Scenic and VerifAI: Tools for Assured AI-Based Autonomy

Sanjit A. Seshia **Edward Kim** **Daniel J. Fremont** **Atul Acharya**

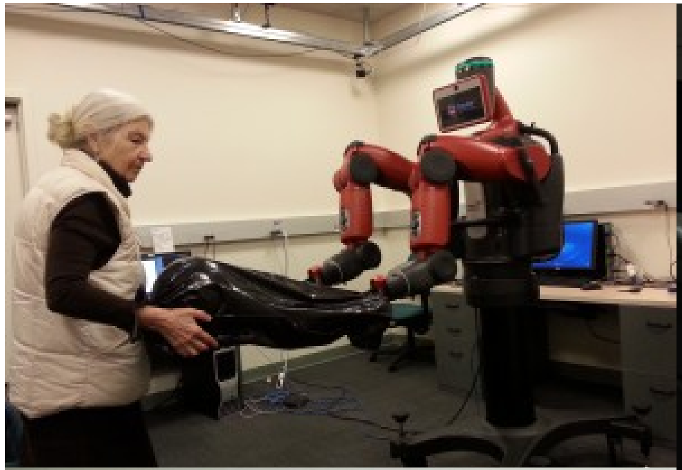



UC Berkeley UC Santa Cruz AAA NCNU

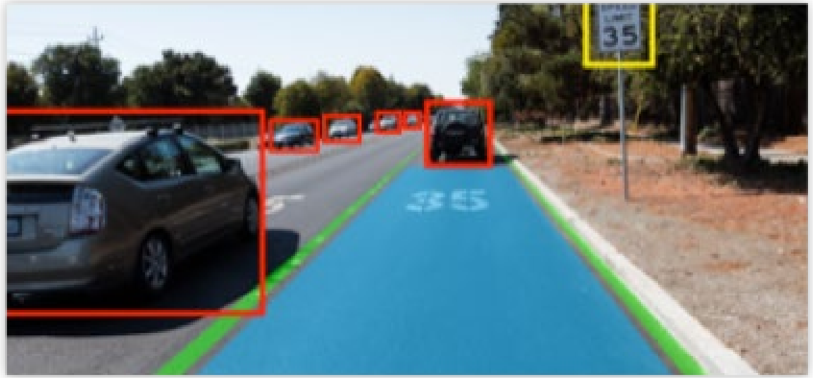
Webinar
August 20, 2020

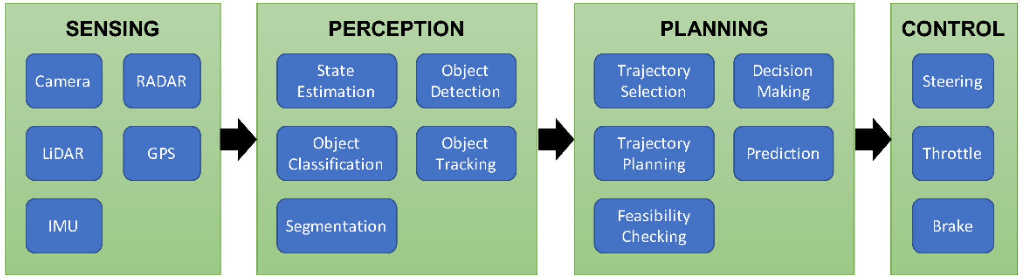
Artificial Intelligence (AI) and Autonomy

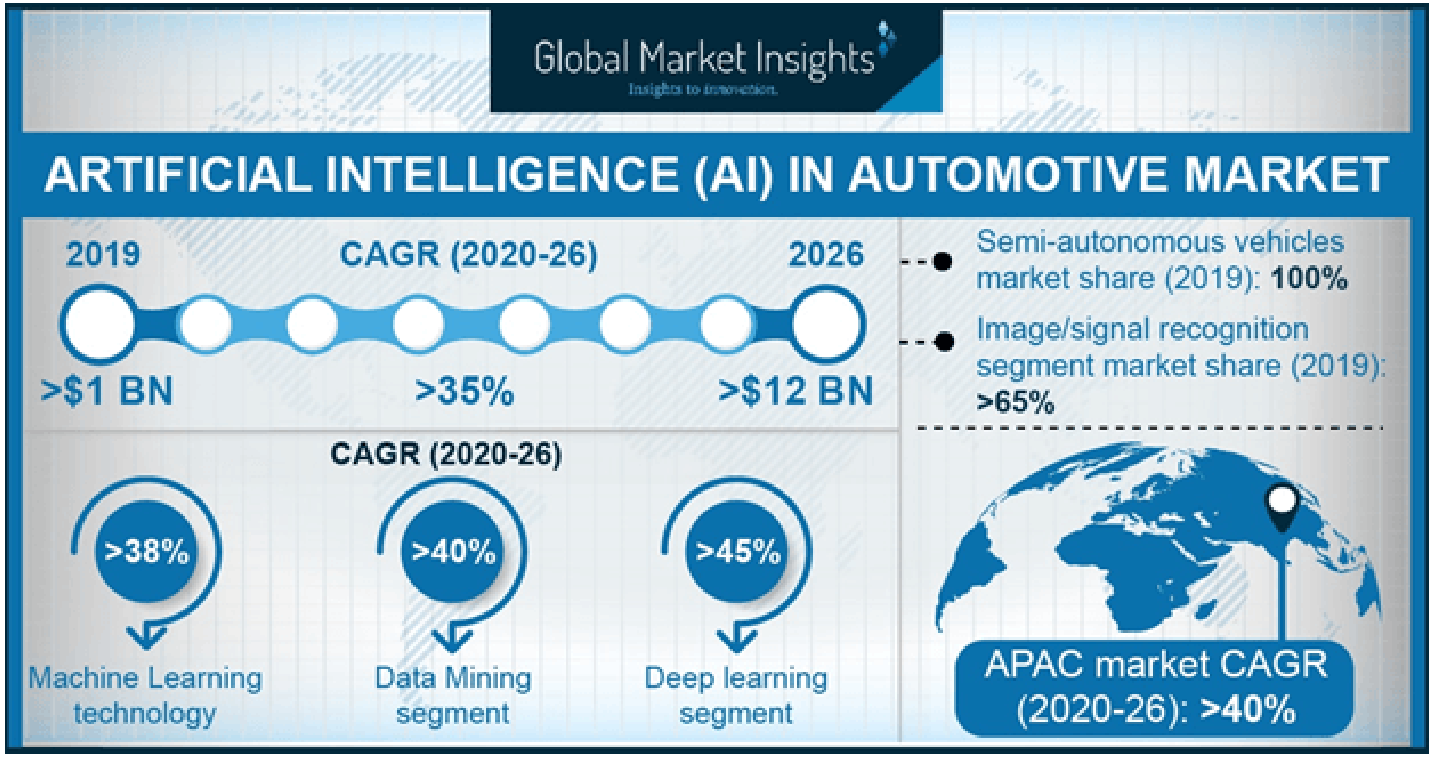
Computational Systems that attempt to mimic aspects of human intelligence, including especially the ability to learn from experience.



