Using Theorem Provers to Guarantee Closed-Loop Properties

Nikos Aréchiga Sarah Loos André Platzer Bruce Krogh

Carnegie Mellon University

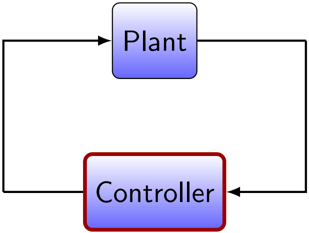
April 27, 2012

The controller design problem

Suppose there is a plant that needs to be controlled. Established control design provides

- stability
- optimality (with respect to some cost function)
- robustness (acceptable performance and stability for a range of disturbances and system parameters)

But these are not the only properties of interest.



Another property: safety

- A subset of the state space is identified as unsafe (state variable constraints)
- A safety property formally specifies that the system state will never enter the unsafe set

Traditional control design methods cannot guarantee safety properties.

One approach: iterative design and verification

One way to design a safe controller:

1. Design a controller in the usual way (for stability, robustness, optimality)
2. Try to show the closed-loop system is safe for the given controller (using, e.g., reachability analysis or a theorem prover)
3. If unsafe, re-design until safe

One approach: iterative design and verification

One way to design a safe controller:

1. Design a controller in the usual way (for stability, robustness, optimality)
2. Try to show the closed-loop system is safe for the given controller (using, e.g., reachability analysis or a theorem prover)
3. If unsafe, re-design until safe
4. **Problem:** System parameters or specifications often change—so the entire process needs to be repeated

An alternative approach

Given a plant:

- Find constraints on the controller (rather than constraints on the state variables) that will guarantee the closed-loop system is safe
- Use these *safety constraints* in the design process by either
  - checking the safety constraints for a given controller design (and redesign if necessary); or
  - incorporating the safety constraints directly in the design method as additional constraints in the design (a better approach).

Motivation: Checking or incorporating direct constraints on the controller is easier than dealing with state variable constraints

Enter theorem provers

- Use a theorem prover to find general safety constraints for the controller, rather than to check whether the closed-loop system is safe for a given controller.
- Want constraints to admit a broad class of possible controllers, so that the control design method has sufficient freedom to take take care of stability, optimality and robustness
- Requires abstraction of the plant and controller models, non-determinism

Similar to the refinement approach to design. This top-down process is what theorem provers are good at.

Overview

1. KeYmaera: a theorem prover for hybrid systems
2. Description of the proposed approach
3. Example: an intelligent cruise control system (ICC)
4. Designing a controller for the ICC
5. Conclusions and future work

KeYmaera: A theorem prover for hybrid systems

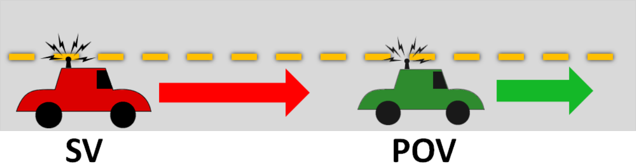
- KeYmaera is a theorem prover for differential dynamic logic (d$\mathcal{L}$)
- d$\mathcal{L}$ semantics interpret hybrid systems as transition relations over $\mathbb{R}^n$
- Quantifier elimination is used to decide first order formulas over real numbers

Proposed Approach

Rather than verify a particular controller:

- Use KeYmaera to verify that a general class of controllers is safe in closed loop
- Extract sufficient conditions for safety of the controller from the KeYmaera model (safety constraint)
- Use conventional controller design techniques to satisfy standard criteria (e.g. performance, optimality) and either
  - verify that a given controller satisfies the safety constraint, or
  - incorporate the safety constraint into the synthesis procedure

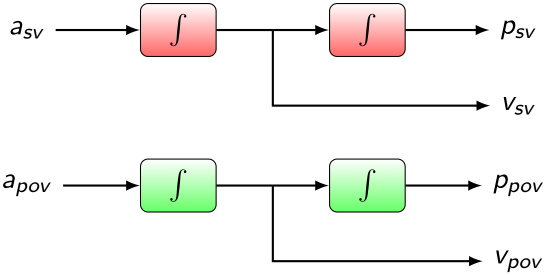
Example: Intelligent Cruise Controller



- Two cars platooning on a highway
- Objective: design a controller for the Subject Vehicle (SV)
- SV tries to maintain a constant distance from the Primary Other Vehicle (POV)
- SV can sense POV position and velocity, as well as its own
- Only use this controller within a defined operating regime

