Autonomous Cars & Software Testing – Part 3 of 3

Introduction

This is the third part of a three-part article on autonomous cars and software testing.

The first part of the article introduced the reasons for developing autonomous cars and the high-level technological challenges that are involved. It then considered some of the ethical issues, and suggested areas where regulation is needed.

In the second part, the new technologies that are needed for autonomous systems were described in more detail, such as in the areas of sensors, vehicle-to-vehicle communications, and machine learning.

This part provides explanations of how traditional automotive lifecycle practices of requirements specification and architectural design must change for autonomous cars. The article then suggests how autonomous cars will provide new challenges and opportunities for software testing.

The first two parts are aimed at anyone who wants to understand more about autonomous cars, while this part is more focused at those involved in managing or performing software testing of the autonomous systems inside them.

Specifying Requirements for Autonomous Vehicles

The main difference between the requirements for autonomous cars and other vehicle systems is in the specification of how the autonomous system should perform the 'decision-making' function, as shown in Figure 1.

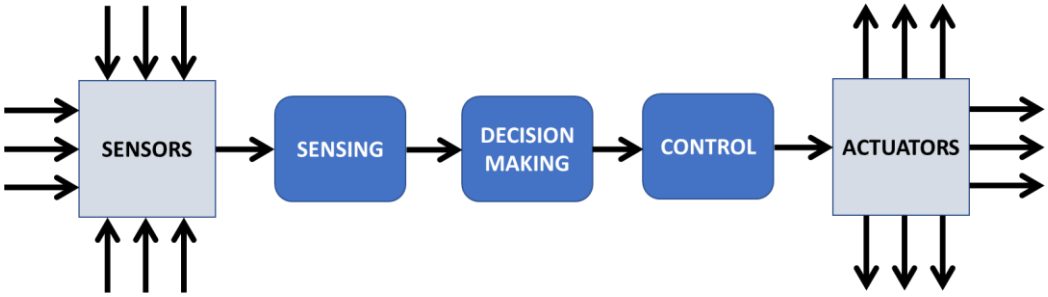

Figure 1: Basic Autonomous Car Functions

For a SAE Level 5 autonomous system [1], in which the car takes complete control away from the driver (e.g. no steering wheel), this requires the system to be able to react to any driving situation safely with no human input. In some ways this makes it easier to specify the requirements of such a system, as we know it must be able to cope with anything a car could encounter with no restrictions or constraints, so the requirement is quite clear, if exceedingly difficult to deliver.

However, fully autonomous cars are still quite some way into the future, and before then it is highly likely that we will first produce SAE Level 3 and 4 cars. For these lower levels, the range of situations in which the autonomous system should be able to fully control the car is constrained to a specific set of driving scenarios (i.e. specific functionality in lower risk environments, such as keeping a safe distance at low speed). In terms of requirements specification, these driving scenarios can be considered in two parts. The first part defines the function provided by the autonomous car and second part defines the set of constraints within which we expect this function to be able to operate safely.

Defining Constraints for the Autonomous Functions

In the US NHTSA 'Federal Automated Vehicles Policy', published in 2016 [2], one of the key parts of the safety assessment is the definition of the 'Operational Design Domain (ODD)' for the autonomous car. According to the NHTSA, at a minimum, this ODD would include the following information to define the constraints on a function:

- Roadway types (interstate, local, etc.) on which the function is intended to operate safely;
- Geographic area (city, mountain, desert, etc.);
- Speed range;
- Environmental conditions in which the function will operate (weather, daytime/night-time, etc.); and
- Other domain constraints.

When the car leaves the constraints of the ODD (e.g. the car moves onto a different road type or the weather conditions change – or the system determines it is failing), the system is expected to safely hand back control to the human driver. As was discussed in one of the earlier parts of this article, this handover is known to be a dangerous area, as the time taken for a driver to regain situational awareness can be quite long. It is reasonable to expect systems to put the car into a minimal risk situation (e.g. pulled over at the side of the road with hazard warning lights flashing) if the car's driver does not take back control within a reasonable time.

