ASSURING AUTONOMY
INTERNATIONAL PROGRAMME

2.7 – Using simulation

Practical guidance – cross-domain and automotive

Authors: Dr Philippa Ryan, Dr Gareth Fetcher (Adelard LLP), and Kazuki Imai (Witz Corporation)

Simulation has emerged as one of the most important means of assurance for Machine Learning (ML) embedded in control systems. This guidance presents a summary of issues relating to the use of simulation in assuring ML, as well as experience gained from the TIGARS demonstrator project in the autonomous vehicle domain.

It should be noted that simulation discussions in this guidance are limited to simulation environments for verification and reinforcement learning of ML, rather than, for example, simulations of overall traffic flow once autonomy has been incorporated.

Simulation is an approach widely used and encouraged (e.g. by the NHSTA [1]) to train and verify the performance of ML used in autonomous vehicles. Simulation can be performed at many different levels of abstraction, some of which are described below:

- Fully virtual simulation – where the ML is executed in isolation with fully electronic input data and data capture (e.g. running an image classification Convolutional Neural Network (CNN) on a PC with sample image file)
- Hardware in the Loop (HIL) – where the ML is run on representative hardware, however the inputs and outputs are managed virtually or in an artificial environment (e.g. putting an autonomous vehicle inside a room with a bank of monitors and capturing decisions via data logging). Simpler cut down versions may also be used (e.g. a sub-system in isolation but with hardware sensors)
- Real-world limited trial – where the autonomous system is run on representative hardware but in a controlled environment such as on a test track
- Real-world trial – where the autonomous system is put into the public environment, with no control of test conditions

Simulation may require substantial computer resources to create an environment with enough fidelity to gather meaningful results.

Table 1 below summarises the pros and cons of different combinations of virtual and real-world simulation. In practical terms it may be desirable to use different types at different stages of ML development. This would be dependent on the risk associated with the system, as that would inform the amount of evidence required to demonstrate adequate safety.

Environment	ML	Strengths	Weaknesses
• Virtual	• Virtual	• Can control and model many different environment options, which may be hard to replicate in real world testing • Can create accident sequences to test corner cases without risk of accident • Potentially cheap and quick • Can do early in lifecycle to assess performance • Can monitor every aspect of performance • Can use for reinforcement learning • Potentially strong repeatability • Easier to detect how/where faults occurred with monitoring	• Unrealistic input data e.g. computer generated environment* or modelled sensor functionality which may not match the resolution and real-time performance of a real sensor • Extensive computer resources will be required to achieve the performance required for adequate modelling and collecting data e.g. in terms of processing power and fast access memory • ML may not perform this way in real life • Hard to involve user if needed • Potentially unrepresentative results (e.g. no feedback from bumpy surface, compromise of equipment from wet surface, temperature changes)
• Virtual / artificial	• On target hardware (Hardware in Loop (HIL))	• Can control many different environment options • Can create accident sequences with very limited or no risk • Can involve end user • Gain trust in ML hardware • Potentially strong repeatability • Easier to detect how / where faults occurred with monitoring	• Unrealistic input data – ML may be real but some of the input data may not be realistic e.g. if working in a room with lots of monitors • Computing power required may be large • Outputs may be more realistic but still constrained by environment (e.g. no actual movement or slower/faster responses) • User may not behave as they would in real environment or may have simulation sickness [2]
• Real world but controlled (e.g. test track)	• On target hardware	• Input data is real and may contain unanticipated events • Can get useful feedback on performance with low risk to third party • Can involve users if needed	• Less control over the environment • Much harder to repeat results • Harder to detect how/where faults occurred
• Real world trials	• On target hardware	• Input data is real and may contain unanticipated events • Can involve users if needed	• No control over environment • Riskier to third parties depending on mitigations in place • Hard to repeat results • Hard to detect how / where faults occurred

*Consider the situation where the simulation provides conflicting and unrealistic sensor data (e.g. blocky low-resolution models, moving trees and unrealistically fast pedestrians [3]). While it might be useful for the ML to identify this as invalid input data, if used for training care will be needed not to reinforce invalid behaviour.

