Formal Verification of V2I aided Autonomous Driving: A Hybrid Systems Approach

Term Paper for 15624 course project

Ishan Pardesi
Dhruv Mahajan

1 Abstract

With the rapidly growing interest in autonomous car technology, it is becoming increasingly important to come up with techniques to ensure safety of autonomous cars. Another emerging trend in the automotive space is the connected car paradigm. It has been known that embedding intelligence into the road infrastructure can help boost safety of autonomous cars significantly. Here we present a model of a V2I aided autonomous driving system and go on to formally verify its safety using Differential Dynamic Logic.

2 Introduction

The world is rapidly moving towards complete autonomy in mobility. Driverless cars have already made it to the roads and companies have started commercial operations. We are not very far from the time when autonomous cars will become all pervasive[2]. As we see this shift towards autonomous driving, safety and reliability of automobiles is more important than ever before. It is only by formally verifying the algorithms used by autonomous cars can it be ensured that there will be a reduction in the number of accidents on the roads, the most potent promise of autonomy. Another notable, but generally overlooked aspect of driverless cars in the ability of smart infrastructure to ensure safety. A highway equipped with appropriate sensors and controllers can go a long way in helping automobiles make real time control decisions to ensure safety and efficiency.

In this paper, the problem of safety of autonomous cars has been addressed considering a smart highway system which communicates relevant information to all automobiles on the highway at discrete positions. A real world scenario is modelled where an autonomous car tries to safely and efficiently navigate its way across a smart highway using Differential Dynamic Logic for Hybrid Systems [7]. The objective here is to prove that the car does not experience a collision on the highway. The novelty of the paper is that it proposes a model to ensure safety of V2I aided autonomous driving and then formally verifies it using Differential Dynamic Logic.

3 Related Work

There is an extensive body of literature that describes various road maneuvers that can be performed by an autonomous vehicle. The problem of trajectory planning for an autonomous vehicle has also been discussed extensively by the research community ([4], [9], [3] and [5]). In most of the existing literature, the problem of autonomy is always approached from the perspective of the car. The car relies exclusively on its sensing and control capabilities to ensure safety and efficiency. However, the infrastructure can play a very powerful role in ensuring the safety and efficiency of the car. [10] proposes a model for intersection management using smart infrastructure. [1] looks at various scenarios where vehicle to infrastructure communication can assist autonomous driving. [6] discusses the idea of using the infrastructure to enforce a speed limit on a car and goes on to model and formally verify the same.

This paper proposes and formally verifies an intelligent highway system using differential dynamic logic [7]. It builds on the work done by [5] and [6]. The highway has intelligent nodes

deployed at various discrete locations on the road. These nodes communicate information about the road ahead to the car on the highway. This gives the car more time to make control decisions by augmenting the sensing capabilities of the car. The system is relevant to consider as it models multiple number of scenarios closely related to the real world on a highway where the environment is mostly adversarial and leverages the capability of the road infrastructure to ensure safety of the car.

This work contributes to the existing body of literature by successfully creating and verifying a model of V2I aided autonomous driving which is partly inspired by the work done by [6]. Though the paper proves safety of an autonomous car in a limited number of scenarios, the model can be extended as a general framework to prove safety of an autonomous car in the presence of smart road infrastructure.

4 Methodology

4.1 Differential Dynamic Logic

Differential Dynamic Logic has been used to model and verify the system[7]. It uses hybrid programs as a program notation to model hybrid systems (systems that are a combination of discrete and continuous dynamics). For understanding the model described in this paper, it is important to understand the program notation of hybrid programs.

Let α and β represent two hybrid programs defined as follows:

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

1. The non deterministic choice operator $\alpha \cup \beta$ follows that either α or β will be chosen randomly while executing a hybrid program (eg. Car can choose to accelerate or brake while on the highway).

2. The sequential composition $\alpha; \beta$ expresses that first α executes and then β executes.

3. α^* repeats α zero or more times (eg. The car will continue running on the highway till it reaches its destination).

4. Discrete assignment $x := e$ assigns a discrete value e to the variable x (eg. assign a random acceleration to the car). A variation of the discrete assignment is the random assignment $x := *$. This operator randomly assigns a value to variable x.

5. $x' = f(x)\&Q$ represents the evolution of the ODE $x' = f(x)$ within the domain Q.

6. $?Q$ is a test condition that only lets the formula proceed if Q is true.

Differential Dynamic Logic provides the box modality operator. The $d\mathcal{L}$ formula $[\alpha]P$ is true only when P is true for all runs of the hybrid program α. The system described in this paper uses this to model the safety condition: as long as the car is running on the highway, it will remain safe. Differential Dynamic Logic also comes with a verification technique that has been used to prove the correctness of the models.

4.2 Modelling the Environment

The environment is a straight two lane highway. The highway has intelligent nodes (sensing and control units) installed at every $\frac{D}{2}$ distance units. Each node has a range of 'D' units as described in the figure 1. Each node informs the car about the road conditions that lie within the range of the node. For this article, road conditions translate to three scenarios - presence of a stationary obstacle, presence of a moving obstacle, change in the number of lanes. Every time the car enters the range of a node, it is given information about the road conditions, 'D' units ahead. It is also assumed that once information is communicated to the car, there will not be any uncalled for changes in the environment. The car has full awareness of what is happening on the road for the next 'D' units.

4.3 Car Controls

The car can accelerate, brake or change lanes. Every time it crosses a node on the highway, it gets real time information from the node about the obstacles placed in the next D distance. The car is given sufficient amount of time to react to ensure that it does not have a collision.

4.4 The highway

The nodes on the highway are used to communicate information to the car. For the simplicity of the model, the environment is only allowed to change introduce static obstacles on the highway that will block only one lane at a time. New objects/changes on the road after always introduced after $\frac{D}{2}$ for the part of the road segment between 0 and D. The car gets information of the upcoming changes in the road conditions through the intelligent nodes installed at $\frac{D}{2}$ distance units. Every intelligent node covers a range of $\frac{D}{2}$ radius. To give an example, a node installed at $3D$ is able to give the information of region $3D$ to $3D+\frac{D}{2}$ to the car when it is at $2D+\frac{D}{2}$. Once the information is communicated to the car, the environment cannot be changed. It is important to note that every time the car reaches a mode, there is a shift in the origin. The car starts moving from $X_{angel} = 0$ and goes on till $X_{angel} = \frac{D}{2}$. As it gets into the range of the next node, the frame of reference is shifted by $\frac{D}{2}$.

