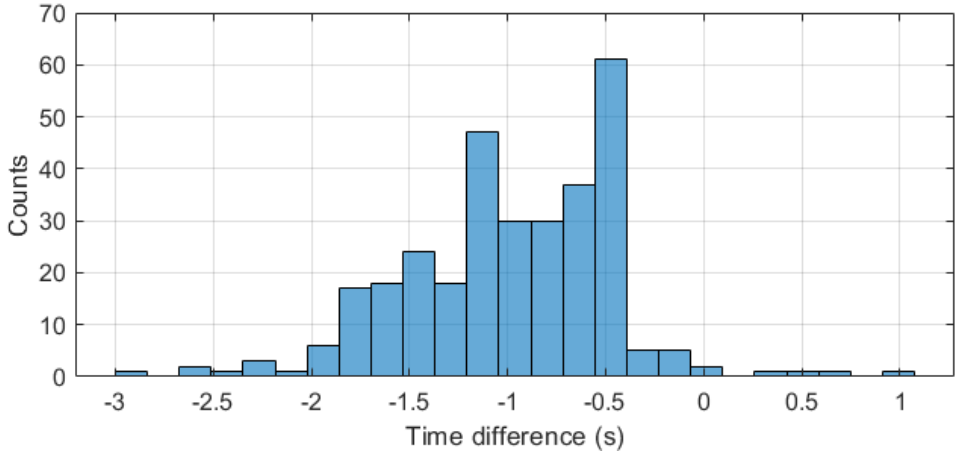

Figure 4.9: The difference of total time spent to complete the interaction between MPC approach and PID approach in situations where both approaches do not stop and wait. The histogram strongly indicates that the total time spent in MPC approach is shorter than PID approach.

Figures 4.7, 4.8, and 4.9 show the histograms of the time difference for the above 3 situations. In general, the MPC approach is better than PID approach. Detail description can be found in the figure captions.

4.5.2 Different Pedestrian Density

Simulations of different pedestrian density were also conducted. The numerical results are shown in table 4.2.[2] Generally speaking, the MPC approach is better than PID approach in terms of the time to complete the interaction, although the performance degrades as the pedestrian density decreases.

There is a steady-state error of $\approx 0.16m/s$ at the desired speed $v_r = 4m/s$ for the PID approach. The maximum delay caused by this steady-state error to complete the interaction is $\approx 0.4s$, which is calculated by assuming $v_r^{PID} = v_r$ all the time. Therefore, if the maximum delay of PID is considered, MPC approach is still better than PID approach in pedestrian-dense scenario (30 pedestrians in the simulation). However, in less-dense scenarios (20 or 10 pedestrians), it is hard to conclude that MPC approach is better than PID approach, although the simulation result still shows negative time difference.

4.6 Conclusion

This chapter investigated the possibility of applying model predictive control (MPC) supplemented with social force based vehicle-crowd interaction (VCI) model to regulate the longitudinal speed of the autonomous vehicle that faces a crowd of crossing pedestrians. The MPC problem was formulated based on state constraints and a safe distance to achieve collision avoidance and maximally maintaining desired speed. The formulation was successfully converted into a standard quadratic programming (QP) problem, which can be easily solved by standard QP toolbox. Preliminary results demonstrated the merits of the proposed MPC approach by comparing it with classical pure PID approach.

Future work is required to solve the following issues:

- In the pedestrian motion prediction process, this constant vehicle speed assumption can be improved by incorporating the VCI model into the MPC synthesis. Because of the non-linearity of VCI model, this incorporation requires modifying the VCI model so that the MPC can be properly synthesized and successfully solved.

[1] `https://youtu.be/JlR3aZ1saDU`

[2] N.A. in column 3 row 4: The number of instances in this situation is very small, hence the result is not provided here.

Table 4.2: Average Time Difference (in seconds) Between MPC and PID Approaches with Different Pedestrian Density in Different Situations (A. General: time spent to complete the interaction in all situations; B. Stop-and-Wait: longest waiting time when both approaches stop and wait for pedestrian crossing; C. Non-stop: time spent to complete the interaction when both approaches do not stop and wait)

# of Ped.	General	Stop-and-Wait	Non-stop
30	-1.2665	-1.9457	-0.9843
20	-0.5243	-1.8338	-0.7630
10	-0.4153	N.A.	-0.5394

- The performance of the PID approach can be improved by systematically tuning the PID parameters. Specifically, the steady-state error should be minimized or eliminated, and other effects such as rise time, overshot, settling time, and stability should also be carefully treated.
- In addition to hard constraints on the control action, a quadratic term of control effort could also be included in the MPC cost function, so that the overall MPC performance can be improved by taking the control action in consideration.

Chapter 5

Predicting Pedestrian Crossing Intention

This chapter is derived from the published work in [85].

5.1 Background

Autonomous driving technology has made significant progress in the past few years. However, to develop vehicle intelligence that is comparable to human drivers, understanding and predicting the behaviors of traffic agents is indispensable. This chapter aims to develop behavior understanding algorithms for vulnerable road users. Specifically, a vision-based pedestrian crossing intention prediction algorithm is proposed.

Behavior understanding plays an crucial role in autonomous driving system. It establishes trust between people and autonomous driving vehicles. By explicitly showing passengers how the system make its decisions, people will be more willing to accept this technology.

In level 4 autonomous driving, pedestrian crossing behavior is one of the most important behaviors that needs to be studied urgently. In urban scenarios, vehicles frequently interact with crossing pedestrians. If the autonomous system failed to handle vehicle-pedestrian interaction, casualties will most likely occur. With accurate intention prediction, the decision-making and planning modules in autonomous driving systems can access additional meaningful information, hence generating more safe and efficient maneuvers.

Nowadays, visual sensors such as front-facing cameras are becoming the standard configuration of autonomous driving systems. In the tasks of object detection and tracking, both the software and hardware of vision components are mature and ready for mass production. This provides a perfect platform on which vision-based behavior prediction algorithms can be deployed. Researchers and engineers in the prediction field can just focus on algorithm design. When the algorithm is ready, deployment becomes relatively trivial. This makes the proposed algorithm, vision-based pedestrian intention prediction promising. As long as the prediction algorithm is appropriately tested and verified, mass deployment becomes straightforward.

Vision-based pedestrian crossing intention prediction has been explored for several years. Early works [86] usually utilized a single frame as input to a convolutional neural network (CNN) based prediction system. This approach ignores the temporal aspect of image frames, which play a critical role in the intention prediction task. Later on, with the

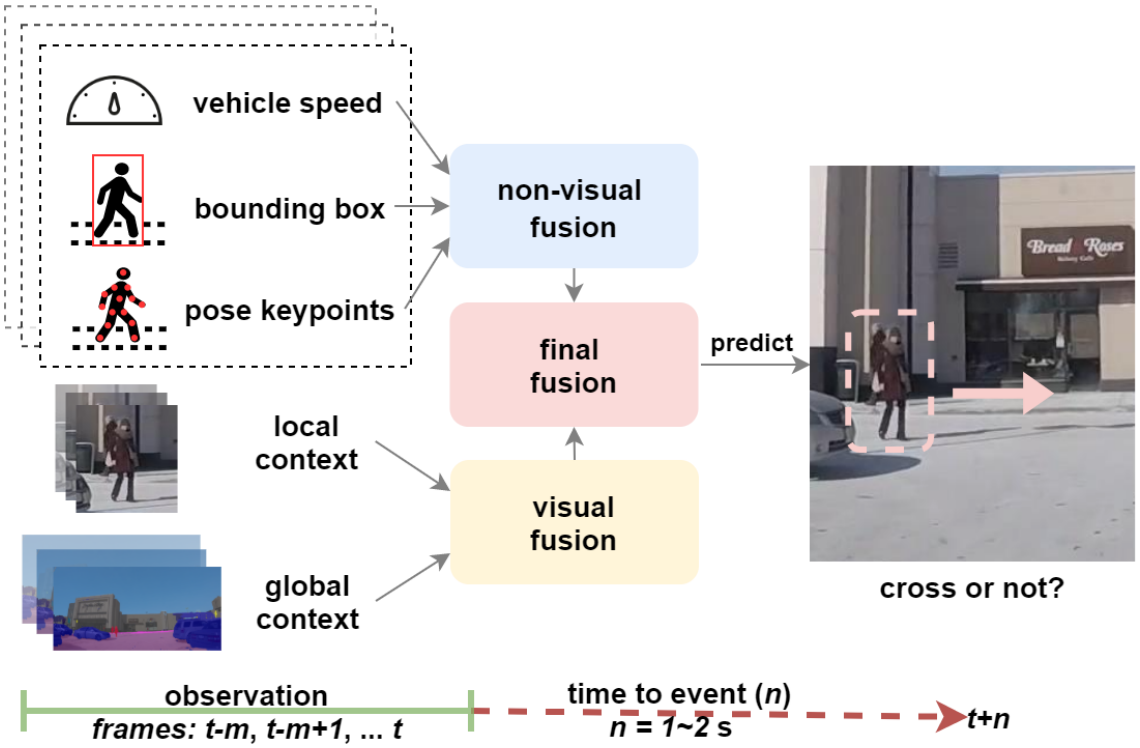

Figure 5.1: Predicting pedestrian crossing intention is a multi-modal spatio-temporal problem. Our method fuses inherently different spatio-temporal phenomena with CNN-based visual encoders, RNN stacks, and attention mechanisms to achieve state-of-the-art performance.

maturity of recurrent neural networks (RNNs), pedestrian crossing intention was predicted by considering both the spatial and temporal information [87, 88, 89]. This led to different ways of fusing different features, e.g., the detected pedestrian bounding boxes, poses, appearance, and even the ego-vehicle information [90, 91, 92, 93, 94]. The most recent benchmark of pedestrian intention prediction was released by [95], in which the PCPA model achieved the state-of-the-art in the most popular dataset JAAD [86]. However, PCPA lacks the consideration of global contexts such as road and other road users. We believe they are nonnegligible in pedestrian crossing intention prediction. Furthermore, the existing fusion strategies may not be optimal.

In this work, we focus on improving the performance of vision-based prediction of pedestrian crossing intention, i.e., whether a pedestrian detected by a front-facing camera will cross the road or not in a short time horizon (1-2s). Our work leverages the power of deep neural networks and fuses the features from different channels. As shown in Figure 5.1, the proposed model considers both non-visual and visual information. They are extracted from a sequence of video frames 1-2s before the crossing / not crossing (C/NC) event. Non-visual information includes the pedestrian's bounding box, pose key points, and ego-vehicle speed. Visual information contains local context and global context. Local context is the enlarged pedestrian appearance based on the bounding box position. Global context is the semantic segmentation of road, pedestrians (all pedestrians in the scene), and vehicles. They are used because they significantly affect the target pedestrian's crossing decision. We proposed a hybrid way of fusing the the non-visual and visual features, which is justified by comparing different strategies of feature fusion.

Our main contributions are as follows:

- A novel vision-based pedestrian intention prediction framework for ADSs and ADASs. The proposed method employs a novel neural network architecture for utilizing different spatio-temporal features with a hybrid fusion strategy.
- Extensive ablation studies on different feature fusion strategies (early, later, hierarchical, or hybrid), input configurations (adding/removing input channels, using semantic segmentation masks as explicit global context), and visual encoder options (3D CNN or 2D convolution with RNN + attention) to identify the best model layout.
- Demonstrating the efficiency of the proposed method on the commonly used JAAD dataset [86], and achieving state-of-the-art performance on the most recent pedestrian action prediction benchmark [95].

5.1.1 Related Work

Vision-based pedestrian crossing prediction traces back to the works [96] that utilize the Caltech Pedestrian Detection Benchmark [97]. However, the Caltech dataset does not explicitly annotate the crossing behavior of the pedestrians. This gap was later filled by the introduction of JAAD dataset [86] that offers high-resolution videos and explicit crossing behavior annotations. With the release of JAAD dataset, a simple baseline was also created that uses a 2D convolutional neural network (CNN) to encode the features in a given previous frame and then uses a linear support vector machine (SVM) to predict the C/NC event.

Spatio-temporal modeling. Instead of using a single image, most recent works use

image sequences as input to the prediction model due to the importance of temporal information in the prediction task. This leads to spatio-temporal modeling.

Spatio-temporal modeling can be achieved by first extracting visual (spatial) features per frame via 2D CNNs [98] or graph convolution networks (GCNs) [99], and then feeding these features into RNNs such as long-short term memory (LSTM) model [100] and gate recurrent unit (GRU) model [101]. For example, [87, 88, 89] use 2D convolution to extract the visual features from image sequence, and RNNs to encode the temporal information among these features. The encoded sequential visual features are fed into a fully-connected layer to obtain the final intention prediction.

Another way of extracting the sequential visual features is utilizing 3D CNN [102]. It directly captures the spatio-temporal features by replacing the 2D kernels of the convolution and the pooling layers in 2D CNN with 3D counterparts. For example, [103, 104] use 3D CNN based framework (3D DenseNet) to directly extract the sequential visual features from the pedestrian image sequence. The final prediction is achieved in a similar way of using a fully-connected layer.

The crossing intention prediction task can also be combined with scene prediction. A couple of works [105, 106] attempted to decompose the prediction task into two stages. In the first stage, the model predicts a sequence of future scenes using an encoder/decoder network. Then, pedestrian actions are predicted based on the generated future scenes using a binary classifier.

Feature fusion. Instead of end-to-end modeling of visual features, information such as pedestrian's bounding box, body-pose keypoints, vehicle motion, and the explicit global scene context can also be modeled as separate channels as inputs to the prediction model. This requires a proper way of fusing the above information.

For example, [90, 107, 108, 109, 91] introduced human poses/skeletons in pedestrian crossing prediction tasks since human pose can be considered as a good indicator of human behaviors. By extracting the pose keypoints from cropped pedestrian images, crossing behavior classifiers are built based on the human pose feature vectors. Improvement in prediction accuracy shows the effectiveness of using pose features. However, these methods either only rely on human pose features without considering other important features or pay less attention to feature fusion.

Some other methods focused on novel fusion architecture. For instance, [92] proposed SF-GRU, a stacked RNN-based architecture, to hierarchically fuse five feature sources (pedestrian appearance, surrounding context, pose, bounding box, and ego-vehicle speed) for pedestrian crossing intention prediction. Nevertheless, this method does not take global context into account. [93] proposed a multi-modal based prediction system that integrates four feature sources (local scene, semantic map, pedestrian motion, and ego-motion). The global context (semantic map) is utilized, but it lacks other important features such as human pose. [94] proposed a multi-task based prediction framework to take advantages of feature sharing and multi-task learning. It fuses four feature sources (semantic map, pedestrians' trajectory, grid locations, and ego-motion). However, local context and human pose are not considered in the model.

Very recently, more datasets such as PIE [89] and PePScenes [110] provide more annotations for fusing different features. A benchmark was also released with the PCPA model [95]. They create more room for researchers to explore the task of vision-based pedestrian crossing intention prediction.

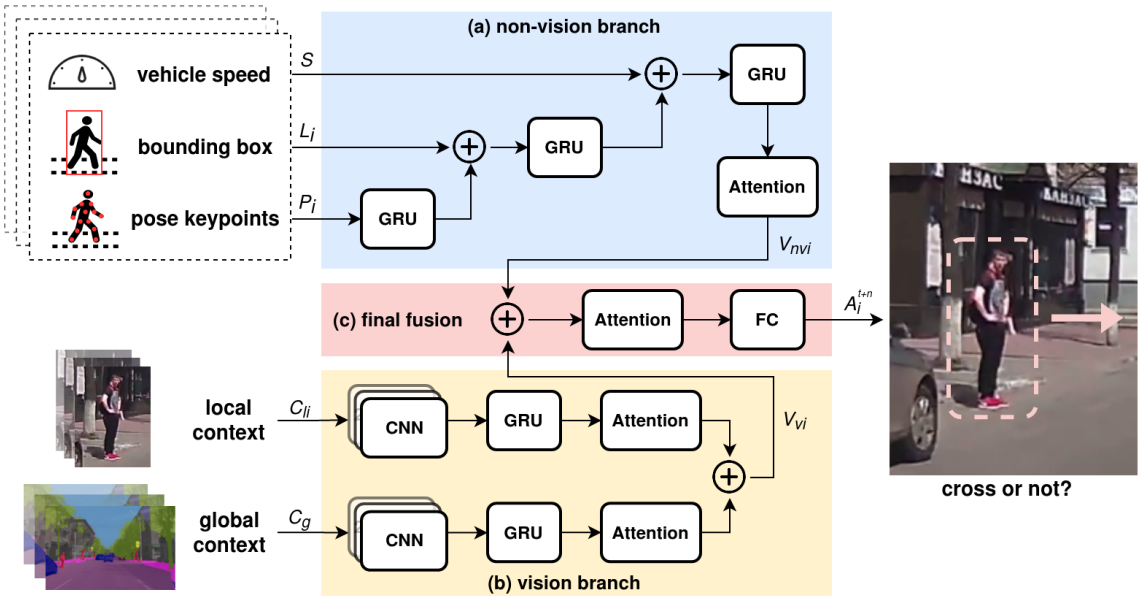

Figure 5.2: **Overview of the proposed pedestrian crossing intention prediction model.** The yellow part denotes the fusion of visual features. 2D convolutional features of local context and global context are encoded by GRU and fed to the attention blocks respectively. Two outputs are concatenated as final visual features. The blue part denotes the fusion of local features (non-visual). These non-visual features are encoded by GRU and fused hierarchically, and then fed to an attention block to obtain the final non-visual features. The red part denotes the final fusion. Final visual features and final non-visual features are concatenated and fed to an attention block. A fully-connected (FC) layer is then applied to make the final prediction.

5.2 Proposed Method

5.2.1 Problem formulation

The task of vision-based pedestrian crossing intention prediction is formulated as follows. Given a sequence of observed video frames from the vehicle's front view and the relevant information of ego-vehicle motion, the goal is to design a model that can estimate the probability of the target pedestrian i's action $A_i^{t+n} \in \{0, 1\}$ of crossing the road, where t is the specific time of the last observed frame and n is the number of frames from the last observed frame to the crossing / not crossing (C/NC) event.

