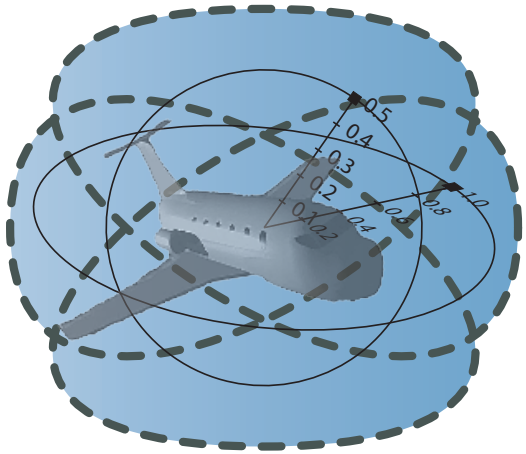
15-819/18-879: Logical Analysis of Hybrid Systems

11: Logic & Hybrid Programs

André Platzer

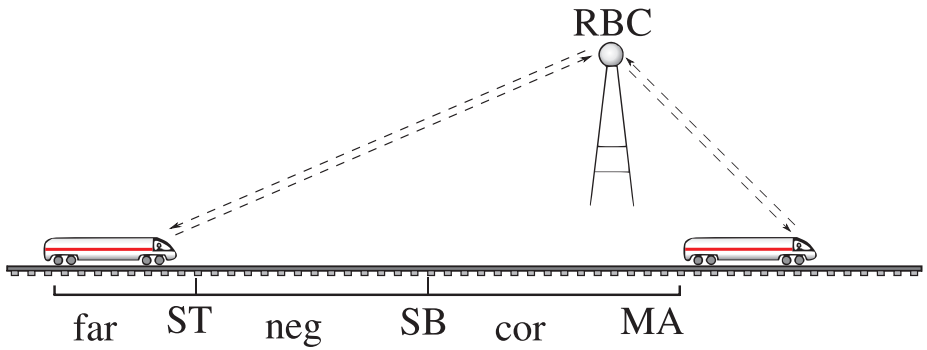
aplatzer@cs.cmu.edu
Carnegie Mellon University, Pittsburgh, PA



Outline

Outline

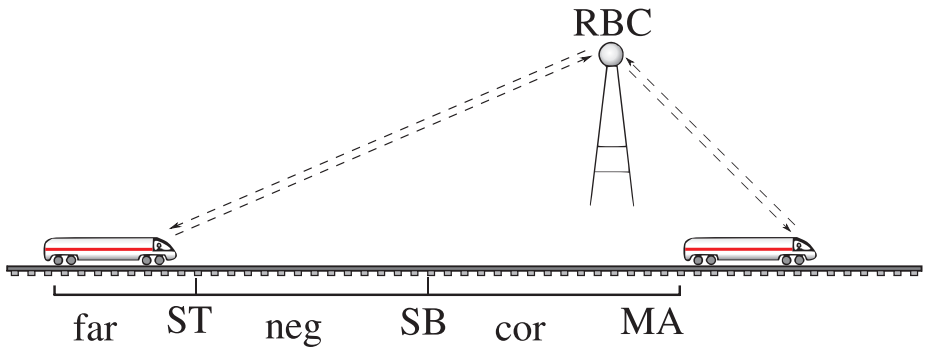
Verifying Parametric Hybrid Systems



ETCS objectives:

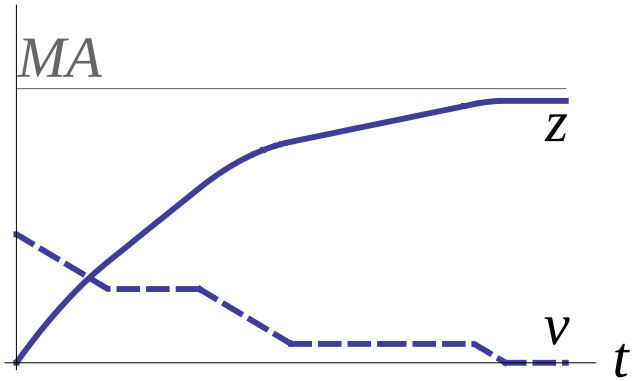
1. Collision free
2. Maximise throughput & velocity (300 km/h)
3. $2.1 * 10^6$ passengers/day

