GAME THEORETIC MODEL PREDICTIVE CONTROL FOR AUTONOMOUS DRIVING

A Dissertation

by

QINGYU ZHANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Reza Langari
Committee Members,	Sivakumar Rathinam
	Alireza Talebpour
	Steve Suh
Head of Department,	Andreas A. Polycarpou

August 2019

Major Subject: Mechanical Engineering

ABSTRACT

This study presents two closely-related solutions to autonomous vehicle control problems in highway driving scenario using game theory and model predictive control.

We first develop a game theoretic four-stage model predictive controller (GT4SMPC). The controller is responsible for both longitudinal and lateral movements of Subject Vehicle (SV) . It includes a Stackelberg game as a high level controller and a model predictive controller (MPC) as a low level one.

Specifically, GT4SMPC constantly establishes and solves games corresponding to multiple gaps in front of multiple-candidate vehicles (GCV) when SV is interacting with them by signaling a lane change intention through turning light or by a small lateral movement. SV's payoff is the negative of the MPC's cost function , which ensures strong connection between the game and that the solution of the game is more likely to be achieved by a hybrid MPC (HMPC). GCV's payoff is a linear combination of the speed payoff, headway payoff and acceleration payoff. . We use decreasing acceleration model to generate our prediction of TV's future motion, which is utilized in both defining TV's payoffs over the prediction horizon in the game and as the reference of the MPC. Solving the games gives the optimal gap and the target vehicle (TV). In the low level , the lane change process are divided into four stages: traveling in the current lane, leaving current lane, crossing lane marking, traveling in the target lane. The division identifies the time that SV should intiate actual lateral movement for the lateral controller and specifies the constraints HMPC should deal at each step of the MPC prediction horizon. Then the four-stage HMPC controls SV's actual longitudinal motion and execute the lane change at the right moment.

Simulations showed the GT4SMPC is able to intelligently drive SV into the selected gap and accomplish both discretionary land change (DLC) and mandatory lane change (MLC) in a dynamic situation. Human-in-the-loop driving simulation indicated that GT4SMPC can decently control the SV to complete lane changes with the presence of human drivers.

Second, we propose a differential game theoretic model predictive controller (DGTMPC) to

address the drawbacks of GT4SMPC. In GT4SMPC, the games are defined as table game, which indicates each players only have limited amount of choices for a specific game and such choice remain fixed during the prediction horizon. In addition, we assume a known model for traffic vehicles but in reality drivers' preference is partly unknown. In order to allow the TV to make multiple decisions within the prediction horizon and to measure TV's driving style on-line, we propose a differential game theoretic model predictive controller (DGTMPC).

The high level of the hierarchical DGTMPC is the two-player differential lane-change Stackelberg game. We assume each players uses a MPC to control its motion and the optimal solution of leaders's MPC depends on the solution of the follower. Therefore, we converts this differential game problem into a bi-level optimization problem and solves the problem with the branch and bound algorithm. Besides the game, we propose an inverse model predictive control algorithm (IMPC) to estimate the MPC weights of other drivers on-line based on surrounding vehicle's real-time behavior, assuming they are controlled by MPC as well. The estimation results contribute to a more appropriate solution to the game against driver of specific type. The solution of the algorithm indicates the future motion of the TV, which can be used as the reference for the low level controller. The low level HMPC controls both the longitudinal motion of SV and his real-time lane decision. Simulations showed that the DGTMPC can well identify the weights traffic vehicles' MPC cost function and behave intelligently during the interaction. Comparison with level-k controller indicates DGTMPC's Superior performance.

ACKNOWLEDGMENTS

First, I would like to give my sincere gratitude to my advisor Dr. Reza Langari for his continuous support of my Ph.D study and related research, for his patience, motivation, passion and immense knowledge. He guidance helped me conquer the challenges I met in my research one after another. What he has taught me is the most valuable treasure of my Ph.D life.

In addition to my advisor, I would like to thank the engineers from Ford motor company: Dimitar Filev, Eric Tseng, Steven Szwabowski and Shankar Mohan. From them have I learned not only expertise, but also the dedicated attitude towards works and the kindness of supporting each other.

I would also like to thank the rest of my committee: Dr. Sivakumar Rathinam, Dr. Alireza Talebpour, and Dr. Steve Suh, for their insightful comments and encouragement.

I would like to thank all my lab mates. It has been a pleasure working with them.

Finally I want to express special thanks to my family for their unconditional love. In particular, I would like to thank my parents. I would not have been able to start my Ph.D life in United States without their support.

CONTRIBUTORS AND FUNDING SOURCE

Contributors

This work was supervised by a dissertation committee consisting of Professor Reza Langari, Sivakumar Rathinam and Alireza Talebpour and Steve Suh of the Department of Mechanical Engineering.

All work for the dissertation was completed by the student, under the advisement of Reza Langari of the Department of Mechanical Engineering, and Dimitar Filev, Eric Tseng, Steven Szwabowski and Shankar Mohan of Ford Motor Company.

Funding Source

This graduate study was supported by Ford Motor Company under the grant URP 2016-8009R.

NOMENCLATURE

Acronyms

ACC	Adaptive cruise control
AV	Autonomous vehicle
CV	Candidate vehicle
DGTMPC	Differential game theoretic model mredictive control
DLC	Discretionary lane change
DV	Dummy vehicle
HV	Human vehicle
IDM	Intelligent driver model
GCV	Game theoretic candidate vehicle
GT	Game theory
GT4SMPC	Game theoretic four-stage model predictive control
IDM	Intelligent driver model
IMPC	Inverse model predictive control
LDAGTC	Linear-decreasing-acceleration-model-based controller
LHS	Left hand side
MIQP	Mixed integer quadratic programming
MLC	Mandatory lane change
MPC	Model predictive control
RHS	Right hand side
POLC	Post-lane-change
PRLC	Pre-lane-change

PV	Preceding vehcle
QP	Quadratic programming
SRV	Surrounding vehicle
SV	Subject vehicle
TV	Target vehicle
V2V	Vehicle to vehicle

Symbols

a	Player's strategy/action in the Game
A	Strategy Set
G	Constraint
J	MPC cost function
k	MPC step index
k_{acc}	Coefficient of acceleration payoff
k_{sf}	Coefficient of safety payoff
k_{sp}	Coefficient of space payoff
n	Discrete time index of prediction horizon
N	MPC prediction horizon
N'	The last step of traveling in the current lane in the MPC prediction horizon
p	Longitudinal position
Q_v	Weight velocity term in MPC cost function
Q_{vN}	Weight of terminal velocity term in MPC cost function
Q_x	Weight of position term in MPC cost function
R_u	Weight of acceleration term in MPC cost function
R_H	Weight of slack variable term in MPC cost function

t	Continuous time index
T_s	Sampling time
u	Acceleration
u_{IDM}^{+}	Maximum acceleration in IDM model
u_{IDM}^{-}	Maximum deceleration in IDM model
U_{sf}	Safety payoff
U_{sp}	Space payoff
U	Payoff
v	Longitudinal velocity
w	MPC weight vector
x	Vehicle states
β	Aggressiveness
β_l	Aggressiveness lower bound
β_u	Aggressiveness upper bound
Δu_{max}	Maximum jerk
Δu_{min}	Minimum jerk
λ	Margin variable

Note: Sometime, a variable is given with a subscript of a certain type of vehicle (like "SV"). Such subscript indicates the belonging of this variable.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION AND LITERATURE REVIEW

1.1 Background and Objectives

The fact that governments, vehicle manufacturers and IT companies are spending billions of dollars in research and experiment of autonomous vehicles (AV) indicated AV has been considered a promising technology. The reason is that AV could bring multiple potential improvements to our society, including increased traffic safety, better environment, reduced utilization of manpower and so forth [1].

However, besides the advantages AV has, there remain some challenges we have to face. One of them is AV-human-driver interaction. As more AVs hit roads, we will soon face blended traffic situations that contain both autonomous, semi-autonomous and human-driven vehicles. Although safety is always of the top priority, a conservative setup in such a situation could lead to a some issues. First, the behavior of an excessively conservative vehicle might look weird from a human driver's perspective. Second, in a blended traffic of both human driver and self-driving cars, human drivers would finally realize the overly-cautious characteristic of such an autonomous car and bully the car for more advantages during the interaction. Therefore, to smoothen the adaptation process, researchers must not only emphasize safety, but should also ensure that self-driving vehicles act in a human-like manner so as to behave consistently with ordinary driving norms [2, 3, 4].

Therefore, this study will proposed a category of AV motion controllers that can address the above issue in highway driving situation.

The controller will consist of two layers. The high level controller that makes strategic decisions and the low level control that controls the AV's specific movements.

Based on kinematic information of both surrounding vehicles (SRV) and subject vehicle (SV), the high level controller should be able to make balanced decisions that consider both SV's safety and driving advantage. Such decisions should be reasonable enough to be made by human drivers. This also indicates that a human-like decision should also includes advanced tasks like overtaking

or MLC, which would require that the high level decision maker should also consider the coordination between longitudinal and lateral movement. In addition, such decision should consider the interaction between human driver and AV. In other words, the high level controller should be able to think from human drivers' perspective and make movements based on their possible strategies.

The low level controller controls the longitudinal motion and lane change. Besides the ability to achieve the decision made by high level controller, the low level controller should also ensure a collision-free path.

1.2 Literature Review

1.2.1 Autonomous Vehicle Decision and Control

We do a thorough literature review on AV decision making and control technique in this section.

The control approaches are divided into three categories in the following text: Separate control, coordinated control and planning-type approaches.

Separate control means controlling the longitudinal motion and lateral motion of the vehicle separately. Coordinated control indicates a certain level of cooperation between the lateral and longitudinal motion. Therefore, the vehicle may perform more complicated behavior like overtaking by acceleration. Path planning approaches plans AV's future path in advance, and a controller is responsible for tracking such path.

1.2.1.1 Longitudinal Control

In terms of longitudinal motion, there are several widely-used models. The General Motors model (or the GHR model) is proposed by Denos C. Gazis, et al. [5]. It was named after GM because it was estimated using data collected on GM test track. The collision avoidance model developed by Gipps is another famous model[6]. The advantage of this model is that it ensures a safety distance between the subject car and the front car, which the GM model fails to consider. Optimal velocity model [7], different from the previous two, is a dynamic model that suggests the car's acceleration is proportional to its optimal speed and actual speed. The next widely-used car following model is the Intelligent Driver Model [8], which define the acceleration as a function of

velocity , net distance gap and the velocity difference in a way different from GM model.

Many other searchers are working on expanding the above model from single-regime to multiple ones. Ahmed considered the following two regimes for his acceleration model in [9]: free-flow regime and car-following regime. He also proposed a good way to evaluate parameters in his model according to real traffic data. Yang also considered emergency regime in MITSIM they established [10]. In [11], the paper includes three more regimes. ACC is another widely-studied driver assistance technique [12, 13] and many variants have been raised like Multi-Objective ACC[14, 15], Model Predictive ACC [16, 14], cooperative ACC [17, 18] etc.

Many other mature control methods has also been used for vehicle longitudinal control. Nouveliere proposed a second-order sliding mode control scheme [19]. In [20, 21], Lyapunov direct methods are used for either linear model or nonlinear one.

1.2.1.2 Lateral Decision and Control

Different from vehicle's longitudinal motion, vehicle's lateral motion is sometimes considered discrete in terms of lanes. Thus researches either consider lane change process as a "decision" by ignoring the specific lateral movement or use models like "bicycle" model to control the vehicle's lateral position continuously. Lateral control discussed in this section is only limited to steering control for regular lane change.

Rule-based lane change decision are widely accepted in papers especially those about microscopic traffic due to the large amount of vehicles in the simulation models. Lane change rule is generally based on gap acceptance in consideration of vehicles' positions, velocities and accelerations.

A common way of designing rules is to categorize lane change into several types according to the urgency of lane change due to the fact that lane changes are usually classified into different types. Gipps and Hidas categorize lane changing into three types: "Essential" , "Desirable" and "Unnecessary" according to vehicle's position to the end of the lane [22, 23]. Ahmed in his dissertation classifies lane change into MLC and DLC according to the purpose of lane change and includes a probabilistic framework before triggering the corresponding rule for each type [9].

The lane change behaviors in SUMO [24] are divided into three types. Strategic (mandatory) lane change, cooperative lane change and tactical (discretionary) lane change, which is a compound of predecessor's work. In [25] the same longitudinal controller is used while MLC and DLC are distinguished by critical gap acceptance analysis. References [26] and [27] all proposed their own way of integrating MLC and DLC in a microscopic traffic study. The former paper, in particular, proposed a vehicle automation and communication system, which controls the longitudinal motion as well as the lateral motion of all the individual vehicles in the system in order to achieve maximum traffic efficiency. A lane-change model that is compatible with a number of car-following models is illustrated in [28]. This lane change rule is also based on standard gap acceptance models. Other researchers consider categorization of lane change in a different way.

Besides gap acceptance, other researchers have set up the problem in a cooperative framework. Kesting thought of modelling lane change behavior in a cooperative way [28]. The rule aims to minimize braking of both the subject car and other vehicles induced by subject car's lane change. In [29] and [30], the authors assumed that two vehicles use Vehicle to Vehicle (V2V) communication to complete the merging behavior cooperatively. Van Arem et al. [31] and Xu et al. [32] used cooperative adaptive cruise control along with V2V technology to achieve collaborative highway merging.

Researchers have also aimed to design controllers that target the precise control of one vehicle in a specific situation. These approaches usually control the steering wheel to track a reference obtained from certain path planners. Schildbach and Borrelli propose an MPC that can plan lane-change behaviors while taking into account the behavior of surrounding vehicles in [33]. In[34], [35] and [36], researchers presented several diversified optimal feedback controllers for vehicle's steering wheel task. Other well-known methods include PID [37], fuzzy control [38] etc.

Intuitively safety plays a more important role in an MLC scenario as compared to DLC because there is limited time to make a decision before crashing into an obstacle or reaching the end of a merging area. The decision could be difficult if there are competing vehicles in the target lane. This requires that one simultaneously evaluate the possibility of entering any one of multiple gaps

instead of considering them sequentially, i.e., one at a time. Thus, designing an autonomous driving system that is capable of handling an MLC scenario effectively requires the balance of factors associated with safety and driving advantages, for example, driver's desire for speed, headway or ideal lane, etc.

Despite many research results on lane changing, some points have been less adequately addressed. Lane change is coordination between vehicle's longitudinal and lateral control, thus longitudinal controllers should serve to create a better lane-change opportunity. A Moreover, V2V technology is yet to be used in practice and competing vehicles in reality are not always cooperative. Therefore, cooperative control scheme needs to be fully validated under a non-cooperative scenario. In addition, human drivers (based on whose behavior lane change strategies are modelled) can actually consider several gaps in their adjacent lane(s) at the same time and choose one that typically optimizes safety and driving advantage. In other words, considering the human driver as a reference is likely of value in this context.

1.2.1.3 Coordinated Control

Generally approaches of coordinated control evaluates longitudinal and lateral motion in the same framework. Utility function is defined as a criterion to represent each decision's advantage to AV.

Belleman introduced Markov Decision Process(MDP) to model systems that are not only controllable but also stochastic [39]. In MDP, the evolution of the system does not purely depends on transition probability but also on action. Brechtel et al. developed a discrete MDP model whose action space includes both longitudinal acceleration and lateral velocities for lane change and collision avoidance [40]. Coskun et al. proposed a fuzzy MDP model for mandatory lane change which allows AV to adjust its longitudinal velocity to fit the velocity of the main traffic stream while executing the MLC [41]. Li et al. illustrate a controller that integrates MPD and game theory. Counting in the interaction between vehicles and model uncertainty, the controller is able evaluate totally seven actions in both longitudinal and lateral direction[42].

Game theory is another promising method of decision making for autonomous vehicle in this

category. It can not only predict surrounding vehicle's behavior by considering their reaction to the subject vehicle, but also flexibly set up the game as needed, which means lateral movement, longitudinal movement or complicated movement can be considered simultaneously under the same criterion and coordination of longitudinal and lateral motion can be easily achieved. The detailed literature review of game theory is given in Section 1.2.2.

Hybrid system is an adequate method for modeling traffic because drivers intuitively consider lane selection as the discrete state and longitudinal position as continuous state [43, 44, 45]. Such feature enables researchers to consider different constraints or continuous dynamics that may be applicable in each discrete state.

1.2.1.4 Path Planning Approaches

Planning-type approaches generally includes two steps. Generating the path and tracking the path. The first step sets up a specific near-future path for the vehicle and the second step is controlling the vehicle to track the path. Usually these approaches' contribution is in the first step while the second step can be achieved by well-known control techniques like PID, sliding mode, h-infinity and so forth.

Many popular path planning algorithms and their variant, like A*, D* [46, 47] are based on the cell decomposition algorithm. These methods decompose the environment into cells and the planner computes explicit robot motions within each cell. With several years evolvement, some variant of the above algorithms have been proposed. [48] described hybrid A* algorithm, which was validated in the DARPA Urban Challenge. The algorithm first obtain a kinematically feasible trajectory phase then improves the solution via numeric non-linear optimization.

Another category is sampling based approaches like probability Road Map (PRM) [49] and rapidly-exploring random tree (RRT) [50]. These approaches construct a set of collision-free trajectories at high efficiency due to its probabilistic feature. In Aoude's papers, the author raised a new algorithm called RR-GP, which is a combination of RRT and Gaussian Process (GP). GP is responsible for obstacle movement prediction and RRT is responsible for feasible path generation. [51, 52] proposed an improved RRT and showed its application to the 2007 DARPA challenge. [53]

presented a new algorithm called Rapidly exploring Random Graph (RRG), and showed that the cost of the best path in the RRG converges to the optimum. [54] developed a game theoretic path planning method based on RRG for multi-robot motion planning problem. In [55, 56, 57], authors illustrated a sampling-based path planning algorithm called B-spline curves. Smooth-primitive constrained-optimization-based path-tracking algorithm.

Potential field method [58] assumes that robots moves in a field of forces. The position to be reached is an attractive pole and obstacles are repulsive surfaces. The control effort can then be easily obtained by summing up the attraction and repulsion. Gradient descending or Newton method are often used to find the optimal waypoint in the field [59].

The dynamics window approach [60, 61] forms the search space of trajectory by considering the dynamics of robots, specifically, the reachable velocities in a time scope.

State lattice [62, 63, 64] samples the state space regularly to generate the path, while taking into account the kinematic and dynamic constraints of the robot. [65, 66] showed their improved work so that the method can be used in more complicated environments with a frequently-varying dominant direction like urban areas. [67] applied the resolution-equivalent grid method to state lattice to solving the coupling between state variables induced by dynamic constraints at high speed.

1.2.1.5 End to End Control

Artificial neural networks can be considered an end to end control approach that is totally different from other. As long as enough training data is give, the networks can function as a controller that mimics the objects which we get the training data from. [68] proposed a deep-neural-network-based autonomous driving scheme. The system was trained in-real time with the training data generated from internal simulators and can intelligently estimate future behavior of surrounding vehicles. [69] demonstrated an autonomous driving system using NVIDIA Drive PX. Instead of decomposing problem into a series of tasks like lane marking detection, path planning, and control, their end-to-end system optimizes all processing steps simultaneously. The system was trained with driver's activity data including lane keeping, lane change and so forth. IT showed

the trained network is able to finish a 10 mile driving task with little help from driver.

1.2.2 Game Theory

Game theory is the formal study of conflict and cooperation between rational agents. It provides researchers a mathematical language to model and to analyze scenarios where there exist strategic interactions.

Game theory has been applied in various ways to study the effects of policy, decisions, and the actions of individual agents in a transportation system. These studies can be broadly classified into two categories: infrastructural regulation studies (traffic control problem) and agent-oriented studies (vehicle placement or route decisions). In the first category, researchers have employed game theory in relation to dynamic traffic control or assignment problems. For instance, Chen and Ben-Akiva [70] adopted a non-cooperative game model to study the interaction between a traffic regulation system and traffic flow to optimally regulate the flow on a highway or an intersection while Zheng-Long and De-Wang [71] addressed the ramp-metering problem via Stackelberg game theory. Su et al. [72] have also used game theory to simulate the evolution of a traffic network.

In the second category, vehicles are regarded as game participants and traffic rules are generally considered to be implicit in the respective decision models [73, 74, 75]. In this context Kita [76] has worked to address the merging-giveaway interaction between a through car and a merging car as a two-person non-zero-sum game. This approach can be regarded as a game theoretic interpretation of Hidas' driver courtesy scheme [77] from the viewpoint that the vehicles share the payoffs or heuristics of the lane-changing process. This leads to a reasonable traffic model although the approach does not address the uncertainties resulting from the actions of other drivers. In recent studies, Talebpour et al. [78, 79] have considered the notion of incomplete information as part of the game formulation process and have developed a model that in certain respects addresses the aforementioned concern. Likewise, Altendorf and Flemisch flow [80] have developed a game theoretic model that addresses the issue of risk-taking by drivers and its impact on the traffic flow. Their study focuses on the cognitive aspects of this decision making process and its impact on traffic safety. Likewise, M. Wang, et al. [81] have used a differential game based controller to

control a given vehicle's car-following and lane-changing behavior while in [82], the authors have applied an Iterative Snow-Drift (ISD) game on cross-a-crossing scenario. In [83], Stackelberg game theory was used solve conflicts in shared space zones while in [84] the authors compared heavy vehicle and passenger car lane-changing maneuvers on arterial roads and freeways. Elvik [85] offers a rather complete review of related works in this area, albeit up to 2014. Meng et al. [86] integrate the idea of receding horizon of MPC with game theory and proposed a game theoretic solution to dynamic lane change situations with incomplete information.

As is shown above, diversified types of games have been applied to transpiration research. The most common one is modeling driver's lane change or merging behavior via a non-cooperative simultaneous game and a pure Nash equilibrium [87, 88]. Others have considered mixed strategy Nash solutions for similar game theory schemes [89, 90, 91]. Recently, advanced game types such as grey game [92], incomplete information game [80, 93] and Stackelberg game [94, 95, 96, 97] are introduced. Differential games are games that concerns control of dynamic systems. One group of researchers proposed several differential-game-based vehicle control frameworks in [98, 81], from non-cooperative to cooperative approaches. Differential games are also increasingly used in related fields such as vehicle/robot formation control [99, 100].

1.2.3 Model Predictive Control

Model predictive control (MPC) is a widely-used control technique. The most important characteristic of MPC is its ability to determine a sequence of control inputs for the model over prediction horizon given the cost function and constraints. However, only the first input of the sequence is applied to the model. The optimization problem is solved periodically using the latest information and the horizon recedes as time goes so MPC is also called receding horizon control [101].

Many researchers have employed MPC to endow autonomous vehicles prediction capability. MPC predicts the future behavior of a system over a finite time horizon and compute a sequence of optimal control inputs that minimizes a cost function, while ensuring satisfaction of given constraints. Gao et al. proposed a nonlinear MPC, the cost function of which considers the distance of the subject vehicle to the obstacle so that MPC is able to generate and follow a collision-free

path[102]. In [103], Rosolia et al. proposed a non-convex MPC for autonomous vehicle, where the non-convexity lies in non-convex constraint of the MPC. To avoid the non-convexity, some people set up their models such that MPC controls only the longitudinal motion[104]. Others employed hybrid MPC by defining the lanes as the discrete state and longitudinal motion as continuous state, so that different constraints or continuous dynamics may be applied[44, 45].

Attempts of combining GT and MPC have been made in some papers. Na et al. proposed a linear quadratic game-based MPC for the control of active front steering system [105, 106]. Yoo et al. raised a collision avoidance technique where model predictive controller controls the vehicle to avoid the unsafe zone generated by the NE [90]. Distributed MPC is another topic on this specific cross-subject field [107, 108]. Despite some literature on this topic, little can be found in combining a non-cooperative game and MPC in a hierarchical way.

1.2.4 Driver Aggressiveness and Estimation

Human drivers' strategies are largely dependent on their driving style. To accurately predict human drivers' future behavior, an autonomous driving system should take into account other vehicles' aggressiveness. However, there are mainly two challenges to achieving this. For one, there is no widely accepted definition for aggressive driving. Most published papers define this notation by driver's self-evaluation in questionnaires [116, 117] or rely on observation of aggressive behaviors in field studies [118, 119]. For another thing, if vehicles are not connected, real-time estimation is almost the only way to identify a driver's aggressiveness, but few published works exist on this subject.

Previous studies on aggressiveness estimation are almost always conducted offline and are based on field observation or questionnaires[120, 121, 122, 123]. These researches mainly study aggressiveness by considering frequency of aggressive behaviors, including honking a horn, tailgating, swearing, forcing, etc., and statistically relate them to drivers' features, like age or gender, statistically. However, these methods are difficult to use with an autonomous driving system for such data can generally only be collected and processed offline.

Another category of papers studies driver aggression's influence in a simulation environment.

Drivers' aggression are often divided into two or three types (conservative, neutral and aggressive) and each type of drivers has quantitatively distinct driving style. For instance, aggressive drivers prefer higher desired speed [124], larger acceleration, smaller following distance [125] when performing car following and larger weight on payoff and smaller weight on collision risk [95] when doing lane change. These papers usually suggest a statistic distribution of these types of drivers on the road and check their influence on the traffic.

In spite of a certain amount of research regarding driving aggressiveness, no published works have proposed a viable approach of estimating their quantity on-line.

1.3 Game Theory Basics

Game theory was systematically established by von Neumann and Oskar Morgenstern in 1944 [126]. Then John Nash introduced the famous Nash equilibrium in noncooperative game theory, at which none of the players can deviate from his own strategy and gain better payoff [127]. From then on, game theory has been applied to numerous fields including economic theory, engineering, sociology, psychology and so forth.

Game is the object of game theory and is a formal mathematical model of an interactive situation. Generally, a game consists of the following elements:

- Player. Players are agents that are involved in the interaction
- Sequence. Sequence is the order that agents play their strategy
- Strategies. Strategies are actions a player can take
- Payoff. Payoff is the reward that a player can gain after every player plays his or her strategy
- Level of cooperation. A game is either cooperative or non-cooperative. In cooperative game, players are optimizing the sum of their payoff. In non-cooperative one, each player is maximizing their own payoff.
- Information Completeness. If any parameter in the payoff of any player is a probability

distribution, then information is incomplete. Then the players are trying to maximize the expected payoff.

- Equilibrium. Equilibrium is a pair of strategies (in order sometimes) that satisfies a set of predefined rules. Equilibria are not necessarily the Pareto optimal, but they generally reflect real human's decision logic. There are many kinds of equilibria. Nash equilibrium, Stackelberg equilibrium, Correlated equilibrium, Sequential equilibrium and so on. They have different definitions and thus fit different kinds of interactions.

Stackelberg game was raised by Heinrich Freiherr von Stackelberg in 1943 [128]. The solution of a Stackelberg game is called Stackelberg equilibrium. Stackelberg game is a two-player sequential game played by a leader and a follower. Here we refer to the leader as "him" and the follower as "her". The leader first commits his strategy and the follower observes the leader's strategy and responds with her own strategy. Each player is trying to maximize his/her own payoff. The leader in Stackelberg games is endowed the power of predicting follower's strategy given his own.

The definition of Stackelberg Equilibrium is formally given in (1.1) to (1.2), where L stands for leader, F stands for follower, a_L and a_F are leader's and follower's strategies, a_L^* and a_F^* are leader's and follower's optimal strategies, A_L, A_F are leader's and follower's strategy spaces, a_L^* is follower's optimal strategy set, U_L and U_F are leader's and follower's payoffs.

$$a_L^* = \underset{a_L \in A_L}{\operatorname{argmax}} \left(\min_{a_F \in A_F^*} U_L(a_L, a_L) \right) \tag{1.1}$$

$$A_F^*(a_L) \triangleq \{a_F^* \in A_F : U_F(a_L, a_F^*) \geq U_F(a_L, a_F), \forall a_F \in A_F\} \tag{1.2}$$

For games of small scale in this paper, Stackelberg equilibrium can be found using exhaustive search, i.e., checking all possible joint strategies one by one.

1.4 Miscellaneous

This section includes some notes that helps readers better understand the main contents of the thesis.

1.4.1 Vehicle Naming

In this section we explain the naming of all the vehicles that appear in the thesis.

