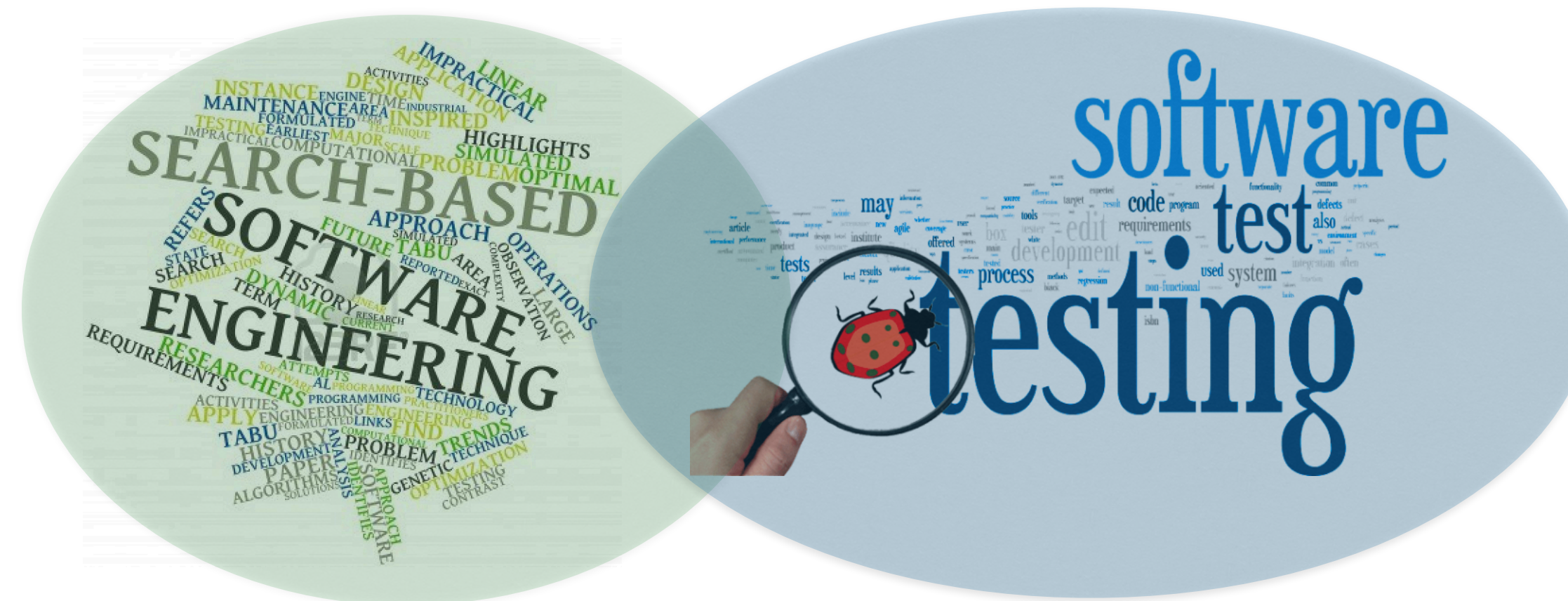


Search-Based Software Testing Tool Competition 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Alessio Gambi

Passau University



Fiorella Zampetti

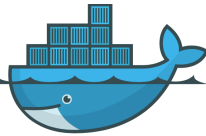
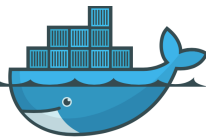
University of Sannio



Vincenzo Riccio

University of Lugano

History SBST Tool Competition

	Year	Venue	Coverage tool	Mutation Tool	#CUTs	#Projects	#Participants (+ baseline)	Statistical Tests	
Round 1	2013	ICST	Cobertura	Javalanche	77	5	2	✗	
Round 2	2014	FITTEST	JaCoCo	PItest	63	9	4	✗	
Round 3	2015	SBST	JaCoCo	PItest	63	9	8	✗	
Round 4	2016	SBST	DEFECT4J (Real Faults)		68	5	4	✗	
Round 5	2017	SBST	JaCoCo	PItest + Our Env.	69	8	2 (+ 2)	✓	
Round 6	2018	SBST	JaCoCo	PItest + Our Env.	59	7	2 (+ 2)	✓ + combined analysis	
Round 7	2019	SBST	JaCoCo	PItest + Our Env.	69	8	2 (+ 2)	✓ + combined analysis	+  docker
Round 8	2020	SBST	JaCoCo	PItest + Our Env.	69	8	1 (+ 1)	✓ + combined analysis	+  docker

SBST Tool Competition - 2021

What is New?

Java tool competition: As for recent years, we invite researchers to participate in the competition with their unit test generation tool for **Java**. Tools will be evaluated against a benchmark with respect to code coverage and mutation score.

Cyber-physical systems (CPS) testing competition: In addition to the traditional Java tool competition, we also organize a CPS testing competition on self-driving cars simulation environments. Specifically, in collaboration with the BeamNG research team (<https://beamng.gmbh/research/>), this competition focuses on the

- Generation of scenarios using BeamNG self-driving cars simulator

Class Under Test (CUT)

```
class Triangle {
    int a, b, c; //sides
    String type = "NOT_TRIANGLE";

    Triangle (int a, int b, int c){...}

    void computeTriangleType() {
        1. if (a == b) {
        2.     if (b == c)
        3.         type = "EQUILATERAL";
        4.     else
        5.         type = "ISOSCELES";
        6.     } else {
        7.         if (a == c) {
        8.             type = "ISOSCELES";
        9.         } else {
        10.            if (b == c)
        11.                type = "ISOSCELES";
        12.            else
        13.                type = "SCALENE";
        14.        }
        15.    }
    }
}
```

Test Case

```
@Test
public void test(){
    // Constructor (init)
    // Method Calls
    // Assertions (check)
}
```



```
@Test
public void test(){
    Triangle t = new Triangle (1,2,3);
    t.computeTriangleType();
    String type = t.getType();
    assertTrue(type.equals("SCALENE"));
}
```

Figure 1: Example of test generation for a simple Java class.

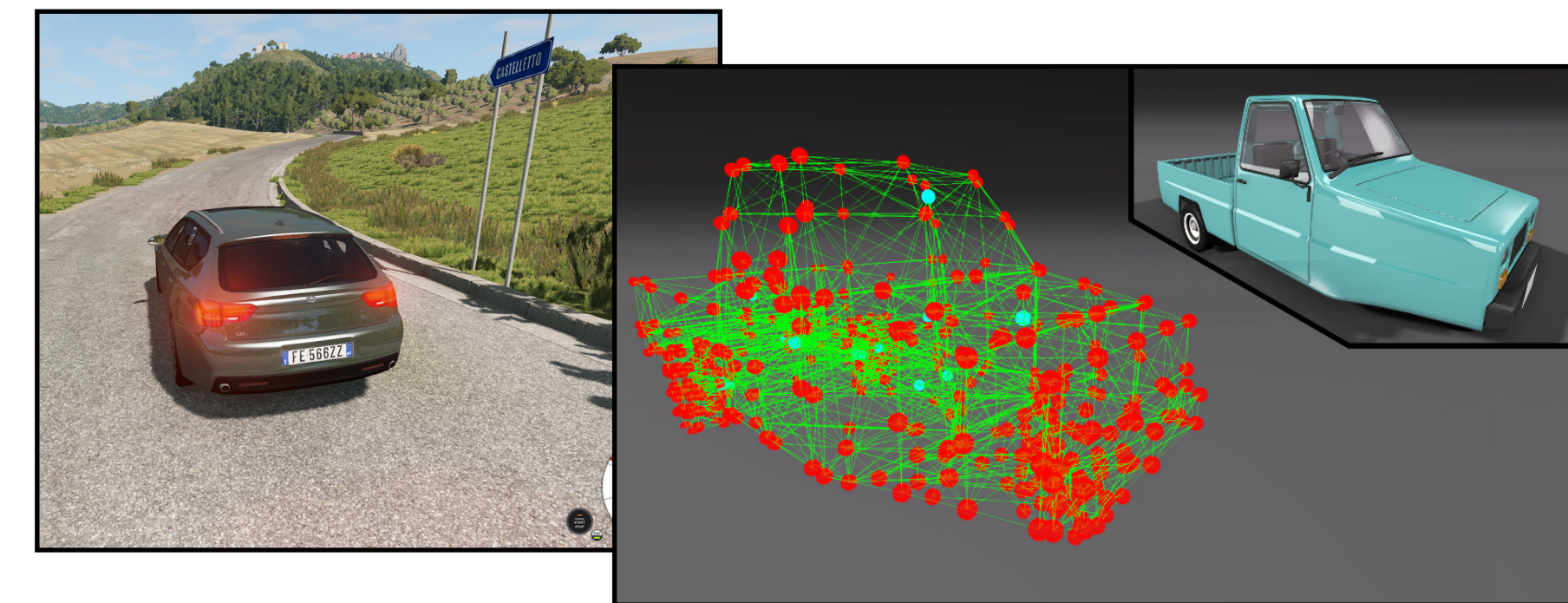


Figure 2: Example of CPS testing tool simulation environment.

New!!!

SBST Tool Competition - 2021

What is New?

Java tool competition: As of 2017, 100% of Java code is written using IDEs.

Cyber-
tradition
competit
collabora
[research/](#)

- Generat

10 Tools Participating to the Competition

- Five times more tools than last 2020!!!
- 2 Tools from Industrial Organizations!!!



Class Under Test (CUT)

```
class Triangle {
    int a, b, c;
    String
```

Test Case

```
Test
public void test(){
    // Constructor (init)
    // Method Calls
    // Assertions (check)
```

```
void test(){
    Triangle t = new Triangle (1,2,3);
    TriangleType type = t.getType();
    assertTrue(type.equals("SCALENE"));
}
```

simple Java class.



SBST Tool Competition - 2021

Co-chairs 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Fiorella Zampetti

University of Sannio



Alessio Gambi

Passau University



Vincenzo Riccio

University of Lugano



Class Under Test (CUT)

```
class Triangle {  
  int a, b, c; //sides  
  String type = "NOT_TRIANGLE";  
  
  Triangle (int a, int b, int c){...}  
  
  void computeTriangleType() {  
    1. if (a == b) {  
    2.   if (b == c)  
    3.     type = "EQUILATERAL";  
    4.   else  
    5.     type = "ISOSCELES";  
    6.   } else {  
    7.     if (a == c) {  
    8.       type = "ISOSCELES";  
    9.     } else {  
    10.      if (b == c)  
    11.        type = "ISOSCELES";  
    12.      else  
    13.        type = "SCALENE";  
    14.    }  
    15.  }  
  }  
}
```

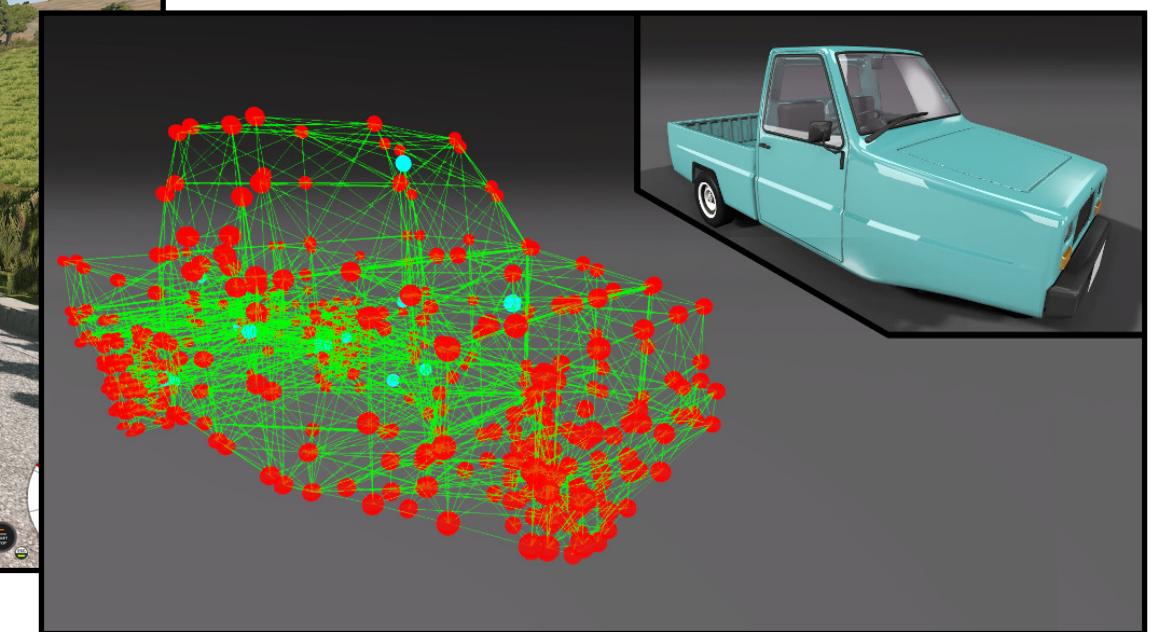
Test Case

```
@Test  
public void test(){  
  // Constructor (init)  
  // Method Calls  
  // Assertions (check)  
}
```



```
@Test  
public void test(){  
  Triangle t = new Triangle (1,2,3);  
  t.computeTriangleType();  
  String type = t.getType();  
  assertTrue(type.equals("SCALENE"));  
}
```

Java tool competition



Cyber-physical systems (CPS) testing competition

SBST Tool Competition - 2021

Co-chairs 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Fiorella Zampetti

University of Sannio



Alessio Gambi

Passau University



Vincenzo Riccio

University of Lugano

Class Under Test (CUT)

```
class Triangle {  
    int a, b, c; //sides  
    String type = "NOT_TRIANGLE";  
  
    Triangle (int a, int b, int c){...}  
  
    void computeTriangleType() {  
        1. if (a == b) {  
            2.     if (b == c)  
            3.         type = "EQUILATERAL";  
            else  
            4.         type = "ISOSCELES";  
        } else {  
            5.     if (a == c) {  
            6.         type = "ISOSCELES";  
            } else {  
            7.         if (b == c)  
            8.             type = "ISOSCELES";  
            else  
            9.             type = "SCALENE";  
        }  
    }  
}
```

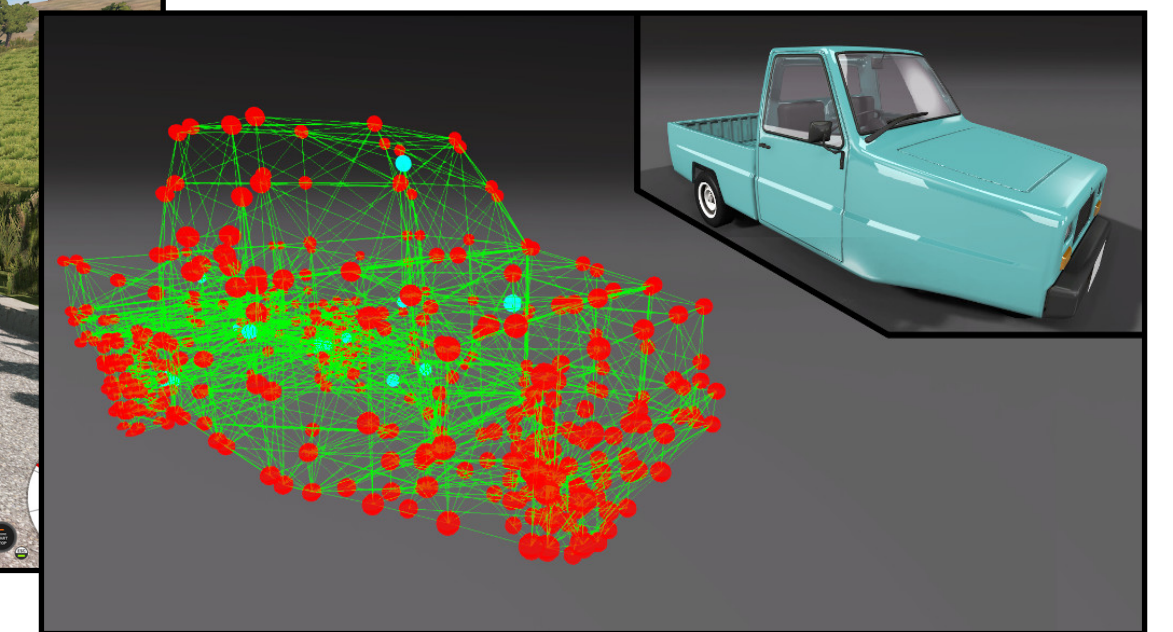
Test Case

```
@Test  
public void test(){  
    // Constructor (init)  
    // Method Calls  
    // Assertions (check)  
}
```



```
@Test  
public void test(){  
    Triangle t = new Triangle (1,2,3);  
    t.computeTriangleType();  
    String type = t.getType();  
    assertTrue(type.equals("SCALENE"));  
}
```

Java tool competition



Cyber-physical systems (CPS) testing competition

SBST Tool Competition - 2021

Co-chairs 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Fiorella Zampetti

University of Sannio



Alessio Gambi

Passau University



Vincenzo Riccio

University of Lugano

Class Under Test (CUT)

```
class Triangle {
  int a, b, c; //sides
  String type = "NOT_TRIANGLE";

  Triangle (int a, int b, int c){...}

  void computeTriangleType() {
    1. if (a == b) {
    2.   if (b == c)
    3.     type = "EQUILATERAL";
    4.   else
    5.     type = "ISOSCELES";
    6.   } else {
    7.     if (a == c) {
    8.       type = "ISOSCELES";
    9.     } else {
    10.      if (b == c)
    11.        type = "ISOSCELES";
    12.      else
    13.        type = "SCALENE";
    14.    }
    15.  }
  }
}
```

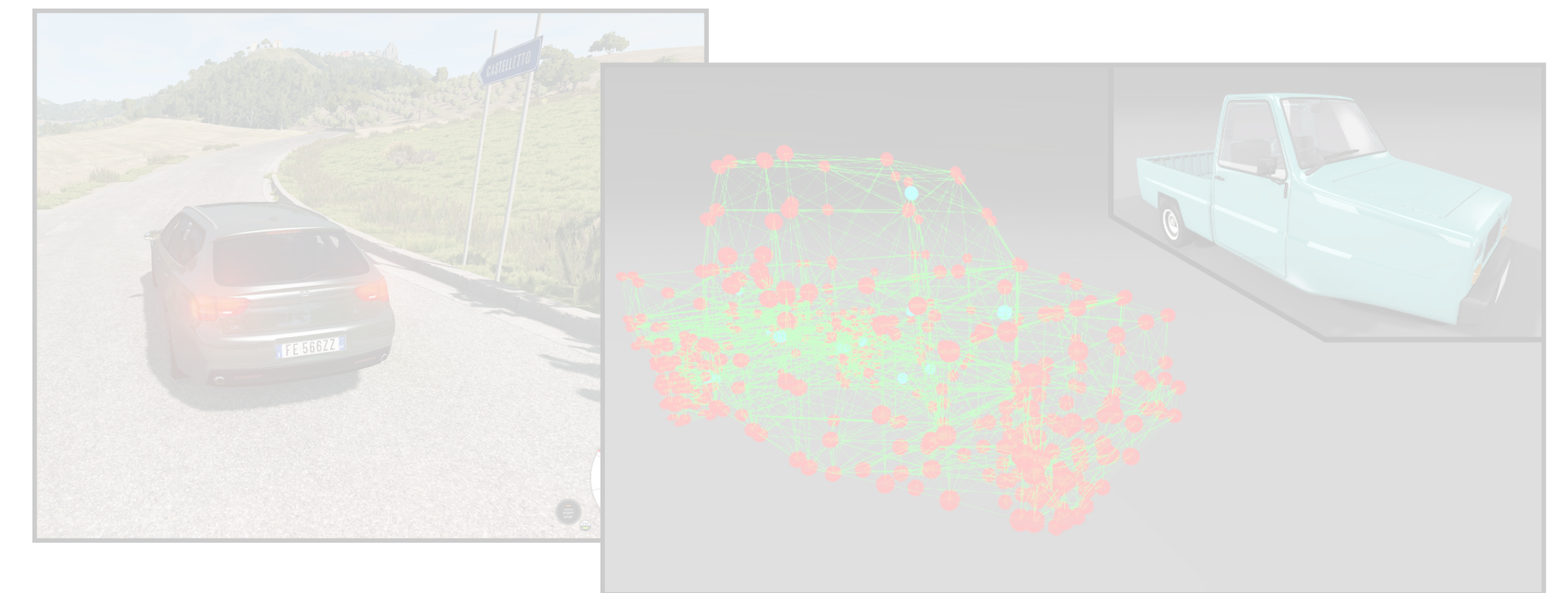
Test Case

```
@Test
public void test(){
  // Constructor (init)
  // Method Calls
  // Assertions (check)
}
```