Verifying Parametric Hybrid Systems

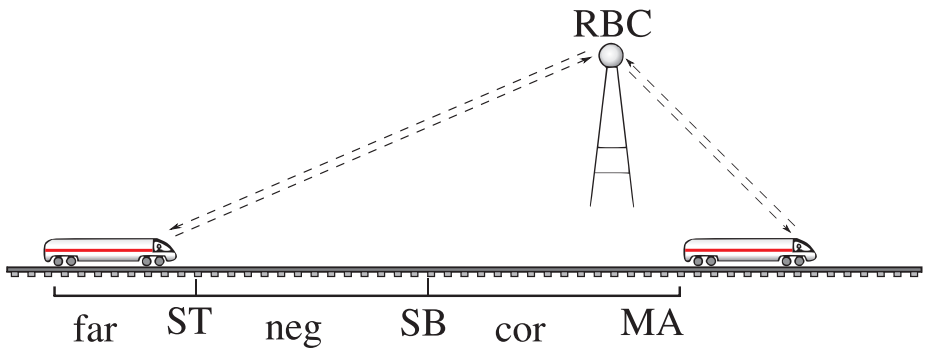


Parametric Hybrid Systems

continuous evolution along differential equations + discrete change

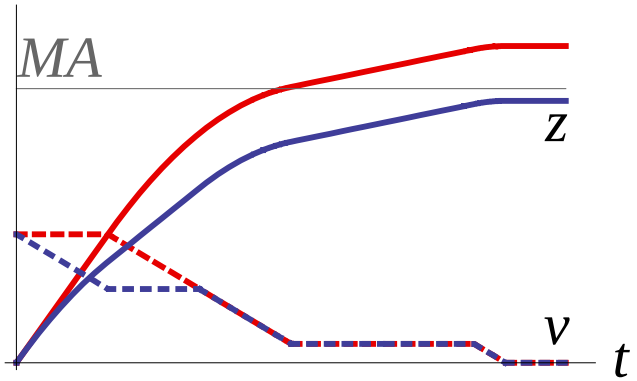


Verifying Parametric Hybrid Systems

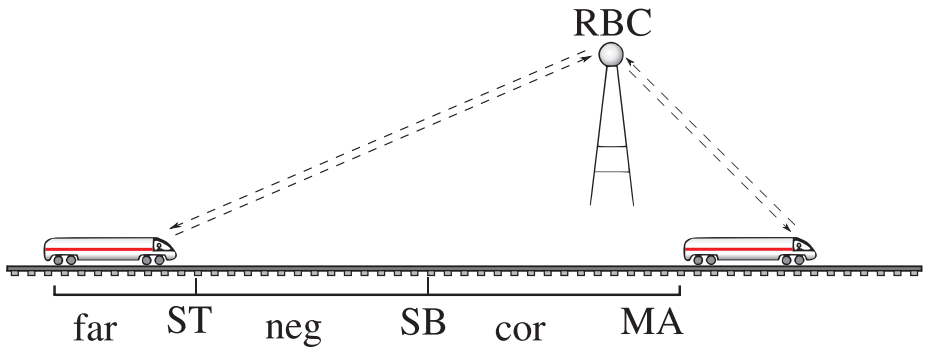


Parametric Hybrid Systems

continuous evolution along differential equations + discrete change

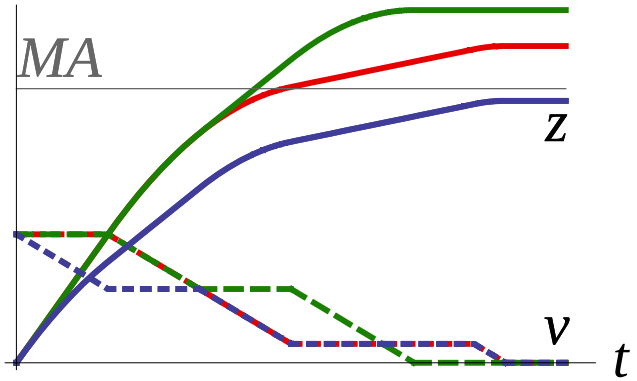


Verifying Parametric Hybrid Systems

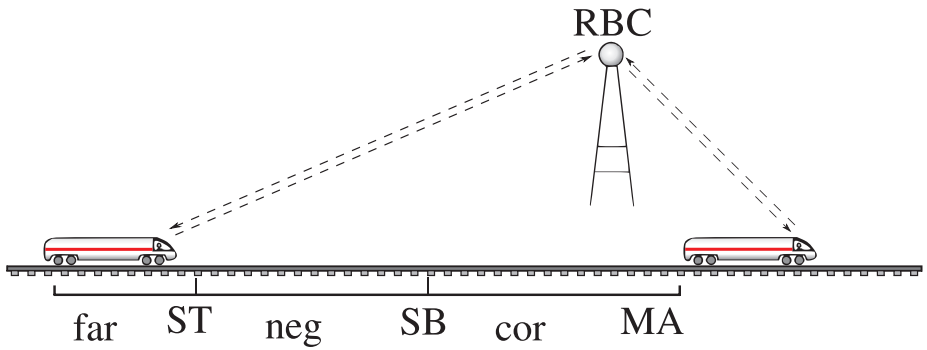


Parametric Hybrid Systems

continuous evolution along differential equations + discrete change



Verifying Parametric Hybrid Systems

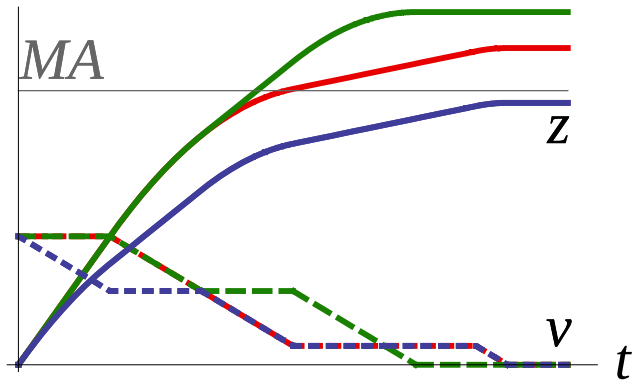


Parametric Hybrid Systems

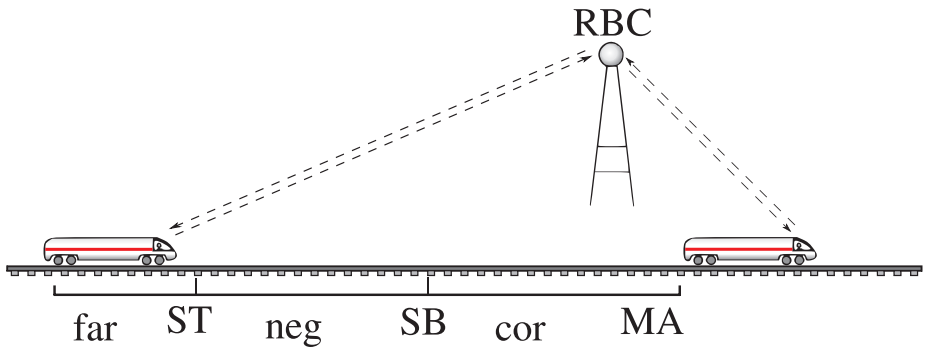
continuous evolution along differential equations + discrete change

- Parameters have nonlinear influence
- Handle SB as free symbolic parameter?
- Challenge: verification (falsifying is easier)
- Which constraints for SB?

$\forall MA\, \exists SB$ "train always safe"

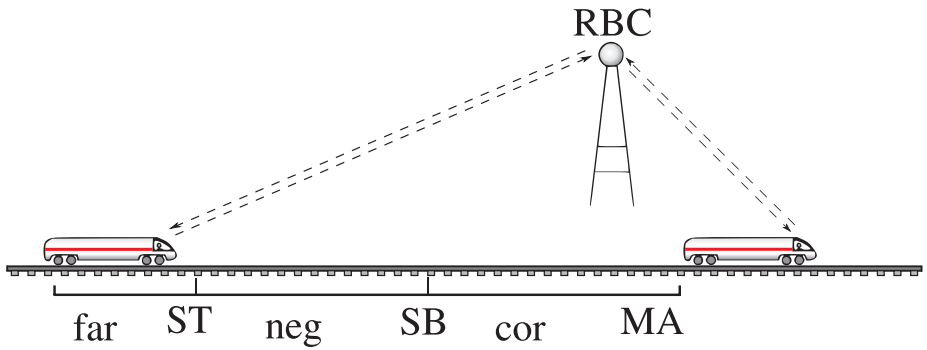


Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓

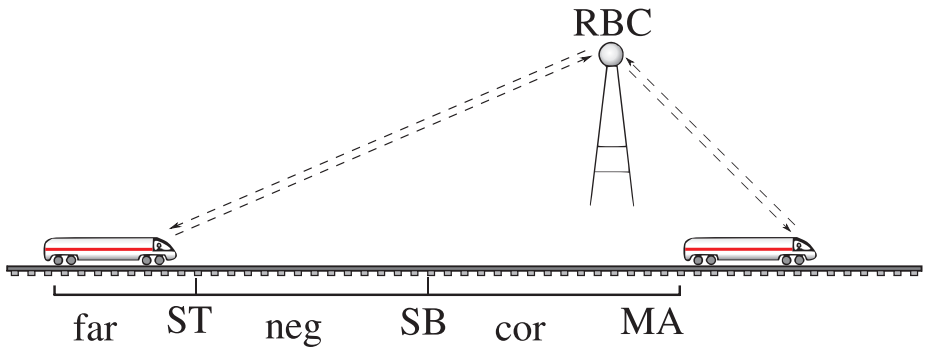
Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓

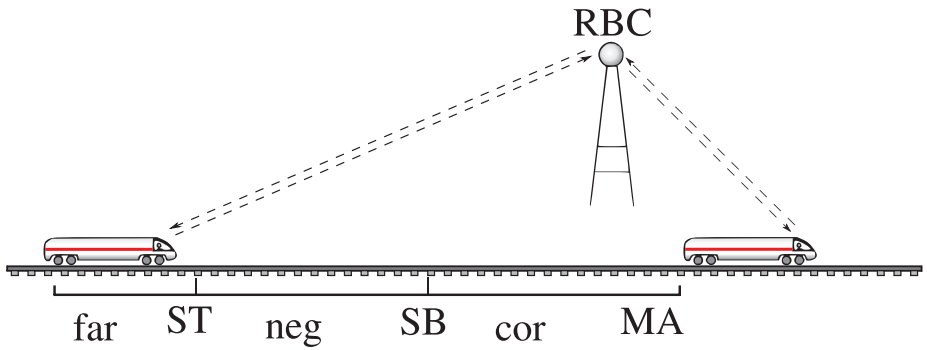
- × no finite-state bisimulation for HS
- × no general handling of free parameters
- × with parameters, everything gets nonlinear!

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathsf{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

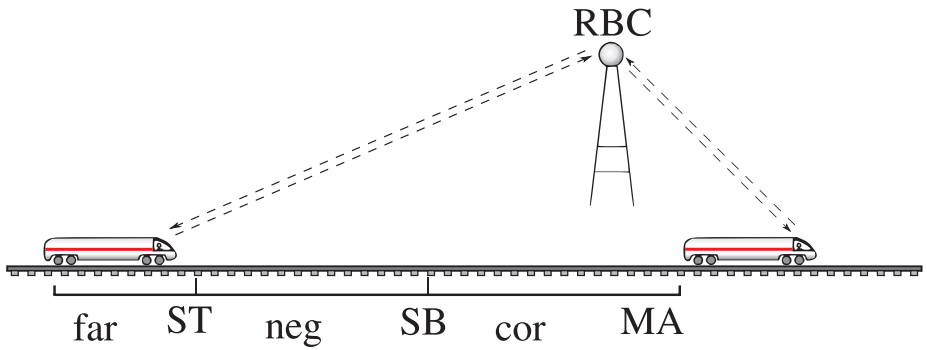
Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathsf{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×

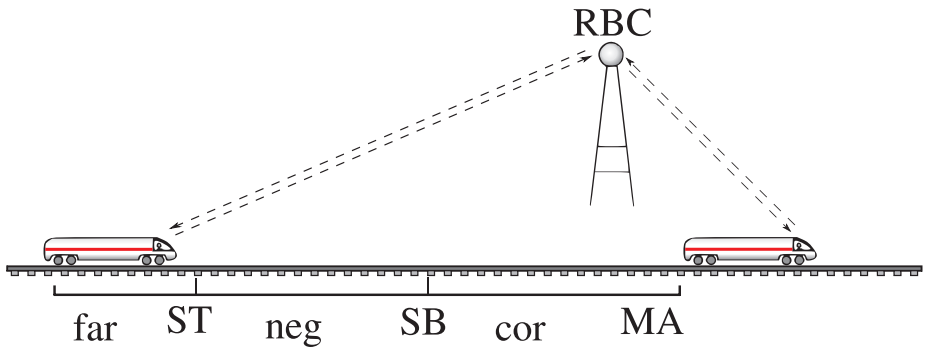
- × declaratively axiomatise operational model
- × expressiveness for characterisation?
- × automation

