Quantified Differential Invariants

André Platzer

Carnegie Mellon University, Pittsburgh, PA
Outline

1 Motivation

2 Quantified Differential Dynamic Logic QdŁ
   - Design
   - Syntax
   - Semantics

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Air Traffic Control
   - Derivations and Differentiation
   - Soundness and Completeness

4 Conclusions
Q: Verify my plane?

Challenge
Q: Verify my plane? A: Hybrid systems

Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
Q: Verify my plane? A: Hybrid systems Q: But there’s lots of planes!

Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
Q: Verify lots of planes?

Challenge
Q: Verify lots of planes? A: Distributed systems

Challenge (Distributed Systems)

- Local computation (finite state automaton)
- Remote communication (network graph)
Q: Verify lots of planes? A: Distributed systems
Q: But they move!

Challenge (Distributed Systems)

- Local computation (finite state automaton)
- Remote communication (network graph)
Complex Physical Systems:

Q: Verify lots of moving planes?

Challenge

Continuous dynamics
(differential equations)
Discrete dynamics
(control decisions)
Structural dynamics
(remote communication)
Dimensional dynamics
(appearance)
Q: Verify lots of moving planes?  A: Distributed hybrid systems

**Challenge (Distributed Hybrid Systems)**

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (remote communication)
Complex Physical Systems: Distributed Hybrid Systems

Q: Verify lots of moving planes? A: Distributed hybrid systems

Challenge (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (remote communication)
- Dimensional dynamics (appearance)
Q: Verify lots of moving planes? A: Distributed hybrid systems Q: How?

Challenge (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (remote communication)
- Dimensional dynamics (appearance)
<table>
<thead>
<tr>
<th>State of the Art:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shift [DGV96]</strong> The Hybrid System Simulation Programming Language</td>
</tr>
<tr>
<td><strong>Hybrid CSP [CJR95]</strong> Semantics in Extended Duration Calculus</td>
</tr>
<tr>
<td><strong>HyPA [CR05]</strong> Translate fragment into normal form.</td>
</tr>
<tr>
<td><strong>χ process algebra [vBMR⁺06]</strong> Simulation, translation of fragments to PHAVER, UPPAAL</td>
</tr>
<tr>
<td><strong>R-Charon [KSPL06]</strong> Modeling Language for Reconfigurable Hybrid Systems</td>
</tr>
<tr>
<td><strong>Φ-calculus [Rou04]</strong> Semantics in rich set theory</td>
</tr>
<tr>
<td><strong>ACP_{hs} [BM05]</strong> Modeling language proposal</td>
</tr>
<tr>
<td><strong>OBSHS [MS06]</strong> Partial random simulation of objects</td>
</tr>
</tbody>
</table>
### State of the Art: Modeling and Simulation

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid CSP [CJR95] Semantics in Extended Duration Calculus</td>
<td>Φ-calculus [Rou04] Semantics in rich set theory</td>
</tr>
<tr>
<td>HyPA [CR05] Translate fragment into normal form.</td>
<td>ACP_{hs}^{srt} [BM05] Modeling language proposal</td>
</tr>
<tr>
<td>χ process algebra [vBMR+06] Simulation, translation of fragments to PHAVER, UPPAAL</td>
<td>OBSHS [MS06] Partial random simulation of objects</td>
</tr>
</tbody>
</table>
No formal verification of distributed hybrid systems

**Shift [DGV96]** The Hybrid System Simulation Programming Language

**R-Charon [KSPL06]** Modeling Language for Reconfigurable Hybrid Systems

**Hybrid CSP [CJR95]** Semantics in Extended Duration Calculus

**Φ-calculus [Rou04]** Semantics in rich set theory

**HyPA [CR05]** Translate fragment into normal form.

**ACP_{srt} [BM05]** Modeling language proposal

**χ process algebra [vBMR+06]** Simulation, translation of fragments to PHAVER, UPPAAL

**OBSHS [MS06]** Partial random simulation of objects
1 Motivation

2 Quantified Differential Dynamic Logic Qd\(\mathcal{L}\)
   - Design
   - Syntax
   - Semantics

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Air Traffic Control
   - Derivations and Differentiation
   - Soundness and Completeness

4 Conclusions
Outline (Conceptual Approach)

1 Motivation

2 Quantified Differential Dynamic Logic QdL
   - Design
   - Syntax
   - Semantics

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Air Traffic Control
   - Derivations and Differentiation
   - Soundness and Completeness

4 Conclusions
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x' = d, \quad d' = f(\omega, d) \]

- Discrete dynamics (control decisions)

- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x' = d, \quad d' = f(\omega, d) \]
- Discrete dynamics (control decisions)
  \[ \omega := \text{if .. then } 0 \text{ else } 2 \]
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x' = d, \quad d' = f(\omega, d) \]

- Discrete dynamics (control decisions)
  \[ \omega := \text{if} .. \text{then} 0 \text{else} 2 \]

- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x' = d, \quad d' = f(\omega, d) \]
- Discrete dynamics (control decisions)
  \[ \omega := \text{if .. then } 0 \text{ else } 2 \]
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x(i)' = d(i), \quad d(i)' = f(\omega(i), d(i)) \]
- Discrete dynamics (control decisions)
  \[ \omega(i) := \text{if .. then} \ 0 \ \text{else} \ 2 \]
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

**Model (Distributed Hybrid Systems)**

- **Continuous dynamics** (differential equations)
  \[ \forall i \ x(i)' = d(i), \ d(i)' = f(\omega(i), d(i)) \]

- **Discrete dynamics** (control decisions)
  \[ \forall i \ \omega(i) := \text{if.. then } 0 \text{ else } 2 \]

- **Structural dynamics** (communication/coupling)

---

Andre Platzer (CMU)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- **Continuous dynamics** (differential equations)
  \[ \forall i \, x(i)' = d(i), \, d(i)' = f(\omega(i), d(i)) \]

- **Discrete dynamics** (control decisions)
  \[ \forall i \, \omega(i) := \text{if } \ldots \text{then } 0 \text{ else } 2 \]

- **Structural dynamics** (communication/coupling)
  \[ c(i) := \text{negotiate}(i,j) \]
Q: How to model distributed hybrid systems

A: Quantified Hybrid Programs

Model (Distributed Hybrid Systems)

- Continuous dynamics
  (differential equations)
  \[ \forall i \ x(i)' = d(i), \ d(i)' = f(\omega(i), d(i)) \]

- Discrete dynamics
  (control decisions)
  \[ \forall i \ \omega(i) := \text{if } \ldots \text{then } 0 \text{ else } 2 \]

- Structural dynamics
  (communication/coupling)
  \[ c(i) := \text{negotiate}(i,j) \]

- Dimensional dynamics
  (appearance)
Q: How to model distributed hybrid systems

A: Quantified Hybrid Programs

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ \forall i \ x(i)' = d(i), \ d(i)' = f(\omega(i), d(i)) \]

- Discrete dynamics (control decisions)
  \[ \forall i \ \omega(i) := \text{if .. then 0 else 2} \]

- Structural dynamics (communication/coupling)
  \[ c(i) := \text{negotiate}(i,j) \]

- Dimensional dynamics (appearance)
  \[ n := \text{new Aircraft} \]
### Syntax

**Definition (Quantified hybrid program $\alpha$)**

- $\forall i : C \ x(i)' = \theta$ (quantified ODE)
- $\forall i : C \ x(i) := \theta$ (quantified assignment)
- $?\chi$ (conditional execution)
- $\alpha; \beta$ (seq. composition)
- $\alpha \cup \beta$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)

### Kleene algebra

- $\alpha \cup \beta$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)

- **Jump & Test**
### Definition (Quantified hybrid program $\alpha$)

- $\forall i : C \ x(i)' = \theta$ (quantified ODE)
- $\forall i : C \ x(i) := \theta$ (quantified assignment)
- $?\chi$ (conditional execution)
- $\alpha; \beta$ (seq. composition)
- $\alpha \cup \beta$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)

### Kleene algebra

\[
\text{jump \& test} \quad \{ \quad \text{Kleene algebra} \\
\]

### $\text{DATC} \equiv (\text{ctrl}; \text{fly})^*$

- $\text{ctrl} \equiv \forall i : A \ \omega(i) := \text{if } \forall j : A \ \text{far}(i, j) \ \text{then } 0 \ \text{else } 2$
- $\text{fly} \equiv \forall i : A \ x(i)'' = d(i), \ d(i)' = f(\omega(i), d(i))$

---

André Platzer (CMU)
### Definition (Quantified hybrid program $\alpha$)

- $\forall i : C \ x(i)' = \theta$ (quantified ODE)
- $\forall i : C \ x(i) := \theta$ (quantified assignment)
- $?\chi$ (conditional execution)
- $\alpha; \beta$ (seq. composition)
- $\alpha \cup \beta$ (nondet. choice)
- $\alpha^*$ (nondet. repetition)

#### Jump & Test

- $\forall i : C \ x(i)' = \theta$
- $\forall i : C \ x(i) := \theta$
- $?\chi$
- $\alpha; \beta$
- $\alpha \cup \beta$
- $\alpha^*$

#### Kleene Algebra

- $\forall i : C \ x(i)' = \theta$
- $\forall i : C \ x(i) := \theta$
- $?\chi$
- $\alpha; \beta$
- $\alpha \cup \beta$
- $\alpha^*$

---

**DATC** $\equiv$ $(\text{appear} ; \text{ctrl} ; \text{fly})^*$

**appear** $\equiv$ $n := \text{new } A ;$ $?(\forall j : A \ \text{far}(j, n))$

**ctrl** $\equiv$ $\forall i : A \ \omega(i) := \text{if } \forall j : A \ \text{far}(i, j) \ \text{then } 0 \ \text{else } 2$

