WHITE PAPER

Accelerate the Development of Advanced Driver-Assistance Systems

Autonomous Drive Emulation (ADE)

Forecasts for the adoption of connected autonomous vehicles (CAVs) remain optimistic. However, those predictions are dependent on unshakable confidence in the minds of consumers, regulators and the insurance industry. Building confidence in advanced driver-assistance systems (ADAS) means hundreds of millions of miles of road testing—actual or simulated—to fully explore corner cases and thoroughly validate new designs.

More Testing, Done Sooner, Enhances ADAS Performance

The following definition can be found on Wikipedia, and it highlights the fact that ADAS requires sensors to be able to perceive the environment of a car and the current driving situation to make the right decisions.

> "*Advanced driver-assistance systems (ADAS) are electronic systems that assist drivers in driving and parking functions. Through a safe human-machine interface, ADAS increases car and road safety. ADAS systems use automated technology, such as sensors and cameras, to detect nearby obstacles or driver errors, and respond accordingly*." [1]

Today, software-based simulation systems are being used to develop and test ADAS functionality. However, traditional integration- and system-level tests cannot ensure proper ADAS operation under real-world conditions. Further, these tests are usually performed late in the development process, making design changes costly and time-consuming, and potentially delaying start of production (SOP).

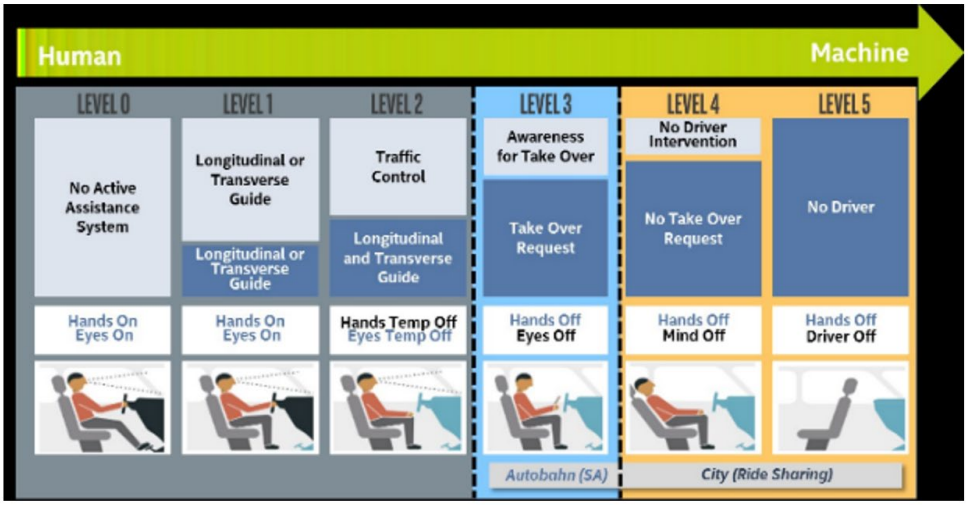
Staying on track with a target SOP date starts with higher-level scenario testing performed earlier in the development process, ideally with minimal use of scarce and costly prototype vehicles. Detailed emulation of real-world conditions enables thorough debugging and troubleshooting of complete subsystems long before vehicles take to the road. This includes accurate simulations that enable thorough testing of camera systems (e.g., realistic visual scenarios) and radar systems (e.g., reflections from moving a stationary objects) in the lab. A technique called ray tracing is essential to the creation of such simulations.

KEYSIGHT TECHNOLOGIES

[1] https://en.wikipedia.org/wiki/Advanced_driver-assistance_systems

Designing for Higher Levels of Autonomy

The goal of ADAS is to not only assist the driver but to eventually enable fully automated or autonomous driving. The Society of Automotive Engineers (SAE) has defined five levels of driving automation in SAE J3016 (considered the global leader in technical learning for the mobility industry). Figure 1 provides a brief overview of the five levels of driving automation.



Sensor	L0	L1	L2	L3	L4 / L5
RADAR	N/A	0-3	0-5	3-6	6-20
Cameras	N/A	0-1	3-5	3-6	3-6
Ultrasonic	N/A	4-8	8-12	RADAR	RADAR
LIDAR	N/A	0	1	2-3	3+
Secure V2X	No	No	Some	Yes	Yes

Figure 1. The need for human interaction and attention decreases at higher levels of driving automation.

