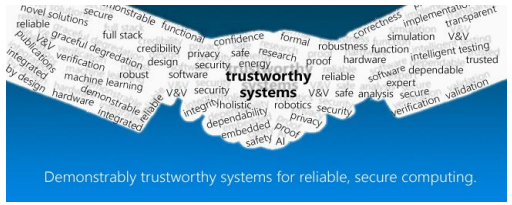
Automated Approaches to Test Generation in Simulation-based Verification

Verification Futures 2022

Dr Greg Chance
Trustworthy Systems Lab
University of Bristol

University of BRISTOL



Overview

Background
- Capri & Robopilot
- Research aims

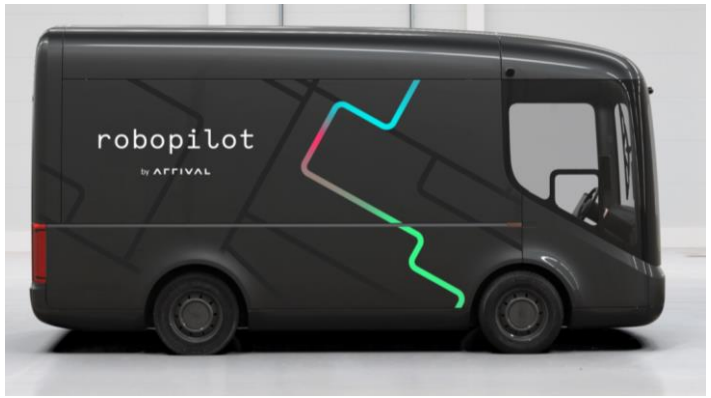
Agency-Directed Test Generation
- 'Good' Test Case
- Gridworld Case Study
  - Agent types
  - Results
- Additional Agent Improvements
  - Q-Agent
  - EBM for spawn location
- Porting Agents

3D testing environment (Carla)

Introduction

- Addressing challenge of AV testing
- How do we detect software failures?
- Gain confidence in the correctness of the system?
- Simulation allows
  - Full control of the environment
  - Increase rare event frequency
  - Provide evidence to regulators
- How to effectively generate tests?
  - Can this be automated?
  - Can we still achieve interesting tests if automated?
- How much testing for sufficient confidence? And can we be more targeted?
  - 275 million miles road testing (12.5 years with 100 vehicles)
  - 7 billion miles of simulation achieved (Waymo)
- What are the current methods?
  - Can **agency** assist in test generation?
  - Does this method outperform random? At what cost?