In the proposed model, explicit features such as pedestrian's bounding box, pose keypoints, local context (cropped image around the pedestrian), and global context (semantic segmentation) are firstly extracted. They are used together with the vehicle's speed as separate channels that serve as the input to the prediction model. Therefore, our model has the following input sources:

- The sequential local context around pedestrian i:

$$C_{li} = \{c_{li}^{t-m}, c_{li}^{t-m+1}, ..., c_{li}^{t}\};$$

- The 2D location trajectory of pedestrian i denoted by bounding box coordinates (top-left points and bottom-right points):

$$L_i = \{l_i^{t-m}, l_i^{t-m+1}, ..., l_i^t\};$$

- Pose keypoints of pedestrian i:

$$P_i = \{p_i^{t-m}, p_i^{t-m+1}, ..., p_i^t\};$$

- Speed of ego-vehicle:

$$S = \{s^{t-m}, s^{t-m+1}, ..., s^t\};$$

- The sequential global context denoted by the mask of semantic segmentation:

$$C_g = \{c_g^{t-m}, c_g^{t-m+1}, ..., c_g^t\}.$$

Each source has a sequence of length $m + 1$. The input sources are illustrated in Figure 5.2.

5.2.2 Input acquisition

Local context and 2D location trajectory. Local context C_{li} provides visual features of the target pedestrian. 2D location trajectory L_i gives the position change of the target pedestrian in the image. They can be extracted by a detection (e.g. YOLO [111]) and tracking (e.g. SORT [112]) system. In our work, we directly use the ground truth C_{li} and L_i from the dataset, because pedestrian detection and tracking are not the primary focus of this work. Specifically, the local context $C_{li} = \{c_{li}^{t-m}, c_{li}^{t-m+1}, ..., c_{li}^t\}$ consists of a sequence of RGB images of size $[224, 224]$ pixels around the target pedestrian. The 2D location trajectory $L_i = \{l_i^{t-m}, l_i^{t-m+1}, ..., l_i^t\}$ consists of target pedestrian's bounding box coordinates, i.e.,

$$l_i^{t-m} = \{x_{it}^{t-m}, y_{it}^{t-m}, x_{ib}^{t-m}, y_{ib}^{t-m}\},$$

where $x_{it}^{t-m}, y_{it}^{t-m}$ denotes the top-left point and $x_{ib}^{t-m}, y_{ib}^{t-m}$ bottom-right point.

Pedestrian pose keypoints. Pedestrian pose keypoints represent the target pedestrian's detailed motion, i.e., the posture at each frame while moving. They can be obtained by applying a pose estimation algorithm on the local context C_{li}. Since the applied JAAD dataset does not provide ground truth pose keypoints, we utilize pre-trained OpenPose model [113] to extract the pedestrian pose keypoints $P_i = \{p_i^{t-m}, p_i^{t-m+1}, ..., p_i^t\}$, where p is a 36D vector of 2D coordinates that contain 18 pose joints, i.e.,

$$p_i^{t-m} = \{x_{i1}^{t-m}, y_{i1}^{t-m}, x_{i2}^{t-m}, y_{i2}^{t-m}, ..., x_{i18}^{t-m}, y_{i18}^{t-m}\}.$$

Ego-vehicle speed. Ego-vehicle speed S is a major factor that affects the pedestrian's crossing decision. It can be directly read from the ego-vehicle's system. Since the dataset contains the annotation of ego-vehicle's speed, we directly use the ground truth labels for the vehicle speed $S = \{s^{t-m}, s^{t-m+1}, ..., s^t\}$.

Global context. Global context $C_g = \{c_g^{t-m}, c_g^{t-m+1}, ..., c_g^t\}$ offers the visual features that account for multi-interactions between the road and road users, or among road users. In our work, we use pixel-level semantic masks to represent the global context. The semantic masks classify and localize different objects in the image by labeling all the

pixels associated with the objects the a pixel value. Since the JAAD dataset does not have annotated ground truth of semantic masks, we use DeepLabV3 model [114] pretrained on Cityscapes Dataset [115] to extract the semantic masks and select important objects (e.g. road, street, pedestrians and vehicles) as the global context. For the model to learn the interactions between the target pedestrian i and these objects, the target pedestrian is masked by an unique label. The mask area uses the target pedestrian i's bounding box (obtained from L_i). The semantic segmentation of all input frames are scaled to be the size of $[224, 224]$ pixels, which is the same as the local context.

5.2.3 Model architecture

The overall architecture is shown in Figure 5.2. It consists of CNN modules, RNN modules, attention modules, and a novel way of fusing different features.

CNN module. We use VGG19 [98] model pre-trained on ImageNet dataset [116] to build the CNN module. Sequential RGB images are collected as a 4D array input with the dimensions of [number of observed frames, row, cols, channels] ($[16, 224, 224, 3]$ in this work), and then loaded by the CNN module. First, the feature map of every image from the fourth maxpooling layer of VGG19 is extracted with size $[512, 14, 14]$. Second, every feature map is averaged by a pooling layer with a 14×14 kernel, and then flattened and concatenated, to obtain a final feature tensor with size $[16, 512]$, as sequential visual features.

RNN module. We use gated recurrent unit (GRU) [101] to build the RNN module. The reason of choosing GRU is that GRU is more computationally efficient than its counterpart LSTM [100], which is older, and its architecture is relatively simple. The applied GRUs have 256 hidden units, which result in a feature tensor of size $[16, 256]$

Attention module. Attention module [117], by selectively focusing on parts of features, is used for better memorizing sequential sources. Sequential features (e.g. the output of RNN-based encoder) are represented as hidden states $h = \{h_1, h_2, ..., h_e\}$. The attention weight is computed as:

$$\alpha = \frac{exp(score(h_e, \tilde{h_s}))}{\sum_{s'} exp(score(h_e, \tilde{h_{s'}}))},$$

where $score(h_e, \tilde{h_s}) = h_e^T W_s \tilde{h_s}$ and W_s is a weight matrix. Such attention weight trades off the end hidden state h_e with each previous source hidden state $\tilde{h_s}$. The output vector of the attention module is produced as

$$V_{attention} = tanh(W_c[h_c; h_e]),$$

where W_c is a weight matrix, and h_c is the sum of all attention weighted hidden states as $h_c = \sum_{s'} \alpha \tilde{h_{s'}}$. The output of the attention module in our work is a feature tensor with size $[1, 256]$.

Hybrid fusion. We applied a hybrid way of fusing the features from different sources. The strategy is shown in Figure 5.2. The proposed architecture has two branches, one for non-visual features and one for visual features.

The non-vision branch fuses three non-visual features (bounding boxes, pose keypoints, and vehicle speed). They are hierarchically fused according to their complexity and level of abstraction. The later stage of fusion, the closer impact of the fused feature on final prediction. This is illustrated in Figure 5.2(a). First, sequential pedestrian pose keypoints P_i are fed to a RNN-based encoder. Second, the output of the first stage is concatenated

with 2D location trajectory L_i and fed to a new RNN-based encoder. Last, the output of the second stage is concatenated with ego-vehicle speed S and fed to a final RNN-based encoder. The output of the final encoder is then fed to an attention block to obtian the final non-visual feature vectors V_{nvi}.

The vision branch fuses two visual features, consisting of local context (enlarged pedestrian appearance around the bounding box) and global context (semantic segmentation of important objects in the whole scene), as shown in Figure 5.2(b). Local context C_{li} is encoded by first extracting spatial features from the CNN module (as explained in the previous section) and then extracting temporal features from the GRU module. Global context C_g is encoded in the same way. Both local and global features are then fed into their attention modules, and finally, concatenated together to generate final visual feature vectors V_{vi}.

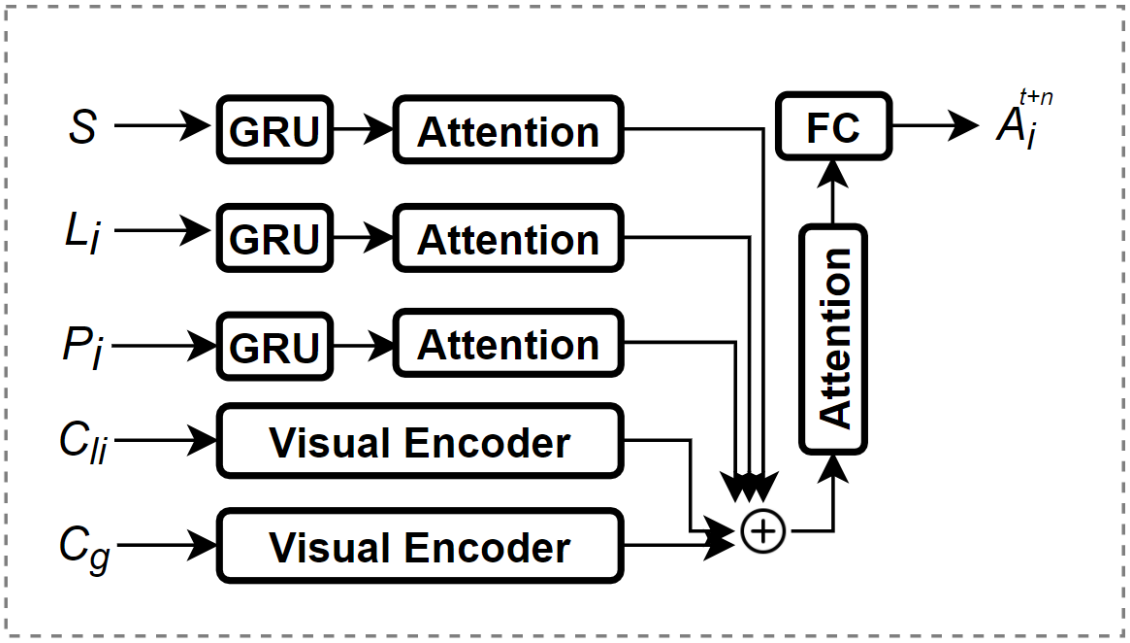

Figure 5.3: Illustration of Later Fusion

Lastly, as shown in Figure 5.2(c), the final non-visual feature vectors V_{nvi} and the final visual feature vectors V_{vi} are concatenated and fed into another attention block, followed by a fully-connection (FC) layer to obtain the final predicted action:

$$A_i^{t+n} = f_{FC}(f_{attention}(V_{nvi}; V_{vi})).$$

5.3 Experiments

5.3.1 Dataset and Benchmark

The proposed model was evaluated using JAAD dataset [86]. It contains two subsets, JAAD behavioral data (JAAD_{beh}) and JAAD all data (JAAD_{all}). JAAD_{beh} contains pedestrians who are crossing (495 samples) or are about to cross (191 samples). JAAD_{all} has additional pedestrians (2100 samples) with non-crossing actions. To create a fair benchmark, the dataset configuration follows the same one as in [95]. It uses a data

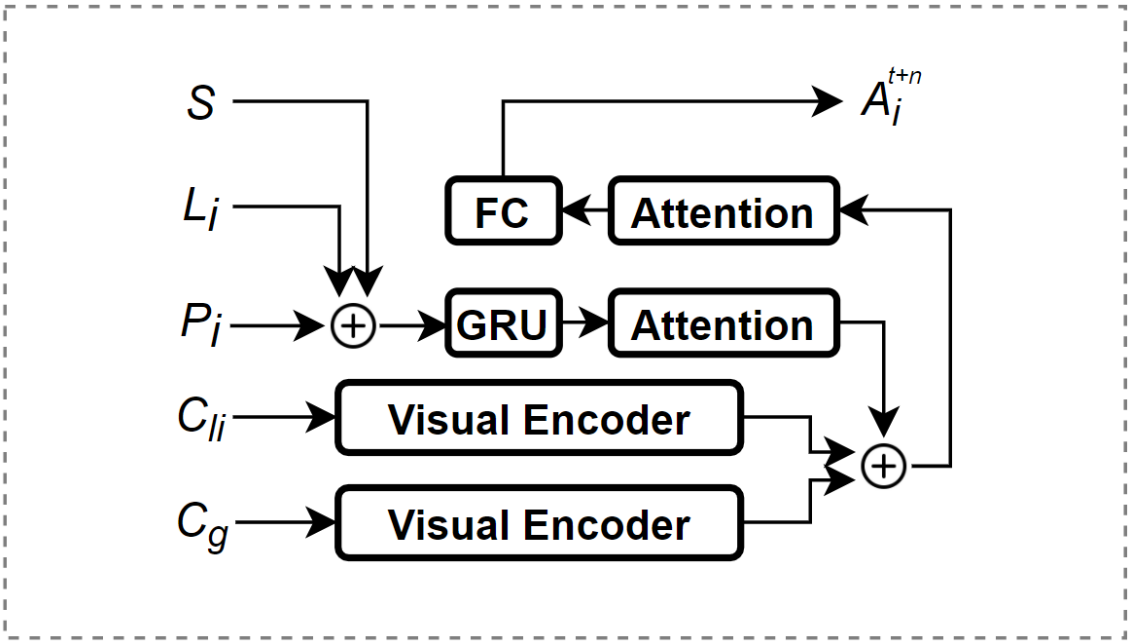

Figure 5.4: Illustration of Early Fusion

sample overlap of 0.8, local context scale of 1.5. The evaluation metrics use accuracy, AUC, F1 score, precision, and recall. They are the most recognized metrics and are used by most related works.

5.3.2 Implementation

In the expriments, the propsoed model was compared with the following methods: SingleRNN [87], SF-GRU [92] and PCPA[95]. We adopted the benchmark implementation released with PCPA model [95]. This benchmark collects the implementations of most pedestrian intention prediction methods. Our model was developed based on the this benchmark. We use a dropout of 0.5 in the attention module, L2 regularization of 0.001 in FC layer, binary cross-entropy loss, Adam optimizer [118], learning rate = 5×10^{-7}, epochs = 40, and batch size = 2. All models were trained and tested on the same split of the dataset, as suggested by the benchmark. Note that JAAD dataset does not provide explicit vehicle speed. Instead, the driver's action is recorded as an abstract encoding of the vehicle speed. The action contains [stopped (0), moving slow (1), moving fast (2), decelerating (3), accelerating (4)].

5.3.3 Ablation study

An ablation study was also conducted to compare different strategies of fusing different features. In addition to baseline methods (SingRNN [87], SF-GRU [92] and PCPA [95]) and the proposed model (Ours), a total of 7 variants of the proposed model (Ours1, Ours2, ... , Ours7, as indicated in table 5.3 and table 5.4) were trained and compared with the proposed one. First, for the visual encoder, we tried (1) 2D CNN combined with RNN (VGG and GRU in our experiments) and (2) 3D CNN as proposed in the PCPA model. Second, we tried the models with and without the global feature (semantic

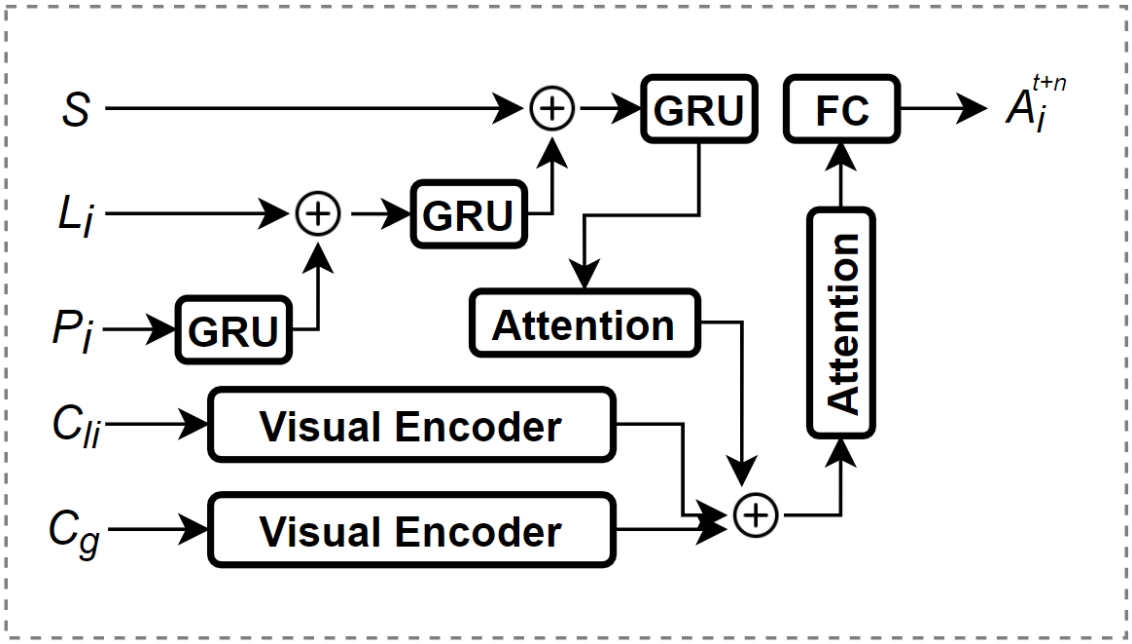

Figure 5.5: Illustration of Hierarchical Fusion

segmentation). Last, we tried different fusion strategies that include later fusion, early fusion, and hierarchical fusion so that they can be compared with the proposed hybrid fusion strategy. Later fusion (Figure 5.3) is the same as the one proposed in PCPA [95]. Early fusion (Figure 5.4) concatenates non-visual features and visual features directly and then send them into one RNN module followed by an attention module. Hierarchical fusion (Figure 5.5) gradually fuses both visual features and non-visual features by RNN modules using the same way as in Figure 5.2(a), followed by an attention module.

5.4 Results

Table 5.1: Quantitative Results on JAAD Behavior Subset

Models	Model Variants			JAAD_{beh}				
	Visual Encoder	Global Context	Fusion Approach	Accuracy	AUC	F1	Precision	Recall
SingleRNN [87]	VGG + GRU	✗	✗	0.59	0.52	0.71	0.64	0.80
SF-GRU [92]	VGG + GRU	✗	hierarchical-fusion	0.58	**0.56**	0.65	**0.68**	0.62
PCPA [95]	3D CNN	✗	later-fusion	0.53	0.53	0.59	0.66	0.53
Ours	VGG + GRU	✓	hybrid-fusion	**0.62**	0.54	**0.74**	0.65	**0.85**

• The **bold** result means the best in the models.

