Safe Reinforcement Learning via Formal Methods

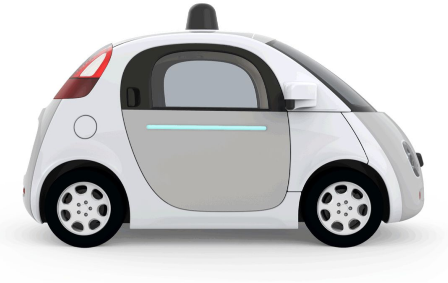
Nathan Fulton and André Platzer

Carnegie Mellon University

Safety-Critical Systems

"How can we provide people with cyber-physical systems they can bet their lives on?" - Jeannette Wing

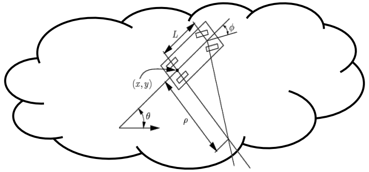
Autonomous Safety-Critical Systems



How can we provide people with **autonomous** cyber-physical systems they can bet their lives on?

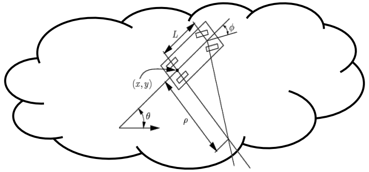
Model-Based Verification

Reinforcement Learning



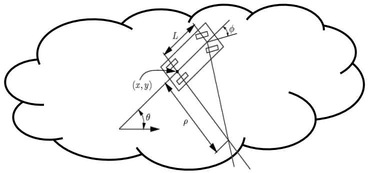
φ

Model-Based Verification Reinforcement Learning

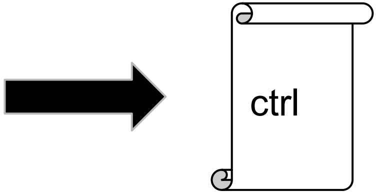


pos < stopSign

Model-Based Verification



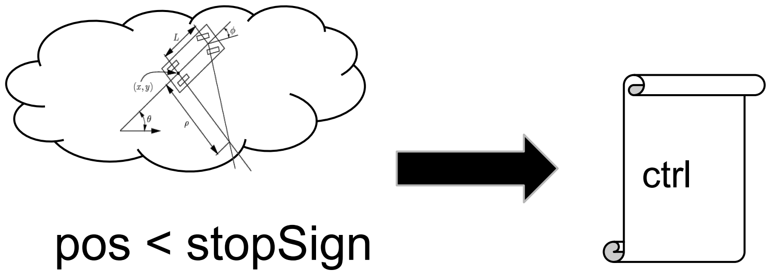
pos < stopSign



Reinforcement Learning

Model-Based Verification

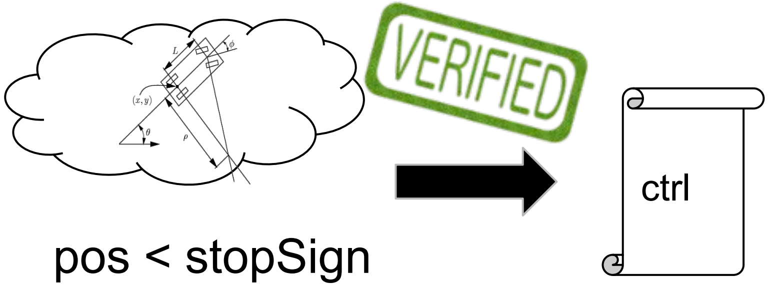
Reinforcement Learning



Approach: prove that control software achieves a specification with respect to a model of the physical system.

Model-Based Verification

Reinforcement Learning



Approach: prove that control software achieves a specification with respect to a model of the physical system.

Model-Based Verification

Reinforcement Learning

φ

Benefits:

- Strong safety guarantees
- Automated analysis

Model-Based Verification

Reinforcement Learning

φ

Benefits:

- Strong safety guarantees
- Automated analysis

Drawbacks:

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”

Model-Based Verification

Reinforcement Learning

φ

Benefits:

- Strong safety guarantees
- Automated analysis

Drawbacks:

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model

Model-Based Verification

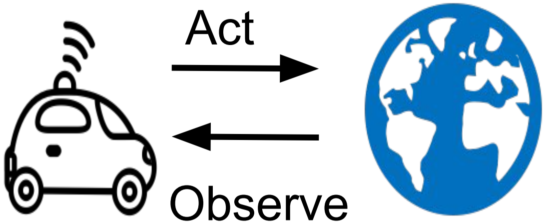
Benefits:

- Strong safety guarantees
- Automated analysis

Drawbacks:

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model.

Reinforcement Learning



Model-Based Verification

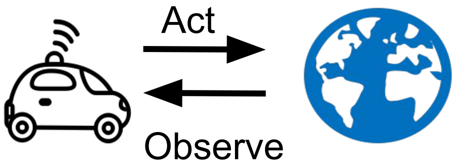
Benefits:

- Strong safety guarantees
- Automated analysis

Drawbacks:

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model.

Reinforcement Learning



Benefits:

- No need for complete model
- Optimal (effective) policies

Model-Based Verification

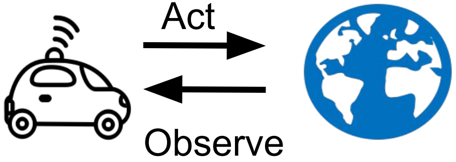
Benefits:

- Strong safety guarantees
- Automated analysis

Drawbacks:

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model.

Reinforcement Learning



Benefits:

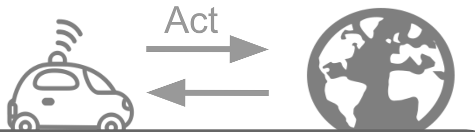
- No need for complete model
- Optimal (effective) policies

Drawbacks:

- No strong safety guarantees
- Proofs are obtained and checked by hand
- Formal proofs = decades-long proof development

Model-Based Verification

Reinforcement Learning



Bene

- S ... del
- A ... s

Goal: Provably correct reinforcement learning

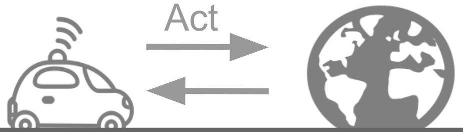
Draw

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model

- No strong safety guarantees
- Proofs are obtained and checked by hand
- Formal proofs = decades-long proof development

Model-Based Verification

Reinforcement Learning



Goal: Provably correct reinforcement learning

1. **Learn Safely**
2. **Learn a Safe Policy**
3. **Justify claims of safety**

Bene

- S
- A

del

s

Draw

- Control policies are typically non-deterministic: answers “what is safe”, not “what is useful”
- Assumes accurate model

- No strong safety guarantees
- Proofs are obtained and checked by hand
- Formal proofs = decades-long proof development

Model-Based Verification

Accurate, analyzable models often exist!

```
{
   {?safeAccel;accel ∪ brake ∪ ?safeTurn; turn};
   {pos’ = vel, vel’ = acc}
}*
```

Model-Based Verification

Accurate, analyzable models often exist!

```
{
    {?safeAccel;accel ∪ brake ∪ ?safeTurn; turn};
    {pos’ = vel, vel’ = acc}
}*
```

discrete control

Continuous motion

Model-Based Verification

Accurate, analyzable models often exist!

```
{
   {?safeAccel;accel ∪ brake ∪ ?safeTurn; turn};
   {pos' = vel, vel' = acc}
}*
```

Continuous motion

discrete, ***non-deterministic*** control

Model-Based Verification

Accurate, **analyzable** models often exist!

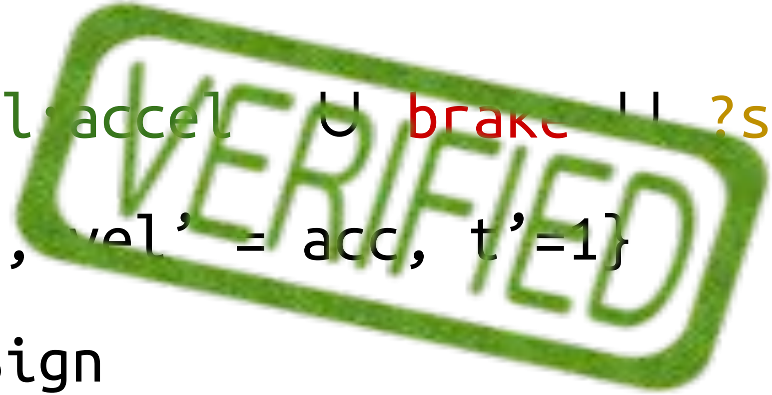
```
init → [{
   { ?safeAccel;accel  ∪ brake ∪ ?safeTurn; turn};
   {pos’ = vel, vel’ = acc, t’=1}
}*]pos < stopSign
```

Model-Based Verification

Accurate, **analyzable** models often exist!

formal verification gives strong safety guarantees

```
init → [{
    { ?safeAccel;accel ∪ brake || ?safeTurn; turn};
    {pos’ = vel, vel’ = acc, t’=1}
}*]pos < stopSign
```



Model-Based Verification

Accurate, **analyzable** models often exist!

formal verification gives strong safety guarantees

=

- **Computer-checked proofs of safety specification.**

Model-Based Verification

Accurate, **analyzable** models often exist!

formal verification gives strong safety guarantees

=

- **Computer-checked proofs of safety specification**
- **Formal proofs mapping model to runtime monitors**

Model-Based Verification Isn't Enough

Perfect, analyzable models don't exist!