**fly** $\equiv$ $\forall i : A \ x(i)'' = d(i), \ d(i)' = f(\omega(i), d(i))$
Quantified Differential Dynamic Logic Qd\(\mathcal{L}\): Syntax

### Definition (Quantified hybrid program \(\alpha\))

\[
\begin{align*}
\forall i : C \ x(i)' &= \theta & \text{(quantified ODE)} \\
\forall i : C \ x(i) &:= \theta & \text{(quantified assignment)} \\
?\chi & & \text{(conditional execution)} \\
\alpha; \beta & & \text{(seq. composition)} \\
\alpha \cup \beta & & \text{(nondet. choice)} \\
\alpha^* & & \text{(nondet. repetition)} \\
\end{align*}
\]

\[
\begin{align*}
\text{jump \\& test} \\
\text{Kleene algebra}
\end{align*}
\]

\[
\text{DATC} \equiv (\text{appear}; \text{ctrl}; \text{fly})^*
\]

\[
\begin{align*}
\text{appear} & \equiv n := \text{new } A; \ ?(\forall j : A \ \text{far}(j, n)) \\
\text{ctrl} & \equiv \forall i : A \ \omega(i) := \text{if } \forall j : A \ \text{far}(i, j) \ \text{then } 0 \ \text{else } 2 \\
\text{fly} & \equiv \forall i : A \ x(i)'' = d(i), d(i)' = f(\omega(i), d(i)) \\
\end{align*}
\]

\text{new } A \text{ is definable!}
Definition (Quantified hybrid program $\alpha$)

\[
\begin{align*}
\forall i &: C \ x(i)' = \theta && \text{(quantified ODE)} \\
\forall i &: C \ x(i) := \theta && \text{(quantified assignment)} \\
?\chi && \text{(conditional execution)} \\
\alpha; \beta && \text{(seq. composition)} \\
\alpha \cup \beta && \text{(nondet. choice)} \\
\alpha^* && \text{(nondet. repetition)} \\
\end{align*}
\]

\[\{ \text{jump \& test} \}\]
\[\{ \text{Kleene algebra} \}\]

\[\text{DATC} \equiv (\text{appear}; \text{ctrl}; \text{fly})^*\]

\[\text{appear} \equiv n := \text{new } A; \quad ?(\forall j : A \ \text{far}(j, n))\]

\[\text{ctrl} \equiv \forall i : A \ \omega(i) := \text{if } \forall j : A \ \text{far}(i, j) \ \text{then } 0 \ \text{else } 2\]

\[\text{fly} \equiv \forall i : A \ x(i)'' = d(i), \ d(i)' = f(\omega(i), d(i))\]
Quantified Differential Dynamic Logic $\text{QdL}$: Syntax

**Definition (QdL Formula $\phi$)**

\[ \neg, \land, \lor, \rightarrow, \forall x, \exists x, =, \leq, +, \cdot \quad (\mathbb{R}\text{-first-order part}) \]

\[ [\alpha] \phi, \langle \alpha \rangle \phi \quad (\text{dynamic part}) \]

\[ \forall i, j : A \ far(i, j) \rightarrow \]

\[ [(\text{appear}; \ ctrl; \ fly)^*] \ \forall i, j : A \ (i = j \lor (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq p^2) \]
Definition (Quantified hybrid program $\alpha$: transition semantics)

$\forall i : C \ x(i) := \theta$

- $v$ transitions to $w$ if $w(x)(v^e_i[i]) = v^e_i[\theta]$ (for all $e$)
- and otherwise unchanged
Quantified Differential Dynamic Logic \( \mathcal{QdL} \): Semantics

**Definition (Quantified hybrid program \( \alpha \): transition semantics)**

\[
\forall i : C \ x(i)' = \theta
\]

\[\begin{align*}
\forall i : C \ x(i)' &= \theta \\
\frac{d \varphi(t)_i^e \{ x(i) \}}{dt}(\zeta) &= \varphi(\zeta)_i^e \{ \theta \} \quad \text{(for all } e )
\end{align*}\]
Definition (Quantified hybrid program $\alpha$: transition semantics)

$\alpha; \beta$

$V \xrightarrow{\alpha} S \xrightarrow{\beta} W$
Definition (Quantified hybrid program $\alpha$: transition semantics)

$v \xrightarrow{\alpha} s \xrightarrow{\beta} w$

$x \rightarrow v \rightarrow s \rightarrow w$

André Platzer (CMU)
Definition (Quantified hybrid program \( \alpha \): transition semantics)

\[
\alpha; \beta
\]