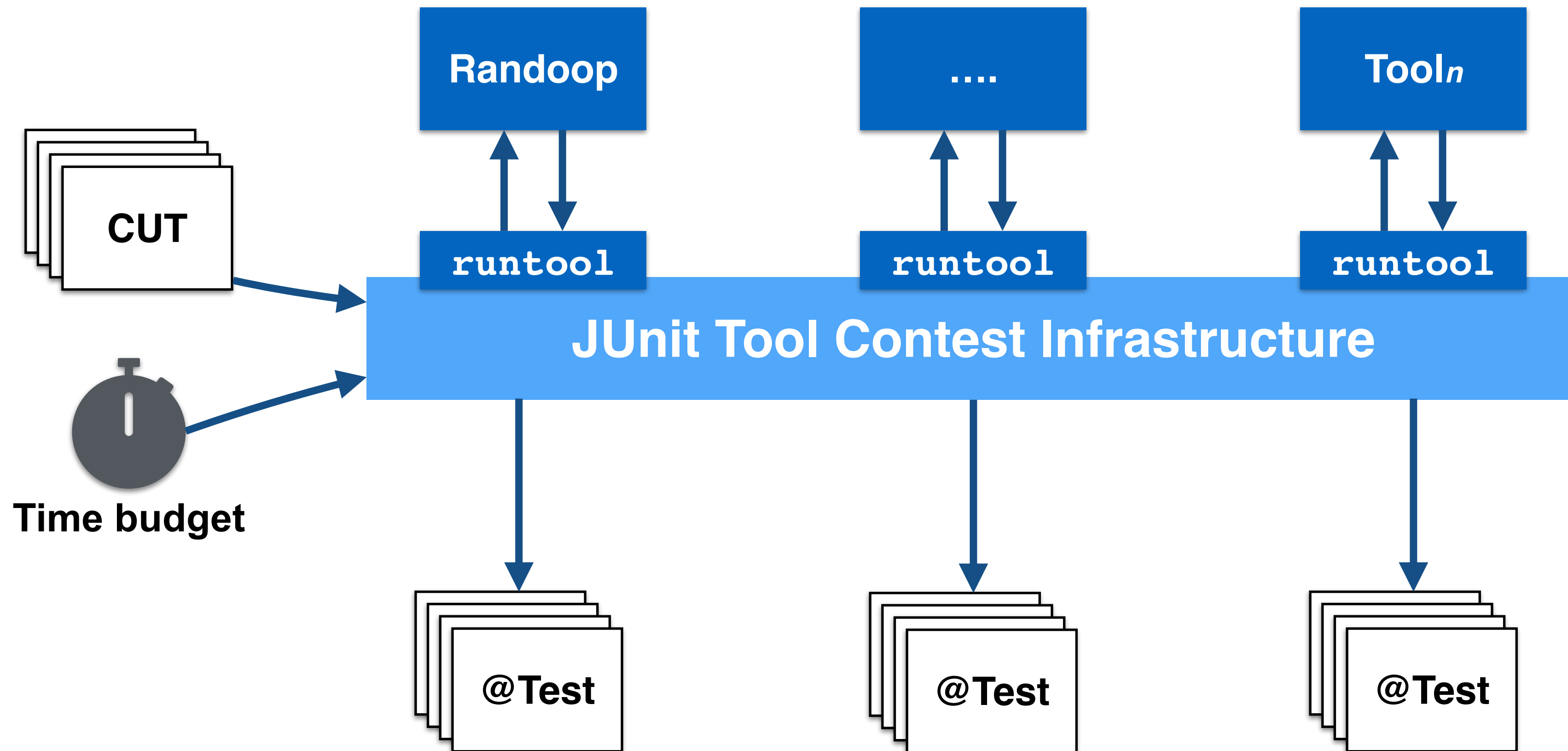
```
@Test
public void test(){
  Triangle t = new Triangle (1,2,3);
  t.computeTriangleType();
  String type = t.getType();
  assertTrue(type.equals("SCALENE"));
}
```

Java tool competition

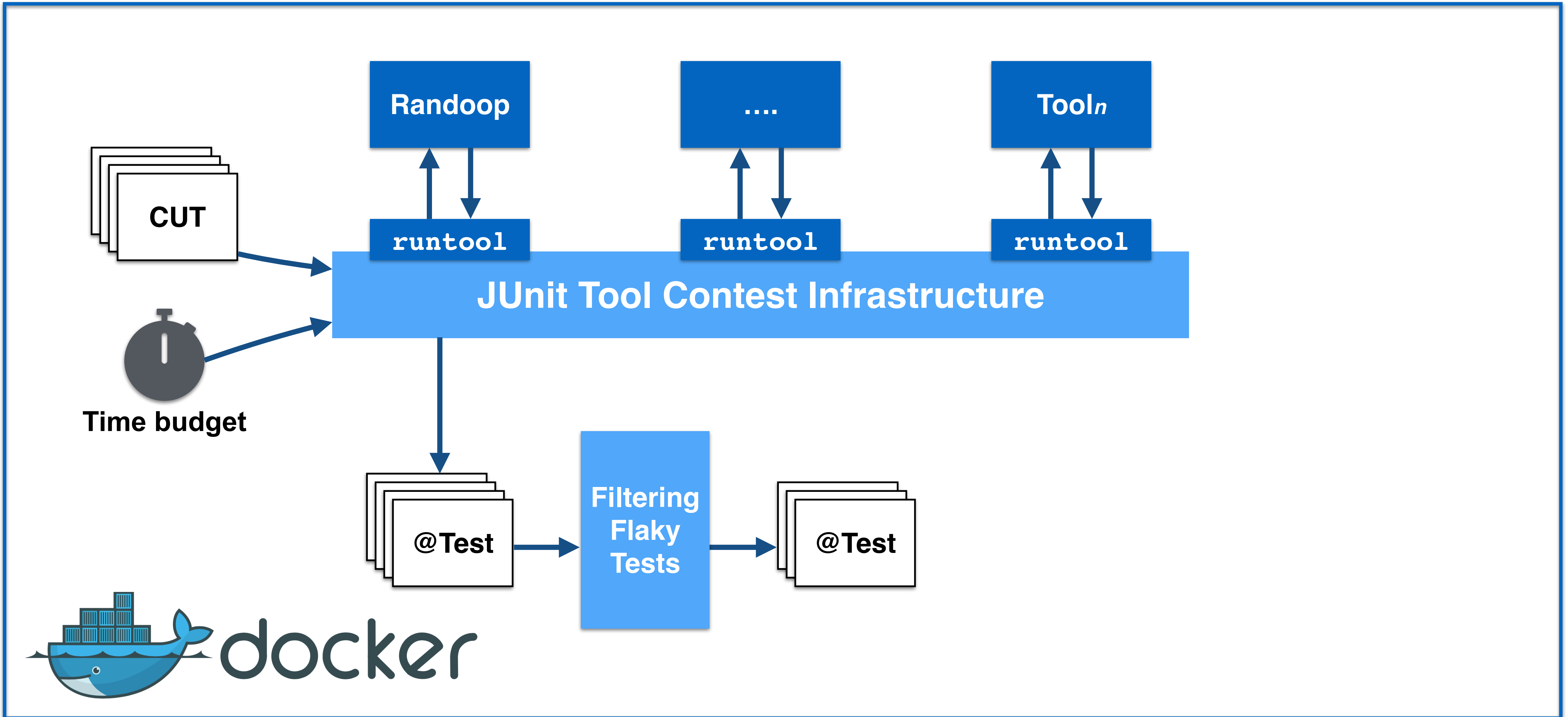


Cyber-physical systems (CPS) testing competition

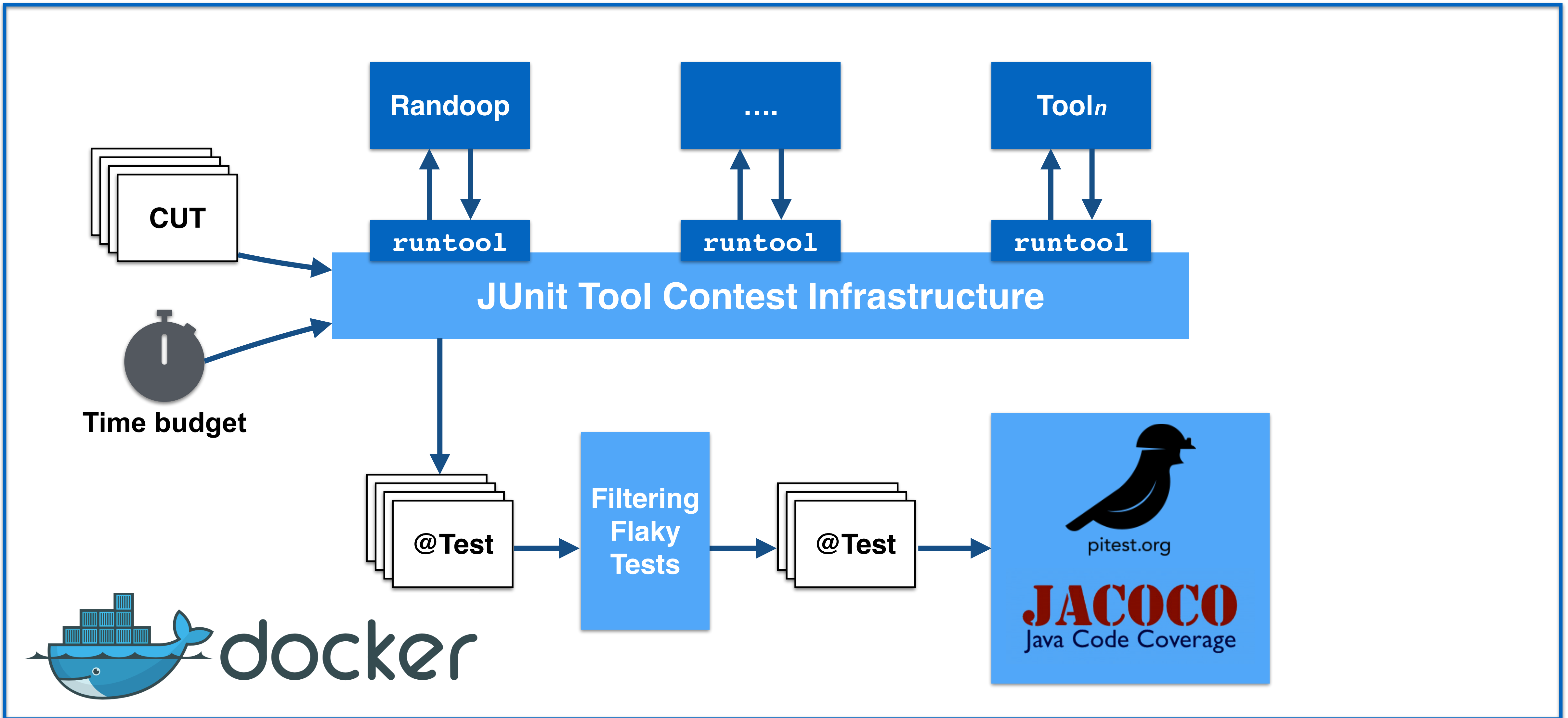
Java tool competition Infrastructure



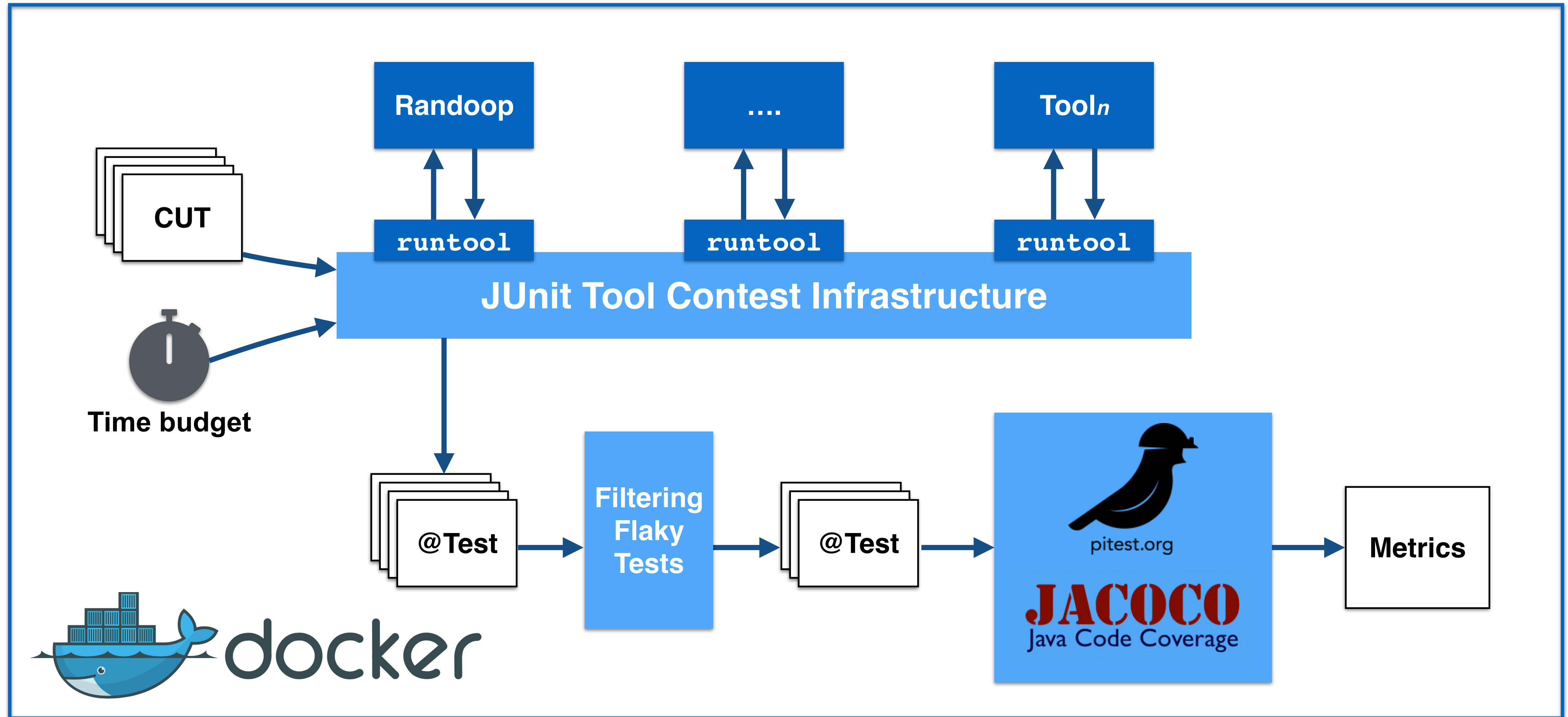
Java tool competition Infrastructure



Java tool competition Infrastructure



Java tool competition Infrastructure



Scoring Formula

$$\text{covScore}(T, B, C, R) = \mathbf{1} \times \text{Cov}_i + \mathbf{2} \times \text{Cov}_b + \mathbf{4} \times \text{Cov}_m$$

$$\text{tScore}(T, B, C, R) = \text{covScore}(T, B, C, R) \times \min \left(1, \frac{2 \times B}{\text{genTime}} \right)$$

$$\text{Score}(T, B, C, R) = \text{tScore}(T, B, C, R) + \text{penalty}(T, B, C, R)$$

T = Generated Test

B = Search Budget

C = Class under test

R = independent Run

Cov_i = statement coverage

Cov_b = branch coverage

Cov_m = Strong Mutation

genTime = generation time

penalty = percentage of flaky test
and non-compiling tests

<https://github.com/JUnitContest/junitcontest>

Class Under Test (CUT)

```
class Triangle {
    int a, b, c; //sides
    String type = "NOT_TRIANGLE";

    Triangle (int a, int b, int c){...}

    void computeTriangleType() {
        1. if (a == b) {
        2.     if (b == c)
        3.         type = "EQUILATERAL";
        4.     else
        5.         type = "ISOSCELES";
        6. } else {
        7.     if (a == c) {
        8.         type = "ISOSCELES";
        9.     } else {
        10.        if (b == c)
        11.            type = "ISOSCELES";
        12.        else
        13.            type = "SCALENE";
        14.    }
        15. }
    }
}
```

Test Case

```
@Test
public void test(){
    // Constructor (init)
    // Method Calls
    // Assertions (check)
}
```



```
@Test
public void test(){
    Triangle t = new Triangle (1,2,3);
    t.computeTriangleType();
    String type = t.getType();
    assertTrue(type.equals("SCALENE"));
}
```

Benchmark Projects

- **Selection criteria**
 - GitHub repositories
 - Project builds using Maven or Gradle
 - Contains JUnit 4 test suite

Benchmark Projects

- **Selection criteria**
 - GitHub repositories
 - Project builds using Maven or Gradle
 - Contains JUnit 4 test suite
- **6 projects selected**

Guava

<https://github.com/google/guava>

Seata

<https://github.com/seata/seata>

Okio

<https://github.com/square/okio>

Spoon

<https://github.com/INRIA/spoon/>

FastJSON

<https://github.com/alibaba/fastjson>

Weka

<https://github.com/Waikato/weka-3.8>

Contest Methodology

Search budgets

**30
seconds**

2 min.

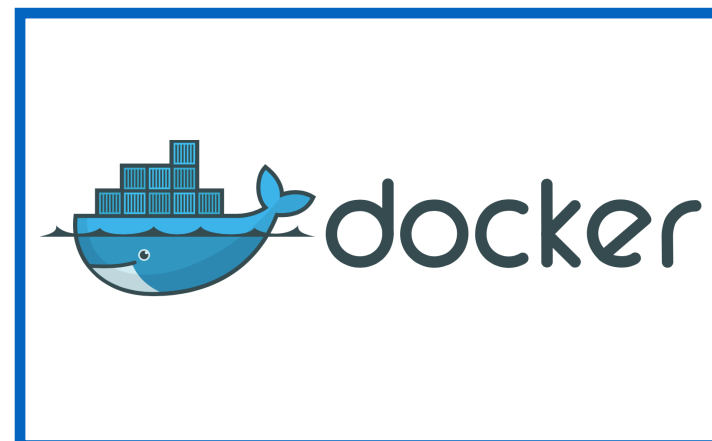
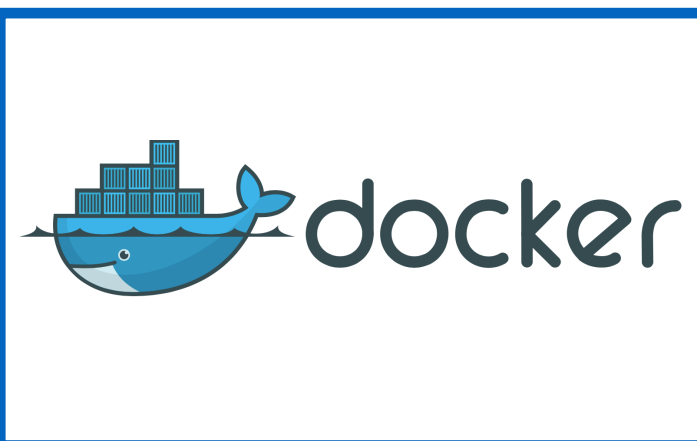
Classes under test

98 classes

Repetitions

10 repetitions

Execution environment



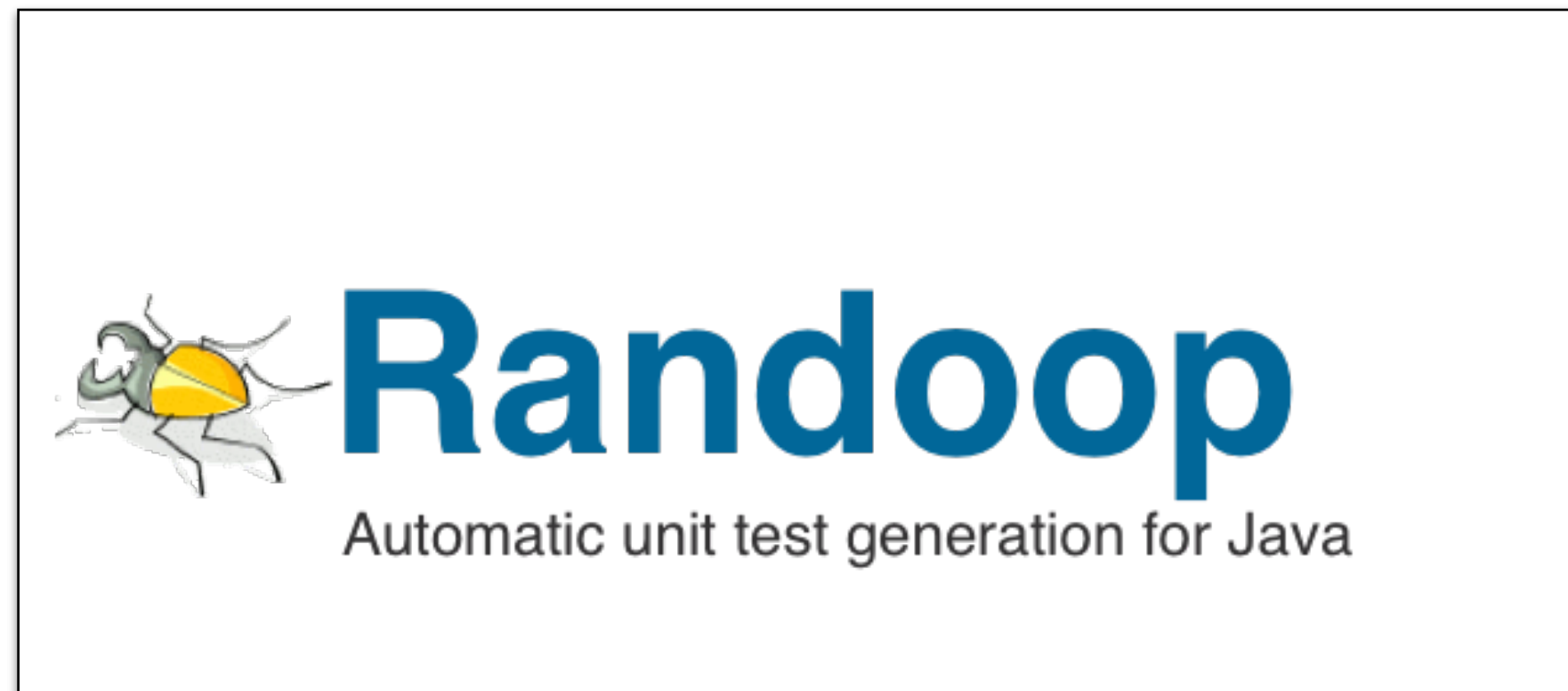
Statistical analysis

Friedman's test

Post-hoc Conover

The Tools

Baseline



V.S.