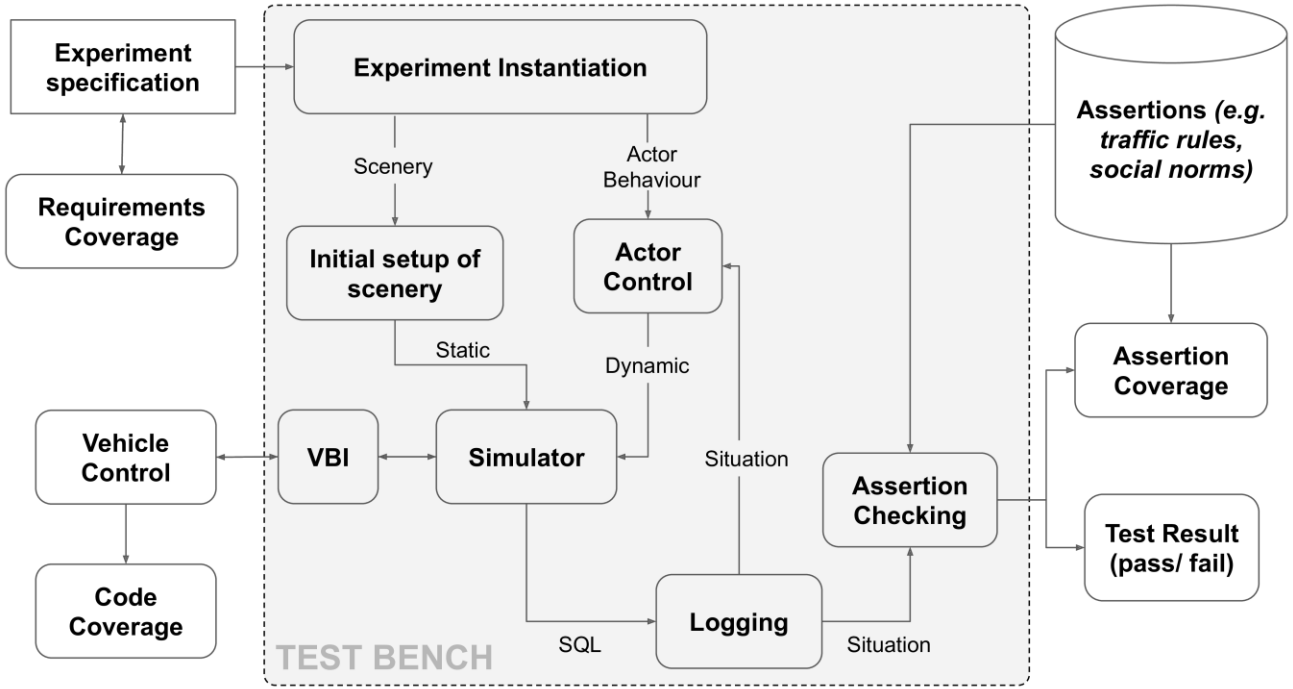
CAPRI Driverless POD



ROBOPILOT autonomous delivery truck

Research Questions

- Verification
  - Correctness in a system relative to requirements
  - Testing ensure behaviour does not differ from intention
- Simulation
  - Full control of environment
  - Provide evidence to regulators
- Testbench
  - Provide stimulus to DUV (Design under Verification)
  - Automated = more efficient
- Test generation
  - Random is useful
  - Selecting ***minimal effective*** set of tests
  - Automating valid test generation challenging
  - Automating realistic (interesting) test cases
- AV as DUV
  - Tests to interact with AV (change behaviour)
  - May conflict with mission goal
- Use of agency to verify the DUV
  - Software agents goals -> verification goals
  - Agents can interact/ coordinate
  - Agents sense environment
- Research questions
  - Can agents beat random test generation?
  - Can they generate 'good' test cases?



Agency-Directed Test Generation

Can test generation be automated through the use of agency?

Properties of a 'good' test case

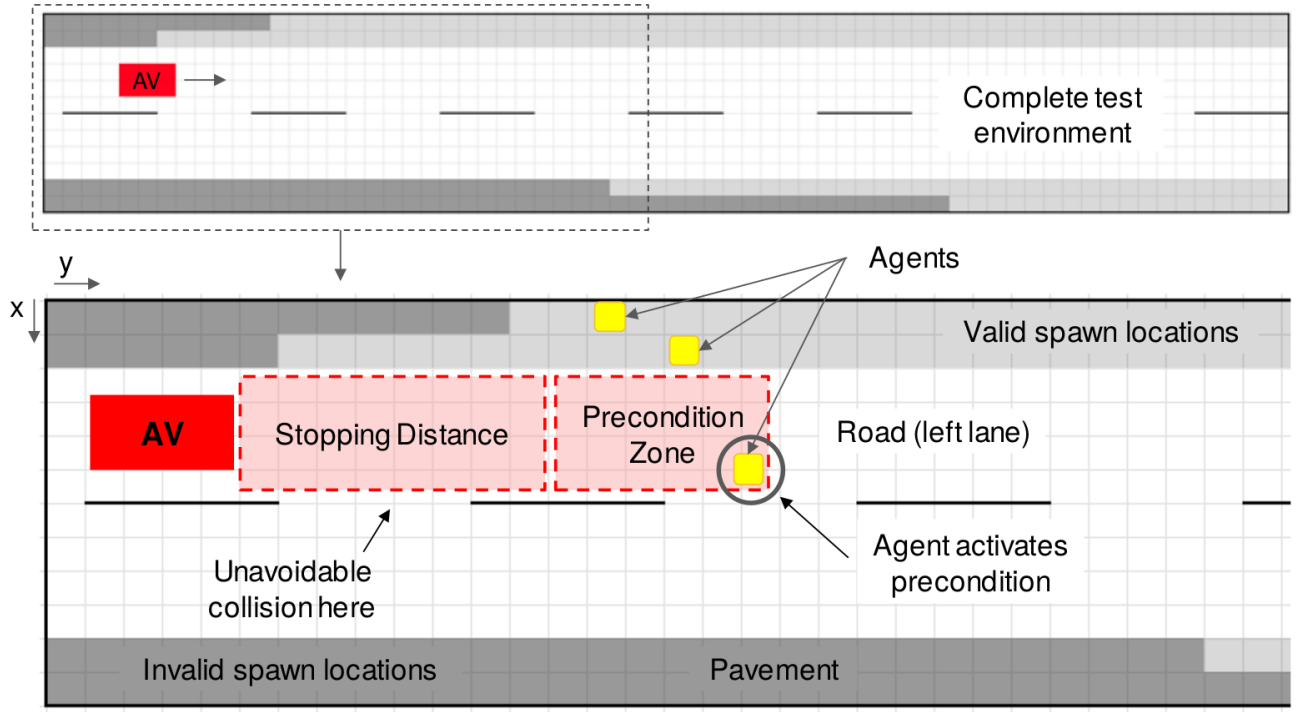
Optimum Test Case Property	Agent Success Metric
Effectiveness at detecting failures	**Accuracy** of test generation per episode
Efficiency in tests to reach verification goals	**Tests/tick** simulation
Economy of resource usage	**CPU Time** used and **LOC**
Robustness to changes	Use agent in **alternative environment**
Realistic representation of reality	Compare to **reality**

