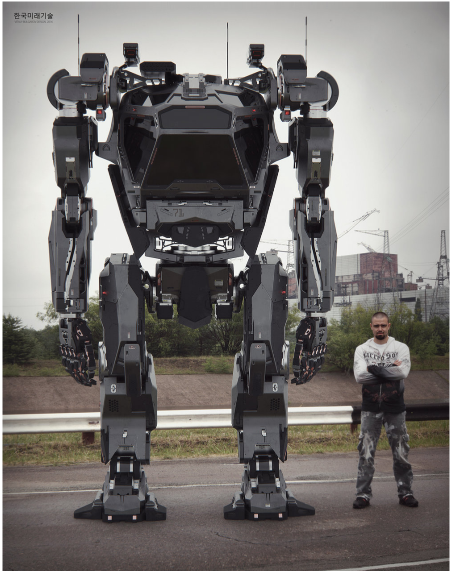
VeriPhy: Verified Controller Executables from Verified Cyber-Physical System Models

Brandon Bohrer[1], Yong Kiam Tan[1], Stefan Mitsch[1], Magnus O. Myreen[2], and André Platzer[1]

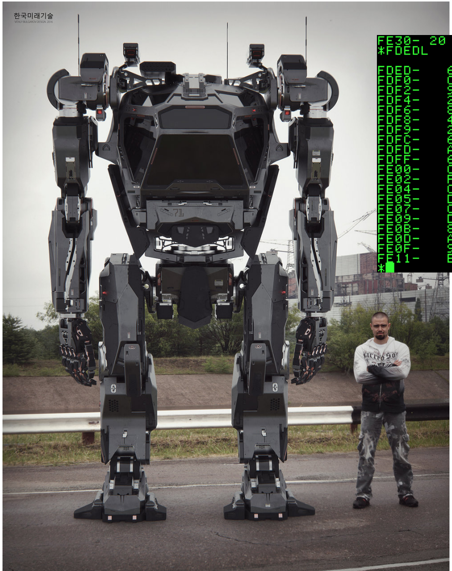
Carnegie Mellon University[1]
Chalmers University of Technology[2]

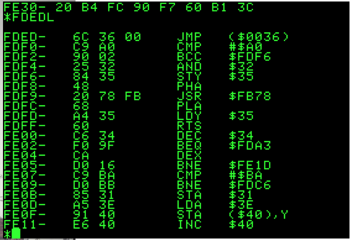
PLDI'18

A Real Cyber-Physical System



A Scary Cyber-Physical System

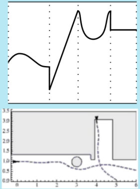




VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)

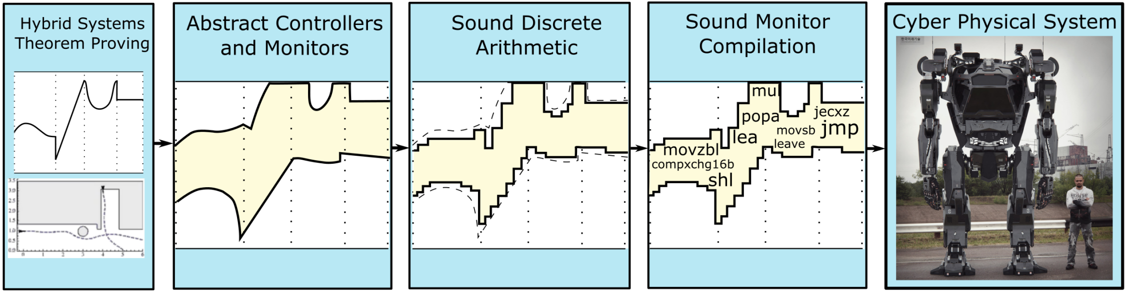
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)

