SWAT tool at the SBST 2021 Tool Competition

Dmytro Humeniuk
Polytechnique Montréal
Montreal, Canada
dmytro.humeniuk@polymtl.ca

Giuliano Antoniol
Polytechnique Montréal
Montreal, Canada
giuliano.antoniol@polymtl.ca

Foutse Khomh
Polytechnique Montréal
Montreal, Canada
foutse.khomh@polymtl.ca

***Abstract*—SWAT is a test case generating tool for testing cyber-physical systems (CPS). In the context of SBST 2021 CPS testing competition, it has been adapted to generating virtual roads to test a lane keeping assist system. It has achieved the best ratio between valid and generated test cases, producing over 95% valid test cases in both testing configurations.**

***Index Terms*—test cases, virtual roads, competition**

I. Introduction

Cyber-physical systems, such as cars with adaptive cruise control, robotic systems or smart buildings are becoming an important part of the modern world. Testing such systems poses new challenges, not yet addressed by software quality assurance techniques. Therefore, in addition to the Java unit testing tool competition, CPS testing tool competition was introduced this year as a part of the SBST2021 Workshop[1]. This novel competition aims to promote open research on the challenges of CPS testing. This year it was dedicated to generating test cases for a lane keeping assist system. In this paper we describe the principle of operation of the SWAT tool, submitted to the competition. The results of SWAT tool performance are outlined in the competition report [1].

II. SWAT tool description

In the current implementation, the tool uses Markov process to generate the test cases. As the input user specifies the time budget as well as the size of the two-dimensional map to use for scenarios. The tool generates the virtual roads as a sequences of points, defining the road spine. These roads are further transferred to BeamNG [2] based self-driving vehicle simulator for evaluation. The goal is to generate valid roads that force the ego-car, i.e., the test subject, to drive off its lane. According to the competition rules [2], the main requirements for valid roads are:

1) roads must never intersect or overlap;
2) turns must have a geometry that allows the ego-car to completely fit in the lane while driving on them; turns with small radius, are disallowed;
3) roads must completely fit the given squared map boundaries.

In the following subsections we describe how the SWAT tool achieves valid road generation, how to use SWAT as well as our plans for future work on improving the tool performance.

[1]https://github.com/se2p/tool-competition-av

[2]https://github.com/se2p/tool-competition-av/blob/main/documentation/GUIDELINES.md

A. SWAT tool principle of operation

As inspired by the AsFault tool [3], the virtual road generation starts by creating a road backbone vector, defined by two points. We further refer to it as to ”road vector” . The vector length corresponds to a two lane road width, i.e. 8 meters. The vector end is defined as the point with the highest Y-coordinate. The initial position is randomly selected in the middle of one of the map margins. Then, affine transformations are applied to the initial vector. Finally, the road defining coordinates are calculated as the transformed vectors middle points. We use three types of transformations:

- parallel transition, corresponding to road going straight;
- clockwise rotation, corresponding to road turning right;
- anticlockwise rotation, corresponding to road turning left.

In the fig.1 you can see the obtained road spine points (yellow) after applying affine transformations to the initial vector a, selected among other possible initial vectors b, c and d. The sequences of transformations are defined automatically,

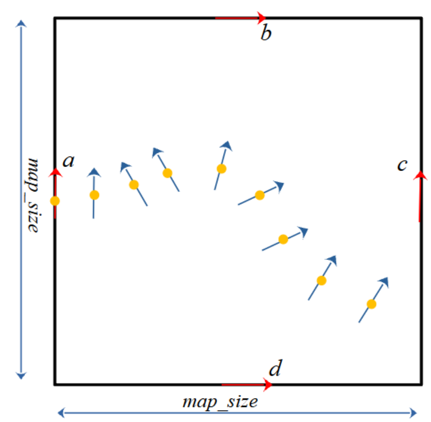

Fig. 1. Possible initial vectors a, b, c and d (red). Road spine points (yellow) generated after applying vector affine transformations (blue) starting from vector a

using the Markov chain with three states: ”go straight”, ”turn right” and ”turn left”. The designed Markov chain with the probabilities for changing states is shown in the fig.2. We fine-tuned the probabilities empirically, so that on average, longer loads are produced. To each selected state, the value is randomly assigned from the list of accepted values. In current configuration, the range of values for going straight is from 5 to 50 meters, for turning left or right - from 10 to 70 degrees.

