Outline

1 Motivation

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

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Deduction Modulo with Free Variables & Skolemization
   - Actual Existence and Creation
   - Soundness and Completeness
   - Quantified Differential Invariants

4 Applications
   - Distributed Car Control
   - Surgical Robot

5 Conclusions
Complex Physical Systems:

Q: I want to verify my car

Challenge
Q: I want to verify my car  
A: Hybrid systems

**Challenge (Hybrid Systems)**

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
Q: I want to verify my car
A: Hybrid systems
Q: But there’s a lot of cars!

Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
Q: I want to verify a lot of cars

Challenge
Q: I want to verify a lot of cars  
A: Distributed systems

Challenge (Distributed Systems)

- Local computation (finite state automaton)
- Remote communication (network graph)
Q: I want to verify a lot of cars  
A: Distributed systems  
Q: But they move!

**Challenge (Distributed Systems)**

- Local computation  
  (finite state automaton)
- Remote communication  
  (network graph)
Q: I want to verify lots of moving cars
Q: I want to verify lots of moving cars A: Distributed hybrid systems

Challenge (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (remote communication)
Q: I want to verify lots of moving cars
A: Distributed hybrid systems

Challenge (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (remote communication)
- Dimensional dynamics (appearance)
Q: I want to verify lots of moving cars
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)
No formal verification of distributed hybrid systems

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid CSP [6] Semantics in Extended Duration Calculus</td>
</tr>
<tr>
<td>Φ-calculus [9] Semantics in rich set theory</td>
</tr>
<tr>
<td>ACP$^{srt}_{hs}$ [10] Modeling language proposal</td>
</tr>
<tr>
<td>χ process algebra [8] Simulation, translation of fragments to PHAVER, UPPAAL</td>
</tr>
</tbody>
</table>
Contributions

1. System model and semantics for distributed hybrid systems: QHP
2. Specification and verification logic: QdL
3. Proof calculus for QdL
4. First verification approach for distributed hybrid systems
5. Sound and complete axiomatization relative to differential equations
6. Prove collision freedom in a (simple) distributed car control system, where new cars may appear dynamically on the road
7. Logical foundation for analysis of distributed hybrid systems
8. Fundamental extension: first-order $x(i)$ versus primitive $x$
Outline

1 Motivation

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

3 Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Deduction Modulo with Free Variables & Skolemization
   - Actual Existence and Creation
   - Soundness and Completeness
   - Quantified Differential Invariants

4 Applications
   - Distributed Car Control
   - Surgical Robot

5 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
   - Deduction Modulo with Free Variables & Skolemization
   - Actual Existence and Creation
   - Soundness and Completeness
   - Quantified Differential Invariants

4 Applications
   - Distributed Car Control
   - Surgical Robot

5 Conclusions
Model for 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)

 André Platzer (KIT || CMU)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x'' = a \]
- Discrete dynamics (control decisions)
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x'' = a \]
- Discrete dynamics (control decisions)
  \[ a := \text{if..then } A \text{ else } -b \]
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x'' = a \]

- Discrete dynamics (control decisions)
  \[ a := \text{if } \ldots \text{then } A \text{ else } -b \]

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

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x'' = a \]
- Discrete dynamics (control decisions)
  \[ a := \text{if..} \text{then} A \text{else} -b \]
- Structural dynamics (communication/coupling)

André Platzer (KIT || CMU)
Logic of Distributed Hybrid Systems
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ x(i)'' = a(i) \]

- Discrete dynamics (control decisions)
  \[ a(i) := \text{if..then } A \text{ else } -b \]

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

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ \forall i \ x(i)'' = a(i) \]
- Discrete dynamics (control decisions)
  \[ \forall i \ a(i) := \text{if}..\text{then} A \text{else} b \]
- Structural dynamics (communication/coupling)
Q: How to model distributed hybrid systems

Model (Distributed Hybrid Systems)

- **Continuous dynamics**
  (differential equations)
  \[ \forall i \ x(i)'' = a(i) \]

- **Discrete dynamics**
  (control decisions)
  \[ \forall i \ a(i) := \text{if..then A else } -b \]

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

A: Quantified Hybrid Programs

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ \forall i \ x(i)'' = a(i) \]

- Discrete dynamics (control decisions)
  \[ \forall i \ a(i) := \text{if } \ldots \text{then } A \text{ else } -b \]

- Structural dynamics (communication/coupling)
  \[ \ell(i) := \text{carInFrontOf}(i) \]