Table 5.2: Quantitative Results on JAAD All Dataset

Models	Model Variants			JAAD_{all}				
	Visual Encoder	Global Context	Fusion Approach	Accuracy	AUC	F1	Precision	Recall
SingleRNN [87]	VGG + GRU	✗	✗	0.79	0.76	0.54	0.44	0.71
SF-GRU [92]	VGG + GRU	✗	hierarchical-fusion	0.76	0.77	0.53	0.40	0.79
PCPA [95]	3D CNN	✗	later-fusion	0.76	0.79	0.55	0.41	**0.83**
Ours	VGG + GRU	✓	hybrid-fusion	**0.83**	**0.82**	**0.63**	**0.51**	0.81

• The **bold** result means the best in the models.

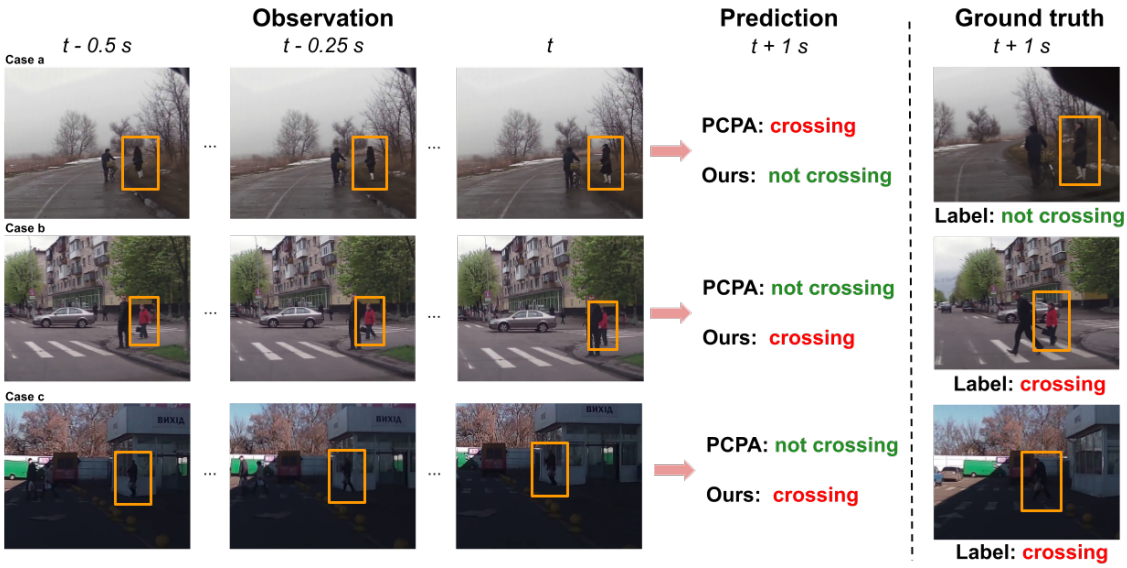

Figure 5.6: **Qualitative results on the JAAD dataset** produced by PCPA [95] and our proposed model (Ours). The target pedestrians in images are enclosed by **bounding boxes. The prediction results as well as ground truth labels are represented as crossing or not crossing.**

5.4.1 Quantitative Results

Table 5.1 shows the qualitative results on JAAD_{beh} dataset. It compares the proposed model with baseline models of SingleRNN [87], SF-GRU [92] and PCPA [95]. The proposed model achieved the best scores in accuracy, F1, and recall. F1 score is an balanced metirc considering both recall and percision. For binary classification, it is the most important indicator of how good the model is. Our model achieved about 4% improvement in F1. In addition to F1, accuracy is another important metric. Our model also achieved the best.

Table 5.2 shows the qualitative results on JAAD_{all} dataset. JAAD_{all} has additional samples of non-crossing behaviors. It is larger than JAAD_{beh}. The data distribution is more similar to real world scenarios. As illustrated by table 5.2, the proposed achieved the best in accuracy, AUC, F1, precision. Similar to the results in JAAD_{beh}, our model achieved the best in terms of the two important metrics, F1 and accuracy.

Table 5.3 and table 5.4 show the results of ablation study on JAAD_{beh} dataset and JAAD_{all}, respectively. Different model variants are denoted by Ours1, Ours2, ..., Ours7. By comparing Ours5 with Ours4 and Ours1 with the PCPA model, it shows that introducing global context can improve the model performance. If we further compare Ours4 with the PCPA model, it shows that using 2D CNN plus RNN instead of 3D CNN for visual feature encoding also has the advantage of extracting spatio-temporal features, hencing improving model performance. In terms of fusion strategies, the proposed hybrid fusion strategy achieved the best performance, which can be identified by comparing Ours with Ours5, Ours6, and Ours7.

Table 5.3: Ablation Study on JAAD Behavior Subset

Models	Model Variants			JAAD_{beh}				
	Visual Encoder	Global Context	Fusion Approach	Accuracy	AUC	F1 Score	Precision	Recall
Ours	VGG + GRU	✓	hybrid-fusion	0.62	0.54	**0.74**	0.65	**0.85**
Ablations								
Ours1	3D CNN	✓	later-fusion	0.59	0.53	0.69	0.65	0.75
Ours2	3D CNN	✓	early-fusion	0.59	0.54	0.69	0.65	0.74
Ours3	3D CNN	✓	hierarchical-fusion	0.57	0.48	0.70	0.62	0.81
Ours4	VGG + GRU	✗	later-fusion	0.59	0.51	0.72	0.63	0.83
Ours5	VGG + GRU	✓	later-fusion	**0.64**	**0.59**	0.73	**0.68**	0.78
Ours6	VGG + GRU	✓	early-fusion	0.60	0.56	0.70	0.67	0.73
Ours7	VGG + GRU	✓	hierarchical-fusion	0.54	0.50	0.64	0.63	0.65

• The **bold** result means the best in the models.

Table 5.4: Ablation Study on JAAD All Dataset

Models	Model Variants			JAAD_{all}				
	Visual Encoder	Global Context	Fusion Approach	Accuracy	AUC	F1 Score	Precision	Recall
Ours	VGG + GRU	✓	hybrid-fusion	**0.83**	**0.82**	**0.63**	**0.51**	0.81
Ablations								
Ours1	3D CNN	✓	later-fusion	0.77	0.77	0.54	0.42	0.76
Ours2	3D CNN	✓	early-fusion	0.77	0.74	0.51	0.41	0.69
Ours3	3D CNN	✓	hierarchical-fusion	0.78	0.77	0.55	0.43	0.75
Ours4	VGG + GRU	✗	later-fusion	0.75	0.79	0.54	0.40	**0.85**
Ours5	VGG + GRU	✓	later-fusion	0.77	0.80	0.56	0.43	0.84
Ours6	VGG + GRU	✓	early-fusion	0.79	0.74	0.52	0.43	0.66
Ours7	VGG + GRU	✓	hierarchical-fusion	0.80	0.81	0.59	0.46	0.84

• The **bold** result means the best in the models.

5.4.2 Qualitative Results

Figure 5.6 provides qualitative results for the proposed model of pedestrian crossing intention prediction. We mainly compared the proposed method with the PCPA model. In the provided examples, our method correctly predicted the crossing intention but the PCPA failed. Taking a closer look at the examples, the following argument is raised. Without utilizing the global context, the task of crossing intention prediction may face the problems of (1) unknown direction of the pedestrian (Case a in Figure 5.6), (2) occlusion (Case b in Figure 5.6), and (3) poor vision (Case c in Figure 5.6). Global context can provide additional information to account for the interaction between the whole scene and the target pedestrian.

Figure 5.7 provides more qualitative results to analyze the advantages of the proposed model over the PCPA model as well as a few of failure cases. Figure 5.7-(a) and Figure 5.7-(b) show the cases when the proposed model generated correct predictions but the PCPA failed. The main reason is that our model considers the global visual context that contains the semantic segmentation of the drivable area. The model can learn from this whether the pedestrian is moving toward or on the drivable area, which is an important indicator of pedestrian crossing intention.

Figure 5.7-(c) and Figure 5.7-(d) show the cases when both the proposed model and the PCPA failed. Figure 5.7-(c) shows an intersection scenario. The pedestrian (yellow bounding box) has already crossed the ego road but near the edge of the road on the other side. This may mislead the model to generate a prediction of crossing. The failure in figure 5.7-(d) was mainly due to the poor illumination such that the model cannot obtain enough detailed feature.

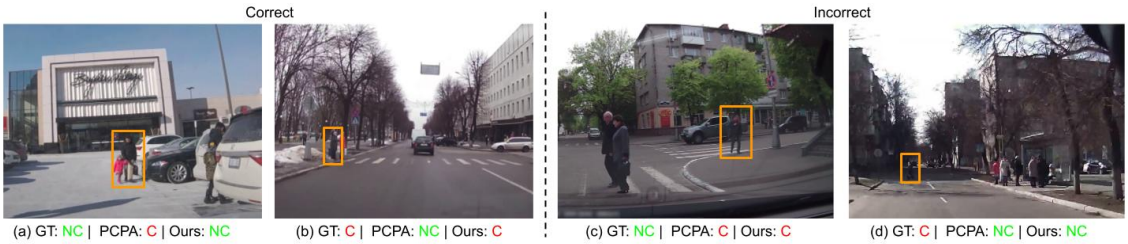

Figure 5.7: More qualitative results. (a) and (b) show the cases of correct predictions by the proposed model but the PCPA failed. (c) and (d) show the results when both the proposed and the PCPA model failed.

5.5 Conclusion

In this chapter, we proposed a novel method for vision-based pedestrian crossing intention prediction. Our method explicitly considers the global context as a channel representing the interaction between the target pedestrian and the whole scene. We also proposed a hybrid fusion strategy for different features using 2D CNNs, RNNs, and attention mechanisms. Experiments on the JAAD dataset show that the proposed method achieves the state-of-the-art against baseline methods in the pedestrian action prediction benchmark.

Future work can focus on improving our model's robustness in unexpected situations, e.g., poor vision and occlusion. Additionally, feature fusion with more information sources can be explored. Finally, fine-tuning the model for particular pedestrian subsets, such as children and disabled people, can increase overall safety and performance.

Chapter 6

Pedestrian Emergence Estimation and Occlusion-Aware Risk Assessment

This chapter is derived from the published work in [119].

6.1 Background

612,500 pedestrians were killed in 2013 by road traffic injuries, which was the number one cause of death among the age group 15-29 [120]. Fully automated driving systems are seen as possible remedies for reducing road traffic fatalities due to the fact that they do not possess the fundamental issues of human drivers, such as failure to comply with the rules, lack of attention while driving, etc. Furthermore, decision-making for automated driving systems (ADS) is a challenging area that plays a key role in fully automated systems. Especially, developing intelligent systems taking precautions actions for objects that are currently unobservable but interacting with the ego vehicle in the future has attracted much attention recently. Currently, only up to Level 3 systems [121] are available in the market [122].

The motion prediction and risk assessment are vital in taking precautious actions. Lefevre et al. [123] categorized the motion prediction and risk assessment methods into four categories, and stated that despite being computationally more demanding, interaction-aware methods are more reliable than other methods. As an alternative to the interaction-aware methods, occlusion-aware [124] risk assessment methods have been proposed. Those alternatives vary from solutions based on partially/mixed observable Markov decision processes (POMDP/MOMDP) [9, 125, 126, 127] , to solutions based on set-based methods [128, 129, 130, 131, 132, 133]. Set-based methods and exploitation of behavior of other participants in a rule-based fashion were used [134, 135, 136, 137, 12]. Althoff et al. [134] improved the reachability analysis to obtain PID controllers and implemented their method on a real car.

There are currently two shortcomings of available methods in the literature. First, little attention has been paid to using visible information and prior knowledge to predict pedestrian emergence out of occluded areas. Second, human driver performance under occlusion and limited visibility conditions have been mostly neglected.

This chapter introduces a novel occlusion-aware risk assessment system for ADSs. The

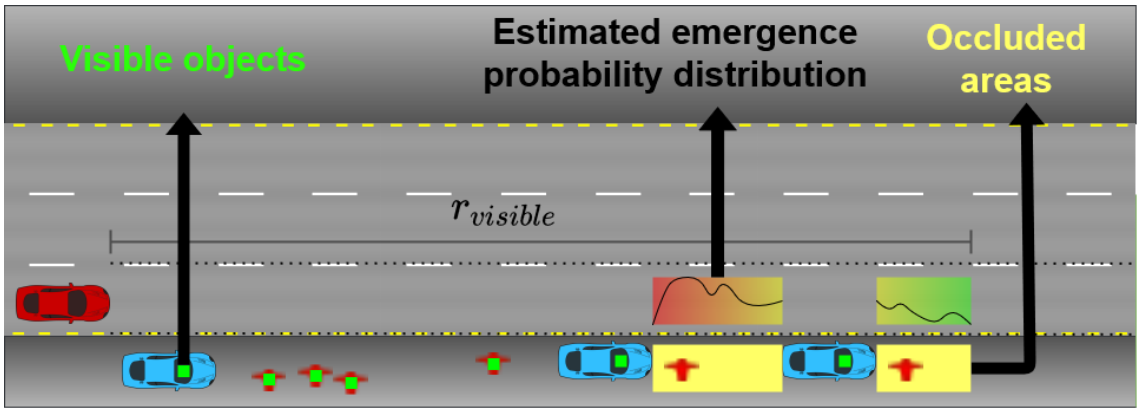

Figure 6.1: An example of environmental cues and how a human driver would easily decide a similar scheme for emergence probability

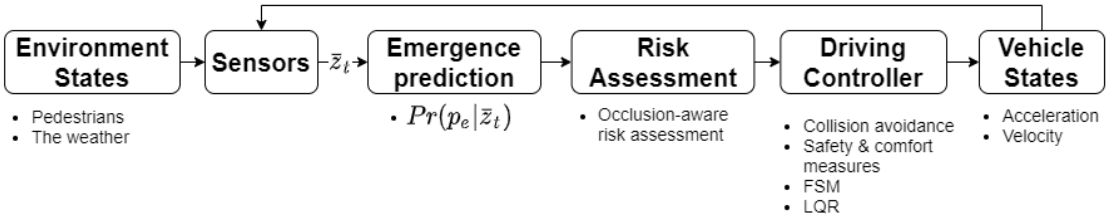

Figure 6.2: Overview of the proposed framework

proposed method can estimate pedestrian emergence probability from occluded areas and adjust its driving policy accordingly. Our method's overview is demonstrated in Fig. 6.1, where the red ego vehicle uses information such as visible pedestrians and parked cars to assess the probability of emerging pedestrians to derive an optimal driving policy.

The main contributions can be summarized as follows:

- Using contextual information for estimating pedestrian emergence from occluded areas
- Employing the estimated emergence probabilities in an occlusion-aware risk assessment framework
- Incorporating the proposed occlusion-aware risk assessment framework into a longitudinal vehicle controller for realizing comfortable and safe driving

The chapter continues as follows: Section 6.2 explains the framework of the current proposal in detail. Section 6.3 introduces 3 baseline controllers to compare against the proposed method and provide details about the simulation environment. Section 6.4 evaluates the proposed method and demonstrates that the proposed method's performance is significantly better over several metrics quantitatively. Section 6.5 concludes and discusses possible future directions.

6.2 Methodology

Here, we propose a novel occluded emerging pedestrian distribution estimation and risk assessment method. The framework includes three parts: (1) estimating pedestrian emergence from occlusions; (2) risk assessment; and (3) the controller. The overview of the proposed method is demonstrated in Fig. 6.2. A simulation environment is created using Python programming. The parked cars are placed randomly on the two sides of the road to create the occlusions. The occlusions and visible objects to the ego vehicle are calculated using a simple visibility polygon algorithm. For this task, we assume that the ego vehicle has sensors that can identify each object and locate the visible objects within the visible range $r_{visible}$ and the viewing angle. Nevertheless, our proposal can be compatible with any sensors that can detect, i.e. can label an object as in "a vehicle" or "a pedestrian", and locate the objects. To generate realistic scenarios, pedestrians are modeled as point-mass objects with instant velocity change because the time passed until a pedestrian accelerates to its walking velocity is negligible. We investigated the empirical data of the pedestrian velocity from [138]. Accordingly, we used a Gaussian distribution with $\mu_{v\_ped} = 1.5m/s$ and $\sigma_{v\_ped} = 0.6m/s$. In addition, since not all the pedestrians that a driver sees on the sidewalk will cross the street, we generated some pedestrians that would not cross the street.

6.2.1 Estimating Pedestrian Emergence from Occlusions

The goal is to find the distribution of the emerging pedestrians from occlusions. Our solution is to use contextual information such as the presence of the parked cars, crosswalks, and visible pedestrians. As we set forth previously, the POMDP representations and Bayesian filtering are computationally demanding. Therefore, we suggest a solution that generates a posterior belief without explicit Bayesian filtering. Specifically, to reduce the computational demand of the estimation of the posterior distribution, a collection of piecewise weighted sigmoid functions was utilized. Then, the probability approximation becomes:

$$\Pr(p_e|\bar{z}_t) \simeq \frac{1}{1 + e^{-\bar{w}^T \bar{z}_t}} \tag{6.1}$$

$$\bar{z}_t = [1, n_1, n_2, d_1, d_2, d_3]^T \tag{6.2}$$

where $\Pr(p_e|\bar{z}_t)$ is the pedestrian emergence from occlusion probability, p_e is the pedestrian emergence event, and $\bar{z}_t$ is the observation vector; n_1 is the normalized density of parked cars, n_2 is the normalized density of visible pedestrians, d_1 is the normalized distance to the crosswalk, d_2 is the normalized distance to the closest parked car, and d_3 is the normalized distance to the closest visible pedestrian. Heuristics have chosen the weight vector of the observations in (6.1). In the case of unobservability, n_1, n_2, n_3 is considered 0 whereas d_1, d_2, d_3 considered 1 which is the normalized value of $r_{visible}$.

