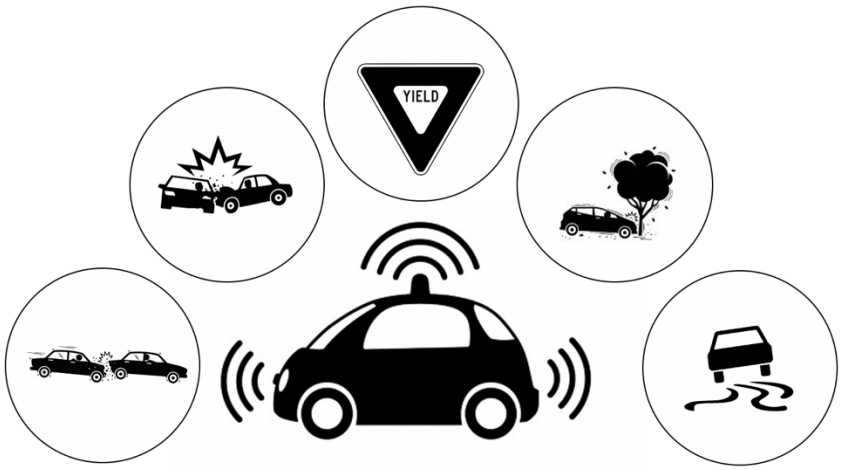
# Formalization and Improvements to the Responsibility Sensitive Safety Model for Self-Driving Cars

Department of Computer Science
Megan Strauss
Carnegie Mellon University
mstrauss@andrew.cmu.edu

December 10, 2021

**Abstract**

In this project, we investigate the Responsibility Sensitive Safety model for self-driving cars as presented by Shalev-Schwartz et al. The model proposed would give guidelines for cars to follow to ensure that they would not collide. This project formalized this into a differential dynamic logic model. Portions of this model were proved using KeymaeraX and suggestions as to why other parts of the model were unable to be proven are investigated. This project then created a modified version of this model which would be easier to implement, which only did necessary computation using a timer, rather than doing infinitely many computations as the original model implies would happen.



# I. Introduction and Motivation

As technology develops, the automation of everyday tasks becomes more and more feasible and appealing. Most recently, larger tasks, such as grocery shopping checkout, have even been automated. The automation of transportation has been an idea that has been progressing for decades, and the idea of a self-driving vehicle has been around with the first working version of one coming out in 1939 from General Motors at the world fair. Self-driving cars have become an important research topic since they are both safer and more convenient than cars controlled by humans. For self-driving cars to be used, however, they must be provably safe, which is where difficulty lies. A way to do this, is to model cars driving and prove the safety of the model.

A hybrid system is a system that has both discrete and continuous dynamics. Since cars are controlled discretely, and move continuously with the effects of physics, it can be considered as such and this project will prove the safety of a model of self-driving cars through hybrid system logic.

It should be noted that creating a system of self-driving vehicles that is provably safe and scalable is extremely difficult. The ultimate goals of automating driving would be to decrease the rate of accidents to 0, which is difficult because there are so many factors that go into the cause of an accident that it is both difficult to list all factors, and even harder to consider all of them when creating a model for self-driving cars to follow. John McDermid explains five challenges that self-driving cars must overcome to be rolled out on a large scale. He mentions social acceptability and government regulations that are holding the field back. More related to technology, he mentions that machine learning is not advanced enough to trust safely for self-driving car applications, since there are too many factors that go into the safety of cars. The other problem he mentions is the fact that there will undoubtedly be obstacles that the car encounters on the road that it has never been exposed to in training, and proving that the car will react safely to these is difficult(McDermid, n.d.). For these reasons, proving that a model for self-driving cars is always safe is near impossible, but this may not be necessary. It is much easier to prove that very few accidents will happen, and this is reasonable since society should be able to tolerate a very small number of accidents. For this reason, it is not needed to consider every possible factor in creating a model for self-driving cars in order to prove safety, and thus the model studied in this paper only takes into consideration the behavior of other cars on the road.

This project was a combination of work done for the 15-424 Logical Foundations of Cyber Physical Systems course at Carnegie Mellon and an independent study done in the same semester. Looking into the model itself and the proofs of the original model in the longitudinal direction were for the purpose of the independent study, and modeling proving the time-triggered models as well as the model for lateral movement were done for the course project.

# II. The Responsibility Sensitive Safety Model

## 2.1 Shalev-Schwartz Proposed Model

The Responsibility Sensitive Safety (RSS) model for self-driving cars was published by Shalev-Schwartz et al which describes a safe and scalable list of rules and claims that no car following

this model would cause an accident. The originally published paper suggests looking at car accidents by looking at the car at fault for the accident. It describes a way of driving that is both safe and non-conservative (does not practice over caution).

The model itself consists of formalizations of five basic rules.

1. Do not hit someone from behind
2. Do not cut in recklessly
3. Right-of-way is given, not taken
4. Be careful in areas with limited visibility
5. If you can avoid an accident without causing another one, you must do it(Shalev-Shwartz et al., 2018).

