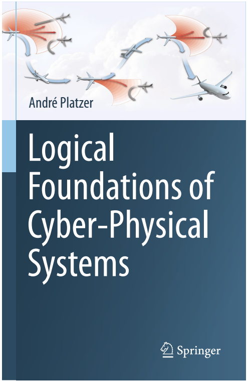
19: Verified Models & Verified Runtime Validation

Logical Foundations of Cyber-Physical Systems



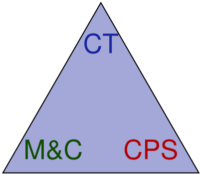
Stefan Mitsch

Carnegie Mellon University
Computer Science Department

Outline

Outline

proof in a model vs. truth in reality
tracing assumptions
turning provers upside down
correct-by-construction
dynamic contracts
proofs for CPS implementations



models vs. reality
inevitable differences
model compliance
architectural design

tame CPS complexity
runtime validation
online monitor
prediction vs. run

Outline

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

S Right answer to wrong question

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \to [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  ↯ Unsatisfiable

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- S Right answer to wrong question
- A Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \to [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct
  - ↯ Empty behavior

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \to [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct
  - ↯ Empty behavior
  - ↯ Model vs. control implementation

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct
  - ↯ Empty behavior
  - ↯ Model vs. control implementation
- *plant* Proof, so all behavior correct

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- S Right answer to wrong question
- A Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- $ctrl$ Proof, so all behavior correct
  - ↯ Empty behavior
  - ↯ Model vs. control implementation
- $plant$ Proof, so all behavior correct
  - ↯ No runs

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Wrong?

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct
  - ↯ Empty behavior
  - ↯ Model vs. control implementation
- *plant* Proof, so all behavior correct
  - ↯ No runs
  - ↯ Plant model vs. real physics

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Models Predictions need models!

- *S* Right answer to wrong question
- *A* Proof, so can't forget condition
  - ↯ Unsatisfiable
  - ↯ Too picky to turn on
- *ctrl* Proof, so all behavior correct
  - ↯ Empty behavior
  - ↯ Model vs. control implementation
- *plant* Proof, so all behavior correct
  - ↯ No runs
  - ↯ Plant model vs. real physics

What Else Could Possibly Go Wrong?

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Models Predictions need models!

- S Ri…
- A Pr…
  - ϟ U…
  - ϟ T…
- $ctrl$ Proof, so all behavior correct
  - ϟ Empty behavior
  - ϟ Model vs. control implementation
- $plant$ Proof, so all behavior correct
  - ϟ No runs
  - ϟ Plant model vs. real physics

Challenge

Verification results about models

only apply if CPS fits to the model

Simultaneous model validation and proof transfer

Outline

Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

A Monitor easy if measurable
Veto turns CPS off

Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

A Monitor easy if measurable
Veto turns CPS off

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

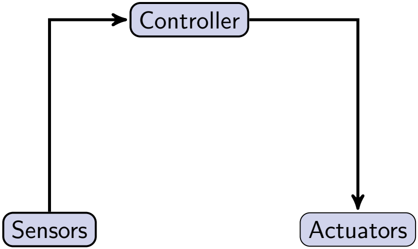
- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Synthesize or Monitor

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!



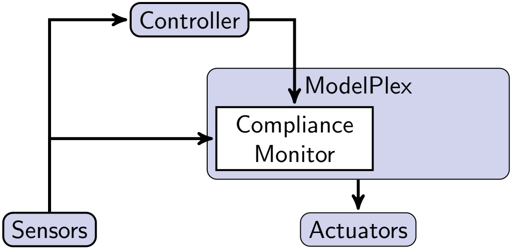
Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Synthesize or Monitor

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!
- *ctrl* Refinement proofs



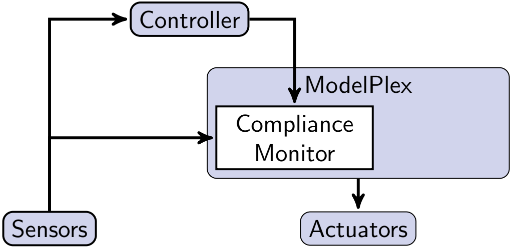
Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Synthesize or Monitor

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!
- *ctrl* Refinement proofs
  Monitor each control decision
  Veto overrides decision



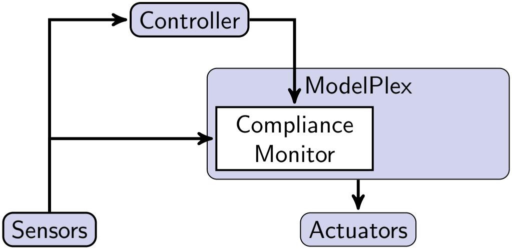
Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!
- *ctrl* Refinement proofs
  Monitor each control decision
  Veto overrides decision



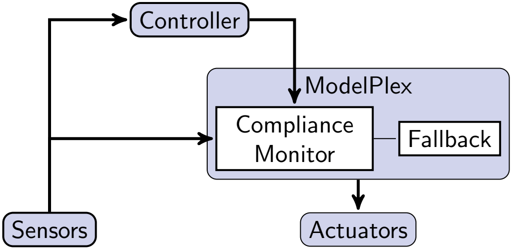
Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!
- *ctrl* Refinement proofs
  Monitor each control decision
  Veto overrides decision
- *plant* No source code for physics
  Observe and compare



Runtime Monitor for Runtime Validation of Model

Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitor

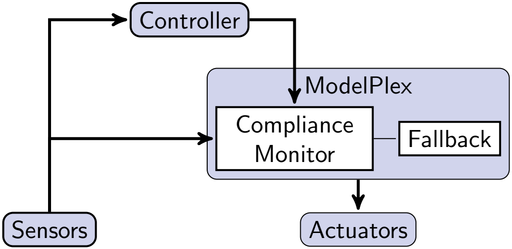
Monitors must be correct

A Monitor easy if measurable
Veto turns CPS off

S Too late to monitor
CPS already unsafe!

