Carnegie Mellon University

15-424

Foundations of Cyber Physical Systems

NASCAR Refueling Challenges: The Strategy behind a Pit Stop

Connie Wang
crwang

May 1, 2016

Abstract

Deciding when to refuel is a huge challenge in NASCAR races because teams are prohibited from monitoring exact fuel levels. The only information that teams can gather is on pit stops, where they can calculate how much fuel was added, and they can know the car has been filled to capacity. This leads to a lot of guesswork on the part of the teams, which can result in running out of fuel on track, which not only slows down the car, but also is highly dangerous since competitors' cars are still traveling upwards of 200mph. This paper verifies two models of a NASCAR racecar with different constraints for the fuel consumption and refueling process. Both models were proven to be safe in that the car always stays on the circular track and the car never runs out of fuel. Therefore, the controls within my models outline strategy for which NASCAR racecars can guarantee that there is always a control decision such that the racecar can traverse the racetrack without running out of fuel.

Contents

1 Introduction

In NASCAR, fuel plays a hugely significant role in the outcome of the race. Races are hundreds of laps with most tracks ranging between $\frac{1}{2}$ - $2\frac{1}{2}$ miles long, forcing teams to make multiple pit stops in order to have enough fuel to finish the race.[5] During a pit stop, the "pit crew" of mechanics can change out the tires and refuel in around 11 seconds, but with cars traveling upwards of 200 mph, 11 seconds can be the difference between 1st place and 20th place.

Changing tires is also a significant part of the pit stop strategy because the tire rubber degrades throughout the race as the cars lap the track. However, refueling is the limiting factor in the total time that a pit stop requires because refueling is capped at a specific rate (for safety purposes), and so decisions about refueling often have more of an impact than decisions about changing tires. In this vein of thought, this paper analyzes the decisions that a NASCAR team faces when devising a strategy for refueling in terms of a racecar as a cyber physical system.

Cyber physical systems (CPS), which deal with the intersection of computer systems and the physical world, must deal with both continuous dynamics and discrete decisions. CPS model real world systems, providing abstractions of the system which can be analyzed and verified. The verification of cyber physical systems allows a certain peace of mind, although verification of a model does not guarantee safety of the actual system. Therefore, any models in CPS must reflect reality for maximum value. We rely on cyber physical systems in everyday life, and so verification of these systems is a safety critical skill for those in the industry. Accurate models of refueling are especially vital in NASCAR because of the restriction on the sensors teams are allowed to have. In this paper, I will discuss multiple models of refueling in NASCAR and the corresponding proofs that guarantee safety.

2 Related Work

NASCAR pitting strategy has been studied in depth, as it is such an important piece of competitive racing. Deck et al discussed game theory strategies for the sequential pitting decisions, in order to optimize the number of cars a given driver can pass. [1] However, that paper studied pit stops purely as a tire exchange as fresh tires have better grip on the track and therefore can withstand higher velocities through the turn without losing traction. The paper did not take into account the need to have a pit stop to refuel, and so those game theory based strategies are not accurately representing racing conditions.

Additionally, many have done extensive research into fuel consumption and optimizing fuel efficiency under different conditions [4]. In NASCAR races, efficiency isn't the goal, but understanding fuel consumption guides decisions on pit strategies. For example, in NASCAR races, fuel consumption is typically much higher when the car is driving "in traffic" compared to when the car is in clean air.[3] On the other side, however, lie the benefits of drafting. If a car is in the right position following another car, then it can ride the "draft" behind the lead car and use less fuel. Because the lead car is redirecting the flow of air around its car, a talented driver could slip into the air stream behind the lead car and consequently have less air resistance to deal with, boosting speeds by around 5mph.[2]

Figure 1: The lead cars create an air stream that following cars can use to draft, reducing drag, but there is also turbulence that can increase drag.