\( v \xrightarrow{\alpha} s \xrightarrow{\beta} w \)
Definition (Quantified hybrid program $\alpha$: transition semantics)

$$v \xrightarrow{\alpha} s_1 \xrightarrow{\alpha} s_2 \xrightarrow{\ldots} s_n \xrightarrow{\alpha} w$$

$$\alpha^*$$
Definition (Quantified hybrid program $\alpha$: transition semantics)

$$\alpha^*$$

$V \xrightarrow{\alpha} S_1 \xrightarrow{\alpha} S_2 \xrightarrow{\alpha} \ldots \xrightarrow{\alpha} S_n \xrightarrow{\alpha} W$
Definition (Quantified hybrid program $\alpha$: transition semantics)

$v \xrightarrow{\alpha} w_1 \xleftarrow{\alpha \cup \beta} v \\
\xrightarrow{\beta} w_2$
Definition (Quantified hybrid program $\alpha$: transition semantics)

$v \xrightarrow{\alpha} w_1 \cup \beta \xrightarrow{\beta} w_2$

Diagram:

$X$ axis:
- $v$ to $w_1$ (red)
- $v$ to $w_2$ (blue)

$Y$ axis:
- $v$ to $w_1$
- $w_2$

André Platzer (CMU)
Definition (Quantified hybrid program $\alpha$: transition semantics)

$$
\text{if } v \models \chi
$$

André Platzer (CMU)

Quantified Differential Invariants
Definition (Quantified hybrid program $\alpha$: transition semantics)

$v \models \chi$

- no change if $v \models \chi$
- otherwise no transition

Andre Platzer (CMU)
Definition (QdL Formula $\phi$

$$
\phi \vdash [\alpha] \phi \vdash \phi
$$

Details
Definition (QdŁ Formula $\phi$)

$\langle \alpha \rangle \phi$

$\phi$

$\alpha$-span

$\beta$-span

Details

André Platzer (CMU)
Definition (QdŁ Formula $\phi$)

$$[\alpha] \phi$$

$$\alpha\text{-span}$$
Definition (QdŁ Formula $\phi$)

$v_\alpha$-span $\{\alpha\}$

$\langle \beta \rangle \phi$

$\beta$-span

$[\alpha] \phi$

$\alpha$-span

Details

André Platzer (CMU)

Quantified Differential Invariants
Definition (QdL Formula $\phi$)

$\langle \beta \rangle \phi$

$[\alpha] \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$

$\langle \beta \rangle \phi$
Definition (QdŁ Formula $\phi$)

$\langle \beta \rangle \phi$

$[\alpha] \phi$

$\beta$-span

$\langle \beta \rangle \phi$

$[\alpha] \phi$

$\alpha$-span

compositional semantics $\Rightarrow$ compositional calculus!
Outline (Verification Approach)

1 Motivation

2 Quantified Differential Dynamic Logic $QdL$
   - Design
   - Syntax
   - Semantics

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Air Traffic Control
   - Derivations and Differentiation
   - Soundness and Completeness

4 Conclusions
\[
\forall i (i = u \rightarrow \phi(\theta)) \\
\phi([\forall i \ x(i) := \theta]x(u))
\]
∀i (i = [∀i x(i) := θ]u → φ(θ))

φ([∀i x(i) := θ]x(u))
∀i (i = [∀i x(i) := θ] u → φ(θ))

φ([∀i x(i) := θ] x(u))

∃t ≥ 0 ⟨∀i S(t)⟩ φ

⟨∀i x(i)′ = θ⟩ φ

∀i x(i) := θ

∀i x(i)′ = θ

v

w

φ

φ
∀i (i = [∀i x(i) := θ]u → φ(θ))

φ([∀i x(i) := θ]x(u))

∃t ≥ 0 ⟨∀i S(t)⟩φ

⟨∀i x(i)' = θ⟩φ

∀i S(t)
\[
\forall i (i = [\forall i x(i) := \theta]u \rightarrow \phi(\theta)) \implies \phi([\forall i x(i) := \theta]x(u))
\]

\[
\exists t \geq 0 \langle \forall i S(t) \rangle \phi \implies \langle \forall i x(i)' = \theta \rangle \phi
\]

solve infinite-dimensional diff. eqn.?
Proof Calculus for Quantified Differential Dynamic Logic

compositional semantics \Rightarrow \text{compositional rules!}
\[
\frac{[\alpha] \phi \land [\beta] \phi}{[\alpha \cup \beta] \phi}
\]
\[
\frac{[\alpha] \phi \land [\beta] \phi}{[\alpha \cup \beta] \phi}
\]

\[
\frac{[\alpha][\beta] \phi}{[\alpha; \beta] \phi}
\]
Proof Calculus for Quantified Differential Dynamic Logic

\[
\frac{\left[\alpha\right]\phi \land \left[\beta\right]\phi}{\left[\alpha \cup \beta\right]\phi}
\]

\[
\frac{\left[\alpha\right]\left[\beta\right]\phi}{\left[\alpha ; \beta\right]\phi}
\]

\[
\frac{\phi \quad (\phi \rightarrow \left[\alpha\right]\phi)}{\left[\alpha^*\right]\phi}
\]
Air Traffic Control
Verification?

