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

- Disturbances in actuation
- Uncertainty in sensing
- Deviation from typical dynamics
- Adversarial agents
- ...
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?**
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
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.
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
Definition (Hybrid Programs)

Assign  \( x := \theta \)

Sequence  \( \alpha; \beta \)

Test  \(?\varphi\)

Iteration  \( \alpha^* \)

Choice  \( \alpha \cup \beta \)

ODEs  \( \{x'_1 = \theta_1, \ldots, x'_n = \theta_n & H\} \)
### Definition (Hybrid Programs)

- **Assign** $x := \theta$
- **Sequence** $\alpha; \beta$
- **Test** $?\phi$
- **Iteration** $\alpha^*$
- **Choice** $\alpha \cup \beta$
- **ODEs** $\{x'_1 = \theta_1, \ldots, x'_n = \theta_n \& H\}$

Differential Dynamic Logic (dL) formulas describe reachability properties of hybrid programs using modalities: $[\alpha]\phi$ and $\langle \alpha \rangle \phi$. 
Specifying Hybrid Systems
Example: A Hybrid Systems Specification in dL

\[
\{ ?(x \geq \frac{(AT + v)^2}{2} + obs); a := A \cup a := -B \}; \\
c := 0; \{x' = v, v' = a, c' = 1 \land v \geq 0 \land c \leq T \} \\
\}^* x \leq obs
\]
Example: A Hybrid Systems Specification in dL

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

• Parametric controller design
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 dL

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

\[ \rightarrow \]

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

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

\[ A > 0 \land B > 0 \land T > 0 \land v \geq 0 \land \]
\[ \frac{v^2}{2B} + obs \leq x \leq obs \rightarrow \]
\[ \{ \{ x \geq \frac{((A)^T + v)^2}{2(B)} + obs \}; a := A \cup a := -B \}; \]
\[ c := 0; \{ x' = v, v' = a, c' = 1 \land v \geq 0 \land c \leq T \} \]
\[ \}^* x \leq obs \]
Example: Adding Actuation Error

\[ A > 0 \land B > 0 \land T > 0 \land v \geq 0 \land 0 < \epsilon < A \land \epsilon < B \land \frac{v^2}{2B \pm \epsilon} + obs \leq x \leq obs \rightarrow \]

\[
\{ \begin{array}{l}
?(x \geq \frac{(A \pm \epsilon)T + v}{2(B \pm \epsilon)} + obs); a := A \pm \epsilon \cup a := -B \pm \epsilon; \\
c := 0; \{ x' = v, v' = a, c' = 1 \land v \geq 0 \land c \leq T \}
\end{array} \}
\]

\]x \leq obs}
Example: Adding Actuation Error

\[ A > 0 \land B > 0 \land T > 0 \land v \geq 0 \land 0 < \epsilon < A \land \epsilon < B \land \]
\[ \frac{v^2}{2B-\epsilon} + obs \leq x \leq obs \rightarrow \]
\[ \{ \{ ?(x \geq \frac{((A+\epsilon)T+v)^2}{2(B-\epsilon)} + obs); a := A+\epsilon \cup a := -B-\epsilon \}; \]
\[ c := 0; \{ x' = v, v' = a, c' = 1 \land v \geq 0 \land c \leq T \} \}
\]
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+\epsilon,B-\epsilon), 1) <(
  QE, QE,
  splitCases(1) <(
    chase(1) & ODE & QE
    chase(1) & ODE & QE
  )
))
```
✓ 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)
System $\alpha$ refines system $\beta$ ($\alpha \leq \beta$) if every state reachable by $\alpha$ is also reachable by $\beta$. 

\begin{center}
\begin{tikzpicture}
  \node[shape=circle,draw=black] (A) at (0,0) {$\nu$};
  \node[shape=circle,draw=black] (B) at (1,1) {$\omega_1$};
  \node[shape=circle,draw=black] (C) at (1,-1) {$\omega_2$};
  \node[shape=circle,draw=black] (D) at (2,0) {$\omega_3$};
  \draw[->,thick,red] (A) -- (B) node [above] {$\beta$};
  \draw[->,thick,red] (A) -- (C) node [above] {$\beta$};
  \draw[->,dashed,blue] (A) -- (D) node [above] {$\alpha$};
  \draw[->,thick,red] (B) -- (D) node [above] {$\beta$};
  \draw[->,thick,red] (C) -- (D) node [above] {$\beta$};
\end{tikzpicture}
\end{center}
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).
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:

$$\frac{[\beta] \varphi \quad \alpha \leq \beta}{[\alpha] \varphi}$$
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:
  
  $\begin{align*}
  [\beta]\varphi & \quad \alpha \leq \beta \\
  \underline{[\alpha]\varphi}
  \end{align*}$

- $\leq$ has a well-understood algebraic structure.
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 Misch, André Platzer, Brandon Bohrer, Jan-David Quesel)

Advertisement: KeYmaera X Tutorial at FM this year!
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 \(\text{dGLC}\); 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 (dGLC); 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!