KeYmaera and S\(\text{\textalpha}nx\) for Hybrid System Modeling and Verification

Stefan Mitsch, Sarah M. Loos, and André Platzer
Computer Science Department, Carnegie Mellon University

**CASE STUDIES**

### Aircraft Controllers

[1]: As airspace becomes ever more crowded, air traffic management must reduce both space and time between aircraft to increase throughput, making on-board collision avoidance systems ever more important. We prove the collision avoidance systems never allow aircraft to get too close to one another, even when new planes approach an in-progress avoidance maneuver.

### Robot Obstacle Avoidance

[2]: Nowadays, robots interact frequently with a dynamic environment outside limited manufacturing sites and in close proximity with humans. Thus, safety of motion and obstacle avoidance are vital safety features of such robots. We formally verify that our controller avoids both stationary and moving obstacles.

### Robot Obstacle Avoidance: Passive Safety

### Robotic Surgery

[6]: We analyzed a control algorithm that provides directional force feedback for a surgical robot. We found two bugs and described exactly what could go wrong. We then developed a new algorithm that provides safe operation along with directional force feedback. We created a machine-checked proof that guarantees the new algorithm is safe for all inputs.

### Train Control

[3]: We prove that the European Train Control System protocol remains correct even in the presence of perturbation by disturbances in the dynamics and when a PI controlled speed supervision is used.

**METHOD**

Automotive, aircraft, railway, and robotics controllers are hybrid systems, which we model and verify using differential dynamic logic (dL) [7,8]. Below are a few selected rules from the dL proof calculus.

\[
\begin{align*}
\forall x. (\phi(x) \Rightarrow \phi(x')) & \Rightarrow \forall x. \phi(x) \\
\exists x. \phi(x) & \Rightarrow \exists x. \exists y. \phi(x) \\
\forall x. \exists y. \phi(x, y) & \Rightarrow \exists y. \forall x. \phi(x, y)
\end{align*}
\]

The basic operators are (non-deterministic) assignment \((x := f(x), x := 1)\), non-deterministic choice \((a \cup B)\), and non-deterministic repetition \((a^*\)\). dL formulae are built with expressions over the reals using the usual logical connectives, quantifiers, and modality operators.

**TOOLS**


**HOW CAN I APPLY THE METHODS?**

Our formal verification methods use a hybrid system model that specifies both its discrete and continuous behavior. This tends to be easier if the original specification already models physics (e.g., differential equations for robot kinematics). In addition, the safety condition(s) that must be met by the system must be formally specified. Sometimes, these are obvious (e.g., distance to static obstacles always non-zero), in other cases the process of finding a safety condition already provides value in itself (e.g., who is to blame for a collision of two moving agents?).

**REFERENCES**