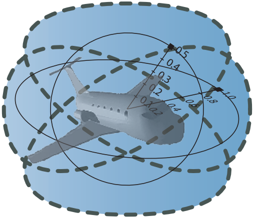
# Logical Foundations of Cyber-Physical Systems

André Platzer

aplatzer@cs.cmu.edu
Logical Systems Lab
Computer Science Department
Carnegie Mellon University, Pittsburgh, PA

http://symbolaris.com/



# Outline

## Can you trust a computer to control physics?

# Can you trust a computer to control physics?

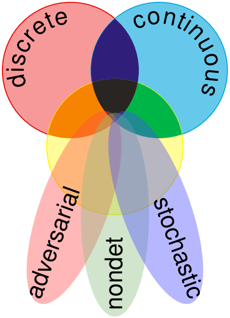
## Rationale

1. Safety guarantees require analytic foundations
2. Foundations revolutionized digital computer science & society
3. Need even stronger foundations when software reaches out into our physical world

# CPS are Multi-Dynamical Systems

## CPS Dynamics Bee

CPS are characterized by multiple facets of dynamical systems.




## CPS Compositions

CPS combine many simple dynamical effects.

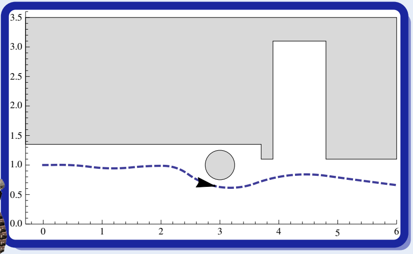
## Tame Parts

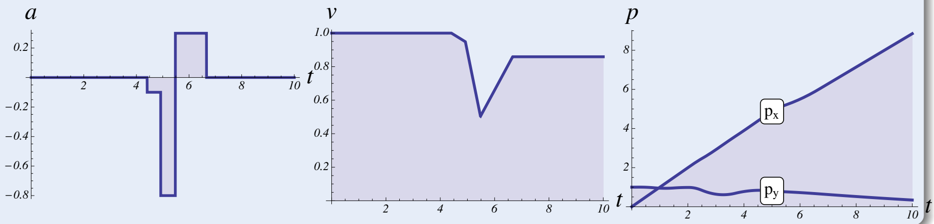
Exploiting compositionality tames complexity.

## Challenge (Hybrid Systems)

Fixed rule describing state evolution with both

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)

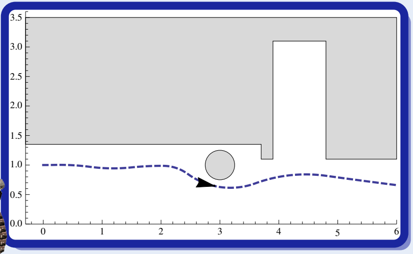



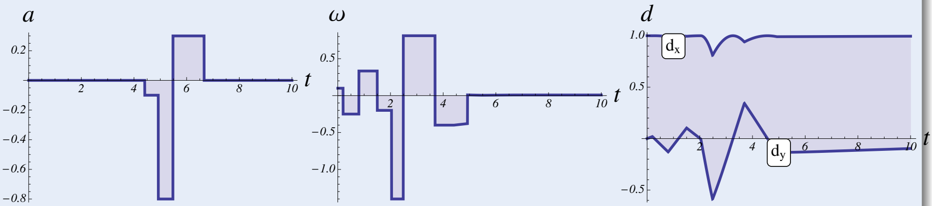



## Challenge (Hybrid Systems)

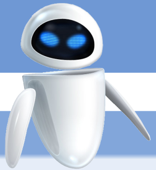
Fixed rule describing state evolution with both

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)

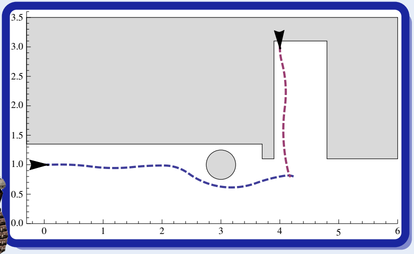
# CPS Analysis

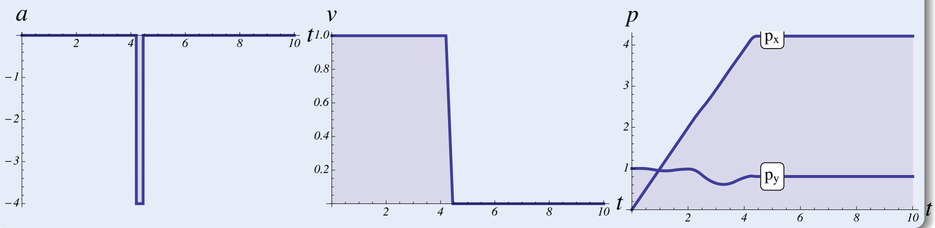


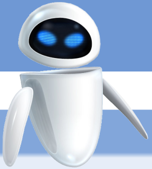
## Challenge (Hybrid Systems)

Fixed rule describing state evolution with both

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)



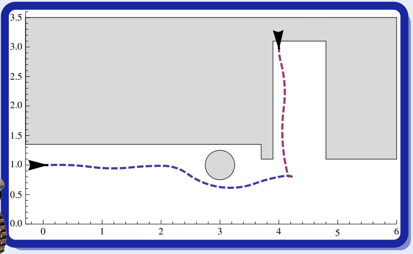



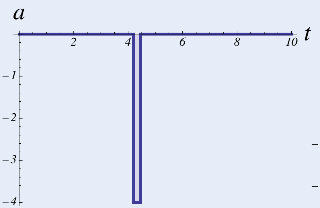


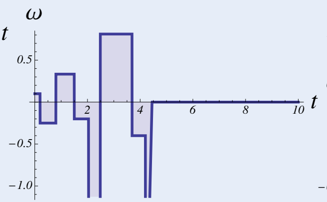
## Challenge (Hybrid Systems)

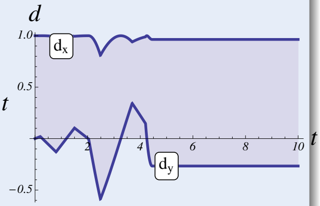
Fixed rule describing state evolution with both

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)

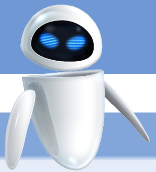


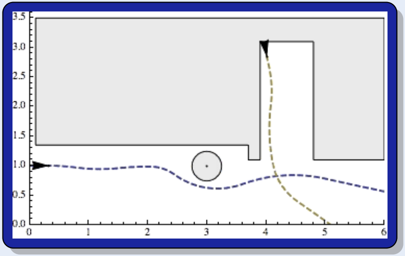
# CPS Analysis: Other Agents

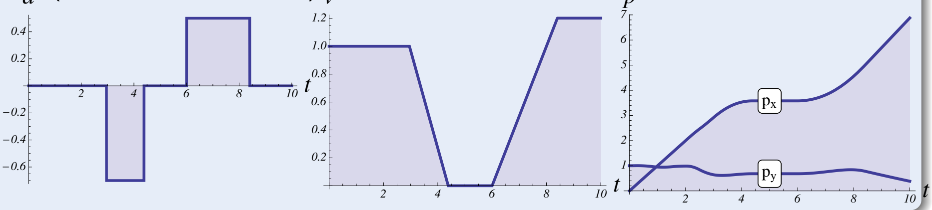


## Challenge (Hybrid Games)

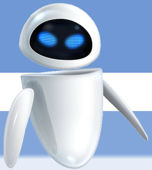
Game rules describing play evolution with

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
- Adversarial dynamics (Angel ◇ vs. Demon ◻)





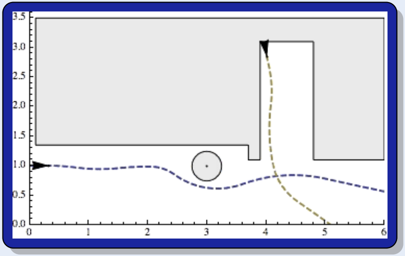
# CPS Analysis: Other Agents

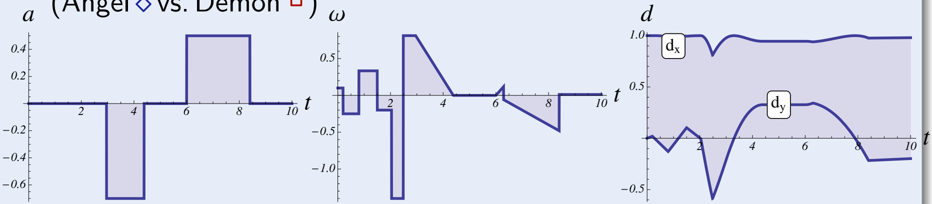


## Challenge (Hybrid Games)

Game rules describing play evolution with

- Discrete dynamics (control decisions)
- Continuous dynamics (differential equations)
- Adversarial dynamics (Angel ◇ vs. Demon □)

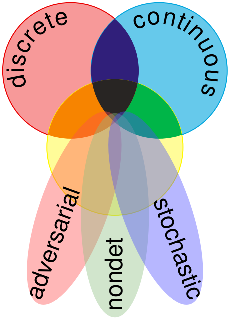


# CPS are Multi-Dynamical Systems

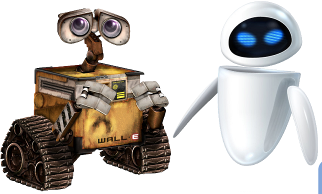
**hybrid systems**

HS = discrete + ODE



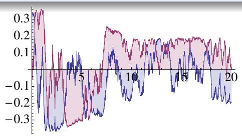

**hybrid games**

HG = HS + adversary



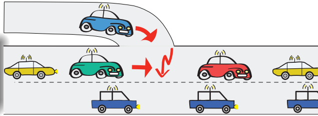
**stochastic hybrid sys.**

SHS = HS + stochastics

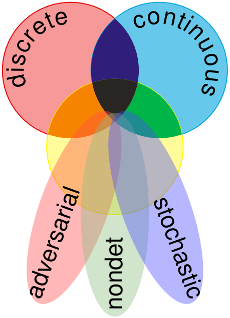



**distributed hybrid sys.**

DHS = HS + distributed

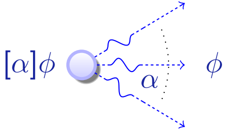


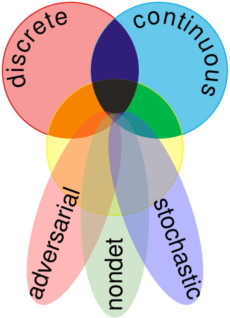
# Family of Differential Dynamic Logics



**differential dynamic logic**

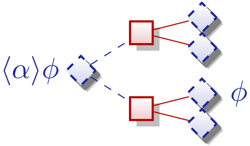
$d\mathcal{L} = \mathsf{DL} + \mathsf{HP}$



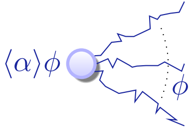

**differential game logic**

$dG\mathcal{L} = \mathsf{GL} + \mathsf{HG}$




**stochastic differential DL**

$Sd\mathcal{L} = \mathsf{DL} + \mathsf{SHP}$




**quantified differential DL**

$Qd\mathcal{L} = \mathsf{FOL} + \mathsf{DL} + \mathsf{QHP}$

# Differential Dynamic Logic d$\mathcal{L}$: Syntax

**Definition (Hybrid program $\alpha$)**

$$x := \theta \mid ?H \mid x' = f(x) \,\&\, H \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$

**Definition (d$\mathcal{L}$ Formula $\phi$)**

$$\theta_1 \geq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \forall x\, \phi \mid \exists x\, \phi \mid [\alpha]\phi \mid \langle\alpha\rangle\phi$$

# Differential Dynamic Logic d$\mathcal{L}$: Syntax

Discrete Assign | Test Condition | Differential Equation | Nondet. Choice | Seq. Compose | Nondet. Repeat

**Definition (Hybrid program $\alpha$)**

$$x := \theta \mid ?H \mid x' = f(x) \,\&\, H \mid \alpha \cup \beta \mid \alpha ; \beta \mid \alpha^*$$

**Definition (d$\mathcal{L}$ Formula $\phi$)**

$$\theta_1 \geq \theta_2 \mid \neg \phi \mid \phi \wedge \psi \mid \forall x\, \phi \mid \exists x\, \phi \mid [\alpha]\phi \mid \langle \alpha \rangle \phi$$

All Reals | Some Reals | All Runs | Some Runs

# Differential Dynamic Logic d$\mathcal{L}$: Semantics

## Definition (Hybrid program $\alpha$)

$$
\begin{aligned}
\rho(x := \theta) &= \{(v, w) \ : \ w = v \text{ except } [\![x]\!]_w = [\![\theta]\!]_v\} \\
\rho(?H) &= \{(v, v) \ : \ v \models H\} \\
\rho(x' = f(x)) &= \{(\varphi(0), \varphi(r)) \ : \ \varphi \models x' = f(x) \text{ for some duration } r\} \\
\rho(\alpha \cup \beta) &= \rho(\alpha) \cup \rho(\beta) \\
\rho(\alpha; \beta) &= \rho(\beta) \circ \rho(\alpha) \\
\rho(\alpha^*) &= \bigcup_{n \in \mathbb{N}} \rho(\alpha^n)
\end{aligned}
$$

## Definition (d$\mathcal{L}$ Formula $\phi$)

$$
\begin{aligned}
&v \models \theta_1 \geq \theta_2 && \text{iff} \quad [\![\theta_1]\!]_v \geq [\![\theta_2]\!]_v \\
&v \models [\alpha]\phi && \text{iff} \quad w \models \phi \text{ for all } w \text{ with } v\rho(\alpha)w \\
&v \models \langle\alpha\rangle\phi && \text{iff} \quad w \models \phi \text{ for some } w \text{ with } v\rho(\alpha)w \\
&v \models \forall x\, \phi && \text{iff} \quad w \models \phi \text{ for all } w \text{ that agree with } v \text{ except for } x \\
&v \models \exists x\, \phi && \text{iff} \quad w \models \phi \text{ for some } w \text{ that agrees with } v \text{ except for } x \\
&v \models \phi \wedge \psi && \text{iff} \quad v \models \phi \text{ and } v \models \psi
\end{aligned}
$$

# Differential Dynamic Logic: Axiomatization

equations of truth

[:=] $[x := \theta]\phi(x) \leftrightarrow \phi(\theta)$

[?] $[?H]\phi \leftrightarrow (H \rightarrow \phi)$

['] $[x' = f(x)]\phi \leftrightarrow \forall t{\geq}0\, [x := y(t)]\phi$ $\quad (y'(t) = f(y))$

[∪] $[\alpha \cup \beta]\phi \leftrightarrow [\alpha]\phi \wedge [\beta]\phi$

[;] $[\alpha; \beta]\phi \leftrightarrow [\alpha][\beta]\phi$

[*] $[\alpha^*]\phi \leftrightarrow \phi \wedge [\alpha][\alpha^*]\phi$

K $[\alpha](\phi \rightarrow \psi) \rightarrow ([\alpha]\phi \rightarrow [\alpha]\psi)$

I $[\alpha^*](\phi \rightarrow [\alpha]\phi) \rightarrow (\phi \rightarrow [\alpha^*]\phi)$

C $[\alpha^*]\forall v{>}0\, (\varphi(v) \rightarrow \langle\alpha\rangle\varphi(v-1)) \rightarrow \forall v\, (\varphi(v) \rightarrow \langle\alpha^*\rangle\exists v{\leq}0\, \varphi(v))$

LICS'12

# Differential Dynamic Logic: Axiomatization

equations of truth

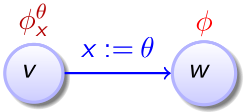
G $\dfrac{\phi}{[\alpha]\phi}$

MP $\dfrac{\phi \to \psi \quad \phi}{\psi}$

$\forall$ $\dfrac{\phi}{\forall x\, \phi}$

LICS'12

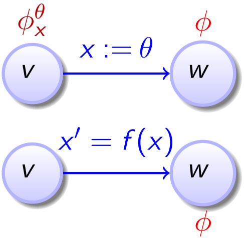
$$\frac{\phi_x^\theta}{[x := \theta]\phi}$$



# Proofs for Hybrid Systems

$$\frac{\phi_x^\theta}{[x := \theta]\phi}$$

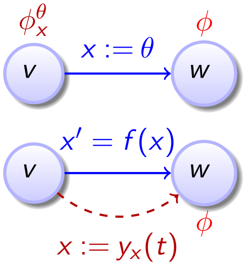
$$\frac{\forall t{\geq}0\, [x := y_x(t)]\phi}{[x' = f(x)]\phi}$$



# Proofs for Hybrid Systems

$$\frac{\phi_x^\theta}{[x := \theta]\phi}$$

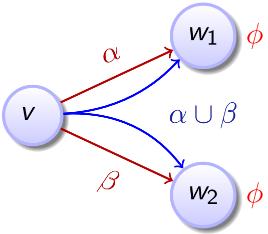
$$\frac{\forall t{\geq}0\, [x := y_x(t)]\phi}{[x' = f(x)]\phi}$$



# Proofs for Hybrid Systems

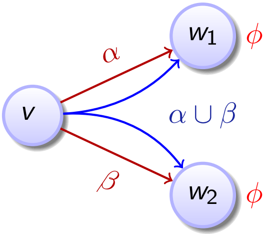
compositional semantics $\Rightarrow$ compositional rules!