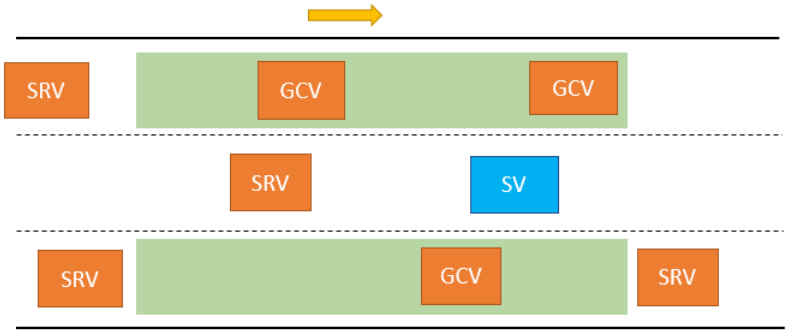

Figure 1.1: Vehicle naming

Fig.1.1 describes a classic highway driving situation. In the figure, the blue vehicle is the vehicle we control and we call it *subject vehicle* (SV). All the orange vehicles are called *surrounding vehicles* (SRV).

Intuitively, the control of SV only needs to consider a limited surrounding area to make decision, so we define a *candidate vehicle scope* for SV as a blue transparent area with green outline in Fig.1.1. The lateral range of the perception scope involves two adjacent lanes and SV's current lane. All the SRVs in the scope are named *game theoretic candidate vehicles* (GCV). GCV is a candidate because SV considers entering the gap in front of GCV. For explanation purpose, we call the vehicle that is in front of a certain GCV preceding vehicle of that GCV (PV). The naming of GCV indicates that SV could interact with GCVs through games. Of all the GCVs, SV will finally choose at most one GCV as the target. And we call it *target vehicle* (TV). And the gap in front of TV is *target gap*. The relationship between each different types of SRVs are shown in Fig. 1.2.

As a routine in game theory, we call SV 'he' and call SRV/GCV/TV 'she' for the convenience of explanation.

According to our definition, obviously, a TV is a GCV and a GCV is an SRV, so in the following

text, the same variables with subscript of “SRV”, “GCV” and “TV” share the same definition. We will use one of the three subscripts that best fits the role of that vehicle in that situation.

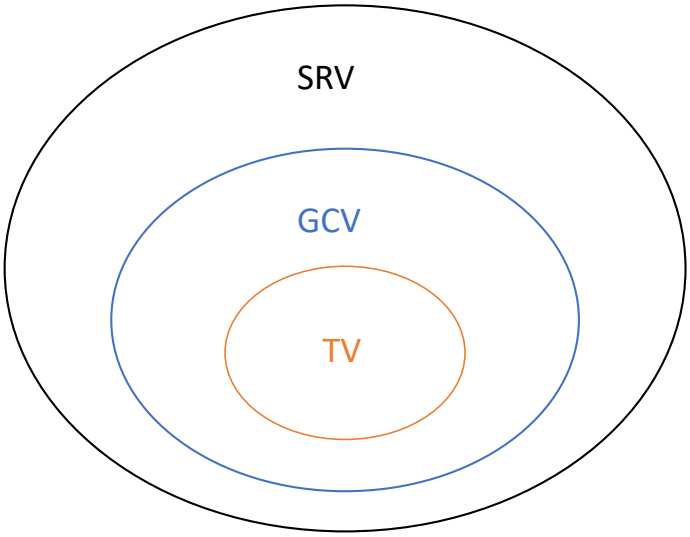

Figure 1.2: Relationships between vehicle names.

1.4.2 Vehicle Kinematic Model

The simple kinematic model in this section is used throughout the whole dissertation.

The longitudinal kinematic model of SV is given as (1.3).

$$\begin{bmatrix} \dot{p}_{SV}(t) \\ \dot{v}_{SV}(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{SV}(t) \\ v_{SV}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_{SV}(t) \tag{1.3}$$

where p_{SV}, v_{SV} and u_{SV} are the longitudinal position, velocity and acceleration of SV respectively.

For implementation in a simulation environment, we follow the the Euler method[129]. We introduce (1.4) to discretize the continuous state space.

$$p_{SV}(k) \approx (p_{SV}(t + T_s) - p_{SV}(t))/T_s \tag{1.4}$$

where T_s is the sampling time.

Combining (1.4) and (1.3) gives us gives the discrete state-space form of SV's kinematic model as (1.5).

$$\begin{bmatrix} p_{SV}(k+1) \\ v_{SV}(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{SV}(k) \\ v_{SV}(k) \end{bmatrix} + \begin{bmatrix} 0 \\ T_s \end{bmatrix} u_{SV}(k) \tag{1.5}$$

where k is the discrete time index.

For lateral movement of SV we simply assume the lane of SV is the discrete state for vehicle's lateral movement. There is no continuous lateral motion.

For SRVs, we assume they have the exact same longitudinal and lateral model as SV.

1.4.3 Controllers of SV and SRV

In the model, there are two types of controllers: SV's controller and SRV's controller. SV always uses GTMPC. The controllers of SRV are different in each chapter and will be described at the beginning of each simulation section. The common feature of those SRV's controllers in each chapter is that they can respond to SV's strategy from a game theoretic approach.

1.5 Organization of the Dissertation

The dissertation will include two pieces of works.

Chapter 2 is about a game theoretic four-stage MPC (GT4SMPC). The chapter is organized as follows. In Section 2.1, the Stackelberg lane change game is clearly defined. In section 2.2, GT4SMPC is introduced. In Section 2.3, the platform is explained and simulations are given. In Section 2.4, conclusions are drawn.

Chapter 3 is about the DGTMPC. In Section 3.1 the structure of DGTMPC is introduced. In Section 3.3 the model of lane change differential game is explained. In Section 3.2, the process of using branch and bound algorithm to solve the Stackelberg differential game is shown. In Section 3.5, the IMPC algorithm is given and described. In Section 3.6, the controller is validated in simulation environment. In Section 3.7, conclusions are drawn.

In Chapter 4, we summarize the whole dissertation, discusses about the potential application

of the work and future improvements.

2. GAME THEORETIC FOUR-STAGE MODEL PREDICTIVE CONTROL

In this chapter we proposed a game theoretic four-stage model predictive controller (GT4SMPC) for highway driving. The GT4SMPC has a two-layer structure. The high-level controller of GT4SMPC is a game theoretic decision model and the low-level controller is a model predictive controller (MPC). The high-level controller is responsible for decision making of surrounding vehicle (SRV). MPC in the lower level controller controls SV's longitudinal motion and SV's lane decision.

To make intelligent decisions on the highway, GT4SMPC constantly establishes and solves Stackelberg games corresponding to multiple game candidate vehicles (GCV). Such interaction is initiated by SV's signaling a lane change intention through turning light or a small lateral movement. By selecting one target vehicle (TV) among multiple GCVs based on SV's payoffs indicated by the Stackelberg equilibrium of the games, GT4SMPC is able to enter the gap in front of the TV.

Satisfactory simulations results showed that the GT4SMPC can achieve intelligent highway driving in simulated dynamic traffic. To further validate the effectiveness GT4SMPC we also conducted a series of human-in-the-loop driving simulations. The experiment results showed the proposed GT4SMPC is able to interact with human driver intelligently.

The schematic diagram of GT4SMPC is given in Fig.2.1.

The high-level controller is where all the games are established. It chooses the closest several vehicles in the game scope defined in Fig. 2.2 as GCVs. Then for each GCV, a game is established and solved to determine the optimal payoff of SV if SV choose to interact with that GCV. A simplified version of the low level MPC is used to generate SV's payoff in the game given GCV's strategy. We also define a payoff function for GCV based on its current headway and its desired speed, which is a function of both players' strategies as well. Knowing all the payoffs of each strategy pair enables us to compute the solution of each game. Comparing the results of each game gives the optimal GCV for SV to interact with, i.e., TV, along with the target gap, both of which are fed to the low-level controller.

The low level controller is a four-stage hybrid model predictive controller, whose name comes from the four stages of the lane change process we define in this paper. Given the signals of the high-level controller, the low-level one is able to drive the vehicle to the desired position and initiate the lane change at the best moment.

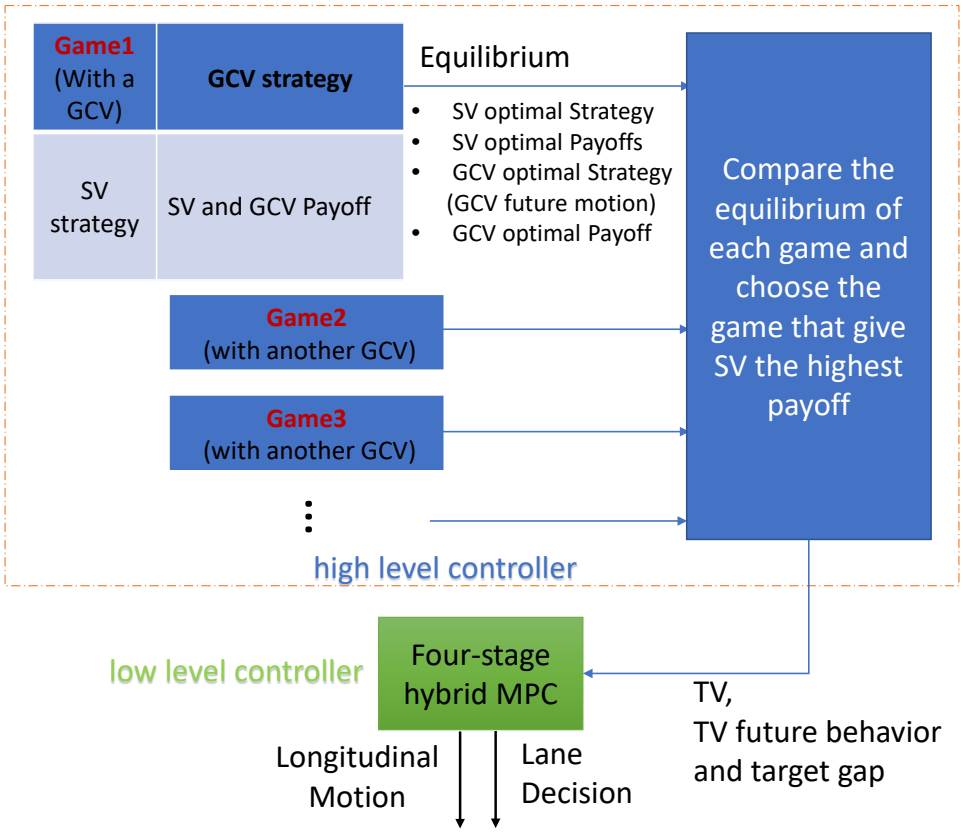

Figure 2.1: Controller schematic diagram.

2.1 Stackelber Game

We assume SV are playing games with GCVs during the lane change process. Here we target the Stackelberg equilibrium of the game because the interaction during the lane-change process can be simplified as a sequential game. SV plays its strategy first by switching on the turning

signal or slightly shifting towards the target lane, and GCV is assumed to respond to SV in a few seconds by possibly applying a different acceleration. The above sequential game is updated and played recursively for the duration of the interaction.

Stackelberg game was raised by Heinrich Freiherr von Stackelberg in 1943 [128]. The solution of a Stackelberg game is called Stackelberg equilibrium. Stackelberg game is a two-player sequential game played by a leader and a follower. Here we refer to the leader as "him" and the follower as "her". The leader first commits his strategy and the follower observes the leader's strategy and responds with her own strategy. Each player is trying to maximize his/her own payoff. The leader in Stackelberg games is endowed the power of predicting follower's strategy given his own.

The definition of Stackelberg equilibrium is given as (2.1)-(2.2).

$$a_L^* = \underset{a_L \in A_L}{\operatorname{argmax}} \left(\min_{a_F \in A_F^*} U_L(a_L, a_F) \right) \tag{2.1}$$

$$A_F^*(a_L) \triangleq \{a_F^* \in A_F : U_F(a_L, a_F^*) \geq U_F(a_L, a_F), \forall a_F \in A_F\} \tag{2.2}$$

where a_L^* and a_F^* are the optimal strategies of leader and follower respectively, A_L and A_F are the strategy set of leader and follower, U_L and U_F are payoff functions of leader and follower. In this paper, as we have defined above, SV is the leader and GCV is the follower. Their payoffs are already given in (2.17) and (2.3). SV's strategy set is all the possible values N_0' can be, where N_0' is SV's strategy. GCV's strategy set includes all the possible accelerations that GCV can choose as well as instantaneous lane change to adjacent lanes.

According to [130], equilibrium of small-scale games like those in this paper can be easily found by checking all the possible solutions.

To fully define the lane-change Stackelber game, we need to define its players, and their strategies and payoffs.

2.1.1 Game Scope and Players

In Fig. 2.2, the game scope is defined as the green area. This indicates the assumption that SV is able to interact with every vehicles within that area and the possibility that SV could enter any

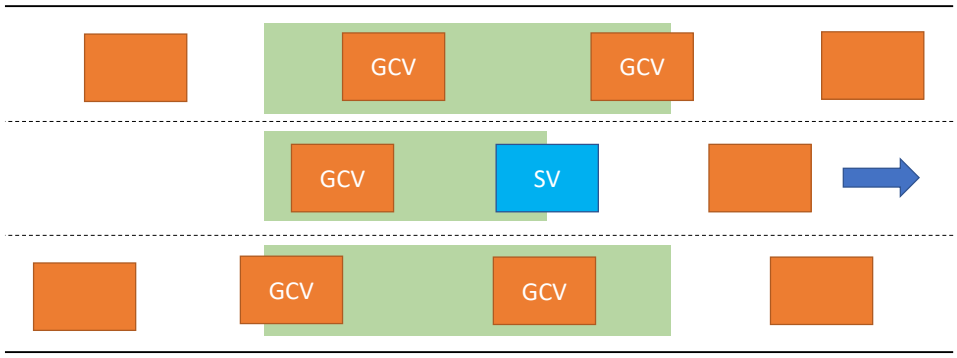

Figure 2.2: The green area is the game scope.

gap "guarded" by a vehicle in green area in adjacent lane. In addition, SV's action also has some impact on the vehicle that follows him at any time, so we assume interaction between them too. All the vehicles in green area are called game candidate vehicle (GCV).

For each GCV and SV pair, we establish a two-player Stackelberg game between them. Thus in each game, the players are SV and one GCV. We assume SV is the leader in the game for two reasons. First, if SV initiates a lane change, it is more reasonable to define SV as the leader. Second, we do not want our SV just passively react to the traffic but actively pursue advantages. For more details of the Stackelberg game, readers may refer to [130].

2.1.2 GCV Strategies and Payoffs

In our set-up, we assume the GCV is able to move in four directions as her response to SV in the game. To be specific, we define several longitudinal accelerations and instantaneous lane switch as GCV's strategy (Fig. 2.3).

GCV's payoff is defined as (2.3).

$$U_{GCV}(k) = \sum_{k=1}^{N} (U_{pos}(k) + U_{vel}(k)) \tag{2.3}$$

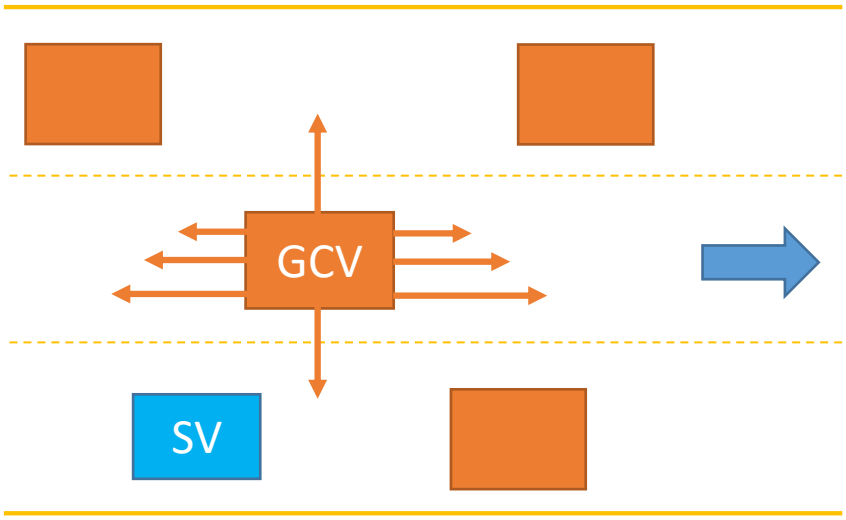

Figure 2.3: GCV strategy space.

where

$$U_{pos}(k) = \begin{cases} p_{GCV}(k), & \text{if } \Delta p(k) \geq \Delta p_d \\ p_{PV}(k) + (k_p - 1)\Delta p_d - k_p \Delta p(k), & \text{if } \Delta p(k) < \Delta p_d \end{cases} \tag{2.4}$$

$$U_{vel}(k) = k_v |v_d - v_{GCV}(k)| \tag{2.5}$$

$$\Delta p(k) = p_{PV}(k) - p_{GCV}(k) \tag{2.6}$$

In (2.3) to (2.6), U_{pos} is the position part of the payoff, U_{vel} is the velocity part of the payoff, p_{GCV} is the longitudinal position of GCV, v_{GCV} is GCV's velocity, v_d is GCV's desired speed, PV is the preceding vehicle of GCV after all players play their strategies, p_{PV} is the PV's longitudinal position, $\Delta p(k)$ is GCV's current space headway, Δp_d is GCV's desired space headway, k_p and k_v are two negative parameters relating to the two parts of the payoff function.

The integration form of (2.3) indicates that GCV is considering the benefit of both spacing and velocity over the prediction horizon like SV. Equation (2.4) means that GCV wants not only to be as close to her destination as possible, but also to maintain safe distance by keeping desired space

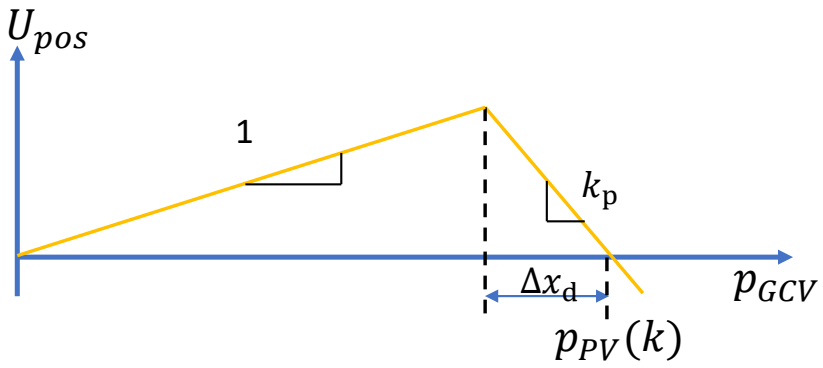

Figure 2.4: The position part of GCV's payoff function.

headway from any PV. Equation (2.5) is GCV's desire for the ideal speed. The position part of GCV's payoff function can be drawn as Fig. 2.4.

Which vehicle is PV really depends on two players' strategies. If GCV changes lane, then SV is assumed to change lane successfully and GCV's new preceding vehicle becomes PV. If GCV does not change lane, then the new PV could be SV if SV changes lane successfully or the original PV if SV fails. This reflects the interaction between two players of the game.

2.2 The Four-Stage MPC, SV Strategies and Payoffs

In this section, we introduce the so called four-state MPC and talk about how we define SV's strategies and payoffs in the game.

First, we would like to divide the lane change process of SV into four stages:

1. Traveling in the current lane. (TCL)
2. Leaving current lane. (LCL)
3. Crossing lane marking. (CLM)
4. Traveling in the target lane. (TTL)

Such classification is for pure control purpose in terms of constraints handling as well as longitudinal and lateral control coordination.

While in TCL or LCL stage, SV is assumed to only consider the constraints associated with current lane. When in CLM stage, SV should pay heed to constraints of both lanes. When SV arrives at TTL stage, only constraints related to target lane needs attention.

To better understand the role each stage plays in MPC, we draw Fig. 2.5. In the figure, the red rectangle is the prediction horizon. As time goes, it gradually moves towards the right side. N_0', N_1' and N_2' are certain steps symbolizing the borders of adjacent stages in the prediction horizon if the horizon covers that border.

N_0' will not show in the MPC cost function, because the purpose of LCL stage is to help GT4SMPC plan the best moment to start the lateral movement of the lane switch process. Such signal is sent to the steering controller so that the predicted moment when SV crosses lane marking, which is computed by MPC, matches SV's actual lateral motion, over which GT4SMPC does not have full control.

For simplicity, $N_1' - N_0'$ and $N_2' - N_1'$ are fixed during the process of solving the MPC problem. Therefore for a certain game happening at a certain simulation step, N_0', N_1' and N_2' are dependent. If one of them is determined, then all of them are. In reality, $N_1' - N_0'$ and $N_2' - N_1'$ can vary depending on how fast SV plans to execute the lane change. Such variation can be directly applied to the current controller, but consequently it produces extra computation load because of a larger strategy space of SV.

Based on the definition of the four stages, the four-stage hybrid MPC is proposed as (2.7). Please notice that the MPC is for both payoff generation of the game in high level and the control of SV's motion in low level.

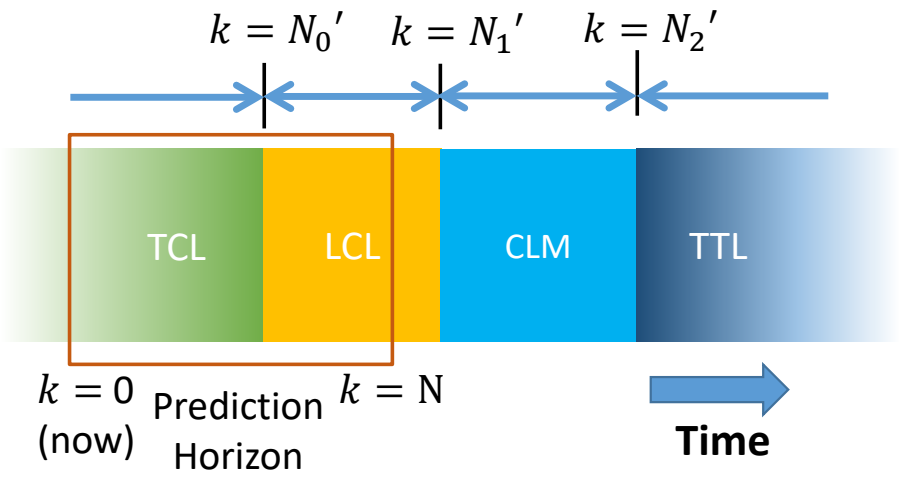

Figure 2.5: The four stages of lane change in MPC.

$$
\begin{aligned}
J(v,u,h_1,h_2,N_1',N_2') = & \sum_{k=1}^{N_1'}(Q_v|v(k)-v_{des}|^2 + R_u|u(k)|^2 + R_{h1}|h_1(k)|^2) \\
& + \sum_{k=N_1'+1}^{N_2'}(Q_v|v(k)-v_{des}|^2 + R_u|u(k)|^2 \\
& + R_{h1}|h_1(k)|^2 + R_{h2}|h_2(k)|^2) \\
& + \sum_{k=N_2'+1}^{N}(Q_v|v(k)-v_{des}|^2 + R_u|u(k)|^2 + R_{h2}|h_2(k)|^2) \\
& + Q_{vN}|v(N)-v_{des}|^2
\end{aligned} \tag{2.7}
$$

subject to

$$v(k) \geq 0 \tag{2.8}$$

$$u_{min} \leq u(k) \leq u_{max} \tag{2.9}$$

$$\Delta u_{min} \leq u(k) - u(k-1) \leq \Delta u_{max} \tag{2.10}$$

$$h_1(k) \geq 0 \tag{2.11}$$

$$h_2(k) \geq 0 \tag{2.12}$$

for $0 \leq k \leq N_2'$

$$p(k) < h_1(k) + p_{CPV}(k) \tag{2.13}$$

$$p(k) > h_1(k) + p_{CFV}(k) \tag{2.14}$$

for $N_1' + 1 \leq k \leq N$

$$p(k) < h_2(k) + p_{PPV}(k) \tag{2.15}$$

$$p(k) > h_2(k) + p_{GCV}(k) \tag{2.16}$$

where N is the prediction horizon, $u(k)$ is SV's longitudinal acceleration, i.e., a control variable. $h_1(k)$ and $h_2(k)$ are two slack variables in constraints, v_{des} is SV's desired velocity, Q_v is the weight on the desired speed of SV, Q_{vN} is the weight on the terminal term, R_u, R_{h1} and R_{h2} are the weights on u, h_1 and h_2 respectively, u_{max} and u_{min} are the upper and lower bound of u. Δu_{max} and Δu_{min} are the bounds of $u(k) - u(k-1)$. PPV, CFV and CPV are just local naming of surrounding vehicles in terms of a specific GCV according to Fig. 2.6.

The second row of (2.7) is SV's cost in TCL and LCL stages, the third and forth rows are his cost in CLM stage, and the fifth row is his cost in TTL stage and the last row is the terminal cost. Simply speaking, in all stages, SV cares about both speed and control effort, i.e., acceleration. In

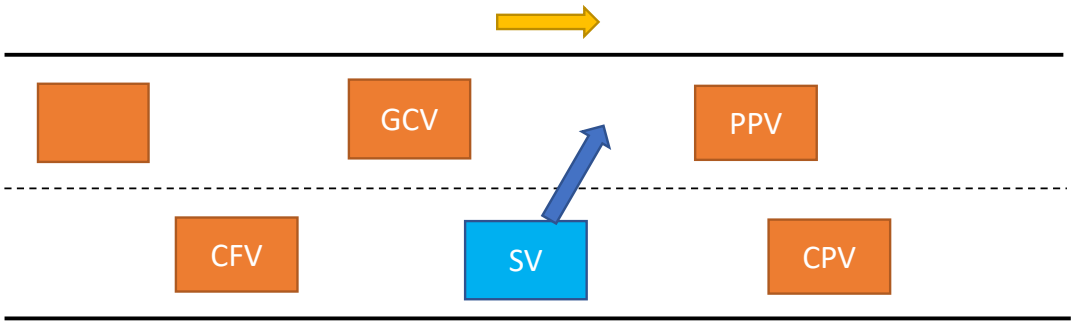

Figure 2.6: The local naming of MPC formulation.

TCL and LCL stages, SV tries to minimize the slack variable associated to the safety concern in current lane. In TTL stage, SV is minimizing that of the target lane. In CLM stage, SV cares about both. The meanings of constraints (2.8) to (2.12) are self-evident. (2.13) and (2.14) are the soft constraints that help SV keep distance from preceding and following vehicles in the current lane, whose slack variables also appear in (2.7). Similarly, (2.15) and (2.16) are the safety constraints associated with the target lane.

We need to point out that (2.7) is abused here. If N_1' or N_2' is not in the prediction horizon, then corresponding row will not appear in the cost function .

From (2.7) we can see that GCV's strategy is also influencing SV's trajectory planning in the game. Here we use a linear decreasing acceleration model in[131] as SV's prediction of GCV's future motion in SV's game if GCV's strategy is certain longitudinal acceleration. If GCV's startegy is lane change, then we assume lane change is completely immediate and GCV follows constant velocity.

Given the MPC, we simply define the payoff function of SV as (2.17).

$$U_{SV} = -J \tag{2.17}$$

In addition, since we know that N_0', N_1' and N_2' are dependent, any one of them can be considered as SV's strategy of the game.

2.3 Simulations

In this section, we validate GT4SMPC's performance in two scenarios.

The first scenario is a normal highway driving situation. SV is trying to achieve high speed in a comparatively slow traffic by competing with traffic vehicles. All the traffic vehicles are controlled by a linear decreasing acceleration game theoretic controller (LDAGTC).

In the second scenario, we test GT4SMPC in a mandatory lane change scenario. The blue vehicle is controlled by a human in real time. The human will intentionally prevent SV from lane change so that we can observe how GT4SMPC behaves in such an extreme case. The black vehicles are still controlled by LDAGTC.

LDAGTC is a linear-decreasing-acceleration-model-based controller that can respond to SV's lane change intention in a game theoretic way. According to the definition of Stackelberg game, a follower, which is a traffic vehicle controlled by LDAGTC in the simulation, does not predict the action of leader (SV), but passively observes it and makes the corresponding response. Therefore, a traffic vehicle actually does not need to establish a game of her own. She only observes SV's current state, and infer SV's strategy and his trajectory in a certain way. Here we assume LDAGTC uses constant acceleration model to predict SV's future motion, and a lane-change rule based on gap acceptance to determine how soon SV is going to execute the lane change. Then LDAGTC will choose certain acceleration to either prevent SV's lane change intention or yield to SV as long as that acceleration maximizes her payoff. Here, we still use (2.3) as traffic vehicle's payoff function.