6.2.2 Risk Assessment

Risk assessment is the most crucial part and the main contribution of this work. Passengers can feel both the force and the change in the force exerted on their bodies. The change in force is widely known as *jerk*. Furthermore, both acceleration and jerk are perceived omnidirectionally by a human body. Therefore, a $a_{comfort}$, and a $j_{comfort}$ value can be

defined under the assumption that values between $[-a_{comfort}, a_{comfort}]$ for acceleration, and values between $[-j_{comfort}, j_{comfort}]$ for jerk are defined as *comfortable*.

Due to physical limitations, before reaching a steady acceleration of choice, the acceleration's magnitude rises linearly for a ramp time t_{ramp}. Then, the ramp time from the magnitude of 0 to $a_{comfort}$ is denoted by $t_{ramp,comfort}$ whereas the ramp time from the magnitude of 0 to a_{max} is denoted by $t_{ramp,min}$. Also, the distance traveled before stopping by the ego vehicle, with the magnitude of deceleration decided to be $a_{comfort}$ and the ramp time decided to be $t_{ramp,comfort}$, is denoted by $d_{stop,comfort}$ whereas the distance traveled with the magnitude of deceleration decided to be a_{max} and the ramp time physically possible being $t_{ramp,min}$ is denoted by $d_{stop,min}$ Note that, $t_{ramp,min}$, a_{max}, and d_{min} are physical limitations and constant for the ego vehicle whereas $t_{ramp,comfort}$, $a_{comfort}$ and $d_{stop,comfort}$ are variables. Moreover, the minimum distance using the comfortable values is denoted by $d_{stop,comfort,min}$. Using the notation, the ego vehicle can have imaginary zones; here, it is defined as *risk zones*. Assuming that $d_{stop,min} \leq d_{stop,comfort,min}$, the so-called risk zones:

- *danger zone* spans $[0, d_{stop\_min}]$
- *discomfort zone* spans $[d_{stop\_min}, d_{stop\_comfort}]$
- *safety zone* spans $[d_{stop\_comfort}, r_{visible}]$ meters ahead, away from the ego vehicle.

6.2.3 Driving Policy

The proposed occlusion-aware risk assessment framework is incorporated into a safe and robust driving policy. The driving policy is given in algorithm 3.

Longitudinal control. A controller is required to achieve the necessary control actions. We use a modified, LQR-based control strategy based on a point-mass discrete-time vehicle dynamics model given by:

$$\bar{x}_{crs,k+1} = v_{k+1} = v_k + \Delta t * a_k \tag{6.3}$$

$$d_{k+1} = d_k - \Delta t * v_k \tag{6.4}$$

where v_k is the velocity of the ego vehicle, a_k is the acceleration value of the ego vehicle d_k, in (6.4), is the lateral distance to the imaginary line tangent to the close side of the visible pedestrian to the vehicle, and also perpendicular to the ego vehicle's direction. Here, we used k to distinguish discrete-time representation from continuous-time representation. Henceforth, we replace k with t. Using the vector notation again with the discrete time equations:

$$\bar{x}_{yld,t+1} = \begin{bmatrix} d_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d_t \\ v_t \end{bmatrix} + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix} \bar{u}_t \tag{6.5}$$

$$\bar{u}_t = a_t \tag{6.6}$$

When there is no visible pedestrian to be yielded, the state-space is represented by $\bar{x}_{crs,t}$. By contrast, when there is at least one visible pedestrian, the state-space is represented by $\bar{x}_{yld,t}$; the distance between the closest pedestrian and the vehicle is one of the states. Then, the traditional LQR optimization scheme uses the quadratic cost function to optimize the

control action. The cost function is defined as follows:

$$c_t = (\bar{x}_t - \bar{x}_{ref})^T Q(\bar{x}_t - \bar{x}_{ref}) + \bar{u}_t^T R \bar{u}_t \tag{6.7}$$

$$J = \sum_{t=0}^{T-1} c_t \tag{6.8}$$

$$\bar{u}_t = -K(\bar{x}_t - \bar{x}_{ref}) \tag{6.9}$$

where K is the Kalman gain, and the optimal control is obtained by minimizing the cumulative cost J provided that appropriate Q and R matrices are selected.

However, the traditional LQR is an unconstrained-optimization scheme. Consequently, this proposal with the traditional LQR will generate unrealistic and high-frequency control actions under noise in the measurements or the system. Those high-frequency modes of the action can be very uncomfortable and dangerous for the passengers. The equations of the LQR scheme can be revised with a small modification to limit the jerk:

$$a_{t+1} = a_t + \Delta t * j_t \tag{6.10}$$

$$\begin{bmatrix} d_{t+1} \\ v_{t+1} \\ a_t \end{bmatrix} = \begin{bmatrix} 1 & -\Delta t & 0 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_t \\ v_t \\ a_{t-1} \end{bmatrix} + \begin{bmatrix} 0 \\ j_1 * \Delta t^2 \\ j_1 * \Delta t \end{bmatrix} j_t \tag{6.11}$$

$$\begin{bmatrix} v_{t+1} \\ a_t \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_t \\ a_{t-1} \end{bmatrix} + \begin{bmatrix} j_2 * \Delta t^2 \\ j_2 * \Delta t \end{bmatrix} j_t \tag{6.12}$$

where the state-space for yielding with limited jerk is described in (6.11), and the state-space for cruising with limited jerk is described in (6.12). Since the jerk of a vehicle is not a realizable control input, the LQR controller's implementation with the modification will be similar; the controller will actuate the control input a_t obtained from (6.11) or (6.12). j_1 is the maximum allowed jerk while yielding, whereas j_2 is the maximum allowed jerk while cruising as the ego vehicle must have more agility while yielding. Therefore, j_1 is chosen as $2m/s^3$, the maximum jerk for aggressive driving; by contrast, j_2 is chosen as $0.9m/s^2$ the maximum jerk for normal driving [139].

Finally, we have chosen appropriate Q and R matrices for both the *cruising* and *yielding* and computed the resulting full-state-feedback coefficients K in MATLAB–R2019a as follows:

$$Q_{crs} = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix} \qquad R_{crs} = \begin{bmatrix} 1000 \end{bmatrix}$$

$$Q_{yld} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \qquad R_{yld} = \begin{bmatrix} 1500 \end{bmatrix}$$

$$K_{crs} = \begin{bmatrix} 0.9047 & 0.9074 \end{bmatrix}$$

$$K_{yld} = \begin{bmatrix} -0.0532 & 0.3139 & 0.3792 \end{bmatrix}$$

Collision Avoidance. After calculating the span of so-called risk zones, assessing the current risk, it is also necessary to determine the collision risk. The quantity measurements, i.e., the normalized density are discrete; therefore, discontinuities in the function appear at the critical instance of observing or losing sight of a parked car or a pedestrian.

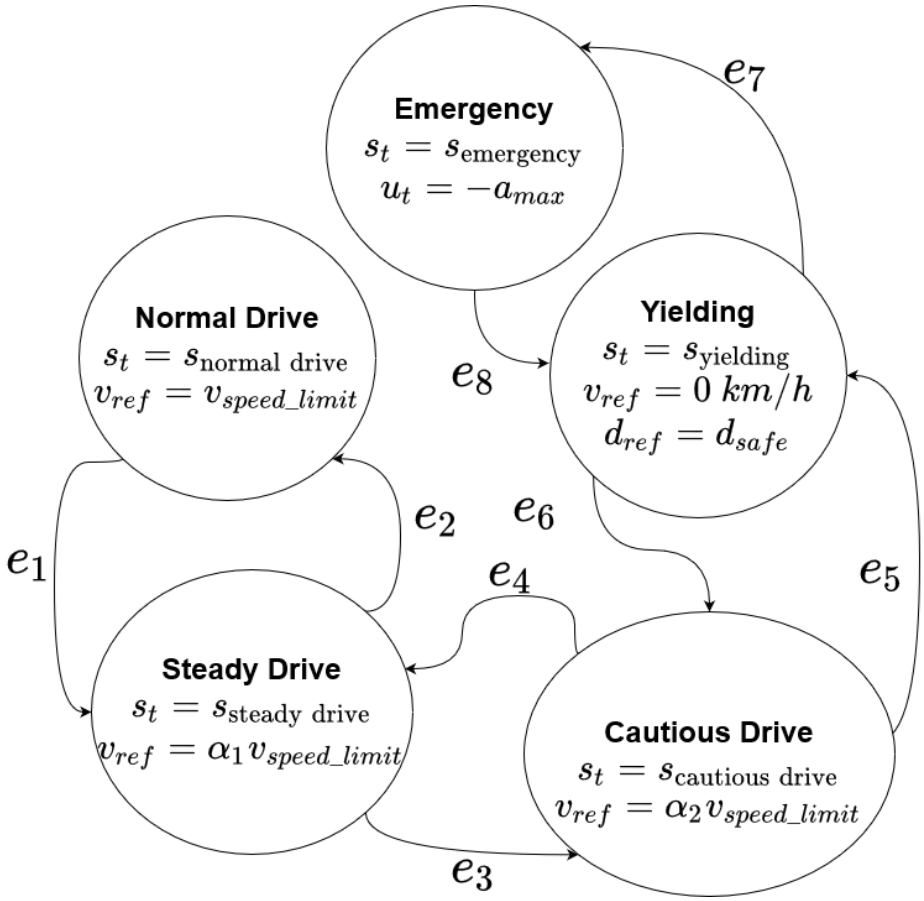

Figure 6.3: The proposed FSM to utilize the quantified risk

Consequently, we use thresholds to divide the function into regions. for a robust control architecture.

We designed an FSM controller with distinct states for each region in the function. As the risk decreases, the ego vehicle could be more encouraged to drive the speed limit while slowing down to a proportion of the speed limit in a risky situation. We named the FSM states for each region as follows:

- *Normal Drive*: The ego vehicle is given the reference velocity of the speed limit, v_{speed_limit},
- *Steady Drive*: The ego vehicle is given the reference velocity of some proportion of the speed limit, $\alpha_1 v_{speed\_limit}$,
- *Cautious Drive*: The ego vehicle is given the reference velocity of another proportion of the speed limit, $\alpha_2 v_{speed\_limit}$.

Also, we consider two additional crucial driving modes: yielding pedestrians when necessary and engaging in maximum braking to avoid collision under dangers. The resulting FSM architecture is demonstrated in Fig. 6.3. Additionally, the state transition conditions of the designed controller are given in Table 6.1. The interpretation of risk inside different *risk zones* should be different because even a small risk of emergence in *danger zone* should be treated with utmost care. In contrast, the same level of risk of emergence

Table 6.1: State Transitions for the FSM in Fig. 6.3

State Transition	Explanation
e_1	$Pr(p_e\|\bar{z}_t) > l_{steady}$
e_2	$Pr(p_e\|\bar{z}_t) \leq l_{steady}$
e_3	$Pr(p_e\|\bar{z}_t) > l_{cautious}$
e_4	$l_{steady} \leq Pr(p_e\|\bar{z}_t) \leq l_{cautious}$
e_5	A visible pedestrian to be yielded
e_6	No visible pedestrian to be yielded
e_7	$TTC_{brake} \geq TTC_{emergency}$
e_8	$TTC_{brake} < TTC_{emergency}$

in *discomfort zone* might not require similar alertness. Therefore, we assigned different threshold values for the different risk zones.

Combining all, the ego vehicle observes the environment from its sensors to predict the distribution of the emerging pedestrians from occlusions; this information is utilized, after risk assessment, differently per *risk zone*, and this inference is going to be converted into a high-level command such as *normal drive*, *yielding*, etc.; finally, the controller is going to choose the appropriate control action, acceleration, to reach the reference state within decided control limits.

The ego vehicle does not require to take precautious action against any pedestrians entering the expected path from inside the safety zone, the algorithm's maximum look-ahead distance should be $d_{stop,comfort}$. Then, the future risk is computed by considering contextual information within a predefined window with the spatial resolution Δd. The maximum risk per risk zone is compared against the thresholds to choose the appropriate deceleration/acceleration limit a_{limit} and jerk limit j_{limit} with the appropriate FSM state to reach the reference velocity v_{ref}. It is also crucial to determine the deceleration limit by the friction coefficient between the tires and the road. Here we assume that the ego vehicle can measure the friction coefficient for different weather conditions.

6.3 Experiments

In order to evaluate the proposal, a straight road is chosen. The road is 96 meters long, and it has three lanes whose width is 3 meters. The scenarios are divided into three categories based on the crowdedness; In *suburban scenarios (sc1)*, there are one or two pedestrian, one or two parked cars and a crosswalk; in *mildly crowded urban scenarios (sc2)*, there are multiple parked cars, multiple pedestrians and a crosswalk; in *very crowded urban scenarios(sc3)*, the parking slots are full, and there are multiple pedestrians and a crosswalk. This way it is going to be possible to observe the strengths and weaknesses of the proposal in various possible scenarios.

We compare the proposal with the 3 baselines that represents a driving aspect.

- *Baseline1 (B1)*: occlusion-unaware, and drives the legal speed limit of the road ($30km/h$). It yields to the pedestrians that are on the road will likely enter its expected path.

- *Baseline2 (B2)*: occlusion-unaware, and drives the two-third of the legal speed limit

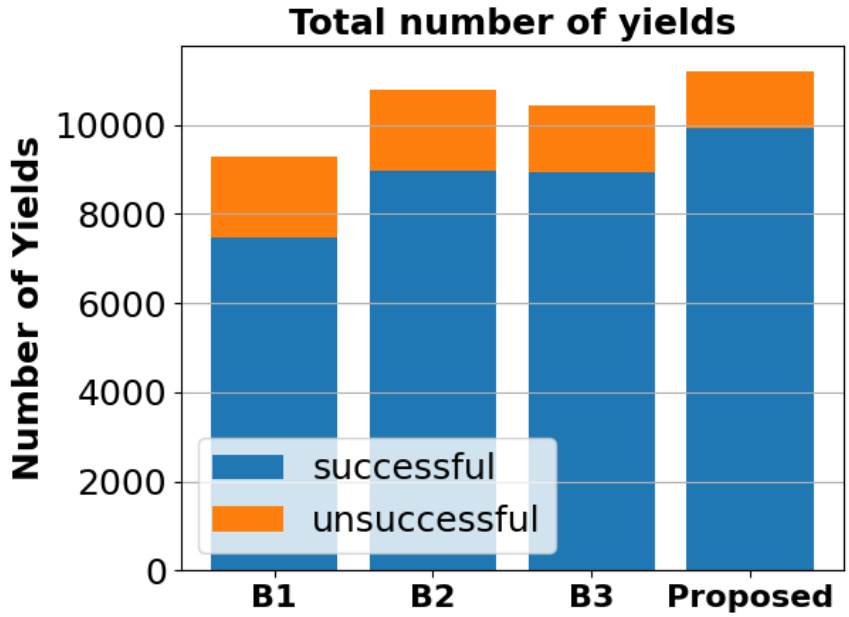

Figure 6.4: Total successful and unsuccessful yields (*mt1*) of different controllers in *mildly crowded urban scenarios* (*sc2*)

of the road ($20km/h$). It yields to the pedestrians that are on the road will likely enter its expected path.

- *Baseline3 (B3)*: occlusion-unaware, and drives the one-third of the legal speed limit of the road ($10km/h$) only if it observes a crosswalk which closer than a specific distance; otherwise, it drives the speed limit. It yields to the pedestrians that are on the road will likely enter its expected path.

6.3.1 Metrics

An extensive survey on important metrics to determine the driving quality of AVs has been made by [140]. Combining the metrics from [140] with additional safety metrics, we have decided to use the following metrics:

- *mt1*: The total number (successful/unsuccessful) of yields
- *mt2*: Deceleration (mean, std)
- *mt3*: The total number of successful finishes
- *mt4*: Time of emergency braking (mean, std)

6.4 Results

The results of the simulation, 1000 episodes per scenario, is demonstrated in Table 6.2. In terms of comfort, the proposed method clearly outperforms in terms its yielding capabilities (*mt1*) even though all baseline methods have the exact same FSM policy for

Table 6.2: Overall Performance of Proposed Controller and Baselines over 1000 episodes

Metrics	***B1***	***B2***	***B3***	**Proposed**
mt1(*sc1*)	(817/94)	(934/24)	(912/33)	**(937/23)**
mt1 (*sc2*)	(7482/1811)	(8980/1821)	(8929/1496)	**(9911/1294)**
mt1 (*sc3*)	(8012/1754)	(9645/1875)	(9449/1535)	**(10310/1404)**
mt2(*sc1*)	(-2.90, 1.55)	(-1.12, 0.99)	(-1.42, 1.23)	**(-1.07, 0.95)**
mt2 (*sc2*)	(-2.13, 1.78)	(-1.31, 1.30)	(-1.05, 1.29)	**(-0.79, 0.90)**
mt2 (*sc3*)	(-1.99, 1.75)	(-1.29, 1.28)	(-1.02, 1.27)	**(-0.80, 0.91)**
mt3(*sc1*)	898	992	966	**996**
mt3 (*sc2*)	652	916	856	**986**
mt3 (*sc3*)	664	956	870	**988**
mt4(*sc1*)	(0.27, 0.35)	(0.08, 0.18)	(0.09, 0.22)	**(0.07, 0.16)**
mt4 (*sc2*)	(0.63, 0.57)	(0.32, 0.38)	(0.30, 0.38)	**(0.11, 0.18)**
mt4 (*sc3*)	(0.63, 0.60)	(0.30, 0.36)	(0.30, 0.39)	**(0.12, 0.19)**

yielding. Specifically, the proposed method outperforms the baselines in *sc2*, by 32.46%, 10.37%, 11.00% respectively. This clearly shows that, the proposed method is pedestrian friendly, in that without resorting to emergency braking, it yields to as many pedestrians as possible. In terms of safety, outperformance of the proposed method is clearly visible for the number of successful finishes (*mt3*) out of 1000 episodes per scenario where the proposed method outperforms the baselines in *sc2* by 51.23%, 7.64%, 15.19% respectively (performance in *sc2* is demonstrated in Fig. 6.5). This metric is crucial as it is directly related to collision risk of the method. Furthermore, the proposed method also outperforms all other baselines both in the *average deceleration* (*mt2*) and the *average time of emergency braking* (*mt4*) which could be interpreted as an indication of how successfully a method anticipates the incoming risk such that it resorts to minimal deceleration value and emergency braking (performance in *sc2* is demonstrated in Fig. 6.6). One important remark is that in Fig. 6.6 total number of yields per controller is different due to the fact that this metric is only available for successful path completions. In other words, an aggressive driving style may end up in a collision which in return means that the successful yields in the episode will not be considered.

