Verification of Controllers for Advanced Driver Assistance Systems

An M.Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Master of Technology

by

Manish Semwal
(204101032)

under the guidance of

Dr. Purandar Bhaduri

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Verification of Controllers for Advanced Driver Assistance Systems**" is a bonafide work of **Manish Semwal (Roll No. 204101032)**, carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Dr. Purandar Bhaduri**

Professor,
Department of Computer Science & Engineering,
Indian Institute of Technology Guwahati, Assam.

Jun, 2022
Guwahati.

Abstract

This report considers the problem of the lane-change maneuver for automated vehicles in dense traffic on highways. To be able to safely change lanes or evolve in the same lane, an automated vehicle must follow certain safety conditions which include the condition that no vehicle enters the critical zone of another vehicle and that there is proper synchronization among nearby vehicles before making a lane change, which could be achieved by using the claim-reserve paradigm for lane-change maneuver, in which the vehicles first make claims to the target lane and if there is no conflict of claims, they change their claim to reserve and perform lane change operation else, they withdraw their claim. Finally, these safety conditions were verified using theorem prover KeYmaera X which is a powerful tool to verify hybrid systems that are modeled using hybrid programs and differential dynamic logic.

Contents

List of Figures **1**

1 Introduction **4**

1.1 Motivation 5

1.2 Problem Statement 5

1.2.1 The multi-lane highway 6

1.2.2 The safety condition 7

1.3 Proposed Solution 7

2 Review of Existing Work **9**

2.1 Critical Zone 9

2.2 Critical Zone Modelling 10

2.2.1 Assumptions: 10

2.2.2 Acceleration profiles for the evasive manoeuvres 11

2.2.3 Leading vehicles in target lane 12

3 Compositional Verification of multi-lane traffic maneuver **13**

3.1 Modelling the dynamic behavior of a car 13

3.2 Verification of multi-lane traffic maneuvers 14

4 KeYmaera X : a tool for the verification of hybrid systems **15**

4.1 Differential dynamic logic 15

4.2 Hybrid programs 16

4.3 KeYmaera X 17

5 Verification of lane-change maneuvers using KeYmaera X 19

6 Conclusion 24

References 27

List of Figures

1.1 A multi-lane highway with several cars. The large rectangle shows the view of car A, i.e. the part of the environment visible to A[BHLO17]. 6

2.1 Traffic scenario with vehicles travelling in two lanes. Ego vehicle is shown in red. T1 and T2 are trailing vehicles. L1 and L2 are leading vehicles. Shaded region in red are the critical zones [CSB+17]. 10

2.2 The longitudinal acceleration profile for braking. The ego vehicle follows a constant jerk profile until time t_0, where it reaches the maximum deceleration limit, a_x^{max}. The vehicle follows a constant acceleration profile from time t_0 to final time t_f [CSB+17]. 11

2.3 The lateral acceleration profile for steering. The ego vehicle is modelled to follow a constant jerk profile until time t_0, where it reaches the maximum acceleration a_y^{max}. The ego vehicle follows a constant acceleration profile from time t_0 to t_1. From t_1 to t_2, it follows a deceleration profile with constant jerk until zero acceleration is achieved. Constant zero acceleration is maintained from time t_2 to final time t_f [CSB+17]. 11

2.4 Critical zone calculation for the leading vehicle when the ego vehicle brakes to avoid a collision. The leading vehicles crashes to a stop and the ego vehicle brakes to avoid collision, travelling a distance of S_{ego} before stopping [CSB+17]. 12

3.1 Behavior of a car A changing from its current lane n to a neighboring target lane m is either $n-1$ or $n+1$: **a** simple algorithm, **b** protocol with a claim transition $c(m)$ as in [BHLO17]. 14

5.1 A multi-lane highway with 1 as our ego vehicle as six cars 19

5.2 Snippet 1 - lane change maneuver in multi-lane highway modeled in KeYmaera X 21

5.3 Snippet 2 - lane change maneuver in multi-lane highway modeled in KeYmaera X 22

5.4 Lane change maneuver in multi-lane highway model verified in KeYmaera X 23

Chapter 1

Introduction

Cyber-Physical Systems have become a hot issue due to rapid advancements in science and technology. These automated technologies reduce human involvement in task completion and are gradually taking over human roles in a variety of fields. They perform flawlessly and entirely eliminate the difficulties caused by human mistakes, if they are designed and installed appropriately. Cyber-Physical Systems are used in a wide range of applications, from autopilots and autonomous vehicles to autonomous drones that could be used as weapons and pacemakers. As the number of these agents grows, they come into closer contact with humans, making it even more critical to ensure that they operate properly. Despite the fact that testing is commonly used to verify the effectiveness of other programming applications, it is very much evident that it is not sufficient to guarantee safety and reliability in the case of Cyber-Physical Systems. This is due to the fact that testing all possible scenarios that such a framework can encounter while running in the real world is impossible. When other measures, such as testing, are unlikely to suffice, formal verification provides us with techniques to verify the critical security aspects. These strategies are used by KeYmaera X to adequately show the safety features of hybrid systems.

Out of the mentioned Cyber-Physical Systems, our work is focused on autonomous vehicles, we try to verify the safety of vehicles in dense traffic on a multi-lane highway, for that we try to model and verify the safety of vehicles in a multi-lane highway scenario in theorem prover KeYmaera X, using differential dynamic logic and hybrid programs.

1.1 Motivation

As the development of the automated driving vehicles bring them closer to deployment, the importance of them being safe becomes extremely crucial. A study have shown that about 20 percent of the highway accidents involve lane change [NBCF15]. In an accident the driver is at highest risk of injuries and death. Therefore, for the increased traffic safety and reduction of human traffic deaths, a safe, smooth and efficient lane change is essential.

According to The United States Department of Transportation (USDOT), the rise of driverless cars will lead to 90 percent fall of traffic deaths. Autonomous vehicles would calculate headway (space) much accurately which will result in increased lane capacity. The traffic will be smoother and the congestion will be less, which will reduce travelling time, save fuel and reduce CO_2 emissions. Hence, travelling will be safer, cheaper and more eco-friendly.

The autonomous vehicles can help in reducing the incidents caused by human errors in driving a vehicle, they can bring down the incidents caused by drugged driving, speeding, and distractions could help people with disabilities, who can reduce their dependencies on other people a bit. They increase productivity as driving time can now be utilized.

1.2 Problem Statement

An automated vehicle needs to make decisions in complex environments of the traffic. Changing lanes on a highway is a sophisticated driving mode, since the vehicle has to adapt to the movements of the nearby vehicles. In a dense traffic, it could be difficult to find and select suitable gaps in the adjacent lanes and to perform a lane-change maneuver.