Hybrid Systems Theorem Proving

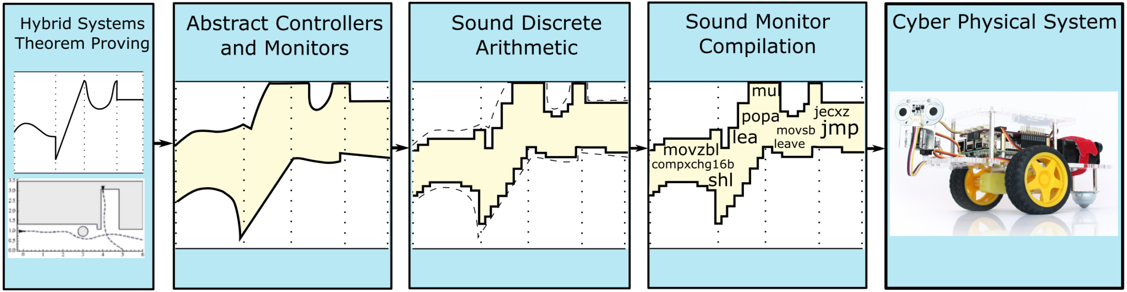


Cyber Physical System

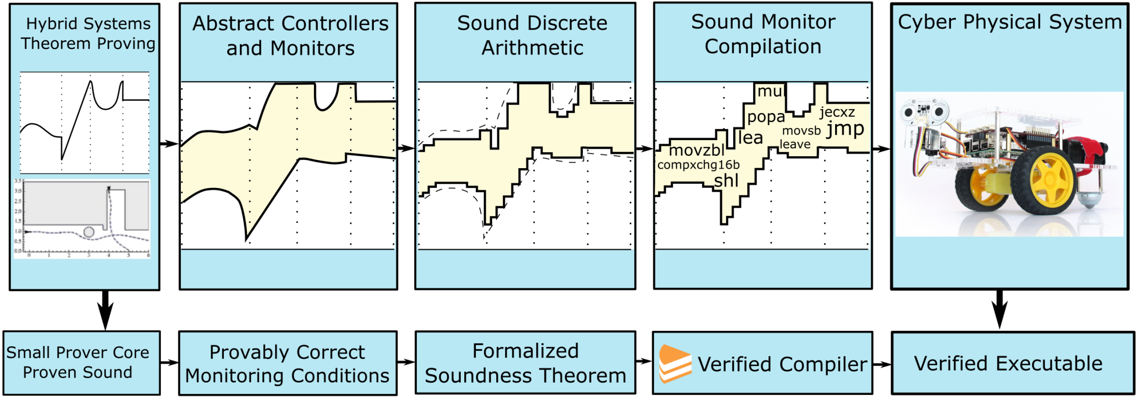
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



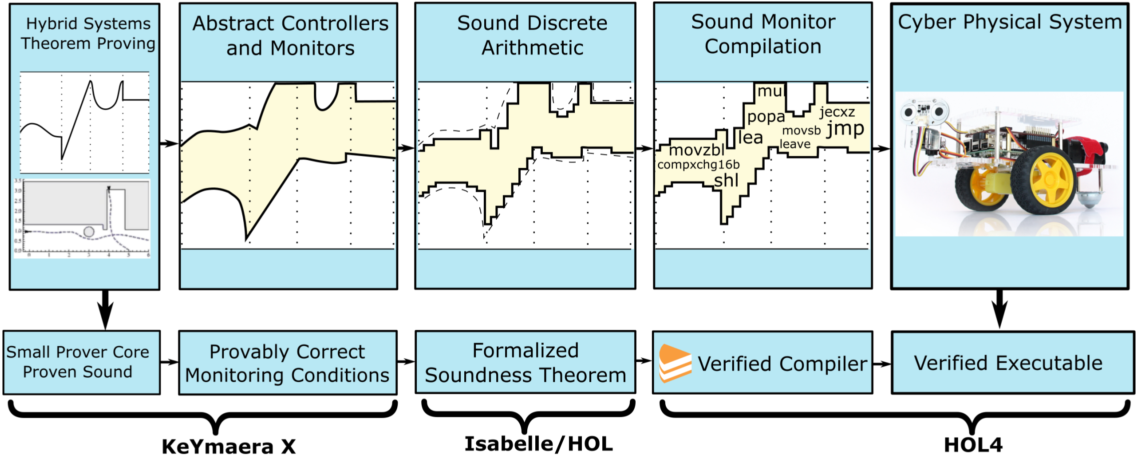
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



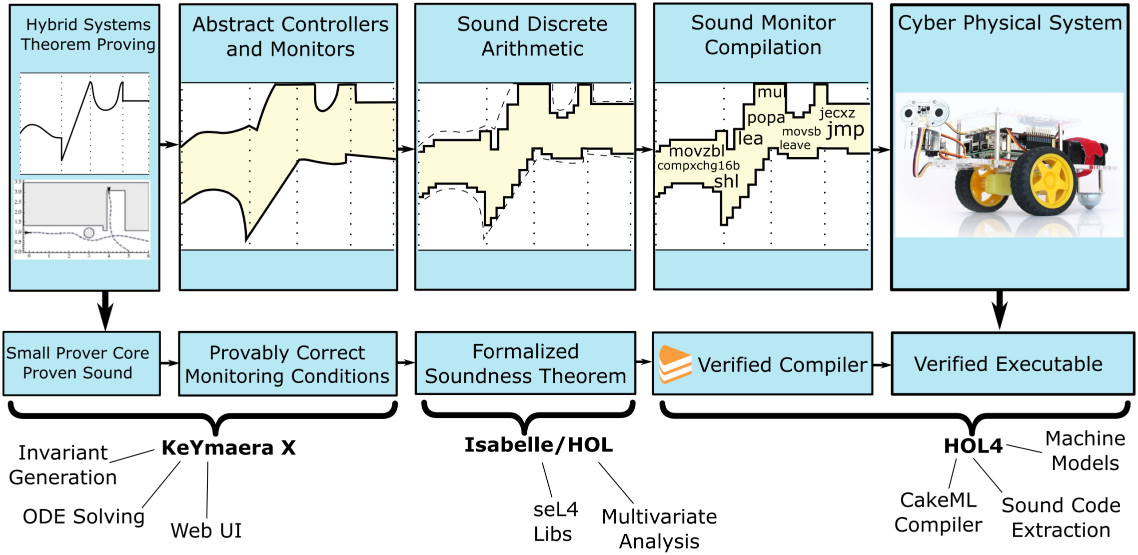
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



