Reachability Analysis in the KeYmaera X Theorem Prover

SNR 2017 | Uppsala, Sweden | April 22, 2017

Nathan Fulton

Other System Contributors: Stefan Mitsch, André Platzer, Brandon Bohrer, Yong Kiam Tan, Jan-David Quesel, ...

Trustworthy Foundations

Interactive Reachability Analysis

- Demonstration
- Bellerophon language and library

Automation and Tooling

Conclusions & Resources

KeYmaera X enables trustworthy automation for hybrid systems analysis:

- A well defined **logical foundations,**
- implemented in a **small trustworthy core**
- that ensures correctness of **automation** and **tooling**.

Trustworthy Foundations

Hybrid Programs

$a := t$

$\mathbf{a=a_0}$
$b=b_0$
$c=c_0$
...

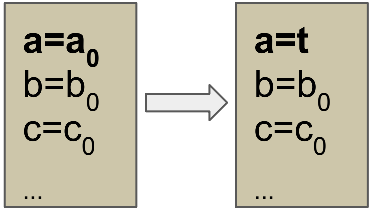
$\Rightarrow$

$\mathbf{a=t}$
$b=b_0$
$c=c_0$
...

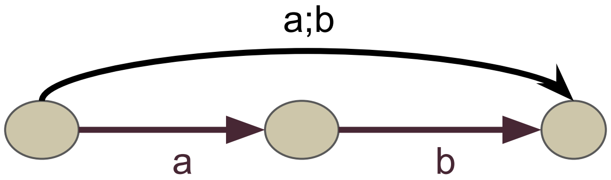
Trustworthy Foundations

Hybrid Programs

$a := t$

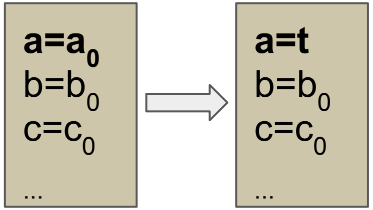


$a;b$

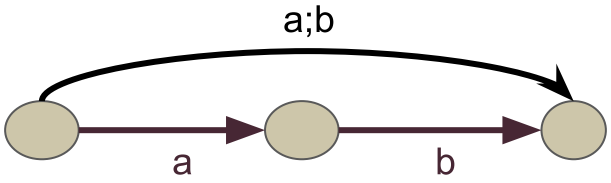


Hybrid Programs

a := t



a;b



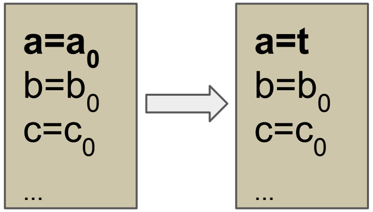
?P

If P is true: no change

If P is false: terminate

Hybrid Programs

a := t

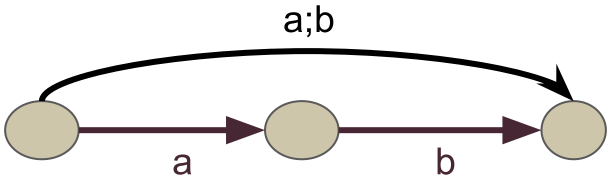


?P

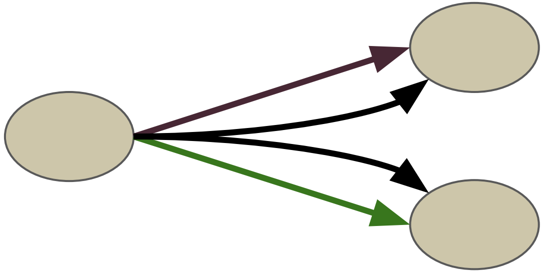
If P is true: no change

If P is false: terminate

a;b

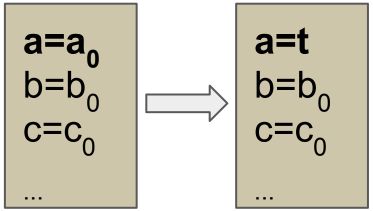


a U b

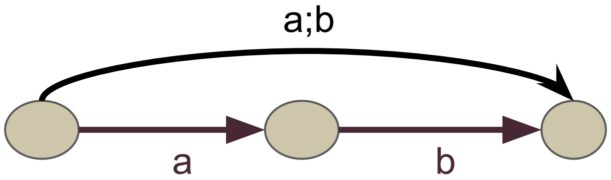


Trustworthy Foundations
Hybrid Programs

a := t



a;b

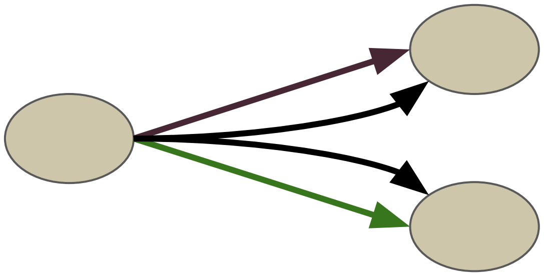


?P

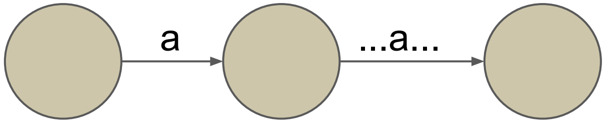
If P is true: no change

If P is false: terminate

a U b



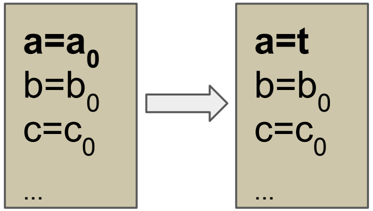
a*



Trustworthy Foundations