Primarily, our aim is to verify the safety conditions of automated vehicles in dense traffic on multi-lane highways, which include evolving safely in the current lane without entering the *critical zone* aka *safety envelope* of other vehicles and safe lane-change maneuvers i.e. to be able to change lane without collision with other vehicles as well as without entering the *critical zone* of other vehicles while lane-change maneuver. For verification, theorem prover $KeYmaera\ X$ is suggested.

1.2.1 The multi-lane highway

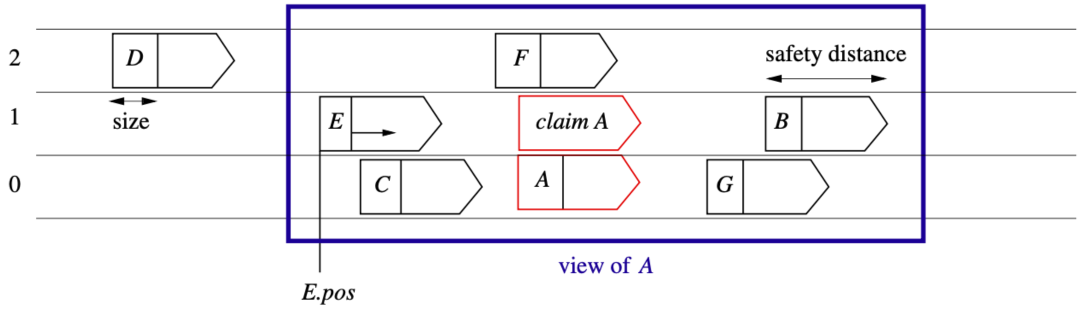


Fig. 1.1 A multi-lane highway with several cars. The large rectangle shows the view of car A, i.e. the part of the environment visible to A[BHLO17].

Consider a multi-lane highway scenario as shown in Fig. 1.1, where the lanes are numbered from 0 to 2, which act as a coordinate system in which the cars could evolve. Each car has a position along the road and a set of current lanes. Since the safety of a vehicle depends on its local environment, we consider the vehicles that are in the *view* of ego vehicle. Every vehicle is able to perceive its *view* using a set of sensors. Lastly, we assume all the cars on the highway follow the same protocol for lane-change maneuvers[BHLO17].

Each vehicle has an envelope around it which is often referred to as *safety envelope* or $criticalzone$ of a vehicle, it is that region around a vehicle in which no vehicle should ever enter to guarantee safety. Let the length of *safety envelope* be given by sd i.e. *safety distance*.

In Fig. 1.1, if car A in lane 0 and car F in lane 2 observe that their is space available in lane 1, adjacent to both A and F and try to change lanes simultaneously, a collision possible, which could be resolved using following set of features :

- *res*: The set of reserved lanes. It could have at most two elements, the current lane and the adjacent target lane, if vehicle wants to change lane.
- *clm*: The set of claimed lanes. It could have at most one element, the adjacent target lane, if vehicle wants to change lane.

1.2.2 The safety condition

The dangerous situation of a collision is formalized by the following condition:

$$col = \exists c_1, c_2 : (c_1 \neq c_2 \wedge (c_1.res \cap c_2.res \neq \phi) \wedge safetyOverlap(c_1, c_2)) \tag{1.1}$$

Where $safetyOverlap(c_1, c_2)$ is true if there is an overlap of critical zones also called $safety$ *envelope* i.e. overlap of regions of two different cars ranging from current position of the vehicle to current position + safety distance *sd*. The overlap can be formalized as:

$$\begin{aligned} safetyOverlap(c_1, c_2) =& (c_1.pos \leq c_2.pos \leq c_1.pos + c_1.sd) \\ & \vee (c_2.pos \leq c_1.pos \leq c_2.pos + c_2.sd) \end{aligned} \tag{1.2}$$

We say that the system is $safe$ if there is no overlap of $safety$ *envelope* of any two cars on any given lane, that is, if the collision condition *col* is false. Therefore, the safety condition is:

$$safe \equiv \neg col = \forall c_1, c_2 : (c_1 = c_2 \vee (c_1.res \cap c_2.res = \phi) \vee \neg safetyOverlap(c_1, c_2)) \tag{1.3}$$

1.3 Proposed Solution

To guarantee safety for lane-change maneuver by Advance Driver Assistant Systems in automated vehicles in dense traffic on multi-lane highways, we verified their safety conditions which include evolving safely in the current lane without entering the *critical zone* aka *safety envelope* of other vehicles and safe lane-change maneuvers i.e. to be able to change lane without collision with other vehicles as well as without entering the *critical zone* of other vehicles while lane-change maneuver. Our objective was to propose verification of the existing safety conditions for lane-change maneuvers in ADAS, these have been discussed in detail in Chapter 4 and 5.

Chapter 2

Review of Existing Work

For lane changes, one of the common assessment for safety of a vehicle is to consider a safety distance *sd* . This distance could be either based on a fixed time gap or time to collide. However, if our considered safety distance is conservative it would leave less space for maneuvering the vehicle in the traffic, on the other hand if we consider an optimistic estimate it could risk safety of the vehicles.

A lane change algorithm is expected to be safe and at the same time should minimize the safety distance so that there is more room for maneuverability. This will allow the vehicles to save travel time as, more maneuverability means the vehicle will be able to move faster, if required.

2.1 Critical Zone

Let us consider the traffic scenario depicted in Fig. 2.1. The figure depicts five vehicles where E is the *ego vehicle* (subject vehicle), L1 and L2 are the two leading vehicles and T1 and T2 are the trailing vehicles. L1 and T1 are in the *host lane* (current lane of the ego vehicle) whereas L2 and T2 are in the adjacent lane. Previous studies argue that the lane changes can be performed safely as long as the *time headway* (the time difference between any two successive vehicles when they cross a given point) to all surrounding vehicles always is kept sufficiently high [MS15] [NBCF15]. Such critical time headway zones are shown as shaded regions around the vehicles in Fig. 2.1.

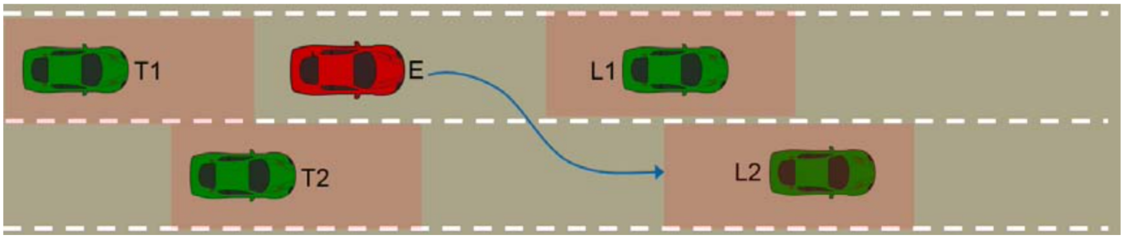


Fig. 2.1 Traffic scenario with vehicles travelling in two lanes. Ego vehicle is shown in red. T1 and T2 are trailing vehicles. L1 and L2 are leading vehicles. Shaded region in red are the critical zones [CSB+17].