An adaptive cruise control system is considered a level-1 operation, and the Tesla autopilot system is considered a level 2 operation. Mercedes just announced that they will equip their new S class with a version of AUTOPILOT for a level 3 operation under certain conditions.

The table beneath the figure indicates the number of sensors required to implement each level of driving automation. Levels 4 and 5 include estimates because development is still in progress.

The American Automobile Association (AAA) tested the performance of available ADAS for five different cars from different brands.[2] The test results were rather disillusioning if one considers the main takeaways:

- For controlled closed-course evaluations, each test vehicle's ADAS system generally performed according to the owner's manual.
- During a roughly 4,000-mile test drive on public roads that included mainly freeways, 521 events were noted, translating to approximately one event every 8 miles. Most notably, 73 percent of the events were related to the lane-keeping functionality of the system.

[2] See the article "*AAA Finds Active Driving Assistance Systems Do Less to Assist Drivers and More to Interfere*."

Understanding the Challenges in ADAS Validation

In summary, AAA concluded ADAS *interfered* more than it *assisted*. This rather harsh verdict glosses over test results that show ADAS worked as expected in the controlled environment. Thus, one could easily assume these scenarios had been previously tested during the development cycle. However, the results of AAA's test drive also indicate the difficulty in testing ADAS for all the different driving scenarios one might encounter on public roads.

This points to a couple of key challenges automotive test engineers face when validating their ADAS designs:

- It is difficult to achieve sufficiently high test-coverage that fully addresses the seemingly uncountable number of potential real-world scenarios.
- In the future, the deployment of fully autonomous operation (i.e., level 5) will require a significant increase in the number of on-board sensors. Consequently, the verification of a fully autonomous systems will require increasingly complicated test setups.

To address these challenges, it's worth a closer look at the automotive development cycle with an emphasis on verification. The diagram in Figure 2 is from the ISO 26262 framework. Called the V-model, it depicts a development cycle focused on ensuring functional safety:

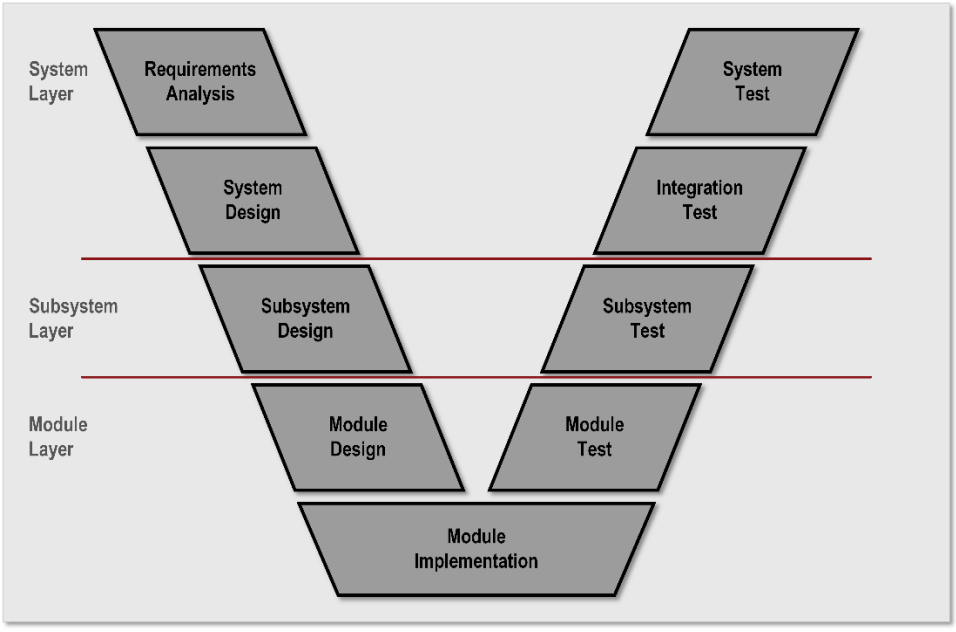


Figure 1. The V-model provides guidance for the design, implementation, integration and testing of ADAS and AV designs.

