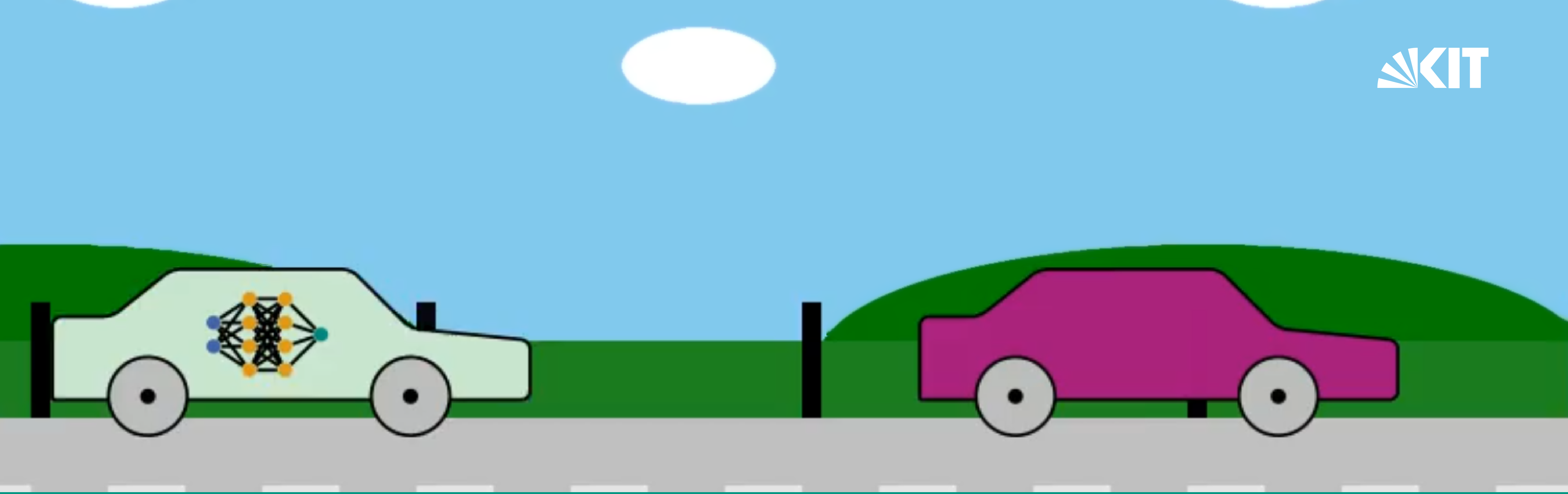


Verification of Autonomous Neural Car Control with KeYmaera X

ABZ 2025 Case Study Challenge

Enguerrand Prebet, **Samuel Teuber**, André Platzer | 12th of June 2025

The ABZ Case Study

Leuschel et al. 2025

Symbolic dL-model for highway car control
$\rightarrow$ infinite-time guarantee: absence of collision

What does that imply for concrete controllers?

The ABZ Case Study

Leuschel et al. 2025

Symbolic dL-model for highway car control
$\rightarrow$ infinite-time guarantee: absence of collision

What does that imply for concrete Neural Network controllers?

The ABZ Case Study

Leuschel et al. 2025

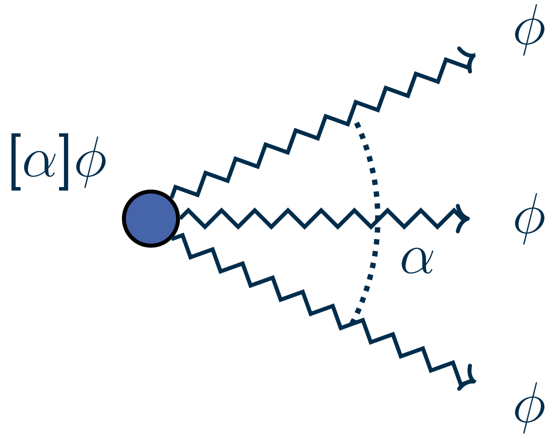
Symbolic dL-model for highway car control
$\rightarrow$ infinite-time guarantee: absence of collision

What does that imply for concrete Neural Network controllers?

Safe abstract dL-model $\longleftrightarrow$ Neural network controller $\longleftarrow$ black/whitebox?

Differential Dynamic Logic

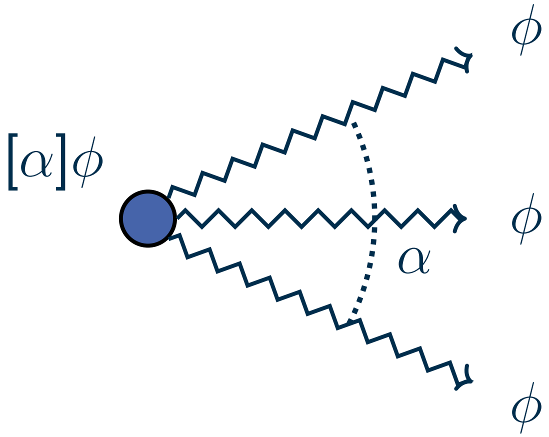
FOL($\mathbb{R}$) + program modalities



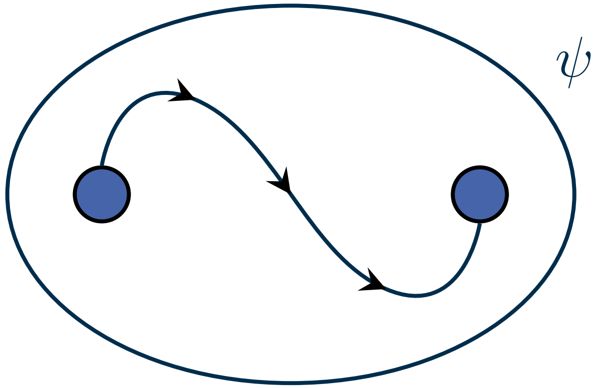
Hoare triple: init $\rightarrow$ [sys]post

Differential Dynamic Logic

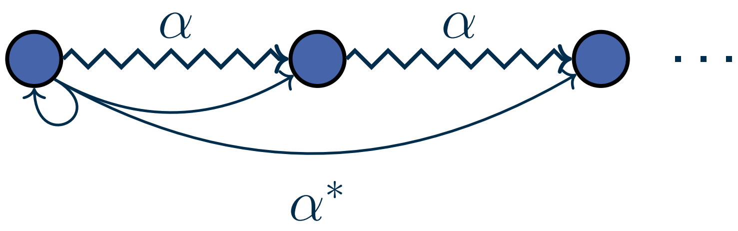
FOL($\mathbb{R}$) + program modalities + differential systems



Hoare triple: init $\rightarrow$ [sys]post



$x' = \theta \& \psi$



Proving properties

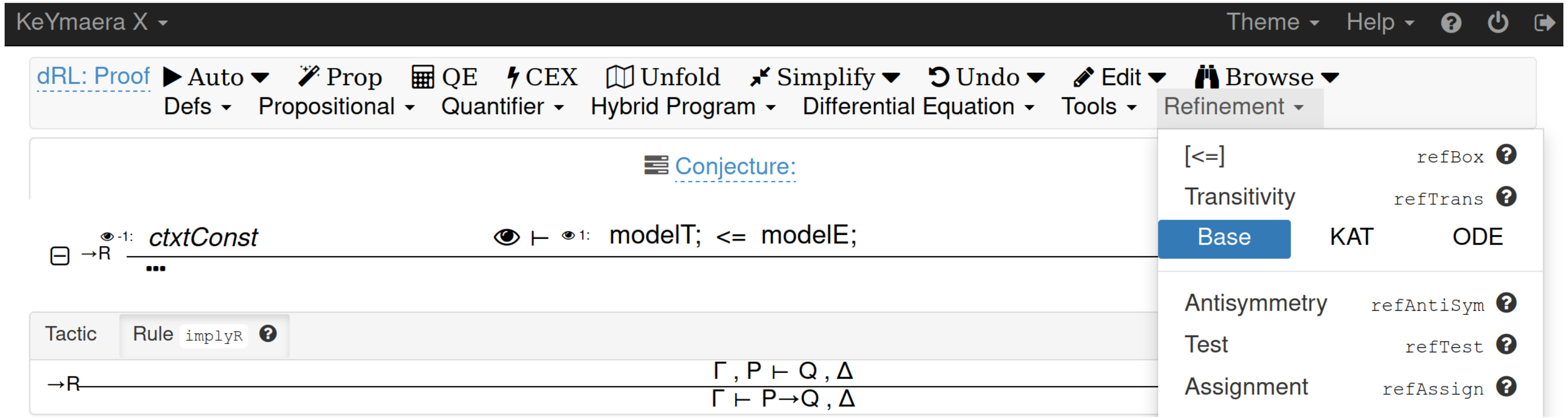
Uniform-substitution based calculus:

$$p() \rightarrow [a]p()$$
$$[x := f()]p(x) \leftrightarrow p(f())$$

$$\text{(US)}\ \frac{\phi}{\sigma(\phi)}\ \text{if } \sigma(\phi) \text{ defined}$$