Because autonomous cars will eventually have to be regulated (the NHTSA Policy is still only guidance), the different ODDs will need to be specified in a consistent (ideally standard) format. Various schemes have been proposed for specifying these constraints, and the AdaptiVe (Automated Driving Applications Technologies for Intelligent Vehicles) project [3], which was co-funded by the EU and various car manufacturers, has created a detailed classification scheme for those factors that affect autonomous car functions. At the top level, this scheme uses three categories (vehicle, driver and environment) that need to be defined. As an example, to demonstrate the level of detail required by the scheme, the environment is broken into the sub-categories of road, visibility and traffic, which, in turn, is described by the following parameters:

- mixed traffic (yes, no)
- traffic participants (non-motorized, motorized: slow, motorized: fast)
- traffic flow (moving traffic, slow moving traffic, stationary traffic)
- road type (motorway, highway, interstate, rural road, arterial road, urban road, residential district road, parking area/parking deck and garage)

- road accessibility (public, private)
- road condition (good, slippery, bumpy)
- road geometry (straight, curved, steep)
- road infrastructure (physical cut-off, good lane markings, guardrails, deer fences, emergency lanes, hard shoulder and traffic lights).
- good visibility
- reduced visibility due to obstacles (vehicles, infrastructure)
- reduced visibility due to weather (fog, heavy spray, heavy rain, heavy snow).

Each of the above traffic parameters is subsequently described in more detail in the AdaptiVe scheme documentation [3].

Defining Autonomous Vehicle Functions

In addition to understanding the ODD, which provides constraints on where and when the autonomous functions can operate, we also need to specify the actual functions provided by the autonomous systems.

The classification scheme from the AdaptiVe project [3] defines, in detail, 33 separate functional classes for autonomous car systems from SAE levels 1 to 5:

- Cruise Control
- Adaptive Cruise Control (ACC)
- Lane Keeping Assistance (Type I – correction when needed)
- Lane Keeping Assistance (Type II – continuous steering)
- Lane Keeping Assistance (Type III – continuous steering – no human)
- Active Lane Change Assistance
- Combined ACC and LKA Type II
- Active Traffic Light Assistance
- Narrowing Assistance
- Construction Zone Assistance
- Traffic Jam Assistance
- Highway Assistance
- Overtaking Assistance
- Parking Assistance with steering
- Parking Assistance with steering and accelerating/braking
- Key Parking
- Traffic Jam Chauffeur
- Highway Chauffeur
- Overtaking Chauffeur
- Platooning
- Driverless Valet Parking
- Tele-Operated Driving – Urban

- Traffic Jam Pilot
- Highway Pilot
- Overtaking Pilot
- Urban Robot Taxi
- Automated Mining Vehicles
- Automated Marshalling of Trucks
- Universal Robot Taxi

Autonomous Car Safety Rules

If we can define the functions that are provided by the autonomous car (see above) and the environmental constraints within which these functions are expected to work safely (e.g. the operational design domain), the third, and arguably most important part of the specification of requirements are the rules which these functions must work within. These rules should define the safety constraints for the autonomous car functions.