3 Modeling Decisions

3.1 Constant Velocity

For the sake of simplicity, I started with all initial models having a constant velocity. Introducing acceleration would have added a lot of arithmetic because the fuel consumption would need to be calculated by calculating the time it would take to cover a certain distance given an acceleration. Being able to calculate the time would require solving the kinematic equation $d = v_intial * t + \frac{1}{2}a * t^2$ for the time t, which involves a square root. Using square roots in Keymaera X is significantly slower than arithmetic without square roots, and so I chose to start with constant velocity for ease of analysis. During NASCAR races, speeds are variable, but typically fall in the range right around 180mph - 200mph, so assuming a constant velocity is a relatively small compromise to make for modeling the actual race. The real compromise for not including acceleration and deceleration comes from modeling the actual pit stop, but those are infrequent enough that the average velocity still comes out fairly high.

3.2 Time Triggered vs Event Triggered

My initial models are time triggered models, in that the ordinary differential equation (ODE) can only run for a maximum of time T. Given the constant velocity assumption, limiting the run of the ODE by a time T is essentially equivalent to defining a lap time T. Therefore, the system will have a control decision at least every time T interval (although the controls could be triggered in less than time T).

My secondary model is an event triggered model, in that the ODEs can only run while in a certain domain, which is the part of the track without the pit lane. In this model (Appendix A.3), the control will be triggered when the car enters pit lane ($x >= 0$). Typically, CPS avoid event triggered models because of the implementation challenge in detecting events. However, if the model is implemented as a NASCAR driver making decisions instead of an autonomous system, then a human can detect when they are in a certain location fairly easily.

3.3 Single Car

Because of the complex forces behind drafting and the effect of aerodynamics on fuel efficiency, and the general complication of modeling 40 cars, I instead chose to focus my model on the strategy of a single car, assuming fuel consumption is dependent only on the velocity. Because of the exclusion of other cars, my models also implicitly assume that there are no crashes or other racing incidents, which would be accurate in an ideal world. Unfortunately, many NASCAR races do have multiple crashes, but as crashes are unplanned and often end the race for the involved cars, there's no reason to include a crash in modeling a strategy for refueling.

3.4 Circular Track

In NASCAR, 34 out of 36 races are oval tracks navigated counter-clockwise. However, I modeled the tracks as circles in order to simplify the arithmetic. Because the tracks are already ovals with relatively short straightaways, modeling the tracks as circles preserves the general idea of circular motion without compromising too much on the reality of a NASCAR race. My models also preserve the concept of counterclockwise motion in order to match the reality of a NASCAR race.

Figure 2: NASCAR races are predominantly oval tracks, like the Dover International Speedway, with a pit lane extending the length of one of the straightaways.

Modeling movement along a circular track was the subject of a few of the labs for CMU course 15-424, and I used those labs as a starting point for my models. Since sine and cosine are not expressible in the real arithmetic implemented in Keymaera X, counterclockwise circular motion at a specific velocity v is expressed with the following ODEs.

$$\{x' = v * y, y' = -v * x, dx = -dy, dy = dx\}$$

In this model, dx, dy represent the direction vector of the velocity, and x, y represent the Cartesian coordinates of the vehicle moving in a circle. Therefore,

$$\begin{aligned} x &= rad * sin(t) \\ y &= rad * cos(t) \\ dx &= cos(t) \\ dy &= sin(t) \end{aligned}$$

3.5 Safety Conditions

Since I decided to approximate the track as a circle, then the first safety requirement is that the car always stays on the track.

$$x^2 + y^2 = rad^2$$

This property is important because the car should not be able to cut across the track to get to the "pit area" to refuel. Instead, all cars must follow the track and have enough fuel to get around.

The second safety property is the more relevant property about fuel, which is that the fuel never runs out while the car is on the track.

$$fuel \geqslant 0$$

This property is the focus of this paper, as proving that fuel is always greater than zero will prove that there is a strategy to finish the race without running out of fuel on the track.