The ego vehicle should not enter the *critical zone*, a region around each vehicle in which no other vehicle should enter to guarantee safety, of any vehicle at any point in time, to guarantee a collision free motion. The critical zones can be calculated by using a constant time gap as the safety indicator, which depends on the vehicle's deceleration capabilities, for some ego vehicle, the higher the leading vehicle's deceleration capabilities are the larger ego vehicle's critical zone would be, since if the leading vehicle applies brake suddenly, then the ego vehicle should be able to slow down or stop without collision. Here, a method to model the critical zone for an autonomous lane change manoeuvre is discussed, where the autonomous vehicle can make use of the ability to either *brake* or *steer* to avoid collisions [CSB+17].

2.2 Critical Zone Modelling

2.2.1 Assumptions:

- Linear motion models are used to model the motion of the ego vehicle and the surrounding vehicles.

- The critical zone calculated around the surrounding vehicles at every instant is based on an assumed worst case behaviour for the ego vehicle to handle and adapt.

- The ego vehicle uses specific longitudinal and lateral acceleration profile to plan the evasive manoeuvre in case of any worst case scenario[CSB+17].

2.2.2 Acceleration profiles for the evasive manoeuvres

The longitudinal and lateral acceleration profiles of the ego vehicle during the evasive manoeuvre are determined considering the best manoeuvring capability (steering or braking) of the ego vehicle to avoid collisions in case of emergency situations. The longitudinal and lateral acceleration profile are shown in Fig. 2.2 and Fig. 2.3.

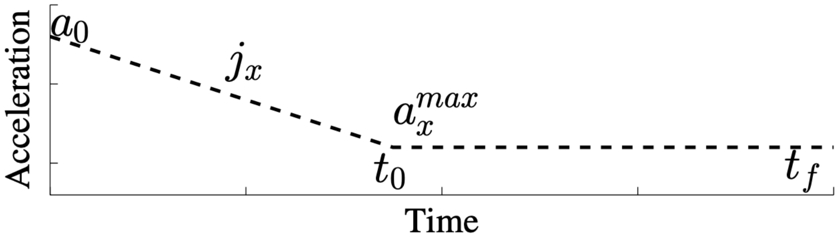


Fig. 2.2 The longitudinal acceleration profile for braking. The ego vehicle follows a constant jerk profile until time t_0, where it reaches the maximum deceleration limit, a_x^{max}. The vehicle follows a constant acceleration profile from time t_0 to final time t_f [CSB+17].

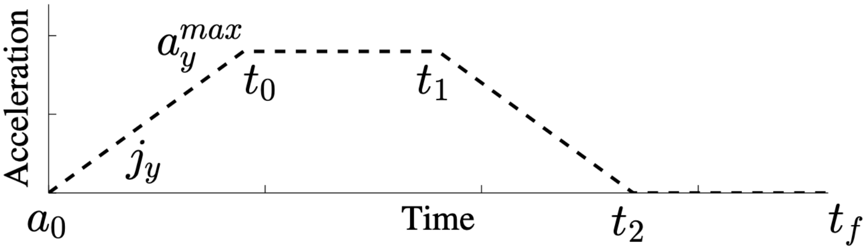


Fig. 2.3 The lateral acceleration profile for steering. The ego vehicle is modelled to follow a constant jerk profile until time t_0, where it reaches the maximum acceleration a_y^{max}. The ego vehicle follows a constant acceleration profile from time t_0 to t_1. From t_1 to t_2, it follows a deceleration profile with constant jerk until zero acceleration is achieved. Constant zero acceleration is maintained from time t_2 to final time t_f [CSB+17].

2.2.3 Leading vehicles in target lane

To model the critical zone around the leading vehicle, let us assume that the leading vehicle is travelling with a constant velocity. It comes to an immediate stop at some time t, as shown in Fig. 2.4. This will give least time to react to the ego vehicle. A critical time $T_{critical}$, the latest time before which the ego vehicle has to initiate an emergency action, i.e. either brake or steer to avoid the collision, is calculated to represent the critical zone of the leading vehicle. For ego vehicle travelling at a given velocity v_{ego}, the braking distance S_{ego} is calculated from the assumed acceleration profile in Fig. 2.2. The critical time gap T_{brake} to be maintained by the ego vehicle to avoid collision with the leading vehicle by initiating braking is then calculated as

$$T_{brake} = \frac{S_{ego} + S_{min}}{v_{ego}} \tag{2.1}$$

$$T_{critical} = min(T_{brake}, T_{steer}) \tag{2.2}$$

$$S_{critical} = T_{critical} * v_{ego} \tag{2.3}$$

where S_{min} is the minimum longitudinal safety gap between the two vehicles after stopping [CSB+17].

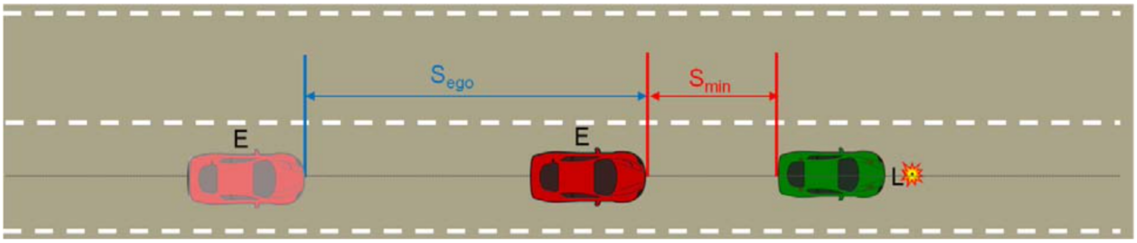


Fig. 2.4 Critical zone calculation for the leading vehicle when the ego vehicle brakes to avoid a collision. The leading vehicles crashes to a stop and the ego vehicle brakes to avoid collision, travelling a distance of S_{ego} before stopping [CSB+17].

Chapter 3

Compositional Verification of multi-lane traffic maneuver

3.1 Modelling the dynamic behavior of a car

To describe the dynamic behavior of the lane change control component during lane change, the following interactions are introduced:

- $c(m)$: Introduce a claim for lane m.
- $wd - c(m)$: Withdraw the claim for lane m.
- $r(m)$: Change a claim for lane m into a reservation for lane m.
- $wd - r(m)$: Withdraw the reservation for lane m.

