Bridging Simulation and the Real World with VerifAI and Scenic

Sanjit A. Seshia
Professor
EECS Department, UC Berkeley

Students/Postdocs: D. Fremont, E. Kim, Y. V. Pant, H. Ravanbakhsh
Collaborators: A. Acharya, X. Bruso, P. Wells (AAA), S. Lemke, Q. Lu, S. Mehta (LG)
Improving Safety of Autonomous Vehicles with Formal Methods

• Need to handle **complex neural network-based perception & prediction** including interaction with planning & control

• Need toolchain that **integrates design and verification with data generation and training/testing** of ML components

• Simulation important for **complex environment scenarios** for which real world data is hard/impossible to gather
**SCENIC: Environment Modeling and Data Generation**

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

```python
from gta import Car, curb, roadDirection

ego = Car

spot = OrientedPoint on visible curb
badAngle = Uniform(1.0, -1.0) * (10, 20) deg
Car left of (spot offset by -0.5 @ 0),
    facing badAngle relative to roadDirection
```

Some Applications of Scenic

• Data Generation, (Re)-Training
  – More controllable, interpretable
  – Improves performance significantly
  – Rare scenarios, controlled distributions, etc.

• Debugging Failures
  – Vary scenarios systematically
  – Explain failures of ML

• Design Space Exploration

Car detection with occlusions

Test Hypothesis: does the car model lead to a mis-detection?
VERIFAI: A Toolkit for the Design and Analysis of AI-Based Systems [CAV 2019]

https://github.com/BerkeleyLearnVerify/VerifAI

VERIFICATION
- Fuzz Testing
- Falsification

DEBUGGING
- Failure Analysis
- Data Augmentation/Retraining
- Parameter Synthesis

SYNTHESIS

<table>
<thead>
<tr>
<th>System</th>
<th>Environment (Scenic pgm)</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>Search</td>
<td>Monitor</td>
</tr>
<tr>
<td>Error Analysis</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Semantic Feature Space |

ROBOTICS

AUTONOMOUS DRIVING

AIRCRAFT
Key Questions

1. We can find LOTS of safety violations in simulation.
   - Do these transfer to the real world? How well?

2. Real world testing is expensive, tedious, and time-consuming.
   - Can we use formally-guided simulation to effectively design real-world tests/experiments?

Formal Scenario-Based Test Generation with VerifAI and Scenic

Create Simulated World → Simulation Model

Specify Scenario → Scenic Program → Temporal Logic Falsification in VerifAI

Specify Safety Properties/Metrics → Safety Property

Test Cases (Safe/Unsafe) → Test Case Selection → Test Cases for Track → Test Execution on the Track → Test Data → Data Analysis

Collaboration with AAA and LG:
- AAA’s GoMentum Testing Grounds (near Concord, CA)
- LGSVL Simulator (open source, including full AV Stack simulation) and LG’s AV running Baidu’s Apollo software
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),
    with walkDistance (4, 7),
    with hesitateTime (1, 3)

Snippet of Scenic program
Results: Falsification and Test Selection

1294 simulations explored
2% violated safety property
Total 7 test cases selected

- S2: robustly safe
- M2: marginally safe
- F2: collision
Results: Does Safety in Simulation $\rightarrow$ Safety on the Road?

Unsafe in simulation $\rightarrow$ unsafe on the road: \textit{62.5\% (incl. collision)}
Safe in simulation $\rightarrow$ safe on the road: \textit{93.5\% (no collision)}
Results: Why did the AV Fail?

**Perception Failure:** Apollo 3.5 lost track of the pedestrian several times.
Results: How well do the trajectories match?

Green – AV real
Blue – AV sim
Orange – Ped real
Yellow – Ped sim
Conclusion

- Scenic allows easy modeling of complex AV driving scenarios and associated data generation
- VerifAI covers range of design, verification, and debugging tasks for AI based autonomous systems
- Scenic+VerifAI can be used to bridge the sim-to-real gap
  - Concrete evidence for effectiveness of formally-guided simulation
  - Reduce expense of real-world testing with systematic test generation
- Future Work:
  - More detailed automated analysis of failure cases
  - Evaluation on more complex, higher-dimensional scenarios
  - Improvements in track testing equipment and their connection to simulation

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