Model-Based Verification Isn't Enough

Perfect, analyzable models don't exist!

How to implement?

```
{
  { ?safeAccel;accel ∪ brake ∪ ?safeTurn; turn};
  {pos' = vel, vel' = acc}
}*
```

Only accurate sometimes

Model-Based Verification Isn't Enough

Perfect, analyzable models don't exist!

How to implement?

```
{
  { ?safeAccel;accel  U  brake  U  ?safeTurn; turn};
  {dx'=w*y, dy'=-w*x, ...}
}*
```

Only accurate sometimes

Our Contribution

Justified Speculative Control is an approach toward provably safe reinforcement learning that:

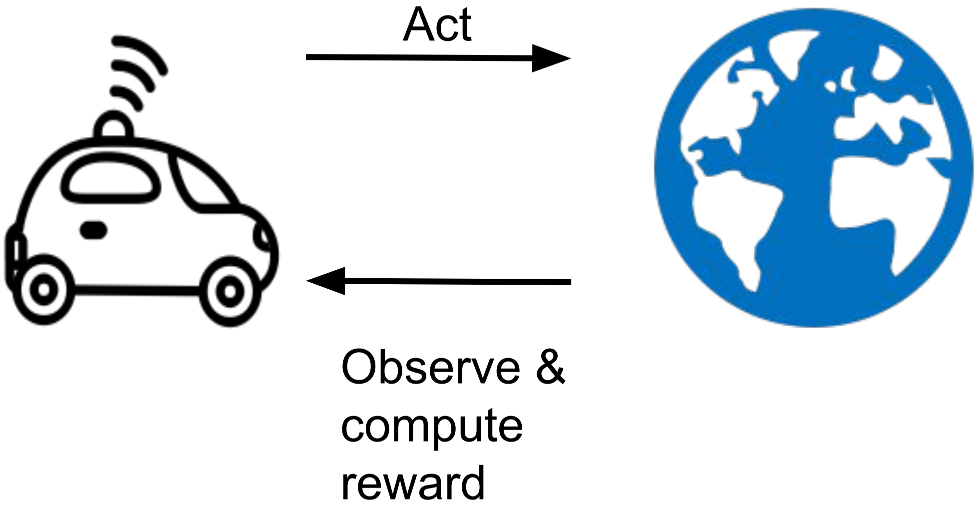
1. learns to resolve non-determinism without sacrificing formal safety results

Our Contribution

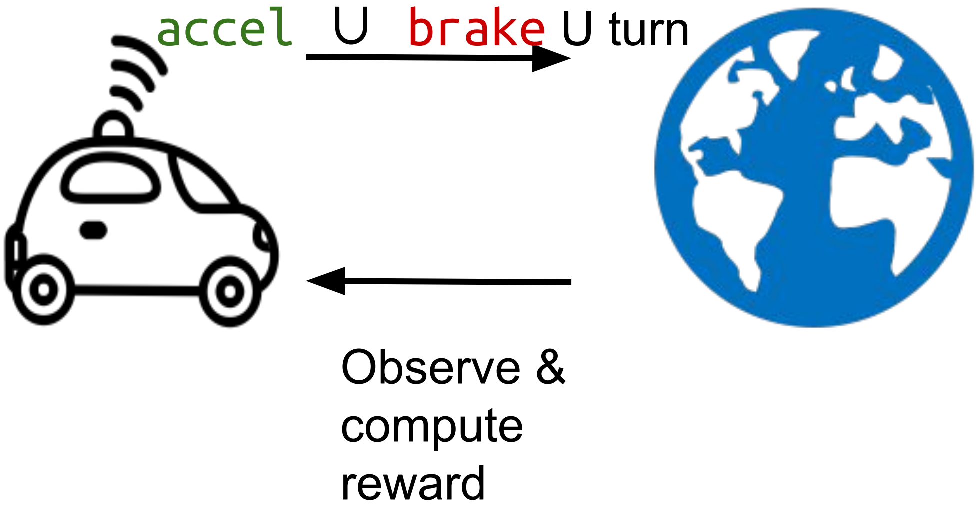
Justified Speculative Control is an approach toward provably safe reinforcement learning that:

1. learns to resolve non-determinism without sacrificing formal safety results
2. allows and directs speculation whenever model mismatches occur