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

A: Quantified Hybrid Programs

Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  \[ \forall i \, x(i)'' = a(i) \]
- Discrete dynamics (control decisions)
  \[ \forall i \, a(i) := \text{if..then} \ A \ \text{else} -b \]
- Structural dynamics (communication/coupling)
  \[ \ell(i) := \text{carInFrontOf}(i) \]
- Dimensional dynamics (appearance)
  \[ n := \text{new Car} \]
**Definition (Quantified hybrid program \( \alpha \))**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \forall i : C \ x(i)' = \theta )</td>
<td>(quantified ODE)</td>
</tr>
<tr>
<td>( \forall i : C \ x(i) := \theta )</td>
<td>(quantified assignment)</td>
</tr>
<tr>
<td>(?Q)</td>
<td>(conditional execution)</td>
</tr>
<tr>
<td>( \alpha;\beta )</td>
<td>(seq. composition)</td>
</tr>
<tr>
<td>( \alpha \cup \beta )</td>
<td>(nondet. choice)</td>
</tr>
<tr>
<td>( \alpha^* )</td>
<td>(nondet. repetition)</td>
</tr>
</tbody>
</table>

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

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

\(?Q\)

\( \alpha;\beta \)

\( \alpha \cup \beta \)

\( \alpha^* \)

\( \{ \text{jump \\& test} \} \)

\( \{ \text{Kleene algebra} \} \)
Definition (Quantified hybrid program $\alpha$)

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

\{ jump & test \  \\
\{ Kleene algebra \}

Andre Platzer (KIT || CMU)
Quantified Differential Dynamic Logic $\mathcal{QdL}$: Syntax

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

$$
\forall i : C \ x(s)' = \theta \quad \text{(quantified ODE)}
$$

$$
\forall i : C \ x(s) := \theta \quad \text{(quantified assignment)}
$$

$$
?Q \quad \text{(conditional execution)}
$$

$$
\alpha ; \beta \quad \text{(seq. composition)}
$$

$$
\alpha \cup \beta \quad \text{(nondet. choice)}
$$

$$
\alpha^* \quad \text{(nondet. repetition)}
$$

\begin{align*}
DCCS & \equiv (ctrl; drive)^* \\
ctrl & \equiv \forall i : C \ a(i) := \text{if} \forall j : C \ \text{far}(i,j) \ \text{then} A \ \text{else} b \\
drive & \equiv \forall i : C \ x(i)'' = a(i)
\end{align*}
## Quantified Differential Dynamic Logic $\mathcal{QdL}$: Syntax

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

\[
\forall i : C \ x(s)' = \theta \\
\forall i : C \ x(s) := \theta \\
?Q \\
\alpha ; \beta \\
\alpha \cup \beta \\
\alpha^* \\
\]

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

### Kleene algebra

\[DCCS \equiv (\text{appear}; \text{ctrl}; \text{drive})^*\]

- **appear**: $n := \text{new } C; \ ?(\forall j : C \ \text{far}(j, n))$
- **ctrl**: $\forall i : C \ a(i) := \text{if } \forall j : C \ \text{far}(i, j) \text{ then } A \ \text{else } -b$
- **drive**: $\forall i : C \ x(i)'' = a(i)$

---

André Platzer (KIT || CMU)
Quantified Differential Dynamic Logic $\mathcal{QdL} :$ Syntax

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

\[
\forall i : C \quad x(s)' = \theta \quad \text{(quantified ODE)}
\]

\[
\forall i : C \quad x(s) := \theta \quad \text{(quantified assignment)}
\]

\[
?Q \quad \text{(conditional execution)}
\]

\[
\alpha; \beta \quad \text{(seq. composition)}
\]

\[
\alpha \cup \beta \quad \text{(nondet. choice)}
\]

\[
\alpha^* \quad \text{(nondet. repetition)}
\]

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

\[
\text{Kleene algebra} \quad \{ \}
\]

\[
DCCS \equiv (appear; ctrl; drive)^*
\]

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

**ctrl** \equiv \( \forall i : C \ a(i) := \text{if } \forall j : C \ far(i, j) \text{ then } A \text{ else } \neg b \)

**drive** \equiv \( \forall i : C \ x(i)'' = a(i) \)

\new C \text{ is definable!}
Quantified Differential Dynamic Logic $\mathcal{QdL}$: Syntax

Definition ($\mathcal{QdL}$ Formula $\phi$)

$\neg, \land, \lor, \to, \forall x, \exists x, =, \geq, +, \cdot$  \hspace{1cm} ($\mathbb{R}$-first-order part)

$[\alpha] \phi, \langle \alpha \rangle \phi$  \hspace{1cm} (dynamic part)

$[(\text{appear}; \text{ctrl}; \text{drive})^*] \forall i \neq j : C x(i) \neq x(j)$
Quantified Differential Dynamic Logic $\mathcal{QdL}$: Syntax

**Definition (Qd$L$ Formula $\phi$)**

$\neg, \land, \lor, \Rightarrow, \forall x, \exists x, =, \geq, +, \cdot$  
($\mathbb{R}$-first-order part)

