

Logic and Hybrid Systems

Manasvi Saxena

Formal Systems Lab, UIUC

Hybrid Systems

- ▶ Dynamical Systems exhibiting both discrete (jump) and continuous (flow) behaviors.
- ▶ Physical systems - thermostats, airplanes and trains.
- ▶ Continuous dynamics generally specified using Differential Equations.

- ▶ Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).

Differential Dynamic Logic (dL)

- ▶ Main focus - Differential Dynamic Logic for Hybrid Systems (Andre Platzer).
- ▶ Practical deductive verification of hybrid systems.
- ▶ Introduces Hybrid Program - program notation for hybrid systems.
- ▶ Dynamic Logic for Hybrid Programs, a generalization of Dynamic Logic.

Hybrid Automata

- ▶ Commonly used to model Hybrid Systems, via Graphs.
- ▶ Nodes specify continuous dynamics. Edges describe discrete transitions.
- ▶ Intuitive, but not suitable for deductive verification.

Hybrid Automata

- ▶ Commonly used to model Hybrid Systems, via Graphs.
- ▶ Nodes specify continuous dynamics. Edges describe discrete transitions.
- ▶ Intuitive, but not suitable for deductive verification.

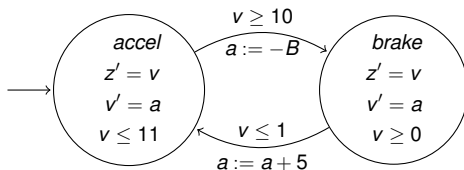


Figure: Hybrid Automata (simplified) of a Train Control System

Differential Dynamic Logic

Motivations

- ▶ **First Order Logic** - No builtin means for referring to state transitions.
- ▶ **Temporal Logics** - Modal operators allow referring to state transitions. But valid formulas only express generic facts.

Dynamic Logic (DL) - Combines operational system models with operators for reasoning.

- ▶ Provides parameterized modal operators, $[\alpha]$, $\langle\alpha\rangle$ that refer to states reachable by system α .
- ▶ $[\alpha]\phi$ expresses all states reachable by α satisfy ϕ , allowing reasoning about discrete systems.
- ▶ Say $(b > 0) \rightarrow [a := -b](a < 0)$ expresses a discrete transition. We can prove $(b > 0) \vdash (a < 0)[b/a]$ using DL's calculus.
- ▶ No built in notion for describing or reasoning about continuous dynamics.

Differential Dynamic Logic

Motivations

- ▶ Generalize DL so operational models α can be used in modal formulas like $[\alpha]\phi$. dL refers to generalized models as “Hybrid Programs”.

Differential Dynamic Logic

Syntax and Semantics

dL formulas built over

- ▶ V , set of real-valued logical variables.
- ▶ Signature $\Sigma = \Sigma_{\text{rigid}} \cup \Sigma_{\text{fl}}$
- ▶ Set $\text{Trm}(\Sigma, V)$ of *terms* - classical FOL polynomial expressions over V with additional skolem terms $s(X_1, \dots, X_n)$, where $X_1, \dots, X_n \in V$.

Differential Dynamic Logic

Syntax and Semantics

Hybrid Programs

Consider $x_i \in \Sigma$, $\theta_i, \vartheta_i \in \text{Trm}(\Sigma, V)$ for $1 \leq i \leq n$, χ a (Σ, V) FOL-formula, $\alpha, \beta \in \text{HP}(\Sigma, V)$. Set $\text{HP}(\Sigma, V)$, is defined inductively as -

- ▶ $(x_1 := \theta_1, \dots, x_n := \theta_n) \in \text{HP}(\Sigma, V)$
- ▶ $(x'_1 = \vartheta_1, \dots, x'_n = \vartheta_n) \& \chi \in \text{HP}(\Sigma, V)$. $x'_i = \vartheta_i$ is a differential equation where x'_i is the first order time derivative of x_i .
- ▶ $(?\chi) \in \text{HP}(\Sigma, V)$.
- ▶ $\alpha \cup \beta \in \text{HP}(\Sigma, V)$.
- ▶ $\alpha; \beta \in \text{HP}(\Sigma, V)$.
- ▶ $\alpha^* \in \text{HP}(\Sigma, V)$.