ctrl Refinement proofs
Monitor each control decision
Veto overrides decision

plant No source code for physics
Observe and compare
Veto triggers best fallback



Runtime Monitor for Runtime Validation of Model

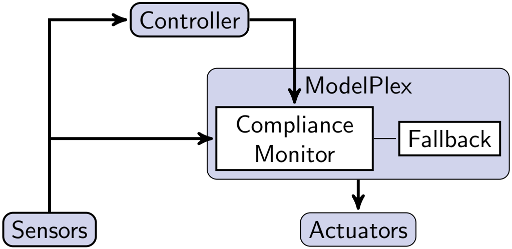
Proposition (System Proved Safe)

$$A \rightarrow [(ctrl; plant)^*]S$$

Monitors must be correct

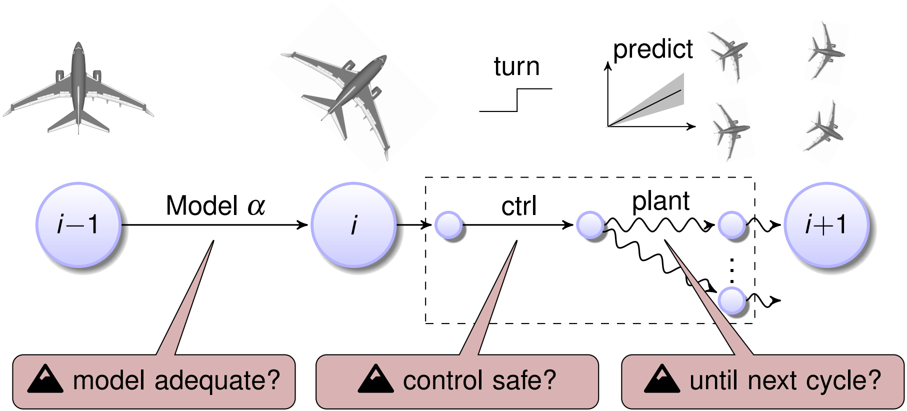
Monitor Verified runtime validation!

- *A* Monitor easy if measurable
  Veto turns CPS off
- *S* Too late to monitor
  CPS already unsafe!
- *ctrl* Refinement proofs
  Monitor each control decision
  Veto overrides decision
- *plant* No source code for physics
  Observe and compare
  Veto triggers best fallback



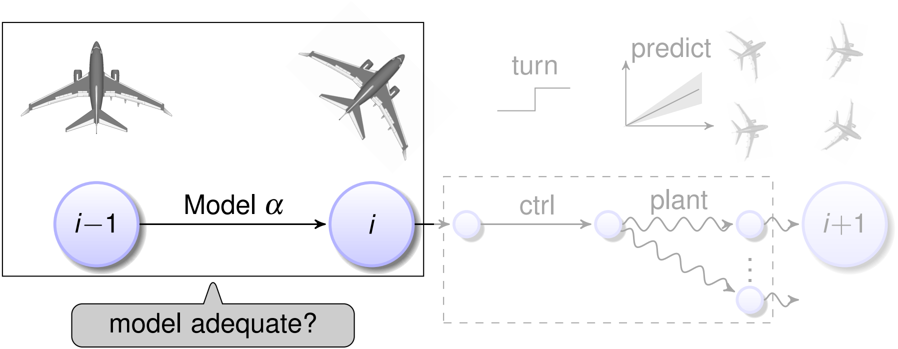
Model Validation and Proof Transfer

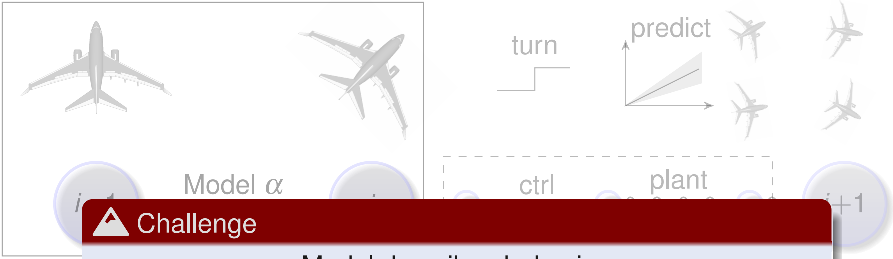
Ensure that verification results about models
apply to CPS implementations