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathsf{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

Verification Approaches for Hybrid Systems



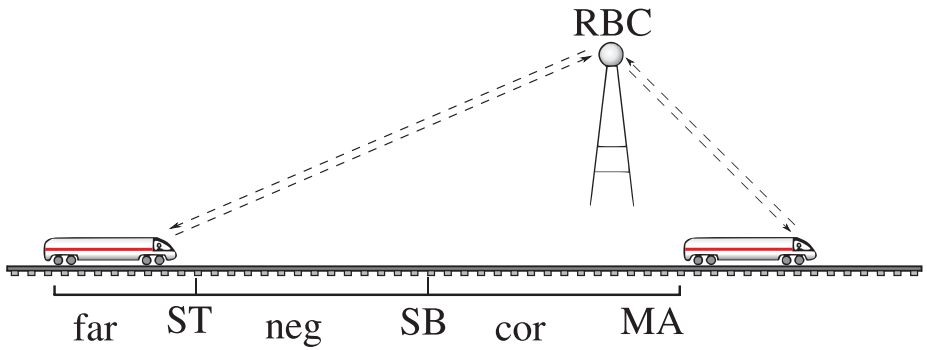
problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

✓ $[RBC]\mathrm{partitioned} \rightarrow \exists SB\, \langle \mathrm{Train} \rangle [RBC]\mathrm{safe}$

× intermediate states

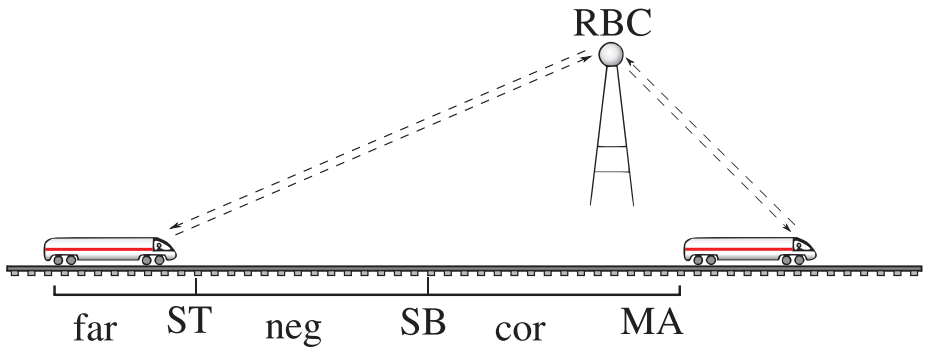
× automation

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathsf{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	×

Verification Approaches for Hybrid Systems



problem	technique	Op	Par	T	CI	Aut
$ETCS \models z < MA$	TL-MC	✓	×	✓	×	✓
$\models (\mathrm{Ax}(ETCS) \rightarrow z < MA)$	TL-calculus	×	×	✓	..	×
$\models [ETCS]\, z < MA$	DL-calculus	✓	✓	×	✓	?

differential dynamic logic

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

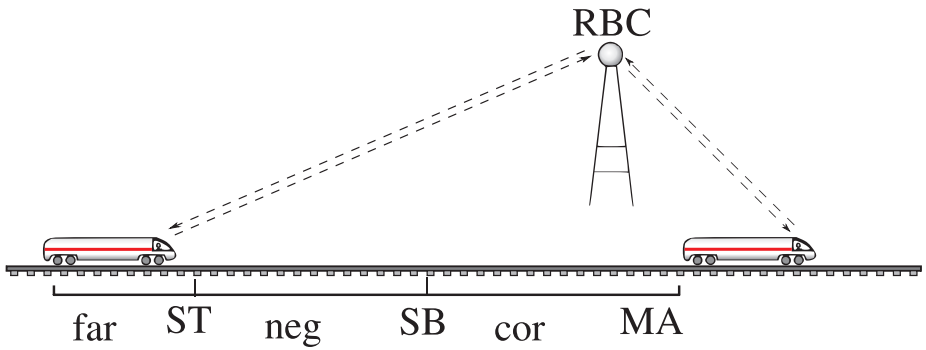
Outline

Outline

dℒ Design: The Logic of Hybrid Systems

differential dynamic logic

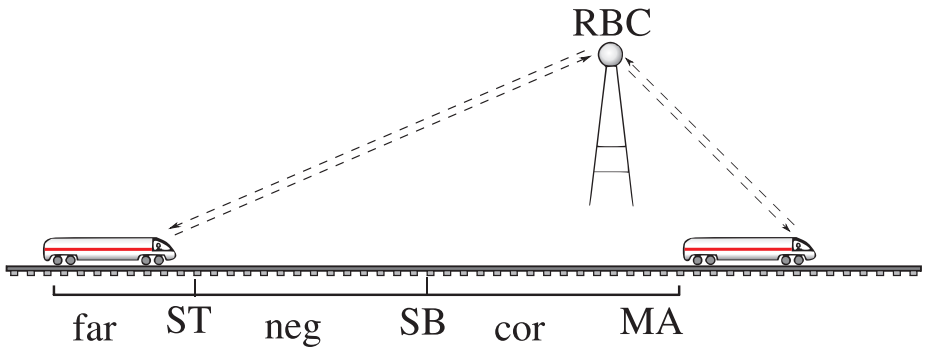
$$\mathsf{d}\mathcal{L} = \qquad \mathsf{DL} + \mathsf{HP}$$