Note that the model proved in this paper will only cover a subset of these rules.

The purpose of introducing this model was the claim that it is both scalable and safe. The scalability claim comes with regard to the way that the proof that this model is safe can be inductively applied to any number of cars (so long as the precondition, as discussed below, is met). This is not investigated in detail in this project.

The safety claim comes from an induction argument that is loosely proven in their paper. The purpose of this paper is to prove this using Differential Dynamic Logic, which is a logic created to prove the correctness of hybrid systems(Platzer, 2010). This project investigates the situation of two cars on a road and shows that following this model, they will not collide.

The RSS model defines their driving rules by utilizing the idea that two cars should always maintain a well-defined safe distance between the two of them, and if this safe distance is violated, they should respond in a defined "proper response". The paper defines a lane-based coordinate system based on the direction of the road they are driving on. This leads to the assumption that the two cars being formally modelled will be driving on the same road.

## 2.2 Coordinate System

The lane-based coordinate system defines a coordinate of the car's current position with respect to the road it is on. Since roads are often not straight, it lets the I position of the car follow the curve of the car as if it was projected into a straight line. To do this, the road, $r$, is divided into pieces which are either arcs or straight lines. The $y$ position of the car is then with respect to the road's curve and is a coordinate on one of the pieces of the road. The $x$ coordinate of the car is defined as the distance from the center of the road and is with respect to the line perpendicular to the tangent line to the road's curve at any point.

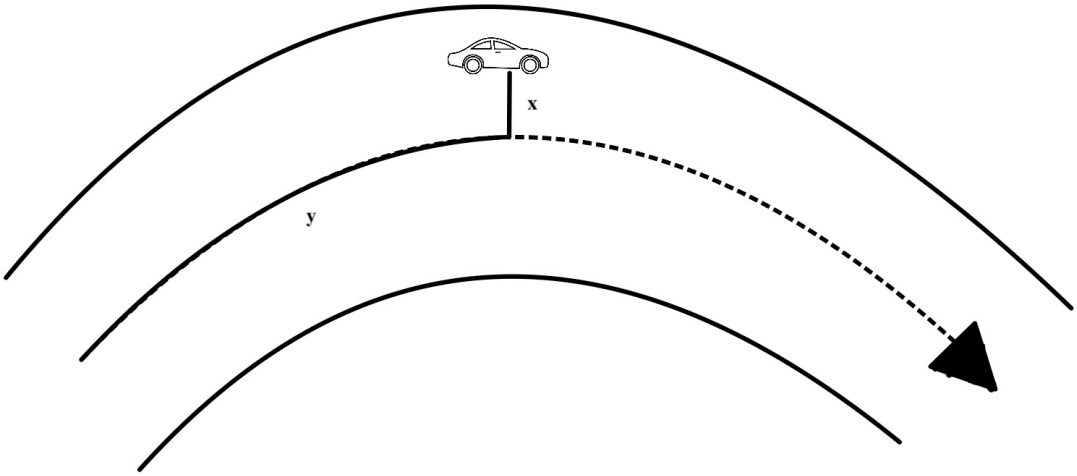



*Figure 1: Diagram of components of car's position in lane-based coordinate system*

## 2.3 Longitudinal Velocity and Safe Distance

The longitudinal velocity that a car is driving can then be referred to as the change in *y* position over time. The longitudinal velocity can be thought of as the forward motion of the car and will be referred to the y direction movement of the car throughout this project.

The safe distance between two cars is defined in two parts. The longitudinal, $d_{min,y}$, and lateral, $d_{min,y}$. It is defined with respect to a few preset constants and the velocities of both cars. Parameter $a_{maxAccel}$ is the maximum acceleration that either car can apply in the direction that they are driving (positive or negative with respect to the road's direction). Parameters $a_{minBrake}, a_{minBrakeCorrect}$ can be thought of as the minimum braking acceleration that will be experienced if the car brakes at full force. $a_{minBrakeCorrect}$ corresponds to the minimum braking value experienced to a car going in the allowed direction of the lane. $a_{maxBrake}$ is then defined as the maximum acceleration that the car will experience while braking at full force. These parameters are defined for both the longitudinal and lateral directions.

The last parameter, $\rho$, corresponds to the minimum response time that the cars will have before changing their behavior to avoid the accident. This can be thought of the amount of time that the cars will remain at an unsafe distance before they should be moving towards a safe distance.