Table 1 – Simulation variants and their strengths and weaknesses

Different combinations of virtual and real-world simulations can be used and, in practical terms, it may be desirable to use different types at different stages of ML development. This would be dependent on the risk associated with the system, as that would inform the amount of evidence required to demonstrate adequate safety.

Summary of approach

- Simulation can have many roles in the development and assurance lifecycle: the roles of the different simulation variants should be specified and justified
- Confidence in the simulation environment needs to be established. In other words, how much we can trust it, and how much do we need to trust it. This will include:
    - confidence in any simulation software (in the quality of its construction)
    - confidence in the fidelity of the sensor data compared to real-life
    - trust in the results produced (both positive and negative)
- Although many tools are available off the shelf to support simulation, in our experience, they did not perform as anticipated (ViViD had many timing issues) and they may not have been developed to the quality traditionally expected for safety critical systems testing
- Adjustments in system behaviour may be needed to accommodate the simulation environment and these will need to be justified so that test evidence can be used in the overall assurance cases

Further details on this guidance can be found in [5].

Example of application of guidance

This section provides an overview of the simulation performed on the TIGARS Evaluation Vehicle (TEV) golf cart, with an acceleration control system containing ML. The study was undertaken to elucidate the gaps between actual and simulation environments for testing systems including ML models. It is used to highlight pragmatic issues in using simulation on real projects using off the shelf components integrated with bespoke software, by a sub-system developer.

The ML used was a version of the You Only Look Once (YOLO) [4] CNN which has been trained to detect people and vehicles, as well as other objects. The system under test uses a combination of distance calculations via parallax images, LiDAR and image classification to determine speed and acceleration settings. The system responds to other vehicles and pedestrians in its environment depending on their type(s) and distance from the vehicle.

Tests were conducted in the following two simulation environments:

- TEV test room – combination of virtual and artificial environment with HIL
- Virtualized Verification into automatic Driving (ViViD) – fully virtual environment

TEV test room environment

These tests were conducted using a chassis dynamo. In the chassis dynamo environment, the TEV runs over the dynamo rollers. During the test, an environmental situation is reproduced by installing a panel of a person or a car in front of the golf cart. Since the space in which the chassis dynamo can be used is narrower than real life, the threshold values of the distance from the front vehicle when accelerating, decelerating, or stopping were adjusted proportionally. The configuration of the TEV test room is shown in Figure 1. The chassis dynamo environment is seen in Figure 2.

