Fine-Tuning Language Models Using Formal Methods Feedback: A Use Case in Autonomous Systems

Yunhao Yang*, Neel P. Bhatt*, Tyler Ingebrand*, William Ward, Steven Carr,
Zhangyang Wang, and Ufuk Topcu

May 14th, 2024

TEXAS
The University of Texas at Austin

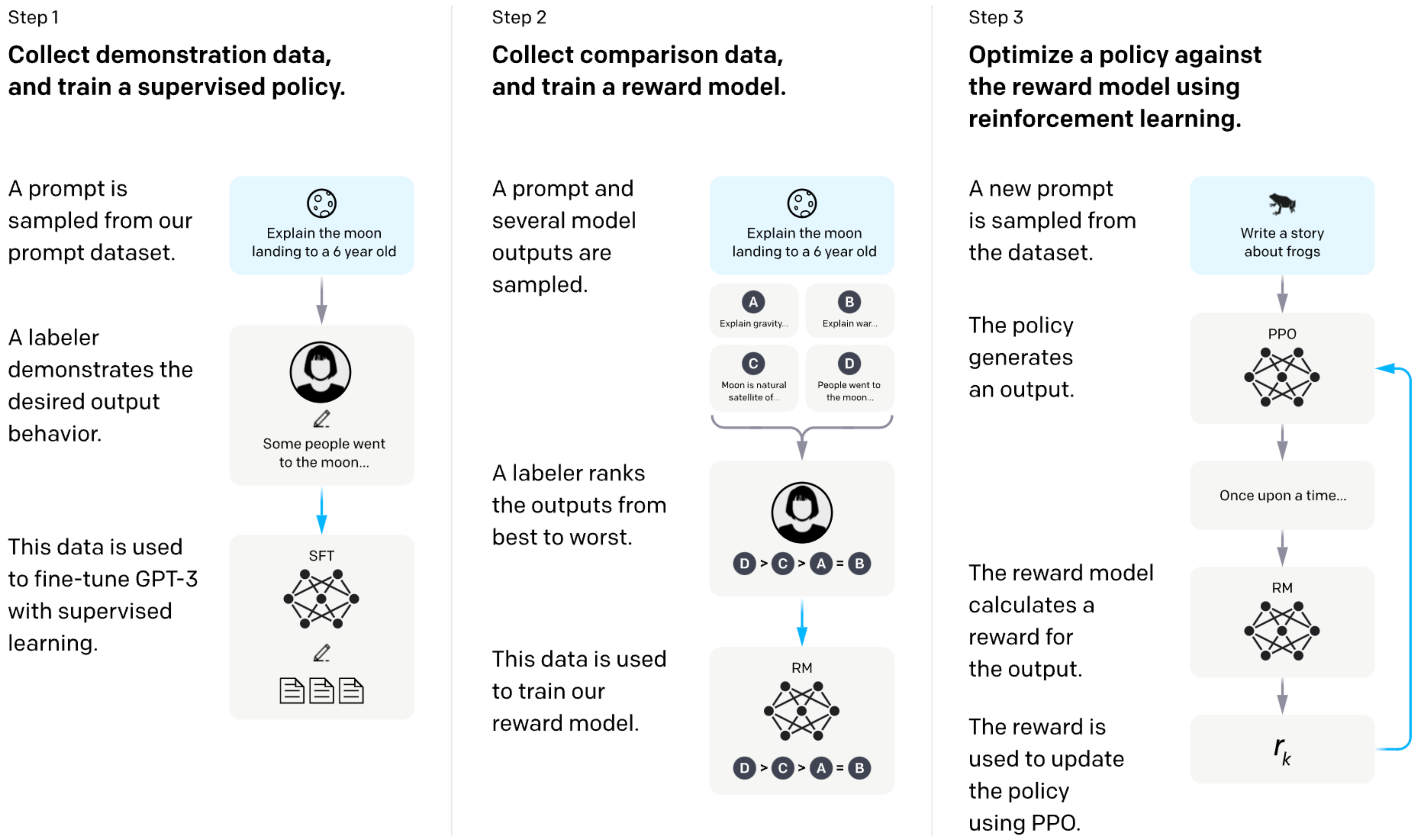
autonomy.oden.utexas.edu

CENTER FOR
autonomy

* Equal Contribution

Reinforcement Learning via Human Feedback (RLHF)

Example: OpenAI Scheme for Instruct GPT



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
D > C > A = B
RM
D > C > A = B

Step 3

Optimize a policy against the reward model using reinforcement learning.

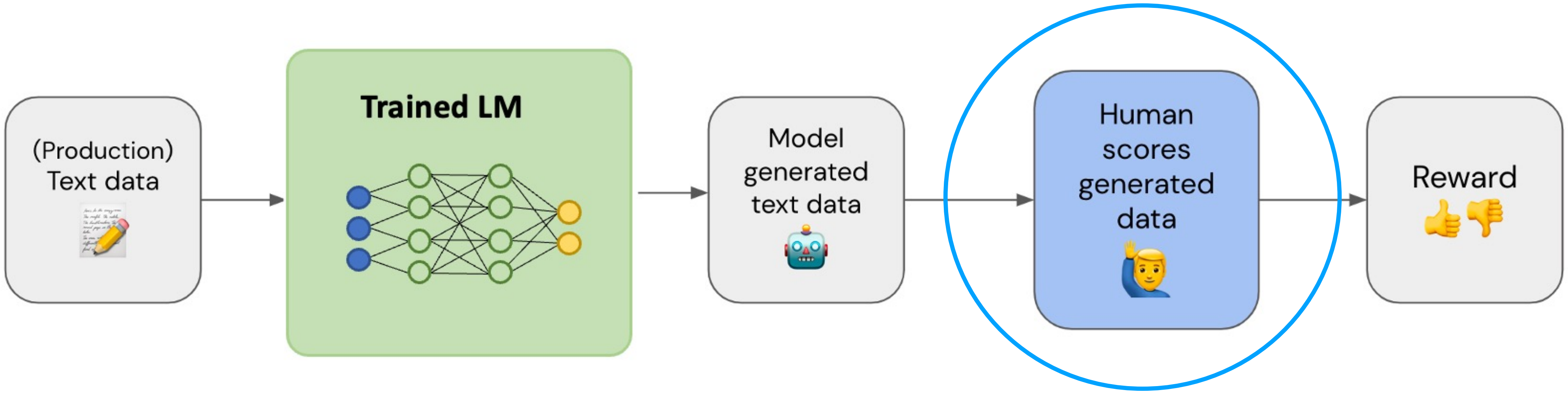
A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

Fine-Tuning Language Models Using Human Feedback



(Production) Text data 📝

Trained LM

Model generated text data 🤖

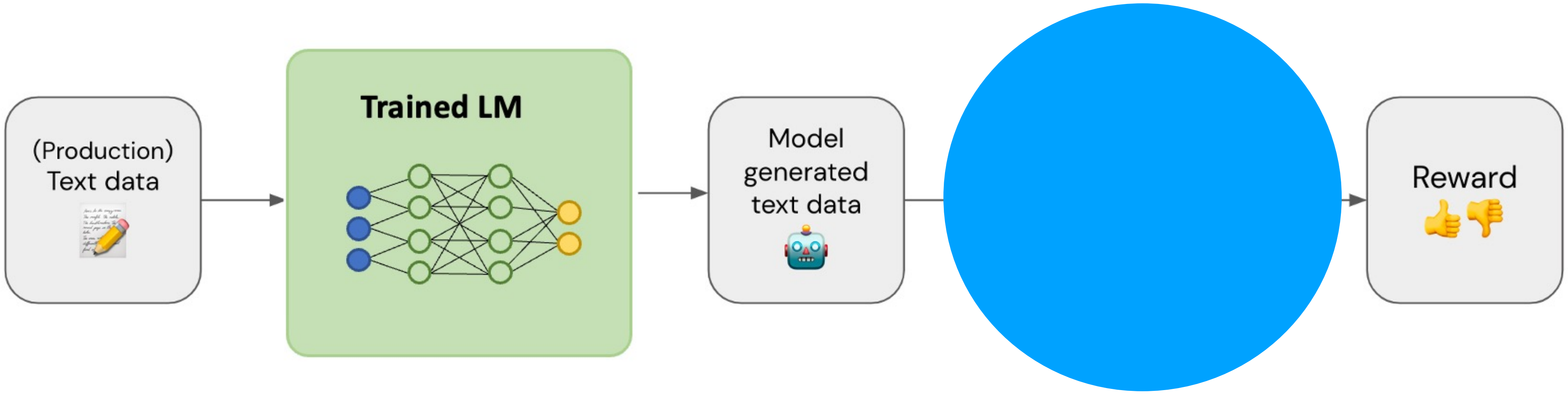
Human scores generated data 🙋‍♂️

Reward 👍👎

Labor-Intensive

Subjective/Inconsistent Feedback

Fine-Tuning Language Models Using Formal Methods Feedback?

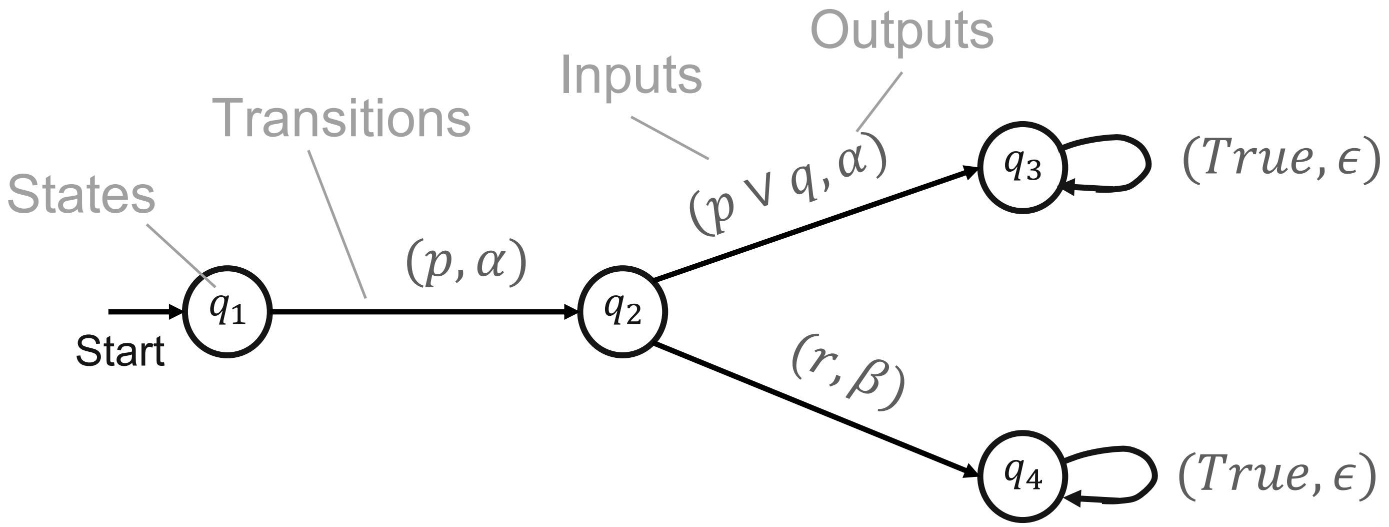


Formal Methods:
Automaton-Based Representation, Model Checking, Temporal Logic Specification, etc.

Labor-Intensive ✓

Subjective/Inconsistent Feedbac ✓

Background: Automaton-Based Representations

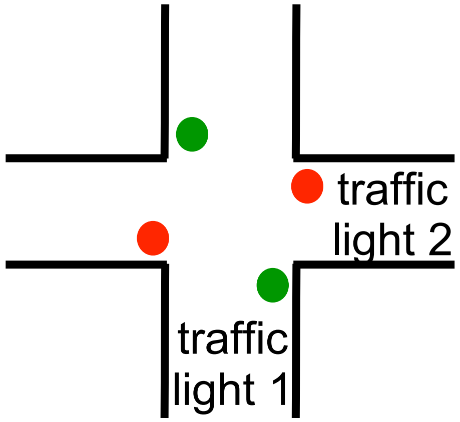


Why automaton-based representations? They are used for

- model checking, planning,…
- reactive synthesis, games on graphs, …
- probabilistic verification and synthesis, and
- reinforcement learning.

A (Very) Brief Introduction to Model Checking

Are the controller’s outcomes **guaranteed** to satisfy user-specified requirements when implemented against a system model?

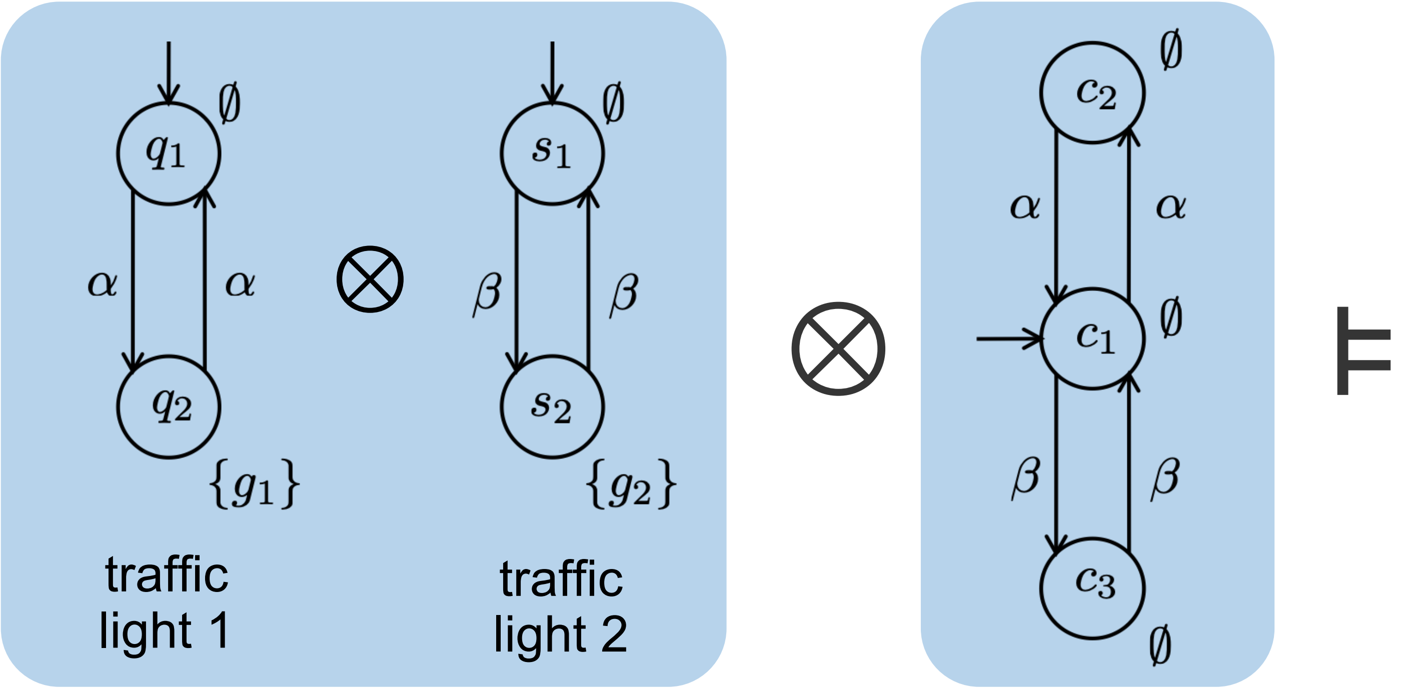


Model,
environment assumptions,
other side information

controller

system requirements

$$“\mathcal{M} \otimes \mathcal{C} \models \Phi”$$



$$\square \lozenge g_1 \wedge \square \lozenge g_2 \wedge \square(\neg g_1 \vee \neg g_2)$$

“Each light will always eventually be green, and the lights will never simultaneously be green.”