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

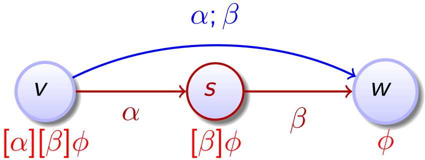


# Proofs for Hybrid Systems

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

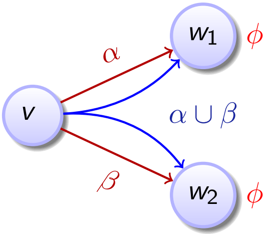



$$\frac{[\alpha][\beta]\phi}{[\alpha; \beta]\phi}$$

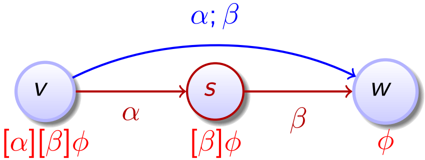


# Proofs for Hybrid Systems

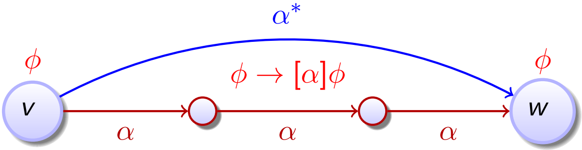
$$\frac{[\alpha]\phi \land [\beta]\phi}{[\alpha \cup \beta]\phi}$$




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




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



# Complete Proof Theory of Hybrid Systems

**Theorem (Sound & Complete)** (J.Autom.Reas. 2008, LICS'12)

d$\mathcal{L}$ *calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations* **or** *discrete dynamics.*

Proof 25pp

**Corollary (Complete Proof-theoretical Alignment & Bridging)**

proving continuous = proving hybrid = proving discrete

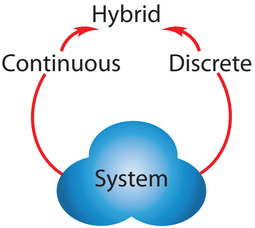
# Complete Proof Theory of Hybrid Systems

**Theorem (Sound & Complete)** (J.Autom.Reas. 2008, LICS'12)

*d$\mathcal{L}$ calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations* **or** *discrete dynamics.* ▸ Proof 25pp

**Corollary (Complete Proof-theoretical Alignment & Bridging)**

proving continuous = proving hybrid = proving discrete



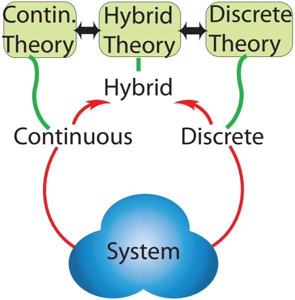
# Complete Proof Theory of Hybrid Systems

**Theorem (Sound & Complete)** (J.Autom.Reas. 2008, LICS'12)

*d$\mathcal{L}$ calculus is a sound & complete axiomatization of hybrid systems relative to either differential equations* **or** *discrete dynamics.* Proof 25pp

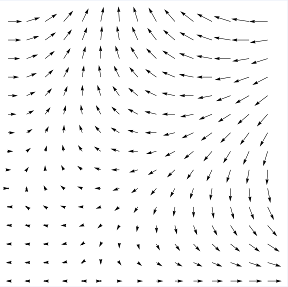
**Corollary (Complete Proof-theoretical Alignment & Bridging)**

proving continuous = proving hybrid = proving discrete

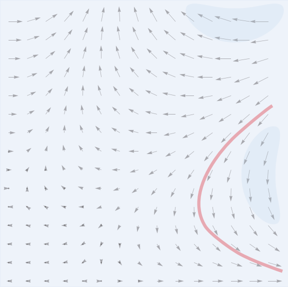


# Differential Invariants for Differential Equations

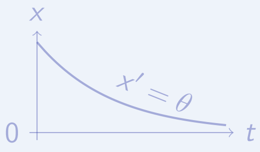
## Differential Invariant

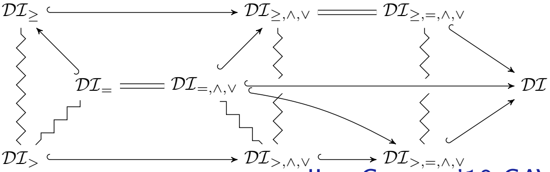


## Differential Cut



## Differential Ghost




**Logic**

Provability study

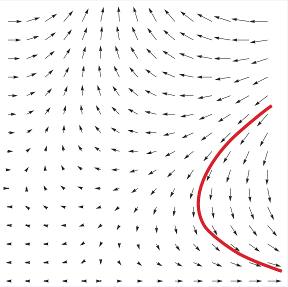
**Math**

Character-istic PDE

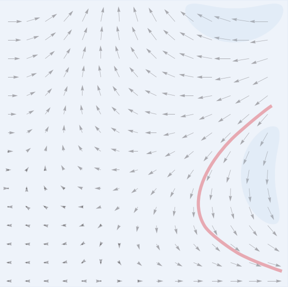
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

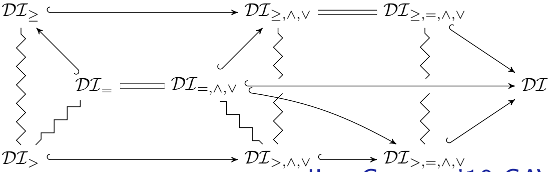
## Differential Invariant



## Differential Cut



## Differential Ghost


## Logic

Provability study

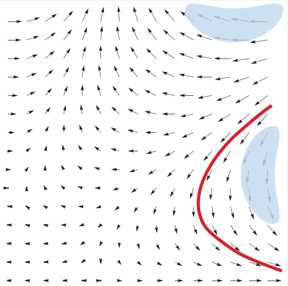
## Math

Character-istic PDE

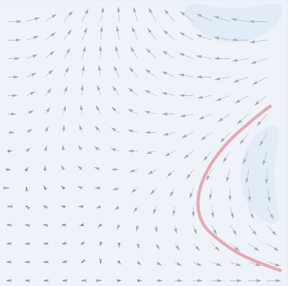
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

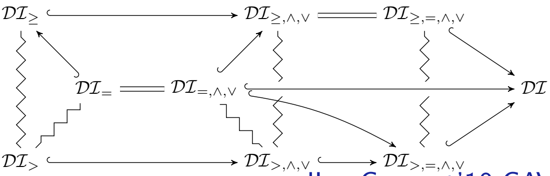
## Differential Invariant



## Differential Cut



## Differential Ghost


## Logic

Provability study

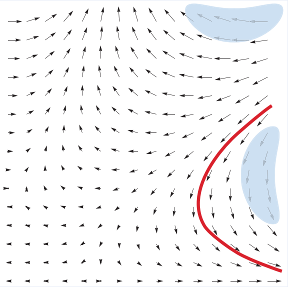
## Math

Character-istic PDE

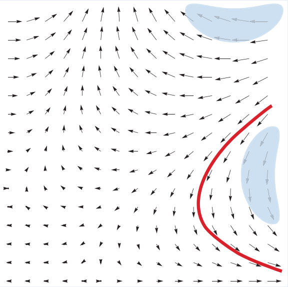
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

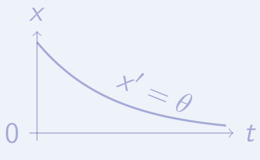
## Differential Invariant

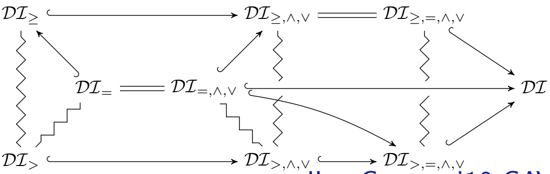


## Differential Cut



## Differential Ghost




## Logic

Provability study

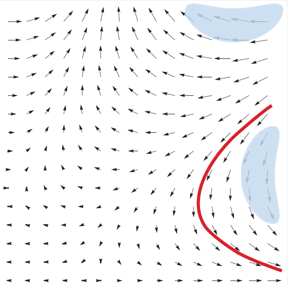
## Math

Character-istic PDE

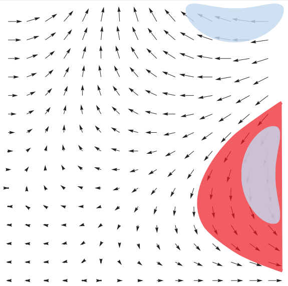
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

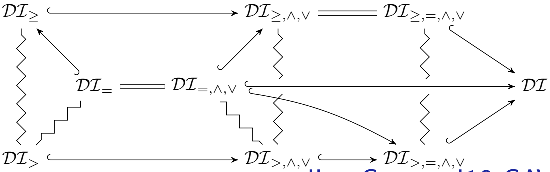
## Differential Invariant



## Differential Cut



## Differential Ghost


**Logic**

Provability study

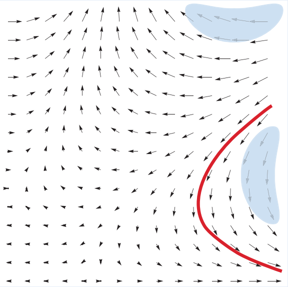
**Math**

Character-istic PDE

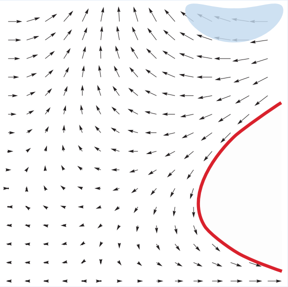
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

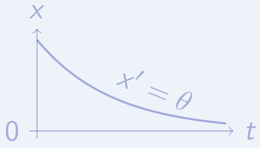
## Differential Invariant

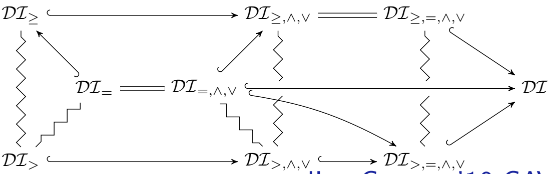


## Differential Cut



## Differential Ghost




## Logic

Provability study

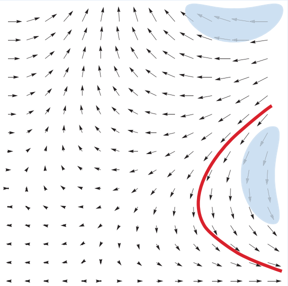
## Math

Character-istic PDE

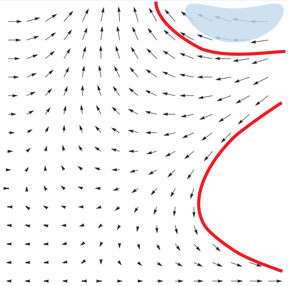
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

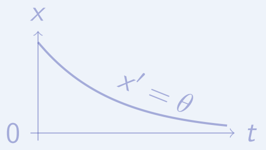
## Differential Invariant

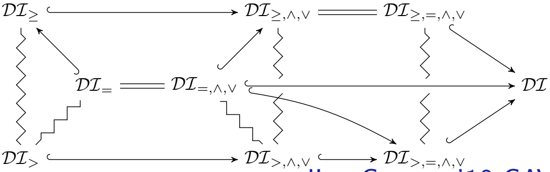


## Differential Cut



## Differential Ghost




## Logic

Provability study

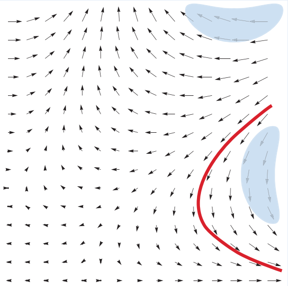
## Math

Character-istic PDE

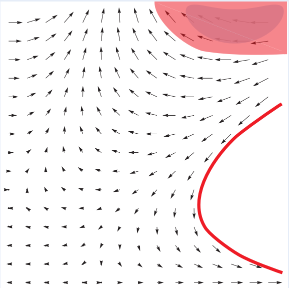
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

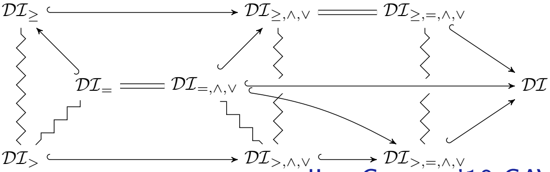
## Differential Invariant



## Differential Cut



## Differential Ghost


## Logic

Provability study

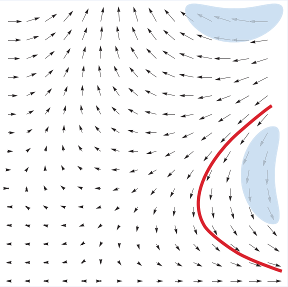
## Math

Character-istic PDE

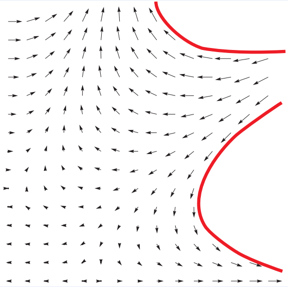
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

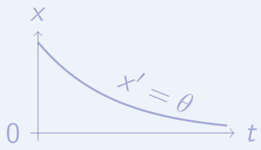
## Differential Invariant

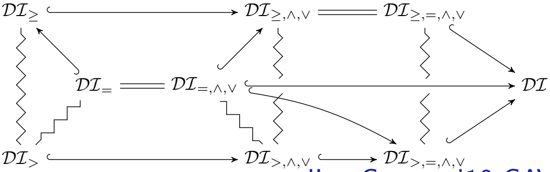


## Differential Cut



## Differential Ghost




## Logic

Provability study

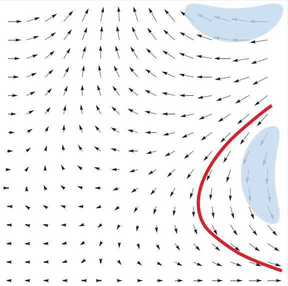
## Math

Character-istic PDE

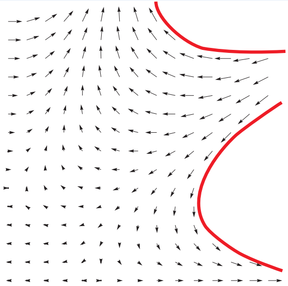
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

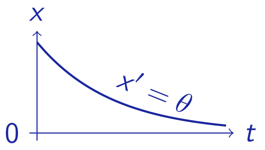
## Differential Invariant

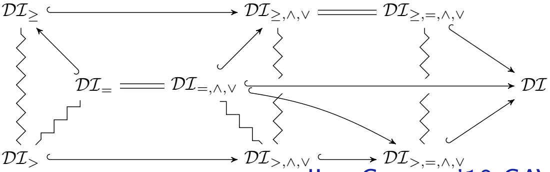


## Differential Cut



## Differential Ghost




## Logic

Provability study

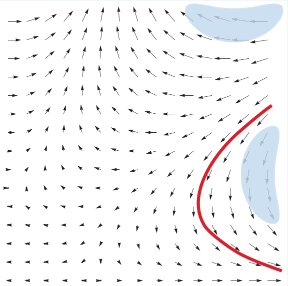
## Math

Character-istic PDE

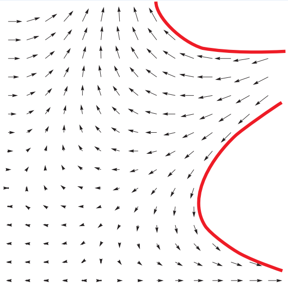
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

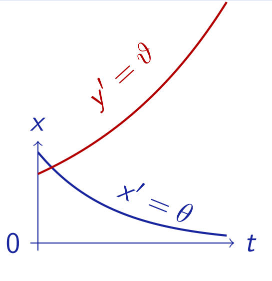
## Differential Invariant

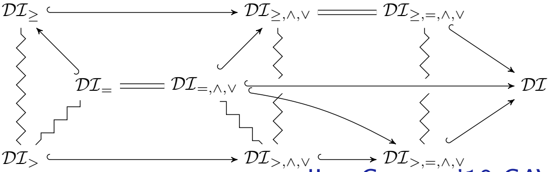


## Differential Cut



## Differential Ghost




## Logic

Provability study

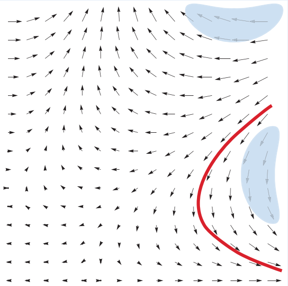
## Math

Character-istic PDE

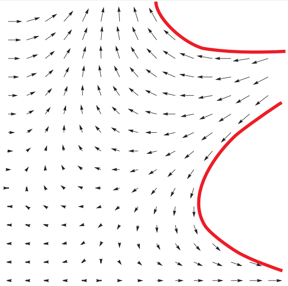
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12

# Differential Invariants for Differential Equations

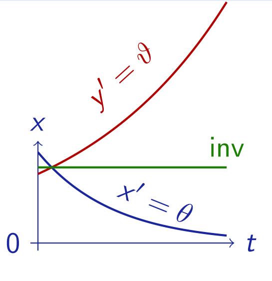
## Differential Invariant

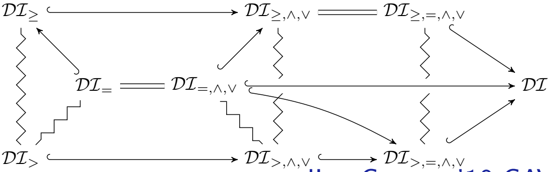


## Differential Cut



## Differential Ghost



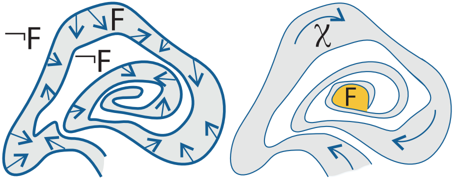

## Logic

Provability study

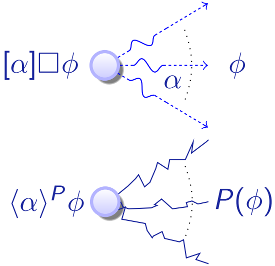
## Math

Character-istic PDE

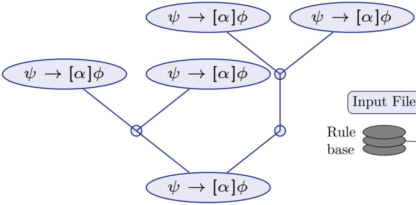
JLogComput'10,CAV'08,FMSD'09,LMCS'12,ITP'12



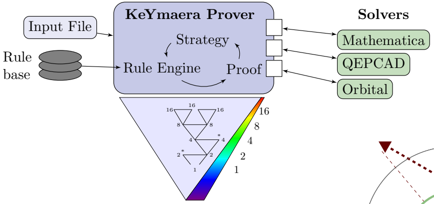
¬F
F
¬F
χ
F

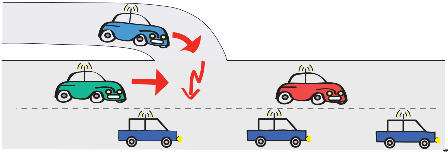
$[\alpha]\Box\phi$
$\alpha$
$\phi$
$\langle\alpha\rangle^P\phi$
$P(\phi)$

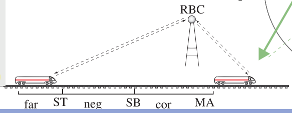



$\psi \to [\alpha]\phi$
$\psi \to [\alpha]\phi$
$\psi \to [\alpha]\phi$
$\psi \to [\alpha]\phi$
$\psi \to [\alpha]\phi$

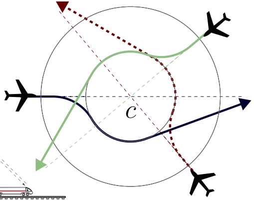



KeYmaera Prover
Input File
Strategy
Rule base
Rule Engine
Proof
Solvers
Mathematica
QEPCAD
Orbital
16
8
4
2
1