Refinements as formulas: $\alpha \leq \beta$

All implemented in theorem prover KeYmaera X



Our dL Model

Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \underbrace{\texttt{ctrl}_o; (\texttt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}};$$

- $\texttt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$

Our dL Model

Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \underbrace{\texttt{ctrl}_o; (\texttt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}};$$

- $\texttt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$
- $\texttt{ctrl}_e$: a_e , if $\neg\texttt{safe}(a_e)$, overrides with one in $[-B_{\max}, -B_{\min}]$ behind
  RSS-like $[A_{\min}, A_{\max}]$ in front

Our dL Model

Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T);}_{\text{control}} \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}$$

- $\mathtt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$
- $\mathtt{ctrl}_e$: a_e , if $\neg\mathtt{safe}(a_e)$, overrides with one in $[-B_{\max}, -B_{\min}]$ behind
- $\mathtt{accelCorr}$: ensures $0 \leq v_e, v_o \leq V$ RSS-like $[A_{\min}, A_{\max}]$ in front

Our dL Model

Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T);}_{\text{control}} \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}$$

- $\mathtt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$
- $\mathtt{ctrl}_e$: a_e , if $\neg\mathtt{safe}(a_e)$, overrides with one in $[-B_{\max}, -B_{\min}]$ behind
- $\mathtt{accelCorr}$: ensures $0 \leq v_e, v_o \leq V$ RSS-like $[A_{\min}, A_{\max}]$ in front
- dyn: cars move

$$\begin{aligned} &x_e' = v_e, v_e' = a_e, \\ &x_o' = v_o, v_o' = a_o, \ \& \ t \leq t_e + T \\ &t' = 1 \end{aligned}$$

Our dL Model

Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \big(\underbrace{\texttt{ctrl}_o; (\texttt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\texttt{accelCorr}; \texttt{dyn}}_{\text{plant}}\big)^*$$

- $\texttt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$
- $\texttt{ctrl}_e$: a_e , if $\neg\texttt{safe}(a_e)$, overrides with one in $[-B_{\max}, -B_{\min}]$ behind
- $\texttt{accelCorr}$: ensures $0 \leq v_e, v_o \leq V$ RSS-like $[A_{\min}, A_{\max}]$ in front
- dyn: cars move

$$\begin{aligned} &x_e' = v_e, v_e' = a_e, \\ &x_o' = v_o, v_o' = a_o, \ \ \& \ t \leq t_e + T \\ &t' = 1 \end{aligned}$$

Our dL Model

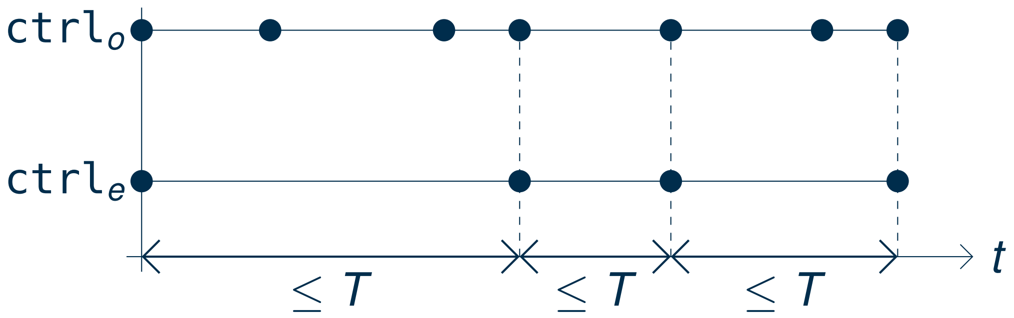
Two unordered cars $\rightarrow$ core question, even for multilane

$$\text{sys} ::= \big(\underbrace{\texttt{ctrl}_o; (\texttt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\texttt{accelCorr}; \texttt{dyn}}_{\text{plant}}\big)^*$$

- $\texttt{ctrl}_o$: sets a_o to a value in $[-B_{\max}, A_{\max}]$
- $\texttt{ctrl}_e$: a_e , if $\neg\texttt{safe}(a_e)$, overrides with one in $[-B_{\max}, -B_{\min}]$ behind
- $\texttt{accelCorr}$: ensures $0 \le v_e, v_o \le V$ RSS-like $[A_{\min}, A_{\max}]$ in front
- $\texttt{dyn}$: cars move

$$\begin{aligned} &x_e' = v_e, v_e' = a_e, \\ &x_o' = v_o, v_o' = a_o, \ \ \& \ t \le t_e + T \\ &t' = 1 \end{aligned}$$

Desynchronised controllers:



Safety proofs

Theorem

These formulas are proved in dL:

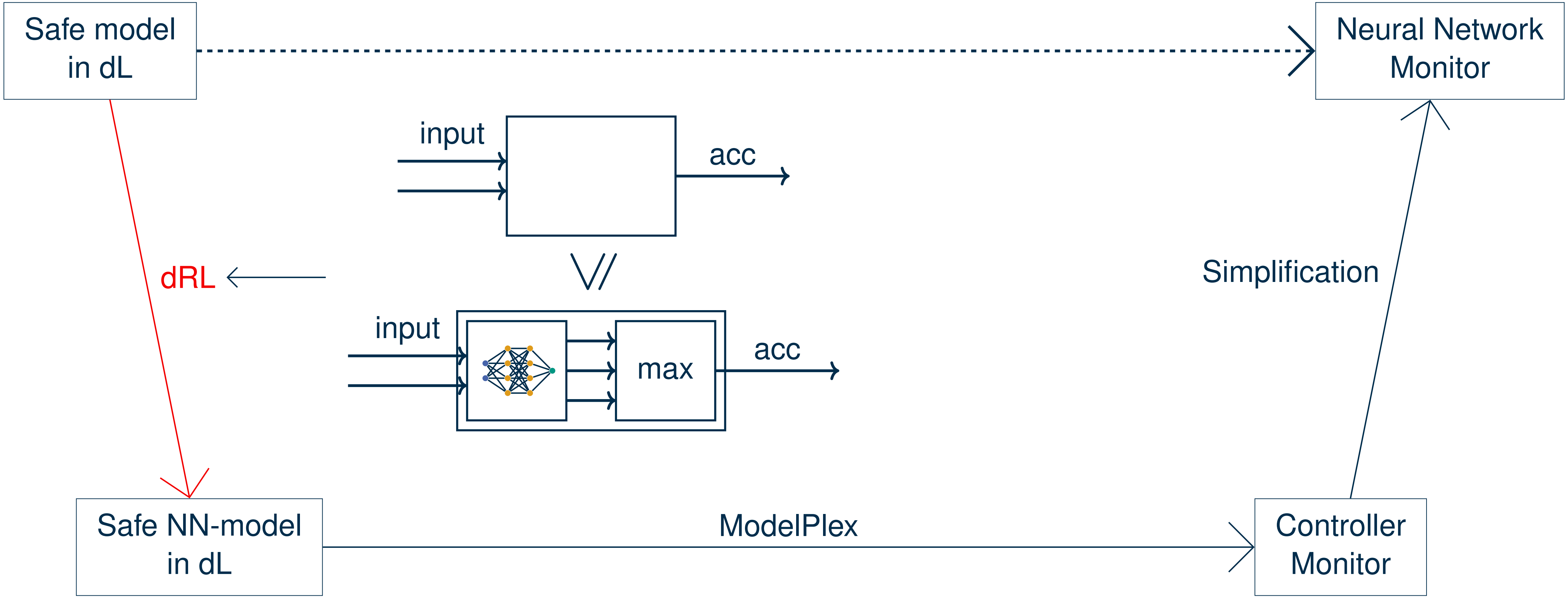
$$\texttt{ctx} \land x_e + L \leq x_o \land \text{init} \rightarrow [\text{sys}]x_e + L \leq x_o \qquad \texttt{ctx} \land x_o + L \leq x_e \land \widetilde{\text{init}} \rightarrow [\text{sys}]x_o + L \leq x_e$$

$$\text{init} ::= x_e + \frac{v_e^2}{2B_{\text{min}}} + L \leq x_o + \frac{v_o^2}{2B_{\text{max}}} \qquad \widetilde{\text{init}} ::= x_e + \frac{(v_e - V)^2}{2(-A_{\text{min}})} + L \leq x_o + \frac{(v_o - V)^2}{2(-A_{\text{max}})}$$