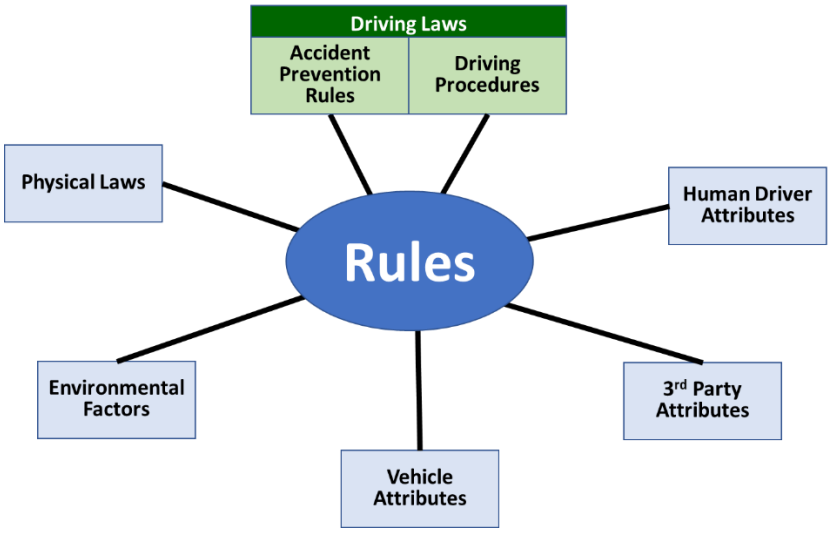

Figure 2: Factors affecting Autonomous Driving Rules

The set of rules used by the decision-making function of the autonomous system are derived from a variety of sources, as shown in Figure 2. The legal driving laws should be used to identify those rules that are there to prevent accidents (e.g. speed limits, safe braking distances) and those procedures that ensure an efficient flow of traffic (e.g. give right of way to traffic on a roundabout). A problem with driving laws is that they are country-specific and will need modification for each country where an autonomous car is sold. Physical laws need to be considered as these are used along with vehicle attributes (e.g. weight), environmental factors (e.g. road surface) and human driver attributers (e.g. reaction time) to calculate actual stopping distances at different speeds. The rules also need to consider 3rd party attributes, such as the speed of pedestrians (who might rush out between parked cars in front of us) and other road infrastructure, such as barriers.

Ideally these rules would be common across all autonomous car systems, but initially they will have to be country-specific because they must align with country-specific legislation (e.g. driving laws), so that human drivers not using autonomous cars do not need to learn new 'rules of the road' when they interact with autonomous cars. However, those rules that are derived simply for safety (i.e. to avoid collisions) should be able to be kept common across all countries (accidents, and the need to prevent them, are similar everywhere). For instance, you can imagine a 'safe distance to car ahead' rule. This rule could define a formula for determining the minimum distance that should be left to the car in front, dependent on the speeds of the two vehicles, the road surface, the visibility to the car in front, and the maximum rate of braking of the car. Once agreed, such a rule would form part of the regulations for any autonomous system that included a function that controlled the distance to a vehicle in front (e.g. appropriate for adaptive cruise control, but not for self-parking), and could be applied as shown in Figure 3. Here we see that the autonomous system is implementing an adaptive cruise control function – and so would only need to apply those safety rules that were relevant to keeping a safe distance to the vehicle in front (e.g. any rules concerned with the safety of turning the vehicle out of the lane would be irrelevant – and covered by a separate rule).

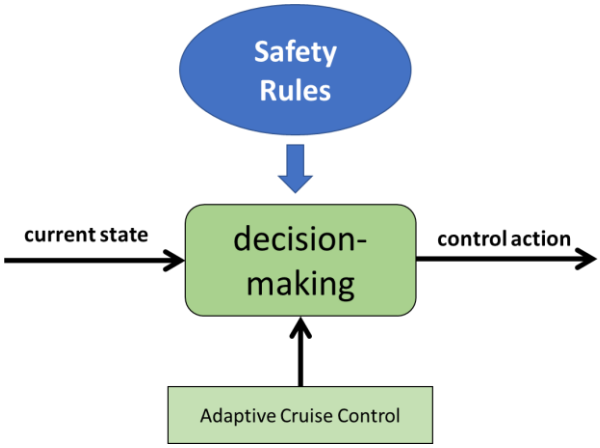

Figure 3: Decision-Making with Rules

Other similar safety rules could be defined, for example, for avoiding collisions with pedestrians who step out into the road, and for overtaking or pulling in front of other vehicles. The developers of autonomous systems would need to ensure their systems complied with these safety rules, although they may find that for the comfort of the passengers their systems would not need to always work on the limits defined by these rules. Regulatory bodies would use these safety rules to define the test cases used for certification. In practice, the regulatory body would need to own the rules and be sure that a complete set were identified. An incomplete set would mean that some unsafe scenarios were not tested (and their avoidance probably not implemented by some builders of autonomous systems).

