Table 1: Hybrid Programs

Program Statement	Meaning
$\alpha; \beta$	Sequentially composes β after α.
$\alpha \cup \beta$	Executes either α or β nondeterministically.
α^*	Repeats α zero or more times nondeterministically.
$x := \theta$	Evaluates term θ and assigns result to x.
$x := *$	Assigns an arbitrary real value to x.
$\{x_1' = \theta_1, ..., x_n' = \theta_n \& F\}$	Continuous evolution[1].
$?F$	Aborts if formula F is not true.

is a comparison operator $=, \neq, \geq, >, \leq, <$. The quantifiers quantify over the reals. We denote by $s \models P$ the fact that P is true in state s; e.g., we denote by $s \models [\alpha]P$ the fact that $(s,t) \in [\![\alpha]\!]$ implies $t \models P$ for all states t. When P is true in every state (i.e., valid) we simply write $\models P$.

Example 2 (Safety specification for straight-line car model)**.**

$$\underbrace{v{\geq}0 \wedge A{>}0}_{\textit{initial condition}} \rightarrow [\big(\underbrace{(a{:=}A \cup a{:=}0)}_{\textit{ctrl}}\, ; \underbrace{\{p'{=}v,\ v'{=}a\}}_{\textit{plant}}\big)^*]\underbrace{v{\geq}0}_{\textit{post cond.}}$$

This formula states that if the car begins with a non-negative velocity, then it will also have a non-negative velocity after choosing new acceleration or coasting and moving for a nondeterministic period of time.

ModelPlex: Verified Runtime Validation

Central to our approach is the ability to check, at runtime, whether or not the current state of the system can be explained by a d$\mathcal{L}$ formula. The KeYmaera X theorem prover provides a mechanism for translating a d$\mathcal{L}$ formula of the form $P \rightarrow [\alpha^*]Q$ into a formula of real arithmetic, which checks whether the present behavior of a system fits to this model. The resulting arithmetic is checked at runtime and is accompanied by a correctness proof. This algorithm, called ModelPlex (Mitsch and Platzer 2016), can be used to extract provably correct monitors that check compliance with the model as well as with the controller. If non-equivalence transformations have been used in the ModelPlex monitor synthesis proofs, the resulting monitor may be conservative, i.e. give false alarms. But if the monitor formula evaluates to true at runtime, the execution is guaranteed to be safe.

Controller Monitors ModelPlex controller monitors are boolean functions that monitor whether or not the controller portion of a hybrid systems model has been violated. The monitor takes two inputs – a "pre" state and a "post" state. The controller monitor returns true if and only if there is an execution of the $ctrl$ fragment of the program that, when executed on the "pre" state, produces the "post" state. For example, the controller monitor for Model 2 is:

$$(v_{post} = v \wedge p_{post} = p \wedge a_{post} = A) \vee$$
$$(v_{post} = v \wedge p_{post} = p \wedge a_{post} = 0)$$

[1]A continuous evolution along the differential equation system $x_i' = \theta_i$ for an arbitrary duration within the region described by formula F.

where a_{post} is the value of a chosen by the controller. Similarly, v_{post} and p_{post} are the values v and a chosen by the controller. Therefore, this controller monitor states that the controller may choose $a := A$ or $a := 0$, but may not change the values of p or v.

We write the controller monitor as a function

$$\mathit{CM} : S \times A \rightarrow \mathit{Bool}$$

mapping a current state $s \in S$ and an action $\mathit{act} \in A$ to a boolean value. This formulation is equivalent to the pre/post state formulation (e.g., $v_{post} = \mathit{act}(v_{pre})$) where

$$\mathit{act}(s)$$

is the state reached by performing the action *act* in state s.

Model Monitors ModelPlex can also produce full model monitors, which check that the *entire* system model is accurate – including the model of the system's physics – for a single control loop. The full model monitor returns true only if the controller for the system chooses a control action that is allowed by the model of the system and also the observed physics of the system correspond to the differential equations describing the system's physical dynamics. The full model monitor for Model 2 is:

$$(t_{post} \geq 0 \ \wedge \ a_{post} = A \ \wedge \ v_{post} = At_{post} + v_{pre} \ \wedge$$
$$p_{post} = \frac{At_{post}^2}{2} + v_{pre}t_{post} + p_{pre}) \vee$$
$$(t_{post}{\geq}0 \wedge v_{post}{=}v_{pre} \wedge p_{post}{=}v_{post}t_{post} + p_{pre} \wedge a_{post}{=}0)$$

Each side of the disjunction corresponds to a control decision, and the constraints on v and p come from solving the differential equation $p' = v, v' = a$.

We write the ModelPlex monitor as a function

$$\mathit{MM} : S \times A \times S \rightarrow \mathit{Bool}$$

where S is a set of states and A is a set of actions allowed by the controller; the first argument is the state before the control action, the second argument specifies the control action, and the third argument specifies the state after following the plant with the chosen control action.

The ability to perform verified runtime monitoring is essential to our approach – these arithmetic expressions are the conditions that allow us to determine when to use a speculative controller, and when to avoid deviating from the various options available in the verified nondeterministic control policy. We define model monitors semantically.