Towards a Hybrid Dynamic Logic for Hybrid Dynamic Systems

André Platzer\textsuperscript{1,2}

\textsuperscript{1}Carnegie Mellon University, Pittsburgh, PA, USA
\textsuperscript{2}University of Oldenburg, Department of Computing Science, Germany
aplatzer@cs.cmu.edu

LICS International Workshop on Hybrid Logic 2006
Towards a Hybrid Dynamic Logic for Hybrid Dynamic Systems

André Platzer\textsuperscript{1,2}

\textsuperscript{1}Carnegie Mellon University, Pittsburgh, PA, USA

\textsuperscript{2}University of Oldenburg, Department of Computing Science, Germany

aplatzer@cs.cmu.edu

LICS International Workshop on Hybrid Logic 2006
Hybrid Dynamic Logic

Logic with state-references and program-modalities

Hybrid Dynamic Systems

Hybrid dynamic systems are subject to both continuous evolution along differential equations and discrete change.
Hybrid dynamic systems are subject to both continuous evolution along differential equations and discrete change.

Example (Safety-Critical)

- Car / train / aircraft / chemical process / artificial pancreas
- discrete: digital controller of plant
- continuous: physical model of plant
Hybrid dynamic systems are subject to both continuous evolution along differential equations and discrete change.

1. Verify intricate dynamics in isolation
2. Integrability of local correctness
Hybrid Dynamic Systems

Hybrid dynamic systems are subject to both continuous evolution along differential equations and discrete change.

Challenges (Compositional Verification)

1. Verify intricate dynamics in isolation
2. Integrability of local correctness
   1. state-based reasoning: (transition to abstract state $i$)
   2. introspection: (statement about other state $\Theta_i \phi$)
Outline

1 Motivation

2 The Logic $d\mathcal{L}_h$
   - Syntax
   - Semantics
   - Compositional Introspection

3 The $d\mathcal{L}_h$ Calculus
   - Sequent Calculus
   - State-based Reasoning
   - Soundness & Co

4 Conclusions & Future Work
Outline

1. Motivation

2. The Logic $dL_h$
   - Syntax
   - Semantics
   - Compositional Introspection

3. The $dL_h$ Calculus
   - Sequent Calculus
   - State-based Reasoning
   - Soundness & Co

4. Conclusions & Future Work
The Logic $\mathcal{L}_h$: Syntax

$d\mathcal{L}_h$ formulas = first-order logic + dynamic logic + hybrid logic

$[\alpha]\phi$, $\langle \alpha \rangle \phi$

Definition (System actions $\alpha$)

$\dot{x} = f(x)$ (continuous evolution)

$x := \theta$ (discrete mode switch)

$\phi?$ (conditional execution)

$\alpha; \gamma$ (seq. composition)

$\alpha \cup \gamma$ (nondet. choice)

$\alpha^*$ (nondet. repetition)
\[ \dot{x} = -x + 0.5 \quad \dot{x} = f(x) \]

\[ x > 1 \quad \rightarrow \quad \langle \dot{x} = -x; \ x := x + 0.5; \ \dot{x} = f(x) \rangle \text{ safe} \]
\[ \dot{x} = -x \]

\[ \dot{x} = f(x) \]

\[ x > 1 \rightarrow \langle \dot{x} = -x; x := x + 0.5; \dot{x} = f(x) \rangle \text{ safe} \]
Compositional Introspection in ETCS Braking

\[
[poll\text{-}sensor; \ a := \text{accel-sys}; \ \ddot{z} = a](z \geq m \rightarrow \Box_{i}\text{slope})
\]
[poll-sensor; a := accel-sys; i?; \dot{z} = a](z \geq m \rightarrow @i slope)
Outline

1. Motivation

2. The Logic $\mathcal{L}_h$
   - Syntax
   - Semantics
   - Compositional Introspection

3. The $\mathcal{L}_h$ Calculus
   - Sequent Calculus
   - State-based Reasoning
   - Soundness & Co

4. Conclusions & Future Work
Sequent Calculus (excerpt)

(R1) \[ i \langle x := \theta \rangle j \vdash i F_x^\theta \]
\[ \vdash i \langle x := \theta \rangle j \vdash j F \]