A (Very) Brief Introduction to Model Checking

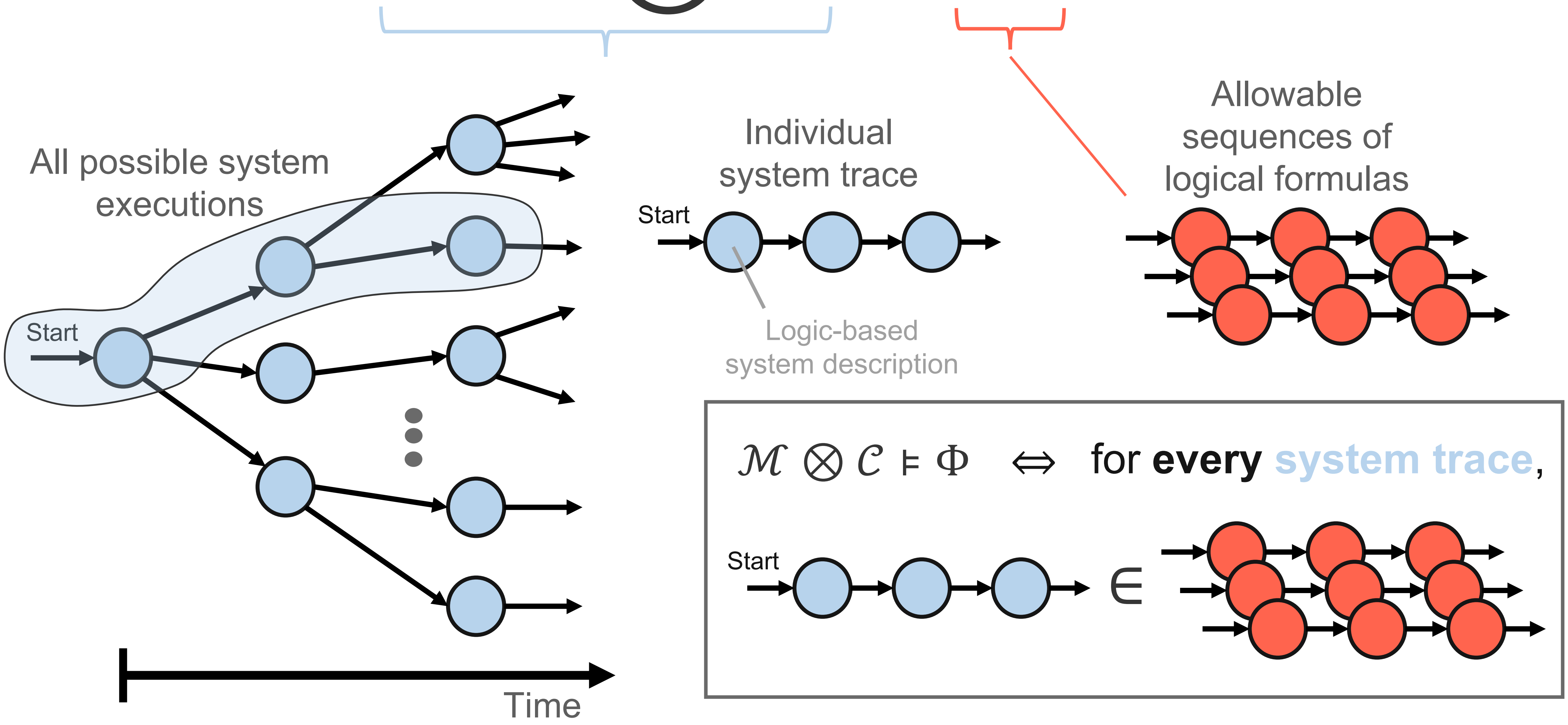
Are the controller's outcomes **guaranteed** to satisfy user-specified requirements when implemented against a system model?

Model,
environment assumptions,
other side information

Controller

System requirements

$$\text{“}\mathcal{M} \otimes \mathcal{C} \models \Phi\text{”}$$



All possible system executions

Start

Time

Individual system trace

Start

Logic-based system description

Allowable sequences of logical formulas

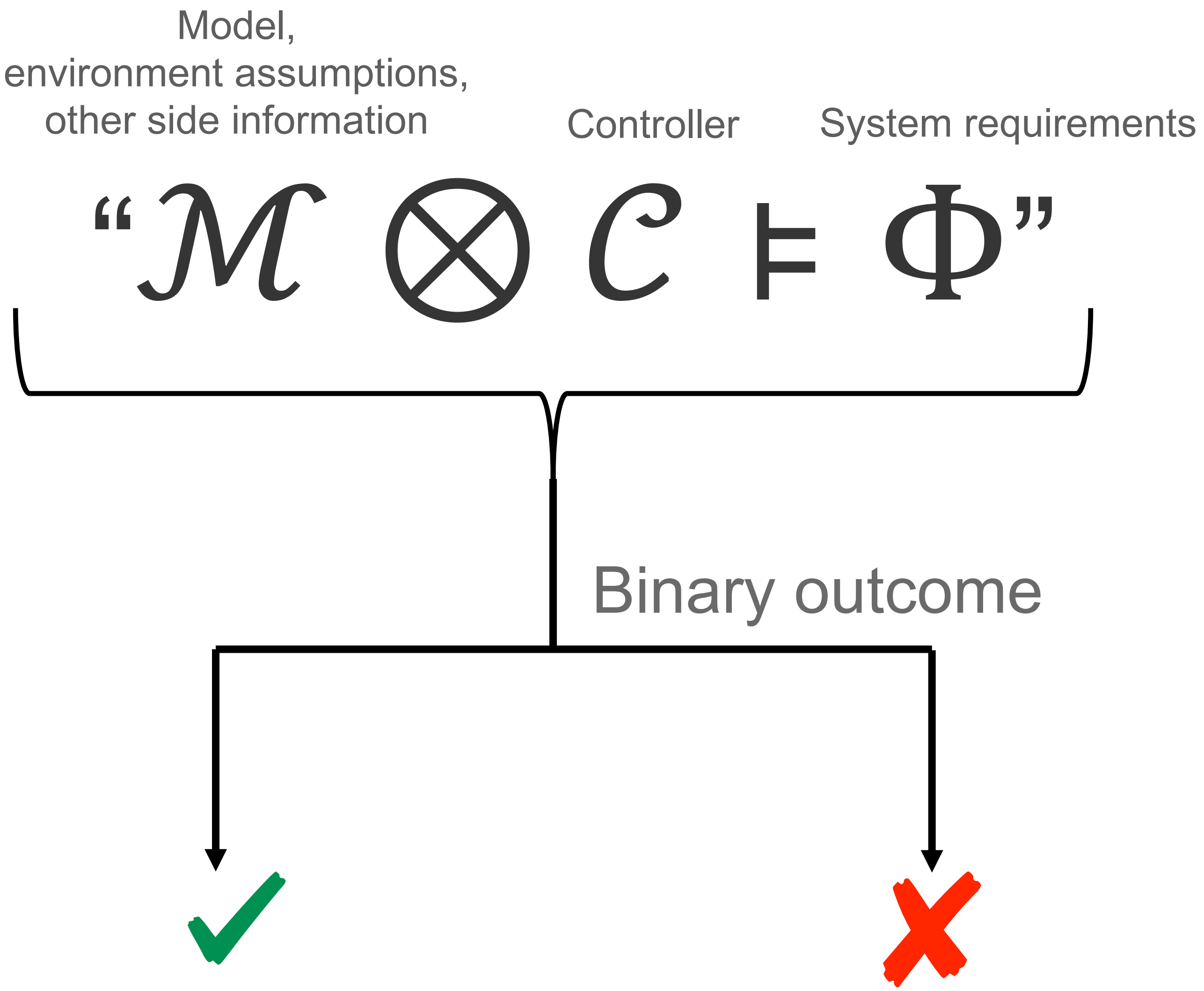
$\mathcal{M} \otimes \mathcal{C} \models \Phi \iff$ for **every** system trace,

Start

$\in$

A (Very) Brief Introduction to Model Checking

Are the controller’s outcomes **guaranteed** to satisfy user-specified requirements when implemented against a system model?



Model,
environment assumptions,
other side information

Controller

System requirements

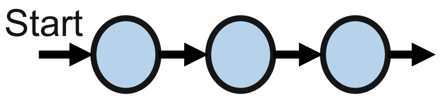
“$\mathcal{M} \otimes \mathcal{C} \models \Phi$”

Binary outcome

✓

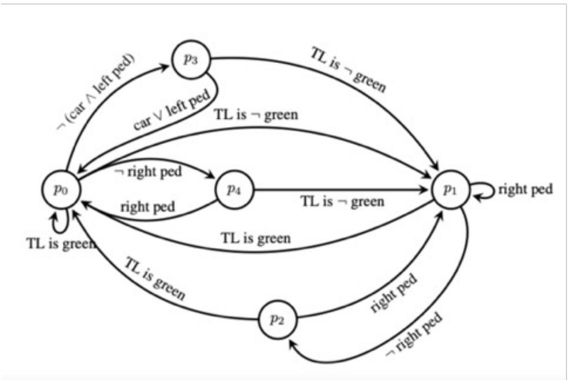
✗

Byproduct: Counterexample trace that **violates** the specification.



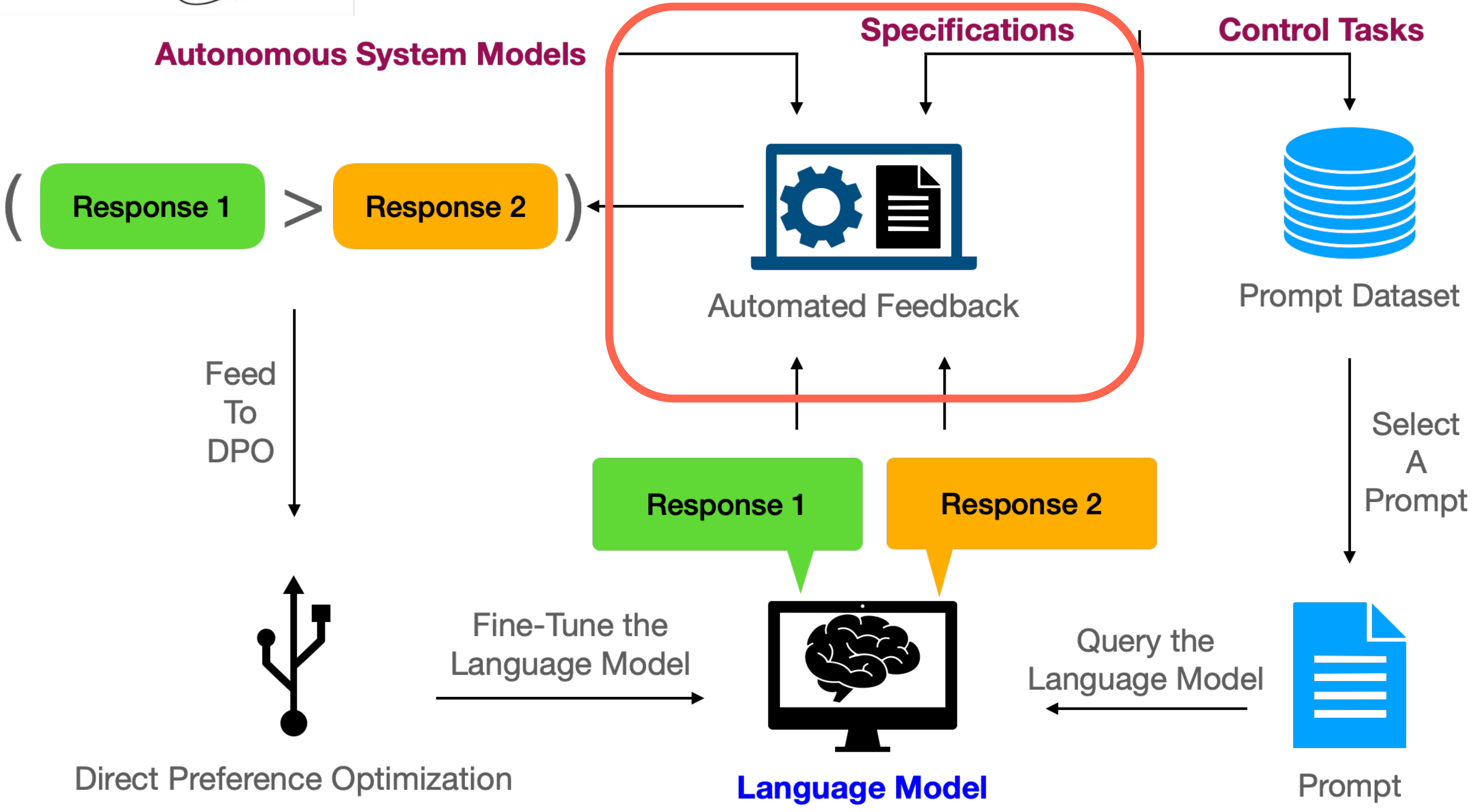
Fine-Tuning Language Models Using Formal Methods Feedback

