DEGREE PROJECT IN ELECTRICAL ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

Platoon Coordination under Signal Temporal Logic Specifications

Master Thesis

AKASH SINGH

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Platoon Coordination under Signal Temporal Logic Specifications

AKASH SINGH

Master in Systems, Controls and Robotics
Date: September 27, 2020
Supervisor: Christos Verginis
Examiner: Dimos Dimorgonas
School of Electrical Engineering and Computer Science
Swedish title: Platoon-samordning under specifikationer för temporär logisk signal

Abstract

Autonomous Driving has been hailed as one of the biggest game-changers for solving a lots of problems of the future, such as traffic congestion, efficiency of transportation, carbon emissions, road safety etc. Vehicle platooning has been a well studied idea in this field for efficient highway driving, which has been around since a few decades now. This work aims at planning for tasks that will be undertaken by an autonomous vehicle in a computationally efficient manner instead of usual methods that are calculation heavy, like MPC, and present how they can enable the vehicle to move around highway on its own or in coordination with a platoon. Using computationally simple planners leaves enough bandwidth for processors to look for unprecedented changes in a dynamic environment.

The major contribution of this thesis is in demonstrating the formulation of specification for such tasks in a dynamic and heterogeneous environment using Signal Temporal Logic (STL). Control Barrier Functions (CBF) are then used to represent them as constraints for the system which are in turn fulfilled by a quadratic optimization-based controller. The STL definition allows more insight into planning for the system from a time-based perspective, while employing CBFs as constraints guarantees the system never leaves its specified set of safe states. Two different maneuvers, namely lane changing and overtaking-merging with a platoon, are performed. The merging task also demonstrates cooperation between vehicles using the same method, which shows its suitability for platoon coordination tasks. These results are presented from a computer based simulation which, along with the system and controller, also models different traffic behaviours. Particularly, the average speed of the traffic and the traffic density are modelled to verify the suitability of the controller in varying conditions and understand its limits. Also, experiments with real robots were performed in laboratory environment, which were done with the single-integrator holonomic vehicle model to assess the feasibility of the method in real world.

Sammanfattning

Autonom körning har hyllats en av de största spelväxlarna för att lösa många framtidsproblem, såsom trafikstockningar, effektivitet i transporter, koldioxidutsläpp, trafiksäkerhet etc. En Kolonn har varit en väl studerad idé inom detta område för en effektiv motorvägskörning, som har funnits sedan några decennier nu. Detta arbete syftar till att planera för uppgifter som kommer att utföras av ett autonomt fordon på ett beräkningseffektivt sätt, istället för vanliga beräkningsmetoder som MPC, och presentera hur de kan göra det möjligt för fordonet att röra sig på motorvägen på egen hand eller i samordning med en kolonn. Användning av beräkningsenligt planerare lämnar tillräckligt med bandbredd för att processorer ska kunna leta efter aldrig tidigare skådade förändringar i en dynamisk miljö.

Det viktigaste bidraget med denna avhandling är att demonstrera formuleringen av specifikation för sådana uppgifter i en dynamisk och heterogen miljö med hjälp av Signal Temporal Logic (STL). Control Barrier Functions (CBF) används sedan för att representera dem som begränsningar för systemet som i sin tur uppfylls av en kvadratisk optimeringsbaserad styrenhet. STL-definitionerna möjliggör mer insikt i planering av systemet ur ett tidsbaserat perspektiv, medan användning av CBF som begränsningar garanterar att systemet aldrig lämnar sin specificerade uppsättning säkra tillstånd. Två olika manövrar, nämligen körbyte och omkörning-sammanslagning med en kolonn, utförs. Den sammanslagna uppgiften visar också samarbete mellan fordon med samma metod, vilket visar att det är lämpligt för samordning av plutoner. Dessa resultat presenteras från en datorbaserad simulering, som tillsammans med systemet och styrenheten också modellerar olika trafikbeteenden. Särskilt modelleras trafikens genomsnittliga hastighet och trafiktätheten för att verifiera styrenhetens lämplighet och förstå dess gränser. Experiment med riktiga robotar utfördes också i laboratoriemiljö, som gjordes med en integrerad holonomisk fordonsmodell för att bedöma metodens genomförbarhet i den verkliga världen.

Acknowledgements

First thanks for this work, without a doubt, goes to my supervisor, Chris, for your precious guidance and freedom that I had while choosing and working with this project. You helped me countless of times whenever I hit a roadblock with a method or concept, went out of the way to debug my codes and been a friend throughout this work. If it were not for you, I doubt if I could have finished what I set out to achieve. Furthermore, I would like to express my gratitude to my examiner Dimos for offering me this opportunity to carry out the thesis work under his guidance.

Countless thanks go out to all the people and colleagues at the Automatic Control Department who created such an enjoyable work atmosphere, atleast until as long as offices were not locked down due to Covid pandemic. Thanks to my desk neighbors Johan and Marti for all the discussions over lunch and otherwise and the fikas at the department. Similarly the people at the Smart Mobility Lab, where I carried out the experiments during latter phase of my thesis work, helped to complete the tasks smoothly. Sincere thanks to Robin for explaining the lab and its tools to us, and for being always available whenever we had troubles. Thanks to Francesca and Mustafa because of whom it was a bit less boring to work in an almost empty lab.

The toughest part of these months was definitely the time spent alone at home due to health safety measures in place. It is so easy to slip into the abyss of dullness in such times. But I am thankful that I had John as my friend. You kept me motivated and kept pushing me in numerous ways, be it our over-the-video exercise sessions or our conversations about life, love and work. But our friendship extends much beyond and further than this period. I can not express how grateful I am for all the support and love you have shown to me in this foreign land and helped me explore the beautiful Sweden and its beautiful culture and values. At the same time I have to thank Anirvan. Your feedback, your advice for my personal and academic matters, your awesome dinners, and the trips with you, have kept me afloat countless number of times in the rough waters. You have been a constant source of inspiration to me. Tianze, you are one of the first friends that I made and I'm glad we are still going strong. You are a very generous person and I appreciate our friendship a lot. Of course, I cannot conclude this chapter without thanking you Priti. There are not enough words to express my gratitude for all the emotional support you have provided me while being seven seas apart. I have been really lucky to have such, and many more great friends. I wish you all much happiness and progress in life.

Last but not the least, my family who have been the strongest pillars of

support in my life. I want to thank my Mom, Dad and *Didi* for supporting me no matter what. Thank you for teaching me to keep fighting and making me who I am. I am grateful to the almighty for blessing me with the best family there could be.

Contents

Chapter 1

Introduction

With the advent of autonomous driving, one more feature that is unlocked in the field of road transportation is that of cooperative driving. Formation of a platoon of heavy-duty vehicles, driving at close inter-vehicular distance is one such application which is particularly explored in this work. It presents great potential to reduce road congestion, and to improve fuel efficiency [1] for individual vehicles besides other advantages, such as increased safety, better drivability etc. This provides a clear motivation to delve deep into control of vehicular platoons and evaluate methods that have not been tried before.

1.1 Motivation and Literature Review

To be able to successfully join with a platoon, a vehicle needs to finish some constituent sub-tasks on its own leading upto merging with the platoon with cooperation from other vehicles of the platoon. An additional advantage would be to have some upper limits over how long it takes to complete those tasks safely.

The available control engineering techniques were explored keeping these objectives in mind. In robotics and static environment situations, temporal logics [2] have been used to complete complex tasks which can be sequential, periodic or reactive. It is a rigorous formalism for specifying certain desired behaviours of the discrete systems. When temporal logics are defined for specifying properties of a real-valued continuous signals, they are commonly called Signal Temporal Logic (STL) [3]. Thus allowing a validation methodology for continuous and hybrid systems, and also allowing to construct complex tasks having strict deadlines, unlike Linear Temporal Logics [2], where no real-time deadlines are addressed. In this project tasks such as lane changing, overtak-

ing & merging with the platoon are considered and temporal requirements are assigned to them. STL formulas are defined to represent these tasks which are then fulfilled by the final controller. In order to develop a controller from these STL formulas a suitable method is required which is not too computationally intensive, unlike the exploration of the entire state space using HJ reachability [4] or some other reachability analysis method which involve choosing one of the safe paths among those possible. In this work guaranteeing safety of the system is intended, while moving towards completion of the task based on the current state of the vehicle and environment without having to look at all the reachable states, having to extrapolate the unknowns or having to use a receding horizon control. Lyapunov's stability theorem provides inspiration for methods which are useful for ensuring forward invariance of a set.

Barrier function initially employed in optimization [5], are now commonly used in control systems to drive a set forward invariant [6]. A method to transform the STL formulas into constraints which can then be employed by an optimization-based controller has been demonstrated in [7], which is particularly of our interest here and where most of the inspiration for this thesis work is drawn from. In continuation of the problem statement in [7], this work extends the framework by including more STL fragments (disjunction operator per se), and by showing how this method can work in a dynamic and heterogeneous environment where not all agents are necessarily cooperative and by presenting a decentralized methodology for multi-agent control. However, the underlying idea remains the same, i.e designing a controller that tries to find optimal inputs by minimizing a quadratic program, constrained by the time-varying control barrier functions which are derived from chosen STL formulas.

The inputs from the resulting controller are shown to guarantee the satisfaction of the assigned temporal tasks while ensuring safety of the vehicle. Traffic around our target vehicle and the platoon is assumed to have constant speed, and no abrupt behaviour while following all the intended traffic rules.

The structure of the rest of this thesis is as follows: Chapter 2 provides necessary background on the concepts used, such as STL and control barrier functions. Chapter 3 describes the model used, and the construction of the specified STL formulas and safety barrier certificates to be incorporated into the optimization-based controller. Chapter 4 describes how the simulations/experiments are designed and the results obtained. In Chapter 5, we discuss the results and other observations. Chapter 6 concludes the findings, along with proposing future improvements.

Chapter 2

Background

2.1 Signal Temporal Logic

Temporal logic allows for specification of relative order of events [2], which can be described as properties of a system. It is a formalism to also prove these properties of time-dependent systems. Various variants of temporal logics exist, with Linear Temporal Logic (LTL) being one of the most widely used temporal logic for discrete sequence of atomic states. It finds major use in program specification and verification in software engineering. It extends the propositional logic such as Disjunction ($\vee$), Conjunction($\wedge$), negation($\neg$), by combining them with temporal modalities such as "next"($\bigcirc$), "always"($\Box$ or G), "eventually"($\Diamond$ or F) and "Until"($\mathcal{U}$). The figure 2.1 explains how these modalities are evaluated over a sequence of events.

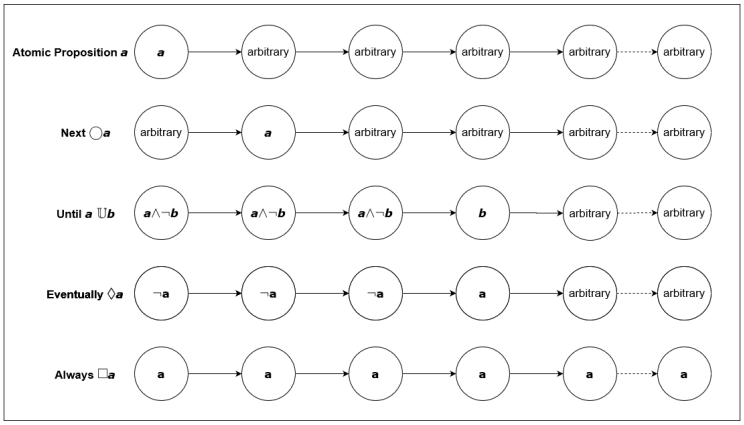


Figure 2.1: Evaluating LTL propositions over a sequence of events

Signal Temporal Logic (STL) is an extension of LTL for real-time and

real-valued properties. The temporal modalities in LTL are abstract instead of referring to precise timing of events, but STL can reflect those precise timing requirements in its formalism. STL is a predicate logic consisting of predicates μ that are obtained after evaluation of a predicate function $h : \mathbb{R}^n \to \mathbb{R}$ over a signal $y : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ [7] as

$$\mu = \begin{cases} \text{True} & \text{if } h(y(t)) \geq 0 \\ \text{False} & \text{if } h(y(t)) < 0 \end{cases}$$

For example, in STL a property is formulated as

$$h(y(t)) > 0 \Rightarrow F_{[0,5s]}(y(t) > 0),$$

where $h(y(t))$ is a predicate over real-valued signal $y(t)$ between real time $[0, 5s]$. It is read as eventually, within next 5s, $y(t) > 0$ would be true.

The STL syntax defines an STL formula ϕ and is given by

$$\phi := \text{True} | \mu | \neg\phi | \phi_1 \vee \phi_2 | \phi_1 \wedge \phi_2 | \phi_1 \, \mathcal{U} \, [a, b] \phi_2,$$

where ϕ_1 and ϕ_2 are STL formulas and $a, b \in \mathbb{R}_{\geq 0}$ with $a \leq b$.
The Eventually operator ($\Diamond$) is defined as $\text{F}_{[a,b]}\phi = \text{True} \, \mathcal{U}_{[a,b]} \, \phi$.
The Always operator ($\Box$) is defined as $\text{G}_{[a,b]}\phi = \neg(\text{F}_{[a,b]}\neg\phi)$.

STL semantics :

The STL semantics for a signal $y : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ satisfying formula ϕ is given below [3] :

$$\begin{array}{lcl} (y,t) \models \mu & \Leftrightarrow & h(y(t)) \geq 0 \\ (y,t) \models \neg\phi & \Leftrightarrow & \neg((y,t) \models \phi) \\ (y,t) \models \phi_1 \vee \phi_2 & \Leftrightarrow & (y,t) \models \phi_1 \vee (y,t) \models \phi_2 \\ (y,t) \models \phi_1 \wedge \phi_2 & \Leftrightarrow & (y,t) \models \phi_1 \wedge (y,t) \models \phi_2 \\ (y,t) \models \phi_1 \, \mathcal{U}_{[a,b]} \, \phi_2 & \Leftrightarrow & \exists t_1 \in [t+a, t+b] \text{ s.t } (y,t_1) \models \phi_2 \\ & & \wedge \, \forall t_2 \in [t, t_1], (y,t_2) \models \phi_1 \\ (y,t) \models F_{[a,b]}\phi & \Leftrightarrow & \exists t_1 \in [t+a, t+b] \text{ s.t } (x,t_1) \models \phi \\ (y,t) \models G_{[a,b]}\phi & \Leftrightarrow & \forall t_1 \in [t+a, t+b] \text{ s.t } (x,t_1) \models \phi \end{array} \tag{2.1}$$

The satisfaction relation $(y,t) \models \phi$ means signal $y : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ satisfies ϕ at time t.