3.6 Efficiency Condition

Additionally, as a race, all teams are actively trying to push the limits on how far they can go without refueling, and so my model represents that in the controls by only refueling if the car does not have enough fuel to last another lap. In the initial models, this condition is represented by testing the fuel value and only stopping if the fuel would run out before the next lap.

4 Model Details

4.1 Linear Model

In my initial linear model (Appenix A.1), I use a linear approximation to model fuel consumption. Using a linear model will simplify the arithmetic and since NASCAR cars rarely travel less than 60mph, a linear model is a good approximation of how fuel is actually consumed and guarantee safety. Fuel consumption was modeled as having a negative derivative based on the slope of the fuel consumption line.

$$fuel' = -fc * v$$

Where fc is a constant representing the fuel consumption proportional to the velocity.

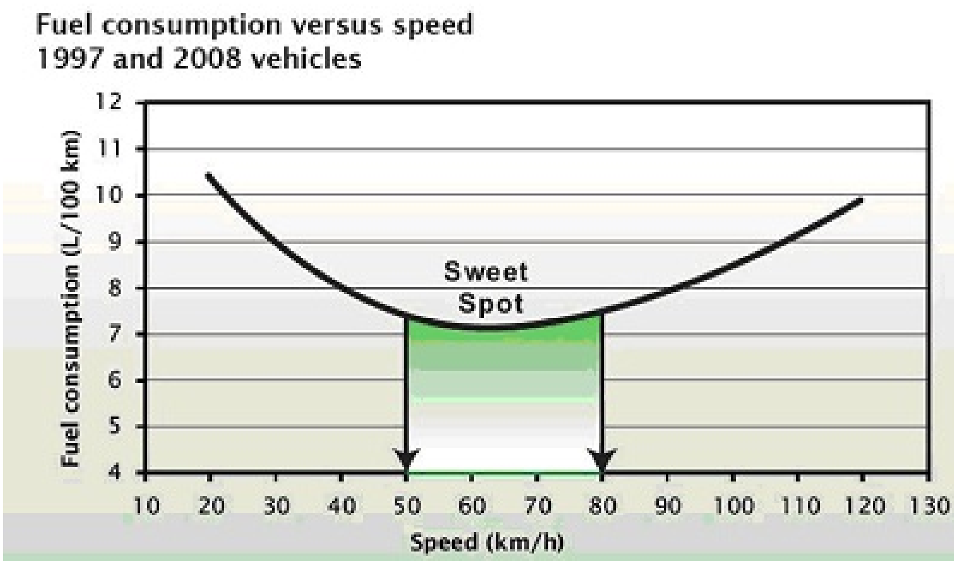

Figure 3: Fuel consumption has follows a curve that mostly maps to a quadratic curve, but above 60mph, a linear model could be fairly accurate.

4.2 Quadratic Model

Technically, however, fuel consumption follows a more complex curve. In a study done by the US Department of Transportation, a model of fuel consumption was developed based on regression analysis. [4]

$$FC = \frac{1}{FE}$$
$$FE = a\left(\frac{T}{2} + b\right)^c$$

where:
FC is fuel consumption in L/km
FE is fuel efficiency in km/L
T is torque in N-m
a, b, c are regression coefficients

So in my second model, (Appendix A.2), I use a quadratic approximation to model fuel consumption. This model has a more accurate representation of fuel consumption based on the US DOT model, as that model shows that there is not a strictly linear relationship between fuel consumption and torque. Note: torque and speed are linearly related, an increase in torque is going to accelerate the car and therefore is associated with an increase in speed.

$$fuel' = -(fc * v * t + c)$$

Where fc and c are the constants of fuel consumption.