Competitors

EVASUITE

UtBot

Kex

EVASUITE - DSE

Results (1)

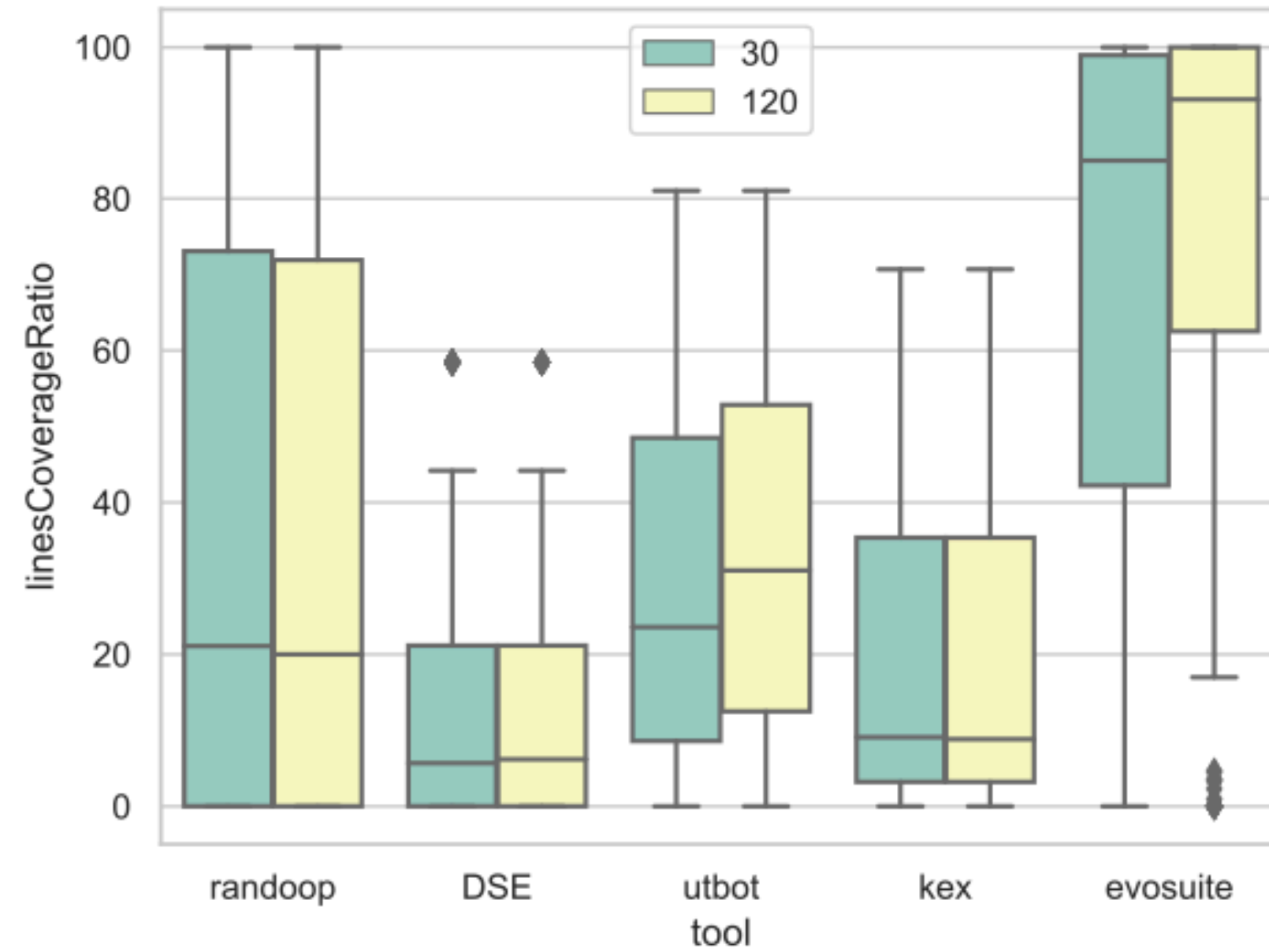


Fig. 1: Line Coverage for Randoop, Evosuite(DSE), Utbot, Kex and Evosuite for 30 and 120 seconds.

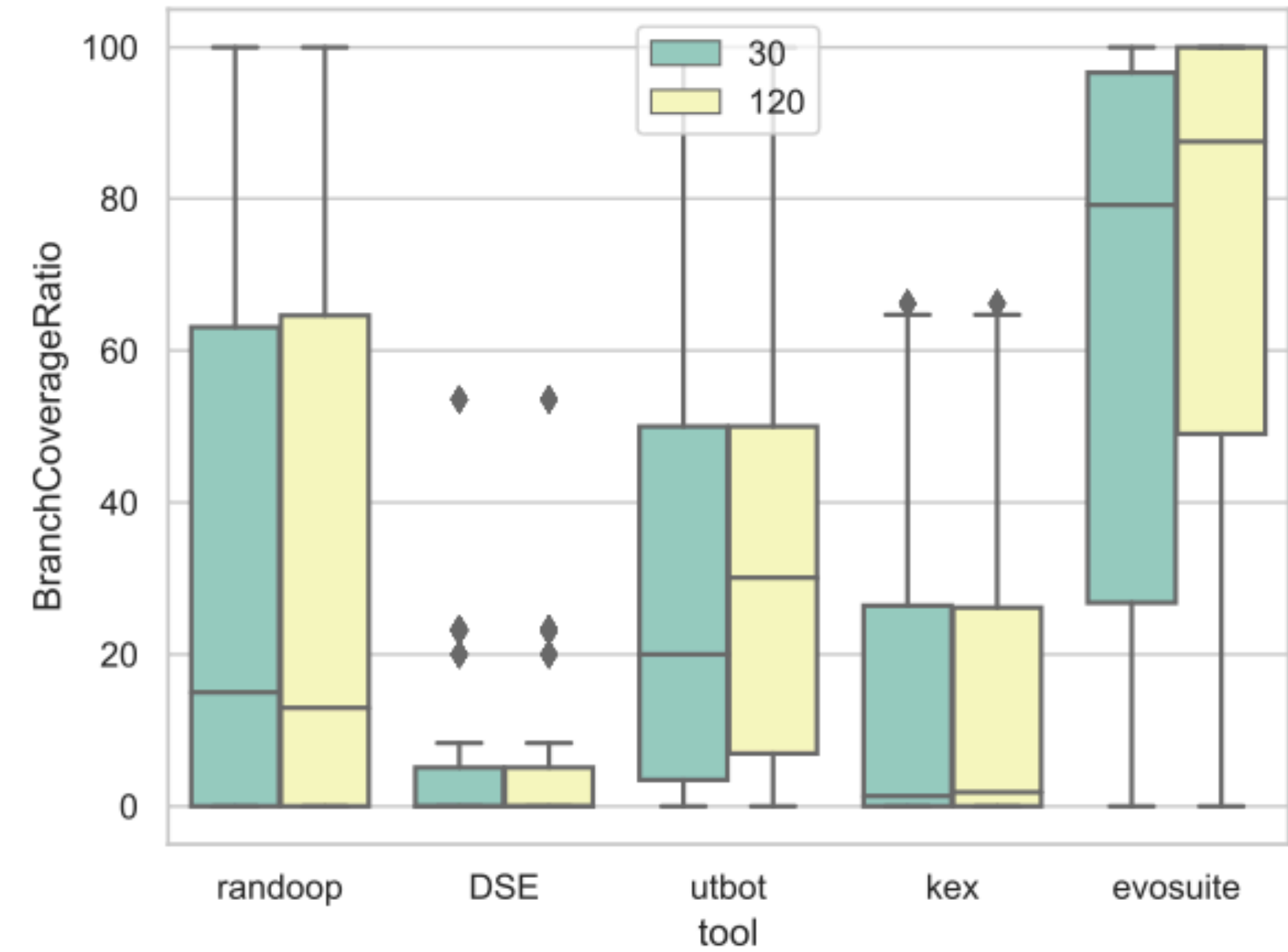


Fig. 2: Branch Coverage for Randoop, Evosuite(DSE), Utbot, Kex and Evosuite for 30 and 120 seconds.

Results (2)

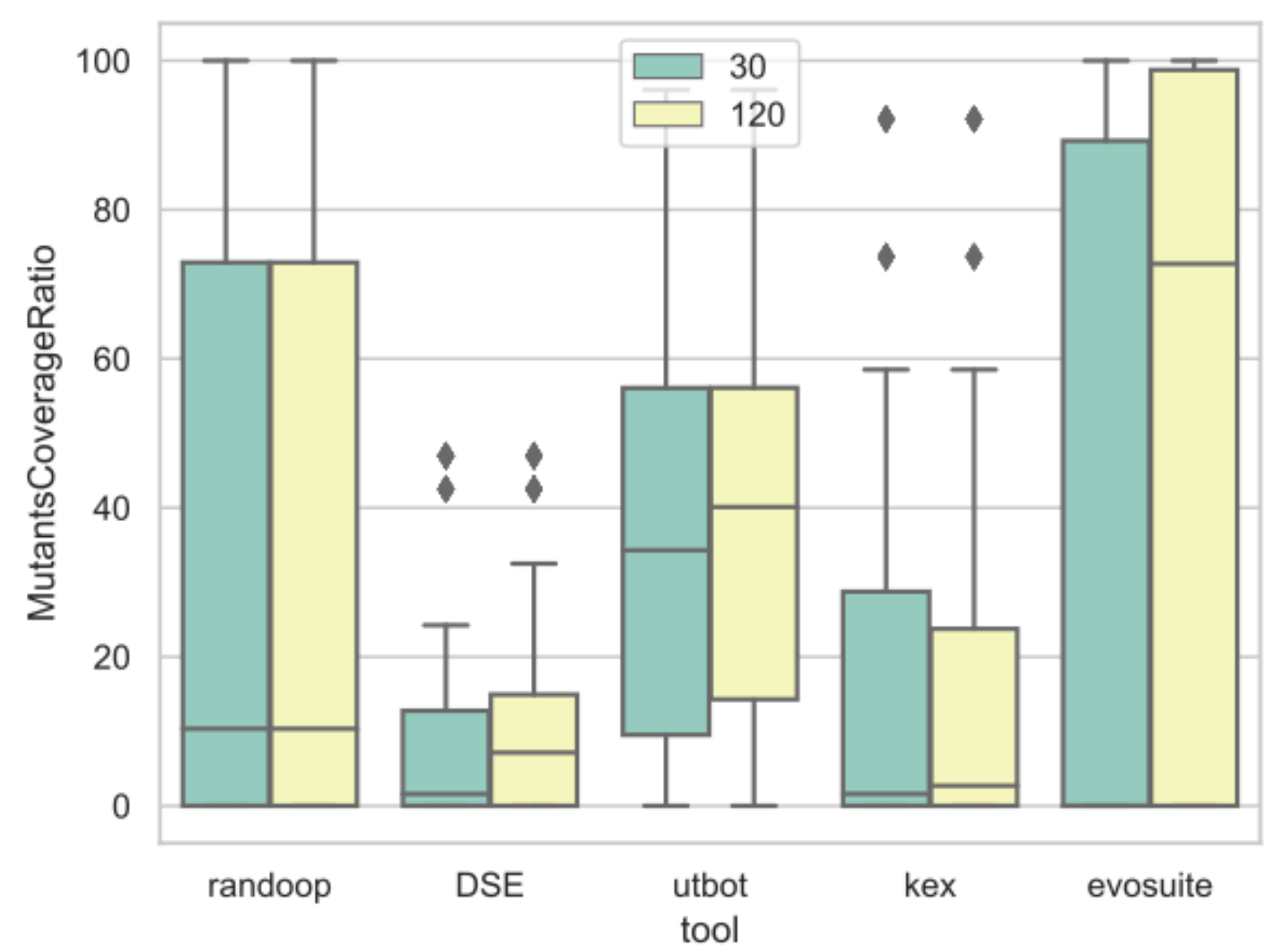


Fig. 3: Mutant Coverage for Randoop, Evosuite(DSE), Utbot, Kex and Evosuite for 30 and 120 seconds.

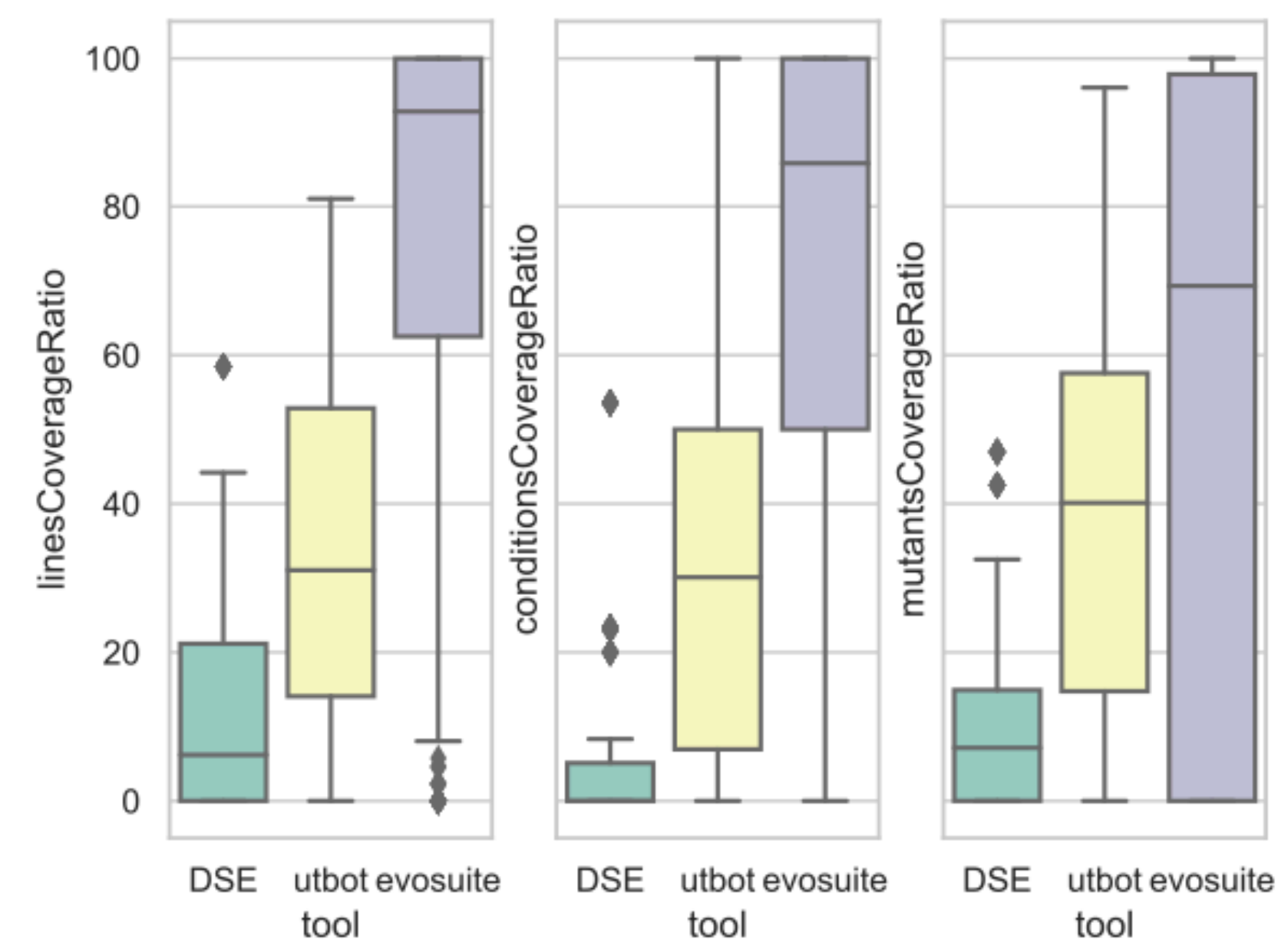
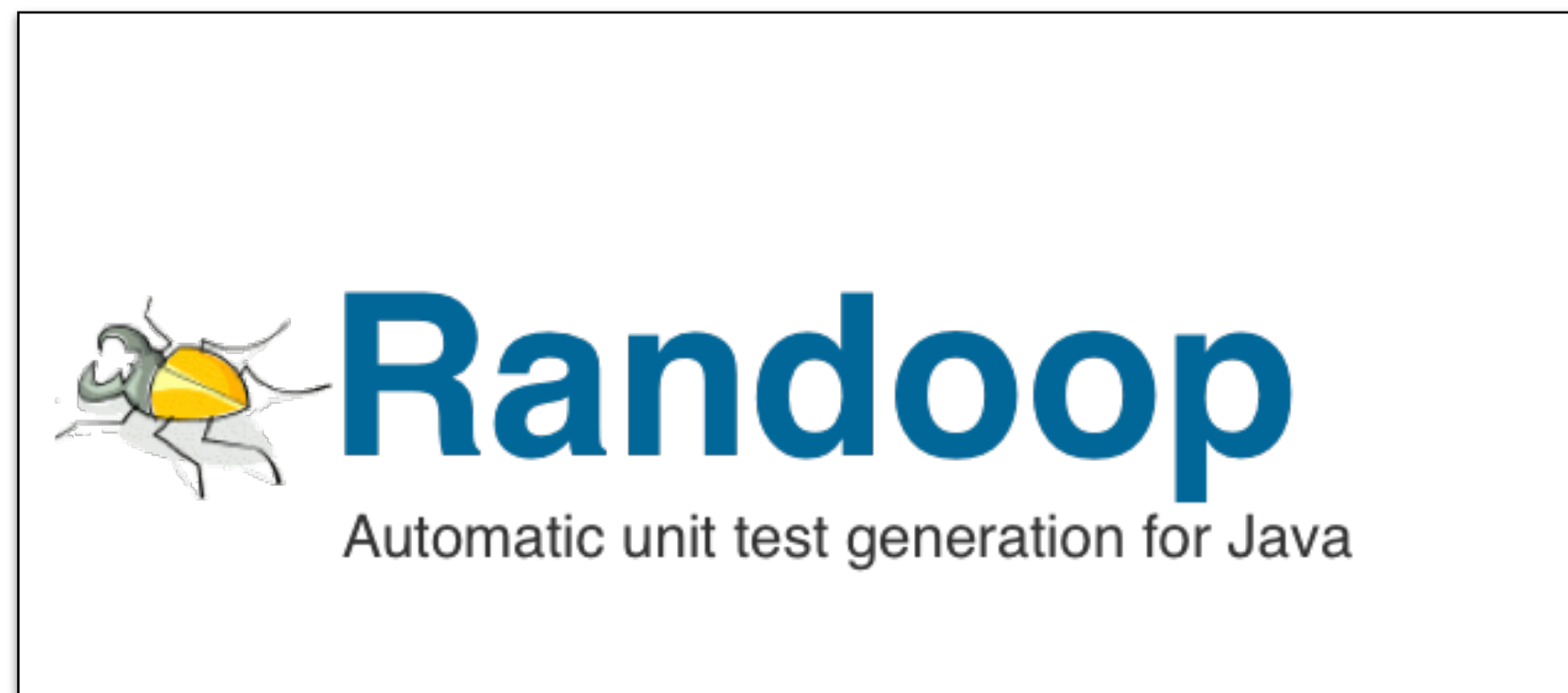


Fig. 4: Coverage for Evosuite(DSE), Utbot and Evosuite on a time budget of 5 minutes.

Final Ranking

Baseline



V.S.

Competitors

EVASUITE



UtBot

Kex

EVASUITE - DSE

Lessons Learnt

- Identified **aspects** to **improve** and **bugs** that could be **fixed** in the infrastructure
- Docker **simplifies** the evaluation procedure
 - **More participants to the competition!**
 - From Academia & Industry

What's Next?

- **Contest Infrastructure**

- <https://github.com/JUnitContest/junitcontest>
- Improve usability
 - Facilitate setup of an evaluation
 - Facilitate evaluation in other contexts
 - Update the user documentation
- Storage and versioning of the results (and participating tools?)