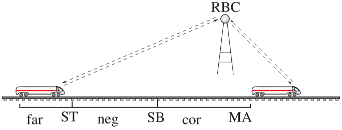


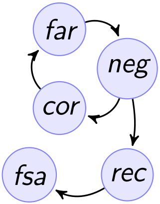
RBC
far
ST
neg
SB
cor
MA

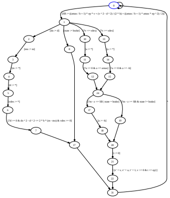



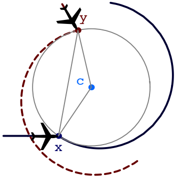
$c$

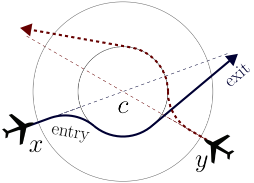
# Successful CPS Proofs

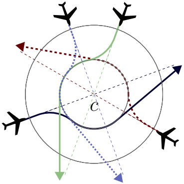


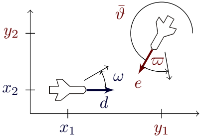



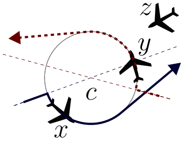



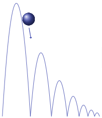


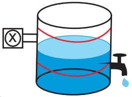



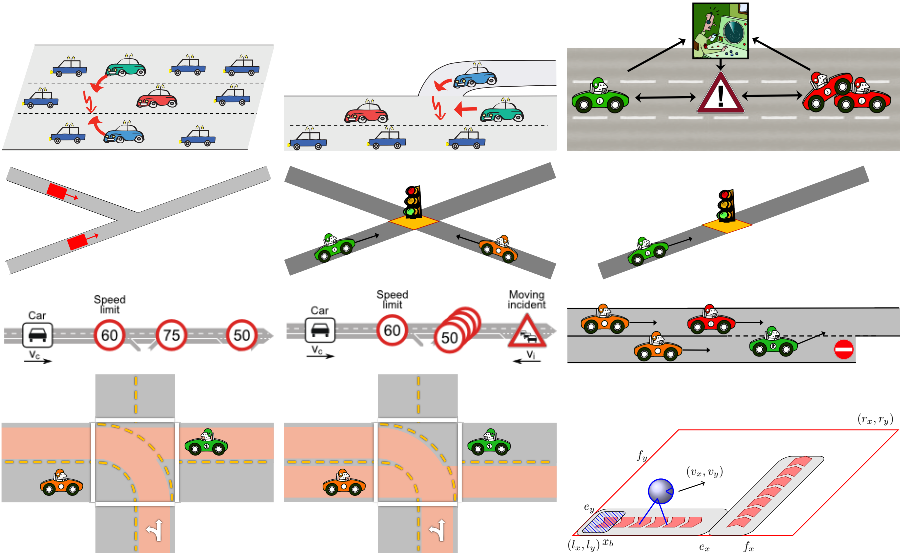





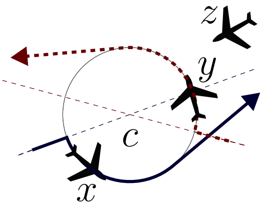
ICFEM'09,CAV'08,FM'09,HSCC'11

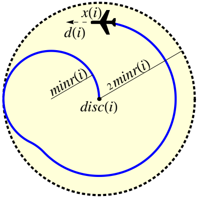
# Successful CPS Proofs

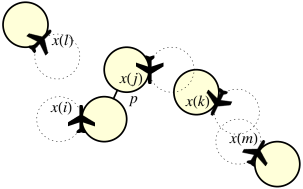


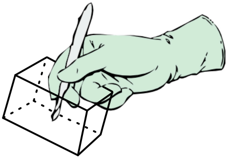

FM'11,LMCS'12,ICCPS'12,ITSC'11,IJCAR'12

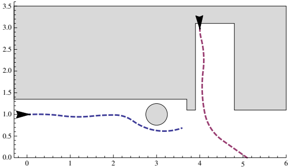
# Successful CPS Proofs

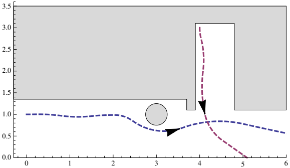


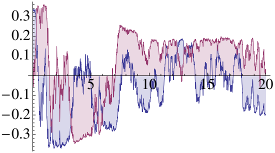



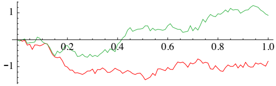



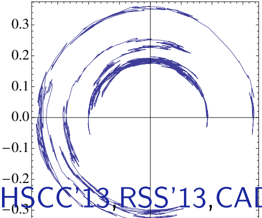





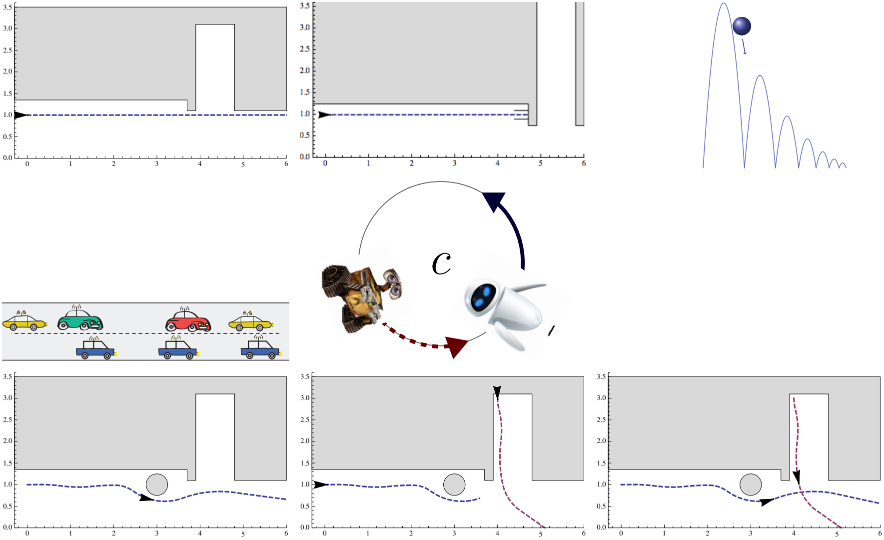








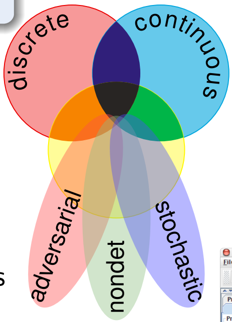
HSCC'11,HSCC'13,HSCC'13,RSS'13,CADE'12

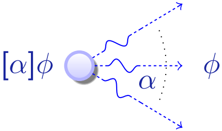



CMU 15-424/624 F'13 Students

**differential dynamic logic**

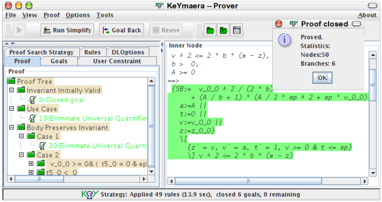
$d\mathcal{L} = \mathrm{DL} + \mathrm{HP}$

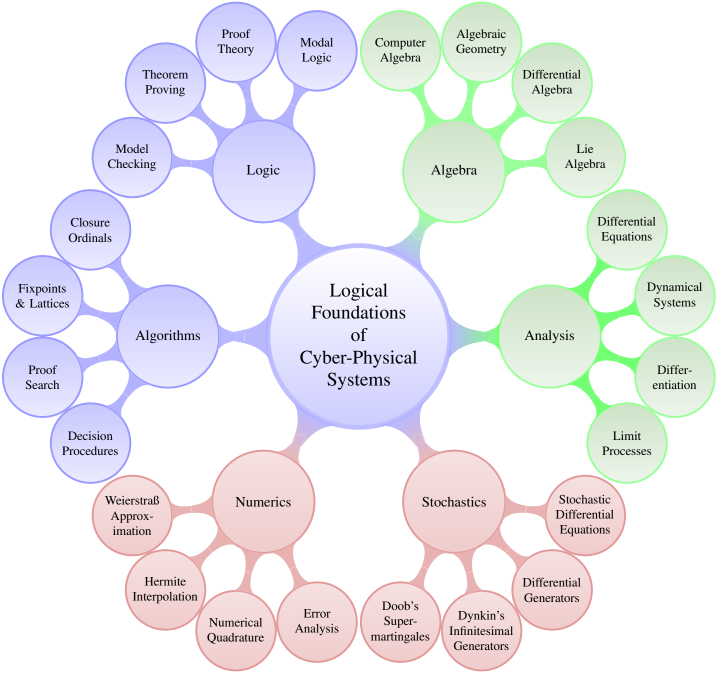


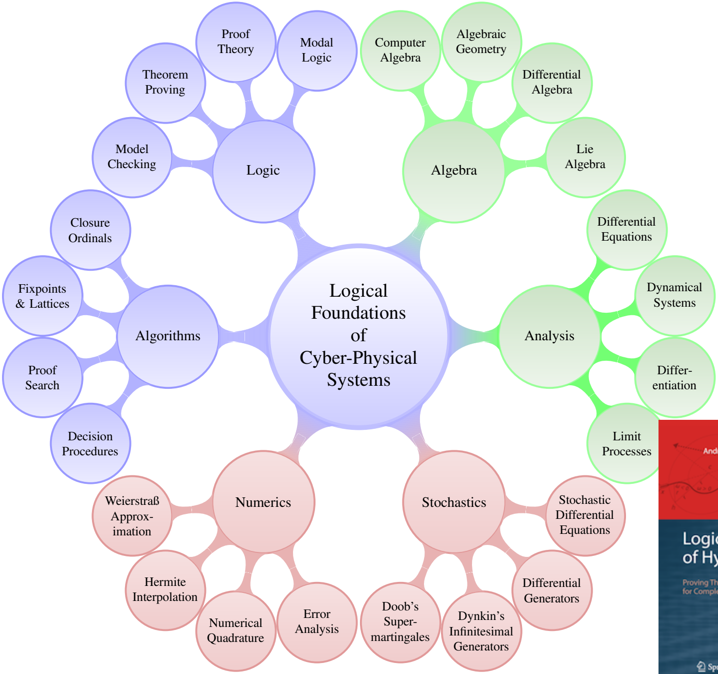



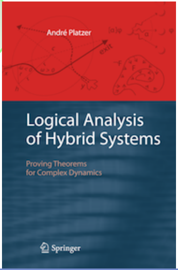

- Multi-dynamical systems
- Combine simple dynamics
- Tame complexity
- Logic & proofs for CPS
- Theory of CPS
- Applications

KeYmaera




Logical
Foundations
of
Cyber-Physical
Systems
Logic
Proof
Theory
Modal
Logic
Theorem
Proving
Model
Checking
Algebra
Computer
Algebra
Algebraic
Geometry
Differential
Algebra
Lie
Algebra
Algorithms
Closure
Ordinals
Fixpoints
& Lattices
Proof
Search
Decision
Procedures
Analysis
Differential
Equations
Dynamical
Systems
Differ-
entiation
Limit
Processes
Numerics
Weierstraß
Approx-
imation
Hermite
Interpolation
Numerical
Quadrature
Error
Analysis
Stochastics
Stochastic
Differential
Equations
Differential
Generators
Doob's
Super-
martingales
Dynkin's
Infinitesimal
Generators


Logical Foundations of Cyber-Physical Systems
Logic
Proof Theory
Modal Logic
Theorem Proving
Model Checking
Algebra
Computer Algebra
Algebraic Geometry
Differential Algebra
Lie Algebra
Analysis
Differential Equations
Dynamical Systems
Differ-entiation
Limit Processes
Algorithms
Closure Ordinals
Fixpoints & Lattices
Proof Search
Decision Procedures
Numerics
Weierstraß Approx-imation
Hermite Interpolation
Numerical Quadrature
Error Analysis
Stochastics
Stochastic Differential Equations
Differential Generators
Dynkin's Infinitesimal Generators
Doob's Super-martingales


André Platzer
Logical Analysis of Hybrid Systems
Proving Theorems for Complex Dynamics
Springer

André Platzer.
Logics of dynamical systems.
In *LICS* [9], pages 13–24.
doi:10.1109/LICS.2012.13.

André Platzer.
Differential dynamic logic for hybrid systems.
*J. Autom. Reas.*, 41(2):143–189, 2008.
doi:10.1007/s10817-008-9103-8.

André Platzer.
The complete proof theory of hybrid systems.
In *LICS* [9], pages 541–550.
doi:10.1109/LICS.2012.64.

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 20(1):309–352, 2010.
doi:10.1093/logcom/exn070.

André Platzer and Edmund M. Clarke.

Computing differential invariants of hybrid systems as fixedpoints.
*Form. Methods Syst. Des.*, 35(1):98–120, 2009.
Special issue for selected papers from CAV'08.
doi:10.1007/s10703-009-0079-8.

André Platzer.
The structure of differential invariants and differential cut elimination.
*Logical Methods in Computer Science*, 8(4):1–38, 2012.
doi:10.2168/LMCS-8(4:16)2012.

André Platzer.
A differential operator approach to equational differential invariants.
In Lennart Beringer and Amy Felty, editors, *ITP*, volume 7406 of *LNCS*, pages 28–48. Springer, 2012.
doi:10.1007/978-3-642-32347-8_3.

André Platzer.
*Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*.
Springer, Heidelberg, 2010.
doi:10.1007/978-3-642-14509-4.

*Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25–28, 2012*. IEEE, 2012.

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 20(1):309–352, 2010.
Advance Access published on November 18, 2008.
doi:10.1093/logcom/exn070.

André Platzer and Jan-David Quesel.
KeYmaera: A hybrid theorem prover for hybrid systems.
In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008.
doi:10.1007/978-3-540-71070-7_15.

André Platzer.
Differential dynamic logic for verifying parametric hybrid systems.
In Nicola Olivetti, editor, *TABLEAUX*, volume 4548 of *LNCS*, pages 216–232. Springer, 2007.

doi:10.1007/978-3-540-73099-6_17.

André Platzer.
Quantified differential dynamic logic for distributed hybrid systems.
In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *LNCS*, pages 469–483. Springer, 2010.
doi:10.1007/978-3-642-15205-4_36.

André Platzer.
A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems.
*Logical Methods in Computer Science*, 8(4):1–44, 2012.
Special issue for selected papers from CSL'10.
doi:10.2168/LMCS-8(4:17)2012.

André Platzer.
Quantified differential invariants.
In Emilio Frazzoli and Radu Grosu, editors, *HSCC*, pages 63–72. ACM, 2011.
doi:10.1145/1967701.1967713.

André Platzer.
Stochastic differential dynamic logic for stochastic hybrid programs.
In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *CADE*, volume 6803 of *LNCS*, pages 431–445. Springer, 2011.
doi:10.1007/978-3-642-22438-6_34.

André Platzer.
Logics of dynamical systems.
In *LICS* [9], pages 13–24.
doi:10.1109/LICS.2012.13.

André Platzer.
Differential dynamic logic for hybrid systems.
*J. Autom. Reas.*, 41(2):143–189, 2008.
doi:10.1007/s10817-008-9103-8.

André Platzer.
The complete proof theory of hybrid systems.
In *LICS* [9], pages 541–550.
doi:10.1109/LICS.2012.64.

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 20(1):309–352, 2010.
doi:10.1093/logcom/exn070.

André Platzer and Edmund M. Clarke.
Computing differential invariants of hybrid systems as fixedpoints.
*Form. Methods Syst. Des.*, 35(1):98–120, 2009.
Special issue for selected papers from CAV'08.
doi:10.1007/s10703-009-0079-8.

André Platzer.
The structure of differential invariants and differential cut elimination.
*Logical Methods in Computer Science*, 8(4):1–38, 2012.
doi:10.2168/LMCS-8(4:16)2012.

André Platzer.
A differential operator approach to equational differential invariants.
In Lennart Beringer and Amy Felty, editors, *ITP*, volume 7406 of *LNCS*, pages 28–48. Springer, 2012.

doi:10.1007/978-3-642-32347-8_3.

André Platzer.
*Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*.
Springer, Heidelberg, 2010.
doi:10.1007/978-3-642-14509-4.

*Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25–28, 2012*.
IEEE, 2012.

André Platzer.
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 20(1):309–352, 2010.
Advance Access published on November 18, 2008.
doi:10.1093/logcom/exn070.

André Platzer and Jan-David Quesel.
KeYmaera: A hybrid theorem prover for hybrid systems.

In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008. doi:10.1007/978-3-540-71070-7_15.

André Platzer.
Differential dynamic logic for verifying parametric hybrid systems.
In Nicola Olivetti, editor, *TABLEAUX*, volume 4548 of *LNCS*, pages 216–232. Springer, 2007.
doi:10.1007/978-3-540-73099-6_17.

André Platzer.
Quantified differential dynamic logic for distributed hybrid systems.
In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *LNCS*, pages 469–483. Springer, 2010.
doi:10.1007/978-3-642-15205-4_36.

André Platzer.
A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems.
*Logical Methods in Computer Science*, 8(4):1–44, 2012.
Special issue for selected papers from CSL'10.

doi:10.2168/LMCS-8(4:17)2012.

André Platzer.
Quantified differential invariants.
In Emilio Frazzoli and Radu Grosu, editors, *HSCC*, pages 63–72. ACM, 2011.
doi:10.1145/1967701.1967713.

André Platzer.
Stochastic differential dynamic logic for stochastic hybrid programs.
In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *CADE*, volume 6803 of *LNCS*, pages 431–445. Springer, 2011.
doi:10.1007/978-3-642-22438-6_34.

# Outline Background

# Outline Background

| | Op | Par | T | Cl | Tec | Aut | Cex | Dim | |
|---|---|---|---|---|---|---|---|---|---|
| HenzingerH94, HyTech | ✓ | × | ✓ | × | ✓ | ✓ | ✓ | | LHA |
| LafferrierePY99 | ✓ | × | ✓ | × | ✓ | | ✓ | | forgetful reset |
| Fränzle99 | ✓ | × | ✓ | × | ✓ | | ✓ | × | robust systems |
| CKrogh03, CheckMate | ✓ | × | ✓ | × | ✓ | ✓ | ✓ | | polyhedral |
| Frehse05, PHAVer | ✓ | × | ✓ | × | ✓ | ✓ | ✓ | 8 | LHA (+affine) |
| MysorePM05 | ✓ | × | ✓ | × | ✓ | ● | ✓ | 4 | bounded prefix |
| TomlinPS98,MBT05 | ○ | × | × | × | ○ | ○ | ● | 4 | HJB numPDE |
| RatschanS07, HSolver | ✓ | × | | × | ✓ | ✓ | × | 4 | interval |
| MannaS98, STeP | ✓ | | | × | ✓ | ○ | × | 7 | inv$\mapsto$VCG, flat |
| ÁbrahámSH01, PVS | ● | | | × | ● | ○ | × | $\approx$9 | HA$\hookrightarrow$PVS, -“- |
| ZhouRH92, EDC | × | ● | ✓ | .. | × | × | × | × | no maths |
| DavorenN00, L$\mu$ | × | × | | ✓ | ○ | × | × | × | prop. H-semantics |
| RönkköRS03, HGC | ✓ | × | × | × | × | × | × | × | HGC$\hookrightarrow$HOL |
| SSManna04 | ● | ○ | | × | ✓ | | × | 4/1 | equational system |
| CTiwari05 | ● | ○ | | × | ✓ | | × | 6/0 | linear, -”- |
| PrajnaJP07, barrier | ● | × | | × | ● | | × | 3 | needs 10000-dim |
| d$\mathcal{L}$ & dTL | ✓ | ✓ | ✓ | ✓ | ✓ | ● | × | 28 | expr., compos. |

| | Dom | Op | Base | Modal | Quant | Cmpl | Aut |
|---|---|---|---|---|---|---|---|
| DL | $\mathbb{N}$ | | $\mathrm{FOL}_{(\mathbb{N})}$ | | FV+unify | $/\mathbb{N}$ | |
| d$\mathcal{L}$ | $\mathbb{R}$ | $x'$ | $\mathrm{FOL}_{\mathbb{R}}$ | ODE | FV+requant+QE | /ODE | IBC |

# Outline Background

7. Formal Details
   - Soundness Proof
   - Completeness Proof
8. Differential Algebraic Dynamic Logic DAL (Excerpt)
   - Differential Invariants
9. Differential Temporal Dynamic Logic dTL (Excerpt)
10. Deduction Modulo Real Algebraic and Computer Algebraic Constraints
11. European Train Control System
12. Collision Avoidance Maneuvers in Air Traffic Control
13. Hybrid Automata Embedding
14. Distributed Hybrid Systems
15. Car Control Verification
16. Stochastic Hybrid Systems

# Soundness

**Proof (Soundness).**

- $x' = f(x)$
- Side deductions
- Free variables & Skolemisation

□

Return

# Incompleteness

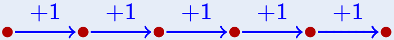
**Theorem**

*Discrete fragment and continuous fragment of* $\mathsf{d}\mathcal{L}$ *characterize* $\mathbb{N}$

**Proof.**

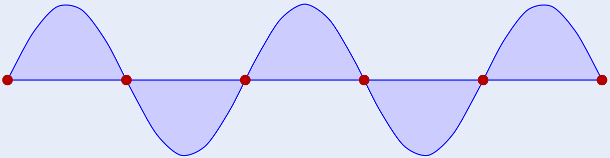
Discrete fragment:

$$\langle (x := x + 1)^* \rangle\ x = n$$




Continuous fragment:

$$\langle s'' = -s, \tau' = 1 \rangle (s = 0 \land \tau = n) \qquad \leadsto s = \sin$$



□

# Outline Background

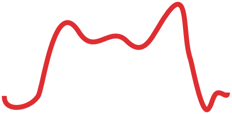
7. Formal Details
   - Soundness Proof
   - Completeness Proof
8. Differential Algebraic Dynamic Logic DAL (Excerpt)
   - Differential Invariants
9. Differential Temporal Dynamic Logic dTL (Excerpt)
10. Deduction Modulo Real Algebraic and Computer Algebraic Constraints
11. European Train Control System
12. Collision Avoidance Maneuvers in Air Traffic Control
13. Hybrid Automata Embedding
14. Distributed Hybrid Systems
15. Car Control Verification
16. Stochastic Hybrid Systems

# Incomplete! But are we missing proof rules?

# Incomplete! But are we missing proof rules?

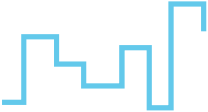
**Relativity**

Cook,Harel: discrete-DL/$\text{data}_{\mathbb{N}}$ hybrid-d$\mathcal{L}$/$\text{data}_{\mathbb{R}}$ ??