For cars driving in the same longitudinal direction, the longitudinal safe distance is

$$d_{min,y} = max\left(v_1\rho + \frac{1}{2}a_{maxAccel}^{long\rho^2} + \frac{\left(v_1 + \rho a_{maxAccel}^{long}\right)^2}{2a_{minBrake}} - \frac{v_2^2}{2a_{maxBrake}^{long}}, 0\right)$$

For two cars driving in opposite longitudinal directions, we let

$$d_{min,y} = \frac{v_1 + v_{1,\rho}}{2}\rho + \frac{\left(v_{1,\rho}\right)^2}{2a_{minBrakeCorrect}^{long}} + \frac{|v_2| + v_{2,\rho}}{2}\rho + \frac{\left(v_{2,\rho}\right)^2}{2a_{minBrake}^{long}}$$

Where $v_{1,\rho} = v_1 + \rho a_{maxAccel}$ and $v_{2,\rho} = |v_2| + \rho a_{maxAccel}$ (Shalev-Shwartz et al., 2018)

2.4 Lateral Velocity and Safe Distance

The lateral safe distance is defined with respect to some $\mu$ −lateral velocity, defined as follows:

"*Consider a point located at a lateral location* l *at time* t. *Its* μ*-lateral velocity at time* t *is defined as follows. Let* tout > t *be the earliest future time in which the point's lateral position, denoted* lout*, is either* l - μ/2 *or* l + μ/2 *(if no such time exists we set* tout = ∞*). If at some time* $t' \in$ (t, tout) *the point's lateral position is* l*, then the* μ*-lateral-velocity is* 0. *Otherwise, the* μ*-lateral-velocity is* (lout - l)/(tout - t)." *(Shalev-Shwartz et al., 2018)*

The reason it is defined like this is due to the fact that slight fluctuations in lateral velocity are expected, but only lateral fluctuation of a certain magnitude needs to be accounted for. The lateral safe distance is then given by

$$d_{min,x} = \mu + min\left(\frac{v_1 + v_{1,\rho}}{2}\rho + \frac{\left(v_{1,\rho}\right)^2}{2a_{minBrake}^{lat}} - \left(\frac{v_2 + v_{2,\rho}}{2}\rho + \frac{\left(v_{2,\rho}\right)^2}{2a_{minBrake}^{lat}}\right)\right.$$

Where $v_{1,\rho} = v_1 + \rho a_{maxAccel}^{lat}$ and $v_{2,\rho} = v_2 - \rho a_{maxAccel}^{lat}$, regardless of which direction the cars are travelling(Shalev-Shwartz et al., 2018).

2.5 Unsafe Distance and Proper Response

Two cars are then at an unsafe distance if they have both an unsafe lateral and longitudinal distance. The purpose of the safe distance is not to ensure that cars never get closer than a safe distance away from each other, but rather, to ensure that if they do, they will have enough time to react without colliding. The paper describes a proper response in the lateral and longitudinal direction.

For two cars driving in the same direction, the longitudinal proper response for both cars, assuming $c_1$ is the back car,

1. $c_1$ should accelerate at rate less or equal to $a^{\text{long}}_{maxAccel}$ for at most $\rho$ time, and then it must accelerate at rate at most $-a^{long}_{minBrake}$ until reaching a safe longitudinal distance.

2. $c_2$ should accelerate at rate at most -$a^{long}_{maxBrake}$ until reaching a safe longitudinal distance

For two cars driving in opposite directions, the longitudinal proper response, assuming that $c_2$ is driving with a negative velocity

1. $c_1$ should accelerate at rate less or equal to $a^{long}_{maxAccel}$ for at most $\rho$ time, and then it must accelerate at rate at most $-a^{long}_{minBrakeCorrect}$ until reaching a full stop.

2. $c_2$ should accelerate at rate at least -$a^{long}_{maxAccel}$ (since it is driving backwards) for at most $\rho$ time, and at least -$a^{long}_{minBrake}$ until reaching a full stop.

For the lateral response, assume without loss of generality that $c_1$ is to the left of $c_2$. The lateral proper response is then that both cars can drive at any lateral acceleration that has absolute value less or equal to $a^{\text{lat}}_{maxAccel}$. Then

1. $c_1$ should accelerate at most -$a^{\text{lat}}_{minBrake}$ until reaching a $\mu$ −lateral velocity of zero.

2. $c_2$ should accelerate at least $a^{\text{lat}}_{minBrake}$ until reaching a $\mu$ −lateral velocity of zero.

(Shalev-Shwartz et al., 2018).

With this, we can move into formalizing a model for this system.

## III. Event-Triggered Model

Creating an event-triggered model is the most accurate way to translate the idea behind the paper into dL. The model behind this assumes preconditions, then, in a loop, applies a controller, and then follows continuous physical dynamics for an arbitrary amount of time until the controller is applied again.

### 3.1 Variables

Below lists a table of variables used in the model.

| **Name** | **KeymaeraX** | **Description** |
|---|---|---|
| $x_1, x_2, y_1, y_2$ | x1, x2 | Components of cars' positions |
| $v_{1,x}, v_{2,x}, v_{1,y}, v_{2,y}$ | v1, v2 | Velocities of cars in both directions |
| $a_{1,x}, a_{2,x}, a_{1,y}, a_{2,y}$ | a1, a2 | Acceleration of cars in both directions |
| $a_{maxBrake}$ | aMaxBrake | Maximum acceleration that brake can apply |
| $a_{minBrake}$ | aMinBrake | Minimum acceleration that braking at full force will apply in wrong direction of driving |
| $a_{minBrakeCorrect}$ | aMinBrakeCorrect | Minimum acceleration that braking at full force will apply in correct direction of driving |
| $a_{maxAccel}$ | aMaxAccel | Maximum acceleration that can be applied |
| $\rho$ | rho | Reaction time |
| $\mu$ | mu | Minimum lateral velocity to be noticed |

3.2 General Structure of Models

The model proven for this paper was split into three cases.