Implementing a complete set of safety rules would not guarantee that the autonomous car would have no accidents. Although the decision-making software would not deliberately issue a command that should result in a collision, there could still be accidents caused by other vehicles, pedestrians, mechanical failures, faults in the sensing function, and defects in the decision-making software.

Mobileye (part of Intel) has designed a framework called Responsibility-Sensitive Safety (RSS) that includes rules for assigning blame for accidents involving autonomous cars, based on a mathematical model (it also covers economic scalability) [4]. This framework, in effect, defines safety rules similar to those described above.

Testing the Software for Autonomous Cars

The testing of the software controlling autonomous cars will need to build on the traditional approaches to software testing that have been around for many decades, which are already required by the ISO 26262 standards [5] and defined in the ISO/IEC/IEEE 29119 [6] and ISO/IEC 20246 [7] standards.

However, autonomous car software introduces some new challenges for software testers. For instance, ensuring the safety of autonomous cars in the many different scenarios they will encounter, will make the testing both a complex and time-consuming activity, which will mean that virtual environments and test automation will be increasingly important. Also, the testing of the machine learning, which is an integral part of the software that implements the sensing and decision-making functions shown in Figure 1, is a new area that is constantly changing. There are also other areas, such as the testing of the driver-autonomous car interface that will need to be carefully considered.

Not Just Traditional Testing

One of the big challenges to testing autonomous car software is the safety-related nature of the systems. Users of autonomous cars will expect these cars to be quite a bit safer than those driven by humans, which will mean that their average serious crash rate (resulting in death or serious injury) will have to be far above 150,000,000 km. Different commentators have estimated different values for the number of kilometres that would need to be driven by test cars to reach statistically significant levels of confidence in the safety of the systems, ranging from one billion to eight billion km of serious crash-free testing, which is not realistic if we expect to see autonomous cars on our roads in the near future.

The complex nature of the systems, and the range of unpredictable environments they will have to work in, may also mean that the traditional approach of using safety cases may become so complicated, due to the number of safety arguments involved and their complexity, that a more measurable approach, such as testing becomes more important.

Testing against Safe Scenarios

In the previous section, on specifying requirements for autonomous cars, we introduced the concepts of specifying the autonomous functions that the system will perform along with corresponding constraints and safety rules that they must follow. These requirements can be considered to be a ‘safe scenario’. Any autonomous vehicle that provides a specific autonomous function will need to follow the relevant safety rules (e.g. not get too close to the car in front) that apply to that function within the defined constraints (e.g. in daylight) and stop providing the function when outside the defined constraints.

From a regulatory perspective, that means the regulatory body would expect to be able to test any vehicle that provides autonomous functions to check that it meets the relevant safety rules, while within the defined constraints. Obviously, the car manufacturer will need to test that their car meets this requirement, but, given recent problems with trust in manufacturer testing, it is likely that the regulatory body will also run their own independent tests.

Although the safe scenarios may be defined in a standard, these scenarios would be used to generate a potentially infinite number of test cases, when the many combinations of different vehicle speeds, weather conditions, road surfaces, pedestrians, road formats, etc. are considered. However, the safe scenarios will be limited to ensuring road safety, and, as much as possible, the number of these scenarios should be kept to a minimum to allow safety testing to be accomplished within relatively short timescales and without costing too much.

In practice, the regulatory body needs to ensure that no test case based on a safe scenario can cause the car to break a safety rule, which means that their test suites may well comprise of many edge-case tests that are more likely to 'break' the system. The car manufacturers will also need to execute such safety-related tests, but they will also need to ensure that a far wider range of scenarios are also tested.

Autonomous Car Two-Part System Architecture

For a manufacturer of an autonomous car, ensuring a car passed the tests based on safe scenarios would be an absolute necessity. First, for the safety of their customers and other road users, and second, to ensure their cars were allowed on the roads and were insurable. However, the car manufacturers will also need to perform a second set of tests that are not based on safe scenarios, but on the non-safety based functionality that is needed to drive the car.