Growing Use of Machine Learning/Artificial Intelligence in Safety-Critical Autonomous Systems







Source: gminsights.com

Growing Concerns about Safety:

- Numerous papers showing that *Deep Neural Networks can be easily fooled*
- *Accidents*, including some *fatal,* involving potential failure of AI/ML-based perception systems in self-driving cars

Can we address the Design & Verification Challenges of AI/ML-Based Autonomy with Formal Methods?

Precise, Programmatic **Environment/Scenario Modeling**



φ Mathematical **Specification of Requirements and Metrics**

Scalable Algorithms for **Verification and Testing**

Methodologies for **Provably-Robust System Design**

Berkeley DeepDrive DARPA iCyPhy NSF VeHiCaL

http://vehical.org

S. A. Seshia, D. Sadigh, S. S. Sastry.
Towards Verified Artificial Intelligence. July 2016. https://arxiv.org/abs/1606.08514.

Scenic

High-Level, Probabilistic Programming Language for Modeling Environment Scenarios

VerifAI

Requirements Specification + Algorithms for Design, Verification, Testing, Debugging

GitHub **Open-Source Tools**

https://github.com/BerkeleyLearnVerify/Scenic
https://github.com/BerkeleyLearnVerify/VerifAI

for

Industry

Improve assurance of the systems you build

Academia

Use these tools in your research

Government/ Regulators

Evaluate the safety of AI-based autonomous systems

Share Scenarios and Metrics

Community

Develop Corpus of Tools and Data

Outline for this Webinar

Part I: Overview

- Challenges for Assurance of Autonomous Driving Systems
- Overview of VerifAI and Scenic
- Case Study on Formal Scenario-Based Testing in Simulation and on the Road

Part II: Tutorial

- Spatial modeling, data generation, and debugging ML-based perception with Scenic
- Spatio-temporal scenario modeling, testing, falsification, debugging, retraining with Scenic and VerifAI

Conclusion & Outlook

Challenges for Assuring Safety of ADS

What We Mean By Safety in Autonomous Driving

Safety → "absence of unreasonable risk"

SAFETY	RISK

ISO 26262

Functional Safety (FuSa)
- *Hazards due to E/E system*

ISO/PAS 21448

(**SOTIF**) **Safety of the Intended Functionality**
- *Hazards due to nominal system operation*

Risk = f (Severity, Exposure, Controllability)
[ASIL, ISO 26262]

→ Severity
- types of injuries

→ Exposure
- frequency of hazards

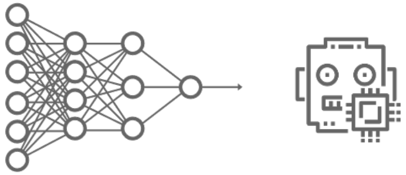
→ Controllability
- how much driver can prevent injury

No system has absolutely zero risk

Improving Safety in Automated Driving Systems: Needs

Challenges for safety-critical systems

SENSE → PERCEIVE → PREDICT → PLAN → ACT



Handle **complex neural-network** based **perception** and **prediction** tasks, including planning and control

Toolchain that integrates **design** and **verification** with **data generation** and training/testing of ML components

Simulation is important for complex, **real-world scenarios** for which **real world data is difficult/dangerous**

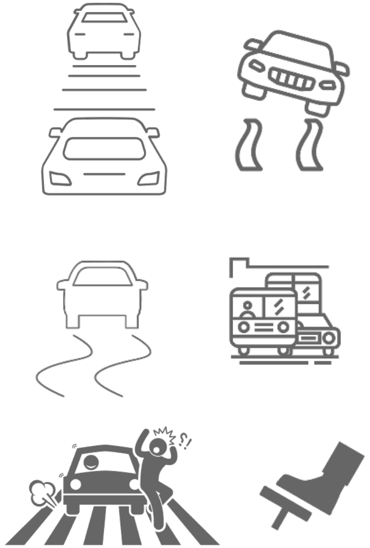
Why Testing ADS is Complex

Verification & Validation: Assurance of "*positive risk balance*"
Scenario-based testing is one standard approach for V&V

Wide variety of functions and scenarios…

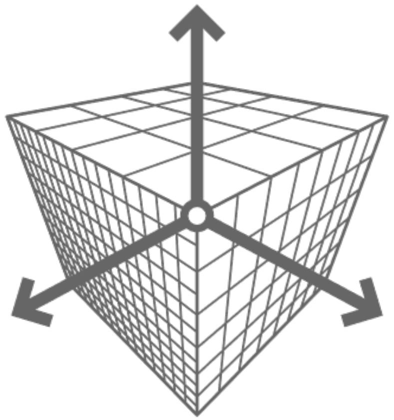
ADAS

ACC
FCW
CAS
AEB
P-AEB
LKA
LDW
BSM
ISA
…



NHTSA, EuroNCAP, JNCAP

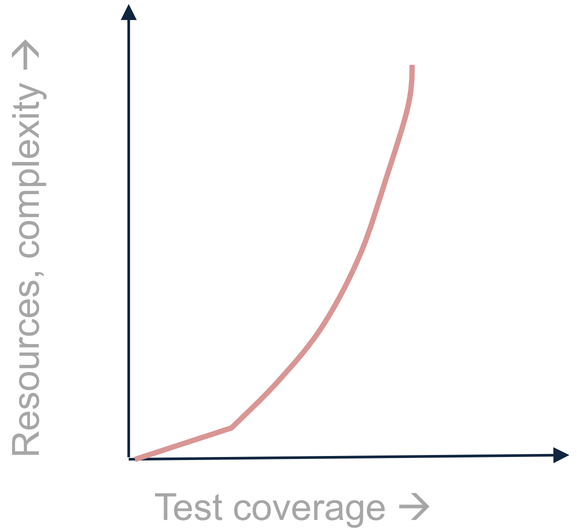
…lead to high-dimensionality of parameter search



f(speed, objects, agents, …)

Test matrix

…resulting in high complexity



Operational Design Domain: What and Why

Operating environment within which an ADS can **safely** perform its dynamic driving task (**DDT**)

Formulation

Category / sub-category / attributes
Static and dynamic elements
Additive / subtractive elements