Outline

Outline

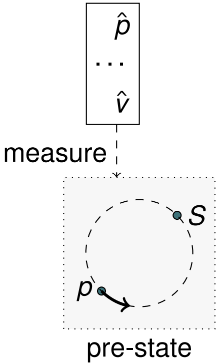




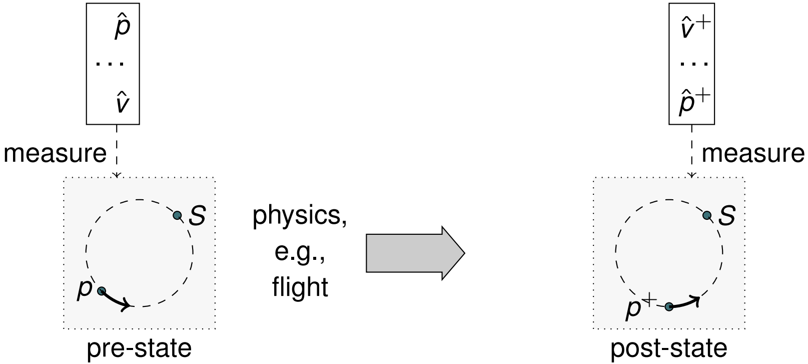
Challenge

Model describes behavior,
but at runtime we get sampled observations

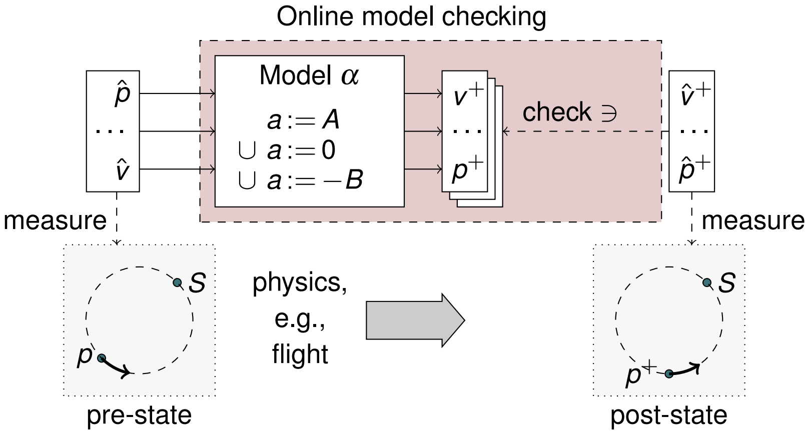
Model Validation



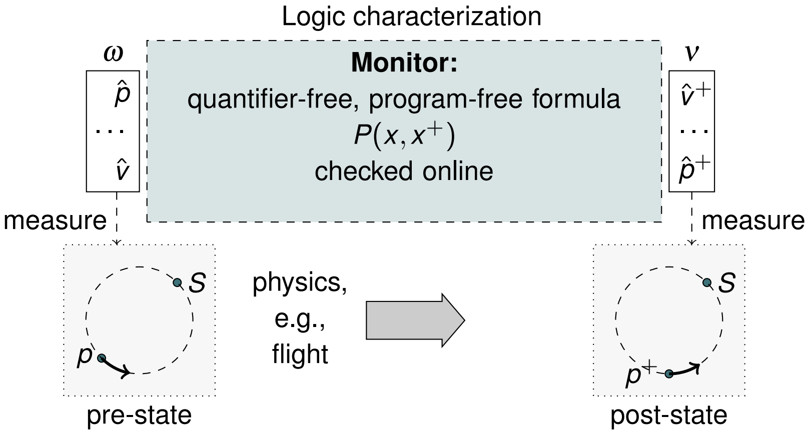
Model Validation

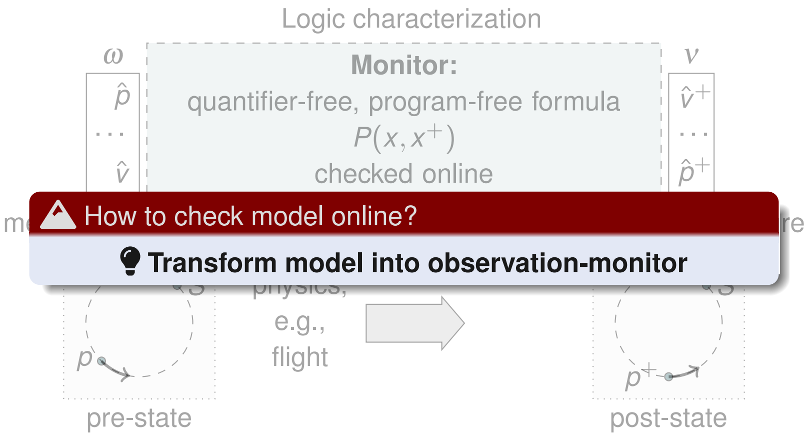


Model Validation



Model Validation





How to check model online?

Transform model into observation-monitor

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

control changes (x, v) to (x^+, v^+)

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \to [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$(x = 0 \land v^+ = -cv \;\lor\; x > 0 \land v^+ = v) \land x^+ = x$

control changes (x, v) to (x^+, v^+)

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$$\big(x = 0 \wedge v^+ = -cv \;\vee\; x > 0 \wedge v^+ = v\big) \wedge x^+ = x$$

test+domain

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$$(x = 0 \land v^+ = -cv \;\lor\; x > 0 \land v^+ = v) \land x^+ = x$$

Example (Plant Monitor)

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$\left(x = 0 \land v^+ = -cv \;\lor\; x > 0 \land v^+ = v\right) \land x^+ = x$

Example (Plant Monitor)

from solution

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$\big(x = 0 \wedge v^+ = -cv \;\vee\; x > 0 \wedge v^+ = v\big) \wedge x^+ = x$

Example (Plant Monitor)

$v^+ = v - g \cdot \Delta t \wedge x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \wedge \Delta t \geq 0$

from solution

domain

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$\left(x = 0 \land v^+ = -cv \;\lor\; x > 0 \land v^+ = v\right) \land x^+ = x$

Example (Plant Monitor)

$v^+ = v - g \cdot \Delta t \land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land \Delta t \geq 0 \land x \geq 0 \land x^+ \geq 0$

domain

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$(x = 0 \land v^+ = -cv \ \lor \ x > 0 \land v^+ = v) \land x^+ = x$

Example (Plant Monitor)

$v^+ = v - g \cdot \Delta t \land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land \Delta t \geq 0 \land x \geq 0 \land x^+ \geq 0$

Example (Model Monitor, combines controller and plant monitor)

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

substitute in

Example (Controller Monitor)

$(x = 0 \land v^+ = -cv \ \lor \ x > 0 \land v^+ = v) \land x^+ = x$

Example (Plant Monitor)

$v^+ = v - g \cdot \Delta t \land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land \Delta t \geq 0 \land x \geq 0 \land x^+ \geq 0$

Example (Model Monitor, combines controller and plant monitor)

$(\qquad)$

$\land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land x \geq 0 \land x^+ \geq 0$

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \rightarrow [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

substitute in

Example (Controller Monitor)

$(x = 0 \land v^+ = -cv \ \lor \ x > 0 \land v^+ = v) \land x^+ = x$

Example (Plant Monitor)

$v^+ = v - g \cdot \Delta t \land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land \Delta t \geq 0 \land x \geq 0 \land x^+ \geq 0$

Example (Model Monitor, combines controller and plant monitor)

$$(x^+ = 0 \land v^+ = -c(v - g \cdot \Delta t) \ \lor \ x^+ > 0 \land v^+ = v - g \cdot \Delta t)$$
$$\land x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \land x \geq 0 \land x^+ \geq 0$$

Bouncing Ball Monitors

Proposition (Can bounce around safely)

$$A \to [(\{x' = v, v' = -g \,\&\, x \geq 0\}; (?x = 0; v := -cv \cup ?x \neq 0))^*]S$$

Example (Controller Monitor)

$$(x = 0 \wedge v^+ = -cv \ \vee \ x > 0 \wedge v^+ = v) \wedge x^+ = x$$

Takeaway

Monitors are subtle, in desperate need of correctness proof.

What proof implies a safe system if the monitors pass?