For instance, if an autonomous car function needs to determine and follow a route from A to B, this is not, in itself, safety-related, but it still needs to be tested. In terms of system architecture, the autonomous car systems may well use a two-part architecture, whereby the smaller part of the system manages the safety-related functions, while the larger part of the system manages the remaining functions (and both are kept as independent as possible).

From a testing perspective, the testing of the safety-related functions will necessarily be far more rigorous than for the rest of the system.

Virtual Testing of Autonomous Car Systems

As was mentioned earlier, cars on public roads would need to be tested over distances in the region of billions of kilometres to provide statistically significant confidence in their systems. This is obviously impractical, and so we need a mechanism to speed up the test process. This can be done by using virtual testing on simulators, where we can run many tests in a relatively small timeframe (we can run them much faster than real time) that cover a wide variety of scenarios in different simulated environments.

The hierarchy of different test environments is shown in Figure 4. The pyramid shows that the virtual tests provide the foundation, and vast majority, of the necessary tests (the figure is not to scale – in reality the proportion of virtual tests would be far larger). At this level, we can also test

scenarios that are too dangerous to test in real vehicles. The later tests, initially in a restricted proving ground and then on public roads, provide opportunities to validate the results of the earlier tests and to test scenarios and identify problems that could not have been done in the previous, less realistic, environment.

To be able to provide this level of virtual testing, we will need sophisticated test environments that provide a high level of realism and that can run, record and analyse results very fast. The generation of tests will also need to be automated from defined scenarios, and it is likely that this automated generation will also be based on feedback from earlier tests. This will be an area that is a great opportunity – and challenge – for automotive software testers.

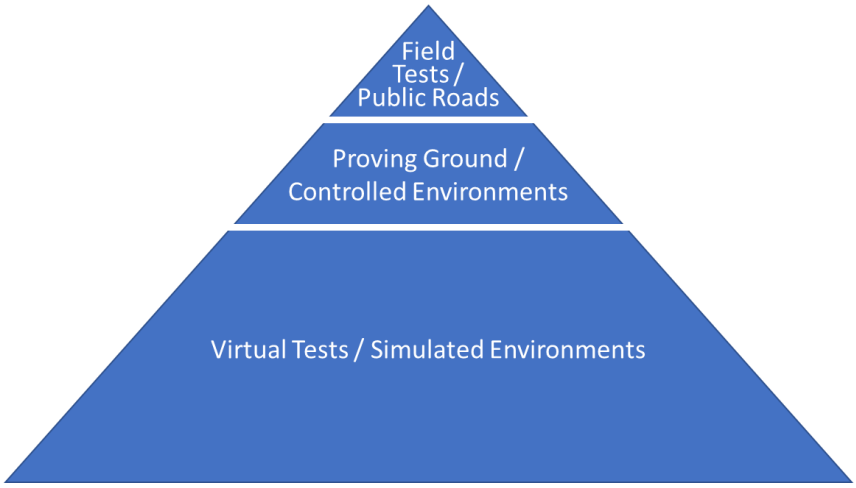

Figure 4: Different Test Levels

Testing Machine Learning Systems

In previous parts of this article, we saw that current implementations of the sensing and decision-making functions of an autonomous car (as shown in Figure 1), are typically based on a form of artificial intelligence, more specifically machine learning. This is because the multi-dimensional optimization that needs to be performed by these functions cannot be done effectively using traditional design and programming techniques. However, testing machine learning systems is quite different from traditional testing for several reasons.

First, the results from machine learning systems are non-deterministic. This means that you cannot predict the exact result that the system will produce, as it may produce different results from the same inputs simply because it uses slightly different starting conditions for its randomized algorithms. For instance, imagine you are testing a route planning function that initially randomly selects many possible routes and then uses a fitness function to select the best of these randomly selected routes. Given the random nature of the how the initial set of routes are identified, you cannot know how good the 'best' route (according to the fitness function) will be. If testing, you would need to set limits on how good the selected route would need to be to pass the test (e.g. the selected route must be less than 195 km rather than selected route must be exactly equal to the shortest possible route of 193.25 km).