1. Two cars driving in opposite longitudinal directions (only y-positions considered)
2. Two cars driving in the same longitudinal direction (only y-positions considered)
3. Two cars driving (only x-positions considered)

The argument that this split still ensures a proof of the entire system is due to the assumption that cars cannot drive with velocities below zero, and thus two cars driving in opposite longitudinal directions will stay driving in opposite directions and similarly with two cars driving in the same direction. By the definition of safe distance (that either the cars have a safe longitudinal distance OR a safe lateral distance), we know that in the case that the distance between cars is unsafe, the car needs to react in both longitudinal and lateral directions, but these are independent, so both

models can be occurring simultaneously, and can be split for the purpose of the proof. Note that this means that each of these proofs implies that that the direction that is not considered in the proof is also unsafe (i.e. in longitudinal proofs, assume lateral distance is also unsafe).

Note that since the models are split into the three cases, the position of the car (in whatever direction the model is considering) is considered as `x1` and `x2` for car 1 and car 2 respectively.

The three models follow the same basic structure and differ slightly in their initial assumptions and controllers. This section will discuss the basic structure of the models

3.2.1 Preconditions

First, the models assumes that the braking parameters are all absolute values, so all of these must be greater than zero. We also can safely assume that the maximum braking value is greater than the minimum braking value. From here, we split the model into two scenarios based on whether the cars are going the same or opposite directions in the longitudinal direction. We can do this since we assume that the cars cannot reverse, and thus they will always stay in the same case.

For cars going the same direction. Without loss of generality, the model assumes both cars are driving with positive velocity. There is also an added assumption that the distance between the cars initially is safe. For cars going in opposite directions, the model lets car 2 drive in the wrong direction (negative velocity), without loss of generality, and also assumes a safe distance at the start.

The preconditions are formalized in the KeymaeraX language below.

```
 safeDist(v1, v2)<=x2-x1

& aMaxAccel>0

& aMaxBrake > 0

& rho>0

& aMinBrakeCorrect > 0

& aMinBrake>0

& aMinBrake<aMaxBrake

& aMinBrakeCorrect<aMaxBrake
```

3.2.2 Continuous Dynamics

The dynamics that the car follows in the paper are the basic laws of kinematics, and thus, after setting the acceleration of cars 1 and 2, $a_1, a_2$ respectively, we get that the dynamics that the system follows are

$$\{x_1' = v_1, x_2' = v_2, v_1' = a_1, v_2' = a_{2,}\}$$

In KeymaeraX:

```
{x1' = v1, x2' = v2, v1' = a1, v2' = a2}
```

3.3.3 Controller

The model in the paper describes the cars as having free range over their control unless they are in an unsafe situation, in which case they must follow the proper response. For this reason, there are two controllers in the formalized model. The main controller runs in the case that the distance between the cars is safe. It is able to choose an acceleration that is within the minimum and maximum bounds on acceleration, and then it runs continuous dynamics for arbitrary time with the evolution domain constraint that the distance between the cars stays safe, based on the safe distance for the respective directions of the two cars. This controller is the same for all three models. Below is the regular controller in KeymaeraX.

```
HP control ::= {

        /*Set acceleration and check that it is within bounds*/

        a1 := *;

        a2 := *;

        ?( a1<=aMaxAccel

          & a1>=-aMaxBrake

          & a2>=-aMaxAccel

          & a2<=aMaxBrake);

        /*Run continous dynamics while safe distance*/

        {v1' = a1, x1' = v1, v2' = a2, x2' = v2

            & safeDist(v1, v2)<= x2-x1

}};
```

In the case that the cars are at an unsafe distance, a proper response controller is run, which implements the proper response as described. These are described in detail in later sections.

The final problem for each model chooses which controller to run based on whether the distance between the cars is safe. It makes this decision an arbitrary number of times, representing the cars continuing to drive and check by making these decisions.

```
[{
    /*Choose which controller to use based on distance*/
    {?safeDist(v1, v2)<=x2-x1; control;}
    ++ {?safeDist(v1, v2)>=x2-x1; longProperResponseOpposite;}
   }*
)]
```

3.3 Model 1: Cars driving in opposite longitudinal directions

3.3.1 Preconditions

In addition to the preconditions described above, an extra assumption was needed to provide a proof of this model. This was the fact that the position that car 2 would stop at would be greater or equal to the position that car 1 would stop at if car 1 and car 2 were to respond properly. The other additional precondition was the assumption that car 1 was driving with non-negative velocity and car 2 was driving with non-positive velocity and the fact that the position of car 1 is less than the position of car 2. The explanation for the last assumption is as follows:

The only case that is interesting to consider for car 1 driving with non-negative velocity and car 2 driving with non-positive velocity is the case that car 1 has position less than car 2. Consider the following two cases