At the upper-left, the model starts with the overall system definition, and it then moves down through system- and component-level design of both hardware and software. At the bottom, prototypes are constructed and then the final implementation is created. The right-hand leg of the V covers verification, moving upward through the testing of individual components, the resulting subsystems, and then the complete system.

Managing the Cumulative Costs of Testing

When test engineers progress from module tests towards system validation, they may start with pure simulations, especially when the testing of software is involved. However, when progressing up the V-model to submodules and eventually the complete system, the software must interact with the real hardware. This increases the complexity of the required test systems, and this has implications for the associated cost of testing.

One way to counteract these potential costs is to find design flaws earlier in the verification cycle, as exemplified by the V-model. This helps minimize the cost of verification tests, and it also helps prevent failures that can be very costly to remedy if found later in the verification process.

Even so, validating the complete system requires road testing of a fully integrated vehicle. This approach has two noteworthy shortcomings: it is expensive (as indicated above); and results obtained during random, real-world environmental conditions are not easily reproducible. Reproducibility becomes important if the system-under-test is producing intermittently erroneous behavior. After making changes to the system, the ability to verify the effectiveness of any and all fixes is absolutely essential.

Including Software and Hardware in the Loop

Let's consider an adaptive cruise control system as an example. Based on inputs from a radar sensor, the system should control the brake and accelerator systems to either meet the desired cruising speed or maintain a safe distance from a vehicle directly ahead. Verifying the control algorithm starts with a simulation system that replicates the road environment (e.g., driving conditions and surrounding traffic), the radar sensor, and the physical behavior of the car (e.g., reactions of braking and acceleration systems). This approach can verify the behavior of the algorithm in different scenarios. This type of test setup is referred to as "software in the loop" or SiL.

Over time, hardware elements can be added to the system. For instance, a test setup could include a microprocessor system such as an electronic control unit (ECU) that will later be integrated with the vehicle and, for example, a radar sensor. The operating environment and the respective reactions of braking or accelerating for this setup would be implemented through simulation. This type of test setup is referred to as "hardware in the loop" or HIL.

To address such applications, Keysight has introduced an HIL-based system called the autonomous drive emulation (ADE) platform. It supports the testing of ADAS at the component and system levels. The block diagram in Figure 3 outlines the high-level architecture of the ADE platform.