Hybrid Programs

$a := t$

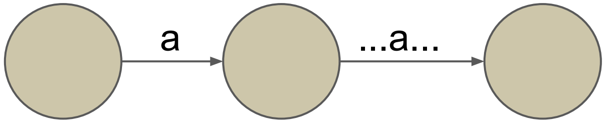


$?P$

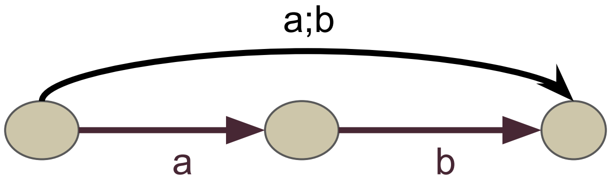
If P is true: no change

If P is false: terminate

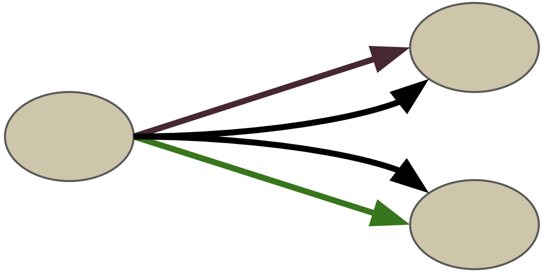
a^*



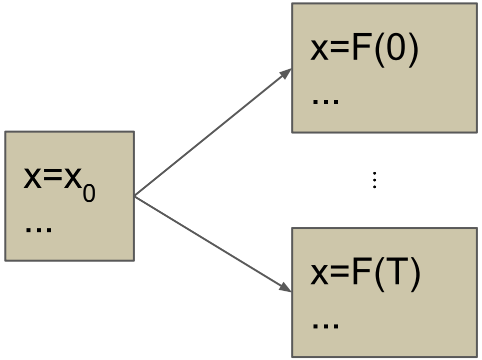
$a;b$



$a \cup b$

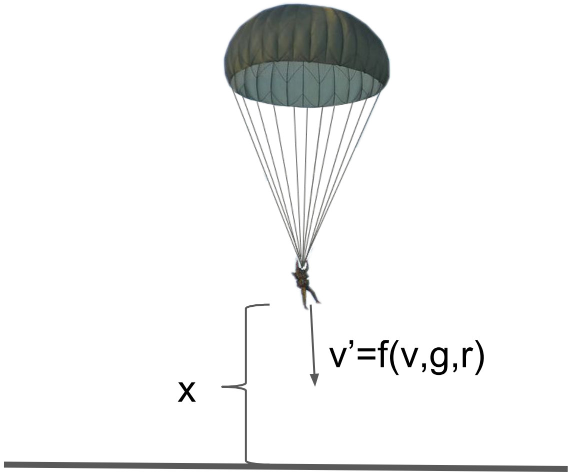


$x' = f$



Hello, World

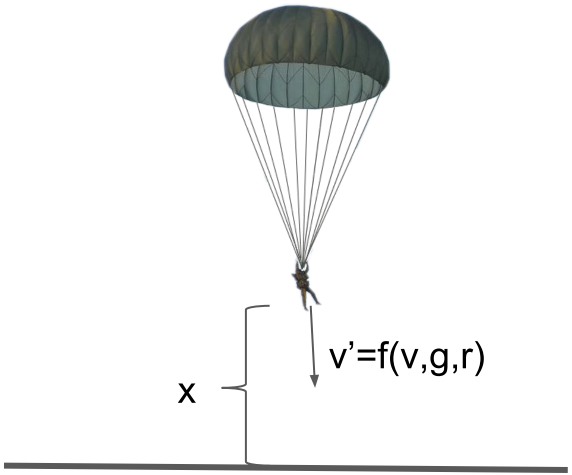
```
{
  {?Dive ∪ r := r_p};
  t:=0;
  {x' = v,
   V' = f(v,g,r), t'=1
   & 0≤x & t≤T}
}*
```



Control: Continue diving if safe, else open parachute.
Plant: Downward velocity determined by gravity, air resistance.

Hello, World

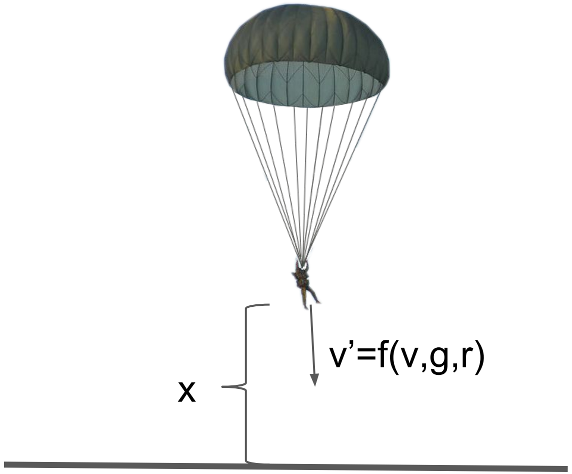
```
{
  {?Dive ∪ r := r_p};
  t:=0;
  {x' = v,
   V' = f(v,g,r), t'=1
   & 0≤x & t≤T}
}*
```



Control: Continue diving if safe, else open parachute.
Plant: Downward velocity determined by gravity, air resistance.

Hello, World

```
{
   {?Dive U r := r_p};
   t:=0;
   {x' = v,
    V' = f(v,g,r), t'=1
    & 0≤x & t≤T}
}*
```

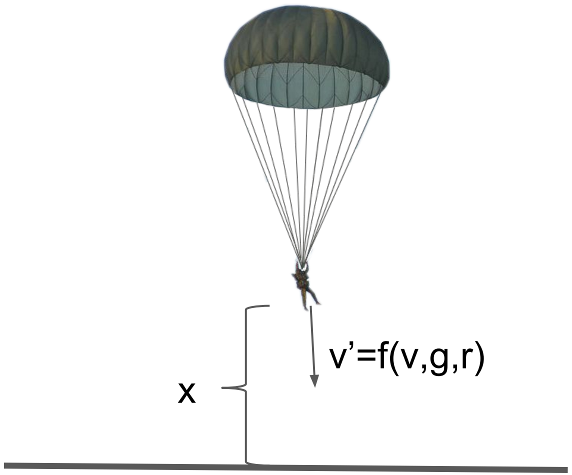


Control: Continue diving if safe, else open parachute.

Plant: Downward velocity determined by gravity, air resistance.

Hello, World

```
{
   {?Dive ∪ r := r_p};
   t:=0;
   {x′ = v,
    V′ = f(v,g,r), t′=1
    & 0≤x & t≤T}
}*
```

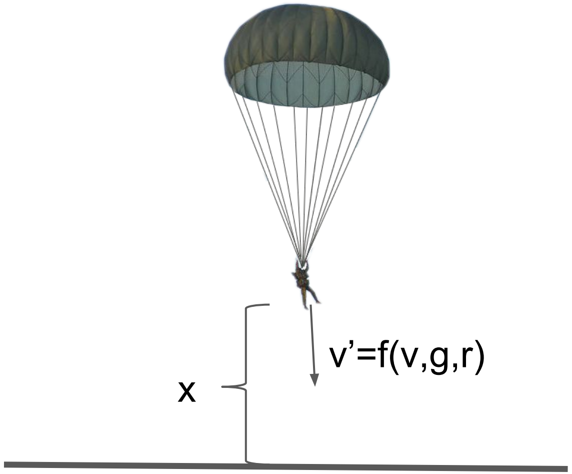


Control: Continue diving if safe, else open parachute.

Plant: Downward velocity determined by gravity, air resistance.

Hello, World

```
{
  {?Dive ∪ r := r_p};
  t:=0;
  {x' = v,
   V' = f(v,g,r), t'=1
   & 0≤x & t≤T}
}*
```



Control: Continue diving if safe, else open parachute.

Plant: Downward velocity determined by gravity, air resistance.

