

# Automatically Robustifying Verified Hybrid Systems in KeYmaera X

---

Nathan Fulton

Carnegie Mellon University

September 13, 2016

Dagstuhl, Germany

A system is **robust** if it operates correctly despite:

- Disturbances in actuation
- Uncertainty in sensing
- Deviation from typical dynamics
- Adversarial agents
- ...

# Robustness

A system is **robust** if it operates correctly despite:

- Disturbances in actuation
- Uncertainty in sensing
- Deviation from typical dynamics
- Adversarial agents
- ...

Expressible by systematically  
modifying a hybrid system

A system is **robust** if it operates correctly despite:

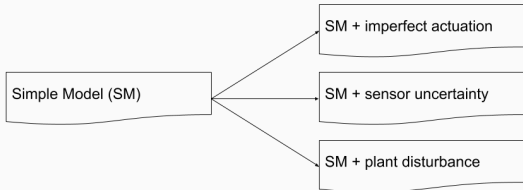
- Disturbances in actuation
- Uncertainty in sensing
- Deviation from typical dynamics
- Adversarial agents
- ...

Expressible by systematically  
modifying a hybrid system

Can we automatically robustify hybrid systems?

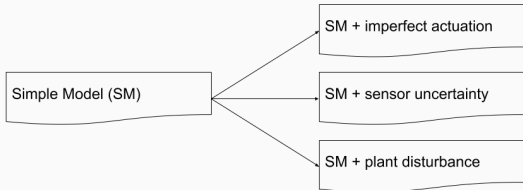
# Automatic Incremental Robustification

Typical verification approach: begin with a **simplified model**, then incrementally add **complexity**.



# Automatic Incremental Robustification

Typical verification approach: begin with a **simplified model**, then incrementally add **complexity**.

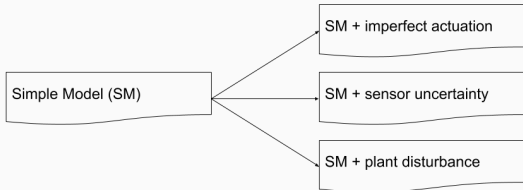


Advantages:

- Initial verification task exposes essential aspects of the safety argument.
- Successive verification tasks are tractable.

# Automatic Incremental Robustification

Typical verification approach: begin with a **simplified model**, then incrementally add **complexity**.



## Advantages:

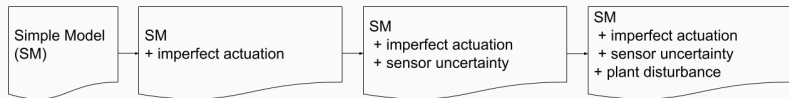
- Initial verification task exposes essential aspects of the safety argument.
- Successive verification tasks are tractable.

## Disadvantages:

- Re-verification is expensive.
- Verification efforts are non-compositional.

# Automatic Incremental Robustification

Typical verification approach: begin with a **simplified model**, then incrementally add **complexity**.



## Advantages:

- Initial verification task exposes essential aspects of the safety argument.
- Successive verification tasks are tractable.

## Disadvantages:

- Re-verification is expensive.
- Verification efforts are non-compositional.



## Definition (Hybrid Programs)

**Assign**  $x := \theta$

**Sequence**  $\alpha; \beta$

**Test**  $?\varphi$

**Iteration**  $\alpha^*$

**Choice**  $\alpha \cup \beta$

**ODEs**  $\{x'_1 = \theta_1, \dots, x'_n = \theta_n \ \& \ H\}$

## Definition (Hybrid Programs)

**Assign**  $x := \theta$

**Sequence**  $\alpha; \beta$

**Test**  $?\varphi$

**Iteration**  $\alpha^*$

**Choice**  $\alpha \cup \beta$

**ODEs**  $\{x'_1 = \theta_1, \dots, x'_n = \theta_n \ \& \ H\}$

Differential Dynamic Logic (d $\mathcal{L}$ ) formulas describe reachability properties of hybrid programs using modalities:  $[\alpha]\varphi$  and  $\langle\alpha\rangle\varphi$ .

