

**15-424/15-624/15-824 Lab 2 Revision 1**  
**15-424/15-624/15-824 Foundations of Cyber-Physical Systems**  
**Course TAs: Nathan Fulton (nathanfu@cs), Anastassia Kornilova (akornilo@andrew)**

Betabot Due Date: 2/12/16, worth 20 points

Veribot Due Date: 2/18/16, worth 80 points

## 1. Event-triggered Highway Driving

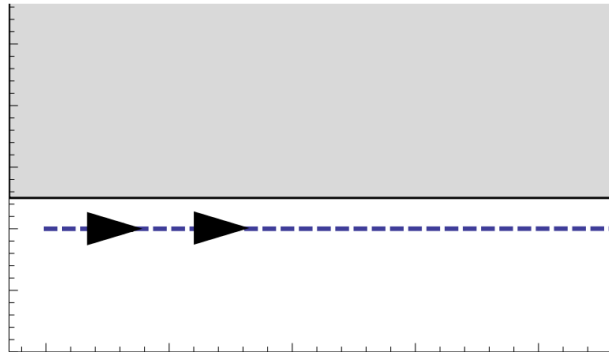


Figure 1: Lead and control car

In this problem, you will design a hybrid program (HP) to model a controlled car (`ctrl`) following a lead car (`lead`) along a straight road.

You can get templates for the problems from <http://symbolaris.com/course/fcps16/lab2.zip>.

- The lead car should keep a constant and nonnegative velocity (i.e.  $vel_{lead} \geq 0$ ).
- The driver of the controlled car can only choose to accelerate at rate ( $A > 0$ ), or brake at rate  $-B$ , where ( $B > 0$ ). The choice of acceleration  $A$  should only be available to the driver when it is safe – a condition that you will have to define.
- The controlled car has continuous access to the lead car's position and velocity (i.e. the controller you design should be *event-triggered*).
- Assume the cars are infinitesimal points. In other words, a crash occurs only if the position of the controlled car exceeds the position of the lead car (i.e. a crash occurs only if  $pos_{ctrl} > pos_{lead}$ ).
- The controller should always have a valid choice (i.e. the transition semantics of the controller should never be empty).

1.1 [Betabot] What is a good safety condition for this system? A good efficiency condition? Add this to `L2Q1.discussion.andrewid.txt`.

1.2 [Betabot] Fill in the missing parts of the HP below to model this system. Also fill in your safety condition from 1.1. Save this file as `L2Q1.andrewid.kyx`. While a proof is not required at the Betabot due date, you should, nevertheless, strive to get the model and controller correct, because that will give you a better basis for the Veribot that you will be proving.

1.3 [Veribot] Use KeYmaera X to prove that the HP you designed in 1.2 satisfies your safety condition and translate your manual proof into a tactic. Save the resulting tactic as `L2Q1.andrewid.kyt` and the corresponding `.kyx` file as `L2Q1.andrewid.kyx`.

1.4 [Veribot] **Bonus:** Drivers get uncomfortable when their car gets too close to the car ahead. Update your safety condition to require that the cars never come within constant distance  $c$  of each other. Then update your model to satisfy this requirement and prove it in KeYmaera. Only attempt the bonus problem *after* proving safety without the buffer – you are required to submit both versions to get credit.

## 2. Time-triggered Highway Driving

In this problem, you should allow the lead car to either accelerate at rate  $A$  or brake at rate  $-B$  and also change the controller from being event-driven to being time-driven. This means that when your car chooses an acceleration, it may be stuck with that choice for some time. Your model will now have a “stop watch” which must be set to 0 before each continuous evolution.

- The lead car may accelerate or brake arbitrarily at rate  $A$  or  $-B$ . The controlled car never has access to the lead car’s acceleration.
- In part 1, your car could only accelerate or brake. This means that once it comes to a stop, it has no option but to accelerate. If acceleration is not safe, then the controller has no transition. You have more freedom in your controller design for this question to address this issue.
- The controlled car has intermittent access to the lead car’s position and velocity. The time between updates is variable, but is guaranteed to be less than time  $T$  (i.e. your controller must be *time-triggered*).
- The controller should always have a valid choice (i.e. the transition semantics of the hybrid program should not be empty).

2.1 [Betabot] Using the template, design a time-triggered controller and model the system as a hybrid program. Then, write a  $d\mathcal{L}$  formula that shows your safety condition from question 1.1 is still satisfied by this controller. Submit this file as `L2Q2_andrewid.kyx`.

2.2 [Veribot] Using KeYmaera X, prove that your  $d\mathcal{L}$  formula is true and convert your proof into a tactic. Save and submit the tactic as `L2Q2_andrewid.kyt`, along with an updated version of your .kyx file `L2Q2_andrewid.kyx`.

2.3 [Veribot] **Question:** Compare and contrast the Event-triggered and Time-triggered highway driving. Describe their relationship. Which was easier to prove safe? Which would be easier to implement? Why and what caused these differences? Submit your answer to this question in `L2Q2_discussion_andrewid.txt`.

## 3. Submission Checklist

Labs can be submitted in groups of two. If that is the case, then the submissions should look like `L2Q?_andrewid1_andrewid2.kyx` and `L2Q?_andrewid1_andrewid2.kyt`. Only one of you needs to submit it on autolab.

- (a) **BetaBots:** submit a zip file on autolab containing your preliminary .kyx files for each of the tasks as well as the BetaBot discussion file. This will enable us to give you feedback halfway through the assignment, so that you don’t get stuck! If you want, you can include some *small* comments about your approach and questions you might have. *Due Fri 02/12.*  
While a proof is not required at the Betabot due date, you should, nevertheless, strive to get the model and controller correct, because that will give you a better basis for the Veribot that you will be proving.
- (b) **Final submission:** The final submission works the same way, but you must include your final model in a .kyx file as well as completed tactic in a .kyt file. To receive credit, proofs must be complete (i.e. the “Property Proved” dialog appears after copy/pasting your .kyt file contents

into the tactic input in the Web UI. Also submit your veribot discussion file and your extracted database as `keymaerax_extracted_andrewid.sqlite` *Due Thu 02/18.*

Use the provided templates, and *do not forget to fill in the section at the top*. It gives us important information when grading your submission!

The BetaBot zip file should contain:

- `L2Q1_andrewid.kyx`
- `L2Q2_andrewid.kyx`
- `L2Q1_discussion_andrewid.txt`

The VeriBot zip file should contain:

- `L2Q1_andrewid.kyx`
- `L2Q1_andrewid.kyt`
- `L2Q2_andrewid.kyx`
- `L2Q2_andrewid.kyt`
- `L2Q2_discussion_andrewid.txt`
- `keymaerax_exported_andrewid.sqlite`

Revisions:

- `positive` → `nonnegative`