One main concern for the simulations is information completeness. According to the text above, SV and traffic vehicles share follower's payoff function, follower's model and the current states of all vehicles, but we also assume traffic vehicles do not know SV's strategy or MPC's trajectory and has to infer them. Such assumption brings uncertainty to the game: traffic vehicle's actual behaviors might be different from SV's prediction. Therefore, the information of the simulation is incomplete. In addition, we let human control a traffic vehicle in Scenario II. Since GT4SMPC can by no means know a human's model or his/her payoff function, it could prove GT4SMPC's performance and robustness in situations where traffic vehicles' model is completely

unknown.

In both simulations, the high level of the controller is operated at 2Hz and the low level is operated at 5Hz. This means if the high level is not updated since its last activity, the low level will continue using the previous result of the high level. In high level the prediction horizon N is 4 steps, in low level it is 20. The prediction time is always 4s. The selection of these parameters is a result of balancing MPC's speed and performance. A shorter prediction time undermines MPC's motivation of lane change while a longer one involves too much future uncertainty. The selected frequencies make sure GT4SMPC can well handle the dynamic traffic in real time.

2.3.1 Scenario I

Multiple vehicles are traveling rightward on a three-lane road (Fig. 2.7). The red vehicle (SV) is controlled by GT4SMPC. Black vehicles, each of which is given a number for the convenience of explanation, are controlled by LDAGTC. There are two black vehicles in each lane. No.1 (right one, not shown in Fig. 2.7) and No.2 (left one) are traveling in the upper lane at 10m/s, No.3 (right one) and No.4 (left one) are traveling in the middle lane at 15m/s. No.5 (right one) and No.6 (left one) are traveling in the lower lane at 20m/s. SV starts at 10m/s and wants to achieve high speed in such a traffic.

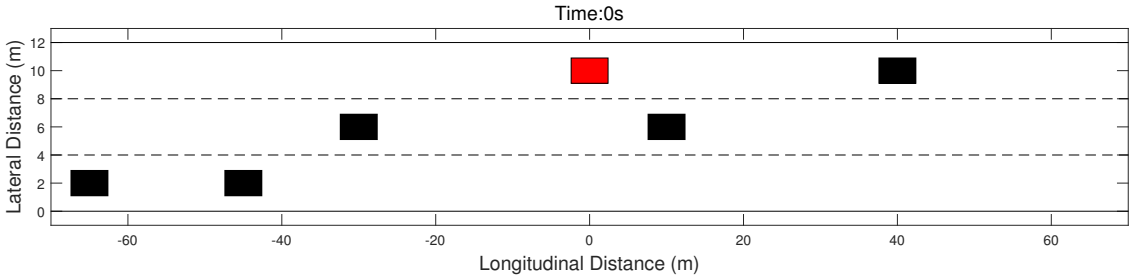

Figure 2.7: Initial condition of Scenario I.

Fig. 2.7 to Fig. 2.9 shows how SV made two consecutive lane changes to reach the desired speed.

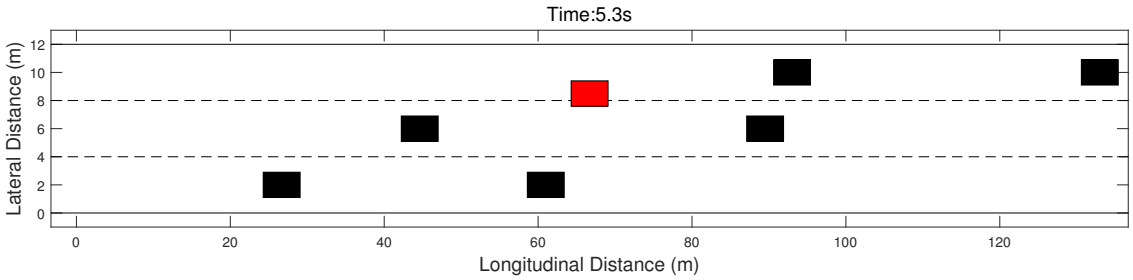

Figure 2.8: SV's first lane switch.

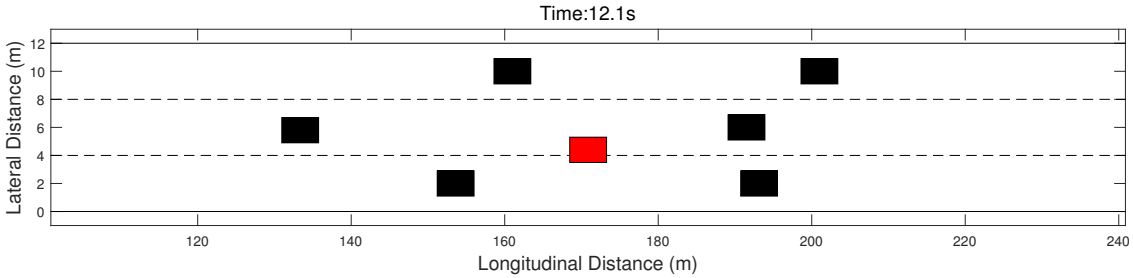

Figure 2.9: SV's second lane switch.

According to the game scope that we have defined in Fig. 2.2, SV is constantly observing the surrounding gaps. At the beginning, since the vehicles in the top lane are slow and those in the middle lane are faster, SV would be willing to enter the middle lane. Thus SV chose No.4 to interact with (Fig. 2.11). By predicting No.4's future behavior in a game theoretic way, SV thought that No.4 would accelerate mildly , which is indeed close to No.4's actual acceleration (Fig. 2.12). Thus, SV plans a lane change between t=3.8s and t=7s as is shown in Fig. 2.10. The lines of N_0', N_1' and N_2' in the figure divide each lane change into the four stages. According to the definition of N_0', N_1' and N_2', SV is in TCL stage between t=0s and t=3.8s, in LCL stage between t=3.8s and t=5s, in CLM stage between t=5s and t=7s and in TTL stage after t=7s. This is how SV went through the four stages during the lane change.

Similar to the first lane change, SV attempted the second lane change after entering the middle lane and seeing the higher speed of vehicles in the bottom lane. This time SV predicts that the new TV No.6 is likely to maintain speed in the game. According to Fig. 2.12, No.4's real behavior is

almost the same as SV prediction. Then SV accomplished the lane change between t=10.2s and t=13.6s.

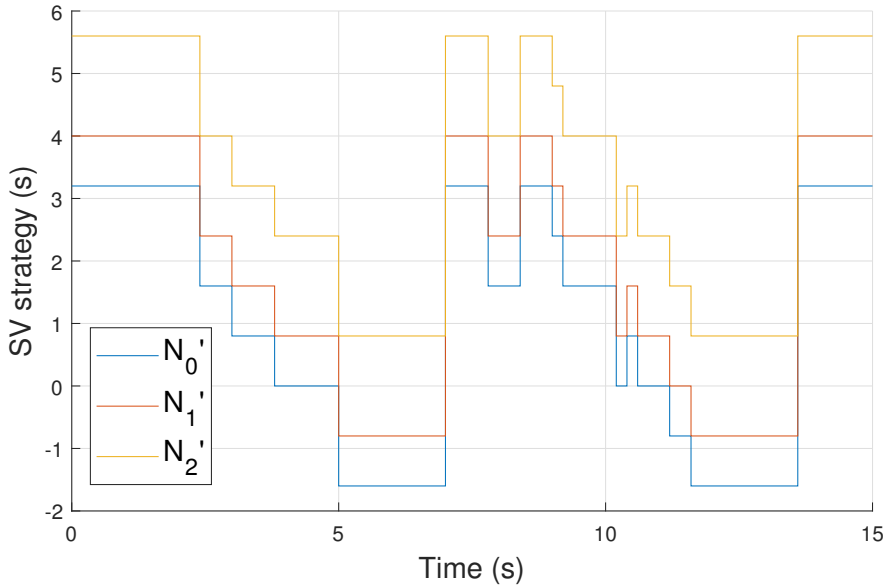

Figure 2.10: SV strategy.

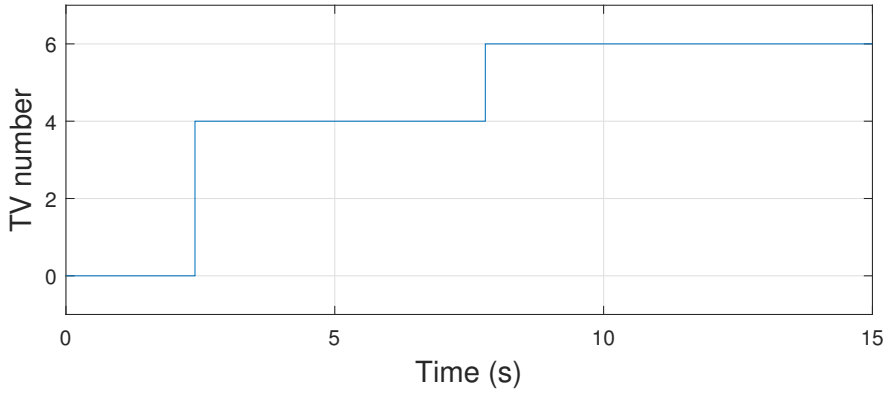

Figure 2.11: TV selection.

We would like to point out here that actually the traffic vehicle can lane change and GT4SMPC also considers that possibility. We intentionally created this scenario where traffic vehicles do not

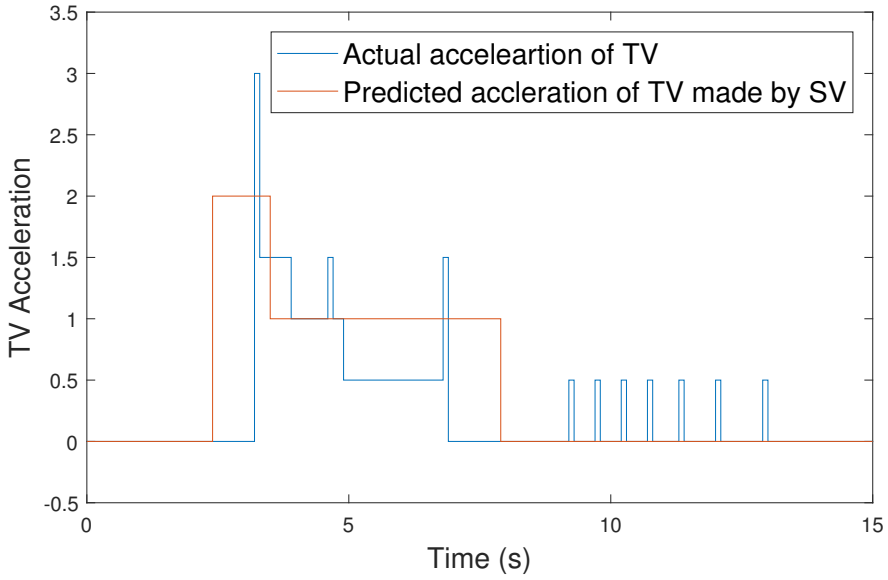

Figure 2.12: Predicted TV acceleration and actual TV acceleration.

change lane in order to make SV's life more difficult.

2.3.2 Scenario II

Now we would like to test GT4SMPC's performance at the presence of human driver. We build a driver interface in MATLAB SIMULINK environment so that human can control the blue vehicle on the road in Fig. 2.13. During the simulation, the human driver can use the left and right arrows on the keyboard to adjust the blue vehicle's longitudinal velocity and use the up and down arrows to select her desired lane. A built-in lateral controller will drive the blue vehicle into the target lane. In addition, this scenario is also designed to show GT4SMPC's capability of running in real time.

We set up a scenario as Fig. 2.13. The red vehicle (SV) is still controlled by GT4SMPC and wants to achieve higher speed. The blue vehicle (No.4) is controlled by a human. The black vehicles use LDAGTC. The black vehicles in the lower lane have high desired speed (20 m/s), where the left one is No.3 and the right one is No.2. Working as an obstacle, another black vehicle (No.1) is traveling slowly (5 m/s) in front of the red vehicle, which is shown in Fig. 2.14. During the simulation, the human vehicle will intentionally prevent the red vehicle from lane change.

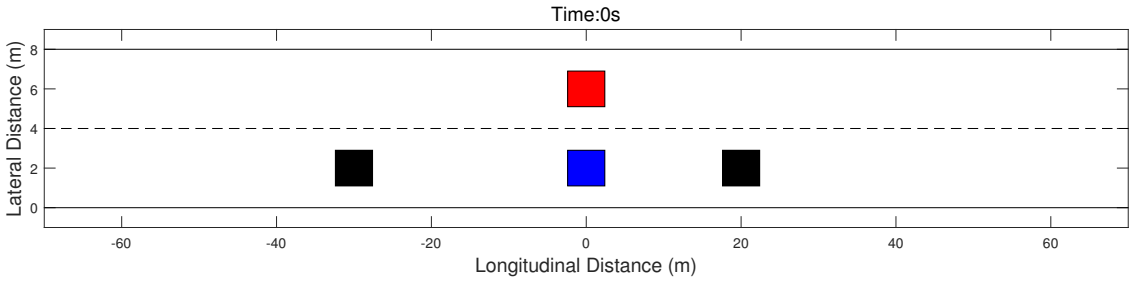

Figure 2.13: Initial condition of Scenario II.

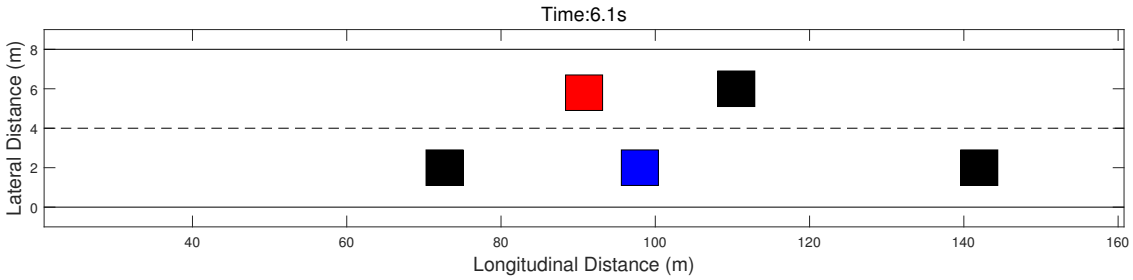

Figure 2.14: SV's lane change intention being prevented by human driver.

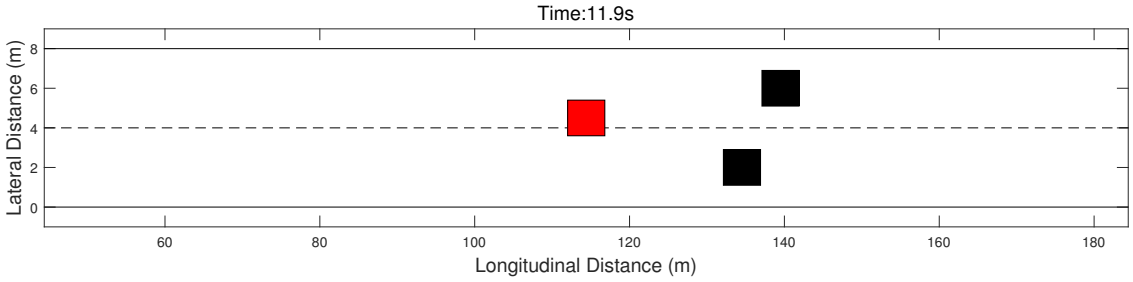

Figure 2.15: SV's final lane switch.

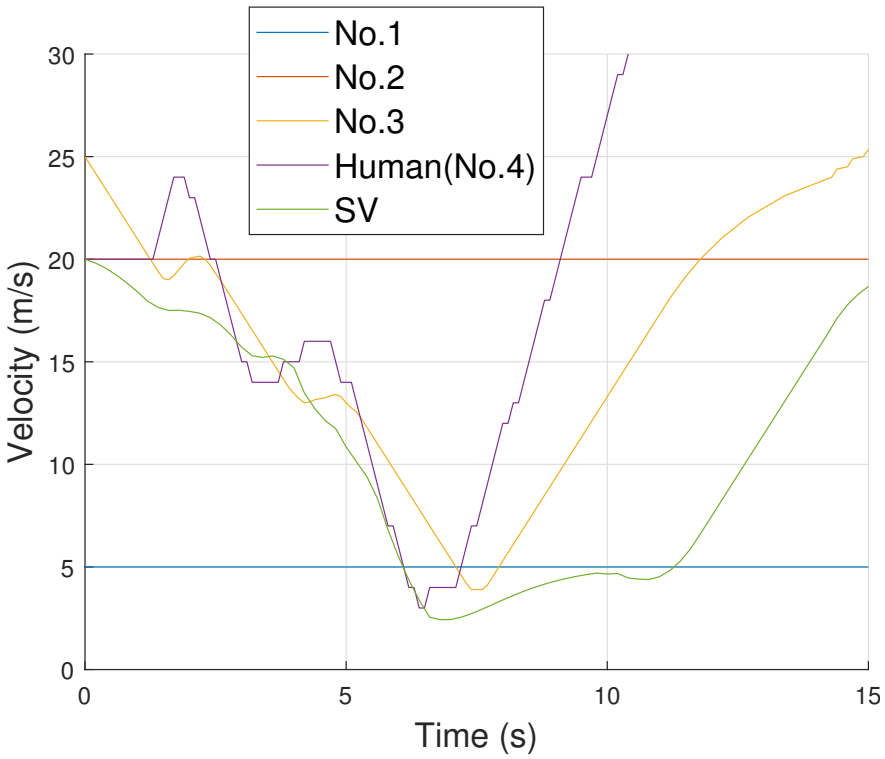

Figure 2.16: Vehicles' velocities.

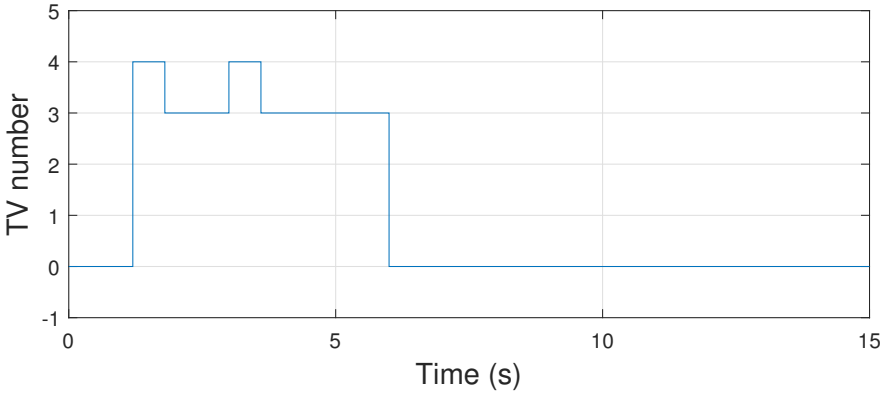

Figure 2.17: TV selection.

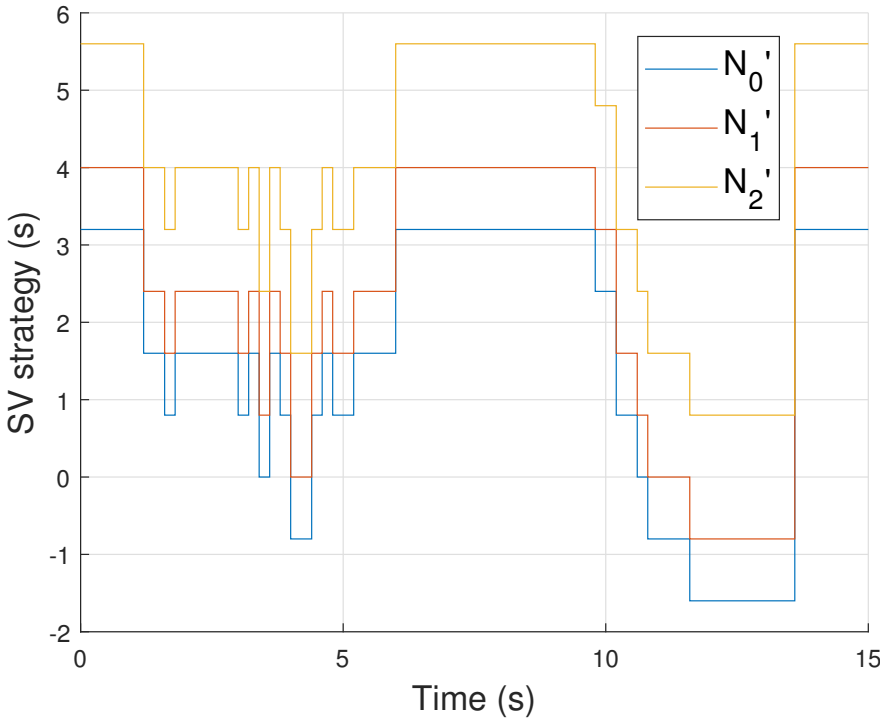

Figure 2.18: SV strategy.

According to Fig. 2.13 to Fig. 2.16, as SV was approaching No.1, the human vehicle was decelerating and trying to block SV. Targeting either No.3 or No.4 during t=0s to t=6s (Fig. 2.17), GT4SMPC was always planning a lane change in the near future and N_0' even hit 0 occasionally, which triggered the lane change, but the lane change was then aborted after GT4SMPC's awareness of human driver's hostile behavior, according to Fig. 2.18. Therefore, SV kept staying in the merging lane and was not able to overtake the human driver at all before t=6s.

After human driver started accelerating after t=6s, No.3 also began to accelerate to prevent SV's lane change intention, securing her position advantage. Taking this into consideration, SV continued selecting the current lane as the target lane until No.3 passed SV at around t=9s. Then SV decided to tug behind No.3 and completed the lane change according to Fig. 2.18. In Fig. 2.17, the fact that TV became 0 after t=6s means that SV was not trying to overtake any vehicle anymore.

The above simulation proves that GT4SMPC is able to run in real time. Solving the game is

indeed a heavy task for the high level controller and it takes MATLAB about 0.3s to get the result, so we set the game frequency at 2Hz, which is an acceptable value if we consider human driver's frequency of decision making. The low level controller can actually run as fast as 20Hz, but we still let it run at 5Hz, which is fast enough for our simulations.

The two simulations showed GT4SMPC's performance in different cases. Scenario I proved GT4SMPC's ability of predicting surrounding vehicle's behavior in a game theoretic situation. Scenario II showed GT4SMPC's robustness in an extreme situation created by human driver.

2.4 Conclusion

The chapter proposed a novel game theoretic model predictive controller for autonomous highway driving. The four-stage MPC divides lane change process into four stages of different constraints, forming a hybrid MPC. With the help of game theory, the controller takes the possible response of traffic vehicles into account and is able to make rational decisions in dynamic traffics. Two simulations demonstrated GT4SMPC's decent performance without knowing the exact model of surrounding vehicles. Since this piece of work is based on vehicle kinematic model, future works include testing the controller on vehicle dynamic models and using a driving simulator to study the interaction between GT4SMPC and human drivers in a virtual while realistic environment.

However, the GT4SMPC also has two biggest drawbacks. First, the GCV model is assumed to play a fixed strategy during the prediction horizon. We wish that the GCVs in our model could make a sequence of decisions during their prediction horizons. Second, we have assumed fixed payoff function for GCV, which means in our simulations all the drivers are assumed to be exactly the same. In real life, however, drivers are different and it's not reasonable to use the same payoff function to represent all the GCVs.

3. DIFFERENTIAL GAME THEORETIC MODEL PREDICTIVE CONTROL

In this section we propose a differential game theoretic model predictive controller (DGTMPC) for autonomous driving. The motivation of the new controller mainly comes from the weaknesses of the GT4SMPC: the fixed strategy and the fixed payoff function of GCV.

We consider the fixed strategy assumption over the prediction horizon unrealistic because in real life drivers are able to make a series of decisions (like Fig.3.1) when necessary. For instance, for a driver yielding to a another vehicle cutting in front of her, she could plan a strategy like this: decelerate to create a big gap and then accelerate to achieve the original speed. In spite of the simplicity of the strategy, the previous GT4SMPC did not include such decision sequence for GCV at all. Therefore, in our new DGTMPC we assume two players are playing a Stackelgerg differential game where each player is able to make a decision at each step of the N-step game , where N is the prediction horizon.

The fixed payoff function is another unrealistic assumption for the GT4SMPC as well as for almost all researches that model traffic via a game theory approach. In literatures, parameters that are related to driving style in the game theoretic model are mainly determined in two ways. The first one is using values directly given by authors[132, 42, 80]. The first approach is groundless since authors do not give enough support for choosing such values. The second one is off-line calibration [93, 133, 134, 135]. In this category, researches use real drivers' data from on-line database or experiments to calibrate parameters in the payoff functions. This approach seems solid but actually the calibration result only reflects the average driving style of the drivers in the data set or only a small bulk of the drivers involved in the experiment.

Targeting the above two issues, the DGTMPC in this section considers a much larger action space for the GCV in the game and uses inverse MPC to capture a specific driver's feature through on-board observation.

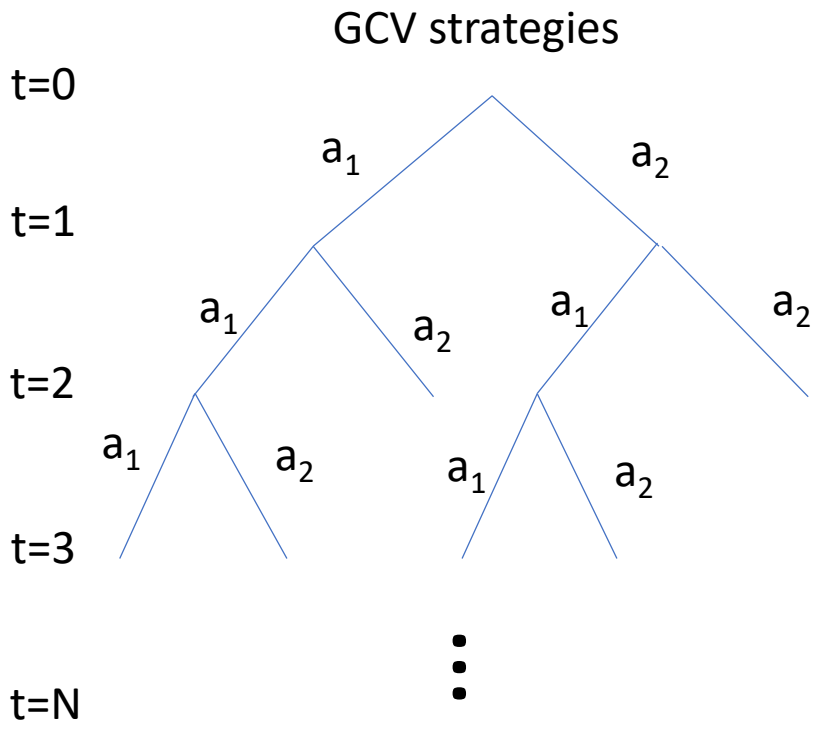

Figure 3.1: GCV's decision tree.

3.1 DGTMPC Schematic Diagram

The schematic diagram of DGTMPC is given in Fig. 3.2. As GT4SMPC, DGTMPC has two levels: the high level and the low level. The high-level controller not only selects the TV and the target gap for the low level controller but also gives a prediction of the TV's future motion within the prediction horizon. Then the low level controller uses these information as the reference to control the precise motion of the SV.

Inside the high-level controller there are two blocks: the differential game block and the inverse MPC block. The differential block solves the Stackelberg differential games and outputs all the information the the lower-level controller needs including the TV, the TV's future motion and the equilibrium of the game. The differential game has two modes: overtaking mode and the guarding mode. In the overtaking mode, SV is planning multiple games and in each game SV is considering overtake a GCV. Then a TV is selected from multiple GCV along with a target gap.