2.2 Control Barrier Functions

Control barrier function(CBF) is a widely used tool in control theory dealing with derivation of controllers that ensures a safe set remains forward invariant

[6] [8] [9] [10] [11] [12], which is an accepted definition for safety of a system. Forward invariance for a dynamical system means that solutions for the set evolve within the set [13] and never leave its boundary. Similar to Control Lyapunov Functions, they can also be used to prove certain properties of the system, without needing to explicitly calculate the entire reachable set, which saves us from a lot of computation.

Barrier functions have been traditionally used in optimization, where they are added to a cost function in order to avoid undesirable zones [12]. Mathematically this can described as

$$B(\boldsymbol{x}) \to \infty \text{ as } \boldsymbol{x} \to \partial\mathcal{D},$$

where $B(\boldsymbol{x})$ is the barrier function, or reciprocal barrier function to be exact, associated with the set $\mathcal{D}$. The region outside of the set $\mathcal{D}$ is considered undesirable; $\partial\mathcal{D}$ here symbolizes boundary of the set. In theory, two classes of Barrier function are defined : Reciprocal (RBF) and Zeroing Barrier function (ZBF). The reciprocal Barrier function differs from ZBF in that, RBF goes to ∞ outside the desirable set $\mathcal{D}$, instead of 0 as in the case of ZBF. ZBF is more commonly used, since it is well-defined outside of $\mathcal{D}$ [12].

The idea is to design the controller such that the system states are not driven outside of the safe set. That's where the Lyapunov's stability theorem comes to our help [14].

Here affine control systems are considered:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}, \tag{2.2}$$

where f and g are locally lipschitz, $\boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n$, $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^m$.

In order for a feedback law, $\boldsymbol{u} = k(\boldsymbol{x})$, that stabilizes the system 2.2 to exist, it is sufficient to show that there exists a positive definite function, $V(\boldsymbol{x})$ that can be driven to zero [12], i.e if

$$\exists \boldsymbol{u} = k(\boldsymbol{x}) \text{ s.t. } L_f V(\boldsymbol{x}) + L_g V(\boldsymbol{x}) k(\boldsymbol{x}) \leq -\alpha(V(\boldsymbol{x})),$$

where L_f and L_g are directional-derivatives along f and g respectively, defined as

$$L_f V(\boldsymbol{x}) = \frac{\partial f}{\partial \boldsymbol{x}}.f$$

α is a class κ function. A continuous function $\alpha : [0, a) \to [0, \infty)$ belongs to κ function class [15] if:

- $\alpha(0) = 0$,

- the function is strictly increasing.

This gives birth to the conditions for existence of Control Lyapunov functions (CLFs), V which are positive definite and follow:

$$\inf_{\boldsymbol{u}\in\mathbb{U}}[L_fV(\boldsymbol{x}) + L_gV(\boldsymbol{x})k(\boldsymbol{x}) + \alpha(V(\boldsymbol{x}))] \leq 0 \tag{2.3}$$

Extending the motivation from CLF which provide tools to stabilize a system, tools to ensure safety of a system, which mathematically means keeping a set invariant, can be found. Suppose $\mathcal{C}$ is the superset for $V(\boldsymbol{x})$ s.t $\mathcal{C} = \boldsymbol{x} \in \mathbb{R}^n : V(\boldsymbol{x}) \geq 0$. The controllers derived from 2.3 will render set $\mathcal{C}$ invariant, which is proved using Nagumo's theorem in [16]. But the results from generalizing lyapunov for safety can be overly restrictive, since only set $\mathcal{C}$ is required to be invariant and not its sub-level sets [12]. Thus in order to find a Control Barrier Function $h(\boldsymbol{x})$, the requirement of being a positive definite function is now relaxed and α is now an extended class κ_∞ function to accommodate the increased domain. $\dot{h}$ is required to be bounded only from below by a negative number, thus the inequality can now be reversed. Now $\dot{h}$ is allowed to be both positive, which forces forward invariance of the level sets, and negative, which allows the system to move between the level sets. These modifications help us in achieving the objectives mentioned above. Being an extended κ_∞ function means $\alpha(-\infty, \infty) \to (-\infty, \infty]$ is now defined on entire real line and satisfies the following [15]:

- α belongs to class κ function
- $\lim_{r\to\infty} \alpha(r) \to \infty$

For a particular function $h(\boldsymbol{x})$ a superlevel set $\mathfrak{S}$ is defined such that:

$$\begin{aligned} \mathfrak{S} &:= \{\boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n : h(\boldsymbol{x}) \geq 0\} \\ \partial\mathfrak{S} &:= \{\boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n : h(\boldsymbol{x}) = 0\} \\ \text{Int}(\mathfrak{S}) &:= \{\boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n : h(\boldsymbol{x}) > 0\} \end{aligned} \tag{2.4}$$

Thus we are ready to define a CBF as follows [11]:

Definition 1 : Let a dynamical system 2.2 and the set $\mathfrak{S}$ defined by 2.4 for a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$. If there exists a locally Lipschitz extended class κ_∞ function α and set $\mathfrak{S} \subseteq \mathcal{D} \subseteq \mathbb{R}^n$ such that $\forall \boldsymbol{x} \in \mathcal{D}$,

$$\sup_{\boldsymbol{u}\in\mathbb{U}}[L_fh(\boldsymbol{x}) + L_gh(\boldsymbol{x})\boldsymbol{u} + \alpha(h(\boldsymbol{x}))] \geq 0 \tag{2.5}$$

then the function h is called Control Barrier Function(CBF) or Zeroing Control Barrier Function to be exact, for the set $\mathfrak{S}$. In a general sense, it can be said that the set of all CLFs is a subset of the set of all CBFs. That is, CBFs are more general but include CLFs as a special case.

Now the inequality constraints set on the derivative of candidate CBFs are in fact affine in $\boldsymbol{u}$. And thus an optimization-based controller, can be used to get the control inputs that would render a set forward invariant.

2.3 Time Varying Control Barrier Functions

Taking inspiration from the previous section and [7], a time-varying control barrier function can be defined. Consider a continuous and differentiable function $b : \mathcal{D} \times [t_0, t_1] \rightarrow \mathbb{R}$. Similar to equation 2.4, we define a set which explicitly depends on time for b.

$$\mathfrak{C}(t) := \{\boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n : b(\boldsymbol{x}, t) \geq 0\}$$

The function $b(\boldsymbol{x}, t)$ can be called a candidate control barrier function if it can be ensured that $\mathfrak{C}(t)$ is non-empty for each $t \in [t_0, t_1]$ [7]. Also, a function $\boldsymbol{x} : [t_0, t_1] \rightarrow \mathbb{R}^n$ needs to exist such that $\boldsymbol{x}(t) \in \mathfrak{C}(t), \forall t \in [t_0, t_1]$. This function needs to be absolutely continuous in t for finding a solution to 2.2, i.e for a $\boldsymbol{u} : [t_0, t_1] \rightarrow \mathbb{U}$ to exist.

As defined in [7], a valid time-varying control barrier function is:

Definition 2 : A candidate barrier control function $b(\boldsymbol{x}, t)$ is a valid control barrier function for 2.2 if there exists a locally Lipschitz continuous class κ_∞ function α such that, for all $(\boldsymbol{x}, t) \in \mathfrak{C}(t) \times [t0, t1]$,

$$\sup_{\boldsymbol{u} \in \mathbb{U}} \frac{\partial b(\boldsymbol{x}, t)^T}{\partial \boldsymbol{x}}(f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}) + \frac{\partial b(\boldsymbol{x}, t)}{\partial t} + \alpha(b(\boldsymbol{x}, t)) \geq 0 \tag{2.6}$$

2.4 Optimization-based Control

Mathematically, an optimization problem has the form [17]:

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & f_i(\boldsymbol{x}) \leq b_i, i = 1, 2, ...m \end{aligned}$$

Here, vector $\boldsymbol{x} = [x_1, x_2...x_n]^T$ is the variable which is being optimized. The function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective function in $\boldsymbol{x}$ which is being minimized, subject to the constraint functions in $\boldsymbol{x}$, $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, where $i = 1, ..., m$. The

constants $b_i, ..., b_m$ are the bounds of the constraints. These constraints represent the model's properties or performance, actuator constraints etc depending upon their use.

Optimization theory has been well studied and there are well-developed tools which are able to solve some standard forms of optimization by exploiting the convexity of the constraints and/or objective function. The major challenge in optimization problems lie in representing/approximating the system as these standard forms.

In this work, since we managed to develop affine constraints, we restrict our usage to quadratic optimization, where our objective function will be quadratic.

Chapter 3

Methodology

In this chapter the vehicle model is discussed and the description of the behavioural specifications using STL is provided. Next a discussion on how to formulate the time-varying control barrier functions for different STL formulae is provided. All this leads to the derivation of the optimization problem that will be solved. The implementation of cooperative control and implementing it in a decentralised fashion is also discussed.

3.1 Vehicle Modelling

While deriving the dynamic model for a vehicle, certain valid assumptions are made that are applicable for a highway driving scenario such as the steering input being small and traffic moving at a constant speed. Also, the longitudinal direction is considered on the x-axis and lateral direction on the y-axis in figure 3.1.

Both bicycle and single-integrator holonomic model are considered in this work. The computer simulations of the tasks are performed using both models, while the lab experiments with real robots are performed using the holonomic model as per the hardware availability.

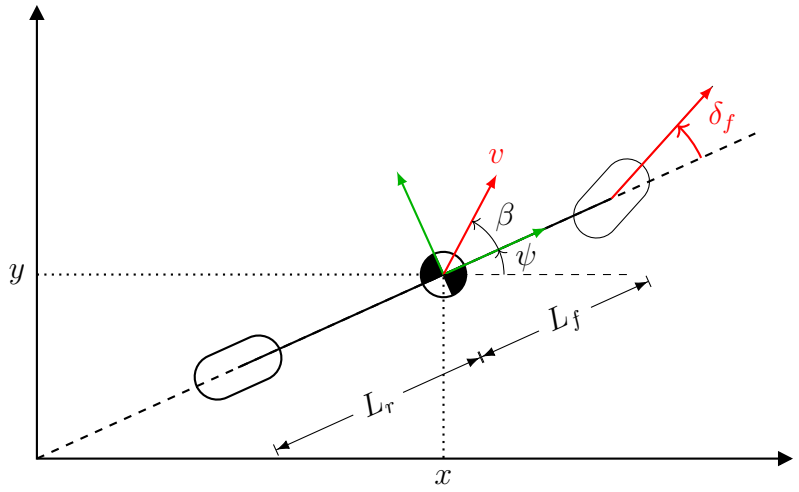


Figure 3.1: 2DOF Bicycle model

The 2-DOF bicycle model used in this work is inspired by the model used in [18]. The equations of motion as derived from the figure 3.1 are as follows:

$$\begin{aligned}
\dot{x} &= v\cos(\psi + \beta(u_2)) \\
\dot{y} &= v\sin(\psi + \beta(u_2)) \\
\dot{\psi} &= \frac{v}{L_r}\sin(\beta(u_2)) \\
\dot{v} &= u_1 \\
\delta_f &= u_2 \\
\text{where } \beta(u_2) &= \arctan(\tan(u_2)\frac{L_r}{L_r + L_f})
\end{aligned} \tag{3.1}$$

Here the 2 inputs to the system are the acceleration, u_1 and steering angle,u_2.

Based on the assumptions that highway driving uses small angles to steer, small angle approximation can be used for simplifying the system. Based on this approximation, trigonometric properties and further assuming $L_r = L_f = L$, a simplified model for the system that is used is:

$$\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{v}\\ \dot{\psi}\end{bmatrix}}_{\dot{\boldsymbol{x}}} = \underbrace{\begin{bmatrix}v\cos(\psi)\\ v\sin(\psi)\\ 0\\ 0\end{bmatrix}}_{f(\boldsymbol{x})} + \underbrace{\begin{bmatrix}0 & -\frac{v\sin(\psi)}{2}\\ 0 & \frac{v\cos(\psi)}{2}\\ 1 & 0\\ 0 & \frac{v}{L}\end{bmatrix}}_{g(\boldsymbol{x})} \underbrace{\begin{bmatrix}u_1\\ u_2\end{bmatrix}}_{\boldsymbol{u}} \tag{3.2}$$

Besides eq. 3.2, holonomic model has also been used for showing how to apply same method to different vehicle models. The single integrator model

as shown in eq 3.3 is used in that case.

$$\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}}_{\dot{\boldsymbol{x}}} = \underbrace{\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}}_{g(\boldsymbol{x})}\underbrace{\begin{bmatrix}u_1\\ u_2\end{bmatrix}}_{\boldsymbol{u}} \tag{3.3}$$

3.2 Formulating Control Barrier Functions for STL Specifications

After the rules and relaxations have been established, we can begin with the first step of our framework, which is defining safe set for our system. This is done by identifying certain signals governed by STL logic, relating to different states/properties of the vehicle. The super set of all the states which keep these signals positive is the safe set of the system.

If the composite STL formula that has been defined for the task is ϕ_m, and $b_m(\boldsymbol{x}(t), t)$ is the final control barrier function for the composite formula defined over the set $\mathcal{D} \times [t_0, t_1] \rightarrow \mathbb{R}_{\geq 0}$, then the safe set, $\mathfrak{D}$ is defined as:

$$\mathfrak{D}(t) := \boldsymbol{x} \in \mathcal{D} \subseteq \mathbb{R}^n : b_m(\boldsymbol{x}(t), t) \geq 0 \tag{3.4}$$

So for a given system 2.2 with initial condition $\boldsymbol{x}(t_0)$ that follows $(\boldsymbol{x}, t_0) \models \phi_m$, where the system specifications are encoded in the STL formula ϕ_m, the problem is to derive a control law $u(\boldsymbol{x}, t)$ for time interval $t \in [t_0, t_1]$ so that the solution $\boldsymbol{x} : \mathcal{D} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ to 2.2 satisfies ϕ_m for all the defined times, i.e $(\boldsymbol{x}, t) \models \phi_m \forall t \in [t_0, t_1]$

Now we will discuss in detail how to formulate a composite $b_m(\boldsymbol{x}(t), t)$ from ϕ_m, and subsequently the control law $u(\boldsymbol{x}, t)$.

In this work, the following STL fragments have been considered:

$$\Psi ::= T|\mu|\neg\mu|\Psi_1 \wedge \Psi_2 \tag{3.5a}$$

$$\phi ::= G_{[a,b]}\Psi|F_{[a,b]}\Psi|\Psi_1 U_{[a,b]}\Psi_2|\phi_1 \wedge \phi_2|\phi_1 \vee \phi_2 \tag{3.5b}$$

where Ψ_1 and Ψ_2 belong to same class of fragments as in 3.5a and ϕ_1 and ϕ_2 belong to those in 3.5b. The major difference being formulas of class Ψ do not consist of temporal operators, but those of class ϕ do.