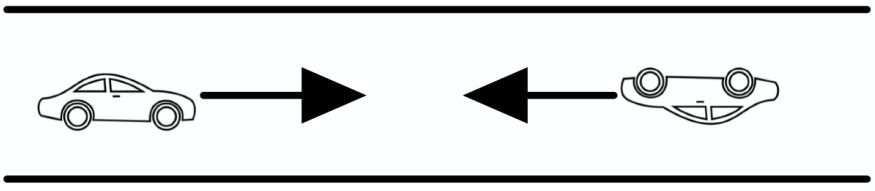


*Figure 2: Position of car 1 less than that of car 2*

*Figure 3: Position of car 1 less than that of car 2*

In the second case, there would never be a collision because the cars would continuously get further from each other. For this reason, we let our initial assumption be the fact that car 1 has position less or equal to car 2.

The extra preconditions are as follows:

```
        x2 - (v2^2)/(2*aMinBrake)
           >= x1 - (v1^2)/(-2*aMinBrakeCorrect)
      & x1<=x2
      & v1>=0
      & v2<=0
```

3.3.2 Proper Response Controller

The proper response controller for this model is a formalized version of the proper response defined in the paper. The acceleration is set to a valid response acceleration, i.e. $a_{1,y} \leq -a_{minBrakeCorrect}$ and $a_{2,y} \geq a_{minBrake}$, and the continuous dynamics are run until both cars reach a full stop. When either car reaches a full stop, it must wait for the other car to reach a stop, and then the cars have the option to go back to the original controller.

```
HP longProperResponseOpposite ::= {
        /*Change acceleration to correct unsafe distance*/
        a1 := *; ?(a1<=-aMinBrakeCorrect);
        a2 := *; ?(a2>=aMinBrake);
```

```
        {{v1' = a1, x1' = v1, v2' = a2, x2' = v2
            & (v1>=0 & v2<=0)
            &     x2 - (v2^2)/(2*aMinBrake)
               >= x1 - (v1^2)/(-2*aMinBrakeCorrect)}
        /*Set acceleration*/
        if (v1=0){a1 := *; ?a1<=0;} if(v2=0) {a2 := *; ?a2>=0;}
        }*
     /*Don't move on until both cars have reached a full stop*/
        ?v1=0&v2=0;
};
```

3.3.3 Postcondition

The postcondition, using the definition from the paper that two cars will only collide if they overlap paths (i.e. if their x-positions are equal they have not yet collided), we check that the position of car 1 has not exceeded the position of car 2.

```
x1<=x2
```

3.3.4 Proof strategy

The strategy used to prove this was to add an invariant to the loop in the main controller. This one stated that any run of either controller, the position that car 1 would stop at would be less or equal to the position that car 2 would stop at, if car 1 applied $-a_{minBrakeCorrect}$ acceleration and car 2 applied $a_{minBrake}$ acceleration. The invariant also includes the fact that the velocities of the two cars stay within the desired range. Formalized this is the invariant.

```
@invariant(  v1>=0 & v2<=0 &
               x2 - (v2^2)/(2*aMinBrake)
            >= x1 - (v1^2)/(-2*aMinBrakeCorrect))
```

In the regular control case, there was no extra need for loop invariants, since the safe distance definition implied the invariant. In the proper response case, another invariant for the loop that continues running the continuous dynamics of the proper response (and deals with velocity when it reaches 0). This invariant ensured at the stopping position of car 1 was less or equal to that of

car 2 using the fact that the current position of car 1 was less than the stopping position of car 1 and the current position of car 2 was greater or equal to the stopping position, which means that the cars will never collide.

```
@invariant(

              v1>=0

            & v2<=0

            & x1 <= x1 - (v1^2)/(2*-aMinBrakeCorrect)

            & x2 >=x2 - (v2^2)/(2*aMinBrake)

            & x2 - (v2^2)/(2*aMinBrake)

          >= x1 - (v1^2)/(-2*aMinBrakeCorrect));
```

3.4 Model 2: Two cars driving in the same longitudinal direction.

3.4.1 Preconditions

The only extra assumption needed for this case was the fact that both cars drive with velocities that are non-negative. We also, without loss of generality, assume that car 1 is the back car and set its horizontal position to be less than that of car 2.

```
v1<=0 & v2<=0 & x1<=x2
```

3.4.2 Proper Response Controller

The proper response controller sets the values of $a_{1,y} \leq -a_{minBrake}$ and $a_{2,y} \geq -a_{maxBrake}$. Then it runs continuous dynamics until the situation becomes safe again, or until the cars collide (one of these two things must happen).

```
HP longProperResponseSame ::=

        /*Change to proper response accelerations */

        a1 := *; ?a1 <= -aMinBrake;

        a2 := *; ?a2>=-aMaxBrake;

        /*run continuous dynamics until safe*/

        {v1' = a1, x1' = v1, v2' = a2, x2' = v2
```

```
          & safeDist(v1, v2)<=x2-x1 & v1>=0 & v2>=0 & x1<=x2}

    /*Run until safe distance is reached or the cars collide*/

     ?safeDist(v1, v2)<=x2-x1 | x2<x1;

    };
```