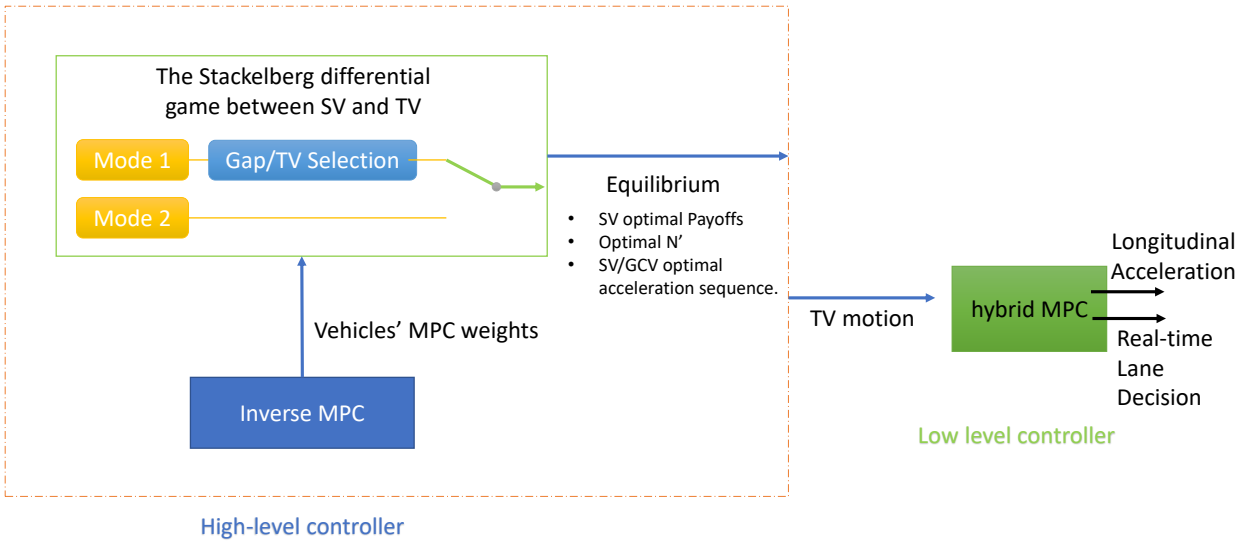

Figure 3.2: Schematic diagram of DGTMPC.

In the guarding mode, SV is being overtaken by the TV. A switch logic is designed to switch between these two modes. The inverse MPC block estimates the weights of the cost function of SRVs in the differential game through real-time observation of SRV's movements given the assumption of the differential game that all SRVs follow MPC. The lower level is a hybrid model predictive controller. It receives information from the high level and outputs both SV's longitudinal acceleration and real-time lane decision.

3.2 Differential Game Basics

In spite of the fact that meaning of "dynamic" in dynamic system means "the state of the system varies over time according to differential equations involving time derivatives" [136], the "dynamic" in the term "dynamic game" has more general meaning. According to [130] a game is dynamic if at least one player is allowed to use a strategy that depends on previous actions. On the other hand, a game in which the players act only once and independently of each other is definitely called a static game.

Differential game can be considered as a branch of dynamic games. It is accepted as a general name for games wherein the evolution of the states of the game is described by differential equations and the players act throughout a time interval [130]. For the discrete case of a differential game, which is the type of the game this research focuses on, players make an action at each step of the duration of the game.

An open-loop N-person discrete differential game of fixed duration with complete information involves the following elements:

1. The player set $\boldsymbol{P} = 1, ..., n_p$.

2. A time interval [1,N] which is specified a priori and which denotes the duration of the game.

3. The trajectory space of the game $\mathbf{X}$ whose elements are denoted as $\{\boldsymbol{x}(k), 0 < k < N\}$. For any time index $k \in [1, N]$, $\boldsymbol{x}(k) \in \mathbf{X}$.

4. The control space of the game $\mathbf{U}_i$ of player i whose elements are denoted as $\{u_i(k), 0 < k < N\}$. We also define $\boldsymbol{u} = \boldsymbol{u}_1 \cup ... \cup \boldsymbol{u}_{n_p}$, $\mathbf{U} = \mathbf{U}_1 \cup ... \cup \mathbf{U}_{n_p}$,

5. The strategy space A_i of player i which is a class of mapping $a_i : [1, N] \times \mathbf{X} \rightarrow \mathbf{U}_i$ where $u_i(k) = a_i(k, \boldsymbol{x})$ Each element in A_i is a feasible strategy that is subject to the constraints of each player $G_i(\boldsymbol{x}(k), \boldsymbol{u}(k)) \leq 0$

6. A difference equation which describes the dynamics of the system.

$$\boldsymbol{x}(k+1) = f(k, \boldsymbol{x}(k), \boldsymbol{u}(k)), \;\; \boldsymbol{x}(0) = \boldsymbol{x}_0$$

7. Cost functions for each player $J_i : [1, N] \times \mathbf{X} \times \mathbf{U} \to \mathbf{R}$ is defined as:

$$J_i(\boldsymbol{u}(k)) = -\sum_{k=0}^{N} j_i(k, \boldsymbol{x}(k), \boldsymbol{u}(k))$$

The concept of equilibrium of Nash or Stackelberg game in the previous chapter can be applied to differential games. An N-tuple of strategies $\{a_i^* \in A_i, i \in \boldsymbol{P}\}$ is a Nash equilibrium if the following inequalities are satisfied for any $\{a_i \in A_i, i \in \boldsymbol{P}\}$

$$\begin{cases} J_1^* \triangleq J_1(a_1^*, a_2^*, ..., a_{n_p}^*) \leq J_1(a_1, a_2, ..., a_{n_p}) \\ J_2^* \triangleq J_1(a_1^*, a_2^*, ..., a_{n_p}^*) \leq J_2(a_1, a_2, ..., a_{n_p}) \\ ... \\ J_{n_p}^* \triangleq J_1(a_1^*, a_2^*, ..., a_{n_p}^*) \leq J_{n_p}(a_1, a_2, ..., a_{n_p}) \end{cases} \tag{3.1}$$

For a two player differential game, a 2-tuple of strategies $\{a_L^*, a_F^*\}$ is a Stackelberg equilibrium if the following inequalities satisfied for any $\{a_L \in A_L, a_F \in A_F\}$

$$\begin{cases} a_L^* = \text{argmin}_{a_L \in A_L} \left(\max_{a_F \in A_F^*} J_L(a_L, a_F) \right) \\ A_F^*(a_L) \triangleq \{a_F^* \in A_F : J_F(a_L, a_F^*) \leq J_F(a_L, a_F)\} \end{cases} \tag{3.2}$$

3.3 The Lane Change Differential Game

The researches of differential game was initiated by Rufus Isaacs in [137]. Since then all kinds of differential games have been applied to various aspects of engineering field. The application of differential games to ground vehicle control did not draw much attention until recent years.

Currently, the literature on differential-game-based autonomous vehicle control can be categorized into two categorries with respect to the scenario of the study. The first one is autonomous racing [138, 139, 140]. Both vehicles in the game want to reach the destination ahead of the other. The second one is collision avoidance in the motion planning problem [141, 142, 140]. Both players need to reach its own destination, say traveling across unsignalized intersection, without colliding with the other. In addition to the above two categories, Huang uses mean field game [143] to study the behavior of mass behaviors instead of individual agents in traffic flow. Wang et al. apply cooperative differential game to car-following and lane change model [81]. Lygeros et al. study the platooning behavior of two vehicles from the perspective of zero-sum game. However, few literature models general highway driving scenario as non-cooperative differential games.

Solving the differential game means finding the equilibrium, where the majority of the literature targets either the Nash equilibrium or the Stackelberg one. On one hand, there are multiple ways to find the Nash equilibrium of a differential game nowadays, notwithstanding the limited application of each approaches. In Isaac's research [137] , the Nash equilibrium of the pursuit and evasion game can be found by solving for the Hamilton-Jacobi-Isaac equation, which is basically a two player version of Hamilton-Jacobi-Belleman equation in optimal control. Iterated best response algorithm fits the underlying nature of Nash equilibrium and is more practical in some cases because it does not suffer from the curse of dimensionality [144, 145, 146]. In [147], the general form of open-loop Nash equilibrium of an unconstrained linear quadratic game can be obtained by solving the Riccati equation. In [141], Dreves finds the Nash equilibrium from the perspective of a single optimal control problem in the case that cost functions of both players are independent. On the other hand, both the analytic form and the numeric form of the Stackelberg equilibrium of the differential game have been studied by previous researchers. In [130], the analytic form of both the open-loop and feedback solution to the unconstrained Stackelberg differential game are given, where the former is obtained by solving a two-point boundary value problem and the later is found by solving a Hamilton-Jacobi-Bellman equation. Long and Sorger showed that the problem can be turned to a standard optimal control problem in certain case [148]. Bensoussa et al.

employ the maximum principle to derive the necessary conditions for the Stackelberg solutions in stochastic game [149]. The numeric form is often studied for the constrained case where analytic solution cannot be derived. Given the fact that the open-loop solution to the constraint Stackelberg game is an equivalent to the bi-level optimization problem, researchers usually use the latter term in their papers to refer to both types of problems. Branch and bound [150, 151, 152], brand and cut [153, 154], and trust-region method [155, 156, 157] are among the most accepted solution to the bi-level optimization problem in spite of its expensive computation . Zhu and Martinez solve the receding-horizon dynamic Stackelberg game by solving an N-horizon quadratic program[158]. Meng and Zeng use genetic algorithm to solve for the one-leader N-follower Stackelberg game [159].Despite plenty of researches on this topic, solving the differential game remains a hard problem due to the fact that each appraoch is only applicable to conditional scenario.

In this section we try to apply the idea of Stackelberg differential game to lane change scenario in highway driving. Specifically, we set up two lane change scenarios as two types of differential game. For the first type, SV is constantly considering overtaking others to gain more driving benefits. For the second type, SV is guarding the gap in front of him from another vehicle who is trying to enter the gap. We set up games for these two scenarios separately because SV is playing a different role in these scenarios.

In either scenario, we always define SV as the leader and TV as the follower of the Stackelberg differential game. We give two reasons for this assumptions: First, playing leader means SV/DGTMPC knows of what the other players need and can take advantage of it. Second, in Stakelberg game, the follower is completely passive, so it is less meaningful to study the problem from a passive perspective.

Rather than the dissimilar model for the two players in the static Stackelberg game in Chapter 2, in the Stackelberg differential game both players use MPC to control their behaviors and compete with each other.

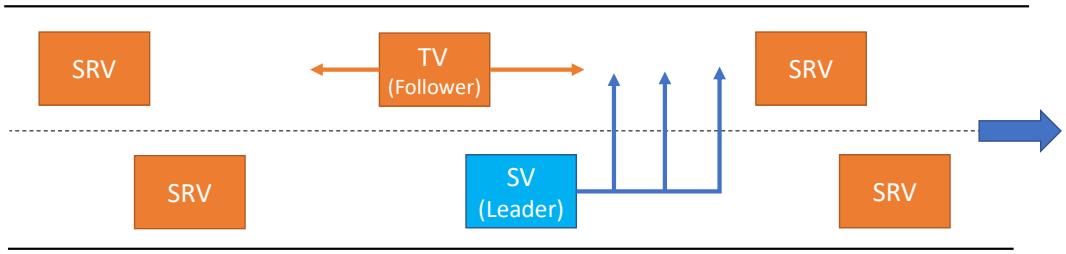

Figure 3.3: The first game scenario.

3.3.1 DGTMPC Mode 1

In the first game mode(Fig. 3.3). SV is trying to overtake a TV. As always, we define the game through defining the players, strategies and payoffs respectively. The players of the game are SV and TV. The action of the leader is his longitudinal acceleration at each step over the prediction horizon ($u_L(k)$ for $k = 1, 2, ..., N$) and the lane change moment he choose (N'_L). The follower's action is just her longitudinal acceleration at each step of the prediction horizon ($u_F(k)$ for $k = 1, 2, ..., N$). The payoff of each player is the negative of each player's cost function ($-J_L$ and $-J_F$).

In mode 1, the MPC problems of the leader and the follower are defined as (3.3) to (3.8) and (3.9) to (3.13) respectively.

$$\min_{u_L, h_L} J_L(v_L, u_L, h_L) = \sum_{k=1}^{N} (Q_{vL}|v_L(k) - v_{des}|^2 + R_{uL}|u_L(k)|^2 + R_{hL}|h_L(k)|^2) \tag{3.3}$$

s.t.

$$v_L(k) \geq 0 \tag{3.4}$$

$$u_{min} \leq u_L(k) \leq u_{max(k)}$$

$$0 \leq h_L(k) \leq h_{max} \tag{3.5}$$

For $1 \leq k \leq N'_L$

$$p_L(k) < -h_L(k) + p_{LPV}(k) \tag{3.6}$$

For $N'_L + 1 \leq k \leq N$

$$p_L(k) < -h_L(k) + p_{FPV}(k) \tag{3.7}$$

$$p_L(k) > p_F(k) + h_L(k) \tag{3.8}$$

$$\min_{u_F, h_F} J_F(v_F, u_F, h_F) = \sum_{k=1}^{N} (Q_{vF}|v_F(k) - v_{des}|^2 + R_{uF}|u_F(k)|^2 + R_{hF}|h_F(k)|^2) \tag{3.9}$$

s.t.

$$v_F(k) \geq 0 \tag{3.10}$$

$$u_{min} \leq u_F(k) \leq u_{max}$$

$$0 \leq h_F(k) \leq h_{max} \tag{3.11}$$

For $1 \leq k \leq N'_L$

$$p_F(k) < -h_F(k) + p_{FPV}(k) \tag{3.12}$$

For $N'_L + 1 \leq k \leq N$

$$p_F(k) < -h_F(k) + p_L(k) \tag{3.13}$$

In (3.3) and (3.9), the meaning of each variables are defined exactly the same as those in Chapter 2. We locally define the names of SRVs with respect to a TV as Fig. 3.4. The formulation of the MPC is also a mimic of the MPC problem in Chapter 2 except two main changes. First,

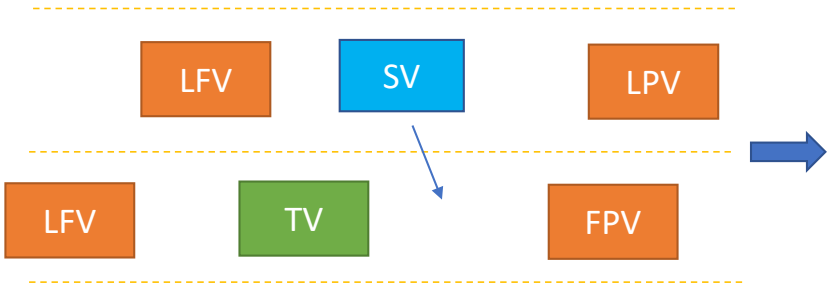

Figure 3.4: The local naming of the differential game.

the rear safety constraint is now abandoned when either leader or follower is in the original lane. This is due to the fact that most drivers do not care about following vehicle but only the preceding vehicle while driving on the road. Second, SV is the vehicle who does lane change, so it only have one safety constraint (3.6) before lane change, but two safety constraints (3.7) and (3.8) after lane change. TV is the vehicle who guards the gap, so it has one safety constraint (3.12) before lane change as well as one constraint (3.13) after lane change.

3.3.2 DGTMPC Mode 2

In the second game mode(Fig. 3.5), TV is trying to over take SV and SV can choose either to yield or to block. The players of the game remain the same. The action of the leader is only his longitudinal acceleration at each step over the prediction horizon ($u_L(k)$ for $k = 1, 2, ..., N$) and now the follower's action includes both her longitudinal acceleration at each step of the prediction horizon ($u_F(k)$ for $k = 1, 2, ..., N$) and the lane change moment she choose (N'_F). The payoff of each player is still the same as mode 1.

In mode 2, two players' MPC problems are defined as (3.14) to (3.16) and (3.17) to (3.21) respectively.

$$\min_{u_L, h_L} J_L(v_L, u_L, h_L) = \sum_{k=1}^{N} (Q_{vL}|v_L(k) - v_{des}|^2 + R_{uL}|u_L(k)|^2 + R_{hL}|h_L(k)|^2) \tag{3.14}$$

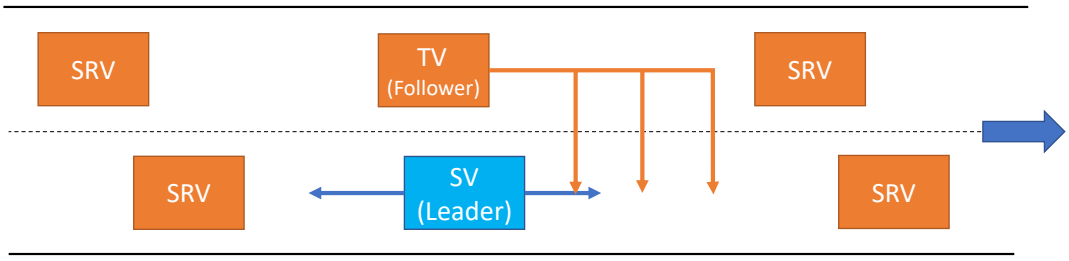

Figure 3.5: The second game scenario.

s.t.

$$v_L(k) >= 0$$

$$u_{min} \leq u_L(k) \leq u_{max(k)}$$

$$0 \leq h_L(k) \leq h_{max}$$

For $1 \leq k \leq N'_L$

$$p_L(k) < -h_L(k) + p_{LPV}(k) \tag{3.15}$$

For $N'_L + 1 \leq k \leq N$

$$p_L(k) < -h_L(k) + p_F(k) \tag{3.16}$$

$$\min_{u_F, h_F} J_F(v_F, u_F, h_F) = \sum_{k=1}^{N} (Q_{vF}|v_F(k) - v_{des}|^2 + R_{uF}|u_F(k)|^2 + R_{hF}|h_F(k)|^2) \tag{3.17}$$

s.t.

$$v_F(k) >= 0 \tag{3.18}$$

$$u_{min} \leq u_F(k) \leq u_{max}$$

$$0 \leq h_F(k) \leq h_{max}$$

For $1 \leq k \leq N'_L$

$$p_F(k) < -h_F(k) + p_{FPV}(k) \tag{3.19}$$

For $N'_L + 1 \leq k \leq N$

$$p_F(k) < -h_F(k) + p_{LPV}(k) \tag{3.20}$$

$$p_F(k) > p_L(k) + h_F(k) \tag{3.21}$$

As is shown from (3.3) to (3.21), the roles of two players are exchanged in the mode 2 of the game, so now SV only one safety constraint (3.6) before lane change, and one safety constraint (3.16), but TV has one safety constraint (3.19) before lane change but two constraints (3.20) and (3.21) after lane change.

3.3.3 DGTMPC Mode Swtich Logic

Since DGTMPC can only work in one of the two modes at a moment, we develop a mode switch logic as Fig. 3.6. Before describing the logics, we make the assumption that two players cannot turn on the turning signal at the same time so that players will not be overtaking each other simultaneously. By default, the DGTMPC works at mode 1, seeking lane change opportunity to reach desired speed while constantly monitoring SRV's lane change intention. If SV identifies any advantage of a lane change, SV will then turn on the turning light until lane change succeeds or fails. If SV's turn signal is off and is in mode 1, SV will switch to mode 2 as soon as he sees the

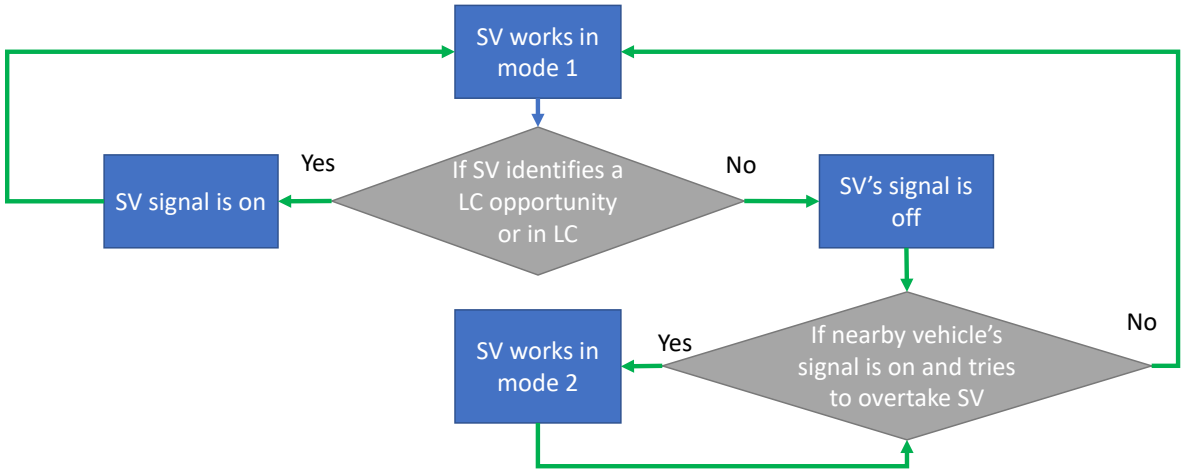

Figure 3.6: The switch logic of DGTMPC between mode 1 and mode 2.

turning signal of a TV who is trying to overtake SV. SV will switch back to mode 1 from mode 2 when TV gives up or finishes her lane change.

3.4 Solving the Stackelberg Differential Game

In this section, we describe the complete mathematical formulation of the Stackelberg differential game and show how we find the Stackelberg equilibrium of the game.

Stackelberg game is a sequential game where leader's first play his strategy and then based on the leader's strategy the follower makes her best response. Therefore, the Stackelberg differential game can be written as a bi-level MPC problem as (3.22) to (3.27).

$$\min_{u_L(k)} J_L(v_L, u_L, h_L) = \sum_{k=1}^{N} (Q_{vL}|v_L(k) - v_{des}|^2 + R_{uL}|u_L(k)|^2 + R_{hL}|h_L(k)|^2) \tag{3.22}$$

$$s.t. \quad G_L^{MPC}(\boldsymbol{x}_L, \boldsymbol{x}_F, \boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0 \tag{3.23}$$

$$\boldsymbol{x}_L(k+1) = A\boldsymbol{x}_L(k) + B\boldsymbol{u}_L(k) \tag{3.24}$$

$$\min_{u_F(k)} J_F(v_F, u_F, h_F) = \sum_{k=1}^{N} (Q_{vF}|v_F(k) - v_{des}|^2 + R_{uF}|u_F(k)|^2 + R_{hF}|h_F(k)|^2) \tag{3.25}$$

$$s.t. \quad G_F^{MPC}(\boldsymbol{x}_L, \boldsymbol{x}_F, \boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0 \tag{3.26}$$

$$\boldsymbol{x}_F(k+1) = A\boldsymbol{x}_F(k) + B\boldsymbol{u}_F(k) \tag{3.27}$$

where

$$\boldsymbol{x}_L = \begin{bmatrix} p_L \\ v_L \end{bmatrix}, \boldsymbol{x}_F = \begin{bmatrix} p_F \\ v_F \end{bmatrix}, \boldsymbol{u}_L = \begin{bmatrix} u_L \\ h_L \end{bmatrix}, \boldsymbol{u}_F = \begin{bmatrix} u_F \\ h_F \end{bmatrix}, A = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ T_s \end{bmatrix}$$

The problem is "bi-level" because it has an inner problem and an outer one. The outer problem (3.22) to (3.24) is the leader's optimization problem subject to leader's system dynamics (3.24), which is an equivalent of (1.5), and to all constraints in leader's MPC problem (3.23), which is an equivalent of (3.4) to (3.8) or (3.3.2) to (3.16). Besides all the equality and inequality constraints, the outer problem should also satisfy the inner problem: the variables associated with the follower are the optimal solution to the inner problem. Similar to the outer problem, the inner problem (3.25) to (3.27) is the follower's optimization problem subject to follower's system dynamics (3.27), which is ac tally the same as leader's, and to all constraints in follower's MPC problem (3.26), which is an equivalent of (3.10) to (3.13) or (3.18) to (3.21).

There are totally three steps to solve the Stackelberg differential game or the bi-level MPC problem in our research. For a certain N'

1. Convert the bi-level MPC problem to a bi-level quadratic program (QP).

2. Convert the bi-level QP into a single level QP.

3. Use Branch and Bound algorithm to solve the single level QP problem. The solution gives the equilibrium of the game.

3.4.1 Convert the Bi-Level MPC Problem to a Bi-Level QP

To solve a linear MPC problem, the most-adopted approach is to convert the MPC problem into a quadratic program by taking into the dynamics of the system into consideration. By doing this, the same control variable of different steps of the MPC become different variables in the new QP problem. If the QP problem is convex, then the optimum of the problem, which is also the solution of the MPC problem, can be found by optimization algorithms like active-set algorithm[160] or interior point algorithm[161].

Reference [162] derives the process of rewriting a linear MPC problem into a quadratic program. For an MPC problem as follows,

$$\begin{aligned}
\min_{\boldsymbol{u}(k)} J(\boldsymbol{x},\boldsymbol{u}) &= \sum_{k=1}^{N}[(\boldsymbol{x}(k)-\boldsymbol{x}_{ref})^{\intercal}Q(\boldsymbol{x}(k)-\boldsymbol{x}_{ref})+(\boldsymbol{u}(k)-\boldsymbol{u}_{ref})^{\intercal}R(\boldsymbol{u}(k)-\boldsymbol{u}_{ref})] \\
s.t. \quad & G^{MPC}(\boldsymbol{x},\boldsymbol{u}) \leq 0 \\
& \boldsymbol{x}(k+1) = A\boldsymbol{x}(k)+B\boldsymbol{u}(k)
\end{aligned}$$

it is equivalent to the following quadratic program:

$$\begin{aligned}
\min_{\boldsymbol{u}(k)} J(\boldsymbol{u}) &= J_{const}+0.5\boldsymbol{u}(k)^{\intercal}H\boldsymbol{u}(k)+\boldsymbol{u}(k)^{\intercal}[F(\boldsymbol{x}_0-\boldsymbol{x}_{ref})-H\boldsymbol{u}_{ref}] \\
s.t. \quad & G(\boldsymbol{u}) \leq 0
\end{aligned}$$

where

$$H \triangleq \Gamma^{\intercal} Q \Gamma + R \tag{3.28}$$

$$F \triangleq \Gamma^{\intercal} Q \Omega \tag{3.29}$$

$$\Gamma = \begin{bmatrix} B & 0 & ... & 0 & 0 \\ AB & B & ... & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ A^{N-1}B & A^{N-2}B & ... & ... & B \end{bmatrix} \tag{3.30}$$

$$\Omega \triangleq \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \tag{3.31}$$

$$G = Lu - W \tag{3.32}$$

where the exact definition of L and W can be found on page 108 of [162].

Following this idea, we are able to rewrite the bi-level MPC as a bi-level QP problem as

$$\min_{\boldsymbol{u}_L} J_L(\boldsymbol{u}_L, \boldsymbol{u}_F)$$

s.t.

$$G_L(\boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0$$

$$\min_{\boldsymbol{u}_L} J_F(\boldsymbol{u}_L, \boldsymbol{u}_F)$$

s.t.

$$G_F(\boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0$$

3.4.2 Convert the Bi-Level QP into a Single Level QP

To solve the bi-level QP problem, we would need to get rid of the inner problem. Luckily, the inner problem can be completely replaced by KKT condition according to [163]. Then the bi-level problem can be reformulated as a single level QP problem with non-linear constraints [164]:

$$\max_{\boldsymbol{u}_L} J_L(\boldsymbol{u}_L, \boldsymbol{u}_F)$$

s.t.

$$\boldsymbol{G}(\boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0 \tag{3.33}$$

$$\frac{\partial J_F(\boldsymbol{u}_L, \boldsymbol{u}_F)}{\partial \boldsymbol{u}_F} + \frac{\partial \boldsymbol{\lambda}^\intercal \boldsymbol{G}}{\partial \boldsymbol{u}_F} = 0$$

$$\lambda_j G_j = 0 \tag{3.34}$$

$$\lambda_j \geq 0$$

where

$$\boldsymbol{G} = G_L \cup G_F = \begin{bmatrix} G_1 \\ G_2 \\ \vdots \\ G_m \end{bmatrix}, \ \boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{bmatrix} \tag{3.35}$$

and λ_i is the Lagrange multiplier corresponding to the ith inequity constraint in $\boldsymbol{G}$.

With the existence of the complimentary condition (3.34) the single-level QP problem is no more convex in spite of a convex cost function and linearity of most of its constraints. And such non-convexity is the main challenge of solving this problem.

3.4.3 Branch and Bound Algorithm

Enumeration is one of the most prime way to find optimum. However, the branch and bound algorithm provides us a clever way of enumerating all possible cases of an optimization problem. The idea of branch and bound is to eliminate unnecessary cases by adding constraints to the prob-

Algorithm 1 The branch and bound algorithm for bi-level QP problem.