Let us discuss the conditions that $b(\boldsymbol{x}, t)$[1] has to follow to account for the semantics of a given STL formula as compiled from [7]. Note that these general conditions are valid for formulas belonging to both class ψ and class ϕ.

[1] $b(\boldsymbol{x}(t), t)$ and $b(\boldsymbol{x}, t)$ notations have been used interchangeably, but they essentially refer to the same class of functions.

- It always needs to hold that $b(\boldsymbol{x_0}, 0) \geq 0$.
- With a negation operator, $\neg$, the corresponding predicate function should be chosen as $-b(\boldsymbol{x}, t)$.
- With a conjunction operator, $\wedge$, involving n predicates, i.e. $\psi = \psi_1 \wedge \psi_2 \wedge ...\psi_n$ the predicate function $b(\boldsymbol{x}, t)$ that ensures a temporal behaviour leading to satisfaction of ψ is $\min_{i \in [1...n]} b_i(\boldsymbol{x}, t)$, where b_i are corresponding predicate functions of participating predicates ψ_i. But due to non-differentiable nature of min function, a smooth under-approximation is used instead:

$$\min_{i \in [1...n]} b_i(\boldsymbol{x}, t) \approx -ln\left(\sum_{i=1}^{n} exp(-b_i(\boldsymbol{x}, t))\right) \tag{3.6}$$

Now we discuss about the conditions for $b(\boldsymbol{x}, t)$ meant for the formulas of class ϕ, i.e ones with temporal operators, again as in [7]. Let $h_1(\boldsymbol{x}, , h_2(\boldsymbol{x},$ be the predicate functions associated with predicates μ_1, μ_2 that are used below.

- Formulas with single temporal operator, i.e either always ($G_{[a,b]}\mu$), eventually ($F_{[a,b]}\mu$) or until ($\mu_1 \mathcal{U}_{[a,b]} \mu_2$) operator, but without any conjunction operator.
  - For $G_{[a,b]}\mu$ choose $b(\boldsymbol{x}, t)$ s.t. $\forall t^{'} \in [a, b] : b(\boldsymbol{x}, t^{'}) \leq h(\boldsymbol{x})$
  - For $F_{[a,b]}\mu$ choose $b(\boldsymbol{x}, t)$ s.t. $\exists t^{'} \in [a, b] : b(\boldsymbol{x}, t^{'}) \leq h(\boldsymbol{x})$
  - For $\mu_1 \ \mathcal{U}_{[a,b]} \ \mu_2$ choose $b(\boldsymbol{x}, t) \coloneqq -ln\left(e^{-b_1(\boldsymbol{x},t)} + e^{-b_2(\boldsymbol{x},t)}\right)$ s.t. $\exists t^{'} \in [a, b] : b_2(\boldsymbol{x}, t^{'}) \leq h_2(\boldsymbol{x})$ and $\forall t^{''} \in [a, t^{'}] : b_1(\boldsymbol{x}, t^{''}) \leq h_1(\boldsymbol{x})$
- Formulas with single temporal operator, either always ($G_{[a,b]}\Psi$), eventually ($F_{[a,b]}\Psi$) or until ($\Psi_1 \ \mathcal{U}_{[a,b]} \ \Psi_2$) operator, but the predicates may now contain conjunction operators i.e $\Psi_1 = \mu_1 \wedge \mu_2, \Psi_2 = \mu_3 \wedge \mu_4$
  - For $G_{[a,b]}\Psi$ choose $b(\boldsymbol{x}, t) \coloneqq -ln\left(e^{-b_1(\boldsymbol{x},t)} + e^{-b_2(\boldsymbol{x},t)}\right)$ s.t. $\forall t^{'} \in [a, b] : b_1(\boldsymbol{x}, t^{'}) \leq h_1(\boldsymbol{x},$ and $b_2(\boldsymbol{x}, t^{'}) \leq h_2(\boldsymbol{x},$
  - For $F_{[a,b]}\Psi$ choose $b(\boldsymbol{x}, t) \coloneqq -ln\left(e^{-b_1(\boldsymbol{x},t)} + e^{-b_2(\boldsymbol{x},t)}\right)$ s.t. $\exists t^{'} \in [a, b] : b_1(\boldsymbol{x}, t^{'}) \leq h_1(\boldsymbol{x},$ and $b_2(\boldsymbol{x}, t^{'}) \leq h_2(\boldsymbol{x},$
  - For $\Psi_1 \ \mathcal{U}_{[a,b]} \ \Psi_2$ choose $b(\boldsymbol{x}, t) \coloneqq -ln\left(\sum_{i=1}^{4} e^{-b_i(\boldsymbol{x},t)}\right)$ s.t. $\exists t^{'} \in [a, b] : b_3(\boldsymbol{x}, t^{'}) \leq h_3(\boldsymbol{x},$ & $b_4(\boldsymbol{x}, t^{'}) \leq h_4(\boldsymbol{x},$ and $\forall t^{''} \in [a, t^{'}] : b_1(\boldsymbol{x}, t^{''}) \leq h_1(\boldsymbol{x},$ & $b_1(\boldsymbol{x}, t^{''}) \leq h_1(\boldsymbol{x},$

3.2.1 Handling Disjunction Operators

An additional operator for the class of formulas in 3.5b that has been added in this work is the disjunction operator, $\vee$. Given the dynamic nature of the environment, alternative constraints are needed to fall back to in case the current state of environment is such that it does not permit any additional change in state of the vehicle according to the primary STL formula. In that case, the vehicle can continue being in the current safe state, or another secondary task can be assigned to the vehicle while it waits for the environment to become favourable again.

With a disjunction operator, $\vee$, involving n predicates, i.e. $\psi = \psi_1 \vee \psi_2 \vee ...\psi_n$ the predicate function $b(\boldsymbol{x}, t)$ that ensures a temporal behaviour leading to satisfaction of ψ is $\max_{i \in [1...n]} b_i(\boldsymbol{x}, t)$, where b_i are corresponding predicate functions of participating predicates ψ_i. Note that this applies to both ψ-class and ϕ-class fragments from 3.5.

In logic algebra, the disjunction and conjunction operators are complimentary to each other, and using De Morgan's law, they can be transformed as follows:

$$\mu_1 \vee \mu_2 = \overline{\bar{\mu_1} \wedge \bar{\mu_2}}$$

So taking inspiration from eq. 3.6, a similar smooth approximation for $\max_{i \in [1...n]} b_i(\boldsymbol{x}, t)$ would be

$$\max_{i \in [1...n]} b_i(\boldsymbol{x}, t) \approx ln \left(\sum_{i=1}^{n} exp(b_i(\boldsymbol{x}, t)) \right) \tag{3.7}$$

But there is a problem with this approximation, since it is an over approximation, which means all the constituting $b_i(\boldsymbol{x}, t)$ would be negative and still $ln\left(\sum_{i=1}^{n} exp(b_i(\boldsymbol{x}, t))\right)$ can be positive. And this is not what is desired.

Thus any number of STL formula joined together by disjunction operator are treated as separate predicates. Their predicate functions are formulated separately and a switching sequence is assigned to them based on a priority order. Thus note that the switching sequence is decided in advance. Since the magnitude of the predicate function, $b_i(\boldsymbol{x}, t)$ is also a measure of robustness of how well that constraint, ϕ_i is being followed, controller can switch to an alternate predicate function, $b_{i+1}(\boldsymbol{x}, t)$ meant for ϕ_{i+1} once the value of the $b_i(\boldsymbol{x}, t)$ falls below a certain threshold depending on design and performance requirements. Then controller can switch back to the primary STL formula ϕ_i, after spending some pre-defined time in the safe states reachable by ϕ_{i+1}. This cycle can be repeated until the primary STL formula meets its objective.

Alternate switching mechanism can be adopted, but the current work employs the method mentioned above.

3.3 Formulating Convex Quadratic Optimization-based Controller

Since a valid control barrier function has been formulated, $b_m(\boldsymbol{x}, t)$, a controller can now be designed to keep the set $\mathfrak{D}(t)$ invariant and such that $(\boldsymbol{x}, 0) \models \phi_m$. Taking inspiration from [7], we have:

$$\begin{aligned} &\min_{\hat{\boldsymbol{u}} \in \mathbb{U}} \hat{\boldsymbol{u}}^T \boldsymbol{Q} \hat{\boldsymbol{u}} \\ &\text{s.t } \frac{\partial b_m(\boldsymbol{x}, t)^T}{\partial \boldsymbol{x}} (f(\boldsymbol{x} + g(\boldsymbol{x})\hat{\boldsymbol{u}})) + \frac{\partial b_m(\boldsymbol{x}, t)}{\partial t} + \alpha(b_m(\boldsymbol{x}, t)) \geq 0 \end{aligned} \tag{3.8}$$

Since the constraint is always affine in $\boldsymbol{u}$, convex programming techniques like `quadprog` can be used to find a solution. For the case, $\frac{\partial b_m(\boldsymbol{x},t)^T}{\partial \boldsymbol{x}} = \boldsymbol{0}_n$, when the system becomes uncontrollable, one can switch to an alternate control law, such that $\dot{\boldsymbol{x}}(t) = \boldsymbol{0}_n$, which means that the system remains invariant with respect to the safe set. So the final control law can be written as:

$$\boldsymbol{u}(\boldsymbol{x}, t) := \begin{cases} -g(\boldsymbol{x})^T (g(\boldsymbol{x}) g(\boldsymbol{x})^T)^{-1} f(\boldsymbol{x}), & \text{if } \frac{\partial b_m(\boldsymbol{x}(t'), t')^T}{\partial \boldsymbol{x}} = \boldsymbol{0}_n \\ & \text{for some } t' \in [a, t] \\ \hat{\boldsymbol{u}} \text{ from 3.8}, & \text{otherwise} \end{cases} \tag{3.9}$$

for some $t \in [a, b]$ and where $b_m(\boldsymbol{x}, t)$ is the control barrier function defined over interval $[a, b]$.

3.3.1 Distributing Common STL Tasks Between Multiple Agents

In case there are 2 or more agents collectively working towards fulfilling a task, some changes need to be made in the way the controllers are formulated. The decentralised controller approach is more practical, especially for the vehicular applications. By decentralised approach, it is meant here that each agent now calculates its own feedback control law based on the knowledge of its own and its neighbor's states while still accomplishing the global goal. This is another extension to the work in [7] where only the centralised controller is used to control various agents.

Let the coupled CBF be of the form $b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t)$, i.e combined of the states of agent i and j. The constraints for a centralized QP problem in eq 3.8 can be rewritten as:

$$\min_{\hat{\boldsymbol{u}} \in \mathbb{U}} \hat{\boldsymbol{u}}^T \boldsymbol{Q} \hat{\boldsymbol{u}}$$
$$\text{s.t } -\frac{\partial b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t)^T}{\partial \boldsymbol{x}}(f(\boldsymbol{x} + g(\boldsymbol{x})\hat{\boldsymbol{u}})) \leq \frac{\partial b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t)}{\partial t} + \alpha(b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t))$$

where $\boldsymbol{x} = [\boldsymbol{x_1}, ..., \boldsymbol{x_i}, \boldsymbol{x_j}, ..]^T$ and $\hat{\boldsymbol{u}} = [\boldsymbol{u_1}, ..., \boldsymbol{u_i}, \boldsymbol{u_j}, ..]^T$

Without the loss of generality, consider 2 agents i and j, and the system dynamics modelled as $\dot{\boldsymbol{x}} = g(\boldsymbol{x})\boldsymbol{u}$ for both the agents. The constraint to be solved by optimization then becomes:

$$\begin{bmatrix} -\frac{\partial b_m}{\partial \boldsymbol{x_i}} & -\frac{\partial b_m}{\partial \boldsymbol{x_i}} \end{bmatrix} \begin{bmatrix} \boldsymbol{u_i} \\ \boldsymbol{u_j} \end{bmatrix} \leq \frac{\partial b_m}{\partial t} + \alpha(b_m) \tag{3.10}$$

In order to decentralize the CBF between 2 agents, we propose to distribute the constant terms between them, as follows:

$$[-\frac{\partial b_m}{\partial \boldsymbol{x_i}}][\boldsymbol{u_i}] \leq \frac{\eta_i}{\eta_i + \eta_j}\frac{\partial b_m}{\partial t} + \alpha(b_m) \tag{3.11a}$$
$$[-\frac{\partial b_m}{\partial \boldsymbol{x_j}}][\boldsymbol{u_j}] \leq \frac{\eta_j}{\eta_i + \eta_j}\frac{\partial b_m}{\partial t} + \alpha(b_m) \tag{3.11b}$$

where η_i and η_j are scalars their values can be decided based on the agility of the agents, or other design and performance considerations. For example, a more agile agent can be assigned a larger fraction of the constant terms to take care of.

Adding up the equations 3.11a and 3.11b leads us to the original constraint in 3.10, proving the equivalence of the two methods.

Thus, now we have a decentralized QP problem with $b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t)$ as coupled CBF and $b_{m2}(\boldsymbol{x_i}, t)$ as de-coupled CBF.

$$\min_{\hat{\boldsymbol{u_i}} \in \mathbb{U}} \hat{\boldsymbol{u_i}}^T \boldsymbol{Q} \hat{\boldsymbol{u_i}}$$
$$\text{s.t } -\frac{\partial b_m(\boldsymbol{x_i}, \boldsymbol{x_j} t)^T}{\partial \boldsymbol{x_i}}(g(\boldsymbol{x_i})\hat{\boldsymbol{u_i}})) \leq \frac{\eta_i}{\eta_i + \eta_j}\left(\frac{\partial b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t)}{\partial t} + \alpha_1(b_m(\boldsymbol{x_i}, \boldsymbol{x_j}, t))\right)$$
$$-\frac{\partial b_{m2}(\boldsymbol{x_i}, t)^T}{\partial \boldsymbol{x_i}}(g(\boldsymbol{x_i})\hat{\boldsymbol{u_i}})) \leq \frac{\partial b_{m2}(\boldsymbol{x_i}, t)}{\partial t} + \alpha_2(b_{m2}(\boldsymbol{x_i}, t)) \tag{3.12}$$

Same proof can be generalized for more than 2 agents which are coupled by one CBF, or when system dynamics has an $f(\boldsymbol{x})$ term which can be adjusted in the similar way as shown above.

Chapter 4

Simulation, Experiment Design and Results

In this chapter the two scenarios, that have been the main focus of this thesis, and how to actuate them using the framework described above are discussed. The design of the simulation/experiment is explained and then corresponding results are presented. Different models of vehicles, and different highway maneuvers have been used to demonstrate usability of the method in different situations.