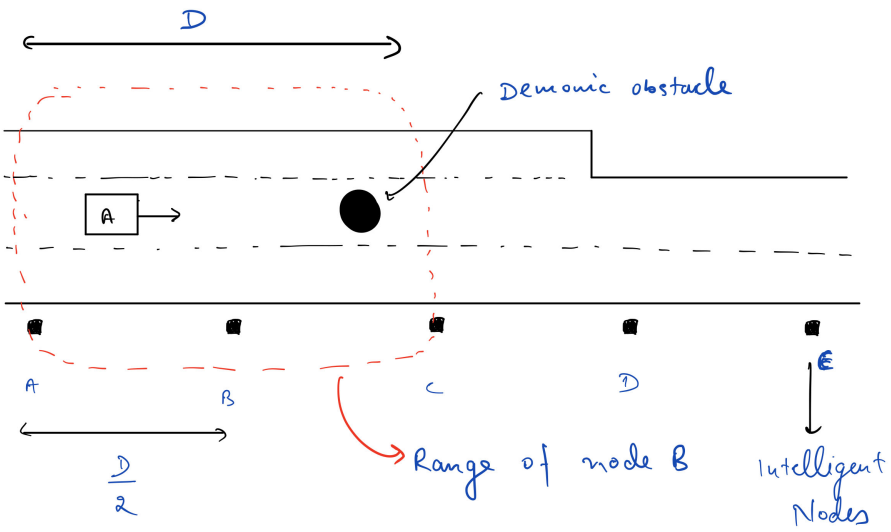

Figure 1: Intelligent Highway

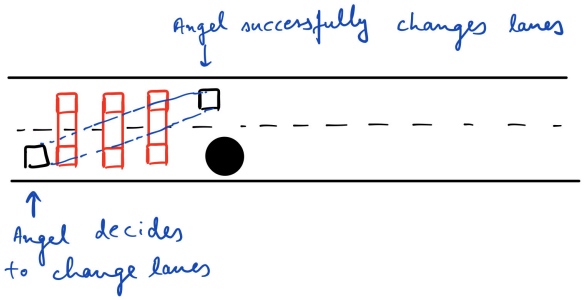

Figure 2: Lane Change maneuver

4.5 Lane Change Maneuver

This work builds on top of the work done by [5]. Though [5] considers infinite number of cars on the highway, this work only consider one car and one obstacle on the highway. It is assumed that the lane changing maneuver takes a fixed amount of time to complete irrespective of the velocity or acceleration of the car. For this time period, it is assumed that the car is present in both the lanes, the one that it is leaving and the one that it is entering.

5 Models Discussion

5.1 General Assumptions

1. The maximum acceleration and braking(deceleration) of the car is limited by its design. We denote it by $A(A > 0)$ and $-B(B > 0)$ respectively.

2. The maximum velocity of the car V_{angel} is limited by the speed limit of the highway $V_{SL}, (V_{angel} \leq V_{SL})$, which is assumed to be 70 m/s or 155 mph. since we know that cars don't go beyond 155 mph on a highway.

3. The inter node distance, $\frac{D}{2}$ is taken to be 150 meters, which according to the research team is a fairly well estimated real-world scenario value when considering factors such as: (1) the range, bandwidth and delay of the wireless communication protocol (enforcing an upper limit), (2) sensing capabilities of the autonomous car suggesting a lower limit as it is inappropriate to have nodes installed at small distances, and (3) the reaction time of the car including path planning, actuation and communication delays at the maximum velocity (of the highway).

4. The environment introduces new obstacle(s) after every $\frac{D}{2}$ distance.

5. The car cannot come to a stop on the highway lane and maintains a positive velocity at all times.

6. A lane changing maneuver takes a finite amount of time, $\delta > m * T, m \in \mathbf{N} \wedge m \neq 0$, where T is the controller cycle time.

7. The time period for the control cycle T is assumed to be equal to $\frac{1}{Vsl}$ to ensure that sampling happens once every meter, even when the car is moving at maximum velocity.

In the explanation for the model provided below, (X_{angel}, Y_{Angel}) represents the coordinates of the car. (X_{d0}, Y_{d0}) represents the obstacle situated between 0 and $\frac{D}{2}$. (X_{d1}, Y_{d1}) represents the obstacle between $\frac{D}{2}$ and D. (X_{dn}, Y_{dn}) represent the obstacle immediately ahead of the car.

5.2 Model 1

5.2.1 Objective

Initially a simple model is presented where the aim is to formally prove that the car can change lanes safely if it encounters an obstacle in front in the same lane. The car is moving with a fixed velocity and does not accelerate or brake. The car would engage into a lane change maneuver only if it senses that the remaining distance after time T from the obstacle is not adequate for a lane change maneuver. Thus, the car would change the lanes if and only if it faces a critical situation. There's only one static obstacle in this model.

5.2.2 Pre-Conditions

Model Preconditions - Model 1

1: (
2: $D = 300 \ \wedge \ V_{SL} = 70 \ \wedge \ \delta = 1 \wedge$
3: $X_{angel} = 0 \ \wedge \ Y_{angel} = 0 \ \wedge \ V_{angel} <= V_{SL} \ \wedge \ V_{angel} > 0 \wedge$
4: $X_{d0} >= \frac{D}{2} \ \wedge \ X_{d0} < D \ \wedge \ Y_{d0} = 0 \ \wedge \ Y_{new} = 0 \wedge$
5: $V_{SL} \, . \, \delta <= \frac{D}{2} \wedge$
6: $V_{angel} \, . \, \delta < (X_{d0} - X_{angel}) \wedge$
7: $T > 0 \wedge \ t_1 <= \delta \ \wedge \ lcFlag = 0$
8:)

The code snippet above shows the initial conditions of the car on the highway road. This includes the initial values of the different variables, values of the constants and the constraints on values of constants, if any.

Line number 2 assigns realistic values to parameters like D, V_{SL} and δ. $\frac{D}{2}$(inter-node distance) is chosen to be 150 meters, V_{SL} is taken to 70 meters per second (or 155 Mph) and δ (average time to change lanes) is assumed to be 1 second. Car's starting position is initialized in line number 3 to be at X=0 and lane 0. Car is also assigned a positive velocity but well within the maximum allowed velocity of V_{SL} imposed by the highway. The obstacle is placed randomly on the road somewhere between $X = \frac{D}{2}$ and X=D (in line number 4) so that the car has at least $\frac{D}{2}$ distance to react with respect to obstacle's position. This also aligns with the assumption of car having the information of the upcoming block of road of $\frac{D}{2}$ distance at least $\frac{D}{2}$ distance before that block arrives.

Line number 5 is more of a constraint of δ in form of a relation between D, V_{SL} and δ. It simply says that the distance taken by the car at max velocity to change the lanes will be less than or equal to the inter node distance (block size of the road). This condition is more useful in the later models where a fixed value of 1 second is not assigned to δ but is allowed to be within a range of values with minimum being 1 second. The need for the car to be safe before starting the journey in this model requires another constraint on the values of velocity and position of the car and the position of the obstacle. This is given in line number 6. Like previous condition, this condition is actually useful in the subsequent models where the obstacle is allowed to be placed in the first block of the road (same as the car).

The final line in the preconditions includes constraints on the values of controller's time period (T), variable to measure lane change time t_1 and the flag variable to track lane changing , i.e. lcFlag. lcFlag=0 indicates that initially lane change is not in progress.