Differential Dynamic Logic

Hybrid Program Example

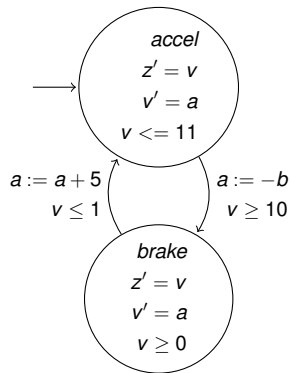
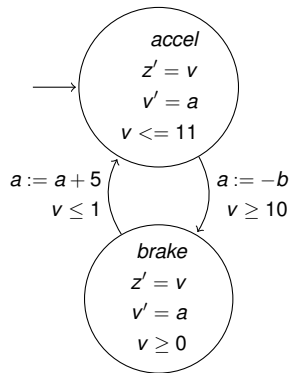


Figure: Hybrid Automata of Simple Train Control System

Differential Dynamic Logic

Hybrid Program Example



$q := accel;$
 $((?q = accel; z' = v, v' = a \& v \leq 11)$
 $\cup (?q = accel \wedge v \geq 10; a := -b; q := brake; ?v \geq 0)$
 $\cup (?q = brake; z' = v, v' = a \& v \geq 0)$
 $\cup (?q = brake \wedge v \leq 1; a := a + 5; q := accel))^*$

Figure: Hybrid Automata of Simple Train Control System

Differential Dynamic Logic

Syntax and Semantics

dL Formulas

The set $\text{Fml}(\Sigma, V)$ of dL formulas is defined as -

- ▶ If p is a n -ary predicate symbol in Σ and $\theta_1, \dots, \theta_n \in \text{Trm}(\Sigma, V)$, then $p(\theta_1, \dots, \theta_n) \in \text{Fml}(\Sigma, V)$
- ▶ If $\phi, \psi \in \text{Fml}(\Sigma, V)$ then $\phi \vee \psi, \phi \rightarrow \psi, \phi \wedge \psi \in \text{Fml}(\Sigma, V)$
- ▶ If $\phi \in \text{Fml}(\Sigma, V)$ and $x \in V$ then $\exists x.\phi, \forall x.\phi \in \text{Fml}(\Sigma, V)$
- ▶ If $\phi \in \text{Fml}(\Sigma, V)$ and $\alpha \in \text{HP}(\Sigma, V)$ then $[\alpha]\phi, \langle \alpha \rangle \phi \in \text{Fml}(\Sigma, V)$

Differential Dynamic Logic

Syntax and Semantics

Some Notation

- ▶ Interpretation $I : \Sigma_t \text{extrigid} \rightarrow \mathbb{R}$
- ▶ A state is a map $\nu : \Sigma_{fl} \rightarrow \mathbb{R}$.
- ▶ Assignment of logical variables is a map $\eta : V \rightarrow \mathbb{R}$.
- ▶ The models are Kripke Structures, where nodes are hybrid system states.

Differential Dynamic Logic

Syntax and Semantics

dL Valuation of dL Terms

Say $val_{l,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation l , assignment η and state ν .

Differential Dynamic Logic

Syntax and Semantics

dL Valuation of dL Terms

Say $val_{l,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation l , assignment η and state ν .

- ▶ (State Variables) $val_{l,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$

Differential Dynamic Logic

Syntax and Semantics

dL Valuation of dL Terms

Say $val_{l,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation l , assignment η and state ν .