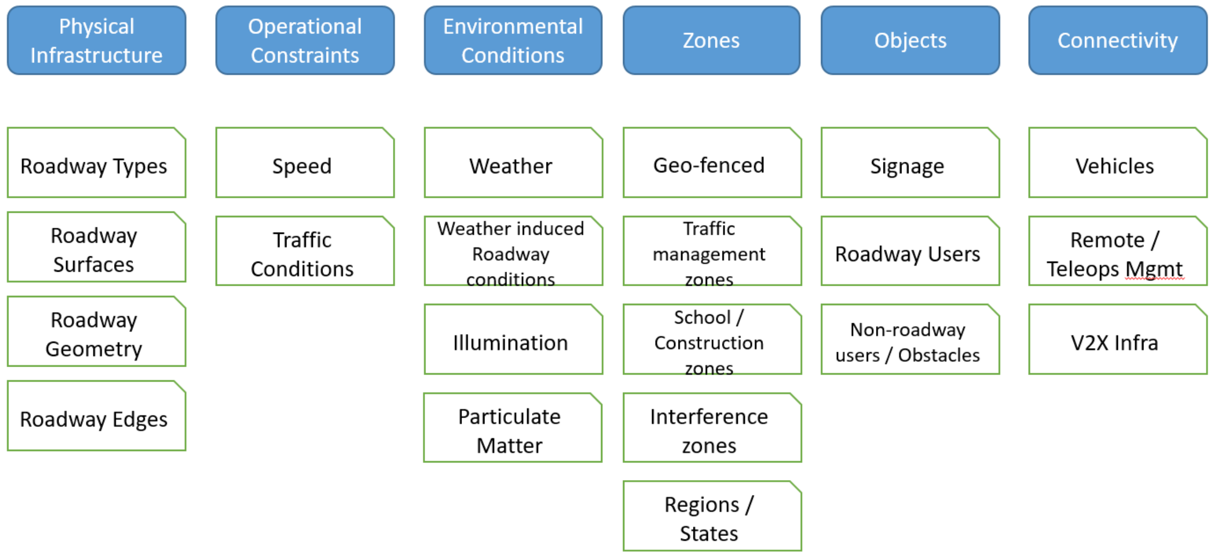
Requirements

Precisely definable
Comprehensible (human / machine)
Measurable
Monitorable (by ADS / operator)

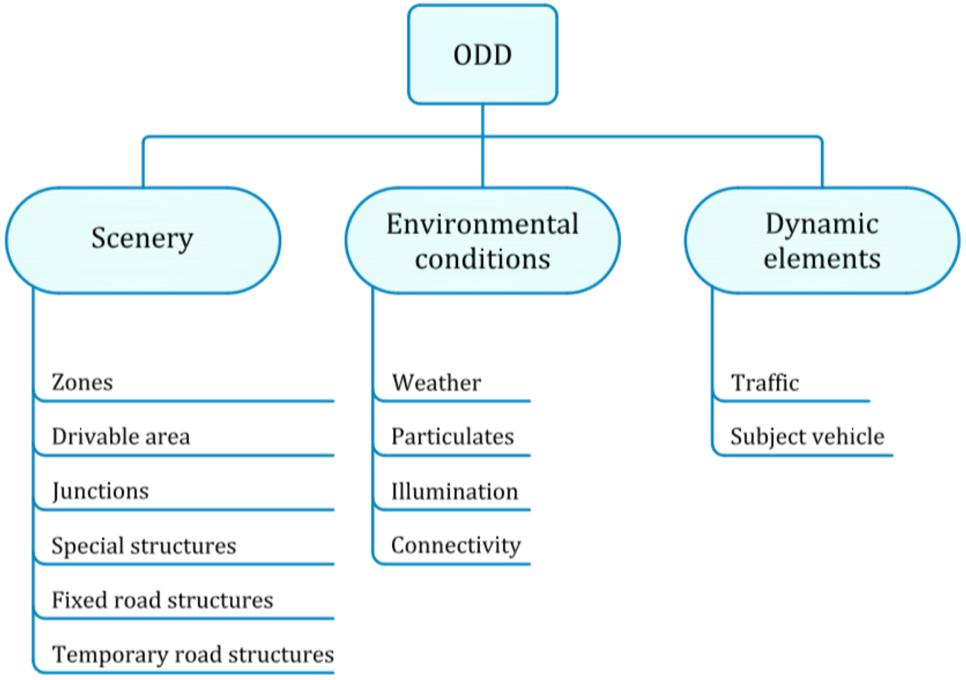
Boundary conditions

ODD detection / departure
Min Risk Condition (MRC)
Min Risk Maneuver (MRM)
Fail Safe / Fail Operational

ODD Classification



Source: NHTSA 13882 ADS Scenario Framework



Source: BSI PAS 1883

Safety Metrics: How is Success / Failure Measured

System performance is context-dependent (mission/scenario/test-case/etc.)

“Disengagement” is not a safety metric

Standards / Proposals…

ANSI / UL 4600 – “Safety Performance Indicators (SPIs)”

IEEE P2846 – “motion control based metrics”

Intel’s Responsibility-Sensitive Safety (**RSS**)

NVIDIA’s Safety Force-Field (**SFF**)

…Convergence

Vehicle Dynamics Based
Min Safe Distance Violation
Proper Response Action
Min Safe Distance Factor
Min Safe Distance Calc Error
Collision Incident
Rules-of-road violation
ADS Active
Human Traffic Control Detection Error Rate
Time to Collision (TTC)
Post-Encroachment Time
Aggressive Driving
Collision Avoidance Capability (CAC)

SPIs
Incident rates
Violation rates
- By human exposure
- By item exposure

Hazard occurrence rates
Unmitigated hazard rates
Psychological comfort rates
ODD departure rates
…
Post-deployment defect rates
Field failure rates
Misclassification rates
…

Sources:
1. “Driving Safety Performance Assessment Metrics for ADS-equipped Vehicles”, Wishart, et al (SAE WCX 2020)
2. “Collision Avoidance Capability Metric”, Silberling, et al (SAE WCX 2020)

Bridging Simulation and Real World

Testing on road/track is expensive but important, hence

- Need to carefully design road/track tests (e.g. NHTSA, NCAP, IIHS, …)
- Customize test plans based on ODD, autonomy functions, infra, …

- Ensure that models in sim are fit for their test purpose
- Ensure match between simulation scenarios and road testing scenarios

- Need fallback options (e.g. MRC) in case safety cannot be assured
- Test boundary conditions very well