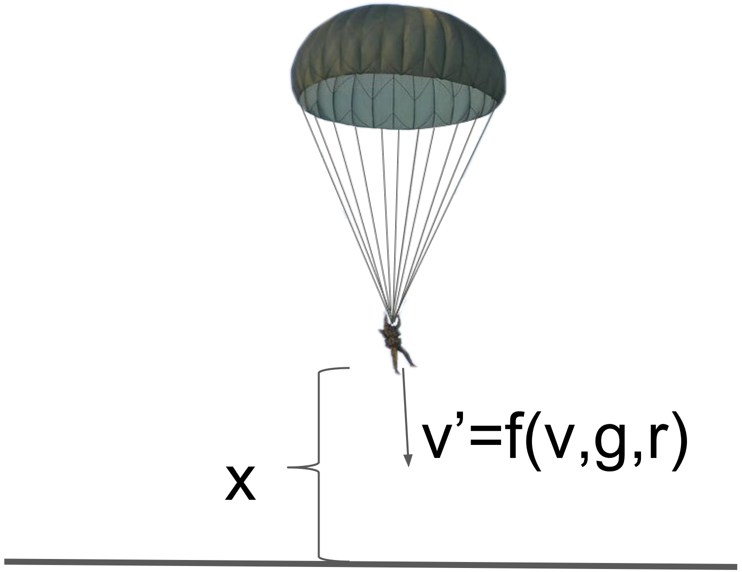
Reachability Specifications

[a]P “after every execution of a, P”

<a>P “after some execution of a, P”

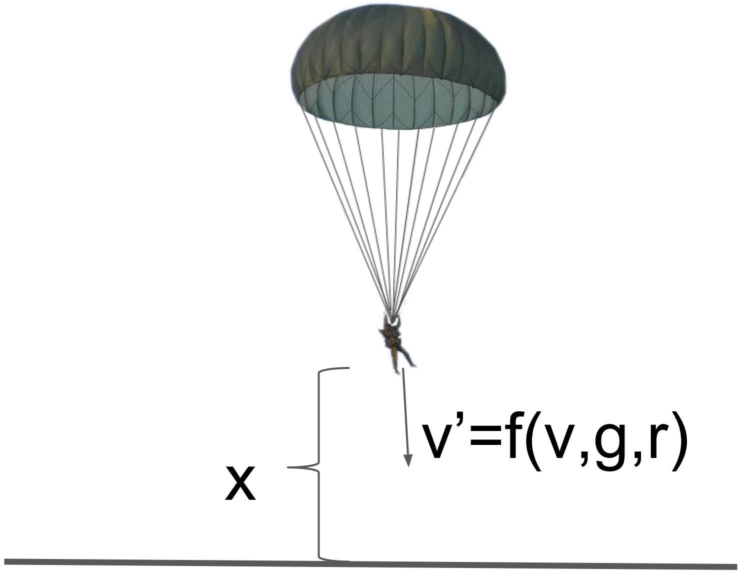
Reachability Specifications

$$
\begin{aligned}
&(\mathtt{Dive}\ \&\ g>0\ \&\ \ldots)\rightarrow\\
&[\{\\
&\quad \{?\mathtt{Dive} \cup r := r_p\};\\
&\quad \{x' = v,\\
&\quad\ V' = f(v,g,r)\\
&\quad\ \&\ 0 \le x\}\\
&\}^*]\ (x=0 \rightarrow m \le v)
\end{aligned}
$$



Reachability Specifications

```
(Dive & g>0 & …)→
[{
  {?Dive ∪ r := r_p};
  {x' = v,
   V' = f(v,g,r)
   & 0≤x}
}*] (x=0→m≤v)
```



$\underbrace{(x=0 \rightarrow m \leq v)}$

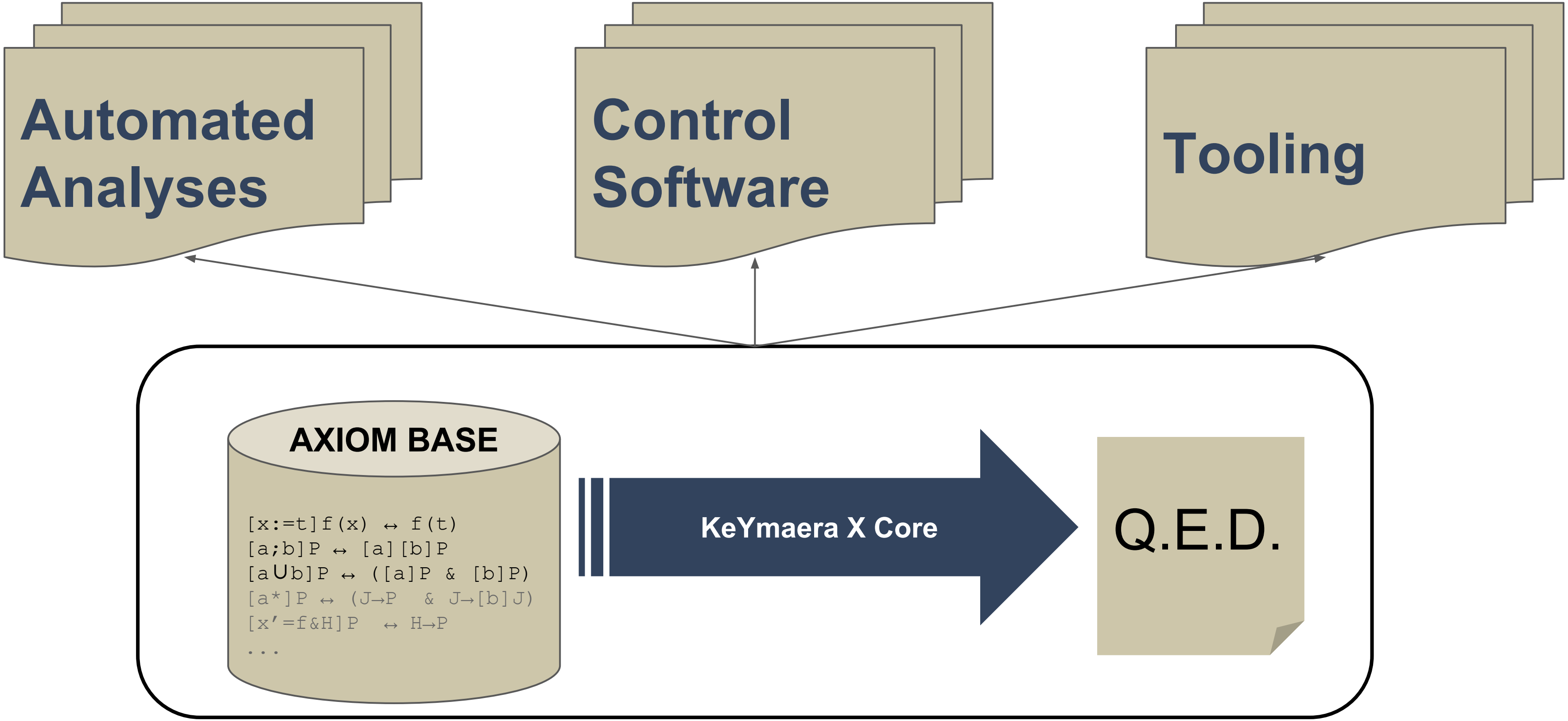
If the parachuter is on the ground, their speed is safe (m≤v≤0)

Dynamical Axioms

```
[x:=t]f(x)  ↔  f(t)
[a;b]P      ↔  [a][b]P
[a∪b]P      ↔  ([a]P & [b]P)
[a*]P       ↔  (J→P  & J→[b]J)
[x'=f&H]P   ↔  H→P
...
```

Trusted Core

Trustworthy Implementations



Prover Core Comparison

Tool	Trusted LOC (approx.)
KeYmaera X	**1,682 (out of 100,000+)**
KeYmaera	65,989
Isabelle/Pure	8,113
Coq	20,000
HSolver	20,000
dReal	50,000
SpaceEx	100,000

Interactive Reachability Analysis in KeYmaera X

KeYmaera X enables interactive verification and tool development:

Interactive Reachability Analysis in KeYmaera X

KeYmaera X enables interactive verification and tool development:

- A **standard library** of common proof techniques.