Abstract Model

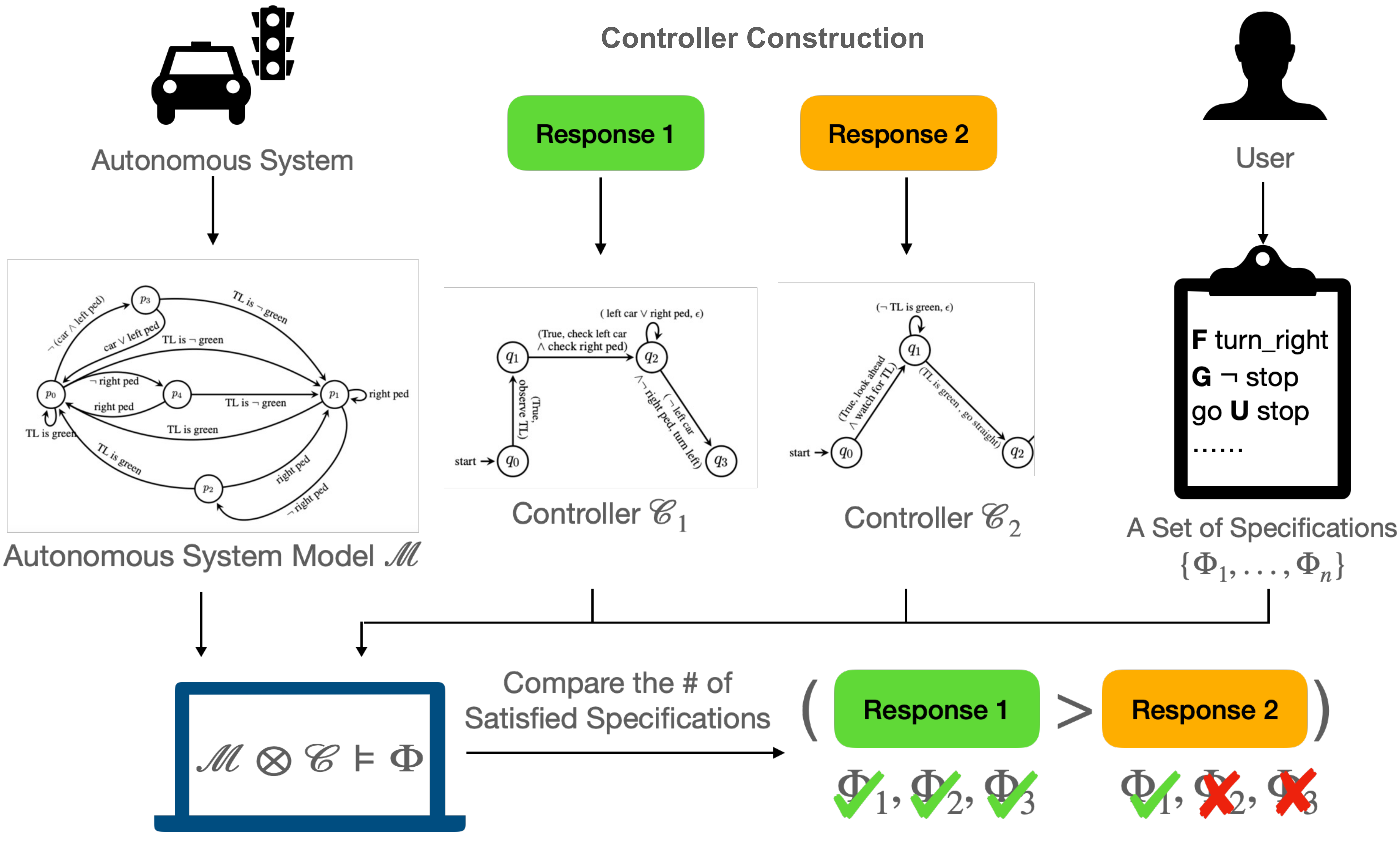


High-Fidelity Simulator

User



Fine-Tuning Language Models Using Formal Methods Feedback



Modeling the Autonomous System

Autonomous System

Autonomous System Model $\mathscr{M}$

Controller Construction

Response 1

Response 2

Controller $\mathscr{C}_1$

Controller $\mathscr{C}_2$

User

F turn_right
G ¬ stop
go **U** stop
……

A Set of Specifications
$\{\Phi_1, \ldots, \Phi_n\}$

$\mathscr{M} \otimes \mathscr{C} \models \Phi$

Model Checker

Compare the # of Satisfied Specifications

(**Response 1** > **Response 2**)

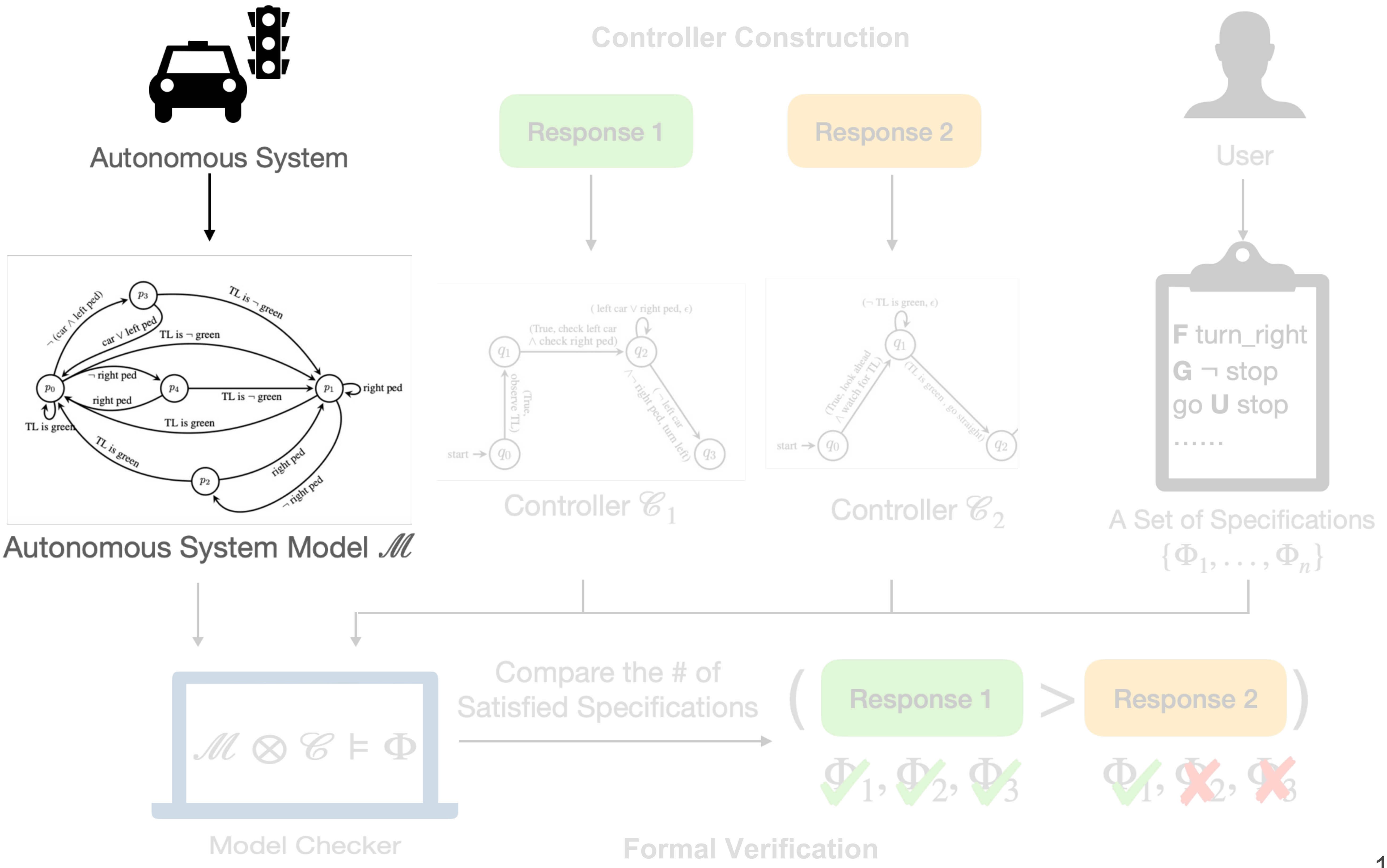
Φ_1 ✓, Φ_2 ✓, Φ_3 ✓

Φ_1 ✓, Φ_2 ✗, Φ_3 ✗

Formal Verification

Fine-Tuning Language Models Using Formal Methods Feedback

Modeling the Autonomous System



Autonomous System

Autonomous System Model $\mathscr{M}$

Controller Construction

Response 1

Response 2

User

Controller $\mathscr{C}_1$

Controller $\mathscr{C}_2$

F turn_right
G ¬ stop
go **U** stop
……

A Set of Specifications
$\{\Phi_1, \ldots, \Phi_n\}$

$\mathscr{M} \otimes \mathscr{C} \models \Phi$

Model Checker

Compare the # of Satisfied Specifications

(Response 1 > Response 2)

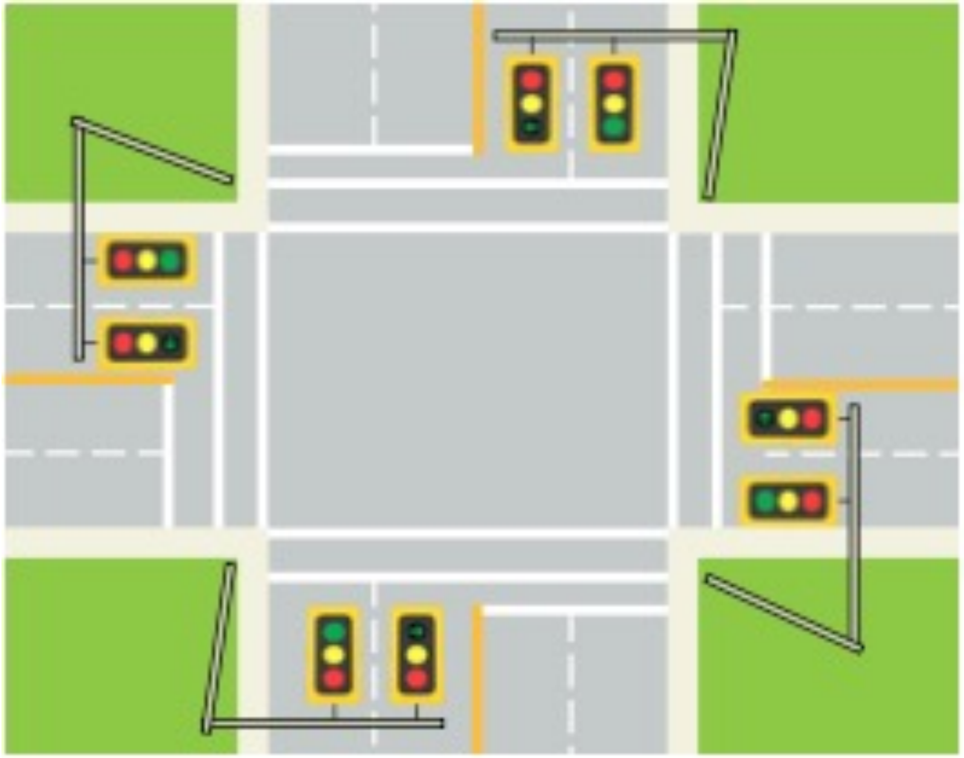
Φ_1 ✓, Φ_2 ✓, Φ_3 ✓

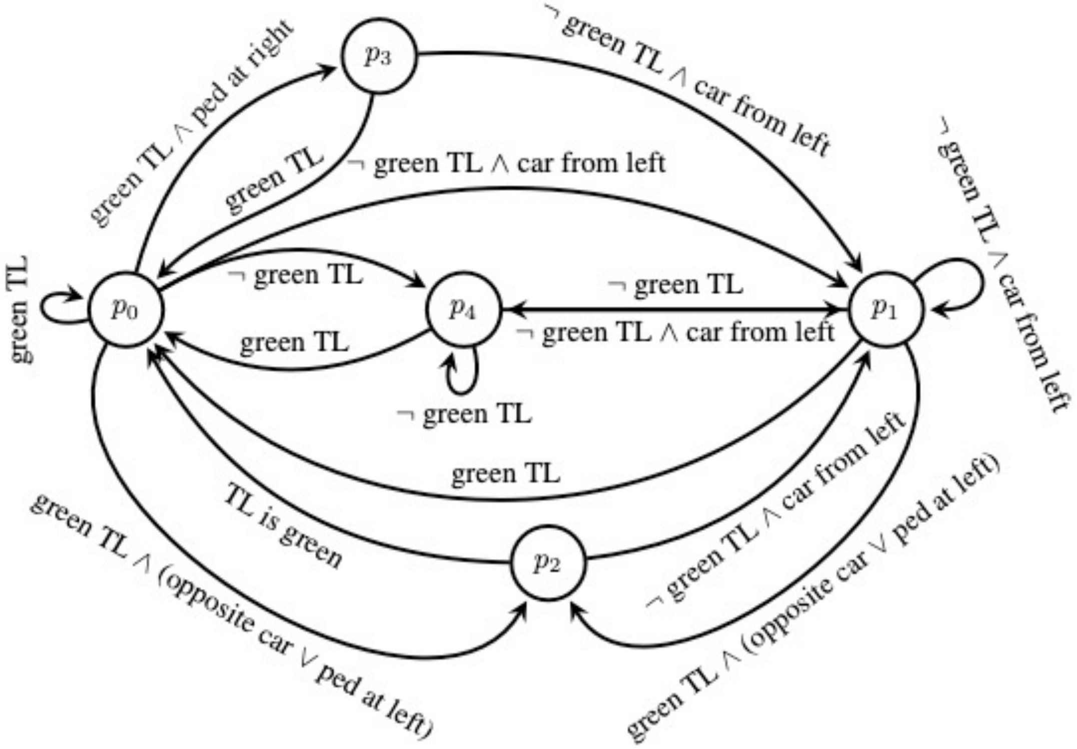
Φ_1 ✓, Φ_2 ✗, Φ_3 ✗

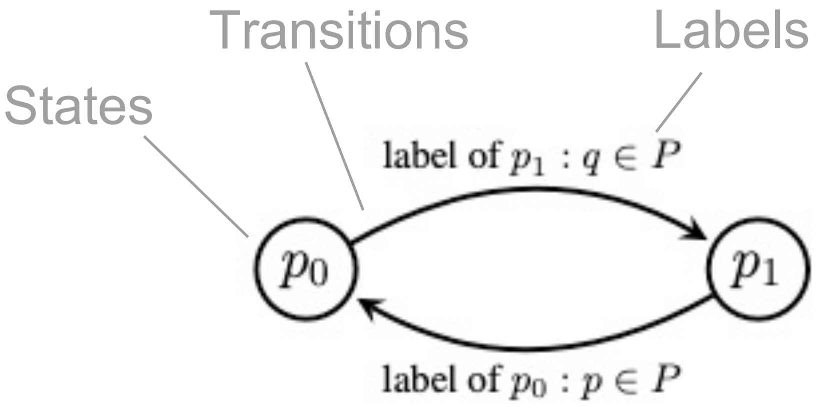
Formal Verification

Formal Methods Feedback in an Autonomous Driving System

Modeling the Autonomous System




p_0
p_1
p_2
p_3
p_4
green TL ∧ ped at right
green TL
¬ green TL ∧ car from left
¬ green TL ∧ car from left
¬ green TL ∧ car from left
green TL
¬ green TL
green TL
¬ green TL
¬ green TL ∧ car from left
¬ green TL
green TL
TL is green
green TL ∧ (opposite car ∨ ped at left)
¬ green TL ∧ car from left
green TL ∧ (opposite car ∨ ped at left)