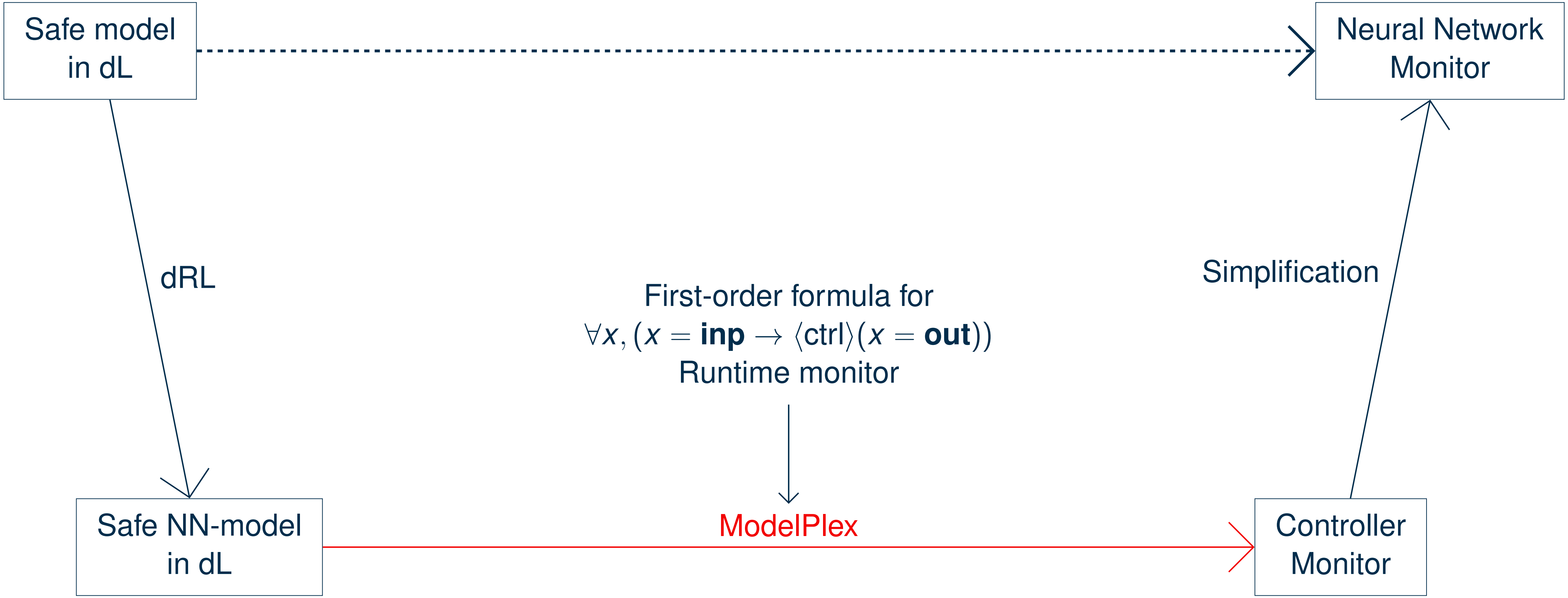
KeYmaera X proofs and experiments online: `https://doi.org/10.5281/zenodo.14959858`

ModelPlex for Neural Network Controllers

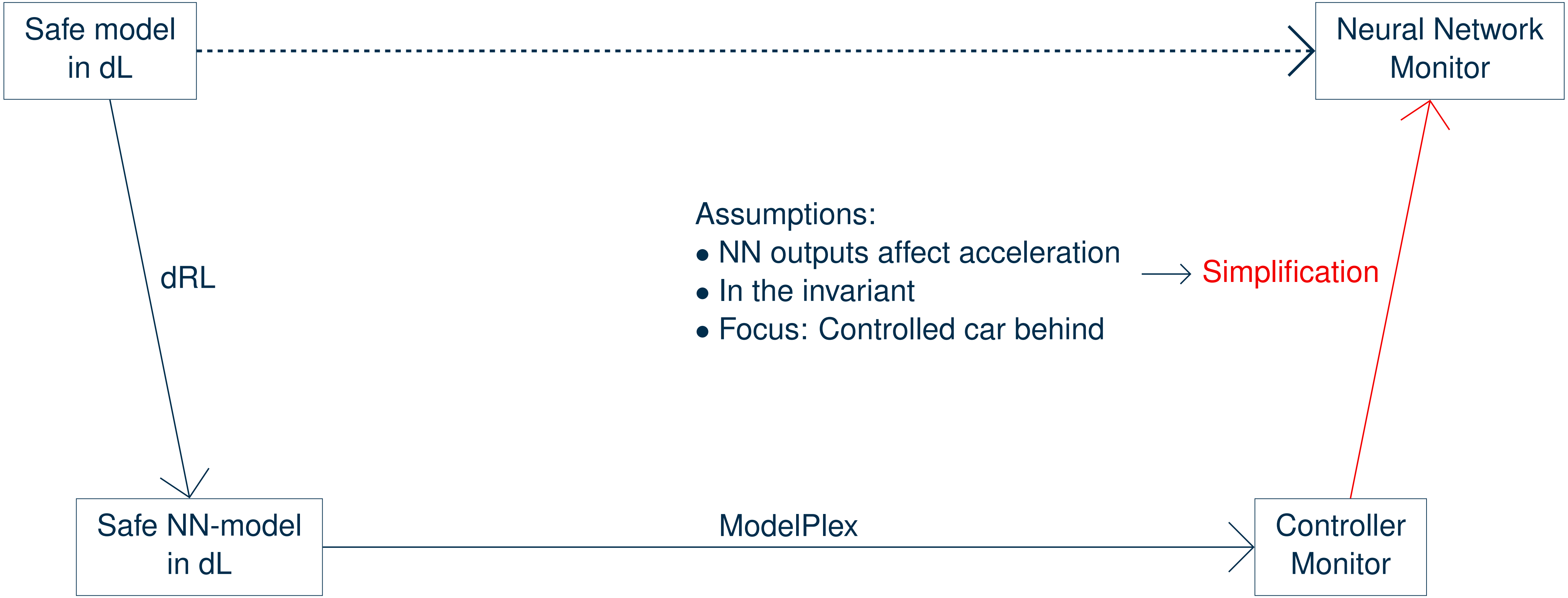
ModelPlex for Neural Network Controllers



ModelPlex for Neural Network Controllers



ModelPlex for Neural Network Controllers



Applications of ModelPlex

Via dL: Correct by Construction Monitoring Condition

$$
\begin{aligned}
& y_1^+ \geq y_2^+ \wedge y_1^+ \geq y_3^+ \\
& \quad \vee\, y_2^+ > y_1^+ \wedge y_2^+ \geq y_3^+ \wedge \\
& \qquad \left(B_{\min} \leq 0 \leq A_{\max} \wedge v_e \geq 0 \wedge \mathrm{pos}_e(B_{\min}) + (\frac{0}{B_{\min}} + 1) T v_e + L < \mathrm{pos}_o\right) \\
& \quad \vee\, y_3^+ > y_1^+ \wedge y_3^+ > y_2^+ \wedge \big(B_{\min} \leq A_{\max} \wedge v_e + A_{\max} T < 0 \wedge \mathrm{pos}_e(A_{\max}) + L < \mathrm{pos}_o \\
& \qquad \vee\, B_{\min} \leq A_{\max} \wedge v_e + A_{\max} T \geq 0 \wedge \mathrm{pos}_e(B_{\min}) + (\frac{-A_{\max}}{B_{\min}} + 1)(\frac{A_{\max}}{2} T^2 + T v_e) + L < \mathrm{pos}_o\big)
\end{aligned}
$$

Given concrete inputs and outputs, this form tells us what actions are provably safe.
But how do we put this knowledge into practice?

Applications of ModelPlex

Via dL: Correct by Construction Monitoring Condition

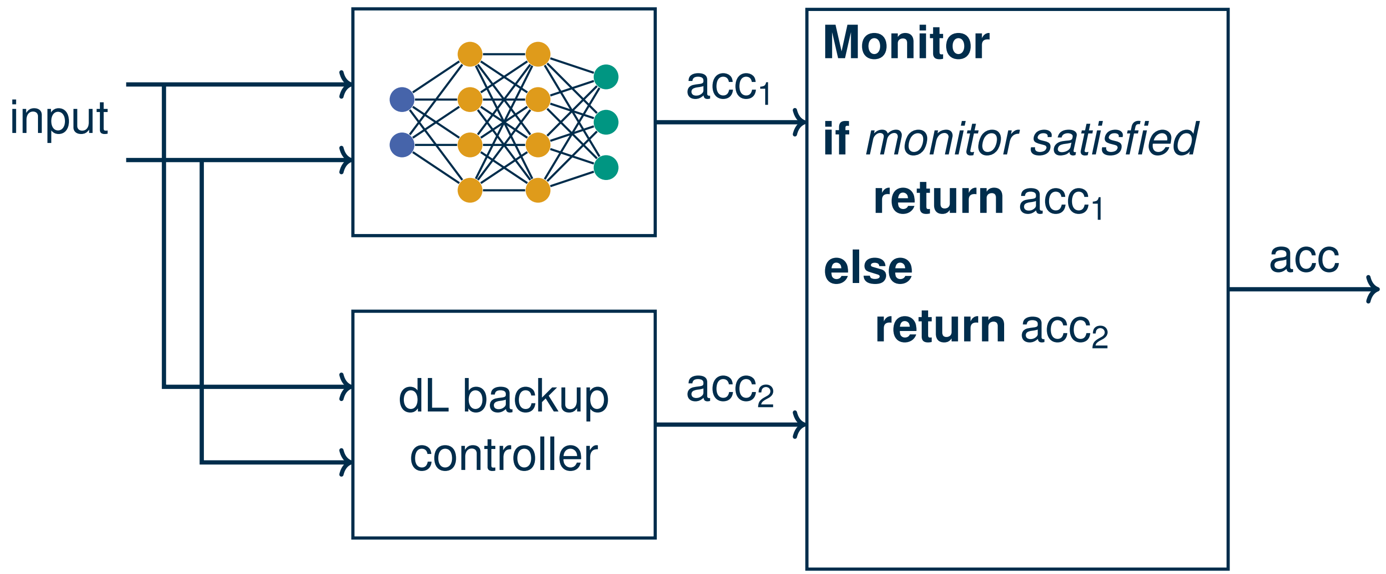
$$
\begin{aligned}
& y_1^+ \geq y_2^+ \wedge y_1^+ \geq y_3^+ \\
& \quad \vee\, y_2^+ > y_1^+ \wedge y_2^+ \geq y_3^+ \wedge \\
& \qquad \left(B_{\min} \leq 0 \leq A_{\max} \wedge v_e \geq 0 \wedge \mathrm{pos}_e(B_{\min}) + (\frac{0}{B_{\min}} + 1)\, T v_e + L < \mathrm{pos}_o\right) \\
& \quad \vee\, y_3^+ > y_1^+ \wedge y_3^+ > y_2^+ \wedge \big(B_{\min} \leq A_{\max} \wedge v_e + A_{\max} T < 0 \wedge \mathrm{pos}_e(A_{\max}) + L < \mathrm{pos}_o \\
& \qquad \vee\, B_{\min} \leq A_{\max} \wedge v_e + A_{\max} T \geq 0 \wedge \mathrm{pos}_e(B_{\min}) + (\frac{-A_{\max}}{B_{\min}} + 1)(\frac{A_{\max}}{2} T^2 + T v_e) + L < \mathrm{pos}_o\big)
\end{aligned}
$$