- ▶ (State Variables) $val_{l,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$
- ▶ (Logical Variables) $val_{l,\eta}(\nu, x) = \eta(x), x \in V$

Differential Dynamic Logic

Syntax and Semantics

dL Valuation of dL Terms

Say $val_{l,\eta}(\nu, \cdot)$ is evaluation w.r.t. interpretation l , assignment η and state ν .

- ▶ (State Variables) $val_{l,\eta}(\nu, a) = \nu(a), x \in \Sigma_{fl}$
- ▶ (Logical Variables) $val_{l,\eta}(\nu, x) = \eta(x), x \in V$
- ▶ (Rigid Symbols)
 $val_{l,\eta}(\nu, f(\theta_1, \dots, \theta_n)) = f_l(val_{l,\eta}(\nu, \theta_1), \dots, val_{l,\eta}(\nu, \theta_n))$
where f is n-ary rigid symbol in Σ

Differential Dynamic Logic

Syntax and Semantics

Valuation of dL Formulas

- ▶ $val_{l,\eta}(\nu, p(\theta_1, \dots, \theta_n)) = p_l(val_{l,\eta}(\nu, \theta_1), \dots, val_{l,\eta}(\nu, \theta_n))$
- ▶ $val_{l,\eta}(\nu, \varphi \wedge \psi) = \top$ iff $val_{l,\eta}(\nu, \varphi) = \top \wedge val_{l,\eta}(\nu, \psi) = \top$.
Similarly for \rightarrow, \neg, \vee
- ▶ $val_{l,\eta}(\nu, \exists x. \varphi) = \top$ iff $val_{l,\eta[x \mapsto d]}(\nu, \varphi) = \top$ for some $d \in \mathbb{R}$

Differential Dynamic Logic

Syntax and Semantics

Valuation of dL Formulas

- ▶ $val_{l,\eta}(\nu, p(\theta_1, \dots, \theta_n)) = p_l(val_{l,\eta}(\nu, \theta_1), \dots, val_{l,\eta}(\nu, \theta_n))$
- ▶ $val_{l,\eta}(\nu, \varphi \wedge \psi) = \top$ iff $val_{l,\eta}(\nu, \varphi) = \top \wedge val_{l,\eta}(\nu, \psi) = \top$.
Similarly for \rightarrow, \neg, \vee
- ▶ $val_{l,\eta}(\nu, \exists x. \varphi) = \top$ iff $val_{l,\eta[x \mapsto d]}(\nu, \varphi) = \top$ for some $d \in \mathbb{R}$
- ▶ $val_{l,\eta}(\nu, [\alpha]\varphi) = \top$ iff $val_{l,\eta}(\omega, \varphi) = \top$ for all states ω with $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$.
- ▶ $val_{l,\eta}(\nu, \langle \alpha \rangle \varphi) = \top$ iff $val_{l,\eta}(\omega, \varphi) = \top$ for some state ω with $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$.

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

Evaluation $\rho_{l,\eta}(\alpha)$ of HP α . $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$ means state ω is reachable from ν by operations of α .

- ▶ $(\nu, \omega) \in \rho_{l,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff
 $\nu[x_1 \mapsto \text{val}_{l,\eta}(\nu, \theta_1), \dots, x_n \mapsto \text{val}_{l,\eta}(\nu, \theta_n)] = \omega$ and
 $\forall y \in (\Sigma_{fl} - \{x_1, \dots, x_n\}). \text{val}_{l,\eta}(\nu, y) = \text{val}_{l,\eta}(\omega, y).$

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

Evaluation $\rho_{l,\eta}(\alpha)$ of HP α . $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$ means state ω is reachable from ν by operations of α .