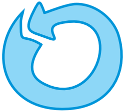


continuous

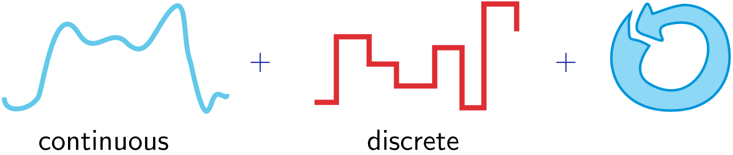
+

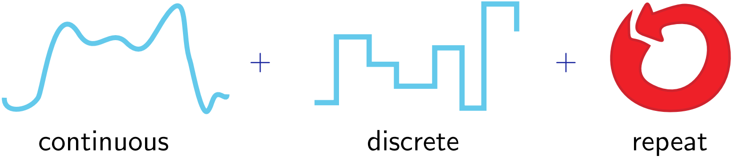


+

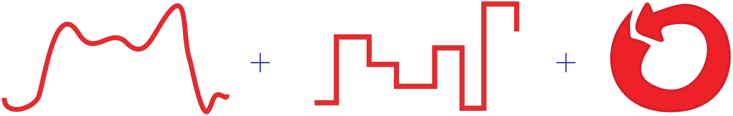


# Sources of Incompleteness



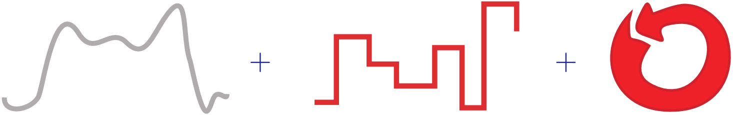

continuous
+
discrete
+
repeat

# Sources of Incompleteness

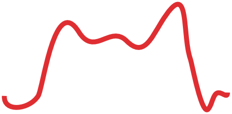


continuous + discrete + repeat

# Relative Completeness



continuous + discrete + repeat

 +  + 

continuous discrete repeat

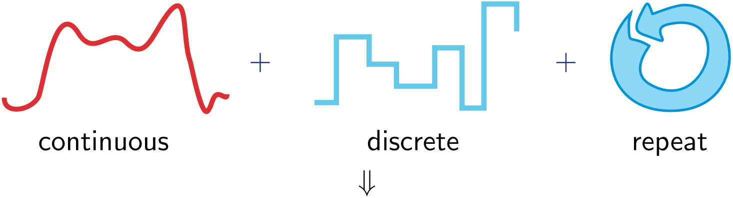
## Theorem (Relative Completeness)

d$\mathcal{L}$ calculus is complete relative to first-order logic of differential equations.

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathrm{FOD}} \vdash \phi$$

where $\mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

Proof Outline 15p



# Relative Completeness

## Theorem (Relative Completeness)

$d\mathcal{L}$ calculus is complete relative to first-order logic of differential equations.

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathrm{FOD}} \vdash \phi$$

where $\quad \mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

▸ Proof Outline 15p

$$\Downarrow$$

## Relativity

Cook,Harel: discrete-DL/data P.: hybrid-$d\mathcal{L}$/differential equations

# Relative Completeness

**Theorem (Relative Completeness)**

d$\mathcal{L}$ calculus is complete relative to first-order logic of differential equations.

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathrm{FOD}} \vdash \phi$$

where $\mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

▸ Proof Outline 15p

$\Downarrow$

**Corollary (Proof-theoretical Alignment)**

verification of hybrid systems = verification of dynamical systems!

## Theorem (Relative Completeness)

$d\mathcal{L}$ calculus is complete relative to first-order logic of differential equations.

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathrm{FOD}} \vdash \phi$$

where $\mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

▸ Proof Outline 15p

$$\Downarrow$$

## Corollary (Deductive Power)

$d\mathcal{L}$ calculus is *supremal hybrid* verification technique

# Relative Completeness Proof

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathsf{FOD}} \vdash \phi$$

$$\text{where} \quad \mathsf{FOD} = \mathsf{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \dots, x_n' = \theta_n]F$$

## Proof (Relative Completeness, 10 pages)

Return.

1. Strong invariants and variants expressible in d$\mathcal{L}$
2. d$\mathcal{L}$ expressible in FOD
3. valid d$\mathcal{L}$ formulas d$\mathcal{L}$-derivable from corresponding FOD axioms
4. finite FOD formula characterising unbounded hybrid repetition
5. FOD characterises $\mathbb{R}$-Gödel encoding
6. First-order expressible & program rendition: $\forall \phi\ \exists F \in \mathsf{FOD} \quad \vDash \phi \leftrightarrow F$
7. Propositionally & first-order complete
8. Relative complete for first-order safety $F \to [\alpha]G$
9. Relative complete for first-order liveness $F \to \langle \alpha \rangle G$

□

# Relative Completeness Proof

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathsf{FOD}} \vdash \phi$$

$$\text{where} \quad \mathsf{FOD} = \mathsf{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \dots, x_n' = \theta_n]F$$

## Proof (Relative Completeness, 10 pages)

Return

1. Strong invariants and variants expressible in $d\mathcal{L}$
2. $d\mathcal{L}$ expressible in FOD
3. valid $d\mathcal{L}$ formulas $d\mathcal{L}$-derivable from corresponding FOD axioms
4. finite FOD formula characterising unbounded hybrid repetition
5. FOD characterises $\mathbb{R}$-Gödel encoding
6. First-order expressible & program rendition: $\forall \phi \ \exists F \in \mathsf{FOD} \quad \vDash \phi \leftrightarrow F$
7. Propositionally & first-order complete
8. Relative complete for first-order safety $F \rightarrow [\alpha]G$
9. Relative complete for first-order liveness $F \rightarrow \langle\alpha\rangle G$

□

# Relative Completeness Proof

$$\vDash \phi \quad \text{iff} \quad Taut_{\text{FOD}} \vdash \phi$$

$$\text{where} \quad \text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$$

## Proof (Relative Completeness, 10 pages)

Return

1. Strong invariants and variants expressible in d$\mathcal{L}$
2. d$\mathcal{L}$ expressible in FOD
3. valid d$\mathcal{L}$ formulas d$\mathcal{L}$-derivable from corresponding FOD axioms
4. finite FOD formula characterising unbounded hybrid repetition
5. FOD characterises $\mathbb{R}$-Gödel encoding
6. First-order expressible & program rendition: $\forall \phi\ \exists F \in \text{FOD} \quad \vDash \phi \leftrightarrow F$
7. Propositionally & first-order complete
8. Relative complete for first-order safety $F \to [\alpha]G$
9. Relative complete for first-order liveness $F \to \langle \alpha \rangle G$

$\square$

# Relative Completeness Proof

$$\vDash \phi \quad \text{iff} \quad Taut_{\mathrm{FOD}} \vdash \phi$$

$$\text{where} \quad \mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$$

## Proof (Relative Completeness, 10 pages)

◂ Return

1. Strong invariants and variants expressible in d$\mathcal{L}$
2. d$\mathcal{L}$ expressible in FOD
3. valid d$\mathcal{L}$ formulas d$\mathcal{L}$-derivable from corresponding FOD axioms
4. finite FOD formula characterising unbounded hybrid repetition
5. FOD characterises $\mathbb{R}$-Gödel encoding
6. First-order expressible & program rendition: $\forall \phi \ \exists F \in \mathrm{FOD} \quad \vDash \phi \leftrightarrow F$
7. Propositionally & first-order complete
8. Relative complete for first-order safety $F \to [\alpha]G$
9. Relative complete for first-order liveness $F \to \langle \alpha \rangle G$

□

# Relative Completeness Proof

$$\vDash \phi \quad \text{iff} \quad Taut_{\text{FOD}} \vdash \phi$$

$$\text{where} \quad \text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \dots, x_n' = \theta_n]F$$

## Proof (Relative Completeness, 10 pages)

Return.

1. Strong invariants and variants expressible in d$\mathcal{L}$
2. d$\mathcal{L}$ expressible in FOD
3. valid d$\mathcal{L}$ formulas d$\mathcal{L}$-derivable from corresponding FOD axioms
4. finite FOD formula characterising unbounded hybrid repetition
5. FOD characterises $\mathbb{R}$-Gödel encoding
6. First-order expressible & program rendition: $\forall \phi \; \exists F \in \text{FOD} \quad \vDash \phi \leftrightarrow F$
7. Propositionally & first-order complete
8. Relative complete for first-order safety $F \rightarrow [\alpha]G$
9. Relative complete for first-order liveness $F \rightarrow \langle\alpha\rangle G$

□

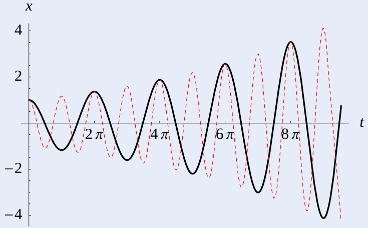
# Relative Completeness Proof

where $\quad \text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

Return

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$



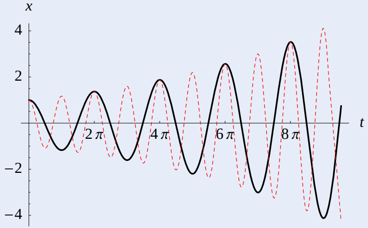
# Relative Completeness Proof

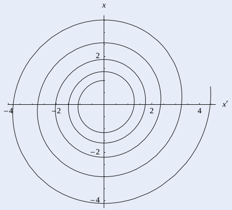
where $\quad \text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

◂ Return

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$



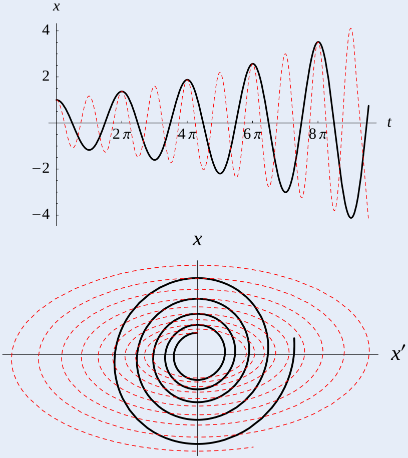



where $\text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

Return

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$



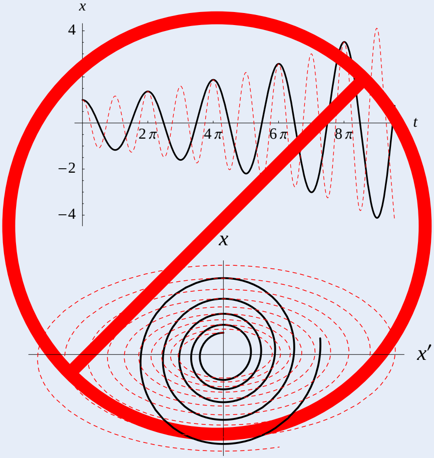
# Relative Completeness Proof

where $\text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \ldots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

Return.

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$ not differentiable!



where $\quad \mathrm{FOD} = \mathrm{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \dots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

Return

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$

$$\sum_{i=1}^{\infty} \frac{a_i}{2^i} = 0.a_1 a_2 \dots \qquad \sum_{i=1}^{\infty} \frac{b_i}{2^i} = 0.b_1 b_2 \dots$$

$$\leftrightarrow \sum_{i=0}^{\infty} \left( \frac{a_i}{2^{2i+1}} + \frac{b_i}{2^{2i+2}} \right) = 0.a_1 b_1 a_2 b_2 \dots$$

# Relative Completeness Proof

where $\quad \text{FOD} = \text{FOL}_{\mathbb{R}} + [x_1' = \theta_1, \dots, x_n' = \theta_n]F$

## Proof ($\mathbb{R}$-Gödel encoding)

Return

FOD characterises constructive bijection $\mathbb{R} \to \mathbb{R}^2$

$$\sum_{i=1}^{\infty} \frac{a_i}{2^i} = 0.a_1 a_2 \dots$$

$$\sum_{i=1}^{\infty} \frac{b_i}{2^i} = 0.b_1 b_2 \dots$$

$$\sum_{i=0}^{\infty} \left( \frac{a_i}{2^{2i+1}} + \frac{b_i}{2^{2i+2}} \right) = 0.a_1 b_1 a_2 b_2 \dots$$

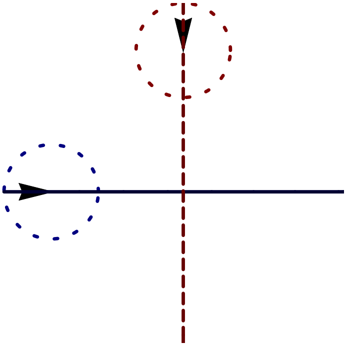
$$2^n = z \quad \leftrightarrow \quad \langle x := 1; \tau := 0; x' = x \ln 2 \land \tau' = 1 \rangle (\tau = n \land x = z)$$

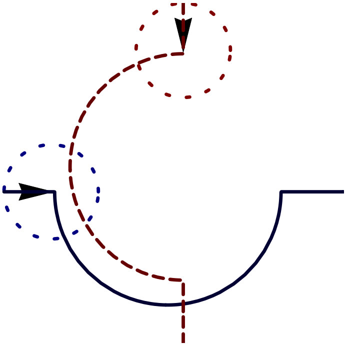
$$\ln 2 = z \quad \leftrightarrow \quad \langle x := 1; \tau := 0; x' = x \land \tau' = 1 \rangle (x = 2 \land \tau = z)$$

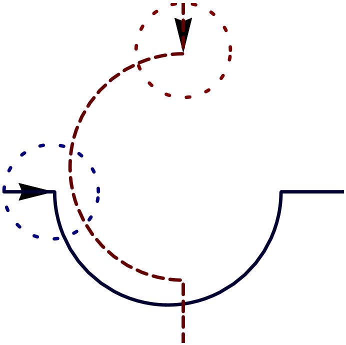
□

# Outline Background

# Air Traffic Control

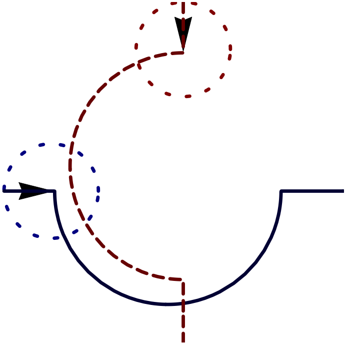


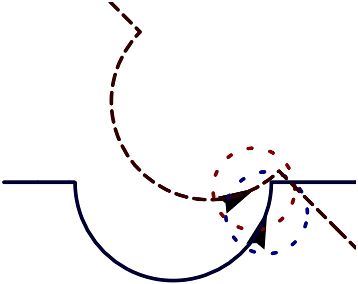




## Verification?

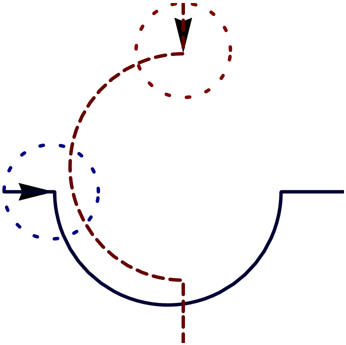
looks correct

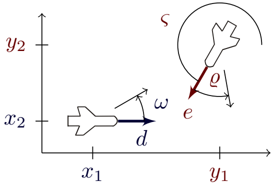




**Verification?**

looks correct NO!



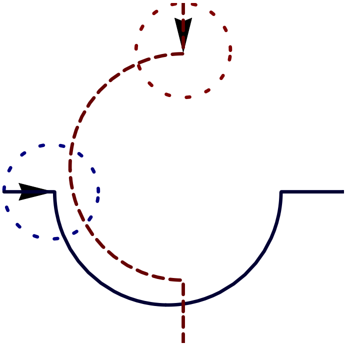


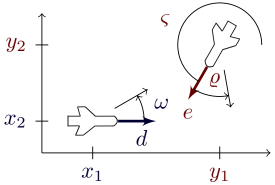

$$\begin{bmatrix} x_1' = -v_1 + v_2 \cos\vartheta + \omega x_2 \\ x_2' = v_2 \sin\vartheta - \omega x_1 \\ \vartheta' = \varpi - \omega \end{bmatrix}$$

## Verification?

looks correct NO!