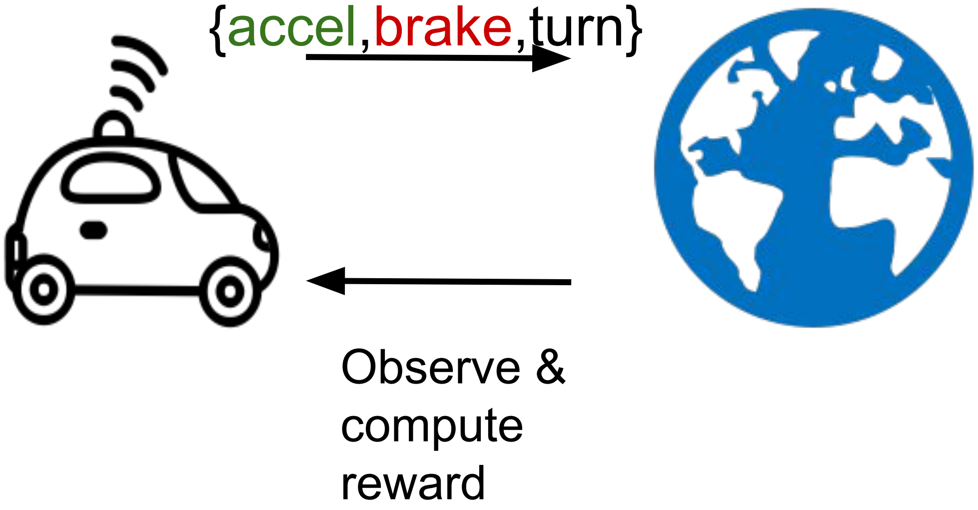
Learning to Resolve Non-determinism



Learning to Resolve Non-determinism



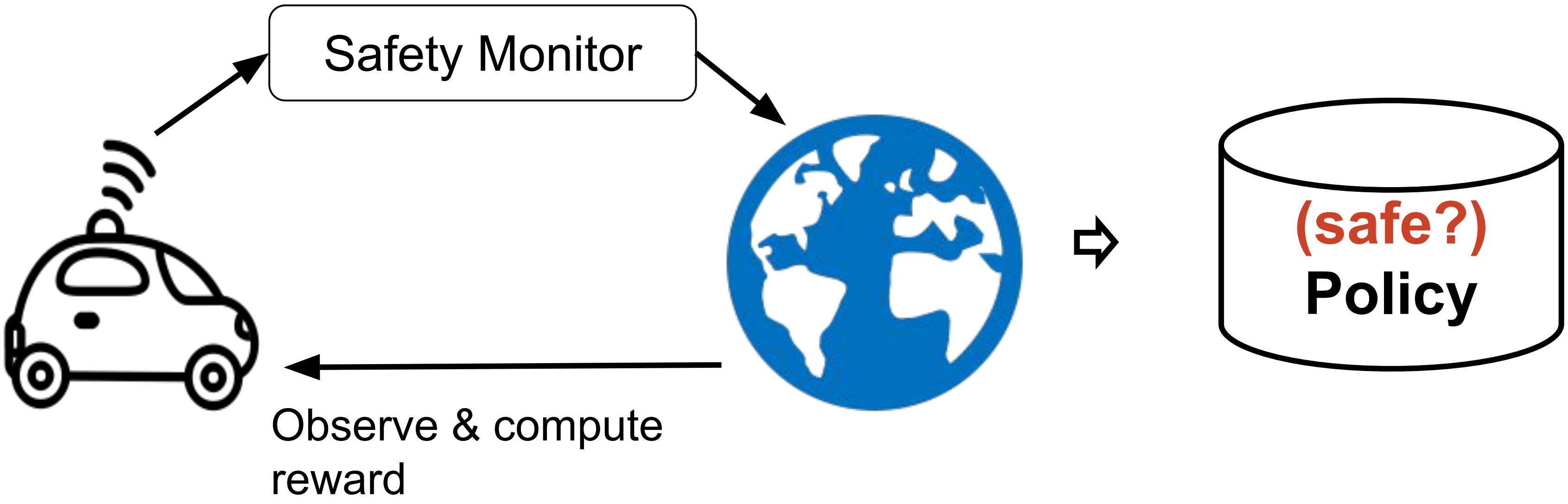
Learning to Resolve Non-determinism



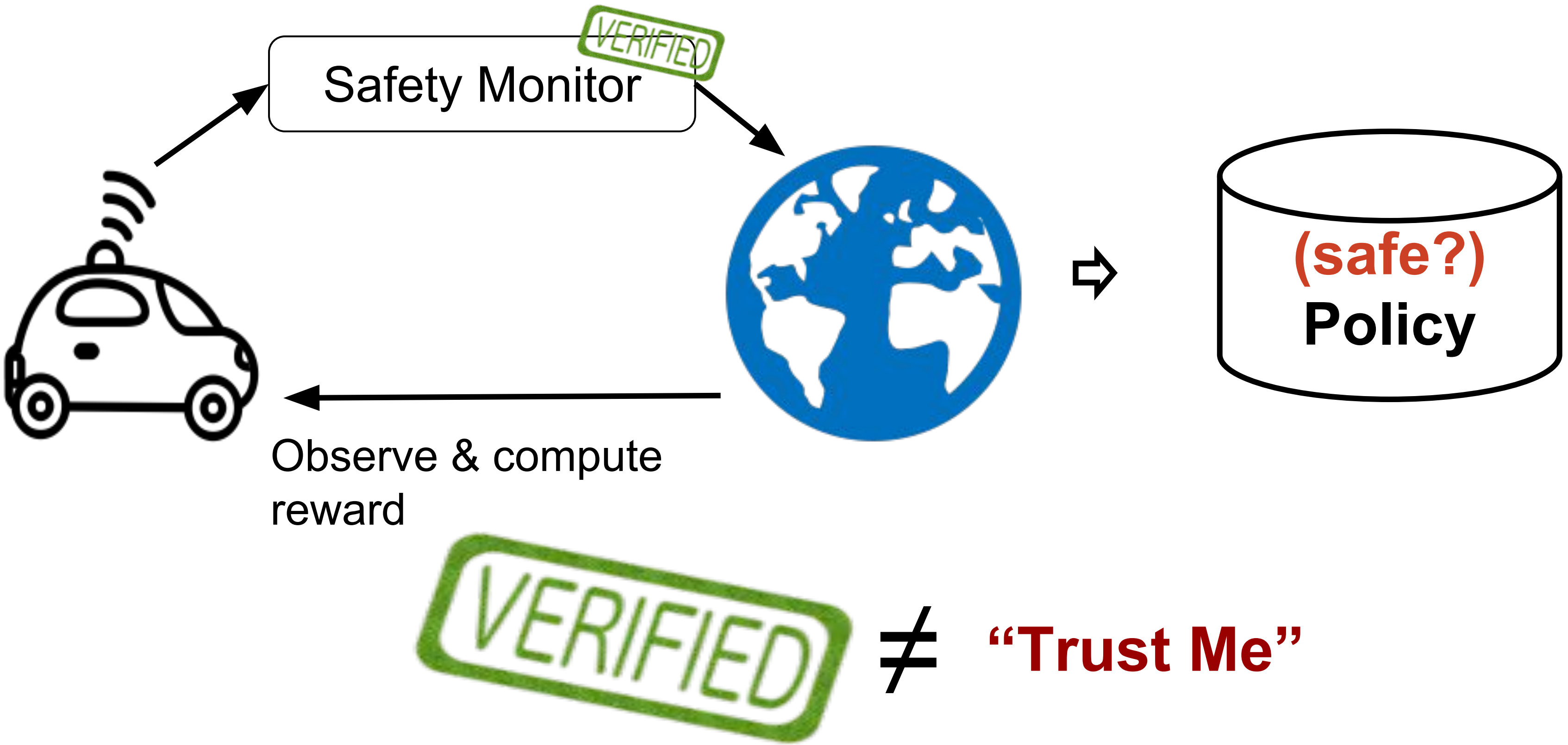
Learning to Resolve Non-determinism

Learning to Resolve Non-determinism

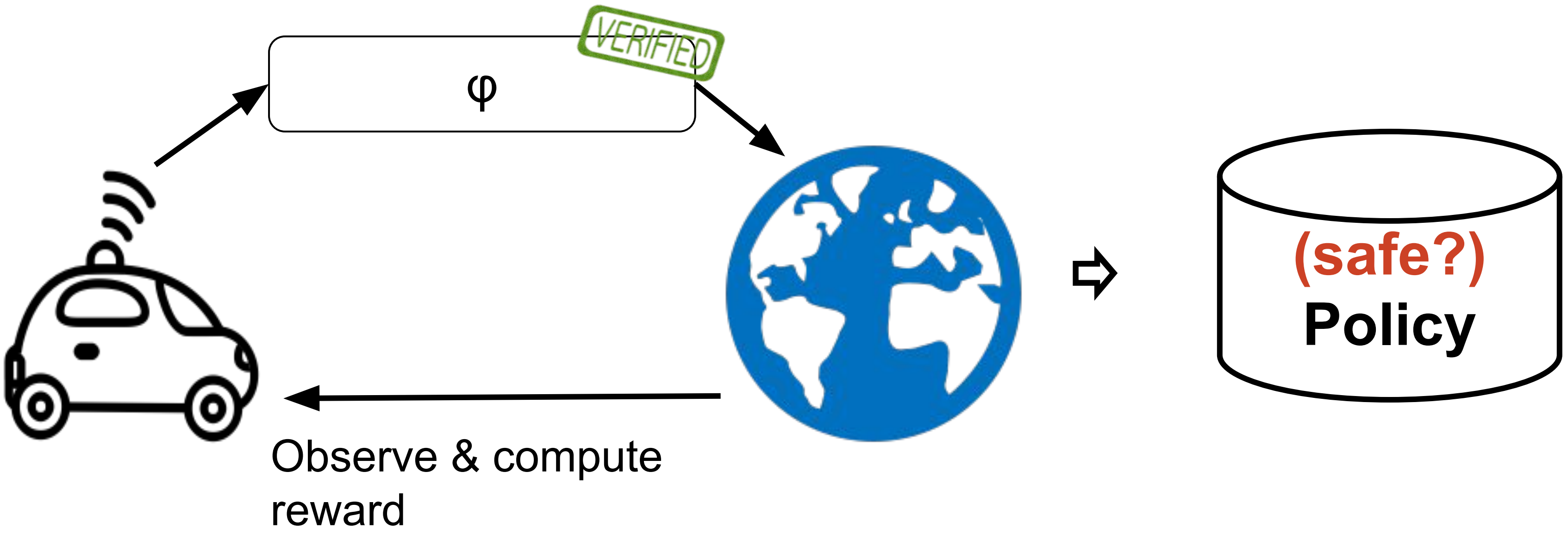
Learning to **Safely** Resolve Non-determinism