- ▶ $(\nu, \omega) \in \rho_{l,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff
 $\nu[x_1 \mapsto \text{val}_{l,\eta}(\nu, \theta_1), \dots, x_n \mapsto \text{val}_{l,\eta}(\nu, \theta_n)] = \omega$ and
 $\forall y \in (\Sigma_{\#} - \{x_1, \dots, x_n\}). \text{val}_{l,\eta}(\nu, y) = \text{val}_{l,\eta}(\omega, y).$
- ▶ $\rho_{l,\eta}(\alpha \cup \beta) = \rho_{l,\eta}(\alpha) \cup \rho_{l,\eta}(\beta)$

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

Evaluation $\rho_{l,\eta}(\alpha)$ of HP α . $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$ means state ω is reachable from ν by operations of α .

- ▶ $(\nu, \omega) \in \rho_{l,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff
 $\nu[x_1 \mapsto \text{val}_{l,\eta}(\nu, \theta_1), \dots, x_n \mapsto \text{val}_{l,\eta}(\nu, \theta_n)] = \omega$ and
 $\forall y \in (\Sigma_{fl} - \{x_1, \dots, x_n\}). \text{val}_{l,\eta}(\nu, y) = \text{val}_{l,\eta}(\omega, y).$
- ▶ $\rho_{l,\eta}(\alpha \cup \beta) = \rho_{l,\eta}(\alpha) \cup \rho_{l,\eta}(\beta)$
- ▶ $\rho_{l,\eta}(?\chi) = \{(\nu, \nu) : \text{val}_{l,\eta}(\nu, \chi) = \top\}$

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

Evaluation $\rho_{l,\eta}(\alpha)$ of HP α . $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$ means state ω is reachable from ν by operations of α .

- ▶ $(\nu, \omega) \in \rho_{l,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff
 $\nu[x_1 \mapsto \text{val}_{l,\eta}(\nu, \theta_1), \dots, x_n \mapsto \text{val}_{l,\eta}(\nu, \theta_n)] = \omega$ and
 $\forall y \in (\Sigma_{\#} - \{x_1, \dots, x_n\}). \text{val}_{l,\eta}(\nu, y) = \text{val}_{l,\eta}(\omega, y).$
- ▶ $\rho_{l,\eta}(\alpha \cup \beta) = \rho_{l,\eta}(\alpha) \cup \rho_{l,\eta}(\beta)$
- ▶ $\rho_{l,\eta}(\text{?}\chi) = \{(\nu, \nu) : \text{val}_{l,\eta}(\nu, \chi) = \top\}$
- ▶ $\rho_{l,\eta}(\alpha; \beta) = \{(\nu, \omega) : (\nu, Z) \in \rho_{l,\eta}(\alpha) \wedge (Z, \omega) \in \rho_{l,\eta}(\beta) \text{ for some state } Z\}$

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

Evaluation $\rho_{l,\eta}(\alpha)$ of HP α . $(\nu, \omega) \in \rho_{l,\eta}(\alpha)$ means state ω is reachable from ν by operations of α .

- ▶ $(\nu, \omega) \in \rho_{l,\eta}(x_1 := \theta_1, \dots, x_n := \theta_n)$ iff $\nu[x_1 \mapsto \text{val}_{l,\eta}(\nu, \theta_1), \dots, x_n \mapsto \text{val}_{l,\eta}(\nu, \theta_n)] = \omega$ and $\forall y \in (\Sigma_{fl} - \{x_1, \dots, x_n\}). \text{val}_{l,\eta}(\nu, y) = \text{val}_{l,\eta}(\omega, y)$.
- ▶ $\rho_{l,\eta}(\alpha \cup \beta) = \rho_{l,\eta}(\alpha) \cup \rho_{l,\eta}(\beta)$
- ▶ $\rho_{l,\eta}(\text{?}\chi) = \{(\nu, \nu) : \text{val}_{l,\eta}(\nu, \chi) = \top\}$
- ▶ $\rho_{l,\eta}(\alpha; \beta) = \{(\nu, \omega) : (\nu, Z) \in \rho_{l,\eta}(\alpha) \wedge (Z, \omega) \in \rho_{l,\eta}(\beta) \text{ for some state } Z\}$
- ▶ $(\nu, \omega) \in \rho_{l,\eta}(\alpha^*)$ iff for $n \in \mathbb{N}$, there are states $\nu = \nu_0, \dots, \nu_n = \omega$ s.t. $(\nu_i, \nu_{i+1}) \in \rho_{l,\eta}(\alpha)$ for $0 \leq i < n$.