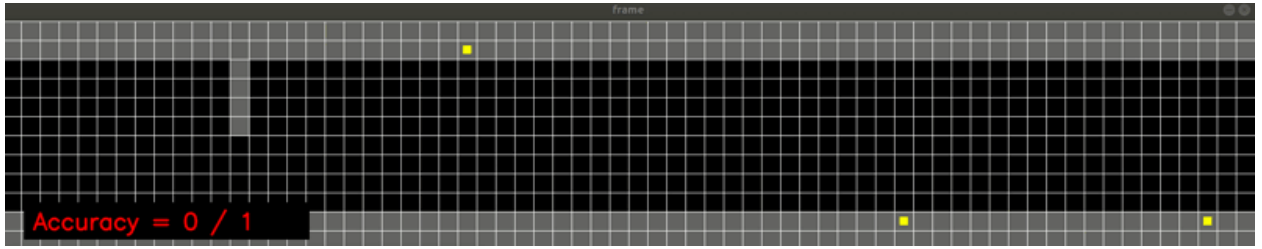
Methodology

- Compare **Agency-directed** test generation vs. **Random**

Test Environment for Agents

- Test Environment
  - Measure success
  - Develop agents
  - Python gridworld
- Pedestrian safety test
  - Euro-NCAP CPNA-25
- No AI in controller
  - AV moves fixed speed
- Dynamics
  - AV 30 km/h (~9m/s)
  - Agents 4.5 km/h (~1.5m/s)
- Episode terminates
  - Precondition zone - Successful test
  - AV exits road – Unsuccessful test
  - Repeated $1x10^4$





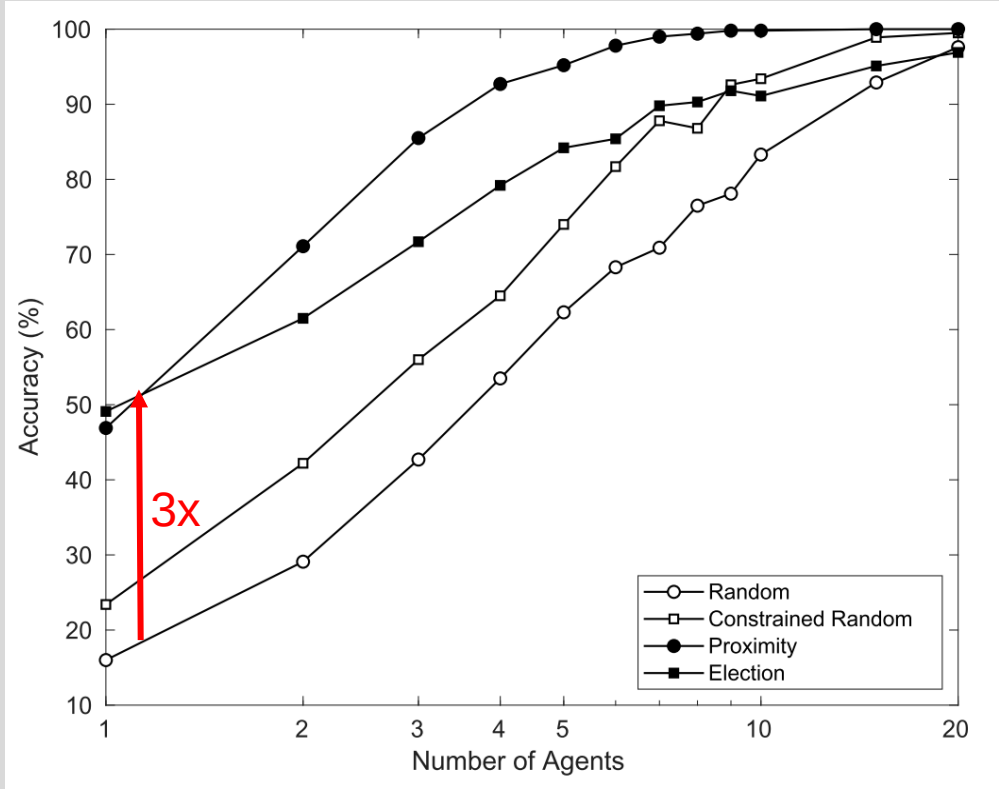
Python gridworld for pedestrians with random actions

Agent Types

Agent Type	Action
Gaussian (Random)	Move in **random direction** each sim tick
Constrained (Random)	Move along pavement, **randomly cross road**
Proximity (Agency)	Move along pavement, cross road when **vehicle is r<15**
Election (Agency)	As proximity but **elect single agent** which is closest