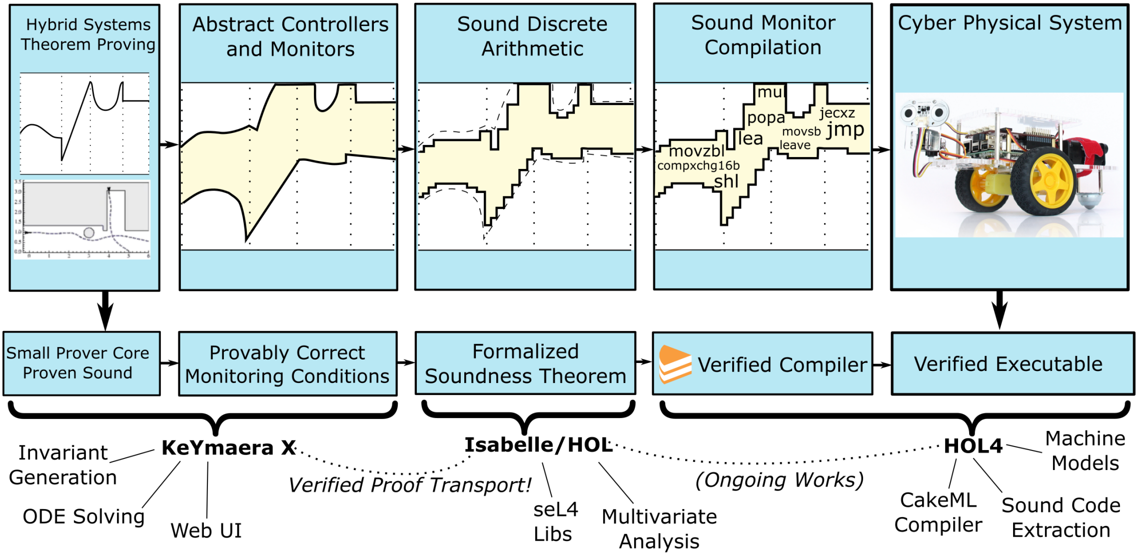
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



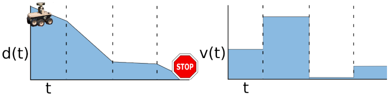
VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



VeriPhy: Automatic, Verified EXEs from Controllers (VeriPhy.org)



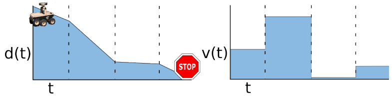
HPs Model Control and Environment



$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;$$

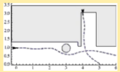
$$\overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$

HPs Model Control and Environment



$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1\ \&\ t \leq \varepsilon\}}^{env.}\Big)^*$$

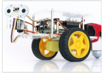
Far Enough?



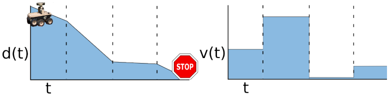
HP + dL Pf.

Monitors : Sandbox : Interval : CakeML : FFIs + EXE

KeYmaera X — Isabelle — HOL4



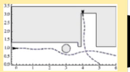
HPs Model Control and Environment



$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{\textit{drive}} \cup \overbrace{v := 0}^{\textit{stop}}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{\textit{env.}}\Big)^*$$

Far Enough?

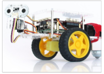
Velocity Envelope



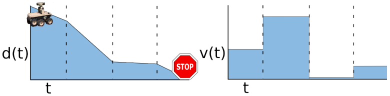
HP + dL Pf.

Monitors | Sandbox | Interval | CakeML | FFIs + EXE

KeYmaera X — Isabelle — HOL4



HPs Model Control and Environment

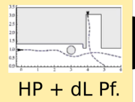


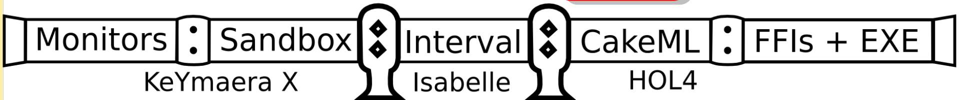
$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$

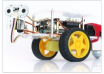
Far Enough?

Velocity Envelope

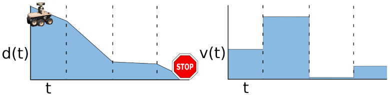
Fallback







HPs Model Control and Environment



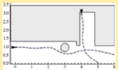
$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$

Far Enough?

Physics

Velocity Envelope

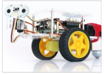
Fallback



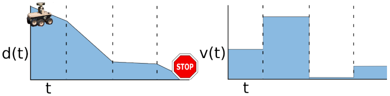
HP + dL Pf.

Monitors | Sandbox | Interval | CakeML | FFIs + EXE

KeYmaera X — Isabelle — HOL4



HPs Model Control and Environment



$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$

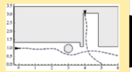
Far Enough?