Differential Dynamic Logic

Syntax and Semantics

Transition Semantics of Hybrid Programs

$(\nu, \omega) \in \rho_{l,\eta}((x'_1 = \vartheta_1 \dots x'_n = \vartheta_n) \& \chi)$ iff there is a flow of some duration $r \geq 0$, from ν to ω respecting the differential equations and evolution domain. Formally, there is a function

$f : [0, r] \rightarrow \text{Sta}(\Sigma)$ such that -

- ▶ $f(0) = \nu$ and $f(r) = \omega$.
- ▶ $\forall \delta : [0, r]. \text{val}_{l,\eta}(f(\delta), x_i)$ is continuous and $\text{val}_{l,\eta}(f(\delta), \chi) = \top$.
- ▶ $\forall \epsilon : (0, r). \text{val}_{l,\eta}(f(\epsilon), \dot{\vartheta}_i) = \dot{f}(\epsilon)$.
- ▶ For any $z \notin \{x_1, x_2, \dots, x_n\}$, $\text{val}_{l,\eta}(f(\zeta), z)$ remains constant for $\zeta \in [0, r]$. In other words, all other state variables remain unchanged.

Differential Dynamic Logic

Syntax and Semantics

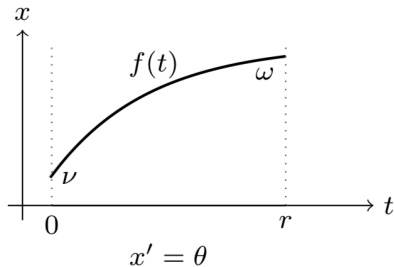


Figure: Unbounded Evolution

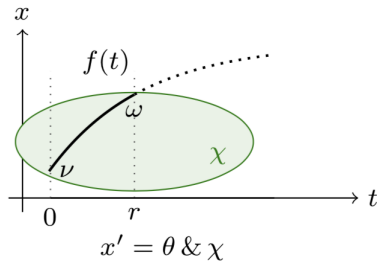


Figure: Evolution bound by χ

Differential Dynamic Logic

Examples

Some Intuition behind dL

- ▶ Generally, want to reason about Embedded Systems (ES) or Cyber Physical Systems (CPS).

Differential Dynamic Logic

Examples

Some Intuition behind dL

- ▶ Generally, want to reason about Embedded Systems (ES) or Cyber Physical Systems (CPS).
- ▶ Loosely speaking, an ES or CPS is built upon the integration of Computation (Controller) and Physical Systems (Plant). Examples include autopilot, ABS (Anti Lock Braking Systems), fly-by-wire (human interaction).
- ▶ Expressed in dL as an HP $\alpha \equiv (ctrl; plant)^*$.

Differential Dynamic Logic

Examples

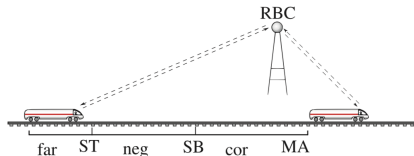
Some Intuition behind dL

- ▶ Generally, want to reason about Embedded Systems (ES) or Cyber Physical Systems (CPS).
- ▶ Loosely speaking, an ES or CPS is built upon the integration of Computation (Controller) and Physical Systems (Plant). Examples include autopilot, ABS (Anti Lock Braking Systems), fly-by-wire (human interaction).
- ▶ Expressed in dL as an HP $\alpha \equiv (ctrl; plant)^*$.
- ▶ Formulas of interest are of the form $\varphi \rightarrow [\alpha]\psi$.
- ▶ Formula above says that all states satisfying φ will always transition to states satisfying ψ , where α dictates the transition relation.