Given concrete inputs and outputs, this form tells us what actions are provably safe.
But how do we put this knowledge into practice?

Monitoring | Shielding | Verification

Monitoring / Sandboxing (VeriPhy)

Check NN actions during runtime at each step

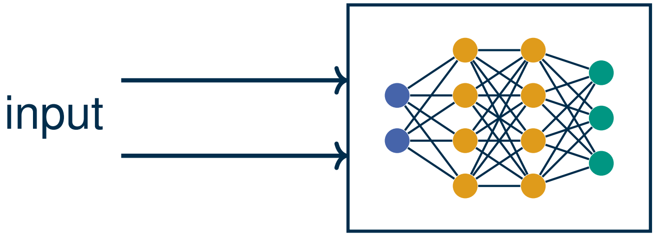


Can be combined with correct-by-construction sandbox synthesis (Bohrer et al. 2018)

Mitsch and Platzer 2016

Shielding (Justified Speculative Control)

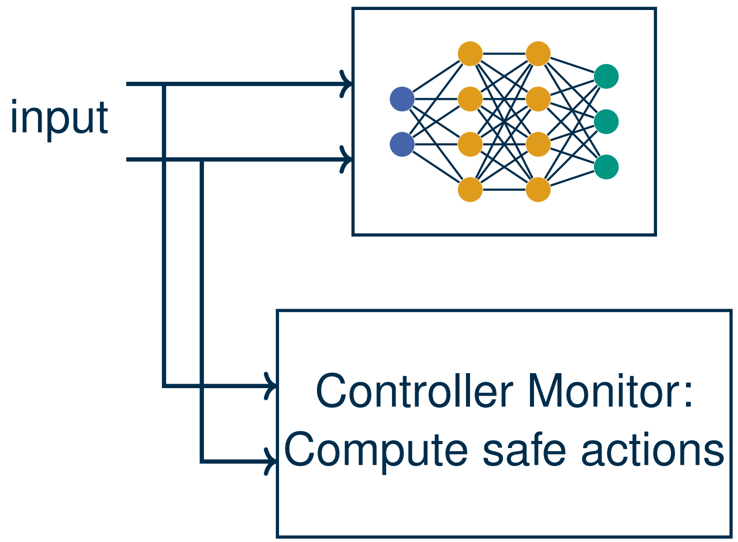
Insight: RL Agents often learn a **distribution** of actions
$\Rightarrow$ Constrain action space



Fulton and Platzer 2018

Shielding (Justified Speculative Control)

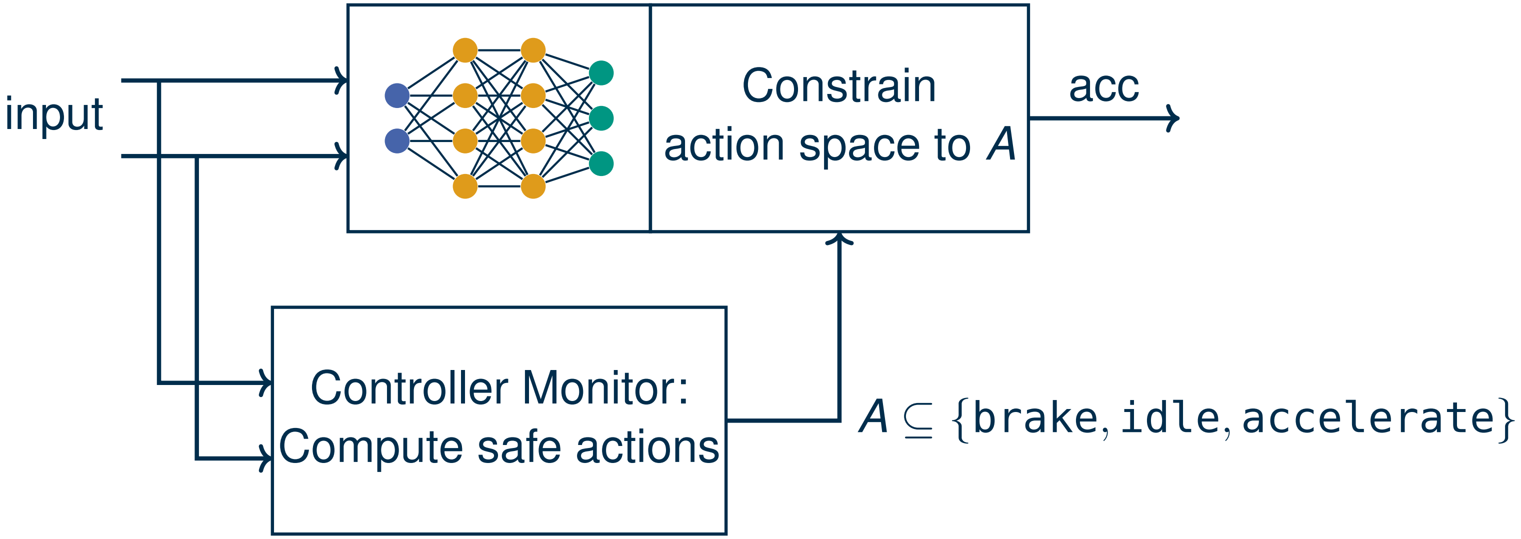
Insight: RL Agents often learn a **distribution** of actions
$\Rightarrow$ Constrain action space



Fulton and Platzer 2018

Shielding (Justified Speculative Control)

Insight: RL Agents often learn a **distribution** of actions
$\Rightarrow$ Constrain action space

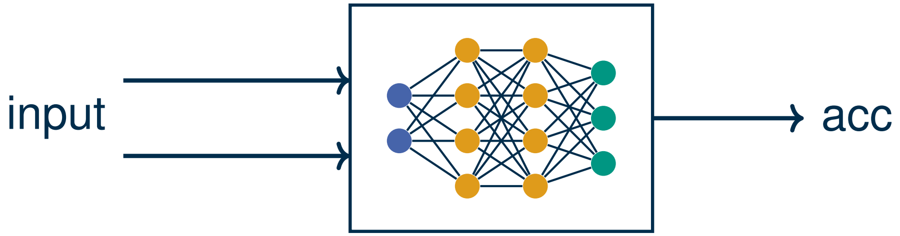


Provably safe actions during **training & deployment**!
Can also take into account **model monitoring**

Fulton and Platzer 2018

Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller



Teuber, Mitsch, and Platzer 2024

Verification (VerSAILLE)

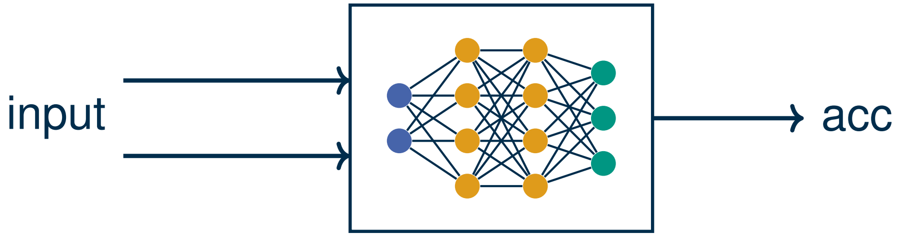
Objective: A priori guarantees on safety of NN controller