4.3 Proposed Linear Event Triggered Model

A progression up from those models would be to add an abstraction of pit lane. This new model (Appendix A.3) allows the car to refuel only on one half of the track, e.g. when $x \geqslant 0$. This requires that the controls consider how much fuel will be required to return to the pit lane, not how much fuel would be required to last one more time interval. Therefore, when a car is in an area where it can make a pit stop, then when deciding whether or not to stop, it must take into account if there is enough fuel to get back to pit lane if it doesn't stop now. To approximate how much time it would take to get back to the current position, I use the current velocity and the bounding box that encloses the circular track (the box with perimeter $8 * rad$). This bounding box will overestimate the track distance, which can guarantee the car can always make it back to pit lane.

5 Proofs

The linear and quadratic model proofs had similar structures. Because of the two distinct postconditions, the proof had two major branches, one to prove that the car always stays on the track (Section 5.1.1), and the other to prove that the car always has enough fuel (Section 5.1.2).

5.1 Invariants

5.1.1 Circular Motion

The invariants necessary for circular motion were fairly straightforward.

$$x^2 + y^2 = rad^2 \tag{1}$$
$$dx^2 + dy^2 = 1 \tag{2}$$
$$dx * v = -y \tag{3}$$
$$dy * v = x \tag{4}$$
$$rad >= 0 \tag{5}$$

Formula (1) is also one of the safety conditions; the car must be on the track at all times. Formula (2) is an invariant that guarantees that dx and dy actually represent the direction vector of velocity. Formulas (3) and (4) describe the relationship between the direction vectors and the actual values of x and y. Since dx and y are both cosines and dy and x both sines, their relationships are simply proportional given the velocity. Then, formula (5) guarantees that the radius is non-negative to ensure that the travel is actually counterclockwise.

5.1.2 Fuel Consumption

$$max > fc * v * T \tag{6}$$
$$fc > 0 \tag{7}$$
$$T > 0 \tag{8}$$
$$fuelinit > fc * v * T \tag{9}$$

In order to prove that the fuel would be greater than 0 at the end of a run of time T, it was necessary to use the fact that the initial level of fuel $fuelinit$ was greater than the amount of fuel that would be required for a run of time T, expressed in formula (9). To prove that in the linear model, it was necessary to ensure that the maximum fuel that the tank could hold would be enough to last for time T, formula (6). And the related constants, formulas (7) and (8), also have to be positive in order for the arithmetic to work out.

5.2 Proof Techniques

5.2.1 Discrete Ghost

Use of a discrete ghost was applied to prove both the linear and quadratic models because the ODEs do not have a real arithmetic solution (they are sines and cosines), and so a ghost was cut into the domain constraint to keep some extra information about the fuel level. This technique is essentially introducing a new variable that's not in the ODE as a value that doesn't change so that.

As seen above, to prove the second branch, that $fuel \geqslant 0$, it was important to establish a discrete ghost $fuelinit$. This variable $fuelinit$ was introduced to keep track of the value of fuel before the ODE ran, so $fuelinit$ was assigned to the value of $fuel$ right as time was set to 0 before the ODE.

$$t := 0$$
$$fuelinit := fuel$$

5.2.2 Differential Cut

The differential cut is a proof technique that cuts an invariant into the domain of a differential equation.

In terms of the linear model, this cut using $fuelinit$ was

$$fuelinit > fc * v * T \tag{10}$$
$$fuel = fuelinit - fc * v * t \tag{11}$$

Then, given formula (10), it is clear that formula (11) would prove that $fuel \geqslant 0$. This cut was possible because the control decision only allows the car to continue without refueling if $fuel > fc * v * T$, which would then guarantee $fuelinit > fc * v * T$. Then, simple arithmetic can guarantee that $fuel > 0$.

The proof for the quadratic was structured similarly, with the distinction that the invariant and cut using $fuelinit$ was the following:

$$fuelinit > fc * v * T^2 + c * T$$

5.2.3 Differential Weakening

Given the discrete ghosts and the differential cuts, the domain constraints were used to prove the post conditions by a differential weakening step. The differential weakening proof rule throws away the context, and just uses the domain constraints of an ODE to prove the postconditions. Using differential weakening is only possible if the domain constraints are strong enough to prove the postcondition without using any of the preconditions as context.