Machine learning systems are made up of layers of 'neurons' that are connected to form a neural network, with each of the connections labelled with probabilities or weights. The neural network is generated based on a set of training data (e.g. pictures of road signs paired with labels describing them). It is impossible to predict in advance what these weightings will be (training sets often contain many thousands of pairs of inputs and each extra pair potentially changes the weightings in the neural network), and so it is also impossible to predict exactly what results the system will produce when presented with test data. Here, again, we typically need to consider whether the actual set of results is within acceptable limits (e.g. it correctly identifies the road sign in 99% of cases).

Another problem with testing machine learning systems is that the generated neural network can be very difficult to interpret by a human, so there is often little benefit in manually reviewing the generated neural network for validity in the same way you can review code.

Challenges with Testing Machine Learning Systems – Test Data

When we are testing a machine learning system, one thing we cannot do is use the same data for testing that we used to train the system. Fairly obviously, this would not be a reasonable test – we are using an artificially intelligent system not to check whether it can recognize a previous situation it has seen before (we would not need artificial intelligence for that), but to use the situations it has seen before to react sensibly when it encounters new situations. Therefore, our test data set must be completely independent from the training data set (otherwise we risk what is known as overfitting of data). However, when we find test cases that cause the machine learning system to fail, we should feed these back into the system as training data, so that in the future, the system can cope with similar situations. Test generation is problematic, as it can be very expensive to manually generate this test data – and the cost of test generation is likely to be similar to the cost of training data generation.

If we are testing a machine learning system for an autonomous car, we would need to collect a lot of test data from the real world, and it would need to be correctly labelled. For the testing of the safe scenarios, this could hopefully be automated by use of a scenario generator based on the defined safe scenarios. It should be possible to record real world data and then combine this with the safe scenarios, and use traditional test design techniques to ensure a reasonable level of coverage of the scenarios. For testing that the sensing function correctly interprets the sensor data, the generation of test data sets is, at least initially, typically a more human-intensive activity, although not particularly highly skilled.

Challenges with Testing Machine Learning Systems – Black Swans

Until it was discovered, there was no concept of a black swan outside Australia, because in other countries all swans were white. So, intelligent non-Australians, when presented with a black swan had no idea what it was (and would probably have argued that black swans did not exist). In the same way, an artificially intelligent system will have problems with some situations and images that

it has never encountered before. Thus, it is important that as many scenarios as possible are presented to the machine learning system as training data. From a testing perspective, we need to test that enough unusual scenarios have been presented to the system so that we can have confidence that it can handle similar unusual scenarios. We know that in traditional testing it is often the boundary cases (often called edge cases in AI) that cause problems to the system, and so the testing must cover as many of these edge cases as possible. It is likely that scenarios recorded by vehicles on public roads will be a good source of such scenarios, and non-autonomous cars are already gathering such data for testing.

Challenges with Testing Machine Learning Systems – Research Ideas

There are many researchers considering the problem of building and testing machine learning systems, and there are several testing areas being considered. For instance, it has been found that measuring test coverage of the neurons in a neural network in an approach analogous to white box test coverage of traditional systems may be useful. This approach can be used to ensure sufficient tests are generated to cover all the neurons, and it has also been used to compare neuron coverage when one machine learning system is used to test the output of another one.

Due to their nature, machine learning systems can sometimes generate outputs that are very different from a previous output even though the input has changed very little. When testing, we can use this apparent discontinuity to monitor when a small change to a scenario results in a big change to the system's response. An adversarial approach to the testing of the sensing function uses this idea by taking human generated test inputs and using tools to slightly change the images, perhaps by only a few pixels that would not be noticeable by a human eye, and doing this many times until one of these small changes fools the AI sensing function.