Neural Network Verification:

$$\forall \bar{x} \quad \phi(\bar{x}, g(\bar{x}))$$

Before Deployment

At Runtime



Teuber, Mitsch, and Platzer 2024

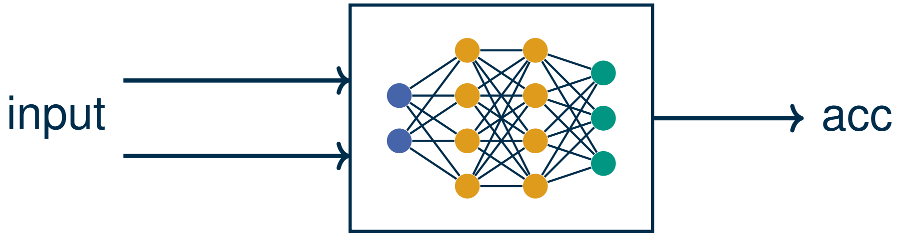
Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

$\left(\alpha_{\text{ctrl}} \; ; \alpha_{\text{plant}}\right)^*$ Safe

Before Deployment

At Runtime



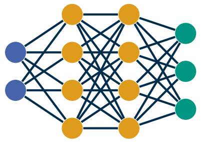
Teuber, Mitsch, and Platzer 2024

Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

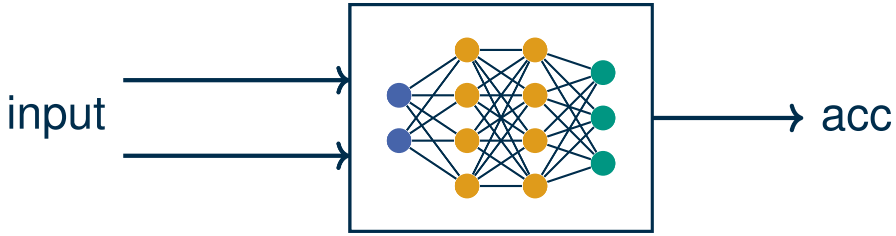
$\left(\alpha_{\text{ctrl}} \;;\, \alpha_{\text{plant}}\right)^{*}$ Safe

g



Before Deployment

At Runtime



Teuber, Mitsch, and Platzer 2024

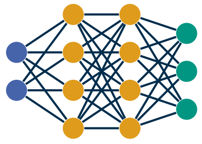
Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

$\left(\alpha_{\text{ctrl}}\ ;\alpha_{\text{plant}}\right)^*$ Safe

Before Deployment

g

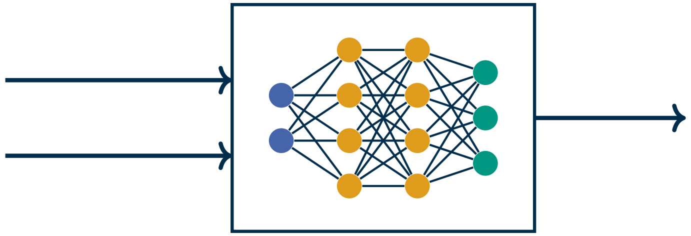


Nondeterministic Mirror:
$\alpha_g := \mathrm{mirror}(g)$

$\left(\underbrace{\alpha_g}\ ;\alpha_{\text{plant}}\right)^*$

At Runtime

input



acc

Teuber, Mitsch, and Platzer 2024

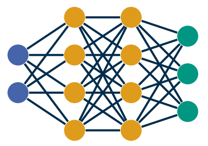
Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

$$\text{invariant} \wedge \neg\text{monitor} \xleftarrow[\text{based on ModelPlex \& Loop Invariants}]{\text{VerSAILLE}} \left(\alpha_{\text{ctrl}}\ ;\alpha_{\text{plant}}\right)^* \quad \text{Safe}$$

Before Deployment

g



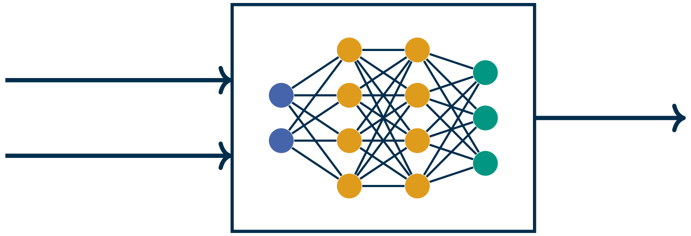
Nondeterministic Mirror:

$\alpha_g := \text{mirror}\,(g)$

$$\big(\underbrace{\alpha_g}\ ;\alpha_{\text{plant}}\big)^*$$

At Runtime

input



acc

Teuber, Mitsch, and Platzer 2024

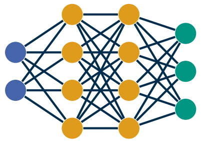
Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

$\text{invariant} \land \neg\text{monitor} \xleftarrow[\text{based on ModelPlex \& Loop Invariants}]{\text{VerSAILLE}} (\alpha_{\text{ctrl}} \; ; \alpha_{\text{plant}})^*$ Safe

NN Verification (NCubeV):
Show that unsat

g



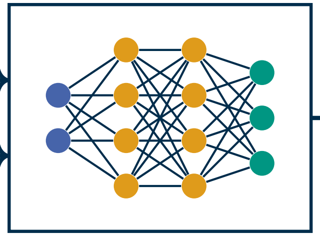
Nondeterministic Mirror:
$\alpha_g := \text{mirror}(g)$

$(\underbrace{\alpha_g}\; ; \alpha_{\text{plant}})^*$

Before Deployment

At Runtime

input



acc

Teuber, Mitsch, and Platzer 2024

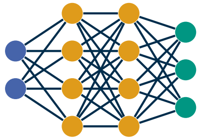
Verification (VerSAILLE)

Objective: A priori guarantees on safety of NN controller

invariant $\land$ $\neg$monitor $\xleftarrow[\text{based on ModelPlex \& Loop Invariants}]{\text{VerSAILLE}}$ $(\alpha_{\text{ctrl}}\ ;\alpha_{\text{plant}})^*$ Safe

NN Verification (NCubeV):
Show that unsat

Refinement

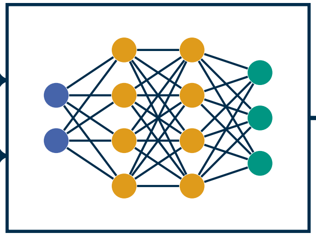
g 

Before Deployment

Nondeterministic Mirror:
$\alpha_g := \text{mirror}(g)$

$(\underbrace{\alpha_g}\ ;\alpha_{\text{plant}})^*$

At Runtime

input  acc

A priori and **infinite-time horizon** safety guarantees

Teuber, Mitsch, and Platzer 2024

Model-To-Simulation Gap (1)

Behaviour of FASTER

Spec:

“This action ***increases the speed (up to $\mathbf{v_{max}}$) with an acceleration*** *up to a_{max} m/s^2.*
Once the car reaches v_{max}, the acceleration is 0 m/s^2.”

Model-To-Simulation Gap (1)

Behaviour of `FASTER`

Spec:

“This action ***increases the speed (up to v_{max}) with an acceleration*** *up to a_{max} m/s^2.*
Once the car reaches v_{max}, the acceleration is 0 m/s^2.”

Simulator:

Uses the configuration `DiscreteMetaAction`: `FASTER` increases the **reference velocity** v_r.
Subsequently, a **low-level** proportional controller adjusts the acceleration.

Model-To-Simulation Gap (1)

Behaviour of FASTER

Spec:
*“This action **increases the speed (up to v_{max}) with an acceleration** up to a_{max} m/s^2.*
Once the car reaches v_{max}, the acceleration is 0 m/s^2.”

Simulator:
Uses the configuration `DiscreteMetaAction`: FASTER increases the **reference velocity** v_r.
Subsequently, a **low-level** proportional controller adjusts the acceleration.

$\Rightarrow$ **FASTER can lead to braking if $v_r < v$!**

Model-To-Simulation Gap (1)

Behaviour of FASTER

Spec:
*“This action **increases the speed (up to v_{max}) with an acceleration** up to a_{max} m/s^2.
Once the car reaches v_{max}, the acceleration is 0 m/s^2.”*

Simulator:
Uses the configuration `DiscreteMetaAction`: `FASTER` increases the **reference velocity** v_r.
Subsequently, a **low-level** proportional controller adjusts the acceleration.

$\Rightarrow$ **`FASTER` can lead to braking if $v_r < v$!**

We adjusted the simulator’s configuration and retrained a new set of NNs using the provided scripts.