6 Discussion

While I did achieve my goals of proving the initial linear and quadratic models, I didn't make as much progress into further models as I had planned because the initial proofs were more involved that I had originally considered. Through attempting to construct valid proofs, I was able to build a more accurate initial model, and so even though I wasn't able to prove the secondary model, my initial model was more complex than I originally planned. There are many directions that this project could go in the future, and with more time, I would love to dig a little deeper into the world of NASCAR refueling strategies. Fuel consumption is a hugely important topic, and there are many factors that go into making decisions about refueling during a NASCAR race.

Some of the features I considered adding to my models are below.

One feature that none of the previously mentioned models take into account is the acceleration and deceleration of the car. All models so far have assumed that the velocity is constant, but a more accurate model would take into account the constant accelerations and decelerations involved in actual NASCAR racing. Acceleration and deceleration add complexity in calculating the amount of fuel necessary to continue because the fuel consumption is related to velocity and the above models all assume that the velocity is constant. Therefore, if acceleration and deceleration were included, the fuel consumption would need to be calculated by calculating the time it would take to cover a certain distance given an acceleration, i.e. solving the kinematic equation $d = v_intial * t + \frac{1}{2}a * t^2$.

Down the avenue of more complex models, the next model could include a decision both for when to refuel if the car is in the pit area, and another decision for whether or not to accelerate/decelerate when the car is on the side of the track that doesn't allow refueling. This model would be highly nuanced because the decision of whether or not to refuel cannot be completely independent of the decision to accelerate/decelerate because the acceleration will change the amount of fuel consumed before the next time the car is in the pit area.

A future model could also include a more clear cut condition for efficiency. I didn't have a specific efficiency post condition, but rather I built efficiency into the control so that a car wouldn't stop if it still had enough fuel to continue. For any future model that would use a time limit to ensure efficiency, the actual refueling would need to be modeled as an ODE instead of a variable assignment, to account for the total time that refueling takes.

Because my models focused solely on one car, there is no way to account for varying fuel efficiency based on race traffic or other cars. Another model in future studies could explore possibly modeling a race with more than one car, and then modify the strategy to perhaps allow refueling when it is not strictly necessary. For example, if a car has enough fuel to get around the track one more time, but it would be advantageous to pit in order to avoid on track traffic, then perhaps the optimal strategy would be for that car to pit one lap earlier.

7 Deliverables

The goal of this project was to be able to prove the previously listed models safe, and to achieve a sufficiently high level of accuracy in modeling NASCAR strategy. My models are in Appendix A and the completed proofs are in Appendix B. The model Appendix A.1 corresponds to the proof Appendix B.1, and the model Appendix A.2 corresponds to the proof Appendix B.2. There is no completed proof for the model Appendix A.3.

Appendix

A Models

A.1 Initial Linear

```
Functions.
  R rad.
  R max.
  R fc.
  R T.
End.

ProgramVariables.
  R x.
  R y.
  R dx.
  R dy.
  R v.
  R fuel.
  R fuelinit.
  R t.
End.

Problem.
  (
    x^2 + y^2 = rad^2 &
    dx^2 + dy^2 = 1 &
    dx*v = -y &
    dy*v =  x &
    v > 0 &
    rad >= 0 & max > fc * v * T & fc > 0 & T > 0 &
    fuel = max & max > 0 & fuelinit > fc * v * T
  )
  ->
  [
    {
      /* Controls */
      {
       /* Fuel will not be empty, continue */
       ?fuel > fc*v * T;
++
       /* Fuel will be empty, fill back to max */
       ?fuel <= fc*v * T; fuel := max;}

       t := 0;
       fuelinit := fuel;
```

```
      {x' = v * dx, y' = v * dy,
       dx' = -dy, dy' = dx, t' = 1, fuelinit' = 0,

       /* Fuel consumption is linear with respect to velocity */
       fuel' = -fc * v

       & v >= 0 & t <= T}

    }*@invariant( fuel >= 0 &
                  x^2 + y^2 = rad^2 &
                  dx^2 + dy^2 = 1 &
                  dx*v = -y &
                  dy*v = x &
                  v > 0 &
                  rad >= 0 & max > fc * v * T & fc > 0 & T > 0 & fuelinit > fc * v * T
                  )

  ](x^2 + y^2 = rad^2 &
    fuel >= 0) /* Safety condition. */
End.
```