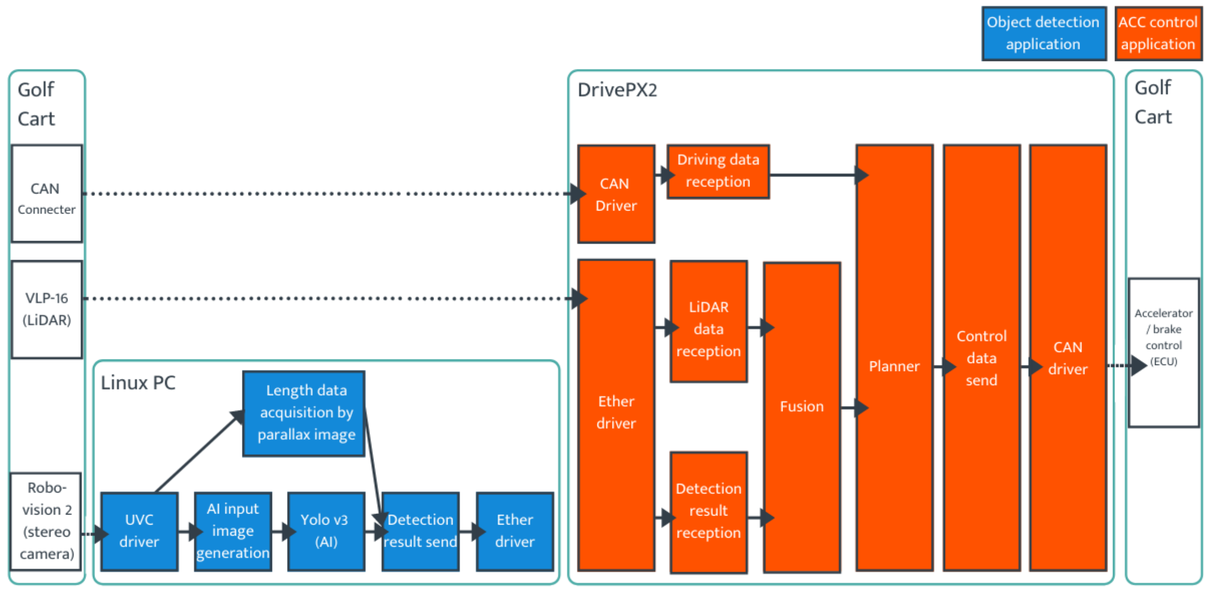

Figure 1: TEV test room configuration diagram

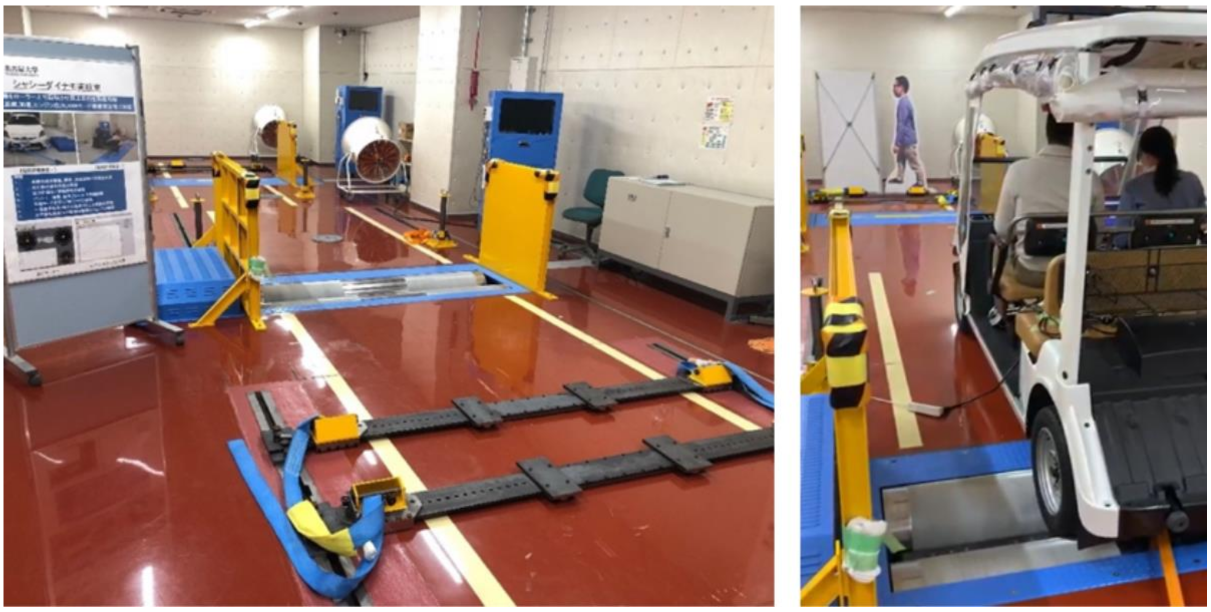

Figure 2: Chassis dynamo environment

ViViD environment

ViViD provides a fully virtual simulation environment for the TEV golf cart. Using ViViD, sensor information can be acquired by User Datagram Protocol (UDP) communication. It can be configured so that obstacles such as vehicles and pedestrians can be inserted into the environment, as well as failure injection within sensor data. The configuration of the ViViD environment is shown in Figure 3. The tests were carried out with the TEV driving on a typical road as shown in Figure 4.