- **For the next edition**

- More tools
- More CUTs
- [Python as new language to experiment!](#)

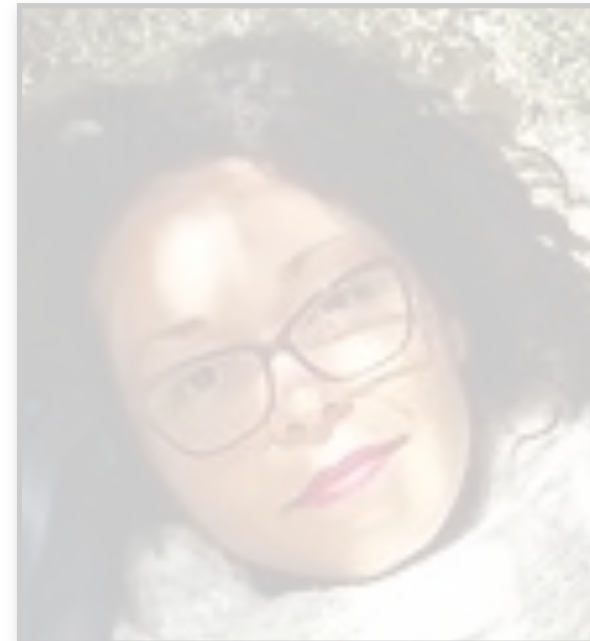
SBST Tool Competition - 2021

Co-chairs 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Fiorella Zampetti

University of Sannio



Alessio Gambi

Passau University



Vincenzo Riccio

University of Lugano

Class Under Test (CUT)

```
class Triangle {  
  int a, b, c; //sides  
  String type = "NOT_TRIANGLE";  
  
  Triangle (int a, int b, int c){...}  
  
  void computeTriangleType() {  
    1. if (a == b) {  
    2.   if (b == c)  
    3.     type = "EQUILATERAL";  
    4.   else  
    5.     type = "ISOSCELES";  
    6.   } else {  
    7.     if (a == c) {  
    8.       type = "ISOSCELES";  
    9.     } else {  
    10.      if (b == c)  
    11.        type = "ISOSCELES";  
    12.      else  
    13.        type = "SCALENE";  
    14.      }  
    15.    }  
  }  
}
```

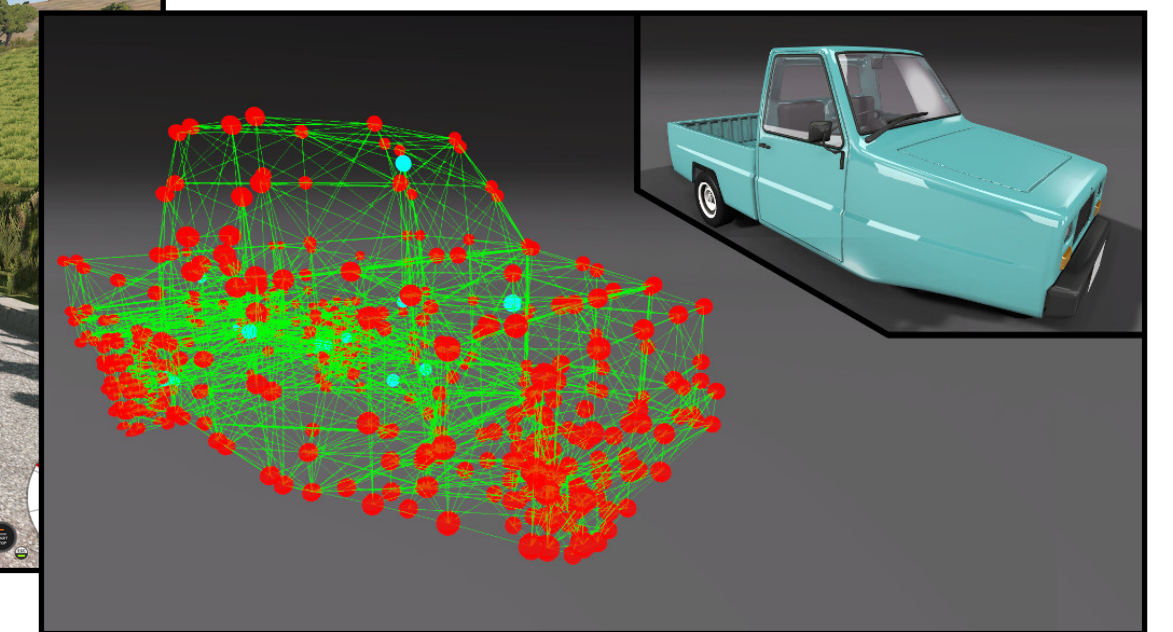
Test Case

```
@Test  
public void test(){  
  // Constructor (init)  
  // Method Calls  
  // Assertions (check)  
}
```



```
@Test  
public void test(){  
  Triangle t = new Triangle (1,2,3);  
  t.computeTriangleType();  
  String type = t.getType();  
  assertTrue(type.equals("SCALENE"));  
}
```

Java tool competition



Cyber-physical systems (CPS) testing competition

SBST Tool Competition - 2021



Alessio Gambi

Passau University



Vincenzo Riccio

University of Lugano

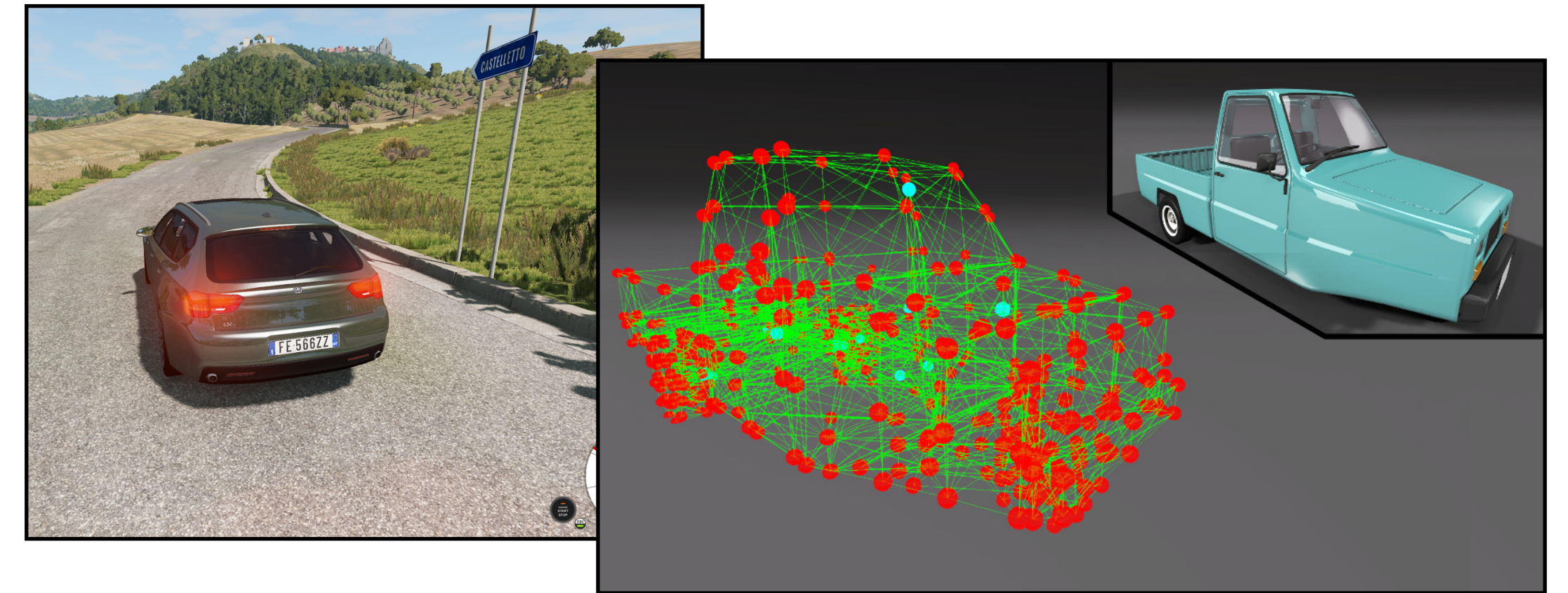


Figure 2: Example of CPS testing tool simulation environment.

Cyber-physical systems (CPS) testing competition: In addition to the traditional Java tool competition, we also organize a CPS testing competition on self-driving cars simulation environments. Specifically, in collaboration with the BeamNG research team (<https://beamng.gmbh/research/>), this competition focuses on the

- Generation of scenarios using BeamNG self-driving cars simulator

A Dream Come True



That Can Also Be a Nightmare

WIRED



Friday briefing: Uber's self-driving software was responsible for pedestrian fatality

Crash investigators have found that disabled features and poor object identification led to the killing of a pedestrian by one of Uber's autonomous vehicles, vast canyons are buried in the ice between Antarctica's mountains

—
By **WIRED**

25 May 2018



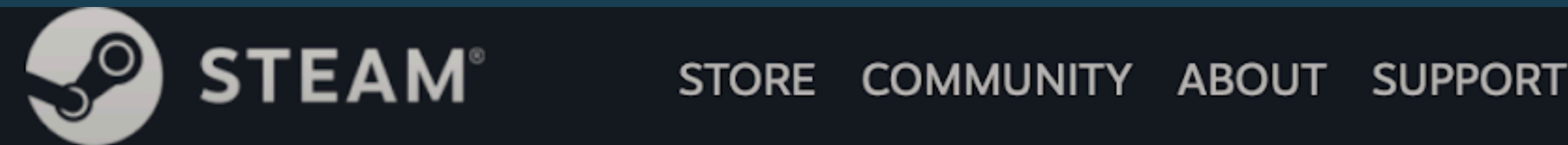
Testing Self-Driving Cars



Testing Self-Driving Cars

**Time-consuming
Limited realism
Impractical**

Simulation-based Testing



Your Store ▾

Browse ▾

Points Shop

News

Steam Labs

All Games > Simulation Games > BeamNG.drive

BeamNG.drive

CUSTOMER REVIEWS

Overall Reviews:

Overwhelmingly Positive (40,939 reviews) ?

REVIEWS

"The Most Impressive Physics Engine You've Never Seen"
IGN

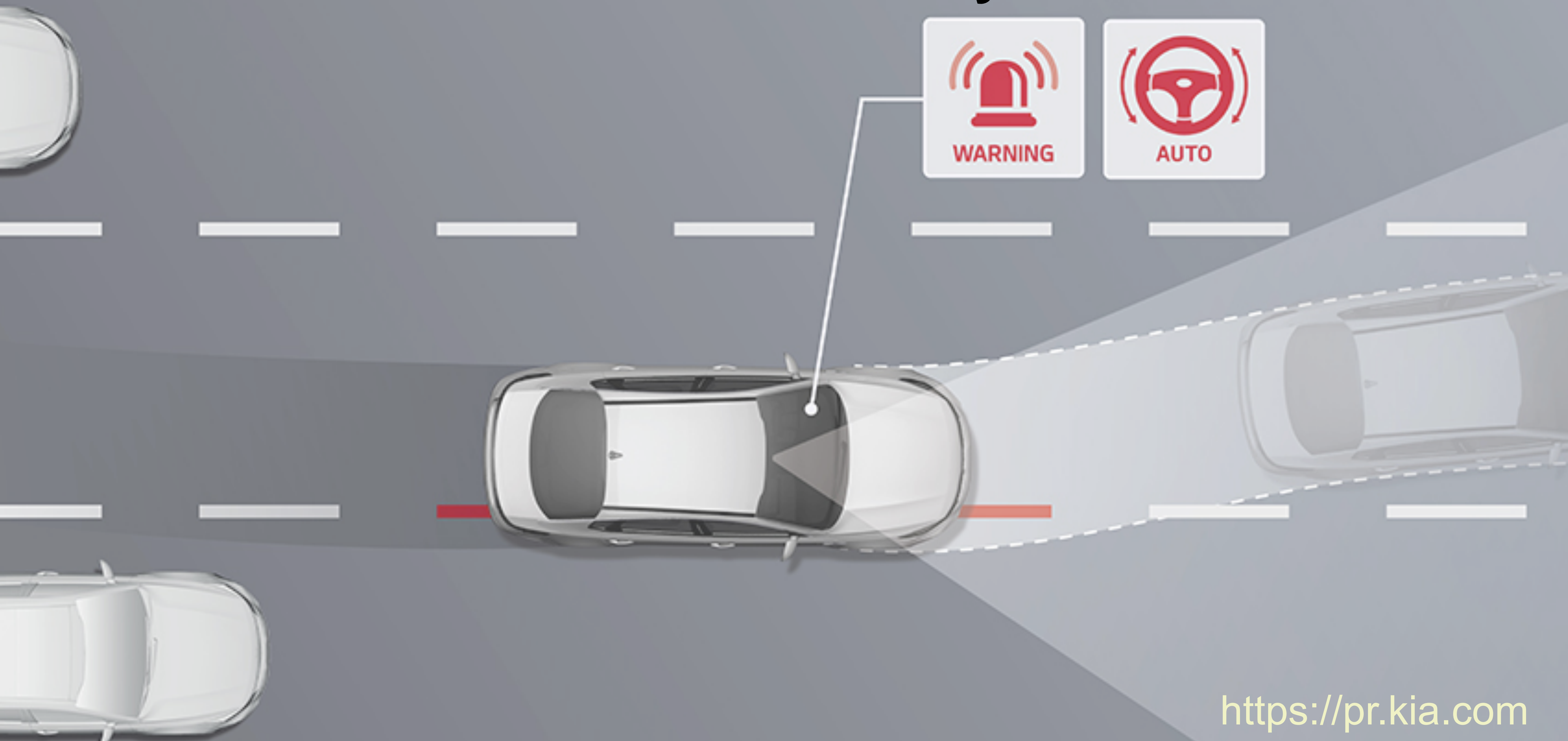
"BeamNG's Amazingly Realistic Car Crashes"
Gameinformer

"Amazing Car Crashes + Hilarious Greenlight Trailer = Magic"
Kotaku



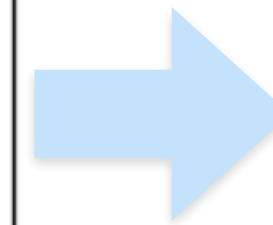
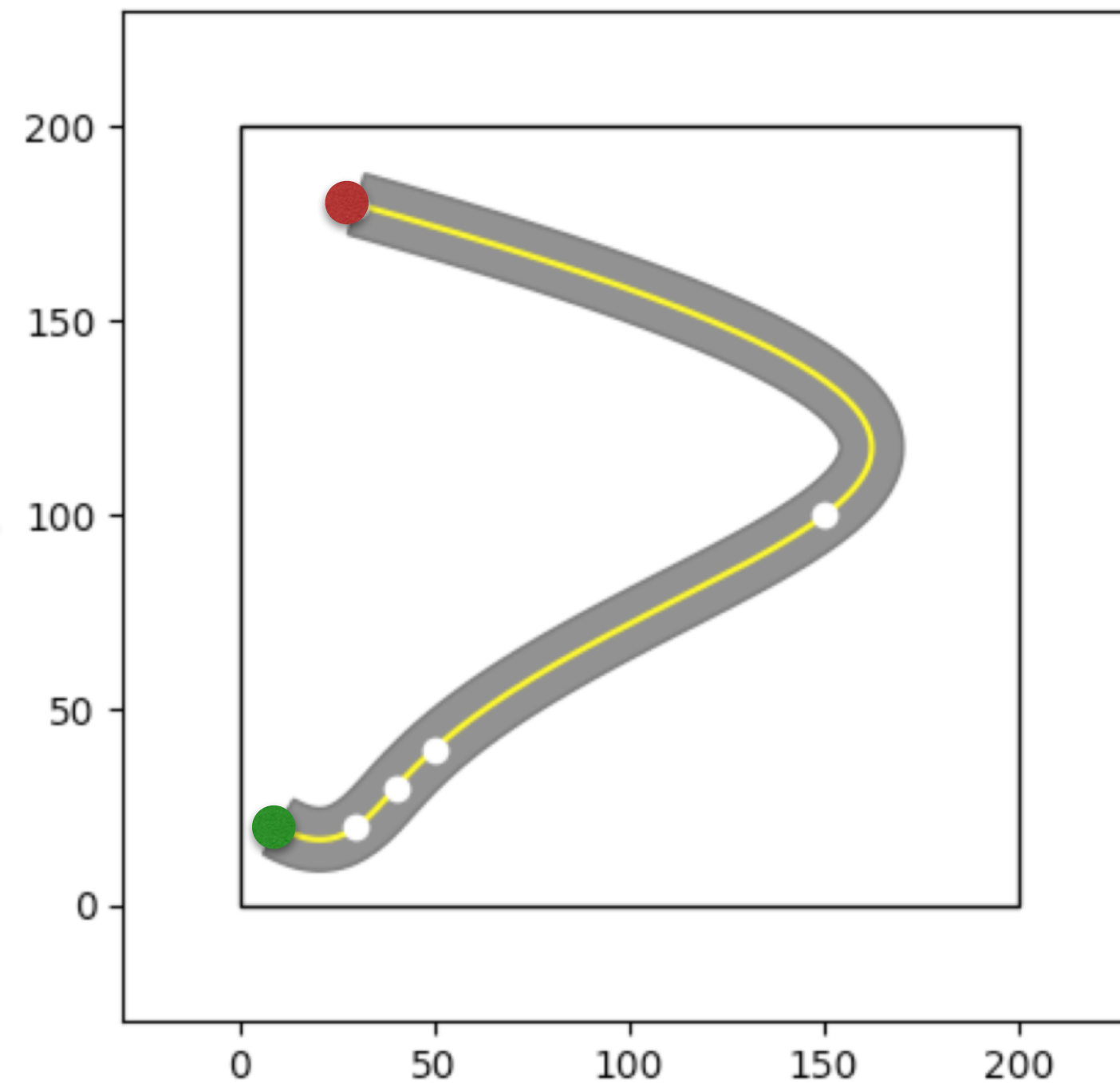
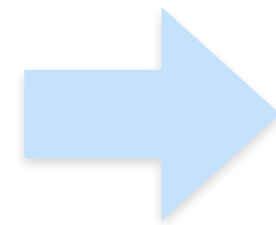
A dynamic soft-body physics vehicle simulator capable of doing just about anything.

Lane Keeping Assist System



What is a Test Case?

- Start = (10.0, 20.0)
- A = (30.0, 20.0)
- B = (40.0, 30.0)
- C = (50.0, 40.0)
- D = (150.0, 100.0)
- End = (30.0, 180.0)





What is a Failure?

Out of

Bound

Episode

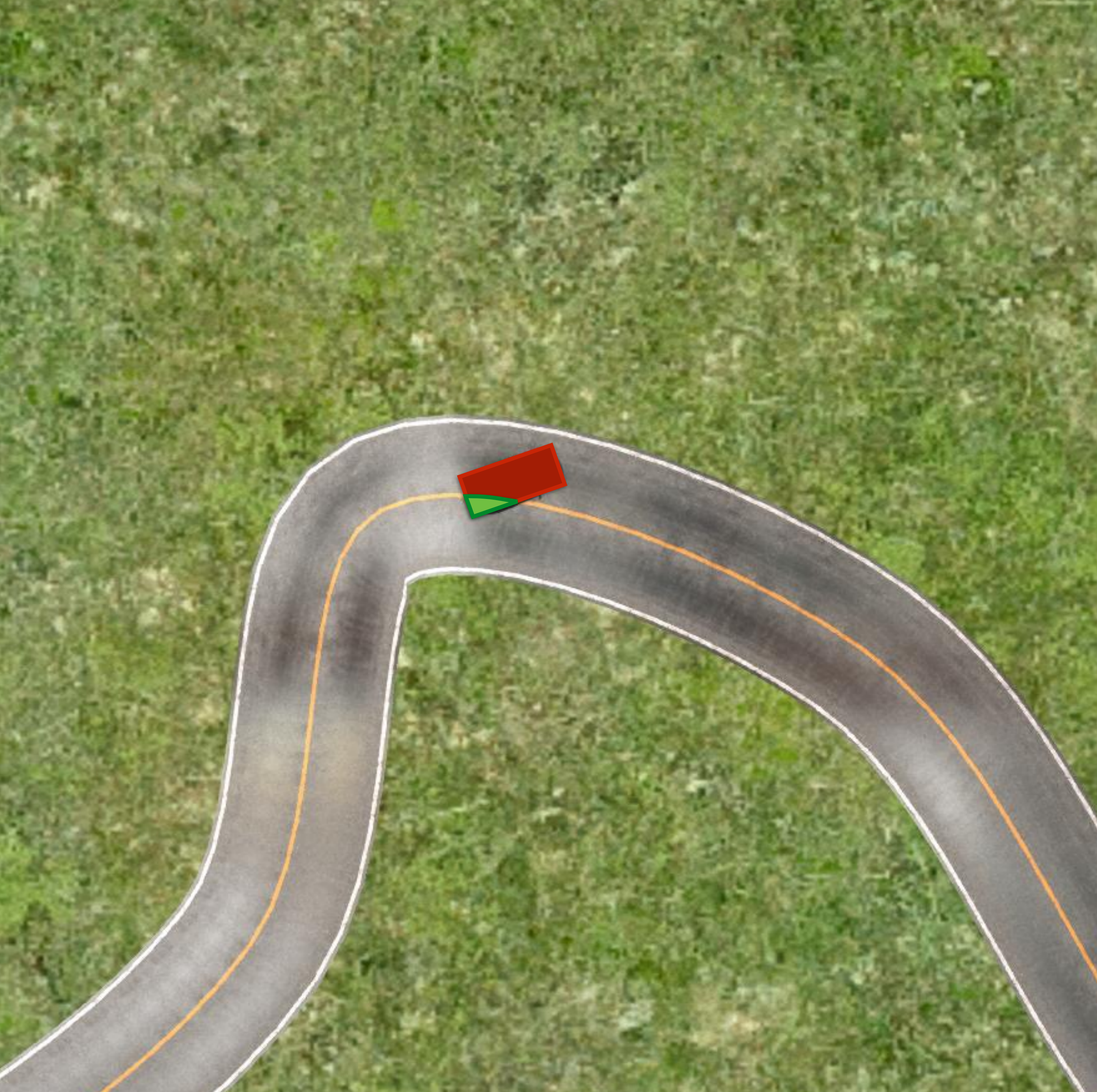


What is a Failure?