Let the notation A.qRC means that car A is in control state q, it has reserved the lanes in the set R, and it claims the lanes in the set C. Fig. 3.1 shows the two possible behavior of a car A, which wants to change lane from current lane n to adjacent lane $n + 1$ or $n - 1$ represented as m. In **(a)**, car A directly changes lane from n to m, which is a dangerous idea, since if some other vehicle shows the same behavior and switches to lane m at the same time, then there is a chance of collision. In **(b)**, the vehicle first claims the target lane, if no other vehicle is claiming the target lane then it is safe to change the lane. Else wait for some arbitrary amount of time and then try again[BHLO17].

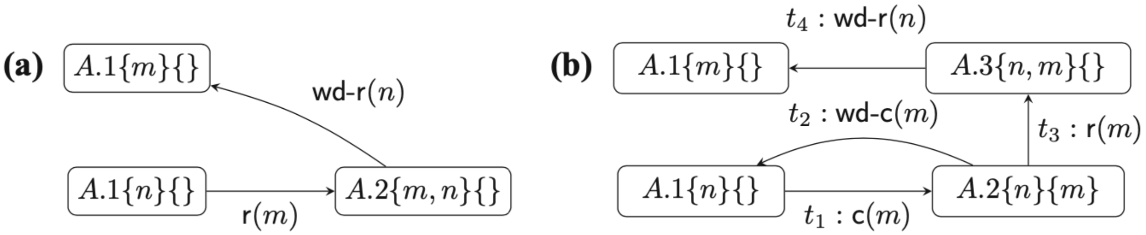


Fig. 3.1 Behavior of a car A changing from its current lane n to a neighboring target lane m is either $n-1$ or $n+1$: **a** simple algorithm, **b** protocol with a claim transition $c(m)$ as in [BHLO17].

3.2 Verification of multi-lane traffic maneuvers

For the velocity controller VC we consider:

VC.A1 : During a continuous flow, the front distance fd does not decrease faster than the current velocity cv (i.e. there is no car in front that goes in the opposite direction): fd' > -cv, where fd' denotes the timewise derivation of fd.

VC.A2 : After a discrete change of the front distance fd (which may be caused by an interaction $r(m)$ or $wd - r(n)$ of the lane controller), the front distance is still larger than the safety distance: $sd < fd$.

VC.G1 : During a continuous flow, the safety distance sd remains smaller than the front distance fd : $sd < fd$. This guarantee ensures, if $VC.A2$ is also satisfied, then the condition $sd < fd$ is always true.

For the steering controller SC we consider:

SC.A1 : When an input $r(m)$ is received, | m − current lane | = 1 holds.

SC.G1 : After the input of an $r(m)$ interaction, the current left-to-right position will become centered around the middle of lane m, and then an output wd-r(n) will be sent (where n is the previous current lane)[BHLO17].

Chapter 4

KeYmaera X : a tool for the verification of hybrid systems

KeYmaera X is an automatic and interactive formal verification tool for hybrid systems. Hybrid systems are mathematical models for dynamical systems, they involve continuous as well as discrete dynamics of a system. Hybrid systems can model highly complex Cyber-Physical Systems like autopilot avionic systems, autonomous drones, pacemakers, autonomous cars, etc. Presently, KeYmaera X is the premier theorem prover for hybrid systems which implements differential dynamic logic (dL), a specification and verification logic for hybrid systems. KeYmaera X has several techniques that it uses to verify properties and its proofs are rigorous and complete, they don't leave out any exceptions or corner cases. Although KeYmaera X is itself a powerful tool and many times guesses the invariant on its own but sometimes we need to supply an invariant to it, which is necessary to prove properties using induction.

4.1 Differential dynamic logic

Differential dynamic logic (dL) is a logic for specifying and verifying hybrid systems combining discrete and continuous dynamical systems. It is an extension of classical logic and can be defined by the following grammar:

$$P, Q ::= e \geq \tilde{e} \mid \neg P \mid P \wedge Q \mid P \vee Q \mid P \rightarrow Q \mid \forall_x P \mid \exists_x P \mid [\alpha]P \mid < \alpha > P \tag{4.1}$$

- P and Q are dL formulas
- e and ẽ are terms
- x is a variable
- *box* and *diamond* operator are the modal operators.

The functions of various operators been have described in Table 4.1.

dL	Operator	Meaning
$\theta_1 \sim \theta_2$	Comparison	True iff $\theta_1 \sim \theta_2$ with $\sim \in \{>, \geq, =, \neq, \leq, <\}$
$\neg\phi$	Negation/not	True if ϕ is false
$\phi \wedge \psi$	Conjunction/and	True if both ϕ and ψ are true
$\phi \vee \psi$	Disjunction/or	True if ϕ is true or if ψ is true
$\phi \rightarrow \psi$	Implication/implies	True if ϕ is false or ψ is true
$\phi \leftrightarrow \psi$	Bi-implication/equivalent	True if ϕ and ψ are both true or both false
$\forall_x \phi$	Universal quantifier	True if ϕ is true for all values of variable x in R
$\exists_x \phi$	Existential quantifier	True if ϕ is true for some values of variable x
$[\alpha]\phi$	$[\cdot]$ modality/box	True if ϕ is true after all runs of hybrid program α
$< \alpha > \phi$	$< \cdot >$ modality/diamond	True if ϕ is true after at least one run of hybrid program α

Table 4.1 Operators and (informal) meaning in differential dynamic logic (dL)

4.2 Hybrid programs

Hybrid programs are the programs that involve continuous evolution as well as conventional discrete programs, they describe what a dynamical system does over a period of time. Hybrid programs can be defined by following grammar:

$$\alpha, \beta ::= x := f(x) \mid ?Q \mid x' = f(x) \& Q \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^* \tag{4.2}$$

- α and β are the hybrid programs
- x is a variable
- x' represents time derivative of variable x
- $f(x)$ is a function of x that evolves with time
- Q is the evolution domain constraint, which is given by a First-Order Logic formula
- $x' = f(x)\&Q$ means that the differential equation $x' = f(x)$ can only evolve till it satisfies the evolution domain constraint Q

The functions of various operators have been described in Table. 4.2.

Statement	Effect
$\alpha;\beta$	Sequential composition where β starts after α finishes
$\alpha \cup \beta$	Nondeterministic choice, following either alternative α or β
$\alpha*$	Nondeterministic repetition, repeating α n times for any $n \in \mathrm{N}$
$x := \theta$	Discrete assignment of the value of term θ to variable x (jump)
$x := *$	Nondeterministic assignment of an arbitrary real number to x
$(x'_1 = \theta_1, ..., x'_n = \theta_n \& F$	Continuous evolution of x_i along the differential equation system $x'_i = \theta_i$ restricted to evolution domain F
$?F$	Test if formula F holds at current state, abort otherwise

Table 4.2 Statements of hybrid programs (F is a first-order formula, α, β are hybrid programs)

4.3 KeYmaera X

The input to KeYmaera X is a differential dynamic logic (dL) formula of the form :

$$init \rightarrow [\{ctrl; plant\}*](safe) \tag{4.3}$$

- *init* : It is the set of initial conditions for the system.