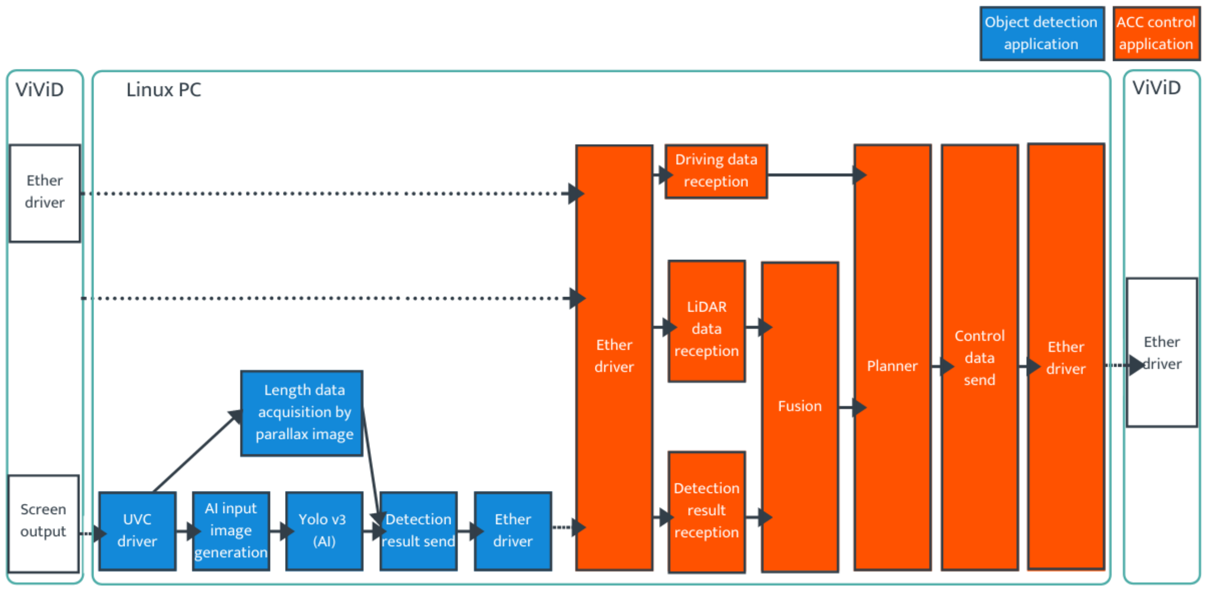

Figure 3: ViViD configuration diagram

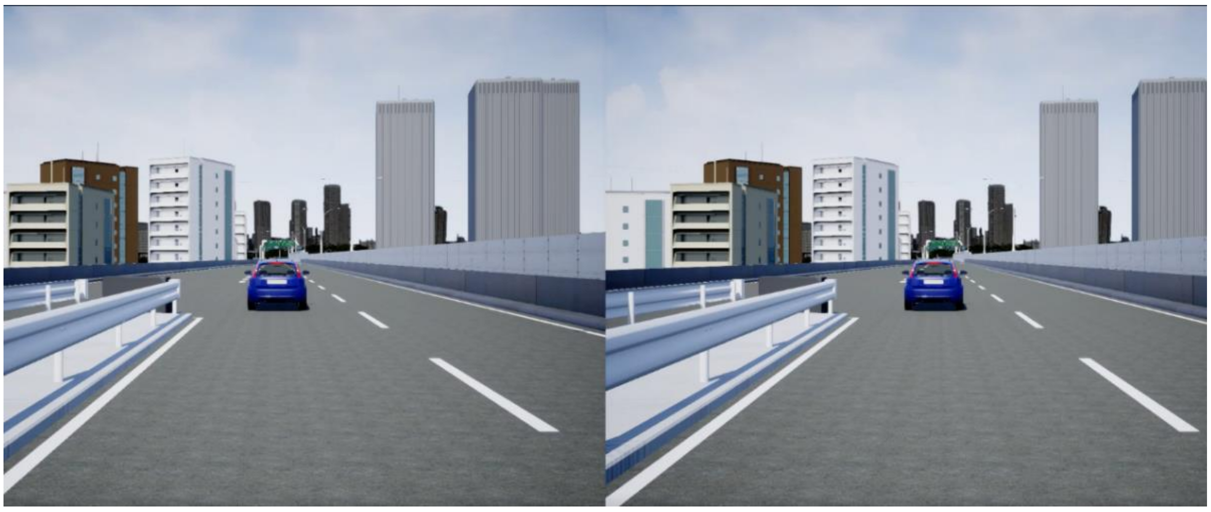

Figure 4: ViViD environment

Experimental findings

The results of the experiments described above are discussed below.

The system on the TEV uses comparisons of distance information from the object detection and LiDAR. There were many issues with timing in the ViViD environment which impacted on the effectiveness of the testing.

The LiDAR simulation software was too slow to be used at full fidelity in ViViD. As a result only part of the LiDAR data (the front ±15 degrees) was used to ensure a similar execution time as the real system. This was justified as it had no impact on the test cases being run within the ViViD environment.

A very high specification machine was required to run the test application and simulator together, otherwise there were unacceptable delays sending the video output to the test application and in running the YOLO component. Even then, there were issues providing a predictable frame rate from the simulator, since only approximately 57 seconds of real-time data could be processed in around 1 minute. This had a cumulative effect on the simulation.

There were further complications with variations in the execution cycle, which changed from test to test. This meant that the tests were not repeatable. Only a rigid real-time execution of the simulator would have solved this, something that was impractical with off the shelf software. An attempt was made to lock-step time stamps from the LiDAR and object detection with the slowest input data, but the overall time lag meant this was not a complete solution. One important knock-on effect of the lack of repeatability is on regression testing. Tests cases re-run on a changed system cannot be assumed to execute with the changed functionality as the only variant, so the results of regression testing would need to be closely examined to ensure the results observed are valid and representative.