Standing at the initial node $n0$. Define $S_0 = G$, $S_+ = \emptyset$, $S_- = \emptyset$, $\underline{J} = inf$
Do the following iteratively
Try to solve the problem without (3.33), but add the following constraints: (a) Set $\lambda_i = 0$ for G_i in S_0 and S_-. (b) Set $G_i = 0$ for G_i in S_+. Get solution $\boldsymbol{u}$ and cost J
if the problem is feasible **then**
    if $J < \underline{J}$ **then**
        if $G(u) \leq 0$ and $\lambda G(u) = 0$ **then**
            Let $\underline{J} = J$ and mark u as the solution. Close the current node.
        else if $S_0 = \emptyset$ **then**
            Close the current node.
        else if $G_i \leq 0$ is satisfied for all $G_i \in S_0$ **then**
            Close the current node.
        else
            Find the the biggest G_i in S_0. Generate two new nodes under the current node. In one node, move G_i to S_+. In the other, move G_i to S_-. close the current node.
        end if
    else
        J is not smaller than the previous smallest value, close the current node.
    end if
else
    Problem is not feasible, close the current node.
end if
After closing a node, try to find a new active node. If no active node is left, terminate the algorithm.

lem one by one. If the problem is infeasible with constraints specified by a part of the conditions, then the problem will not be feasible with additional constraints either. Following this idea, we are able to truncate the search tree and only explore the nodes on necessary branches in an efficient way. Each nodes in the search tree stands for a specific case. The branch and bound algorithm for solving the bi-level QP problem was raised by Bard and Moore [150]. Following the idea of the original branch and bound, the paper provides ample details of managing constraint sets of the problem in an efficient way. The algorithm is given as Algorithm 1.

In Algorithm 1, S_0 is a set that includes all the constraints that has not been considered, S_+ is a set that includes all the constraints that is activated, S_- is a set that includes all the constraints that are not activated, $\underline{J}$ is the lowest cost the algorithm has reached so far.

Line 1 is the initialization of the algorithm. Line 1 tries to solve the problem at a certain node we have reached specified by the elements in S_0 S_+ and S_- of that node. Line 4 to 5 are the two necessary conditions, which are feasibility of the problem, better cost of regarding the current solution as the optimal one for the current branch. If Line 4 or 5 is not satisfied, then we close the current node and the current branch as Line 16 or 19. Otherwise there could be four possibilities as well. Line 6 checks the complete constraints and complimentary condition satisfaction of the original problem, if the solution satisfies this condition then it can be considered a the optimal solution for the current branch. Else we use Line 8 to check if S_0 is an empty set. If so, then all the constraints have been considered and the problem remains infeasible, so we close the current node. Line 10 checks if any constraints in S_+ or S_- is preventing the solution from satisfying the original constraints. If so, then the current branch is not worth further exploration. Line 13 is the branch operation. By selecting a constraint G_i that is violating the original constraints most as the next constraint we consider, we create two branches under the current node. For one node we assume G_i is active and for the other, we assume it is not. Line 21 checks if there is any new node that has not been visited. If not, we terminate the algorithm.

Although the algorithm is intelligent enough to avoid unnecessary computation, computation remains a problem due to the large numbers of constraints for branching in the differential lane change problem. In this research, three tricks are used to speed up the computation of the branch and bound algorithm so that it can be used for real-time implementation. First, I ignored the jerk constraints of the MPC comparing to (2.10) in the GT4SMPC. Second, I associate some of the constraints so that the algorithm does not have to check all the cases. For example, (3.11) is related with (3.13) in a way such that if $h = 0$ is active, then (3.13) is inactive, and vice versa. Such rule can be applied to mode 2 as well. Third, Matlab parallel computation box is used to compute the Stacelberg equilibrium for different N' simultaneously.

3.5 Inverse Model Predictive Control

Inverse optimization is a category of problem that we infer the parameters of the optimization problem assuming we already know the optimal solution to the problem. In this section we study

the inverse model predictive control (IMPC) problem so that we endow our controller the capability of evaluating other driver's drive style, which would greatly help DGTMPC make pertinent decision against SRVs from a game theoretic perspective.

Existing literatures show that not many researchers pay attention to the adaption of the game parameters, when they apply game theory to autonomous driving. A common way to determine the value of the parameters is from a empiric perspective [42, 132, 138]. This means the player model in the game is completely fixed. We all know that drivers are different. This means they could make diverse decision even in the exact same scenario. Therefore a fixed model is groundless. Another category of approaches obtain the parameters through off-line model calibration or learning using data from either public data set or experiments [93, 165, 140]. This approach of calibrating or learning a model seems solid, it still gives a fixed model in the end. Such calibrated model might work for macroscopic problems, it is not likely to work on individual drivers due to the same drawback of the first approach. Therefore, our research looks into an IMPC algorithm that could distinguish each surrounding driver's driving preference in a dynamic traffic.

Researches have studied different types of inverse optimization that are related to IQP/IMPC problem. Some researchers use the inverse linear quadratic regulator (ILQR) to recover the control technique of a human subject. Popular approaches to solve the ILQR problem involve linear matrix inequities approach[166, 167] and particle swarm optimization [168]. Borrowing the idea of ILQR, Ramadan et al. solves an IMPC problem by setting up a convex optimization problem and adding the algebraic Riccatti equation as one of the equality constraint. However, his approach does not consider complex constraints and thus cannot be applied to our problem. Invesr reinforcement learning (IRL) is recently a hot topic due to its value in robotic control. Popular IRL algorithms include apprentice learning [169] and entropy-based approach [170, 171].

The IMPC problem for our research is defined as the following optimization problem.

$$\boldsymbol{w}^* = \text{argmin} \sum_{k=1}^{N} |\boldsymbol{u}_F(k) - \boldsymbol{u}_F^*(k)| \tag{3.36}$$

s.t.

$$\min \widetilde{J}_F(\boldsymbol{u}_L, \boldsymbol{u}_F) \tag{3.37}$$

s.t.

$$G_F(\boldsymbol{u}_L, \boldsymbol{u}_F) \leq 0 \tag{3.38}$$

where

$$\boldsymbol{w} = [\widetilde{Q}_{vF}, \widetilde{R}_{uF}, \widetilde{R}_{hF}] \tag{3.39}$$

$$\begin{aligned} \widetilde{J}_F &= \boldsymbol{w} \sum_{k=1}^{N} (\boldsymbol{u}_F - \boldsymbol{u_r})^2 \\ &= \sum_{k=1}^{N} (\widetilde{Q}_{vF} |v_F(k) - v_{des}|^2 + \widetilde{R}_{uF} |u_F(k)|^2 + \widetilde{R}_{hF} |h_F(k)|^2) \end{aligned} \tag{3.40}$$

$$\boldsymbol{u_r} = [v_{des}, 0, 0] \tag{3.41}$$

and $\boldsymbol{w}^*$ is the optimal $\boldsymbol{w}$.

IMPC or inverse QP (IQP) is generally considered as a hard job due to the existence of complimentary slackness. Specifically, if any constraint of the MPC problem is active, then the relationship between the optimal solution that we already know and the weights we are trying to estimate is not a one to one mapping according to KKT condition. This can be proved by trying to solve the IMPC problem with KKT condition. Remember that the 1st order necessary condition of KKT condition indicates that the gradient of the Lagrange function is zero at the optimal point:

$$0 = \nabla L(x^*, \lambda^*) = \nabla f(x^*) - \sum \lambda_i^* G_i(x^*) \tag{3.42}$$

If at least one of the inequality constraint is active, then we would never have enough inequities

to solve for any other unknown parameters in f in (3.42) because we always need to solve for the extra Lagrange multiplier associated with that constraint.

Liwei Zhang and his group might be the only researchers who work on constrained IQP/IMPC. He solve ths problem by turning the IQP into a convex programming problem, where a reference weight matrix is designated to the QP problem [172, 173]. However, such reference is randomly generated in the paper and the author did not provide any detail for it.

3.5.1 From Inverse Reinforcement Learning to IMPC

In this section, we will show how we derive our IMPC algorithm from apprentice learning algorihtm of IRL in [169].

First we define value function as

$$V_{\pi,R}(s_t) = E[\sum_{t=0}^{\infty} \alpha^{\intercal} R(s_t)|\pi]$$

where γ is the forgetting rate, R is the reward function, t is time, s_t is the state of the system at time, π is the policy.

An optimal policy of an reinforcement learning (RL) problem should satisfy

$$V_{\pi^*,R^*}(s_t) \leq V_{\pi,R^*}(s_t) \tag{3.43}$$

where π^* is the optimal policy and R^* is the original reward function (to the corresponding IRL problem). Equation (3.43) is straightforward because the best policy should result in the best expected accumulated sum of the rewards.

We do the following to $V_{\pi,R}(s_t)$:

$$\begin{aligned} V_{\pi^*,R^*}(s_t) &= E[\sum_{t=0}^{\infty} \alpha^\intercal R(s_t)|\pi] \\ &= E[\sum_{t=0}^{\infty} \alpha^\intercal \omega^\intercal \phi(s_t)|\pi] \\ &= \omega^\intercal E[\sum_{t=0}^{\infty} \alpha^\intercal \phi(s_t)|\pi] \\ &= \omega^\intercal \mu(\pi) \end{aligned} \tag{3.44}$$

where ϕ is a feature vector of R and μ is the expected cumulative discounted sum of feature values. Equation (3.44) rewrites $V_{\pi,R}(s_t)$ as the weighted sum of μ by replacing R with a weighted sum of feature vectors ϕ.

Then the IRL problem can be defined as finding an ω^* such that

$$\omega^{*\intercal}\mu(\pi) \leq \omega^\intercal \mu(\pi), \quad \forall \pi \tag{3.45}$$

Abbeel and Ng raised a solution called apprenticeship learning for the above IRL problem, as is shown in Algorihtm 2. The idea of the algorithm is to find some ω that maximizes the margin between the optimal μ and the μ we got in the last iteration so that the gap between the computed μ and $\mu^{(i)}$ decreases with the number of iteration.

However, two challenges prevent us from the direct application of such an IRL algorithm to the IMPC/IQP problem. First, the complimentary slackness prevents us from accurately estimate the weights in the cost function if constraint is active. This can be seen from the analysis we just made on KKT condition or the following example: the optimum of both problems is $x = 1$ despite

Algorithm 2 Apprentice learning for inverse reinforcement learning.

1: Initialize with certain π_0
2: Iterate 3 to for i=1,2,...
3: Find a reward function such that the expert/teacher maximally outperforms all previously found policy π:

$$\begin{array}{ll} \max & \gamma \\ \text{s.t.} & w^T\mu(\pi^*) \geq w^T\mu^{(i')}(\pi) + \gamma, i' = 1, 2, ...i-1 \\ & ||w||_2 \leq 1 \end{array}$$

4: Find optimal policy π_i for the current guess of the reward function If γ is very small, exit the algorithm.

the different weights in the cost functions.

$$\begin{array}{lll} \text{Problem 1:} & \min_{x\in\mathbb{R}} & (x+1)^2 \\ & \text{s.t.} & x \geq 1 \\ \text{Problem 2:} & \min_{x\in\mathbb{R}} & (x+2)^2 \\ & \text{s.t.} & x \geq 1 \end{array}$$

and the second challenge is that we are not able to observe the whole output sequence over the prediction horizon of an MPC, but only the first step of the sequence.

For the first challenge we filter the observed data before we apply it to the inverse process so that we can minimize the influence of the complimentary slackness. For the second challenge we do the following: we observe for N time steps and consider all the outputs in the filled rectangles of Fig. 3.7 as the estimated MPC output sequence at $t = 0$. The filled rectangles in the figure means the observation, or the first output of a TV's MPC output sequence. However, this method of resolving the second challenge could bring up another issue because it does not give us the accurate solution of the MPC but just estimation. If we directly apply the observation and to a similar process to the apprentice learning, the algorithm will become over-constraining and thus infeasible due to the uncertainty of such a replacement. Being a variant of apprentice learning that overcomes the above two challenges, the IMPC algorithm is proposed as Algorithm 3 and 4.

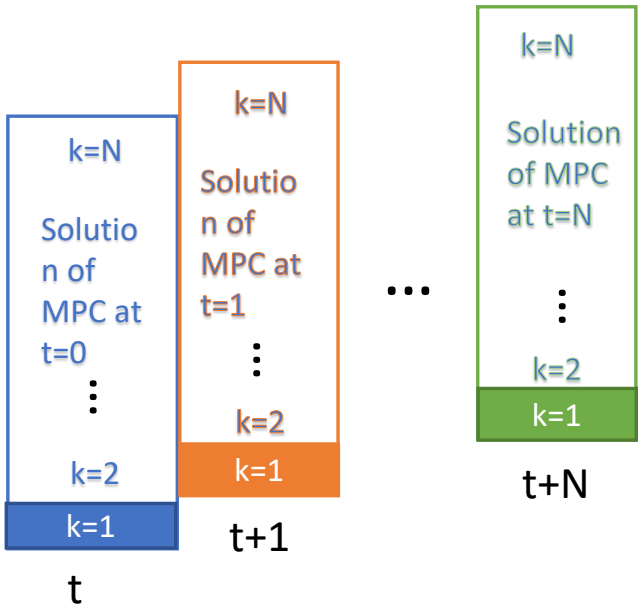

Figure 3.7: TV's MPC solution at each time step.

The IMPC Algorithm (Algorithm 3) works with the simulation/real time and explains how we gather and filter observed data, and how we maintain the constraint pool that is used for Algorithm 4. The max margin algorithm (Algorithm 4) iterates the margin maximization process to finds the weights that maximizes the sum of the margins between the observation and the estimated results of all the inequalities in the constraint pool until convergence and returns the weights to Algorithm 3.

In Algorithm 3, Line 1 initialize the algorithm. Line 2 and line 3 observe the environment and get the data needed for the whole prediction horizon of the MPC. To overcome the first challenge, line 4 checks the activity of the constraints to make sure that the observation contains some information for the accurate estimation of the weights. The less the number of the active constraints the more the current observation is contributing to accurate estimation. Line 5 solves the MPC problem using the traffic information $t - N$ ago as the initial condition and estimated weights as the weights of the MPC cost function. Then the solution is used to compare with the observation and get a new inequality for the constraint pool according to 6 and 7. Line 3 executes the max-margin

Algorithm 3 The IMPC algorithm

1: Initialize $\boldsymbol{w}$ with $\boldsymbol{w}^{(0)} = \epsilon$. let t=1, j=1
2: At time t, observe the target (target vehicle in our case) and get $\boldsymbol{u}^*(t)$
3: If we have observed for at least N steps, go to 4. Else t=t+1 and go to 2
4: Test $\boldsymbol{u}^*(t-N)$, $\boldsymbol{u}^*(t-N+1)$, ..., $\boldsymbol{u}^*(t)$ with the constraints. If all the constraints are inactive for at least one step in the horizon, then go to 5. Else, $t = t+1$ and go to 2.
5: Using the traffic information at $t-N$ as the initial condition and the current $\boldsymbol{w}^{(i)}$, we solve the MPC problem and get $\boldsymbol{u}_j$.
6: Compare the output of 5 and the observation by substituting them into the cost function. Then we get a new inequality $\boldsymbol{w}^T\boldsymbol{\mu}_j^* + \gamma_j \leq \boldsymbol{w}^T\boldsymbol{\mu}_j^{(i)}$, $j = j+1$
7: Add the inequality to the constraint pool.
8: Using the constraint pool, estimate w with the max-margin algorithm, then $t = t+1$, go back to 2

Algorithm 4 Max-margin algorithm

1: Initialize with $i = 1$
2: We solve the following problem to get the estimation of $\boldsymbol{w^{(i)}}$

$$\begin{aligned} \max_{\boldsymbol{w}^{(i)},\gamma_j} \quad & \textstyle\sum_{j=1}^{m} \gamma_j \\ \text{s.t.} \quad & \boldsymbol{w}^T\boldsymbol{\mu}_j^* + \gamma_j \leq \boldsymbol{w}^T\boldsymbol{\mu}_j^{(i')}, \text{ for } i' = 1,2,...,i \text{ and } j = 1,2,...,m \\ & ||\boldsymbol{w^{(i)}}||_2 = 1 \\ & \boldsymbol{w^{(i)}} \geq 0 \end{aligned}$$

3: If $\sum_{j=1}^{m} \gamma_j/m \leq \epsilon$, return $\boldsymbol{w^{(i)}}$ and go back to the high level of the algorithm
4: Else use $\boldsymbol{w^{(i)}}$ to solve the MPC problem, get $\boldsymbol{u}_j^{(i)}$, and compute $\boldsymbol{\mu}_j^{(i)}$ with the following equation for every j. Let $i = i+1$, then go to 2 .

$$\boldsymbol{\mu}_j^{(i)} = \sum_{k=t-N}^{t} (\boldsymbol{u}_j^{(i)}(k) - \boldsymbol{u}_r)^2$$

algorithm.

In Algorithm 4, line 2 solves the maximization problem for the current iteration and gets the estimated value of the weights. Unlike apprentice learning (Algorithm 2), we assign a margin variable λ_j to each of the inequality of the constraint pool instead of using one margin variable. By doing so, the infeasibility issue of the second challenge is addressed. The equality constrain $||w||_2 = 1$ actually plays the exact same role as the inequality constraint $||w||_2 \leq 1$ in Algorithm 2 due to the fact that the equal sign is always active in a maximization problem. Line 3 is the convergence condition for the algorithm. Line 4 solves the MPC using the current weights and gets $\boldsymbol{\mu}_j^{(i)}$ needed for constructing the constraints.

3.5.2 A Conditional Proof of Convergence of Max-Margin Algorithm

A conditional proof of the convergence of max-margin algorithm is given in this section. We assume MPC is unconstrained and we can observe the whole control sequence of the MPC. The proof includes three steps.

1. Prove that $(\sum_j \gamma_j^{(i+1)})/(\sum_j \gamma_j^{(i)}) < 1$, which indicates the convergence of the optimization process.

2. Prove the computed output $\boldsymbol{\mu}^{(i)}$ in the algorithm is now close to the observation $\boldsymbol{\mu}^*$.

3. We prove that if the weights we found, $\boldsymbol{w}^{(i)}$, give us the solution $\boldsymbol{\mu}^{(i)}$ very close to the observation $\boldsymbol{\mu}^*$ for multiple time steps $(k, k+1, k+2, ...)$, then we know the weights we estimated are close to the real one.

Step 1: First we define

$$\boldsymbol{\mu}_j^* = \sum_{k=t-N}^{t} (\boldsymbol{u}_j^*(k) - \boldsymbol{u}_r)^2$$
$$\boldsymbol{\mu}_j^{(i)} = \sum_{k=t-N}^{t} (\boldsymbol{u}_j^{(i)}(k) - \boldsymbol{u}_r)^2$$

Then the actual MPC cost function that we are trying to estimate can be rewritten as

$$J = \boldsymbol{w}\boldsymbol{\mu}_j^*$$

We assume we have observed the target MPC for $N + m - 1$ steps. Then we get totally m inequalities, which gives

$$\boldsymbol{w}^T\boldsymbol{\mu}_j^* + \gamma_j \leq \boldsymbol{w}^T\boldsymbol{\mu}_j^{(i)}, \text{ for } i = 1, 2, 3... \text{ and } j = 1, 2, 3...$$

We sum up $j = 1, 2, 3....$ and get

$$\boldsymbol{w}^T \sum_{j=1}^{m} \boldsymbol{\mu}_j^* + \sum_{j=1}^{m} \gamma_j \leq \boldsymbol{w}^T \sum_{j=1}^{m} \boldsymbol{\mu}_j^{(i)}$$

In [169], the authors already proved that for a specific θ^*, $\tau_j^{(i+1)}/\tau_j^{(i)}$ converges at certain rate between $(0, 1)$ for the max-margin algorithm, with which our IMPC algorithm share the exactly principle.

$$\tau^{(i+1)}/\tau^{(i)} \leq \frac{1}{1 + \frac{|\theta^*|^2}{|\theta_(i+1) - \theta^*|^2}} \tag{3.46}$$

where all the variables can be found in the similar maximization problem

$$\begin{aligned} \max \quad & \tau \\ \text{s.t.} \quad & \omega^T \theta^* \leq \omega \theta^{(i)} + \tau \\ & ||\omega||_2 \leq 1 \end{aligned}$$

We simply do the following substitution to (1).

$$\begin{aligned} \tau &= -\sum_{j=1}^{m} \gamma_j \\ \theta^* &= -\sum_{j=1}^{m} \boldsymbol{\mu}_j^* \\ \theta^{(i)} &= -\sum_{j=1}^{m} \boldsymbol{\mu}_j^{(i)} \end{aligned}$$

This gives

$$\sum_{j=1}^{m} \gamma_j^{(i+1)} / \sum_{j=1}^{m} \gamma_j^{(i)} \leq \frac{1}{1 + \frac{|\sum_{j=1}^{m} \boldsymbol{\mu}_j^*|^2}{|\sum_{j=1}^{m} \boldsymbol{\mu}_j^{(i)} - \sum_{j=1}^{m} \boldsymbol{\mu}_j^*|^2}}$$

Step 2: We already know that $\sum_{j=1}^{m} \gamma_j^{(i)}$ approaches zero as i grows and each γ is greater than 0, this means every γ_j is actually approaching 0. In other words,

$$\boldsymbol{w}^T (\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_j^*) = |\boldsymbol{w}||\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_j^*| \cos(\alpha) \leq \epsilon$$

where α is the angle between $\boldsymbol{w}$ and $\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_j^*$.

Since $\boldsymbol{w}$ is selected such that γ_j is maximum, α should not be close to π. And we also know that the norm of $|\boldsymbol{w}| = 1$, which means $|\boldsymbol{\mu}_j^* - \boldsymbol{\mu}_j^{(i)}|$ approaches 0 as i grows, or

$$\boldsymbol{\mu}_j^{(i)} - \boldsymbol{\mu}_j^* \leq \epsilon$$

Step 3: Remember that the cost function of MPC can be written as

$$J = \boldsymbol{w}\mu$$

And we also know that optimal $\boldsymbol{\mu}_j^*$ should satisfy the following due to its optimum nature.

$$\boldsymbol{w}^T(\boldsymbol{u}_j^* - \boldsymbol{u}_r) = 0$$

Thus given the fact that computed output $\boldsymbol{\mu}_j^{(i)}$ using the estimated weight is close to the $\boldsymbol{\mu}_j^*$, we are able to write the following equation set

$$\begin{aligned} \boldsymbol{w}^T\boldsymbol{u}_1^{(i)} &= \boldsymbol{u}_r \pm \sqrt{\epsilon} \\ &\dots \\ \boldsymbol{w}^T\boldsymbol{u}_j^{(i)} &= \boldsymbol{u}_r \pm \sqrt{\epsilon} \end{aligned}$$

As long as we have enough number of equations in the equation set that are linear independent, we are able to determine unique weight $\boldsymbol{w}$. In our case, the observation of the target MPC almost always give linear-independent equations. Then we can say the step 8 of the IMPC algorithm converges to a unique weight.

3.6 Simulations

In this section, we thoroughly validate the DGTMPC controller in four subsections. In the first part, we show the performance of the IMPC algorithm. In the second and third part, we analyze DGTMPC's performance in guarding mode and overtaking mode repectively. In the last part, we compare our controller's performance with the level-k game theoretic controller [42] in various aspects.

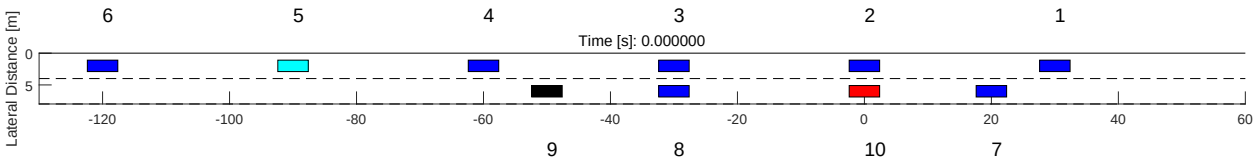

Figure 3.8: Initial places of the vehicles in scenario I.

3.6.1 Validating IMPC's Performacne

To validate IMPC's estimation accuracy, we would first need to set up a suitable simulation environment (Scenario I), where IMPC is used to estimate the weights of a traffic vehicle's MPC cost function assuming that vehicle is using MPC to control itself. In a two-lane highway driving scenario as Fig. 3.8. The vehicles are numbered as the figure, where No. 10 (red) is the SV controlled by DGTMPC, No.5 is an SRV controlled by HMPC defined as (3.9), all the other vehicles are controlled by IDM. One important thing that we do here is that we put a hostile vehicle No.9 (black) in the traffic. This vehicle's only mission is to make a sudden lane change in front of No. 5 to interrupt her steady state so that SV can gather enough information for IMPC process. If a TV is always in steady state, then obviously SV would never be able to estimate TV's MPC weights.

In the simulation, the initial places of all the vehicles are already shown in Fig. 3.8. Vehicles in the upper lane have an initial speed of 20m/s and vehicles in the lower lane have an initial speed of 15m/s. All the vehicles have the same desired speed as their initial speed except SV and the hostile vehicle, who desire for a speed of 20m/s. In this section, we will not show how SV behaves in such a scenario but just focus on how well SV can estimate the MPC weights of TV (No.5). Remember that No. 7 will do a sudden lane change to overtake No. 5, so the actual trajectory for each vehicle during the estimation process can be shown in Fig. 3.9 and 3.10.

The estimation result of the IMPC algorithm in this scenario is given as Table 3.1. The left three columns show the exact value of the weights in No.5's MPC cost function. The right three columns are the final estimation result of those weights. We can see that the estimation result is indeed close to the actual value despite some errors. The estimation result can give us a rough idea on the driving style of a driver and this information would be useful for the game formulation. In

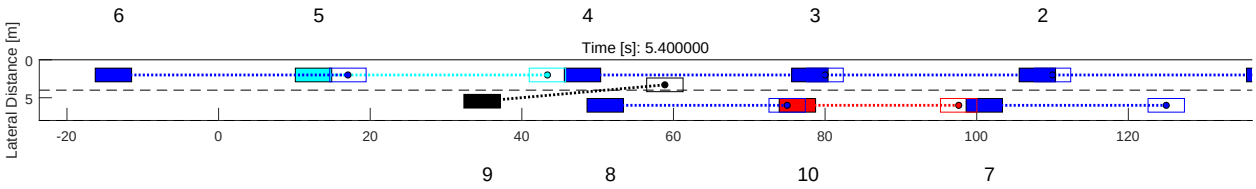

Figure 3.9: No.7 made a sudden lane change.

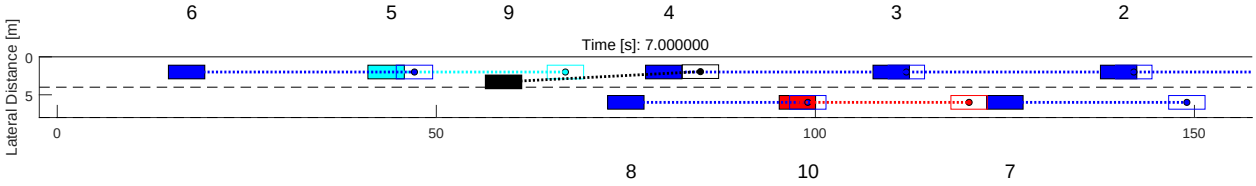

Figure 3.10: No. 5 start to decelerate after No.7's sudden lane switch.

addition to Table 3.1, we also include Fig. 3.11 and 3.12 here to show how the weights converges to a certain value as simulation time goes. The actual values of the estimated weights are given in the captions of each figure. We can identify the following facts form the figures. First, the algorithm gives its first estimation 4s after 4s simulation starts because SV need to observe the TV for the whole prediction horizon to get the complete output sequence of MPC. Second, from t=4s to t=10s the algorithm was giving estimated values of the weights because the TV was adjusting its headway at the beginning so she was not in steady state. However, TV's behavior at that time period was not informative enough due to the fact that the initial condition was close to the steady state and the observation uncertainty mentioned in the second challenge. Therefore, the estimated

Table 3.1: Actual weights versus the weight estimation results of IMPC.

Q_v	R_u	R_h	$\tilde{Q}_v$	$\tilde{R}_u$	$\tilde{R}_h$
0.09	0.95	0.29	0.14	0.89	0.44
0.11	0.53	0.84	0.11	0.55	0.83
0.58	0.58	0.58	0.6	0.5	0.62
0.8	0.53	0.27	0.83	0.46	0.32
0.85	0.17	0.51	0.8	0.29	0.52

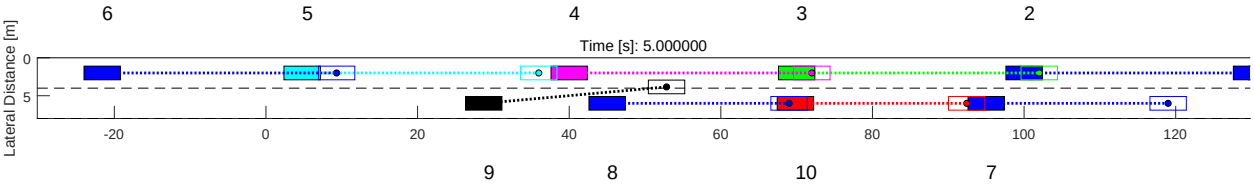

Figure 3.11: The estimation result of weights in No.5's MPC, where its original weights are Q_v=0.8, R_u=0.54 R_h=0.27.