Examp

$v^+ = $

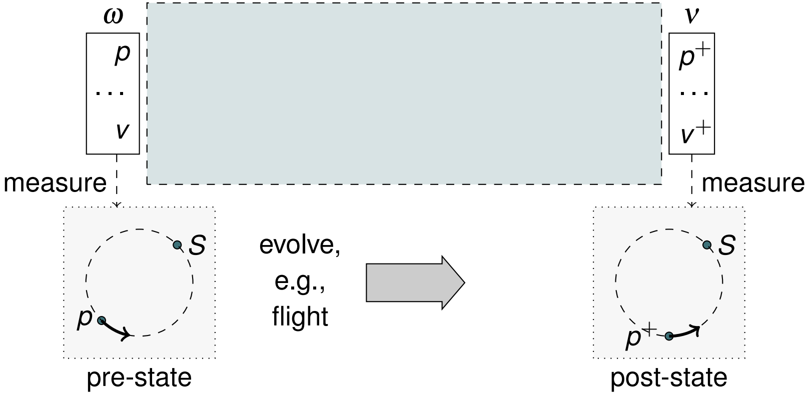
Example (Model Monitor, combines controller and plant monitor)

$$(\qquad)$$

$$\wedge x^+ = x + v \cdot \Delta t - \frac{g}{2}(\Delta t)^2 \wedge x \geq 0 \wedge x^+ \geq 0$$

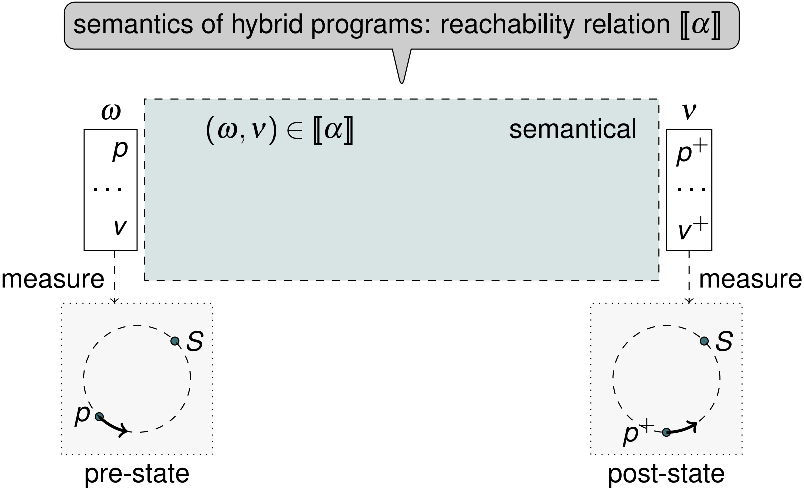
Outline

When are two states linked through a run of model α?



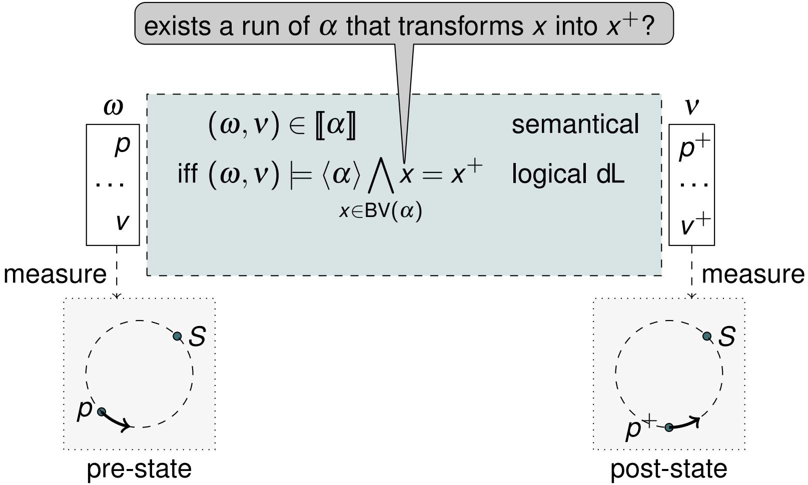
Characterizing State Relations in Logic

When are two states linked through a run of model α?



Characterizing State Relations in Logic

When are two states linked through a run of model α?



Characterizing State Relations in Logic

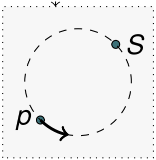
When are two states linked through a run of model α?

ω: p, $\ldots$, v

$(\omega, \nu) \in [\![\alpha]\!]$	semantical
iff $(\omega, \nu) \models \langle \alpha \rangle \bigwedge_{x \in \mathrm{BV}(\alpha)} x = x^{+}$	logical dL
iff $(\omega, \nu) \models P(x, x^{+})$	arithmetical

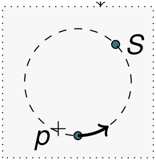
ν: p^{+}, $\ldots$, v^{+}

measure



pre-state

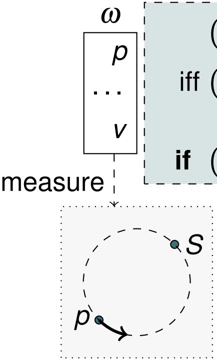
measure



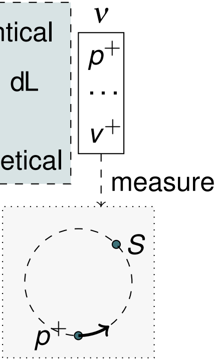
post-state

When are two states linked through a run of model α?

$(\omega, \nu) \in [\![\alpha]\!]$	semantical
iff $(\omega, \nu) \models \langle\alpha\rangle \bigwedge_{x \in \mathrm{BV}(\alpha)} x = x^+$	logical dL
if $(\omega, \nu) \models P(x, x^+)$	arithmetical



pre-state



post-state

Logic reduces online safety to offline proof plus runtime monitor

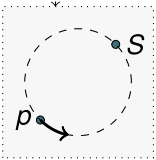
ω: p … v (measure)

ν: p^+ … v^+ (measure)

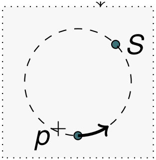
	$(\omega,\nu) \in [\![\alpha]\!]$	semantical
iff	$(\omega,\nu) \models \langle\alpha\rangle \bigwedge_{x\in BV(\alpha)} x = x^+$	logical dL
if	$(\omega,\nu) \models P(x,x^+)$	arithmetical

✔offline proof

$\models P(x,x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x\in BV(\alpha)} x{=}x^+$



pre-state



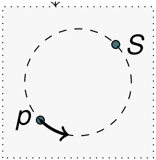
post-state

Logic reduces online safety to offline proof plus runtime monitor

ω: p, …, v $\qquad$ ν: p^+, …, v^+

	$(\omega,\nu) \in [\![\alpha]\!]$	semantical
iff	$(\omega,\nu) \models \langle\alpha\rangle \bigwedge_{x\in \mathrm{BV}(\alpha)} x = x^+$	logical dL
if	$(\omega,\nu) \models P(x,x^+)$	arithmetical