# Air Traffic Control

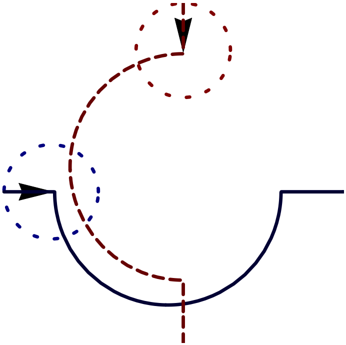


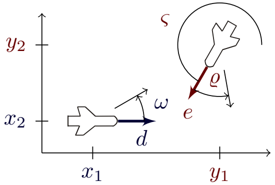



$$\begin{bmatrix} x_1' = -v_1 + v_2 \cos\vartheta + \omega x_2 \\ x_2' = v_2 \sin\vartheta - \omega x_1 \\ \vartheta' = \varpi - \omega \end{bmatrix}$$

## Example ("Solving" differential equations)

$$x_1(t) = \frac{1}{\omega\varpi}\Big(x_1\omega\varpi\cos t\omega - v_2\omega\cos t\omega\sin\vartheta + v_2\omega\cos t\omega\cos t\varpi\sin\vartheta - v_1\varpi\sin t\omega$$
$$+ x_2\omega\varpi\sin t\omega - v_2\omega\cos\vartheta\cos t\varpi\sin t\omega - v_2\omega\sqrt{1-\sin\vartheta^2}\sin t\omega$$
$$+ v_2\omega\cos\vartheta\cos t\omega\sin t\varpi + v_2\omega\sin\vartheta\sin t\omega\sin t\varpi\Big)\ldots$$






$$\begin{bmatrix} x_1' = -v_1 + v_2 \cos\vartheta + \omega x_2 \\ x_2' = v_2 \sin\vartheta - \omega x_1 \\ \vartheta' = \varpi - \omega \end{bmatrix}$$

## Example ("Solving" differential equations)

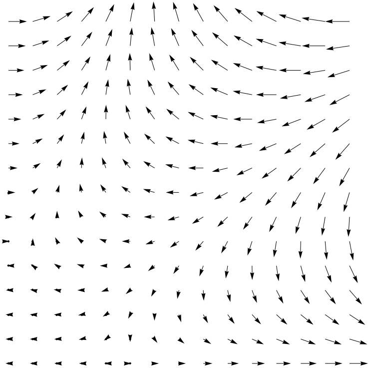
$$\forall t \geq 0 \quad \frac{1}{\omega\varpi}\big(x_1\omega\varpi \cos t\omega - v_2\omega \cos t\omega \sin\vartheta + v_2\omega \cos t\omega \cos t\varpi \sin\vartheta - v_1\varpi \sin t\omega$$
$$+ x_2\omega\varpi \sin t\omega - v_2\omega \cos\vartheta \cos t\varpi \sin t\omega - v_2\omega\sqrt{1 - \sin\vartheta^2} \sin t\omega$$
$$+ v_2\omega \cos\vartheta \cos t\omega \sin t\varpi + v_2\omega \sin\vartheta \sin t\omega \sin t\varpi\big) \ldots$$

# Outline Background

# Differential Invariants for Differential Equations

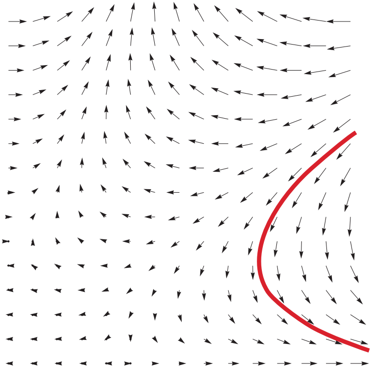
**"Definition" (Differential Invariant)**

"Formula that remains true in the direction of the dynamics"



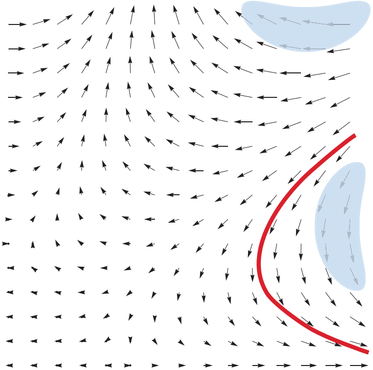
**"Definition" (Differential Invariant)**

"Formula that remains true in the direction of the dynamics"



"Definition" (Differential Invariant)

"Formula that remains true in the direction of the dynamics"



# Outline Background

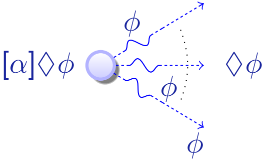
# Temporal Modalities + Dynamic Modalities

| problem | technique | Op | Par | T | closed |
|---|---|---|---|---|---|
| $train \models z < M$ | TL-MC | ✓ | × | ✓ | × |
| $\models (\mathrm{Ax}(train) \to z < M)$ | TL-calculus | × | ... | ✓ | ... |
| $\models [train]\, z < M$ | DL-calculus | ✓ | ✓ | × | ✓ |
| $\models [train]\Box\, z < M$ | dTL-calculus | ✓ | ✓ | ✓ | ✓ |

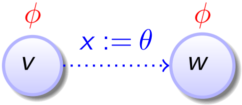
| problem | technique | Op | Par | T | closed |
|---|---|---|---|---|---|
| $train \models z < M$ | TL-MC | ✓ | × | ✓ | × |
| $\models (Ax(train) \rightarrow z < M)$ | TL-calculus | × | ... | ✓ | ... |
| $\models [train]\, z < M$ | DL-calculus | ✓ | ✓ | × | ✓ |
| $\models [train]\Box\, z < M$ | dTL-calculus | ✓ | ✓ | ✓ | ✓ |

**differential temporal dynamic logic**

$$dTL = TL + DL + HP$$

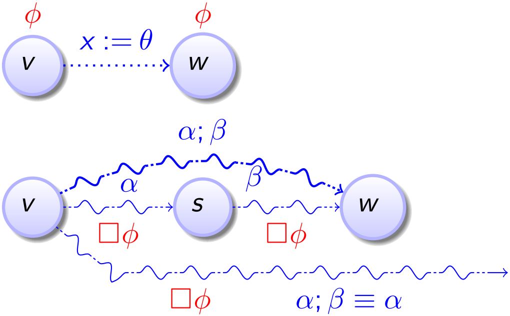


$$\frac{\phi \land [x := \theta]\phi}{[x := \theta]\Box\phi}$$



$$\frac{\phi \land [x := \theta]\phi}{[x := \theta]\Box\phi}$$

$$\frac{[\alpha]\Box\phi \land [\alpha][\beta]\Box\phi}{[\alpha;\beta]\Box\phi}$$

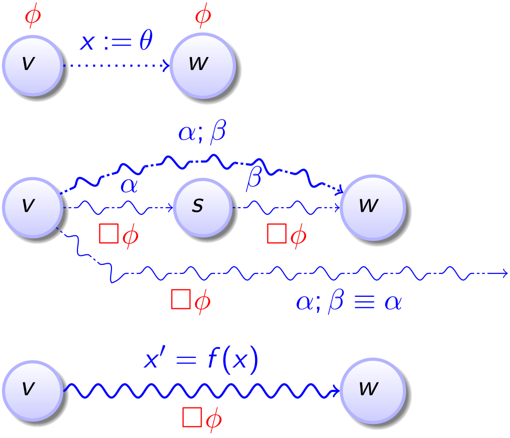


# Proof Calculus for Temporal dTL

$$\frac{\phi \land [x := \theta]\phi}{[x := \theta]\Box\phi}$$

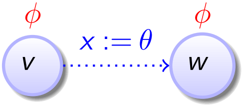
$$\frac{[\alpha]\Box\phi \land [\alpha][\beta]\Box\phi}{[\alpha;\beta]\Box\phi}$$

$$\frac{[x' = \theta]\phi}{[x' = \theta]\Box\phi}$$

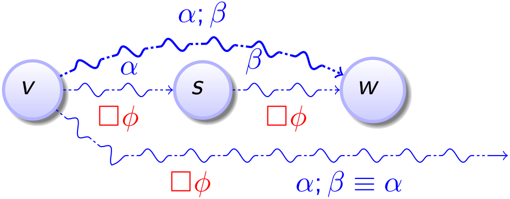


# Proof Calculus for Temporal dTL

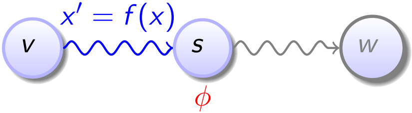
$$\frac{\phi \land [x := \theta]\phi}{[x := \theta]\Box\phi}$$



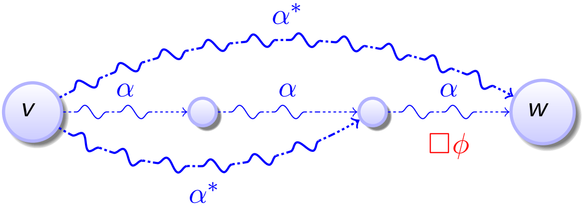

$$\frac{[\alpha]\Box\phi \land [\alpha][\beta]\Box\phi}{[\alpha;\beta]\Box\phi}$$




$$\frac{[x' = \theta]\phi}{[x' = \theta]\Box\phi}$$



$$\frac{[\alpha^*][\alpha]\Box\phi}{[\alpha^*]\Box\phi}$$



# Soundness and Completeness

**Theorem (Relative Completeness)** (P. 2008)

dTL *calculus is a sound & complete axiomatization relative to* d$\mathcal{L}$.

**Corollary (Continuous Relative Completeness)**

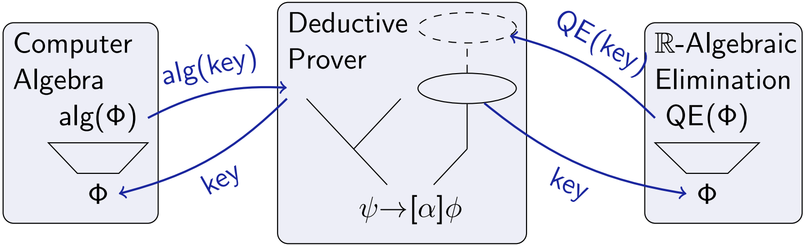
dTL *calculus is a sound & complete axiomatization relative to differential equations.*

**Corollary (Discrete Relative Completeness)**

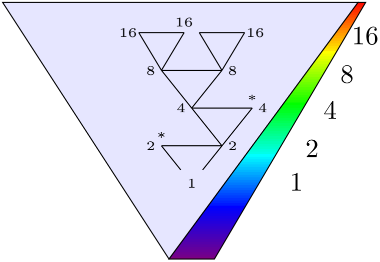
dTL *calculus is a sound & complete axiomatization relative to discrete systems.*

# Outline Background

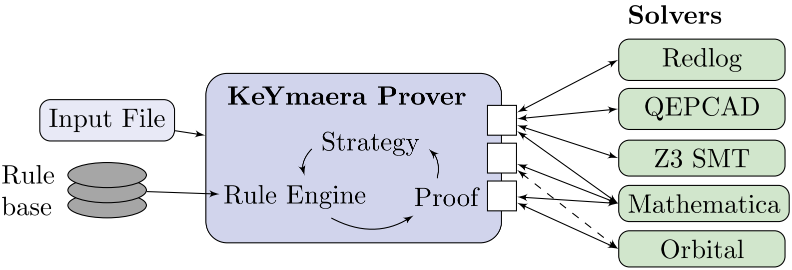
# KeYmaera Verification Architecture



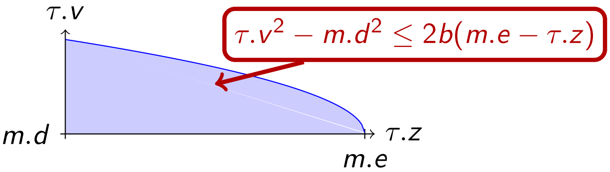

56 interactions?




0–1 interactions!

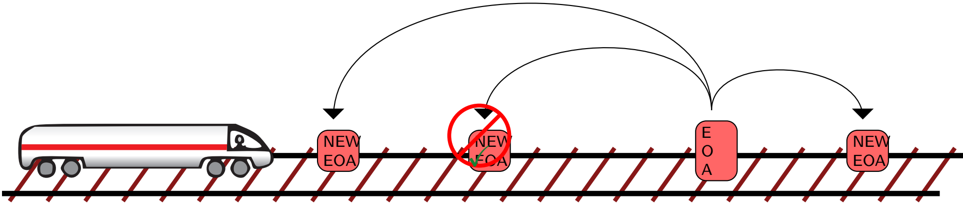

Solvers
Redlog
QEPCAD
Z3 SMT
Mathematica
Orbital
KeYmaera Prover
Input File
Strategy
Rule
base
Rule Engine
Proof

# Outline Background




**Proposition (Controllability)**

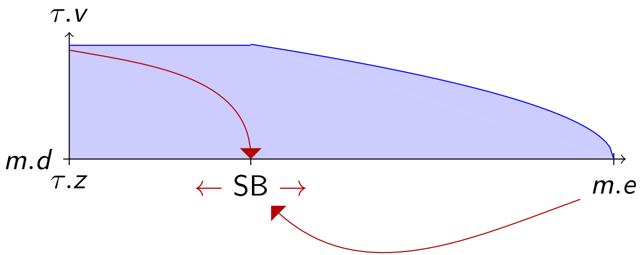
$$[\tau.z' = \tau.v, \tau.v' = -b \,\&\, \tau.v \geq 0](\tau.z \geq m.e \rightarrow \tau.v \leq m.d)$$
$$\equiv \tau.v^2 - m.d^2 \leq 2b(m.e - \tau.z)$$




**Proposition (RBC Controllability)**

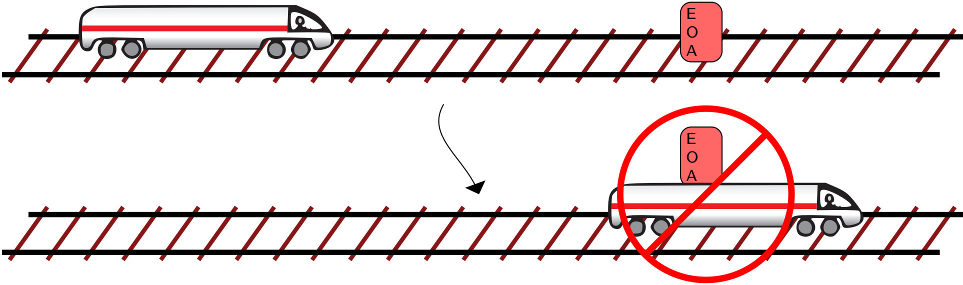
$$m.d \geq 0 \land b > 0 \rightarrow [m_0 := m;\ RBC] \Big( m_0.d^2 - m.d^2 \leq 2b(m.e - m_0.e) \land m_0.d \geq 0 \land m.d \geq 0 \leftrightarrow \forall \tau \big((\langle m := m_0 \rangle \tau.v^2 - m.d^2 \leq 2b(m.e - \tau.z)) \rightarrow \tau.v^2 - m.d^2 \leq 2b(m.e - \tau.z)\big)\Big)$$

# ETCS Reactivity




## Proposition (Reactivity)

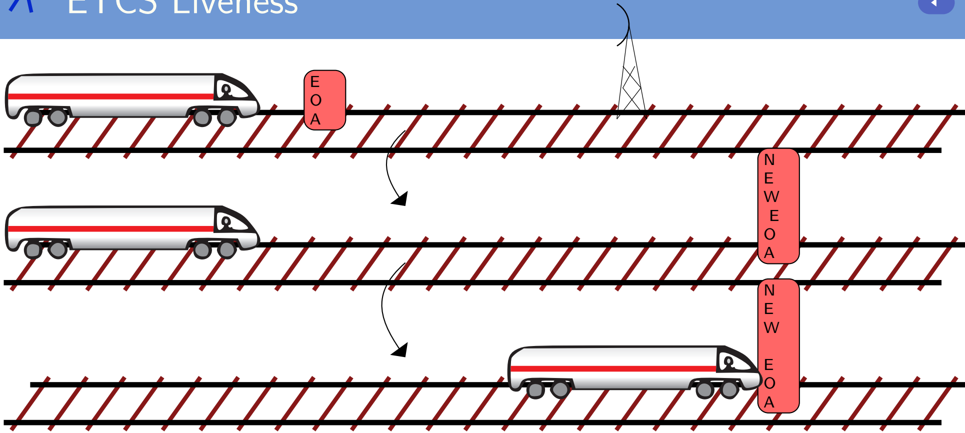
$$\Big(\forall m.e\,\forall \tau.z\,\big(m.e - \tau.z \ge SB \land \tau.v^2 - m.d^2 \le 2b(m.e - \tau.z) \rightarrow$$
$$[\tau.a := A;\ drive]\,\tau.v^2 - m.d^2 \le 2b(m.e - \tau.z)\big)\Big)$$
$$\equiv SB \ge \frac{\tau.v^2 - m.d^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon\,\tau.v\right)$$




**Proposition (Safety)**

$$\tau.v^2 - m.d^2 \le 2b(m.e - \tau.z) \rightarrow$$
$$[ETCS](\tau.z \ge m.e \rightarrow \tau.v \le m.d)$$

# ETCS Liveness




**Proposition (Liveness)**

$$\tau.v > 0 \land \varepsilon > 0 \rightarrow \forall P\ \langle ETCS \rangle\ \tau.z \geq P$$

# Safety Despite Disturbances

**So far: no wind, friction, etc.**

Direct control of the acceleration

**So far: no wind, friction, etc.**

Direct control of the acceleration

**Issue**

This is unrealistic!

**So far: no wind, friction, etc.**

Direct control of the acceleration

**Issue**

This is unrealistic!

**Solution** Take disturbances into account.

**Theorem**

ETCS is controllable, reactive, and safe in the presence of disturbances.

# Safety Despite Disturbances

**So far: no wind, friction, etc.**

Direct control of the acceleration

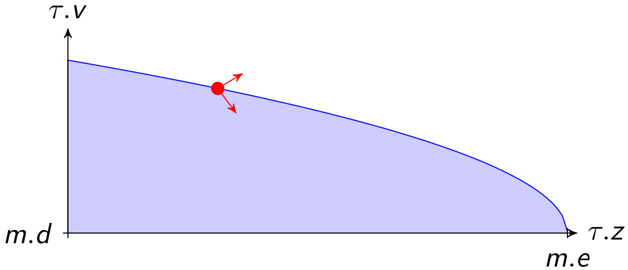
**Issue**

This is unrealistic!

**Solution** Take disturbances into account.

**Theorem**

ETCS is controllable, reactive, and safe in the presence of disturbances.



# Safety Despite Disturbances

**So far: no wind, friction, etc.**

Direct control of the acceleration

**Issue**

This is unrealistic!

**Solution** Take disturbances into account.

**Theorem**

ETCS is controllable, reactive, and safe in the presence of disturbances.

**Proof sketch**

The system now contains $\tau.a - l \leq \tau.v' \leq \tau.a + u$ instead of $\tau.v' = \tau.a$.

$\leadsto$ We cannot solve the differential equations anymore.

$\leadsto$ Use differential invariants for approximation. For details see paper.


Platzer, A.:
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 35(1): 309–352, 2010.

# Smooth Speed Control

**So far**

Almost completely non-deterministic control.

# Smooth Speed Control

**So far**

Almost completely non-deterministic control.

**Issue**

This is unrealistic!

# Smooth Speed Control

**So far**

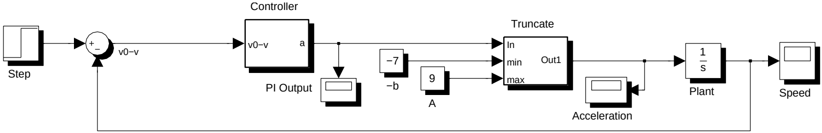
Almost completely non-deterministic control.

**Issue**

This is unrealistic!

**Solution**

Verify proportional-integral (PI) controllers used in trains.



**So far**

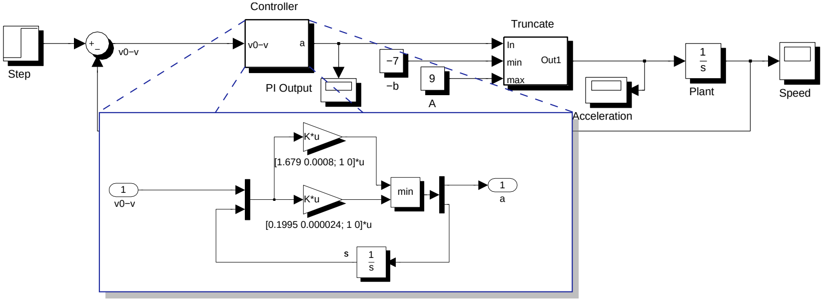
Almost completely non-deterministic control.