Results - Accuracy

- Easy to find tests with lots of agents
- Lots of agents = high resource usage
- Agency-directed more accurate
- Up to 3x more accurate than random
- 1x agency = 4x random

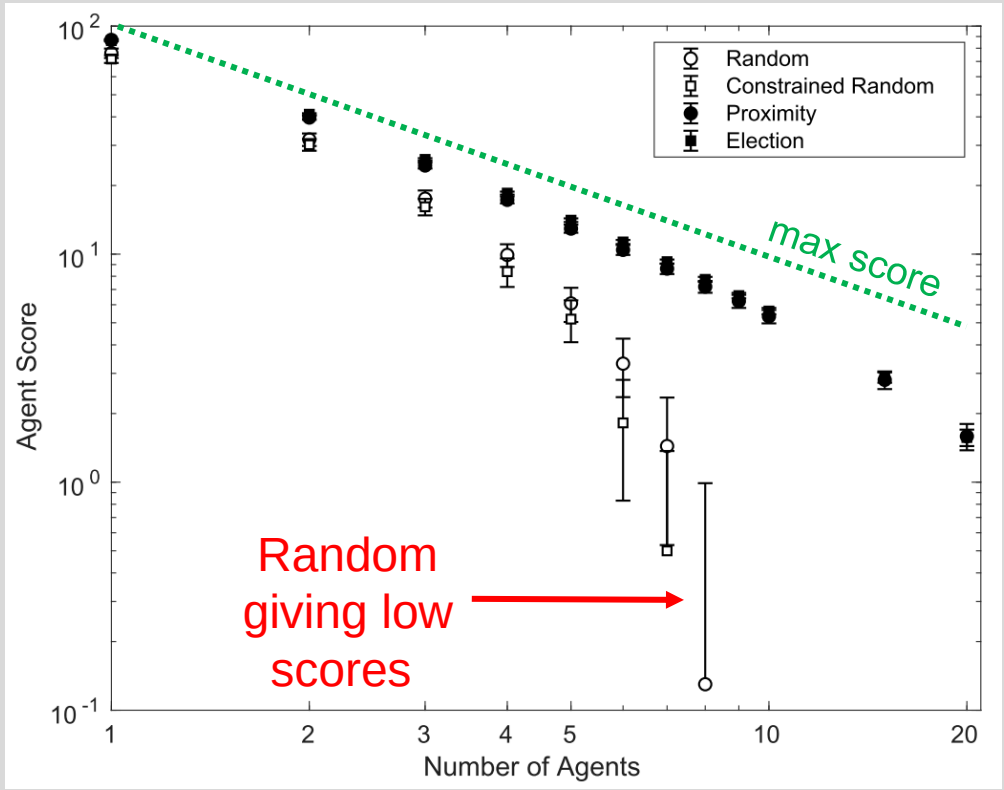


Agency-directed generate tests more successfully than random

Results - Realism

Realism Score

- Based on observation
  - Pedestrians time in road
- Agency-directed higher score
  - Less time in road