Physics

Constraint

Velocity Envelope

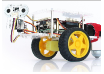
Fallback



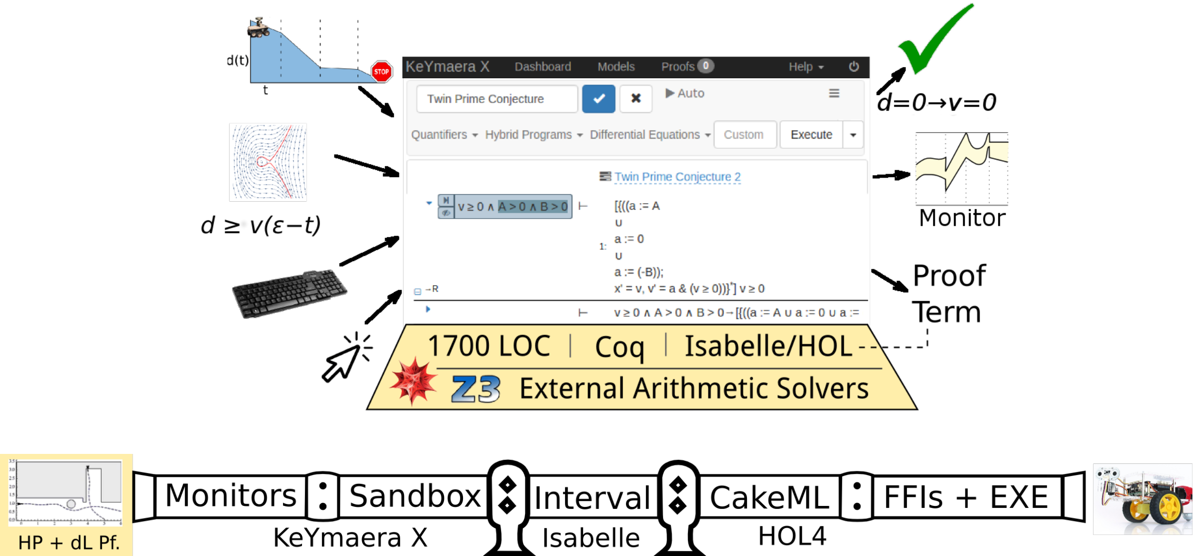
HP + dL Pf.

Monitors : Sandbox | Interval | CakeML : FFIs + EXE

KeYmaera X | Isabelle | HOL4



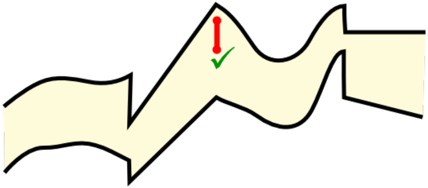
KeYmaera X Enables Model Verification



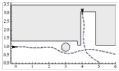
ModelPlex: Provably Correct Monitors

Monitor whether transitions from previous state $\vec{x}$ to next state $\vec{x}^+$ are consistent with control, environment models.

$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$



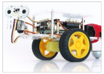
Control Monitor



HP + dL Pf.

Monitors : Sandbox | Interval | CakeML : FFIs + EXE

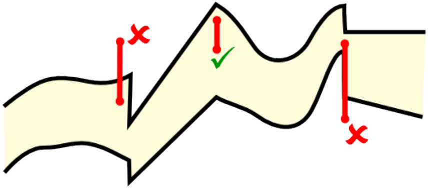
KeYmaera X — Isabelle — HOL4



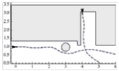
ModelPlex: Provably Correct Monitors

Monitor whether transitions from previous state $\vec{x}$ to next state $\vec{x}^+$ are consistent with control, environment models.

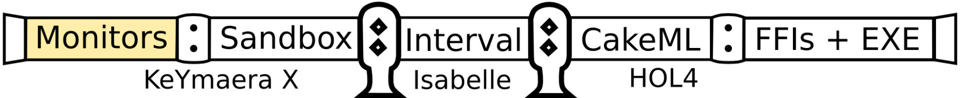
$$\alpha \equiv \Big(\big(\overbrace{?d \geq \varepsilon V;\ v := *;\ ?0 \leq v \leq V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \leq \varepsilon\}}^{env.}\Big)^*$$

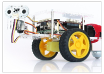


Control Monitor



HP + dL Pf.

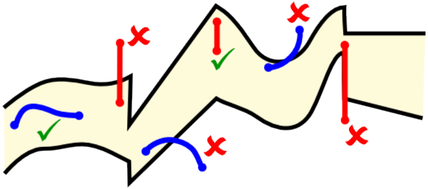




ModelPlex: Provably Correct Monitors

Monitor whether transitions from previous state $\vec{x}$ to next state $\vec{x}^+$ are consistent with control, environment models.

$$\alpha \equiv \Big(\big(\overbrace{?d \ge \varepsilon V;\ v := *;\ ?0 \le v \le V}^{drive} \cup \overbrace{v := 0}^{stop}\big);\ t := 0;\ \overbrace{\{d' = -v,\ t' = 1 \,\&\, t \le \varepsilon\}}^{env.}\Big)^*$$

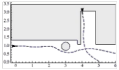


Control Monitor
Plant Monitor

Provable Monitor $\rightsquigarrow$ Provable Sandbox

Sandboxed controller uses external controller when decision is safe, else uses verified fallback. Detects non-compliant plants.