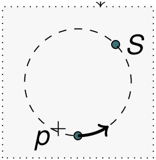
measure $\qquad$ measure



pre-state

✔offline proof

$\models P(x,x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x\in \mathrm{BV}(\alpha)} x{=}x^+$



post-state

$P(x,x^+)$ satisfiable,
but not valid

$\rightsquigarrow$

finish proof at runtime

Logic reduces online safety to offline proof plus runtime monitor

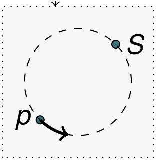
$\vDash A \to [\alpha]S$

✔offline proof

ω: p, …, v ; ν: p^+, …, v^+

$(\omega,\nu) \in [\![\alpha]\!]$	semantical
iff $(\omega,\nu) \vDash \langle\alpha\rangle \bigwedge_{x\in\mathrm{BV}(\alpha)} x = x^+$	logical dL
if $(\omega,\nu) \vDash P(x,x^+)$	arithmetical

measure



pre-state

✔offline proof

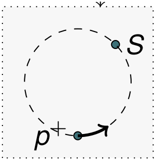
$\vDash P(x,x^+) \to \langle\alpha\rangle \bigwedge_{x\in\mathrm{BV}(\alpha)} x=x^+$

$P(x,x^+)$ satisfiable,
but not valid

$\rightsquigarrow$

finish proof at runtime

measure



post-state

Simultaneous Model Validation and Proof Transfer

Logic reduces online safety to offline proof plus runtime monitor

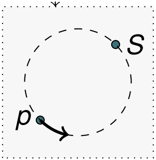
$\models A \rightarrow [\alpha]S \;+\; \omega \in [\![A]\!]$

✔offline proof — runtime monitor

ω: p … v — measure → pre-state

ν: p^+ … v^+ — measure → post-state

$(\omega, \nu) \in [\![\alpha]\!]$	semantical
iff $(\omega, \nu) \models \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$	logical dL
if $(\omega, \nu) \models P(x, x^+)$	arithmetical



pre-state

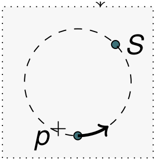
✔offline proof

$\models P(x, x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$

$P(x, x^+)$ satisfiable,
but not valid

$\rightsquigarrow$

finish proof at runtime



post-state

Logic reduces online safety to offline proof plus runtime monitor

$\models A \rightarrow [\alpha]S$ + $\omega \in [\![A]\!]$ + $(\omega,\nu) \in [\![\alpha]\!]$

✔offline proof ⚙runtime monitor

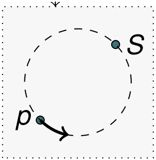
ω: p … v

ν: p^+ … v^+

$(\omega,\nu) \in [\![\alpha]\!]$	semantical
iff $(\omega,\nu) \models \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$	logical dL
if $(\omega,\nu) \models P(x,x^+)$	arithmetical

measure

measure

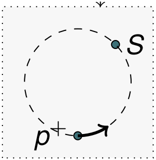


pre-state

✔offline proof

$\models P(x,x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$

$P(x,x^+)$ satisfiable,
but not valid
$\rightsquigarrow$
finish proof at runtime



post-state

Logic reduces online safety to offline proof plus runtime monitor

$\models A \rightarrow [\alpha]S$ + $\omega \in [\![A]\!]$ + $(\omega, \nu) \in [\![\alpha]\!]$ implies $\nu \in [\![S]\!]$

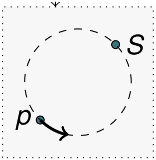
✔offline proof ⚙runtime monitor

ω: p, …, v

$(\omega,\nu) \in [\![\alpha]\!]$	semantical
iff $(\omega,\nu) \models \langle\alpha\rangle \bigwedge_{x\in BV(\alpha)} x = x^+$	logical dL
if $(\omega,\nu) \models P(x,x^+)$	arithmetical

ν: p^+, …, v^+

measure



pre-state

✔offline proof

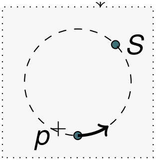
$\models P(x,x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x\in BV(\alpha)} x{=}x^+$

$P(x,x^+)$ satisfiable,
but not valid

$\rightsquigarrow$

finish proof at runtime

measure



post-state

Simultaneous Model Validation and Proof Transfer

Logic reduces online safety to offline proof plus runtime monitor

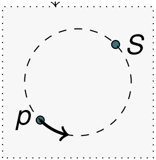
$\models A \rightarrow [\alpha]S$ + $\omega \in [\![A]\!]$ + $(\omega, \nu) \models P(x, x^+)$ implies $\nu \in [\![S]\!]$

✔offline proof ⚙runtime monitor ⚙runtime monitor

ω: p, …, v | ν: p^+, …, v^+

$(\omega, \nu) \in [\![\alpha]\!]$	semantical
iff $(\omega, \nu) \models \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$	logical dL
if $(\omega, \nu) \models P(x, x^+)$	arithmetical

measure



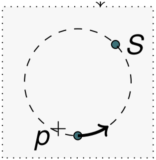
pre-state

✔offline proof

$\models P(x, x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$

$P(x, x^+)$ satisfiable,
but not valid
$\rightsquigarrow$
finish proof at runtime

measure



post-state

Simultaneous Model Validation and Proof Transfer

Logic reduces online safety to offline proof plus runtime monitor

$\models A \rightarrow [\alpha]S$ + $\omega \in [\![A]\!]$ + $(\omega, \nu) \models P(x, x^+)$ implies $\nu \in [\![S]\!]$

✔offline proof — runtime monitor — runtime monitor

ω: p … ν: p^+ …

$(\omega, \nu) \in [\![\alpha]\!]$ semantical

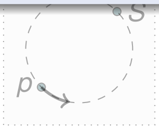
iff $(\omega, \nu) \models \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$ logical dL

Theorem (Model Monitor Correctness with Perfect Sensors)

System safe as long as monitor satisfied **(FMSD'16)**

(extension to imperfect sensors with additional quantifiers)

asure



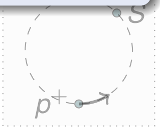
pre-state

$\models P(x, x^+) \rightarrow \langle\alpha\rangle \bigwedge_{x \in BV(\alpha)} x = x^+$