**Issue**

This is unrealistic!

**Solution**

Verify proportional-integral (PI) controllers used in trains.



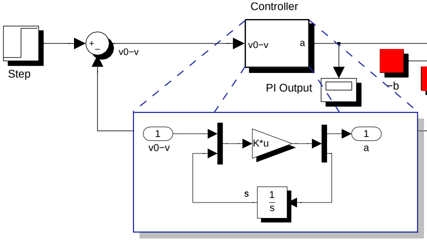
**So far**

Almost completely non-deterministic control.

**Issue**

This is unrealistic!

**Solution** Verify proportional-integral (PI) controllers used in trains.




**Differential equation system**

$$\tau.v' = \min\Big(A, \max\big(-b,\ \ell(\tau.v - m.r) - i\, s - c\, m.r\big)\Big) \land s' = \tau.v - m.r$$

# Smooth Speed Control

**So far**

Almost completely non-deterministic control.

**Issue**

This is unrealistic!

**Solution** Verify proportional-integral (PI) controllers used in trains.

**Theorem**

The ETCS system remains safe when speed is controlled by a PI controller.

**Proof sketch**

Cannot solve differential equations really. Use differential invariants! For details see paper.

Platzer, A.:
Differential-algebraic dynamic logic for differential-algebraic programs.
*J. Log. Comput.*, 35(1): 309–352, 2010.

# Experimental Results (ETCS)

| Case Study | | Int | Time(s) | Mem(Mb) | Steps | Dim |
|---|---|---|---|---|---|---|
| controllability | train | 0 | 0.6 | 6.9 | 14 | 5 |
| controllability | RBC | 0 | 0.5 | 6.4 | 42 | 12 |
| controllability | RBC | 0 | 0.9 | 6.5 | 82 | 12 |
| reactivity | | 13 | 279.1 | 98.3 | 265 | 14 |
| reactivity | | 0 | 103.9 | 61.7 | 47 | 14 |
| safety | | 0 | 2052.4 | 204.3 | 153 | 14 |
| liveness | essentials | 4 | 35.2 | 92.2 | 62 | 10 |
| liveness | simplified | 6 | 9.6 | 23.5 | 134 | 13 |
| controllability | disturbance | 0 | 2.8 | 8.3 | 26 | 7 |
| reactivity | disturbance | 1 | 23.7 | 47.6 | 76 | 15 |
| safety | disturbance | 1 | 5805.2 | 34 | 218 | 16 |

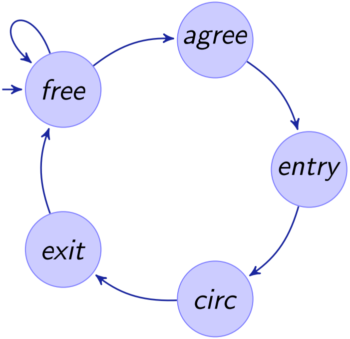
# European Train Control System (ETCS)

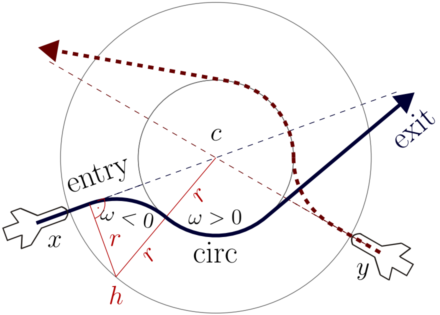
provable automatically!

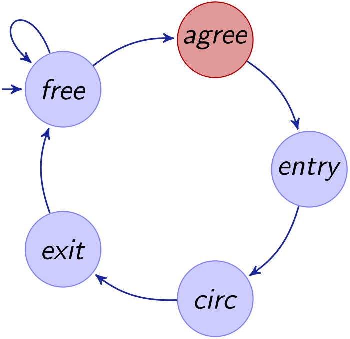
$$
\begin{aligned}
\text{spec} &: \tau.v^2 - \mathbf{m}.d^2 \le 2b(\mathbf{m}.e - \tau.p) \land \tau.v \ge 0 \land \mathbf{m}.d \ge 0 \land b > 0 \\
&\quad \to \ [\text{ETCS}](\tau.p \ge \mathbf{m}.e \to \tau.v \le \mathbf{m}.d) \\
\text{ETCS} &: (\text{train} \cup \text{rbc})^* \\
\text{train} &: \text{spd};\ \text{atp};\ \text{move} \\
\text{spd} &: (?\tau.v \le \mathbf{m}.r;\ \tau.a := *;\ ? - b \le \tau.a \le A) \\
&\quad \cup (?\tau.v \ge \mathbf{m}.r;\ \tau.a := *;\ ?0 > \tau.a \ge -b) \\
\text{atp} &: SB := \frac{\tau.v^2 - \mathbf{m}.d^2}{2b} + \left(\frac{A}{b} + 1\right)\left(\frac{A}{2}\varepsilon^2 + \varepsilon\,\tau.v\right); \\
&\quad (?(\mathbf{m}.e - \tau.p \le SB \lor rbc.message = emergency);\ \tau.a := -b) \\
&\quad \cup (?\mathbf{m}.e - \tau.p \ge SB \land rbc.message \ne emergency) \\
\text{move} &: t := 0;\ (\tau.p' = \tau.v, \tau.v' = \tau.a, t' = 1 \,\&\, \tau.v \ge 0 \land t \le \varepsilon) \\
\text{rbc} &: (rbc.message := emergency) \\
&\quad \cup \Big(\mathbf{m}_0 := \mathbf{m}; \mathbf{m} := *; \\
&\qquad ?\mathbf{m}.r \ge 0 \land \mathbf{m}.d \ge 0 \land \mathbf{m}_0.d^2 - \mathbf{m}.d^2 \le 2b(\mathbf{m}.e - \mathbf{m}_0.e)\Big)
\end{aligned}
$$

```
 state = 0,
 2 * b * (m - z) >= v ^ 2 - d ^ 2,
 v >= 0, d >= 0, v >= 0, ep >  0, b >  0, amax >  0, d >= 0
==>
    v <= vdes
 -> \forall R a_3;
      (    a_3 >= 0 & a_3 <= amax
       ->   (       m - z
               <= (amax / b + 1) * ep * v
                + (v ^ 2 - d ^ 2) / (2 * b)
                + (amax / b + 1) * amax * ep ^ 2 / 2
            -> \forall R t0;
                 (    t0 >= 0
                  -> \forall R ts0;  (0 <= ts0 & ts0 <= t0 -> -b * ts0 + v >= 0 & ts0 + 0 <= ep)
                  ->       2 * b * (m - 1 / 2 * (-b * t0 ^ 2 + 2 * t0 * v + 2 * z))
                        >= (-b * t0 + v) ^ 2
                         - d ^ 2
                      & -b * t0 + v >= 0
                      & d >= 0))
         & (       m - z
               >  (amax / b + 1) * ep * v
                + (v ^ 2 - d ^ 2) / (2 * b)
                + (amax / b + 1) * amax * ep ^ 2 / 2
            -> \forall R t2;
                 (    t2 >= 0
                  -> \forall R ts2;  (0 <= ts2 & ts2 <= t2 -> a_3 * ts2 + v >= 0 & ts2 + 0 <= ep)
                  ->       2 * b * (m - 1 / 2 * (a_3 * t2 ^ 2 + 2 * t2 * v + 2 * z))
                        >= (a_3 * t2 + v) ^ 2
                         - d ^ 2
                      & a_3 * t2 + v >= 0
                      & d >= 0)))
```

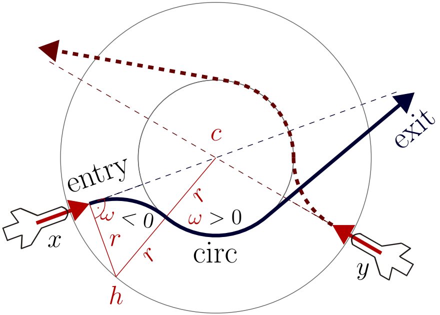
# Outline Background


agree
free
entry
exit
circ

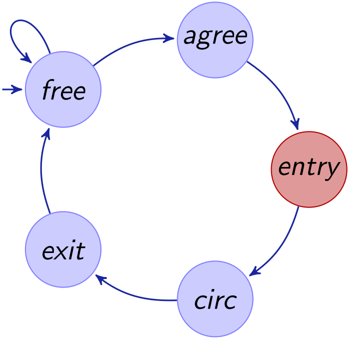

c
entry
exit
r
r
r
$\omega < 0$
$\omega > 0$
x
y
circ
h

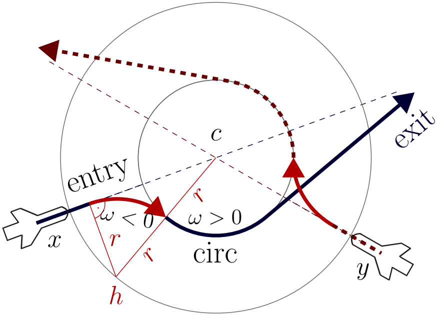


agree
free
entry
exit
circ

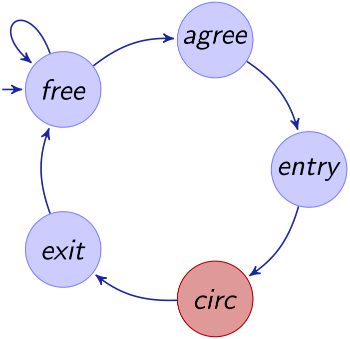



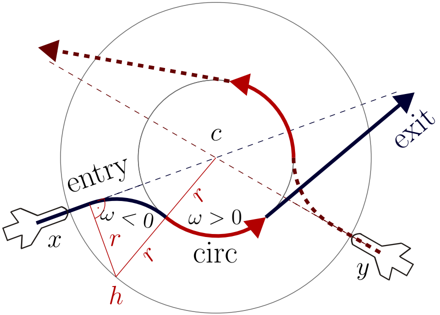
c
entry
exit
$\omega < 0$
$\omega > 0$
r
r
r
x
y
circ
h

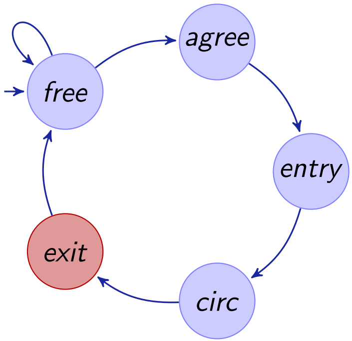

free
agree
entry
circ
exit

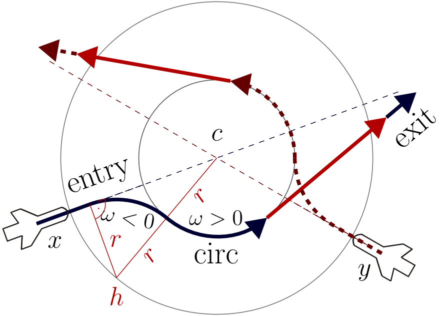

c
entry
exit
x
y
h
r
ω < 0
ω > 0
circ

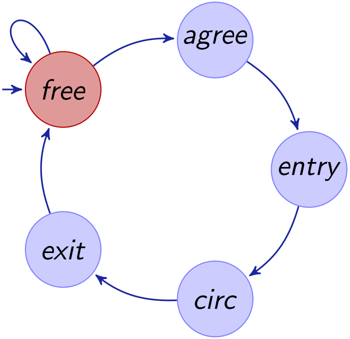
# Flyable Roundabout Maneuver: Overview



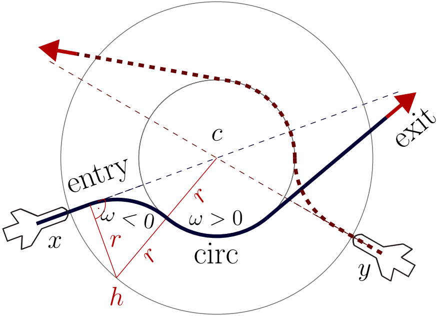




agree
free
entry
exit
circ

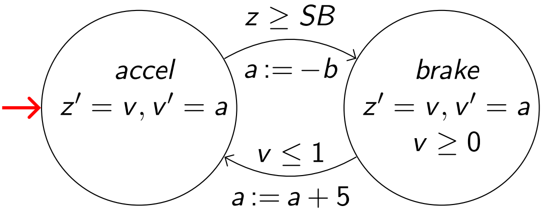

c
entry
exit
$\omega < 0$
$\omega > 0$
r
r
r
r
x
y
circ
h



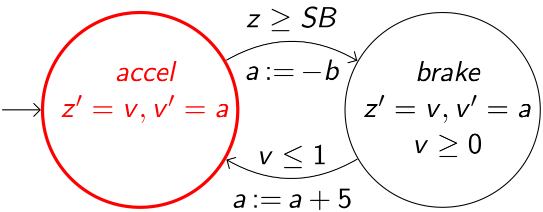
free
agree
entry
circ
exit

c
entry
exit
circ
x
y
h
r
r
r
r
ω < 0
ω > 0

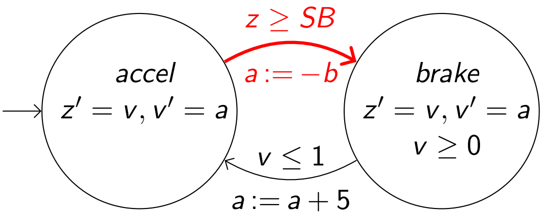



$$
\begin{aligned}
& q := accel; \\
& \big( \quad (?q = accel; \quad z' = v, v' = a) \\
& \cup \ (?q = accel \land z \geq SB; \quad a := -b; \quad q := brake; \quad ?v \geq 0) \\
& \cup \ (?q = brake; \quad z' = v, v' = a \,\&\, v \geq 0) \\
& \cup \ (?q = brake \land v \leq 1; \quad a := a + 5; \quad q := accel) \big)^*
\end{aligned}
$$



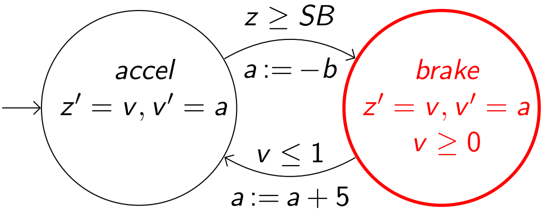

$$\rightsquigarrow$$

$$
\begin{aligned}
&q := accel;\\
&\big(\quad (?q = accel;\quad z' = v, v' = a)\\
&\cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
&\cup\ (?q = brake;\quad z' = v, v' = a \,\&\, v \geq 0)\\
&\cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel)\big)^*
\end{aligned}
$$



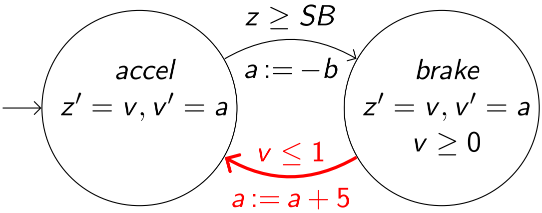

$$\rightsquigarrow$$

$$\begin{aligned}
&q := accel;\\
&(\quad (?q = accel;\ \ z' = v, v' = a)\\
&\cup\ (?q = accel \land z \geq SB;\ \ a := -b;\ \ q := brake;\ \ ?v \geq 0)\\
&\cup\ (?q = brake;\ \ z' = v, v' = a \,\&\, v \geq 0)\\
&\cup\ (?q = brake \land v \leq 1;\ \ a := a + 5;\ \ q := accel)\big)^*
\end{aligned}$$



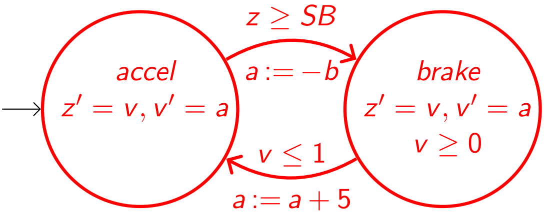

$\rightsquigarrow$

$$
\begin{aligned}
& q := accel; \\
& \big( \quad (?q = accel; \quad z' = v, v' = a) \\
& \cup \ (?q = accel \land z \geq SB; \quad a := -b; \quad q := brake; \quad ?v \geq 0) \\
& \cup \ (?q = brake; \quad z' = v, v' = a \,\&\, v \geq 0) \\
& \cup \ (?q = brake \land v \leq 1; \quad a := a + 5; \quad q := accel)\big)^*
\end{aligned}
$$




$\rightsquigarrow$

$$
\begin{aligned}
& q := accel;\\
& \big( \quad (?q = accel;\quad z' = v, v' = a)\\
& \cup\ (?q = accel \land z \geq SB;\quad a := -b;\quad q := brake;\quad ?v \geq 0)\\
& \cup\ (?q = brake;\quad z' = v, v' = a \,\&\, v \geq 0)\\
& \cup\ (?q = brake \land v \leq 1;\quad a := a + 5;\quad q := accel)\big)^*
\end{aligned}
$$




$\rightsquigarrow$

$$
\begin{aligned}
& q := accel; \\
& \big( \quad (?q = accel; \quad z' = v, v' = a) \\
& \cup \; (?q = accel \land z \geq SB; \quad a := -b; \quad q := brake; \quad ?v \geq 0) \\
& \cup \; (?q = brake; \quad z' = v, v' = a \,\&\, v \geq 0) \\
& \cup \; (?q = brake \land v \leq 1; \quad a := a + 5; \quad q := accel)\big)^*
\end{aligned}
$$

# Outline Background

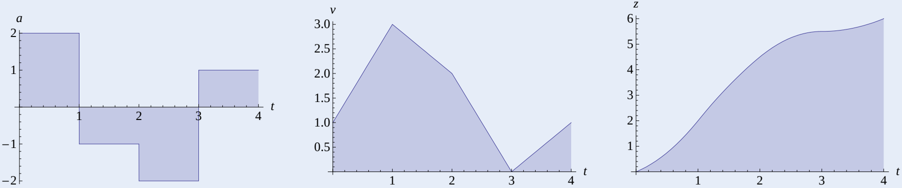
Q: I want to verify my car

## Challenge

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

## Challenge (Hybrid Systems)

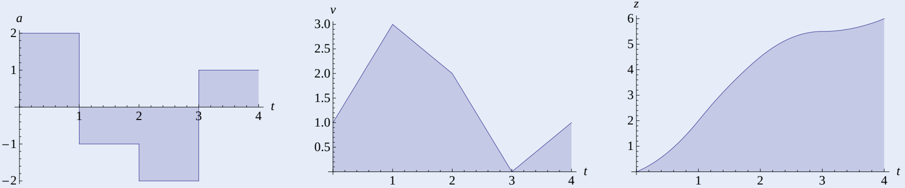
- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)



Q: I want to verify my car A: Hybrid systems Q: But there's a lot of cars!