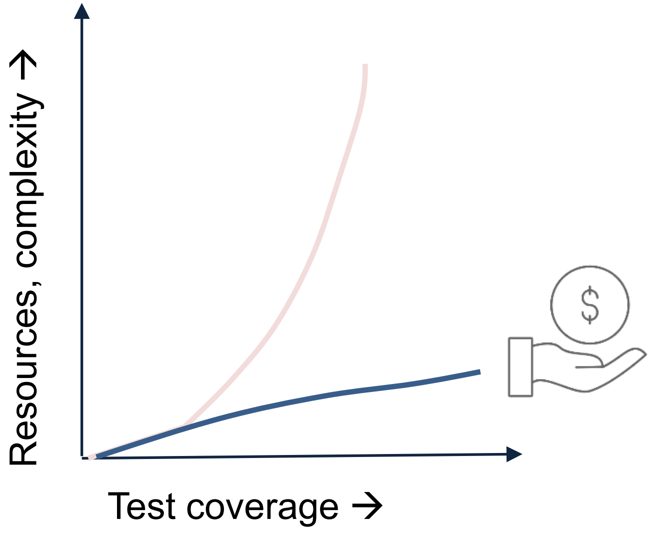
Simulation and Formal Methods can Make ADS Testing Efficient and Bridge the Gap with Road Testing

Simulation

- Efficiently search large space
- Create complex interactions safely

Formal methods

- Temporal logic
- Falsification
- Counterexample-guided retraining
- Parameter synthesis



> *"All models are wrong… but some are useful"* – George P. Box

Overview of Scenic and VerifAI

SCENIC: Environment Modeling and Data Generation

- *Scenic* is a probabilistic programming language defining *distributions over scenes/scenarios*
- *Use cases:* data generation, test generation, verification, debugging, design exploration, etc.

```
model scenic.domains.driving.model

ego = Car

spot = OrientedPoint on visible curb
badAngle = Uniform(1.0, -1.0) * Range(10, 20) deg
parkedCar = Car left of spot by 0.5,
                facing badAngle relative to roadDirection
```

```
model scenic.domains.driving.model

behavior PullIntoRoad():
    while (distance from self to ego) > 15:
        wait
    FollowLaneBehavior(lane=ego.lane)

ego = Car with behavior DriveAvoidingCollisions

spot = OrientedPoint on visible curb
badAngle = Uniform(1.0, -1.0) * Range(10, 20) deg
parkedCar = Car left of spot by 0.5,
                facing badAngle relative to roadDirection,
                with behavior PullIntoRoad
```

Example: Badly-parked car

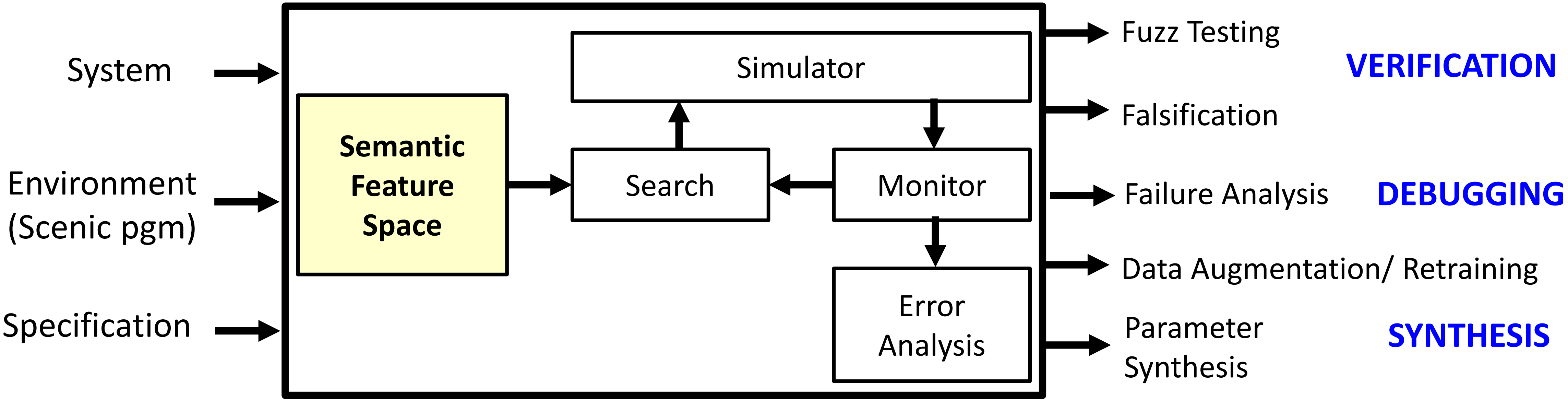
Image created with GTA-V

Video created with CARLA

[D. Fremont et al., "Scenic: A Language for Scenario Specification and Scene Generation", TR 2018, PLDI 2019.]

VERIFAI: A Toolkit for the Design and Analysis of AI-Based Systems

[Dreossi et al. CAV 2019, https://github.com/BerkeleyLearnVerify/VerifAI]



Relevant Use Cases for Scenic and VerifAI

- Scenic Programs can specify ODDs and Test Scenarios
- Can specify Safety Properties/Metrics in VerifAI
- Scenic+VerifAI can
  - Automatically generate tests in simulation
  - Automatically find edge cases to safety
  - Generate data for training and testing ML models and perception
  - Automatically synthesize parameters for ML, planning, control
  - Debug and explain the behavior of perception, planning, control systems
  - Bridge the gap between simulation-based assessment and real-world/road testing
  - …

Industrial Case Study:

Formal Scenario-Based Testing in Simulation and the Real World

3-Way Project Collaboration

A. Acharya, P. Wells, X. Bruso

GoMentum Station proving ground
4Active pulley equipment, pedestrian dummy, OxTS IMU, dGPS, etc.

http://vehical.org

S. A. Seshia, D. Fremont, E. Kim, Y. V. Pant, H. Ravanbakhsh

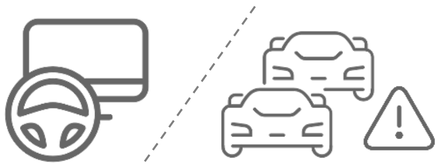
SCENIC scenario description language,
VerifAI toolkit for design and verification of AI based systems

LG Electronics R&D

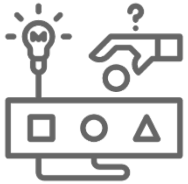
S. Lemke, Q. Lu, S. Mehta

LGSVL Simulator (open source)
LG's research AV with Baidu's Apollo autonomy stack

Key Research Questions



#1 Safety violations in simulation: Do they transfer to the **real world**? How well?