$P(x, x^+)$ satisfiable, but not valid

$\rightsquigarrow$

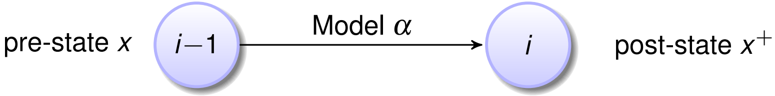
finish proof at runtime



post-state

Correct-by-Construction Synthesis

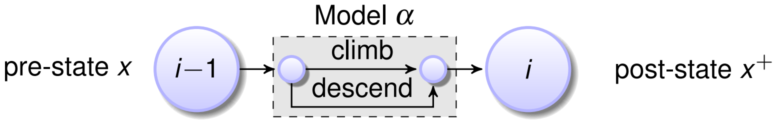
dL proof calculus executes models symbolically



proof attempt

- $\langle\alpha\rangle(\bigwedge_{x\in \mathsf{BV}(\alpha)} x = x^+)$

dL proof calculus executes models symbolically

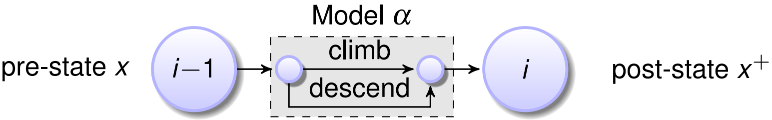


proof attempt

$\langle \text{climb} \cup \text{descend} \rangle (\bigwedge_{x \in \text{BV}(\alpha)} x = x^+)$

$\langle \text{climb} \cup \text{descend} \rangle P \leftrightarrow$
$\langle \text{climb} \rangle P \vee \langle \text{descend} \rangle P$

dL proof calculus executes models symbolically



proof attempt

$\langle \text{climb} \cup \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

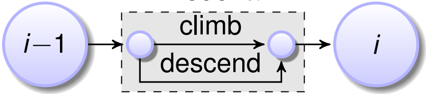
$\vee$

$\langle \text{climb} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$ $\quad$ $\langle \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

dL proof calculus executes models symbolically

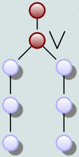
Model α

pre-state x



post-state x^+

proof attempt



$\langle \text{climb} \cup \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

$\vee$

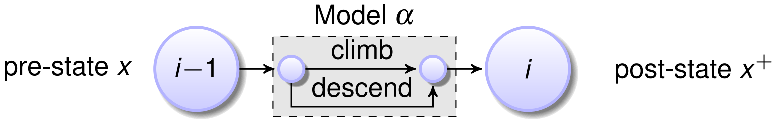
$\langle \text{climb} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

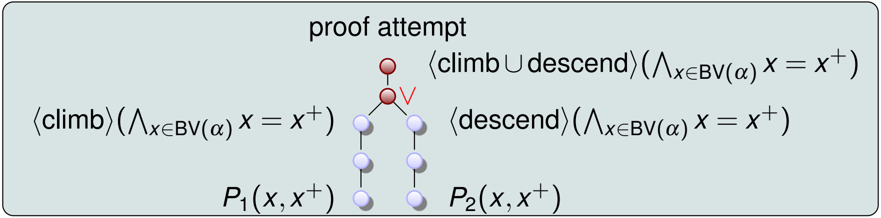
$\langle \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

$P_1(x, x^+)$

$P_2(x, x^+)$

dL proof calculus executes models symbolically

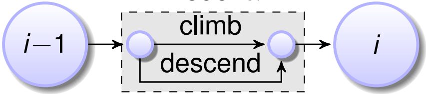




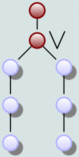
Monitor: $P_1(x, x^+) \vee P_2(x, x^+)$

dL proof calculus executes models symbolically

Model α

pre-state x  post-state x^+

proof attempt



$\langle \text{climb} \cup \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

$\vee$

$\langle \text{climb} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$ $\quad$ $\langle \text{descend} \rangle (\bigwedge_{x \in BV(\alpha)} x = x^+)$

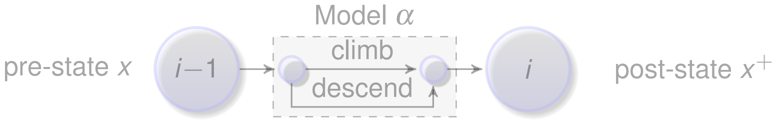
$P_1(x, x^+)$ $\quad$ $P_2(x, x^+)$

Monitor: $\overbrace{P_1(x, x^+) \vee P_2(x, x^+)}$

- The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\rightsquigarrow$ finish proof at runtime

Correct-by-Construction Synthesis

dL proof calculus executes models symbolically



proof attempt

Model Monitor

Immediate detection of model violation

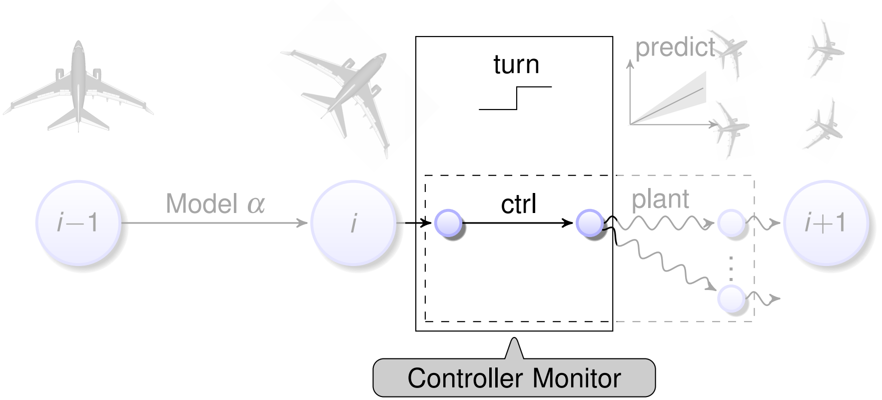
$\rightsquigarrow$ Mitigates safety issues with safe fallback action

$P_1(x, x^+)$ $\quad$ $P_2(x, x^+)$

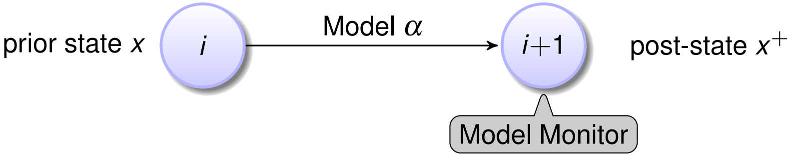
Monitor: $\overbrace{P_1(x, x^+) \vee P_2(x, x^+)}$