$[\alpha] \phi, \langle \alpha \rangle \phi$  
(dynamic part)

$\forall i, j : C$ far$(i, j) \rightarrow [(appear; ctrl; drive)^*] \forall i \neq j : C$ $x(i) \neq x(j)$
**Definition (QdŁ Formula \( \phi \))**

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

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

\[
\forall i, j : C \ far(i, j) \rightarrow [(\text{appear}; \ ctrl; \ drive)^*] \ \forall i \neq j : C \ x(i) \neq x(j)
\]

\[
\text{far}(i, j) \equiv i \neq j \rightarrow x(i) < x(j) \wedge v(i) \leq v(j) \wedge a(i) \leq a(j)
\]

\[
\vee x(i) > x(j) \wedge v(i) \geq v(j) \wedge a(i) \geq a(j) \ldots
\]
Definition (Quantified hybrid program $\alpha$: transition semantics)

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

\[ v \xrightarrow{} w \]

if \( w(x)(v^e_i[s]) = v^e_i[\theta] \) (for all $e$) and otherwise unchanged
Definition (Quantified hybrid program $\alpha$: transition semantics)

- $\forall i: C \ x(s)' = \theta$
- $\forall i \ x(s)' = \theta$

$$\frac{d \varphi(t)^e[x(s)]}{dt}(\zeta) = \varphi(\zeta)^e[\theta] \quad \text{(for all } e)$$
Definition (Quantified hybrid program $\alpha$: transition semantics)

$$\alpha;\beta$$

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

$\alpha; \beta$

$\alpha$

$v \rightarrow s \rightarrow w$

$x$

$\downarrow$

$X$

$s$

$w$

$\uparrow$

$X$

$t$

$\uparrow$

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

$\alpha^*$

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

- If $v \models Q$, there is no change.
- Otherwise, no transition.
Definition (Quantified hybrid program $\alpha$: transition semantics)

$v$ if $v \not\models Q$

no change if $v \models Q$

otherwise no transition
Definition (QdŁ Formula $\phi$)

[*$\alpha$]* $\phi$ $\Rightarrow$ $\phi$

André Platzer (KIT ∥ CMU)

Logic of Distributed Hybrid Systems
Definition (QdL Formula $\phi$)

$$v \langle \alpha \rangle \phi$$

$\langle \alpha \rangle \phi$ - span

$\beta$-span

Compositional semantics $\Rightarrow$ Compositional calculus!
Definition (QdŁ Formula $\phi$)

$[\alpha]\phi$  

$\alpha$-span
Definition (QdŁ Formula $\phi$)

$[\alpha] \phi$ as compositional semantics

$\langle \beta \rangle \phi$ as $\beta$-span

$\alpha$-span as $\alpha$-span

$\beta$-span as $\beta$-span

$\langle \beta \rangle \phi$ as $\beta$-span

$[\alpha] \phi$ as compositional semantics
Definition (QdL Formula $\phi$)

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

$$
[\alpha] \phi
$$

$$
\langle \beta \rangle \phi
$$

$\beta$-span

$[\alpha] \phi$

$\alpha$-span

Compositional semantics $\Rightarrow$ Compositional calculus!
Definition (QdL Formula $\phi$)

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
   - Deduction Modulo with Free Variables & Skolemization
   - Actual Existence and Creation
   - Soundness and Completeness
   - Quantified Differential Invariants
4. Applications
   - Distributed Car Control
   - Surgical Robot
5. Conclusions
\[
\overline{\phi([\forall i x(i) := \theta]x(u))}
\]
\[
\forall i (i = u \rightarrow \phi(\theta)) \\
\phi([\forall i x(i) := \theta] x(u))
\]
∀i (i = [∀i x(i) := θ]u → φ(θ))

\[\phi([∀i x(i) := θ]x(u))\]
\[
\forall i (i = [\forall i x(i) := \theta]u \rightarrow \phi(\theta)) \quad \Rightarrow \quad \phi([\forall i x(i) := \theta]x(u))
\]

\[
\phi([\forall i x(s) := \theta]x(u))
\]
\[
\forall i (i = [\forall i x(i) := \theta] u \rightarrow \phi(\theta))
\frac{\phi([\forall i x(i) := \theta] x(u))}{\text{if } \exists i s = u \text{ then } \forall i (s = u \rightarrow \phi(\theta)) \text{ else } \phi(x(u))}
\frac{\phi([\forall i x(s) := \theta] x(u))}{\phi(\forall i x(s) := \theta)}
\]
\[ \forall i (i = [\forall i x(i) := \theta]u \rightarrow \phi(\theta)) \]

\[ \phi([\forall i x(i) := \theta]x(u)) \]