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

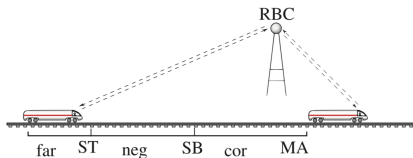


- ▶ Discards traditional fixed segments of track with mutual exclusion. Instead uses decentralized Radio Block Controllers (RBCs)
- ▶ Trains dynamically issues Movement Authorization (MAs).
- ▶ At the end of MA, agent requests MA extension (negotiation). If request is denied, train breaks before exiting old MA zone.

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)



- ▶ Train controller is responsible for staying in MA.
- ▶ Controller has to determine point SB (Start Braking). Before SB, train can move freely (maximizing throughput).
- ▶ Beyond SB (correction phase) start braking such that train remains in MA if RBC refuses extension.

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

Say we have a train -

- ▶ with MA granted upto some track position m
- ▶ at position z , heading with initial speed v towards m
- ▶ point SB as safety distance s relative to m , i.e. $m - s = SB$

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

Say we have a train -

- ▶ with MA granted upto some track position m
- ▶ at position z , heading with initial speed v towards m
- ▶ point SB as safety distance s relative to m , i.e. $m - s = SB$

The following formula expresses that the train remains in its MA under precondition ψ .

$$\psi \rightarrow [(ctrl; drive)^*] z \leq m$$

$$\text{where } ctrl \equiv (?m - z \leq s; a := -b) \cup (?m - z \geq s; a := A)$$

$$drive \equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \ \& \ v \geq 0 \wedge \tau \leq \varepsilon) .$$

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

Say we have a train -

- ▶ with MA granted upto some track position m
- ▶ at position z , heading with initial speed v towards m
- ▶ point SB as safety distance s relative to m , i.e. $m - s = SB$

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

Say we have a train -

- ▶ with MA granted upto some track position m
- ▶ at position z , heading with initial speed v towards m
- ▶ point SB as safety distance s relative to m , i.e. $m - s = SB$

The following formula expresses that the train remains in its MA under precondition ψ .

$$\psi \rightarrow [(ctrl; drive)^*] z \leq m$$

$$\text{where } ctrl \equiv (?m - z \leq s; a := -b) \cup (?m - z \geq s; a := A)$$

$$drive \equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \ \& \ v \geq 0 \wedge \tau \leq \varepsilon) .$$

Differential Dynamic Logic

Motivating Example

European Train Control System (ETCS)

$$\psi \rightarrow [(ctrl; drive)^*] z \leq m$$

where $ctrl \equiv (?m - z \leq s; a := -b) \cup (?m - z \geq s; a := A)$

$drive \equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \ \& \ v \geq 0 \wedge \tau \leq \varepsilon) \ .$

ECTS Safety Verification

In order to verify safety, we need

1. Use the calculus to analyze conditions of safety violation.
Will provide some intuition about the safety precondition.
2. Use the calculus to verify safety, allowing exploration of the calculus' rules.

Differential Dynamic Logic

Proof Calculus

Preliminaries

- ▶ $\phi_{x_1}^{\theta_1}, \dots, \phi_{x_n}^{\theta_n}$ denotes simultaneous substitution of x_i for θ_i in ϕ .
- ▶ Substitution must be “admissible”. Given term t and substitution σ , variables of t or $\sigma(t)$ must not occur in the scope of a quantifier or modality binding. α conversion for renaming is assumed and used when needed.
- ▶ $\forall^\alpha \phi$ denotes the universal closure of ϕ w.r.t. all state variables bound by α . Note quantification over state variables is definable via auxiliary logical variables as $\forall X[x := X]\phi$
- ▶ The calculus consists of propositional (P), first order (F), global (G), dynamic modality (D) rules.
- ▶ The calculus has 32 rules.