Modeling the ICC

- Each car is a double integrator



- SV controller chooses a_{sv} and a_{pov} is free

KeYmaera Model: The two car system in d$\mathcal{L}$

$$
\begin{aligned}
ICC \equiv \ \big(t := 0;\ t' = 1, & \quad (1)\\
p'_{\text{SV}} = v_{\text{SV}},\ v'_{\text{SV}} = a_{\text{SV}}, & \quad (2)\\
p'_{\text{POV}} = v_{\text{POV}},\ v'_{\text{POV}} = a_{\text{POV}}, & \quad (3)\\
(v_{\text{SV}} \geq 0 \ \wedge\ v_{\text{POV}} \geq 0 \ \wedge\ t \leq \varepsilon)\big) & \quad (4)
\end{aligned}
$$

(1) Reset the time variable
(2) Differential equations for the SV
(3) Differential equations for the POV
(4) The system is allowed to evolve for ϵ time, and then the controller samples; cars may not drive backwards

State space representation of ICC

Form:

$$\dot{x} = \mathbf{A}x + \mathbf{B}u + \mathbf{E}d$$

$$y = \mathbf{C}x + \mathbf{D}u + \mathbf{F}d$$

Model v_{pov} as an external disturbance:

$$\begin{bmatrix} \dot{\Delta p} \\ \dot{v}_{sv} \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta p \\ v_{sv} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a_{sv} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_{pov} \\ d_{set} \end{bmatrix}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \Delta p \\ v_{sv} \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} a_{sv} + \begin{bmatrix} 0 & -1 \end{bmatrix} \begin{bmatrix} v_{pov} \\ d_{set} \end{bmatrix}$$

Controller Structure: Introduce an integrator

To eliminate steady state error, introduce a new variable, $\dot{z} = y = \Delta p - d_{set}$. The variable z will represent the integral of the position error. Assume that the state of the integrator is bounded by some parameters Z_{min} and Z_{max}.

$$\begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ \mathbf{D} \end{bmatrix} u + \begin{bmatrix} \mathbf{E} \\ \mathbf{F} \end{bmatrix} d$$

State variable feedback with a setpoint

Proposed form of the control signal:

$$u = K_1(\Delta p - d_{set}) + K_2(v_{pov} - v_{sv}) + K_3 \int (\Delta p - d_{set})\, dt$$

In the implementation, the state of the integrator is bounded with a saturation function to eliminate excessive integrator windup (other methods can be used to address this).

Applying the Proposed Approach to the ICC problem

Step 1. Obtain safety constraint using KeYmaera

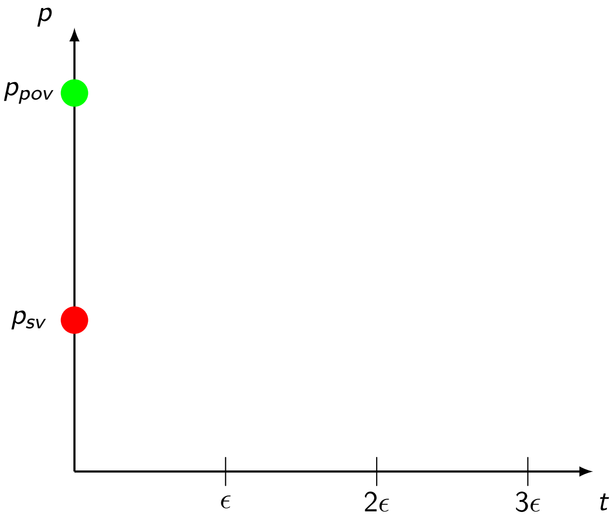
Step 2. Define the operating regime and the form for a specific controller

Step 3. Choose the controller gains using a standard procedure (LQR)