\[ \text{if } \exists i \ s = u \text{ then } \forall i (s = u \rightarrow \phi(\theta)) \text{ else } \phi(x(u)) \]

\[ \phi([\forall i x(s) := \theta]x(u)) \]
\[
\forall i (i = [\forall i x(i) := \theta] u \rightarrow \phi(\theta)) \\
\phi([\forall i x(i) := \theta] x(u)) \\
\text{if } \exists i \ s = u \text{ then } \forall i (s = u \rightarrow \phi(\theta)) \text{ else } \phi(x(u)) \\
\phi([\forall i x(s) := \theta] x(u))
\]
∀i(\(i = [\forall i x(i) := \theta]u \rightarrow \phi(\theta))\)

\[\phi([\forall i x(i) := \theta]x(u))\]

if \(\exists i \, s = u\) then \(\forall i \, (s = u \rightarrow \phi(\theta))\) else \(\phi(x(u))\)

\[\phi([\forall i x(s) := \theta]x(u))\]
∀i (i = [∀ix(i) := θ] u → φ(θ))

\[ \phi([∀ix(i) := θ] x(u)) \]

if \( ∃i s = [A] u \) then ∀i (s = [A] u → φ(θ)) else φ(x([A] u))

\[ \phi([∀ix(s) := θ] x(u)) \]

ϕ
\[ \forall i (i = [\forall i x(i) := \theta]u \rightarrow \phi(\theta)) \]

\[ \phi([\forall i x(i) := \theta]x(u)) \]

\[ \text{if } \exists i s = [\mathcal{A}]u \text{ then } \forall i (s = [\mathcal{A}]u \rightarrow \phi(\theta)) \text{ else } \phi(x([\mathcal{A}]u)) \]

\[ \phi([\forall i x(s) := \theta]x(u)) \]

\[ \forall t \geq 0 [\forall i x(i) := x_i(t)] \phi \]

\[ [\forall i x(i)' = \theta] \phi \]
∀i (i = [∀ix(i) := θ]u → φ(θ))

φ([∀ix(i) := θ]x(u))

if ∃i s = [A]u then ∀i (s = [A]u → φ(θ)) else φ(x([A]u))

φ([∀ix(s) := θ]x(u))

∀t ≥ 0 [∀ix(i) := x_i(t)] φ

[∀ix(i)' = θ] φ
compositional semantics $\Rightarrow$ 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

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

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

\[
\phi \quad (\phi \rightarrow [\alpha] \phi)
\]

\[
\frac{\phi}{[\alpha^*] \phi}
\]
\[ \forall i \neq j \; x(i) \neq x(j) \rightarrow [\forall i \; x(i)'' = -b] \forall j \neq k \; x(j) \neq x(k) \]
\[ \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)' = v(i), v(i)' = -b] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)'' = -b] \forall j \neq k x(j) \neq x(k) \]
\[
\forall i \neq j \ x(i) \neq x(j) \rightarrow \forall t \geq 0 \left[ \forall i \ x(i) := -\frac{b}{2} t^2 + v(i) t + x(i) \right] \forall j \neq k \ x(j) \neq x(k)
\]

\[
\forall i \neq j \ x(i) \neq x(j) \rightarrow \left[ \forall i \ x(i)' = v(i), \ v(i)' = -b \right] \forall j \neq k \ x(j) \neq x(k)
\]

\[
\forall i \neq j \ x(i) \neq x(j) \rightarrow \left[ \forall i \ x(i)''' = -b \right] \forall j \neq k \ x(j) \neq x(k)
\]
∀i ≠ j x(i) ≠ x(j) → s ≥ 0 → [∀ i x(i) := −b/2 s^2 + v(i) s + x(i)] ∀ j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → ∀ t ≥ 0 [∀ i x(i) := −b/2 t^2 + v(i) t + x(i)] ∀ j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀ i x(i)' = v(i), v(i)' = −b] ∀ j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀ i x(i)'' = −b] ∀ j ≠ k x(j) ≠ x(k)
∀i ≠ j \, x(i) ≠ x(j), s ≥ 0 \rightarrow [\forall i \, x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k \, x(j) \neq x(k)

∀i ≠ j \, x(i) \neq x(j) \rightarrow s ≥ 0 \rightarrow [\forall i \, x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k \, x(j) \neq x(k)

∀i ≠ j \, x(i) \neq x(j) \rightarrow \forall t ≥ 0 [\forall i \, x(i) := -\frac{b}{2} t^2 + v(i) t + x(i)] \forall j \neq k \, x(j) \neq x(k)

∀i ≠ j \, x(i) \neq x(j) \rightarrow [\forall i \, x(i)' = v(i), v(i)' = -b] \forall j \neq k \, x(j) \neq x(k)