5.2.3 Post Conditions

Model Post Conditions - Model 1

1: (
2: $(lcFlag = 0 \wedge (X_{d0} \neq X_{angel} \vee Y_{d0} \neq Y_{angel}) \ \bigvee \ (lcFlag = 1 \wedge X_{d0} \neq X_{angel})$
3:)

In this dL model, the post condition is equivalent to the Safety Condition of the car. At any time after the run of the hybrid program, the car must not have hit the obstacle. In terms of formal logic this can be written as the coordinates of the angel car are not equal to the coordinates of the obstacle. If any of the coordinates (x or y) of the car are not equal to the coordinates of the obstacle, then the car is safe.

5.2.4 Controller Design

Model Controller Design - Model 1

1: $\{?(lcFlag = 0);$
2: $\{$
3: $?((Y_{angel} = Y_{d0}) \wedge X_{angel} < X_{d0} \wedge (V_{angel} * \delta + V_{angel} * T >= X_{d0} - X_{angel}));$
4: $\{$
5: $lcFlag := 1;\ t_1 := 0;$
6: $\{(Y_{d0} = 1); Y_{new} := 0;$
7: $\cup$
8: $?(Y_{d0} = 0); Y_{new} := 1;\}$
9: $\}$
10: $\cup$
11: $?\neg(Y_{angel} = Y_{d0} \wedge X_{angel} < X_{d0} \wedge (V_{angel} * \delta + V_{angel} * T >= X_{d0} - X_{angel}));$
12: $/ * else, donothing * /$
13: $\}$
14: $\cup$
15: $?(lcFlag \neq 0);$
16: $\}$
17: $t := 0;$

The code snippet above is the controller for model 1 which essentially checks for the critical condition of distance left from the obstacle when it is in the same lane as obstacle. At the end, it initializes the time variable t to 0 to track the evolution of ODEs in dynamics of the model. In line 1-8, the controllers checks for the critical distance condition in line 3 only if it is currently not changing lanes, and is in the same lane as obstacle and has not crossed the obstacle. Line 5 sets the lane change flag to 1 and resets the t_1 variable. Lines 6-8 assign a value to Y_new (the new lane variable) on the basis of lane of obstacle.

In line 15, it checks if the lane change flag is not set to 0, i.e. if it is 1, then it does nothing and moves on to continue the lane changing maneuver.

5.2.5 Invariant

Model Invariant - Model 1

1: $^*@invariant($
2: $(\ (\ lcFlag = 0 \wedge (X_{d0} \neq X_{angel} \vee Y_{d0} \neq Y_{angel})\ ) \vee (lcFlag = 1 \wedge X_{d0} \neq X_{angel})\ )$
3: $\bigwedge$
4: $((lcFlag = 1 \wedge X_{angel} < X_{d0}) \rightarrow (V_{angel} * (\delta - t_1) < X_{d0} - X_{angel}) \wedge Y_{new} = 1 - Y_{d0})$
5: $\bigwedge$
6: $((lcFlag = 0 \wedge X_{angel} < X_{d0} \wedge Y_{angel} = Y_{d0}) \rightarrow (V_{angel} * \delta < X_{d0} - X_{angel}))$
7: $)$

The invariant is a first order logical formula which always holds (or is true) for all the runs of hybrid system. The invariant here is complex which takes into account various conditions which shall be true during the various actions undertaken by the controller.

The sub formula in line number 2 in the above code snippet is essentially the post condition as it is always true (car has not hit the obstacle) if the car is still running safely. This is in conjunction with the sub formula in line number 4 where we have an implication stating that if lane change is in progress and car is behind the obstacle, then the new lane must be lane not occupied by the obstacle, and the distance left to complete the lane change maneuver is less than than the instantaneous distance between car and the obstacle. This condition needs to be true if the car wants to safely change the lanes.

The sub formula in final line is also in conjunction with the above two sub formulas and is an implication where if the lane change is not in progress and the car happens to be behind obstacle and in the same lane as obstacle, then there must be enough distance gap between the car and obstacle to complete a lane change maneuver.

5.2.6 Differential Dynamics

Model Differential Dynamics - Model 1

1: {
2: ?($lcFlag = 1$);
3: $\{X'_{angel} = V_{angel}, t' = 1, t'_1 = 1 \wedge V_{angel} \geq 0 \wedge t \leq T \wedge t_1 <= \delta\}$
4: $if(t_1 = \delta)\{lcFlag := 0; Y_{angel} := Y_{new};\}$
5: $\bigcup$
6: ?($lcFlag = 0$);
7: $\{X'_{angel} = V_{angel}, t' = 1 \wedge V_{angel} \geq 0 \wedge t \leq T\}$
8: }

The above code snippet is the part of the model where the dynamics or motion of the car is described. The lane changing maneuver has been considered a special case where the car is occupying both the lanes simultaneously, and its progress is monitored by measuring the time of the maneuver and comparing it to δ. Two dynamics equations are introduced here, one for lane changing scenario and another for the default scenario with no lane changing. The dynamics in line 6-7 describe the motion during default mode where the rate of change of position is equal to velocity of the car, and time variable t increases linearly with time. This ODE evolves (or executes) for anytime greater than equal to 0 time till any of the other two condition of $t \leq T$ or $V_{angel} \geq 0$ stops holding true. Similarly for the other dynamics equation for lane changing mode in lines 2-4 describes the similar motion of the car but also monitors if the lane changing is complete via domain constraint condition $t_1 \leq \delta$. In addition, if the lane change is complete, then the lane change flag is reset and the car's lane variable Y_{angel} is assigned the value of new lane.

5.3 Model 2

5.3.1 Objective

This model builds upon the first model and adds the choice of changing lanes anytime during its motion if it is safe. The car doesn't have to wait for the critical moment to engage into a lane change maneuver but can do so anytime it feels like if it is safe.

5.3.2 Post Condition

The post conditions are the same in this model as we are only interested in proving the safety of the car at all times. If it doesn't collide with the obstacle at any time, it is safe.

5.3.3 Pre Conditions

Preconditions or assumptions in this model are mostly the same as the first model. However, two it has two changes. Firstly, the variable Y_{new} is removed as it was found to be redundant and the proof could be done without it. Secondly, the constant T (controller time cycle) is given a fixed value $T = \frac{1}{V_{SL}}$. This value of T makes sure that the controller is invoked to process the environment again within a metre of distance covered (at V_{SL}), allowing a finer control over the dynamics and control of the car.