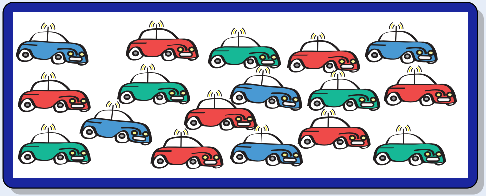
## Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)



Q: I want to verify a lot of cars

## Challenge

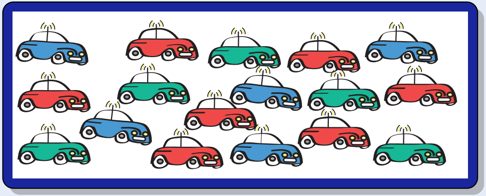


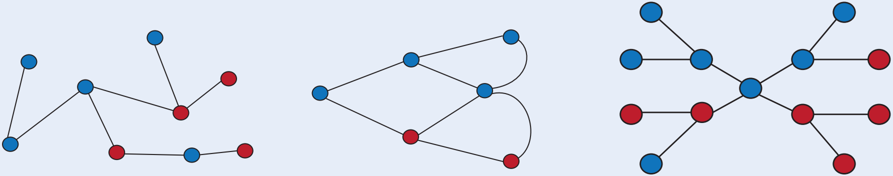
# Cyber-Physical Systems: Distributed Systems

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

## Challenge (Distributed Systems)

- Local computation (finite state automaton)
- Remote communication (network graph)

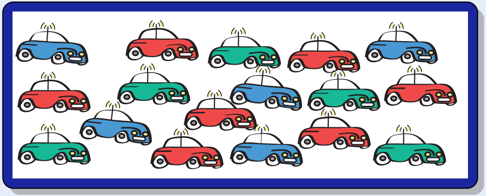


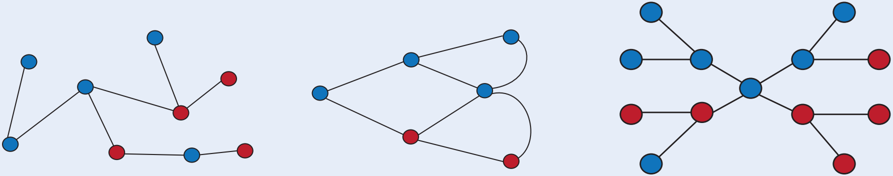


Q: I want to verify a lot of cars A: Distributed systems Q: But they move!

## Challenge (Distributed Systems)

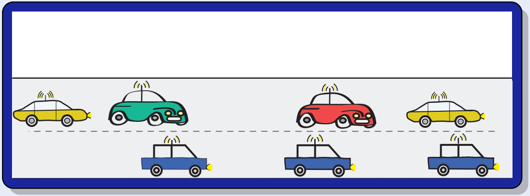
- Local computation (finite state automaton)
- Remote communication (network graph)





Q: I want to verify lots of moving cars

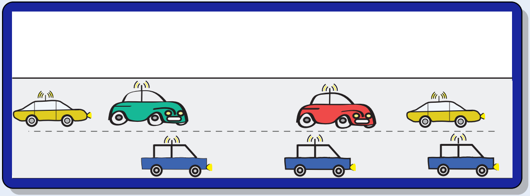
## Challenge



Q: I want to verify lots of moving cars A: Distributed hybrid systems

## Challenge (Distributed Hybrid Systems)

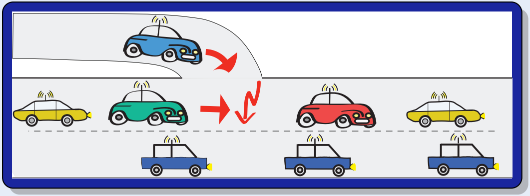
- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (communication/coupling)



Q: I want to verify lots of moving cars A: Distributed hybrid systems

## Challenge (Distributed Hybrid Systems)

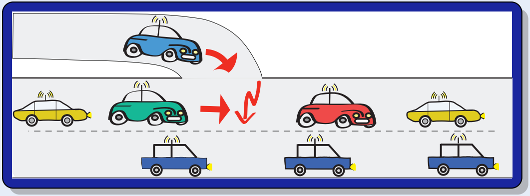
- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (communication/coupling)
- Dimensional dynamics (appearance)



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 (communication/coupling)
- Dimensional dynamics (appearance)

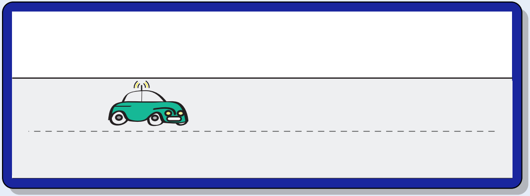


# Model for Distributed Hybrid Systems

Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Structural dynamics (communication/coupling)

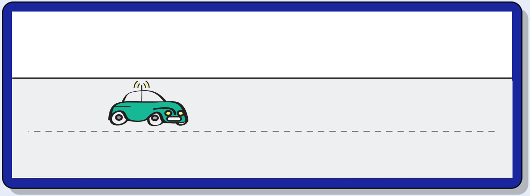


# Model for Distributed Hybrid Systems

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)



# Model for Distributed Hybrid Systems

Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

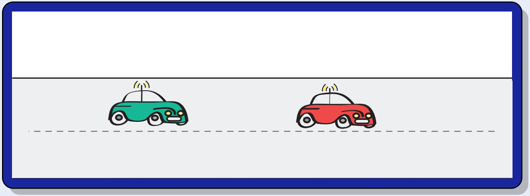
- Continuous dynamics (differential equations)

  $x'' = a$

- Discrete dynamics (control decisions)

$a := \texttt{if} .. \texttt{then}\, a\, \texttt{else} -b\, \texttt{fi}$

- Structural dynamics (communication/coupling)



# Model for Distributed Hybrid Systems

Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

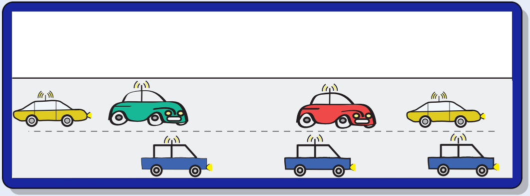
- Continuous dynamics (differential equations)

  $$x'' = a$$

- Discrete dynamics (control decisions)

  $a := \texttt{if} .. \texttt{then}\, a \,\texttt{else} -b\, \texttt{fi}$

- Structural dynamics (communication/coupling)

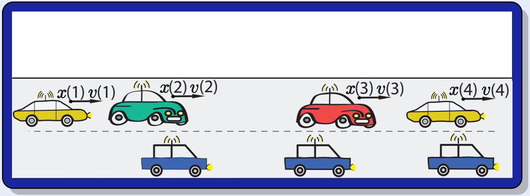


# Model for Distributed Hybrid Systems

Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

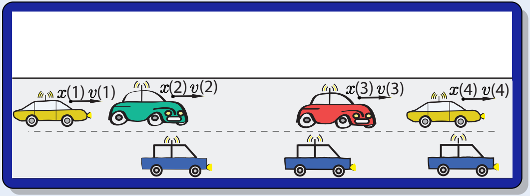
- Continuous dynamics (differential equations)
  $$x'' = a$$
- Discrete dynamics (control decisions)
  $$a := \texttt{if} \,..\, \texttt{then}\, a \,\texttt{else}\, -b \,\texttt{fi}$$
- Structural dynamics (communication/coupling)



Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  $$x(i)'' = a(i)$$
- Discrete dynamics (control decisions)
  $$a(i) := \texttt{if} .. \texttt{then}\, a \,\texttt{else} -b \,\texttt{fi}$$
- Structural dynamics (communication/coupling)



# Model for Distributed Hybrid Systems

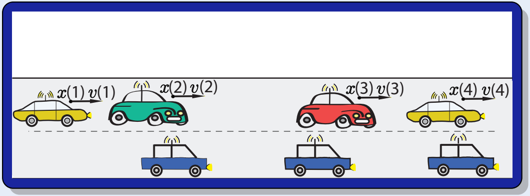
Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  $\forall i\; x(i)' = a(i)$
- Discrete dynamics (control decisions)

$\forall i\; a(i) := \texttt{if} .. \texttt{then}\; a\; \texttt{else}\; -b\; \texttt{fi}$

- Structural dynamics (communication/coupling)



# Model for Distributed Hybrid Systems

Q: How to model distributed hybrid systems

## Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)

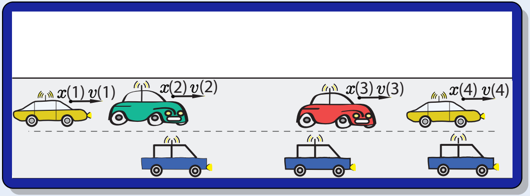
  $\forall i\; x(i)'' = a(i)$

- Discrete dynamics (control decisions)

$\forall i\; a(i) := \texttt{if} .. \texttt{then}\; a\; \texttt{else}\; -b\; \texttt{fi}$

- Structural dynamics (communication/coupling)

  $\ell(i) := \mathit{carInFrontOf(i)}$



# Model for Distributed Hybrid Systems

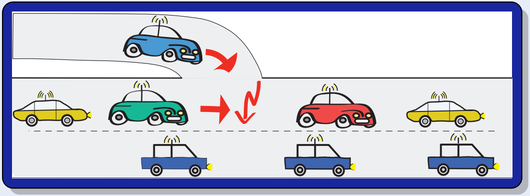
Q: How to model distributed hybrid systems A: Quantified Hybrid Programs

## Model (Distributed Hybrid Systems)

- Continuous dynamics (differential equations)
  $\forall i\, x(i)'' = a(i)$
- Discrete dynamics (control decisions)

$\forall i\, a(i) := \mathtt{if} .. \mathtt{then}\, a\, \mathtt{else}\, -b\, \mathtt{fi}$

- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)

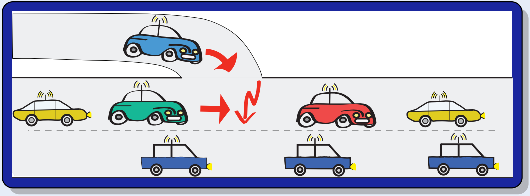


# Model for Distributed Hybrid Systems

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)
  $\forall i\, a(i) := \texttt{if .. then } a \texttt{ else } -b \texttt{ fi}$
- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \texttt{new } Car$

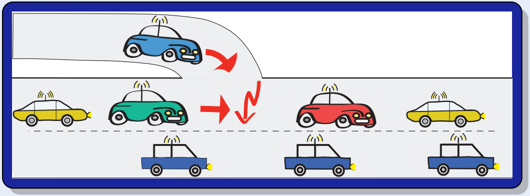


# Model for Distributed Hybrid Systems

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)
  $\forall i\, a(i) := \texttt{if .. then}\ a\ \texttt{else}\ -b\ \texttt{fi}$
- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \texttt{new}\ Car$



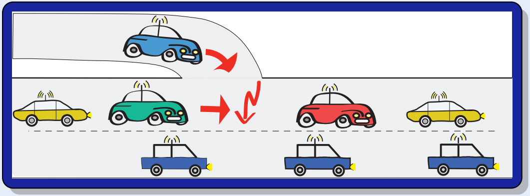
$\Rightarrow$ Communication

$d(i, \ell(i)) := d(i, \ell(i)) + 10$

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)
  $\forall i\, a(i) := \texttt{if} .. \texttt{then}\, a\, \texttt{else}\, -b\, \texttt{fi}$
- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \texttt{new}\ Car$

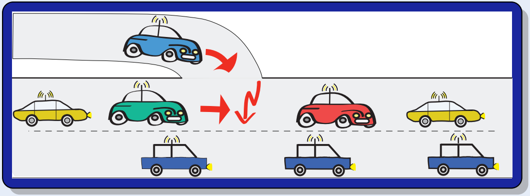


$\Rightarrow$ Communication
$\forall i\, d(i, \ell(i)) := d(i, \ell(i)) + 10$

# Model for Distributed Hybrid Systems

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)
  $\forall i \, a(i) := \mathtt{if} \, .. \, \mathtt{then} \, a \, \mathtt{else} \, -b \, \mathtt{fi}$
- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \mathtt{new} \, Car$

$\Rightarrow$ Communication
$\forall i \, d(i, \ell(i)) := d(i, \ell(i)) + 10$

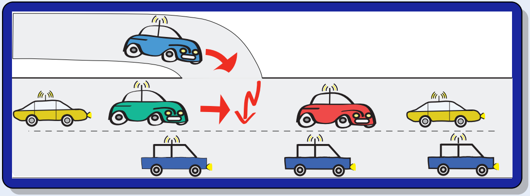
$\Rightarrow$ Discrete structural dynamics
$\ell(i) := \ell(\ell(i))$

# Model for Distributed Hybrid Systems

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)
  $\forall i\, a(i) := \mathtt{if}\, ..\, \mathtt{then}\, a\, \mathtt{else}\, -b\, \mathtt{fi}$
- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \mathtt{new}\, Car$



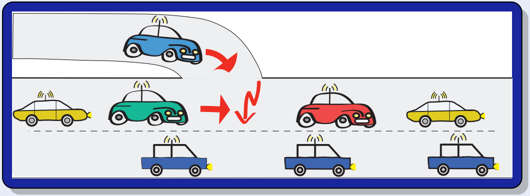
$\Rightarrow$ Communication
$\forall i\, d(i,\ell(i)) := d(i,\ell(i)) + 10$

$\Rightarrow$ Discrete structural dynamics
$\ell(i) := \ell(\ell(i))$

$\Rightarrow$ Continuous structural dynamics
$x(i)'' = a(i) + c(i,\ell(i))a(\ell(i))$

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)

$\forall i\, a(i) := \texttt{if}\ ..\ \texttt{then}\ a\ \texttt{else}\ -b\ \texttt{fi}$

- Structural dynamics (communication/coupling)
  $\ell(i) := carInFrontOf(i)$
- Dimensional dynamics (appearance)
  $n := \texttt{new}\ Car$

$\Rightarrow$ Communication
$\forall i\, d(i, \ell(i)) := d(i, \ell(i)) + 10$

$\Rightarrow$ Discrete structural dynamics
$\ell(i) := \ell(\ell(i))$

$\Rightarrow$ Continuous structural dynamics
$\forall i\, x(i)'' = a(i) + c(i, \ell(i))a(\ell(i))$

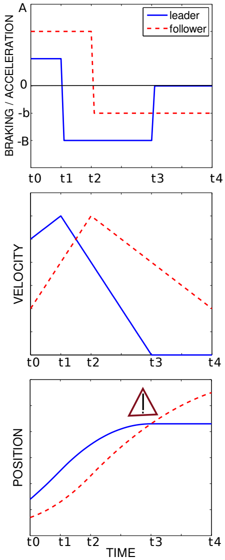
# Outline Background

## Challenge: Local lane dynamics

- A car controller for a differential equation respects separation of local lane.

## Challenge: Local lane dynamics

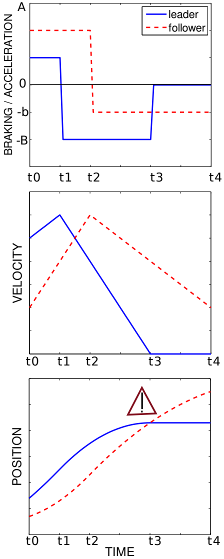
- A car controller for a differential equation respects separation of local lane.
- Follower car maintains safe distance to leader:



## Challenge: Local lane dynamics

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

$$f \ll \ell \rightarrow [(a_i := ctrl;\ x_i'' = a_i)^*]\, f \ll \ell$$

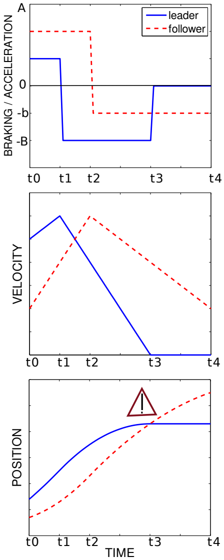


## Challenge: Local lane dynamics

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

$$f \ll \ell \rightarrow [(a_i := ctrl;\ x_i'' = a_i)^*]\, f \ll \ell$$

$$f \ll \ell \equiv (x_f \leq x_\ell) \land (f \neq \ell) \rightarrow (x_\ell > x_f + \frac{v_f^2}{2b} - \frac{v_\ell^2}{2B} \land x_\ell > x_f \land v_f \geq 0 \land v_\ell \geq 0)$$



## Challenge: Global lane dynamics

- All controllers for arbitrarily many differential equations respect separation globally on lane.

# Car Control: Global Lane Control Challenge

## Challenge: Global lane dynamics

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

# Car Control: Global Lane Control Challenge

## Challenge: Global lane dynamics

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

$$[(\forall i\, a(i) := ctrl;\ \forall i\, x(i)'' = a(i))^*]\, \forall i,j\ i \ll j$$

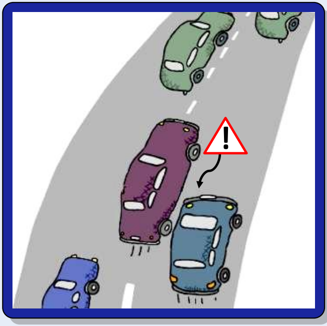
## Challenge: Local highway dynamics

- All controllers for arbitrarily many differential equations respect separation locally on highway.

# Car Control: Local Highway Control Challenge

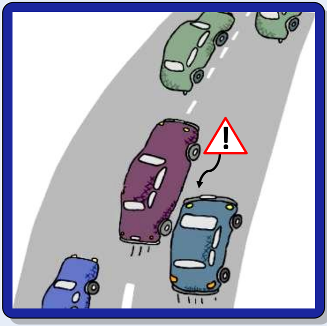
## Challenge: Local highway dynamics

- 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.



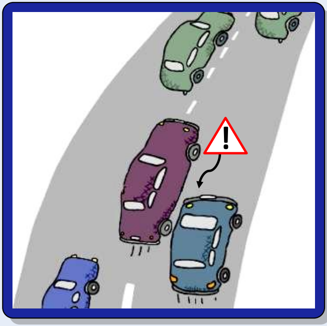
## Challenge: Local highway dynamics

- 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.



## 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.



$$[(n := \mathtt{new}\ C;\ \forall i\, a(i) := ctrl;\ \forall i\, x(i)'' = a(i))^*]\, \forall i,j\, i \ll j$$

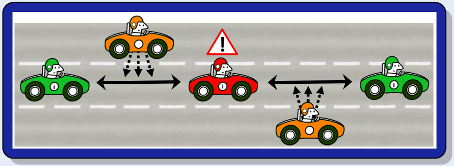
# Car Control: Global Highway Control Challenge

## Challenge: Global highway dynamics

- All controllers for arbitrarily many differential equations respect separation globally on highway.

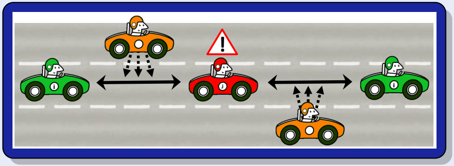
## Challenge: Global highway dynamics

- 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.



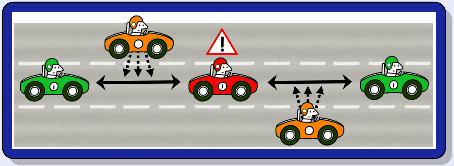
## Challenge: Global highway dynamics

- 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.