Formal Methods Feedback in an Autonomous Driving System

Modeling the Autonomous System

$P = \{green\ light\}$

$S \sim Traffic\ Light$

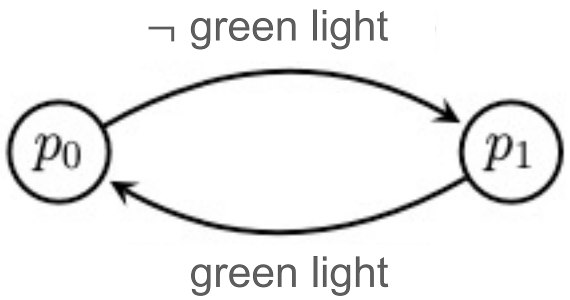
$Q_M = \{p_0\ p_1\}$

$\lambda_M(p_0) = green\ light$

$\lambda_M(p_1) = \neg\ green\ light$

$\delta_M(p_0, p_1) = 1$

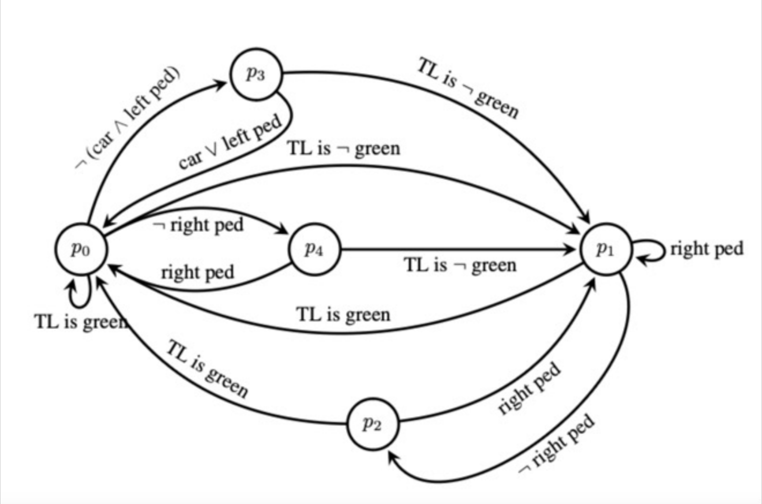
$\delta_M(p_1, p_0) = 1$



Fine-Tuning Language Models Using Formal Methods Feedback

Modeling the Autonomous System

Autonomous System

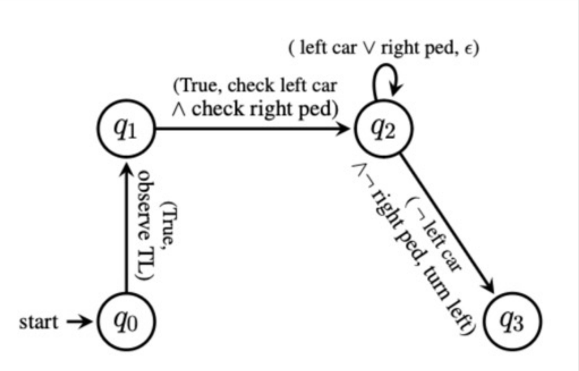


Autonomous System Model $\mathscr{M}$

Controller Construction

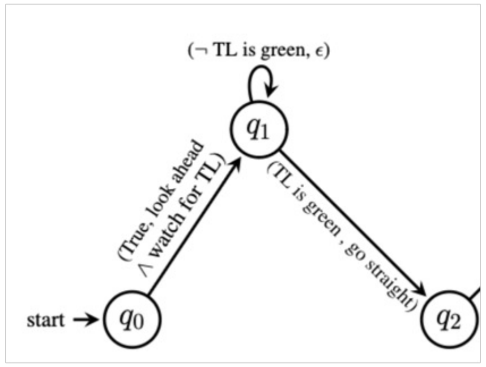
Response 1

Response 2



(left car ∨ right ped, ϵ)
(True, check left car ∧ check right ped)
q_1
q_2
(True, observe TL)
(¬ left car ∧ ¬ right ped, turn left)
start
q_0
q_3

Controller $\mathscr{C}_1$



Controller $\mathscr{C}_2$

User

F turn_right
G ¬ stop
go **U** stop
……

A Set of Specifications
$\{\Phi_1, \ldots, \Phi_n\}$

$\mathscr{M} \otimes \mathscr{C} \models \Phi$

Model Checker

Compare the # of Satisfied Specifications

(Response 1 > Response 2)

Response 1: Φ_1 ✓, Φ_2 ✓, Φ_3 ✓

Response 2: Φ_1 ✓, Φ_2 ✗, Φ_3 ✗

Formal Verification

Formal Methods Feedback in an Autonomous Driving System

Controller Construction

User

Steps for turning right at the traffic light

1. Look straight ahead and watch for traffic light.
2. If the traffic light turns green, start moving forward.
3. As you approach the intersection, look to your left for oncoming traffic.
4. If there is no traffic coming from your left, check pedestrians on your right.
5. If it is safe, turn your vehicle right.

Language Model

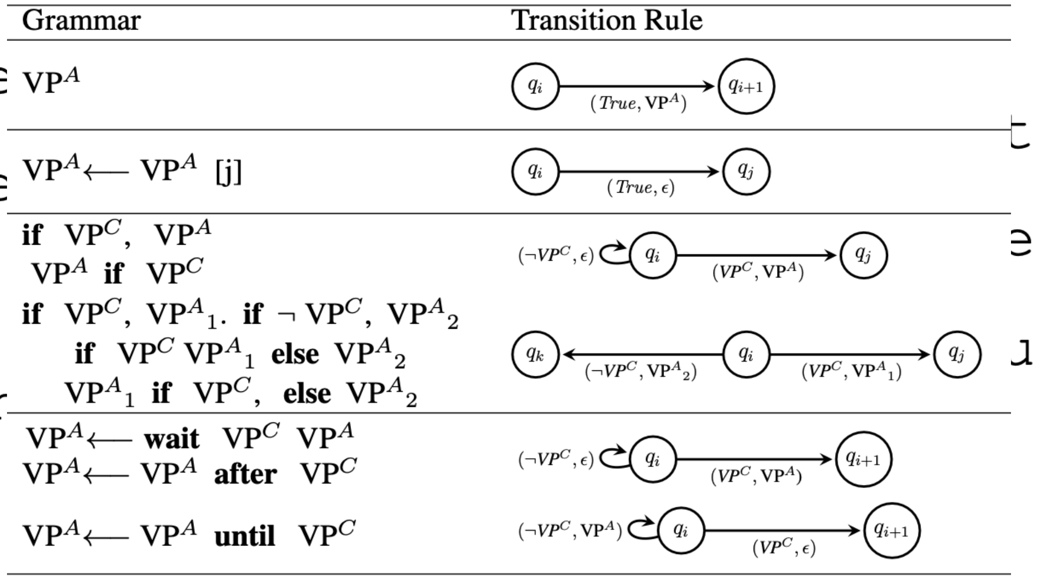
1. Observe the traffic light in front of you.
2. Check for the left approaching car and right side pedestrian.
3. If no car from the left is approaching and no pedestrian on the right, proceed to turn right.

Formal Methods Feedback in an Autonomous Driving System

Controller Construction

1. Look straight ahead and watch for traffic light.
2. If the traffic light turns green, start moving forward.
3. As you approach the intersection, look to your left for oncoming traffic.
4. If there is no traffic coming from your left, check pedestrians on your right.
5. If it is safe, turn your vehicle right.

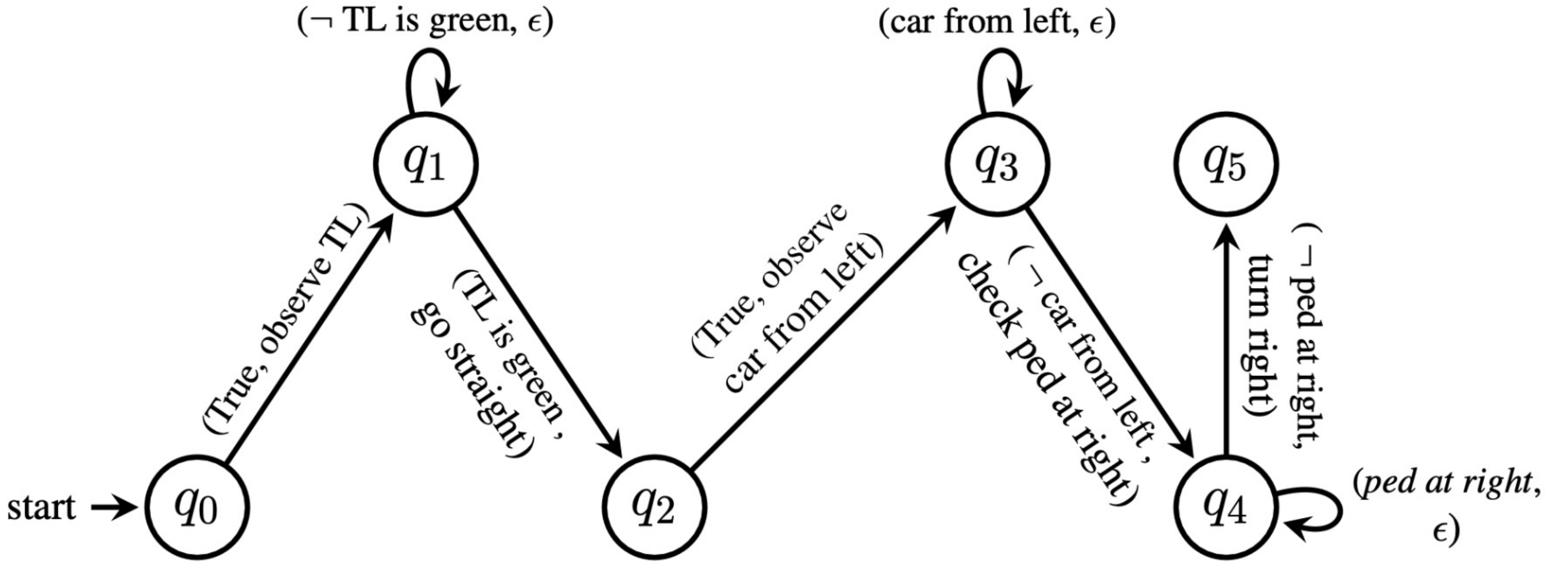
```
1. <obse
2. <if>                                              traight>.
3. <obse
4. <if>                                              edestrian
      at
5. <if>                                              urn
     righ
```



Grammar	Transition Rule
VP^A	$q_i \xrightarrow{(True, \mathrm{VP}^A)} q_{i+1}$
$\mathrm{VP}^A \longleftarrow \mathrm{VP}^A$ [j]	$q_i \xrightarrow{(True, \epsilon)} q_j$
if VP^C, VP^A VP^A **if** VP^C	$(\neg VP^C, \epsilon)$ loop on q_i; $q_i \xrightarrow{(VP^C, \mathrm{VP}^A)} q_j$

if VP^C, ${\mathrm{VP}^A}_1$. **if** $\neg$ VP^C, ${\mathrm{VP}^A}_2$
if VP^C ${\mathrm{VP}^A}_1$ **else** ${\mathrm{VP}^A}_2$
${\mathrm{VP}^A}_1$ **if** VP^C, **else** ${\mathrm{VP}^A}_2$
	$q_k \xleftarrow{(\neg VP^C, {\mathrm{VP}^A}_2)} q_i \xrightarrow{(VP^C, {\mathrm{VP}^A}_1)} q_j$

$\mathrm{VP}^A \longleftarrow$ **wait** VP^C VP^A $\mathrm{VP}^A \longleftarrow \mathrm{VP}^A$ **after** VP^C	$(\neg VP^C, \epsilon)$ loop on q_i; $q_i \xrightarrow{(VP^C, \mathrm{VP}^A)} q_{i+1}$
$\mathrm{VP}^A \longleftarrow \mathrm{VP}^A$ **until** VP^C	$(\neg VP^C, \mathrm{VP}^A)$ loop on q_i; $q_i \xrightarrow{(VP^C, \epsilon)} q_{i+1}$

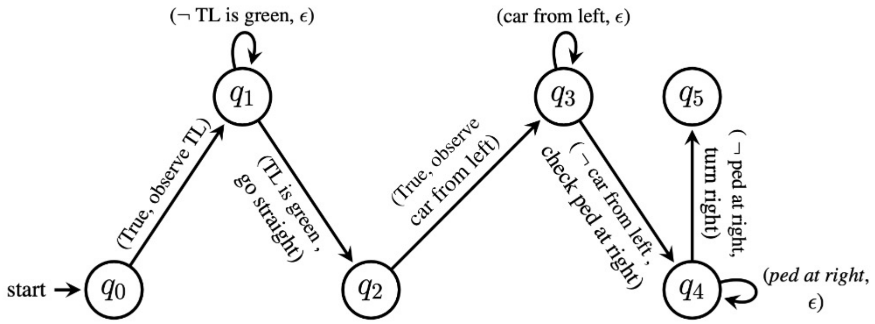


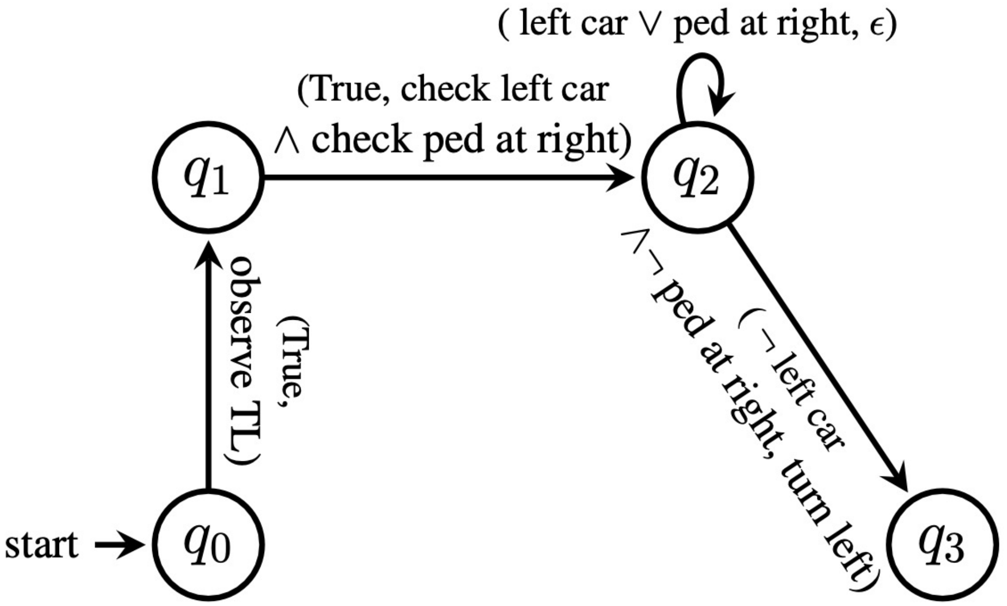
start
q_0
(True, observe TL)
q_1
(¬ TL is green, ε)
(TL is green , go straight)
q_2
(True, observe car from left)
q_3
(car from left, ε)
(¬ car from left , check ped at right)
q_4
(ped at right, ε)
(¬ ped at right, turn right)
q_5

Formal Methods Feedback in an Autonomous Driving System

Controller Construction

1. Look straight ahead and watch for traffic light.
2. If the traffic light turns green, start moving forward.
3. As you approach the intersection, look to your left for oncoming traffic.
4. If there is no traffic coming from your left, check pedestrians on your right.
5. If it is safe, turn your vehicle right.


(¬ TL is green, ϵ)
(car from left, ϵ)
start
q_0
q_1
q_2
q_3
q_4
q_5
(True, observe TL)
(TL is green , go straight)
(True, observe car from left)
(¬ car from left , check ped at right)
(¬ ped at right, turn right)
(ped at right, ϵ)


(left car ∨ ped at right, ϵ)
(True, check left car ∧ check ped at right)
start
q_0
q_1
q_2
q_3
(True, observe TL)
(¬ left car ∧¬ ped at right, turn left)

1. Observe the traffic light in front of you.
2. Check for the left approaching car and right side pedestrian.
3. If no car from the left is approaching and no pedestrian on the right, proceed to turn right.