3.4.3 Postcondition

Again, the postcondition, using the definition from the paper that two cars will only collide if they overlap paths (i.e. if their x-positions are equal they have not yet collided), we check that the position of car 1 has not exceeded the position of car 2.

```
x1<=x2
```

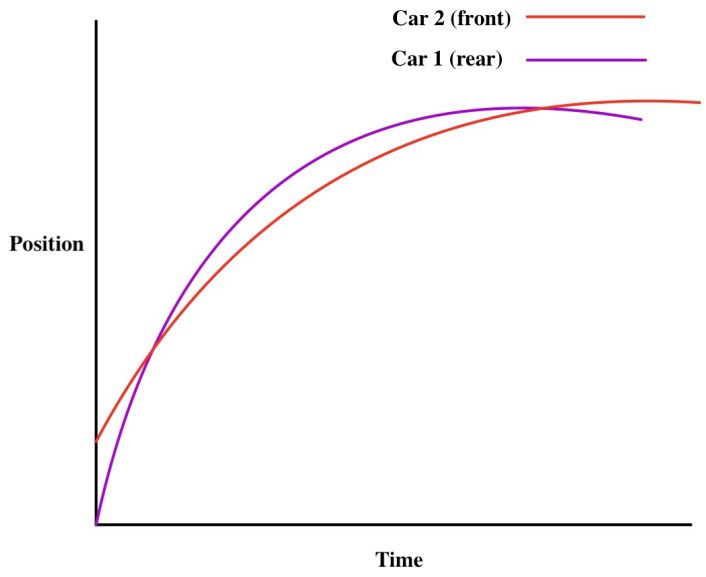
3.4.4 Proof Strategy

The proof strategy for this model was similar to the previous model in that there was an invariant in the main loop, which shows that no matter which controller was run, the cars would never collide. This invariant was that the velocities would stay in the same direction, the position of car 1 would never surpass the position of car 2 and that after any run, the distance between the cars was safe.

```
@invariant(  v1>=0

           & v2>=0

           & x1<=x2

           & safeDist(v1, v2)<=x2-x1)
```

The control case then, did not need any other invariants, since the safe distance evolution domain constraint implied this invariant. For the proper response case, an invariant on the dynamics was added to aid the proof, but more work is needed to be done to find an invariant for this. There is a possibility that the safe distance metric for this case is incorrect, since the definition seems to be derived from rearranging the kinematic equations representing the points where each of the cars stops. There is a case, however that the two cars pass each other twice while in the proper response, for example if car 1 is driving with a very large velocity.

The proof in the original paper does not cover this case and makes the claim that the smallest distance between the two cars is when they reach a full stop or when they begin to respond, which is not the case, considering the following graph representing the two cars' motion.




The rear car passes the front car in the middle of the time period but stops in a position behind the front car. In this case the cars would have collided.

For this reason, the proof of this model was not able to be completed.

3.5 Model 3: modeling lateral direction movement

3.5.1 Preconditions

The only extra precondition for this model is the fact that we have some $\mu$ constant which must be greater than 0.

```
mu>0
```

3.5.2 Proper Response Controller

The proper response controller for this model is similar to the others, where the cars are able to move at whatever acceleration they want for up to $\rho$ time, and then they must complete the response, where $a_{1,x} \leq -a_{minBrake}$ and $a_{2,y} \geq a_{minBrake}$ as defined. Then the continuous dynamics are run until car 1 has a non-positive velocity and car 2 has a non-negative velocity. This ensures that since car 1 is to the left of car 2, they are driving in opposite directions away from each other.

```
HP lateralProperResponse ::= {

/*Count rho time where acceleration within bounds*/
```

```
        t :=0;
        a1 := *; a2 := *;
        ?( a1<=aMaxAccel
          & a1>=-aMaxBrake
          & a2>=-aMaxAccel
          & a2<=aMaxBrake);
        /*run dynamics for up to rho time*/
   {v1' = a1, x1' = v1, v2' = a2, x2' = v2, t' = 1 & t <= rho};
        /*Begin responding,
          set acceleration /*
          a1 := *; ?a1 <=-aMinBrake;
        a2 := *; ?a2 >=aMinBrake;
     /*run dynamics until velocity reaches 0*/

{v1' = a1, x1' = v1, v2' = a2, x2' = v2 & (v1!=0 | v2!=0)}
     /*Check that velocities are as desired before stopping*/
        ?v1<=0 & v2>=0;
    };
```

3.5.3 Postcondition

Once again, the postcondition should just be that the cars have not exceeded each other, and thus is

```
x1<=x2
```

3.5.4 Proof Strategy

The invariant for the main loop for this case was simply the post condition. Each case was easily provable since the cars are able to move in both directions in this case, thus, if they ever get too close to each other, they simply start moving further away and the definition of the safe distance implies that the cars will not collide.