Learning to **Safely** Resolve Non-determinism



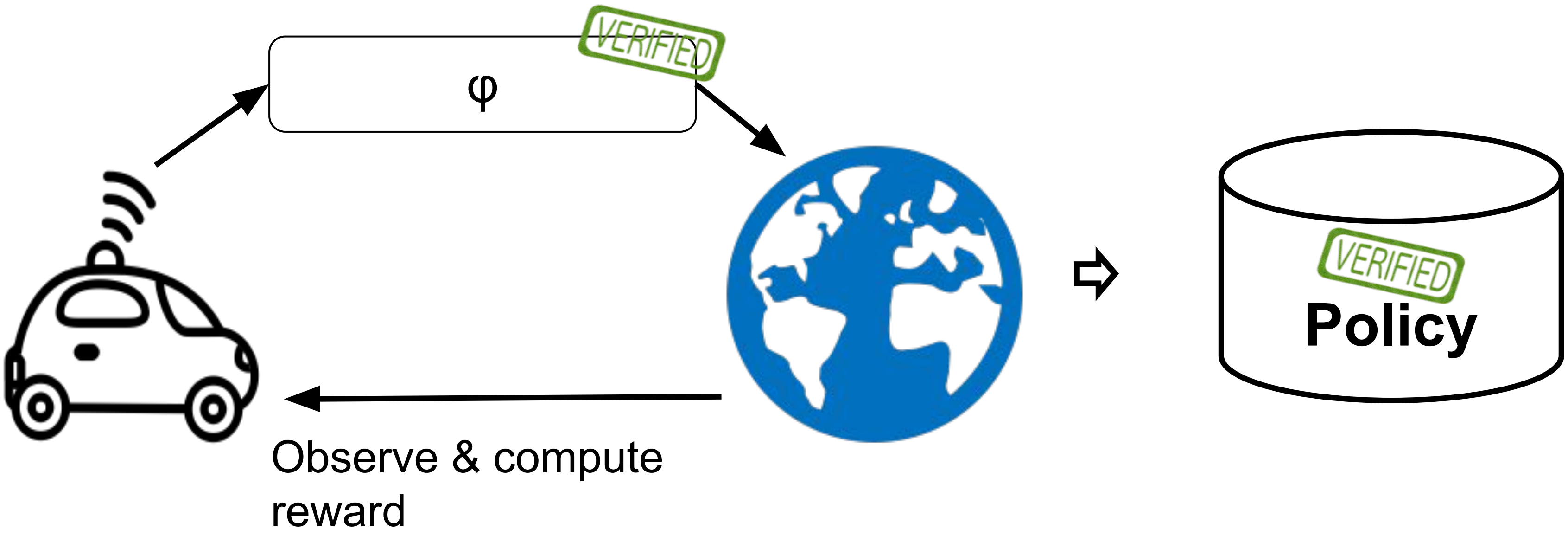
Learning to **Safely** Resolve Non-determinism



Use a theorem prover to prove:

$$(init \rightarrow [\{\{accel \cup brake\}; ODEs\}^*](safe)) \leftrightarrow \varphi$$

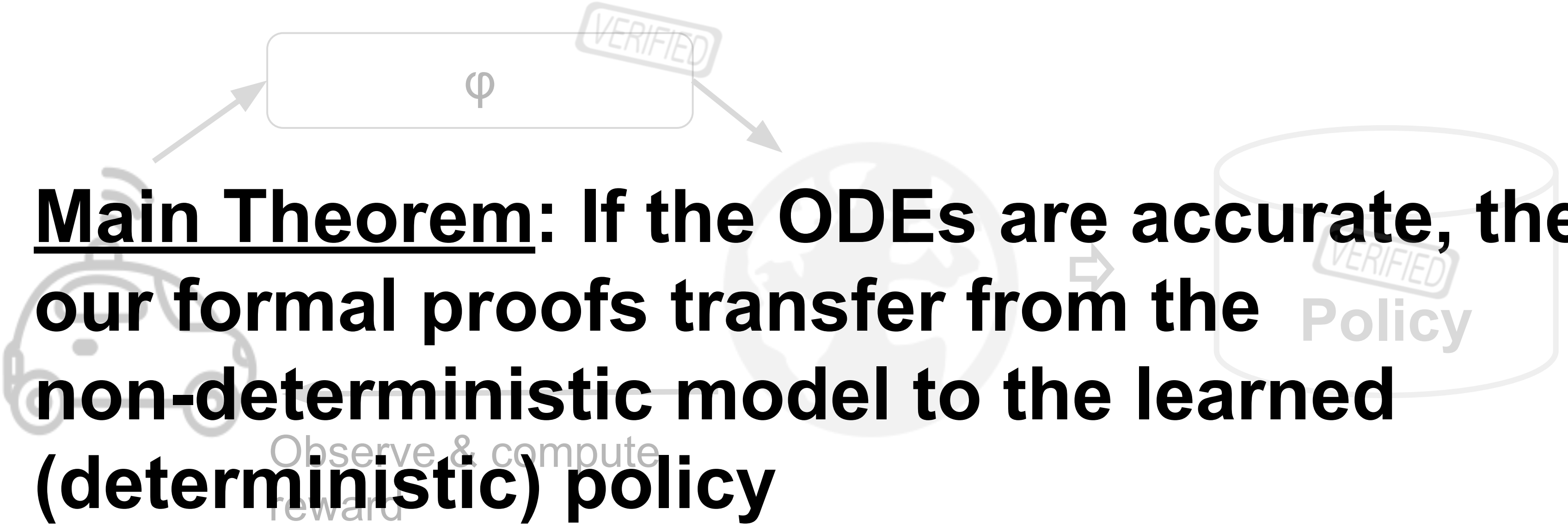
Learning to **Safely** Resolve Non-determinism



Use a theorem prover to prove:

$$(\mathtt{init} \rightarrow [\{\{\mathtt{accel} \cup \mathtt{brake}\};\mathtt{ODEs}\}^*](\mathtt{safe})) \leftrightarrow \varphi$$

Learning to **Safely** Resolve Non-determinism



Main Theorem: If the ODEs are accurate, then our formal proofs transfer from the non-deterministic model to the learned (deterministic) policy

Use a theorem prover to prove:

$(\text{init} \rightarrow [\{\{\text{accel} \cup \text{brake}\};\text{ODEs}\}^*](\text{safe})) \leftrightarrow \varphi$

Learning to **Safely** Resolve Non-determinism

φ

VERIFIED

<u>Main Theorem</u>: If the ODEs are accurate, then our formal proofs transfer from the non-deterministic model to the learned (deterministic) policy via the model monitor.

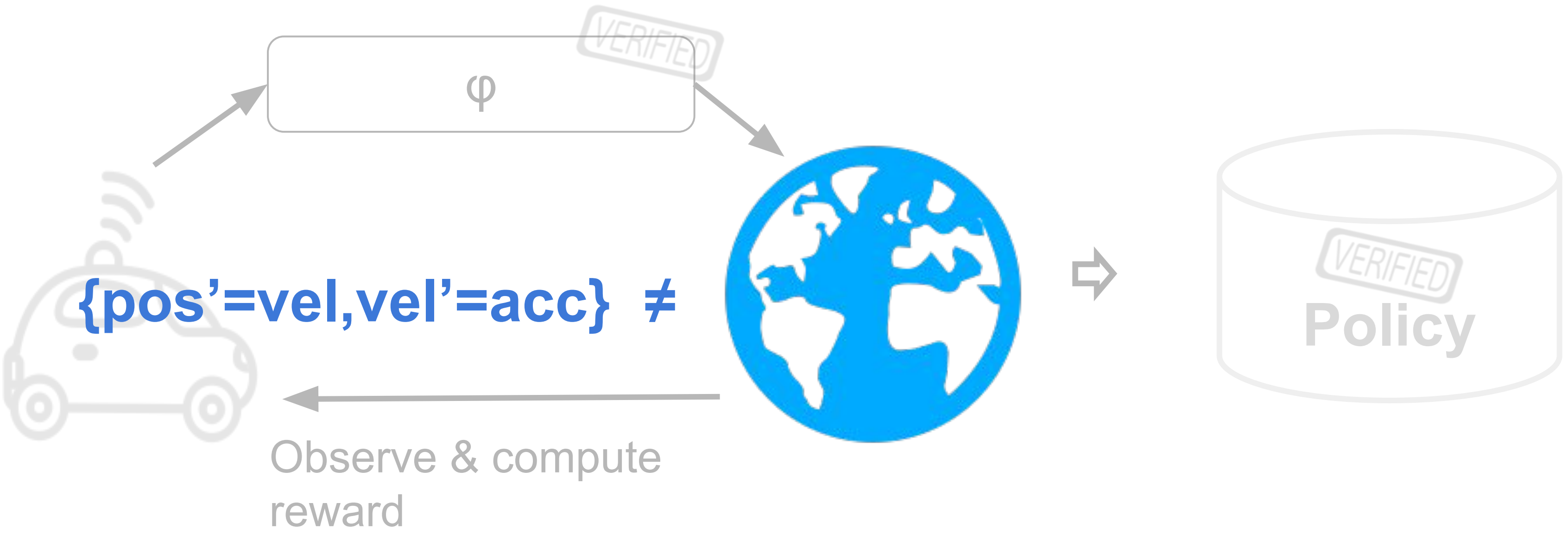
Policy

Observe & compute reward

Use a theorem prover to prove:

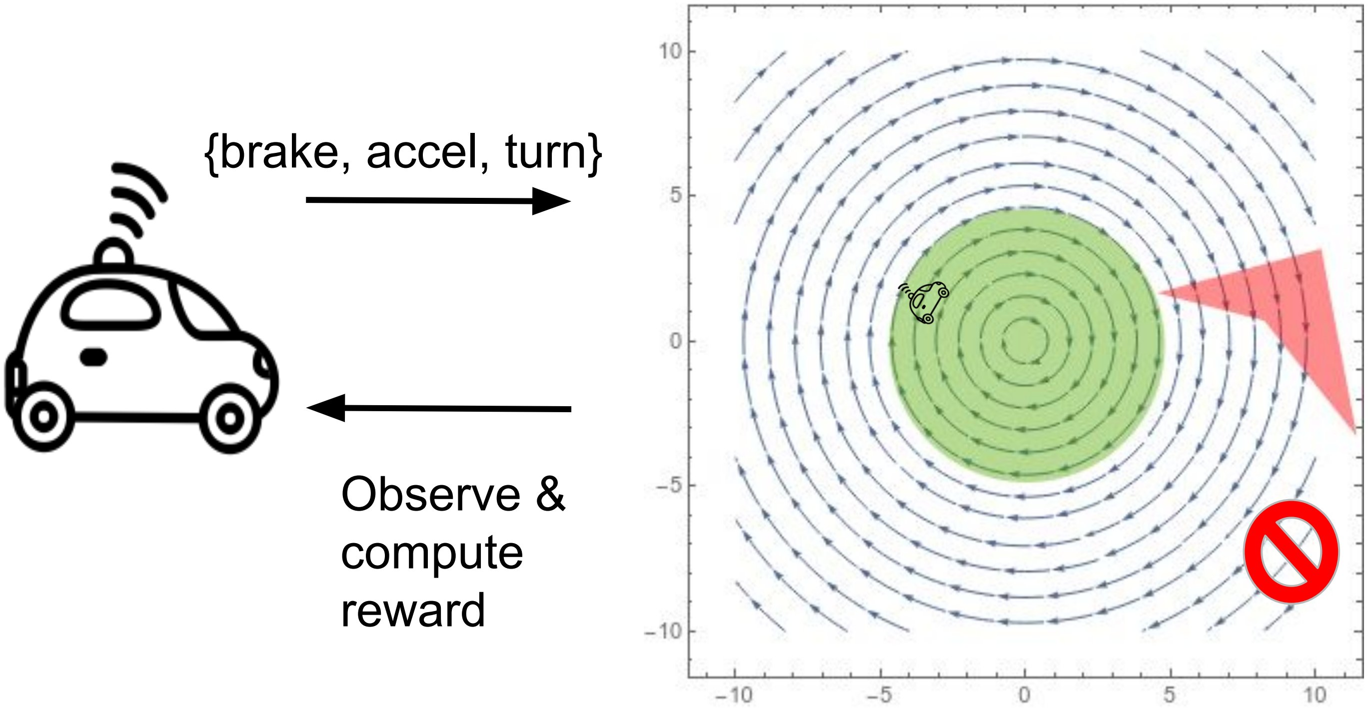
$(\mathrm{init} \rightarrow [\{\{\mathrm{accel} \cup \mathrm{brake}\};\mathrm{ODEs}\}^*](\mathrm{safe})) \leftrightarrow \varphi$

What about the physical model?



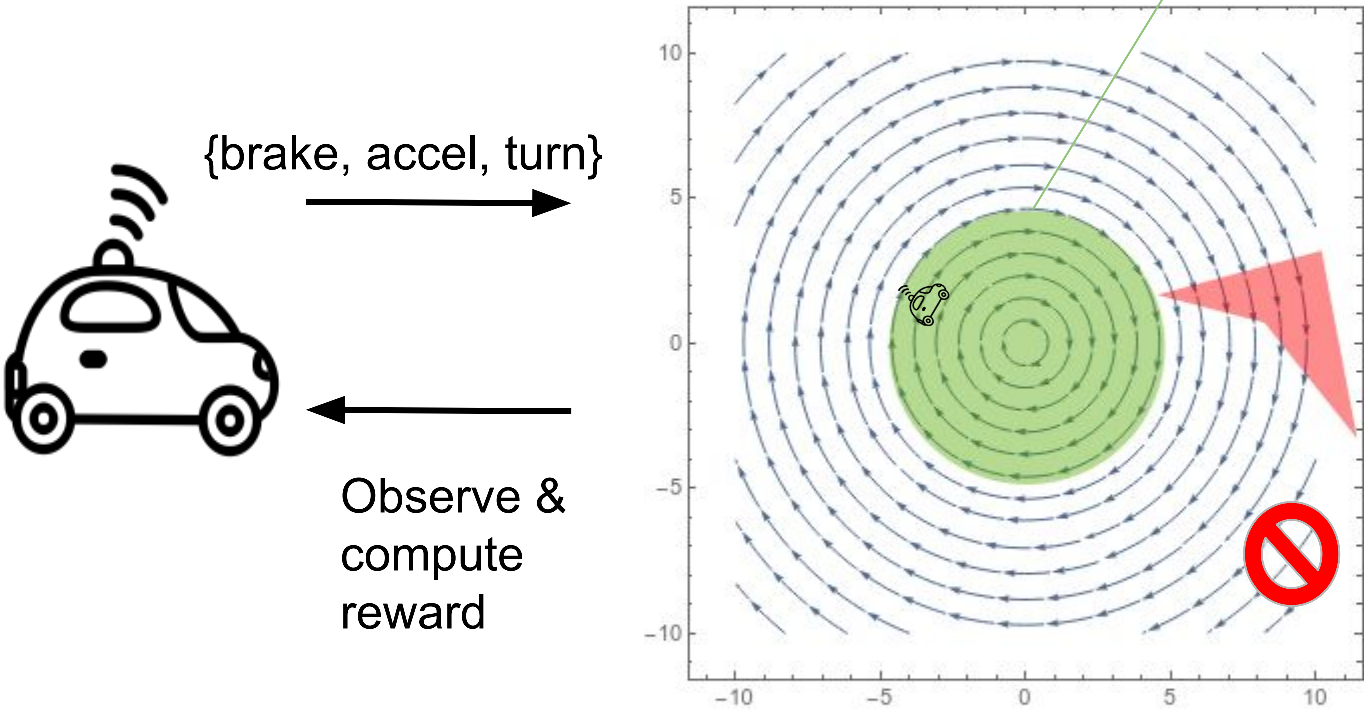
Use a theorem prover to prove: $(\mathtt{init} \rightarrow [\{\{\mathtt{accel} \cup \mathtt{brake}\};\mathtt{ODEs}\}^*](\mathtt{safe})) \leftrightarrow \varphi$

What About the Physical Model?



What About the Physical Model?

Model is accurate.

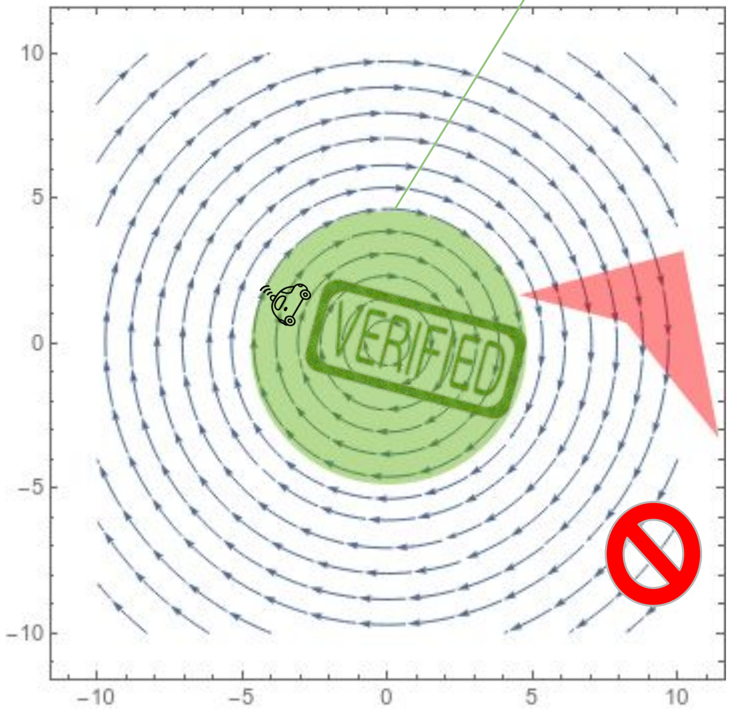


What About the Physical Model?

Model is accurate.

