Autonomous Vehicles: State of the Art, Future Trends, and Challenges

Piergiuseppe Mallozzi, Patrizio Pelliccione, Alessia Knauss, Christian Berger, and Nassar Mohammadiha

Abstract Autonomous vehicles are considered to be the *next big thing*. Several companies are racing to put self-driving vehicles on the road by 2020. Regulations and standards are not ready for such a change. New technologies, such as the intensive use of machine learning, are bringing new solutions but also opening new challenges. This chapter reports the state of the art, future trends, and challenges of autonomous vehicles, with a special focus on software. One of the major challenges we further elaborate on is using machine learning techniques in order to deal with uncertainties that characterize the environments in which autonomous vehicles will need to operate while guaranteeing safety properties.

1 Introduction

During recent years, the automotive sector has been experiencing major transformations. Today's vehicles have a lot of software with millions of lines of code and more processing power than early NASA's spacecraft. Over the past 20 years, the size

P. Mallozzi (✉) · A. Knauss · C. Berger
Chalmers University of Technology, Gothenburg, Sweden

University of Gothenburg, Gothenburg, Sweden
e-mail: mallozzi@chalmers.se; alessia.knauss@chalmers.se; christian.berger@gu.se

P. Pelliccione
Chalmers University of Technology, Gothenburg, Sweden

University of Gothenburg, Gothenburg, Sweden

Università degli Studi dell'Aquila, L'Aquila, Italy
e-mail: patrizio.pelliccione@cse.gu.se

N. Mohammadiha
Chalmers University of Technology, Gothenburg, Sweden

University of Gothenburg, Gothenburg, Sweden

Zenuity, Gothenburg, Sweden
e-mail: nasser.mohammadiha@zenuity.com

Y. Dajsuren, M. van den Brand (eds.), *Automotive Systems and Software Engineering*, https://doi.org/10.1007/978-3-030-12157-0_16

of software has grown by a factor of 10 every 5–7 years.[1] Gigabytes of software run inside the ECUs, which are small computers embedded in the vehicle. In the same period of time, the amount of Electronic Control Units (ECUs) has grown from around 20 to more than 100 pushing companies such as Volvo to renovate the electrical architecture of their vehicles in order to cope with such complexity [45]. With this amount of software embedded in the vehicles, car manufacturers have to become more of software companies in order to deal with today's challenges.[2]

Connected vehicles will benefit from intelligent transport systems (ITSs), smart cities, and the Internet of Things (IoT). They will combine data from inside the vehicle with external data coming from the environment (other vehicles, the road, signs, and the cloud). In such a scenario, different applications will be possible: smart traffic control, better platooning coordination, and enhanced safety in general.

Intelligent vehicles will reach full automation, freeing the driver from performing any task. This is a path that will only be reached gradually. There are six levels of vehicle automation (Sect. 1.1 details these levels). To date, almost all of the vehicles in circulation settle on the levels comprised between levels 0 and 2 (level 2 is defined as "partial automation"), in which the systems are limited to assist the driver without replacing him.

Both companies and academia are putting an enormous effort into developing technical solutions in order to increase the levels of autonomy in vehicles. The business landscape is in a profound transformation as we describe in Sect. 1.2. Companies such as Google and Apple, with no tradition at all in car manufacturing, are racing to put their autonomous vehicles on the road in the near future. Software volume in vehicles has been increasing for years; it is expected to increase by 50% by 2020 [22]. 80–90% of the innovation within the automotive industry is based on electronics, and a big part of electronics is software [30, 45].

1.1 Levels of Vehicle Automation

SAE International[3] has defined six levels of vehicle automation as shown in Fig. 1. In today's vehicles, the driver plays a fundamental role. Besides controlling the vehicle, she/he is also expected to recover the vehicle from failures. With full autonomy (SAE level 5), the driver is no longer in the loop. The vehicle, besides controlling the basic driving maneuvers, should handle all possible situations during driving. As the driver is not monitoring the vehicle and is not a fallback option anymore as in the lower levels of automation, it is expected that nothing goes wrong.

[1]"Surviving in an increasingly computerized and software driven automotive industry"—Martin Hiller, keynote at ICSA 2017 conference: http://icsa-conferences.org/2017/attending/keynotes/.

[2]This Car Runs on Code http://spectrum.ieee.org/transportation/systems/this-car-runs-on-code.

[3]SAE International is a professional association and standards organization for engineering professionals in transportation industries. http://www.sae.org.

	SAE level	Name	Description
The human driver monitors the environment	0	No Automation	Human driver is responsible for steering, throttle and breaking.
	1	Driver Assistance	The vehicle can perform some control function but not everywhere.
	2	Partial Automation	The vehicle can handle steering, throttle and breaking but the driver is expected to monitor the system and take over in case of faults.
The driving system monitors the environment	3	Conditional Automation	The vehicle monitors the surroundings and notifies the driver if manual control is needed.
	4	High Automation	The vehicle is fully autonomous but only in defined use cases
	5	Full Automation	The driver has only to set the destination. The vehicle will handles any surrounding and make any kind of decision on the way.

Fig. 1 SAE levels of autonomy as defined in standard J3016. Source: SAE

A fatal error in the vehicle is a potential threat to the safety of the vehicle and the passengers inside the vehicle. Hence, fully autonomous vehicles have to support fault tolerance. For this reason, there is a need to investigate requirements to allow the system to continue operating safely during faults. Vehicles have to transition from being *fail safe* to becoming *fail operational*.