looks correct
Verification?

looks correct NO!
\[
\begin{bmatrix}
  x'_1 \\
  x'_2 \\
  \vartheta'
\end{bmatrix}
= \begin{bmatrix}
  -v_1 + v_2 \cos \vartheta + \omega x_2 \\
  v_2 \sin \vartheta - \omega x_1 \\
  \varpi - \omega
\end{bmatrix}
\]

Verification?

looks correct NO!
Example ("Solving" differential equations)

\[
\begin{align*}
    x_1(t) &= \frac{1}{\omega \varpi} \left( x_1 \omega \varpi \cos t \omega - v_2 \omega \cos t \omega \sin \vartheta + v_2 \omega \cos t \omega \cos t \varpi \sin \vartheta - v_1 \varpi \sin t \omega \\
    &\quad + x_2 \omega \varpi \sin t \omega - v_2 \omega \cos \vartheta \cos t \varpi \sin t \omega - v_2 \omega \sqrt{1 - \sin \vartheta^2} \sin t \omega \\
    &\quad + v_2 \omega \cos \vartheta \cos t \omega \sin t \varpi + v_2 \omega \sin \vartheta \sin t \omega \sin t \varpi \right) \ldots
\end{align*}
\]
Air Traffic Control

\[ \begin{align*}
\dot{x}_1 &= -v_1 + v_2 \cos \vartheta + \omega x_2 \\
\dot{x}_2 &= v_2 \sin \vartheta - \omega x_1 \\
\dot{\vartheta} &= \varpi - \omega
\end{align*} \]

Example ("Solving" differential equations)

\[ \forall t \geq 0 \quad \frac{1}{\omega \varpi} \left( x_1 \omega \varpi \cos t \omega - v_2 \omega \cos t \omega \sin \vartheta + v_2 \omega \cos t \omega \cos t \varpi \sin \vartheta - v_1 \omega \sin t \omega \\
+ x_2 \omega \varpi \sin t \omega - v_2 \omega \cos \vartheta \cos t \varpi \sin t \omega - v_2 \omega \sqrt{1 - \sin \vartheta^2} \sin t \omega \\
+ v_2 \omega \cos \vartheta \cos t \omega \sin t \varpi + v_2 \omega \sin \vartheta \sin t \omega \sin t \varpi \right) \ldots \]
Idea (Differential Invariant)
Formulat that remains true in the direction of the dynamics

 André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
Idea (Differential Invariant)

Formula that remains true in the direction of the dynamics

Idea (Differential Invariant)

Formula that remains true in the direction of the dynamics

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
Idea (Differential Invariant)

Formula that remains true in the direction of the dynamics

\[ \mathbb{R}^2 \text{ but } \mathbb{R}^\infty ?? \]

André Platzer.

Differential-algebraic dynamic logic for differential-algebraic programs.

Definition (Differential Invariant)

\( F \) closed under total differentiation with respect to differential constraints

\[ (\chi \rightarrow F') \]

\[ \chi \rightarrow F \rightarrow [x' = \theta \land \chi]F \]
Definition (Differential Invariant)

\( F \) closed under total differentiation with respect to differential constraints

\[
\begin{align*}
\chi \to F' \\
\chi \to F \rightarrow [x' = \theta \land \chi] F
\end{align*}
\]

Total differential \( F' \) of formulas?
Quantified Differential Invariants

Definition (Quantified Differential Invariant)

Quantified formula $F$ closed under total differentiation with respect to quantified differential constraints
Definition (Syntactic total derivation $D$)

\[
\begin{align*}
D(r) &= 0 & \text{if } r \text{ a number symbol} \\
D(x(i)) &= x(i)' & \text{if } x : C \rightarrow \mathbb{R}, \ C \neq \mathbb{R} \\
D(a + b) &= D(a) + D(b) \\
D(a \cdot b) &= D(a) \cdot b + a \cdot D(b) \\
D(a/b) &= (D(a) \cdot b - a \cdot D(b))/b^2
\end{align*}
\]
Derivations and Differentiation

Definition (Syntactic total derivation $D$)

- $D(r) = 0$ if $r$ a number symbol
- $D(x(i)) = x(i)'$ if $x : C \to \mathbb{R}$, $C \neq \mathbb{R}$
- $D(a + b) = D(a) + D(b)$
- $D(a \cdot b) = D(a) \cdot b + a \cdot D(b)$
- $D(a/b) = (D(a) \cdot b - a \cdot D(b))/b^2$
- $D(a \geq b) \equiv D(a) \geq D(b)$ accordingly for $>$, $=$
- $D(F \land G) \equiv D(F) \land D(G)$
- $D(\forall i F) \equiv \forall i D(F)$
**Definition (Syntactic total derivation $D$)**