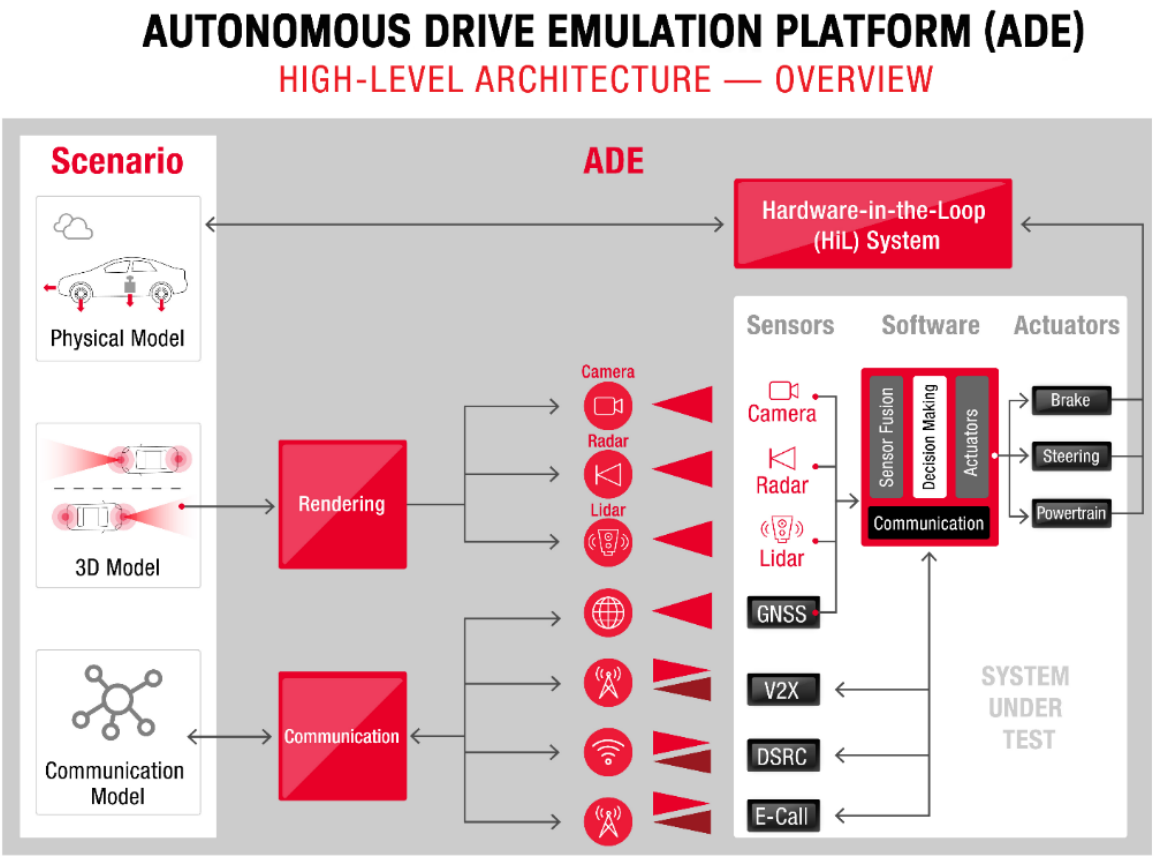


Figure 2. Through integration time-synchronized testing of key subsystems, ADE saves time and money—and provides better test coverage—when compared to single-unit testing.

Creating the Simulated Environment

The test scenario is set up using a simulated environment, which implements the real-world situation all around the vehicle: left and right, top and bottom, front and back. This simulation is used to extract the required information to stimulate sensors such as cameras or radar units, and to link to available communication channels such as vehicle-to-everything (V2X).

To extract the required information for sensors such as radar or cameras during the test, ray-tracing technology can be used. Figure 4 illustrates the fundamental principle.

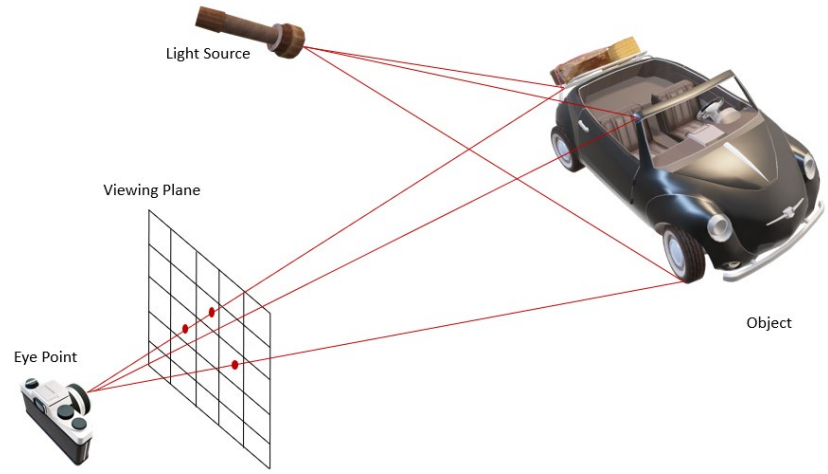


Figure 4. Within a computer-based simulation, ray tracing produces highly realistic views of modeled objects.

The goal is to determine how an object has to be displayed on a viewing plane such that it recreates what a human operator would observe in the real world. For an object to be seen, it needs to be illuminated. In Figure 4, the light source is shown as a flashlight.

The object is represented by a wireframe model in which the surface of the object is split into small triangular tiles, each having a flat surface. For every pixel in the viewing plane, the algorithm can draw a straight line (i.e., trace a ray) from a specified viewpoint (e.g., the driver's seat) through the pixel to the object. This ray will strike one tile of the wireframe of the object, and another ray can be traced from this tile to the light source.

Additional pieces of information, called "material properties," are attached to each tile. Examples include the object's color and reflectivity. Based on this information, along with the angles at which the incident and reflected rays hit the tile, the algorithm can calculate which color and brightness should be used for the individual pixel (remember: each tile has a flat surface so the incident angles can be easily determined). Once this has been done for every pixel of the viewing plane, a 3D image can be displayed on a screen.

The same algorithm can also be used for radar. The light source is the radar transmitter, the relevant material properties would be included, and spatial velocity would be taken into account to calculate the Doppler effects. Even so, the same principles apply.