One limitation is due to the simplification of the risk assessment with another function. The weights of the function that assesses the risk are determined heuristically, and the optimality of resulting controller have not proven. Therefore, there may exist a better weight vector, or a better representation of the risk.

Another limitation the simplification such as assessing the risk on a straight road, assuming that the vehicle is a point-mass object might limit the performance of the proposed method. Although, we have considered several important and realistic phenomena, the delay in actuating the control actions and the delay in sensing objects, the aforementioned assumptions might still deviate the implementation results from the simulated ones.

6.5 Conclusion

This chapter proposed a probabilistic risk assessment and collision avoidance method for emerging pedestrians from occlusions and demonstrates a possible proof-of-concept for the proposal. The proposal was compared against several baselines. The method was

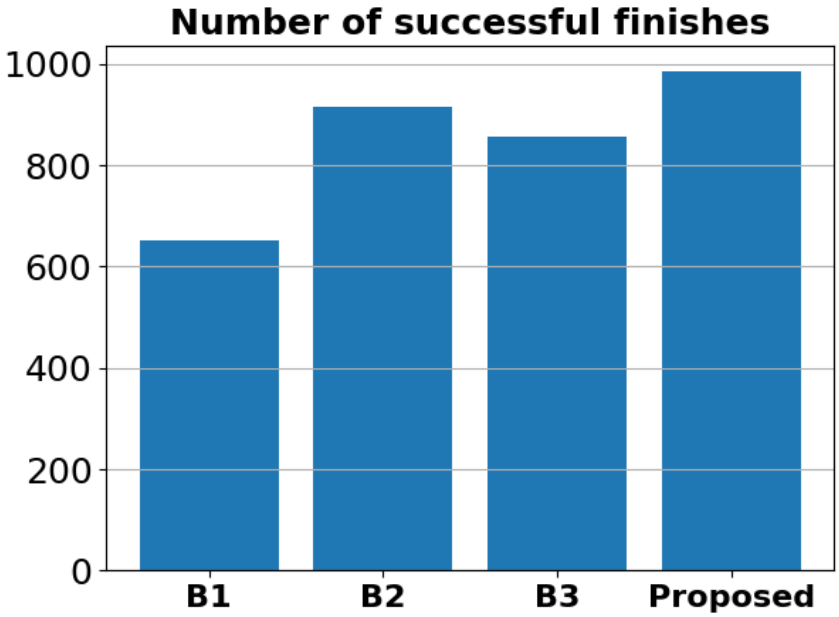

Figure 6.5: The total number of successful finishes (*mt3*) of different controllers in *mildly crowded urban scenarios* (*sc2*)

evaluated against the baselines in three different scenarios, 1000 episodes with randomized initial conditions per scenario type, in the simulation environment in Python built from scratch. The method outperformed these baselines in the predefined metrics. Since the proposal does not rely on accurate map data or accurate and precise localization to achieve occlusion-aware vehicle control, it could be used in which the localization sensor fidelity is low, urban areas, and big metropolitans. On the other hand, there are several limitations that should be overcome before moving to a real-life implementation of this proposal.

Future work can focus on the other possibilities to assess the probability using other contextual information such as the age of the visible pedestrians, the possible actions engaged by the visible pedestrians (presence of children playing soccer at the sidewalk, presence of distracted pedestrians due to use of cellphones or a conversation companion).

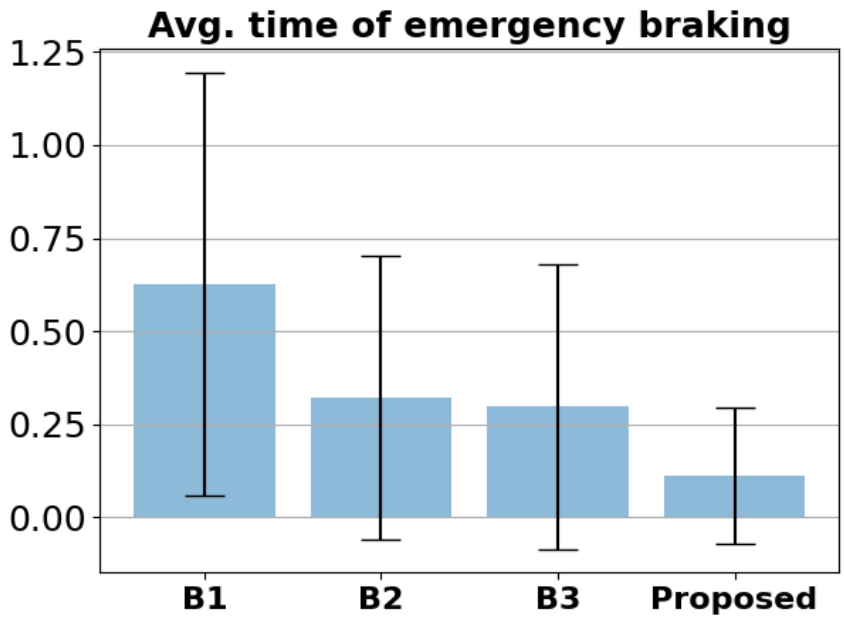

Figure 6.6: Average time of emergency braking (*mt4*) of different controllers in *mildly crowded urban scenarios* (*sc2*)

Algorithm 3 The proposed algorithm (Part 1)

```
1: function PROPOSEDCONTROLLER
2:     a_max ← μ_road * g
3:     s_{t+1} ← s_normal drive
4:     if A visible pedestrian is to be inside the path then
5:         if TTC < TTC_stop then
6:             s_{t+1} ← s_emergency
7:         else
8:             s_{t+1} ← s_yielding
9:     else
10:        current_state ← danger
11:        max_risk ← 0, current_risk ← 0
12:        repeat
13:            # Update current risk zone
14:            if d > d_stop_min then
15:                current_state ← discomfort
16:                max_risk ← 0
17:            # Check the neighboring visible cues
18:            current_risk ← from (6.1)
19:            if max_risk < current_risk then
20:                max_risk ← current_risk
21:            if current_state = danger then
22:                Set l_cautious, l_steady, a_limit and j_limit
23:                if max_risk > l_steady then
24:                    s_{t+1} ← s_steady drive
25:                else if max_risk > l_cautious then
26:                    s_{t+1} ← s_cautious drive
27:            else if current_state = discomfort then
28:                Set l_discomfort, l_steady, a_limit and j_limit
29:                if max_risk > l_steady then
30:                    s_{t+1} ← s_steady drive
31:                else if max_risk > l_cautious then
32:                    s_{t+1} ← s_cautious drive
33:            d ← d + Δd
34:        until d ≥ d_stop_comfort
35:    return s_{t+1}
   =0
```

Chapter 7

Stability Regulation of Learning-Based Continuous Control

This chapter is derived from the published work in [141].

7.1 Background

Automating the task of real-world driving requires the development of robust control systems with safety and stability guarantees. Rule-based methods have been the convention in the development of automated driving systems (ADS) and have shown promises in several driving tasks [142, 143, 144]. These methods generally involve the development of hand-crafted rules to process data and determine the best control output that matches a criterion or minimizes a loss function. Their wide-spread adoption can be attributed to the ease of formulation, safety guarantees, and interpretability. However, these methods often have difficulty with scaling to more complex driving scenarios or multiple objectives.

Reinforcement learning (RL)-based control has been proposed as an alternative paradigm to address sequential tasks and to deal with more complex driving scenarios. These methods are largely attributed to the use of deep neural networks as universal function approximators [145]. Their application has been successful in several control tasks such as lane keeping [146], lane changing [147], highway driving [148, 149], and even urban driving [150]. Rather than hand-crafting rules, these methods learn an optimal control policy using trial-and-error to maximize a reward function. However, the use of these methods has been largely limited to simulation environments with little real-world application. One reason is that the learned policy cannot guarantee safety or stability, which is an important requirement for safety-critical applications, especially with the presence of VRUs. Recent work has looked at improving the safety of RL agents for automated driving by adding constraints on the exploration process or through modifying the optimization process. In [151], a policy is learned to predict and mask unsafe actions. [148] proposes an explicit rule-based safety controller to correct unsafe RL actions. In [152], a risk function is learned, and a trust region constraint is added to guarantee the safety of the driving policy. Despite progress of these approaches, limiting the exploration process can lead to conservative policies without properly exploring the state-action space.

When considering shared spaces between pedestrians, cyclists, and drivers; safety of an

autonomous system becomes critical. The extension of learning-based control methods to these complex environments would require fundamental improvements to the design of deep neural networks. One example would be the consideration of controller stability, which is a common description of safety for control systems. Stability is important as it provides guarantees on the behavior of the system, especially when subject to disturbances. This in turn provides enhanced safety to the driver, passenegers, and the surrounding VRUs.

Lyapunov-based neural control has been an emerging approach to train neural network policies with Lyapunov stability guarantees. Methods such as [153] and [154] define a loss function that measures the violation of the Lyapunov conditions and train a neural network on minimizing this loss. [155] considers an approach to inherently constrain the closed-loop dynamics to be stable and presents a framework to jointly learn the closed-loop dynamics and its Lyapunov function. [156] synthesizes a Lyapunov stable neural network controller by formulating the Lyapunov conditions as a mixed integer problem. In [157], a Lagrangian relaxation of the Lyapunov conditions is proposed to guarantee the safety of an autonomous driving agent trained using offline RL. These methods show promise towards ensuring the stability and safety of the learned controller; however, their application has been mostly within the supervised learning setting due to the challenge of verifying the stability conditions over the entire state space.

In this chapter, we consider a control-theoretic approach to the training of an RL agent. We extend the Deep Deterministic Policy Gradient (DDPG) [158] algorithm with Lyapunov-based stability regulation. We learn the stable system closed-loop dynamics and its associated control-Lyapunov function (cLf) for a given fixed iteration of the DDPG controller policy. The cLf is then used to provide a Lyapunov stability regulation term to the objective of the DDPG agent policy. This allows for the DDPG agent to produce desirable system behaviors by optimizing the reward function, while optimizing the Lyapunov violations to learn actions that lead to stable dynamics. We focus on stability regulation of the RL control policy, as guaranteeing the stability performance over a continuous state space remains challenging. For a discrete and bounded state space, the stability criterion can be guaranteed using learner-verifier frameworks as in [153].

The rest of this chapter is divided as follows: The rest of this section provides the preliminaries of RL and Lyapunov stability. The proposed two-step LSR-DDPG framework is presented in Section 7.2. Section 7.3 outlines the simulation setup and experimental results within the simulated driving setting. Section 7.4 concludes the chapter.

7.1.1 Reinforcement Learning

The sequential decision-making problem of driving is modeled as a Markov Decision Process (MDP). The MDP is defined as a tuple $\langle X, U, R, T \rangle$ where X is the state space, U is the action space, R is the reward function, and T is the transition function. At each step, the RL agent at state $x \in X$ selects a control action $u \in U$. The new state, x', is then determined through the action and transition function, $T(x, u)$. The agent receives from the environment a reward, $r(x, u, x') \in R$, where the reward function is hand-crafted to reward certain behaviors and discourage unsafe actions. The transition function, T, is generally unknown and the goal of the RL agent is to find the optimal control policy $\pi : X \to U$ that maximizes the expected discounted future reward, $V_\pi(x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, u_t) | x_0\right]$, where $\gamma \in [0, 1]$ is the discount factor and $u_t = \pi(x_t)$.

7.1.2 Lyapunov Stability

Consider an n-dimensional nonlinear controlled dynamical system,

$$\dot{x}(t) = f(x_t, u_t) = f_{u_t}(x_t), \tag{7.1}$$

where $x_t \in \mathbb{R}^n$ is the state vector and $u(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ is the control vector. For clarity, we denote the time argument as a subscript, $x(t) = x_t$. Suppose f has an equilibrium point, x_{eq}, with u_{eq} such that $f(x_{eq}, u_{eq}) = 0$. The goal is to find some control policy $u \sim \pi(x_t)$ that stabilizes the dynamical system at the equilibrium point. To do so, we view this problem within the perspective of control-Lyapunov functions. See [159] for a more in-depth treatment of the definitions provided below.

Definition 1 *(Asymptotic Stability) A system is stable at the origin if* $\forall \epsilon \in \mathbb{R}^+$, $\exists \delta(\epsilon) \in \mathbb{R}^+$ *such that if* $||x(0)|| \leq \delta$, *then* $||x(t)|| < \epsilon$, $\forall t \geq 0$. *Furthermore, if* $\lim_{t\to\infty} ||x(t)|| = 0$, *then the system is asymptotically stable at the origin.*

Asymptotic stability provides a guarantee that the system goes to the origin in infinite time. It is important to note that it does not imply how long this convergence would take, which could be important for performance guarantees on the underlying controller.

If we consider a path tracking problem, then the notion of asymptotic stability states that the dynamical system starting with some initial condition, $x(0)$, within a sphere of radius δ will converge after some finite time t to the equilibrium or desired path.

Definition 2 *(Control-Lyapunov Function) Consider the controlled system in* (7.1) *with equilibrium at the origin,* $x_{eq} = 0$. *Suppose there exists a real-valued continuously differentiable function* $V : \mathbb{R}^n \to \mathbb{R}$ *such that the following conditions are met,*

1. $V(0) = 0$,
2. $V(x) > 0 \quad \forall x \in \mathbb{R}^n \setminus \{0\}$,
3. $\exists u : \dot{V}(x, u) = \nabla V(x) \cdot f(x, u) < 0 \quad \forall x \in \mathbb{R}^n \setminus \{0\}$

then the system is asymptotically stabilizable, V is its control-Lyapunov function, and u is the stabilizing action.

The Lyapunov function can be viewed as the "energy" of the system and the third condition on its derivative states that the value of the function along the trajectory of the system f is decreasing. This can be used to construct stabilizing control laws. We refer to the three conditions as the *Lyapunov conditions*.

Definition 3 *(Exponential Stability) Suppose the system in* (7.1) *is asymptotically stabilizable. If the following condition is satisfied* $\forall x \in \mathbb{R}^n \setminus \{0\}$,

$$\nabla V(x) \cdot f(x, u) \leq \text{-}\alpha V(x), \tag{7.2}$$

for some $\alpha > 0$, *then the system is exponentially stable.*

Exponential stability provides a stronger notion of convergence on the underlying dynamics but there is no straight-forward method to find such a function and its stabilizing controller. The use of neural networks to approximate cLf has become a promising direction to show some of these stability properties on a system.

7.2 Lyapunov Stability-Regulated DDPG Framework

We propose a two-step framework to learn a control policy with two objectives: (i) learn stabilizing control actions and, (ii) learn desirable system (or driving) behaviors. We handle the first objective by learning stable system dynamics and its associated control-Lyapunov function using deep neural networks as universal function approximators. The learned dynamics and cLf then provide feedback, through a loss term, to the training of the RL agent. The objective of the RL agent is to learn a desired driving behavior and stabilizing actions. The overall two-step framework is presented in Fig. 7.1. We now describe the details of this framework.

7.2.1 Joint Learning of Dynamics and Control-Lyapunov Function

We consider jointly learning the system dynamics and its corresponding control-Lyapunov function. Let $\hat{f}_\theta$ be some initialization of the system dynamics model and V_ψ be the control-Lyapunov function with network parameters θ and ψ respectively. This initialization can be random. From [155], it follows that the stable controlled system dynamics in (7.1) can be represented as:

$$f(x,u) = \hat{f}_\theta(x,u) - \eta\left(x,u|\hat{f}_\theta, V_\psi\right) \tag{7.3}$$

and

$$\eta\left(x,u|\hat{f}_\theta, V_\psi\right) = \nabla V_\psi(x)\frac{\sigma\left(\nabla V_\psi(x)^T\hat{f}_\theta(x,u) + \alpha V_\psi(x)\right)}{||\nabla V_\psi(x)||_2^2 + a}, \tag{7.4}$$

where $a \ll 1$ is a small positive constant to avoid division by zero at the equilibrium and $\sigma(\cdot)$ is a smooth ReLU function defined as

$$\sigma(x) = \begin{cases} 0 & \text{if} \quad x \leq 0, \\ x^2/2d & \text{if} \quad 0 < x < d, \\ x\text{-}d/2 & \text{otherwise}, \end{cases}$$

and $d > 0$. This definition guarantees the stability of the closed-loop dynamics by projecting $\hat{f}_\theta$ so the condition $\nabla V_\psi(x)^T\hat{f}_\theta(x,u) + \alpha V_\psi(x) \leq 0$ is satisfied. For example, if the condition is satisfied, then $\eta\left(x,u|\hat{f}_\theta, V_\psi\right) = 0$ and $f(x,u) = \hat{f}_\theta(x,u)$.

We initialize the nominal dynamics to be a random feed-forward network and the control-Lyapunov function as an Input Convex Neural Network (ICNN) [160]. The structure of the ICNN ensures that the function is convex in x provided that all weight matrices are non-negative, and all activation functions are convex and non-decreasing. To ensure positive definiteness, the final layer of the neural network is modified as follows:

$$V(x) = \sigma\left(g(x) - g(0)\right) + \epsilon||x||_2^2,$$

where $g(\cdot)$ is the output of the ICNN, $\sigma(\cdot)$ is the smooth ReLU activation function, and $\epsilon > 0$. The choice of $\sigma(\cdot)$ is positive, convex, and non-decreasing with $\sigma(0) = 0$. This enforces the strict positive definiteness of the control-Lyapunov function with the regularization term $\epsilon||x||_2^2$. Furthermore, shifting the output of the ICNN by $g(0)$ ensures that the global minima occurs at the origin (or, without a loss of generality, the desired equilibrium point). This choice of neural network architecture inherently satisfies the first two Lyapunov conditions: (i) V_ψ has only one global minimum at $V(0) = 0$ and (ii) $V_\psi(x) > 0$ elsewhere.