Fine-Tuning Language Models Using Formal Methods Feedback

Modeling the Autonomous System

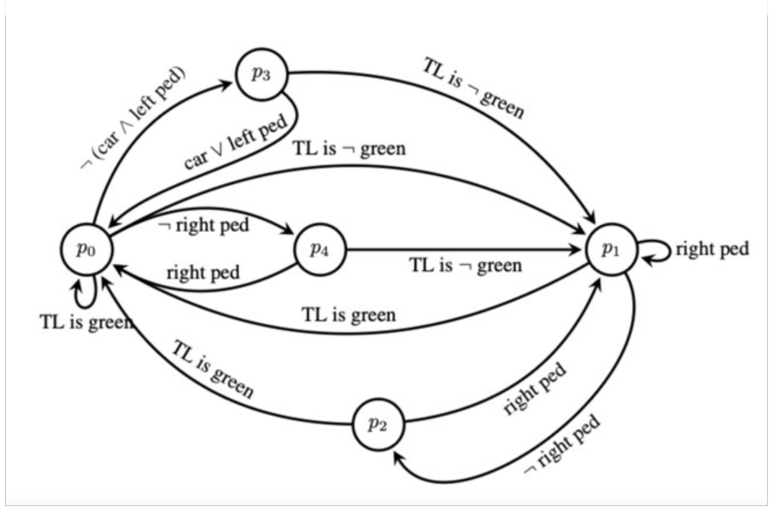
Autonomous System

Controller Construction

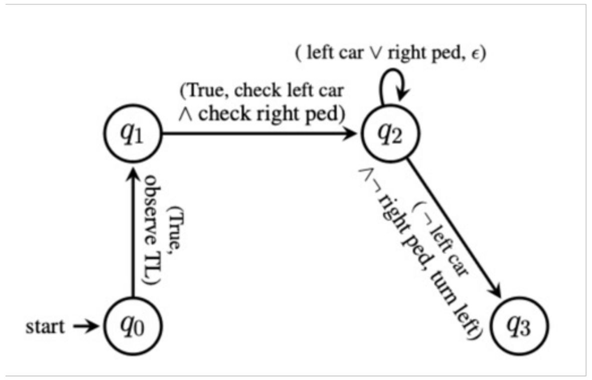
Response 1

Response 2

User

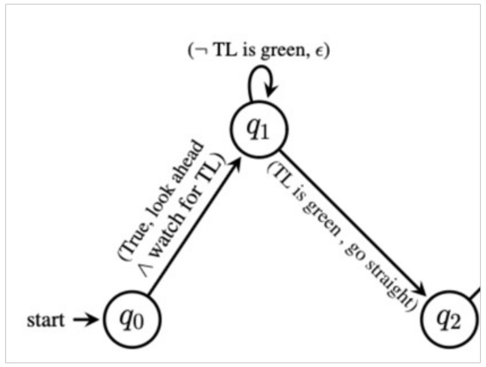


Autonomous System Model $\mathscr{M}$



(left car ∨ right ped, ϵ)
(True, check left car ∧ check right ped)
q_1
q_2
(True, observe TL)
(¬ left car ∧ ¬ right ped, turn left)
start
q_0
q_3

Controller $\mathscr{C}_1$



(¬ TL is green, ϵ)
q_1
(True, look ahead ∧ watch for TL)
(TL is green , go straight)
start
q_0
q_2

Controller $\mathscr{C}_2$

F turn_right
G ¬ stop
go **U** stop
……

A Set of Specifications
$\{\Phi_1, \ldots, \Phi_n\}$

$\mathscr{M} \otimes \mathscr{C} \models \Phi$

Model Checker

Compare the # of Satisfied Specifications

(Response 1 > Response 2)

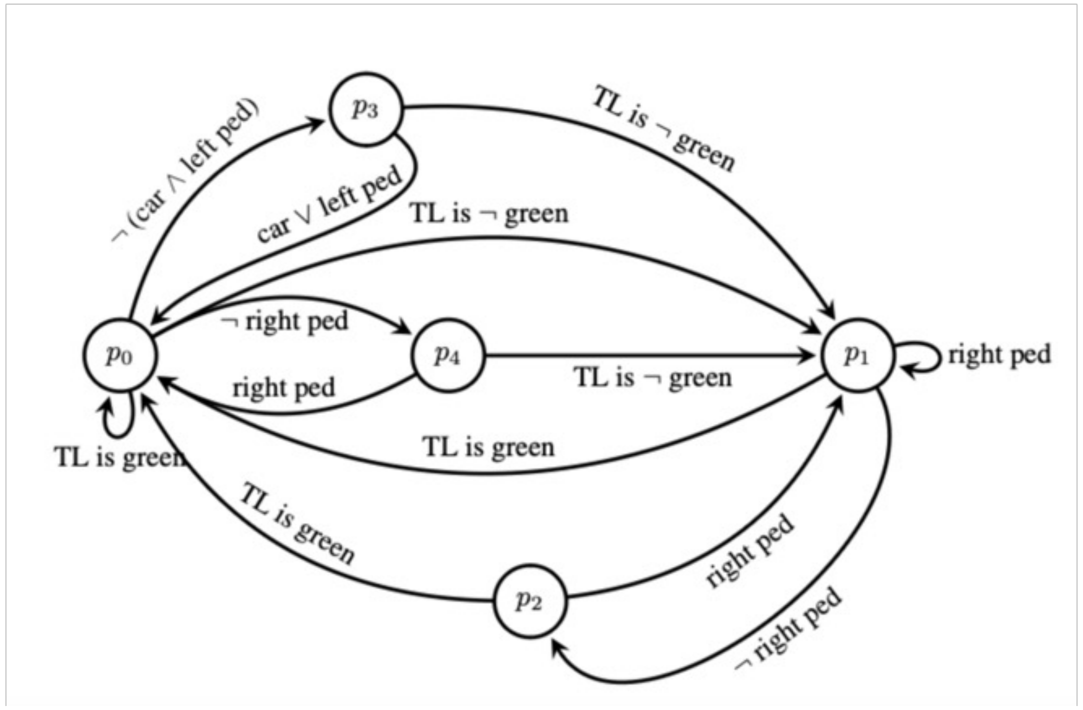
Φ_1 ✓, Φ_2 ✓, Φ_3 ✓

Φ_1 ✓, Φ_2 ✗, Φ_3 ✗

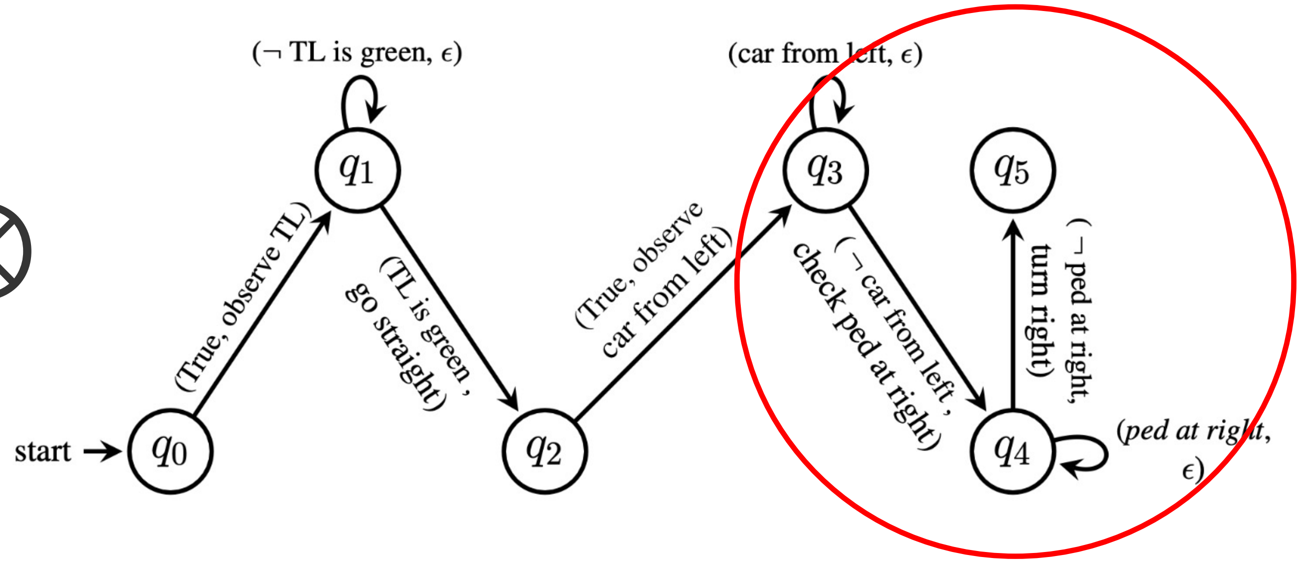
Formal Verification

Formal Methods Feedback in an Autonomous Driving System

Formal Verification



$\otimes$


(¬ TL is green, ϵ)
q_1
(True, observe TL)
(TL is green, go straight)
start
q_0
q_2
(True, observe car from left)
(car from left, ϵ)
q_3
(¬ car from left, check ped at right)
q_5
(¬ ped at right, turn right)
q_4
(ped at right, ϵ)

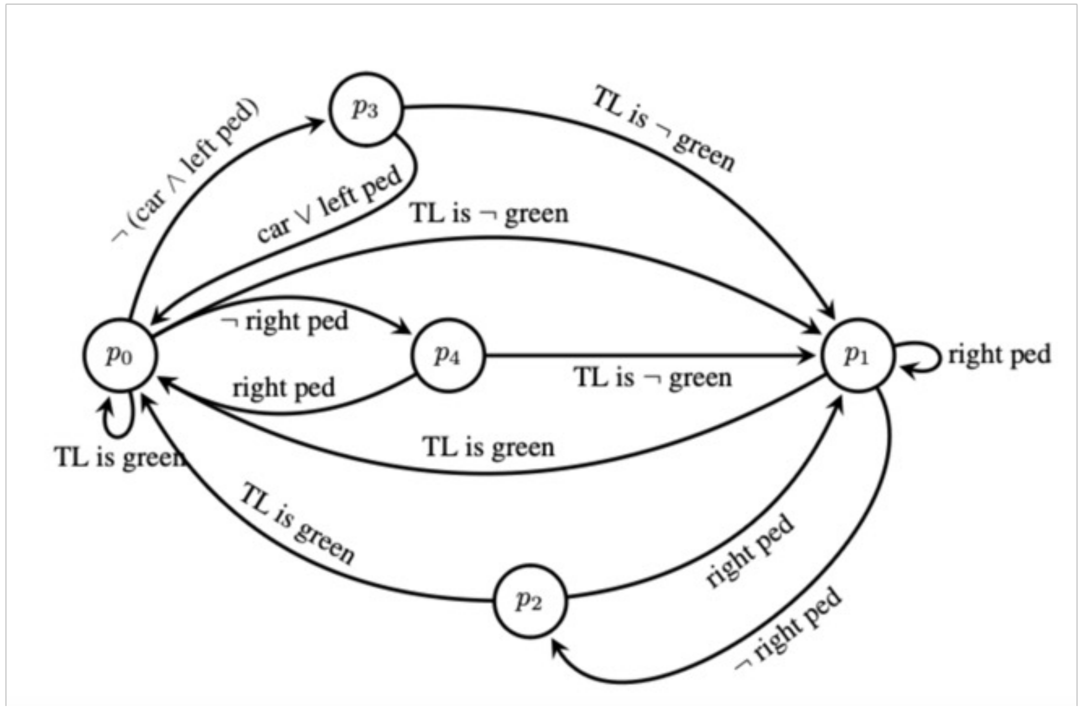
$\square(\neg\text{green traffic light} \rightarrow \neg\text{go straight})$, ✓

$\square(\text{stop sign} \rightarrow \Diamond\,\text{stop})$, ✓

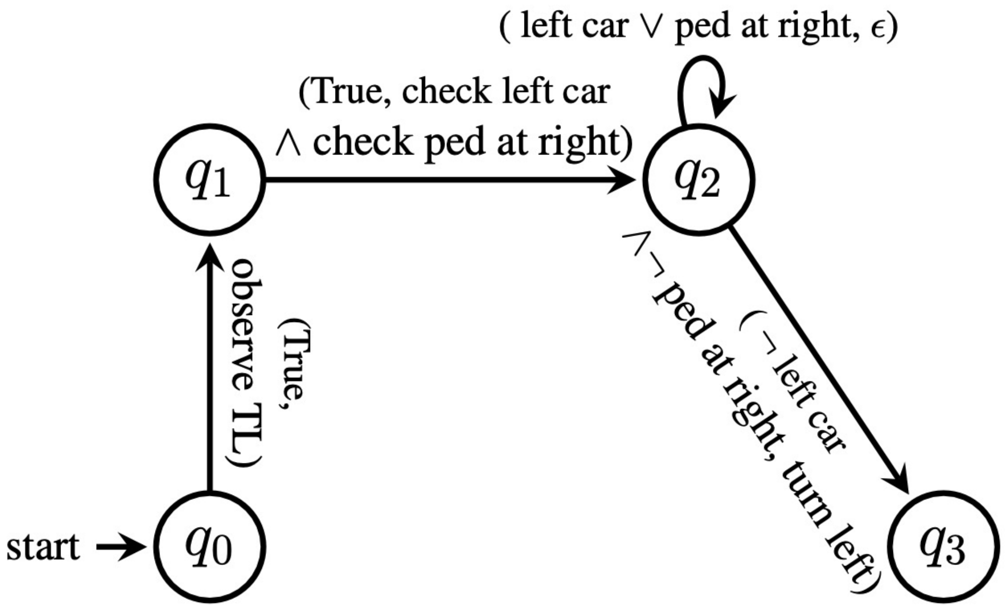
$\square\,\neg\text{turn right} \vee \neg(\text{car from left} \vee \text{pedestrian at right})$, ✗