(R2) \[ i \langle \alpha \rangle a, i \alpha \phi \vdash \]
\[ \vdash i \langle \alpha \rangle \phi \vdash \]

(R3) \[ i \exists t \geq 0 \langle x := y_x(t) \rangle \phi \vdash \]
\[ \vdash i \langle \dot{x} = f(x) \rangle \phi \vdash \]

where \( y_x \) solution of IVP
\[ \dot{x} = f(x) \]
\[ x(0) = x \]

Priority: \( R3 > R2 > R1 \)
State-based Reasoning for Compositional Verification

\[ \text{Abbreviations: } c_2 \equiv (m - z \geq 2e) \text{ and } \text{accel} \equiv (a := -b \cup (c_2?; a := 0.1)); \ddot{z} = a \]


<table>
<thead>
<tr>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall t (a := -b) r, \forall r (\dot{z} = -b) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t (a := -b) r, \forall r (\dot{z} = a) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t (\langle a := -b \rangle r \lor \langle c_2 ; a := 0.1 \rangle r), \forall r (\dot{z} = a) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t (a := -b \cup \langle c_2 ; a := 0.1 \rangle r), \forall r (\dot{z} = a) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t (\langle a := -b \rangle \cup \langle c_2 ; a := 0.1 \rangle \dot{z} = a) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t (\text{accel}) cr \vdash \forall t (\dot{z} = -b) z \geq m$</td>
</tr>
<tr>
<td>$\forall t \neg (\dot{z} = -b) z \geq m, \forall s [\text{tctl}] t, \forall t (\text{accel}) cr \vdash$</td>
</tr>
<tr>
<td>$\forall s [\text{tctl}] \neg (\dot{z} = -b) z \geq m, \forall s [\text{tctl}] t, \forall t (\text{accel}) cr \vdash$</td>
</tr>
<tr>
<td>$\forall s [\text{tctl}] \neg (\dot{z} = -b) z \geq m, \forall s [\text{tctl}] \langle \text{accel} \rangle cr \vdash$</td>
</tr>
<tr>
<td>$\forall s [\text{tctl}] \neg (\dot{z} = -b) z \geq m, \forall s [\text{tctl}; \text{accel}] cr \vdash$</td>
</tr>
<tr>
<td>$\forall s [\text{tctl}] \neg (\dot{z} = -b) z \geq m \vdash \forall s \neg (\text{tctl}; \text{accel}) cr$</td>
</tr>
</tbody>
</table>

Abbreviations: $c_2 \equiv (m - z \geq 2e)$ and $\text{accel} \equiv (a := -b \cup \langle c_2 ; a := 0.1 \rangle); \dot{z} = a$
\[\begin{align*}
\Box_t (\ddot{z} = -b) s, \Box_s \text{crash} & \vdash \Box_s z \geq m \\
\Box_t (\ddot{z} = -b) s, \Box_s \text{crash} & \vdash \Box_t (\ddot{z} = -b) z \geq m \\
\Box_t (a := -b) r, \Box_t (\ddot{z} = -b) \text{crash} & \vdash \Box_t (\ddot{z} = -b) z \geq m \\
\Box_t (a := -b) r, \Box_r (\ddot{z} = a) \text{crash} & \vdash \Box_t (\ddot{z} = -b) z \geq m
\end{align*}\]
\[\begin{align*}
\@t\langle \dot{z} = -b \rangle s, \@s crash & \vdash \@s z \geq m \\
\@t\langle \dot{z} = -b \rangle s, \@s crash & \vdash \@t\langle \dot{z} = -b \rangle z \geq m \\
\@t\langle a := -b \rangle r, \@t\langle \dot{z} = -b \rangle crash & \vdash \@t\langle \dot{z} = -b \rangle z \geq m \\
\@t\langle a := -b \rangle r, \@r\langle \dot{z} = a \rangle crash & \vdash \@t\langle \dot{z} = -b \rangle z \geq m
\end{align*}\]
\[ \begin{array}{l}
\text{State-based Reasoning for Compositional Verification} \\
\text{*} \\
\hline
\\text{\( @t(\ddot{z} = -b)s, @s\text{crash} \vdash @s z \geq m \)} \\
\hline
\\text{\( @t(\ddot{z} = -b)s, @s\text{crash} \vdash @t(\ddot{z} = -b)z \geq m \)} \\
\hline
\\text{\( @t(a := -b)r, @t(\ddot{z} = -b)\text{crash} \vdash @t(\ddot{z} = -b)z \geq m \)} \\
\hline
\\text{\( @t(a := -b)r, @r(\ddot{z} = a)\text{crash} \vdash @t(\ddot{z} = -b)z \geq m \)} \\
\end{array} \]
\[
\begin{align*}
\@t\langle \ddot{z} = -b \rangle_s, \@s crash & \vdash \@s z \geq m \\
\@t\langle \ddot{z} = -b \rangle_s, \@s crash & \vdash \@t\langle \ddot{z} = -b \rangle z \geq m \\
\@t\langle a := -b \rangle_r, \@t\langle \ddot{z} = -b \rangle crash & \vdash \@t\langle \ddot{z} = -b \rangle z \geq m \\
\@t\langle a := -b \rangle_r, \@r\langle \ddot{z} = a \rangle crash & \vdash \@t\langle \ddot{z} = -b \rangle z \geq m
\end{align*}
\]
\begin{align*}
\Theta_t(\dot{z} = -b) s, \Theta_s crash & \vdash \Theta_s z \geq m \\
\Theta_t(\dot{z} = -b) s, \Theta_s crash & \vdash \Theta_t(\dot{z} = -b) z \geq m \\
\Theta_t(a := -b) r, \Theta_t(\dot{z} = -b) crash & \vdash \Theta_t(\dot{z} = -b) z \geq m \\
\Theta_t(a := -b) r, \Theta_r(\dot{z} = a) crash & \vdash \Theta_t(\dot{z} = -b) z \geq m
\end{align*}
Theorem (Soundness)
\( d\mathcal{L}_h \) calculus is sound.