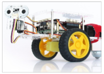
$$
\begin{array}{l}
\vec{x} := *;\\
?\phi\\
(\ \vec{x}^+ := \text{extCtrl}\\
\quad (\ \ ?\text{ctrlMon}(\vec{x}, \vec{x}^+)\\
\quad\ \ \cup\ \text{fallback}\ );\\
\quad \vec{x} := \vec{x}^+\\
\quad \vec{x}^+ := *\\
\quad ?\text{plantMon}(\vec{x}, \vec{x}^+);\\
\quad \vec{x} := \vec{x}^+)^*
\end{array}
\qquad
\begin{array}{l}
V := *;\ \varepsilon := *;\ d := *;\ t := *;\\
?d \geq 0 \land V \geq 0 \land \varepsilon \geq 0;\\
(\ t^+ := *;\ v^+ := *;\ d^+ := d;\\
\quad (\ \ ?\text{ctrlMon}(d, t, v, d^+, t^+, v^+)\\
\quad\ \ \cup\ t^+ := 0;\ v^+ := 0\ );\\
\quad t := t^+;\ v := v^+;\\
\quad d^+ := *;\ t^+ := *;\\
\quad ?\text{plantMon}(d, t, v, d^+, t^+, v^+);\\
\quad d := d^+;\ t := t)^*
\end{array}
$$



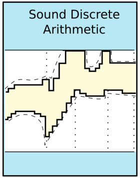
HP + dL Pf.

Monitors : Sandbox ⬥ Interval ⬥ CakeML : FFIs + EXE

KeYmaera X | Isabelle | HOL4



Intervals Make ctrlMon and plantMon Computable

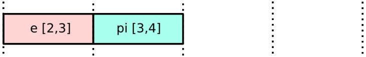


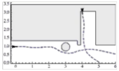
Example: Check whether $\pi < e$, efficiently.
Solution: Conservative interval approximation

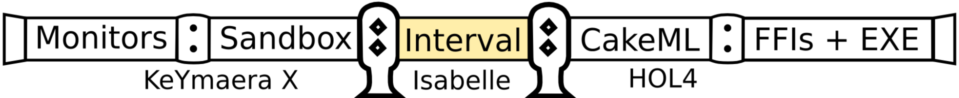
Example

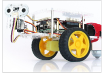
Let $\nu_I = \{pi \mapsto [3,4], e \mapsto [2,3]\}$, then

- $pi <_w e$ is false $(\perp)$

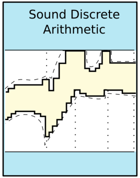








Intervals Make ctrlMon and plantMon Computable

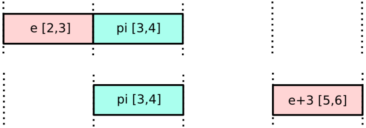


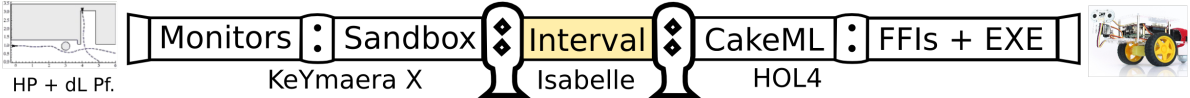
Example: Check whether $\pi < e$, efficiently.
Solution: Conservative interval approximation

Example

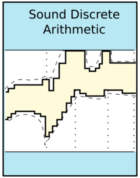
Let $\nu_I = \{pi \mapsto [3,4], e \mapsto [2,3]\}$, then

- $pi <_w e$ is false ($\bot$)
- $pi <_w e + 3$ is true ($\top$)





Intervals Make ctrlMon and plantMon Computable

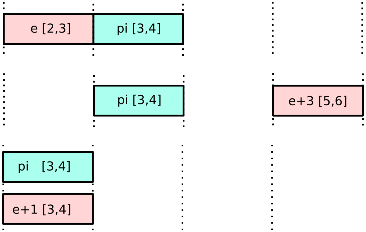


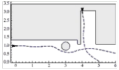
Example: Check whether $\pi < e$, efficiently.
Solution: Conservative interval approximation

Example

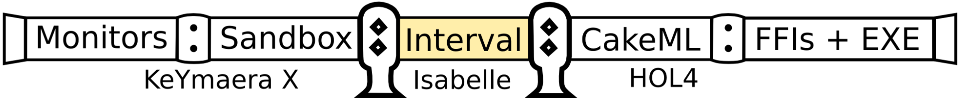
Let $\nu_I = \{pi \mapsto [3,4], e \mapsto [2,3]\}$, then

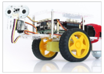
- $pi <_w e$ is false ($\bot$)
- $pi <_w e+3$ is true ($\top$)
- $pi <_w e+1$ is *a known unknown* (U)



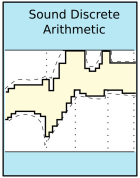


HP + dL Pf.





Intervals Make ctrlMon and plantMon Computable

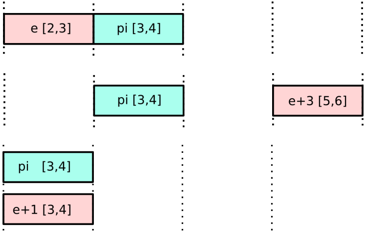


Example: Check whether $\pi < e$, efficiently.
Solution: Conservative interval approximation

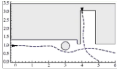
Example

Let $\nu_I = \{pi \mapsto [3,4], e \mapsto [2,3]\}$, then

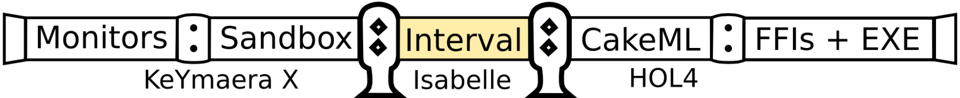
- $pi <_w e$ is false ($\bot$)
- $pi <_w e + 3$ is true ($\top$)
- $pi <_w e + 1$ is *a known unknown* (U)

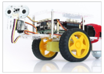


When truth values can be unknown, resulting logic is *3-valued*



HP + dL Pf.