The vehicle being controlled in the simulations and experiments is referred to as Ego vehicle in the text and the non-controlled vehicles are referred to as traffic vehicles. First, here are some constants used to model the environment and system for simulations and experiments.

For the simulations:

All the vehicles are assumed to have a frame of a truck, and the road dimensions have been considered which are approximately average of prevalent European standards. The solution has been arrived upon using these dimensions, but can be easily replaced for a different system accordingly.
Length of the vehicle = 8.2m
Width of the vehicle = 2.5m
Width of the lanes = 3.25m
Peak Acceleration allowed = 4m/s^2

For the experiment:

Since the nexus robotics 4W-mecanum-drive robots are used for carrying out the experiments, the dimensions of the road are modified in the same proportion as the ratio of width of the robots to the width of model used for simulations. The figures used to calculate the control feedback law and get the

results as presented are given below
Length of the vehicle = 40cm
Width of the vehicle = 40cm
Width of the lanes = 52cm
Top Speed allowed = 70mm/s

4.1 Changing lane on a Highway

The 2-DOF bicycle model is used to simulate the dynamic response of the vehicle for this scenario. Hence, the constraints are defined on the states $[x, y, v, \psi]$ of the vehicle.

The primary objective in this case is changing lane on a highway while observing the safety constraints for collision avoidance with the highway traffic and following the lane behaviour. The respective STL predicates can be formed as:

- To maintain the vehicle along the center of lane 1 for t_1 seconds

$$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\left(|y - y_{lane1}| < k_1\right)$$

- To move towards the center of lane 2 within the next t_2 seconds

$$\boldsymbol{\mu_2} = F_{[t'',t''+t_2]}\left(|y - y_{lane2}| < k_1\right)$$

- To always avoid collision with the oncoming traffic vehicles

$$\boldsymbol{\mu_3} = G_{[0,\infty]}\sum_{i}^{n}\left[\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 > 1\right]$$

- Alternate Objective - To slow down so as to let faster moving vehicle pass and keep following original lane

$$\boldsymbol{\mu_{stay}} = F_{[t''',t'''+t_3]}(\dot{v} < 0 \wedge |y - y_{lane1}| < k_{alt1})$$

where (x, y) is the position of the ego vehicle, (x_i, y_i) is the position of i^{th} neighbouring vehicle, y_{lane1} and y_{lane2} correspond to the center of the original lane and the lane to be merged into respectively. $t^{'}, t^{''}$ and $t^{'''}$ are the times of switching into the respective objective. $t_1, t_2, t_3, k_1, k_2, k_{alt1}$ are constants which are decided based on performance requirements and system constraints. $|.|$ represents absolute value function. Here, for collision avoidance, vehicles

are assumed to have ellipsoidal frames. If the center of the ego vehicle stays out of this ellipsoid, it will be able to avoid the collision.

The composite STL predicate that needs to be followed in order to safely fulfill the task is:

$$\boldsymbol{\phi_m} = (\boldsymbol{\mu_1} \wedge \boldsymbol{\mu_2} \wedge \boldsymbol{\mu_3}) \vee \boldsymbol{\mu_{stay}} \tag{4.1}$$

The above formula can be re-written into 2 formulas which can then be sequentially processed. This prevents results from being too conservative due to having formulas that are no longer required or will be needed after some time. Thus the formula is split into

$$\begin{aligned} \boldsymbol{\phi_{m1}} &= (\boldsymbol{\mu_1} \wedge \boldsymbol{\mu_3}) \vee \boldsymbol{\mu_{stay}} \\ \boldsymbol{\phi_{m2}} &= (\boldsymbol{\mu_2} \wedge \boldsymbol{\mu_3}) \vee \boldsymbol{\mu_{stay}} \end{aligned} \tag{4.2}$$

In order to assign priorities to different tasks and to assist in tuning of the controller later, scalar weights are assigned to each of the predicates. Thus the weighted STL formula becomes:

$$\begin{aligned} \boldsymbol{\phi_{m1}} &= (c_1.\boldsymbol{\mu_1} \wedge c_2.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}} \\ \boldsymbol{\phi_{m2}} &= (c_2.\boldsymbol{\mu_2} \wedge c_3.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}} \end{aligned} \tag{4.3}$$

The higher the weight among the alternate choices, the more priority that formula segment has. Since $\boldsymbol{\mu_{stay}}$ is a secondary objective, c_{stay} should be carefully chosen, so that the CBF value calculated for it should not appear to be more robust than the primary objective. At the same time, the values of these weights should be kept low for less reactive systems. The effects of the weights will be discussed in detail along with other results.

As mentioned in section 3.2, control barrier functions have to be formulated for each of the STL predicates. Here, t_1 is chisen as 3seconds, t_2 as 4seconds, t_3 as 1second and k_1 as 0.1m. The lane widths are chosen at 3.25m. So y_{lane1} lies at 0 and y_{lane2} lies at 3.25m. The initial position of the ego vehicle is at $[0, 0]$ and that of i^{th} neighbouring vehicle is at $[0, 3.25]$. In this case, $i = 1$ since only one neighbouring vehicle is considered.

For $\boldsymbol{\mu_1}$, the predicate function, $h_1(\boldsymbol{x})$ is $c_1(k_1 - |y - y_{lane1}|)$. To derive a valid control barrier function, $b_1(\boldsymbol{x}, t)$ from this, the rules mentioned in section 3.2 need to be followed. So, $b_1(\boldsymbol{x}, t)$ takes the form of $c_1(\gamma_1(t) - |y - y_{lane1}|)$. In order to have smooth derivatives and avoid oscillations about the desired equilibrium point, the $b_1(\boldsymbol{x}, t)$ is modeified as $c_1(\gamma_1^2(t) - (y - y_{lane1})^2)$. What needs to be taken care is that $b_1(0, 0) \geq 0$ and $\exists t^{'} \in [0, t_1]$ s.t $b_1 \leq h_1$. This leads us to choosing $\gamma_1(t)$ as $-\dfrac{3.0}{3.5}t + 3.5$, given the initial position of the

ego vehicle, and the performance requirement from the STL formula. So, $\forall t > 2.68$s, $b_1(\boldsymbol{x}, t) \leq h_1(\boldsymbol{x})$ and $b_1(0,0) = 9 > 0$.

For $\boldsymbol{\mu_3}$, the predicate function, $h_3(\boldsymbol{x})$ is $c_3\left(\left[\frac{x-x_i}{a}\right]^2 + \left[\frac{y-y_i}{b}\right]^2 - 1\right)$. Similarly, following rules for the global operator from section 3.2, we have $b_3(\boldsymbol{x}, t) = c_3\left(\left[\frac{x-x_i}{a}\right]^2 + \left[\frac{y-y_i}{b}\right]^2 - \gamma_3(t)\right)$. $\gamma_3(t) = \left(-0.1e^{(-0.5t-1.2)} - 1\right)$ is chosen so that $b_3(0,0) > 0$ and $\forall t, b_3(\boldsymbol{x}, t) \leq h_3(\boldsymbol{x})$.

The composite barrier function for ϕ_{m1} would be $b_{m1}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-b_1} + \mathrm{e}^{-b_3})$. This b_{m1}, and $f(\boldsymbol{x}), g(\boldsymbol{x})$ from eq.3.2 can be plugged into the controller equation in 3.9 to get the control law $u(x, t)$.

In a similar way, for $\boldsymbol{\mu_2}$, the predicate function is $c_2(k_2 - |y - y_{lane2}|)$, and a valid CBF $b_2(\boldsymbol{x}, t)$ would be $c_2(\gamma_2^2(t) - (y - y_{lane2})^2)$, where $\gamma_2(t) = -\frac{3.49}{4}t + 3.5$. The composite barrier function for ϕ_{m2} becomes $b_{m2}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-b_2} + \mathrm{e}^{-b_3})$.

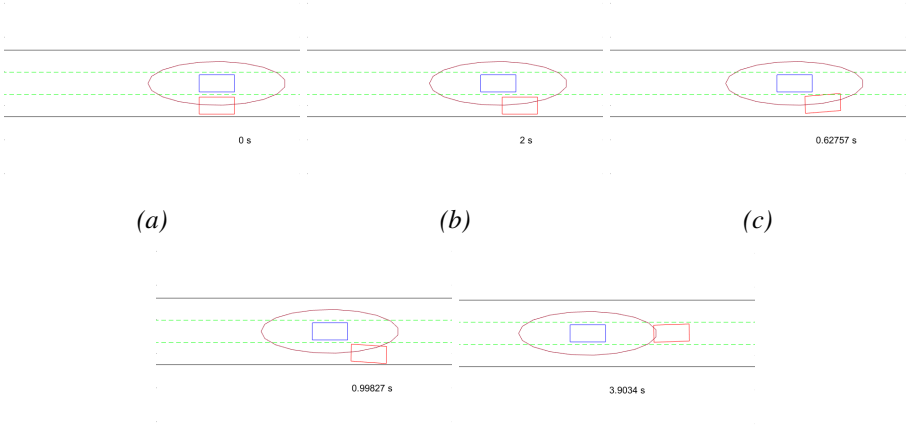
For the alternate objective, $\boldsymbol{\mu_{stay}}$, the CBF for $\dot{v} < 0$ can also be written as $b_{alt1} = c_{alt}(\gamma_{alt1}^2 - (v - \hat{v})^2)$, where $\hat{v}$ could be any lower velocity than the average speed that the system can safely converge to. $\hat{v}$ has been chosen as 1m/s in this case, and then substituting $\gamma_{alt1} = -4.5t + 11$ renders b_{alt} a suitable CBF. For $|y - y_{lane1}| < k_{alt1}$, a suitable CBF is $b_{alt2} = c_{alt2}\left((-\frac{3.24}{4}t + 1.5)^2 - (y - y_{lane1})^2\right)$ if $k_{alt1} = 0.5$ is chosen. Thus the composite CBF for $\boldsymbol{\mu_{stay}}$ is $b_{stay}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-b_{alt1}} + \mathrm{e}^{-b_{alt2}})$.

The controller sequentially shifts from ϕ_{m1} to ϕ_{m2} after the objective of the respective tasks are achieved within agreed margins. Switching between OR conditions within the STL formulas takes place as discussed in section 3.2.1, i.e based on the robustness parameter or the value of $b(\boldsymbol{x}, t)$.

All the constants, signals used to represent the STL formulas in this example in order to achieve the results of this thesis work have been provided again in Appendix A for easier readability.

In the given premise, the initial speed of the ego vehicle is 10m/s, and the traffic moves at speed of 7.5m/s. In this case the input constraints allow the ego vehicle to change the lane by moving in front of the traffic, as seen in figure 4.1. The visual tool for viewing the simulation results has been inspired from the similar tool used in [19]. The plot for magnitude of CBF is shown in figure 4.2b, which shows it remaining positive (almost) all the time, thus confirming safe state of the system as per the design. The magnitude can drop below zero when switching of constraints takes place. This can be adjusted by changing

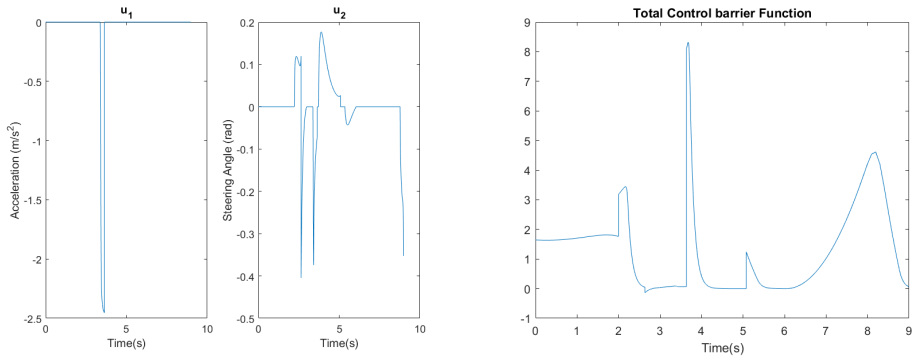
the step size of the solver, or threshold for switching. The plots for inputs and states of the vehicle at all times is shown in figure 4.2a and 4.3.



(a) *(b)* *(c)*

(d) *(e)*

Figure 4.1: (a). The initial configuration, when the ego vehicle is moving at speed of 10*m/s and traffic vehicle at* 7.5*m/s (b). The ego vehicle follows its own lane for first 2s (c). It begins attempting the change of lane, but within 0.7s realizes that its not safe (d). So it follows the alternative STL formula for next 1s (e). Again attempts the primary objective of changing lane and successfully completes within 4s. NOTE: The time displayed is relative, i.e the counter is restarted after each switching of STL formulas.*



(a) Inputs to the Ego Vehicle *(b) Value of the Control Barrier Function*

Figure 4.2: The total CBF is conjunction of constituent STL formulas, which is represented as the under-approximation given by eq 3.6.