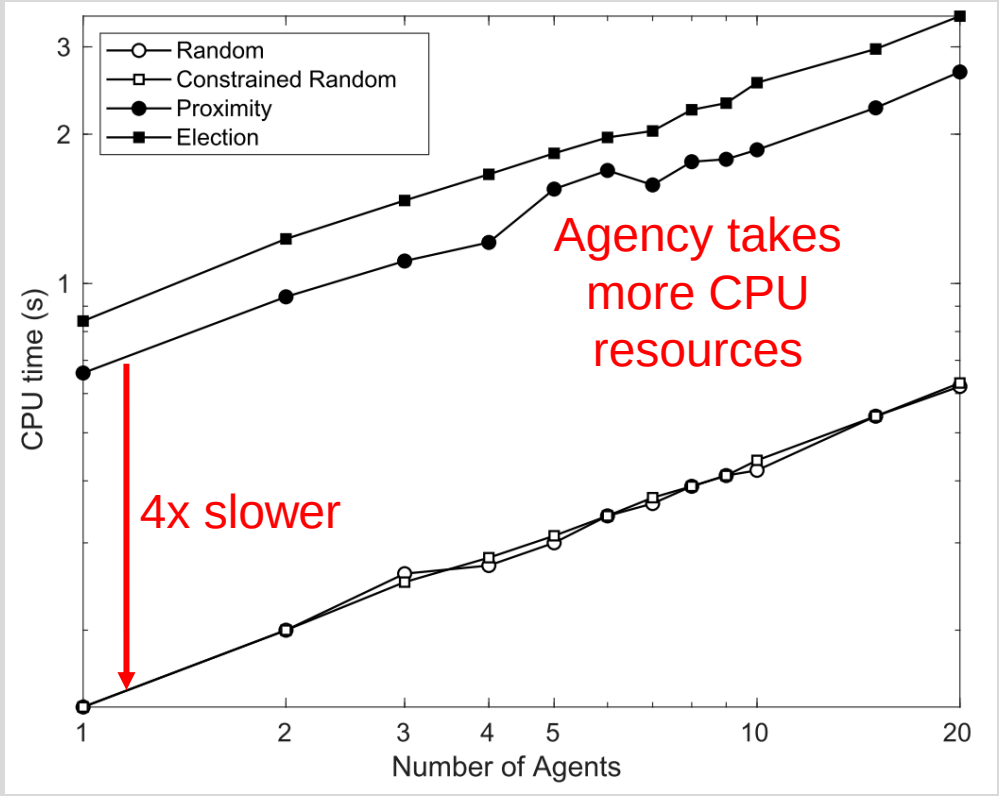


A lower score is a result of a longer test and more time spent in the road

Results - Economy

CPU Time

- Measure of action execution time
- Random very fast!
- Agency less economic

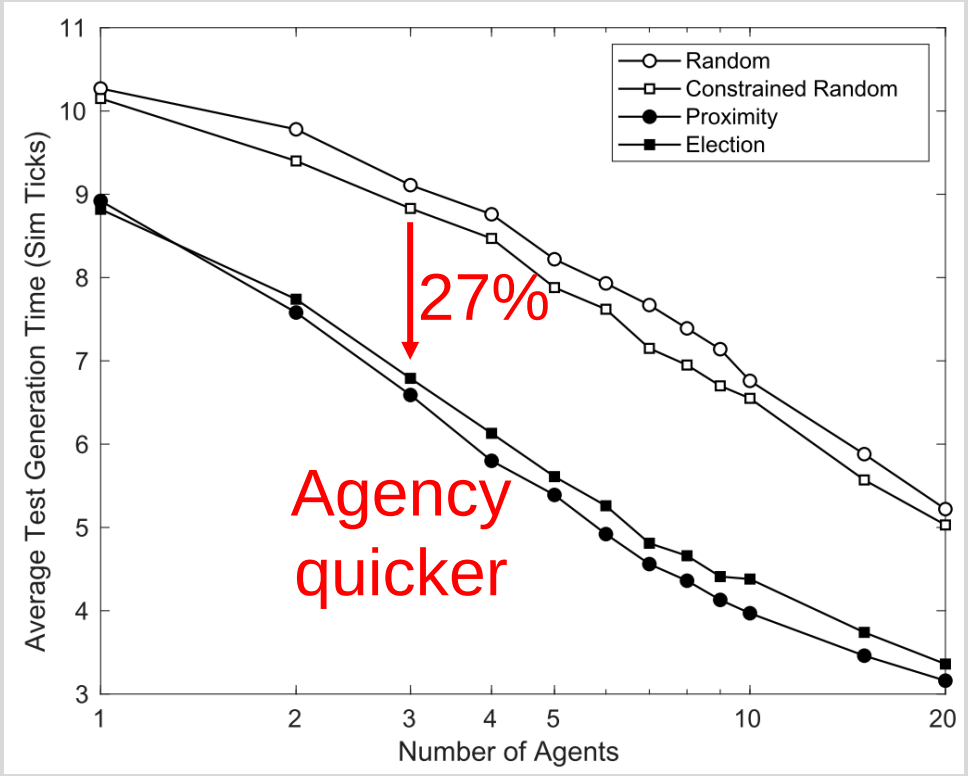


Random methods are faster than agency-directed methods

Results - Efficiency

Test Generation Time

- How long to generate a successful test on average?
- Random takes more sim ticks
- Fewer ticks may be important if simulation is expensive to run



Test generation time in simulation ticks

Results Summary

Method (n=3)	LOC	Accuracy (%)	CPU time (s)	Score	Sim ticks
Random	23	42.7	0.09	0.56	9.11
Constrained	82	56.0	0.08	0.52	8.83
Proximity	86	85.5	0.37	0.78	6.79
Election	235	71.7	0.49	0.83	6.59
	Economy	*Effectiveness*	*Economy*	*Realism*	*Efficiency*

Agents for Verification

- Agency-based test generation methods get to **useful** test cases **quicker**

Chance, G., et al., (2019). An Agency-Directed Approach to Test Generation for Simulation-based Autonomous Vehicle Verification. http://arxiv.org/abs/1912.05434 https://ieeexplore.ieee.org/abstract/document/9176833