∀i ≠ j \, x(i) \neq x(j) \rightarrow [\forall i \, x(i)'' = -b] \forall j \neq k \, x(j) \neq x(k)
\[
\forall i \neq j x(i) \neq x(j), s \geq 0 \rightarrow \forall j \neq k \left( -\frac{b}{2} s^2 + v(j) s + x(j) \neq -\frac{b}{2} s^2 + v(k) s + x(k) \right)
\]

\[
\forall i \neq j x(i) \neq x(j), s \geq 0 \rightarrow \left[ \forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i) \right] \forall j \neq k x(j) \neq x(k)
\]

\[
\forall i \neq j x(i) \neq x(j) \rightarrow s \geq 0 \rightarrow \left[ \forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i) \right] \forall j \neq k x(j) \neq x(k)
\]

\[
\forall i \neq j x(i) \neq x(j) \rightarrow \forall t \geq 0 \left[ \forall i x(i) := -\frac{b}{2} t^2 + v(i) t + x(i) \right] \forall j \neq k x(j) \neq x(k)
\]

\[
\forall i \neq j x(i) \neq x(j) \rightarrow \left[ \forall i x(i)' = v(i), v(i)' = -b \right] \forall j \neq k x(j) \neq x(k)
\]

\[
\forall i \neq j x(i) \neq x(j) \rightarrow \left[ \forall i x(i)'' = -b \right] \forall j \neq k x(j) \neq x(k)
\]
\( \forall i \neq j \ x(i) \neq x(j) \rightarrow \forall j \neq k \quad \forall s \geq 0 (-\frac{b}{2} s^2 + v(j) s + x(j) \neq -\frac{b}{2} s^2 + v(k) s + x(k)) \)

\( \forall i \neq j \ x(i) \neq x(j), s \geq 0 \rightarrow \forall j \neq k \ ( -\frac{b}{2} s^2 + v(j) s + x(j) \neq -\frac{b}{2} s^2 + v(k) s + x(k)) \)

\( \forall i \neq j \ x(i) \neq x(j), s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \)

\( \forall i \neq j \ x(i) \neq x(j) \rightarrow s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \)

\( \forall i \neq j x(i) \neq x(j) \rightarrow \forall t \geq 0 [\forall i x(i) := -\frac{b}{2} t^2 + v(i) t + x(i)] \forall j \neq k x(j) \neq x(k) \)

\( \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)' = v(i), v(i)' = -b] \forall j \neq k x(j) \neq x(k) \)

\( \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)'' = -b] \forall j \neq k x(j) \neq x(k) \)
∀i ≠ j x(i) ≠ x(j) → ∀j ≠ k QE ∀s ≥ 0 (−\frac{b}{2}s^2 + v(j) s + x(j) ≠ −\frac{b}{2}s^2 + v(k) s + x(k))

∀i ≠ j x(i) ≠ x(j), s ≥ 0 → ∃j ≠ k (−\frac{b}{2}s^2 + v(j) s + x(j) ≠ −\frac{b}{2}s^2 + v(k) s + x(k))

∀i ≠ j x(i) ≠ x(j), s ≥ 0 → [∀i x(i) := −\frac{b}{2}s^2 + v(i) s + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → s ≥ 0 → [∀i x(i) := −\frac{b}{2}s^2 + v(i) s + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → ∀t ≥ 0 [∀i x(i) := −\frac{b}{2}t^2 + v(i) t + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀i x(i)' = v(i), v(i)' = −b] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀i x(i)'' = −b] ∀j ≠ k x(j) ≠ x(k)
\[ \forall i \neq j \ x(i) \neq x(j) \rightarrow \forall j \neq k \ (x(j) \leq x(k) \land v(j) \leq v(k) \lor x(j) \geq x(k) \land v(j) \geq v(k)) \]

\[ \forall i \neq j \ x(i) \neq x(j), s \geq 0 \rightarrow \forall j \neq k \ (-\frac{b}{2} s^2 + v(j) s + x(j) \neq -\frac{b}{2} s^2 + v(k) s + x(k)) \]

\[ \forall i \neq j \ x(i) \neq x(j), s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j \ x(i) \neq x(j) \rightarrow s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j \ x(i) \neq x(j) \rightarrow \forall t \geq 0 [\forall i x(i) := -\frac{b}{2} t^2 + v(i) t + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j \ x(i) \neq x(j) \rightarrow [\forall i x(i)' = v(i), v(i)' = -b] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j \ x(i) \neq x(j) \rightarrow [\forall i x(i)'' = -b] \forall j \neq k x(j) \neq x(k) \]
\[ \forall X, Y, V, W (X \neq Y \rightarrow X \leq Y \land V \leq W \lor X \geq Y \land V \geq W) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow \forall j \neq k (x(j) \leq x(k) \land v(j) \leq v(k) \lor x(j) \geq x(k) \land v(j) \geq v(k)) \]