It was found that small scale laboratory experiments were not easy and unexpected problems were encountered. For example, after scaling the parameters of the tests due to the small amount of laboratory space available, the TEV then experienced large variations in the acquired distance from the golf cart vision sensors. The sensors had had relatively high accuracy when detecting objects at a distance originally assumed in the TEV specification. However, as it is complex and expensive to prepare a testing environment that is very similar to the actual deployment environment (e.g. test tracks or large scale experiments) a 'good' simulator may be better suited in some cases and was still felt to provide value.

References

- [1] Automated Driving Systems 2.0 – A vision for safety, NHSTA, September 2017, https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/13069a-ads2.0_090617_v9a_tag.pdf last accessed Nov 2019.
- [2] The Challenges of V&V for Connected and Autonomous Vehicles, S Khastigir, Presentation at Autonomous Systems V&V Workshop. February 2018. https://vavasdotorg.files.wordpress.com/2018/03/the_challenges_of_vv_for_connected_and_autonomous_vehicles-sk-20180201.pdf last accessed Nov 2019.
- [3] Testing Autonomous Vehicle Software in the Virtual Prototyping Environment, B Kim et al. IEEE Embedded Systems Letters, Vol. 9, No 1, March 2017.
- [4] "What's new in YOLO v3?" https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b last accessed October 2019.
- [5] Bloomfield, R., Fletcher, G., Khlaaf, H., Ryan, P., Kinoshita, S., Kinoshit, Y., Takeyama, M., Matsubara, Y., Popov, P., Imai, K. and Tsutake, Y., 2020. Towards Identifying and closing Gaps in Assurance of autonomous Road vehicleS - A collection of Technical Notes Part 2. arXiv preprint arXiv:2003.00790.