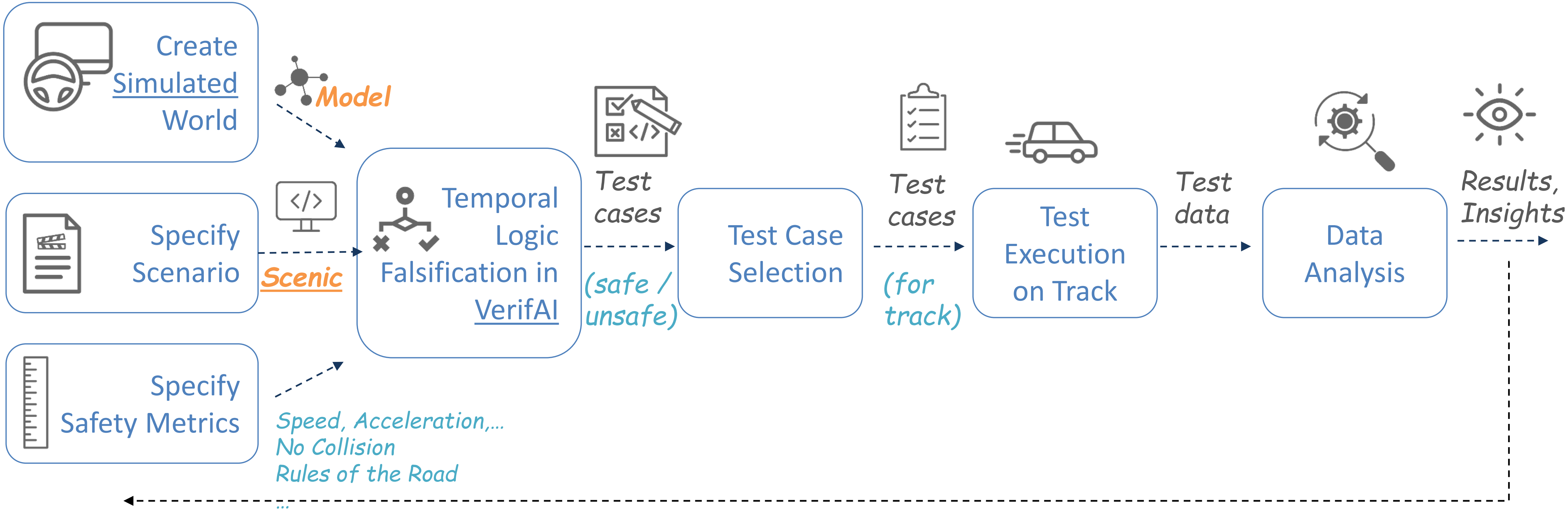


#2 Effective real-world testing: Can we use **formally guided simulation** to design effective **real-world tests**?

First use of formal methods for scenario-based testing of AI-based autonomy in both simulation and real world

Fremont, Kim, Pant, Seshia, Acharya, Bruso, Wells, Lemke, Lu, Mehta, ***"Formal Scenario-Based Testing of Autonomous Vehicles: From Simulation to the Real World"***, Arxiv e-prints, https://arxiv.org/abs/2003.07739 *[appearing **ITSC 2020**]*

Formal Scenario-Based Testing
(with Scenic and VerifAI)



Source: Fremont et al., *"Formal Scenario-Based Testing of Autonomous Vehicles: From Simulation to the Real World"*, Intelligent Transportation Systems Conference (ITSC) 2020, to appear. https://arxiv.org/abs/2003.07739

Scenario Overview: Focus on Vulnerable Road Users (VRUs)

+53%

Pedestrian fatalities: 53% increase in the last decade (2009-2019)
2019: ~6500 (estimated)

17%

Of all traffic fatalities, 17% are **Pedestrians**

67%

Fatalities at **night** (low-light, limited vision environment)

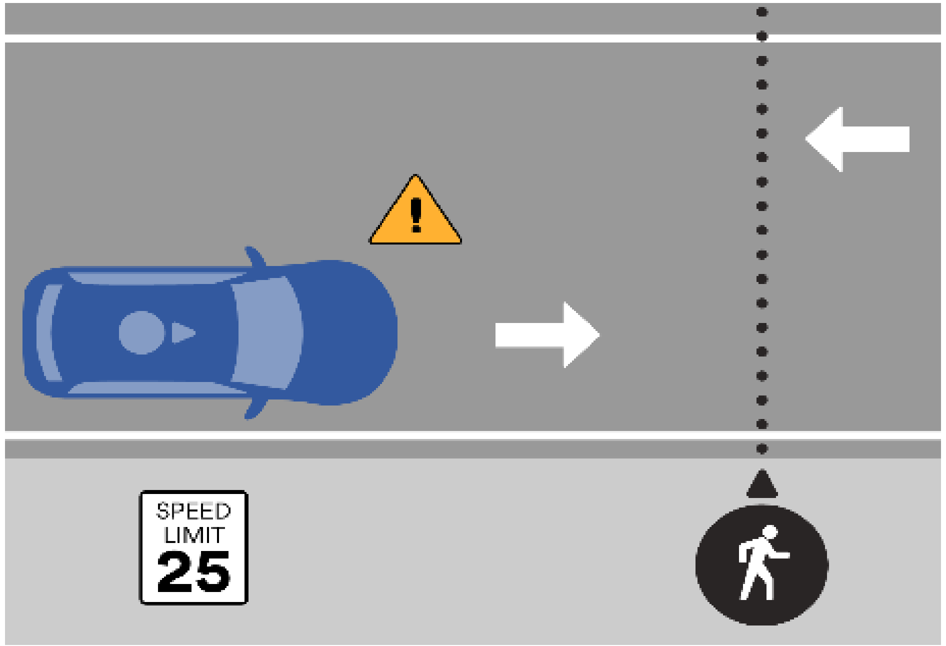
Source:
GHSA: https://www.thecarconnection.com/news/1127308_pedestrian-deaths-reach-30-year-high-in-2019
IIHS: https://www.iihs.org/topics/pedestrians-and-bicyclists

Test Equipment and Use at AAA GoMentum Testing Grounds

Robotic platform for Test Targets

Scenario Execution

[Shows EuroNCAP VRU AEB]