Experimental Results (1)

A first concrete NN

Performance: Standalone NN / Monitoring / Shielding

Original NN		Monitoring (VeriPhy)		Shielding (JSC)	
Reward	Crash	Reward	Crash	Reward	Crash
17.63 $\pm$ 0.21	**0** %	16.72 $\pm$ 0.32	**0** %	**17.63** $\pm$ 0.21	**0** %

This looks good – let's verify it! (1000 simulations)

Experimental Results (1)

A first concrete NN

Performance: Standalone NN / Monitoring / Shielding

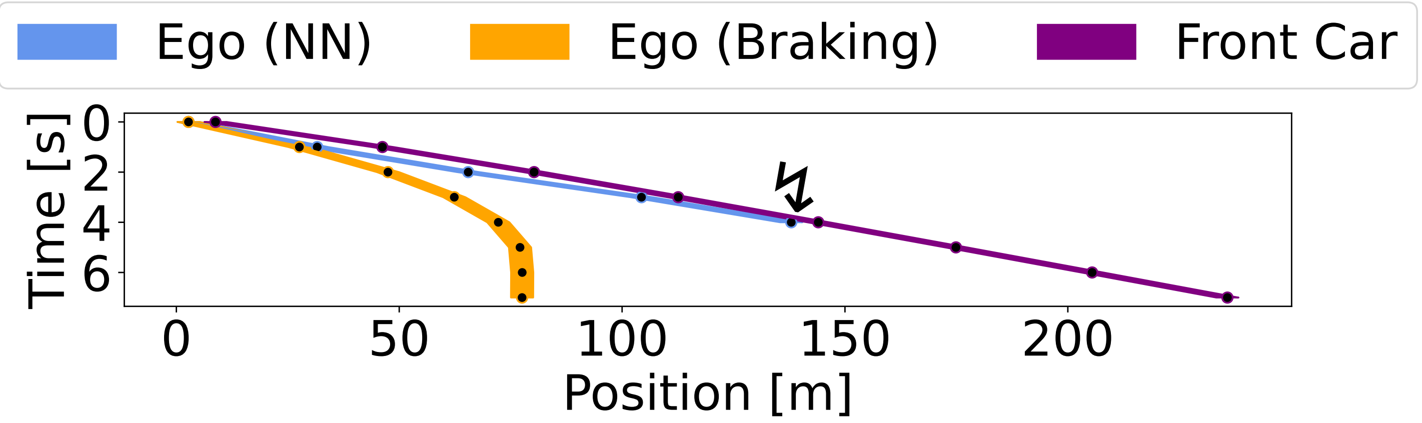
Original NN		Monitoring (VeriPhy)		Shielding (JSC)	
Reward	Crash	Reward	Crash	Reward	Crash
17.63 $\pm$ 0.21	**0** %	16.72 $\pm$ 0.32	**0** %	**17.63** $\pm$ 0.21	**0** %

This looks good – let's verify it! (1000 simulations)

For starters: 2 cars

- Verifier (NCubeV): 3.6 hours
  NN size: 2x256 ReLU nodes
- **14,917 counterexample regions** (exhaustive!)
- Sampling trajectories: 538 concrete crashes

What went wrong?



Model-To-Simulation Gap (2)

Behaviour of other cars

Spec:
*“Maximum **braking** acceleration of **front** vehicle: β_{max}”*

Model-To-Simulation Gap (2)

Behaviour of other cars

Spec:
"*Maximum **braking** acceleration of **front** vehicle:* β_{max}"

Simulator (`highway-env`):
Other cars are controlled by the *Intelligent Driver Model*
Originally used for **congestion modelling**; Cars **rarely/never brake!**

Model-To-Simulation Gap (2)

Behaviour of other cars

Spec:
*“Maximum **braking** acceleration of **front** vehicle: β_{max}”*

Simulator (`highway-env`):
Other cars are controlled by the *Intelligent Driver Model*
Originally used for **congestion modelling**; Cars **rarely/never brake!**

We adjusted the implementation of the other cars to increase likelihood of braking.

Experimental Results (2)

A first concrete NN

Performance: Standalone NN / Monitoring / Shielding

Env	Original NN		Monitoring (VeriPhy)		Shielding (JSC)	
	Reward	Crash	Reward	Crash	Reward	Crash
`default` (IDM)	**17.63** $\pm$ 0.21	**0** %	16.72 $\pm$ 0.32	**0** %	**17.63** $\pm$ 0.21	**0** %
`braking`	5.44 $\pm$ 1.27	99.6%	**16.47** $\pm$ 0.05	**0** %	**16.47** $\pm$ 0.05	**0** %

(1000 simulations)

Experimental Results (2)

A first concrete NN

Performance: Standalone NN / Monitoring / Shielding

Env	Original NN		Monitoring (VeriPhy)		Shielding (JSC)	
	Reward	Crash	Reward	Crash	Reward	Crash
`default` (IDM)	**17.63** $\pm$ 0.21	**0** %	16.72 $\pm$ 0.32	**0** %	**17.63** $\pm$ 0.21	**0** %
`braking`	5.44 $\pm$ 1.27	99.6%	**16.47** $\pm$ 0.05	**0** %	**16.47** $\pm$ 0.05	**0** %

Can we train a better NN?

(1000 simulations)

Experimental Results (2)

A first concrete NN

Performance: Standalone NN / Monitoring / Shielding

Env	Original NN		Monitoring (VeriPhy)		Shielding (JSC)	
	Reward	Crash	Reward	Crash	Reward	Crash
`default` (IDM)	**17.63** $\pm$ 0.21	**0** %	16.72 $\pm$ 0.32	**0** %	**17.63** $\pm$ 0.21	**0** %
`braking`	5.44 $\pm$ 1.27	99.6%	**16.47** $\pm$ 0.05	**0** %	**16.47** $\pm$ 0.05	**0** %

Can we train a better NN?

Modifications:

- 80% of initial states: **within controllable region**
- Front Car: Initiates emergency brake with 15% likelihood
- Smaller NN for better verifiability (2 layers with 16 neurons)

Performance for `braking`: 16.08 ± 0.07 reward / 0 crashes (1000 simulations)

Experimental Results (3)

A better NN?

Verification w.r.t. **full specification** for front scenario:

- 2-5 cars in the front
- Assume $B_{\text{min}} = B_{\text{max}}$

Experimental Results (3)

A better NN?

Verification w.r.t. **full specification** for front scenario:

- 2-5 cars in the front
- Assume $B_{\text{min}} = B_{\text{max}}$

Verification:

- 1.9 hours
- 11,059 counterexample regions
- `default`: 4852 crashes
- `braking`: 8713 crashes

Would braking have saved the car?

Experimental Results (3)

A better NN?

Verification w.r.t. **full specification** for front scenario:

- 2-5 cars in the front
- Assume $B_{\text{min}} = B_{\text{max}}$

Verification:

- 1.9 hours
- 11,059 counterexample regions
- `default`: 4852 crashes
- `braking`: 8713 crashes

Would braking have saved the car?

- `default`: still 181 crashes
- `braking`: still 40 crashes

Experimental Results (3)

A better NN?

Verification w.r.t. **full specification** for front scenario:

- 2-5 cars in the front
- Assume $B_{\text{min}} = B_{\text{max}}$

Verification:

- 1.9 hours
- 11,059 counterexample regions
- `default`: 4852 crashes
- `braking`: 8713 crashes

Would braking have saved the car?

- `default`: still 181 crashes
- `braking`: still 40 crashes

?!?!

Model-To-Simulation Gap (3)

Environment Model

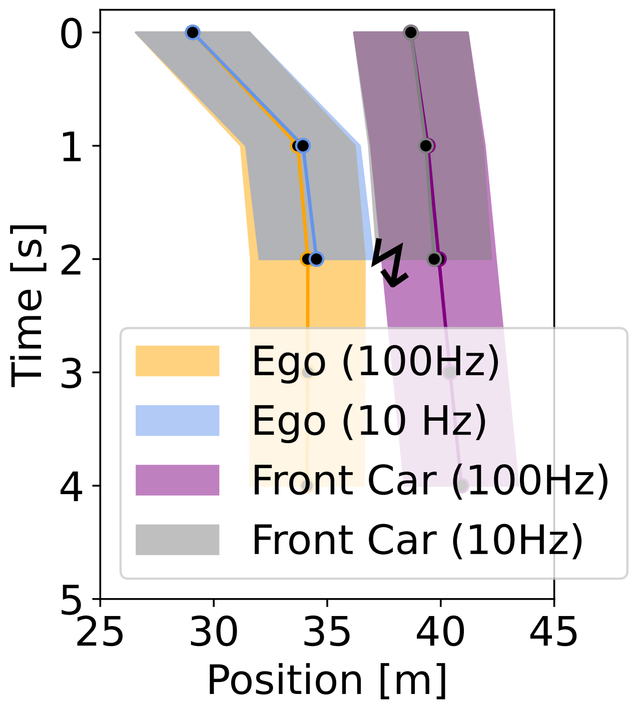
Spec: Continuous evolution of environment

Model-To-Simulation Gap (3)

Environment Model

Spec: Continuous evolution of environment

Simulator (`highway-env`): **Euler Approximations**
⇒ **Euler Crashes:**
Occurrence of crash dependent on precision of approximation