# Specifying Hybrid Systems



## Example: A Hybrid Systems Specification in dL

$$\begin{aligned} & [ \{ \\ & \quad \{ ?(x \geq \frac{(AT + v)^2}{2B} + obs); a := A \cup a := -B \}; \\ & \quad c := 0; \{ x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T \} \\ & \quad \}^* ] x \leq obs \end{aligned}$$

## Example: A Hybrid Systems Specification in dL

$$\begin{aligned} & [ \{ \\ & \quad \{ ?(x \geq \frac{(AT + v)^2}{2B} + obs); a := A \cup a := -B \}; \\ & \quad c := 0; \{ x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T \} \\ & \quad \}^* ] x \leq obs \end{aligned}$$

- Parametric controller design

## Example: A Hybrid Systems Specification in dL

$$\begin{aligned} & [ \{ \\ & \quad \{ ?(x \geq \frac{(AT + v)^2}{2B} + obs); a := A \cup a := -B \}; \\ & \quad c := 0; \{ x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T \} \\ & \quad \}^* ] x \leq obs \end{aligned}$$

- Parametric controller design
- Non-determinism

## Example: A Hybrid Systems Specification in dL

$$A > 0 \wedge B > 0 \wedge T > 0 \wedge v \geq 0 \wedge \frac{v^2}{2B} + obs \leq x \leq obs$$

→

[{

$$\{?(x \geq \frac{(AT + v)^2}{2B} + obs); a := A \cup a := -B\};$$

$$c := 0; \{x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T\}$$

}\*]x ≤ obs

- Parametric controller design
- Non-determinism
- Symbolic constraints on parameters

# Verifying a Simple Hybrid System in KeYmaera X

KeYmaera X is a **trustworthy** and **scriptable** hybrid systems theorem prover.

- Trustworthy: All prover automation passes through a small soundness-critical core (< 2 KLOC).
- Scriptable: KeYmaera X provides a DSL for writing proof search programs.



## Example: Adding Actuation Error

$$A > 0 \wedge B > 0 \wedge T > 0 \wedge v \geq 0 \wedge$$

$$\frac{v^2}{2B} + obs \leq x \leq obs \rightarrow$$

$$[\{ \{?(x \geq \frac{((A)T+v)^2}{2(B)} + obs); a := A \cup a := -B\};$$

$$c := 0; \{x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T\}$$

$$\}^*]x \leq obs$$

## Example: Adding Actuation Error

$$\begin{aligned} & A > 0 \wedge B > 0 \wedge T > 0 \wedge v \geq 0 \wedge 0 < \epsilon < A \wedge \epsilon < B \wedge \\ & \frac{v^2}{2B \pm \epsilon} + obs \leq x \leq obs \rightarrow \\ & [ \{ \\ & \quad \{ ?(x \geq \frac{((A \pm \epsilon)T + v)^2}{2(B \pm \epsilon)} + obs); a := A \pm \epsilon \cup a := -B \pm \epsilon \}; \\ & \quad c := 0; \{ x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T \} \\ & \quad \}^* ] x \leq obs \end{aligned}$$

## Example: Adding Actuation Error

$$\begin{aligned} & A > 0 \wedge B > 0 \wedge T > 0 \wedge v \geq 0 \wedge 0 < \epsilon < A \wedge \epsilon < B \wedge \\ & \frac{v^2}{2B-\epsilon} + obs \leq x \leq obs \rightarrow \\ & [ \{ \\ & \quad \{?(x \geq \frac{((A+\epsilon)T+v)^2}{2(B-\epsilon)} + obs); a := A+\epsilon \cup a := -B-\epsilon\}; \\ & \quad c := 0; \{x' = v, v' = a, c' = 1 \wedge v \geq 0 \wedge c \leq T\} \\ & \quad \}^* ] x \leq obs \end{aligned}$$