- $D(r) = 0$ if $r$ a number symbol
- $D(x(i)) = x(i)'$ if $x : C → \mathbb{R}, C \not= \mathbb{R}$
- $D(a + b) = D(a) + D(b)$
- $D(a \cdot b) = D(a) \cdot b + a \cdot D(b)$
- $D(a/b) = (D(a) \cdot b - a \cdot D(b))/b^2$
- $D(a ≥ b) ≡ D(a) ≥ D(b)$ accordingly for $> , =$
- $D(F \land G) ≡ D(F) \land D(G)$
- $D(∀i F) ≡ ∀i D(F)$

$\mathcal{P} ≡ ∀i, j : A (i = j \lor (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 ≥ p^2)$

$⇒ D(\mathcal{P}) ≡ ∀i, j : A (i' = j' \land 2(x_1(i) - x_1(j))(x_1(i)' - x_1(j)')$

$+ 2(x_2(i) - x_2(j))(x_2(i)' - x_2(j)') ≥ 0)$
Derivations and Differentiation

Definition (Syntactic total derivation $D$)

\[
D(r) = 0 \quad \text{if } r \text{ a number symbol}
\]
\[
D(x(i)) = x(i)' \quad \text{if } x : C \to \mathbb{R}, \ C \neq \mathbb{R}
\]
\[
D(a + b) = D(a) + D(b)
\]
\[
D(a \cdot b) = D(a) \cdot b + a \cdot D(b)
\]
\[
D(a/b) = (D(a) \cdot b - a \cdot D(b))/b^2
\]
\[
D(a \geq b) \equiv D(a) \geq D(b)
\]
\[
D(F \land G) \equiv D(F) \land D(G)
\]
\[
D(\forall i \ F) \equiv \forall i \ D(F)
\]

\[
\mathcal{P} \equiv \forall i, j : A \ (i = j \lor (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq p^2)
\]
\[
\Rightarrow D(\mathcal{P}) \equiv \forall i, j : A \ (i' = j' \land 2(x_1(i) - x_1(j))(x_1(i)' - x_1(j)')
\]
\[
+ 2(x_2(i) - x_2(j))(x_2(i)' - x_2(j)') \geq 0)
\]
Syntactic derivation $D(\cdot)$ coincides with analytic differentiation:

**Lemma (Derivation lemma)**

*Valuation is a differential homomorphism: for all flows $\varphi$ all $\zeta \in [0, r]$*

$$\frac{d \varphi(t)[\theta]}{dt}(\zeta) = \bar{\varphi}(\zeta)[D(\theta)]$$
Syntactic derivation $D(\cdot)$ coincides with analytic differentiation:

**Lemma (Derivation lemma)**

Valuation is a differential homomorphism: for all flows $\varphi$ all $\zeta \in [0, r]$

$$\frac{d}{dt} \varphi(t)[\theta](\zeta) = \bar{\varphi}(\zeta)[D(\theta)]$$

Locally understand QDE as quantified assignments:

**Lemma (Quantified differential substitution principle)**

If $\varphi \models \forall i : C f(i)' = \theta \land \chi$, then $\varphi \models \nu = [\forall i : C f(i)' := \theta]\nu$ for all $\nu$. 
Derivations and Differentiation

Syntactic derivation $D(\cdot)$ coincides with analytic differentiation:

Lemma (Derivation lemma)

Valuation is a differential homomorphism: for all flows $\varphi$ all $\zeta \in [0, r]$

$$\frac{d \varphi(t)[\theta]}{dt}(\zeta) = \bar{\varphi}(\zeta)[D(\theta)]$$

Locally understand QDE as quantified assignments:

Lemma (Quantified differential substitution principle)

If $\varphi \models \forall i : C \; f(i)' = \theta \land \chi$, then $\varphi \models \nu = [\forall i : C \; f(i)' := \theta] \nu$ for all $\nu$.

Theorem (Quantified Differential Invariant)

(QDI) $\chi \rightarrow [\forall i : C \; f(i)' := \theta] D(F)$

$F \rightarrow [\forall i : C \; f(i)' = \theta \land \chi] F$ is sound
∀i: C 2x(i)^3 ≥ 1 → [∀i: C x(i)' = x(i)^2 + x(i)^4 + 2] ∀i: C 2x(i)^3 ≥ 1
\begin{align*}
\forall i : C \ x(i)' &:= x(i)^2 + x(i)^4 + 2 \forall i : C \ 2(x(i)^3)' \geq 0 \\
\forall i : C \ 2x(i)^3 &\geq 1 \Rightarrow [\forall i : C \ x(i)' = x(i)^2 + x(i)^4 + 2] \forall i : C \ 2x(i)^3 \geq 1
\end{align*}
∀ i : C  \( x(i)' = x(i)^2 + x(i)^4 + 2 \) \( \forall i : C \) \( 6x(i)^2 x(i)' \geq 0 \)