Out of

Bound

Episode

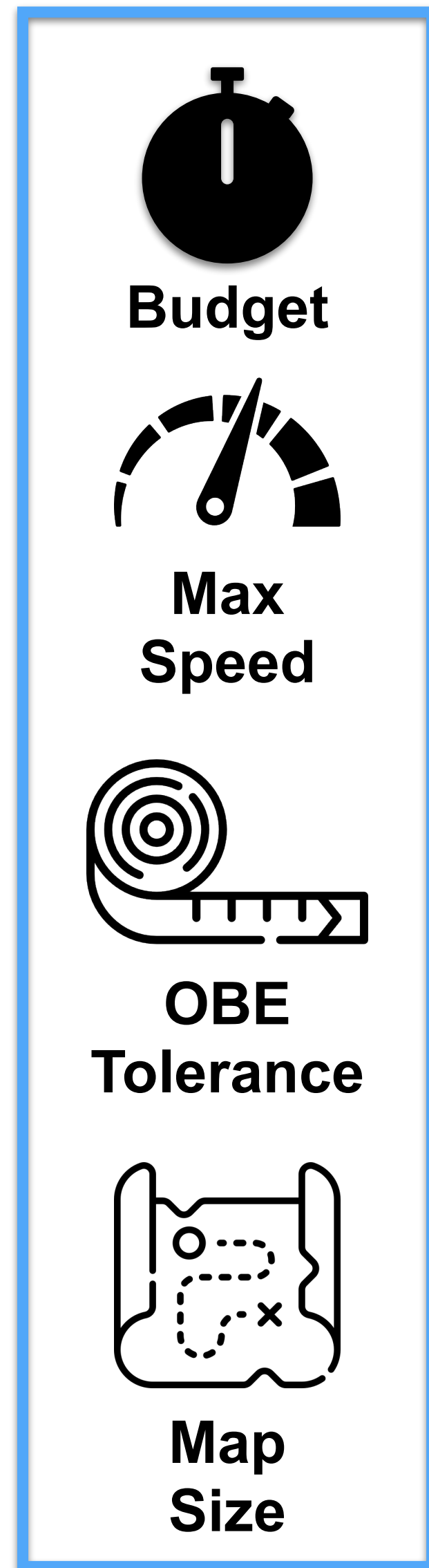


What is a Failure?

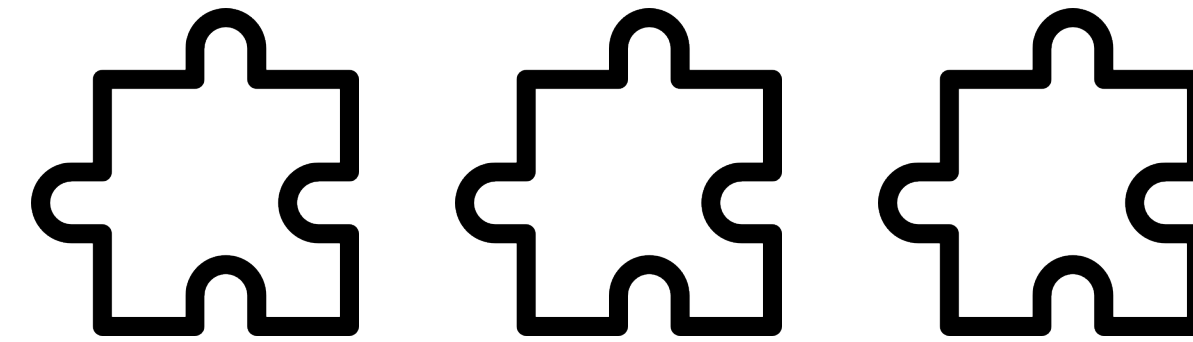
Out of
Bound
Episode

% car outside the lane
>
threshold

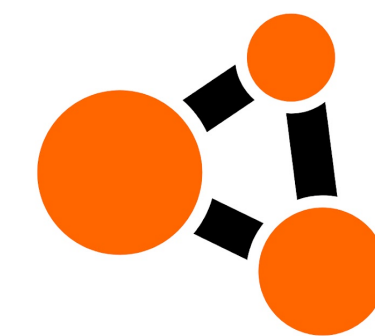
Infrastructure



CPS Tool Contest Code Pipeline

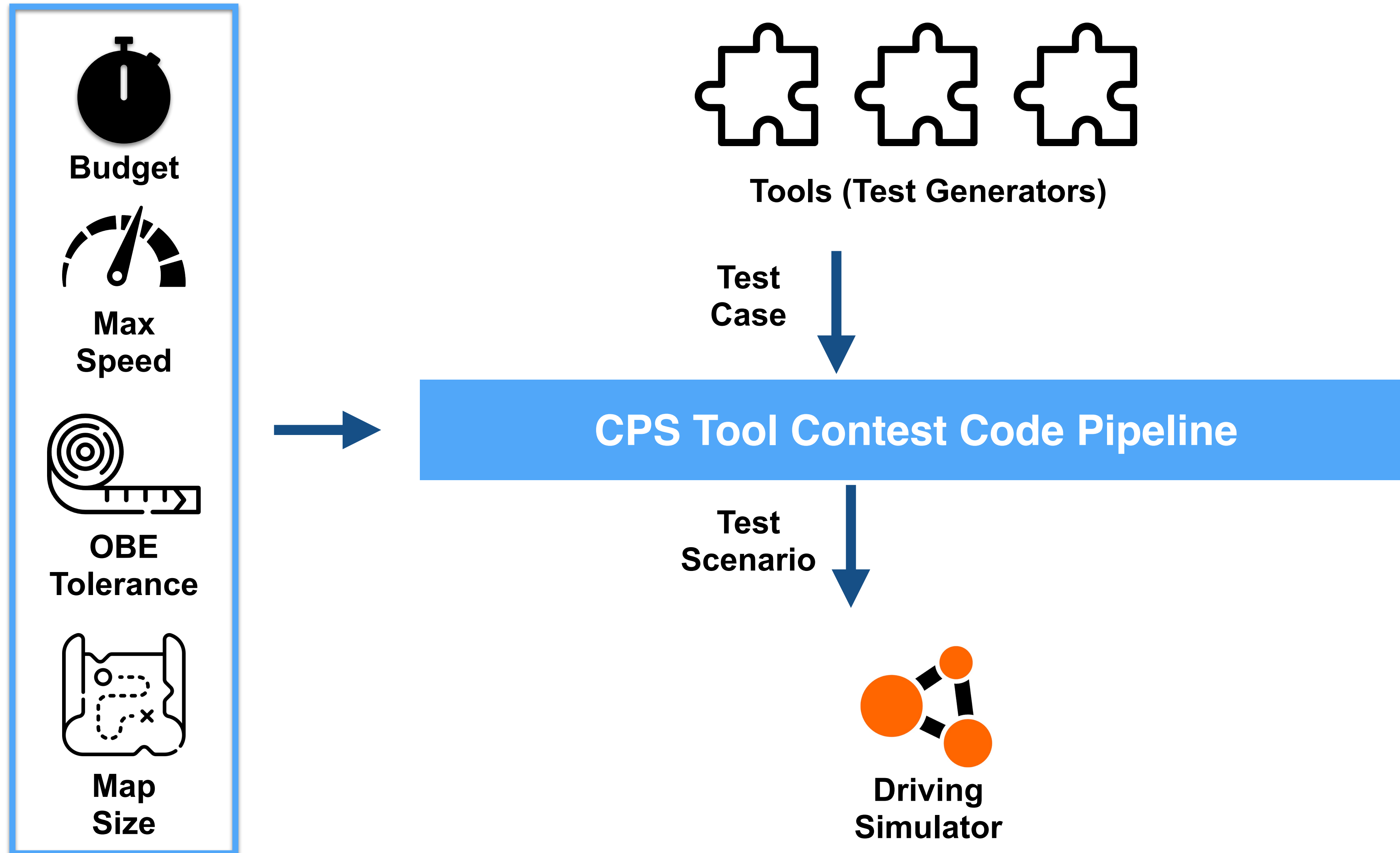


Tools (Test Generators)