## Simple Model

```
ImplyR(1) & loop(p(x,v,a,A,B),
1) <(
  QE, QE,
  splitCases(1) <(
    chase(1) & ODE & QE
    chase(1) & ODE & QE
  ))
```

## Simple Model + Uncertainty

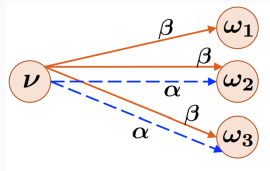
```
ImplyR(1) &
loop(p(x,v,a,A+ $\epsilon$ ,B- $\epsilon$ ), 1) <(
  QE, QE,
  splitCases(1) <(
    chase(1) & ODE & QE
    chase(1) & ODE & QE
  ))
```

# Incremental Robustification via Model/Proof Co-Transformation

- ✓ Tractable initial verification
- ✓ Verification of robustified models re-use ideas from initial safety proof
- ? Compositional robustification
- ✓ Re-verification is expensive (manual effort)
- × Re-verification is expensive (computationally)

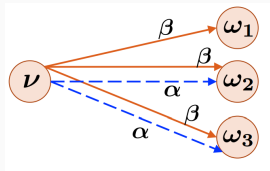
## Incremental Robustification via Refinement

System  $\alpha$  **refines** system  $\beta$  ( $\alpha \leq \beta$ ) if every state reachable by  $\alpha$  is also reachable by  $\beta$ .



## Incremental Robustification via Refinement

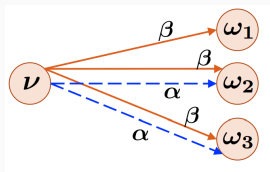
System  $\alpha$  **refines** system  $\beta$  ( $\alpha \leq \beta$ ) if every state reachable by  $\alpha$  is also reachable by  $\beta$ .



- Many robustifications are refinements (after changing environment **and** controller).

## Incremental Robustification via Refinement

System  $\alpha$  **refines** system  $\beta$  ( $\alpha \leq \beta$ ) if every state reachable by  $\alpha$  is also reachable by  $\beta$ .



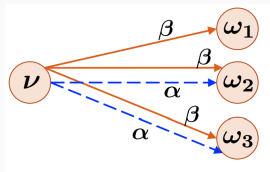
- Many robustifications are refinements (after changing environment **and** controller).
- Refinement makes *direct* use the initial safety property:

$$\frac{[\beta]\varphi \quad \alpha \leq \beta}{[\alpha]\varphi}$$



## Incremental Robustification via Refinement

System  $\alpha$  **refines** system  $\beta$  ( $\alpha \leq \beta$ ) if every state reachable by  $\alpha$  is also reachable by  $\beta$ .



- Many robustifications are refinements (after changing environment **and** controller).
- Refinement makes *direct* use the initial safety property:

$$\frac{[\beta]\varphi \quad \alpha \leq \beta}{[\alpha]\varphi}$$

- $\leq$  has a well-understood algebraic structure.

**Automatic incremental robustification automates common changes to CPS models**

### **Automatic incremental robustification automates common changes to CPS models**

Further Thoughts:

- It would be nice to have automatic robustification procedures for high-fidelity models of common sensors and actuators.
- Notions of robustness are describable in differential game logic (dGL); automation story is unclear.

# Conclusions and Further Thoughts

## Automatic incremental robustification automates common changes to CPS models

Further Thoughts:

- It would be nice to have automatic robustification procedures for high-fidelity models of common sensors and actuators.
- Notions of robustness are describable in differential game logic (dGL); automation story is unclear.

Thanks: KeYmaera X developers (Stefan Mistch, André Platzer, Brandon Bohrer, Jan-David Quesel)

Advertisement: KeYmaera X Tutorial at FM this year!