- $\{ctrl; plant\}$: It is a hybrid program where $ctrl$ represents the discrete dynamics and the $plant$ represents the continuous dynamics. ; is the sequential composition operator, which makes sure $plant$ runs after $ctrl$. $*$ is the non-deterministic repetition operator, it repeats the control loop, here $\{ctrl; plant\}$, any arbitrary number of times including zero.

- $safe$: It is the set of safety conditions that should be true at all times for the system to be safe, it is also called post-condition.

The safety of a car moving in a straight-line can be verified using following model:

$$init \rightarrow [\{ctrl; plant\}*](safe) \tag{4.4}$$

$$init \equiv v \geq 0 \wedge A > 0 \wedge B > 0 \tag{4.5}$$

$$ctrl \equiv a := A \cup a := 0 \cup a := -B \tag{4.6}$$

$$plant \equiv p' = v, v' = a \& v \geq 0 \tag{4.7}$$

$$safe \equiv v \geq 0 \tag{4.8}$$

Here, as indicated by $safe$ the system will be in a safe state if the velocity v of the car never gets negative, which means the car is either at a halt or moving in the forward direction. The initial conditions clarify that initial velocity is non-negative and the forward acceleration A and the braking acceleration B are positive real numbers.

First, the $ctrl$ sub-program sets the current acceleration a to A or 0 or $-B$. Then the $plant$ sub-program evolves the system for some arbitrary period of time. Because of the * operator, the hybrid program $\{ctrl; plant\}$ repeats for zero or more times non-deterministically. After any arbitrary number of repetitions of the hybrid program $\{ctrl; plant\}$, including zero times, the system should be in a safe state i.e. must satisfy $safe$ conditions.

Chapter 5

Verification of lane-change maneuvers using KeYmaera X

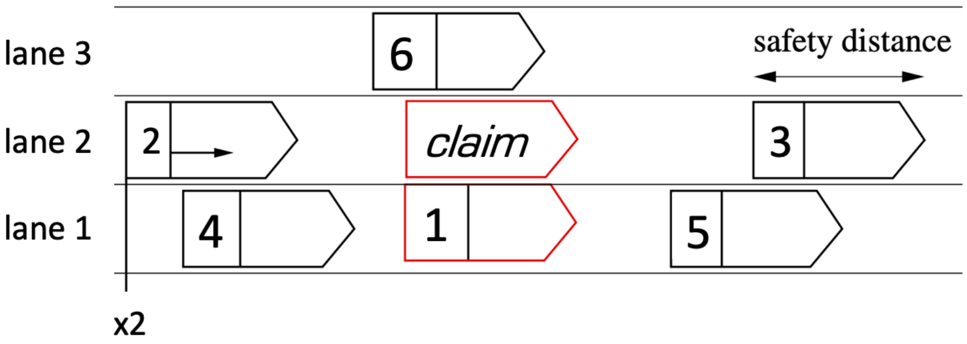
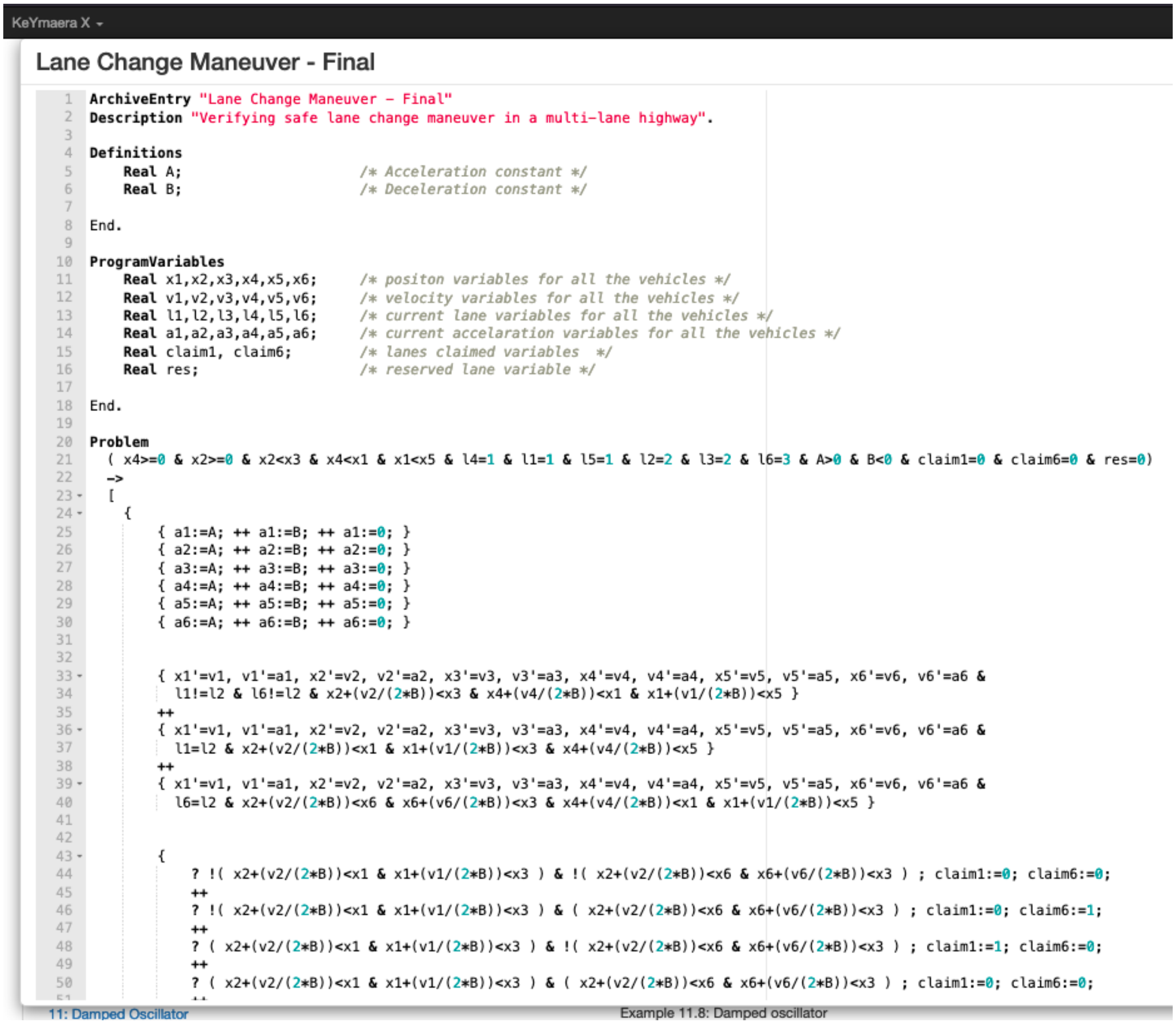