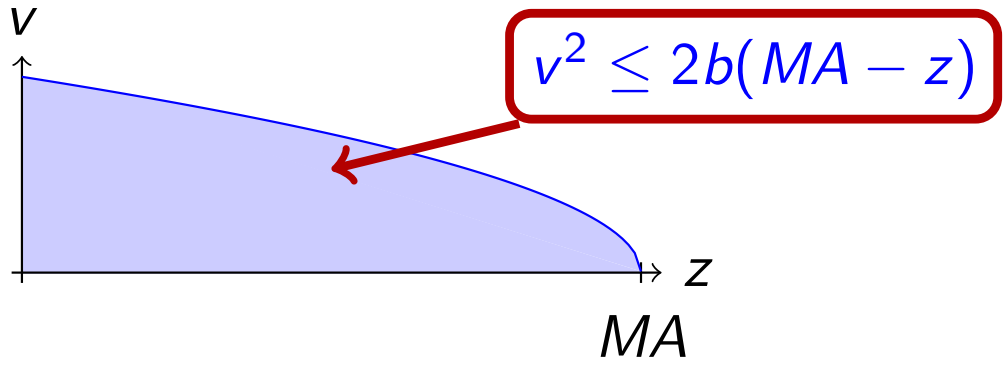


dℒ Design: Regions in First-Order Logic

differential dynamic logic

$d\mathcal{L} = \text{FOL}_{\mathbb{R}}$

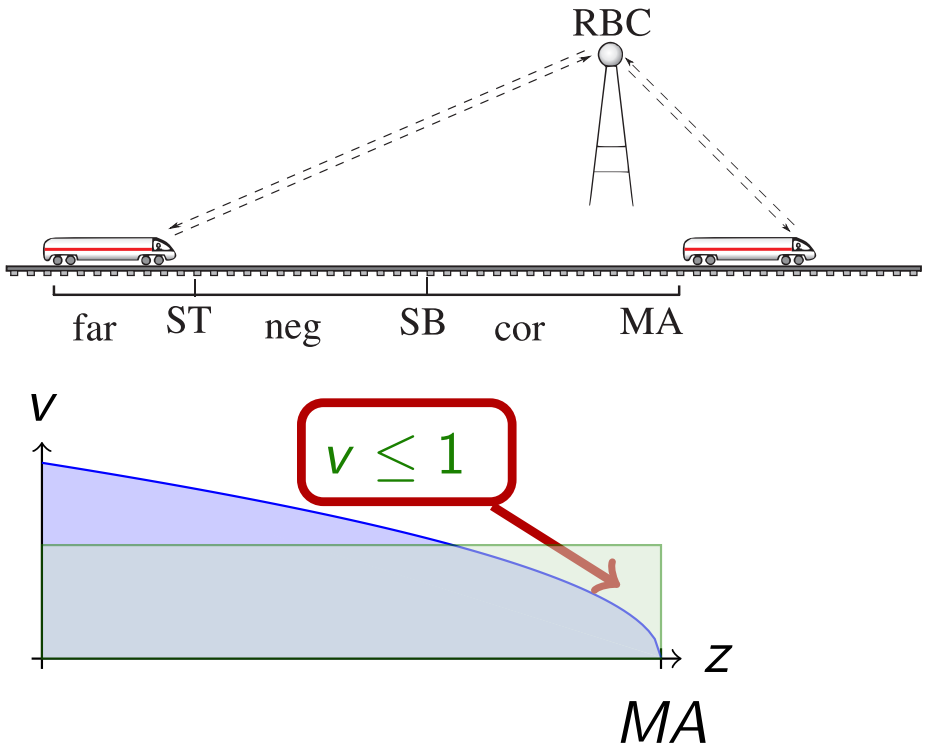




dℒ Design: Regions in First-Order Logic

differential dynamic logic

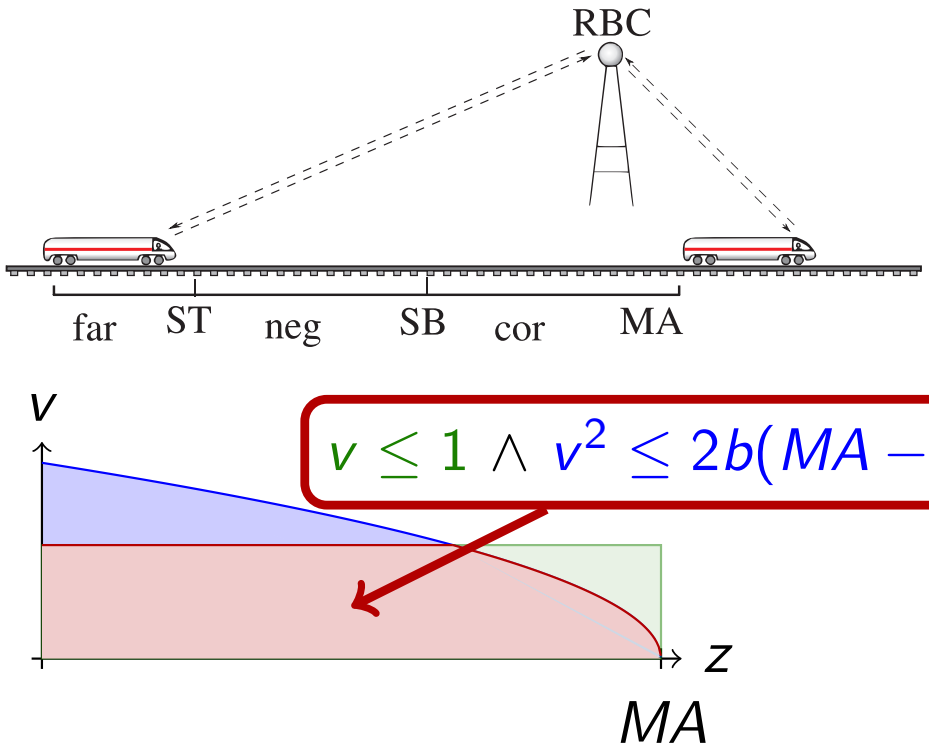
$$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}}$$



dℒ Design: Regions in First-Order Logic

differential dynamic logic

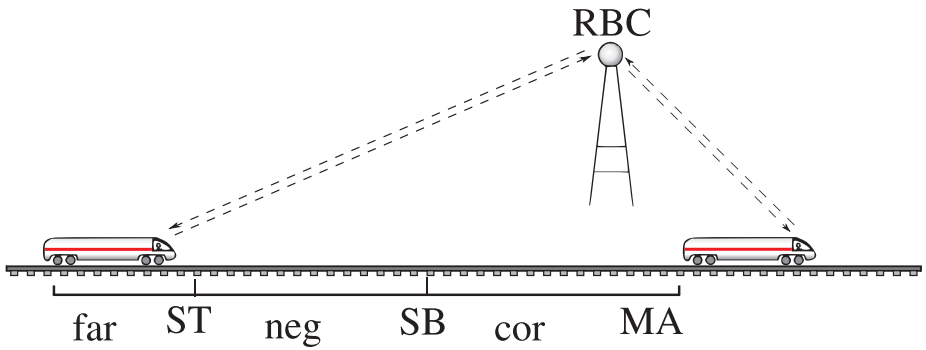
$d\mathcal{L} = \text{FOL}_{\mathbb{R}}$