5.3.4 Controller Design

Model Controller Design - Model 2

```
1: if ((t1 = δ) & (lcFlag = 1))
2:     lcFlag := 0; Y_angel := 1 − Y_angel;
3:
4:   if(lcFlag = 0)
5:     {
6:     if(X_angel > X_d0)
7:     {
8:       {lcFlag := 1; t_1 := 0;}
9:       ∪
10:      {lcFlag := 0;}
11:    }else{
12:      if(Y_angel = Yd0)
13:      {
14:        if(V_angel ∗ δ + V_angel ∗ T >= X_d0 − X_angel)
15:        {
16:        lcFlag := 1; t_1 := 0;
17:        }
18:        else{
19:          {lcFlag := 1; t_1 := 0;}
20:          ∪
21:          {lcFlag := 0;}
22:        }
23:      }else{
24:        ?(V_angel ∗ δ ∗ 2 + V_angel ∗ T < X_d0 − X_angel); lcFlag := 1; t_1 := 0;
25:        ∪
26:        lcFlag := 0;
27:      }
28:    }
29:  }
```

The controller for model 2 is different from model 1 int he sense that it shuffles the order of the control decision taken when $t_1 = \delta \& lcFLag = 1$, as seen in lines 1-2, and the way control decisions for lane changing takes place in lines 4-30. The control decisions like the previous model happen only when car is currently not doing lane change. In line 11-23, the control is for the case when the car is behind the obstacle and in line 6-10 if it has crossed the obstacle. In the latter case, it can choose to do anything arbitrarily.

Whereas in the former case, it needs to check whether the car is in the same lane as obstacle or not. If it is in the same lane, then it checks if it unsafe. If it is not unsafe, then it can either choose to change lanes or continue in the same lane (lines 18-21). If it is unsafe, then it must change lanes (lines 14-17).

Lines 24-27 describes the case when the car is not in the same lane and is behind the obstacle. Then, it can only change lanes if it has sufficient distance to change lanes twice, i.e. go to the same lane as obstacle and then change back. If it is safe, then it can either choose to change lanes or stay on the same lane. If it would be not safe to change lanes, then it must stay on the same lane.

5.3.5 Invariant

Model Invariant - Model 2

```
1: *@invariant(
2:   (lcFlag = 0 ∨ lcFlag = 1) ∧ (Y_angel = 0 ∨ Y_angel = 1) ∧ (Y_d0 = 0 ∨ Y_d0 = 1))
3:       ∧
4:   t_1 ≥ 0 ∧ t_1 ≤ δ
5:       ⋀
6:   (( lcFlag = 1 ∧ Y_angel = Y_d0) ) → (V_angel ∗ (δ − t_1) < X_d0 − X_angel) ∨ (X_angel > X_d0))
7:       ∧
8:   (lcFlag = 0 → (V_angel ∗ δ < X_d0 − X_angel ∧ Y_angel = Y_d0) ∨ (X_angel > X_d0) ∨ (Y_angel ≠ Y_d0)
9:       ∧
10:  ((lcFlag = 1 ∧ Y_angel ≠ Y_d0) → (V_angel ∗ (2 ∗ δ − t_1) < X_d0 − X_angel) ∨ (X_angel > X_d0))
11:                )
```

Progression to this model allowed changing the invariant to a stronger one which keeps a track of more constants as well. The code snippet above shows the new invariant, which takes into account of new property of changing lanes anytime as well. The lines 2-4 makes sure the bounds of time variable t_1, possible values of lcFlag, lanes of the car and obstacle remain preserved across loop iterations while doing the proof.

The remaining line 6-10 includes 3 implications which are responsible for preserving the different scenarios, i.e. changing lanes when the car has already crossed the obstacle, when it has not crossed the obstacle and is in the same lane (as in previous model) or when it is in the different lane as obstacle but has not crossed the obstacle.

Line 6 says if the car is changing lanes and is initially in the same lane as obstacle, then it should either be a safe distance so do a lane change later or should have already crossed the obstacle. Line 10 says if it is changing lanes and the initial lane is not the same as obstacle's lane, then it should have either crossed the obstacle or else it should have distance at least twice more than the single lane change distance so that it can come back into this safe lane by going into the unsafe lane first.

Line 8 says that if lane change is not in progress, then either it should have sufficient distance to change lane once if it is in the same lane and before the obstacle or it should have crossed the obstacle or not in the same lane at all.

5.3.6 Differential Dynamics

Model Differential Dynamics - Model 2

```
1:  {
2:    if(lcFlag = 1){
3:       {X'_angel = V_angel, t' = 1, t'_1 = 1 ∧ V_angel ≥ 0 ∧ t ≤ T ∧ t_1 <= δ}
4:    }
5:    else{
6:       {X'_angel = V_angel, t' = 1 ∧ V_angel ≥ 0 ∧ t ≤ T}
7:    }
8:  }
```

The dynamics of the model have changed slightly from the previous model. The nested if condition which reset the lane change flag and changed the lane of car has been moved to the top of the controller, thus simplifying the dynamics without changing the working of the overall model. The non deterministic choice has been replaced with deterministic if-else condition. The dynamics which describe the motion and monitoring of the lane change time is essentially unchanged.

5.4 Model 3

5.4.1 Objective

Model 3 introduces the notion of smart infrastructure and recurring obstacles. There is a new obstacle that is introduced every time the car crosses a node. In terms of control choices, the car

only has an option to change lanes when absolutely critical as was in model 1. This model builds on top of model 1 as it ensures that the obstacles occur repeatedly and the nodes continue to give the car information about the new obstacles.

5.4.2 Post Conditions

The post conditions are the same as before as the safety condition of the car remains unchanged. If the car does not collide then it is safe.

5.4.3 Pre Conditions

Model Preconditions - Model 3

$$
\begin{aligned}
&(\\
&\quad D = 300 \ \wedge \ V_{SL} = 70 \ \wedge \ \delta = 1 \ \wedge \\
&\quad X_{angel} = 0 \ \wedge \ Y_{angel} = 0 \ \wedge \ V_{angel} <= V_{SL} \ \wedge \ V_{angel} > 0 \ \wedge \\
&\quad X_{d0} >= 0 \ \wedge \ X_{d0} < \tfrac{D}{2} \ \wedge \ Y_{d0} = 0 \ \wedge \\
&\quad X_{d1} >= \tfrac{D}{2} \ \wedge \ X_{d0} < D \ \wedge \ Y_{d1} = 1 \ \wedge \\
&\quad X_{d1} - X_{d0} > V_{angel} * \delta \ \wedge \ X_{dn} = X_{d0} \ \wedge \ Y_{dn} = Y_{d0} \ \wedge \\
&\quad V_{SL} \,.\, \delta <= \tfrac{D}{2} \ \wedge \\
&\quad V_{angel} \,.\, \delta < (X_{d0} - X_{angel}) \ \wedge \\
&\quad T > 0 \wedge \ t_1 <= \delta \ \wedge \ lcFlag = 0 \\
&)
\end{aligned}
$$

The pre-conditions are mostly same as the ones described in model 1. However, in this model, there are two obstacles instead of one at any given time. One obstacle lies between 0 and $\frac{D}{2}$ and is denoted by $(Xd0, Yd0)$, while the other obstacle lies between $\frac{D}{2}$ and D and is denoted by $(Xd1, Yd1)$. Initially it is assumed that the two obstacles are $V_{angel} * \delta$ distance apart. The car is also $V_{angel} * \delta$ distance apart from the first obstacle to ensure that the car is safe initially. Y_{new} has been eliminated from this model. (Xdn, Ydn) have been introduced and they always point to the obstacle immediately in front of the car.