## IV. Towards a Time-Triggered Model

The model described above is the model that was proposed in Shalev-Schwartz 2018 and by design is an event triggered model. The events that it waits on is the violation of the safe distance guideline, and thus, the value of the safe distance needs to be computed continuously since it relies on the velocity of both cars which is continuously changing. Despite making the model efficient, the hardware design of such a system is not feasible, and a time-triggered model would be a more practical option. For this model, rather than computing the safe distance at each time point, the controller would instead check that the distance would be safe within a set time interval. For this, we define a maximum time before checking the controller again.

### 4.1 Preconditions and Postconditions

The preconditions for the time-triggered model are the same as the event-triggered model, as they describe the values of the constants and the initial positions between the cars. The postcondition for this model is also the same as the event-triggered model since the ultimate goal is the same, to not have a car collision.

### 4.2 Time-Triggered Model for Cars Driving in Opposite Longitudinal Directions

#### 4.2.1 Regular Controller

The general idea for the model is similar to that of the event triggered model, however each base controller needs to have a way to define whether the distance is safe for a full period of T time. If this is the case, then both cars can accelerate at any value within bounds and then the regular controller can run continuous dynamics for at most T time, and then check again.

Note that the continuous dynamics are the same except for the fact that there is a time parameter that moves at rate 1. The domain constraints then do not involve the safe distance calculation, since this is done before choosing which controller to run. Instead, domain constraint makes sure the velocity stays safe, and otherwise makes sure that the time that has passed does not exceed the maximum. Keeping track of the velocity is applicable here since we make the assumption that the cars move with velocities only greater or equal to zero.

```
HP control ::= {

        /*Set acceleration*/

        a1 := *; a2 := *;

        ?(  a1<=aMaxAccel
```

```
          & a1>=-aMaxBrake

          & a2>=-aMaxAccel

          & a2<=aMaxBrake);

        {v1' = a1, x1' = v1, v2' = a2, x2' = v2 , t' = 1

            & v2<=0

            & v1>=0

            & t<=T            }

};
```

4.2.2 Proper Response Controller

To begin to think about the proper response portion of the proof, it is easier since the proper response ensures that the accelerations of the two cars are working towards a safe distance. This means that there is no need to keep track of time in the case that the cars are responding. This gives us the following for the proper response in the time triggered model, which is identical to that in the event triggered model.

```
HP longProperResponseOpposite ::= {

        t := 0;

        a1 := *; ?(a1<=-aMinBrakeCorrect);

        a2 := *; ?(a2>=aMinBrake);

   /*Proper response needs to be followed until both cars reach

    a full stop, thus time does not need to be kept track of*/

        {

           {v1' = a1, x1' = v1, v2' = a2, x2' = v2

            & (v1>=0 & v2<=0)

        }

        if (v1=0){a1 := *; ?a1<=0;} if(v2=0) {a2 := *; ?a2>=0;}
```

```
        }*

        ?v1=0 & v2=0;

    };
```

4.2.3 Main Controller

The time-triggered takes a bit more computation to check that each time point is safe. For cars driving in the opposite directions, if the distance is safe at the beginning of the time period, and the distance is safe (following kinematics) after the end of the time period, the distance is safe at all times in between (assuming the cars will not drive backwards, which we require in our model). The main loop of the model formalizes this as follows:

```
/*If current position or position in T time points is unsafe,
run proper response, else run control*/

{?safeDist(v1, v2)<=x2-x1

& safeDist(v1 + T*aMaxAccel , v2 - T*aMaxAccel)<=

(x2  + v2*T - aMaxAccel*T/2)  - (x1 + v1*T + aMaxAccel*T/2);

    control;}

    ++

 {?safeDist(v1, v2)>=x2-x1

| safeDist(v1 + T*aMaxAccel , v2 - T*aMaxAccel)

>= (x2  + v2*T - aMaxAccel*T/2)  - (x1 + v1*T + aMaxAccel*T/2);

    longProperResponseOpposite1;}

    }*
```

4.2.4 Proof strategy

The proof for this case used an invariant in the main loop of the controller which stated that the stopping position of the first car, if it were to use the minimum braking constant in the correct direction is smaller than the stopping position of the second car if it were to use the minimum braking constant. This is equivalent to the point where the cars' velocities reach zero when following the proper response. This works due to the fact that in the proper response car 1 must apply acceleration less or equal to $-a_{minBrake\text{Correct}}$ and car 2 must apply acceleration greater or

equal to $a_{minBrake}$ (this makes the inequalities hold if the acceleration applied is not equal to the exact values).

```
@invariant(  x1 - (v1^2)/(2*-aMinBrakeCorrect)

          <= x2 - (v2^2)/(2*aMinBrake))
```

For the regular control case, since the definition of safe distance implies the invariant, no additional invariants were needed. For the proper response case, an invariant on the dynamics was added to state that at any point, if the cars were to travel up until the final time (T-t), the velocity that the cars would be at would imply that their stopping positions were as desired.

```
@invariant(

   x1 - ((v1 - aMinBrakeCorrect *(T-t))^2)/(2*-aMinBrakeCorrect)
<= x2 - ((v2 + aMinBrakeCorrect *(T-t))/(2*aMinBrake))
```

This was then enough to show the loop invariant.