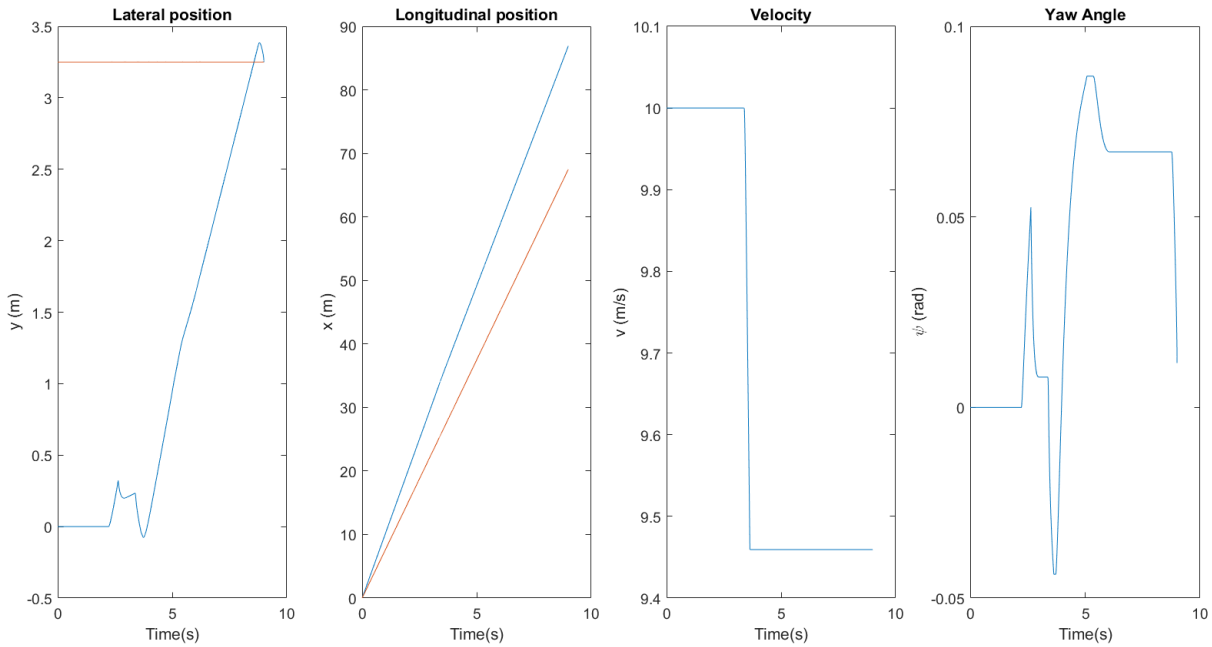
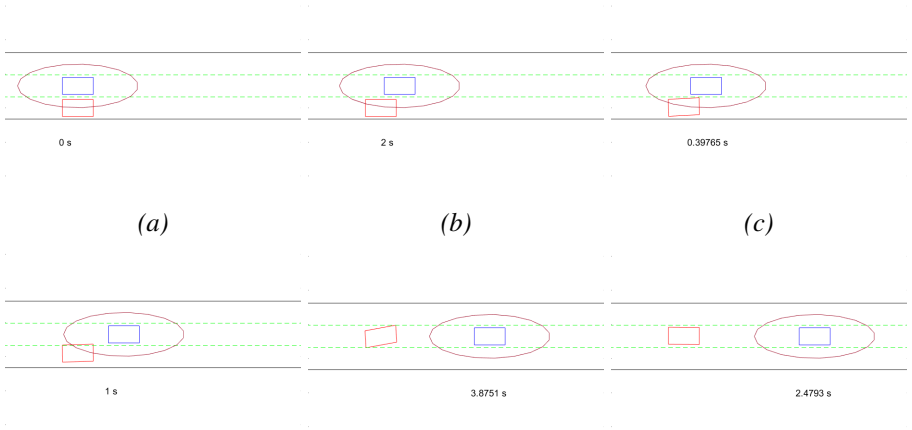


Figure 4.3: States of the Ego Vehicle

The controller can also choose to let the other vehicle pass first before changing lane if, for example, the traffic has higher average speed and the input constraints on the ego vehicle do not allow fast enough acceleration to safely change lane while also maintaining safe distance from traffic. In figure 4.4, the traffic has a steady speed of 12.5m/s and thus ego vehicle moves around it accordingly. The different approach taken by the controller this time is visible in the plots for states of the ego vehicle in figure 4.6. The CBF still remains positive ensuring the safety of the system as seen in figure 4.5.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 4.4: (a). The initial configuration, when the ego vehicle is moving at speed of $10m/s$ *and traffic vehicle at* $12.5m/s$ *(b). The ego vehicle follows its own lane for first 2s (c). Keeps trying to change the lane (d). It follows the alternative STL formula for next 1s while waiting for environment to be safe enough (e). Again attempts the primary objective of changing lane and successfully completes within 4s. (f). Aligns itself with the lane*

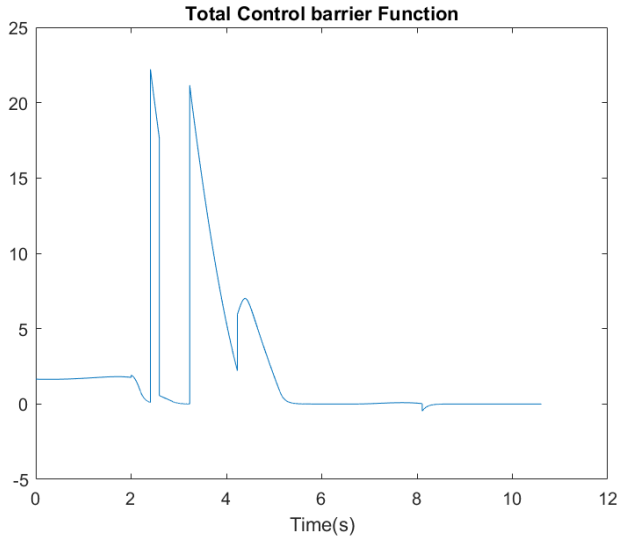


Figure 4.5: Value of Total Control Barrier Function

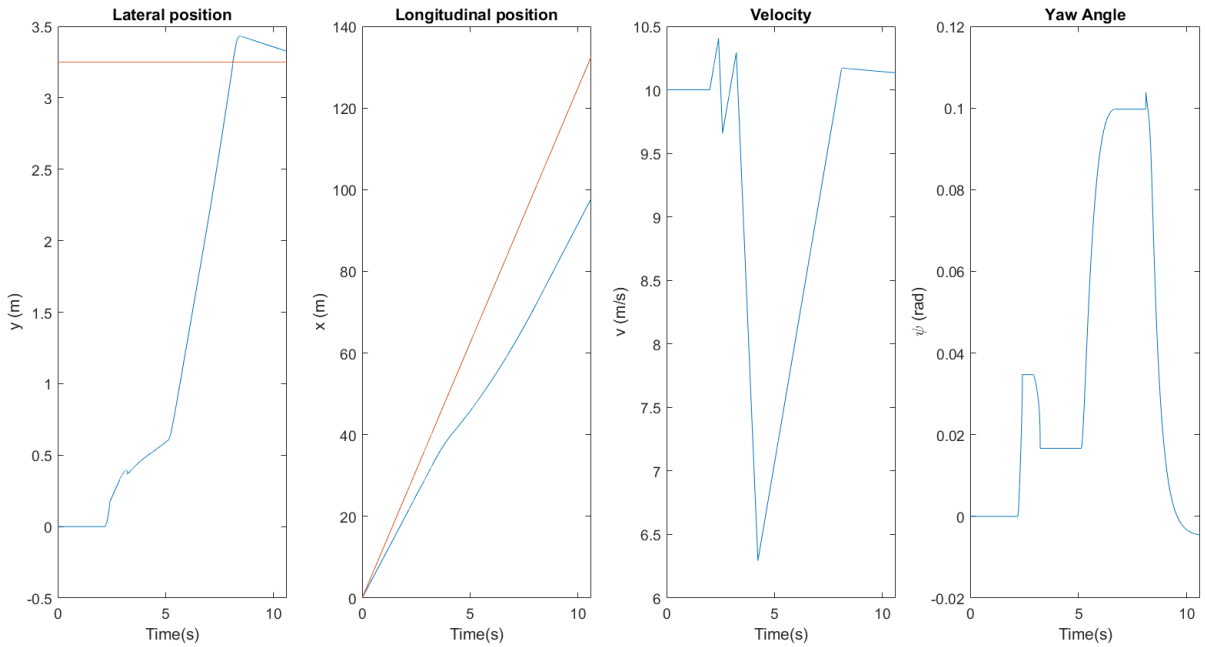


Figure 4.6: States of the Ego Vehicle

In order to check the robustness of the method, the traffic density was slightly changed to observe the behaviour of the controller.

Figure 4.7 shows how the vehicle behaves when the traffic is assumed to be moving at a steady speed of 10m/s and the distance between 2 traffic vehicles is equal to twice the length of the ego vehicle, i.e 16.4m. Safe states of the system are verified by the CBF value plot shown in figure 4.8b. The inputs to the ego vehicle and its state plot is shown in figure 4.8a and 4.9 respectively. In this case, since the controller does not deem it to be safe, the ego vehicle lets all the neighbouring traffic pass before moving into the next lane.

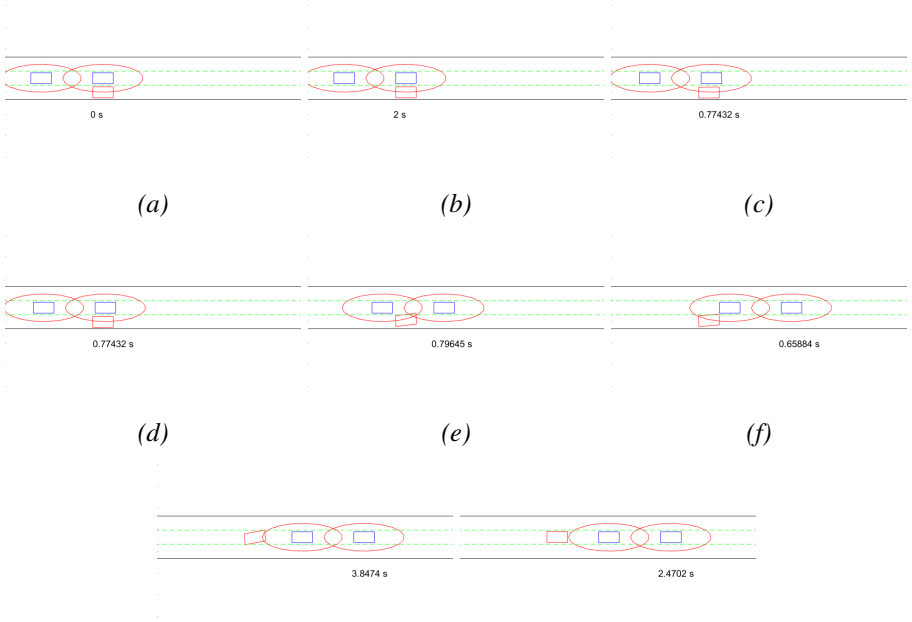
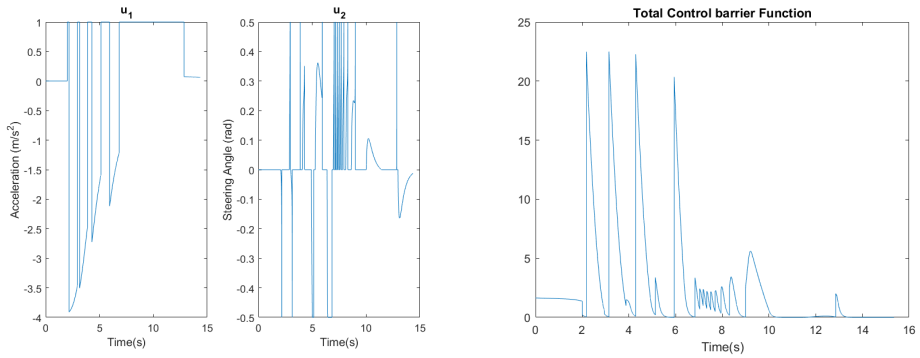


Figure 4.7: (a). The initial configuration, when the ego vehicle is moving at speed of 10*m/s and traffic vehicles also at* 10*m/s. The space between the traffic vehicles is twice that of length of ego vehicle (b). The ego vehicle follows its own lane for first* 2*s (c), (d), (e). Keeps trying to change the lane (f). Final attempt at the primary objective of changing lane is successful and completes within 4s. (g). Aligns itself with the lane*



(a) Inputs to the Ego Vehicle *(b) Value of the Control Barrier Function*

Figure 4.8

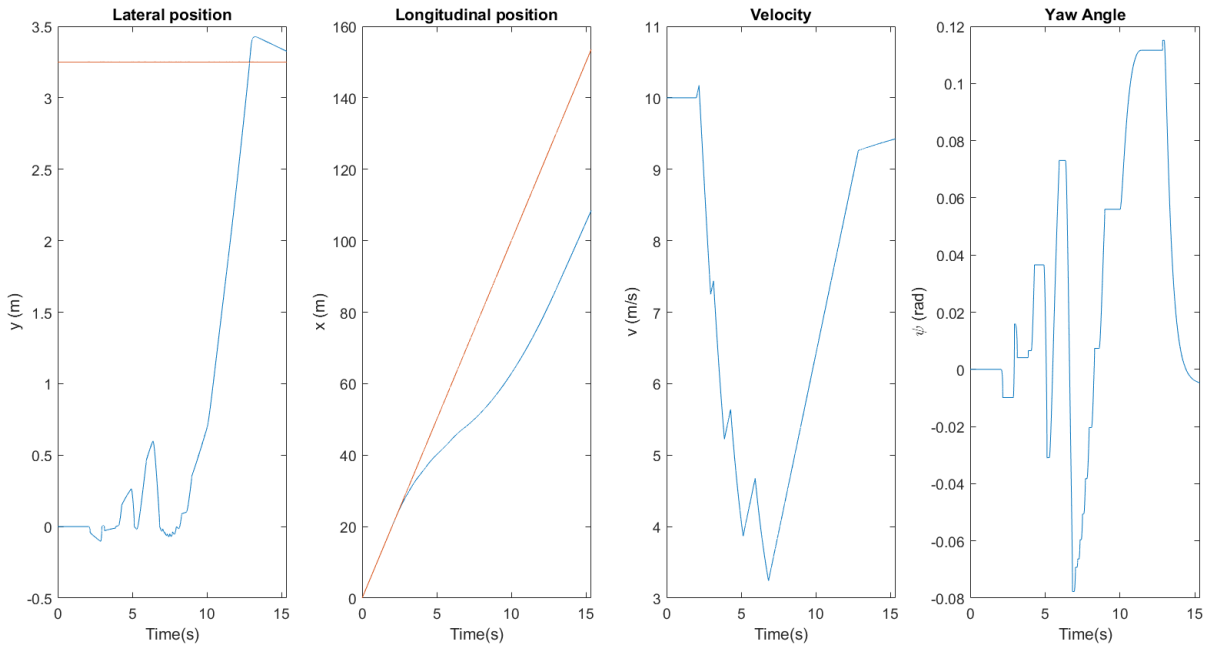
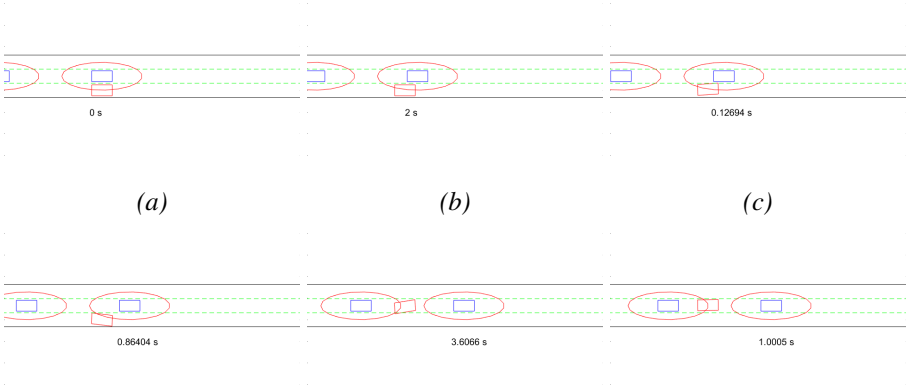


Figure 4.9: States of the Ego Vehicle

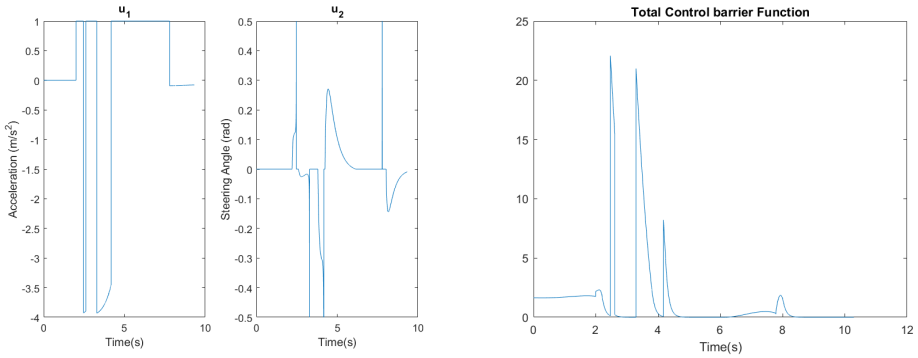
When the distance between the traffic vehicles is increased to 32.8m, i.e equal to 4 times the length of the ego vehicle and the traffic now moves at a constant speed of 12.5m/s, figure 4.10 shows that the controller is now able to safely execute the lane change maneuver without waiting for all the vehicles to pass and the ego vehicle fits in-between the two traffic vehicles. The plot for inputs, CBF and states of the ego vehicle for this case is shown in figure 4.8a, 4.8b and 4.9.