Formal Methods Feedback in an Autonomous Driving System

Formal Verification



$\otimes$



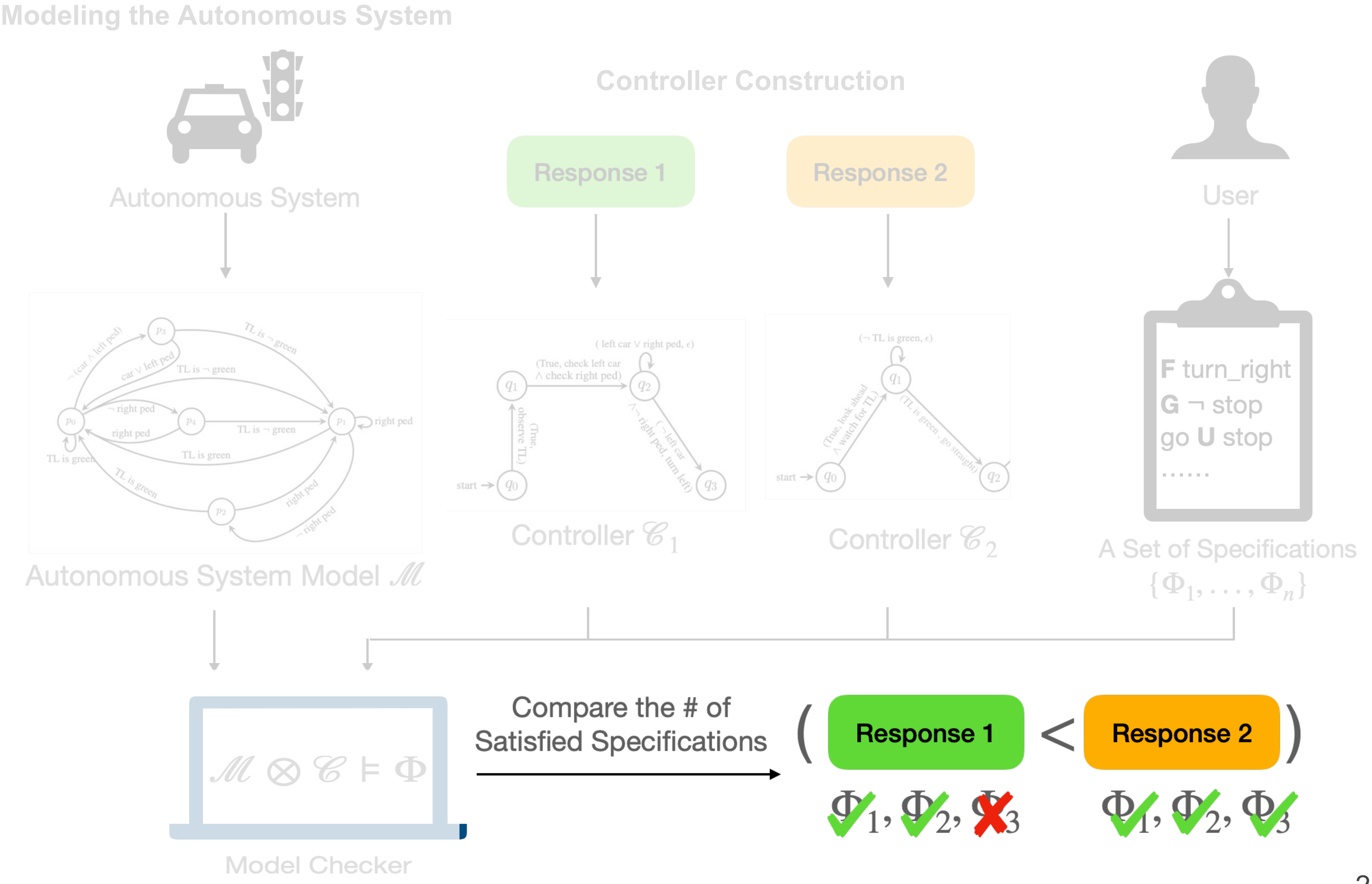
(left car ∨ ped at right, ϵ)
(True, check left car ∧ check ped at right)
q_1
q_2
(True, observe TL)
start
q_0
(¬ left car ∧¬ ped at right, turn left)
q_3

$\square(\neg\text{green traffic light} \rightarrow \neg\text{go straight})$, ✓

$\square(\text{stop sign} \rightarrow \Diamond\, \text{stop})$, ✓

$\square\, \neg\text{turn right} \vee \neg(\text{car from left} \vee \text{pedestrian at right})$, ✓

Fine-Tuning Language Models Using Formal Methods Feedback



Modeling the Autonomous System

Controller Construction

Autonomous System

Response 1

Response 2

User

F turn_right
G ¬ stop
go **U** stop
……

Autonomous System Model $\mathscr{M}$

Controller $\mathscr{C}_1$

Controller $\mathscr{C}_2$

A Set of Specifications
$\{\Phi_1, \ldots, \Phi_n\}$

$\mathscr{M} \otimes \mathscr{C} \models \Phi$

Model Checker

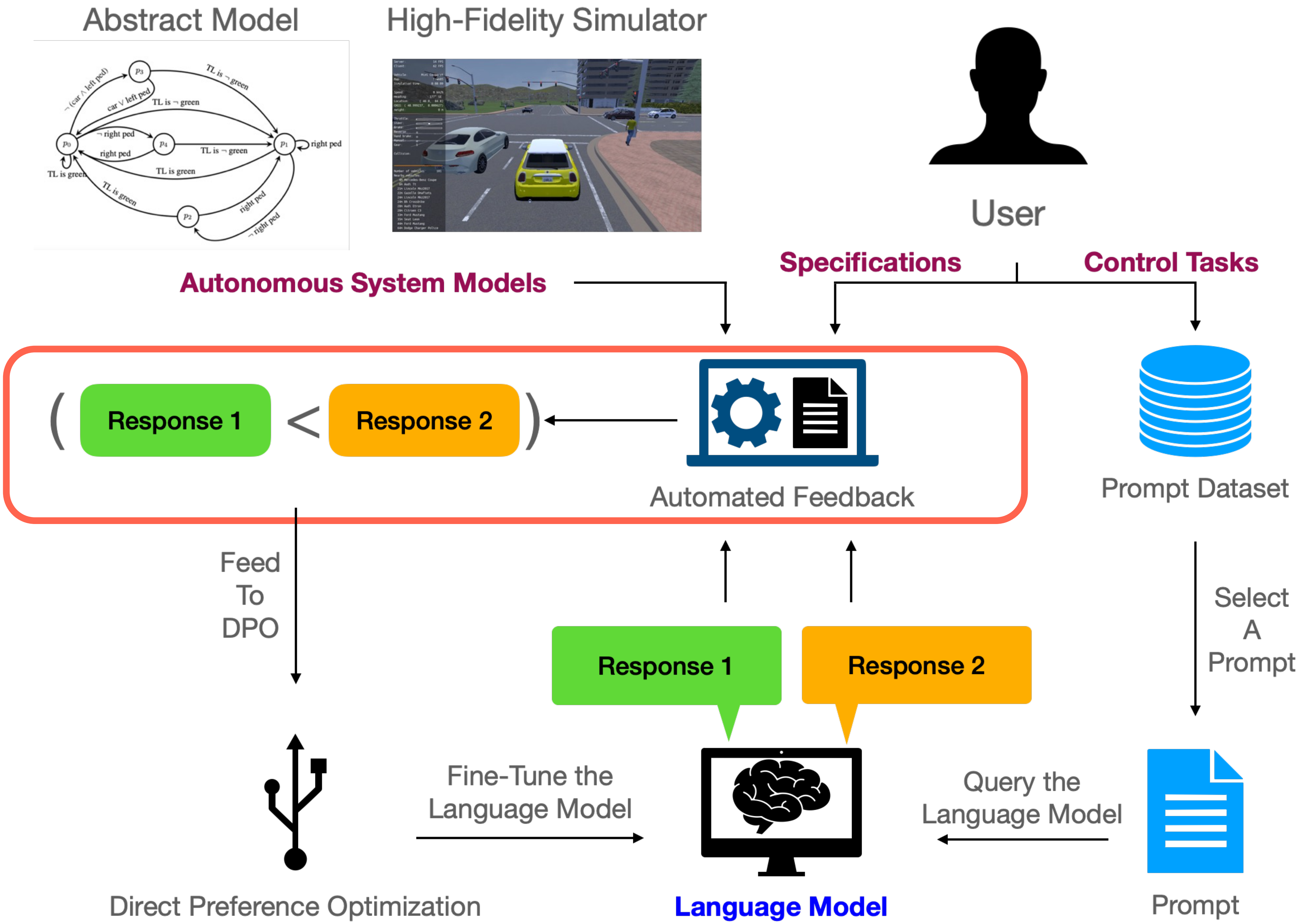
Compare the # of Satisfied Specifications

(**Response 1** < **Response 2**)

Φ_1 ✓, Φ_2 ✓, Φ_3 ✗

Φ_1 ✓, Φ_2 ✓, Φ_3 ✓

Fine-Tuning Language Models Using Formal Methods Feedback



Abstract Model

High-Fidelity Simulator

User

Autonomous System Models

Specifications

Control Tasks

(**Response 1** < **Response 2**)

Automated Feedback

Prompt Dataset

Feed To DPO

Select A Prompt

Response 1

Response 2

Fine-Tune the Language Model

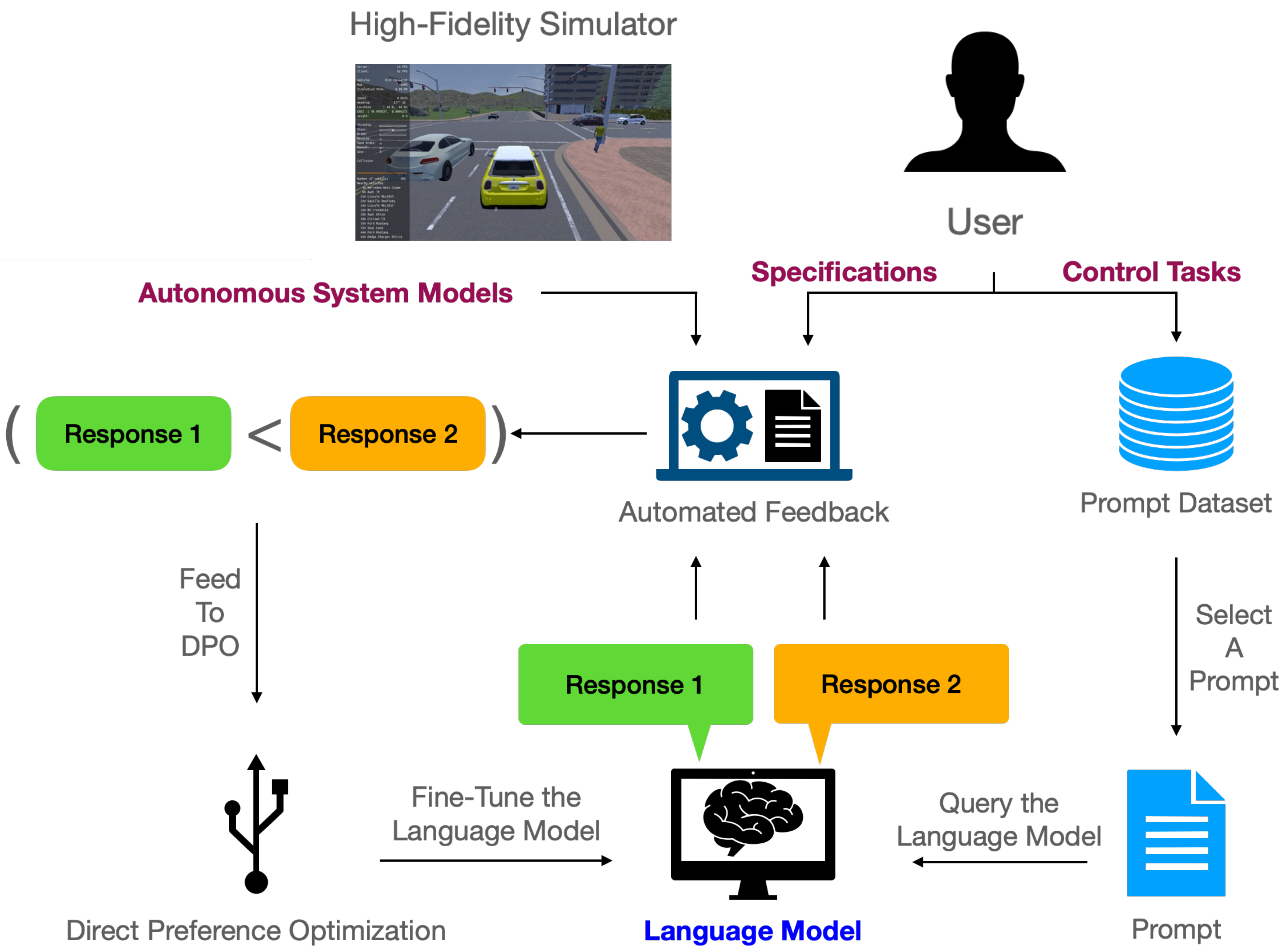
Query the Language Model

Direct Preference Optimization

Language Model

Prompt

Fine-Tuning Language Models Using Formal Methods Feedback



High-Fidelity Simulator

User

Autonomous System Models

Specifications

Control Tasks

(Response 1 < Response 2)

Automated Feedback

Prompt Dataset

Feed To DPO

Select A Prompt

Response 1

Response 2

Fine-Tune the Language Model

Query the Language Model

Direct Preference Optimization

Language Model

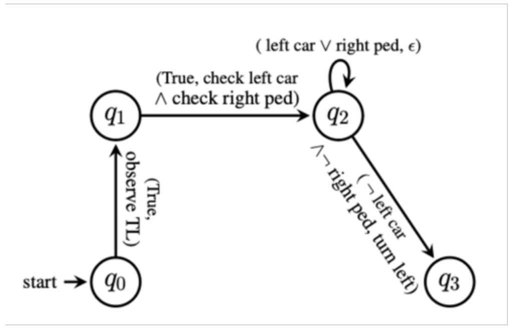
Prompt

Fine-Tuning Language Models Using Formal Methods Feedback

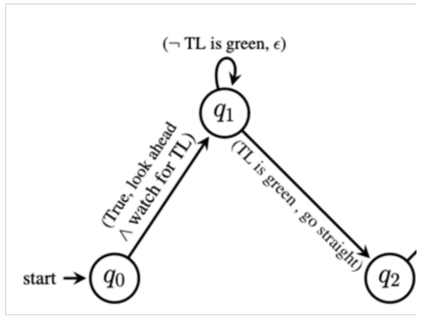
Empirical Evaluation via Simulation

Response 1

Response 2



Controller $\mathscr{C}_1$


(¬ TL is green, ε)
(True, look ahead ∧ watch for TL)
(TL is green , go straight)
start
q_0
q_1
q_2

Controller $\mathscr{C}_2$

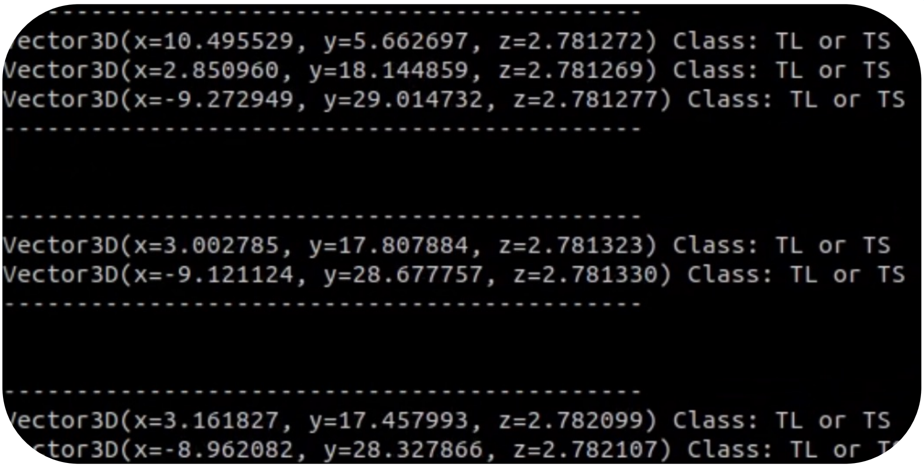
High-Fidelity Simulator

Empirically Collect Execution Info

Execution Traces

Execution Traces

Information from the Simulator



```
-----------------------------------------
ector3D(x=10.495529, y=5.662697, z=2.781272) Class: TL or TS
Vector3D(x=2.850960, y=18.144859, z=2.781269) Class: TL or TS
Vector3D(x=-9.272949, y=29.014732, z=2.781277) Class: TL or TS
-----------------------------------------


-----------------------------------------
Vector3D(x=3.002785, y=17.807884, z=2.781323) Class: TL or TS
Vector3D(x=-9.121124, y=28.677757, z=2.781330) Class: TL or TS
-----------------------------------------


-----------------------------------------
ector3D(x=3.161827, y=17.457993, z=2.782099) Class: TL or TS
ctor3D(x=-8.962082, y=28.327866, z=2.782107) Class: TL or T
```