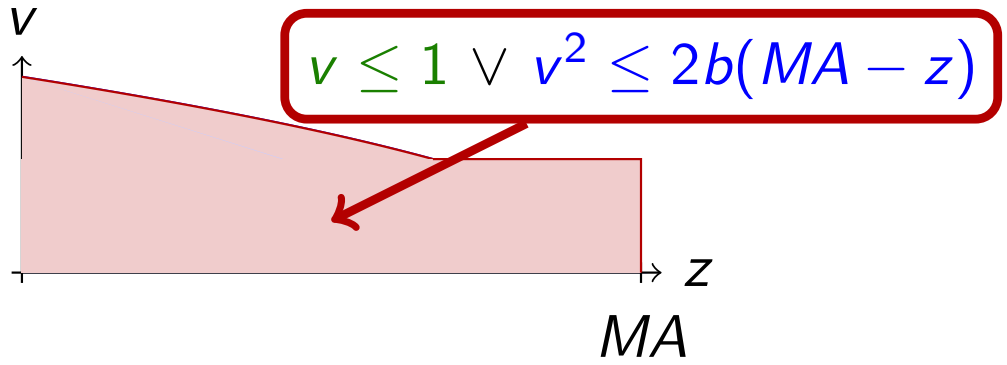


dℒ Design: Regions in First-Order Logic

differential dynamic logic

$$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}}$$





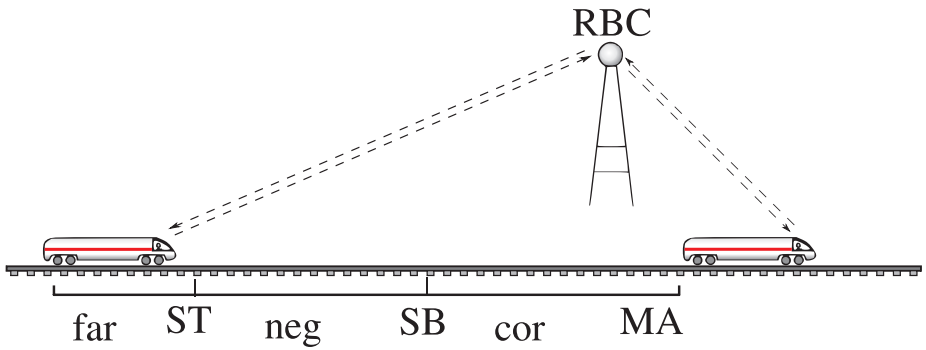
dℒ Design: Regions in First-Order Logic

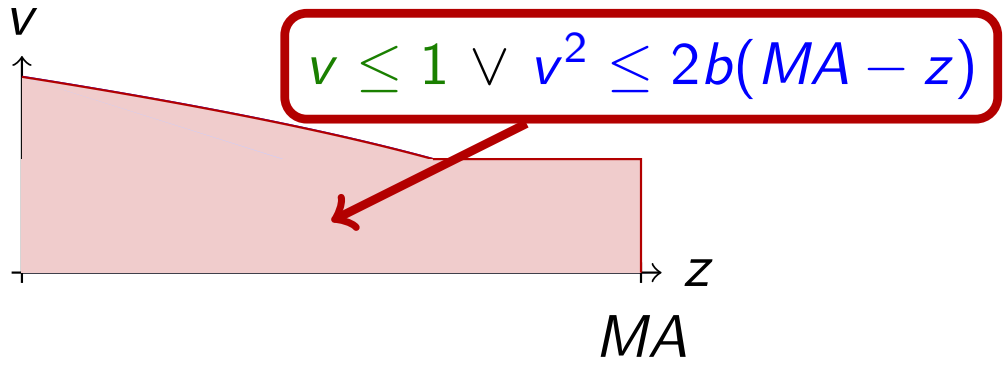
differential dynamic logic

$d\mathcal{L} = \text{FOL}_{\mathbb{R}}$

$\forall MA\, \exists SB\ \ldots$

$\forall t{\geq}0\ \ldots$

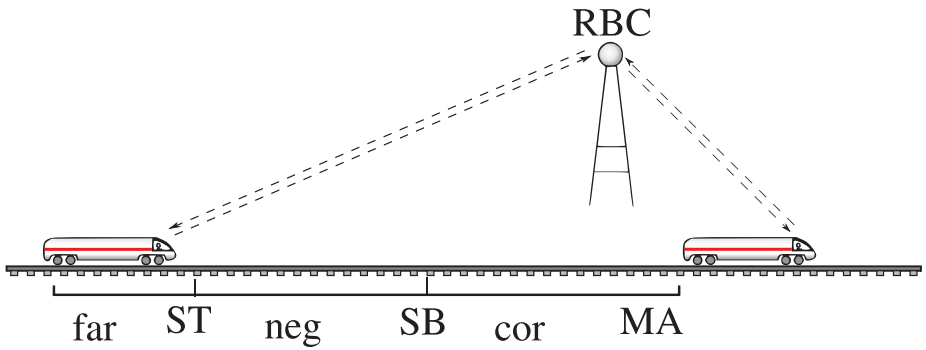


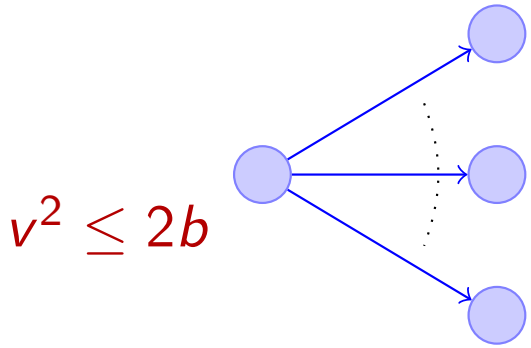


dℒ Design: State Transitions in Dynamic Logic

differential dynamic logic

$$d\mathcal{L} = \mathsf{FOL}_{\mathbb{R}} +$$

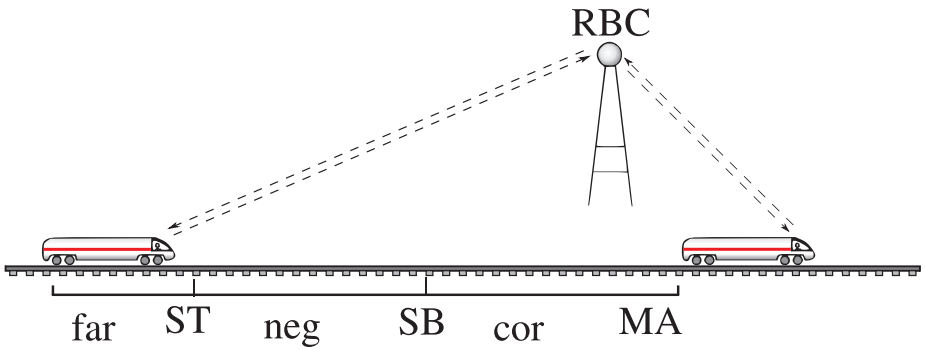


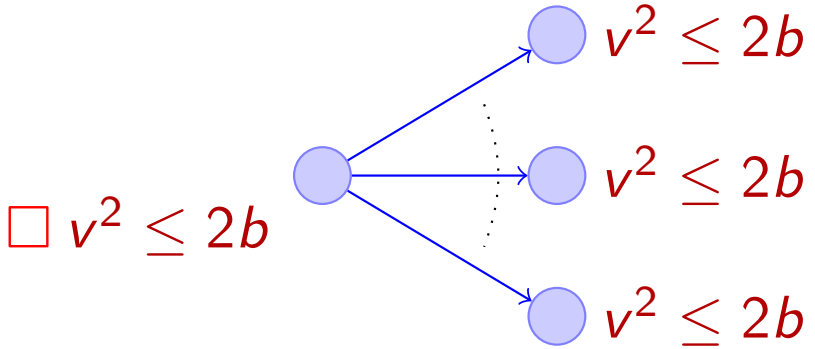


dℒ Design: State Transitions in Dynamic Logic

differential dynamic logic

$$\mathsf{d}\mathcal{L} = \mathsf{FOL}_{\mathbb{R}} + \mathsf{ML}$$

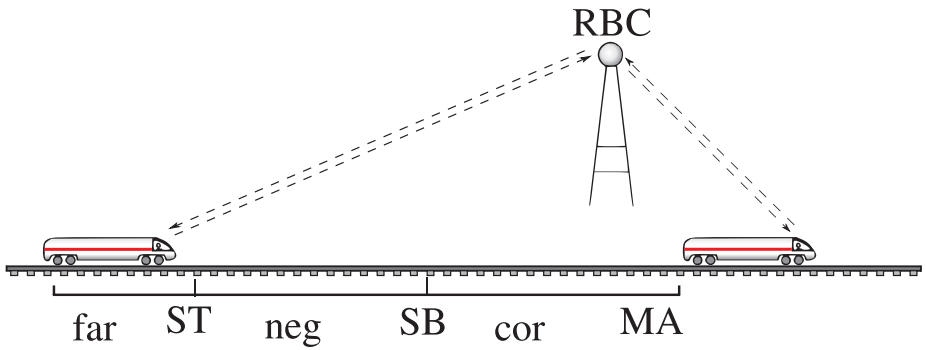


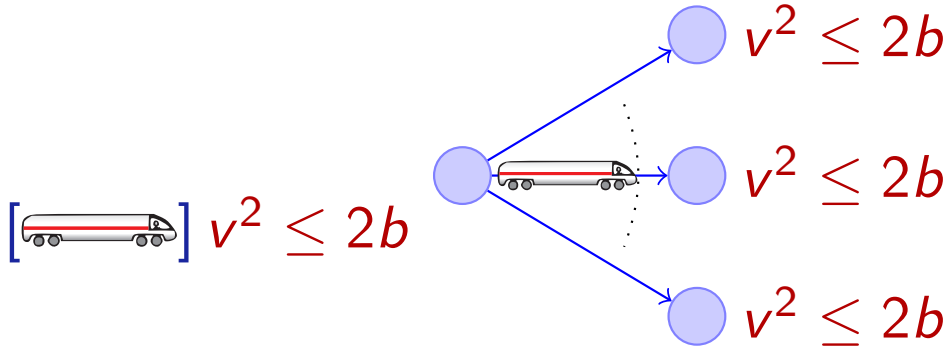


d$\mathcal{L}$ Design: State Transitions in Dynamic Logic

differential dynamic logic

$$\mathrm{d}\mathcal{L} = \mathrm{FOL}_{\mathbb{R}} + \mathrm{DL}$$

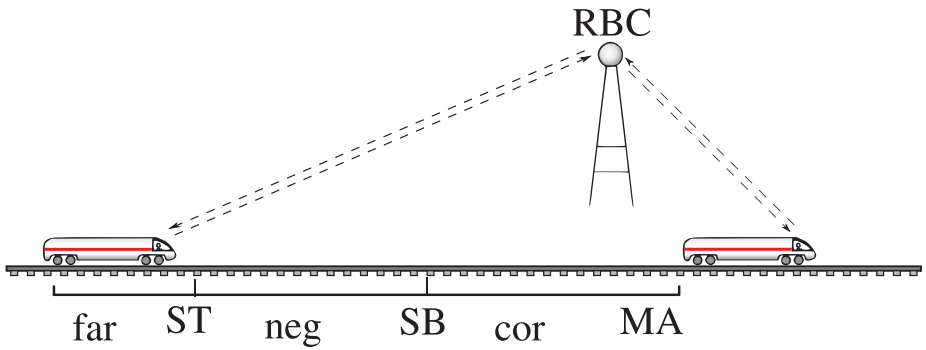




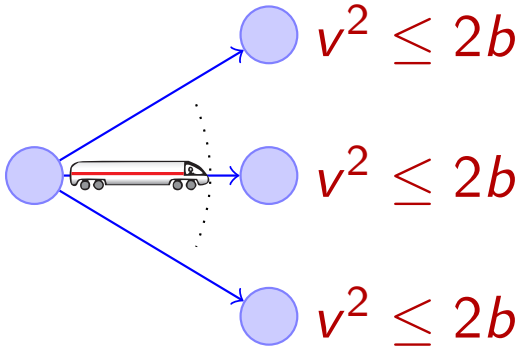
dℒ Design: Hybrid Programs as Uniform Model

differential dynamic logic

$$\mathsf{d}\mathcal{L} = \mathsf{FOL}_{\mathbb{R}} + \mathsf{DL} + \mathsf{HP}$$