- The subgoals that cannot be proved express all the conditions on the relations of variables imposed by the model $\rightsquigarrow$ finish proof at runtime

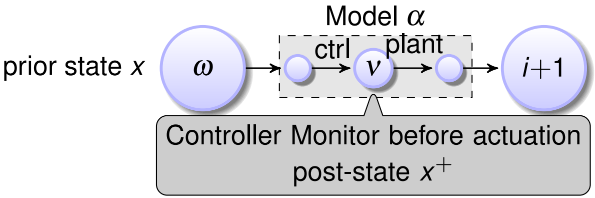
Typical (ctrl; plant)* models can check earlier



Controller Monitor: Veto Early If Noncompliant

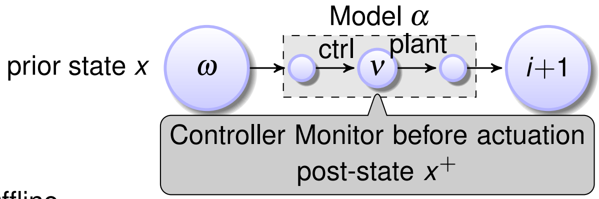


Controller Monitor: Veto Early If Noncompliant



Semantical: $(\omega, \nu) \in [\![\text{ctrl}]\!]$ reachability relation of ctrl

Controller Monitor: Veto Early If Noncompliant



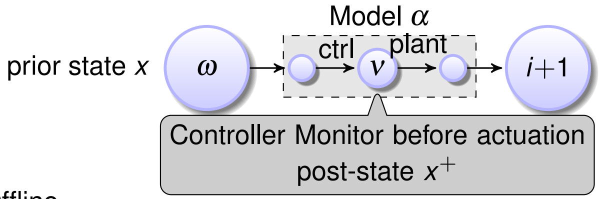
Offline

Semantical: $(\omega, \nu) \in [\![\text{ctrl}]\!]$

$\Updownarrow$ Theorem

Logical dL: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

exists a run of ctrl to a state where $x = x^+$



Offline

Semantical: $(\omega, \nu) \in [\![\text{ctrl}]\!]$

$\Updownarrow$ Theorem

Logical dL: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+)$

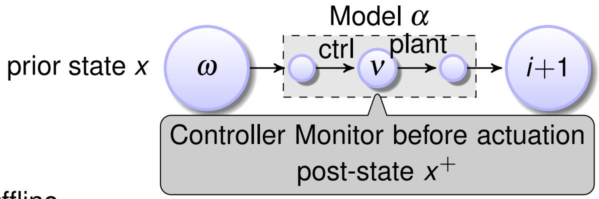
exists a run of ctrl to a state where $x = x^+$

$\Uparrow$ dL proof

Arithmetical: $(\omega, \nu) \models P(x, x^+)$

check at runtime

Controller Monitor: Veto Early If Noncompliant



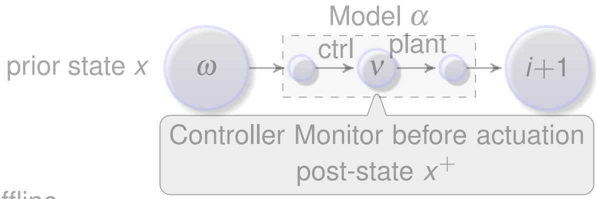
Offline

Semantical:	$(\omega,\nu) \in [\![\text{ctrl}]\!]$	
	$\Updownarrow$ Theorem	
Logical dL:	$(\omega,\nu) \models \langle \text{ctrl} \rangle (x = x^+)$	exists a run of ctrl to a state where $x = x^+$
	$\Uparrow$ dL proof	
Arithmetical:	$(\omega,\nu) \models P(x,x^+)$	check at runtime

Theorem (Controller Monitor Correctness)

Controller safe and in plant bounds as long as monitor satisfied (FMSD'16)

Controller Monitor: Veto Early If Noncompliant



Offline

Controller Monitor

Immediate detection of unsafe control before actuation

$\rightsquigarrow$ Safe execution of unverified implementations in perfect environments

to x^+

$\Uparrow$ dL proof

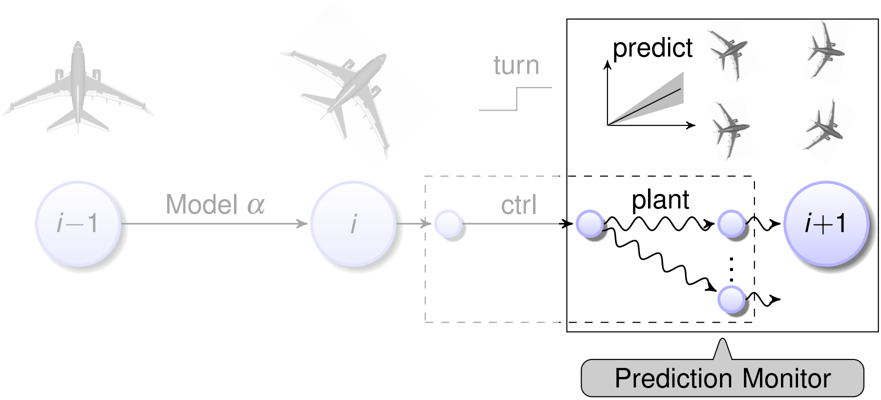
Arithmetical: $(\omega,\nu) \models P(x,x^+)$

check at runtime

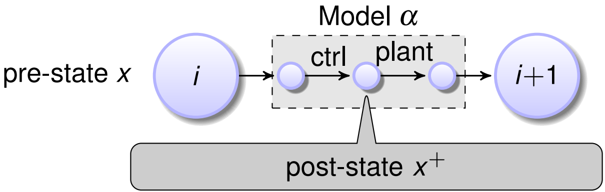
Theorem (Controller Monitor Correctness)

Controller safe and in plant bounds as long as monitor satisfied *(FMSD'16)*

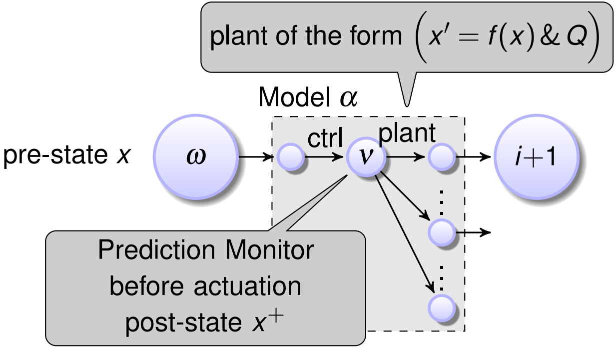
Safe despite evolution with disturbance?