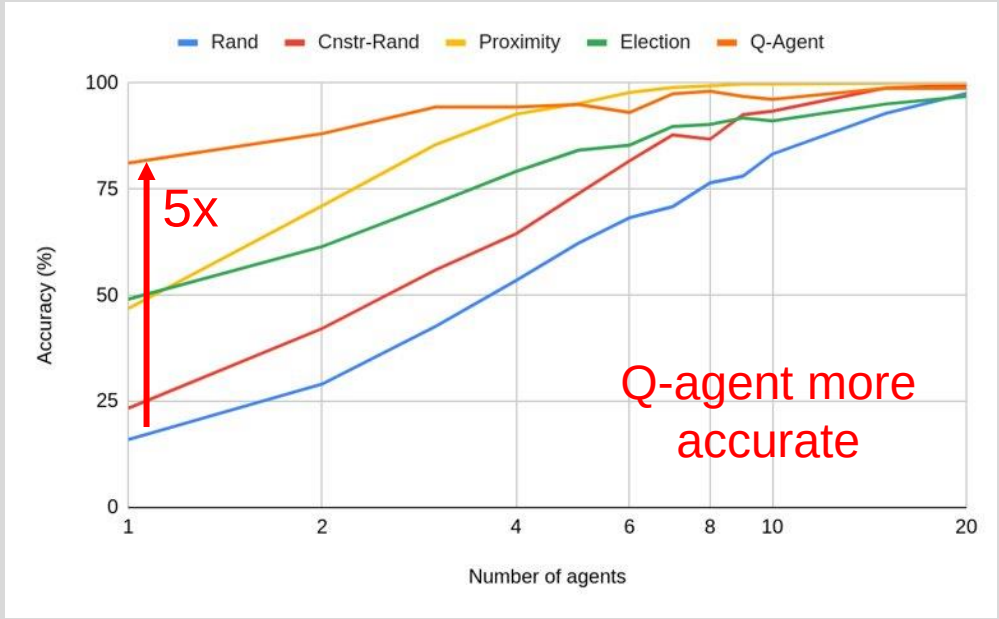
Additional Agent Improvements

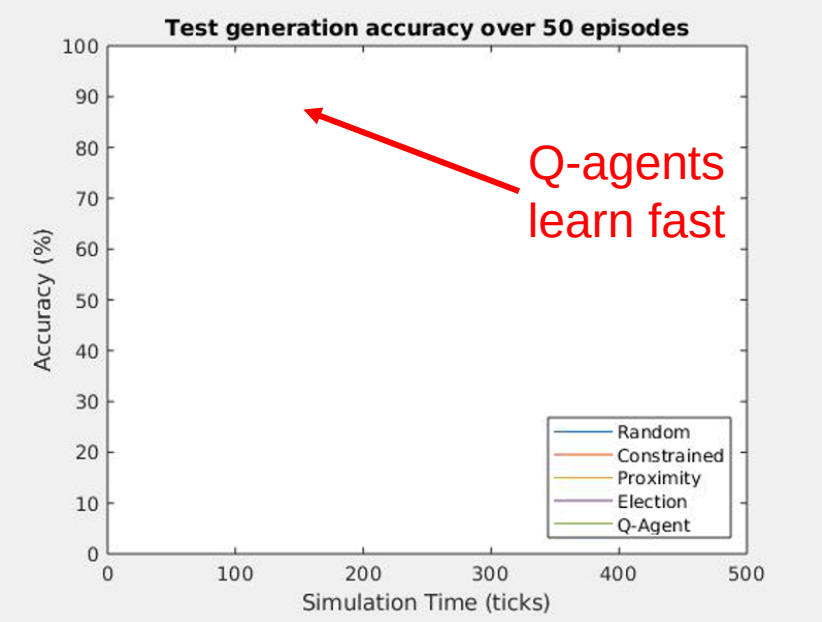
Making agents more effective and robust

Improving Agent Learning

- General agent framework
  - Reinforcement learning
  - Approximate Q-learning
- Functional approximation of environment
  - e.g. is AV moving away/toward agent?
- Agents learn quickly ~10 episodes
- Adapt to other assertions
  - Change agent(s) goal state

Method	LOC	Accuracy (%)	CPU time (s)	Sim ticks
Random	23	42.7	0.09	9.11
Constrained	82	56.0	0.08	8.83
Proximity	86	85.5	0.37	6.79
Election	235	71.7	0.49	6.59
Q-agent	251	94.4	1.71	7.31