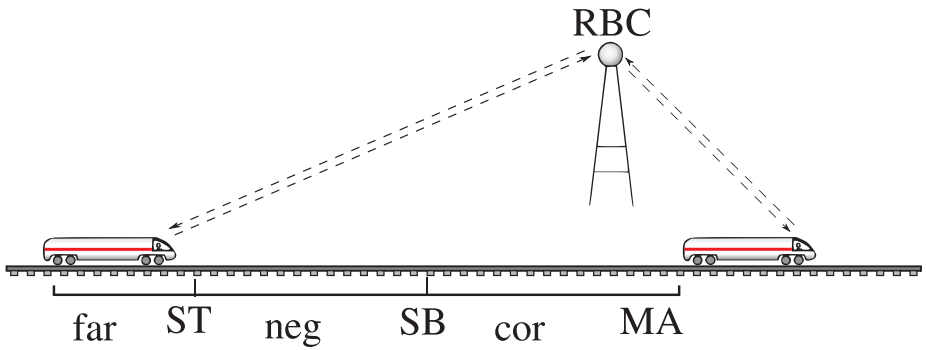
$$[z'' = a]\, v^2 \leq 2b$$



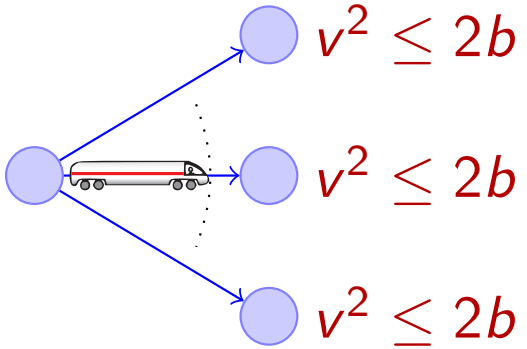
dℒ Design: Hybrid Programs as Uniform Model

differential dynamic logic

$$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$$



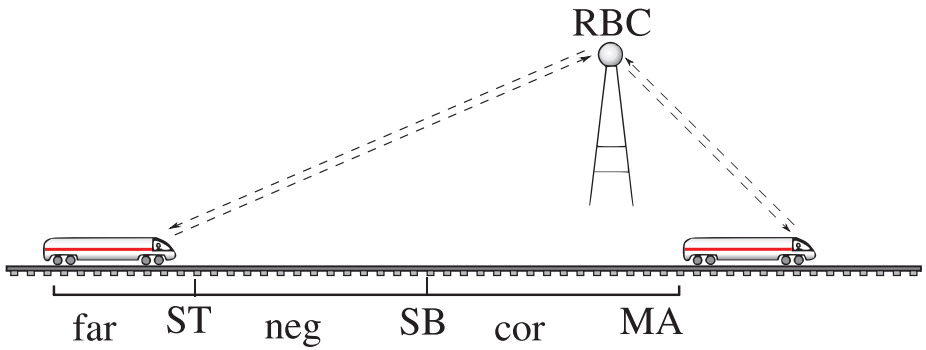
$$[\texttt{if}(z > SB)\, a := -b;\ z'' = a]\, v^2 \leq 2b$$



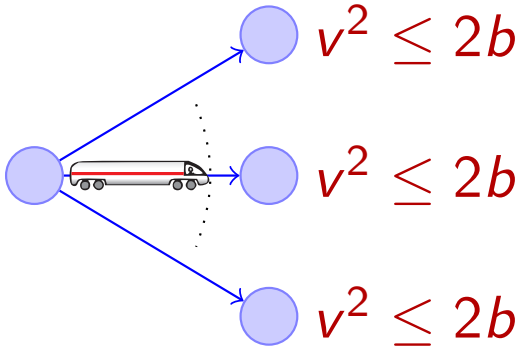
dℒ Design: Hybrid Programs as Uniform Model

differential dynamic logic

$$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$$



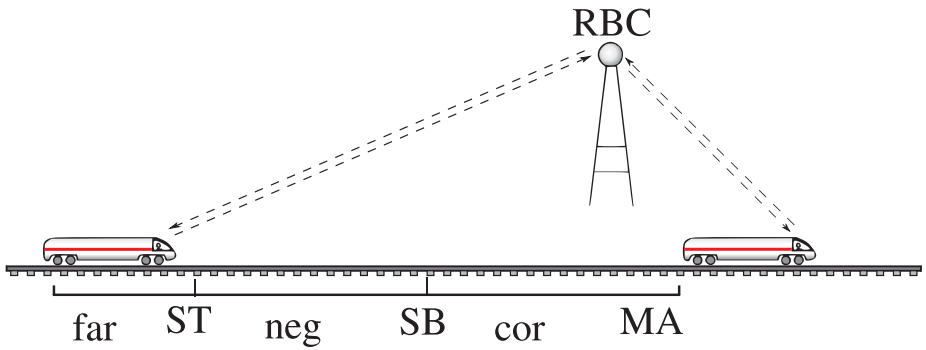
$$[\underbrace{\texttt{if}(z > SB)\, a := -b;\ z'' = a}_{\text{hybrid program}}]\, v^2 \leq 2b$$

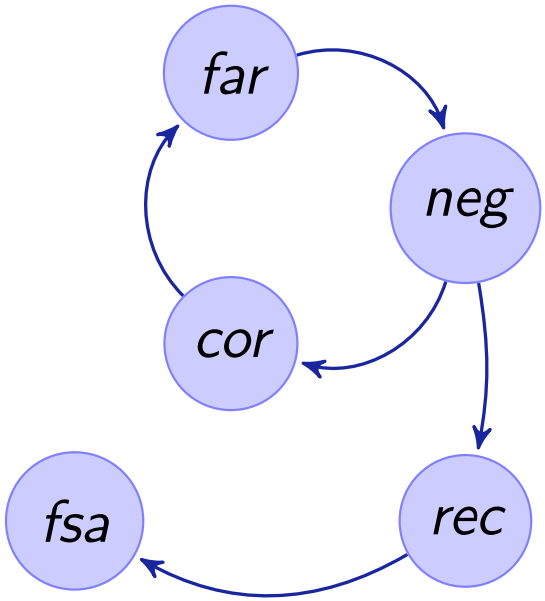


dℒ Design: What about Hybrid Automata?

differential dynamic logic

$$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$$



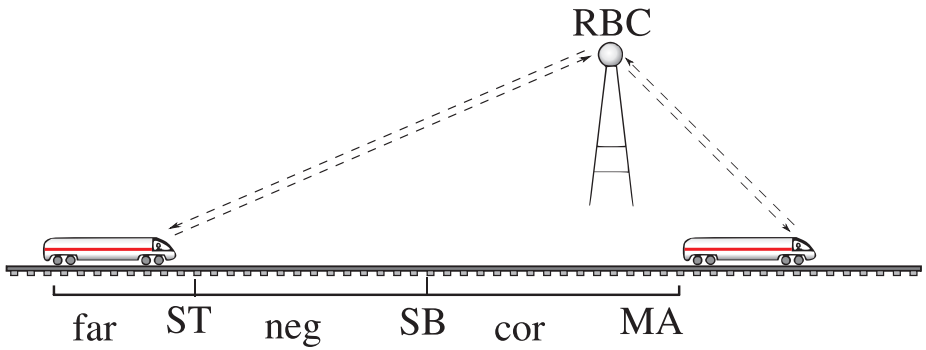


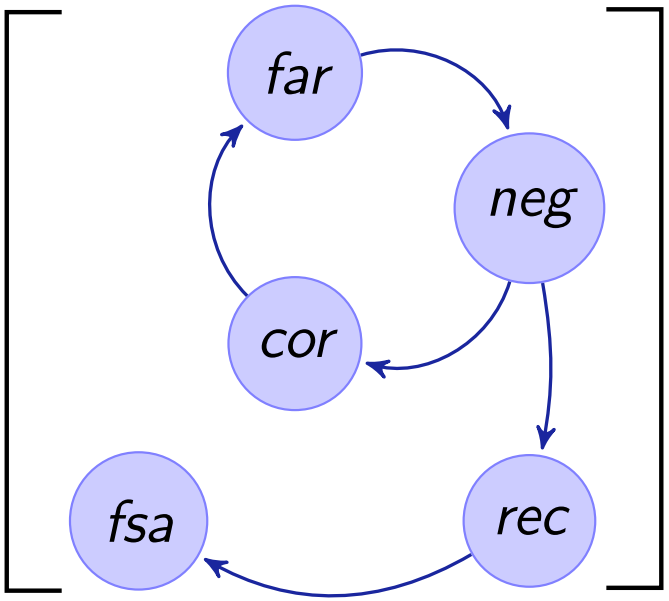
How about hybrid automata?