∀ i : C  \( x(i)' = x(i)^2 + x(i)^4 + 2 \) \( \forall i : C \) \( 2(x(i)^3)' \geq 0 \)

\( \forall i : C 2x(i)^3 \geq 1 \rightarrow \) \( \forall i : C x(i)' = x(i)^2 + x(i)^4 + 2 \) \( \forall i : C 2x(i)^3 \geq 1 \)
\[ \forall i : C \ 6x(i)^2(x(i)^2 + x(i)^4 + 2) \geq 0 \]

\[ \forall i : C \ x(i)' := x(i)^2 + x(i)^4 + 2 \forall i : C \ 6x(i)^2x(i)' \geq 0 \]

\[ \forall i : C \ x(i)' := x(i)^2 + x(i)^4 + 2 \forall i : C \ 2(x(i)^3)' \geq 0 \]

\[ \forall i : C \ 2x(i)^3 \geq 1 \rightarrow \forall i : C \ x(i)' = x(i)^2 + x(i)^4 + 2 \forall i : C \ 2x(i)^3 \geq 1 \]
true

\( \forall i : C \ 6x(i)^2(x(i)^2 + x(i)^4 + 2) \geq 0 \)

\[ \forall i : C \ x(i)' := x(i)^2 + x(i)^4 + 2 \] \( \forall i : C \ 6x(i)^2x(i)' \geq 0 \)

\[ \forall i : C \ x(i)' := x(i)^2 + x(i)^4 + 2 \] \( \forall i : C \ 2(x(i)^3)' \geq 0 \)

\( \forall i : C \ 2x(i)^3 \geq 1 \rightarrow \forall i : C \ x(i)' = x(i)^2 + x(i)^4 + 2 \) \( \forall i : C \ 2x(i)^3 \geq 1 \)
\[ \forall i \ x_1(i)' = d_1(i), \quad d_1(i)' = -\omega d_2(i), \quad x_2(i)' = d_2(i), \quad d_2(i)' = \omega d_1(i) \] 

\[ (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0 \]
\[ i' = j' \land 2(x_1(i) - x_1(j))(x_1(i)' - x_1(j)') + 2(x_2(i) - x_2(j))(x_2(i)' - x_2(j)') \geq 0 \]

\[ [\forall ix_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i)}
\[ i' = j' \land 2(x_1(i) - x_1(j))(x_1(i)' - x_1(j)') + 2(x_2(i) - x_2(j))(x_2(i)' - x_2(j)') \geq 0 \]

\[ [\forall i x_1(i)' = d_1(i), \ d_1(i)' = -\omega d_2(i), \ x_2(i)' = d_2(i), \ d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0 \]
Differential Induction for Aircraft Roundabouts

\[ 0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ [\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0 \]
2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

[\forall i] x_1'(i) = d_1(i), d_1'(i) = -\omega d_2(i), x_2'(i) = d_2(i), d_2'(i) = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0
2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0
0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2
Differential Induction for Aircraft Roundabouts

\[ 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ 0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ [\forall i]x_1(i)' = d_1(i), \ d_1(i)' = -\omega d_2(i), \ x_2(i)' = d_2(i), \ d_2(i)' = \omega d_1(i)] \]

\[ (x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0 \]

\[ \forall i \ x_1(i)' = d_1(i), \ d_1(i)' = -\omega d_2(i), \ x_2(i)' = d_2(i), \ d_2(i)' = \omega d_1(i) ] d_1(i) - d_1(j) = -\omega (x_2(i) - x_2(j)) \]
Differential Induction for Aircraft Roundabouts

\[
2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0
\]

\[
2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0
\]

\[
0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0
\]

\[
[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0
\]
Differential Induction for Aircraft Roundabouts

\[ 2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0 \]

\[ 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ 0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ [\forall ix_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i)) \]

\[ d_1(i)' - d_1(j)' = -\omega(x_2(i)' - x_2(j)') \]

\[ [\forall ix_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)]d_1(i) - d_1(j) = -\omega(x_2(i)' - x_2(j)') \]
Differential Induction for Aircraft Roundabouts