\[ \forall i \neq j x(i) \neq x(j), s \geq 0 \rightarrow \forall j \neq k (-\frac{b}{2} s^2 + v(j) s + x(j) \neq -\frac{b}{2} s^2 + v(k) s + x(k)) \]

\[ \forall i \neq j x(i) \neq x(j), s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow s \geq 0 \rightarrow [\forall i x(i) := -\frac{b}{2} s^2 + v(i) s + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow t \geq 0 [\forall i x(i) := -\frac{b}{2} t^2 + v(i) t + x(i)] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)' = v(i), v(i)' = -b] \forall j \neq k x(j) \neq x(k) \]

\[ \forall i \neq j x(i) \neq x(j) \rightarrow [\forall i x(i)'' = -b] \forall j \neq k x(j) \neq x(k) \]
∀X, Y, V, W (X ≠ Y → X ≤ Y ∧ V ≤ W ∨ X ≥ Y ∧ V ≥ W)

∀i ≠ j x(i) ≠ x(j) → ∀j ≠ k (x(j) ≤ x(k) ∧ v(j) ≤ v(k) ∨ x(j) ≥ x(k) ∧ v(j) ≥ v(k))

∀i ≠ j x(i) ≠ x(j), s ≥ 0 → ∀j ≠ k (−\(\frac{b}{2} s^2\) + v(j)s + x(j) ≠ −\(\frac{b}{2} s^2\) + v(k)s + x(k))

∀i ≠ j x(i) ≠ x(j), s ≥ 0 → [∀i x(i) := −\(\frac{b}{2} s^2\) + v(i)s + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → s ≥ 0 → [∀i x(i) := −\(\frac{b}{2} s^2\) + v(i)s + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → ∀t ≥ 0 [∀i x(i) := −\(\frac{b}{2} t^2\) + v(i)t + x(i)] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀i x(i)' = v(i), v(i)' = −b] ∀j ≠ k x(j) ≠ x(k)

∀i ≠ j x(i) ≠ x(j) → [∀i x(i)'' = −b] ∀j ≠ k x(j) ≠ x(k)
Actual Existence and Creation

Actual Existence Function $E(\cdot)$

$$E(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing object} 
\end{cases}$$
Actual Existence and Creation

Actual Existence Function $E(\cdot)$

$E(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing objects}
\end{cases}$

$[n := \text{new } C] \phi$
Actual Existence Function $E(\cdot)$

$E(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing object}
\end{cases}$

$$\left[ (\forall j: C \ n := j) ; \right] \phi$$

$$\left[ n := \text{new } C \right] \phi$$
Actual Existence Function $\mathbb{E}(\cdot)$

$$\mathbb{E}(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing objects}
\end{cases}$$

$$\frac{[(\forall j : C \ n := j); \ ?(\mathbb{E}(n) = 0); \ ] \phi}{[n := \text{new } C] \phi}$$
Actual Existence Function $E(\cdot)$

$E(i) = \begin{cases} 0 & \text{if } i \text{ denotes a possible object} \\ 1 & \text{if } i \text{ denotes an actively existing objects} \end{cases}$

\[
[\forall j : C \ n := j); \ ?(E(n) = 0); E(n) := 1] \phi
\]

\[
[n := \text{new } C] \phi
\]
Actual Existence and Creation

Actual Existence Function $E(\cdot)$

$E(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing objects}
\end{cases}$

$\left[ (\forall j: C \ n := j); \ ?(E(n) = 0); \ E(n) := 1 \right] \phi$

$\left[ n := \text{new } C \right] \phi$

$\forall i: C! \phi \equiv$

$\forall i: C! f(s) := \theta \equiv$

$\forall i: C! f(s)' = \theta \equiv$

[Diagram of distributed hybrid systems with cars and arrows indicating interactions and changes in states.]
Actual Existence Function $E(\cdot)$

$$E(i) = \begin{cases} 
0 & \text{if } i \text{ denotes a possible object} \\
1 & \text{if } i \text{ denotes an actively existing objects}
\end{cases}$$

$$[(\forall j : C \ n := j); \ ?(E(n) = 0); \ E(n) := 1] \phi$$

$$[n := \text{new } C] \phi$$

$$\forall i : C! \phi \equiv \forall i : C \ (E(i) = 1 \rightarrow \phi)$$

$$\forall i : C! \ f(s) := \theta \equiv \forall i : C \ f(s) := (\text{if } E(i) = 1 \text{ then } \theta \text{ else } f(s))$$

$$\forall i : C! \ f(s)' = \theta \equiv \forall i : C \ f(s)' = E(i) \theta$$
Theorem (Relative Completeness) (LMCS’12)