Usability Testing

As we described earlier, there are known problems with drivers keeping their attention on the driving when some SAE Levels 2 and 3 autonomous systems are used. This is because the system takes so much control that after some time the driver becomes bored, begins to trust the system (probably too much), and so is not in a position to take over the driving if the system needs to hand back control. One response to this problem is for the system to monitor the attentiveness of the driver, but such monitoring can often simply provoke drivers into finding ever more devious ways of fooling the system (as of January 2018, an orange jammed in the steering wheel has been shown to fool the Tesla Autopilot [8]).

There could be other potential problems with the human-system interface, as it is currently unclear on the minimum amount of information the drivers (or passengers) of an autonomous car need to be given when everything does not go perfectly (e.g. the system has had to react to an unexpected situation). There is also the question of how much information an autonomous car needs to give to pedestrians and other road users about what it is intending to do.

These areas, and others, are going to require specialist usability testing to be applied as part of the overall test strategy for autonomous cars.

Conclusions

This article (delivered in three parts) has introduced some of the basic concepts that support the development of autonomous cars. A basic framework of autonomous car functions has been used (see Figure 1) to reason about the challenges that need to be overcome in the areas of sensing and decision-making.

In the first part, we considered the rationale for autonomous cars, and the number of lives that can be saved by replacing human drivers with computerised systems. We introduced the SAE Levels of autonomy [1], which are used to define the amount of autonomy provided by an autonomous car or function and investigated the timescales within which we can expect to see commercial autonomous cars on our roads. We also introduced some of the ethical issues surrounding autonomous cars, such as whether they will protect their occupants ahead of pedestrians (the rule in Germany), and how they will obey the driving regulations. We also considered the role of regulators in controlling the use of autonomous cars and ensuring that manufacturers share safety data in the same way that airlines share their accident reports. With autonomous cars, we can expect the requirement for them to report any safety-related incident automatically to the regulator.

In the second part, we looked at the specialized technology that is employed by autonomous cars. To be able to drive without driver input, these cars need to know exactly where they are, and what is in the surrounding area. This means they require excellent sensing technology, and new forms of sensors are becoming available very quickly. Alongside these sensors, these vehicles are also making use of extremely detailed mapping technology, so that they do not only have to rely on cameras and other sensors to determine the direction they go next. The decision-making part of an autonomous car is controlled by artificial intelligence (AI), and so a brief introduction into AI and machine learning was provided, as this will be at the core of nearly all autonomous vehicles.

In this final part, we have considered how the requirements for autonomous vehicle systems will be specified, and suggested a three-way approach, which defines the autonomous function, the environmental constraints and the safety rules which the car must follow. These requirements will also play a big part in the testing of these systems, especially if they are to be properly regulated and insured. Finally, we considered the new challenges of testing such complex, safety-critical systems, and introduced the need for highly-automated and complex virtual test environments. We also introduced the challenges of testing non-deterministic AI systems, an area that is very new – and very different from traditional software testing.

References

[1] SAE document J3016, “Taxonomy and Definitions for Terms Related to On-Road Automated Motor Vehicles”, revised Sept 2016.

[2] Federal Automated Vehicles Policy, NHTSA, September 2016.

[3] Deliverable D1.0, AdaptiVe (Automated Driving Applications Technologies for Intelligent Vehicles), Final project results, June 2017 (retrieved from www.adaptive-ip.eu, Jan 2018).

[4] Shai Shalev-Shwartz, Shaked Shammah, Amnon Shashua, On a Formal Model of Safe and Scalable Self-driving Cars, Mobileye, 2017 (retrieved from Cornell University Library - arxiv.org/abs/1708.06374, January 2018).

[5] Road vehicles – Functional safety, ISO 26262 Parts 1-10, ISO, 2011.

[6] Software and systems engineering — Software testing, ISO/IEC/IEEE 29119 Parts 1-5, ISO, 2013-15.

[7] Software and systems engineering — Work product reviews, ISO/IEC 20246, ISO, 2017

[8] Tracey Lindeman, Using An Orange to Fool Tesla’s Autopilot Is Probably a Really Bad Idea, Motherboard, 2018 (retrieved from motherboard.vice.com/en_us/article/a3na9p/tesla-autosteer-orange-hack, January 2018).