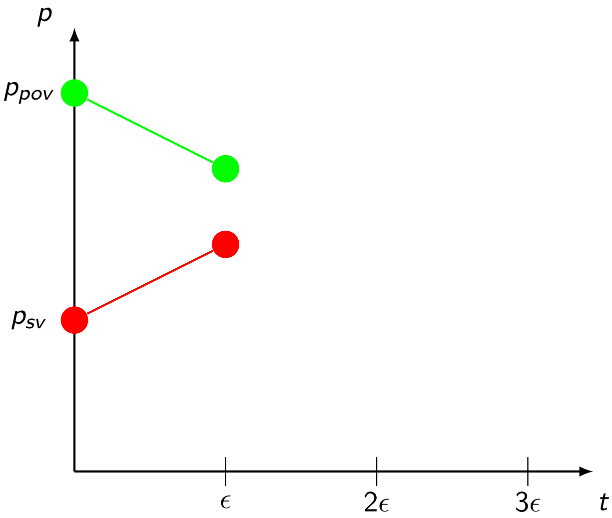
Step 4. Use the safety constraint to find a set point that will yield a safe controller

Note that the safety constraint is a static input-output relation on the controller—no dynamic or closed-loop model is required

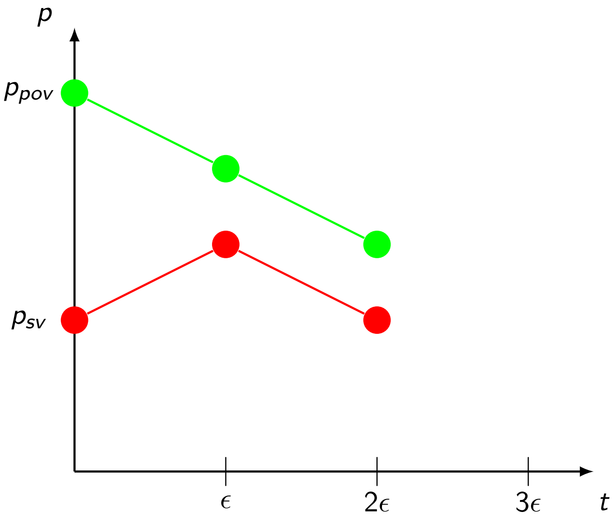
Finding a safety constraint



Finding a safety constraint



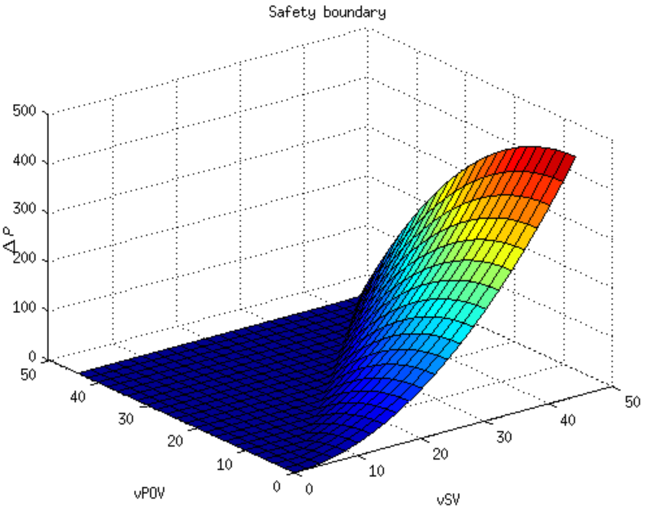
Finding a safety constraint



Model of the safety constraint in KeYmaera

$$
\begin{aligned}
ctrl \equiv\ & \mathrm{P}ov_{ctrl} \;||\; \mathrm{S}v_{ctrl}; && (5)\\
\mathrm{P}ov_{ctrl} \equiv\ & (a_{\mathrm{P}ov} := *;\ ?(-B \leq a_{\mathrm{P}ov} \leq A)) && (6)\\
\mathrm{S}v_{ctrl} \equiv\ & \big(a_{\mathrm{S}v} := *;\ ?(-B \leq a_{\mathrm{S}v} \leq -b)\big) && (7)\\
\cup\ & \big(?\mathbf{Safe}_{\varepsilon} < 0;\ a_{\mathrm{S}v} := *;\ ?(-B \leq a_{\mathrm{S}v} \leq A)\big) && (8)\\
\cup\ & \big(?(v_{\mathrm{S}v} = 0);\ a_{\mathrm{S}v} := 0\big) && (9)\\
\mathbf{Safe}_{\varepsilon} \equiv\ & p_{\mathrm{S}v} + \frac{v_{\mathrm{S}v}^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon v_{\mathrm{S}v}\right) - p_{\mathrm{P}ov} - \frac{v_{\mathrm{P}ov}^2}{2B} && (10)
\end{aligned}
$$