5.4.4 Controller Design

Model Controller Design - Model 3

$$
\begin{aligned}
&\text{if } ((t1 = \delta) \ \&\ (lcFlag = 1))\\
&\quad lcFlag := 0;\ Y_{angel} := 1 - Y_{angel};\\
&\\
&\text{if } (X_{angel} = \tfrac{D}{2})\\
&\quad \{\\
&\quad X_{angel} := 0;\ X_{d0} := X_{d1} - D/2;\ Y_{d0} := Y_{d1};\\
&\quad Y_{d1} := 0;\ ++\ Y_{d1} := 1;\\
&\quad X_{d1} := *;\ ?(X_{d1} >= D/2 \ \&\ X_{d1} < D \ \&\ X_{d1} - X_{d0} > (V_{angel} * delta));\\
&\quad X_{dn} := X_{d0};\ Y_{dn} := Y_{d0};\\
&\quad \}\\
&\\
&\text{if } (X_{angel} = X_{dn})\\
&\quad X_{dn} := X_{d1};\ Y_{dn} := Y_{d1};\\
&\\
&\text{if } (lcFlag = 0)\{\\
&\quad \text{if}((Y_{angel} = Y_{dn}) \ \&\ X_{angel} < X_{dn} \ \&\ (V_{angel} * delta + V_{angel} * T >= X_{dn} - X_{angel}))\\
&\qquad \{lcFlag := 1;\ t1 := 0;\}\\
&\}\\
&\\
&t := 0
\end{aligned}
$$

The controller for model 3 can be understood by dividing it into two separate parts: 1) The control decisions that the car takes. 2) The change in the environment that is happening as a result of the motion of the car.

Line 15 to 18 in the Controller Design code above represent the control choices that the car has. In this model, the car is only allowed to change its lane when absolutely critical to do so.

The rest of the controller primarily accommodates for the changes in the environment. Line 1 and 2 represent the scenario when the car has just finished a lane change maneuver. Line 4 to 10 look at the condition when the car crosses a node. As the car crosses a node, it receives new information about the environment. The frame of the reference is shifted here to ensure that new variables are not required to store the information just received, as well as to maintain simplicity. Every time the car reaches $\frac{D}{2}$, its origin is shifted by $\frac{D}{2}$ and the position of the car is made the 0 of the new reference frame. Besides that, the information regarding (X_{d0}, Y_{d0}) is discarded, (X_{d1}, Y_{d1}) is shifted into (X_{d0}, Y_{d0}) and new values are stored in (X_{d1}, Y_{d1}). The values of (X_{dn}, Y_{dn}) are also updated to point to the next obstacle. Line 12-13 represent the scenario when the X coordinate of the car becomes equal to the X coordinate of the obstacle. In this case, we just update (X_{dn}, Y_{dn}).

5.4.5 Invariant

Model Invariant - Model 3

1: $^*@invariant($
2: $($
3: $(lcFlag = 0 \lor lcFlag = 1) \land (Y_{angel} = 0 \lor Y_{angel} = 1) \land (Y_{d0} = 0 \lor Y_{d0} = 1) \land$
4: $(Y_{d1} = 0 \lor Y_{d1} = 1) \land (((Y_{dn} = Y_{d0}) \land (X_{dn} = X_{d0})) \lor ((Y_{dn} = Y_{d1}) \land (X_{dn} = X_{d1}))) \land$
5: $(X_{d0} >= 0 \land X_{d0} < D/2) \land (X_{d1} >= D/2 \land X_{d1} < D) \land (X_{d1} - X_{d0} > V_{angel} * \delta)$
6: $)$
7: $\bigwedge$
8: $($
9: $t_1 \geq 0 \land t_1 \leq \delta \land X_{angel} \leq \frac{D}{2} \land X_{angel} \geq 0 \land X_{angel} \leq X_{dn}$
10: $\land$
11: $((\, lcFlag = 0 \land Y_{angel} = Y_{dn}\,) \rightarrow (V_{angel} * \delta < X_{dn} - X_{angel}))$
12: $\land$
13: $((lcFlag = 1) \rightarrow (V_{angel} * (\delta - t_1) < X_{dn} - X_{angel}))$
14: $)$
15: $)$

The invariant for this model is a major improvement over the previous models. The invariant for this model has been separated into two parts. Line 2-6 represent the section of the invariant that does not contain any variable that would change due to the ODE. Thus, both the parts of the invariant can be separated at the outset. This simplifies the proof as is explained in the Appendix.

Part 1 of the invariant defines general model constraints. Line 9 also establishes the contraints in which X_{angel} should lie. Line 11 ensures that the car always maintains sufficient distance from an obstacle if it is in the same lane as the obstacle. Line 13 ensures that when changing lanes, the car is sufficiently behind the obstacle.

5.4.6 Differential Dynamics

Model Differential Dynamics - Model 3

1: $\{$
2: $if(lcFlag = 1)\{$
3: $\{X'_{angel} = V_{angel}, t' = 1, t'_1 = 1 \land V_{angel} \geq 0 \land t \leq T \land t_1 <= \delta \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
4: $\}$
5: $else\{$
6: $\{X'_{angel} = V_{angel}, t' = 1 \land V_{angel} \geq 0 \land t \leq T \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
7: $\}$
8: $\}$

The differential dynamics for Model 3 is mostly same as the differential dynamics for second model. The only difference here is the introduction of $X_{angel} <= \frac{D}{2} \wedge X_{angel} <= X_{dn}$ in the domain constraints to ensure that system breaks out of the ODE when the car reaches $\frac{D}{2}$ or reaches the position of the obstacle.

5.5 Model 4

5.5.1 Objective

This model further builds upon the second model and adds the capability of either accelerating or slowing down the car in addition to change lanes freely.

5.5.2 Pre-Conditions

Model Preconditions - Model 4

1: (
2: $B > 0 \wedge A > 0 \wedge A_{angel} > 0 \wedge$
3: $\frac{D}{2} > \frac{V_{SL}^2}{2.B} \wedge$
4: $\frac{V_{SL}^2}{2.B} \leq V_{SL} * \delta \wedge$
5: $\frac{A.T^2}{2} < \frac{D}{2} \wedge$
6: $(Y_{d0} = 0 \vee Y_{d0} = 1) \wedge (Y_{angel} = 0 \vee Y_{angel} = 1)$
7:)

As acceleration capability gets introduced, the model gets augmented with a couple of pre-conditions. These are mainly the constraints on A(max acceleration) and B(max deceleration). The code snippet above lists preconditions in addition to the ones considered in model 2. Line 3 effectively says that the worst case braking distance must be less than inter node distance $\frac{D}{2}$. Line 4 says that the braking distance should also be less than worst case lane change distance. This constraint makes it possible to say that braking is always a safe option if lane change is not feasible. Line 5 says that acceleration A must not be so high that the distance covered by the car in one controller cycle T, starting from rest is greater than inter node distance $\frac{D}{2}$.

In this model, the harsh restriction on $\delta = 1$ has been modified to $\delta \geq 1$ thereby providing more generality in the design. this model further allows the car and obstacle to be in any of the two lanes.