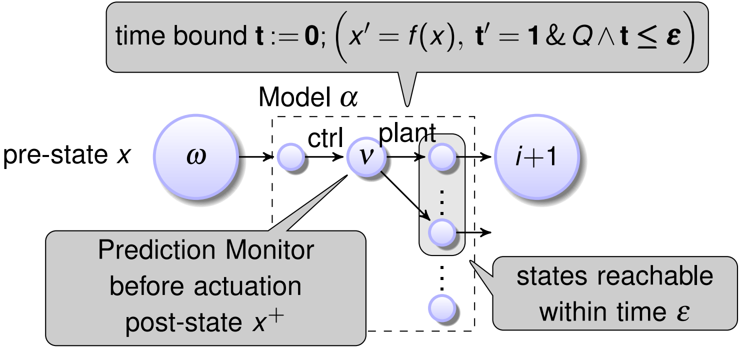
Prediction Monitor: Compliance with Disturbance



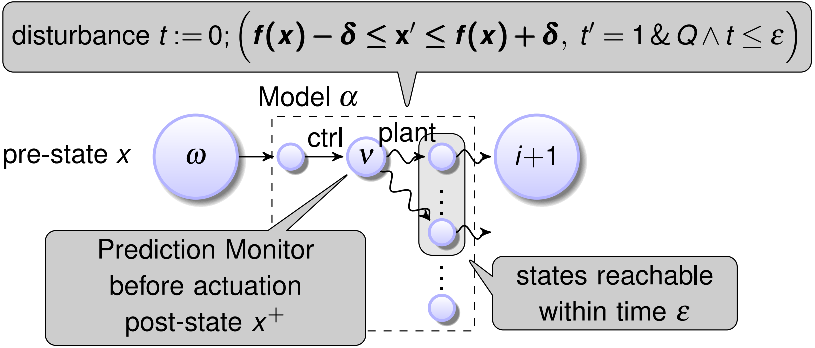
Prediction Monitor: Compliance with Disturbance



Prediction Monitor: Compliance with Disturbance

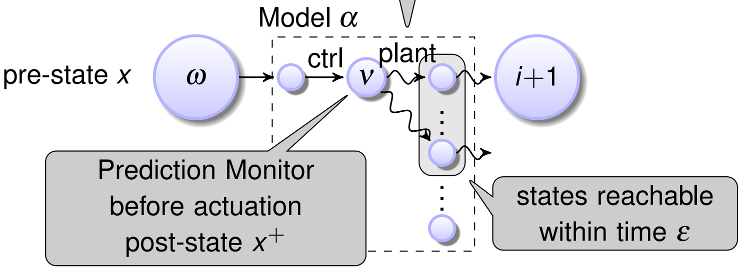


Prediction Monitor: Compliance with Disturbance



Prediction Monitor: Compliance with Disturbance

disturbance $t := 0; \left(\boldsymbol{f(x)} - \boldsymbol{\delta} \leq \mathbf{x}' \leq \boldsymbol{f(x)} + \boldsymbol{\delta},\ t' = 1 \,\&\, Q \wedge t \leq \varepsilon\right)$



Offline

Logical dL: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \wedge [\text{plant}]J)$

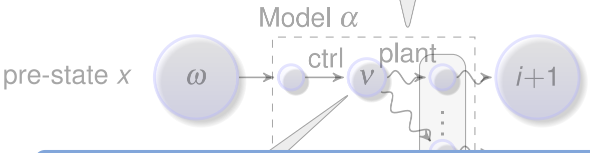
$\Uparrow$ dL proof

Arithmetical: $(\omega, \nu) \models P(x, x^+)$

Invariant J implies safety S
(known from safety proof)

Prediction Monitor: Compliance with Disturbance

disturbance $t := 0; \left(\boldsymbol{f(x) - \delta \leq x' \leq f(x) + \delta},\ t' = 1 \,\&\, Q \land t \leq \varepsilon\right)$



Prediction Monitor with Disturbance

Detect unsafe control before actuation despite disturbance

$\rightsquigarrow$ **Safety in realistic environments**

Offline

Logical dL: $(\omega, \nu) \models \langle \text{ctrl} \rangle (x = x^+ \land [\text{plant}] J)$

$\Uparrow$ dL proof

Arithmetical: $(\omega, \nu) \models P(x, x^+)$

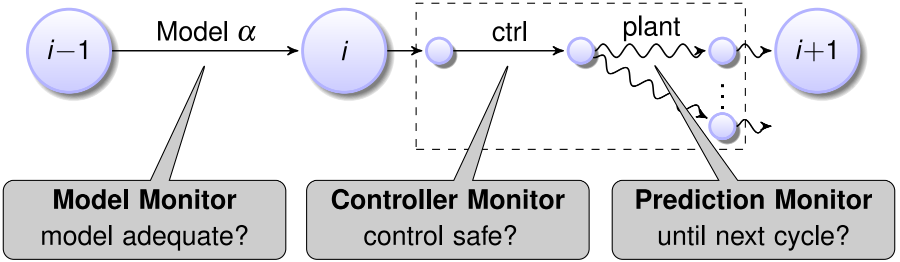
Invariant J implies safety S (known from safety proof)

Outline

Summary

Simultaneous model validation and proof transfer safeguards real CPS

- Validate model compliance
- Characterize compliance with model in logic
- Prover transforms compliance formula to executable monitor
- Model validation and proof transfer by offline + online proof



Stefan Mitsch and André Platzer.
ModelPlex: Verified runtime validation of verified cyber-physical system models.
Form. Methods Syst. Des., 49(1-2):33–74, 2016.
Special issue of selected papers from RV'14.
doi:10.1007/s10703-016-0241-z.

Stefan Mitsch and André Platzer.
ModelPlex: Verified runtime validation of verified cyber-physical system models.
In Borzoo Bonakdarpour and Scott A. Smolka, editors, *RV*, volume 8734 of *LNCS*, pages 199–214. Springer, 2014.
doi:10.1007/978-3-319-11164-3_17.

Stefan Mitsch and André Platzer.
Verified runtime validation for partially observable hybrid systems.
CoRR, abs/1811.06502, 2018.
URL: http://arxiv.org/abs/1811.06502, arXiv:1811.06502.