2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0

2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

[\forall i] x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i) [(x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2] \geq 0

\begin{align*}
d_1(i)' - d_1(j)' &= -\omega(x_2(i)' - x_2(j)') \\
[\forall i] x_1(i)' &= d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' &= d_2(i), d_2(i)' &= \omega d_1(i) \Rightarrow d_1(i) - d_1(j) = -\omega(x_2(i) - x_2(j))
\end{align*}
Differential Induction for Aircraft Roundabouts

\[
2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0
\]

\[
2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0
\]

\[
0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0
\]

\[
[\forall ix_1(i)' = d_1(i), \ d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), \ d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0
\]

\[
-\omega d_2(i) - -\omega d_2(j) = -\omega(d_2(i) - d_2(j))
\]

\[
[\forall ix_1(i)' = d_1(i), \ d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), \ d_2(i)' = \omega d_1(i)]d_1(i) - d_1(j) = -\omega(x_2(i) - x_2(j))
\]
Differential Induction for Aircraft Roundabouts

\[ 2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0 \]

\[ 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ 0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \]

\[ [\forall ix_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i)) \]

\[ -\omega d_2(i) + \omega d_2(j) = -\omega(d_2(i) - d_2(j)) \]

\[ -\omega d_2(i) - -\omega d_2(j) = -\omega(d_2(i) - d_2(j)) \]

\[ [\forall ix_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)]d_1(i) - d_1(j) = -\omega(x_2(i) - x_2(j)) \]
Differential Induction & Differential Cuts

2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0

2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0

[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i) - x_2(j))^2 \geq 0

Proposition (Differential cut)

\[ F \text{ differential invariant of } [\forall i x(i)' = \theta \land H]\phi, \text{ then } [\forall i x(i)' = \theta \land H]\phi \text{ iff } [\forall i x(i)' = \theta \land H \land F]\phi \]

- \omega d_2(i) + \omega d_2(j) = -\omega(d_2(i) - d_2(j))

- \omega d_2(i) - -\omega d_2(j) = -\omega(d_2(i) - d_2(j))

[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)]d_1(i) - d_1(j) = -\omega(x_2(i) - x_2(j))
2(x_1(i) - x_1(j))(-\omega(x_2(i) - x_2(j))) + 2(x_2(i) - x_2(j))\omega(x_1(i) - x_1(j)) \geq 0 \\
2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \\
0 = 0 \land 2(x_1(i) - x_1(j))(d_1(i) - d_1(j)) + 2(x_2(i) - x_2(j))(d_2(i) - d_2(j)) \geq 0 \\
[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)](x_1(i) - x_1(j))^2 + (x_2(i))

\begin{align*}
-\omega d_2(i) + \omega d_2(j) &= -\omega(d_2(i) - d_2(j)) \\
-\omega d_2(i) - -\omega d_2(j) &= -\omega(d_2(i) - d_2(j)) \\
[\forall i x_1(i)' = d_1(i), d_1(i)' = -\omega d_2(i), x_2(i)' = d_2(i), d_2(i)' = \omega d_1(i)]d_1(i) - d_1(j) &= -\omega(x_2(i))
\end{align*}
Soundness and Completeness

**Theorem (Relative Completeness)**

QdŁ calculus is a sound & complete axiomatisation of distributed hybrid systems relative to quantified differential equations.

Proof 16p.
Soundness and Completeness

Theorem (Relative Completeness)

QdL calculus is a sound & complete axiomatisation of distributed hybrid systems relative to quantified differential equations.

Corollary (Proof-theoretical Alignment)

proving distributed hybrid systems $=$ proving dynamical systems!
Theorem (Relative Completeness)

QdŁ calculus is a sound & complete axiomatisation of distributed hybrid systems relative to quantified differential equations.

Corollary (Proof-theoretical Alignment)

proving distributed hybrid systems = proving dynamical systems!

Corollary (Yes, we can!)

distributed hybrid systems can be verified by recursive decomposition
Outline

1 Motivation

2 Quantified Differential Dynamic Logic QdL
   - Design
   - Syntax
   - Semantics

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Air Traffic Control
   - Derivations and Differentiation
   - Soundness and Completeness

4 Conclusions
Conclusions

Quantified differential dynamic logic

\[ \text{QdL} = \text{FOL} + \text{DL} + \text{QHP} \]

- Quantified differential invariants
- Verify quantified differential equations
- Logic for distributed hybrid systems
- Compositional proof calculus
- Sound & complete / diff. eqn.
- First verification approach
- Verified appearance of aircraft
Conclusions

Quantified differential invariants
Verify quantified differential equations
Logic for distributed hybrid systems
Compositional proof calculus
Sound & complete / diff. eqn.
First verification approach
Verified appearance of aircraft


João P. Hespanha and Ashish Tiwari, editors.  

Fabian Kratz, Oleg Sokolsky, George J. Pappas, and Insup Lee.  
R-Charon, a modeling language for reconfigurable hybrid systems.  
In Hespanha and Tiwari [HT06], pages 392–406.

José Meseguer and Raman Sharykin.  
Specification and analysis of distributed object-based stochastic hybrid systems.  
In Hespanha and Tiwari [HT06], pages 460–475.

William C. Rounds.  
A spatial logic for the hybrid $\pi$-calculus.  

D. A. van Beek, Ka L. Man, Michel A. Reniers, J. E. Rooda, and Ramon R. H. Schiffelers.
Syntax and consistent equation semantics of hybrid Chi.