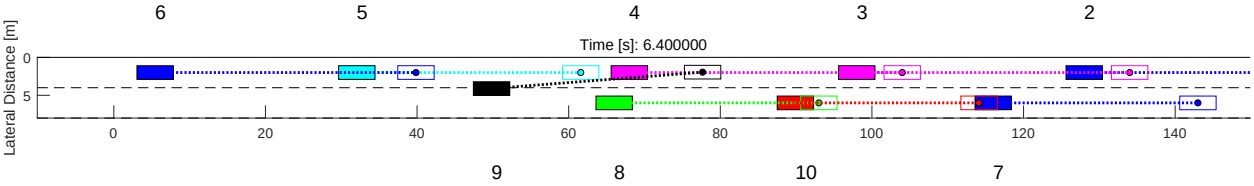

Figure 3.12: The estimation result of weights in No.5's MPC, where its original weights are Q_v=0.58, R_u=0.58 R_h=0.58.

value at that time was not accurate. However, after No.7 made the hostile lane change at about 6s, the process of TV adjusting its headway and returning to steady state provides ample information for good estimation of TV's MPC weights.

3.6.2 Simulation of DGTMPC in Overtaking Mode

In this section, we study DGTMPC's behavior against drivers of different driving style. Instead of using one scenario to test both modes of DGTMPC, we test them in different scenarios. The reason is that creating a scenario that is delicate enough to activate both modes of DGTMPC in a short simulation time requires well-designed manipulation of traffic vehicles, which is not the focus of our research and makes the simulation less straightforward to be understood.

To test overtaking mode of DGTMPC, we still use the same simulation environment as Section 3.6. Same as the previous section, all the vehicles are controlled by IDM except SV and No.5, which are controlled by DGTMPC, and HMPC defined as (3.9). The initial conditions and desired speed of all vehicles are exactly the same as those in the previous section. The coloring of each vehicle is still the same and additionally we mark the real-time TV as green vehicle in the simulation. We still put a hostile vehicle in the traffic to interrupt SV's steady state for weight estimation

purpose. Since the traffic in the upper lane is faster, this motivates the lane change intention of SV and force the interaction between SV and vehicles in the upper lane. We will show two typical cases in this scenario (Scenario I). In the first case, SV will interact with an aggressive driver and in the second SV will interact with a cautious one.

3.6.2.1 Overtaking Mode, Case 1

To define No.5 as an aggressive driver, we designate a weight vector (Q_v, R_u, R_h) of (0.2,0.98,0.1) to No.5's MPC cost function. Since this weight vector means that No.5 cares more about acceleration minimization and velocity maintenance than a safe headway, we think it is 'aggressive'.

The simulation result of case 1 is given in Fig. 3.13 to Fig. 3.17 , where Fig. 3.13 is the traffic when SV has the opportunity to interact with aggressive No.5, Fig. 3.15 is SV's action at every moment, Fig. 3.14 is both vehicle's velocity profile in the simulation, Fig. 3.16 is the weight estimation process of No.5. Fig. 3.17 is SV's GCV scope at every moment of the simulation.

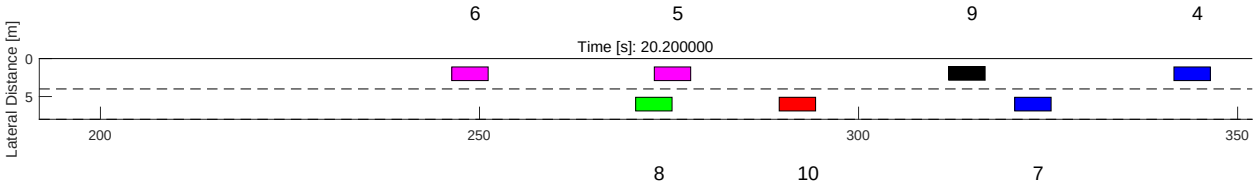

Figure 3.13: When SV has the opportunity to interact with No.5, he choose to stay in the current lane instead of doing lane change.

The simulation result in this case is quite straight forward. At about t=6s, the hostile vehicle (No.9) made a sudden lane change and forced No.5 to slow down (Fig. 3.14). While No. 5 had to adjust its headway and speed to regain its steady state, SV was able to observe No.5's behavior and estimate her MPC weights (Q_v, R_u, R_h) as (0.67, 0.65, 0.35) during the process. During the whole simulation, SV was looking at the vehicles in the upper lane one by one and had planed several changes according to (Fig. 3.15). However, due to the fact that all other vehicles follow IDM and do not respond to SV's lane change intention. SV was not able to accomplish any lane change

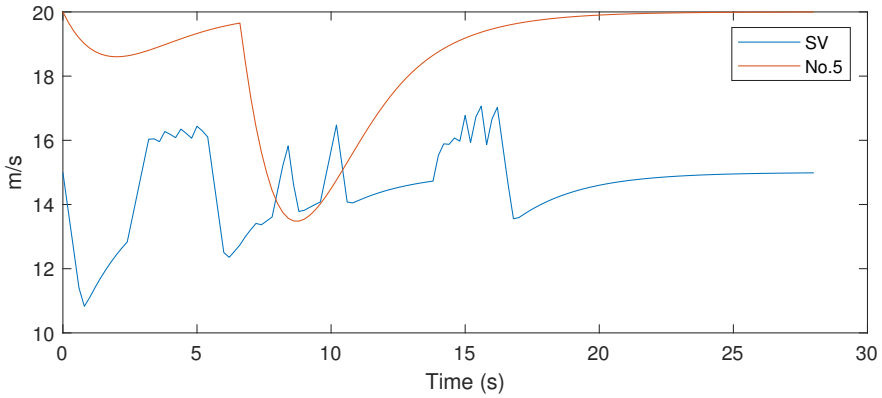

Figure 3.14: SV and No.5 velocity profile.

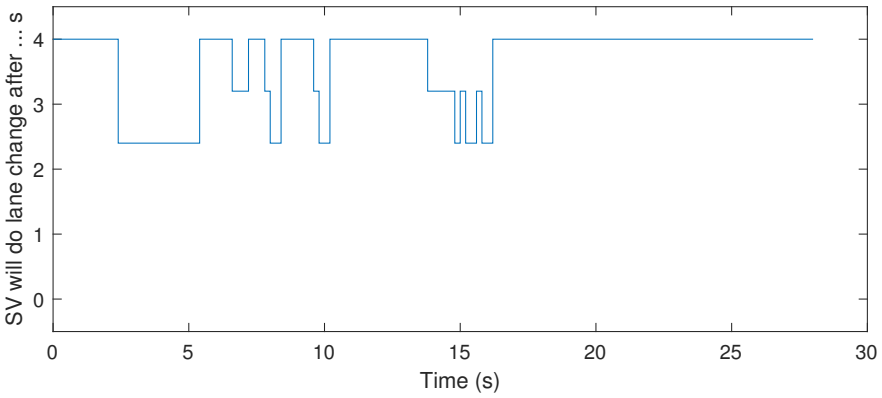

Figure 3.15: SV's lane change intention. 4s means SV does not want to change lane at all.

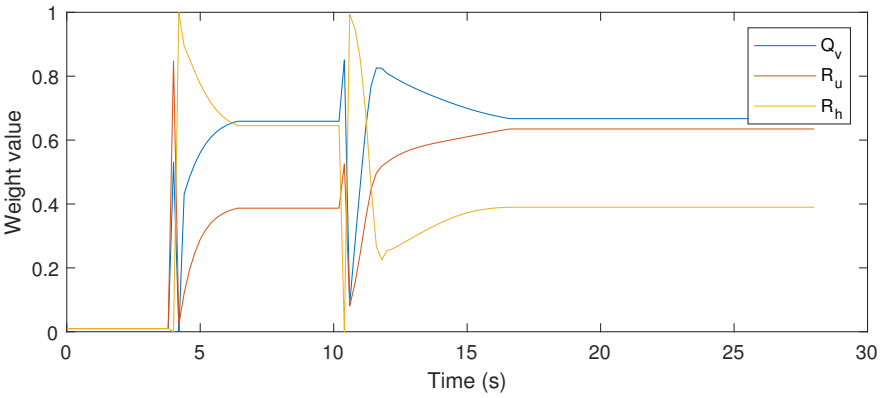

Figure 3.16: Weight estimation of No.5's MPC cost function.

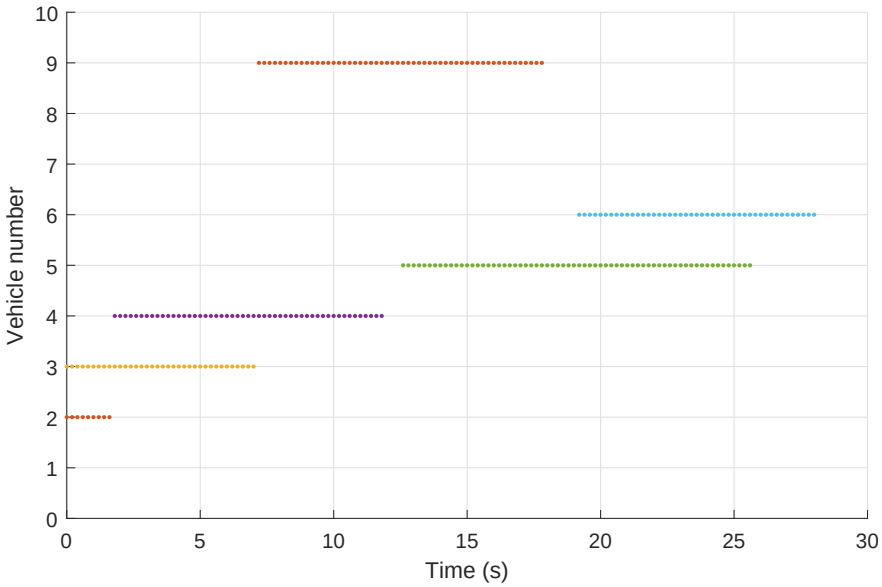

Figure 3.17: The GCV scope of DGTMPC.

before targeting No.5. The interaction process from t=12.5s to t=26s between SV and No.5 is a Stackelberg differential game. After SV realized No.5's high aggressiveness, SV did not attempt any lane change at all (3.15). From the above analysis we can see that although the game theory formulation did not help SV accomplish the lane change in this scenario, SV did play the right strategy in the situation versus aggressive driver (No.5) at the beginning.

3.6.2.2 Overtaking Mode, Case 2

In this case, we define No.5 as a cautious driver and designate a weight vector (Q_v, R_u, R_h) of (0.19,0.19,0.96) to No.5's MPC cost function. Same as case 1, the simulation results are given from Fig. 3.18 to Fig. 3.23.

Same as the previous simulations, from t=6s to t=15s , the steady state of No.5 was interrupted by No.7 and SV was able to estimate No.5's weights as is shown in Fig.3.22. The process of checking the gaps one by one was the same as case 1 until t=15s when SV was able to interact with No.7 according to Fig. 3.23. The IMPC results of $(\widetilde{Q}_v, \widetilde{R}_u, \widetilde{R}_h) = (0.17, 0, 0.97)$ indicated that No.5 is a cautious driver so the by taking into this information into account, the DGTMPC was planning a lane change starting from t=17s and SV finally merged into the upper lane traffic.

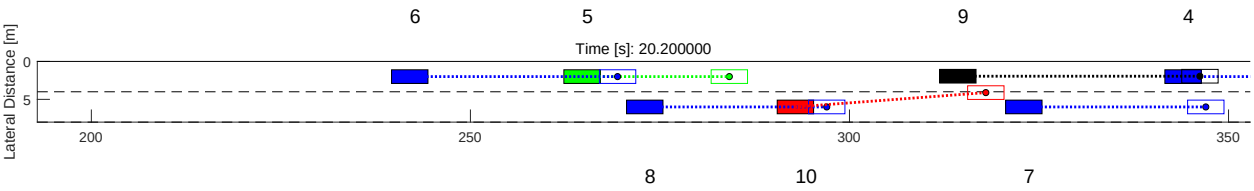

Figure 3.18: When SV has the opportunity to interact with No.5, he choose to turn on the turning light. And No.5 responds to SV's intention by opening a large gap. Then SV initiates a lane change.

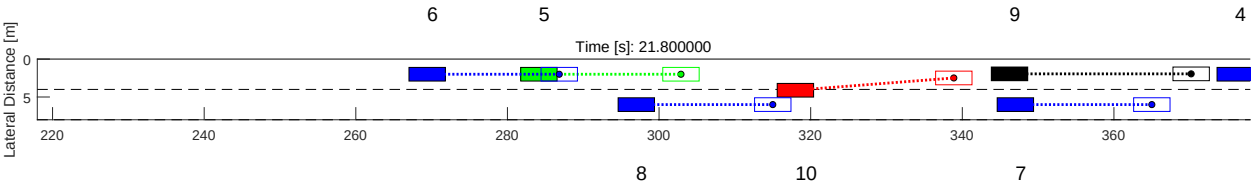

Figure 3.19: SV completes the lane change and merge into the traffic in the upper lane.

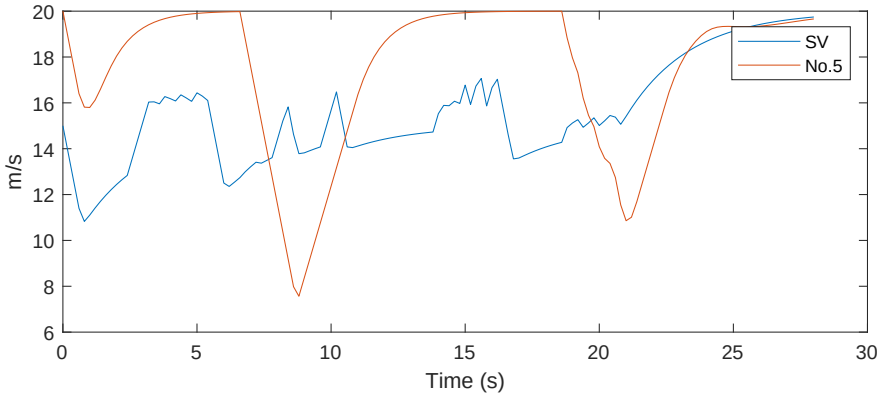

Figure 3.20: SV and Ncyan vehicle velocity profile.

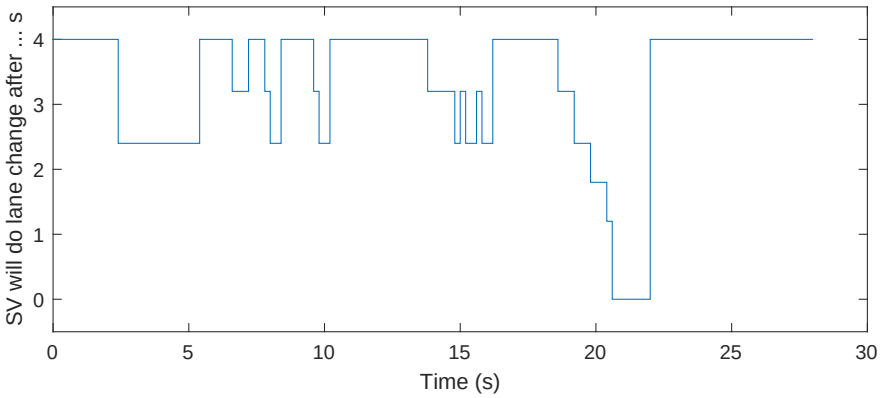

Figure 3.21: SV's lane change intention. 4s means SV does not want to change lane at all.

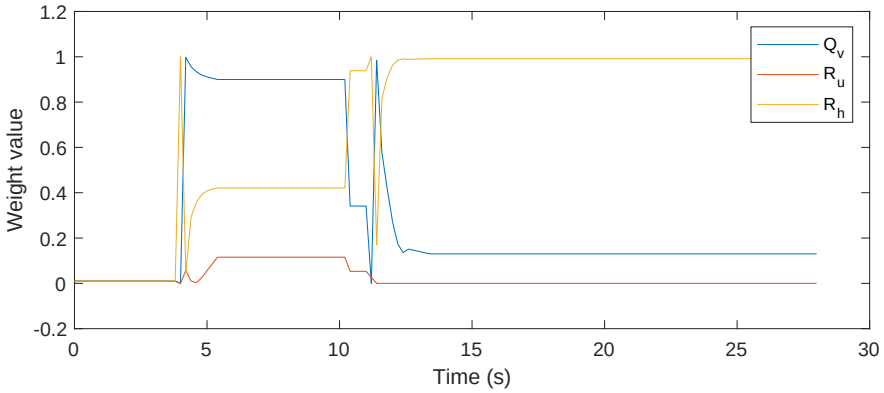

Figure 3.22: Weight estimation of cyan vehicle's MPC cost function.

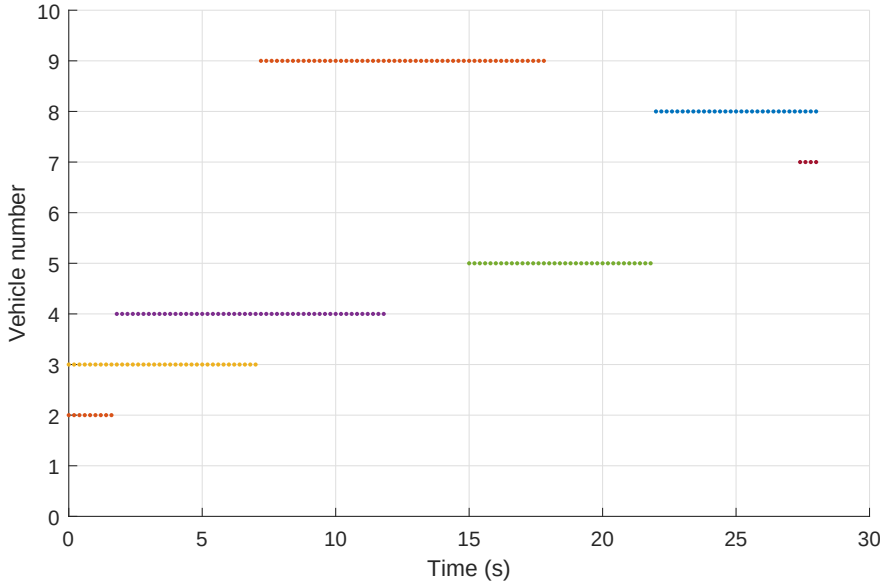

Figure 3.23: The GCV scope of DGTMPC.

The estimation result of IMPC in this case does not seem satisfactory. Comparing the estimation value (0.17, 0, 0.97) with the exact value (0.19,0.19,0.96) we can see that the IMPC algorithm was not able to estimate the weight on acceleration. Here is our explanation. Remember that the first challenge of the IMPC is that if the constraint is active we will never be able to estimate the exact value of the weights. If we look at the acceleration profile of No.5, we can see that during the observation period (t=6s to t=15s) the lower bound of the acceleration ($-6m/s^2$) has been active for a while, which did 'contaminate' the information SV gathered during that period.

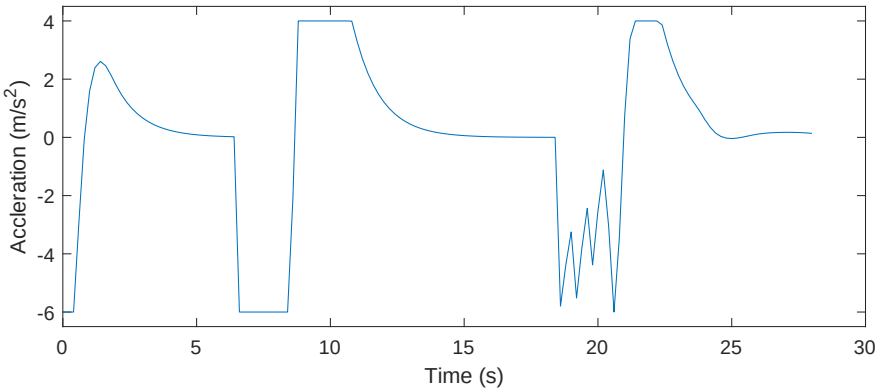

Figure 3.24: The acceleration of No.5 in case 2 .

3.6.3 Simulation of DGTMPC in Guarding Mode

To test the controller's performance in guarding mode, we need to set up another scenario as Fig. 3.25. The red vehicle is SV and the cyan vehicle is a traffic vehicle that is going to overtake SV. There is no hostile vehicle in the simulation. SV is controlled by DGTMPC, cyan vehicle is controlled by a HMPC defined as (3.17). All the other vehicles are controlled by IDM. The initial speed of vehicles in lower lane is 10m/s, that of vehicles in middle lane is 15m/s and that of vehicles in upper lane is 20m/s. All the vehicles have the same desired speed as their initial speed except the cyan vehicle. This motivates the cyan vehicle to enter the upper lane. In this scenario, we did not use hostile vehicle because we provide SV an opportunity to identify cyan vehicle's driving style when the latter enters the middle lane from the bottom lane. To simplify

things, we force cyan vehicle to overtake No.6 when she is in the lower lane and force her to overtake SV when she is in the middle lane, which means the only task of cyan vehicle's HMPC is to overtake a certain vehicle if that gives cyan more benefits. Same as above, we verify DGTMPC's performance against an aggressive driver ($(Q_v, R_u, R_h) = (0.99, 0.13, 0.13)$) and a cautious driver ($(Q_v, R_u, R_h) = (0.13, 0.13, 0.99)$) respectively.

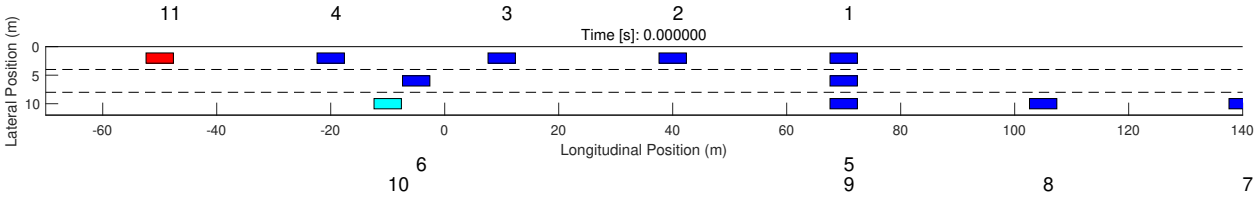

Figure 3.25: Vehicle initial positions in case 1.

3.6.3.1 Guarding Mode, Case 1

case 1 The initial positions of the vehicles are exactly the same as Fig. 3.25. The simulation results of case 1 are given in Fig. 3.26 to Fig. 3.30. Fig. 3.26 and Fig. 3.27 show the first and second lane change of cyan vehicle in the simulation respectively. Fig. 3.28 show both vehicles' velocity. Fig. 3.28 is cyan vehicle's lane change planning, which is a part of her strategy in the game. Fig. 3.30 is SV's estimation of cyan vehicle's MPC weights.

At the beginning, we put cyan in the lower lane where traffic the slow. Since cyan has a desired speed of 20m/s, she changed lane to the adjacent lane, where traffic is faster, at about t=7s. By observing cyan's lane change process, SV was able to identify SV's driving style as aggressive $(\widetilde{Q}_v, \widetilde{R}_u, \widetilde{R}_h) = (0.35, 0, 0.93)$ at t=12s according to Fig. 3.30. After entering the middle lane, cyan saw the even faster traffic in the upper lane and began to seeking opportunity to enter that lane. The fact that we set all the other vehicles as irresponsive to cyan forced the interaction between SV and cyan. When SV is close to the TV at t=13s, interaction begin. Having learned that cyan's high aggressiveness, SV choose to yield to cyan (Fig. 3.28) and gave her enough space to start the lane change at t=18s according to Fig. 3.27 and 3.29. Actually, according to the mechanism of the

Stackelberg differential game, SV knew that even if SV did not yield, cyan would still make an aggressive lane change, leading to a dangerous situation, which neither of the vehicles wanted.

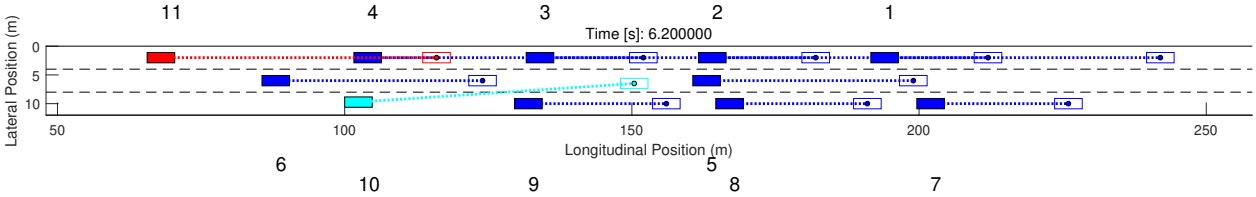

Figure 3.26: First lane change of cyan vehicle.

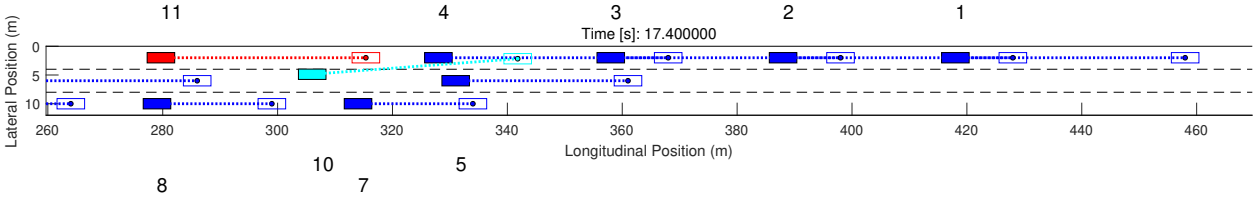

Figure 3.27: Second lane change of cyan vehicle.

In addition, looking at Fig. 3.30, we may see the estimation result did not fit the actual value that we give at the beginning of this section. This is mainly due to the short observation time of cyan's behavior during the lane change. However, this still helped SV make good judgment of cyan's driving preference as well as the right decision against the aggressive driver during the interaction.

3.6.3.2 Guarding Mode, Case 2

In case 2 , we use a slightly different initial position for all the vehicles as Fig. 3.31. The MPC weights of the cyan vehicle are set to $(Q_v, R_u, R_h) = (0.13, 0.13, 0.99)$. All the other settings are exactly the same as case 1. The reason for a different initialization is that the careful-selected initial positions motivates the cyan vehicle to switch lane from the lower one to the middle one, from which DGTMPC would be able to estimate the weights of cyan.

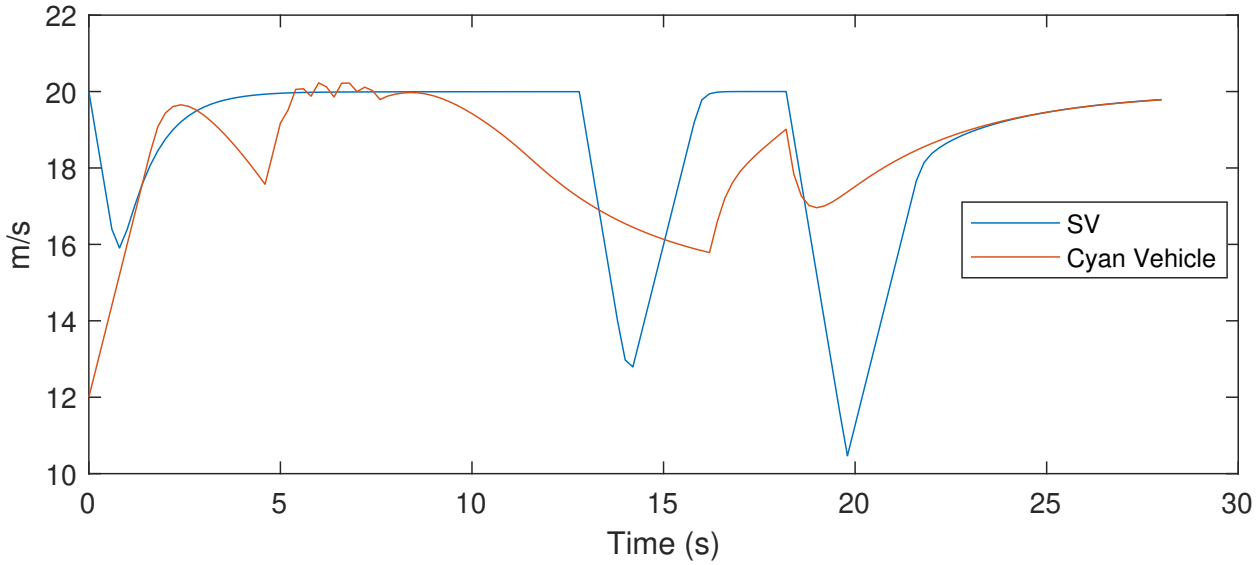

Figure 3.28: SV and cyan vehicle's velocity profile.