dℒ Design: What about Hybrid Automata?

differential dynamic logic

$$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$$



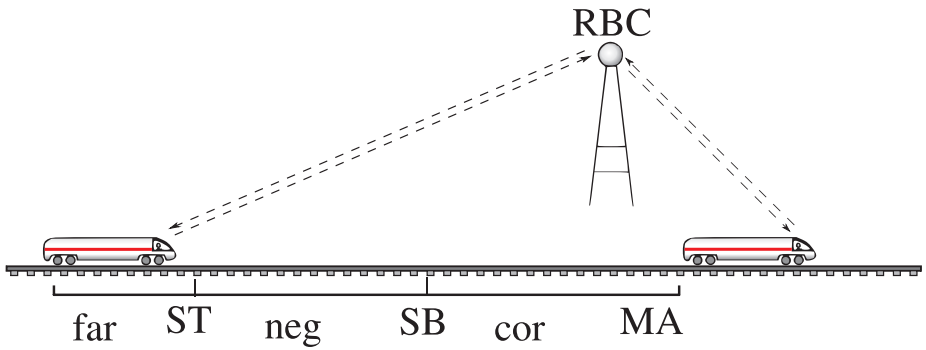


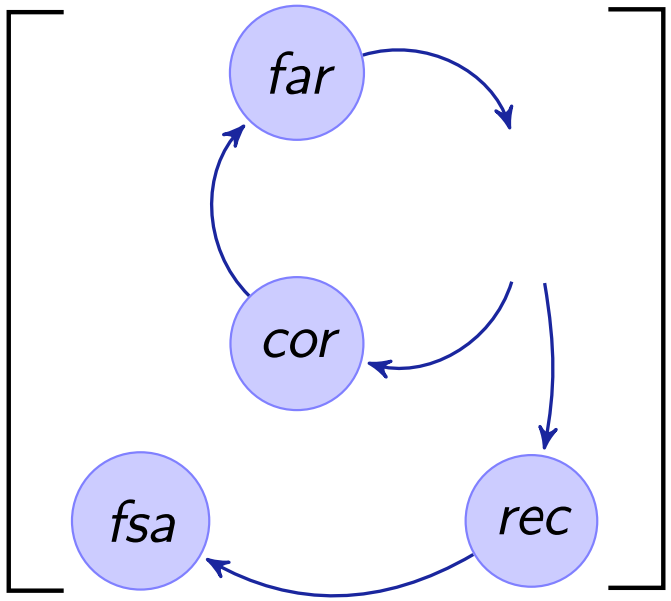
$$v^2 \leq 2b\,..$$

dℒ Design: What about Hybrid Automata?

differential dynamic logic

$$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$$



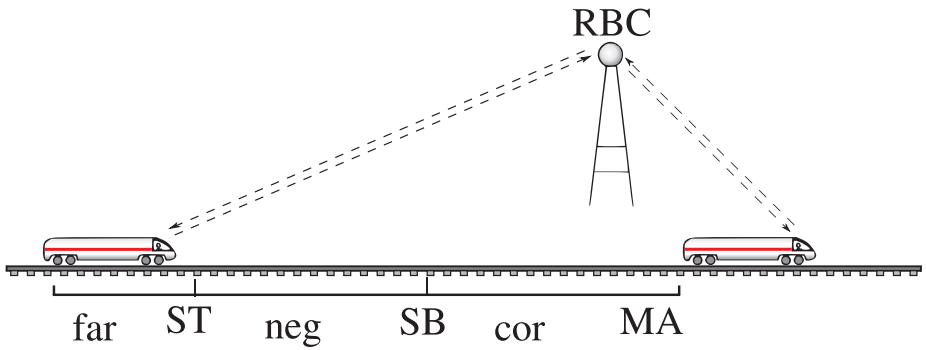


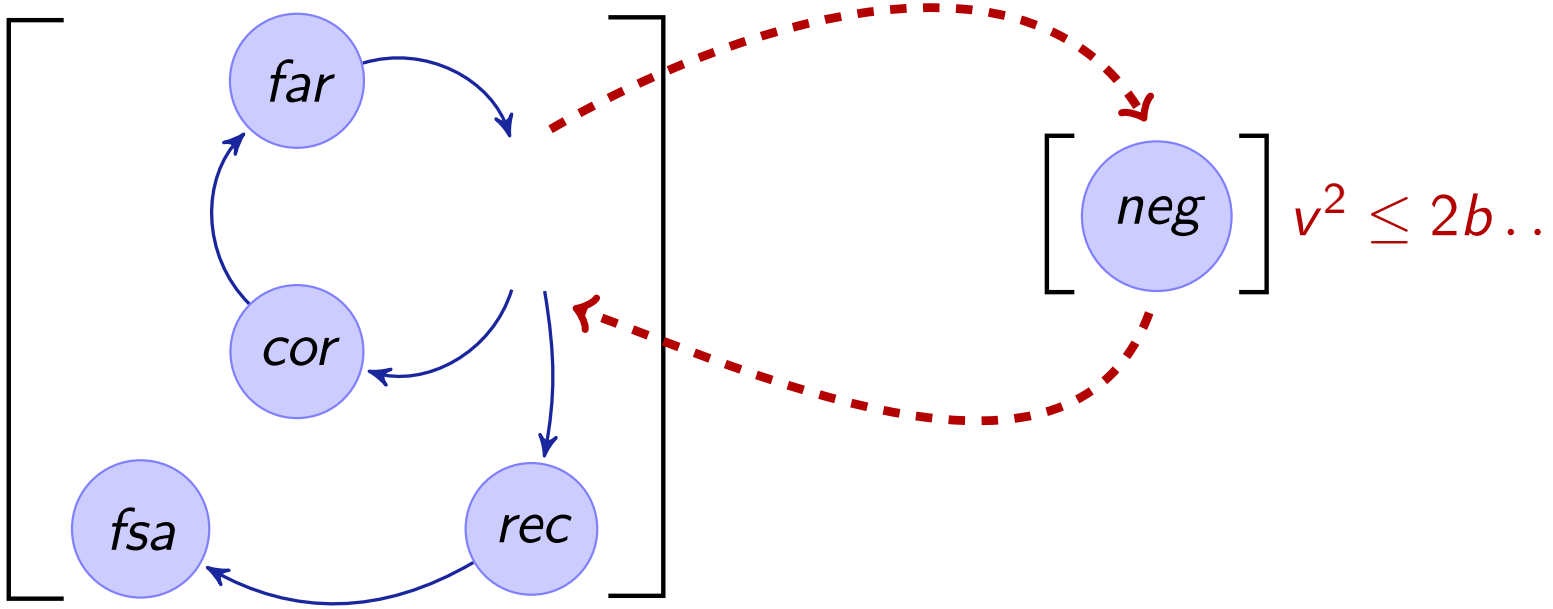
$$[\,\textit{neg}\,]\; v^2 \leq 2b\,..$$

dℒ Design: What about Hybrid Automata?