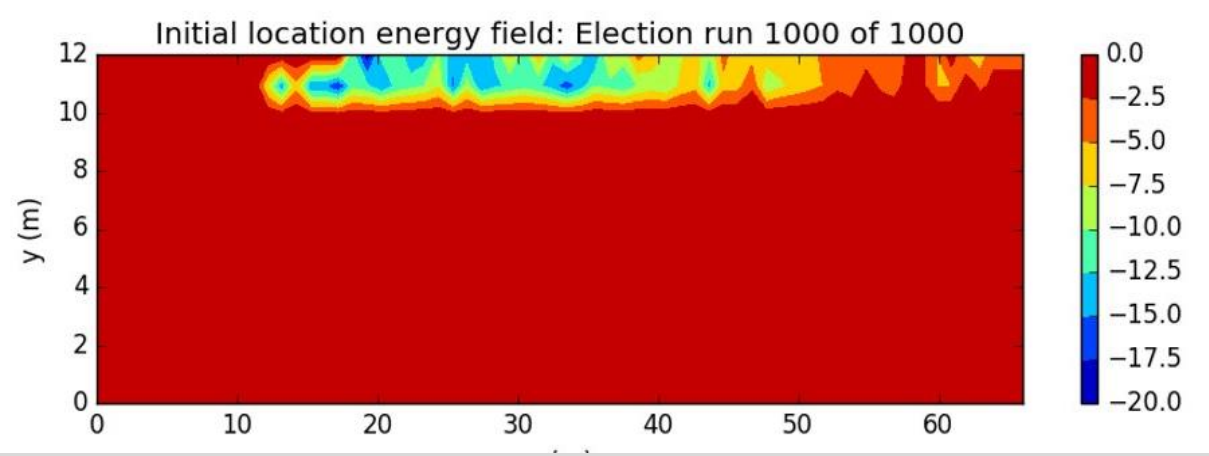
Improving Agent Placement

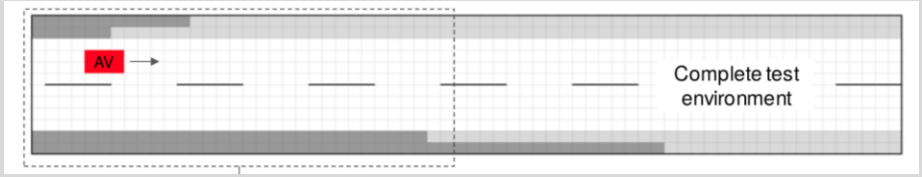
- Energy Based Model
- Previously random start
  - Improved agent accuracy
  - Learnt good start locations

- Find F(x,y) that maximises reward
- Learn Energy Field
  - Build from previous success
  - Exploit field
  - Improves **accuracy**

- Results
  - Random accuracy improvement
  - Agents improved with cost

- Future
  - Estimate EBM on environment, actors





	CPU Time (s)		
	Rand	Prox.	Elec.
No Field	2.05	9.04	12.3
Energy Field	1.98	13.7	17.3
CPU time (%) =	~	+51.5%	+40.9%
Accuracy (%) =	+5.0%	+10.8%	+13.3%

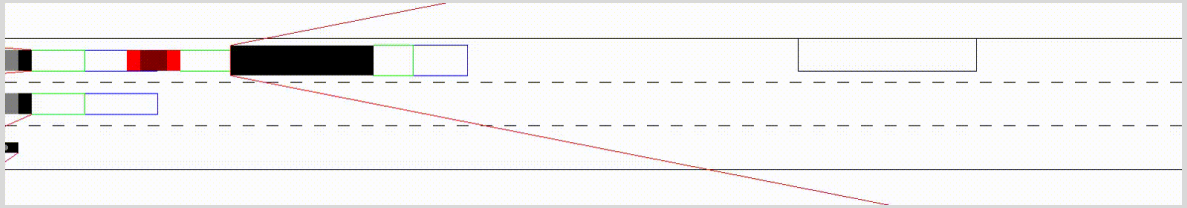
Improving Environment

Improved environment:
- Multi-agent development (CAV-GYM)
- Continuous (non-gridworld)
- Dynamic braking/assertion zones
- Occluded actors
- Easier ML

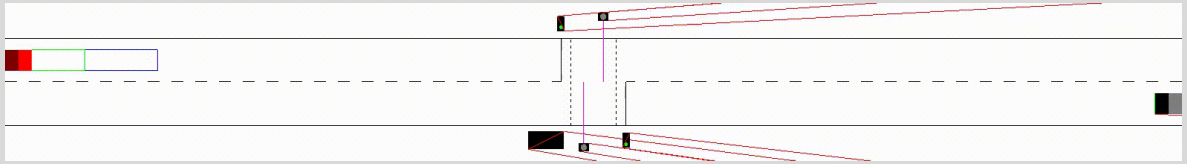
Improved Vehicle Agent
- Used Q-learning
- More capable, less coding
- Feature-based weight updates