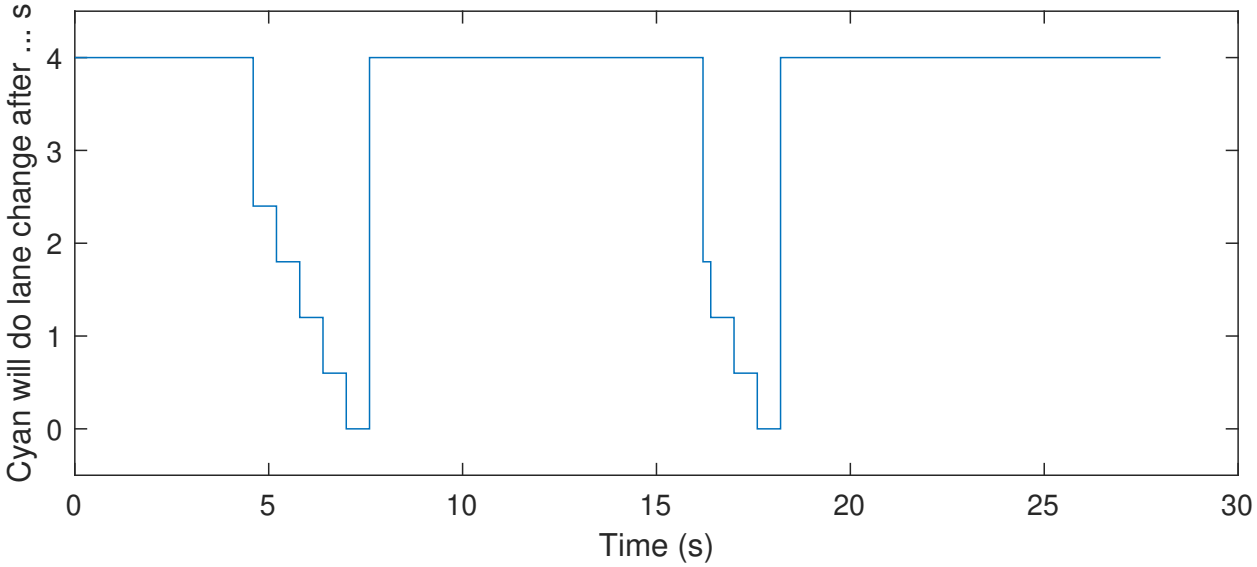

Figure 3.29: TV's lane change planning in the simulation.

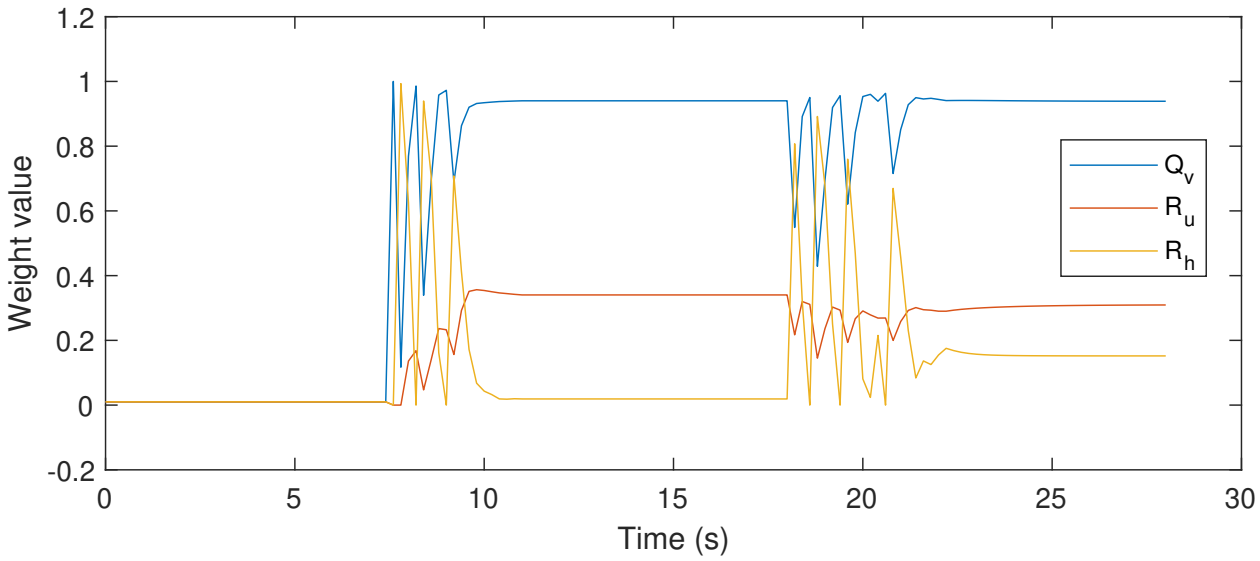

Figure 3.30: Estimation result of cyan vehicle's MPC weight

The simulation results are given from Fig. 3.32 to Fig. 3.37. Fig. 3.32 depicts the first lane change of cyan from lower lane to middle one. Fig. 3.33 shows the interaction process between SV and cyan. In Fig. 3.33 cyan finally merged into the upper lane by tugging behind SV. Fig. 3.35 are two players' velocity. Fig. 3.36 shows TV's lane change planning.

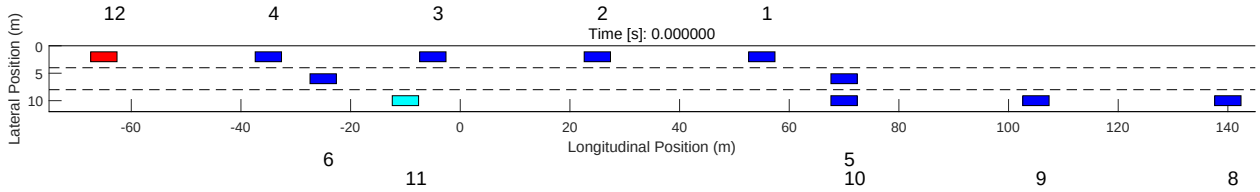

Figure 3.31: Vehicle initial positions in case 2

.

Same as previous scenarios, cyan accomplished a lane change at the beginning of the simulation for pursuit of higher speed as Fig. 3.32. SV was able to well identify TV's driving style as cautious after observing the whole process of cyan's lane change: the IMPC algorihtm gave an estimation of $(\widetilde{Q}_v, \widetilde{R}_u, \widetilde{R}_h) = (0.05, 0.1, 0.98)$. After cyan successfully merged into the middle lane traffic,

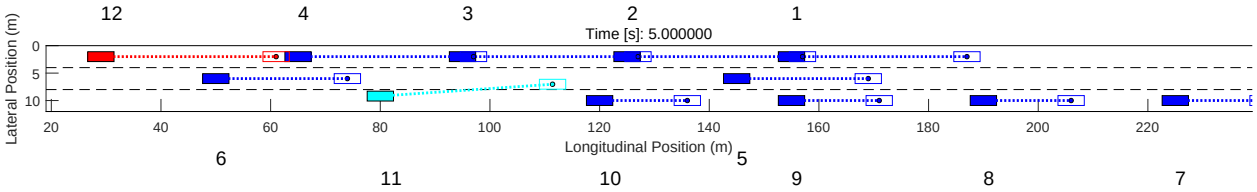

Figure 3.32: Cyan vehicle made the first lane change

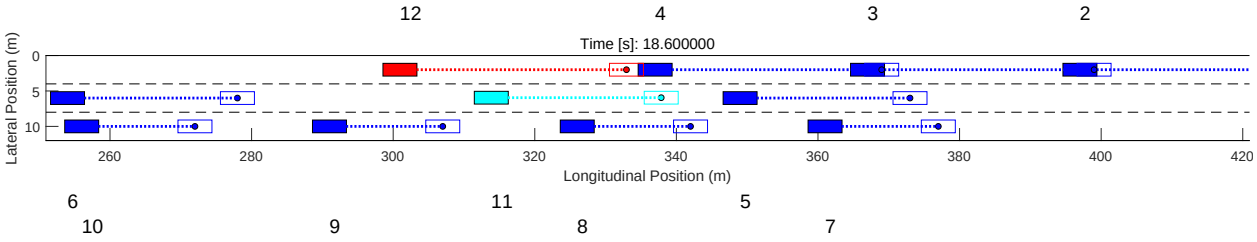

Figure 3.33: Cyan vehicle interacts with SV and choose not to overtake SV

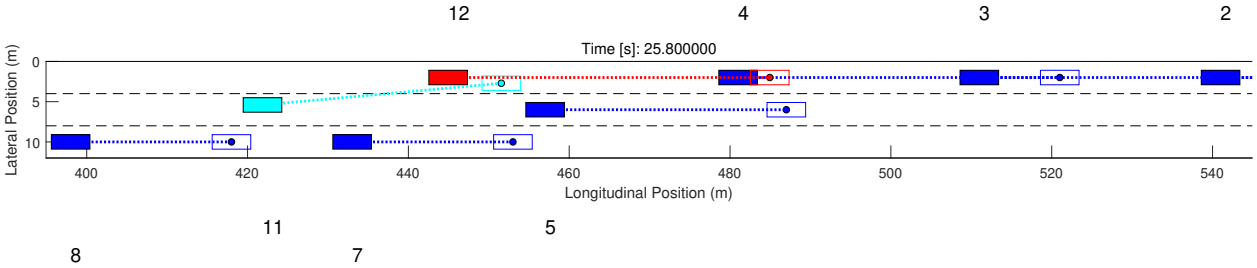

Figure 3.34: Cyan vehicle changed lane after SV.

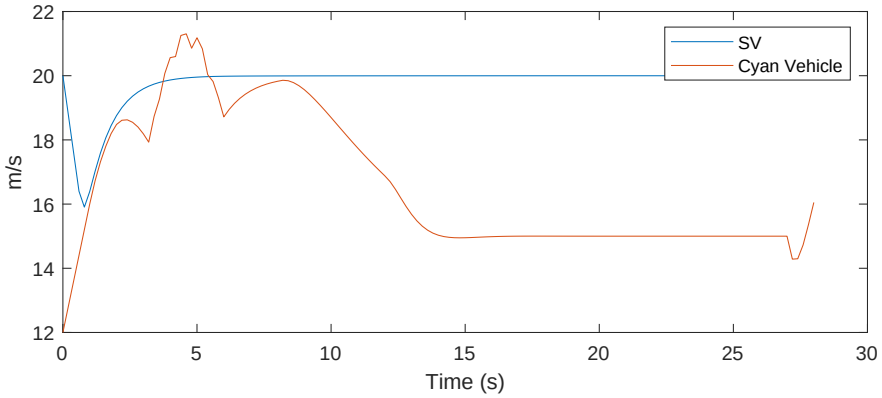

Figure 3.35: Two vehicles' velocity in the simulation.

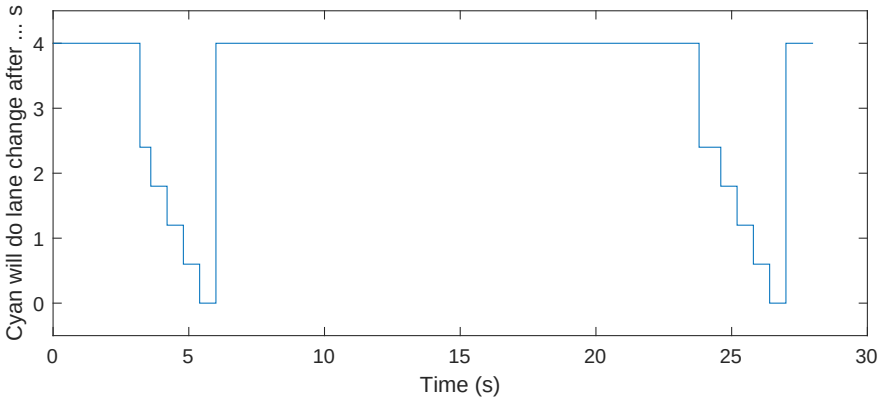

Figure 3.36: TV's lane change planning in the simulation.

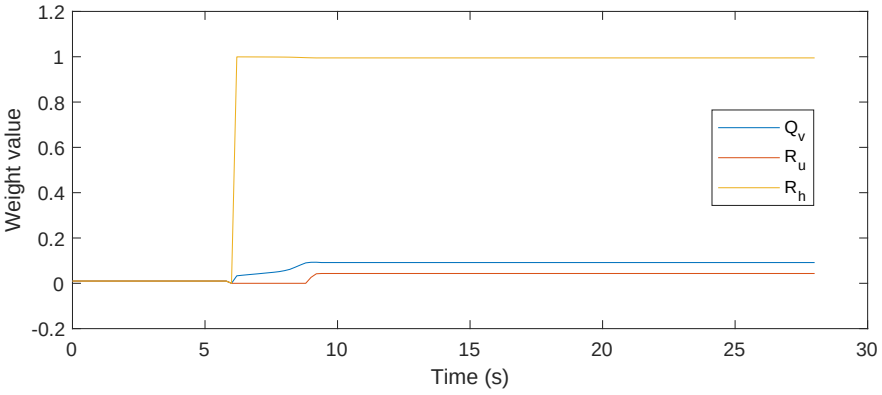

Figure 3.37: Estimation result of cyan vehicle's MPC weight

her interaction with SV was forced by setting all the other vehicles to a fixed IDM model that does not respond to cyan's behavior in adjacent lane. Having learned cyan's low aggressiveness, he chose not to yield because the low aggressiveness meant cyan did not dare to overtake SV at all, if SV did not show any intention of yielding. Then cyan finished her lane change by letting every vehicle in the upper lane pass by. Similar to the IMPC result in case 1, IMPC did not give a result accurate enough in case 2 due to the same reason. However, the result still gave SV some idea of SV's aggressiveness, which made SV determine to secure his position advantage in the simulation facing cyan.

3.6.4 Comparing DGTMPC with Level-k Controller

Comparison helps us understand how good a new idea is comparing to existing ones, so we provide a complete comparison between the DGTMPC and the level-k game theoretic controller of [42] in this section.

The following reasons are given for choosing the level-k controller as the reference. First, same as DGTMPC, the level-k model is among the few AV controllers that are based on non-cooperative game. Second, the Stackeberg game of DGTMPC is actually a special case of level-k thinking, on which the level-k controller is based. Third, both controllers are built in non-connected situation. Fourth, although DGTMPC is an based on MPC and level-k is based on reinforcement learning, such comparison could actually help us identify the feature of each approach, and thus reveals of the fundamental principle of each approaches. More details of the level-k controller and Jaakkola reinforcement learning can be found in [42] and [174].

In this comparison, we did not completely replicate the original level-k set-up but made some modification accordingly so as to achieve a fair comparison. We define the following observation space for the level-k controller:

1. Longitudinal distance between the Left lane front car and SV, quantified as "close" (distance $\leq$ 21 m), "nominal" (21 m < distance $\leq$ 42 m) or "far" (distance > 42 m);

2. Longitudinal distance between the Left lane rear car and SV quantified as "close", "nominal"

or “far”;

3. Longitudinal distance between current lane front car and SV quantified as “close”, “nominal” or “far”;

4. Longitudinal distance between right lane front car and SV quantified as “close”, “nominal” or “far”;

5. Longitudinal distance between right lane rear car and SV quantified as “close”, “nominal” or “far”;

6. Relative speed between left lane front car, quantified as “faster,” (relative speed $\leq 2m/s^2$), “stable” ($-2m/s^2 \leq$ relative speed $\leq 2m/s^2$) or “slower” (relative speed $\geq 2m/s^2$)

7. Relative speed between left lane rear car relative speed, quantified as “faster,”, “stable” or “slower”

8. Relative speed between current lane front car relative speed, quantified as “faster,”, “stable” or “slower”

9. Relative speed between right lane front relative speed, quantified as “faster,”, “stable” or “slower”

10. Relative speed between right lane rear relative speed, quantified as “faster,”, “stable” or “slower”

11. SV lane

The action space of the controller is refined comparing to the original paper as different longitudinal acceleration $\{0, \pm 1, \pm 2, \pm 3, \pm 4\}$ and lane change to left lane or right lane. Such refinement gives the level-k controller a wider range of actions to choose from comparing to the original setup. Instead of the original linear formulation, the reward function is a mimic of the MPC cost function

as follows:

$$R_{level-k} = -(Q_{vL}|v_L(k) - v_{des}|^2 + R_{uL}|u_L(k)|^2 + R_{hL}|h_L(k)|^2) - R_e e \tag{3.47}$$

where R_e is a huge positive number, e is 1 if constraint is violated, 0 otherwise. The constraint is exactly the same as the original paper. We rewrite the reward function because we think the comparison would be more meaningful if the two approaches have similar criterion in decision-making. Same as [42], we use random traffics to train our level-k controller. The average reward of the learning process converges after about 50000 training episodes.

To achieve a fair comparison, we test DGTMPC and level-1 controller in the same random environments. We choose level-1 controller instead of level-0 or level-2 controller as our competitor because Stackelberg game is actually a typical example of level-1 thinking. The setup of the environment is a mimic of that in [42]. The environment is generated in a Monte Carlo way and all the surrounding vehicles are level-0 vehicles. The level-0 controller is a rule based controller and is given as 3.48. We use the following rules to generate the initial traffic. We put 20 cars randomly in a three lane highway. The initial longitudinal positions of the cars follow a uniform distribution on [-100m, 200m]. Their initial longitudinal velocity also follow the uniform distribution on [16m/s, 25m/s]. The initial lane of each car is randomly assigned. The desired speed of both DGTMPC and level-k controller are set at 20.5m/s.

$$u_{level-0} = \begin{cases} -2m/s^2, & \text{if the car in front is nominal and approaching or is close and stable} \\ \Delta x(k)/\Delta x_{lim}, & \text{if the car in front is close and approaching} \\ -4m/s^2, & \text{otherwise.} \end{cases} \tag{3.48}$$

Totally 100 different scenarios are generated for both controllers. We run the simulation of each scenario for 30 seconds then look at the number of collisions and the average velocity for

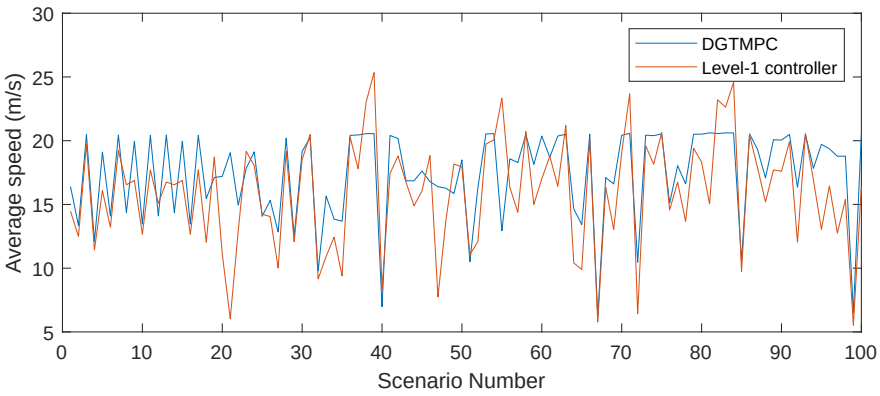

Figure 3.38: The average speed in each scenario of each controller.

both controllers as criterion. Both controller achieved zero collision in the 100 simulations. The average of the average speed of DGTMPC in the 100 scenarios is 15.9m/s while that of level-1 controller is 17.4m/s. The specific average speed of each scenario given in Fig. 3.38. The results show that both controller is safe enough in terms of collision avoidance, but DGTMPC can achieve comparatively higher travel efficiency comparing to level-1 controller.

We provide the following explanation for the performance gap. To generate the policy of level-1 controller or any other RL-based controller, we need to first discretize the observation/state space. Such discretization process brings uncertainty into the problem, because after the discretization there could be infinite number of scenarios that maps to the same message. Therefore the level-1 controller has to be conservative enough to guarantee no collision for any specific scenario that maps to the same message. And that is the reason why the level-1 controller has lower traveling efficiency than DGTMPC . The follow example supports the above statement. It belongs to one of the 100 scenario. Fig. 3.39 shows DGTMPC's decision under that scenario and Fig. 3.40 is that of level-1 controller. While DGTMPC dare to get through a small gap between a blue vehicle in the top lane and the vehicle in front of him, but the level-1 controller gave up that lane change opportunity and choose to tug behind the vehicle in the lower lane.

We summarize the features of two controllers here. The two controllers share the following common features.

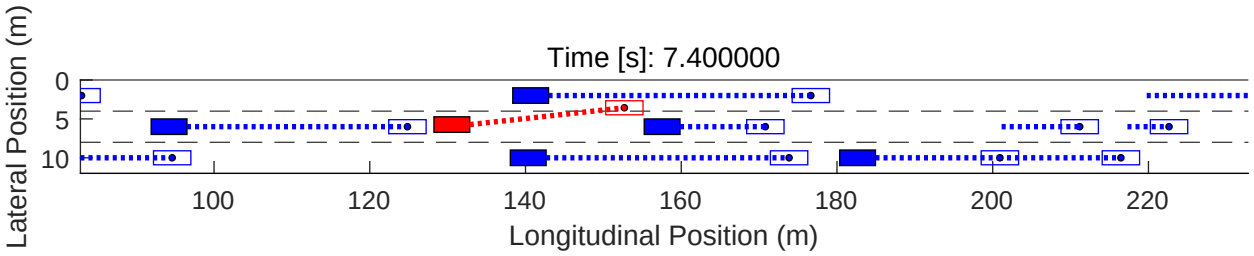

Figure 3.39: DGTMPC's decision under a specific scenario.

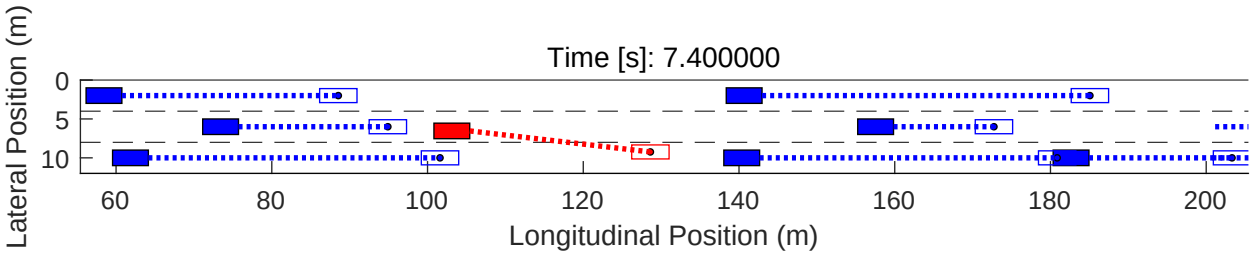

Figure 3.40: Level-1 controller's decision under a specific scenario.

- Both controllers are function-based approach. DGTMPC is minimizing the cost function and level-1 controller is maximizing the reward function.
- Both controllers are built on the leader-follower relationship assumption of game theory.

The DGTMPC owns the following unique features.

- States and actions are continuous, so the traveling efficiency of the controlled vehicle is higher.
- Parameters of DGTMPC can be easily adjusted on-line.
- The DGTMPC models the traffic as many one-leader-one-follower games
- IMPC algorithm identifies the driving style of traffic vehicle and the DGTMPC can take advantage of that information and design corresponding strategy.

The DGTMPC has the following drawbacks.

- The DGTMPC has to be solved on line.
- The cost functions of two players should be convex.

The level-1 controller owns unique features.

- The formulation of reward function is flexible, it does not need to be convex at all.
- As long as it is trained, it is very fast.
- The Jaakkola RL can be considered as an infinite horizon optimization problem.
- The level-1 controller models the traffic as a one-leader-n-follower game.

The level-1 controller has the following drawbacks.

- The controller needs to be robust enough to deal with the uncertainty occurred during the discretization process.
- If any setting of the controller is modified, it needs retraining.
- The training effort grows exponentially with number of states and number of actions.

3.7 Conclusion

In this chapter we propose a Stackelberg differential game based model predictive controller for autonomous driving. This piece of work mainly has two contributions, modeling the high way lane change scenario as two-player Stackelberg differential game and the inverse model predictive control algorithm.

In the Stackelberg differential game, both players actually MPC controllers so that they each can play an action at each step over the prediction horizon. The To solve the Stackelberg differential game, we propose the problem as a bi-level MPC problem. Then it is converted to a single-level problem and is solved with the branch and bound algorithm.

The IMPC algorithm is used to estimate the MPC weights of on-road vehicles on line. The observability of the full solution sequence of MPC and the active constraints are two challenges for

IMPC. The first challenge is overcome by replacing the solution sequence with estimated observation over a period. The second challenge is overcome by assigning one margin variable to each constraints of the apprentice learning problem instead of one. Simulation shows that the algorithm can well estimate the unknown weights of the MPC of traffic vehicles.

The whole controller is validated in two ways. First it is tested in a discretionary lane change scenario against a cautious driver and an aggressive driver. Simulation results showed that DGTMPC can make corresponding strategy based on the observed driving style of surrounding vehicles. Second DGTMPC is compared with level-1 controller in random generated scenarios. Comparison showed that the DGTMPC can achieve higher average traveling speed while guarantee zero collisions in random traffics.

4. SUMMARY AND FUTURE WORK

In this study, two game theoretic model predictive controller were developed for autonomous highway driving.

In the first part, a game theoretic four-stage model predictive controller is proposed. The hierarchical controller includes a high-level controller and a lower level one. The high-level controller models the lane change scenario as a Stackelberg game between subject vehicle (SV) and a target vehicle. The solution of the game becomes the reference for the lower-level controller who controls SV both longitudinally and laterally. The controller was fully validated in simulation and human-in-the-loop simulation and showed decent performance against human drivers.

In the second part, a differential Stackelberg game controller was developed along with an inverse model predictive control algorithm. Rather than the Stackelberg table game in the first part, the differential game defines the action space of both players in a continuous way over the prediction horizon so that both players can play an action at each step of the duration of the game. A branch and bound algorithm is utilized to solve the Stackelberg differential lane change game. Besides, an inverse MPC (IMPC) algorithm is proposed to estimate driver's driving preference, which is used in the differential game to set up the MPC cost function of a target vehicle. Simulations showed that the controller can interact with programmed vehicles intelligently and comparison with a level-1 game theoretic controller indicated its high safety level and better performance.

In summary, two methods based on game theory and model predictive controller have been presented. The hierarchical nature of the controller gives the methods high flexibility. The high-level of the controller is basally a decision maker and the low-level controller achieves the goal of the high level and control the vehicle's specific motion. Therefore, either part can be replaced by an equivalent approach. In addition, the brand and bound algorithm and the inverse MPC algorihtm can be applied to any other engineering problem of the same type.

Following the direction of this research, many future works can be done. First, the inverse MPC is proved in a conditional case in the dissertation. A strict proof of its convergence can be further

explored. Second, in the simulations, we assume the vehicles have full knowledge of the current state of each vehicle in the traffic. The controllers' robustness can be further studied if the sensor information is imperfect. Third, one drawback of the Stackelberg game model is the Stackeberg assumption: the leader can manipulate the follower. In reality, the roles of each player in the game depends. A reasonable approach to determine each player's role in the game would be needed to justify the Stackelberg game model.

REFERENCES

[1] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray, "Autonomous driving in urban environments: approaches, lessons and challenges," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010.

[2] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, and V. Pratt, "Towards fully autonomous driving: Systems and algorithms," in *Intelligent Vehicles Symposium (IV)*, pp. 163–168, IEEE, 2011.

[3] J. E. Naranjo, C. Gonzalez, R. Garcia, and T. De Pedro, "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 9, no. 3, pp. 438–450, 2008.

[4] A. Furda and L. Vlacic, "Enabling safe autonomous driving in real-world city traffic using multiple criteria decision making," *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 1, pp. 4–17, 2011.

[5] D. C. Gazis, R. Herman, and R. W. Rothery, "Nonlinear follow-the-leader models of traffic flow," *Operations Research*, vol. 9, no. 4, pp. 545–567, 1961.

[6] P. G. Gipps, "A behavioural car-following model for computer simulation," *Transportation Research Part B: Methodological*, vol. 15, no. 2, pp. 105–111, 1981.

[7] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation," *Physical Review E*, vol. 51, no. 2, p. 1035, 1995.

[8] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

[9] K. I. Ahmed, *Modeling drivers' acceleration and lane changing behavior*. Thesis, Massachusetts Institute of Technology, 1999.