Remark (Incompleteness)
(unbounded) \( d\mathcal{L}_h \) logic is inherently incomplete.

Proposition (Reducibility)
\( d\mathcal{L}_h \) is reducible to \( d\mathcal{L} \).

Proof (Sketch): states characterised by variable assignments

\[
i \leadsto \vec{i} = \vec{x}
\]

\[
\circ_i \phi \leadsto \langle \vec{x} := \vec{i} \rangle \phi
\]
1 Motivation

2 The Logic $\mathcal{DL}_h$
   - Syntax
   - Semantics
   - Compositional Introspection

3 The $\mathcal{DL}_h$ Calculus
   - Sequent Calculus
   - State-based Reasoning
   - Soundness & Co

4 Conclusions & Future Work
Future Work

- Levels of completeness
- Parallel systems
- Verification tool
Conclusions

- Challenges (Hybrid Dynamic Systems)
  - Verify intricate dynamics in isolation
  - Integrability of local correctness
- $d\mathcal{L}_h$ is a hybrid dynamic logic extending $d\mathcal{L}$ for compositionality:
  - State-based reasoning
  - Introspection
- Calculus with goal-directed interface to mathematical problem solving
The Logic $\mathcal{dL}_h$ (Details)

- Hybrid Dynamic Logic vs. Hybrid Dynamic Systems
- Syntax
- Semantics

Appendix

- ETCS in Mathematica
- Flexible Verification Language
dynamic logic := logic with program-modalities
dynamic system := states vary along ODE
hybrid logic := logic with state-references
hybrid system := interacting discrete & continuous behaviour
### The Logic $d\mathcal{L}_h$: Syntax

#### Definition (Formulas $\phi$)

- $\neg$, $\land$, $\lor$, $\rightarrow$, $\leftrightarrow$, $\forall x$, $\exists x$, $=, \geq, \leq, +, \cdot$ (first-order part)
- $[\alpha]\phi$, $\langle \alpha \rangle \phi$ (dynamic part)
- $i$, $@_i \phi$ (hybrid part)

#### Definition (System actions $\alpha$)

- $x := \theta$ (discrete mode switch)
- $\dot{x} = \theta$ (continuous evolution)
- $\phi?$ (conditional execution)
- $\alpha; \gamma$ (seq. composition)
- $\alpha \cup \gamma$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)
The Logic $d\mathcal{L}_h$: Semantics