Handling Inherent Processing Delays

Modern graphics processing units (GPUs) provide hardware acceleration for ray tracing algorithms. As a result, they are great tools for achieving the computing power required for this approach. However, no matter how fast the GPUs may be, there will always be a processing delay, which can become an issue.

For example, assume that the ray-tracing algorithm has a processing delay of 100 ms to deliver its data to a radar target simulator. This is the time required for the algorithm to take a snapshot of the scene, perform its calculation, and present data to the radar.

Next, let's consider a typical urban traffic situation: two cars are approaching each other and both are driving at a speed of 50 km/h (Figure 5). That's a relative velocity of about 28 m/s, and within the 100-ms interval they have closed the distance by 2.8 m.

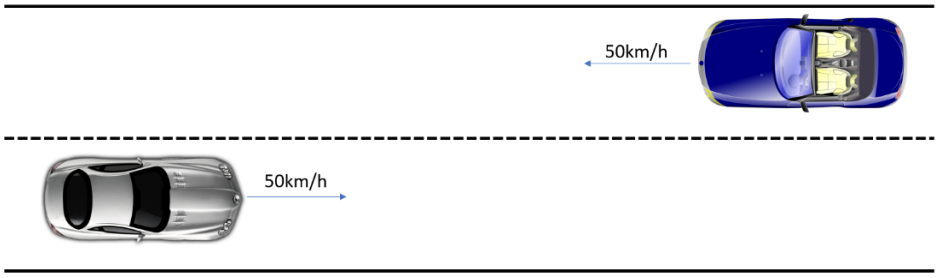


Figure 5. Two vehicles approaching from opposite directions is a common urban driving scenario that can be simulated.

This is an easy scenario for an open-loop system to handle. It can be compared to watching a movie: it doesn't matter if the data is perceived a little later. It is important that, if other sensors are involved, then all are being stimulated consistently. In other words, the processing delay has to be the same for all the sensors to achieve overall measurement synchronism.

Closing the Loop for Highly Realistic Testing

When one also wants to incorporate the ADAS reaction in the test, it requires the closed-loop approach. Here, the reaction of the vehicle is included in the simulation. For instance, if the ADAS decides to hit the brakes in an emergency, this slows down the vehicle and consequently affects the data that is sent to the radar.

Referring back to Figure 3, the loop is closed using an HIL system that also implements the emulation of other components within the car that do not relate directly to the ADAS. To overcome the latency issue, a "nowcasting" algorithm must be implemented. As context, forecasting is a familiar concept from meteorology: weather forecasts use current data to predict what will happen in the future. Nowcasting is the use of "outdated" information – in this case data that is 100 ms old – to predict the current situation.[3]

With this approach, the combination of an ECU, its software, and the real sensors, can be tested. As highlighted above, this requires the simulation of several sensors and communication channels, not only in a synchronized way but also with an algorithm that compensates for the processing delay (keyword: nowcasting).

With these requirements being met, the ADE platform allows testing of an ADAS starting from the component level and progressing to the sub-system level, long before a vehicle is tested on the road. The goal is to decrease the overall cost of testing while improving test coverage at an early stage in verification. Ultimately, this helps improve the performance of ADAS when deployed on public roadways.

[3] For more, please see an article from the University of Graz (Austria) available on ResearchGate.net.

Realizing Your Vision of Mobility

To achieve the goal of fully autonomous driving, OEMs and their suppliers need a way to test in a closed-loop environment using real-world signals in a lab. Keysight's visionary ADE platform brings the road to the lab, thereby enabling the testing of real sensors with real data in a closed-loop system. Ultimately, this translates into confidence, cost savings and a competitive edge in the race to achieve fully autonomous transportation over the roadways.

As you continue to create what comes next, Keysight is ready with test solutions that can accompany you from concept to reality. Our goal is to help you excel—and accelerate—in those areas that are redefining future mobility: sensor systems, wireless links, in-car networks, batteries and cells, and beyond. It's all about getting there first and realizing *your* company's vision of mobility.

To learn more, please visit www.keysight.com/find/ade

Learn more at: www.keysight.com

For more information on Keysight Technologies' products, applications or services, please contact your local Keysight office. The complete list is available at:
www.keysight.com/find/contactus

KEYSIGHT TECHNOLOGIES