Interactive Reachability Analysis in KeYmaera X

KeYmaera X enables interactive verification and tool development:

- A **standard library** of common proof techniques.
- A **combinator language/library** for **decomposing** theorems and **composing** proof strategies.

Bellerophon

Tactic	Meaning
`prop`	Applies propositional reasoning exhaustively.
`unfold`	Symbolically executes discrete, loop-free programs.
`loop(J, i)`	Applies loop invariance axiom to position i.
`dI,dG,dC,dW`	Reasoning principles for differential equations.

Bellerophon

Tactic	Meaning
`prop`	Applies propositional reasoning exhaustively.
`unfold`	Symbolically executes discrete, loop-free programs.
`loop(J, 1)`	Applies loop invariance axiom to position i.
`dI,dG,dC,dW`	Reasoning principles for differential equations.

100+

Interactive Reachability Analysis in KeYmaera X

Bellerophon

100+

Tactic	Meaning
`prop`	Applies propositional reasoning exhaustively.
`unfold`	Symbolically executes discrete, loop-free programs.
`loop(J, i)`	Applies loop invariance axiom to position i.
`dI,dG,dC,dW`	Reasoning principles for differential equations.

Combinator	Meaning
A ; B	Execute A on current goal, then execute B on the result.
$A \mid B$	Try executing A on current goal. If A fails, execute B on current goal.
A*	Run A until it no longer applies.
$A<(B_0,B_1, \ldots ,B_N)$	Execute A on current goal to create N subgoals. Run B_i on subgoal i.

Interactive Reachability Analysis in KeYmaera X

Isolating Interesting Questions

(Dive & g>0
& …)→
[{

}*] **(x=0→m≤v)**

Interactive Reachability Analysis in KeYmaera X

Isolating Interesting Questions

(Dive & g>0
& …)→
[{

}*] **(x=0→m≤v)**

prop ; loop(J,1)

(Dive &
g>0 &
…)→
J

Loop invariant holds initially

J→[

]J

Loop invariant is preserved

J →
x=0→m≤v

Loop invariant implies safety

Isolating Interesting Questions

(Dive & g>0 & …)→
[{

}*] **(x=0→m≤v)**

prop ; loop(J,1)