[10] Q. Yang and H. N. Koutsopoulos, "A microscopic traffic simulator for evaluation of dynamic traffic management systems," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 3, pp. 113–129, 1996.

[11] Y. Zhang, L. Owen, and J. Clark, "Multiregime approach for microscopic traffic simulation," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1644, pp. 103–114, 1998.

[12] C.-Y. Liang and H. Peng, "Optimal adaptive cruise control with guaranteed string stability," *Vehicle System Dynamics*, vol. 32, no. 4-5, pp. 313–330, 1999.

[13] R. Rajamani, *Vehicle dynamics and control*. Mechanical engineering series, New York : Springer, [2012] 2nd ed., 2012.

[14] S. Li, K. Li, R. Rajamani, and J. Wang, "Model predictive multi-objective vehicular adaptive cruise control," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 556–566, 2011.

[15] R. Möbus, M. Baotic, and M. Morari, "Multi-object adaptive cruise control," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 359–374, Springer, 2003.

[16] D. Corona and B. De Schutter, "Adaptive cruise control for a smart car: A comparison benchmark for mpc-pwa control methods," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 2, pp. 365–372, 2008.

[17] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura, "Cooperative adaptive cruise control in real traffic situations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 296–305, 2014.

[18] J. Ploeg, B. T. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 260–265, IEEE, 2011.

[19] L. Nouvelière and S. Mammar, "Experimental vehicle longitudinal control using a second order sliding mode technique," *Control Engineering Practice*, vol. 15, no. 8, pp. 943–954, 2007.

[20] K.-T. Chong and D.-H. Roh, "A lyapunov function approach to longitudinal control of vehicles in a platoon," *IEEE Transactions on vehicular technology*, vol. 50, no. 1, pp. 116–124, 2001.

[21] K. El Majdoub, F. Giri, H. Ouadi, L. Dugard, and F. Z. Chaoui, "Vehicle longitudinal motion modeling for nonlinear control," *Control Engineering Practice*, vol. 20, no. 1, pp. 69–81, 2012.

[22] P. G. Gipps, "A model for the structure of lane-changing decisions," *Transportation Research Part B-Methodological*, vol. 20, no. 5, pp. 403–414, 1986.

[23] P. Hidas, "Modelling lane changing and merging in microscopic traffic simulation," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 5, pp. 351–371, 2002.

[24] J. Erdmann, "Proceedings of the sumo2014 modeling mobility with open data," *Journal of Intelligent & Robotic Systems*, 2014.

[25] M. Ben-Akiva, D. Cuneo, M. Hasan, M. Jha, and Q. Yang, "Evaluation of freeway control using a microscopic simulation laboratory," *Transportation research Part C: emerging technologies*, vol. 11, no. 1, pp. 29–50, 2003.

[26] C. Roncoli, I. Papamichail, and M. Papageorgiou, "Model predictive control for multi-lane motorways in presence of vacs," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 501–507, IEEE, 2014.

[27] T. Toledo, H. Koutsopoulos, and M. Ben-Akiva, "Modeling integrated lane-changing behavior," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1857, pp. 30–38, 2003.

[28] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model mobil for car-following models," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1999, pp. 86–94, 2007.

[29] W. Cao, M. Mukai, and T. Kawabe, "Two-dimensional merging path generation using model predictive control," *Artificial Life and Robotics*, vol. 17, no. 3-4, pp. 350–356, 2013.

[30] B. Ran, S. Leight, and B. Chang, "A microscopic simulation model for merging control on a dedicated-lane automated highway system," *Transportation Research Part C: Emerging Technologies*, vol. 7, no. 6, pp. 369–388, 1999.

[31] B. Van Arem, C. J. Van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 429–436, 2006.

[32] Q. Xu, K. Hedrick, R. Sengupta, and J. VanderWerf, "Effects of vehicle-vehicle/roadside-vehicle communication on adaptive cruise controlled highway systems," in *IEEE 56thVehicular Technology Conference*, vol. 2, pp. 1249–1253, IEEE, 2002.

[33] G. Schildbach and F. Borrelli, "Scenario model predictive control for lane change assistance on highways," in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 611–616, IEEE, 2015.

[34] J. Villagra, B. d'Andrea Novel, H. Mounier, and M. Pengov, "Flatness-based vehicle steering control strategy with sdre feedback gains tuned via a sensitivity approach," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 554–565, 2007.

[35] D. J. Cole, A. J. Pick, and A. M. C. Odhams, "Predictive and linear quadratic methods for potential application to modelling driver steering control," *Vehicle System Dynamics*, vol. 44, no. 3, pp. 259–284, 2006.

[36] J. M. Snider *et al.*, "Automatic steering methods for autonomous automobile path tracking," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*, 2009.

[37] G. Bayar, "Long distance autonomous trajectory tracking for an orchard vehicle," *Industrial Robot: the international journal of robotics research and application*, vol. 40, no. 1, pp. 27–40, 2013.

[38] A. El Hajjaji and S. Bentalba, "Fuzzy path tracking control for automatic steering of vehicles," *Robotics and Autonomous systems*, vol. 43, no. 4, pp. 203–213, 2003.

[39] R. Bellman, "A markovian decision process," *Indiana University Mathematics Journal*, vol. 6, no. 4, p. 15, 1957.

[40] S. Brechtel, T. Gindele, and R. Dillmann, "Probabilistic mdp-behavior planning for cars," in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pp. 1537–1542, IEEE, 2011.

[41] S. Coskun and R. Langari, "Predictive fuzzy markov decision strategy for autonomous driving in highways," in *2th IEEE Conference on Control Technology and Application*, IEEE, 2018.

[42] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on control systems technology*, 2017.

[43] Y. Du, Y. Wang, and C.-Y. Chan, "Autonomous lane-change controller," in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 386–393, IEEE, 2015.

[44] M. Mukai and T. Kawabe, "Model predictive control for lane change decision assist system using hybrid system representation," in *2006 SICE-ICASE International Joint Conference*, pp. 5120–5125, IEEE, 2006.

[45] J. Nilsson and J. Sjöberg, "Strategic decision making for automated driving on two-lane, one way roads using model predictive control," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 1253–1258, IEEE, 2013.

[46] A. Stentz, "Optimal and efficient path planning for unknown and dynamic environments," *International Journal of Robotics & Automation*, vol. 10, no. 3, pp. 89–100, 1995.

[47] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *IEEE International Conference on Robotics and Automation*, pp. 3310–3317, IEEE, 1994.

[48] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[49] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–80, 1996.

[50] S. M. Lavalle, *Rapidly-Exploring Random Trees: A New Tool for Path Planning*. Citeseer, 1998.

[51] Y. Kuwata, G. A. Fiore, J. Teo, E. Frazzoli, and J. P. How, "Motion planning for urban driving using rrt," in *IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS)*, pp. 1681–1686, IEEE, 2008.

[52] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. P. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *Control Systems Technology, IEEE Transactions on*, vol. 17, no. 5, pp. 1105–1118, 2009.

[53] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[54] M. Zhu, M. Otte, P. Chaudhari, and E. Frazzoli, "Game theoretic controller synthesis for multi-robot motion planning part i: Trajectory based algorithms," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1646–1651, IEEE, 2014.

[55] M. Elbanhawi, M. Simic, and R. Jazar, "Receding horizon lateral vehicle control for pure pursuit path tracking," *Journal of Vibration and Control*, p. 1077546316646906, 2016.

[56] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[57] M. Elbanhawi, M. Simic, and R. N. Jazar, "Continuous path smoothing for car-like robots using b-spline curves," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 1, pp. 23–56, 2015.

[58] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[59] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 22, no. 2, pp. 224–241, 1992.

[60] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[61] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *IEEE International Conference on Robotics and Automation*, pp. 1986–1991, IEEE, 2007.

[62] A. Kushleyev and M. Likhachev, "Time-bounded lattice for efficient planning in dynamic environments," in *IEEE International Conference on Robotics and Automation*, pp. 1662–1668, IEEE, 2009.

[63] S. Pancanti, L. Pallottino, D. Salvadorini, and A. Bicchi, "Motion planning through symbols and lattices," in *IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3914–3919, IEEE, 2004.

[64] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics*, vol. 26, no. 3, pp. 308–333, 2009.

[65] M. Rufli and R. Siegwart, "On the design of deformable input-/state-lattice graphs," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3071–3077, IEEE, 2010.

[66] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 987–993, IEEE, 2010.

[67] M. McNaughton, *Parallel algorithms for real-time motion planning*. Carnegie Mellon University, 2011.

[68] S. Shalev-Shwartz, N. Ben-Zrihem, A. Cohen, and A. Shashua, "Long-term planning by short-term prediction," *arXiv preprint arXiv:1602.01580*, 2016.

[69] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, and J. Zhang, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[70] O. J. Chen and M. E. Ben-Akiva, "Game-theoretic formulations of interaction between dynamic traffic control and dynamic traffic assignment," *Transportation Research Record*, vol. 1617, pp. p. 179–188, 1998.

[71] Z.-l. Li and D.-W. Chen, "A stackelberg game approach to ramp metering and variable speed control," *The Proceedings of the 2003 IEEE Intemational Confuence on Intelligent Transportation Systems*, vol. 2, pp. 1061–1063, 2003.

[72] B. B. Su, H. Chang, Y. Z. Chen, and D. R. He, "A game theory model of urban public traffic networks," *Physica a-Statistical Mechanics and Its Applications*, vol. 379, no. 1, pp. 291–297, 2007.

[73] S. Kalam, M. Gani, and L. Seneviratne, "Fully non-cooperative optimal placement of mobile vehicles," *2008 IEEE International Conference on Control Applications, Vols 1 and 2*, pp. 685–690, 2008.

[74] E. Semsar and K. Khorasani, "Optimal control and game theoretic approaches to cooperative control of a team of multi-vehicle unmanned systems," *2007 IEEE International Conference on Networking, Sensing, and Control, Vols 1 and 2*, pp. 628–633, 2007.

[75] P. Yan, M. Y. Ding, and C. P. Zhou, "Game-theoretic route planning for team of uavs," *Proceedings of the 2004 International Conference on Machine Learning and Cybernetics, Vols 1-7*, pp. 723–728, 2004.

[76] H. Kita, K. Tanimoto, and K. Fukuyama, "A game theoretic analysis of merging-giveway interaction: A joint estimation model," *Transportation and Traffic Theory in the 21st Century*, pp. 503–518, 2002.

[77] P. Hidas, "Modelling lane changing and merging in microscopic traffic simulation," *Transportation Research Part C*, vol. 10, pp. 351–371, 2002.

[78] A. Talebpour, H. Mahmassani, and S. Hamdar, "Multiregime sequential risk-taking model of car-following behavior: Specification, calibration, and sensitivity analysis," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2260, pp. 60–66, 2011.

[79] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Modeling lane-changing behavior in a connected environment: A game theory approach," *Transportation Research Part C: Emerging Technologies*, vol. 59, pp. 216–232, 2015.

[80] E. Altendorf and F. Flemisch, "Prediction of driving behavior in cooperative guidance and control: a first game-theoretic approach," *CogSys, Madgeburg*, 2014.

[81] M. Wang, S. P. Hoogendoorn, W. Daamen, B. van Arem, and R. Happee, "Game theoretic approach for predictive lane-changing and car-following control," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 73–92, 2015.

[82] H. Qi, S. Ma, N. Jia, and G. Wang, "Experiments on individual strategy updating in iterated snowdrift game under random rematching," *Journal of theoretical biology*, vol. 368, pp. 1–12, 2015.

[83] R. Schönauer, M. Stubenschrott, W. Huang, C. Rudloff, and M. Fellendorf, "Modeling concepts for mixed traffic: Steps toward a microscopic simulation tool for shared space zones," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2316, pp. 114–121, 2012.

[84] K. Aghabayk, S. Moridpour, W. Young, M. Sarvi, and Y.-B. Wang, "Comparing heavy vehicle and passenger car lane-changing maneuvers on arterial roads and freeways," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2260, pp. 94–101, 2011.

[85] R. Elvik, "A review of game-theoretic models of road user behaviour," *Accident Analysis & Prevention*, vol. 62, pp. 388–396, 2014.

[86] F. Meng, J. Su, C. Liu, and W.-H. Chen, "Dynamic decision making in lane change: game theory with receding horizon," in *Control (CONTROL), 2016 UKACC 11th International Conference on*, pp. 1–6, IEEE, 2016.

[87] 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, pp. 305–312, 1999.

[88] H. X. Liu, W. Xin, Z. Adam, and X. Ban, "A game theoretical approach for modelling merging and yielding behaviour at freeway on-ramp sections," *Elsevier, London*, pp. 197–211, 2007.

[89] J. S. Peng, Y. S. Guo, and Y. M. Shao, "Lane change decision analysis based on drivers' perception-judgment and game theory," in *Applied Mechanics and Materials*, vol. 361, pp. 1875–1879, Trans Tech Publ, 2013.

[90] J. H. Yoo and R. Langari, "Development of a predictive collision avoidance for subjective adjacent risk estimation," in *American Control Conference (ACC)*, pp. 3334–3341, IEEE, 2015.

[91] P. Yulong and X. Huizhi, "The control mechanism of lane changing in jam condition," in *The Sixth World Congress on Intelligent Control and Automation*, vol. 2, pp. 8655–8658, 2006.

[92] J. Peng, R. Fu, Y. Guo, W. Yuan, and C. Wang, "Analysis of lane change decision making based on the finite and zero-sum grey game theory," *Keji Daobao/ Science & Technology Review*, vol. 29, no. 3, pp. 52–56, 2011.

[93] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Modeling lane-changing behavior in a connected environment: A game theory approach," *Transportation Research Procedia*, vol. 7, pp. 420–440, 2015.

[94] N. Li, D. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. Girard, "Game-theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *arXiv preprint arXiv:1608.08589*, 2016.

[95] J. H. Yoo and R. Langari, "Stackelberg game based model of highway driving," in *ASME 5th Annual Dynamic Systems and Control Conference joint with the JSME 11th Motion and Vibration Conference*, pp. 499–508, American Society of Mechanical Engineers, 2012.

[96] J. H. Yoo and R. Langari, "A stackelberg game theoretic driver model for merging," in *ASME Dynamic Systems and Control Conference*, pp. V002T30A003–V002T30A003, American Society of Mechanical Engineers, 2013.

[97] Y.-l. Zhang, "Discretionary lane changing modeling based on stackelberg game theory," *Journal of Transportation Systems Engineering and Information Technology*, vol. 5, p. 010, 2014.

[98] S. P. Hoogendoorn and P. Bovy, *Generic driving behavior modeling by differential game theory*, pp. 321–331. Springer, 2009.

[99] A. Aghajani and A. Doustmohammadi, "Formation control of multi-vehicle systems using cooperative game theory," in *15th International Conference on Control, Automation and Systems (ICCAS)*, pp. 704–709, IEEE, 2015.

[100] D. Gu, "A differential game approach to formation control," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 85–93, 2008.

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

[102] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in *ASME 2010 dynamic systems and control conference*, vol. 1, pp. 265–272, American Society of Mechanical Engineers, 2010.

[103] U. Rosolia, S. De Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Transactions on Control Systems Technology*, 2016.

[104] F. E. Sancar, B. Fidan, J. P. Huissoon, and S. L. Waslander, "Mpc based collaborative adaptive cruise control with rear end collision avoidance," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pp. 516–521, IEEE, 2014.

[105] X. Na and D. J. Cole, "Linear quadratic game and non-cooperative predictive methods for potential application to modelling driver–afs interactive steering control," *Vehicle System Dynamics*, vol. 51, no. 2, pp. 165–198, 2013.

[106] X. Na and D. J. Cole, "Game-theoretic modeling of the steering interaction between a human driver and a vehicle collision avoidance controller," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 1, pp. 25–38, 2015.

[107] J. Maestre, D. M. De La Pena, E. Camacho, and T. Alamo, "Distributed model predictive control based on agent negotiation," *Journal of Process Control*, vol. 21, no. 5, pp. 685–697, 2011.

[108] J. Maestre, D. Munoz De La Pena, and E. Camacho, "Distributed model predictive control based on a cooperative game," *Optimal Control Applications and Methods*, vol. 32, no. 2, pp. 153–176, 2011.

[109] L. Mozaffari, A. Mozaffari, and N. L. Azad, "Vehicle speed prediction via a sliding-window time series analysis and an evolutionary least learning machine: A case study on san fran-

cisco urban roads," *Engineering science and technology, an international journal*, vol. 18, no. 2, pp. 150–162, 2015.

[110] J. Park, D. Li, Y. L. Murphey, J. Kristinsson, R. McGee, M. Kuang, and T. Phillips, "Real time vehicle speed prediction using a neural network traffic model," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pp. 2991–2996, IEEE, 2011.

[111] L. Shen, *Freeway travel time estimation and prediction using dynamic neural networks*. Thesis, Florida International University, 2008.

[112] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[113] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *Journal of transportation engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[114] B. Jiang and Y. Fei, "Traffic and vehicle speed prediction with neural network and hidden markov model in vehicular networks," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pp. 1082–1087, IEEE, 2015.

[115] J. Jing, D. Filev, A. Kurt, E. Özatay, J. Michelini, and Ü. Özgüner, "Vehicle speed prediction using a cooperative method of fuzzy markov model and auto-regressive model," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pp. 881–886, IEEE, 2017.

[116] T. A. Dingus, S. G. Klauer, V. L. Neale, A. Petersen, S. E. Lee, J. Sudweeks, M. Perez, J. Hankey, D. Ramsey, S. Gupta, *et al.*, "The 100-car naturalistic driving study, phase ii-results of the 100-car field experiment," tech. rep., Virginia Tech Transportation Institute, 2006.

[117] D. Jovanović, K. Lipovac, P. Stanojević, and D. Stanojević, "The effects of personality traits on driving-related anger and aggressive behaviour in traffic among serbian drivers,"

Transportation research part F: traffic psychology and behaviour, vol. 14, no. 1, pp. 43–53, 2011.

[118] I. A. Kaysi and A. S. Abbany, "Modeling aggressive driver behavior at unsignalized intersections," *Accident Analysis & Prevention*, vol. 39, no. 4, pp. 671–678, 2007.

[119] F. Badin, F. Le Berr, H. Briki, J. Dabadie, M. Petit, S. Magand, and E. Condemine, "Evaluation of evs energy consumption influencing factors, driving conditions, auxiliaries use, driver's aggressiveness," in *Electric Vehicle Symposium and Exhibition (EVS27), 2013 World*, pp. 1–12, IEEE, 2013.

[120] J. L. Deffenbacher, E. R. GETTING, and R. S. Lynch, "Development of a driving anger scale," *Psychological reports*, vol. 74, no. 1, pp. 83–91, 1994.

[121] T. Lajunen and D. Parker, "Are aggressive people aggressive drivers? a study of the relationship between self-reported general aggressiveness, driver anger and aggressive driving," *Accident Analysis & Prevention*, vol. 33, no. 2, pp. 243–255, 2001.

[122] D. Shinar and R. Compton, "Aggressive driving: an observational study of driver, vehicle, and situational variables," *Accident Analysis & Prevention*, vol. 36, no. 3, pp. 429–437, 2004.

[123] W. Vanlaar, H. Simpson, D. Mayhew, and R. Robertson, "Aggressive driving: A survey of attitudes, opinions and behaviors," *Journal of Safety Research*, vol. 39, no. 4, pp. 375–381, 2008.

[124] B. Higgs, M. Abbas, and A. Medina, "Analysis of the wiedemann car following model over different speeds using naturalistic data," *Road Safety and Simulation, Indianapolis, Indiana*, 2011.

[125] H.-j. Cho and Y.-t. Wu, "A car-following model for intelligent transportation systems management," *WSEAS TRANSACTIONS on BUSINESS and ECONOMICS*, 2007.

[126] J. Von Neumann and O. Morgenstern, *Theory of games and economic behavior (commemorative edition)*. Princeton university press, 2007.

[127] J. F. Nash *et al.*, "Equilibrium points in n-person games," *Proceedings of the national academy of sciences*, vol. 36, no. 1, pp. 48–49, 1950.

[128] H. Von Stackelberg, *Market structure and equilibrium*. Springer Science & Business Media, 2010.

[129] H. J. Stetter, *Analysis of discretization methods for ordinary differential equations*, vol. 23. Springer, 1973.

[130] T. Basar and G. J. Olsder, *Dynamic noncooperative game theory*, vol. 23. Siam, 1999.

[131] G. Long, "Acceleration characteristics of starting vehicles," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1737, pp. 58–70, 2000.

[132] J. H. Yoo and R. Langari, "Stackelberg game based model of highway driving," in *ASME 2012 5th Annual Dynamic Systems and Control Conference joint with the JSME 2012 11th Motion and Vibration Conference*, pp. 499–508, American Society of Mechanical Engineers, 2012.

[133] Y. Rahmati and A. Talebpour, "Towards a collaborative connected, automated driving environment: A game theory based decision framework for unprotected left turn maneuvers," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1316–1321, IEEE, 2017.

[134] N. Mehr, R. Li, and R. Horowitz, "A game theoretic macroscopic model of bypassing at traffic diverges with applications to mixed autonomy networks," *arXiv preprint arXiv:1809.02762*, 2018.

[135] K. Kang and H. A. Rakha, "Modeling driver merging behavior: a repeated game theoretical approach," *Transportation research record*, vol. 2672, no. 20, pp. 144–153, 2018.

[136] "Dynamic systems, latest research and news|nature." `https://www.nature.com/subjects/dynamical-systems`. Accessed: 2019-05-12.

[137] R. Isaacs, *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*. Courier Corporation, 1999.

[138] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, 2019.

[139] G. Williams, B. Goldfain, P. Drews, J. M. Rehg, and E. A. Theodorou, "Autonomous racing with autorally vehicles and differential games," *arXiv preprint arXiv:1707.04540*, 2017.

[140] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions.," in *Robotics: Science and Systems*, vol. 2, Ann Arbor, MI, USA, 2016.

[141] A. Dreves and M. Gerdts, "A generalized nash equilibrium approach for optimal control problems of autonomous cars," *Optimal Control Applications and Methods*, vol. 39, no. 1, pp. 326–342, 2018.

[142] B. J. Goode and M. J. Roan, "A differential game theoretic approach for two-agent collision avoidance with travel limitations," *Journal of Intelligent & Robotic Systems*, vol. 67, no. 3-4, pp. 201–218, 2012.

[143] K. Huang, X. Di, Q. Du, and X. Chen, "A game-theoretic framework for autonomous vehicles velocity control: Bridging microscopic differential games and macroscopic mean field games," *arXiv preprint arXiv:1903.06053*, 2019.

[144] R. Spica, D. Falanga, E. Cristofalo, E. Montijano, D. Scaramuzza, and M. Schwager, "A real-time game theoretic planner for autonomous two-player drone racing," *arXiv preprint arXiv:1801.02302*, 2018.

[145] Y. Vorobeychik and M. P. Wellman, "Stochastic search methods for nash equilibrium approximation in simulation-based games," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, pp. 1055–1062, International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[146] D. M. Reeves and M. P. Wellman, "Computing best-response strategies in infinite games of incomplete information," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 470–478, AUAI Press, 2004.

[147] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank, *Matrix Riccati equations in control and systems theory*. Birkhäuser, 2012.

[148] N. Van Long and G. Sorger, "A dynamic principal-agent problem as a feedback stackelberg differential game," *Central European Journal of Operations Research*, vol. 18, no. 4, pp. 491–509, 2010.

[149] A. Bensoussan, S. Chen, and S. P. Sethi, "The maximum principle for global solutions of stochastic stackelberg differential games," *SIAM Journal on Control and Optimization*, vol. 53, no. 4, pp. 1956–1981, 2015.

[150] J. F. Bard and J. T. Moore, "A branch and bound algorithm for the bilevel programming problem," *SIAM Journal on Scientific and Statistical Computing*, vol. 11, no. 2, pp. 281–292, 1990.

[151] D. Arias, A. Mota, L. Mota, and C. Castro, "A bilevel programming approach for power system operation planning considering voltage stability and economic dispatch," in *2008 IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, pp. 1–6, IEEE, 2008.

[152] J.-H. Ryu, V. Dua, and E. N. Pistikopoulos, "A bilevel programming framework for enterprise-wide process networks under uncertainty," *Computers & Chemical Engineering*, vol. 28, no. 6-7, pp. 1121–1129, 2004.

[153] S. T. DeNegre and T. K. Ralphs, "A branch-and-cut algorithm for integer bilevel linear programs," in *Operations research and cyber-infrastructure*, pp. 65–78, Springer, 2009.

[154] M. Fischetti, I. Ljubić, M. Monaci, and M. Sinnl, "Intersection cuts for bilevel optimization," in *International Conference on Integer Programming and Combinatorial Optimization*, pp. 77–88, Springer, 2016.

[155] B. Colson, P. Marcotte, and G. Savard, "A trust-region method for nonlinear bilevel programming: algorithm and computational experience," *Computational Optimization and Applications*, vol. 30, no. 3, pp. 211–227, 2005.

[156] H. Farvaresh and M. M. Sepehri, "A branch and bound algorithm for bi-level discrete network design problem," *Networks and Spatial Economics*, vol. 13, no. 1, pp. 67–106, 2013.

[157] J. B. E. Etoa, "Solving convex quadratic bilevel programming problems using an enumeration sequential quadratic programming algorithm," *Journal of Global Optimization*, vol. 47, no. 4, pp. 615–637, 2010.

[158] M. Zhu and S. Martinez, "Stackelberg-game analysis of correlated attacks in cyber-physical systems," in *Proceedings of the 2011 American Control Conference*, pp. 4063–4068, IEEE, 2011.

[159] F.-L. Meng and X.-J. Zeng, "A stackelberg game-theoretic approach to optimal real-time pricing for the smart grid," *Soft Computing*, vol. 17, no. 12, pp. 2365–2380, 2013.

[160] K. G. Murty and F.-T. Yu, *Linear complementarity, linear and nonlinear programming*, vol. 3. Citeseer, 1988.

[161] G. B. Dantzig and M. N. Thapa, *Linear programming 2: theory and extensions*. Springer Science & Business Media, 2006.

[162] G. Goodwin, M. M. Seron, and J. A. De Doná, *Constrained control and estimation: an optimisation approach*. Springer Science & Business Media, 2006.

[163] M. Simaan and J. B. Cruz, "On the stackelberg strategy in nonzero-sum games," *Journal of Optimization Theory and Applications*, vol. 11, no. 5, pp. 533–555, 1973.

[164] Z. H. Gümüş and C. A. Floudas, "Global optimization of nonlinear bilevel programming problems," *Journal of Global Optimization*, vol. 20, no. 1, pp. 1–31, 2001.

[165] N. Li, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. Girard, "Game theory-based traffic modeling for calibration of automated driving algorithms," in *Control Strategies for Advanced Driver Assistance Systems and Autonomous Driving Functions*, pp. 89–106, Springer, 2019.

[166] M. C. Priess, J. Choi, and C. Radcliffe, "Determining human control intent using inverse lqr solutions," in *Proc. ASME Dyn. Syst. Control Conf*, pp. 1–8, 2013.

[167] M. C. Priess, R. Conway, J. Choi, J. M. Popovich, and C. Radcliffe, "Solutions to the inverse lqr problem with application to biological systems analysis," *IEEE Transactions on control systems technology*, vol. 23, no. 2, pp. 770–777, 2014.

[168] H. El-Hussieny, A. Abouelsoud, S. F. Assal, and S. M. Megahed, "Adaptive learning of human motor behaviors: An evolving inverse optimal control approach," *Engineering Applications of Artificial Intelligence*, vol. 50, pp. 115–124, 2016.

[169] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, p. 1, ACM, 2004.

[170] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning.," in *Aaai*, vol. 8, pp. 1433–1438, Chicago, IL, USA, 2008.

[171] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 182–189, 2011.

[172] Y. Lu, Y.-E. Ge, and L.-W. Zhang, "An alternating direction method for solving a class of inverse semidefinite quadratic programming problems," *J. Ind. Manag. Optim*, vol. 12, no. 1, pp. 317–336, 2016.

[173] J. Wu, Y. Zhang, L. Zhang, and Y. Lu, "A sequential convex program approach to an inverse linear semidefinite programming problem," *Asia-Pacific Journal of Operational Research*, vol. 33, no. 04, p. 1650025, 2016.

[174] T. Jaakkola, S. P. Singh, and M. I. Jordan, "Reinforcement learning algorithm for partially observable markov decision problems," in *Advances in neural information processing systems*, pp. 345–352, 1995.