**Definition (Formulas $\phi$)**

\[ \text{val}_\eta(v, [\alpha] \phi) = \text{true} \iff \forall w \text{ with } (v, w) \in \rho_\eta(\alpha) \]
\[ \text{val}_\eta(v, \langle \alpha \rangle \phi) = \text{true} \iff \exists w \text{ with } (v, w) \in \rho_\eta(\alpha) \]
\[ \text{val}_\eta(v, i) = \text{true} \iff \eta(i) = v \]
\[ \text{val}_\eta(v, \Diamond_i \phi) = \text{true} \iff \text{val}_\eta(\eta(i), \phi) = \text{true} \]

**Definition (System actions $\alpha$)**

[Diagram showing the relationship between $v$, $\rho_\eta(\alpha)$, $w_1$, $w_n$, and $[\alpha] \phi$.]

\[ \text{Return} \]

André Platzer (CMU)  Hybrid Dynamic Logic  LICS - HyLo 2006  8 / 10
Definition (Formulas $\phi$)

\[\begin{align*}
\text{val}_\eta(v, [\alpha]\phi) &= \text{true} :\iff \text{val}_\eta(w, \phi) = \text{true} \quad \forall w \text{ with } (v, w) \in \rho_\eta(\alpha) \\
\text{val}_\eta(v, \langle\alpha\rangle\phi) &= \text{true} :\iff \text{val}_\eta(w, \phi) = \text{true} \quad \exists w \text{ with } (v, w) \in \rho_\eta(\alpha) \\
\text{val}_\eta(v, i) &= \text{true} :\iff \eta(i) = v \\
\text{val}_\eta(v, @i\phi) &= \text{true} :\iff \text{val}_\eta(\eta(i), \phi) = \text{true}
\end{align*}\]

Definition (System actions $\alpha$)

\[\begin{align*}
(v, w) \in \rho_\eta(x := \theta) & :\iff w = v[x \mapsto \text{val}_\eta(v, \theta)] \\
(v, w) \in \rho_\eta(\dot{x} = f(x)) & :\iff \dfrac{\text{d}}{\text{d}t} \text{val}_\eta(\cdot, x)(\zeta) = \text{val}_\eta(\zeta, f(x)) \quad \forall \zeta \in (v, w) \\
\rho_\eta(\phi?) & = \{(v, v) : \text{val}_\eta(v, \phi) = \text{true}\} \\
\rho_\eta(\alpha; \gamma) & = \rho_\eta(\alpha) \circ \rho_\eta(\gamma) \\
\rho_\eta(\alpha \cup \gamma) & = \rho_\eta(\alpha) \cup \rho_\eta(\gamma) \\
(v, w) \in \rho_\eta(\alpha^*) & :\iff \exists v \quad v \xrightarrow{s_1} \rho_\eta(\alpha) \xrightarrow{s_2} \cdots \xrightarrow{s_n} \rho_\eta(\alpha) w
\end{align*}\]
### Definition (Formulas \( \phi \))

\[
\begin{align*}
\text{val}_\eta(v, [\alpha]\phi) &= \text{true} \iff \text{val}_\eta(w, \phi) = \text{true} \quad \forall w \text{ with } (v, w) \in \rho_\eta(\alpha) \\
\text{val}_\eta(v, \langle\alpha\rangle\phi) &= \text{true} \iff \text{val}_\eta(w, \phi) = \text{true} \quad \exists w \text{ with } (v, w) \in \rho_\eta(\alpha) \\
\text{val}_\eta(v, i) &= \text{true} \iff \eta(i) = v \\
\text{val}_\eta(v, @i\phi) &= \text{true} \iff \text{val}_\eta(\eta(i), \phi) = \text{true}
\end{align*}
\]

### Definition (System actions \( \alpha \))

\[
\begin{align*}
(v, w) \in \rho_\eta(x := \theta) & \iff w = v[x \mapsto \text{val}_\eta(v, \theta)] \\
(v, w) \in \rho_\eta(\dot{x} = f(x)) & \iff \frac{\text{d}}{\text{d}t}\text{val}_\eta(\cdot, x)(\zeta) = \text{val}_\eta(\zeta, f(x)) \quad \forall \zeta \in (v, w) \\
\rho_\eta(\phi?) & = \{(v, v) : \text{val}_\eta(v, \phi) = \text{true}\} \\
\rho_\eta(\alpha; \gamma) & = \rho_\eta(\alpha) \circ \rho_\eta(\gamma) \\
\rho_\eta(\alpha \cup \gamma) & = \rho_\eta(\alpha) \cup \rho_\eta(\gamma) \\
(v, w) \in \rho_\eta(\alpha^*) & \iff \exists \rho_\eta(\alpha) \xrightarrow{S_1} \rho_\eta(\alpha) \xrightarrow{\text{..}} \rho_\eta(\alpha) \xrightarrow{w}
\end{align*}
\]
The Logic $\mathcal{L}_h$: Semantics