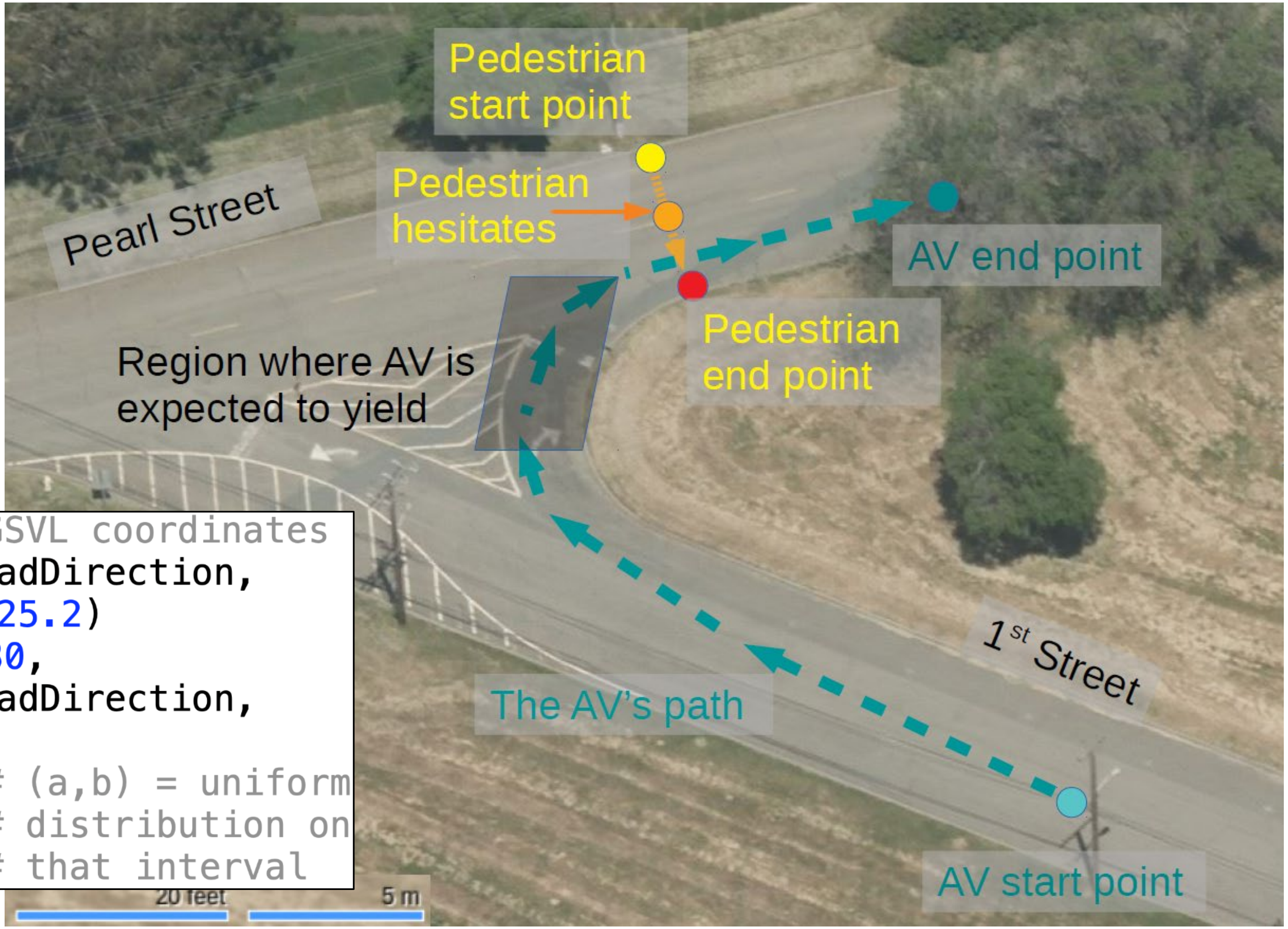
Scenario Evaluation

Object & Event Detection/Response: Metrics & Evaluation

- Object detection
- Time to collision
- Separation distance
- Deceleration profile
- Autonomy Disengagement

Example Scenario: AV making right turn, pedestrian crossing

Lincoln MKZ running Apollo 3.5



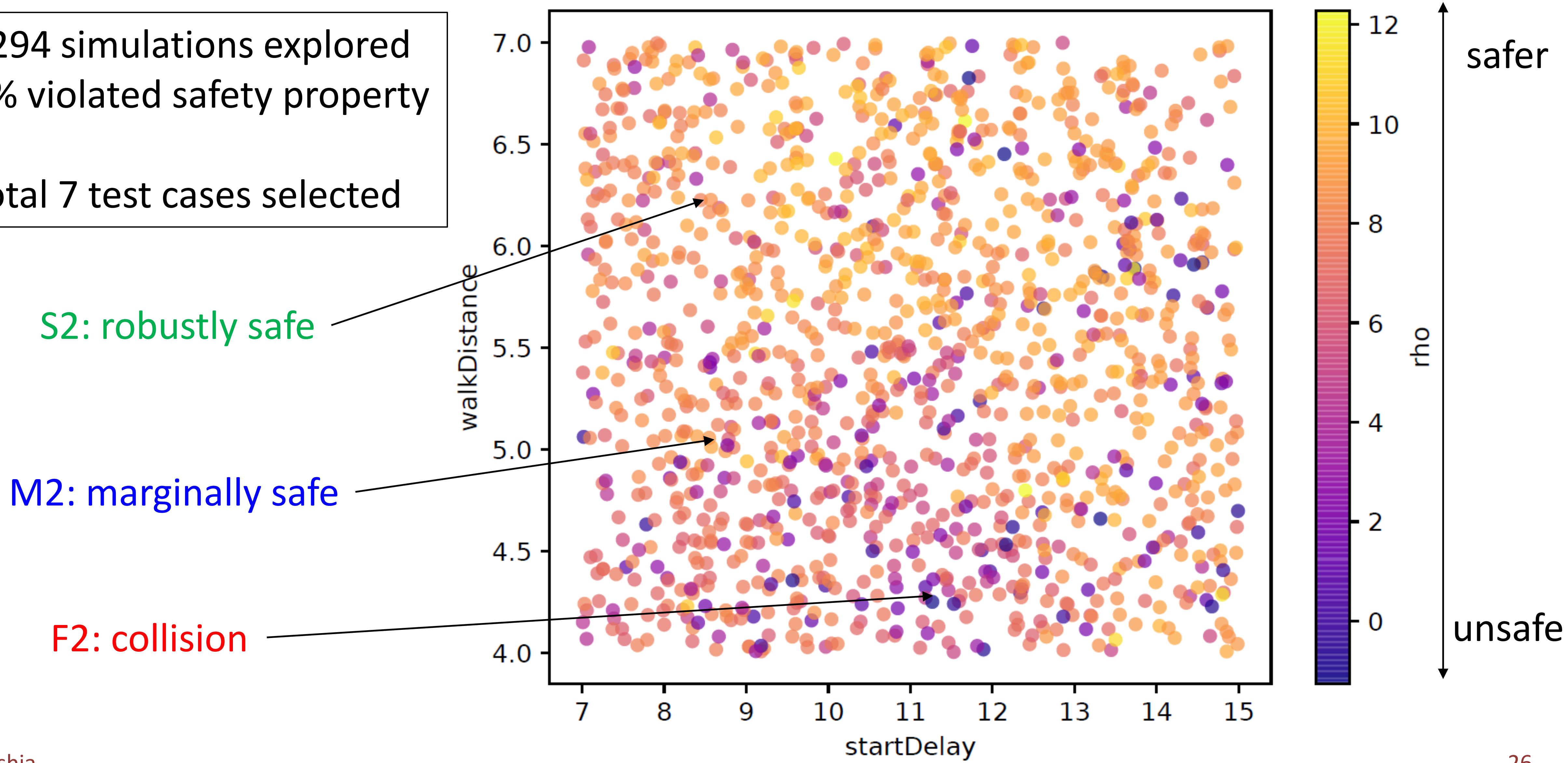

```
ego = EgoCar at 38.6 @ 183.9,  # LGSVL coordinates
       facing 10 deg relative to roadDirection,
       with behavior DriveTo(40 @ 225.2)
ped = Pedestrian at 19.782 @ 225.680,
       facing 90 deg relative to roadDirection,
       with behavior Hesitate,
       with startDelay (7, 15),    # (a,b) = uniform
       with walkDistance (4, 7),   # distribution on
       with hesitateTime (1, 3)    # that interval
```