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

Proof 16p.
Soundness and Completeness

Theorem (Relative Completeness) (LMCS’12)

\[ L_c \] 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 (Quantified Differential Invariant) \(^{(\text{HSCC}'11)}\)

\[
(QdI) \quad \frac{Q \Rightarrow [\forall i : C \ f(i)' := \theta]F'}{F \Rightarrow [\forall i : C \ f(i)' = \theta & Q]F}
\]

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

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

\[ \forall i : C \; x(i)' := x(i)^2 + x(i)^4 + 2 \] \[ \forall i : C \; 2x(i)^3 \geq 0 \] \[ \forall i : C \; 2x(i)^3 \geq 1 \rightarrow [\forall i : C \; x(i)' = x(i)^2 + x(i)^4 + 2] \forall i : C \; 2x(i)^3 \geq 1 \]
A Simple Proof with Quantified Differential Invariants

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

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

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

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

[∀i: C x(i)' := x(i)^2 + x(i)^4 + 2]∀i: C 6x(i)^2x(i)' ≥ 0

[∀i: C x(i)' := x(i)^2 + x(i)^4 + 2]∀i: C (2x(i)^3)' ≥ 0

[∀i: C x(i)' := x(i)^2 + x(i)^4 + 2](∀i: C 2x(i)^3 ≥ 0)'

∀i: C 2x(i)^3 ≥ 1 → [∀i: C x(i)' = x(i)^2 + x(i)^4 + 2]∀i: C 2x(i)^3 ≥ 1
true

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

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

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

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

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

1. Motivation

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

3. Proof Calculus for Distributed Hybrid Systems
   - Compositional Verification Calculus
   - Deduction Modulo with Free Variables & Skolemization
   - Actual Existence and Creation
   - Soundness and Completeness
   - Quantified Differential Invariants

4. Applications
   - Distributed Car Control
   - Surgical Robot

5. Conclusions
Driver’s License Test for Robotic Cars?
Driver’s License Test for Robotic Cars?

André Platzer (KIT || CMU)
Driver’s License Test for Robotic Cars? Proof!

André Platzer (KIT ∥ CMU)
Logic of Distributed Hybrid Systems
A car controller for a differential equation respects separation of local lane.
Car Control: Local Lane Control Challenge

**Challenge: Local lane dynamics**

- A car controller for a differential equation respects separation of local lane.
- Follower car maintains safe distance to leader:

\[
\begin{align*}
&f \ll \ell \\
&f \ll \ell \equiv (x_f \leq x_{\ell}) \land (f \neq \ell) \\
&\rightarrow (x_{\ell} > x_f + v_f^2 f^2 b - v_{\ell}^2 \ell^2 B \land x_{\ell} > x_f \land v_f \geq 0 \land v_{\ell} \geq 0)
\end{align*}
\]

André Platzer (KIT || CMU)
Challenge: Local Lane Dynamics

- A car controller for a differential equation respects separation of local lane.
- Follower car maintains safe distance to leader:
  \[ f \ll \ell \rightarrow [(a_i := ctrl; \ x_i'' = a_i)^*] f \ll \ell \]
Challenge: Local lane dynamics

- A car controller for a differential equation respects separation of local lane.
- Follower car maintains safe distance to leader:

\[ f \ll \ell \rightarrow [(a_i := \text{ctrl}; x_i'' = a_i)^*] f \ll \ell \]

\[ f \ll \ell \equiv (x_f \leq x_\ell) \land (f \neq \ell) \rightarrow \]

\[ (x_\ell > x_f + \frac{v_f^2}{2b} - \frac{v_\ell^2}{2B} \land x_\ell > x_f \land v_f \geq 0 \land v_\ell \geq 0) \]
Challenge: Global lane dynamics

- All controllers for arbitrarily many differential equations respect separation globally on lane.
Challenge: Global lane dynamics

- All controllers for arbitrarily many differential equations respect separation globally on lane.
- Each car safe behind all others
Challenge: Global lane dynamics

- All controllers for arbitrarily many differential equations respect separation globally on lane.
- Each car safe behind all others

\[
[(\forall i \ a(i) := \text{ctrl}; \ \forall i \ x(i)'' = a(i))^*] \forall i, j \ i \ll j
\]
Challenge: Local highway dynamics

- All controllers for arbitrarily many differential equations respect separation locally on highway.
Challenge: Local highway dynamics

- All controllers for arbitrarily many differential equations respect separation locally on highway.
- For each lane: all controllers for the differential equations respect separation even if cars appear or disappear.
All controllers for arbitrarily many differential equations respect separation locally on highway.

For each lane: all controllers for the differential equations respect separation even if cars appear or disappear.