Discrete Dynamics

$$\frac{\phi_{x_1}^{\theta_1} \cdots \phi_{x_n}^{\theta_n}}{\langle x_1 := \theta_1, \dots, x_n := \theta_n \rangle \phi}$$

Continuous Dynamics

$$\frac{\forall t \geq 0 \left((\forall 0 \leq \tilde{t} \leq t \langle \mathcal{S}_{\tilde{t}} \rangle \chi) \rightarrow \langle \mathcal{S}_t \rangle \phi \right)}{[x'_1 = \theta_1, \dots, x'_n = \theta_n \ \& \ \chi] \phi}$$

- ▶ t and \tilde{t} are fresh logical variables. $\langle \mathcal{S}_t \rangle$ is the jump set $\langle x_1 := y_1(t), \dots, x_n := y_n(t) \rangle$, with simultaneous solutions y_1, \dots, y_n of the respective differential equations with constant symbols x_i as symbolic initial values.

F Rules

$$\frac{\vdash \phi(X)}{\vdash \exists x \phi(x)}$$

X is a new logical variable.

$$\frac{\vdash \phi(s(X_1, \dots, X_n))}{\vdash \forall x \phi(x)}$$

X_1, \dots, X_n are all free logical variables of $\forall x \phi(x)$

F rules are inspired from Tableau methods for FOL,
introducing decision procedures into the proof system itself.

F Rules

$$\frac{\vdash \text{QE}(\forall X (\Phi(X) \vdash \Psi(X)))}{\Phi(s(X_1, \dots, X_n)) \vdash \Psi(s(X_1, \dots, X_n))}$$

X is a new logical variable

$$\frac{\vdash \text{QE}(\exists X \bigwedge_i (\Phi_i \vdash \Psi_i))}{\Phi_1 \vdash \Psi_1 \quad \dots \quad \Phi_n \vdash \Psi_n}$$

Logical variable X only occurs in $\Phi_i \vdash \Psi_i$. The intuition is variable X was introduced via skolemization (at some point in the proof tree). The rule reintroduces existentiality over X , allowing for Quantifier Elimination (QE) over the existentially quantified terms.

Differential Dynamic Logic

Proof Calculus

G Rules

$$\frac{\vdash \forall^{\alpha}(\phi \rightarrow [\alpha]\phi)}{\phi \vdash [\alpha^*]\phi}$$

$$\frac{\vdash \forall^{\alpha}(\phi \rightarrow \psi)}{[\alpha]\phi \vdash [\alpha]\psi}$$

Differential Dynamic Logic

Proof Calculus Application

European Train Control System (ETCS)

$$\psi \rightarrow [(ctrl; drive)^*] z \leq m$$

$$\text{where } ctrl \equiv (?m - z \leq s; a := -b) \cup (?m - z \geq s; a := A)$$

$$drive \equiv \tau := 0; (z' = v, v' = a, \tau' = 1 \& v \geq 0 \wedge \tau \leq \varepsilon) .$$

Deduction modulo for MA violation in braking mode

$$\frac{\frac{\frac{v \geq 0, z < m \vdash v^2 > 2b(m - z)}{\vdash v \geq 0 \wedge z < m \rightarrow v^2 > 2b(m - z)}}{v \geq 0, z < m \vdash T \geq 0} \quad \text{D9} \frac{v \geq 0, z < m \vdash -\frac{b}{2}T^2 + vT + z > m}{v \geq 0, z < m \vdash \langle z := -\frac{b}{2}T^2 + vT + z \rangle z > m}}{\frac{v \geq 0, z < m \vdash T \geq 0 \wedge \langle z := -\frac{b}{2}T^2 + vT + z \rangle z > m}{v \geq 0, z < m \vdash \exists t \geq 0 \langle z := -\frac{b}{2}t^2 + vt + z \rangle z > m}}{\frac{v \geq 0, z < m \vdash \langle z' = v, v' = -b \rangle z > m}{\vdash v \geq 0 \wedge z < m \rightarrow \langle z' = v, v' = -b \rangle z > m}}$$