{brake, accel, turn}

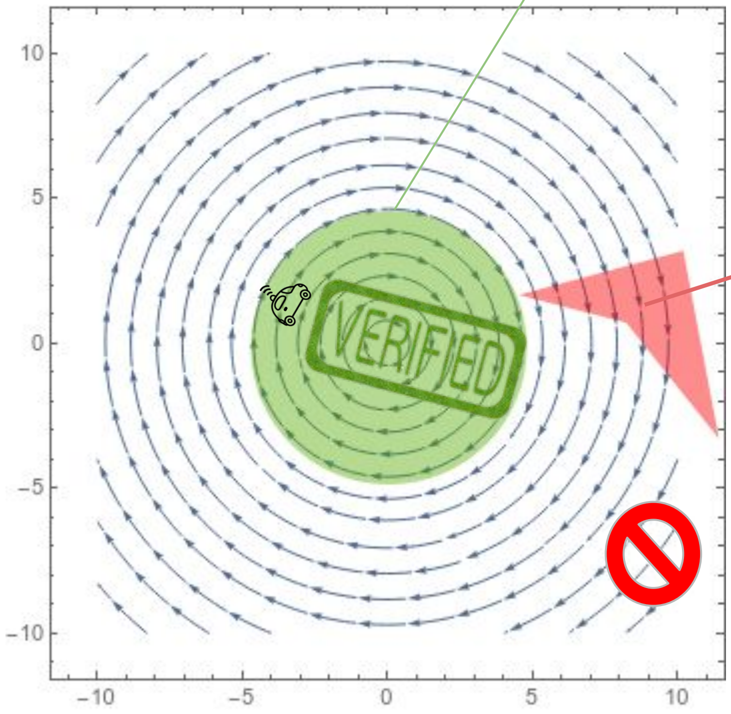
Observe & compute reward



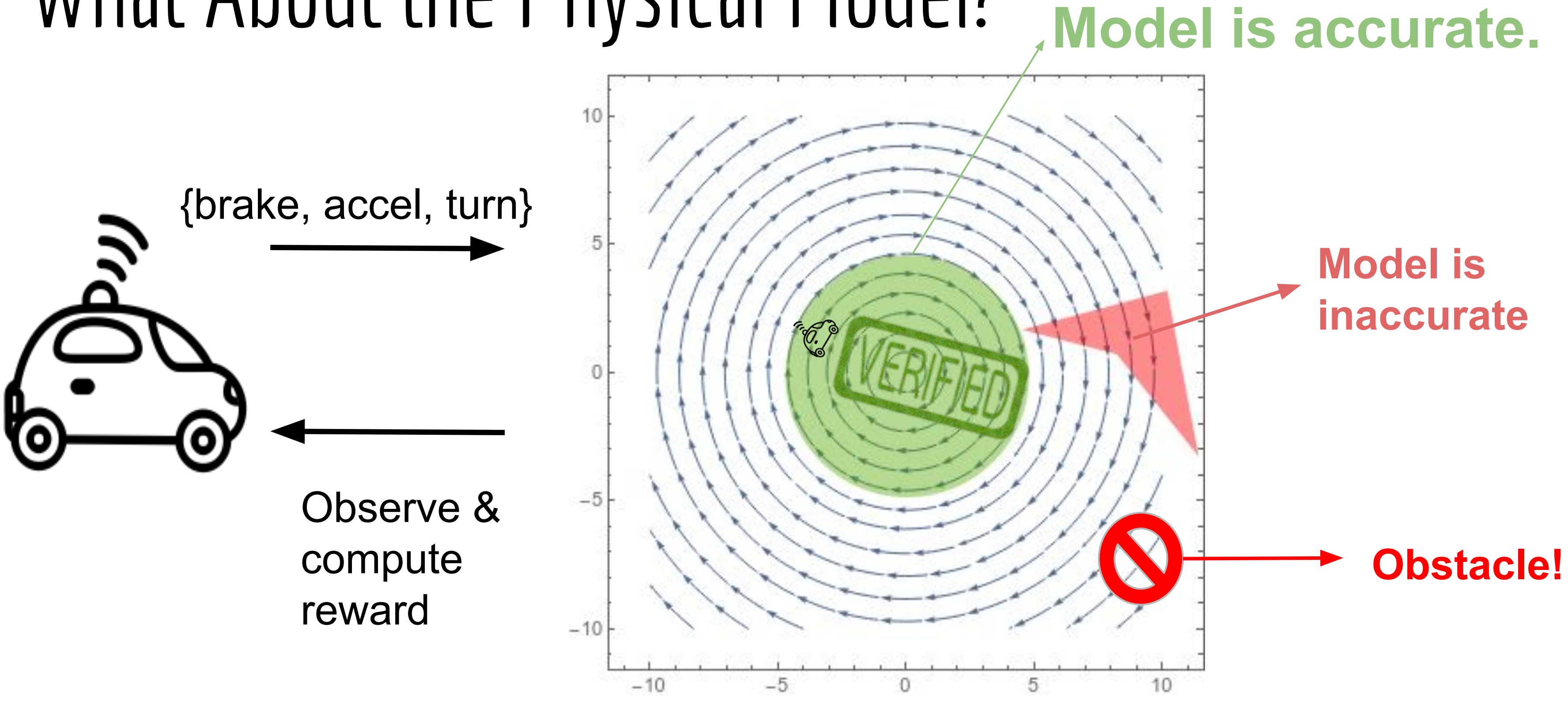
What About the Physical Model?