(a) *(b)* *(c)*

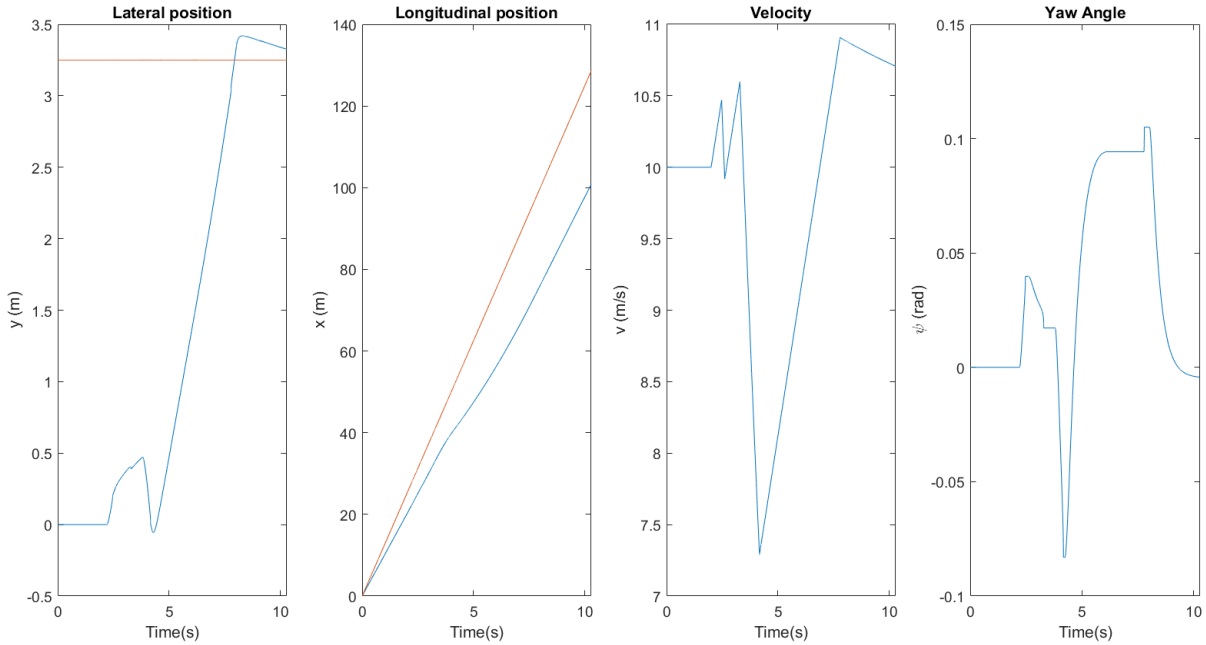
(d) *(e)* *(f)*

Figure 4.10: (a). The initial configuration, when the ego vehicle is moving at speed of $10m/s$ *and traffic vehicles also at* $12.5m/s$*. The space between the traffic vehicles is 4 times that of length of ego vehicle (b). The ego vehicle follows its own lane for first 2s (c), (d). Keeps trying to change the lane (e). Final attempt at the primary objective of changing lane is successful and completes within 4s. (f). Aligns itself with the lane*



(a) Inputs to the Ego Vehicle *(b) Value of the Control Barrier Function*

Figure 4.11



4.2 Experiments with Actual Robots

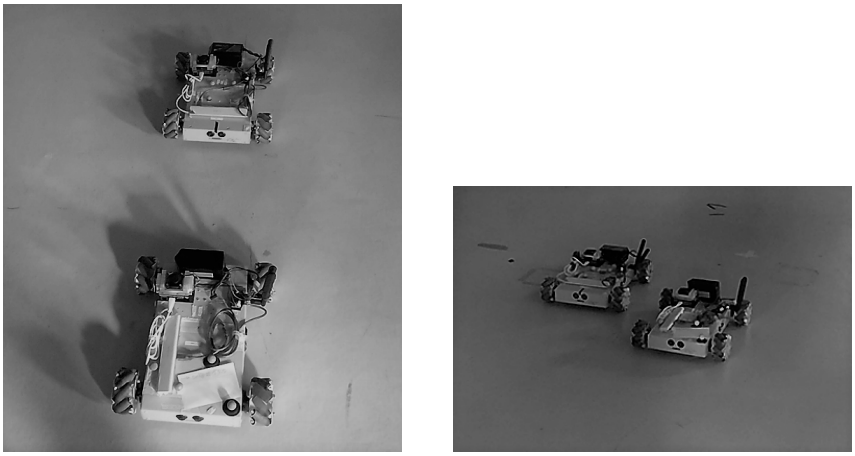


Figure 4.12: 4WD mecanum wheel Nex Robots used in the experiments

The validity of the method in the real world is put to test by using actual robots to fulfill the task described in the previous section, i.e changing lane, while

being able to guarantee safety. The deviation from the ideal scenario that can happen in real world is also observed. The setup uses 2 4WD Mecanum Wheel Mobile Arduino Robotics Car 10011 from Nexus Robots, whose positions are tracked with the help of a MoCap system and are fed instructions from a central controller. The robot's dynamics are modelled by single-integrator holonomic model. Thus, constraints now need to be defined on the states $[x, y]$. The robots have inbuilt velocity controllers which can be fed with instructions wirelessly from another controller within Wifi connectivity range. While the task and sub-tasks to be fulfilled are same as the previous section, the actual STL formulas need to be re-defined to accommodate the change in the model of the vehicle.

The experiments were conducted at Smart Mobility Lab [1] at KTH, using the Nexus robots. But due to limited availability of space to perform experiments and hardware limitations of having only 2 robots available, only the simpler versions of the experiments could be performed.

The predicates for STL formulas for the sub-tasks for this case become:

- To maintain the vehicle along the center of lane 1

$$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\left(|y - y_{lane1}| < k_1\right)$$

- To move towards the center of lane 2 within the next t_2 seconds

$$\boldsymbol{\mu_2} = F_{[t'',t''+t_2]}\left(|y - y_{lane2}| < k_1\right)$$

- To always avoid collision with the oncoming traffic vehicles

$$\boldsymbol{\mu_3} = G_{[0,\infty]}\sum_{i}^{n}\left[\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 > 1\right]$$

- Alternate Objective - To slow down so as to let faster moving vehicle pass

$$\boldsymbol{\mu_{stay}} = F_{[t''',t'''+t_3]}(\dot{x} = 0 \wedge \dot{y} = 0)$$

From the predicates, the actual STL formulas (weighted) for the respective sub-tasks can be formulated as follows:

$$\begin{aligned}\boldsymbol{\phi_{m1}} &= \left(c_1.\boldsymbol{\mu_1} \wedge c_2.\boldsymbol{\mu_3}\right) \vee c_{stay}.\boldsymbol{\mu_{stay}} \\ \boldsymbol{\phi_{m2}} &= \left(c_2.\boldsymbol{\mu_2} \wedge c_3.\boldsymbol{\mu_3}\right) \vee c_{stay}.\boldsymbol{\mu_{stay}}\end{aligned} \tag{4.4}$$

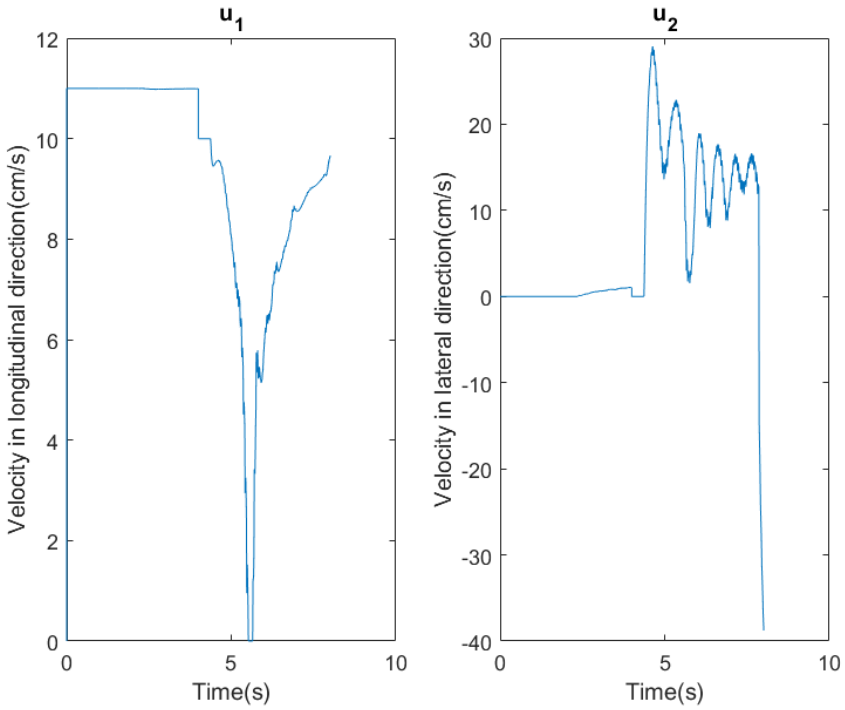
[1] https://www.kth.se/sv/dcs/research/control-of-transport/smart-mobility-lab

The controller employs $\boldsymbol{\phi_{m1}}$ and $\boldsymbol{\phi_{m2}}$ sequentially to derive the feedback law $\boldsymbol{u}(\boldsymbol{x}, t)$.

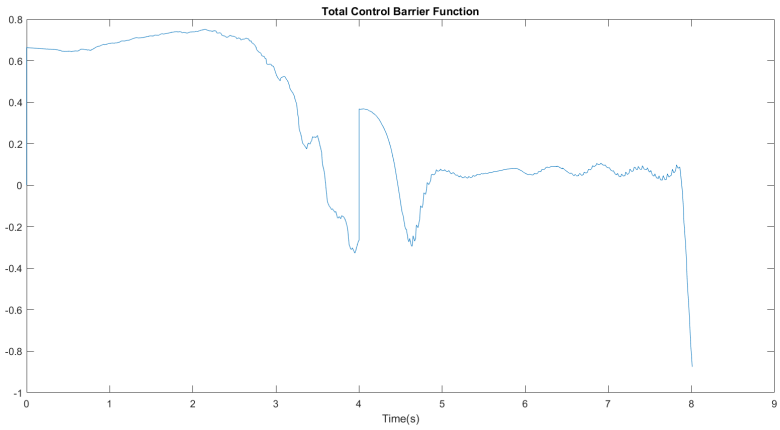
The value of constants and weights used in the predicates and STL formulas is presented in Appendix B in a concise manner. A general observation that can be mentioned here is that the magnitude of weight for the real world cases can be significantly less than what can be used for virtual simulations. This is due to large step size of controller and low agility of the robots.

The update rate of commands given to the robots cannot be more than 100Hz, which is much less than the simulation where step size of 10^{-8}s or lesser was used. The effect on the methods performance due to inertia of the robots, the delay in receiving feedback from MoCap system, and of the general noise in the system is seen here.

Two cases are tested: when the collision is imminent and when it is not. For the former case, the traffic velocity is set at 15cm/s and for the latter case it is 20cm/s. The plots for inputs and value of the CBF for each case is shown in the figure 4.13 and 4.14. The link for the video of the two experiments can be found at `https://youtu.be/dgwerxm-1Rc`.

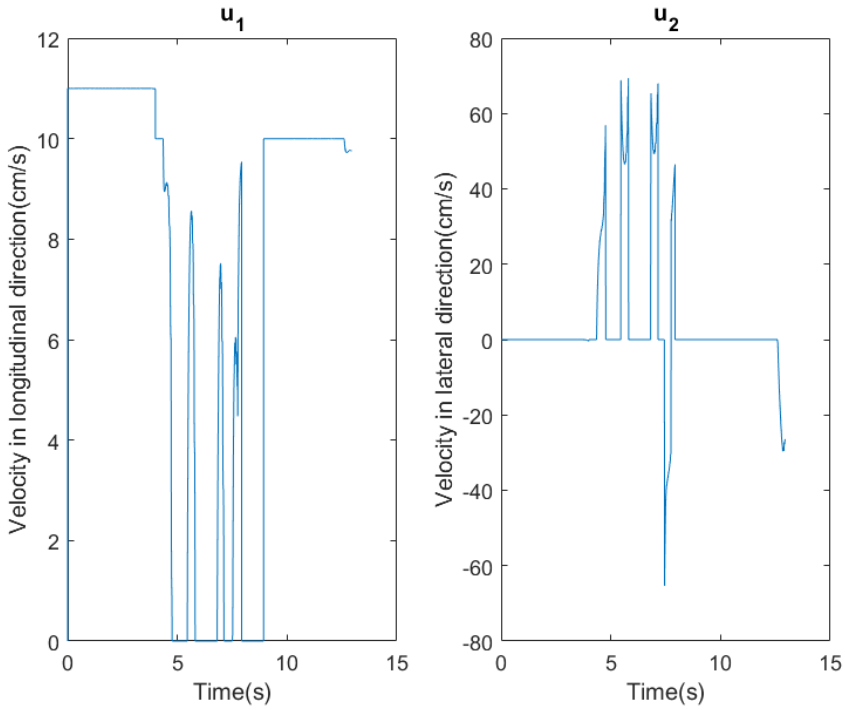


(a) Inputs to the Ego Vehicle

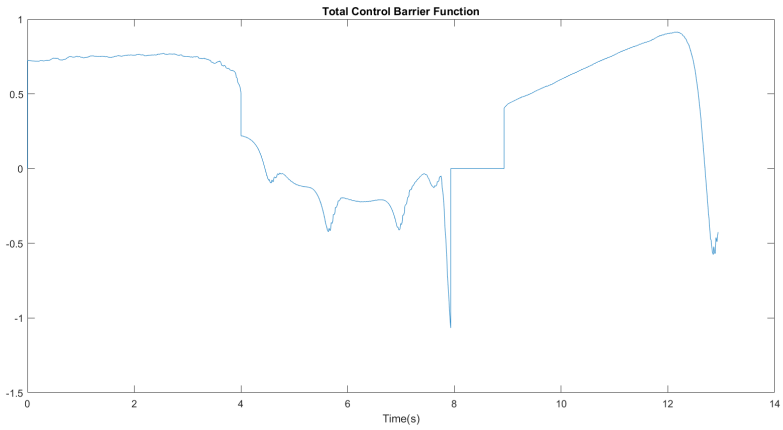


(b) Value of the Control Barrier Function

Figure 4.13: For the case traffic vehicle has constant speed of 20*cm/s and the ego vehicle has average speed of around* 10.5*cm/s*



(a) Inputs to the Ego Vehicle



(b) Value of the Control Barrier Function

Figure 4.14: For the case when traffic vehicle has constant speed of 15*cm/s and the ego vehicle has average speed of around* 10.5*cm/s*

The value of CBF can be seen going below zero at the switching instants which, as explained previously, is affected by the controller's update rate, weights used to define the STL formulas and also due to inertia of the robots in real life. Since the total CBF is an under-approximation of the element-wise

minimum function, the constituent predicate signals making up the total CBF remain positive and thus keep the vehicle safe. This can be improved if a lower step size could be used.