## 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.



$$[\forall l\,(n := \mathtt{new}\ C;\ \forall i\, a(i) := ctrl;\ \forall i\, x(i)'' = a(i))^*]\,\forall l\,\forall i,j\ i \ll j$$

# Outline Background

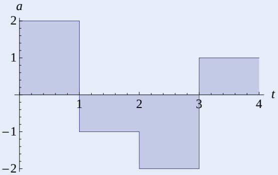
Q: I want to verify trains

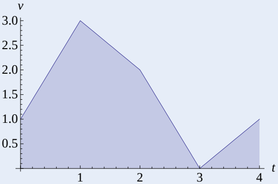
## Challenge

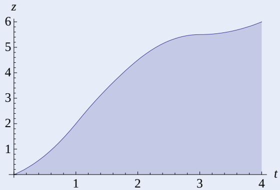
Q: I want to verify trains A: Hybrid systems

## Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)

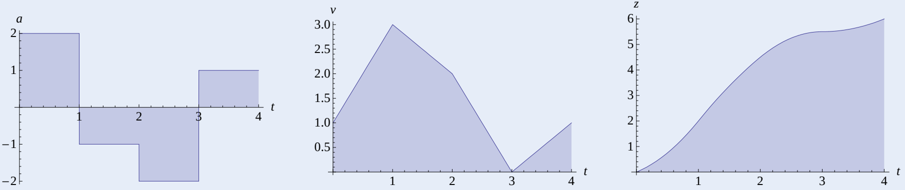



Q: I want to verify trains A: Hybrid systems Q: But there's uncertainties!

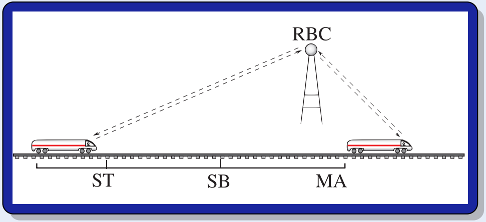
## Challenge (Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)

Q: I want to verify uncertain trains

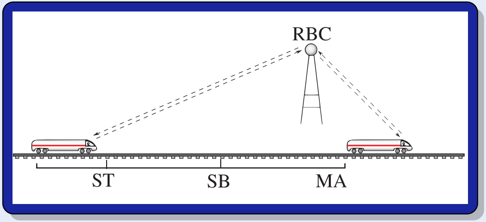
## Challenge

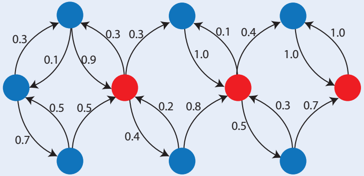


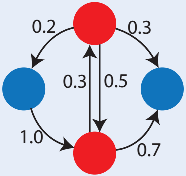
Q: I want to verify uncertain trains A: Markov chains

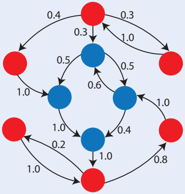
## Challenge (Probabilistic Systems)

- Directed graph
  (Countable state space)
- Weighted edges
  (Transition probabilities)



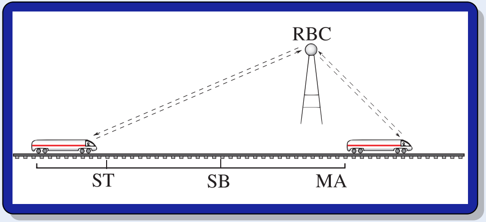



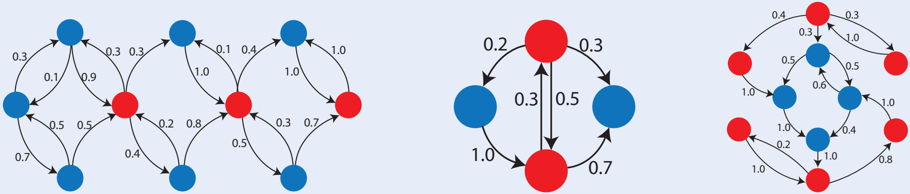



Q: I want to verify uncertain trains A: Markov chains Q: But trains move!

## Challenge (Probabilistic Systems)

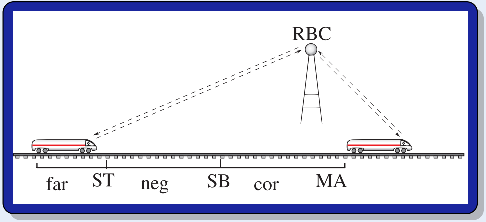
- Directed graph
  (Countable state space)
- Weighted edges
  (Transition probabilities)



Q: I want to verify uncertain systems

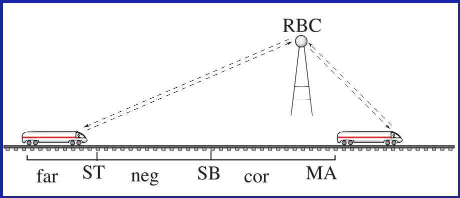
## Challenge



Q: I want to verify uncertain systems A: Stochastic hybrid systems

## Challenge (Stochastic Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Stochastic dynamics (uncertainty)

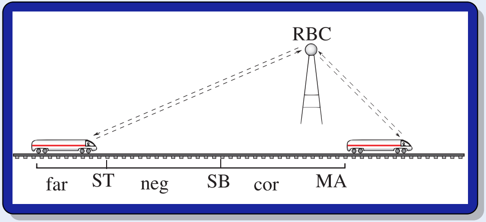


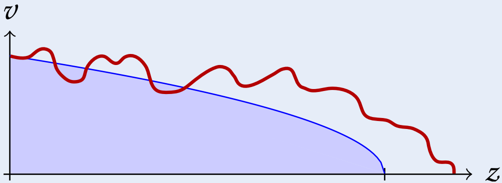
Q: I want to verify uncertain systems A: Stochastic hybrid systems

## Challenge (Stochastic Hybrid Systems)

- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Stochastic dynamics (uncertainty)

- Discrete stochastic (lossy communication)
- Continuous stochastic (wind, track)



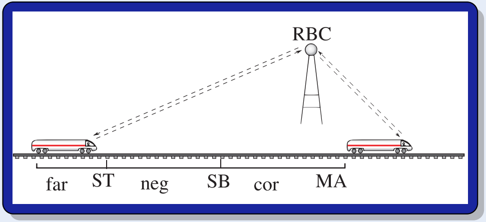



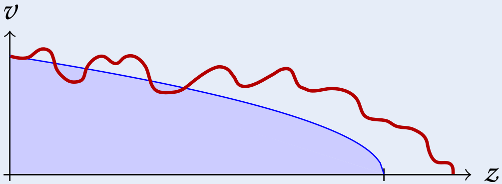
Q: I want to verify uncertain systems A: Stochastic hybrid systems Q: How?

## Challenge (Stochastic Hybrid Systems)

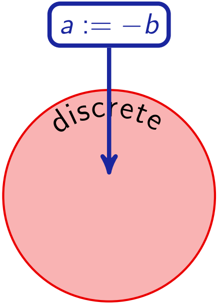
- Continuous dynamics (differential equations)
- Discrete dynamics (control decisions)
- Stochastic dynamics (uncertainty)

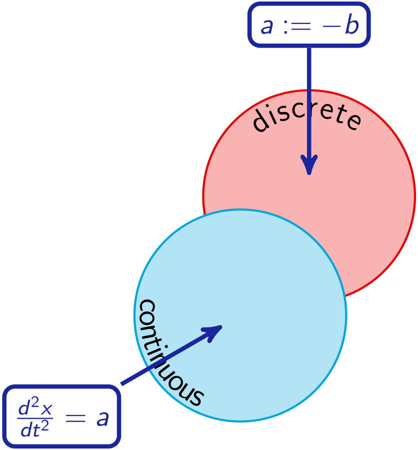
- Discrete stochastic (lossy communication)
- Continuous stochastic (wind, track)



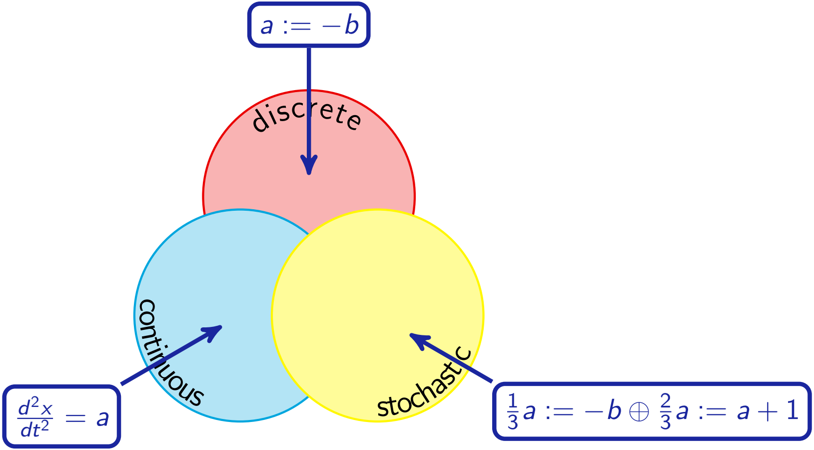
# Model for Stochastic Hybrid Systems



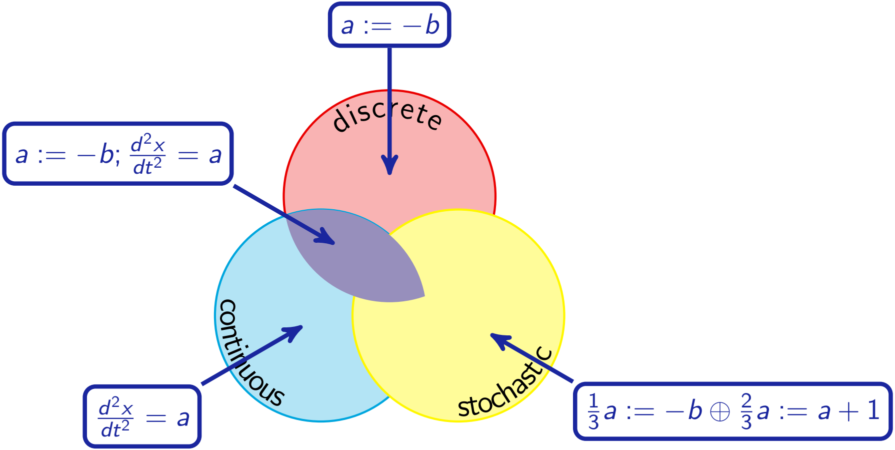


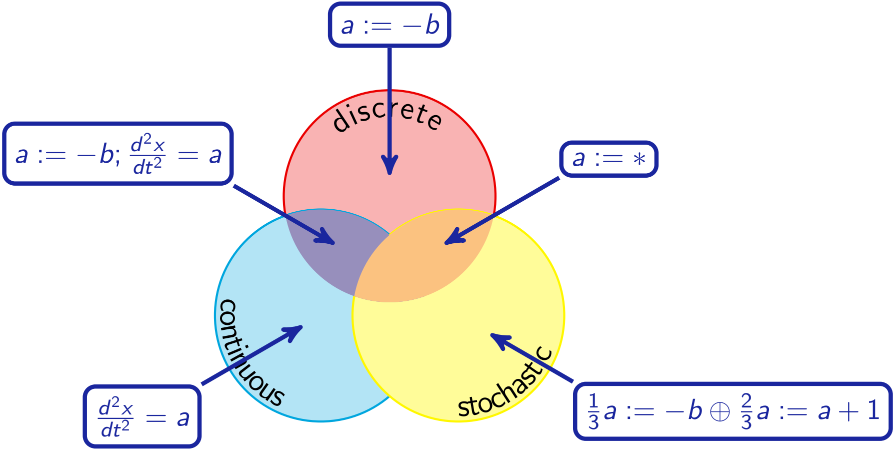
$a := -b$
discrete
continuous
$\frac{d^2x}{dt^2} = a$

# Model for Stochastic Hybrid Systems



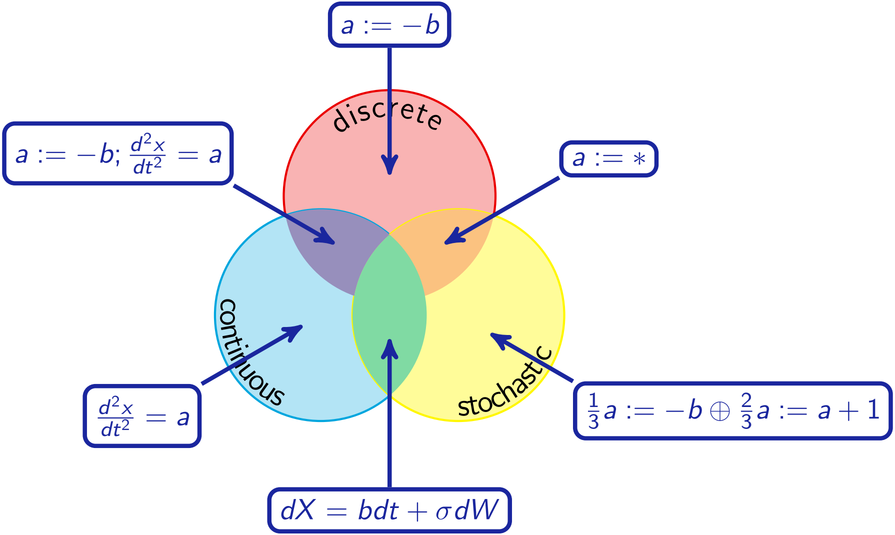
# Model for Stochastic Hybrid Systems



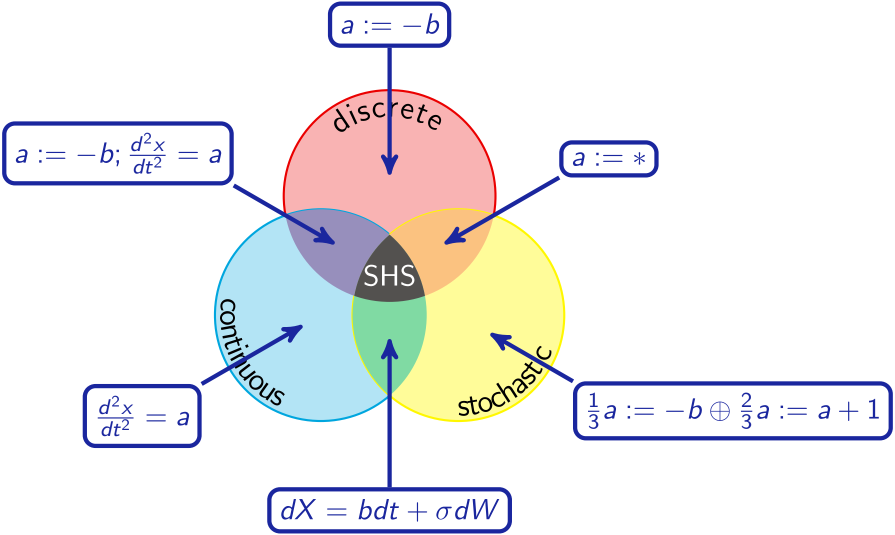


$a := -b$
discrete
$a := -b;\ \frac{d^2x}{dt^2} = a$
$a := *$
continuous
stochastic
$\frac{d^2x}{dt^2} = a$
$\frac{1}{3} a := -b \oplus \frac{2}{3} a := a + 1$

# Model for Stochastic Hybrid Systems

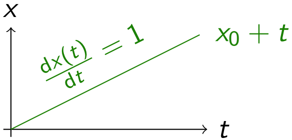


# Model for Stochastic Hybrid Systems



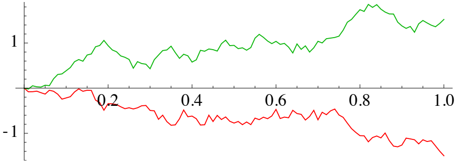
## Definition (Ordinary differential equation (ODE))

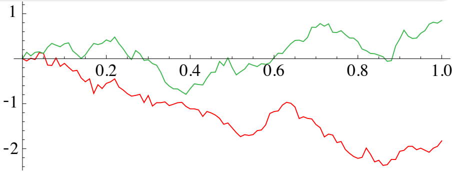
$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = b(x(t)) \quad x(0) = x_0$$




## Definition (Itō stochastic differential equation (SDE))

$$dX_t = b(X_t)dt + \sigma(X_t)dW_t \quad X_0 = Z$$

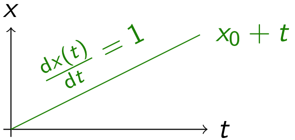


# Stochastic Differential Equations (SDE)

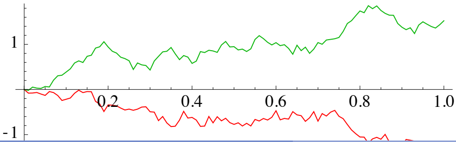
**Definition (Ordinary differential equation (ODE))**

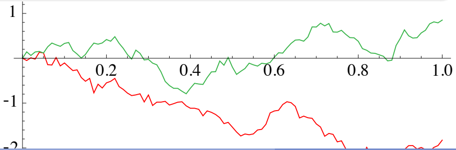
$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = b(x(t)) \quad x(0) = x_0$$




**Definition (Itō stochastic differential equation (SDE))**

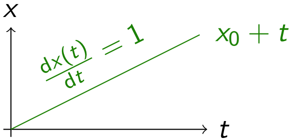
$$X_s = Z + \int_0^s dX_t = Z + \int_0^s b(X_t)dt + \int_0^s \sigma(X_t)dW_t$$



## Definition (Ordinary differential equation (ODE))

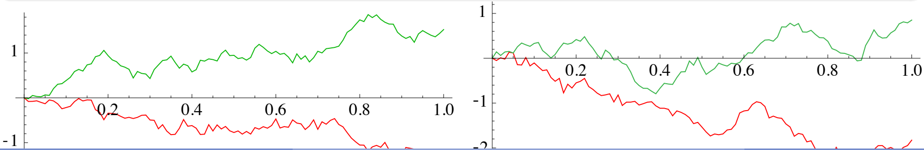
$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = b(x(t)) \quad x(0) = x_0$$




Calculus

## Definition (Itō stochastic differential equation (SDE))

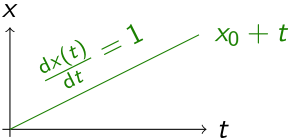
$$X_s = Z + \int_0^s dX_t = Z + \int_0^s b(X_t)dt + \int_0^s \sigma(X_t)dW_t$$

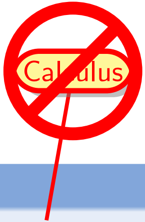


# Stochastic Differential Equations (SDE)

**Definition (Ordinary differential equation (ODE))**

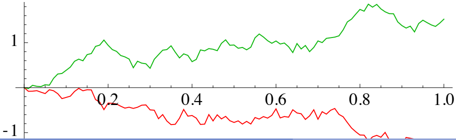
$$\frac{dx(t)}{dt} = b(x(t)) \quad x(0) = x_0$$

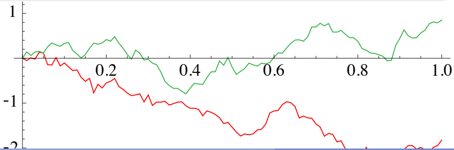



**Definition (Itō stochastic differential equation (SDE))**

$$X_s = Z + \int_0^s dX_t = Z + \int_0^s b(X_t)dt + \int_0^s \sigma(X_t)dW_t$$



# Stochastic Hybrid Programs: Syntax

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

| | | |
|---|---|---|
| $x := \theta$ | (assignment) | jump & test |
| $x := *$ | (random assignment) | jump & test |
| $?H$ | (conditional execution) | jump & test |
| $dx = b dt + \sigma dW \,\&\, H$ | (SDE) | |
| $\alpha; \beta$ | (seq. composition) | algebra |
| $\lambda\alpha \oplus \nu\beta$ | (convex combination) | algebra |
| $\alpha^*$ | (nondet. repetition) | algebra |