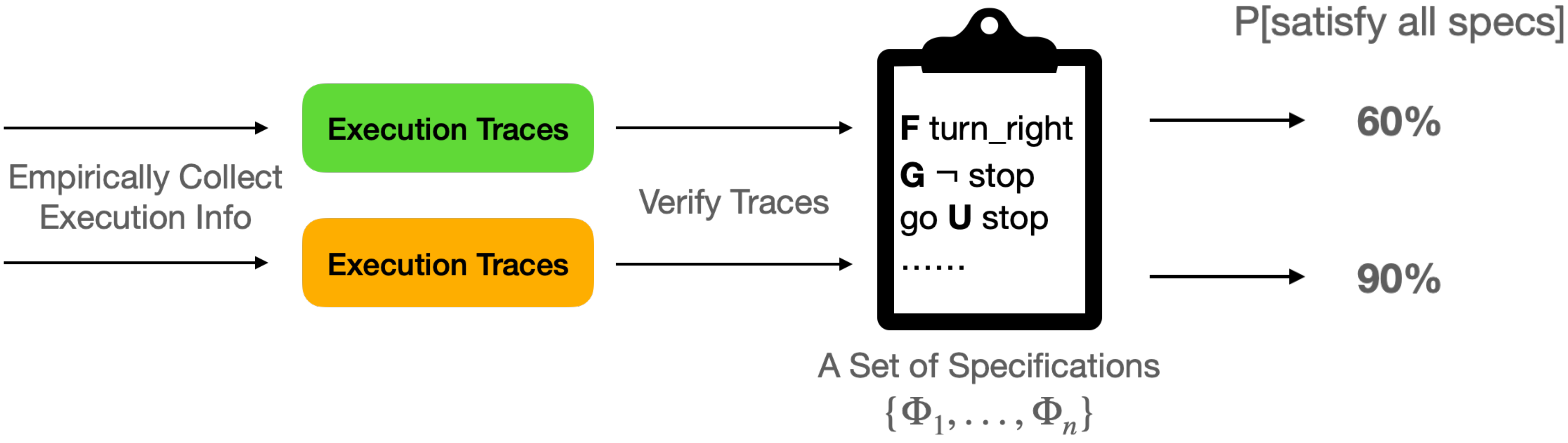
Execution Trace

$(left\_car \wedge go\_straight, pedestrian \wedge stop, \neg pedestrian \wedge go\_straight)$

Fine-Tuning Language Models Using Formal Methods Feedback

Empirical Evaluation via Simulation

(**Response 1** < **Response 2**)

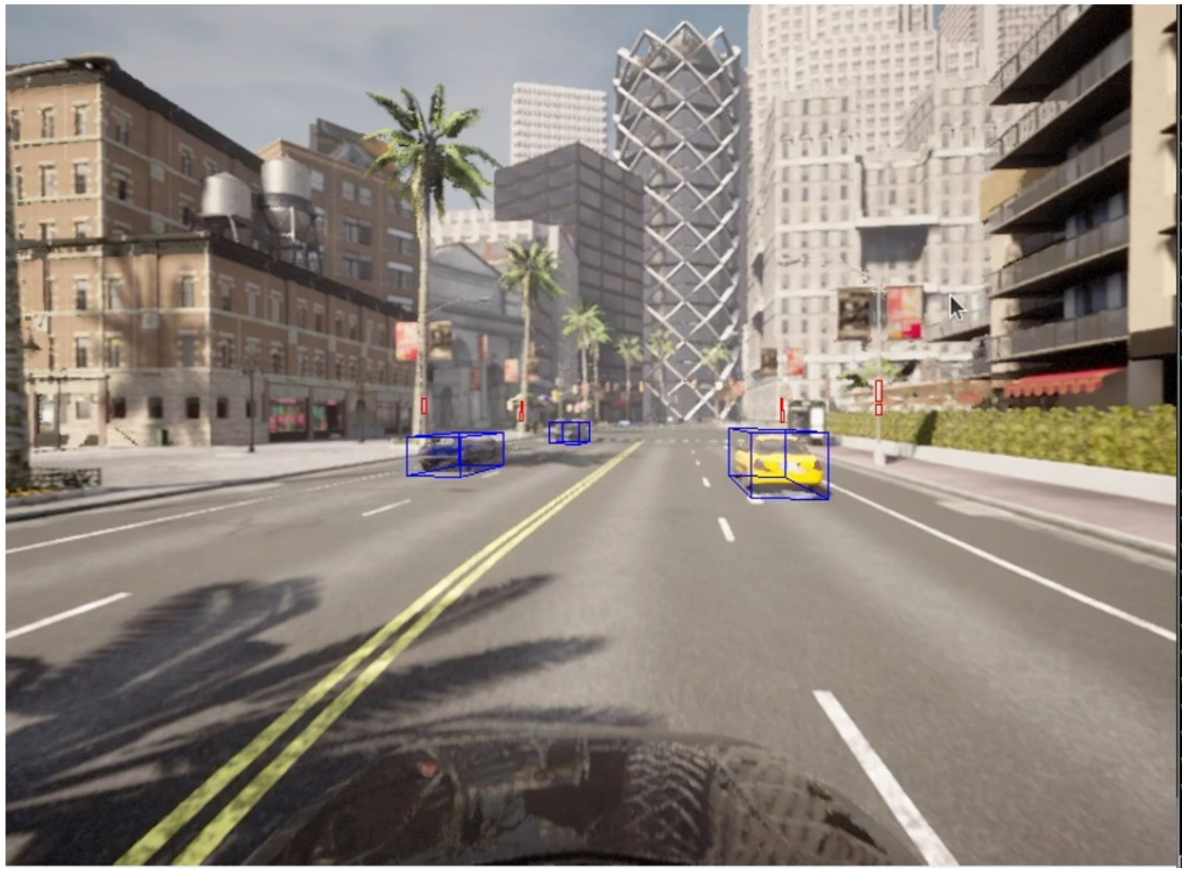


Quantitative Analysis

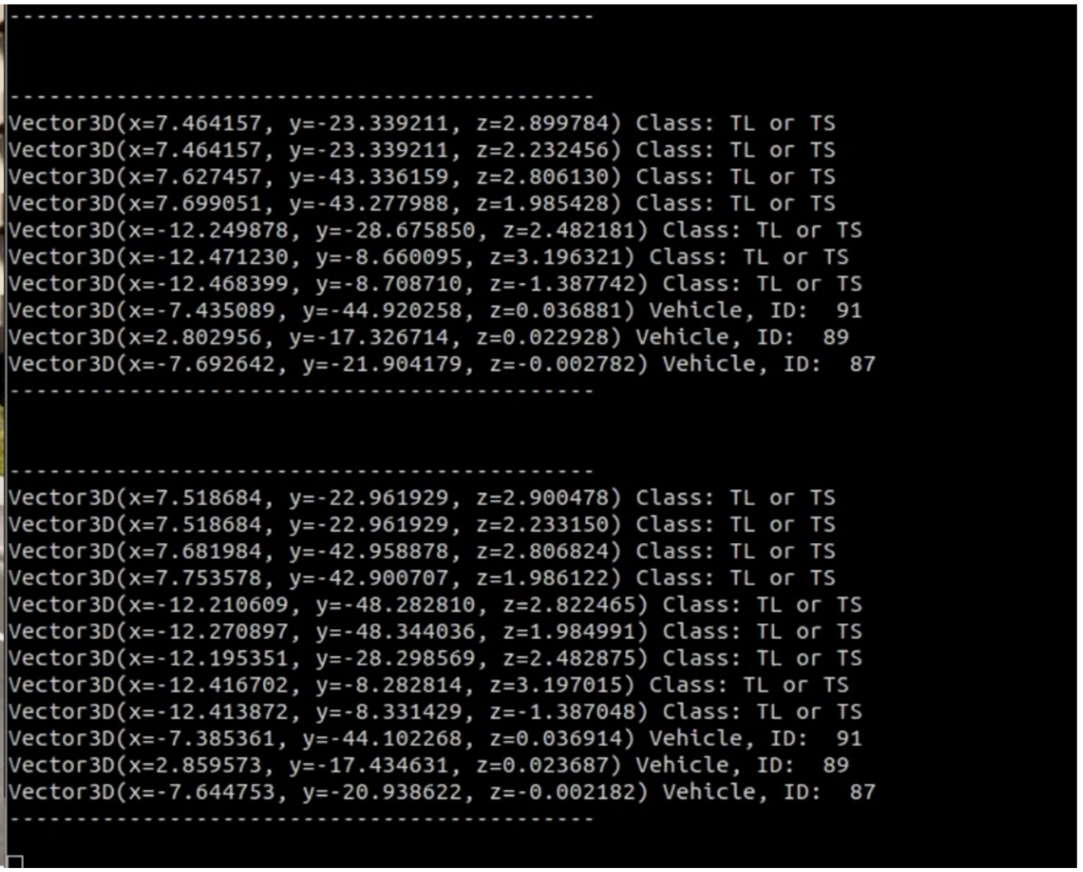
Empirical Evaluation via Simulation

Carla Simulator: Extract execution traces.

Carla Simulation Video



Object and Position Information



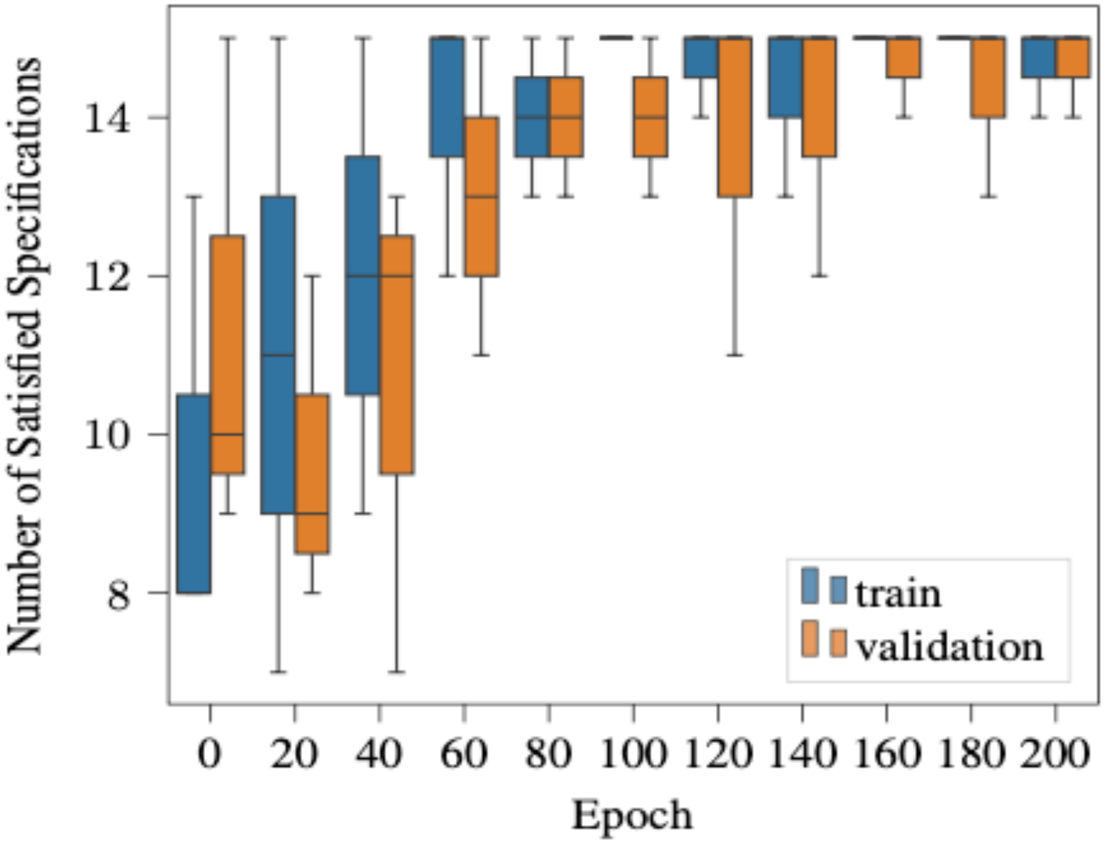
```
-------------------------------------------



-------------------------------------------
Vector3D(x=7.464157, y=-23.339211, z=2.899784) Class: TL or TS
Vector3D(x=7.464157, y=-23.339211, z=2.232456) Class: TL or TS
Vector3D(x=7.627457, y=-43.336159, z=2.806130) Class: TL or TS
Vector3D(x=7.699051, y=-43.277988, z=1.985428) Class: TL or TS
Vector3D(x=-12.249878, y=-28.675850, z=2.482181) Class: TL or TS
Vector3D(x=-12.471230, y=-8.660095, z=3.196321) Class: TL or TS
Vector3D(x=-12.468399, y=-8.708710, z=-1.387742) Class: TL or TS
Vector3D(x=-7.435089, y=-44.920258, z=0.036881) Vehicle, ID:  91
Vector3D(x=2.802956, y=-17.326714, z=0.022928) Vehicle, ID:  89
Vector3D(x=-7.692642, y=-21.904179, z=-0.002782) Vehicle, ID:  87
-------------------------------------------



-------------------------------------------
Vector3D(x=7.518684, y=-22.961929, z=2.900478) Class: TL or TS
Vector3D(x=7.518684, y=-22.961929, z=2.233150) Class: TL or TS
Vector3D(x=7.681984, y=-42.958878, z=2.806824) Class: TL or TS
Vector3D(x=7.753578, y=-42.900707, z=1.986122) Class: TL or TS
Vector3D(x=-12.210609, y=-48.282810, z=2.822465) Class: TL or TS
Vector3D(x=-12.270897, y=-48.344036, z=1.984991) Class: TL or TS
Vector3D(x=-12.195351, y=-28.298569, z=2.482875) Class: TL or TS
Vector3D(x=-12.416702, y=-8.282814, z=3.197015) Class: TL or TS
Vector3D(x=-12.413872, y=-8.331429, z=-1.387048) Class: TL or TS
Vector3D(x=-7.385361, y=-44.102268, z=0.036914) Vehicle, ID:  91
Vector3D(x=2.859573, y=-17.434631, z=0.023687) Vehicle, ID:  89
Vector3D(x=-7.644753, y=-20.938622, z=-0.002182) Vehicle, ID:  87
-------------------------------------------
```