Interval d$\mathcal{L}$ is 3-Valued (Łukasiewicz)

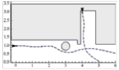
$\wedge$	$\top$	U	$\bot$
$\top$	$\top$	U	$\bot$
U	U	U	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

$\vee$	$\top$	U	$\bot$
$\top$	$\top$	$\top$	$\top$
U	$\top$	U	U
$\bot$	$\top$	U	$\bot$

$$\omega_I[\![(\theta_1 + \theta_2)]\!] = [l_1 \check{+}_w l_2, u_1 \hat{+}_w u_2] \text{ where } \omega_I[\![(\theta_i)]\!] = [l_i, u_i]$$

$$\omega_I[\![(\theta_1 < \theta_2)]\!] = \begin{cases} \top & \text{if } \omega_I[\![(\theta_i)]\!] = (l_i, u_i) \text{ and } u_1 < l_2 \\ \bot & \text{if } \omega_I[\![(\theta_i)]\!] = (l_i, u_i) \text{ and } l_1 \geq u_2 \\ U & \text{otherwise} \end{cases}$$

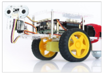
$$(\omega_I, \nu_I) \in [\![(\alpha \cup \beta)]\!] \text{ iff } (\omega_I, \nu_I) \in [\![(\alpha)]\!] \text{ or } (\omega_I, \nu_I) \in [\![(\beta)]\!]$$



HP + dL Pf.

Monitors | Sandbox | Interval | CakeML | FFIs + EXE

KeYmaera X | Isabelle | HOL4

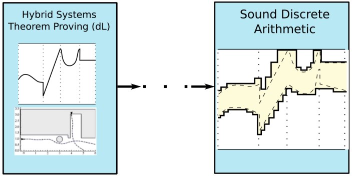


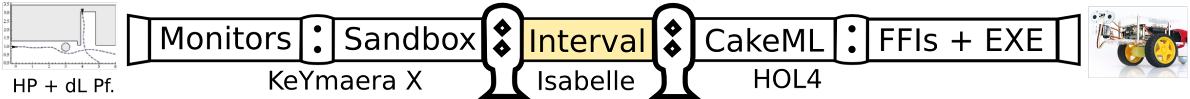
Interval d$\mathcal{L}$ is a Sound Approximation

Theorem (Interval Soundness for Formulas)

- *If* $\omega \in \omega_I$ *and* $\omega_I(\!(\phi)\!) = \top$ *then* $\omega \in [\![\phi]\!]$
- *If* $\omega \in \omega_I$ *and* $\omega_I(\!(\phi)\!) = \bot$ *then* $\omega \notin [\![\phi]\!]$
- *No claims when* $\omega_I(\!(\phi)\!) = U$

Generalizes naturally to programs, but CakeML sandbox only runs simpler formula case





Sandbox HP Already Verified

$V := *;\ \varepsilon := *;\ d := *;\ t := *;$ $\qquad // \ \vec{x} := *$

$?d \geq 0 \wedge V \geq 0 \wedge \varepsilon \geq 0;$ $\qquad // \ ?\phi$

$\big(\ t^+ := *;\ v^+ := *;\ d^+ := d;$ $\qquad // \ \vec{x}^+ := \mathrm{extCtrl}$

$\quad (\quad ?\mathrm{ctrlMon}(d, t, v, d^+, t^+, v^+)$

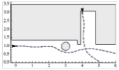
$\quad\quad \cup\ t^+ := 0;\ v^+ := 0\ );$ $\qquad // \ \vec{x}^+ := \mathrm{fallback}$

$\quad t := t^+;\ v := v^+;$ $\qquad // \ \vec{x} := \vec{x}^+$

$\quad d^+ := *;\ t^+ := *;$ $\qquad // \ \vec{x}^+ := *$

$\quad ?(0 \leq t^+ \leq \varepsilon \wedge d^+ \geq v(\varepsilon - t^+));$ $\qquad // \ ?\mathrm{plantMon}(\vec{x}, \vec{x}^+)$

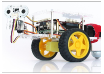
$\quad d := d^+;\ t := t^+$ $\qquad // \ \vec{x} := \vec{x}^+ \big)^*$



HP + dL Pf.

Monitors | Sandbox | Interval | CakeML | FFIs + EXE

KeYmaera X | Isabelle | HOL4



Verified CakeML Source is Generated

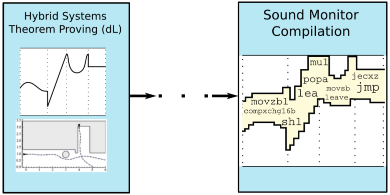
CakeML source incorporates external control, actuation, sensing

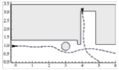
```
fun cmlSandbox state =
    if not (stop ()) then
      state.ctrl+ := extCtrl state;
      state.ctrl := if intervalSem ctrlMon state = ⊤
                    then state.ctrl+
                    else fallback state;
      actuate state.ctrl;
      state.sensors+ := sense ();
      if intervalSem plantMon state = ⊤ then
        Runtime.fullGC ();
        state.sensors := state.sensors+;
        cmlSandbox state
      else violation "Plant Violation"
```