A.2 Initial Quadratic

```
Functions.
  R rad.
  R max.
  R fc.
  R T.
  R c.
End.

ProgramVariables.
  R x.
  R y.
  R dx.
  R dy.
  R v.
  R fuel.
  R fuelinit.
  R t.
End.

Problem.
  (
    x^2 + y^2 = rad^2 &
    dx^2 + dy^2 = 1 &
    dx*v = -y &
    dy*v =  x &
    v > 0 &
    rad >= 0 & max > fc * v * T^2 + c * T & fc > 0 & T > 0 & c > 0 &
    fuel = max & max > 0 & fuelinit > fc * v * T^2 + c * T
  )
  ->
  [
    {
      /* Controls */
      {
       /* Fuel is not empty, continue */
       ?fuel >= fc*v * T^2 + c * T;
++
       /* Fuel is empty, fill back to max */
       ?fuel < fc*v * T^2 + c * T; fuel := max;}

       t := 0;
       fuelinit := fuel;

       {x' = v * dx, y' = v * dy,
        dx' = -dy, dy' = dx, t' = 1, fuelinit' = 0,
```

```
        /* Fuel consumption is linear with respect to velocity */
        fuel' = -(fc * v * 2 * t + c)

        & v >= 0 & t >= 0 & t <= T}

    }*@invariant( fuel >= 0 &
                  x^2 + y^2 = rad^2 &
                  dx^2 + dy^2 = 1 &
                  dx*v = -y &
                  dy*v = x &
                  v > 0 &
                  rad >= 0 & max > fc * v * T^2 + c * T  & fc > 0 & T > 0 & c > 0 &
                  fuelinit > fc * v * T^2 + c * T
                  )

  ](x^2 + y^2 = rad^2 &
    fuel >= 0) /* Safety condition. */
End.
```

A.3 Secondary Linear

```
Functions.
  R rad.
  R max.
  R fc.
End.

ProgramVariables.
  R x.
  R y.
  R dx.
  R dy.
  R v.
  R fuel.
  R fuelinit.
End.

Problem.
  (
    x^2 + y^2 = rad^2 &
    dx^2 + dy^2 = 1 &
    dx*v = -y &
    dy*v =  x &
    v > 0 &
    rad >= 0 & max > fc * v * (8*rad / v) & fc > 0 &
    fuel = max & max > 0 & fuelinit > fc * v * (8*rad / v)
  )
  ->
  [
    {
      /* Controls */
      {
       /* In pit lane, decide whether or not to stop */
       ?x >= 0;

         /* There is enough fuel to get around track again, continue */
         {?fuel >= fc*v * (8*rad / v);

     ++

         /* Fuel is empty, fill back to max */
         ?fuel < fc*v * (8*rad / v); fuel := max;}
      ++
```

```
      /* If not in pit lane, continue */
      ?x < 0;

      }

       fuelinit := fuel;

       {{x' = v * dx, y' = v * dy,
        dx' = -dy, dy' = dx,

        /* Fuel consumption is linear with respect to velocity */
        fuel' = -fc * v

        & v >= 0

        /* if the driver enters the pitting area, trigger a control decision */
        & x <= 0 }

       ++

       {x' = v * dx, y' = v * dy,
        dx' = -dy, dy' = dx,

        /* Fuel consumption is linear with respect to velocity */
        fuel' = -fc * v

        & v >= 0

        /* allow the car to move through the pitting area*/
        & x >= 0 }}


    }*@invariant( fuel >= 0 &
                  x^2 + y^2 = rad^2 &
                  dx^2 + dy^2 = 1 &
                  dx*v = -y &
                  dy*v = x &
                  v > 0 &
                  rad >= 0 & max > fc * v * (8*rad / v) & fc > 0 & fuelinit > fc * v * (
                  )

  ](x^2 + y^2 = rad^2 &
    fuel >= 0) /* Safety condition. */
End.
```