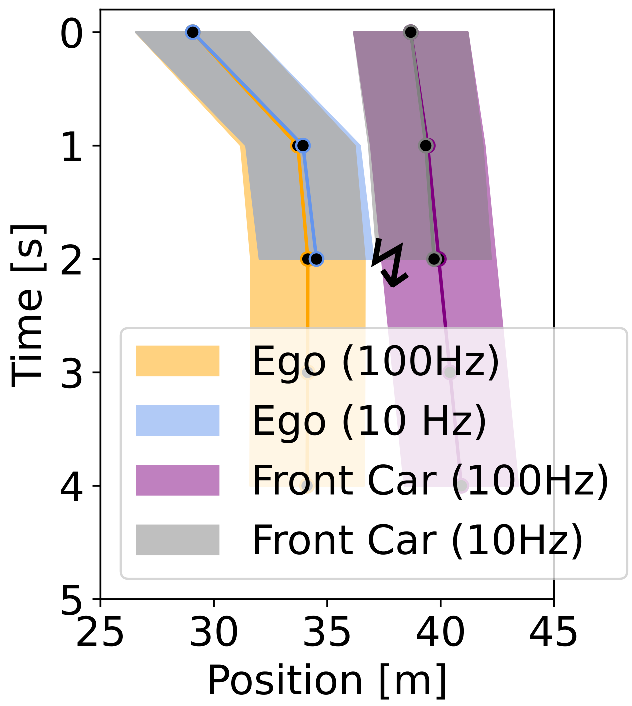
Model-To-Simulation Gap (3)

Environment Model

Spec: Continuous evolution of environment

Simulator (`highway-env`): **Euler Approximations**
$\Rightarrow$ **Euler Crashes:**
Occurrence of crash dependent on precision of approximation

Additionally: *Simulator seems to initialize environment on* ***small*** *subset of admissible states.*



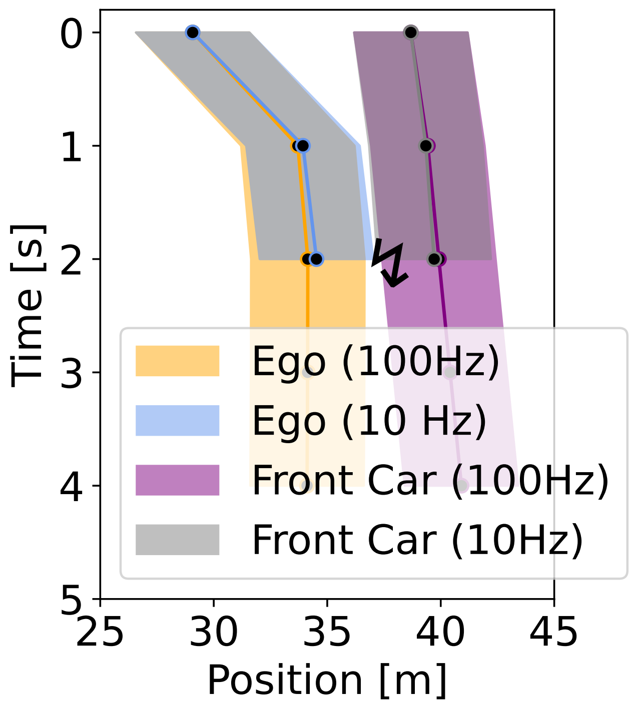
Model-To-Simulation Gap (3)

Environment Model

Spec: Continuous evolution of environment

Simulator (`highway-env`): **Euler Approximations**
$\Rightarrow$ **Euler Crashes:**
Occurrence of crash dependent on precision of approximation

Additionally: *Simulator seems to initialize environment on* ***small*** *subset of admissible states.*



The Model-to-Simulation Gap

- Unifying assumptions across formal models & simulations is **challenging**
- Safe control requires simulators showing full breadth of possible behaviour
- As is, `highway-env` is no reliable basis for training safe car control NNs.

This is a problem beyond this concrete case study!

Conclusion

Contributions

- General **dL model** for highway car control

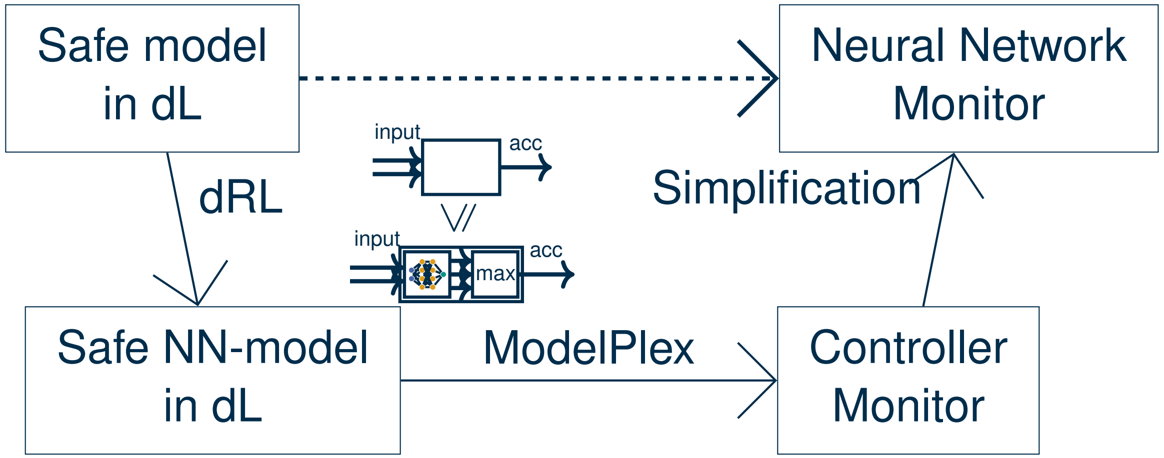
$$\mathsf{sys} ::= \Big(\underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}\Big)^*$$

Conclusion

Contributions

- General **dL model** for highway car control
- Derivation of **real arithmetic constraints** for monitoring/shielding/verification

$$\mathsf{sys} ::= \big(\underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}\big)^*$$

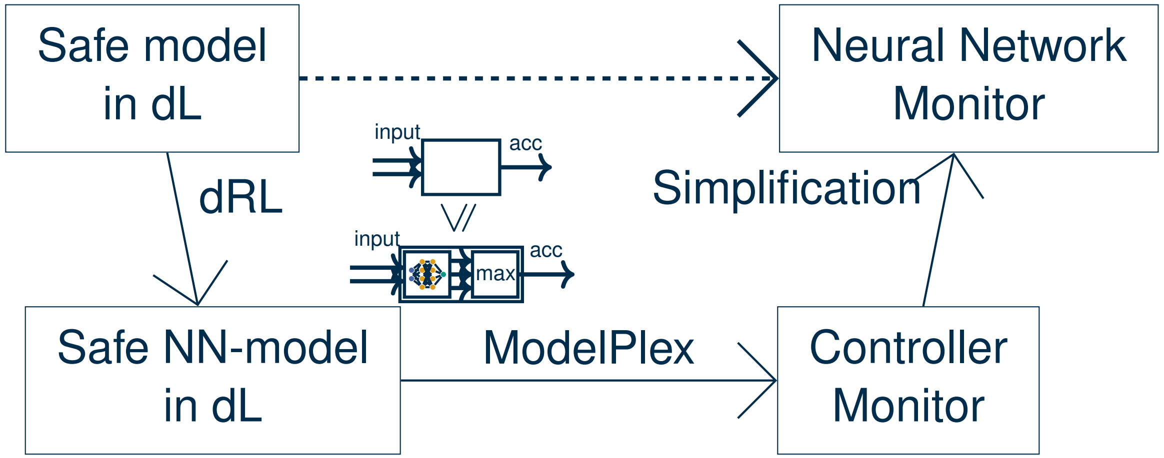


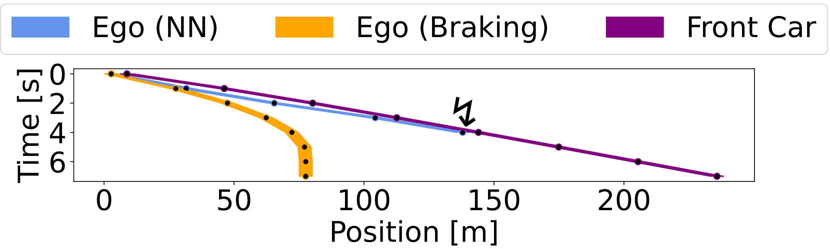
Conclusion

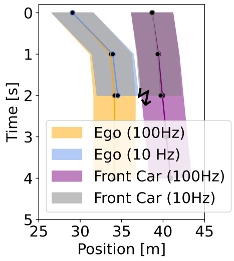
Contributions

- General **dL model** for highway car control
- Derivation of **real arithmetic constraints** for monitoring/shielding/verification
- An ***empirical validation*** of all three dL-based safeguarding techniques

$$\mathsf{sys} ::= \big(\underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}\big)^*$$