The safety condition divides state space



Safety constraint provides a static relation

- d$\mathcal{L}$ programs are interpreted as relations over state space,
- The safety constraint is then a safe transition relation from the state space of the plant to the space of control inputs
- Recall the control equation:

$$u = K_1(\Delta p - d_{set}) + K_2(v_{pov} - v_{sv}) + K_3 \int (\Delta p - d_{set})\, dt$$

- We want $u \leq -b$ when safety constraint requires it
- Recall the integrator state is bounded

Define the operating regime

- Physical and legal limits on possible vehicle speeds, $v_{sv} \leq 33.53m/s \approx 75mph$, $v_{pov} \leq 40.23m/s \approx 90mph$
- Minimum speed at which ICC can be turned on, $v_{sv} \geq 18m/s \approx 40mph$
- Maximum distance at which ICC operates (normal cruise control takes over for larger distances) $\Delta p \leq 50m$
- Driver behavior model: limit at which the driver will take control from the ICC, $(1/12)v_{pov} - (7/40)v_{sv} + (1/6)\Delta p \geq -1$
- Bounds on integrator state to eliminate windup, $Z_{min} = -100$, $Z_{max} = 100$

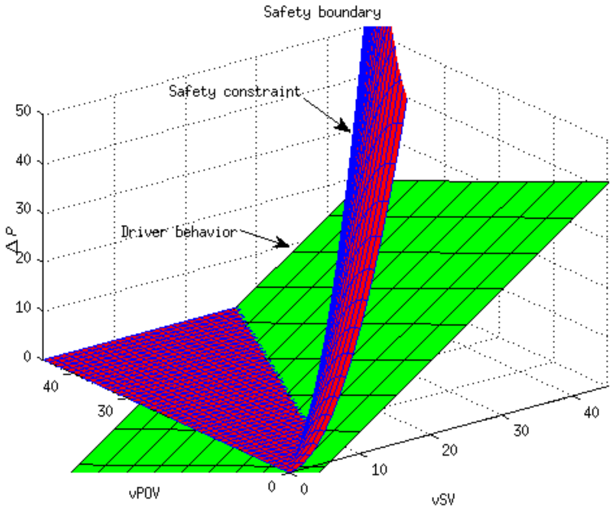
Driver behavior model

Constraint designed to model a driver's decision process. The plane was designed to go through the points:

- $v_{sv} = 20, v_{pov} = 20, \Delta p = 5$
- $v_{sv} = 30, v_{pov} = 31, \Delta p = 10$
- $v_{sv} = 40, v_{pov} = 40, \Delta p = 16$

and exclude points where the cars are closer to a collision situation

Safety condition and operating regime



Design the controller

Choose the gain matrix K through LQR, which chooses the matrix K that minimizes the cost function:

$$J = \int x^T Q x + u^T R u \, dt$$

Using the identity matrix for Q, $R = 1$ and $N = 0$, the gain matrix is:

$$K = \begin{bmatrix} 2.4142 & 2.4142 & 1 \end{bmatrix}$$

Find a set point

Desired controller behavior:

- If the system state is within the operating regime, and
- If the system state is right on the safety boundary,

Then the controller should brake.
Formally,

$$\forall x : x \in \mathcal{O}_{\mathcal{R}} \wedge \mathbf{Safe}_{\varepsilon}(x) = 0 \rightarrow ctrl(x) \leq -b$$

This is a static, first order formula. Quantifier elimination reduces it to an equivalent formula that is just a constraint on the set point.
For this controller,

$$d_{set} \geq 89.23m$$

Problem: extremely conservative

Discussion

Results are very preliminary, the controller is extremely conservative. However, we have accomplished:

- incorporating safety considerations into the design process
- the need for an iterative design-verification process is eliminated
- changes in the system parameters or the operating regime can be addressed as easily as with standard control design

Future Work

- Develop general tools for the proposed procedure
- Have more realistic application scenarios
- Compare to the traditional approach

Using Theorem Provers to Guarantee Closed-Loop Properties

Nikos Aréchiga Sarah Loos André Platzer Bruce Krogh

Carnegie Mellon University

April 27, 2012