differential dynamic logic

$$d\mathcal{L} = \text{FOL}_{\mathbb{R}} + \text{DL} + \text{HP}$$

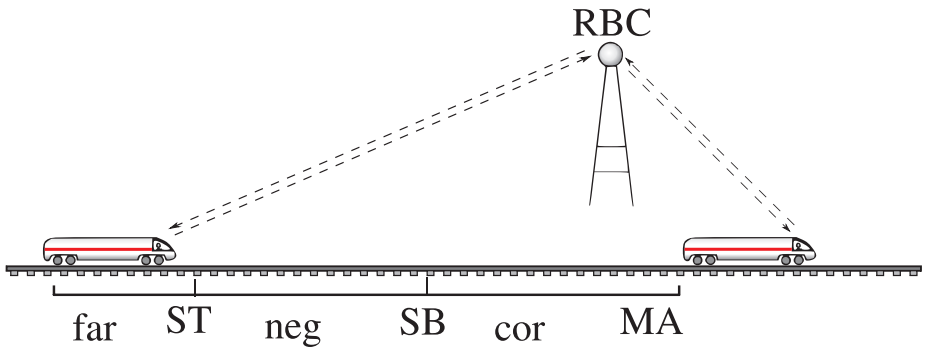


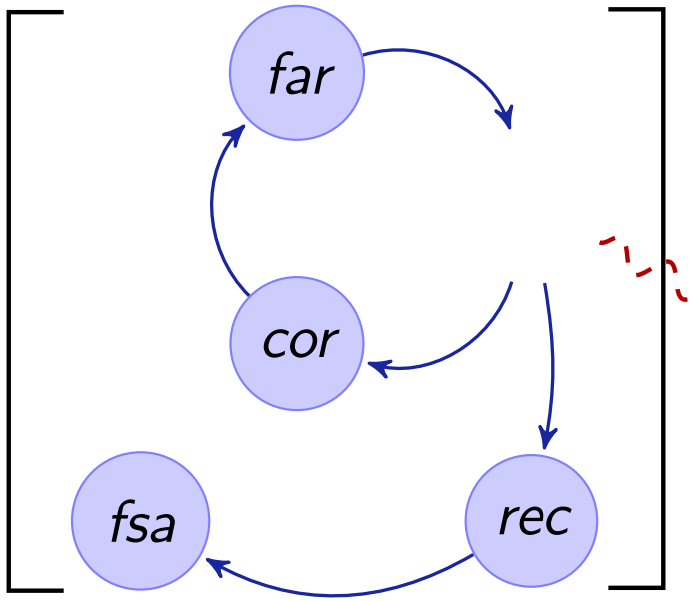


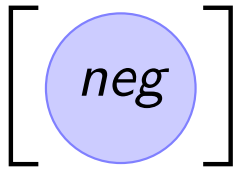
dℒ Design: What about Hybrid Automata?

differential dynamic logic

$$d\mathcal{L} = FOL_{\mathbb{R}} + DL + HP$$







$v^2 \leq 2b\,..$

not compositional

Differential Dynamic Logic dℒ: Syntax

Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$\texttt{if}(\chi)\,\alpha\,\texttt{else}\,\beta$	(conditional execution)	jump & test
$\alpha;\beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	Kleene algebra
α^*	(nondet. repetition)	Kleene algebra

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

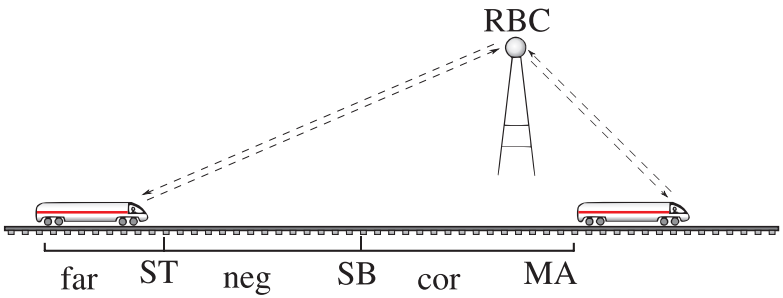
Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$\texttt{if}(\chi)\,\alpha\,\texttt{else}\,\beta$	(conditional execution)	jump & test
$\alpha;\beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	Kleene algebra
α^*	(nondet. repetition)	Kleene algebra

$$ETCS \equiv (ctrl\,;drive)^*$$

$$ctrl \equiv \texttt{if}\ (MA - z < SB)\ \texttt{then}\ a := -b$$
$$\texttt{else}\ a := \ldots$$

$$drive \equiv z'' = a$$
$$\wedge\, v \geq 0 \wedge \tau \leq \varepsilon$$



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

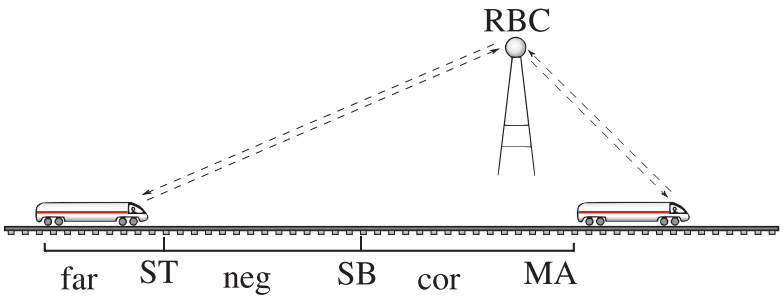
Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$\texttt{if}(\chi)\,\alpha\,\texttt{else}\,\beta$	(conditional execution)	jump & test
$\alpha;\beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	Kleene algebra
α^*	(nondet. repetition)	Kleene algebra

$$ETCS \equiv (ctrl\,;drive)^*$$

$$ctrl \equiv \texttt{if}\ (MA - z < SB)\ \texttt{then}\ a := -b$$
$$\texttt{else}\ a := \ldots$$

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



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

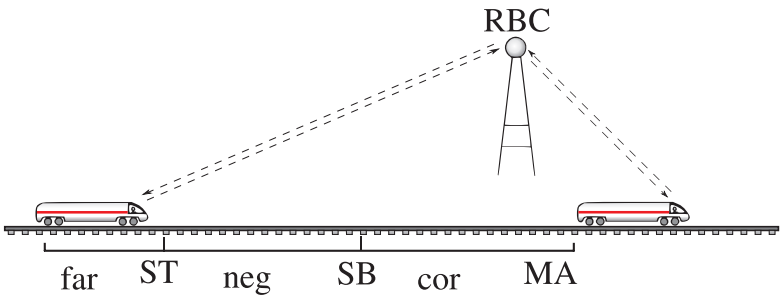
Definition (Hybrid program α)

$x' = f(x) \wedge \chi$	(continuous evolution)	
$x := f(x)$	(discrete jump)	jump & test
$\texttt{if}(\chi)\,\alpha\,\texttt{else}\,\beta$	(conditional execution)	jump & test
$\alpha;\beta$	(seq. composition)	Kleene algebra
$\alpha \cup \beta$	(nondet. choice)	Kleene algebra
α^*	(nondet. repetition)	Kleene algebra

$$ETCS \equiv (ctrl\,;drive)^*$$

$$ctrl \equiv \texttt{if}\ (MA - z < SB)\ \texttt{then}\ a := -b$$
$$\texttt{else}\ a := \ldots$$

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

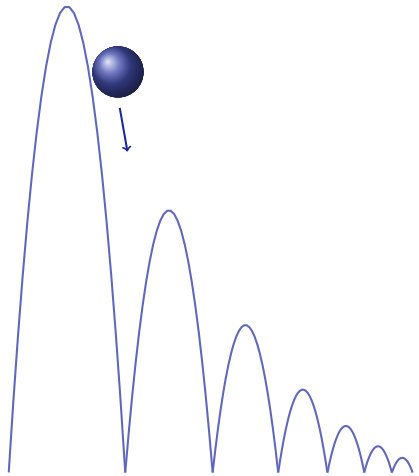


Hybrid Program Example: Controlled Moving Point

```
(
    if (x>0) then
        a := -4          /* move left */
    else
        a := 4           /* move right */
    fi;
    t := 0;              /* reset clock variable t */
    {x'=a, t'=1, t<=c}   /* continuous evolution */
)*                       /* repeat these transitions */
```

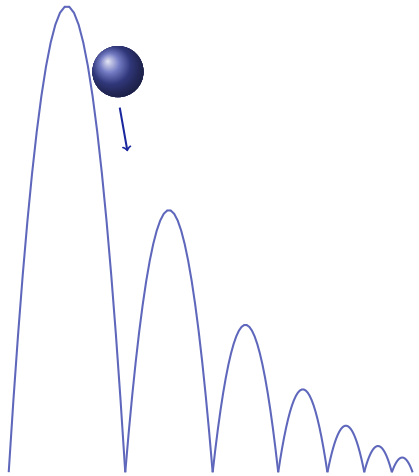
Hybrid Program Example: Bouncing Ball

```
(
    {h'=v,v'=-g,t'=1, h>=0};  /* falling/jumping */
    if (t>0 & h=0) then       /* if on ground */
        v := -c*v;            /* bounce back */
        t := 0
    fi
)*                            /* repeat transitions */
```



Hybrid Program Example: Aerodynamic Bouncing Ball

```
(
    ({h'=v, v'=-g+d*v^2, t'=1, h>=0,v<=0} /* falling */
    ++{h'=v, v'=-g-d*v^2, t'=1, h>=0,v>=0}); /* jumping
    if (t>0 & h=0) then          /* if on ground */
        v := -c*v;               /* bounce back */
        t := 0
    fi
)*                               /* repeat transitions */
```




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


A. Platzer.
Differential dynamic logic for hybrid systems.
J. Autom. Reas., 41(2):143–189, 2008.


A. Platzer.
Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics.
Springer, Heidelberg, 2010.