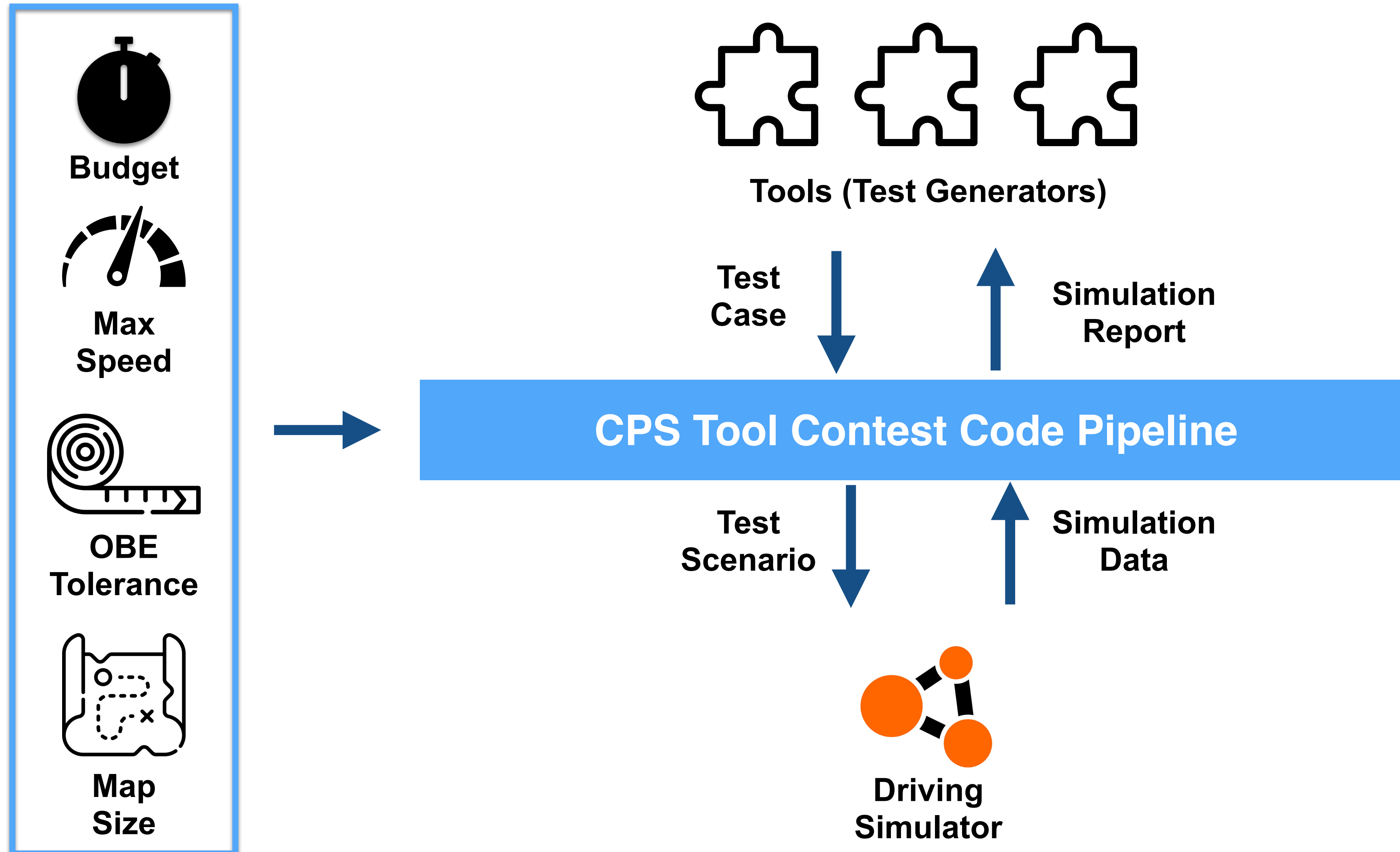


Driving Simulator

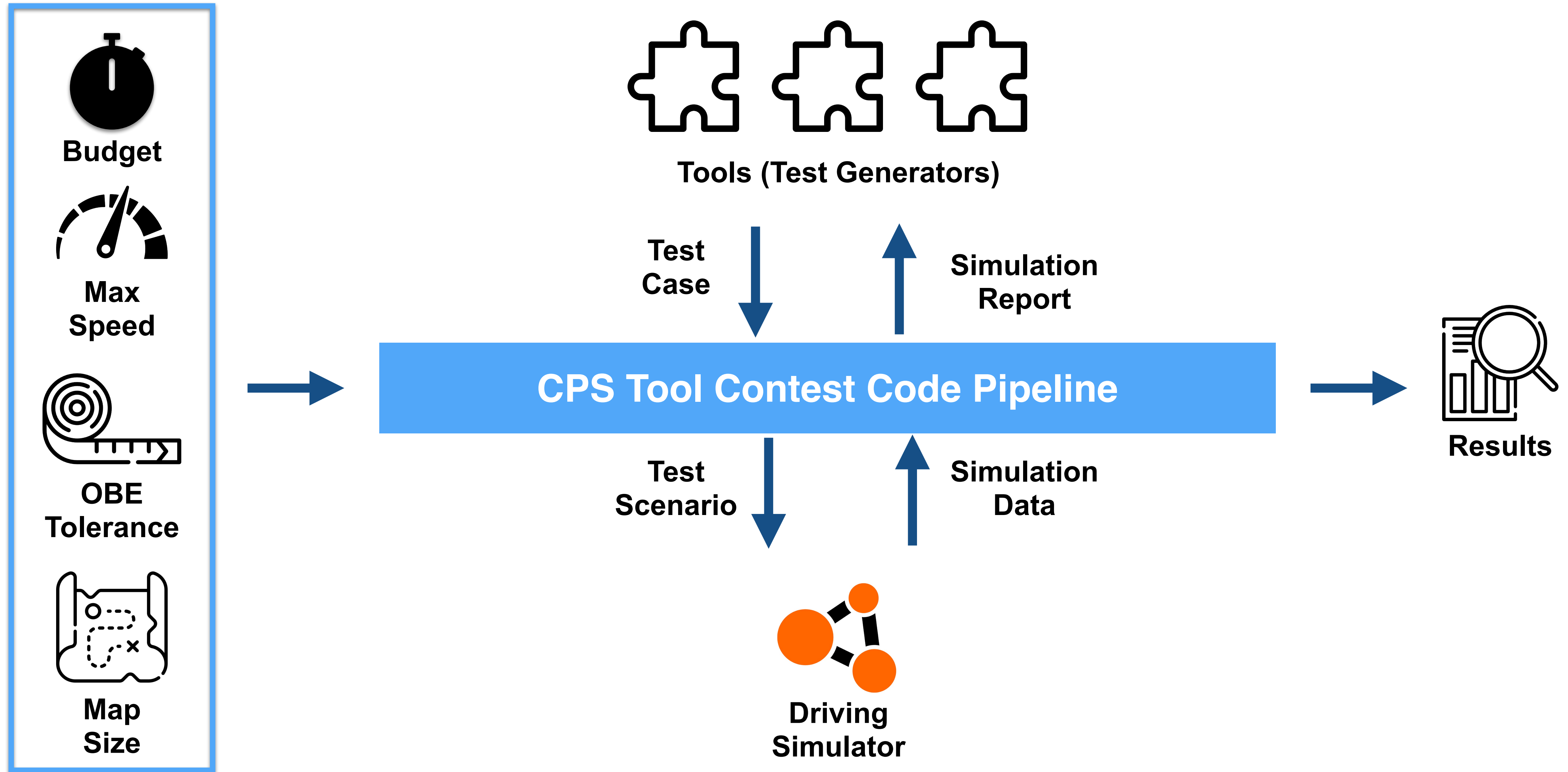
Infrastructure



Infrastructure

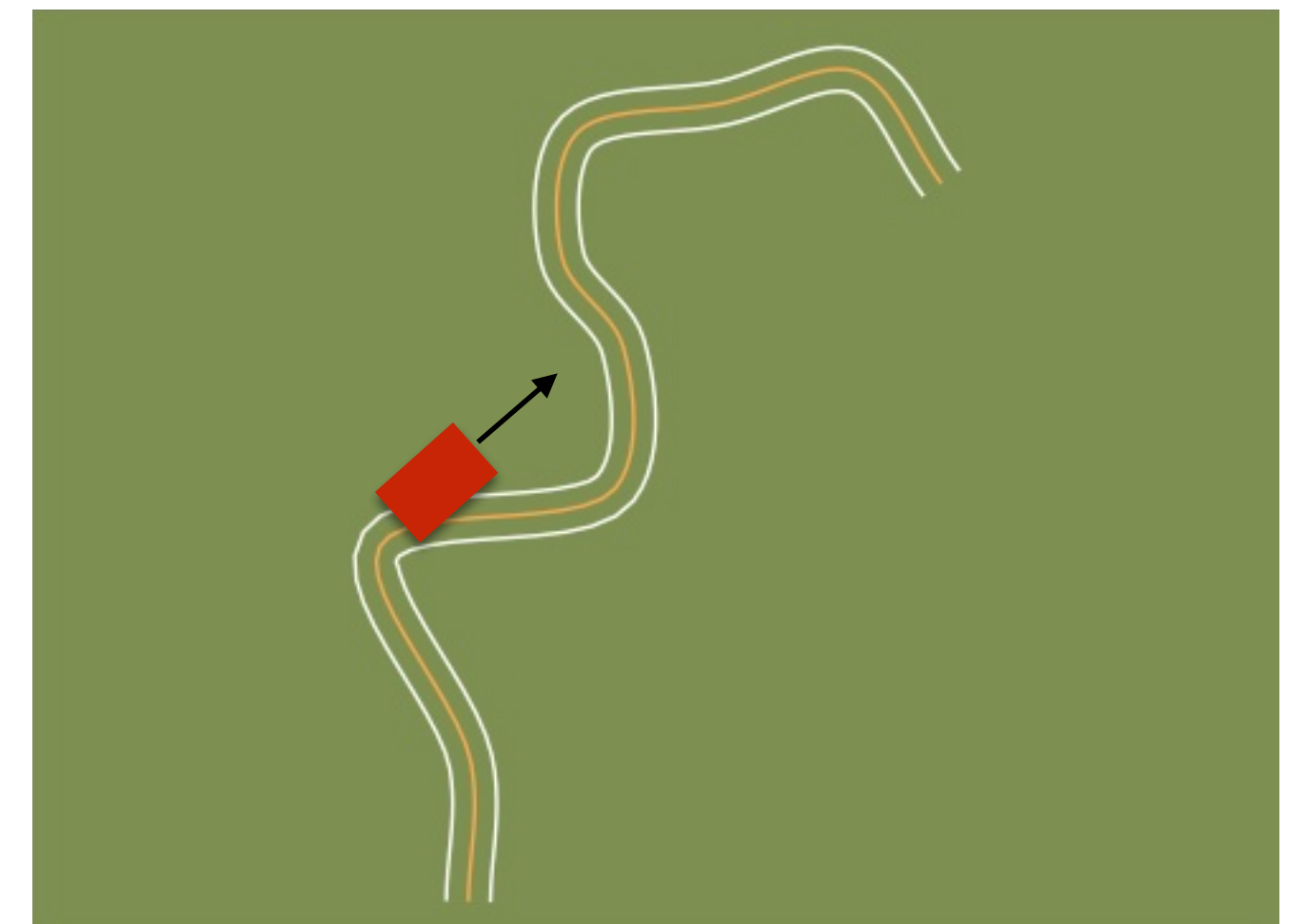
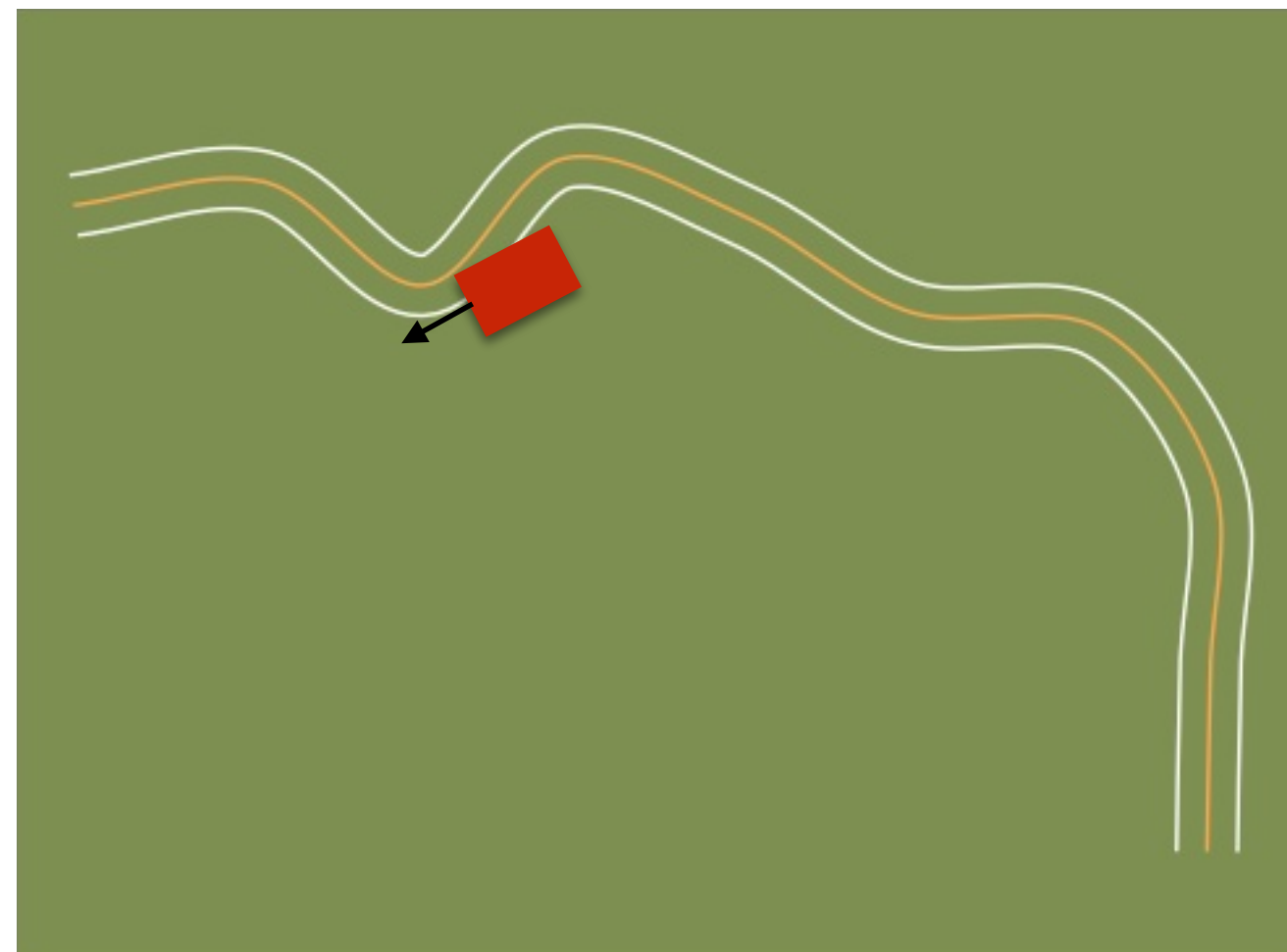
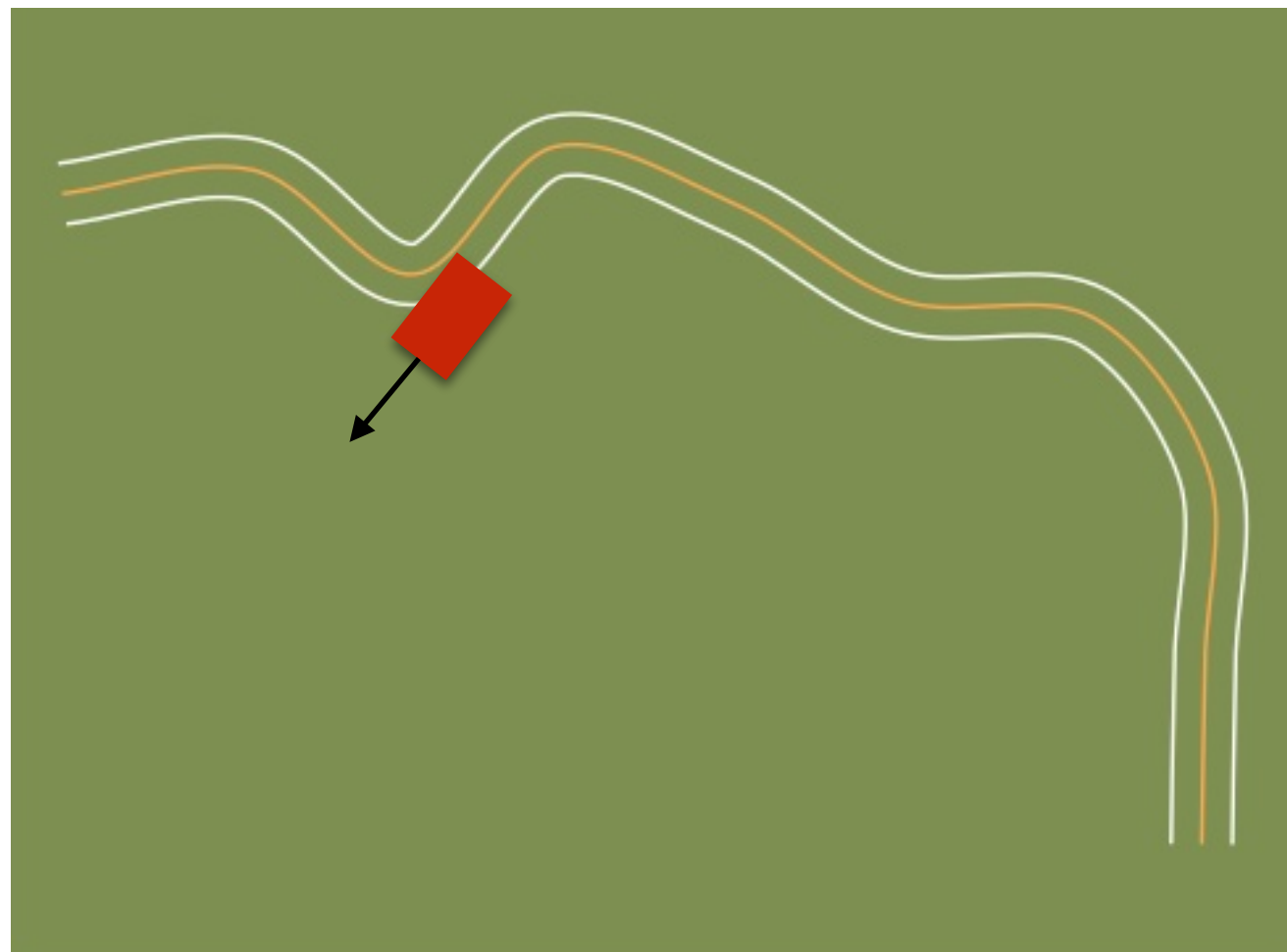


Infrastructure



Metrics: # OBEs

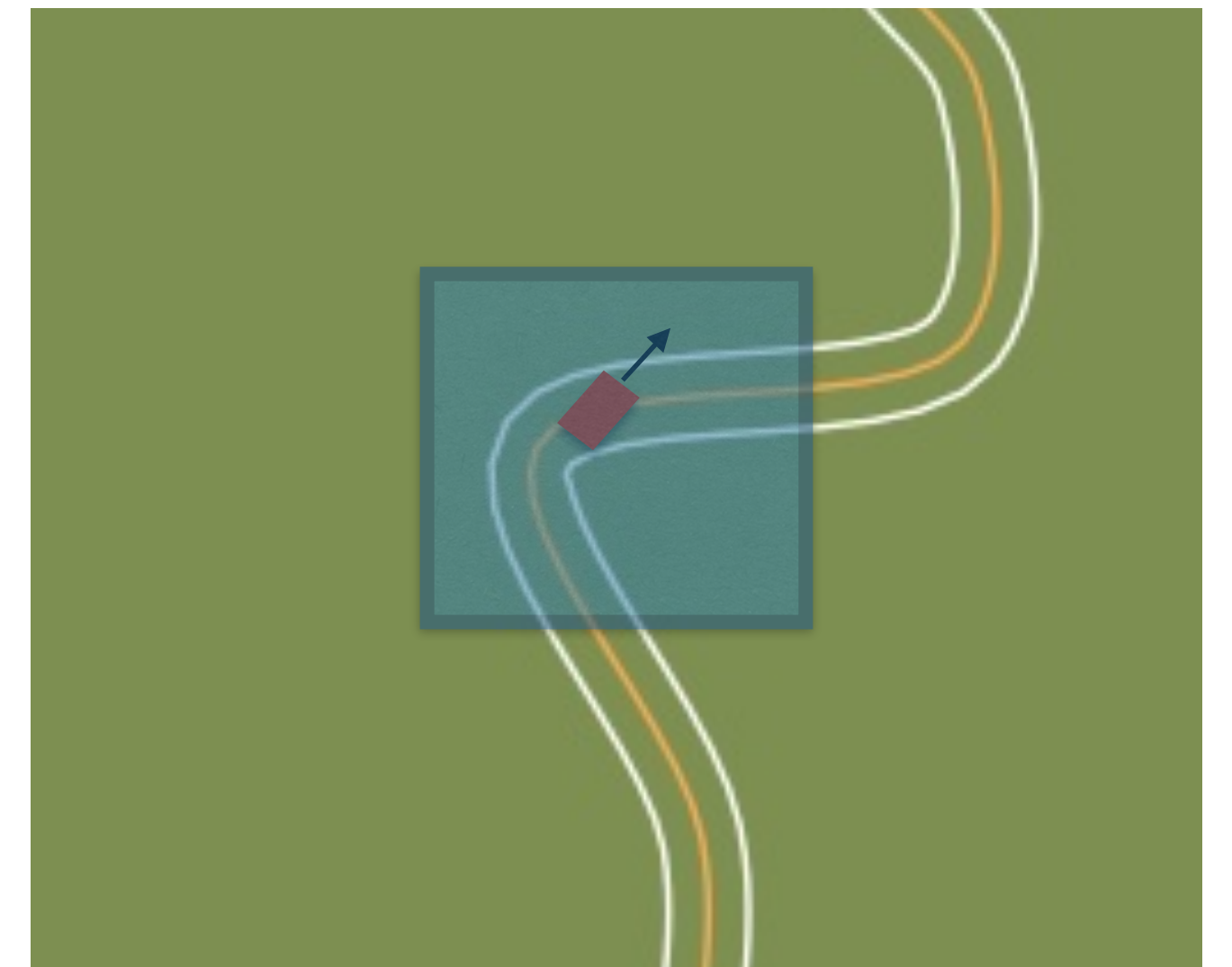
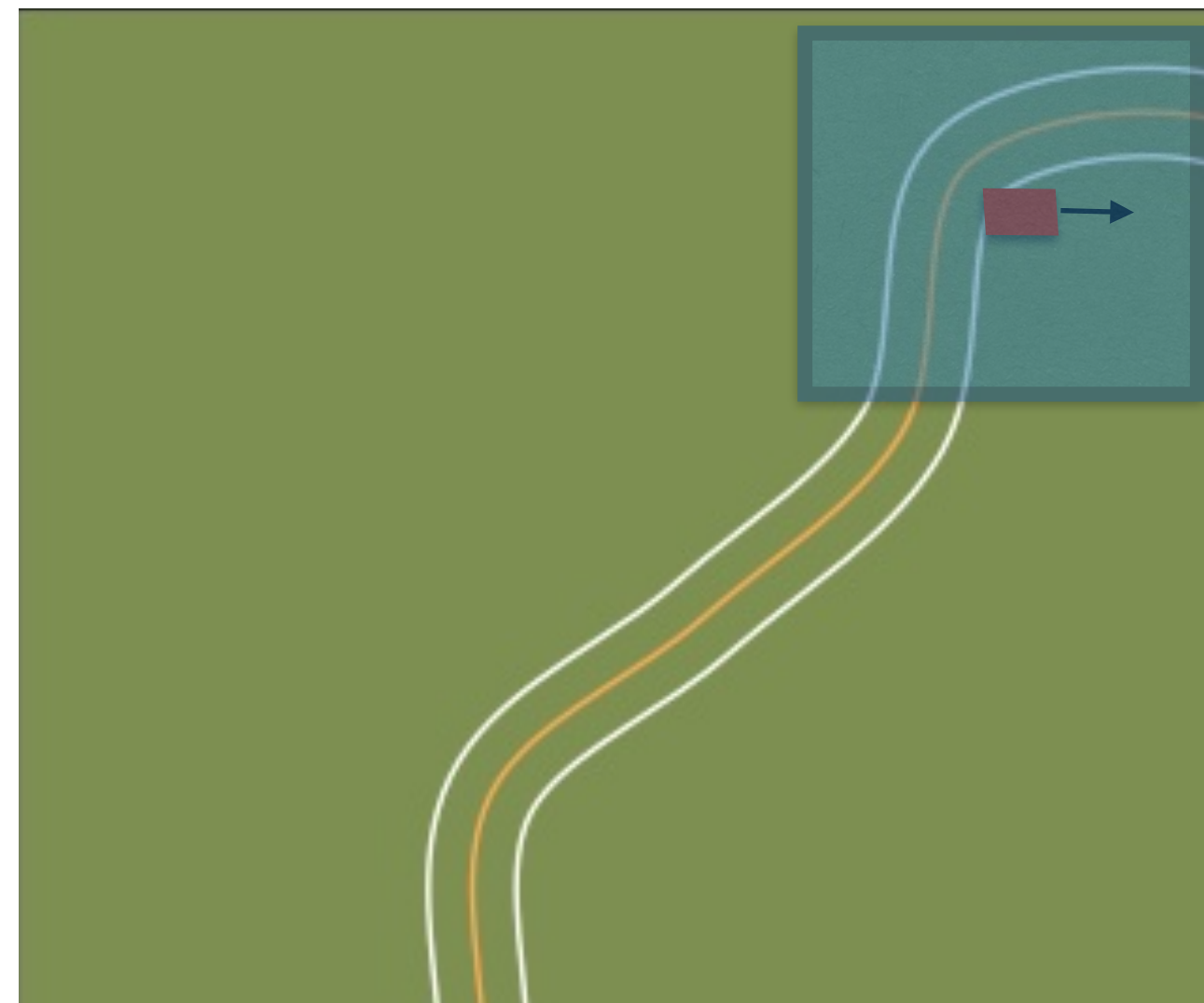
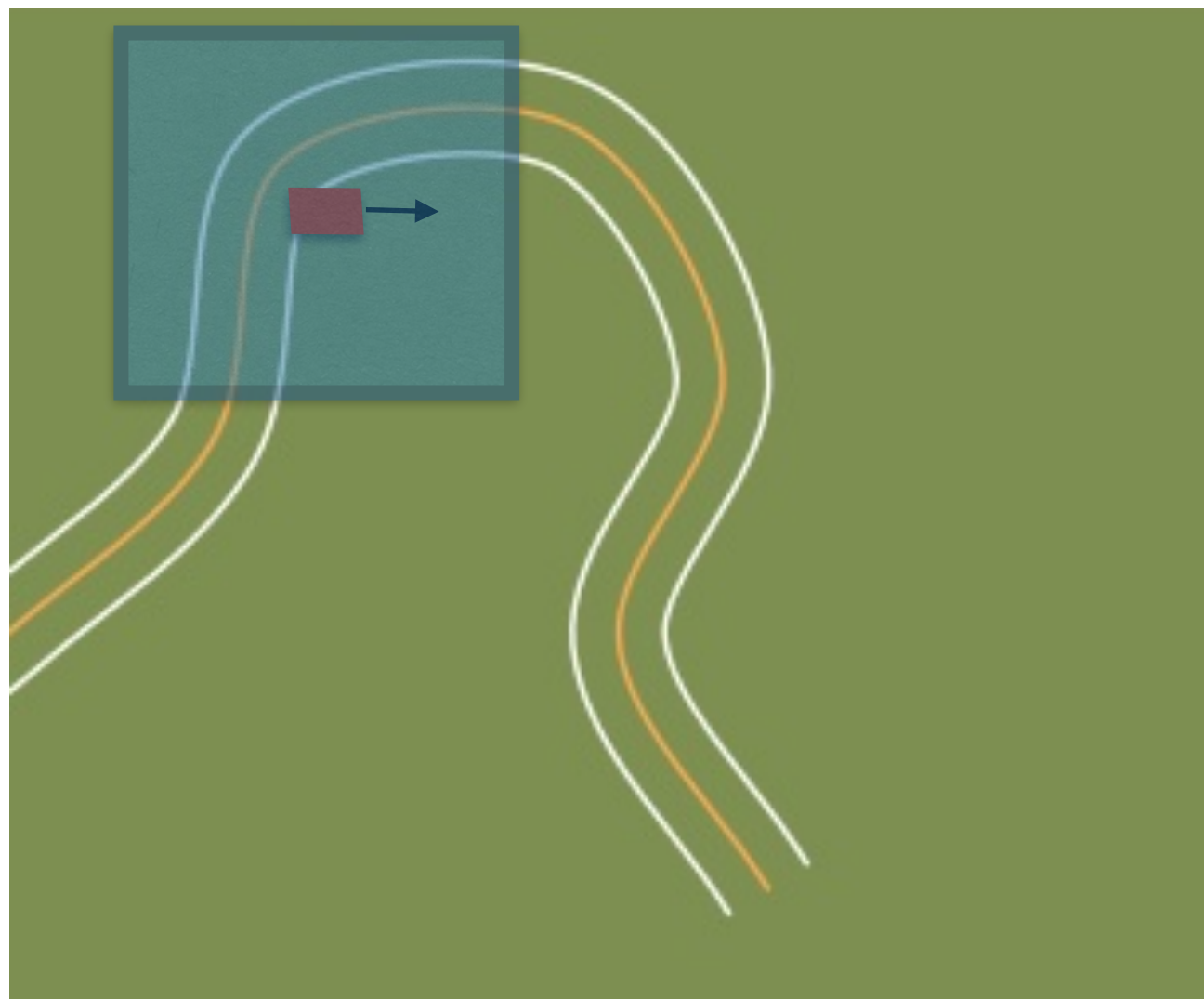
- # Failure-inducing generated test cases