(Dive & g>0 & …)→
J

Loop invariant holds initially

J→[

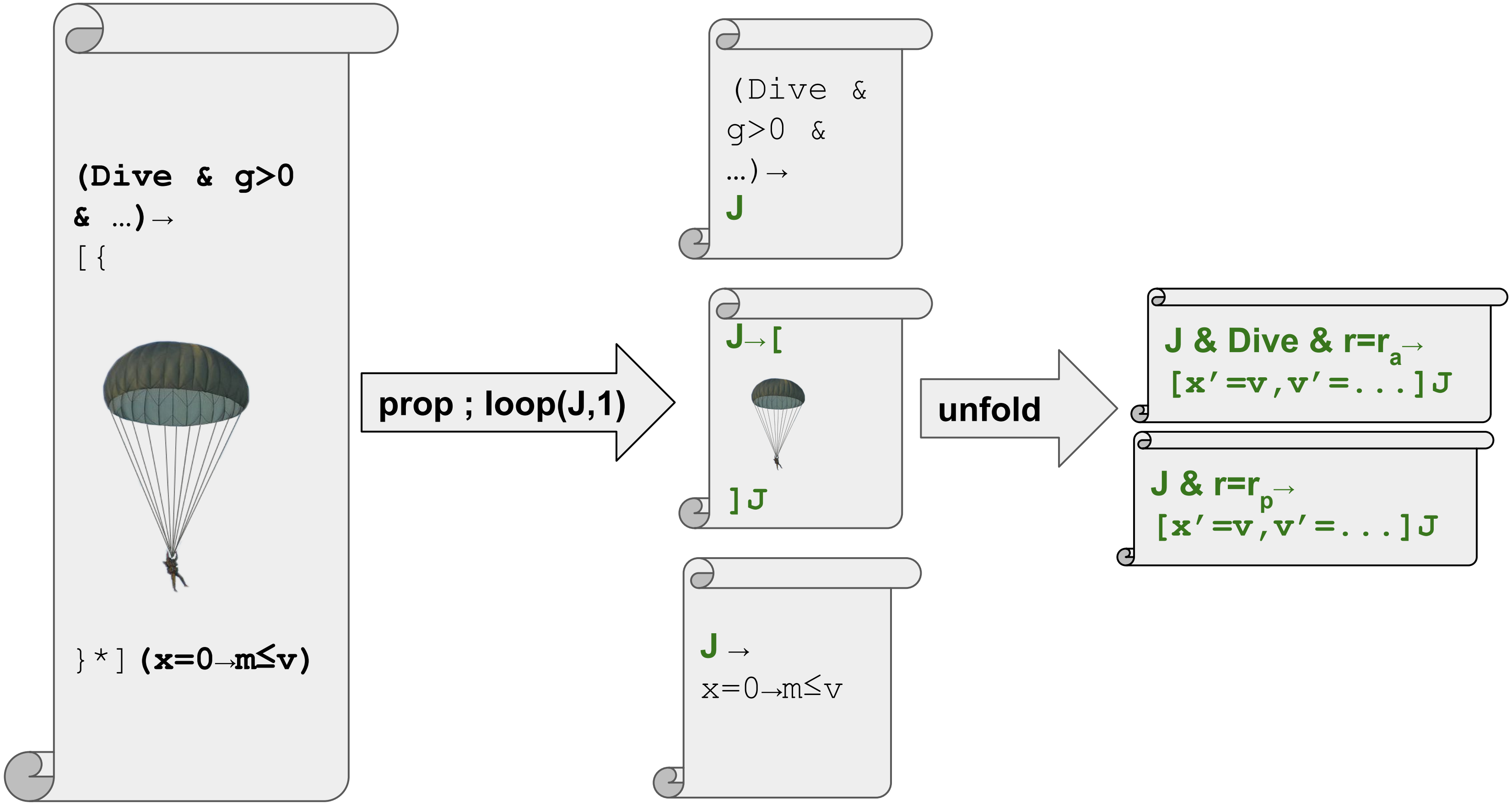
]**J**

Loop invariant is preserved

J →
x=0→m≤v

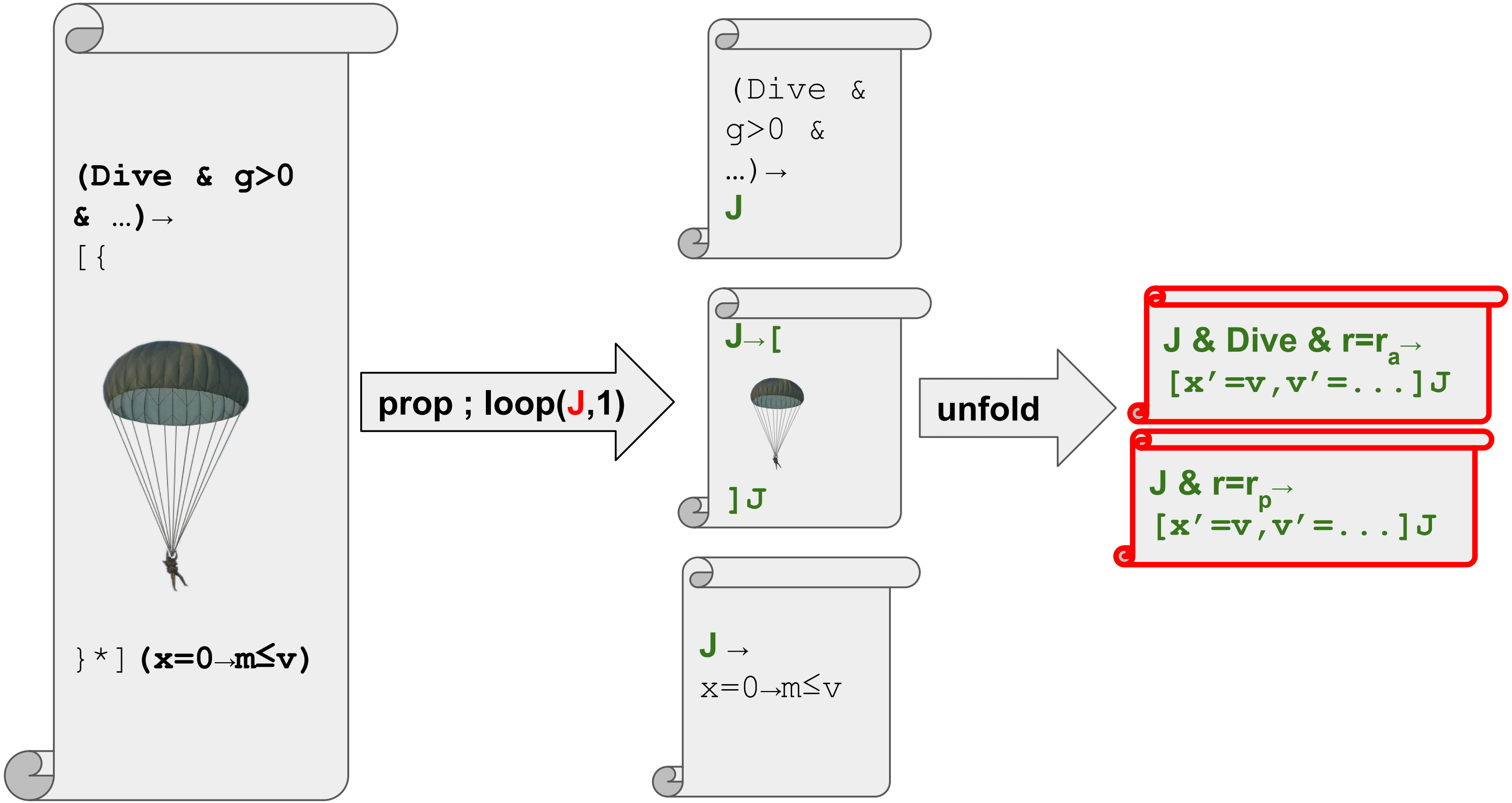
Loop invariant implies safety

Isolating Interesting Questions



Interactive Reachability Analysis in KeYmaera X

Isolating Interesting Questions



Isolating Interesting Questions

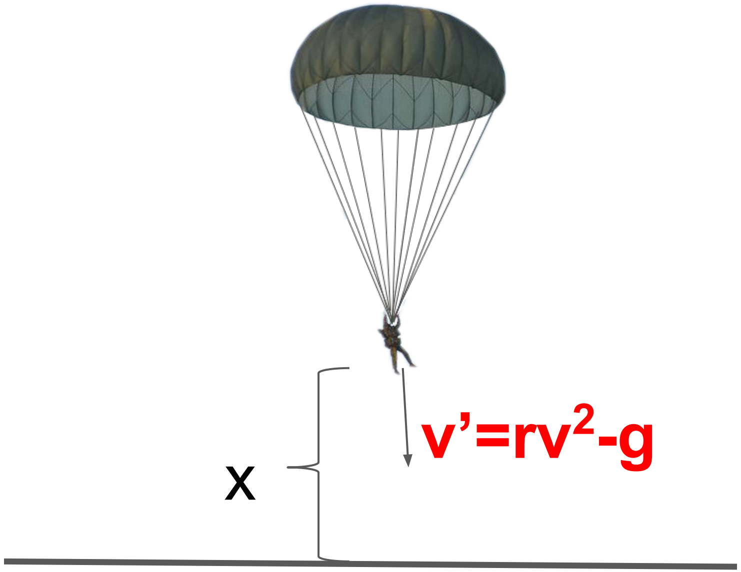
```
prop ;  loop(J, 1) <(
  QE, /* Real arith. solver */
  QE,
  Unfold <(
    … /* parachute open case */
    … /* parachute closed case */
  )
)
```

Differential Induction

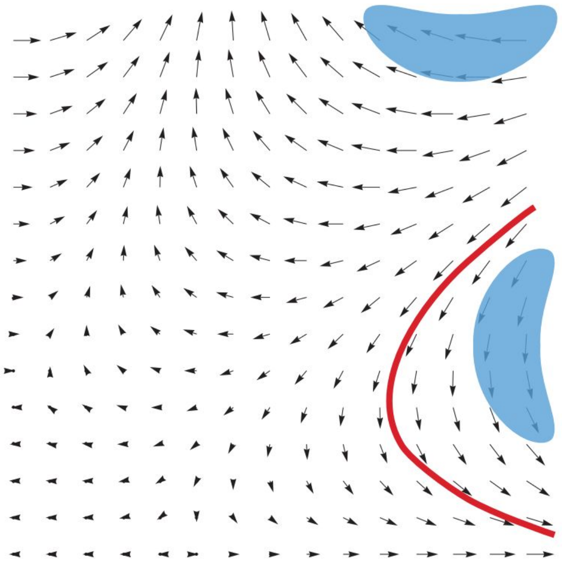
`J = v > -sqrt(g/pr) > m & ...`

Parachute Open Case:

$$\underbrace{\begin{aligned} v &\geq v_0 - gt \\ &\geq v_0 - gT \\ &> -\mathrm{sqrt}(g/pr) \end{aligned}}_{\text{Inductive invariants}}$$



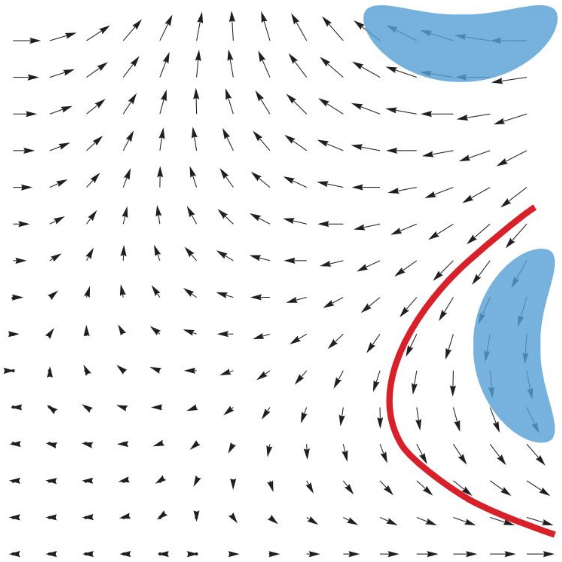
Differential Induction



DI Axiom:

[x'=f&H]P↔(P & (H→[x':=f]P'))