{brake, accel, turn}

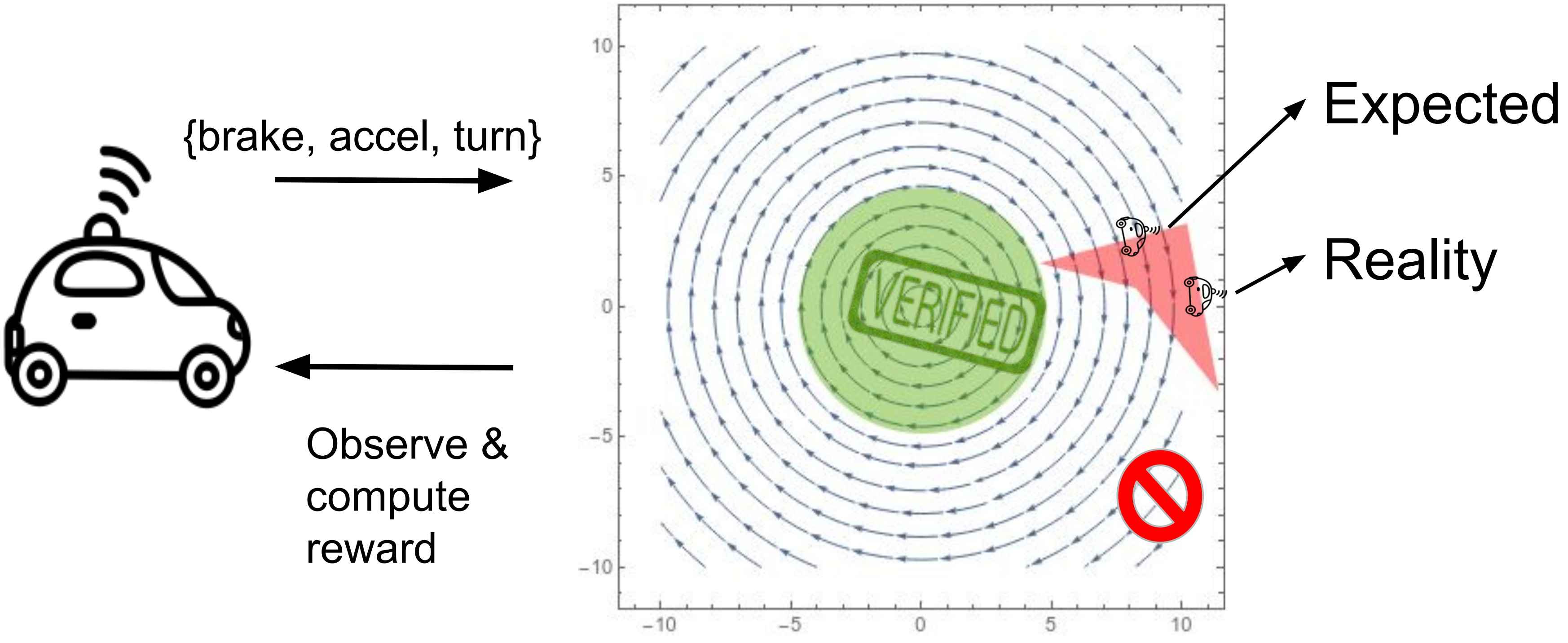
Observe & compute reward



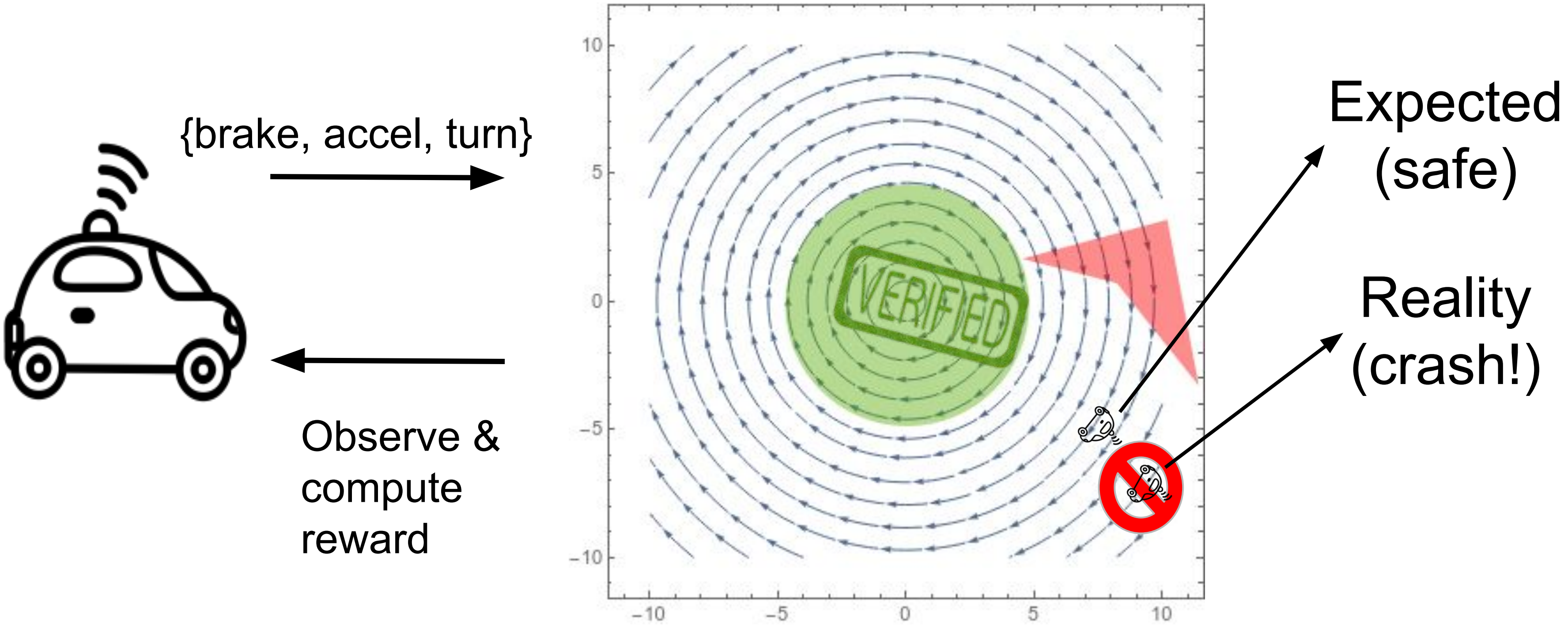
What About the Physical Model?



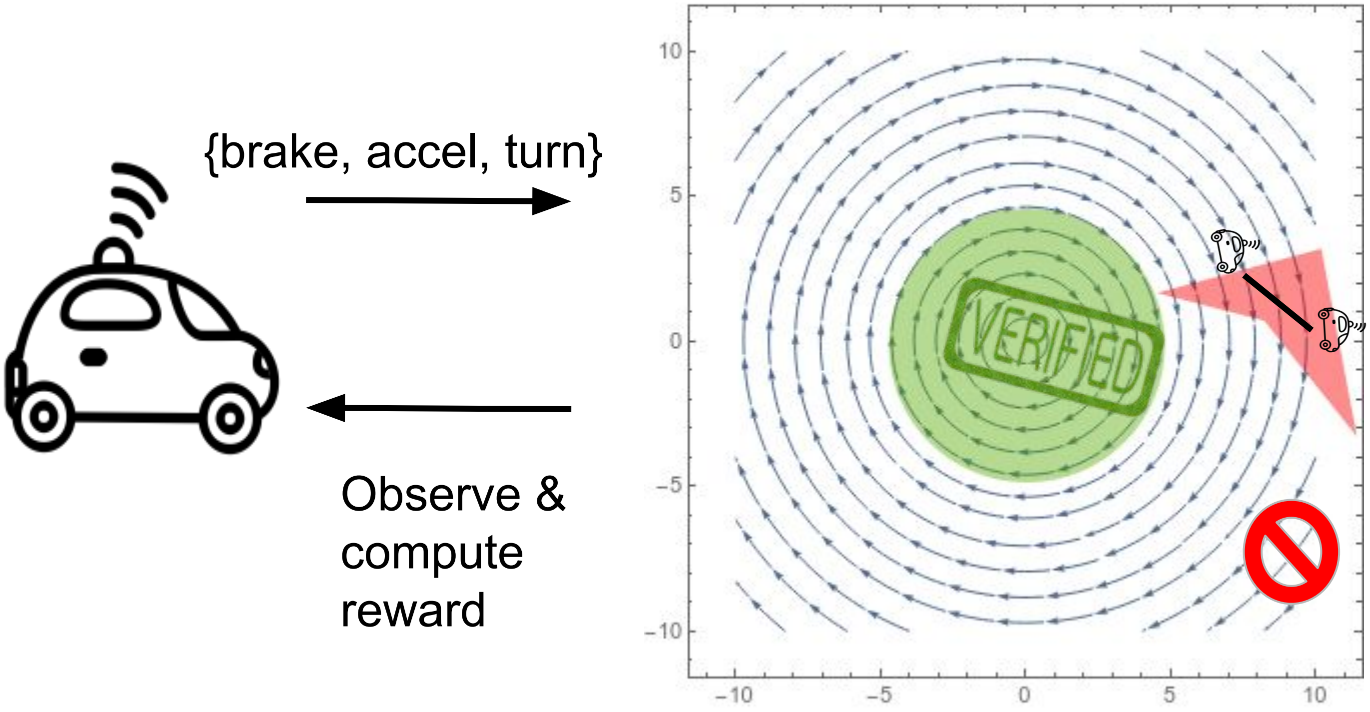
What About the Physical Model?



Speculation is Justified



Leveraging Verification Results to Learn Better



Use a real-valued version of the model monitor as a reward signal

An Example

An Example: The System

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaint; maintVel};
{pos’ = vel, vel’ = acc, t’=1}
```

}*]safe

An Example: The Monitor

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};

{pos' = vel, vel' = acc, t'=1}
```

}*]safe

$$(t_{post} >= 0 \land a_{post} = acc \land v_{post} = acc\, t_{post} + v \land p_{post} = acc\, t_{post}^2/2 + v\, t_{post} + p) \lor$$

$$(t_{post} >= 0 \land a_{post} = 0 \land v_{post} = v \land p_{post} = vt_{post} + p) \lor \text{Etc.}$$

An Example: The Monitor

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};

{pos' = vel, vel' = acc, t'=1}
```

}*]safe

$(t_{post} >= 0 \land a_{post} = acc \land v_{post} = acc\ t_{post} + v \land p_{post} = acc\ t_{post}^2/2 + v\ t_{post} + p) \lor$

$(t_{post} >= 0 \land a_{post} = 0 \land v_{post} = v \land p_{post} = vt_{post} + p) \lor$ Etc.

An Example: The Monitor

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};
{pos’ = vel, vel’ = acc, t’=1}
```

}*]safe

$(t_{post} >= 0 \wedge a_{post} = accel \wedge \mathbf{v_{post} = acc\ t_{post} + v} \wedge p_{post} = acc\ t_{post}^2/2 + v\ t_{post} + p) \vee$

$(t_{post} >= 0 \wedge a_{post} = 0 \wedge \boldsymbol{v_{post} = v} \wedge p_{post} = vt_{post} + p) \vee$ Etc.