Fig. 5.1 A multi-lane highway with 1 as our ego vehicle as six cars

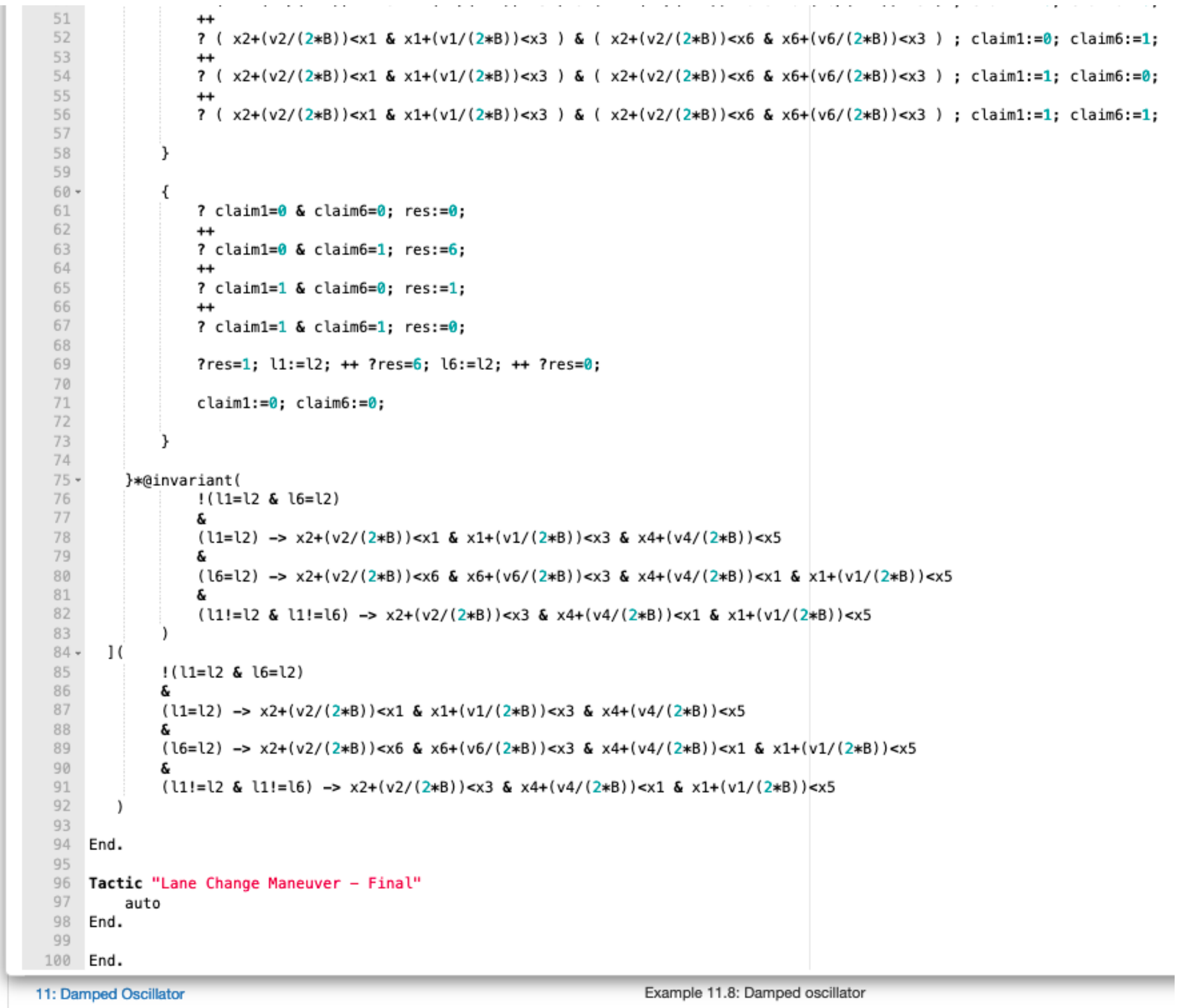
Consider a multi-lane highway scenario in which the vehicles are moving as shown in Fig. 5.1. Let us denote vehicle numbered i with v_i. We can observe that there are three lanes numbered 1, 2, and 3. In lane 1 there are three vehicle v_4, v_1 and v_5. In lane 2 there are two vehicles v_2 and v_3. In lane 3 there is only one vehicle v_6. Each vehicle has an envelope around it which is often referred to as *safety envelope* or *critical zone* of a vehicle, it is that area for a vehicle in which no vehicle should ever enter to guarantee safety. Let the position of a vehicle v_i be given by the variable x_i and the extreme points of the *safety envelope* of a vehicle are given by x_i and $x_i{+}sd$, where sd is current *safety distance* which is defined as the minimum distance a vehicle will travel if it applies brakes as hard as possible i.e. with the maximum deceleration possible. If the maximum possible deceleration of a vehicle is given by B and the current velocity of the vehicle is v, then using Newton's third equation of motion, the $safety\ distance$ can be defined as $sd = v^2/2B$.

We verified the safety of vehicles moving and changing lanes on a multi-lane highway using the $claim - reserve$ paradigm. According to which if a vehicle, let say v_1 is interested in the lane change, then it first claims the lane by sending a lane claim message to all the vehicles in its $view$. If v_1 doesn't receive a message from the vehicles in its $view$ that claim the same lane. Then, v_1 reserves the claimed lane and completes the lane-change maneuver. Else if v_1 as well as v_6 claim the same lane, then both v_1 and v_6 will receive each other's claim too, hence both will withdraw their claim and try to claim after some random amount of time, if possible. Following are the snippets of the code that verify the safety of the model described above.



