Automatically Robustifying Verified Hybrid Systems in KeYmaera X

Nathan Fulton
Carnegie Mellon University

September 13, 2016

Dagstuhl, Germany

Robustness

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

Robustness

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

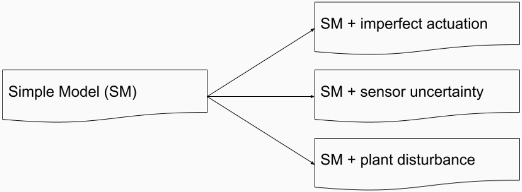
- Disturbances in actuation
- Uncertainty in sensing
- Deviation from typical dynamics
- Adversarial agents
- ...

Expressible by systematically modifying a hybrid system

Can we automatically robustify hybrid systems?

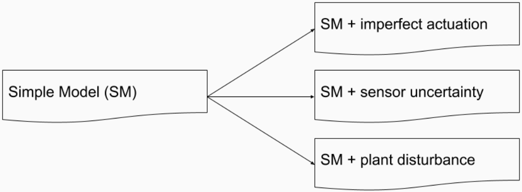
Automatic Incremental Robustification

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**.

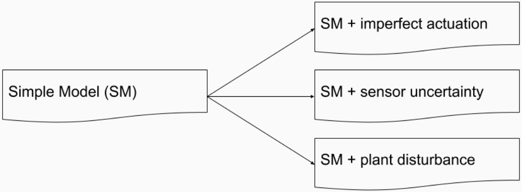


Advantages:

- 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:

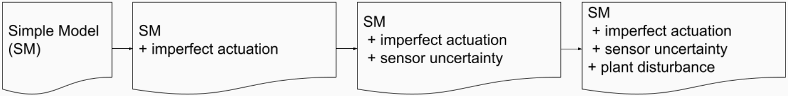
- Initial verification task exposes essential aspects of the safety argument.
- Successive verification tasks are tractable.

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.

Goal: Automatic Incremental Robustification

Specifying Hybrid Systems

Definition (Hybrid Programs)

Assign $x := \theta$

Sequence $\alpha; \beta$

Test $?\varphi$

Iteration α^*

Choice $\alpha \cup \beta$

ODEs $\{x_1' = \theta_1, \ldots, x_n' = \theta_n \;\&\; H\}$

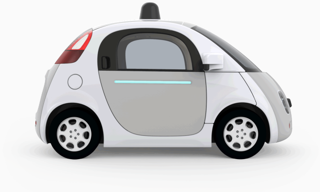
Specifying Hybrid Systems

Definition (Hybrid Programs)

Assign	$x := \theta$
Sequence	$\alpha; \beta$
Test	$?\varphi$
Iteration	α^*
Choice	$\alpha \cup \beta$
ODEs	$\{x_1' = \theta_1, \ldots, x_n' = \theta_n \;\&\; H\}$

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

Specifying Hybrid Systems

[] φ

Example: A Hybrid Systems Specification in d$\mathcal{L}$

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

Example: A Hybrid Systems Specification in d$\mathcal{L}$

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

- Parametric controller design

Example: A Hybrid Systems Specification in d$\mathcal{L}$

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

- Parametric controller design
- Non-determinism

Example: A Hybrid Systems Specification in d$\mathcal{L}$

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

$$\rightarrow$$

$$[\{$$

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

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

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

- 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

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

Example: Adding Actuation Error

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

Example: Adding Actuation Error

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

Co-Transformation of Models and Tactics

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

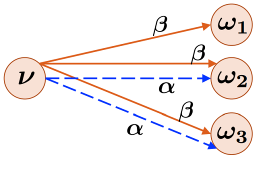
```
ImplyR(1) &
loop(p(x,v,a,A+ϵ,B−ϵ), 1) <(
  QE, QE,
  splitCases(1) <(
    chase(1) & ODE & QE
    chase(1) & ODE & QE
  ))
```

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)

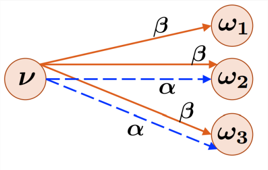
Incremental Robustification via Refinement

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



Incremental Robustification via Refinement

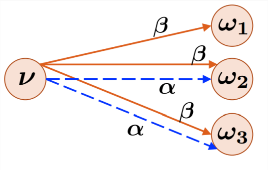
System α **refines** system β ($\alpha \leq \beta$) if every state reachable by α is also reachable by β.



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

Incremental Robustification via Refinement

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

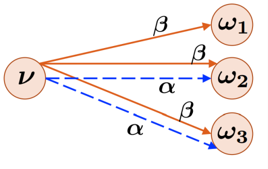


- Many robustifications are refinements (after changing environment **and** controller).
- Refinement makes *direct* use the initial safety property:

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

Incremental Robustification via Refinement

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



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

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

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

Automatic incremental robustification automates common changes to CPS models

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 (d$\mathsf{G}\mathcal{L}$); 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 (dG$\mathcal{L}$); automation story is unclear.

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

Advertisement: KeYmaera X Tutorial at FM this year!