Snippet of Scenic program

Results: Falsification and Test Selection

1294 simulations explored
2% violated safety property

Total 7 test cases selected



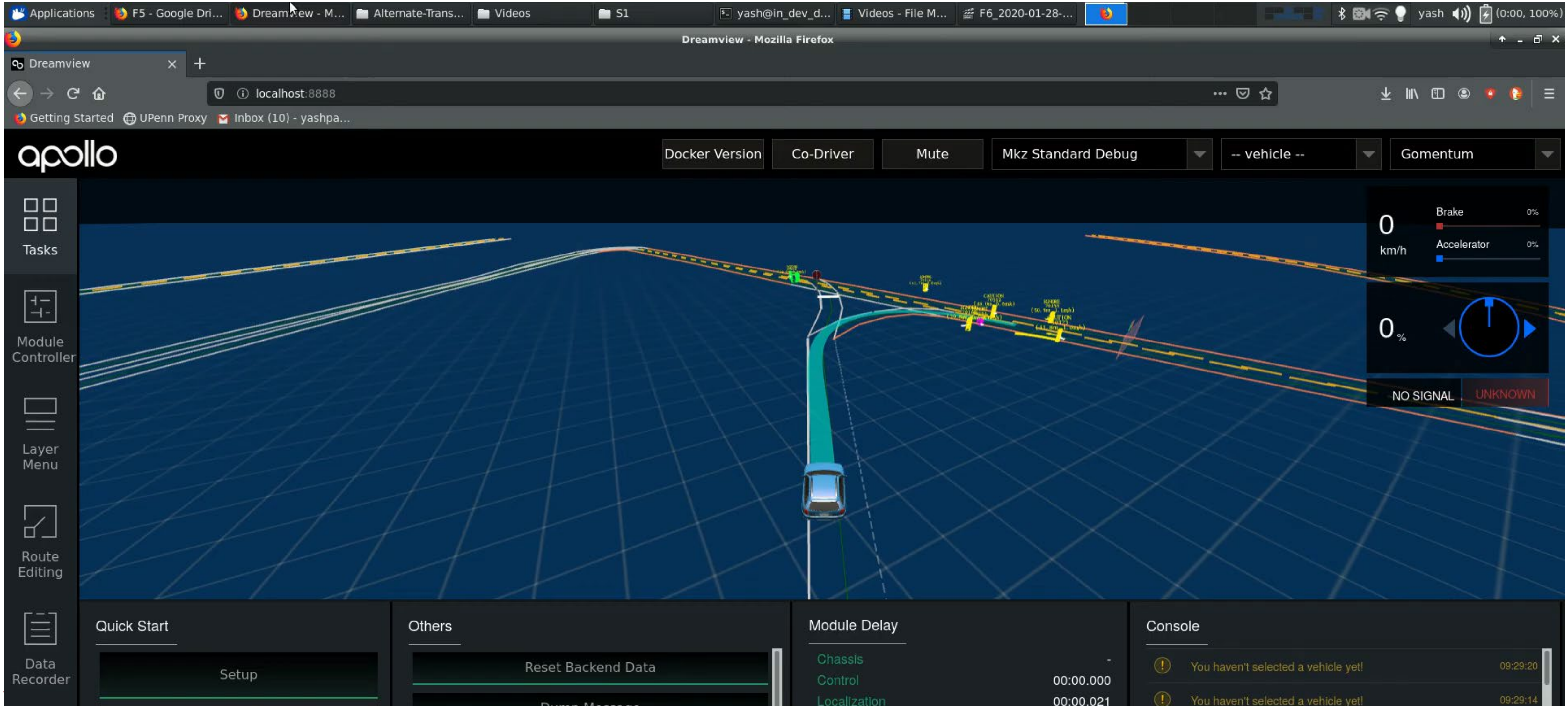
Results: Does Safety in Simulation → Safety on the Road?