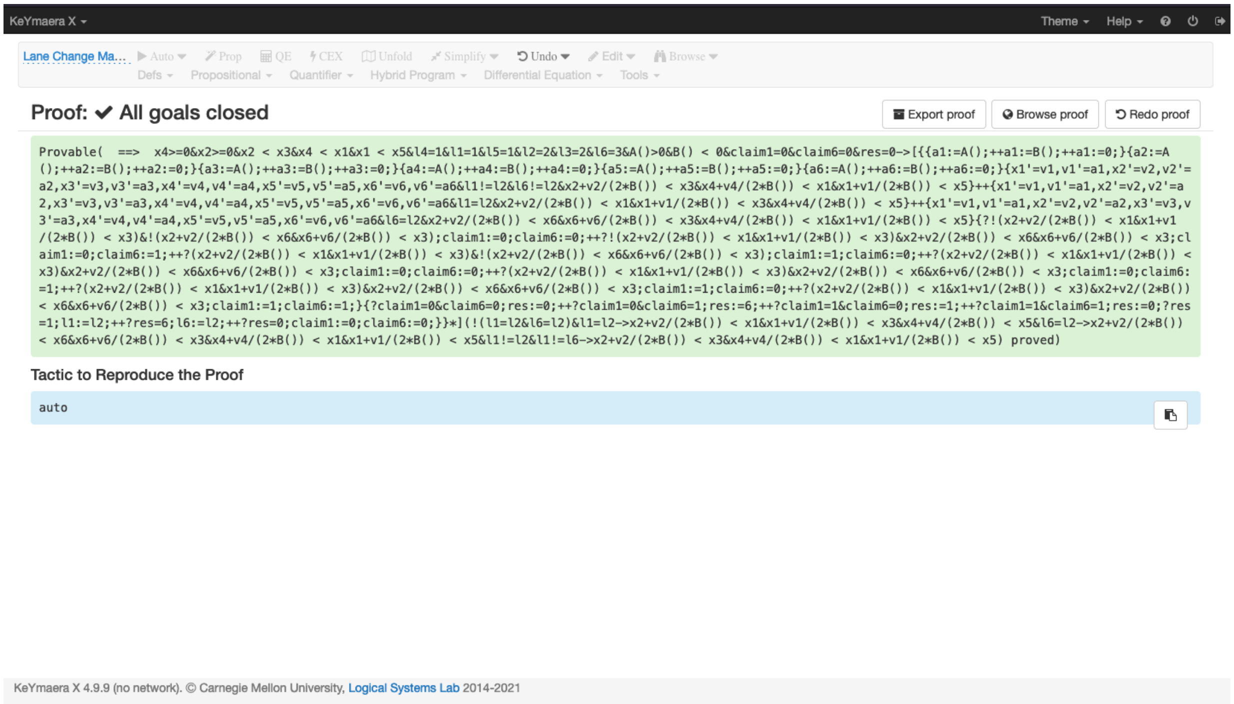
```
ArchiveEntry "Lane Change Maneuver - Final"
Description "Verifying safe lane change maneuver in a multi-lane highway".

Definitions
    Real A;                     /* Acceleration constant */
    Real B;                     /* Deceleration constant */

End.

ProgramVariables
    Real x1,x2,x3,x4,x5,x6;     /* positon variables for all the vehicles */
    Real v1,v2,v3,v4,v5,v6;     /* velocity variables for all the vehicles */
    Real l1,l2,l3,l4,l5,l6;     /* current lane variables for all the vehicles */
    Real a1,a2,a3,a4,a5,a6;     /* current accelaration variables for all the vehicles */
    Real claim1, claim6;        /* lanes claimed variables  */
    Real res;                   /* reserved lane variable */

End.

Problem
  ( x4>=0 & x2>=0 & x2<x3 & x4<x1 & x1<x5 & l4=1 & l1=1 & l5=1 & l2=2 & l3=2 & l6=3 & A>0 & B<0 & claim1=0 & claim6=0 & res=0)
  ->
  [
    {
        { a1:=A; ++ a1:=B; ++ a1:=0; }
        { a2:=A; ++ a2:=B; ++ a2:=0; }
        { a3:=A; ++ a3:=B; ++ a3:=0; }
        { a4:=A; ++ a4:=B; ++ a4:=0; }
        { a5:=A; ++ a5:=B; ++ a5:=0; }
        { a6:=A; ++ a6:=B; ++ a6:=0; }


        { x1'=v1, v1'=a1, x2'=v2, v2'=a2, x3'=v3, v3'=a3, x4'=v4, v4'=a4, x5'=v5, v5'=a5, x6'=v6, v6'=a6 &
          l1!=l2 & l6!=l2 & x2+(v2/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5 }
        ++
        { x1'=v1, v1'=a1, x2'=v2, v2'=a2, x3'=v3, v3'=a3, x4'=v4, v4'=a4, x5'=v5, v5'=a5, x6'=v6, v6'=a6 &
          l1=l2 & x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 & x4+(v4/(2*B))<x5 }
        ++
        { x1'=v1, v1'=a1, x2'=v2, v2'=a2, x3'=v3, v3'=a3, x4'=v4, v4'=a4, x5'=v5, v5'=a5, x6'=v6, v6'=a6 &
          l6=l2 & x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5 }


        {
            ? !( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & !( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=0; claim6:=0;
            ++
            ? !( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & ( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=0; claim6:=1;
            ++
            ? ( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & !( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=1; claim6:=0;
            ++
            ? ( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & ( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=0; claim6:=0;
```

Fig. 5.2 Snippet 1 - lane change maneuver in multi-lane highway modeled in KeYmaera X



```
            ++
            ? ( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & ( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=0; claim6:=1;
            ++
            ? ( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & ( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=1; claim6:=0;
            ++
            ? ( x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 ) & ( x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 ) ; claim1:=1; claim6:=1;

        }

        {
            ? claim1=0 & claim6=0; res:=0;
            ++
            ? claim1=0 & claim6=1; res:=6;
            ++
            ? claim1=1 & claim6=0; res:=1;
            ++
            ? claim1=1 & claim6=1; res:=0;

            ?res=1; l1:=l2; ++ ?res=6; l6:=l2; ++ ?res=0;

            claim1:=0; claim6:=0;

        }

    }*@invariant(
            !(l1=l2 & l6=l2)
            &
            (l1=l2) -> x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 & x4+(v4/(2*B))<x5
            &
            (l6=l2) -> x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5
            &
            (l1!=l2 & l1!=l6) -> x2+(v2/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5
        )
  ](
        !(l1=l2 & l6=l2)
        &
        (l1=l2) -> x2+(v2/(2*B))<x1 & x1+(v1/(2*B))<x3 & x4+(v4/(2*B))<x5
        &
        (l6=l2) -> x2+(v2/(2*B))<x6 & x6+(v6/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5
        &
        (l1!=l2 & l1!=l6) -> x2+(v2/(2*B))<x3 & x4+(v4/(2*B))<x1 & x1+(v1/(2*B))<x5
   )

End.

Tactic "Lane Change Maneuver - Final"
    auto
End.

End.
```

Fig. 5.3 Snippet 2 - lane change maneuver in multi-lane highway modeled in KeYmaera X