Differential Dynamic Logic

Proof Calculus Application

Deduction modulo for MA violation in braking mode

$$\frac{\frac{\frac{v \geq 0, z < m \vdash v^2 > 2b(m - z)}{\vdash v \geq 0 \wedge z < m \rightarrow v^2 > 2b(m - z)}}{v \geq 0, z < m \vdash T \geq 0 \quad \text{D9} \quad \frac{v \geq 0, z < m \vdash -\frac{b}{2}T^2 + vT + z > m}{v \geq 0, z < m \vdash \langle z := -\frac{b}{2}T^2 + vT + z \rangle z > m}}{\frac{v \geq 0, z < m \vdash T \geq 0 \wedge \langle z := -\frac{b}{2}T^2 + vT + z \rangle z > m}{v \geq 0, z < m \vdash \exists t \geq 0 \langle z := -\frac{b}{2}t^2 + vt + z \rangle z > m}}{\frac{v \geq 0, z < m \vdash \langle z' = v, v' = -b \rangle z > m}{\vdash v \geq 0 \wedge z < m \rightarrow \langle z' = v, v' = -b \rangle z > m}}$$

$$\text{QE } (\exists T ((v \geq 0 \wedge z < m \rightarrow T \geq 0) \wedge (v \geq 0 \wedge z < m \rightarrow -\frac{b}{2}T^2 + vT + z > m)))$$

$$\equiv v \geq 0 \wedge z < m \rightarrow v^2 > 2b(m - z) .$$

Differential Dynamic Logic

Proof Calculus Application

European Train Control System (ETCS) Safety

- ▶ Invariant $\phi \equiv v^2 \leq 2b(m - z) \wedge b > 0 \wedge A \geq 0$ can be deduced in the manner of finding the violation condition.
- ▶ The safety proof can then be completed by showing
 - ▶ Invariant holds initially $\psi \vdash \phi$
 - ▶ $\phi \rightarrow [(ctrl; drive)^*]\phi$ holds.
 - ▶ $\phi \rightarrow \varphi_{post}$, where $\varphi_{post} \equiv (z \leq m)$.

Differential Dynamic Logic

Beyond Deductive Verification

- ▶ Verification of a Hybrid System still doesn't guarantee the physical system behaves as intended.
- ▶ Hybrid System involve interaction with real world physics, which can never be captured fully by any model.
- ▶ Runtime monitoring used to obtain real world compliance.

Runtime Monitoring (ModelPlex)

- ▶ ModelPlex (Mitsch et al.) used dL for runtime validation of systems verified in dL.
- ▶ Uses dL to obtain monitoring conditions over Reals (which can be checked via SMT solvers)

$$\begin{array}{c} (v, \omega) \in \rho(\alpha^*) \\ \Downarrow \\ (v, \omega) \models \langle \alpha^* \rangle Y^+ \\ \Uparrow \\ (v, \omega) \models F(x, x^+) \end{array}$$

- ▶ Generate sandbox controller with strong safety guarantees.

Logic and Hybrid System

Conclusion

- ▶ dL provides Hybrid Programs - concise notation for Hybrid Systems.

Logic and Hybrid System

Conclusion

- ▶ dL provides Hybrid Programs - concise notation for Hybrid Systems.
- ▶ A proof calculus suited for verification of CPS and Embedded Systems.

Logic and Hybrid System

Conclusion

- ▶ dL provides Hybrid Programs - concise notation for Hybrid Systems.
- ▶ A proof calculus suited for verification of CPS and Embedded Systems.
- ▶ Rich toolchain (Keymaera), ModelPlex built on top of dL.