Future: game theoretic approach

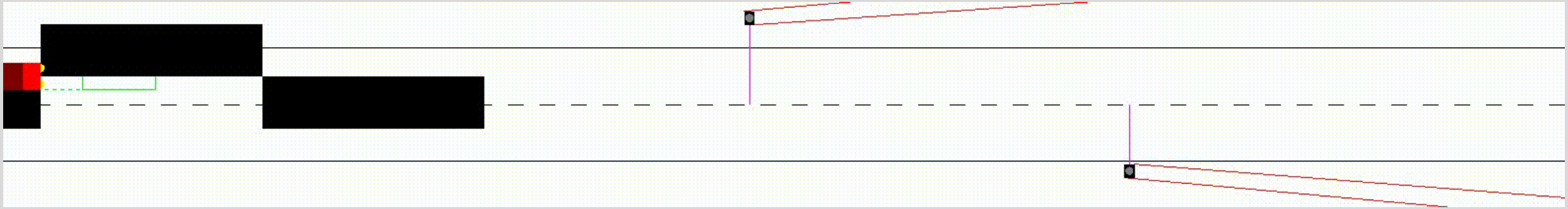
https://github.com/TSL-UOB/CAV-Gym



Bus stop



Pedestrian crossing

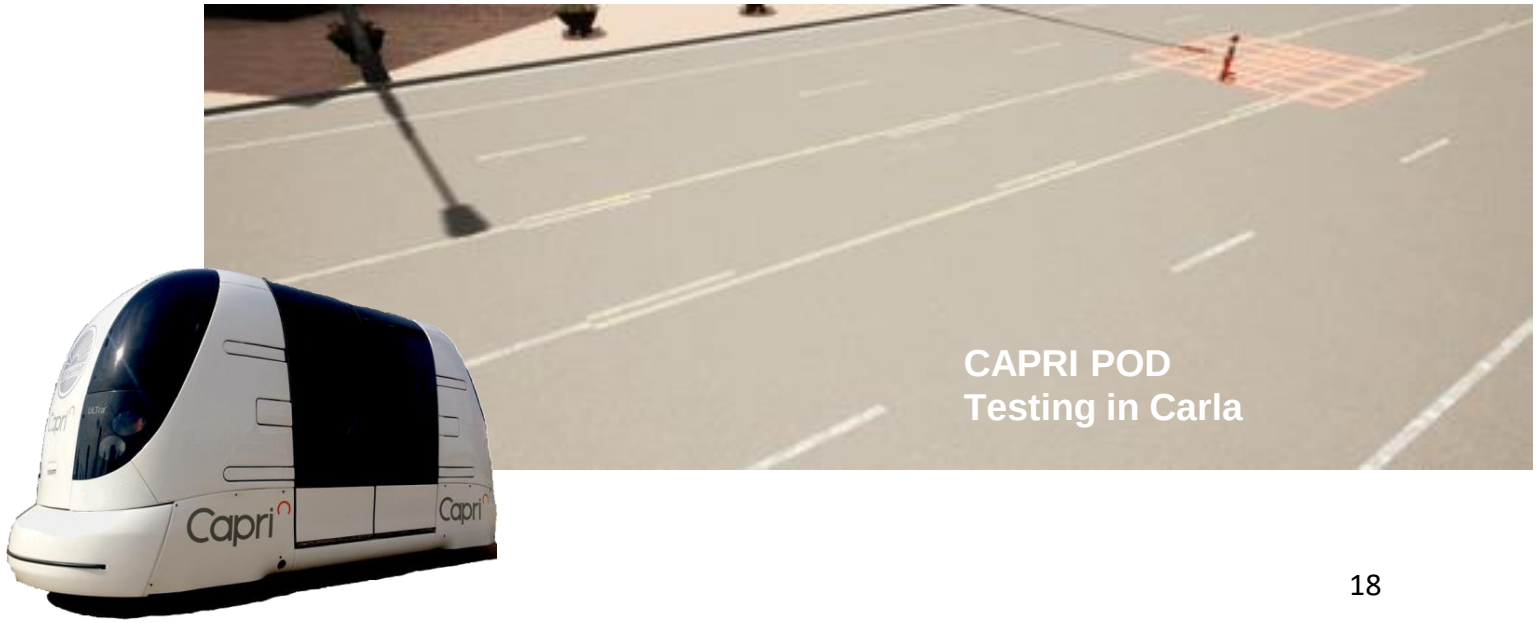


Q-learning Vehicle agent

Porting Trained Agents

Full 3D environment

- Ported agents
- Training agents "offline"
  - Use 2D world for dev
  - Optimise
  - Train faster
- CAPRI integration
  - Proved verification agent
  - Real POD controller



Thank you for your attention :)

Hopefully you learnt about:

- good test cases
- How software agents can beat random test generation
- How Q-learning and EBM can improve agent performance

https://github.com/TSL-UOB/CAV-Gym

I look forward to your questions

greg.chance@bristol.ac.uk



Thanks to collaborators: Kerstin Eder, Abanoub Ghobrial and Kevin McAreavey

University of BRISTOL

EPSRC Engineering and Physical Sciences Research Council

Innovate UK

brl Bristol Robotics Laboratory