5.5.3 Post Condition

The post conditions or safety property in this model remains same as previous models, i.e. the car must not collide with the obstacle at any time despite any number of times the controller has been executed in whatever sequence.

5.5.4 Controller Design

Model Controller Design - Model 4

1:
2: $A_{angel} := *; ?(A_{angel} \geq 0 \wedge A_{angel} \leq A \wedge (V_{angel} + A_{angel} * T) \leq V_{SL} \wedge (V_{angel} + A_{angel} * T) > 0)$
3: $if(V_{angel} * T + \frac{A_{angel} * T^2}{2} + (V_{angel} + A_{angel} * T) * \delta \geq X_{d0} - X_{angel})$
4: {
5: {
6: $lcFlag := 1;$
7: $A_{angel} := *; ?(A_{angel} \geq -B \wedge A_{angel} <= 0 \wedge (V_{angel} + A_{angel} * T) > 0);$
8: $t_1 := 0;$
9: }
10: $\cup$
11: $A_{angel} := -B;$
12: }

The controller for this model builds upon the model of model 2, where it arbitrarily chooses a positive acceleration first and then checks for the safety condition violation. If the condition violates, then it chooses to either change lanes or remain in the same lane depending upon which lane it was in before. Further more, while changing lanes, the car is allowed to choose a negative acceleration so that it can slow down while changing lanes, but never speed up apart from the first controller cycle while changing lanes. The code in line 2 assigns a arbitrary positive value to car's acceleration within limits. The line 3 has the code where the condition is checked whether this acceleration is safe for time T such that car will be able to do a lane change if in the same lane. If yes, then acceleration is chosen again, positive or negative. Else, the car is given a chance to break as well.

5.5.5 Invariant

The invariant for this model is same as model 2 as it employs the same cases.

5.5.6 Differential Dynamics

Model Differential Dynamics - Model 4

1: {
2: $if(lcFlag = 1)\{$
3: $\{X'_{angel} = V_{angel}, V'_{angel} = A_{angel}, t' = 1, t'_1 = 1 \wedge V_{angel} \geq 0 \wedge t \leq T \wedge t_1 <= \delta\}$
4: }
5: $else\{$
6: $\{X'_{angel} = V_{angel}, V'_{angel} = A_{angel}, t' = 1 \wedge V_{angel} \geq 0 \wedge t \leq T\}$
7: }
8: }

The dynamics of this model change a bit as the acceleration gets introduced. The dynamics need to consider the rate of change of velocity now as well. This can be seen above in the code snippet in lines 3 and 6.

5.6 Model 5

5.6.1 Objective

This model further builds upon model 3 and adds the capability of either accelerating or slowing down the car in addition to the repeated introduction of obstacles on the road every $\frac{D}{2}$ distance. This model also allows to change lanes prematurely only if it is in the same lane and thus provides some independence to the car.

5.6.2 Pre-Conditions

As acceleration capability gets introduced, this model also requires the same additional pre-conditions as introduced in model 4. Thus, same preconditions are applicable here.

5.6.3 Post Condition

The post conditions or safety property in this model remains same as model 3, i.e. the car must not collide with the next obstacle at any time despite any number of times the controller has been executed in whatever sequence.

5.6.4 Controller Design

Model Controller Design - Model 5

1: if $(lcFlag = 0)\{$
2: $A_{angel} := *; ?((A_{angel} >= -B \land A_{angel} <= A) \bigwedge$
$(Y_{angel} = Y_{dn} \rightarrow (V_{angel} + A_{angel} * T) * \delta + V_{angel} * T + \frac{A_{angel} * T^2}{2} < X_{dn} - X_{angel}) \bigwedge$
$((V_{angel} + A_{angel} * T) <= Vsl \land (V_{angel} + A_{angel} * T) > 0)$
3: $\cup$
4: $if(Y_{angel} = Y_{dn})$
5: $\{lcFlag := 1;\ t_1 := 0; A_{angel} := 0\}$
6: $\}$

The controller of this model builds upon the controller of model 3, where the affect of acceleration is added while checking for the critical conditions in the controller. In addition, an arbitrary value of acceleration is also chosen between +A and -B. Line 2 lists the modified condition used in model 5 along with the arbitrarily chosen acceleration for the car. The third part of the condition makes sure that the chosen acceleration is not such that the car's velocity will be out of bounds (greater than V_{SL} or less than equal to 0). If the condition is true, then it may choose to do nothing, i.e. continue in the same lane or change lanes its in the same lane as obstacle. IF the condition fails, then it checks for the lane and changes the lane if in the same lane. Else, it was in a different lane and can choose to do nothing.

5.6.5 Invariant

The invariant for this model is same as model 3 as it employs the same cases.

5.6.6 Differential Dynamics

Model Differential Dynamics - Model 5

1: $\{$
2: $if(lcFlag = 1)\{$
3: $\{X'_{angel} = V_{angel}, V'_{angel} = A_{angel}, t' = 1, t'_1 = 1 \land V_{angel} \geq 0 \land t \leq T \land t_1 <= \delta \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
4: $\}$
5: $else\{$
6: $\{X'_{angel} = V_{angel}, V'_{angel} = A_{angel}, t' = 1 \land V_{angel} \geq 0 \land t \leq T \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
7: $\}$
8: $\}$

The dynamics of this model change a bit as the acceleration gets introduced. The dynamics need to consider the rate of change of velocity now as well. This can be seen above in the code snippet in lines 3 and 6.

5.7 Model 6

5.7.1 Objective

This model is an amalgamation of model 4 and 5, thus effectively encapsulating all the various properties and features introduced over the last 5 models. This model adds the freely lane changing capabilities to fifth model, thus freeing it from the constraint of only changing lanes in critical situations.

5.7.2 Pre-Conditions

The pre-conditions remains same as the model 5.

5.7.3 Post Condition

The post condition or safety property also remains the same as model 5. The car and the next obstacle must never collide.

5.7.4 Controller Design

The controller design for this design subtly augments the controller of model 5, thereby adding the functionality of freely changing lanes. It takes the cue from model 4 to implement this.

5.7.5 Invariant

Model Invariant - Model 6

1: $^*@invariant($
2: (
3: $(lcFlag = 0 \lor lcFlag = 1) \land (Y_{angel} = 0 \lor Y_{angel} = 1) \land (Y_{d0} = 0 \lor Y_{d0} = 1) \land$
4: $(Y_{d1} = 0 \lor Y_{d1} = 1) \land (((Y_{dn} = Y_{d0}) \land (X_{dn} = X_{d0})) \lor ((Y_{dn} = Y_{d1}) \land (X_{dn} = X_{d1}))) \land$
5: $(X_{d0} >= 0 \land X_{d0} < D/2) \land (X_{d1} >= D/2 \land X_{d1} < D) \land (X_{d1} - X_{d0} > V_{angel} * \delta)$
6:)
7: $\bigwedge$
8: (
9: $t_1 \geq 0 \land t_1 \leq \delta \land X_{angel} \leq \frac{D}{2} \land X_{angel} \geq 0 \land X_{angel} \leq X_{dn}$
10: $\land$
11: $((\, lcFlag = 0 \land Y_{angel} = Y_{dn}\,) \rightarrow (V_{angel} * \delta < X_{dn} - X_{angel}))$
12: $\land$
13: $((lcFlag = 1) \rightarrow (V_{angel} * (\delta - t_1) < X_{dn} - X_{angel}))$
14: $\land$
15: $((lcFlag = 1 \land Y_{angel} \neq Y_{dn}) \rightarrow (V_{angel} * (2 * \delta - t_1) < X_{dn} - X_{angel}) \lor (X_{angel} > X_{dn}))$
16:)
17:)

The invariant for this model is the most complex of all the models presented above. The invariant of model 5 is augmented by adding the implication for double lane changing from the invariant of model 4. This way the invariant keeps track of the conditions which must hold true across all the cases at all times.