Each car safe behind all others, even if new cars appear or disappear.
Challenge: Local highway dynamics

- All controllers for arbitrarily many differential equations respect separation locally on highway.
- For each lane: all controllers for the differential equations respect separation even if cars appear or disappear.
- Each car safe behind all others, even if new cars appear or disappear.

\[
\left( n := \text{new } C; \ \forall i \ a(i) := \text{ctrl}; \ \forall i \ x(i)'' = a(i)\right)^* \ \forall i, j \ i \ll j
\]
All controllers for arbitrarily many differential equations respect separation globally on highway.
Car Control: Global Highway Control Challenge

Challenge: Global highway dynamics

- All controllers for arbitrarily many differential equations respect separation globally on highway.
- All controllers for the differential equations respect separation even if cars switch lanes.
All controllers for arbitrarily many differential equations respect separation globally on highway.

All controllers for the differential equations respect separation even if cars switch lanes.

On all lanes, all car safe behind all others on their lanes, even if cars switch lanes.
Car Control: Global Highway Control Challenge

Challenge: Global highway dynamics

- All controllers for arbitrarily many differential equations respect separation globally on highway.
- All controllers for the differential equations respect separation even if cars switch lanes.
- On all lanes, all car safe behind all others on their lanes, even if cars switch lanes.

\[
\forall l \left( n := \text{new} \ C; \ \forall i \ a(i) := \text{ctrl}; \ \forall i \ x(i)'' = a(i)^* \right) \forall l \forall i, j \ i \ll j
\]
Surgical Robot Verification: Skull-base Surgery

Virtual fixture boundary

Time
Continuous Control
Robot responds infinitely fast
Time
Robot responds within epsilon
\( \varepsilon \)-Control

Negligible lag?

✓ Admittance controller converting force to velocity (continuous)

× Virtual fixture control algorithm (hybrid)

Safety: speed and space

Other problems?

Redesign to predictive control

HSCC’13
Outline

1 Motivation

2 Quantified Differential Dynamic Logic $\mathcal{QdL}$
   • Design
   • Syntax
   • Semantics

3 Proof Calculus for Distributed Hybrid Systems
   • Compositional Verification Calculus
   • Deduction Modulo with Free Variables & Skolemization
   • Actual Existence and Creation
   • Soundness and Completeness
   • Quantified Differential Invariants

4 Applications
   • Distributed Car Control
   • Surgical Robot

5 Conclusions
Conclusions

quantified differential dynamic logic

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

- Distributed hybrid systems everywhere
- System model and semantics
- Logic for distributed hybrid systems
- Compositional proof calculus
- First verification approach
- Sound & complete / diff. eqn.
- Quantified differential invariants
- Distributed car control verified
- Distributed aircraft control verified
- Robot verified for many obstacles
Conclusions

quantified differential dynamic logic

$$QdL = FOL + DL + QHP$$

- Distributed hybrid systems everywhere
- System model and semantics
- Logic for distributed hybrid systems
- Compositional proof calculus
- First verification approach
- Sound & complete / diff. eqn.
- Quantified differential invariants
- Distributed car control verified
- Distributed aircraft control verified
- Robot verified for many obstacles
I Part: Elementary Cyber-Physical Systems
2. Differential Equations & Domains
3. Choice & Control
4. Safety & Contracts
5. Dynamical Systems & Dynamic Axioms
6. Truth & Proof
7. Control Loops & Invariants
8. Events & Responses
9. Reactions & Delays

II Part: Differential Equations Analysis
10. Differential Equations & Differential Invariants
11. Differential Equations & Proofs
12. Ghosts & Differential Ghosts
13. Differential Invariants & Proof Theory

III Part: Adversarial Cyber-Physical Systems
14-17. Hybrid Systems & Hybrid Games

IV Part: Comprehensive CPS Correctness
André Platzer.
A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems.
Special issue for selected papers from CSL’10.

André Platzer.
Quantified differential dynamic logic for distributed hybrid systems.

André Platzer.
Quantified differential invariants.

Akash Deshpande, Aleks Göllü, and Pravin Varaiya.
SHIFT: A formalism and a programming language for dynamic networks of hybrid automata.


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

Jan A. Bergstra and C. A. Middelburg.  
Process algebra for hybrid systems.  

José Meseguer and Raman Sharykin.  
Specification and analysis of distributed object-based stochastic hybrid systems.  
In Hespanha and Tiwari [18], pages 460–475.

Yanni Kouskoulas, David W. Renshaw, André Platzer, and Peter Kazanzides.  
Certifying the safe design of a virtual fixture control algorithm for a surgical robot.  
In Belta and Ivancic [17], pages 263–272.


Calin Belta and Franjo Ivancic, editors.

João P. Hespanha and Ashish Tiwari, editors.