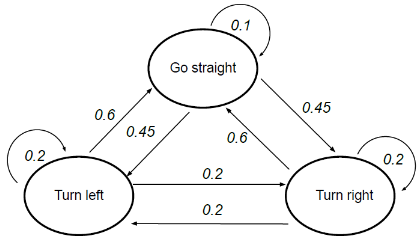

Fig. 2. Markov chain for generating virtual roads

The next challenge is to construct valid roads from the given sequences of commands, i.e. "go straight N meters", "turn left A degrees", "turn right A degrees", etc. Examples of performing "turn left 15 degrees", "turn right 15 degrees" and "go straight N meters" transforms to a vector are shown in figures 3, 4 and 5 respectively.

To perform the rotation, it's important to correctly select the rotation axis, which can be below or above the road vector. The choice depends on the direction in which the road vector moves along the road map. For a left turn, the vector is rotated anti-clockwise around the axis, for the right turn - clockwise.

Our initial strategy was to infer the direction of the movement from the location of the previous vector. This approach, however, produced a relatively high number of invalid roads. The strategy that worked best and that is implemented in the SWAT tool is illustrated in the figures 6 - 9, where a left turn is performed and vector is moving from position A to B.

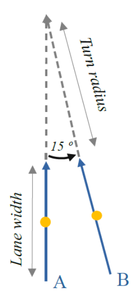

Fig. 3. Turn left transform

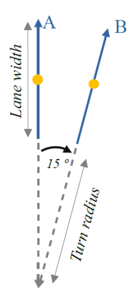

Fig. 4. Turn right transform

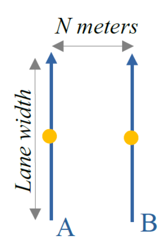

Fig. 5. Go straight

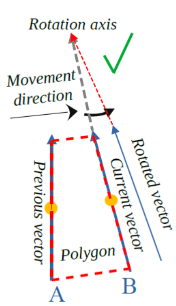

Fig. 6. Correct choice of the axis, rotated vector is outside the polygon

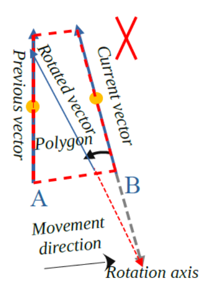

Fig. 7. Incorrect choice of the axis, rotated vector is inside the polygon

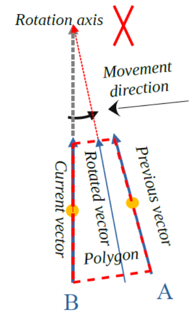

Fig. 8. Incorrect choice of the axis, rotated vector is inside the polygon

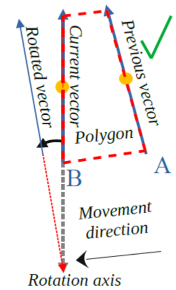

Fig. 9. Correct choice of the axis, rotated vector is outside the polygon

First, the rotation axis is selected to be above the road vector. If after the anti-clockwise rotation (clockwise for right turn) for a small angle value (2 degrees) the rotated vector appears inside the polygon defined by the current and previous road vectors, the location of rotation axis is changed and transform is performed using the new location. If the rotated vector is outside the polygon, the transform is performed using current axis location. The intuition is that the new vector position should not intersect with the path defined by previous vectors.

The road generation stops, when the road vector goes out of the road map bounds.

B. *How to use*

The tool is open source and available at [4], where the instructions on how to run it are provided. Currently it is integrated in the SBST 2021 CPS competition pipeline.

C. *Future improvements*

Current version of the tool was proven to be effective at generating valid virtual roads, achieving the highest ratio between valid and generated test cases among four other tools. The fault revealing power of the generated tests can be improved by adding a search-based technique. We are currently working on implementing a genetic algorithm with two fitness functions: test case diversity and difference between expected car path (generated road) and simulated car path. To evaluate the second fitness function, a simplified car model will be used to execute the test cases generated by the Markov chain, as described in [5].

References

[1] Sebastiano Panichella, Alessio Gambi, Fiorella Zampetti, and Vincenzo Riccio. Sbst tool competition 2021. In *International Conference on Software Engineering, Workshops, Madrid, Spain, 2021*. ACM, 2021.

[2] BeamNG GmbH. BeamNGpy. https://github.com/BeamNG/BeamNGpy, 2021.

[3] Alessio Gambi, Marc Mueller, and Gordon Fraser. Automatically testing self-driving cars with search-based procedural content generation. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 318–328, 2019.

[4] Dmytro Humeniuk, Giuliano Antoniol, and Foutse Khomh. Swat tool. https://github.com/dgumenyuk/tool-competition-av.git, 2021.

[5] Dmytro Humeniuk, Giuliano Antoniol, and Foutse Khomh. Data Driven Testing of Cyber Physical Systems. arXiv:cs.CR/2102.11491, 2021.