CakeML Sandbox is Sound

Theorem (Soundness for CakeML Sandbox, Main Case)

If $(\llbracket\{\omega\}\rrbracket, \llbracket\{\nu\}\rrbracket) \in \llbracket\{\text{cmlSandbox}\}\rrbracket$ *then* $(\llbracket(\omega)\rrbracket, \llbracket(\nu)\rrbracket) \in \llbracket(\text{sandbox})\rrbracket$

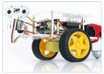




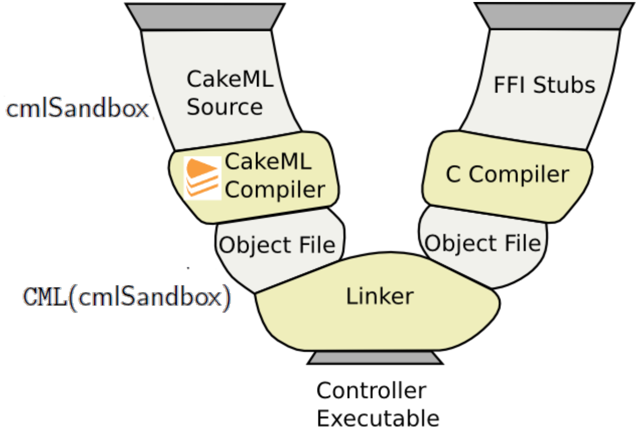
HP + dL Pf.

Monitors : Sandbox : Interval : CakeML : FFIs + EXE

KeYmaera X — Isabelle — HOL4

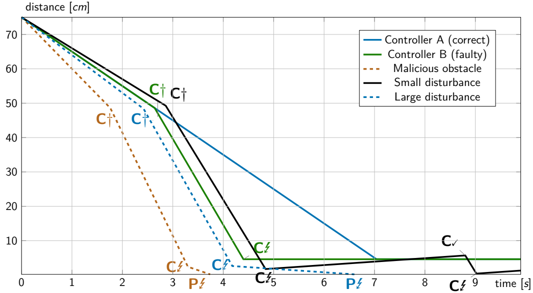


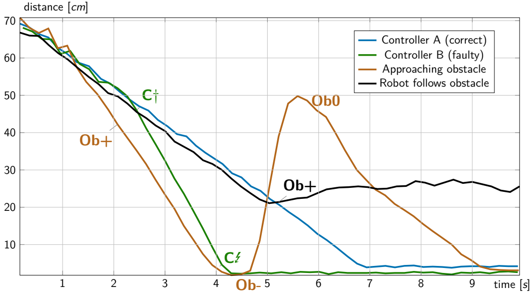
CakeML Compiler Preserves Guarantees



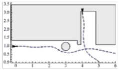
Code Executed on GoPiGo Robot

Operational Suitability?
Arithmetic Precision?

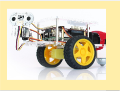




Control Fault **C↯**, Plant Fault **P↯**, Control Spike **C†**, Obstacle Motion **Ob**



HP + dL Pf.



Proof Chain Justifies Transformations

$$\nu \models \psi$$

$$\Uparrow$$

$$(\omega, \nu) \in \underbrace{[\![\text{sandbox}]\!]}_{\text{d}\mathcal{L}\ \text{(KeYmaera X)}}$$

Real arithmetic, nondeterministic

$$\Uparrow$$

$$(\omega_I, \nu_I) \in \underbrace{[\!(\text{sandbox})\!]}_{\text{d}\mathcal{L}\ \text{(Isabelle/HOL)}}$$

Interval word arithmetic, nondeterministic

$$\Uparrow$$

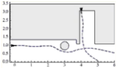
$$([\!\{\omega\}\!], [\!\{\nu\}\!]) \in \underbrace{[\!\{\text{cmlSandbox}\}\!]}_{\text{CakeML (HOL4)}}$$

Interval word arithmetic, deterministic

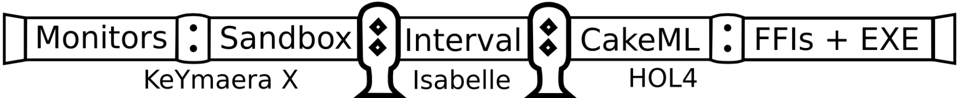
$$\Uparrow$$

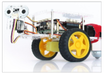
$$(\{\!|\omega|\!\}, \{\!|\nu|\!\}) \in \underbrace{\{\!|\mathtt{CML}(\text{cmlSandbox})|\!\}}_{\text{ARM/x64}}$$

Interval word arithmetic, machine-executable



HP + dL Pf.