5.7.6 Differential Dynamics

Model Differential Dynamics - Model 6

1: {
2: $if(lcFlag = 1)\{$
3: $\{X'_{angel} = V_{angel}, t' = 1, t'_1 = 1 \land V_{angel} \geq 0 \land t \leq T \land t_1 <= \delta \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
4: }
5: $else\{$
6: $\{X'_{angel} = V_{angel}, t' = 1 \land V_{angel} \geq 0 \land t \leq T \land X_{angel} <= \frac{D}{2} \land X_{angel} <= X_{dn}\}$
7: }
8: }

The dynamics of this model are the same as previous model. The dynamics need to consider the rate of change of velocity, which can be seen in the code snippet in lines 3 and 6.

6 Results and Discussion

Three out of six models described above have been successfully proved using keYmaera X. The proof of model 1 and model 2 dictate that a car moving with constant velocity on a two lane

highway can ensure safety as long as it initiates the lane change maneuver sufficiently in advance. The novelty of the research comes from the successful proof of model 3. This model looks at the scenario when there are recurring obstacles on the highway in either of the lanes and the car does not want to stop. As long as it is ensured that the obstacles are sufficiently spaced out to allow for a lane change maneuver after the car crosses an obstacle, it can be ensured that the car will remain safe. Another novelty that this model brings to the table is the road environment that it creates. It is assumed in the model that the information about the road ahead is only communicated to the car using the intelligent nodes located on the highway. Proving that the car remains safe inspite of only having information about the next 'D' distance through the nodes on the highway is an important step in the direction of formal verification of autonomous vehicles using V2I networks. It must be noted here that the significance of such a framework is not limited to verification of the safety of car maneuvers offline/in a simulation. Such a framework can be deployed both in a car and the road infrastructure to ensure safety of autonomous cars in real-time. To illustrate this point, it can be said that a smart sensing and control unit situated on the highway can enforce a car to opt out of a lane change maneuver in real time if it senses that the car does not have sufficient space to navigate the maneuver successfully. This can be of significant help in ensuring safety of both human driven and autonomous vehicles.

The other three models essentially give the car more freedom by allowing it to accelerate, decelerate or change lanes (at any time). Model 4 allows the car to change lanes freely along with acceleration and deceleration. Model 5 builds upon model 3 and adds acceleration capability, thus allowing the car to be guided by road infrastructure and accelerate or decelerate in parallel. Model 6 further combines model 4 & 5, by allowing the car to accelerate, slow down, and freely change lanes, even if it intends to go in to the lane of obstacle while being safe initially.

Model 4,5,6 could not be proved in time for this article. It must be acknowledged here that successfully proving a model using KeYmaeraX still takes significant amount of time. Model 2 and 3 took about 35-40 man hours (combined) to be proved successfully, even though several tricks as mentioned in the Appendix were repeatedly applied. Part of this can be attributed to the fact that every time an issue was encountered with the model, the proof had to be restarted from the beginning. Besides that, the large size of the proof also led to frequent crashes of the software. The software also demanded significant processing power and a general purpose laptop was not sufficient to run the software seamlessly and prove the model easily.

7 Conclusion

The model proposed in this paper can be used as a general framework for formal verification of V2I aided autonomous driving. Such a model does not exist in the literature according to the knowledge of the research team. Though the work conducted as part of this project just proves the safety of a car in the presence of static obstacles on a two lane highway, this framework can be extended to prove a number of other traffic scenarios as was initially proposed. For example, a similar model can be used to prove the safety of a car in the presence of moving obstacles and/or dynamically changes number of lanes on the highway. The existing model can also be proved from a hybrid games [8] perspective. Hybrid games gives the advantage of modeling the environment as an adversary(demon), an idea that is not very different from a real world scenario. The thought here is that the demon(environment) tries to perform whatever is in its capability to ensure that the angel car(autonomous car) eventually experiences a collision(becomes unsafe). However, the demon can only operate within certain constraints. By proving that angel wins the Hybrid game, we prove that the autonomous car is safe in a real world road environment.

8 Appendix

8.1 Proof Techniques

The models have been proved using two broad approaches.

1. The first approach is a more traditional approach wherein all branches are opened one by one and certain simplifications are done on certain branches. Since there are about 5-6 choice operators on average in every model after model 3, the number of branches can go to around 50 or more. Even though a number of tricks have been used to solve the branches early on,

it still consumes considerable amount of time (The tricks have been described below). All the models so far have been proved using this approach.

2. The second approach extensively leverages MR to simplify the proving process. Rather than opening the branches, we start removing one box modality at a time. To illustrate this, we consider an example where we have to prove $\gamma \vdash [\alpha][\beta]P$. Using MR, this formula is broken down into $\gamma \vdash [\alpha]Q$ and $Q \vdash [\beta]P$. It is ensured while modeling that the first modality is the outermost modality. This approach was used to prove model 5. Though the proof could not be completed due to shortage of time, a significant reduction in proving time was observed using this technique. The only challenge in this approach is to ensure that MR formula is chosen correctly.

8.2 Proof Tricks

A number of techniques were used to reduce the time taken to prove a formula. Following is a list of the techniques used:

1. The size of the proof gets drastically reduced by identifying branches that would be vacuously true. For example: In model 3, it is known that the car cannot cross the obstacle while it is changing lanes. Thus if $t_1 = \delta$ & $lcFlag = 1$ is true then $X_{angel} = X_{dn}$ will not be true. Conditions such as this can be made true at the outset by hiding the succedent and applying quantifier elimination. This eliminates a number of branches that would have to be proved otherwise.

2. Intelligent grouping of the invariant can also reduce the size of the proof. Model 3 onwards, the invariants have been divided into two groups, one which deals with the variables that change because of the ODE and the other which does not. This strategy allows the two parts of the invariant to be separated early on in the proof using boxAnd. After separation, the proof for the latter branch is fairly straight forward while the former branch also has a smaller invariant to work with. The proof for the latter branch is looked at in more detail here.
   Now that it is known that the invariant is independent of the ODE, the ODE can be totally eliminated from the branch. This is achieved by performing an MR with the invariant. MR results in two formula: one where the ODE is absent while the rest of the formula is same as before, and second where it has to be proved that new invariant can prove the old invariant after running the ODE. The old and new invariant are the same and the ODE does not change the invariant. Thus, this is simply proved using GV.