B Proofs

B.1 Initial Linear

```
implyR(1) & loop({`fuel>=0&x^2+y^2=rad^2&dx^2+dy^2=1&dx*v=-y&dy*v=x&v>0&rad>=0&max>fc*v*T&fc>0&
<(
master,
master,
composeB(1) &
choiceB(1) &
andR(1)
<(
testB(1) &
implyR(1) &
composeB(1) &
assignB(1) &
composeB(1) &
assignB(1) &
boxAnd(1) &
andR(1) <(diffInvariant({`fuelinit > fc * v * T & fuel = fuelinit - fc * v * t`}, 1) & diffWeal
          diffInvariant({`dx = -y/v & dy = x/v`}, 1) & autoDiffInd(1)
         ),
composeB(1) &
testB(1) &
implyR(1) &
assignB(1) &
composeB(1) &
assignB(1) &
composeB(1) &
assignB(1) &
boxAnd(1) &
andR(1) <(diffInvariant({`fuel = max - fc * v * t`}, 1) & diffWeaken(1) & implyR(1) & QE,
          diffInvariant({`dx = -y/v & dy = x/v`}, 1) & autoDiffInd(1)
         )
)
)
```

B.2 Initial Quadratic

```
implyR(1) & loop({`fuel>=0&x^2+y^2=rad^2&dx^2+dy^2=1&dx*v=-y&dy*v=x&v>0&rad>=0&max>fc*v*T^2+c*T
<(
master,
master,
composeB(1) &
choiceB(1) &
andR(1)
<(
testB(1) &
implyR(1) &
composeB(1) &
assignB(1) &
composeB(1) &
assignB(1) &
boxAnd(1) &
andR(1) <(diffInvariant({`fuelinit >  fc * v * T^2 + c * T & fuel = fuelinit - fc * v * t^2 - c
          diffInvariant({`dx = -y/v & dy = x/v`}, 1) & autoDiffInd(1)
         ),
composeB(1) &
testB(1) &
implyR(1) &
assignB(1) &
composeB(1) &
assignB(1) &
composeB(1) &
assignB(1) &
boxAnd(1) &
andR(1) <(diffInvariant({`fuel = max - fc * v * t^2 - c * t`}, 1) & diffWeaken(1) & implyR(1) &
          diffInvariant({`dx = -y/v & dy = x/v`}, 1) & autoDiffInd(1)
         )
))
```

References

[1] Zhen Zhu Alan Deck, Cary Deck. Decision making in a sequential game: The case of pitting in nascar. Technical report, Chapman University. http://digitalcommons.chapman.edu/cgi/viewcontent.cgi?article=1077&context=esi_pubs.

[2] Eric Baxter. How nascar drafting works. Online; accessed April 2016. http://auto.howstuffworks.com/auto-racing/nascar/nascar-basics/nascar-drafting.htm.

[3] Kenny Bruce. When fuel is all that matters. Online; accessed March 2016, July 2014. http://www.nascar.com/en_us/news-media/articles/2014/7/23/when-fuel-mileage-strategy-matters-jeff-gordon.html.

[4] Karim Chatti. Models for estimating the effects of pavement condition on vehicle operating costs, appendix a: Fuel consumption models, December 2011. http://onlinepubs.trb.org/onlinepubs/nchrp/nchrp_rpt_720Appendixes.pdf.

[5] Steve McCormick. What is nascar? Online; accessed April 2016, December 2014. http://nascar.about.com/od/nascar101/f/whatisnascar.htm.