4.3 Coordinating with a Platoon while Overtaking Traffic Vehicles

The single-integrator holonomic model is adopted for this scenario. Thus, the STL constraints are now defined over the states $[x, y]$.

As pointed out in [7], the longer STL formulas for complex tasks can end up giving very conservative results, so the parts of the formula which are not relevant need to be suppressed. Here, the task is broken into several sub-tasks so as to activate only the relevant constraints for a specified time frame.

Overtaking vehicle(s) in the front to merge with a platoon in the same lane can be broken into several sub-tasks for which the predicates can be chosen as below. The figure 4.15a can be referred to understand the initial configuration.

- **Sub-task 1 -**
  - The distance between ego and i_{th} vehicle is less than k meters after $t_1$seconds
    $$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\left(|x - x_i| < k\right)$$
  - To move the ego vehicle into the left lane to prepare for overtaking
    $$\boldsymbol{\mu_2} = F_{[t',t'+t_1]}\left(|y - y_{lane1}| < k_1\right)$$
- **Sub-task 2 -**
  - To move the ego vehicle in front of i_{th} vehicle by k_2 metres between their centers
    $$\boldsymbol{\mu_3} = F_{[t'',t''+t_2]}\left(x > x_i + k_2\right)$$
  - To move the ego vehicle back into the right (original) lane
    $$\boldsymbol{\mu_4} = F_{[t'',t''+t_2]}\left(|y - y_{lane2}| < k_1\right)$$
- **Sub-task 3 -**

– To maintain platoon-worthy distance between ego vehicle and the platoon

$$\boldsymbol{\mu_5} = F_{[t''',t'''+t_3]}\left(x_p \leq x + k_3\right)$$

– To maintain the vehicle in the same lane

$$\boldsymbol{\mu_6} = F_{[t''',t'''+t_3]}\left(|y - y_{lane2}| < k_1\right)$$

Apart from the eventual constraints mentioned above, the following constraints for collision avoidance always need to be followed.

- To avoid collision with the platoon in the front

$$\boldsymbol{\mu_7} = G_{[0,\infty]}\left(\left[\frac{x - x_p}{a_1}\right]^2 + \left[\frac{y - y_p}{b_1}\right]^2 > 1\right)$$

- To always avoid collision with the oncoming traffic vehicles

$$\boldsymbol{\mu_8} = G_{[0,\infty]}\sum_{i}^{n}\left[\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 > 1\right]$$

- Modified ellipse boundary for collision avoidance while merging with the platoon

$$\boldsymbol{\mu_9} = F_{[t''',t'''+t_3]}\left(\left[\frac{x - x_p}{a_2}\right]^2 + \left[\frac{y - y_p}{b_2}\right]^2 > 1\right)$$

where the symbols used have same meaning as the constraints defined in 4.1. Besides them, (x_p, y_p) refer to the position of the center of the target platoon. $k, k_1, k_2, k_3, a, a_1, a_2, b, b_1, b_2$ are all constants that can be defined according to desired safety/performance margins.

In this case though, the platoon and the ego vehicle have to coordinate. So, a controller for platoon is also required that assists in the keeping the STL formulas satisfied which involves states of both platoon and the ego vehicle.

The composite predicate for this case, combined with weights as explained previously, for the ego vehicle's controller becomes:

$$\begin{aligned} \phi_{m1} &= c_1.\boldsymbol{\mu_1} \wedge c_2.\boldsymbol{\mu_2} \wedge c_8.\boldsymbol{\mu_8} \\ \phi_{m2a} &= c_3.\boldsymbol{\mu_3} \wedge c_4.\boldsymbol{\mu_4} \wedge c_8.\boldsymbol{\mu_8} \text{ and } \phi_{m2b} = c_7.\boldsymbol{\mu_7} \\ \phi_{m3a} &= c_6.\boldsymbol{\mu_6} \wedge c_8.\boldsymbol{\mu_8} \text{ and } \phi_{m3b} = c_5.\boldsymbol{\mu_5} \wedge c_9.\boldsymbol{\mu_9} \end{aligned} \tag{4.5}$$

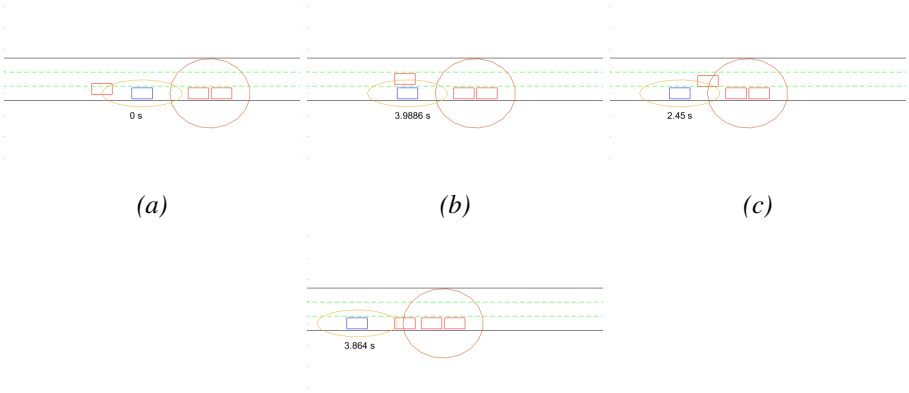
where ϕ_{m2} and ϕ_{m3} are further divided into coupled and non-coupled STL formulas. ϕ_{m2b} and ϕ_{m3b} being the coupled formulas, their corresponding coupled CBF are distributed among the ego vehicle and platoon as per the section 3.3.1. The same coupled STL formulas need to be satisfied by the platoon as well. Thus, the composite predicates for the platoon's controller become:

$$\begin{aligned} \phi_{m2b} &= c_7.\boldsymbol{\mu_7} \\ \phi_{m3b} &= c_5.\boldsymbol{\mu_5} \wedge c_9.\boldsymbol{\mu_9} \end{aligned} \tag{4.6}$$

The CBF for each of the above STL formulas can be defined in the way similar to sec 4.1. The actual constants and signals used to simulate this scenario and achieve results for this thesis work have been provided in Appendix C.

ϕ_{m1}, ϕ_{m2} and then ϕ_{m3} are sequentially processed by the ego vehicle's controller. Note that constraints represented by both ϕ_{m2a} and ϕ_{m2b} are now satisfied simultaneously. Similarly for ϕ_{m3a} and ϕ_{m3b}. At the corresponding times, the platoon's controller solves for ϕ_{m2b} and ϕ_{m3b}.

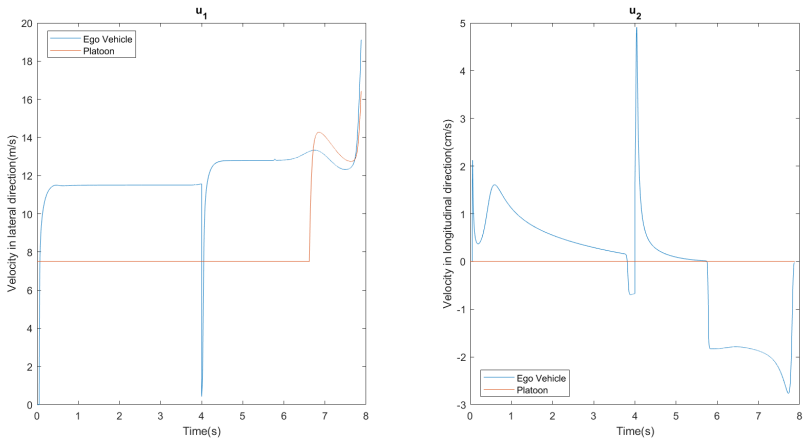
Figure 4.15 shows how the ego vehicle and platoon coordinate in this scenario. The figure 4.16b and 4.16a shows the value of respective CBF and calculated inputs for ego vehicle and platoon. Appendix C lists the weights and hyperparameters in detail, that are used to devise the controller used to achieve these results.



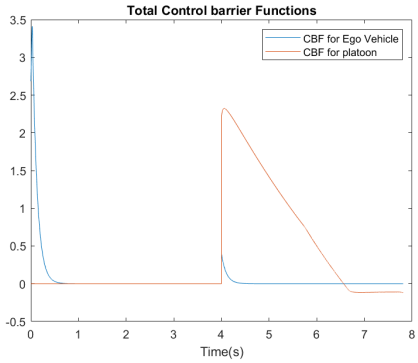
(a) *(b)* *(c)*

(d)

Figure 4.15: (a).The initial configuration, traffic and the platoon are moving at 7.5m/s. The ego vehicle starts from behind the traffic(red) vehicle (b).The ego vehicle changes lane to prepare to overtake, within 4s (c). Overtakes the lane and starts coordinating with the platoon (d). The platoon accelerates forward to make space for successful re-entry into the original lane and to help ego vehicle merge with itself.



(a) Inputs to the Ego Vehicle and platoon



(b) Value of the Control Barrier Function

Figure 4.16

Chapter 5

Discussions and Conclusions

A convex optimization controller based on Signal Temporal Logic framework that can be used in a highway driving scenario has been presented and the results achieved demonstrate the ability of the method. In particular, the system requirements are encoded into STL semantics and the STL formulas are transformed into valid time-varying Control Barrier Functions. The resulting CBF are then used as constraints to find an optimal control feedback law which satisfies the time and performance requirements set by the STL formulas. It was shown how to adapt the method for controlling a vehicle which can maneuver around lanes in traffic, overtake and perform coordinated tasks with other similar vehicles. The focus while presenting these examples was on a dynamic and heterogeneous environment with mixed agents, but the same framework can find numerous other applications in both static and dynamic environments.

The method presents numerous advantages. To begin with, it is easy to express safety and robustness in terms of value of CBF function. Besides, time based guarantees can now be achieved by devising the STL signals for the tasks and then representing them as time-varying CBF's. Since the final optimization problem to find the control input law for our agent is convex, the method is not at all calculation intensive and does not demand huge computation power. The method can be fairly general within certain limits, depending upon how the CBF is formulated from the STL signals and what STL signals are chosen. The controller can be tuned as per designer's requirements. Different weights which give priorities to different tasks, and constants which change the rate at which a task is attempted, can make a controller more aggressive or more conservative as desired.

This thesis work also aimed at verifying the feasibility of the method in a realistic scenario. So as a first step towards it, the experiments at Smart Mo-

bility Lab, KTH were conducted using the Nexus Robotics 4-WD mecanum drive robots. The experimental results have also been provided in the sections above. All the scenarios/examples discussed in theory here could not be tested due to limited availability of space and hardware. Also, the method gets highly limited by actual hardware capabilities, for example the delay in communication, the sampling rate of the controllers involved, inertia of the system etc will determine how effective our method will be and as a result the weights and other hyper-parameters have to be tuned accordingly.

However, the proposed methodology does have some limitations too. The framework handles the tasks in a serial order, so it might not be always possible to easily breakdown every application into sequential sub-tasks. Coming up with relevant STL signals which correspond to completion of the task and safety of the vehicle is usually not the toughest part, but the designer might still have to make some permutations and combinations with different kinds of signals/polynomials before finding the one whose behaviour suits the application. While having hyperparameters to adjust the controller's performance as desired is usually a property to be exploited, but it sometimes proves troublesome due to tuning difficulties. Without having a systematic way of finding them, most of the time random combinations have to be tried.

In conclusion, the method makes some underlying assumptions about the model of the vehicle, uncertainties and delay in communications (for multi-agent control cases), some are reasonable for standard highway situation and others to keep the optimization problem tractable, but is finally able to achieve the objectives of this thesis. However, as seen in the experimental results, the validity of these assumptions need to be further explored in real-world scenarios.

Chapter 6

Future Work

The theoretical framework for trajectory planning and platoon coordination for an autonomous vehicle has been presented and tested in computer simulations. While one of the scenarios has been tested in the real world, with satisfactory results, more real-world tests need to be done.

The method presents great versatility to be able to be successfully used in various applications given its low computation power requirement and fairly easy convex controller design, while guaranteeing the time and safety bounds for the tasks. The saved bandwidth of the controller can then be utilized in some other task.

Several improvements can be further investigated for the future continuation of the work. Firstly, the model of the vehicle used can be made more realistic by including the unknown disturbances and uncertainties. The delay in communication between two agents while coordinating among themselves should also be modelled and included in the framework. Another idea is to explore the adaptive time-varying CBF. As can be seen in results from different simulations, there are cases when switching to secondary STL formula needs to take place, for example in case environmental conditions do not allow primary task to progress further. When the controller switches back to the primary STL formula, it may take a while before CBF value actually begins to correspond to how much of the task has been completed, since it is a constant function that was fixed in advance while designing the controller. The ego vehicle is essentially unreactive in that duration. While the current static method works, making the CBF adaptive to percentage of completion depending on current state of the vehicle can further improve the performance of the method.

Bibliography

[1] A. Alam et al. “Heavy-Duty Vehicle Platooning for Sustainable Freight Transportation: A Cooperative Method to Enhance Safety and Efficiency”. In: *IEEE Control Systems Magazine* 35.6 (2015), pp. 34–56.

[2] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008. ISBN: 026202649X.