7.2.2 Lyapunov Stability Regulated DDPG Control

Now, we present the method of finding an appropriate RL policy regulated on stabilizing the system dynamics such that the condition in (7.2) is satisfied. Consider the control law,

$$u^*(t) = \underset{u \in U}{\operatorname{argmin}} \dot{V}(x, u), \tag{7.5}$$

such a control policy selects the action that maximizes the decrease in the system energy. While this action may be optimal, it can lead to undesirable system trajectories. For example, in the case of automated highway driving, this may result in actions such as jerky or unsafe driving since the control law is only concerned with reaching the equilibrium state. Against this backdrop, we formulate the problem as finding a control law that balances the choice of selecting a stabilizing action and producing a desired system trajectory. Such a control law may be suboptimal; however, it would provide necessary behavior enhancements for the produced state trajectories.

We choose the DDPG [158] algorithm to find an appropriate stabilizing controller. This is a model-free off-policy RL algorithm with an actor-critic architecture for continuous action spaces. The aim of the critic network, $Q(x, u|\theta^Q)$, is to evaluate the Q-value (or action-value) function based on the actor's output and the current state. The aim of the actor network, $\pi(x|\theta^\pi)$, with parameters θ^π is to learn a deterministic policy that maximizes the expected Q-value,

$$\max_{\theta^\pi} \mathbb{E}\left[Q(x, \pi(x|\theta^\pi))\right], \tag{7.6}$$

The Q-value affects the actor's weights using policy gradient descent. To regulate the actor policy on stabilizing the system dynamics, we define an additional loss term, L_c, coined the *Lyapunov violation*. The structure of the cLf neural network only requires that the selected action satisfies condition 3 of the Lyapunov conditions. Therefore, we define the Lyapunov violation as,

$$L_c(x, u) = \nabla V(x) \cdot f(x, u) + \alpha V(x). \tag{7.7}$$

The objective function of the stability-regulated actor policy is then defined as,

$$J(\theta) = \mathbb{E}\left[\rho Q(x, u) - (1 - \rho) L_c(x, u)|_{x=x_t, u=\pi(x_t)}\right], \tag{7.8}$$

where $Q(x, u)$ is the action-value function (critic network) and $\rho \in [0, 1]$ is a weighing term. The objective of the actor policy now becomes to maximize the action-value function and minimize the Lyapunov violation, $L_c(x, u)$. In this case, the action-value function, which depends on the reward function, controls the system behavior while the Lyapunov violation, which depends on the current iteration of the cLf, evaluates stabilizing actions. Since the controller is trained off-policy by sampling batches of experience, the policy loss gradient is defined as follows:

$$\nabla_{\theta^\pi} J(\theta) = -\frac{1}{N} \sum_{(x,u) \in B} \left[\rho \nabla_u Q(x, \pi(x)|\theta^Q) \nabla_{\theta^\pi} \pi(x|\theta^\pi) - (1 - \rho) \nabla_u L_c(x, u)\right], \tag{7.9}$$

where N is the number of samples in each batch and B is the batch of experiences. The defined policy loss in (7.9) may be restrictive as the function is minimized for more positive $L_c(x, u)$. A positive $L_c(x, u)$ represents a stabilizing action. Instead, we modify

the Lyapunov violation term to be the same for all stabilizing actions as any stabilizing action should be admissible and allow the selection between the stabilizing actions to be dictated by the action-value function. The empirical policy loss gradient then becomes,

$$\nabla_{\theta^\pi} J(\theta) = -\frac{1}{N} \sum_{(x,u)\in B} \left[\rho \nabla_u Q(x, \pi(x)|\theta^Q) \nabla_{\theta^\pi} \pi(x|\theta^\pi) - (1-\rho)\hat{L_c}(x,u)\right], \tag{7.10}$$

where $\hat{L_c}(x,u) = \max\left(L_c(x,u), 0\right)$. The critic network is updated by minimizing the mean-squared error loss:

$$L = \frac{1}{N} \sum_{e_i \in B} \left(y_i - Q(x_i, u_i|\theta^Q)\right)^2, \tag{7.11}$$

where the experience $e_i = (x_i, u_i, r_i, x_{i+1})$ and,

$$y_i = r_i + \gamma Q\left(x_{i+1}, \pi(x_{i+1})|\theta^Q\right)$$

is the updated Q-value obtained by the Bellman equation. To allow for exploration of the state space, an Ornstein-Uhlenbeck [161] process noise $\mathcal{N}$ is added to the policy action, $\pi(x_t) = \pi(x_t|\theta^\pi) + n_t$ where $n_t \sim \mathcal{N}_t$. The exploration is stochastic, but the learned control policy is deterministic.

7.2.3 Framework Overview and Remarks

The proposed two-step framework given in Fig. 7.1 iterates between the RL controller and learning stable system dynamics and the associated cLf. The agent first performs simulation rollouts to obtain system trajectories that is used to update the actor and critic networks. The actor network depends on the current fixed iteration of the cLf and dynamics to determine the Lyapunov violation term in the loss function (see (7.8)). After a certain number of updates to the RL controller, the actor policy is fixed and used to collect trajectories of the system dynamics. The cLf and closed-loop dynamics are jointly updated to reflect the new (and fixed) actor policy. The RL controller is then further trained using the updated cLf and system dynamics.

Exploring actions and dynamics The objective of the control policy given in (7.8) depends on the current iteration of the closed-loop dynamics and control-Lyapunov function in (7.3) through the Lyapunov violation term. To better reflect the dynamics of the system and control-Lyapunov function around the fixed deterministic policy, we add a decaying noise parameter ϵ_t to the system trajectory rollouts used to update the closed-loop dynamics. The actions in (7.3) with the exploration term becomes

$$u_t = u_t^* + \epsilon_t, \tag{7.12}$$

where u_t^* is sampled from the fixed policy $\pi(x_t)$. We set ϵ_t to decay linearly to 0 after some number of updates so that the closed-loop dynamics and control-Lyapunov function reflect the converged control policy. In this case, all simulation rollouts reflect the actor's policy. Note that this is different from the Ornstein-Uhlenbeck noise used during learning of the control policy.

Training the control-Lyapunov function Empirically, we found that training the dynamics given in (7.3) using mean-squared error loss resulted in violations of the Lyapunov conditions even after the training has converged. To address this issue, we impose

the L1-loss of the projection term in (7.4). The loss function for the dynamics in (7.3) then becomes:

$$L_f = \frac{1}{|D|} \sum_i \left[(y_i - \hat{y}_i)^2 + |\eta_i| \right], \tag{7.13}$$

where η_i is the L1-loss of the projection term in (7.4) for sample index i, $y_i = x_{t+1}$ is the true label collected during rollout, $\hat{y}_i = f(x_t, u_t)$ is the estimated dynamics from (7.3), and $D = \{(x_t, u_t, x_{t+1})\}_i$ is the sampled system trajectories. When $L_f = 0$ then the stable true dynamics are estimated for the samples used. The L1-loss drives the projection term to zero when the loss function is minimized.

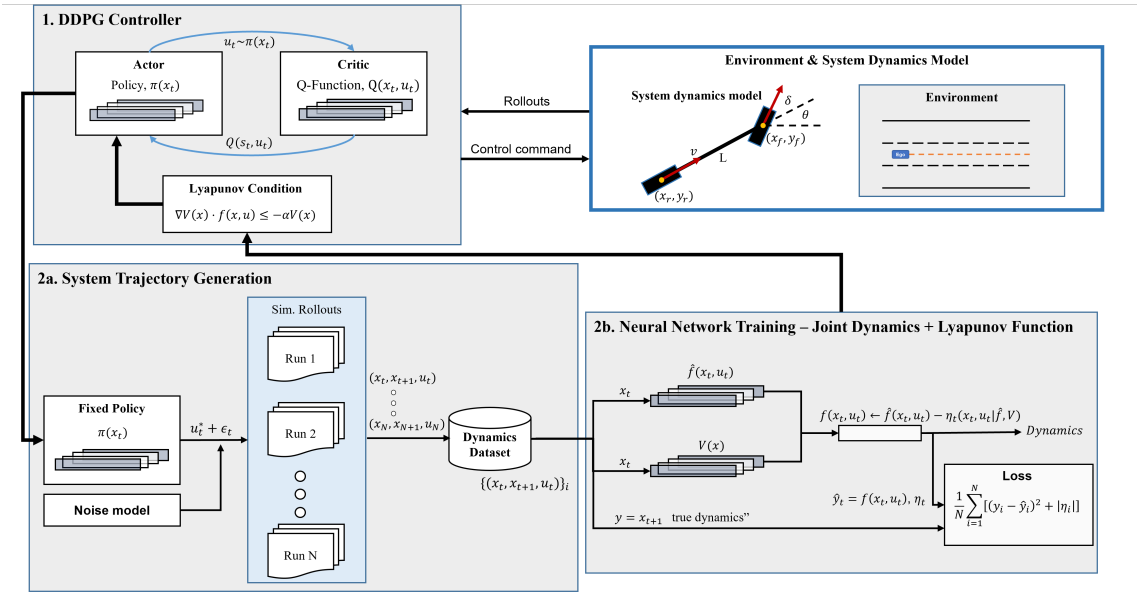

Figure 7.1: Proposed two-step Lyapunov Stability-Regulated DDPG framework. The DDPG controller is regulated by a Lyapunov violation term that is calculated from the current iteration of the control-Lyapunov function. Every M episodes, the dynamics and the control-Lyapunov functions are jointly updated using sampled data points from the fixed actor policy. A noise model is introduced in the sampled actions to allow for better representation of nearby states, which facilitates the exploration of the DDPG controller. This loop is repeated until convergence of the control policy and the Lyapunov violations.

7.3 Simulation Experiments

In this section, we consider the implementation of the LSR-DDPG framework presented in Section 7.2 for the lane-following driving scenario. All experiments are performed using the highway-env [162] simulator.

7.3.1 Simulation Setup

Consider a three-lane road environment with no traffic vehicles. The state space of the vehicle is bounded between the boundaries of the leftmost lane and the rightmost lane. At each initialization, the vehicle is randomly placed within the bounds of the three-lane road. Our goal is to find a stabilizing lateral controller $u(x_t)$, the stable closed-loop vehicle

dynamics $f(x_t, u_t)$, and its associated control-Lyapunov function $V(x)$. The desired path is used as the equilibrium state. The vehicle dynamics are defined by the kinematic bicycle model [163]:

$$\begin{aligned} \dot{y} &= v \sin(\psi + \beta), \\ \dot{\psi} &= \frac{v}{L} \sin \beta, \\ \beta &= \tan^{-1}\left(\frac{\tan \delta}{2}\right), \end{aligned} \tag{7.14}$$

where y is the lateral position, v is the longitudinal velocity, L is the vehicle length, ψ is the vehicle heading, δ is the steering command, and β is the slip angle at the center of gravity. We assume that the vehicle speed is fixed to $25m/s$ across all iterations. The action space is bounded between $\left[-\frac{\pi}{12}, \frac{\pi}{12}\right]$, which is normalized to be in the range $[-1, 1]$. We define the normalized state vector at each iteration as

$$s = [y \quad sin\psi] \in [-1, 1]^2. \tag{7.15}$$

7.3.2 Reward Function and Termination

The reward function is formulated as a positive function of the lateral position and heading with respect to the equilibrium trajectory,

$$\begin{aligned} r_y &= \max(1 - |y - y_{eq}|, 0), \\ r_\psi &= 1 - |\sin(\psi - \psi_{eq})|, \end{aligned}$$

and $r = r_y + r_\psi$. To avoid learning in unsafe states, we terminate the simulation, with zero reward, if the vehicle moves outside the boundaries of the road or if the deviation from the reference trajectory exceeds $\pi/2$.

7.3.3 Network Parameters

The nominal system dynamics is defined as a feedforward network with two hidden layers of size 128 and ReLU activation. The control-Lyapunov function is defined as an ICNN with two hidden layers of size 128 and smooth ReLU activation. The actor network is defined as a feedforward network with two hidden layers of size 128, ReLU activation between the hidden layers, and *tanh* activation at the output. This bounds the output action between $[-1, 1]$. The critic network is defined as a feedforward network with two hidden layers of size 128 and ReLU activation between all layers except the output. The learning rates of the dynamics, actor, and critic networks are set to $1e^{-4}$, $1e^{-3}$, and $1e^{-3}$ respectively. The parameters of all networks are updated using Adam [164] optimizer. The discount factor is set to 0.99, $\alpha = 0.01$, $\rho = 0.5$, and the dynamics noise $\epsilon \sim U[-1, 1]$.

7.3.4 Results

We train the LSR-DDPG controller framework for 1 500 episodes. Without loss of generality, we assume an equilibrium trajectory $\mathcal{T} = [0, 0]$. The state space can be shifted such that the equilibrium trajectory is at the origin. The dynamics are updated every 50 episodes for 300 epochs using 2 000 samples of system trajectories with a $3 - 1$ train-test split. The dynamics noise is linearly annealed and completely removed after 1 000

episodes. To demonstrate our approach, we also train a baseline DDPG controller as in [158] with no stability regulation in the loss function. Results are collected over 5 random seeds shared across both controllers.

Fig. 7.2 shows the training curve of the controllers and the convergence of the trained LSR-DDPG policy and Lyapunov violations. The total Lyapunov violations are calculated from sampled test data at each update interval of the dynamics. It is noted that the total Lyapunov violations decay as the dynamics noise parameter decays. This is a result of the convergence of the control policy and the learned stable dynamics as the exploration of nearby states is removed. The proposed loss function is also minimized along stable trajectories. Comparing the two training curves, it is also noted that the LSR-DDPG controller can generate stabilizing actions whilst still conforming to action trajectories with high rewards, or desired behaviors. Random sample trajectories are then collected for the

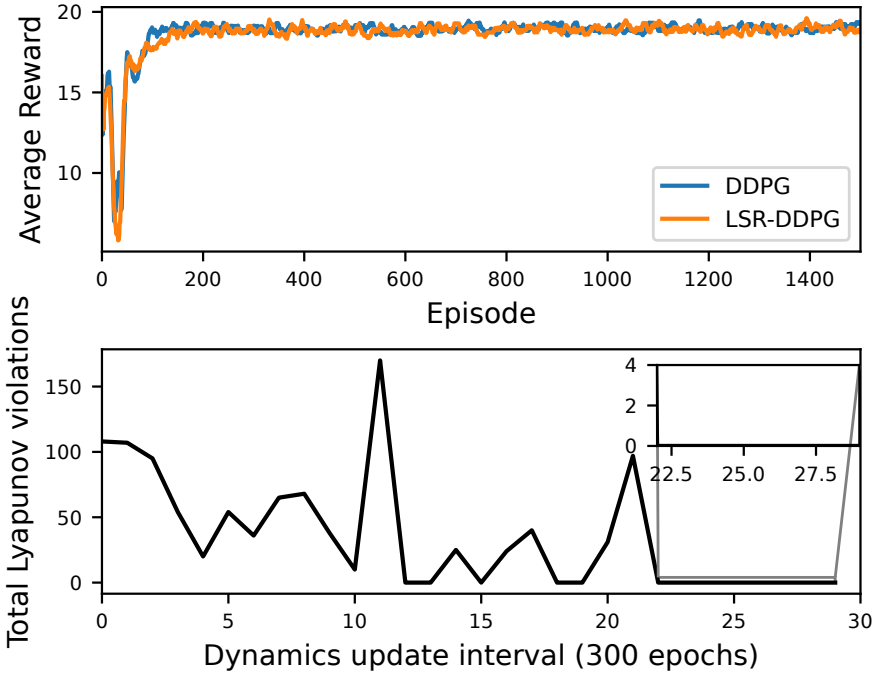

Figure 7.2: Top shows the LSR-DDPG and DDPG training curves. Bottom shows the total Lyapunov violations for the LSR-DDPG fixed actor policy at each update interval over 5 random seeds and its convergence.

final, fixed iteration of the learned control policy, dynamics, and control-Lyapunov function. Fig. 7.3 shows that the learned dynamics is a good predictor of the real dynamics. Moreover, for all samples collected, no Lyapunov violations were recorded.

We then consider a double lane change maneuver on the trained controllers by adjusting the reference trajectory. Fig. 7.4 shows the generated trajectories and the control action at each step. We note that the LSR-DDPG performs more conservative maneuvering, due to the stability regulation, and smoother lane changes while still producing a desired system behavior.