**Definition (System actions $\alpha$)**

$$(v, w) \in \rho_\eta(\dot{x} = f(x)) \iff \frac{d}{dt} \text{val}_\eta(t, x)(\zeta) = \text{val}_\eta(\zeta, f(x)) \quad \forall \zeta \in (v, w)$$

$$\iff \exists f : [v(\tau), w(\tau)] \to \text{Int}$$

- $\gamma_x(\zeta) := \text{val}_\eta(f(\zeta), x)$ continuous on $[v(\tau), w(\tau)]$
- $\dot{\gamma}_x(\zeta) = \gamma_{f(x)}(\zeta), \forall \zeta \in (v(\tau), w(\tau))$
- $\gamma_y$ constant $\forall y \neq x$ and $f(v(\tau)) = v, f(w(\tau)) = w$
Definition (System actions $\alpha$)

$$(v, w) \in \rho_\eta(\dot{x} = f(x)) \iff \left(\frac{d}{dt} \operatorname{val}_\eta(t, x)(\zeta) = \operatorname{val}_\eta(\zeta, f(x)) \quad \forall \zeta \in (v, w)\right)$$

$$\iff \exists f : [v(\tau), w(\tau)] \to \operatorname{Int}$$

- $\gamma_x(\zeta) := \operatorname{val}_\eta(f(\zeta), x)$ continuous on $[v(\tau), w(\tau)]$
- $\dot{\gamma}_x(\zeta) = \gamma_{f(x)}(\zeta), \forall \zeta \in (v(\tau), w(\tau))$
- $\gamma_y$ constant $\forall y \neq x$ and $f(v(\tau)) = v, f(w(\tau)) = w$
The Logic $\mathcal{L}_h$ (Details)
- Hybrid Dynamic Logic vs. Hybrid Dynamic Systems
- Syntax
- Semantics

Appendix
- ETCS in Mathematica
- Flexible Verification Language
antecedent \Rightarrow \langle IVP \rangle \text{query}

antecedent = (z|m|b) \in \text{Reals} \land 0 < z0 < m \land b > 0 \land v0 > 0;

ODE = z''[t] == -b;

IVP = \{ODE, z[0] == z0, z'[0] == v0\};

dsol = \text{Simplify[DSolve[IVP, z[t], t]]}

query = z[t] == m;

\{\{z[t] \rightarrow -\frac{bt^2}{2} + tv0 + z0\}\}

(query/.dsol)[[1]]

Reduce[Assuming[antecedent, \exists t, t \geq 0 \&\& t \in \text{Reals}, Assuming[antecedent, %]], t, \text{Reals}]

Simplify[%, antecedent]

\(-\frac{bt^2}{2} + tv0 + z0 == m\)

\((m < z0 \&\& ( (v0 < 0 \&\& b \geq \frac{v0^2}{2m-2z0}) \lor (v0 \geq 0 \&\& b > 0)))\)\lor

\((m \geq z0 \&\& ( (v0 \leq 0 \&\& b < 0) \lor (v0 > 0 \&\& b \leq \frac{v0^2}{2m-2z0})))\)

\(2b(m-z0) \leq v0^2\)
Example (Verification Tasks)

1. System verification problem (flat / compositional)
   \[ b \geq 10 \rightarrow [\alpha]z \leq m \]

2. (Compositional) refinement
   \[ [S](C)_{safe} \]

3. Abstraction
   \[ f < \epsilon \rightarrow ([\tilde{\alpha}]\phi \rightarrow [\alpha]\phi) \]

4. Level of detail or “layered” time models
   \[ [x := 4]\phi \rightarrow [\dot{t} = 1; x := 4](t \leq 5 \rightarrow \phi) \]