An Example: The Monitor

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};
{pos' = vel, vel' = acc, t'=1}
```

}*]safe

$(t_{post} >= 0 \land a_{post} = acc \land v_{post} = accel\ t_{post} + v \land \mathbf{p_{post} = acc\ t_{post}^2/2 + v\ t_{post} + p}) \lor$

$(t_{post} >= 0 \land a_{post} = 0 \land v_{post} = v \land \boldsymbol{p_{post} = vt_{post} + p}) \lor$ Etc.

An Example: The Monitor

$init \rightarrow$ [{

{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};

{**pos' = vel**, *vel' = acc*, t'=1}

}*]safe

$(t_{post} >= 0 \wedge a_{post} = acc \wedge v_{post} = accel\ t_{post} + v \wedge \mathbf{p_{post} = acc\ t_{post}^2/2 + v\ t_{post} + p}) \vee$

$(t_{post} >= 0 \wedge a_{post} = 0 \wedge v_{post} = v \wedge \boldsymbol{p_{post} = vt_{post} + p}) \vee$ Etc.

An Example: The Monitor

- Q.E. for RCF
- ODE solutions backed by proofs

init → [{

```
{?safeAccel;accel ∪ brake ∪ ?safeMaintain; maintainVel};

{pos’ = vel, vel’ = acc, t’=1}
```

}*]safe

$(t_{post} >= 0 \wedge a_{post} = acc \wedge v_{post} = accel\ t_{post} + v \wedge \mathbf{p_{post} = acc\ t_{post}^2/2 + v\ t_{post} + p}) \vee$

$(t_{post} >= 0 \wedge a_{post} = 0 \wedge v_{post} = v \wedge \boldsymbol{p_{post} = vt_{post} + p}) \vee$ Etc.

An Example: The Reward Signal (simplified)

$$x>=0 \land v>=0 \land A>=0 \rightarrow [\{x' = v,\ v' = A\}]x>=0$$

An Example: The Reward Signal (simplified)

$$x>=0 \land v>=0 \land A>=0 \rightarrow [\{x' = v,\ v' = A\}]x>=0$$

Minimize **max(vError, xError)** where

vError = max(v_{post} - ($A*t_{post}$ + v) , $A*t_{post}$ + v - v_{post})

xError = max(

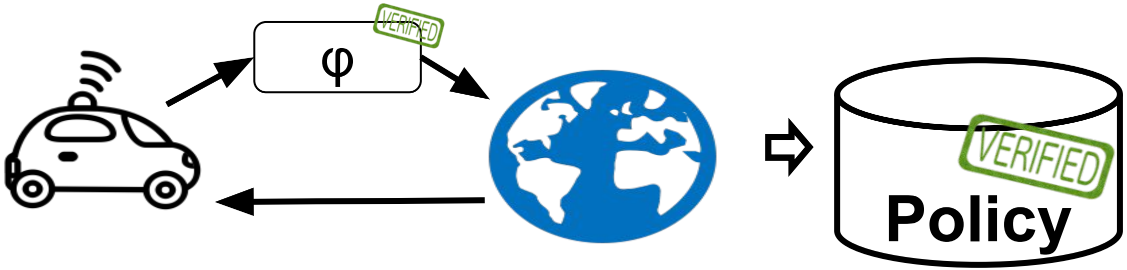
x_{post} - ($A*t_{post}{}^2/2$ + v * t_{post} + x)

($A*t_{post}{}^2/2$ + $v*t_{post}$ + x) - x_{post}

)

Conclusion

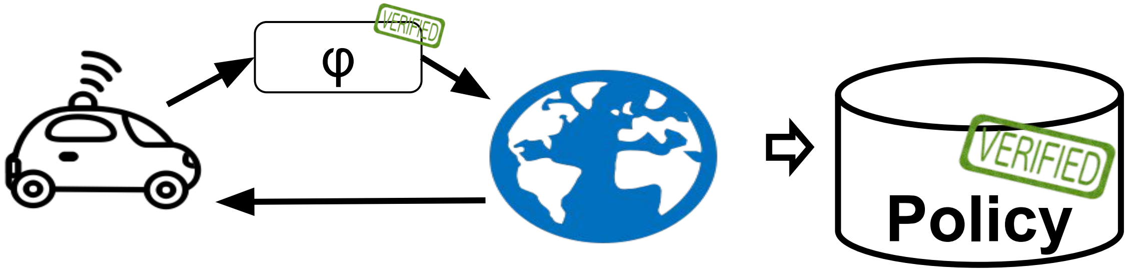
Justified Speculative Control provides the best of logic and learning:



Conclusion

Justified Speculative Control provides the best of logic and learning:

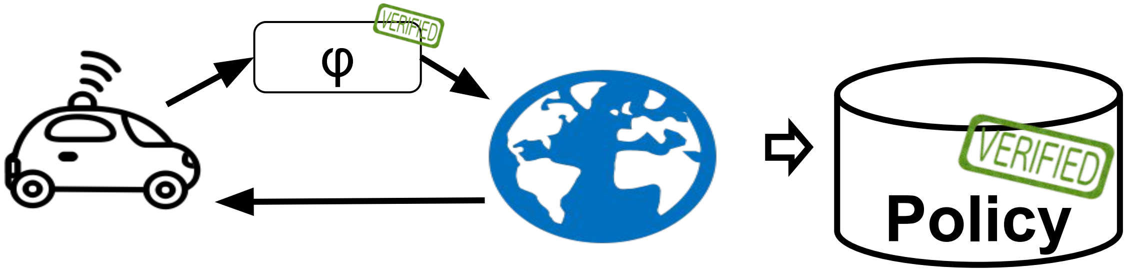
- Formally model the control system (**control + physics**)



Conclusion

Justified Speculative Control provides the best of logic and learning:

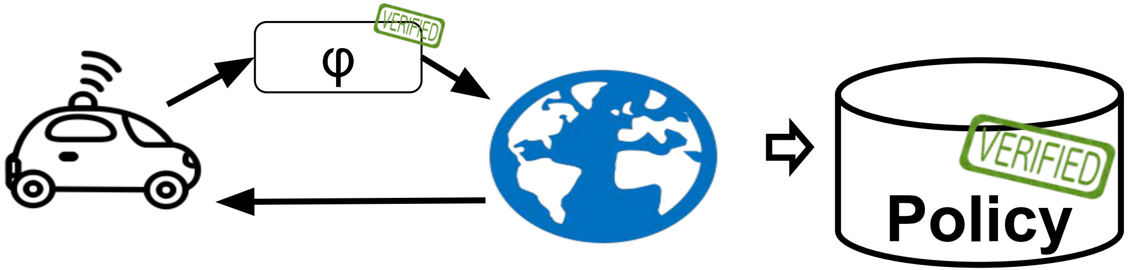
- Formally model the control system (**control + physics**)
- Learn how to resolve non-determinism in models.



Conclusion

Justified Speculative Control provides the best of logic and learning:

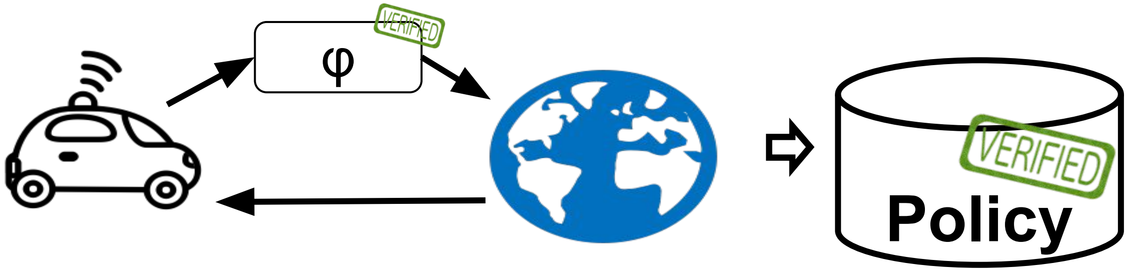
- Formally model the control system (**control + physics**)
- Learn how to resolve non-determinism in models.
- Leverage theorem proving to transfer **proofs** to learned policies.



Conclusion

Justified Speculative Control provides the best of logic and learning:

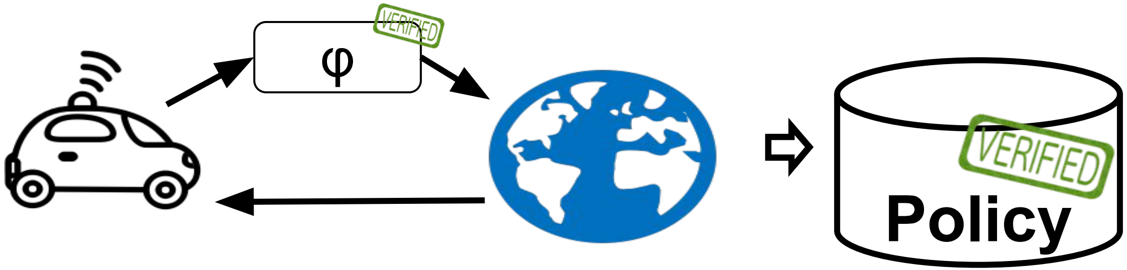
- Formally model the control system (**control + physics**)
- Learn how to resolve non-determinism in models.
- Leverage theorem proving to transfer **proofs** to learned policies.
- Unsafe **speculation is justified** when model deviates from reality



Conclusion

Justified Speculative Control provides the best of logic and learning:

- Formally model the control system (**control + physics**)
- Learn how to resolve non-determinism in models
- Leverage theorem proving to transfer **proofs** to learned policies
- Unsafe **speculation is justified** when model deviates from reality, but **verification results can still be helpful**!



Conclusion

Justified Speculative Control provides the best of logic and learning:

- Formally model the control system (**control + physics**)
- Learn how to resolve non-determinism in models
- Leverage theorem proving to transfer **proofs** to learned policies
- Unsafe **speculation is justified** when model deviates from reality, but **verification results can still be helpful**!