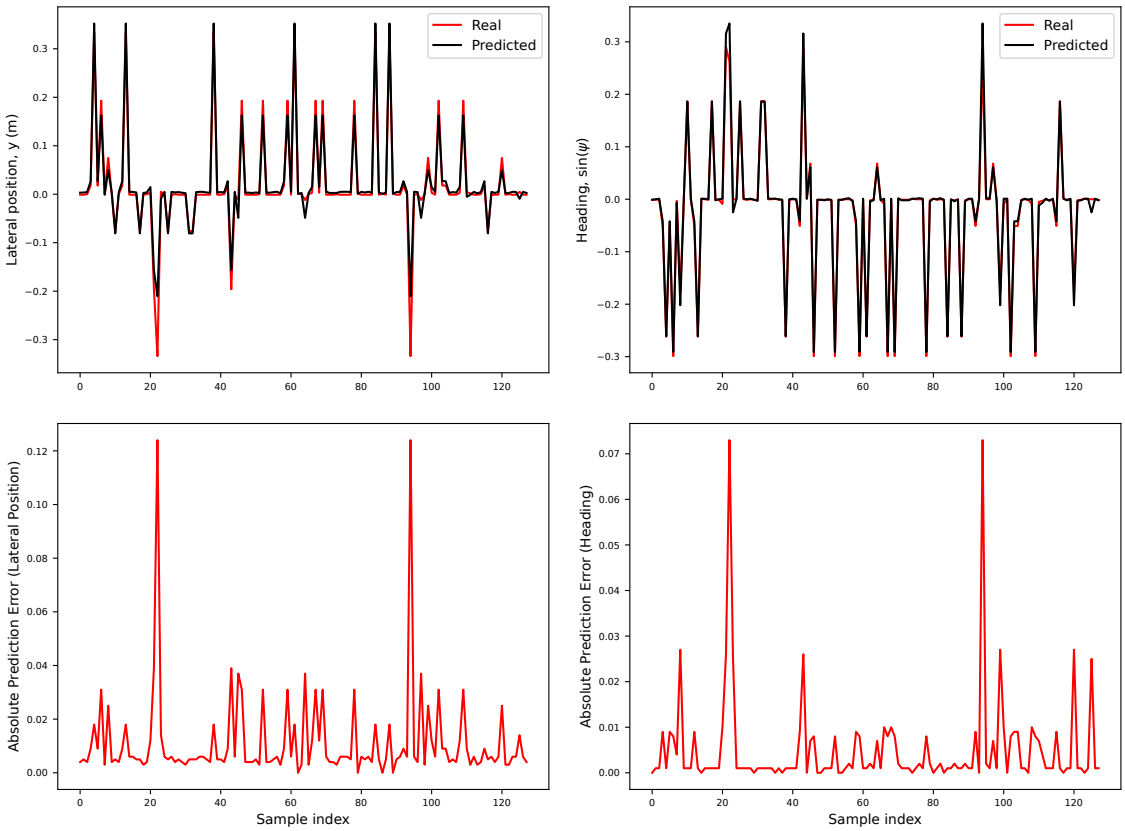

Figure 7.3: Prediction performance of the learned stable dynamics. Samples are randomly collected.

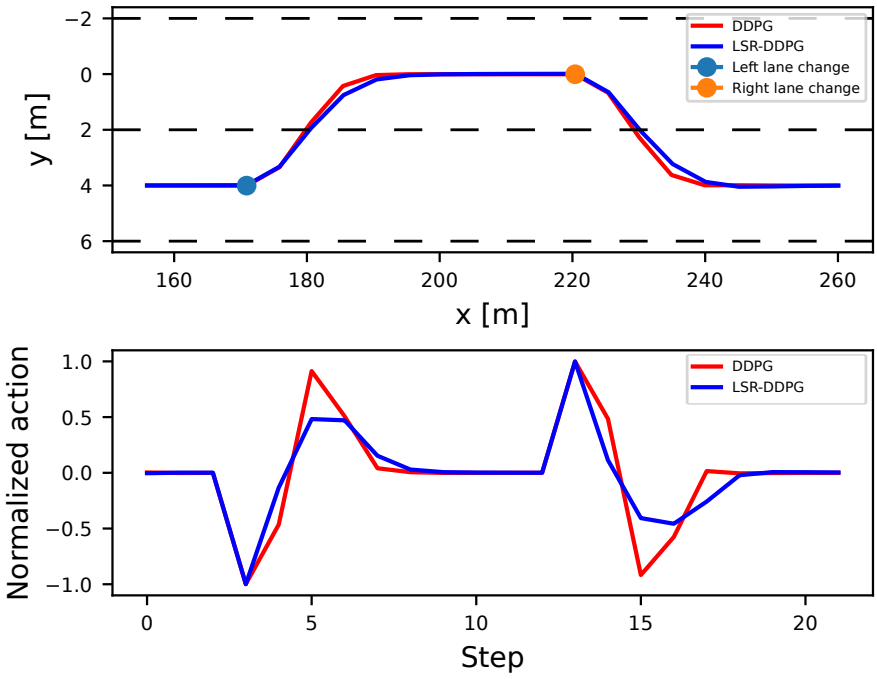

Figure 7.4: Comparison of a double lane change maneuver on LSR-DDPG (blue) and DDPG baseline (red) controllers. Third lane is not shown.

7.4 Conclusion

This chapter presented a framework for regulating an RL controller on the Lyapunov stability criterion to produce a desirable system behavior and to learn stabilizing actions. We proposed a two-step learning scheme to update the control policy and to update the closed-loop stable dynamics with its control-Lyapunov function. The framework was evaluated in a three-lane driving environment and compared with a DDPG baseline. We demonstrated that the proposed framework could learn an appropriate system behavior with stabilizing actions. This in turn improves the overall driving safety of the controller, and consequently, the safety of surrounding VRUs.

Chapter 8

Conclusion

8.1 Conclusion

A variant social force-based model is presented and shown that it is useful to design, control, and evaluate the performance of AVs in Vehicle Pedestrian Interaction (VPI) scenarios when a pedestrian is using uncontrolled crossings. We further extended the pedestrian motion model to the formulation of general motion models and used GFSM for the generation of realistic trajectories. Then we moved towards designing control of vehicles using model predictive control for longitudinal control in vehicle-crowd interaction. Next, we presented a method to predict pedestrian intention prediction. The method showed better results than the current state-of-the-art. Next, we presented a risk assessment of occluded pedestrian and collision avoidance control for AVs. Next, we presented a Framework to improve the stability of the RL controller using Lyapunov stability criteria. The framework was evaluated in a three-lane driving environment and compared with a DDPG baseline. We demonstrated that the proposed framework could learn an appropriate system behavior with stabilizing actions.

8.2 Future Work

Future work can be done to the VPI framework:

- Pedestrian model can be improved by using VPI datasets.
- Control design can consider uncertain pedestrian behavior.
- The solutions presented can be extended to include more scenarios.

For generalized motion models, the following future work can be considered:

- Improvement of motion models for vehicles, and adding new dynamic objects (robots, bicycles, etc.).
- Extending the framework for many stakeholders to design and control complex behaviors of vehicles and pedestrians and investigate the feasibility of scalability.

In model predictive control for vehicle-crowd interaction, the following future work can be considered:

- In the pedestrian motion prediction process, this constant vehicle speed assumption can be improved by incorporating the VCI model into the MPC synthesis. Because of the non-linearity of the VCI model, this incorporation requires modifying the VCI model so that the MPC can be properly synthesized and successfully solved.
- The performance of the PID approach can be improved by systematically tuning the PID parameters. Specifically, the steady-state error should be minimized or eliminated, and other effects such as rise time, overshot, settling time, and stability should also be carefully treated.
- In addition to hard constraints on the control action, a quadratic term of control effort could also be included in the MPC cost function, so that the overall MPC performance can be improved by taking the control action in consideration.

In predicting pedestrian intention, the following future work can be considered:

- Improving the robustness for unexpected situations.
- Fusion with other sensors to achieve better results.
- Using a range of pedestrian behaviors (children, old, disabled, etc) to further tune the model for better prediction.

For the risk assessment work, the following future work can be considered:

- Risk assessment can be improved by using other contextual information.
- Including a range of situations in risk assessment such as children playing nearby, distracted pedestrians, etc.

Appendix A

Research Products for this Project

A.1 Journal Publications

F. T. Johora, D. Yang, J. P. Muller, and U. Ozguner, "On the Generalizability of Motion Models for Road Users in Heterogeneous Shared Traffic Spaces," *IEEE Trans. Intell. Transport. Syst.*, vol. 23, no. 12, pp. 23084–23098, Dec. 2022, doi: 10.1109/TITS.2022.3192138.

D. Yang, H. Zhang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Predicting pedestrian crossing intention with feature fusion and spatio-temporal attention," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 2, pp. 221–230, 2022.

A.2 Conference Publications

D. Yang, K. Redmill, and Ü. Özgüner, "A Multi-State Social Force Based Framework for Vehicle-Pedestrian Interaction in Uncontrolled Pedestrian Crossing Scenarios," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, Las Vegas, NV, USA: IEEE, Oct. 2020, pp. 1807–1812. doi: 0.1109/IV47402.2020.9304561.

D. Yang and Ü. Özgüner, "Combining social force model with model predictive control for vehicle's longitudinal speed regulation in pedestrian-dense scenarios," in *The th Biennial Workshop on Digital Signal Processing for In-Vehicle Systems*, Nagoya University, Japan 2018.

M. Koç, E. Yurtsever, K. Redmill, and Ü. Özgüner, "Pedestrian emergence estima- tion and occlusion-aware risk assessment for urban autonomous driving," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 292–297, IEEE, 2021.

B. Hejase and U. Ozguner, "Lyapunov stability regulation of deep reinforcement learning control with application to automated driving," in *2023 American Control Conference (ACC)*, pp. 4437–4442, IEEE, 2023.

A.3 Dissertation and Thesis

Bilal Hejase, Interpretable and Safe Deep Reinforcement Learning Control in Automated Driving Applications, PhD Dissertation, The Ohio State University, 2023.

Dongfang Yang, Collective Pedestrian Motion Under Vehicle Influence: Social Force Based Modeling and Application in Intelligent Transportation, PhD Dissertation, The Ohio State University, 2020.

Bibliography

[1] U. N. H. T. S. A. (NHTSA), "Traffic safety facts (2017 data) - pedestrians." `https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812681` [Online; accessed 2019-09-30], 2019.

[2] D. Yang, K. Redmill, and Ü. Özgüner, "A multi-state social force based framework for vehicle-pedestrian interaction in uncontrolled pedestrian crossing scenarios," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1807–1812, IEEE, 2020.

[3] E. Becic, "Vehicle automation report - hwy18mh010," *NATIONAL TRANSPORTATION SAFETY BOARD*.

[4] D. Yang, Ü. Özgüner, and K. Redmill, "A social force based pedestrian motion model considering multi-pedestrian interaction with a vehicle," *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, vol. 6, no. 2, pp. 1–27, 2020.

[5] A. Rasouli and J. K. Tsotsos, "Autonomous vehicles that interact with pedestrians: A survey of theory and practice," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 3, pp. 900–918, 2019.

[6] R. Woodman, K. Lu, M. D. Higgins, S. Brewerton, P. A. Jennings, and S. Birrell, "Gap acceptance study of pedestrians crossing between platooning autonomous vehicles in a virtual environment," *Transportation research part F: traffic psychology and behaviour*, vol. 67, pp. 1–14, 2019.

[7] G. Yannis, E. Papadimitriou, and A. Theofilatos, "Pedestrian gap acceptance for mid-block street crossing," *Transportation planning and technology*, vol. 36, no. 5, pp. 450–462, 2013.

[8] B. Chen, D. Zhao, and H. Peng, "Evaluation of automated vehicles encountering pedestrians at unsignalized crossings," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pp. 1679–1685, IEEE, 2017.

[9] T. Bandyopadhyay, K. S. Won, E. Frazzoli, D. Hsu, W. S. Lee, and D. Rus, "Intention-aware motion planning," in *Algorithmic foundations of robotics X*, pp. 475–491, Springer, 2013.

[10] S. M. Thornton, F. E. Lewis, V. Zhang, M. J. Kochenderfer, and J. C. Gerdes, "Value sensitive design for autonomous vehicle motion planning," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1157–1162, IEEE, 2018.

[11] M. Schraner, M. Bouton, M. J. Kochenderfer, and D. Watzenig, "Pedestrian collision avoidance system for scenarios with occlusions," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1054–1060, IEEE, 2019.

[12] N. R. Kapania, V. Govindarajan, F. Borrelli, and J. C. Gerdes, "A hybrid control design for autonomous vehicles at uncontrolled crosswalks," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1604–1611, IEEE, 2019.

[13] P. Liu and Ü. Özgüner, "Predictive control of a vehicle convoy considering lane change behavior of the preceding vehicle," in *American Control Conference (ACC), 2015*, pp. 4374–4379, IEEE, 2015.

[14] M. Goldhammer, S. Köhler, S. Zernetsch, K. Doll, B. Sick, and K. Dietmayer, "Intentions of vulnerable road users–detection and forecasting by means of machine learning," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[15] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[16] W. Zeng, P. Chen, H. Nakamura, and M. Iryo-Asano, "Application of social force model to pedestrian behavior analysis at signalized crosswalk," *Transportation research part C: emerging technologies*, vol. 40, pp. 143–159, 2014.

[17] D. Yang, Ü. Özgüner, and K. Redmill, "Social force based microscopic modeling of vehicle-crowd interaction," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1537–1542, IEEE, 2018.

[18] S. Schmidt and B. Faerber, "Pedestrians at the kerb–recognising the action intentions of humans," *Transportation research part F: traffic psychology and behaviour*, vol. 12, no. 4, pp. 300–310, 2009.

[19] C. Feliciani, L. Crociani, A. Gorrini, G. Vizzari, S. Bandini, and K. Nishinari, "A simulation model for non-signalized pedestrian crosswalks based on evidence from on field observation," *Intelligenza Artificiale*, vol. 11, no. 2, pp. 117–138, 2017.

[20] S. Chandra and A. K. Bharti, "Speed distribution curves for pedestrians during walking and crossing," *Procedia-Social and Behavioral Sciences*, vol. 104, pp. 660–667, 2013.

[21] S. Diamond and S. Boyd, "Cvxpy: A python-embedded modeling language for convex optimization," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2909–2913, 2016.

[22] F. T. Johora, D. Yang, J. P. Müller, and Ü. Özgüner, "On the generalizability of motion models for road users in heterogeneous shared traffic spaces," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 23084–23098, 2022.

[23] E. Clarke, "Shared space-the alternative approach to calming traffic," *Traffic engineering & control*, vol. 47, no. 8, 2006.

[24] B. Hamilton-Baillie, "Shared space: Reconciling people, places and traffic," *Built environment*, vol. 34, no. 2, pp. 161–181, 2008.

[25] I. Kaparias, M. G. Bell, A. Miri, C. Chan, and B. Mount, "Analysing the perceptions of pedestrians and drivers to shared space," *Transportation research part F: traffic psychology and behaviour*, vol. 15, no. 3, pp. 297–310, 2012.

[26] A. Clayden, K. Mckoy, and A. Wild, "Improving residential liveability in the uk: Home zones and alternative approaches," *Journal of Urban Design*, vol. 11, no. 1, pp. 55–71, 2006.

[27] M. Jenks, "Residential roads researched: Are innovative estates safer?," *Architects Journal*, vol. 177, no. 26, 1983.

[28] R. Schönauer, "A microscopic traffic flow model for shared space," *Graz University of Technology*, 2017.

[29] B. Anvari, M. G. Bell, A. Sivakumar, and W. Y. Ochieng, "Modelling shared space users via rule-based social force model," *Transportation Research Part C: Emerging Technologies*, vol. 51, pp. 83–103, 2015.

[30] N. Rinke, C. Schiermeyer, F. Pascucci, V. Berkhahn, and B. Friedrich, "A multi-layer social force approach to model interactions in shared spaces using collision prediction," *Transportation research procedia*, vol. 25, pp. 1249–1267, 2017.

[31] L. W. Lan and C.-W. Chang, "Inhomogeneous cellular automata modeling for mixed traffic with cars and motorcycles," *Journal of advanced transportation*, vol. 39, no. 3, pp. 323–349, 2005.

[32] Y. Zhang and H. Duan, "Modeling mixed traffic flow at crosswalks in microsimulations using cellular automata," *Tsinghua science and technology*, vol. 12, no. 2, pp. 214–222, 2007.

[33] S. Bandini, L. Crociani, C. Feliciani, A. Gorrini, and G. Vizzari, "Collision avoidance dynamics among heterogeneous agents: The case of pedestrian/vehicle interactions," in *AI* IA 2017 Advances in Artificial Intelligence: XVIth International Conference of the Italian Association for Artificial Intelligence, Bari, Italy, November 14-17, 2017, Proceedings 16*, pp. 44–57, Springer, 2017.

[34] F. T. Johora and J. P. Müller, "Modeling interactions of multimodal road users in shared spaces," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3568–3574, IEEE, 2018.

[35] S. Ahmed, F. T. Johora, and J. P. Müller, "Investigating the role of pedestrian groups in shared spaces through simulation modeling," in *International Workshop on Simulation Science*, pp. 52–69, Springer, 2019.

[36] D. Gettman and L. Head, "Surrogate safety measures from traffic simulation models," *Transportation Research Record*, vol. 1840, no. 1, pp. 104–115, 2003.

[37] F. T. Johora and J. P. Müller, "Zone-specific interaction modeling of pedestrians and cars in shared spaces," *Transportation research procedia*, vol. 47, pp. 251–258, 2020.

[38] F. T. Johora and J. P. Müller, "On transferability and calibration of pedestrian and car motion models in shared spaces," *Transportation Letters*, 2020. Accepted.

[39] C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz, "Simulation of pedestrian dynamics using a two-dimensional cellular automaton," *Physica A: Statistical Mechanics and its Applications*, vol. 295, no. 3-4, pp. 507–525, 2001.

[40] S. Bandini, L. Crociani, and G. Vizzari, "An approach for managing heterogeneous speed profiles in cellular automata pedestrian models.," *Journal of Cellular Automata*, vol. 12, no. 5, 2017.

[41] K. Nagel and M. Schreckenberg, "A cellular automaton model for freeway traffic," *Journal de physique I*, vol. 2, no. 12, pp. 2221–2229, 1992.

[42] C. Chai and Y. D. Wong, "Fuzzy cellular automata model for signalized intersections," *Computer-Aided Civil and Infrastructure Engineering*, vol. 30, no. 12, pp. 951–964, 2015.

[43] J. Chen, Z. Li, W. Wang, and H. Jiang, "Evaluating bicycle–vehicle conflicts and delays on urban streets with bike lane and on-street parking," *Transportation letters*, vol. 10, no. 1, pp. 1–11, 2018.

[44] X. Chen, M. Treiber, V. Kanagaraj, and H. Li, "Social force models for pedestrian traffic–state of the art," *Transport reviews*, vol. 38, no. 5, pp. 625–653, 2018.