[3] Oded Maler and Dejan Nickovic. “Monitoring Temporal Properties of Continuous Signals”. In: *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Ed. by Yassine Lakhnech and Sergio Yovine. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 152–166. ISBN: 978-3-540-30206-3.

[4] Mo Chen et al. “Signal Temporal Logic Meets Reachability: Connections and Applications”. In: May 2020, pp. 581–601. ISBN: 978-3-030-44050-3. DOI: 10.1007/978-3-030-44051-0_34.

[5] Anders Forsgren, Philip E. Gill, and Margaret H. Wright. “Interior Methods for Nonlinear Optimization”. In: *SIAM Review* 44.4 (2002), pp. 525–597. ISSN: 00361445. URL: http://www.jstor.org/stable/4148314.

[6] Keng Peng Tee, Shuzhi Sam Ge, and Eng Hock Tay. “Barrier Lyapunov Functions for the control of output-constrained nonlinear systems”. In: *Automatica* 45.4 (Apr. 2009), pp. 918–927. DOI: 10.1016/j.automatica.2008.11.017.

[7] Lars Lindemann and Dimos Dimarogonas. “Control Barrier Functions for Signal Temporal Logic Tasks”. In: *IEEE Control Systems Letters* PP (July 2018), pp. 1–1. DOI: 10.1109/LCSYS.2018.2853182.

[8] Xiangru Xu. “Constrained control of input–output linearizable systems using control sharing barrier functions”. In: *Automatica* 87 (Jan. 2018), pp. 195–201. DOI: 10.1016/j.automatica.2017.10.005.

[9] Li Wang, Aaron Ames, and Magnus Egerstedt. “Safety Barrier Certificates for Heterogeneous Multi-Robot Systems”. In: July 2016, pp. 5213–5218. DOI: 10.1109/ACC.2016.7526486.

[10] Aaron Ames, Jessy Grizzle, and Paulo Tabuada. “Control barrier function based quadratic programs with application to adaptive cruise control”. In: Dec. 2014.

[11] A. D. Ames et al. “Control Barrier Function Based Quadratic Programs for Safety Critical Systems”. In: *IEEE Transactions on Automatic Control* 62.8 (2017), pp. 3861–3876.

[12] Aaron D. Ames et al. *Control Barrier Functions: Theory and Applications*. 2019. arXiv: 1903.11199.

[13] J. Chai and R.G. Sanfelice. “On Notions and Sufficient Conditions for Forward Invariance of Sets for Hybrid Dynamical Systems”. In: *Proceedings of the 54th IEEE Conference on Decision and Control*. Dec. 2015, pp. 2869–2874.

[14] “Chapter 1 - Basic Concepts”. In: *Adaptive Sliding Mode Neural Network Control for Nonlinear Systems*. Ed. by Yang Li, Jianhua Zhang, and Qiong Wu. Emerging Methodologies and Applications in Modelling. Academic Press, 2019, pp. 1–16. ISBN: 978-0-12-815372-7. DOI: 10.1016/B978-0-12-815372-7.00001-X.

[15] “Chapter-4 Lyapunov Stability; Section 4.4”. In: *Non Linear Systems*. Ed. by H.K Khalil. Prentice Hall, 2001.

[16] Franco Blanchini and Stefano Miani. *Set-Theoretic Methods in Control*. Jan. 2007. ISBN: 9780817632557. DOI: 10.1007/978-0-8176-4606-6.

[17] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. DOI: 10.1017/CBO9780511804441.

[18] Philip Polack et al. “The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?” In: June 2017, pp. 812–818. DOI: 10.1109/IVS.2017.7995816.

[19] Filip Klaesson and John Friberg. *Autonomous Overtaking Using Reachability Analysisand MPC*. 2018.

Appendix A

Constants used to derive controller for Sim.1

For the Changing lane on a Highway - Simulation :

The STL signals used are :

$$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\left(|y - y_{lane1}| < k_1\right)$$
$$\boldsymbol{\mu_2} = F_{[t'',t''+t_2]}\left(|y - y_{lane2}| < k_1\right)$$
$$\boldsymbol{\mu_3} = G_{[0,\infty]}\sum_{i}^{n}\left[\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 > 1\right]$$
$$\boldsymbol{\mu_{stay}} = F_{[t''',t'''+t_3]}(\dot{v} < 0 \wedge |y - y_{lane1}| < k_{alt1})$$

,where

$$k_1 = 0.1 \quad k_{alt1} = 0.7$$
$$a = 16 \quad b = 3.2$$
$$y_{lane1} = 0 \quad y_{lane2} = 3.25$$
$$t_1 = 4 \quad t_2 = 4 \quad t_3 = 1$$

The combined weighted STL formula used is :

$$\boldsymbol{\phi_{m1}} = (c_1.\boldsymbol{\mu_1} \wedge c_{3a}.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}}$$
$$\boldsymbol{\phi_{m2}} = (c_2.\boldsymbol{\mu_2} \wedge c_{3b}.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}}$$

, where $c_1 = 1$, $c_2 = 2$, $c_{3a} = 5$, $c_{3b} = 20$ and $c_{stay} = 1$

Corresponding time-varying CBF for each of the STL predicates are :

$$b_1 = (\gamma_1^2(t) - (y - y_{lane1})^2) \quad \text{where } \gamma_1 = -\frac{3}{3.5}t + 3.5$$

$$b_2 = (\gamma_2^2(t) - (y - y_{lane2})^2) \quad \text{where } \gamma_1 = -\frac{3.49}{4}t + 3.5$$

$$b_3 = \left(\left[\frac{x - x_i}{a} \right]^2 + \left[\frac{y - y_i}{b} \right]^2 - \gamma_3(t) \right) \quad \text{where } \gamma_3 = -0.1e^{(-0.5t-1.2)} - 1$$

$$b_{alt1} = c_{alt1}(\gamma_{alt1}^2(t) - (v - \hat{v})^2) \quad \text{where } \gamma_{alt1} = -4.5t + 11,\ \hat{v} = 1,\ c_{alt1} = 10$$

$$b_{alt2} = c_{alt2}(\gamma_{alt2}^2(t) - (y - y_{lane1})^2) \quad \text{where } \gamma_{alt2} = -\frac{3.24}{4}t + 1.5,\ c_{alt2} = 10$$

The final CBF from the conjunction of the individual CBF's above is found by using the formulas below,

$$b_{m1}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_1 b_1} + \mathrm{e}^{-c_{3a} b_3})$$
$$b_{m2}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_2 b_2} + \mathrm{e}^{-c_{3b} b_3})$$
$$b_{stay}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_{alt1} b_{alt1}} + \mathrm{e}^{-c_{alt2} b_{alt2}})$$

Use eq 3.9 to find the input feedback law.

Appendix B

Constants used to derive controller for Exp.1

For the Changing lane - Experiment with Actual robots:

The STL signals used are :

$$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\ (|y - y_{lane1}| < k_1)$$
$$\boldsymbol{\mu_2} = F_{[t'',t''+t_2]}\ (|y - y_{lane2}| < k_1)$$
$$\boldsymbol{\mu_3} = G_{[0,\infty]} \sum_{i}^{n} \left[\left[\frac{x - x_i}{a} \right]^2 + \left[\frac{y - y_i}{b} \right]^2 > 1 \right]$$
$$\boldsymbol{\mu_{stay}} = F_{[t''',t'''+t_3]} (\dot{x} = 0 \wedge \dot{y} = 0)$$

,where

$$k_1 = 1$$
$$a = 78 \quad b = 50$$
$$y_{lane1} = 0 \quad y_{lane2} = 52$$
$$t_1 = 4 \quad t_2 = 4 \quad t_3 = 1$$

The combined weighted STL formula used is :

$$\boldsymbol{\phi_{m1}} = (c_1.\boldsymbol{\mu_1} \wedge c_{3a}.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}}$$
$$\boldsymbol{\phi_{m2}} = (c_2.\boldsymbol{\mu_2} \wedge c_{3b}.\boldsymbol{\mu_3}) \vee c_{stay}.\boldsymbol{\mu_{stay}}$$

, where $c_1 = 2$, $c_2 = 0.5$, $c_{3a} = 2$, $c_{3b} = 0.8$ and $c_{stay} = 1$

Corresponding time-varying CBF for each of the STL predicates are :

$$b_1 = (\gamma_1(t) - (y - y_{lane1})) \quad \text{where } \gamma_1 = -\frac{2.5}{4}t + 3.5$$

$$b_2 = (\gamma_2(t) - (y - y_{lane2})) \quad \text{where } \gamma_1 = -\frac{59.5}{4}t + 60$$

$$b_3 = \left(\left[\frac{x - x_i}{a} \right]^2 + \left[\frac{y - y_i}{b} \right]^2 - \gamma_3(t) \right) \quad \text{where } \gamma_3 = -0.1e^{(-0.5t-1.2)} - 1$$

The final CBF from the conjunction of the individual CBF's above is found by using the formulas below,

$$b_{m1}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_1 b_1} + \mathrm{e}^{-c_{3a} b_3})$$
$$b_{m2}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_2 b_2} + \mathrm{e}^{-c_{3b} b_3})$$

Use eq 3.9 to find the input feedback law.

Appendix C

Constants used to derive controller for Sim.2

For Coordinating with a platoon while overtaking traffic vehicles :

The STL signals used are :

$$\boldsymbol{\mu_1} = F_{[t',t'+t_1]}\left(|x - x_i| < k\right)$$
$$\boldsymbol{\mu_2} = F_{[t',t'+t_1]}\left(|y - y_{lane1}| < k_1\right)$$
$$\boldsymbol{\mu_3} = F_{[t'',t''+t_2]}\left(x > x_i + k_2\right)$$
$$\boldsymbol{\mu_4} = F_{[t'',t''+t_2]}\left(|y - y_{lane2}| < k_1\right)$$
$$\boldsymbol{\mu_5} = F_{[t''',t'''+t_3]}\left(x_p \leq x + k_3\right)$$
$$\boldsymbol{\mu_6} = F_{[t''',t'''+t_3]}\left(|y - y_{lane2}| < k_1\right)$$
$$\boldsymbol{\mu_7} = G_{[0,\infty]}\left(\left[\frac{x - x_p}{a_1}\right]^2 + \left[\frac{y - y_p}{b_1}\right]^2 > 1\right)$$
$$\boldsymbol{\mu_8} = G_{[0,\infty]}\sum_{i}^{n}\left[\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 > 1\right]$$
$$\boldsymbol{\mu_9} = G_{[0,\infty]}\left(\left[\frac{x - x_p}{a_2}\right]^2 + \left[\frac{y - y_p}{b_2}\right]^2 > 1\right)$$

,where

$$k = 1 \quad k_1 = 0.1 \quad k_2 = 20 \quad k_3 = 12.55$$
$$a = 16 \quad b = 3.1 \quad a_1 = 16 \quad b_1 = 8 \quad a_2 = 13.2 \quad b_2 = 10$$
$$y_{lane1} = 3.25 \quad y_{lane2} = 0$$
$$t_1 = 4 \quad t_2 = 4 \quad t_3 = 2.5$$

The combined weighted STL formula used is :

$$\phi_{m1} = c_1.\boldsymbol{\mu_1} \wedge c_2.\boldsymbol{\mu_2} \wedge c_{8a}.\boldsymbol{\mu_8}$$
$$\phi_{m2a} = c_3.\boldsymbol{\mu_3} \wedge c_4.\boldsymbol{\mu_4} \wedge c_{8b}.\boldsymbol{\mu_8} \;\&\; \phi_{m2b} = c_7.\boldsymbol{\mu_7}$$
$$\phi_{m3a} = c_6.\boldsymbol{\mu_6} \wedge c_{8b}.\boldsymbol{\mu_8} \;\&\; \phi_{m3b} = c_5.\boldsymbol{\mu_5} \wedge c_9.\boldsymbol{\mu_9}$$

, where $c_1 = 10$, $c_2 = 100$, $c_3 = 1$, $c_4 = 10$, $c_5 = 1$, $c_6 = 1$, $c_7 = 1$, $c_{8a} = 20$, $c_{8b} = 10$ $c_9 = 1$

Corresponding time-varying CBF for each of the STL predicates are :

$$b_1 = (\gamma_1(t) - |x - x_i|) \quad \text{where } \gamma_1 = -\frac{16.02}{4}t + 17$$
$$b_2 = (\gamma_2(t) - |y - y_{lane1}|) \quad \text{where } \gamma_2 = -\frac{2.9}{4}t + 3$$
$$b_3 = (x - (x_i + k_2) - \gamma_3(t)) \quad \text{where } \gamma_1 = \frac{21.2}{4}t - 43$$
$$b_4 = (\gamma_4(t) - |y - y_{lane2}|) \quad \text{where } \gamma_4 = -\frac{3.6}{2}t + 14.5$$
$$b_5 = (\gamma_5(t) - (x_p - x - k_3)) \quad \text{where } \gamma_1 = -\frac{2.9}{2}t + 14.6$$
$$b_6 = (\gamma_6(t) - |y - y_{lane2}|) \quad \text{where } \gamma_6 = -\frac{0.4}{2}t + 2.1$$
$$b_7 = \left(\left[\frac{x - x_p}{a_1}\right]^2 + \left[\frac{y - y_p}{b_1}\right]^2 - \gamma_7(t)\right) \quad \text{where } \gamma_7 = -0.1e^{(-0.5t-1.2)} - 1$$
$$b_8 = \left(\left[\frac{x - x_i}{a}\right]^2 + \left[\frac{y - y_i}{b}\right]^2 - \gamma_8(t)\right) \quad \text{where } \gamma_8 = -0.1e^{(-0.5t-1.2)} - 1$$
$$b_9 = \left(\left[\frac{x - x_p}{a_2}\right]^2 + \left[\frac{y - y_p}{b_2}\right]^2 - \gamma_9(t)\right) \quad \text{where } \gamma_9 = -0.1e^{(-0.5t-1.2)} - 1$$

The final CBF from the conjunction of the individual CBF's above is found by using the formulas below,

$$b_{m1}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_1 b_1} + \mathrm{e}^{-c_2 b_2})$$
$$b_{m2a}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_3 b_3} + \mathrm{e}^{-c_4 b_4} + \mathrm{e}^{-c_8 b_8})$$
$$b_{m2b}(\boldsymbol{x}, \boldsymbol{x_p} t) = c_7 b_7$$
$$b_{m3a}(\boldsymbol{x}, t) = -\log(\mathrm{e}^{-c_6 b_6} + \mathrm{e}^{-c_8 b_8})$$
$$b_{m3b}(\boldsymbol{x}, \boldsymbol{x_p}, t) = -\log(\mathrm{e}^{-c_5 b_5} + \mathrm{e}^{-c_9 b_9})$$

Use method described in sec 3.3.1 to find the control law for this case.

TRITA-EECS-EX-2020:755
www.kth.se