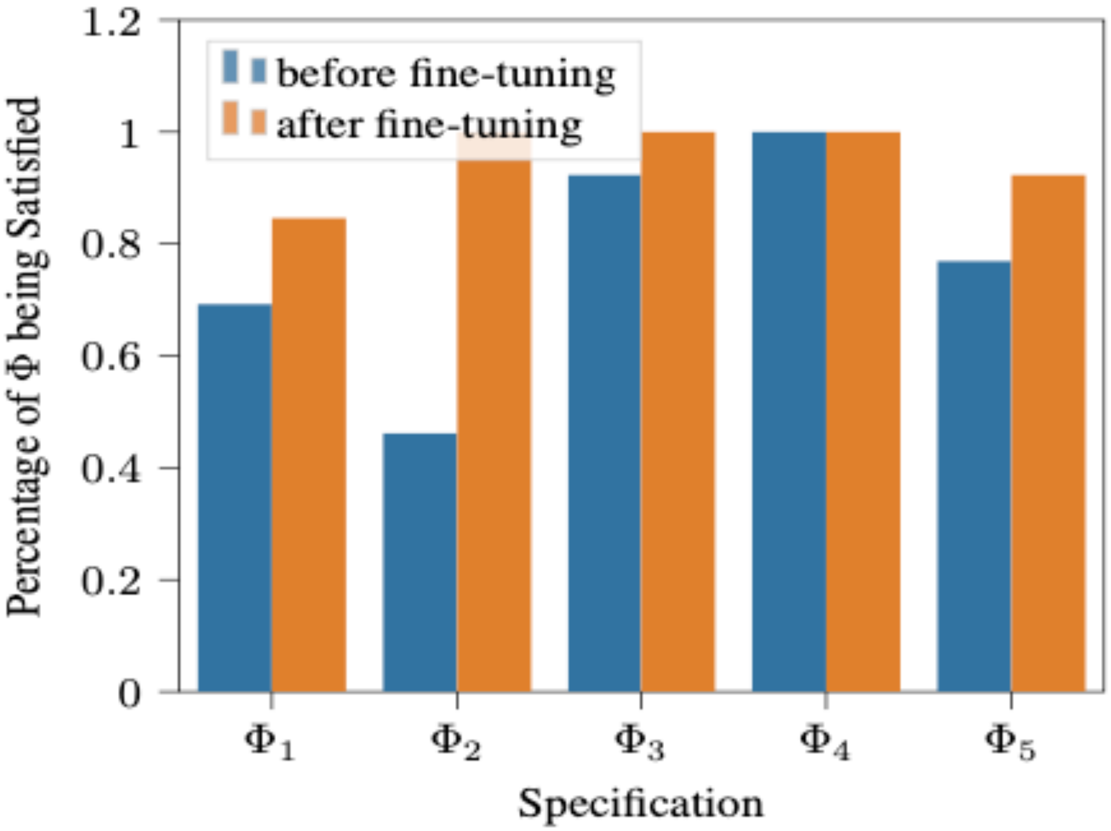
Execution Trace: (desired objects with positions, action),……

Desired objects: pedestrian, car, red/green traffic light, stop sign,……
Actions: go straight, turn left, stop, turn right,……

Quantitative Analysis

Empirical Evaluation via Simulation


Formal Verification

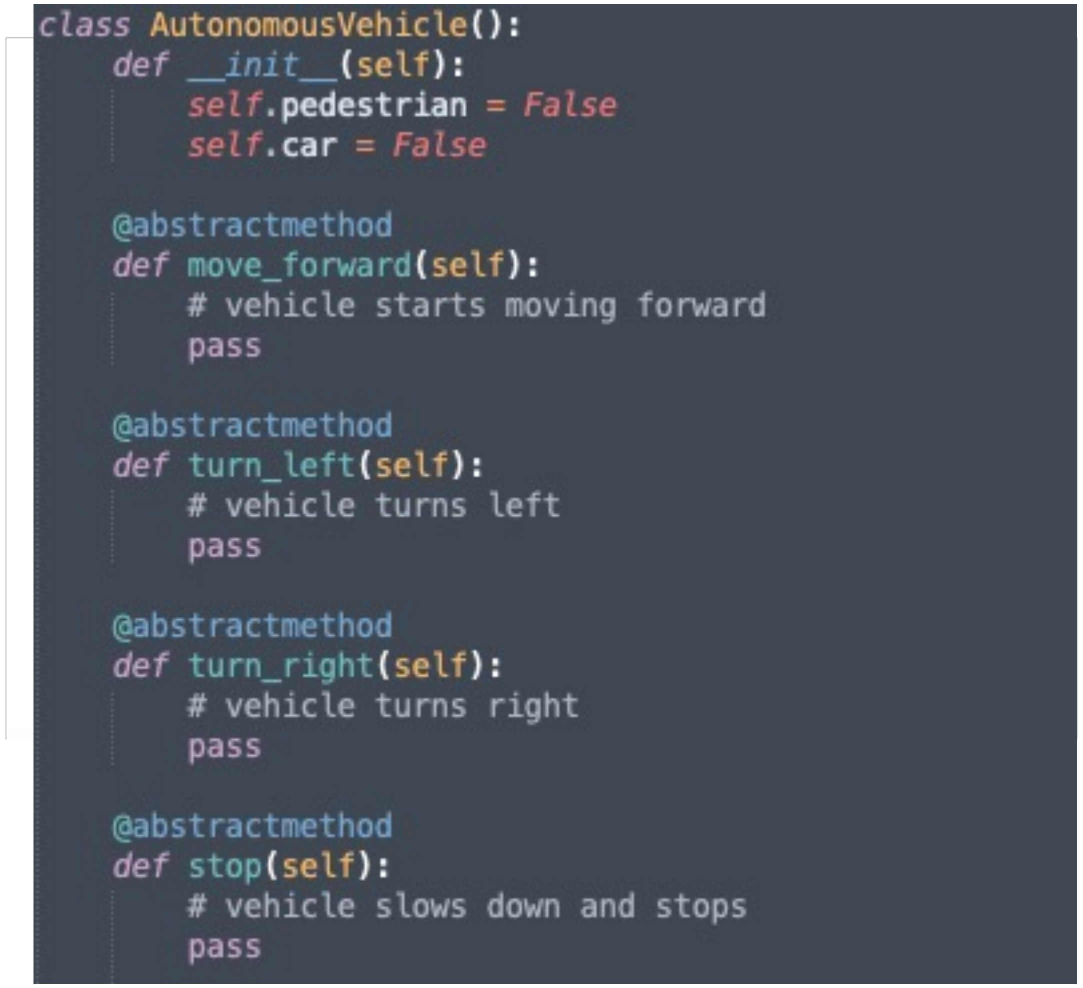

Empirical Evaluation

The results indicate that our approach can improve the language model's ability to satisfy critical requirements.

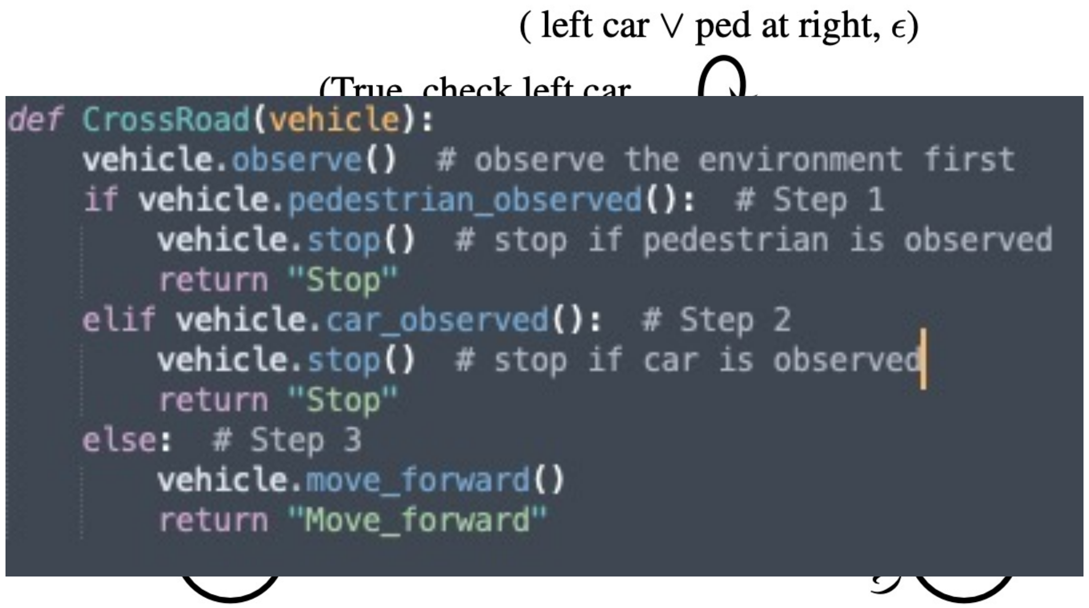
Our approach can act as a starting point to guide the design process for real-world implementations of autonomous driving systems.

Next Step: Verifiable Grounding

Ground the controller to real autonomous driving robots



```
class AutonomousVehicle():
    def __init__(self):
        self.pedestrian = False
        self.car = False

    @abstractmethod
    def move_forward(self):
        # vehicle starts moving forward
        pass

    @abstractmethod
    def turn_left(self):
        # vehicle turns left
        pass

    @abstractmethod
    def turn_right(self):
        # vehicle turns right
        pass

    @abstractmethod
    def stop(self):
        # vehicle slows down and stops
        pass
```



```
def CrossRoad(vehicle):
    vehicle.observe()  # observe the environment first
    if vehicle.pedestrian_observed():  # Step 1
        vehicle.stop()  # stop if pedestrian is observed
        return "Stop"
    elif vehicle.car_observed():  # Step 2
        vehicle.stop()  # stop if car is observed
        return "Stop"
    else:  # Step 3
        vehicle.move_forward()
        return "Move_forward"
```

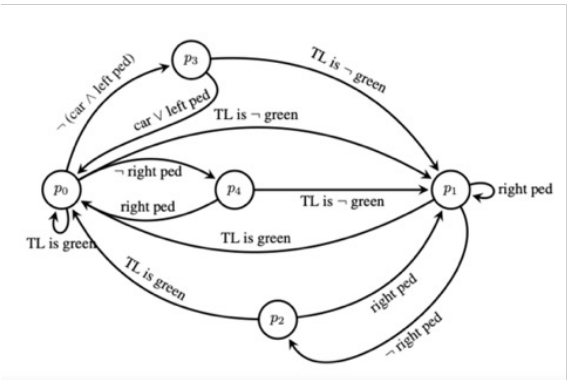
$\square(\neg \text{green traffic light} \rightarrow \neg \text{go straight})$, ✓

$\square(\text{stop sign} \rightarrow \Diamond\, \text{stop})$, ✓

$\square\, \neg \text{turn right} \vee \neg(\text{car from left} \vee \text{pedestrian at right})$, ✓

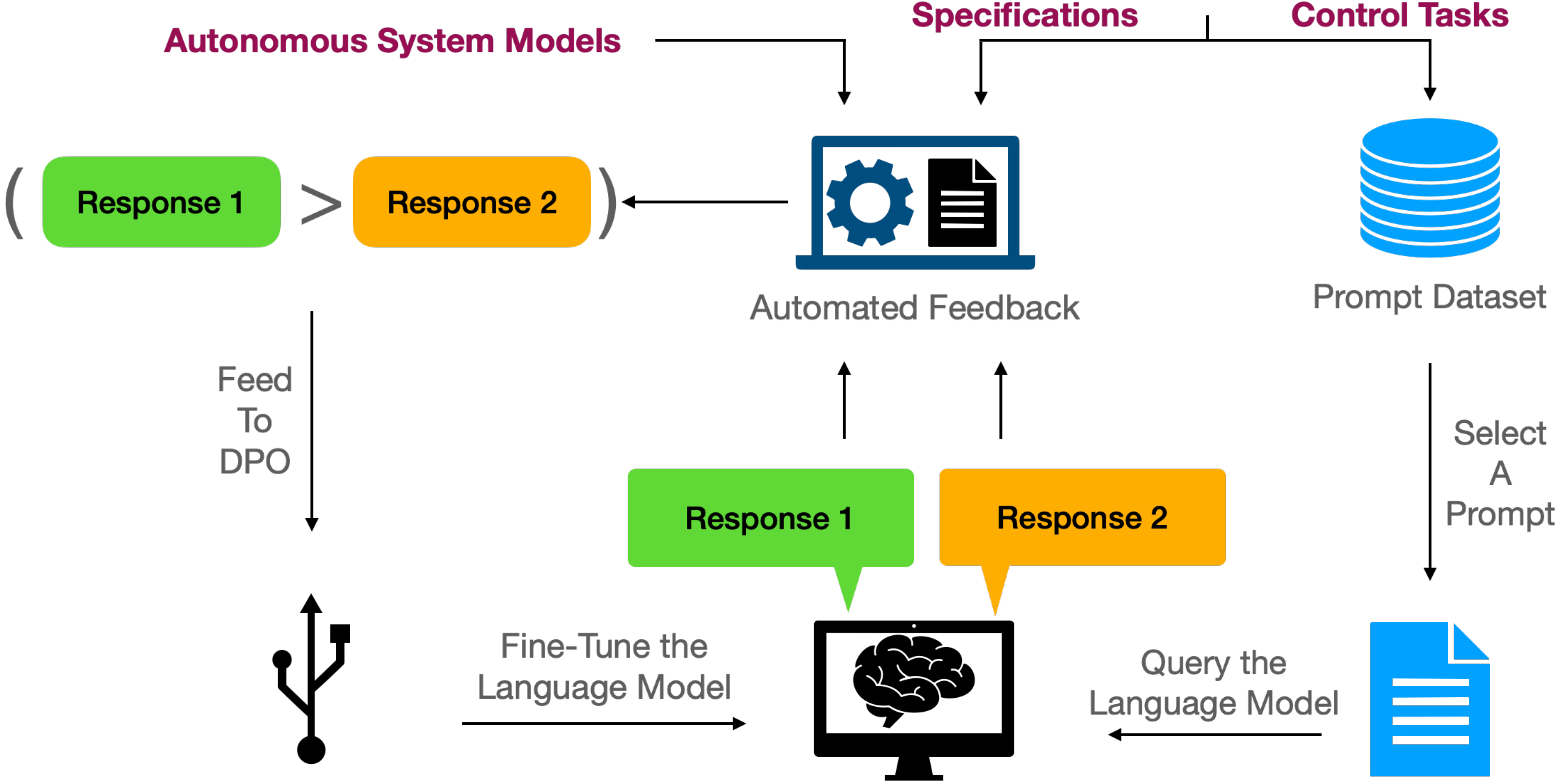
Q&A

Abstract Model



High-Fidelity Simulator

User



Autonomous System Models

Specifications

Control Tasks

(**Response 1** > **Response 2**)

Automated Feedback

Prompt Dataset

Feed
To
DPO

Select
A
Prompt

Response 1

Response 2

Fine-Tune the
Language Model

Query the
Language Model

Direct Preference Optimization

Language Model

Prompt