[45] M. Asano, T. Iryo, and M. Kuwahara, "Microscopic pedestrian simulation model combined with a tactical model for route choice behaviour," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 6, pp. 842–855, 2010.

[46] F. T. Johora, P. Kraus, and J. P. Müller, "Dynamic path planning and movement control in pedestrian simulation," *arXiv preprint arXiv:1709.08235*, 2017.

[47] W. Zeng, H. Nakamura, and P. Chen, "A modified social force model for pedestrian behavior simulation at signalized crosswalks," *Procedia-Social and Behavioral Sciences*, vol. 138, pp. 521–530, 2014.

[48] F. Pascucci, N. Rinke, C. Schiermeyer, V. Berkhahn, and B. Friedrich, "A discrete choice model for solving conflict situations between pedestrians and vehicles in shared space," *arXiv preprint arXiv:1709.09412*, 2017.

[49] H. Fujii, H. Uchida, and S. Yoshimura, "Agent-based simulation framework for mixed traffic of cars, pedestrians and trams," *Transportation research part C: emerging technologies*, vol. 85, pp. 234–248, 2017.

[50] H. Kita, "A merging–giveway interaction model of cars in a merging section: a game theoretic analysis," *Transportation Research Part A: Policy and Practice*, vol. 33, no. 3-4, pp. 305–312, 1999.

[51] T. Bjørnskau, "The zebra crossing game–using game theory to explain a discrepancy between road user behaviour and traffic rules," *Safety science*, vol. 92, pp. 298–301, 2017.

[52] U. Michieli and L. Badia, "Game theoretic analysis of road user safety scenarios involving autonomous vehicles," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1377–1381, IEEE, 2018.

[53] N. Lütteken, M. Zimmermann, and K. J. Bengler, "Using gamification to motivate human cooperation in a lane-change scenario," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 899–906, IEEE, 2016.

[54] M. Kabtoul, A. Spalanzani, and P. Martinet, "Towards proactive navigation: A pedestrian-vehicle cooperation based behavioral model," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6958–6964, IEEE, 2020.

[55] A. Alahi, V. Ramanathan, K. Goel, A. Robicquet, A. A. Sadeghian, L. Fei-Fei, and S. Savarese, "Learning to predict human behavior in crowded scenes," in *Group and Crowd Behavior for Computer Vision*, pp. 183–207, Elsevier, 2017.

[56] C.-H. Yu, A. Liu, and P.-C. Zhou, "A multiagent system for simulating pedestrian-vehicle interaction," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 359–364, IEEE, 2014.

[57] J. Casas, J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday, "Traffic simulation with aimsun," *Fundamentals of traffic simulation*, pp. 173–232, 2010.

[58] M. Fellendorf and P. Vortisch, "Microscopic traffic flow simulator vissim," *Fundamentals of traffic simulation*, pp. 63–93, 2010.

[59] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*, ThinkMind, 2011.

[60] G. Soares, Z. Kokkinogenis, J. L. Macedo, and R. J. Rossetti, "Agent-based traffic simulation using sumo and jade: an integrated platform for artificial transportation systems," in *Simulation of Urban MObility User Conference*, pp. 44–61, Springer, 2013.

[61] M. Fiosins, B. Friedrich, J. Görmer, D. Mattfeld, and J. P. Müller, "A Multiagent Approach to Modeling Autonomic Road Transport Support Systems," in *Autonomic Road Transport Support Systems*, (Basel, CH), pp. 67–85, Springer International Publishing, 2016.

[62] F. T. Johora, H. Cheng, J. P. Müller, and M. Sester, "An agent-based model for trajectory modelling in shared spaces: A combination of expert-based and deep learning approaches," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1878–1880, 2020.

[63] D. Yang, L. Li, K. Redmill, and Ü. Özgüner, "Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 899–904, IEEE, 2019.

[64] M. Aschermann, P. Kraus, and J. P. Müller, "Lightjason: a bdi framework inspired by jason," in *European Conference on Multi-Agent Systems*, pp. 58–66, Springer, 2016.

[65] A. Koefoed-Hansen and G. S. Brodal, *Representations for path finding in planar environments*. PhD thesis, Citeseer, 2012.

[66] I. Millington and J. Funge, *Artificial intelligence for games*. CRC Press, 2009.

[67] A. Johansson, D. Helbing, and P. Shukla, "Specification of a microscopic pedestrian mo del by evolutionary adjustment to video tracking data," *arXiv preprint arXiv:0810.4587*, 2008.

[68] E. W. Weisstein, "Rotation matrix," *https://mathworld. wolfram. com/*, 2003.

[69] D. Marutho, S. H. Handaka, E. Wijaya, *et al.*, "The determination of cluster number at k-mean using elbow method and purity evaluation on headline news," in *2018 international seminar on application for technology of information and communication*, pp. 533–538, IEEE, 2018.

[70] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proceedings of the twenty-first international conference on Machine learning*, p. 29, 2004.

[71] G. Borboudakis and I. Tsamardinos, "Forward-backward selection with early dropping," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 276–314, 2019.

[72] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53 – 65, 1987.

[73] G. Zames, N. Ajlouni, N. Ajlouni, N. Ajlouni, J. Holland, W. Hills, and D. Goldberg, "Genetic algorithms in search, optimization and machine learning.," *Information Technology Journal*, vol. 3, no. 1, pp. 301–302, 1981.

[74] G. Amirjamshidi and M. J. Roorda, "Multi-objective calibration of traffic microsimulation models," *Transportation letters*, vol. 11, no. 6, pp. 311–319, 2019.

[75] C. Schiermeyer, F. Pascucci, N. Rinke, V. Berkhahn, and B. Friedrich, "A genetic algorithm approach for the calibration of a social force based model for shared spaces," in *Proceedings of the 8th international conference on pedestrian and evacuation dynamics (PED)*, 2016.

[76] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature*, vol. 407, no. 6803, pp. 487–490, 2000.

[77] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[78] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1349–1358, 2019.

[79] D. Yang and Ü. Özgüner, "Combining social force model with model predictive control for vehicle's longitudinal speed regulation in pedestrian-dense scenarios," in *The 8th Biennial Workshop on Digital Signal Processing for In-Vehicle Systems*, Nagoya University, Japan, 2018.

[80] N. N. C. for Statistics and Analysis, "2016 traffic safety factsheet pedestrians," 2018.

[81] E. F. Camacho and C. B. Alba, *Model predictive control*. Springer Science & Business Media, 2013.

[82] V. L. Bageshwar, W. L. Garrard, and R. Rajamani, "Model predictive control of transitional maneuvers for adaptive cruise control vehicles," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1573–1585, 2004.

[83] Q. Chao, Z. Deng, and X. Jin, "Vehicle–pedestrian interaction for mixed traffic simulation," *Computer Animation and Virtual Worlds*, vol. 26, no. 3-4, pp. 405–412, 2015.

[84] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital control of dynamic systems*, vol. 3. Addison-wesley Menlo Park, CA, 1998.

[85] D. Yang, H. Zhang, E. Yurtsever, K. A. Redmill, and Ü. Özgüner, "Predicting pedestrian crossing intention with feature fusion and spatio-temporal attention," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 2, pp. 221–230, 2022.

[86] A. Rasouli, I. Kotseruba, and J. K. Tsotsos, "Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 206–213, 2017.

[87] I. Kotseruba, A. Rasouli, and J. K. Tsotsos, "Do they want to cross? understanding pedestrian intention for behavior prediction," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1688–1693, IEEE, 2020.

[88] J. Lorenzo, I. Parra, F. Wirth, C. Stiller, D. F. Llorca, and M. A. Sotelo, "Rnn-based pedestrian crossing prediction using activity and pose-related features," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1801–1806, IEEE, 2020.

[89] A. Rasouli, I. Kotseruba, T. Kunic, and J. K. Tsotsos, "Pie: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6262–6271, 2019.

[90] Z. Fang and A. M. López, "Is the pedestrian going to cross? answering by 2d pose estimation," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1271–1276, IEEE, 2018.

[91] F. Piccoli, R. Balakrishnan, M. J. Perez, M. Sachdeo, C. Nunez, M. Tang, K. Andreasson, K. Bjurek, R. D. Raj, E. Davidsson, *et al.*, "Fussi-net: Fusion of spatio-temporal skeletons for intention prediction network," *arXiv preprint arXiv:2005.07796*, 2020.

[92] A. Rasouli, I. Kotseruba, and J. K. Tsotsos, "Pedestrian action anticipation using contextual feature fusion in stacked rnns," in *BMVC*, 2019.

[93] A. Rasouli, T. Yau, M. Rohani, and J. Luo, "Multi-modal hybrid architecture for pedestrian action prediction," *arXiv preprint arXiv:2012.00514*, 2020.

[94] A. Rasouli, M. Rohani, and J. Luo, "Pedestrian behavior prediction via multitask learning and categorical interaction modeling," *arXiv preprint arXiv:2012.03298*, 2020.

[95] I. Kotseruba, A. Rasouli, and J. K. Tsotsos, "Benchmark for evaluating pedestrian action prediction," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1258–1268, 2021.

[96] J. Hariyono and K.-H. Jo, "Detection of pedestrian crossing road," in *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 4585–4588, IEEE, 2015.

[97] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 304–311, IEEE, 2009.

[98] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[99] B. Liu, E. Adeli, Z. Cao, K.-H. Lee, A. Shenoi, A. Gaidon, and J. C. Niebles, "Spatiotemporal relationship reasoning for pedestrian intent prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3485–3492, 2020.

[100] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[101] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," *Syntax, Semantics and Structure in Statistical Translation*, p. 103, 2014.

[102] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

[103] K. Saleh, M. Hossny, and S. Nahavandi, "Real-time intent prediction of pedestrians for autonomous ground vehicles via spatio-temporal densenet," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9704–9710, IEEE, 2019.

[104] K. Saleh, M. Hossny, and S. Nahavandi, "Spatio-temporal densenet for real-time intent prediction of pedestrians in urban traffic environments," *Neurocomputing*, vol. 386, pp. 317–324, 2020.

[105] M. Chaabane, A. Trabelsi, N. Blanchard, and R. Beveridge, "Looking ahead: Anticipating pedestrians crossing with future frames prediction," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2297–2306, 2020.

[106] P. Gujjar and R. Vaughan, "Classifying pedestrian actions in advance using predicted video of urban driving scenes," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 2097–2103, IEEE, 2019.

[107] Z. Fang and A. M. López, "Intention recognition of pedestrians and cyclists by 2d pose estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4773–4783, 2019.

[108] Z. Wang and N. Papanikolopoulos, "Estimating pedestrian crossing states based on single 2d body pose," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, 2020.

[109] P. R. G. Cadena, M. Yang, Y. Qian, and C. Wang, "Pedestrian graph: Pedestrian crossing prediction based on 2d pose estimation and graph convolutional networks," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2000–2005, IEEE, 2019.

[110] A. Rasouli, T. Yau, P. Lakner, S. Malekmohammadi, M. Rohani, and J. Luo, "Pepscenes: A novel dataset and baseline for pedestrian action prediction in 3d," *arXiv preprint arXiv:2012.07773*, 2020.

[111] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[112] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

[113] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[114] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[115] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.

[116] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[117] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421, 2015.

[118] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[119] M. Koç, E. Yurtsever, K. Redmill, and Ü. Özgüner, "Pedestrian emergence estimation and occlusion-aware risk assessment for urban autonomous driving," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 292–297, IEEE, 2021.

[120] W. H. Organization, *Global status report on road safety 2015*, pp. ix, x. World Health Organization, 2015.

[121] "Taxonomy and definitions for terms related to cooperative driving automation for on-road motor vehicles."

[122] "Automated vehicles for safety," Mar 2021.

[123] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, pp. 1–14, 2014.

[124] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016.

[125] S. Brechtel, T. Gindele, and R. Dillman, "Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps," in *17th international IEEE conference on intelligent transportation systems (ITSC)*, pp. 392–399, IEEE, 2014.

[126] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Scalable decision making with sensor occlusions for autonomous driving," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2076–2081, IEEE, 2018.

[127] M. Schratter, M. Bouton, M. J. Kochenderfer, and D. Watzenig, "Pedestrian collision avoidance system for scenarios with occlusions," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1054–1060, IEEE, 2019.

[128] S. Hoermann, F. Kunz, D. Nuss, S. Renter, and K. Dietmayer, "Entering crossroads with blind corners. a safe strategy for autonomous vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 727–732, IEEE, 2017.

[129] M. Lee, K. Jo, and M. Sunwoo, "Collision risk assessment for possible collision vehicle in occluded area based on precise map," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2017.

[130] P. F. Orzechowski, A. Meyer, and M. Lauer, "Tackling occlusions & limited sensor range with set-based safety verification," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1729–1736, IEEE, 2018.

[131] Ö. Ş. Taş and C. Stiller, "Limited visibility and uncertainty aware motion planning for automated driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1171–1178, IEEE, 2018.

[132] M.-Y. Yu, R. Vasudevan, and M. Johnson-Roberson, "Occlusion-aware risk assessment for autonomous driving in urban environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2235–2241, 2019.

[133] M. Naumann, H. Konigshof, M. Lauer, and C. Stiller, "Safe but not overcautious motion planning under occlusions and limited sensor range," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 140–145, IEEE, 2019.

[134] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[135] S. Magdici and M. Althoff, "Fail-safe motion planning of autonomous vehicles," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 452–458, IEEE, 2016.

[136] S. Kousik, S. Vaskov, M. Johnson-Roberson, and R. Vasudevan, "Safe trajectory synthesis for autonomous driving in unforeseen environments," in *ASME 2017 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers Digital Collection, 2017.

[137] M. Koschi, C. Pek, M. Beikirch, and M. Althoff, "Set-based prediction of pedestrians in urban environments considering formalized traffic rules," in *2018 21st international conference on intelligent transportation systems (ITSC)*, pp. 2704–2711, IEEE, 2018.

[138] M. Saberi, K. Aghabayk, and A. Sobhani, "Spatial fluctuations of pedestrian velocities in bidirectional streams: Exploring the effects of self-organization," *Physica A: Statistical Mechanics and its Applications*, vol. 434, p. 120–128, 2015.

[139] I. Bae, J. Moon, J. Jhung, H. Suk, T. Kim, H. Park, J. Cha, J. Kim, D. Kim, and S. Kim, "Self-driving like a human driver instead of a robocar: Personalized comfortable driving experience for autonomous vehicles," *arXiv preprint arXiv:2001.03908*, 2020.

[140] G. Jahangirova, A. Stocco, and P. Tonella, "Quality metrics and oracles for autonomous vehicles testing," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, pp. 194–204, IEEE, 2021.

[141] B. Hejase and U. Ozguner, "Lyapunov stability regulation of deep reinforcement learning control with application to automated driving," in *2023 American Control Conference (ACC)*, pp. 4437–4442, IEEE, 2023.

[142] C. Hatipoglu, U. Ozguner, and K. A. Redmill, "Automated lane change controller design," *IEEE transactions on intelligent transportation systems*, vol. 4, no. 1, pp. 13–22, 2003.

[143] A. Kurt and Ü. Özgüner, "Hybrid state system development for autonomous vehicle control in urban scenarios," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 9540–9545, 2008.

[144] W. Xiao, N. Mehdipour, A. Collin, A. Y. Bin-Nun, E. Frazzoli, R. D. Tebbens, and C. Belta, "Rule-based optimal control for autonomous driving," in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, pp. 143–154, 2021.

[145] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[146] A. Feher, S. Aradi, and T. Becsi, "Q-learning based reinforcement learning approach for lane keeping," in *2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI)*, pp. 000031–000036, IEEE, 2018.

[147] P. Wang, C.-Y. Chan, and A. de La Fortelle, "A reinforcement learning based approach for automated lane change maneuvers," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1379–1384, IEEE, 2018.

[148] S. Nageshrao, H. E. Tseng, and D. Filev, "Autonomous highway driving using deep reinforcement learning," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 2326–2331, IEEE, 2019.

[149] B. Hejase, E. Yurtsever, T. Han, B. Singh, D. P. Filev, H. E. Tseng, and U. Ozguner, "Dynamic and interpretable state representation for deep reinforcement learning in automated driving," *IFAC-PapersOnLine*, vol. 55, no. 24, pp. 129–134, 2022.

[150] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7153–7162, 2020.

[151] D. Isele, A. Nakhaei, and K. Fujimura, "Safe reinforcement learning on autonomous vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–6, IEEE, 2018.

[152] L. Wen, J. Duan, S. E. Li, S. Xu, and H. Peng, "Safe reinforcement learning for autonomous vehicles through parallel constrained policy optimization," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–7, IEEE, 2020.

[153] Y.-C. Chang, N. Roohi, and S. Gao, "Neural lyapunov control," *Advances in neural information processing systems*, vol. 32, 2019.

[154] A. Abate, D. Ahmed, M. Giacobbe, and A. Peruffo, "Formal synthesis of lyapunov neural networks," *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 773–778, 2020.

[155] J. Z. Kolter and G. Manek, "Learning stable deep dynamics models," *Advances in neural information processing systems*, vol. 32, 2019.

[156] H. Dai, B. Landry, L. Yang, M. Pavone, and R. Tedrake, "Lyapunov-stable neural-network control," *arXiv preprint arXiv:2109.14152*, 2021.

[157] T. Shi, D. Chen, K. Chen, and Z. Li, "Offline reinforcement learning for autonomous driving with safety and exploration enhancement," *arXiv preprint arXiv:2110.07067*, 2021.

[158] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[159] H. K. Khalil, *Nonlinear control*. Pearson Higher Ed, 2014.

[160] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *International Conference on Machine Learning*, pp. 146–155, PMLR, 2017.

[161] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the brownian motion," *Physical review*, vol. 36, no. 5, p. 823, 1930.

[162] E. Leurent, "An environment for autonomous driving decision-making." `https://github.com/eleurent/highway-env`, 2018.

[163] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?," in *2017 IEEE intelligent vehicles symposium (IV)*, pp. 812–818, IEEE, 2017.

[164] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.