4.3 Time-Triggered Model for Cars Driving the Same Longitudinal Direction

This model was omitted since no proof was found for the event-triggered model and more work is needed to find if the safe distance metric is valid for this case.

4.4 Time-Triggered Model for Lateral Movement

4.4.1 Preconditions

The preconditions and postconditions for this problem were identical to that of the event-triggered lateral model, only adding the fact that the maximum time constant is greater than zero.

4.4.2 Controller

The controller for this model was very similar to the controller for regular lateral motion. Rather than checking if the current distance is safe, however, it checked that the current distance is safe and the distance between the cars, if they were to get as close together as possible, was safe. This meant both accelerating towards each other at $a_{\mathrm{maxAccel}}$. If this was the case, we allowed the regular controller to run. If this was not the case, the proper response was taken.

The main deciding controller is as follows:

```
{

    /*Choose which controller to use based on distance*/
```

```
    /*Check closest that cars would get to each other is safe
and current

    position is safe, means that everything is safe*/

    {?safeDist(v1, v2)<=x2-x1

    & safeDist(v1 + T*aMaxAccel , v2 - T*aMaxAccel)<=

          (x2  + v2*T - aMaxAccel*T/2)

          - (x1 + v1*T + aMaxAccel*T/2);

    control;}

    ++

    {?safeDist(v1, v2)>=x2-x1

    | safeDist(v1 + T*aMaxAccel , v2 - T*aMaxAccel)

  >= (x2  + v2*T - aMaxAccel*T/2)  - (x1 + v1*T +aMaxAccel*T/2);

    properResponse;}

    }*
```

The regular controller is identical to the original regular controller except for the domain constraint that just checks that the time is less than the maximum time before calculating safe distance again.

```
HP control ::= {

 /*Set acceleration and check that it is within valid bounds*/

        a1 := *;

        a2 := *;

        t := 0;

        ?(a1<=aMaxAccel

          & a1>=-aMaxBrake

          & a2>=-aMaxAccel
```

```
    & a2<=aMaxBrake);

  /*Run continous dynamics while time is valid*/

  {v1' = a1, x1' = v1, v2' = a2, x2' = v2, t' = 1

      & t<=T }};
```

The proper response controller is, as in the case with opposite longitudinal directions, identical to the original model, since in this case safe distance does not need to be recomputed.

4.2.3 Proof strategy

The proof strategy for this model was very similar to the strategy for the original event-triggered model. The loop invariant for the main controller is simply the postcondition, and an invariant for the case with the proper response stated that the cars were moving towards a safe distance, and that they would reach stopping positions such that car 1 stayed to the left of car 2.

```
@invariant(

   x1 - ((v1 - aMinBrake *(T-t))^2)/(2*-aMinBrake) <=

  x2 - ((v2 + aMinBrake *(T-t))/(2*aMinBrake))
```

Using this, the proof of this model was able to complete.

## V. Discussion

This project investigated the Responsibility Sensitive Safety model for self-driving cars and formalized it in KeymaeraX. The goal of the project was to formally prove the model; however, this could not be done for the case that the cars were driving in the same direction due to the fact that the definition of the safe distance does not seem to take into account enough variables. However, a counterexample was also not found, and thus more work would be needed to show that this was not the case.

In any case, there must be some safe distance that cars can keep (no matter how large) such that the idea behind this model would work, so even if this specific value for safe distance is not valid, it could be replaced by another formula, even if this value might not be as efficient.

This project also accomplished proving a time-triggered model for the RSS model, which gives a more practical and implementable version of this model for use in real cars. Although this model does not take into consideration everything that a car would encounter on the road, proving that cars will not collide with each other is a great contribution to the idea of self-driving cars and a great step forward.

Moving forward with this project, the next steps would be to find a way to either prove the case of the cars moving in the same direction, or to find another formula for a safe distance that would be provable.

## VI. Deliverables

The deliverables for this project are attached and are as follows:

A model and proof for

1. Event-triggered model for cars driving in the opposite longitudinal directions
2. Time-triggered model for cars driving in the opposite longitudinal directions
3. Event-triggered model for lateral motion
4. Time-triggered model for lateral motion

This folder also contains 2 unproven models

1. Event-triggered model for cars driving in the same longitudinal direction
2. Time-triggered model for cars driving in the same longitudinal direction

Bibliography

McDermid, J. (n.d.). *Autonomous cars: Five reasons they still aren't on our roads*. The Conversation. Retrieved November 24, 2021, from http://theconversation.com/autonomous-cars-five-reasons-they-still-arent-on-our-roads-143316

Platzer, A. (2010). Differential Dynamic Logics: Automated Theorem Proving for Hybrid Systems. *KI - Künstliche Intelligenz*, *24*(1), 75–77. https://doi.org/10.1007/s13218-010-0014-6

Shalev-Shwartz, S., Shammah, S., & Shashua, A. (2018). On a Formal Model of Safe and Scalable Self-driving Cars. *ArXiv:1708.06374 [Cs, Stat]*. http://arxiv.org/abs/1708.06374