3. It was observed during the proving process that it takes substantial amount of time (about 5-10 minutes) for KeYmaeraX to solve an ODE. Since a large number of ODEs have to be solved for a reasonably complex model (more than 50), it was not feasible to try proving the model by solving the ODEs. Thus an alternate approach was used. Consider the two ODEs that were used in the model 3.
   ODE 1:

$$X'_{angel} = V_{angel}, t' = 1 \ \wedge \ V_{angel} >= 0 \ \wedge \ t <= T \ \wedge \ X_{angel} <= \frac{D}{2} \ \wedge \ X_{angel} <= X_{dn}$$

   Rather than solving this ODE, it is simplified using a differential cut (dC) with $X_{angel} = old(X_{angel}) + V_{angel} * t \ \wedge \ t >= 0$. Two conditions are obtained after dC: One where the new formula becomes a part of the domain constraints and second where the validity of the new formula has to be proved. The former is proved using dW(differential weakening) and the latter using dI(differential Invariant). This substantially reduces the proving time.

   Similarly for ODE 2:

$$X'_{angel} = V_{angel}, t' = 1, t'_1 = 1 \ \wedge \ V_{angel} >= 0 \ \wedge \ t <= T \ \wedge \ t_1 <= \delta \ \wedge \ X_{angel} <= \frac{D}{2} \ \wedge$$

$$X_{angel} <= X_{dn}$$

   dC with $X_{angel} = old(X_{angel}) + V_{angel} * t \ \wedge \ t >= 0 \ \wedge \ t_1 >= 0 \ \wedge \ t_1 = old(t_1) + t$ is used to simplify the formula and then dW and dI are applied.

4. There has been a consistent effort to reduce the number of variables involved in the model to simplify the proving process. One major example of the same has be the elimination of Y_{new} from model 1 to model 2 and 3. It was identified while proving these new models that since the team was only dealing with two lanes, it was much simpler to use Y_{angel} itself in form of '$1 - Y_{angel}$' to represent the new lane that the car entered, rather than Y_{new}.

5. Another shift from model 1 to model 3 is the introduction of the X_{dn} variable. Using X_{dn} to represent the next obstacle rather than working with X_{d0} and X_{d1} saves a lot of code. This can be understood as if the model were to be built using X_{d0} and X_{d1} then car would have had to localize itself with regard to the next obstacle. Depending on which one would be the next obstacle, different code sections would have to be written. X_{dn} eliminates the need to have separate code sections.

6. Since KeYmaeraX does not allow running multiple proof steps for separate branches in parallel, it is an effective time saving strategy to separate out different branches of the proof into separate proofs and then work on them separately, and later join them together.

8.3 Model 1

Model Lane changing on a 2 lane highway

```
1: (
2:    D = 300 ∧ V_SL = 70 ∧ δ = 1 ∧
3:    X_angel = 0 ∧ Y_angel = 0 ∧ V_angel <= V_SL ∧ V_angel > 0 ∧
4:    X_d0 >= D/2 ∧ X_d0 < D ∧ Y_d0 = 0 ∧ Y_new = 0 ∧
5:    V_SL * δ <= D/2 ∧
6:    V_angel * δ < (X_d0 − X_angel) ∧
7:    T > 0 ∧ t_1 <= δ ∧ lcFlag = 0
8: )
9:    →
10: [
11: {
12:    {
13:      ?(lcFlag = 0);
14:      {
15:         ?((Y_angel = Y_d0) ∧ X_angel < X_d0 ∧ (V_angel * δ + V_angel * T >= X_d0 − X_angel));
16:         {
17:            lcFlag := 1; t_1 := 0;
18:            {
19:              ?(Y_d0 = 1); Y_new := 0;
20:              ∪
21:              ?(Y_d0 = 0); Y_new := 1;
22:            }
23:         }
24:         ∪
25:         ?¬(Y_angel = Y_d0 ∧ X_angel < X_d0 ∧ (V_angel * δ + V_angel * T >= X_d0 − X_angel));
26:            /* else, donothing */
27:         }
28:      ∪
29:      ?(lcFlag ≠ 0);
30:    }
31:    t := 0;
32:    {
33:      ?(lcFlag = 1);
34:         {X'_angel = V_angel, t' = 1, t'_1 = 1 ∧ V_angel >= 0 ∧ t <= T ∧ t_1 <= δ}
35:            if(t_1 = δ){lcFlag := 0; Y_angel := Y_new;}
36:      ⋃
37:      ?(lcFlag = 0);
38:         {X'_angel = V_angel, t' = 1 ∧ V_angel >= 0 ∧ t <= T}
39:    }
40: }*@invariant(
41:    ( ( lcFlag = 0 ∧ (X_d0 ≠ X_angel ∨ Y_d0 ≠ Y_angel) ) ∨ (lcFlag = 1 ∧ X_d0 ≠ X_angel) )
42:       ⋀
43:    ((lcFlag = 1 ∧ X_angel < X_d0) → (V_angel * (δ − t_1 < X_d0 − X_angel) ∧ Y_new = 1 − Y_d0)
44:       ⋀
45:    ((lcFlag = 0 ∧ X_angel < X_d0 ∧ Y_angel = Y_d0) → (V_angel * δ < X_d0 − X_angel))
46:                )
47: ]
48: (
49:    (lcFlag = 0 ∧ (X_d0 ≠ X_angel ∨ Y_d0 ≠ Y_angel) ∨ (lcFlag = 1 ∧ X_d0 ≠ X_angel)
50: )
```

References

[1] Enhancements of V2X Communication in Support of Cooperative Autonomous Driving.

[2] Waymo safety report: On the road to fully self driving. *https://storage.googleapis.com/sdc-prod/v1/safety-report/waymo-safety-report-2017.pdf*, 2017.

[3] Matthias Althoff and John M. Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 2014.

[4] Sarah M. Loos and André Platzer. Safe intersections: At the crossing of hybrid systems and verification. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2011.

[5] Nistor L. Loos S.M., Platzer A. Adaptive cruise control: Hybrid, distributed, and now formally verified. *In: Butler M., Schulte W. (eds) FM 2011: Formal Methods. FM 2011. Lecture Notes in Computer Science, vol 6664. Springer, Berlin, Heidelberg*, 2011.

[6] Stefan Mitsch, Sarah M Loos, and André Platzer. Towards Formal Verification of Freeway Traffic Control. 2012.

[7] André Platzer. Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, 41(2):143–189, Aug 2008.

[8] André Platzer. Differential game logic. *ACM Trans. Comput. Logic*, 17(1), 2015.

[9] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and Decision-Making for Autonomous Vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):187–210, may 2018.

[10] Chairit Wuthishuwong, Ansgar Traechtler, and Torsten Bruns. Safe trajectory planning for autonomous intersection management by using vehicle to infrastructure communication. 2011.