Proof: ✓ All goals closed

```
Provable(  ==>  x4>=0&x2>=0&x2 < x3&x4 < x1&x1 < x5&l4=1&l1=1&l5=1&l2=2&l3=2&l6=3&A()>0&B() < 0&claim1=0&claim6=0&res=0->[{{a1:=A();++a1:=B();++a1:=0;}{a2:=A
();++a2:=B();++a2:=0;}{a3:=A();++a3:=B();++a3:=0;}{a4:=A();++a4:=B();++a4:=0;}{a5:=A();++a5:=B();++a5:=0;}{a6:=A();++a6:=B();++a6:=0;}{x1'=v1,v1'=a1,x2'=v2,v2'=
a2,x3'=v3,v3'=a3,x4'=v4,v4'=a4,x5'=v5,v5'=a5,x6'=v6,v6'=a6&l1!=l2&l6!=l2&x2+v2/(2*B()) < x3&x4+v4/(2*B()) < x1&x1+v1/(2*B()) < x5}++{x1'=v1,v1'=a1,x2'=v2,v2'=a
2,x3'=v3,v3'=a3,x4'=v4,v4'=a4,x5'=v5,v5'=a5,x6'=v6,v6'=a6&l1=l2&x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3&x4+v4/(2*B()) < x5}++{x1'=v1,v1'=a1,x2'=v2,v2'=a2,x3'=v3,v
3'=a3,x4'=v4,v4'=a4,x5'=v5,v5'=a5,x6'=v6,v6'=a6&l6=l2&x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3&x4+v4/(2*B()) < x1&x1+v1/(2*B()) < x5}{?!(x2+v2/(2*B()) < x1&x1+v1
/(2*B()) < x3)&!(x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3);claim1:=0;claim6:=0;++?!(x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3)&x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3;cl
aim1:=0;claim6:=1;++?(x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3)&!(x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3);claim1:=1;claim6:=0;++?(x2+v2/(2*B()) < x1&x1+v1/(2*B()) <
x3)&x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3;claim1:=0;claim6:=0;++?(x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3)&x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3;claim1:=0;claim6:
=1;++?(x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3)&x2+v2/(2*B()) < x6&x6+v6/(2*B()) < x3;claim1:=1;claim6:=0;++?(x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3)&x2+v2/(2*B())
< x6&x6+v6/(2*B()) < x3;claim1:=1;claim6:=1;}{?claim1=0&claim6=0;res:=0;++?claim1=0&claim6=1;res:=6;++?claim1=1&claim6=0;res:=1;++?claim1=1&claim6=1;res:=0;?res
=1;l1:=l2;++?res=6;l6:=l2;++?res=0;claim1:=0;claim6:=0;}}*](!(l1=l2&l6=l2)&l1=l2->x2+v2/(2*B()) < x1&x1+v1/(2*B()) < x3&x4+v4/(2*B()) < x5&l6=l2->x2+v2/(2*B())
< x6&x6+v6/(2*B()) < x3&x4+v4/(2*B()) < x1&x1+v1/(2*B()) < x5&l1!=l2&l1!=l6->x2+v2/(2*B()) < x3&x4+v4/(2*B()) < x1&x1+v1/(2*B()) < x5) proved)
```

Tactic to Reproduce the Proof

```
auto
```

KeYmaera X 4.9.9 (no network). © Carnegie Mellon University, Logical Systems Lab 2014-2021

Fig. 5.4 Lane change maneuver in multi-lane highway model verified in KeYmaera X

Chapter 6

Conclusion

Ensuring the safety of autonomous vehicles on highways is challenging and formally verifying their safety is challenging too. If it's a single lane highway, the safety conditions include that any vehicle doesn't enter the *$safety$ envelope* also called *critical zone* of any other vehicle, which is generally possible by at most two vehicles, the one which is moving just ahead of ego vehicle and the one which is moving just behind the ego vehicle. It becomes more challenging when we have to ensure the safety of the vehicles in a multi-lane highway scenario, where the vehicle may have to make lane-change maneuvers.

In our work, the safety of lane-change maneuvers in dense traffic on a multi-lane highway was verified. We considered a multi-lane highway with three lanes and six vehicles, including the ego vehicle. Since the safety of a vehicle depends on its local environment hence, the vehicles that are out of ego vehicles $view$ can be neglected. $claim - reserve$ paradigm was used to ensure safety before, along, and after the lane-change maneuver. If two vehicles happen to find an opportunity to change lanes and reach the same target lane simultaneously, then the claim of both of these vehicles was nullified, and they may try claiming the target lane again after some time.

For the verification, we used KeYmaera X which is an automatic and interactive formal verification tool for hybrid systems, that implement differential dynamic logic (dL) and hybrid programs. It proves the input by considering the set of initial states to be true, and then inductively reaching the final states with all the states, including each state between the set of initial states and the set of final states, holding the post-condition as true. KeYmaera X provides us with intuitive techniques to verify models, all of its proofs are thorough and thorough, with no exceptions or corner cases. It is an interactive and effective tool for verifying the safety of Cyber-Physical Systems. The first stage in designing a Cyber-Physical system includes modeling it and then verifying it. Many other tools are unable to represent hybrid systems as precisely as KeYmaera X. Verified models can be used in conjunction with more efficient control algorithms to ensure the safety of a system. It does, however, have certain shortcomings. It is difficult to create a model in KeYmaera X, and it is even more difficult to verify its safety. The user must consider a mathematical model that will be used to prove models. Modeling complex systems is substantially more challenging, requiring a great deal of manual inventiveness. Furthermore, a mathematical model, modeled formally rarely can adequately explain all of a real-life system's probable actions.

We were able to verify lane-change maneuvers on multi-lane highways considering that a vehicle can make a discrete jump from the host lane to the target lane if the vehicle is willing to change lane and it is possible to do so without violating any safety conditions. The future work could seek the dynamics that are involved in a lane-change maneuver i.e. the trajectory planning for a lane change maneuver that guarantees safety of all the vehicles. Moreover, how the ego vehicle should behave if the vehicle which is moving just in front of claimed space in the target lane, decelerates or if the vehicle moving just behind the claimed space accelerates, just after the lane-change maneuver has been initiated.

References

[BHLO17] G.V. Bochmann, M. Hilscher, S. Linker, and E.R. Olderog. Synthesizing and verifying controllers for multi-lane traffic maneuvers. *Form Asp Comp 29*, page 583–600, 2017.

[CSB+17] Rajashekar Chandru, Yuvaraj Selvaraj, Mattias Brännström, Roozbeh Kianfar, and Nikolce Murgovski. Safe autonomous lane changes in dense traffic. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2017.

[FH12] Gregory M. Fitch and Jonathan M. Hankey. Investigating improper lane changes: Driver performance contributing to lane change near-crashes. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):2231–2235, 2012.

[MS15] Nikolce Murgovski and Jonas Sjöberg. Predictive cruise control with autonomous overtaking. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 644–649, 2015.

[NBCF15] Julia Nilsson, Mattias Brännström, Erik Coelingh, and Jonas Fredriksson. Longitudinal and lateral control for automated lane change maneuvers. In *2015 American Control Conference (ACC)*, pages 1399–1404, 2015.

[QML+16] Jan-David Quesel, Stefan Mitsch, Sarah Loos, Nikos Aréchiga, and André Platzer. How to model and prove hybrid systems with keymaera: a tutorial on safety. *International Journal on Software Tools for Technology Transfer*, 18:67–91, 02 2016.