Differential Induction



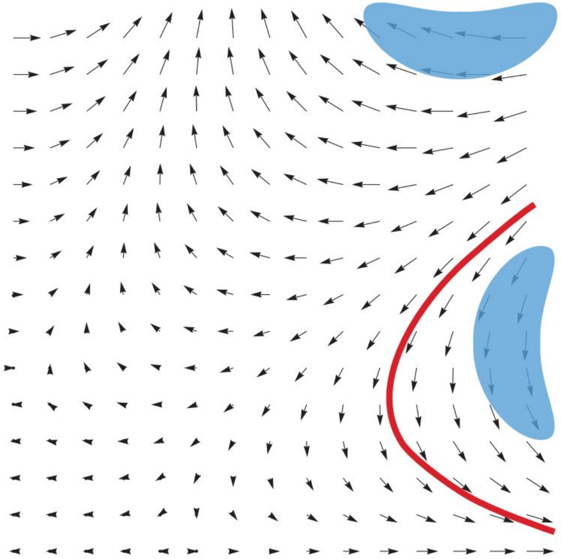
DI Axiom:

[x'=f&H]P↔(P & (H→[x':=f]P'))

Example:

$[v' = r_p v^2 - g, t' = 1]\, v \geq v_0 - gt$

Differential Induction



DI Axiom:

[x’=f&H]P↔(P & (H→[x’:=f]P’))

Example:

$[v'=r_p v^2-g, t'=1]v \geq v_0 - gt$ ↔

… ↔

$[v':=r_p v^2-g][t':=1]v' \geq -g*t'$ ↔

$r_p v^2-g \geq -g$ ↔

$r_p \geq 0$

Differential Induction

dI Tactic:

Side derivation:

$(v \geq v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0)' - (gt)' \leftrightarrow$

$(v)' \geq (v_0)' - (t(g)' + g(t')) \leftrightarrow$

$V' \geq v_0' - (tg' + gt')$

$H = r_p \geq 0 \ \& \ r_a \geq 0 \ \& \ g > 0 \ \& \ \ldots$

DI Axiom:

$[x' = f \& H]P \leftrightarrow (P \ \& \ (H \rightarrow [x' := f]P'))$

Example:

$[v' = r_p v^2 - g, t' = 1] v \geq v_0 - gt \leftrightarrow$

$\ldots \leftrightarrow$

$[v' := r_p v^2 - g][t' := 1] v' \geq -g * t' \leftrightarrow$

$r_p v^2 - g \geq -g \leftrightarrow$

$H \rightarrow r_p \geq 0$

Differential Induction

dI Tactic:

Side derivation:

$(v \geq v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0 - gt)' \leftrightarrow$

$(v)' \geq (v_0)' - (gt)' \leftrightarrow$

$(v)' \geq (v_0)' - (t(g)' + g(t')) \leftrightarrow$

$v' \geq v_0' - (tg' + gt')$

$H = r_p \geq 0$ & $r_a \geq 0$ & $g > 0$ & ...

DI Axiom:

$[x'=f\&H]P \leftrightarrow (P \;\&\; (H \rightarrow [x':=f]P'))$

Example:

$[v'=r_p v^2 - g, t'=1]v \geq v_0 - gt \leftrightarrow$

... $\leftrightarrow$

$[v':=r_p v^2 - g][t':=1]v' \geq -g*t' \leftrightarrow$

$r_p v^2 - g \geq -g \leftrightarrow$

$H \rightarrow r_p \geq 0$

Tactics recover a useful level of abstraction.

Pedantry is the price of trust.

Pedantry is the price of trust.

Bellerophon automates pedantic deductions.

Automation and Tooling

Hybrid Systems Analyses can be built on top of KeYmaera X.

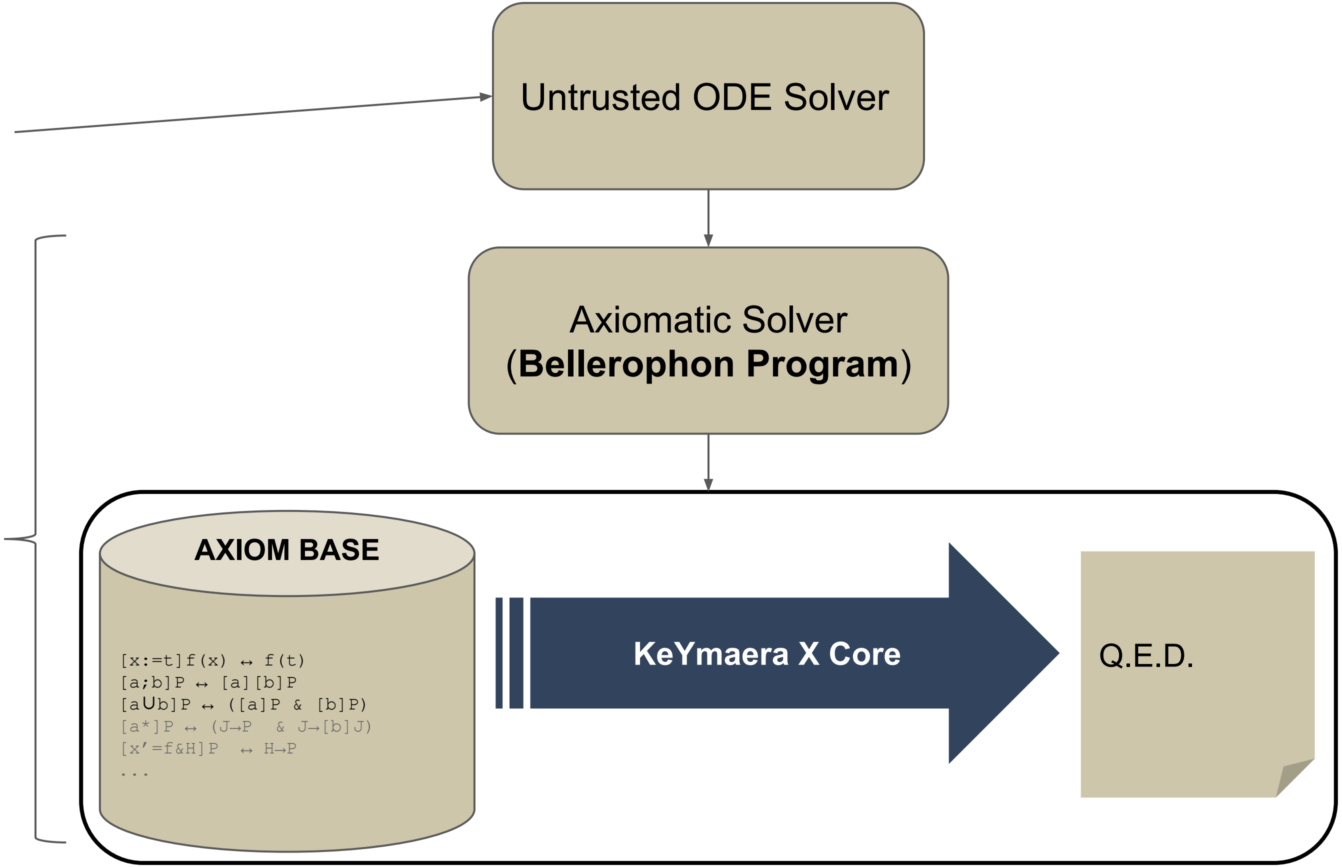
Examples:

- ODE Solver
- Runtime Monitoring

Toward Automated Deduction

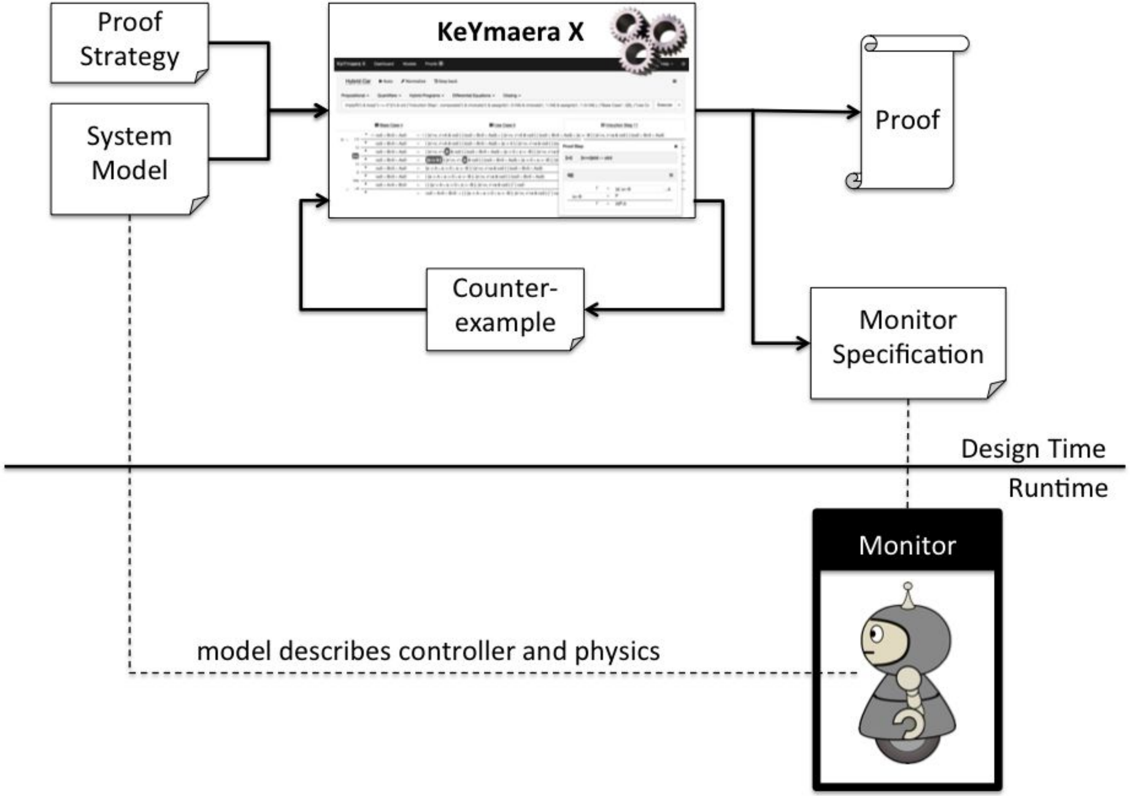
Solving Differential Equations

1. Use untrusted code to find a conjecture.
2. Prove the conjecture systematically.

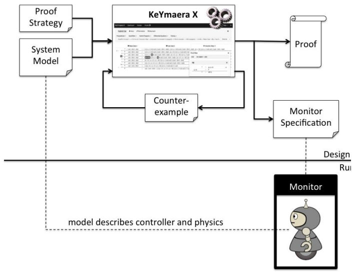


Toward Automated Deduction

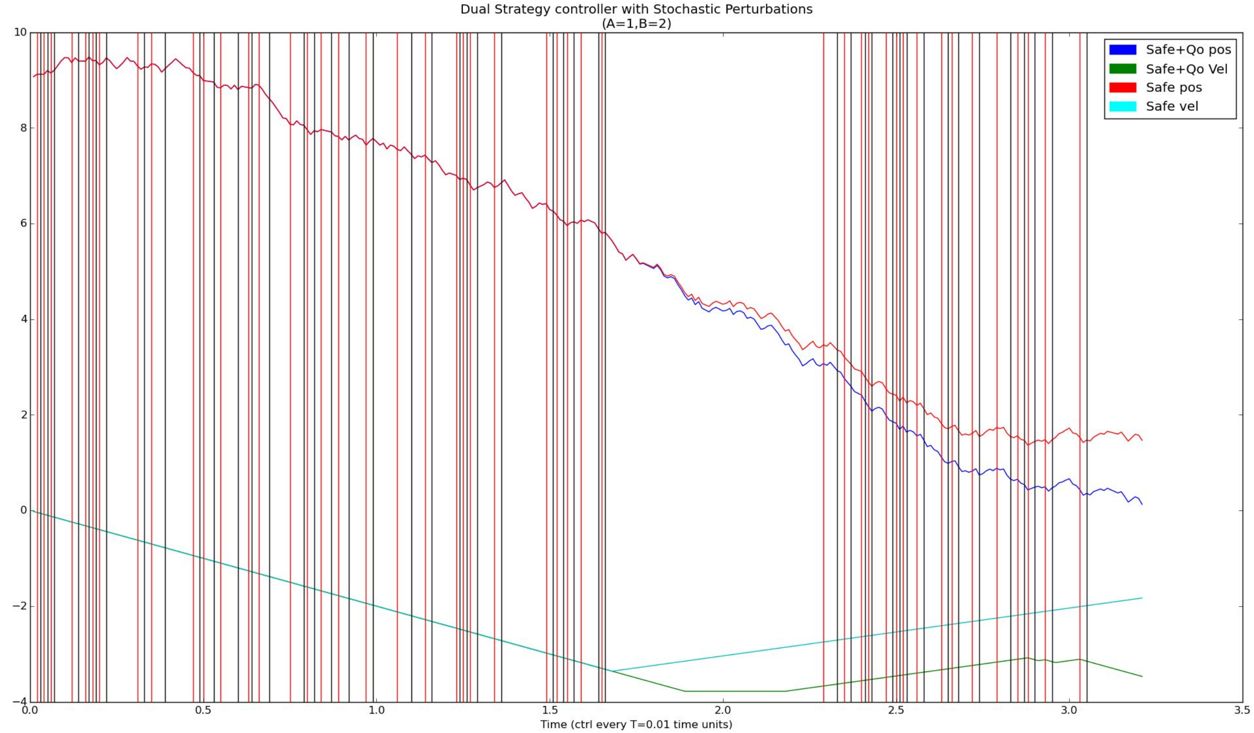
ModelPlex Tactic



Learning how to be Safe



=>



Other Proof Automation & Tooling

- Automated Analysis for nonlinear systems:
  - Pretty decent automation for systems with univariate nonlinearities.
  - Heuristics for multi-variate systems.

Other Proof Automation & Tooling

- Automated Analysis for nonlinear systems:
  - Pretty decent automation for systems with univariate nonlinearities.
  - Heuristics for multi-variate systems.
- Heuristic loop invariant generation for control loops

Other Proof Automation & Tooling

- Automated Analysis for nonlinear systems:
  - Pretty decent automation for systems with univariate nonlinearities.
  - Heuristics for multi-variate systems.
- Heuristic loop invariant generation for control loops
- Taylor Approximations
- ...