OBEs = 3

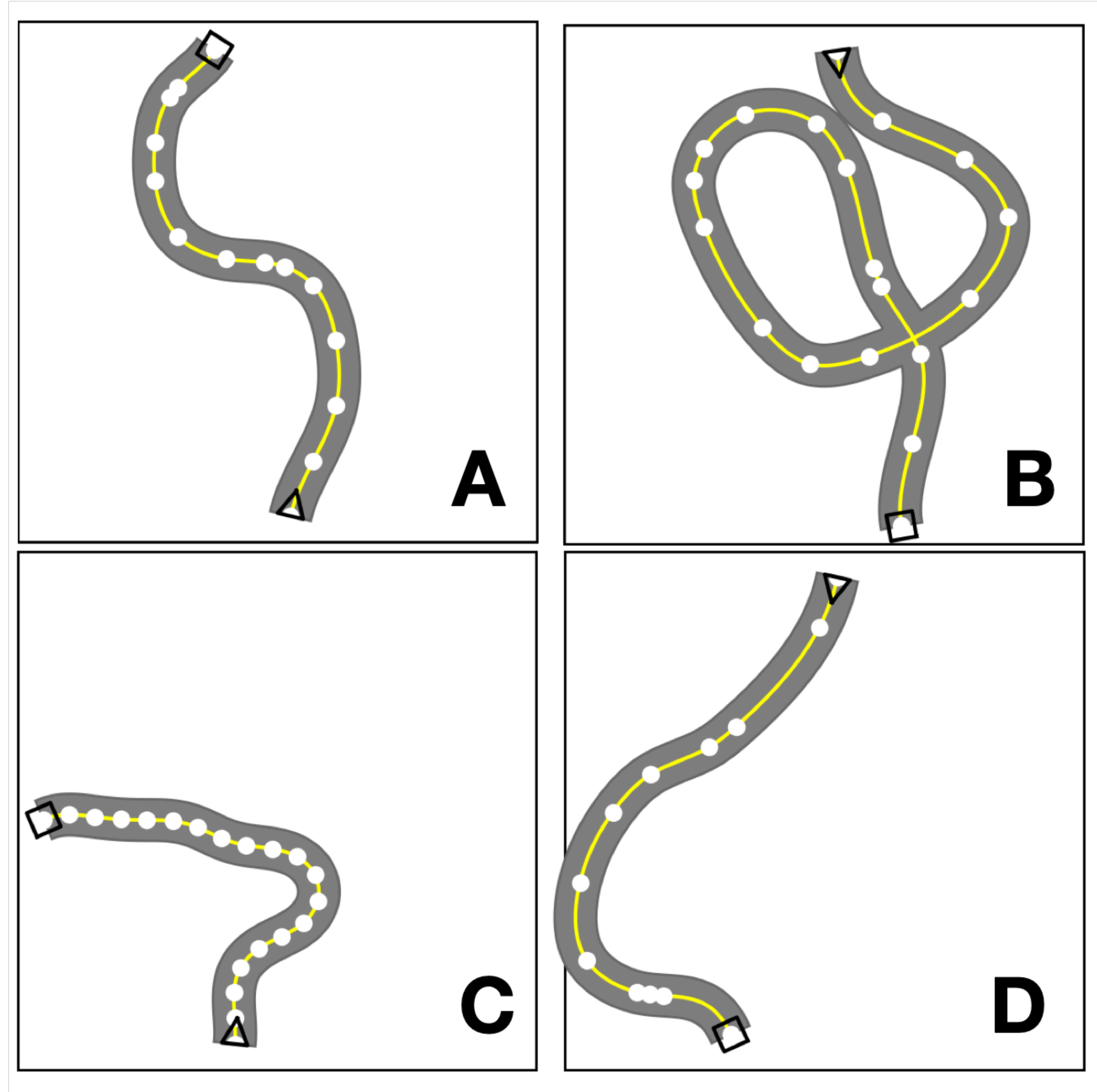
Metrics: Failure Sparseness

- Average maximum distance of the relevant road sectors



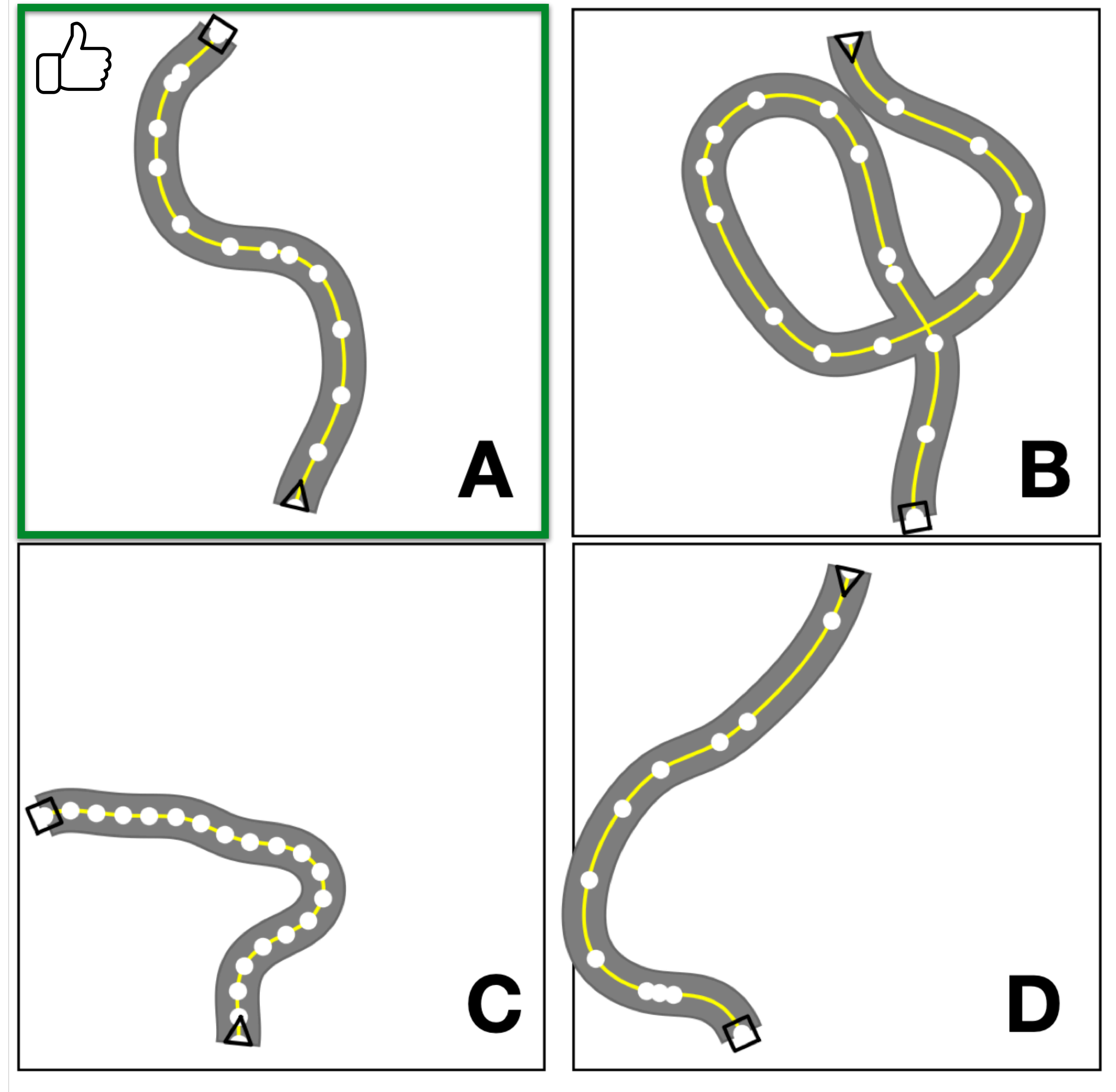
Metrics: Efficiency and Effectiveness

- Number of generated valid test cases within the time budget



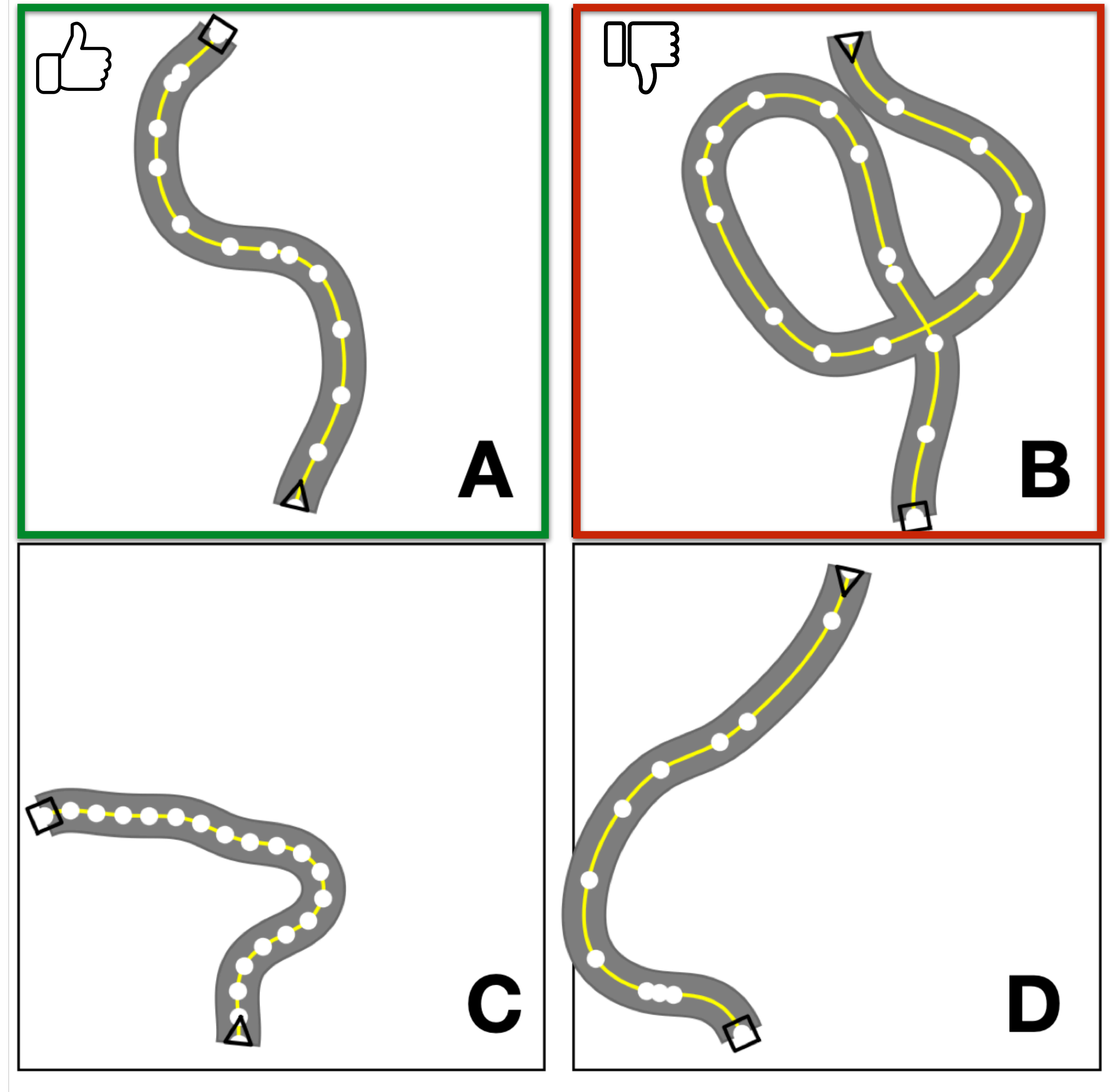
Metrics: Efficiency and Effectiveness

- Number of generated **valid** test cases within the time budget



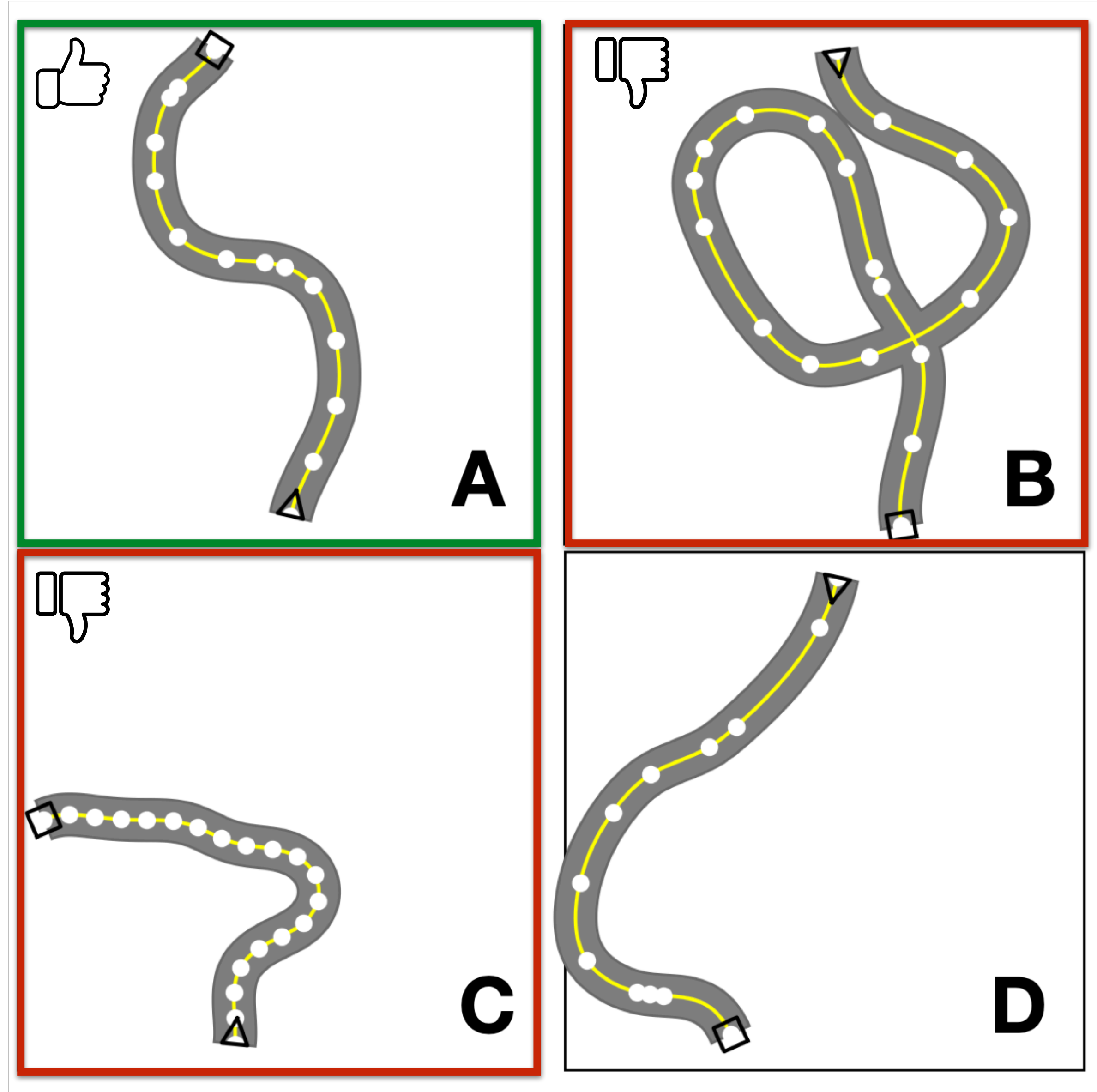
Metrics: Efficiency and Effectiveness

- Number of generated **valid** test cases within the time budget
- A valid road should:
 - not self-intersect



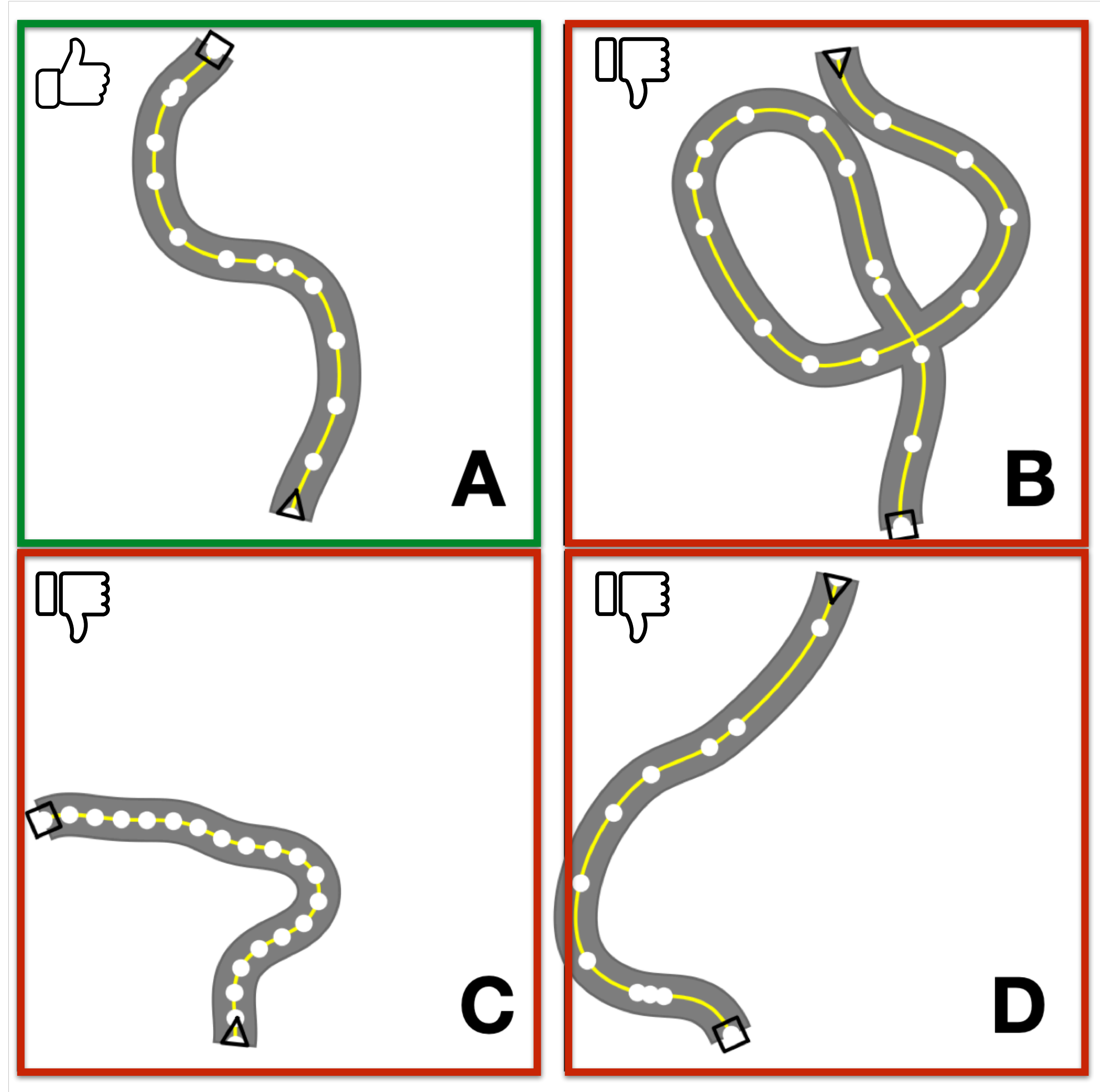
Metrics: Efficiency and Effectiveness

- Number of generated **valid** test cases within the time budget
- A valid road should:
 - not self-intersect
 - not contain overly sharp turns



Metrics: Efficiency and Effectiveness

- Number of generated **valid** test cases within the time budget
- A valid road should:
 - not self-intersect
 - not contain overly sharp turns
 - be fully contained in the specified map