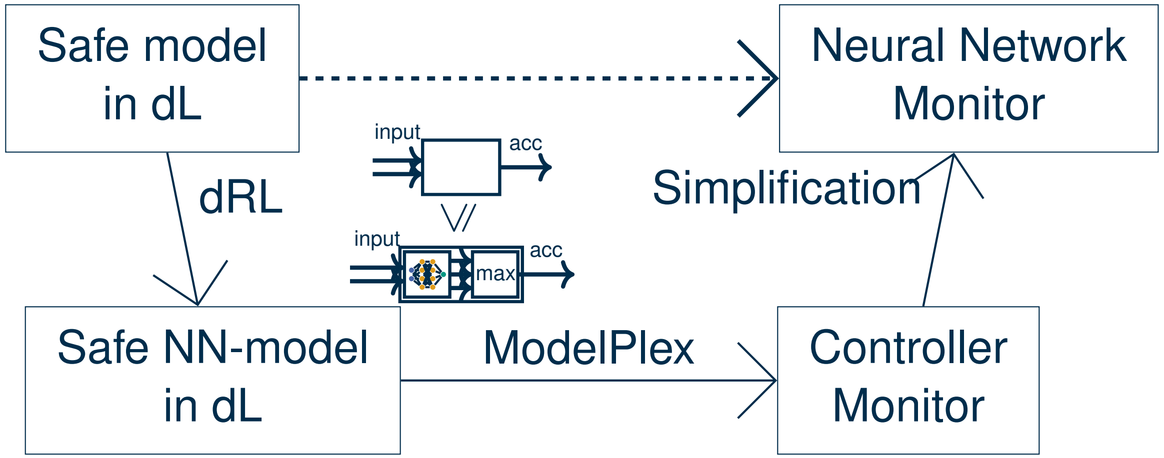
Conclusion

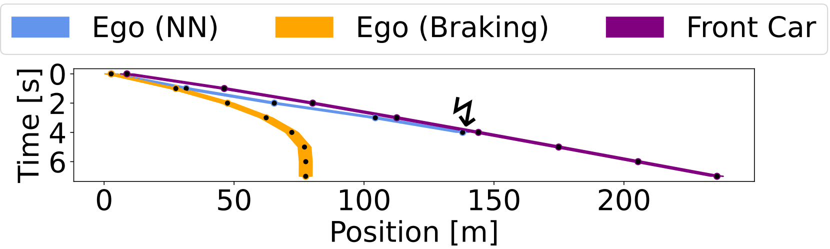
Contributions

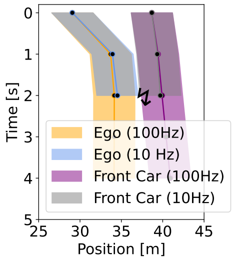
- General **dL model** for highway car control
- Derivation of **real arithmetic constraints** for monitoring/shielding/verification
- An ***empirical validation*** of all three dL-based safeguarding techniques

All presented techniques are **general**!

$$\text{sys} ::= \Big(\underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}\Big)^*$$







Conclusion

Contributions

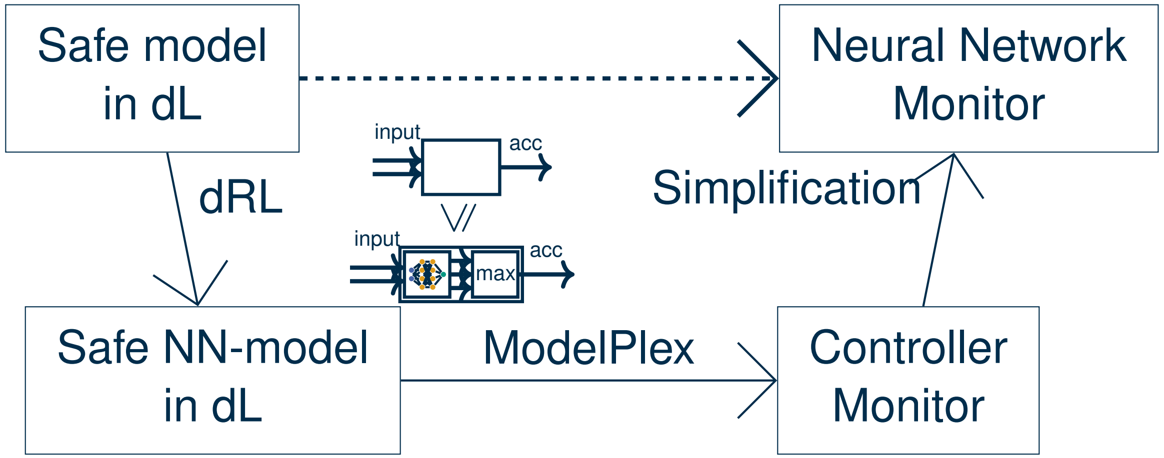
- General **dL model** for highway car control
- Derivation of **real arithmetic constraints** for monitoring/shielding/verification
- An ***empirical validation*** of all three dL-based safeguarding techniques

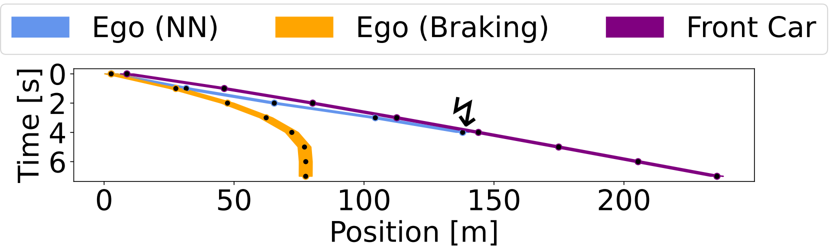
All presented techniques are **general**!

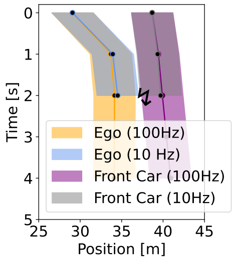
Observations

- Consistency between different views of the system (model, simulation,...) is challenging
- **BUT:** Consistency is paramount to train provably safe ML systems

$$\text{sys} ::= \Big(\underbrace{\mathtt{ctrl}_o; (\mathtt{ctrl}_e \cup ?t < t_e + T)}_{\text{control}}; \underbrace{\mathtt{accelCorr}; \mathtt{dyn}}_{\text{plant}}\Big)^*$$







Literature I

[1] Rose Bohrer et al. "VeriPhy: verified controller executables from verified cyber-physical system models". In: *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2018, Philadelphia, PA, USA, June 18-22, 2018*. Ed. by Jeffrey S. Foster and Dan Grossman. ACM, 2018, pp. 617–630. DOI: 10.1145/3192366.3192406.

[2] Nathan Fulton and André Platzer. "Safe Reinforcement Learning via Formal Methods: Toward Safe Control Through Proof and Learning". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by Sheila A. McIlraith and Kilian Q. Weinberger. AAAI Press, 2018, pp. 6485–6492. DOI: 10.1609/aaai.v32i1.12107.

[3] Stefan Mitsch and André Platzer. "ModelPlex: verified runtime validation of verified cyber-physical system models". In: *Formal Methods Syst. Des.* 49.1-2 (2016), pp. 33–74. DOI: 10.1007/s10703-016-0241-z.

[4] Samuel Teuber, Stefan Mitsch, and André Platzer. "Provably Safe Neural Network Controllers via Differential Dynamic Logic". In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson et al. Curran Associates, Inc., 2024. URL: https://doi.org/10.48550/arXiv.2402.10998.

Satefy formula when behind

$$
\begin{aligned}
x_e + L \le x_o \land \big(& a_e \le B_{\min} \land \mathrm{pos}_e(B_{\min}) + L < \mathrm{pos}_o \\
&\lor B_{\min} \le a_e \land v_e + a_e T < 0 \land \mathrm{pos}_e(a_e) + L < \mathrm{pos}_o \\
&\lor B_{\min} \le a_e \land v_e + a_e T \ge 0 \land \mathrm{pos}_e(B_{\min}) + \mathrm{corrDist} + L < \mathrm{pos}_o\big)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{pos}_e(a_e) &= x_e - \frac{v_e^2}{2a_e} \\
\mathrm{pos}_o &= x_o - \frac{v_o^2}{2B_{\max}} \\
\mathrm{corrDist} &= (\frac{-a_e}{B_{\min}} + 1)(\frac{a_e}{2}T^2 + Tv_e)
\end{aligned}
$$

Full dL Model

$$
\begin{array}{r|l}
\texttt{ctrl}_o & a_o := *; ?(B_{\max} \le a_o \le A_{\max}); \\
\texttt{ctrl}_e & a_e := *; ?(B_{\max} \le a_e \le A_{\max}); t_e := t; \\
 & \texttt{if}(\neg(\texttt{safeBack} \vee \texttt{safeFront})) \\
 & \quad \texttt{if}(x_e \le x_o) \\
 & \qquad a_e := *; ?(B_{\max} \le a_e \le B_{\min}); \\
 & \quad \texttt{else} \\
 & \qquad a_e := *; ?(A_{\min} \le a_e \le A_{\max}); \\
\texttt{accelCorr} & \texttt{if } (v_o = 0 \wedge a_o < 0) \vee (v_o = V \wedge a_o > 0)\, a_o := 0 \\
 & \texttt{if } (v_e = 0 \wedge a_e < 0) \vee (v_e = V \wedge a_e > 0)\, a_e := 0 \\
\texttt{dyn} & x_e' = v_e, v_e' = a_e, x_o' = v_o, v_o' = a_o, t' = 1 \\
 & \qquad \& \; t \le t_e + T \wedge 0 \le v_e \le V \wedge 0 \le v_o \le V
\end{array}
$$