Unsafe in simulation → unsafe on the road: **62.5% (incl. collision)**

Safe in simulation → safe on the road: **93.5% (no collision)**

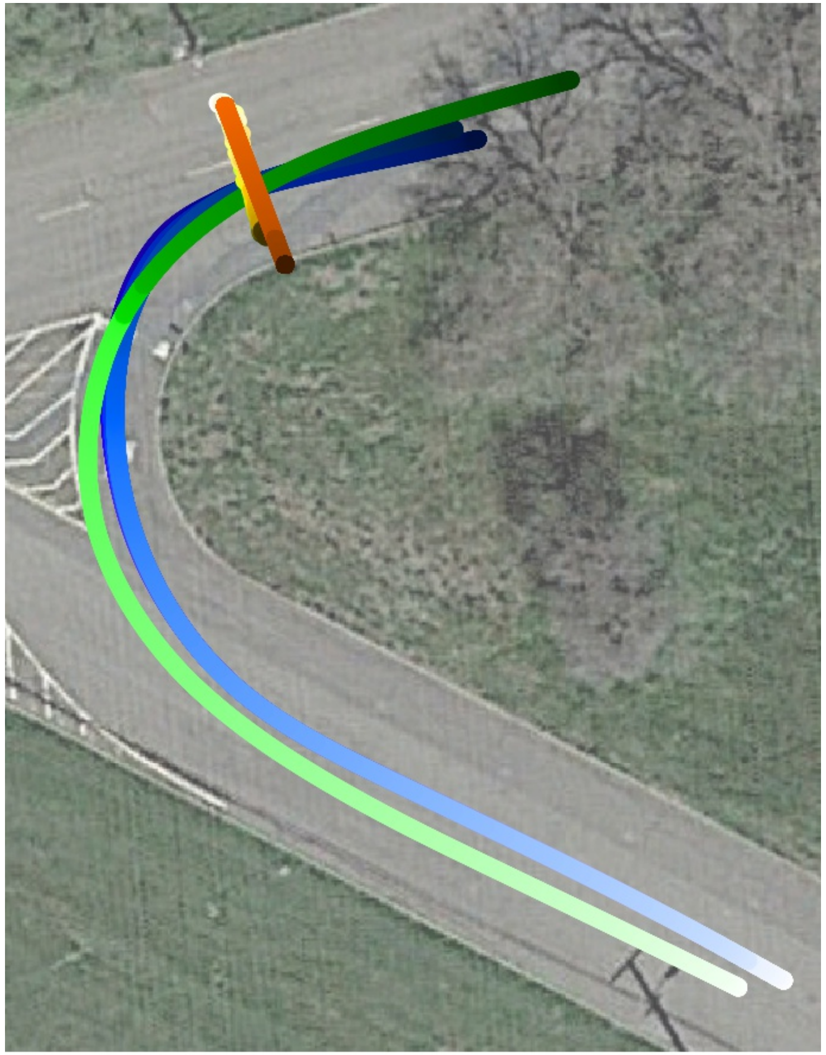
Results: Why did the AV Fail?

Perception Failure: Apollo 3.5 lost track of the pedestrian several times

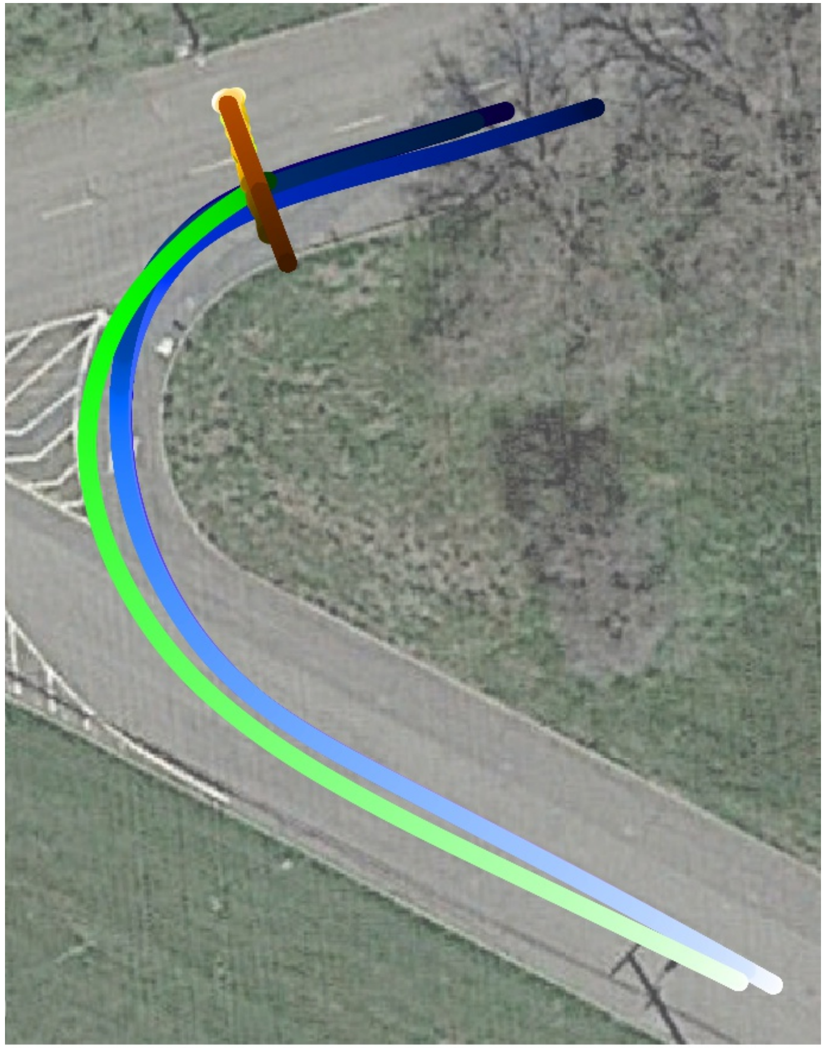


S. A.

Results: How well do the trajectories match?



S1 Run 2



F1 Run 1

Green – AV real
Blue – AV sim

Orange – Ped real
Yellow – Ped sim

Conclusion

- Scenic allows easy modeling of complex scenarios for AI-based autonomy + associated data generation
- VerifAI covers range of design, verification, and debugging tasks for AI-based autonomy
- ITSC 2020 Case Study: Scenic+VerifAI can be used to bridge the simulation-to-real world testing gap
  - Effectively evaluate safety via formally-guided simulation
  - Reduce expense of real-world testing by orders of magnitude
- Up next: 1 hour tutorial will give further details on Scenic and VerifAI and use cases for both tools

Ongoing Work and Directions

- Compiling a library of scenarios in Scenic
- Evaluation on more complex, higher-dimensional scenarios
- New algorithms for formal verification and synthesis
- Tools for automated analysis/triage of failure cases
- Improvements in track testing equipment and their connection to simulation

and more…

We welcome participation from the community!

https://github.com/BerkeleyLearnVerify/VerifAI
https://github.com/BerkeleyLearnVerify/Scenic/

Acknowledgments: Contributors, Co-authors, Collaborators

UC Berkeley

- Johnathan Chiu
- Tommaso Dreossi
- Shromona Ghosh
- Francis Indaheng
- Sebastian Junges
- Kevin Li
- Yash Vardhan Pant
- Hadi Ravanbakhsh
- Jay Shenoy
- Hazem Torfah
- Marcell Vazquez-Chanlatte
- Kesav Viswanadha
- Xiangyu Yue

UC Berkeley

- Kurt Keutzer
- Alberto Sangiovanni-Vincentelli
- Pravin Varaiya
- Alex Kurzhanskiy

UC Santa Cruz

- Ellen Kalvan

Boeing

- Dragos Margineantu
- Denis Osipychev

AAA NCNU

- Xantha Bruso
- Paul Wells

LG Electronics

- Steve Lemke
- Shalin Mehta
- Qiang Lu

NASA Ames

- Corina Pasareanu
- Divya Gopinath

Thank you!