Contest Methodology

Default

SBST21

**Test
Subject**

BeamNG
AI

**Driving
Simulator**

BeamNG.
tech

**Search
budget**

5 hours

**Map
Size**

200 X 200

**Max
Speed**

None



**OBE
Tolerance**

0.95

**Search
budget**

2 hours

**Map
Size**

200 X 200

**Max
Speed**

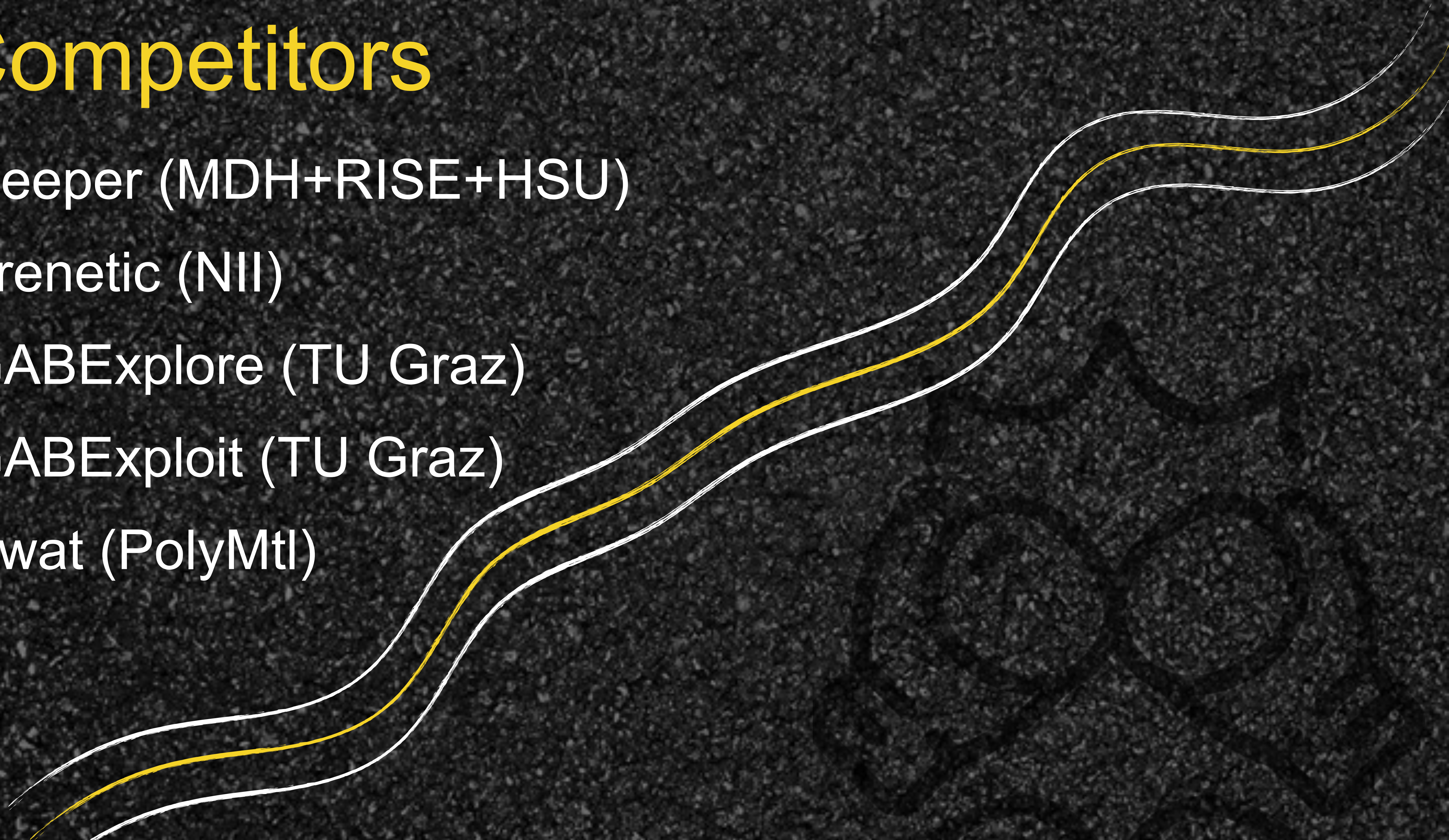
70 km/h

**OBE
Tolerance**

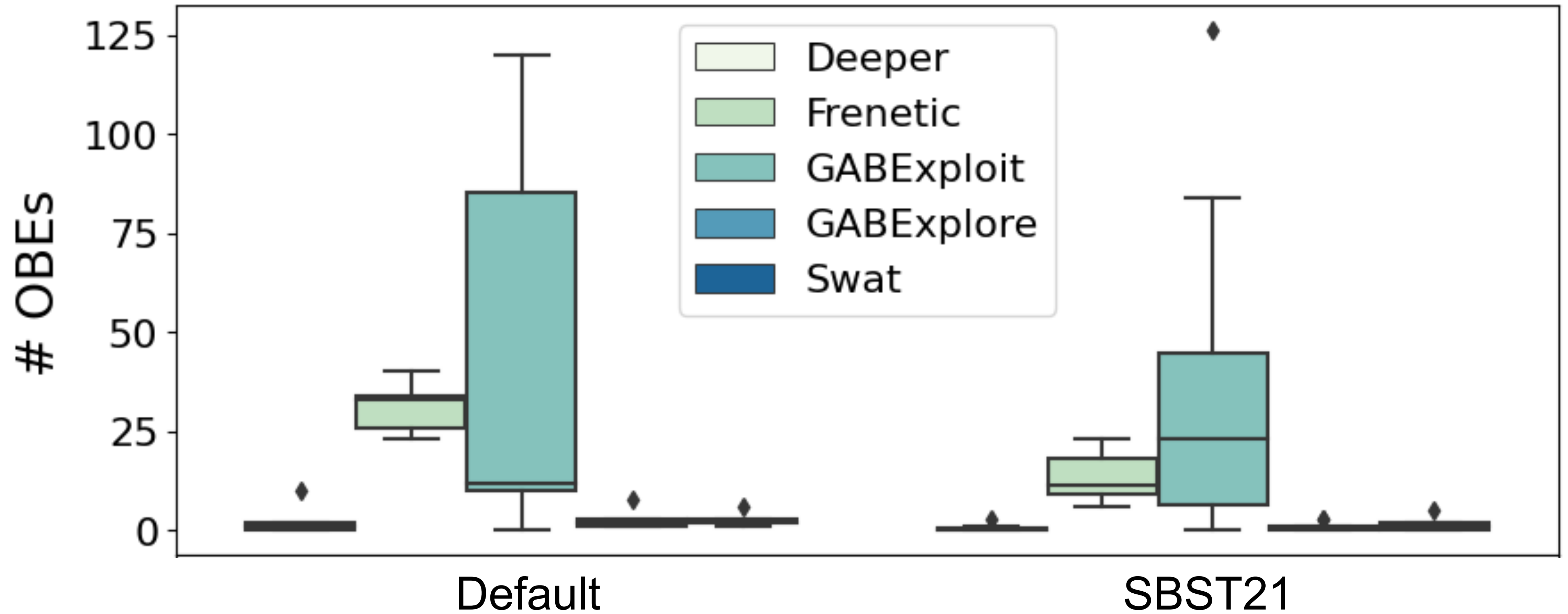
0.85

Competitors

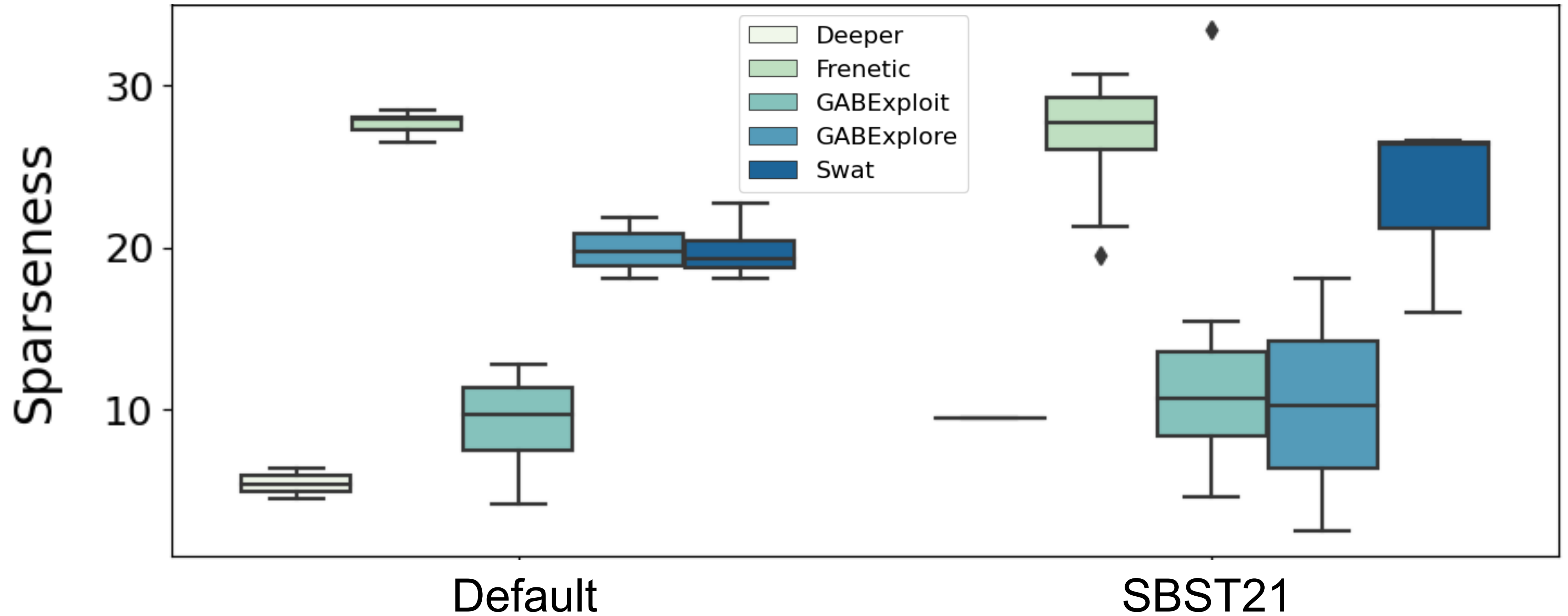
- Deeper (MDH+RISE+HSU)
- Frenetic (NII)
- GABExplore (TU Graz)
- GABExploit (TU Graz)
- Swat (PolyMtl)



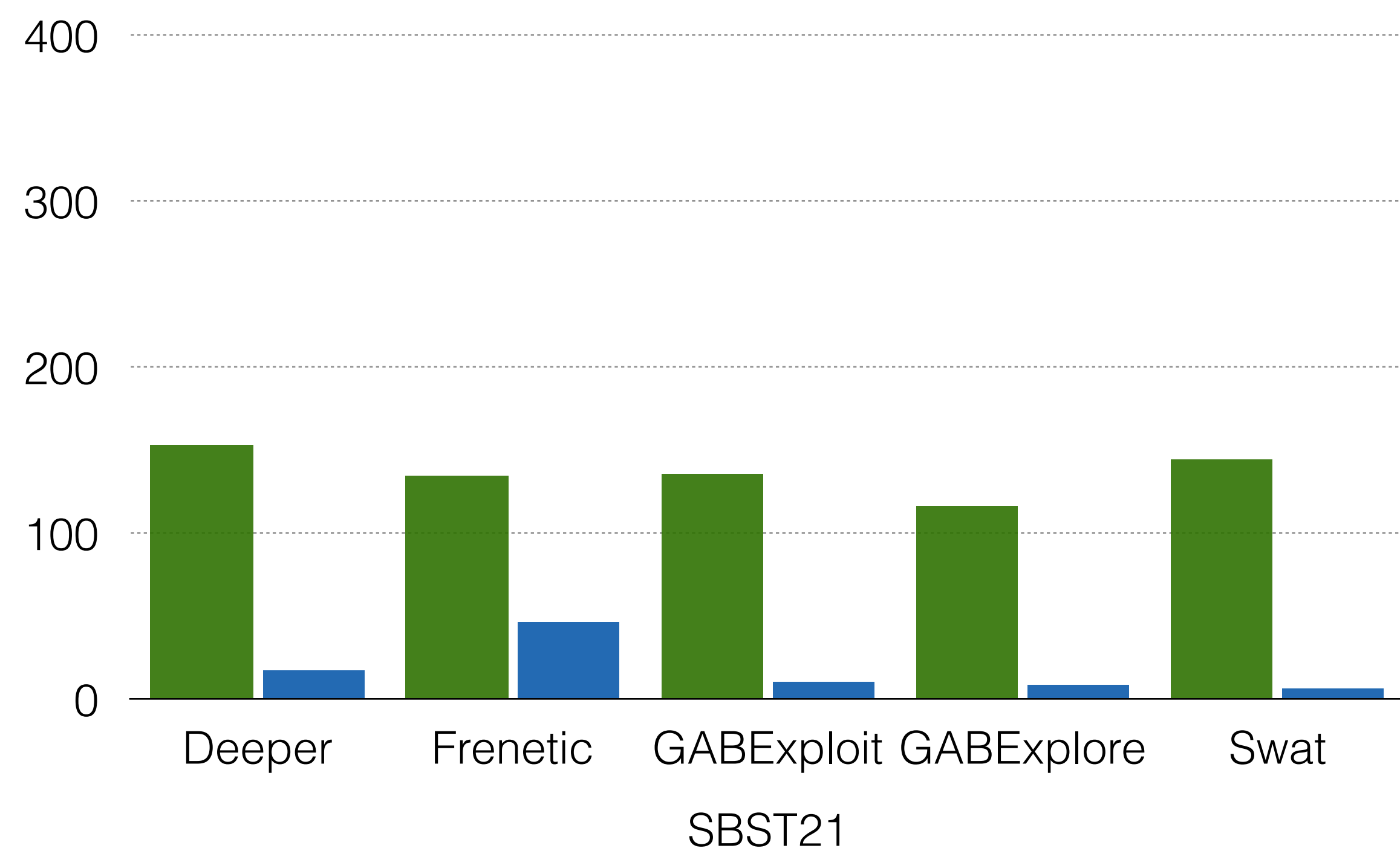
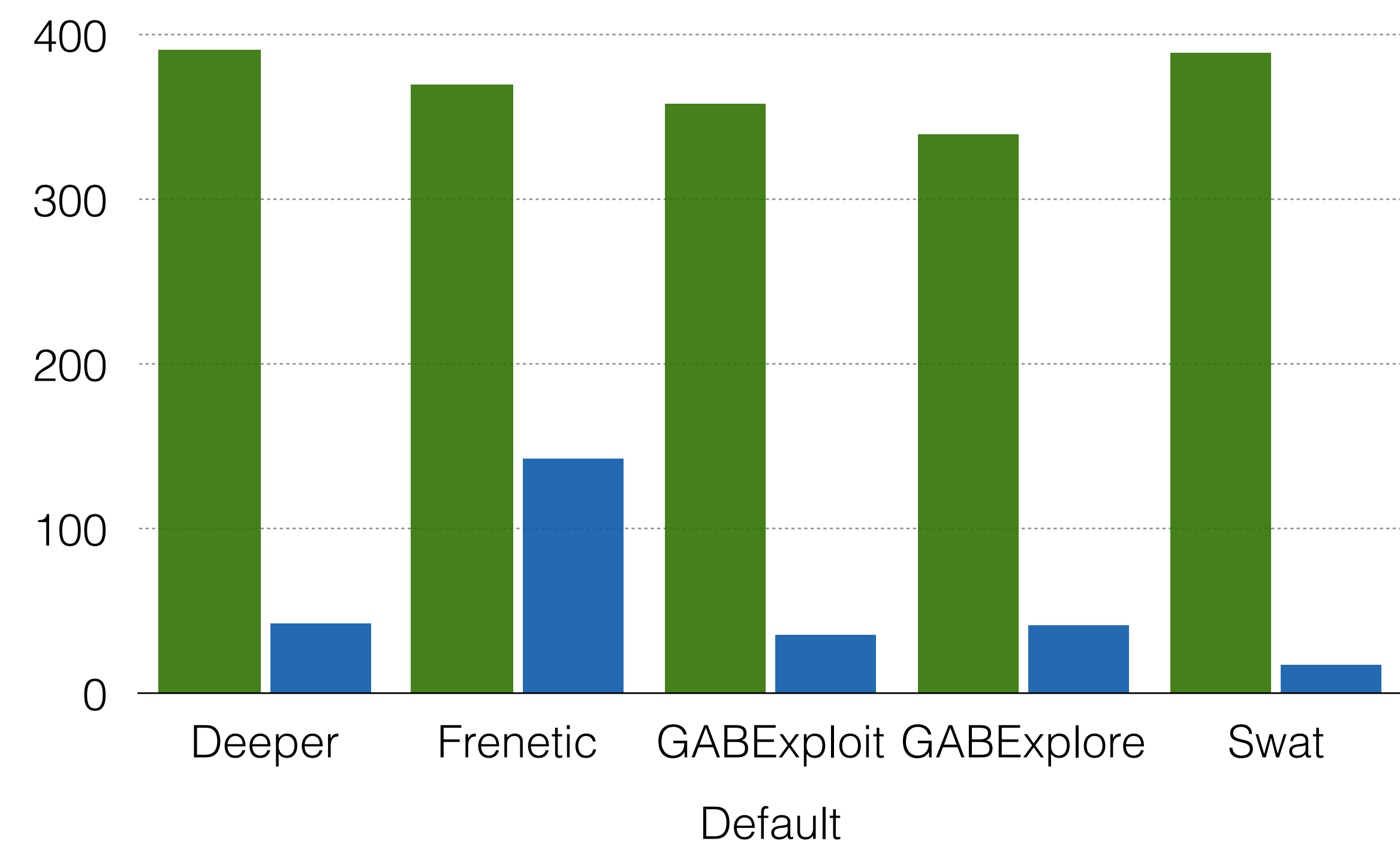
Results: # OBEs



Results: Failure Sparseness



Results: Efficiency and Effectiveness



■ # Valid Test Cases
■ # Invalid Test Cases

Towards Interpretable Failures

- Evaluate test input generators for self-driving software using interpretable **feature maps** (e.g., map coverage)

DEEPHYPERION: Exploring the Feature Space of Deep Learning-Based Systems through Illumination Search

Tahereh Zohdinasab
Università della Svizzera Italiana
Lugano, Switzerland
tahereh.zohdinasab@usi.ch

Alessio Gambi
University of Passau
Passau, Germany
alessio.gambi@uni-passau.de

Vincenzo Riccio
Università della Svizzera Italiana
Lugano, Switzerland
vincenzo.riccio@usi.ch

Paolo Tonella
Università della Svizzera Italiana
Lugano, Switzerland
paolo.tonella@usi.ch

ABSTRACT

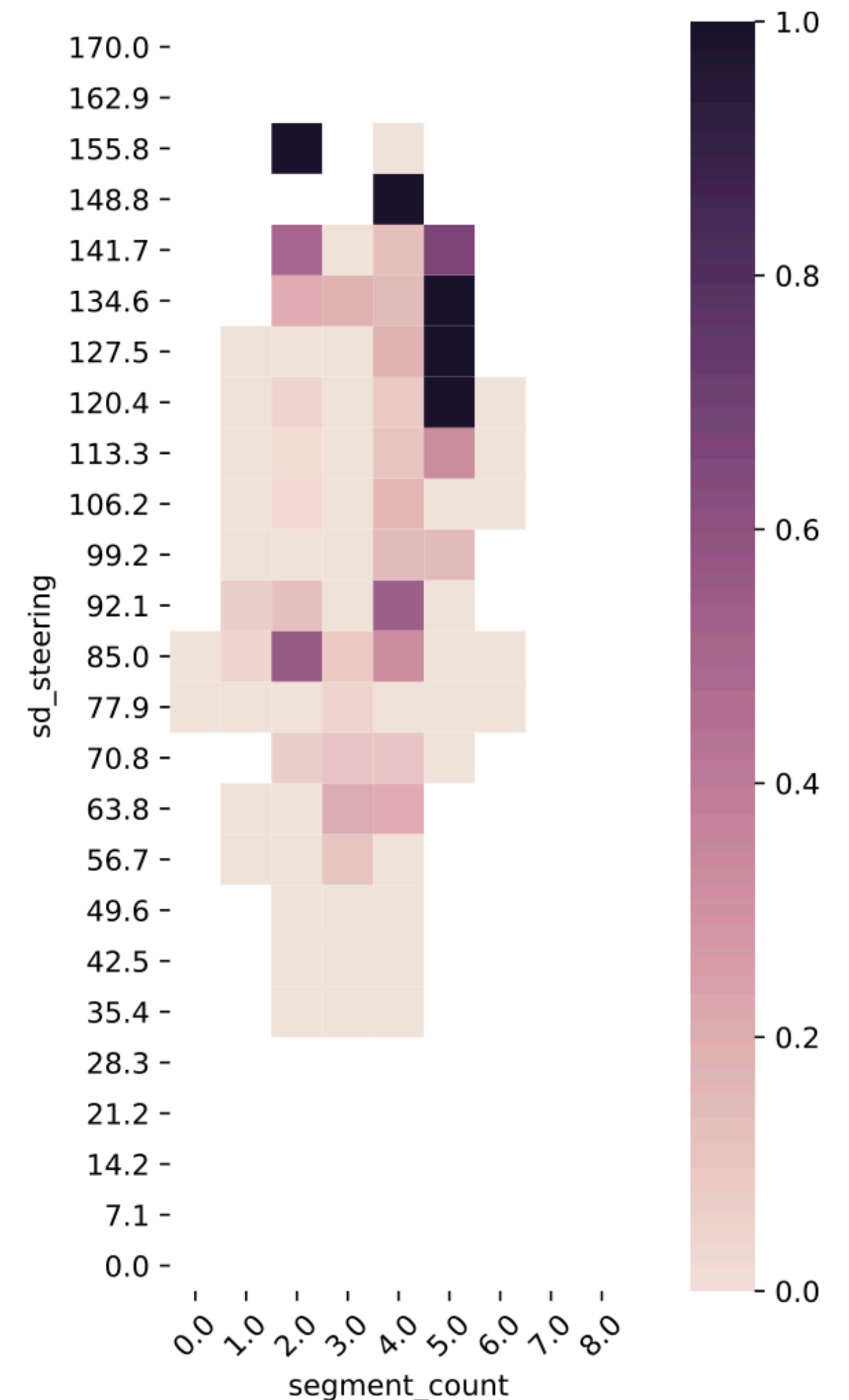
Deep Learning (DL) has been successfully applied to a wide range of application domains, including safety-critical ones. Several DL testing approaches have been recently proposed in the literature but none of them aims to assess how different interpretable features of the generated inputs affect the system's behaviour.

In this paper, we resort to Illumination Search to find the highest-performing test cases (i.e., misbehaving and closest to misbehaving), spread across the cells of a map representing the feature space of the system. We introduce a methodology that guides the users of our approach in the tasks of identifying and quantifying the dimensions of the feature space for a given domain. We developed DEEPHYPERION, a search-based tool for DL systems that illuminates, i.e.,

1 INTRODUCTION

Deep Learning (DL) has become an essential component of complex software systems, including autonomous vehicles and medical diagnosis systems. As a consequence, the problem of ensuring the dependability of DL systems is critical.

Unlike traditional software, in which developers explicitly program the system's behaviour, one peculiarity of DL systems is that they mimic the human ability to learn how to perform a task from training examples [22]. Therefore, it is essential to understand to what extent they can be trusted in response to the diversity of inputs they will process once deployed in the real world, as they could face scenarios that might be not sufficiently represented in the data from which they have learned [13].



Lessons Learnt

- Adopt open infrastructure and intuitive APIs
- Involve and grow the community
- Pull requests are welcome:
<https://github.com/se2p/tool-competition-av>
- Join the discussion on:
<https://join.slack.com/t/driversity>



Don't Drink & Drive



What's Next?

- New test subjects
 - Learning-Based driving agents
 - Path/Trajectory planners
- Training test subjects based on competitors to avoid (representation) bias
- Larger test space/new driving tasks:
 - Environment, weather, 3D roads
 - Obstacles, traffic
- "Open" submission (continuous evaluation)



Search-Based Software Testing Tool Competition 2022

Co-chairs 2021



Sebastiano Panichella

Zurich University of Applied
Science (ZHAW)



Alessio Gambi

Passau University



Fiorella Zampetti

University of Sannio



Vincenzo Riccio

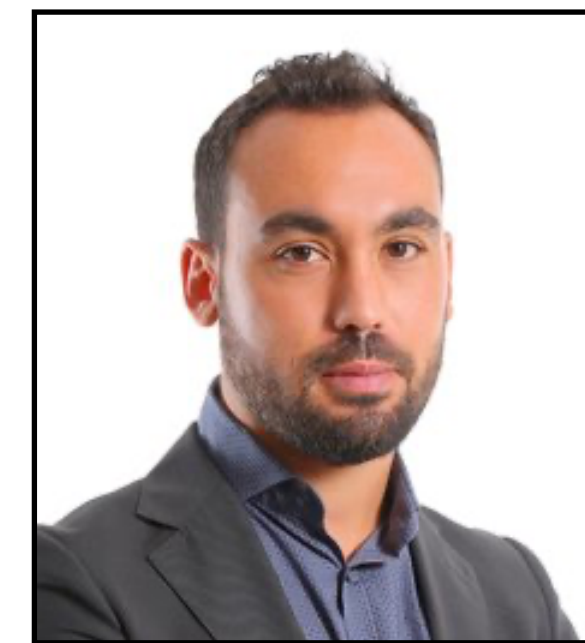
University of Lugano

Co-chairs 2022



Fiorella Zampetti

University of Sannio



Vincenzo Riccio

University of Lugano



Co-chair(s): You? Co-chair(s): You?