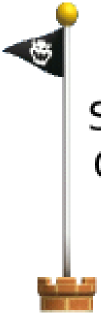




Takeaway Metaphor

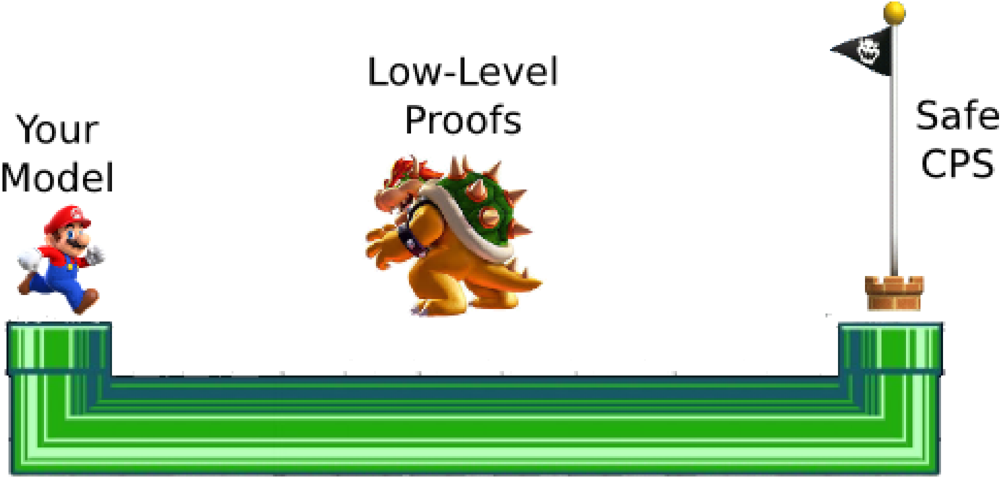
Your Model

Low-Level Proofs



Safe CPS

Takeaway Metaphor



VeriPhy Pipeline (VeriPhy.org)

References I

Brandon Bohrer, Vincent Rahli, Ivana Vukotic, Marcus Völp, and André Platzer, *Formally verified differential dynamic logic*, Certified Programs and Proofs - 6th ACM SIGPLAN Conference, CPP 2017, Paris, France, January 16-17, 2017 (Yves Bertot and Viktor Vafeiadis, eds.), ACM, 2017, pp. 208–221.

Joe Hurd, *The OpenTheory standard theory library*, NFM (Mihaela Gheorghiu Bobaru, Klaus Havelund, Gerard J. Holzmann, and Rajeev Joshi, eds.), LNCS, vol. 6617, Springer, 2011, pp. 177–191.

Magnus O. Myreen and Scott Owens, *Proof-producing synthesis of ML from higher-order logic*, ICFP (Peter Thiemann and Robby Bruce Findler, eds.), ACM, 2012, pp. 115–126.

Isabelle/HOL Cross-Checks KeYmaera X

Problem: Later pipeline stages need understanding of d$\mathcal{L}$ semantics, which KeYmaera X lacks

Isabelle/HOL Cross-Checks KeYmaera X

Problem: Later pipeline stages need understanding of d$\mathcal{L}$ semantics, which KeYmaera X lacks

Solution: Import soundly into Isabelle/HOL from KeYmaera X

- Proof term exported from KeYmaera X, serialized
- Proof checker verified in Isabelle/HOL, extending [BRV+17]

Isabelle/HOL Cross-Checks KeYmaera X

Problem: Later pipeline stages need understanding of d$\mathcal{L}$ semantics, which KeYmaera X lacks

Solution: Import soundly into Isabelle/HOL from KeYmaera X

- Proof term exported from KeYmaera X, serialized
- Proof checker verified in Isabelle/HOL, extending [BRV$^+$17]
- Executable checker code-generated [MO12]
- Scales to 100K's of proof steps ($\approx$6 seconds)
- *Eliminates KeYmaera X core from trusted base!*

Isabelle/HOL → HOL4 Translation is Trusted

Isabelle/HOL Strength: Library Access

- Analysis libraries (absolute must for d$\mathcal{L}$ soundness)
- Machine word libraries (must for interval arithmetic)

Isabelle/HOL → HOL4 Translation is Trusted

Isabelle/HOL Strength: Library Access

- Analysis libraries (absolute must for d$\mathcal{L}$ soundness)
- Machine word libraries (must for interval arithmetic)

Isabelle/HOL Weakness: Weaker Verified Compiler Support

- This is a problem: need to generate source code!

Isabelle/HOL → HOL4 Translation is Trusted

Isabelle/HOL Strength: Library Access

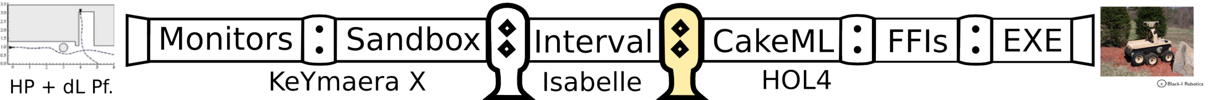
- Analysis libraries (absolute must for d$\mathcal{L}$ soundness)
- Machine word libraries (must for interval arithmetic)

Isabelle/HOL Weakness: Weaker Verified Compiler Support

- This is a problem: need to generate source code!

We jump to HOL4 for access to verified CakeML compiler:

- Manually translate Isabelle/HOL definitions to HOL4
- Justification: Similar logical foundation
- Could be automated in principle, see OpenTheory [Hur11]



Future Work

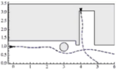
Improve pipeline components:

- Reduce trusted base: OpenTheory, arithmetic witnesses in KeYmaera X
- Floating-point, mixed precision interval arithmetic
- Generalize proof-driven monitor synthesis

Exploit pipeline in case studies:

- UAVs
- High-speed robots
- Your favorite CPS

(c) Black-I Robotics



HP + dL Pf.

(c) Black-I Robotics