Other Proof Automation & Tooling

- Automated Analysis for nonlinear systems:
  - Pretty decent automation for systems with univariate nonlinearities.
  - Heuristics for multi-variate systems.
- Heuristic loop invariant generation for control loops
- Taylor Approximations
- ...
- **Component-based Verification Tooling**

  Mueller et al., *Change and Delay Contracts for Hybrid System Component Verification*, **FASE'17 -- Thursday 10:30-12:30**

Conclusion

KeYmaera X is a hybrid systems theorem prover with:

- A small and trustworthy prover core and
- Excellent infrastructure for **interactively verifying complex systems** and **implementing automated analyses**.

Conclusion

KeYmaera X is a hybrid systems theorem prover with:

- A small and trustworthy prover core and
- Excellent infrastructure for **interactively verifying complex systems** and **implementing automated analyses**.

Project Website (start here)	keymaeraX.org
Online Demo	web.keymaeraX.org
GPL'd Source Code	github.com/ls-lab/KeYmaeraX-release
Course Materials	symbolaris.com/course/fcps17.html

Developers:

- Stefan Mitsch
- Nathan Fulton
- Andre Platzer
- Jan-David Quesel
- Brandon Bohrer
- Yong Kiam Tan
- Markus Voelp

Special Thanks:

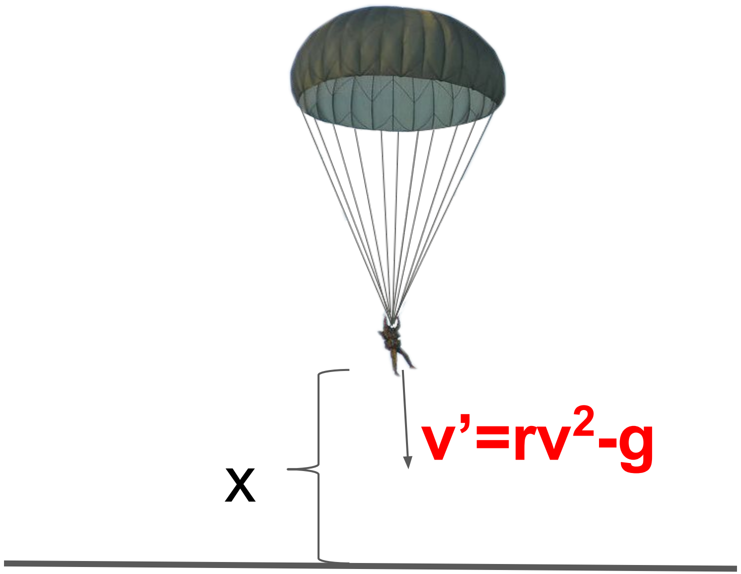
- 15-424 students, Jean-Baptiste Jeanin, Khalil Ghorbal, Daniel Ricketts

Differential Ghosts

Parachute Closed:

$$J \;\&\; t=0 \;\&\; r=r_p \rightarrow [x'=v, v'=rv^2-g \;\&\; 0\leq x \;\&\; t\leq T]\, v>-\mathrm{sqrt}(g/pr) > m$$

Proof requires a **differential ghost** because the property is **not inductive**.



Differential Ghosts

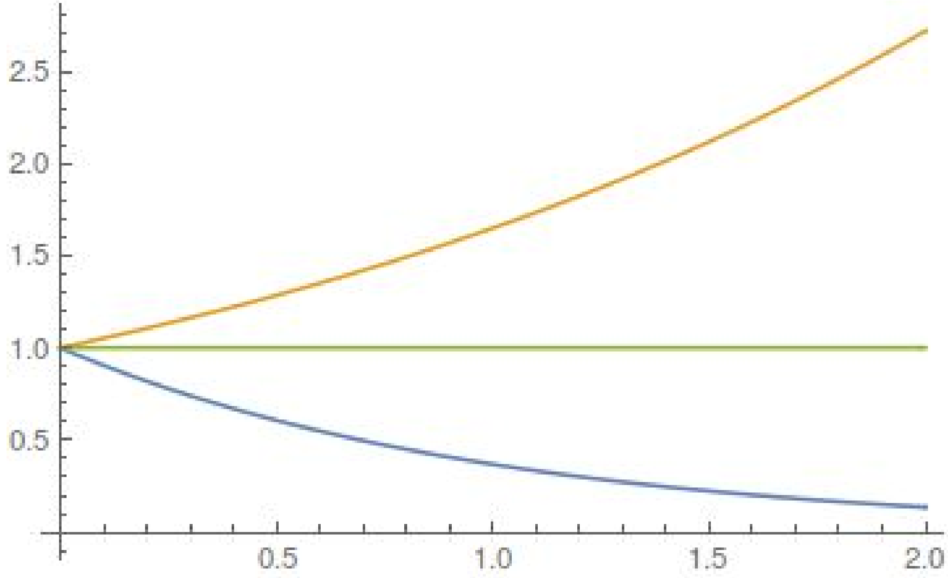
An example differential ghost.

```
x>0 → [x'=-x]x>0
```

Differential Ghosts

An example differential ghost.

```
x>0 → [x'=-x]x>0
Ghost:      y'=y/2
Conserved: 1=xy²
```



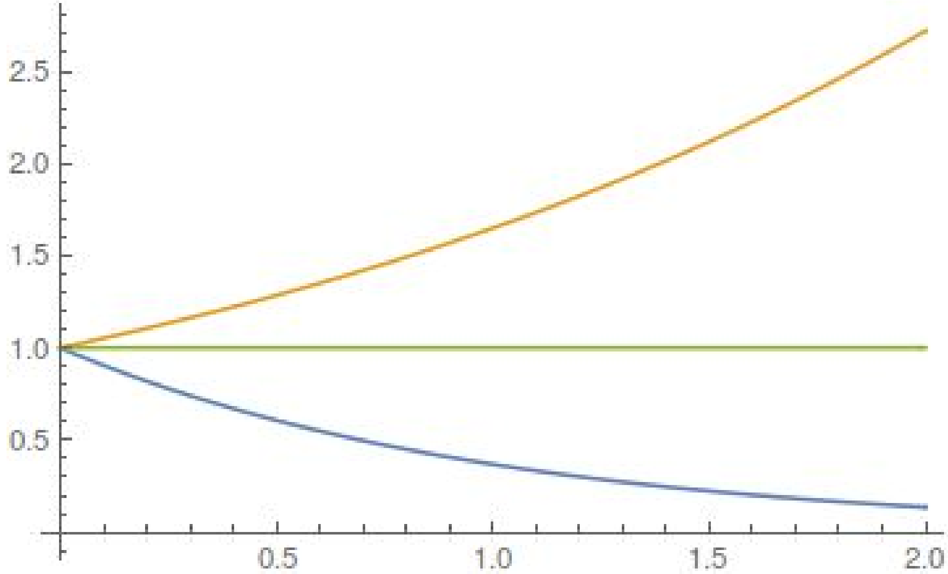
Differential Ghosts

An example differential ghost.

$x>0 \rightarrow [x'=-x]x>0$

Ghost: $y'=y/2$

Conserved: $1=xy^2$



Notice:

$x>0 \leftrightarrow \exists y.1=xy^2$

Therefore, suffices to show:

$1=xy^2 \rightarrow \exists y.[x'=-x, y'=y/2]1=xy^2$