1.2 Autonomous Vehicles Ecosystem

The ecosystem landscape is changing when looking at the lower levels of automation to the higher levels of automation [31]. The players in the ecosystem of levels 1–2, where the driver is supported through different vehicle functions (e.g., active safety features), contain the companies that develop the vehicle (i.e., OEMs), suppliers, the government, and organizations that are responsible to define test methods/test catalogs, as well as legislative organizations. Furthermore, certification bodies are taking the defined test standards as an input and are running different tests based on the standards. Test facilities provide the ground to run such tests in a safe environment. Another big player in the ecosystem are customers who buy the final product. Other players are, for example, researchers, insurance companies, and journalists [31].

For the higher levels of automation (i.e., levels 3–5), additional players enter the ecosystem: companies that focus on communication aspects will play a big role in this ecosystem, as communication will be an important basis for highly automated driving [31]. Communication allows the vehicles to ensure that they have redundant information sources (i.e., several sources of information) to ensure safety, reliability, and quality of autonomous vehicles. As can be seen in this example, data plays an essential role in autonomous vehicles. Hence, companies that focus on data will be additional players in the ecosystem [31]. When it comes to OEMs for highly automated driving, we notice that companies need to shift to being software companies instead of developing software just as a side product, as already mentioned above.

The production of a vehicle requires the integration of thousands of parts. Until now, suppliers have been extremely important in providing different components to the vehicle manufacturers that have become more like assemblers. OEMs are interested in investing in autonomous vehicles because it is such a new and high-impact area. New suppliers are joining the autonomous vehicles business, providing hardware and software solutions, such as Nvidia, Mobileye, and Intel. OEMs are establishing partnerships with suppliers focusing on autonomous vehicles. For example, Zenuity is a joint venture on self-driving cars between Volvo Cars and the supplier Autoliv. BMW announced the alliance with Intel and Mobileye with the plan to bring self-driving cars on the road by 2021. Mercedes joins the forces with Bosch, one of the biggest automotive supplier, to develop level 4 and level 5 vehicles.

Google and Tesla are among the pioneers to push the autonomous driving-related software into the market, but at least 33 companies are working on autonomous vehicles.[4] Most of them had promised to put the autonomous car on the road by 2021, assuming that the necessary regulations will be put in place by then.

In addition to autonomous cars, there is a lot of interest in autonomous trucks. The companies working on developing autonomous trucks include Volvo AB, Daimler, Scania, Iveco, Komatsu, and Caterpillar. Autonomous trucks might have to deal with technology and social challenges even more than self-driving cars. From the technology perspective, autonomous trucks have to coordinate their movements with other trucks so that they can drive closely in a *platoon formation*.

Truck platooning not only reduces road congestion but also dramatically increases the vehicle efficiency by reducing fuel consumption. It is believed that it will also save on labor costs, due to relaxed governmental regulation on obliged driver resting time. This concept has been studied for years, and in April 2016, a convoy of connected trucks completes for the first time cross-border trip in Europe.[5] The trucks platoon involves a leader truck that decides the route and speed and several follower trucks driving in automated mode.

2 Autonomous Driving: State of the Art

In recent years, automation has been playing a big role in reducing the need for human intervention for a variety of tasks. In the automotive domain, new technologies have been developed to increase the autonomy of the vehicles resulting

[4]https://www.cbinsights.com/blog/autonomous-driverless-vehicles-corporations-list/.

[5]https://www.theguardian.com/technology/2016/apr/07/convoy-self-driving-trucks-completes-first-european-cross-border-trip.

in advanced driver-assistance systems (ADAS). Applications such as the adaptive cruise control (ACC), advanced emergency braking system (AEBS), and lane-keeping assist (LKA) are already available in modern cars controlling part of the vehicles and assisting humans. Vehicles have acquired functionalists to help or replace humans in the most common driving tasks, such as identifying road lanes, detecting pedestrians, and avoiding collisions, as further explained in Sect. 2.1.

Another aspect of autonomous vehicles is the transition from the hardware being the main focus of vehicles to software becoming the main enabler in fully autonomous vehicles. Software updates will become as easy and frequent as those on our smartphones. The software will be upgraded over the air, and it will bring more functionality to the vehicle or enhance important aspects such as safety. Fleets of vehicles will collect data in the environment that is then analyzed and redeployed as a new software update. In order to address such increasing complexity in the software, new challenges need to be addressed in the electronic architecture of the vehicles as further explained in Sect. 2.2.

2.1 Vehicle Functionality

One of the main catalysts of building such autonomous vehicles is to enhance the overall safety. In recent years the attention of vehicle manufacturers has been extended from the already consolidated passive safety systems (airbags, seat belts, impact resistance, etc.) to active safety advanced systems, mainly designed to prevent dangerous situations and accidents rather than protect humans in case of a crash. More recently, with developing of connected cars, safety will take another major step, preventing crashes by taking into account information coming from outside the vehicle [46], for example, from a bicycle coming from a side of a building.

In traditional vehicles, the car uses data locally stored in the vehicle and the communication based on signals between the different ECUs (Electronic Control Units) of the car. In autonomous vehicles, the control can rely on aggregated data coming from multiple sensors but also from the infrastructure outside the vehicle such as the cloud. The communication rather than being signal-based within the vehicle is service/IP based with the outside world. Sharing and controlling sensitive information could be of crucial importance in dangerous situations. For example, recently Volvo Cars have developed a mechanism to inform road users and road maintenance of slippery roads [46]. When a car detects slippery conditions on a road, it sends the information to the Volvo Cloud. This information can then be used to predict the road condition for later times [43]. When another car is approaching the slippery part of the road, Volvo Cloud notifies the approaching vehicle that will automatically reduce its speed.

These technology advancements keep bringing new functionality to vehicles that become more and more autonomous. The path toward fully autonomous vehicles is marked by incremental advances in different categories of assistance. Reaching level

5 of autonomy is more an evolutionary path rather than a revolution from one day to another. Assisting the driver is the first step toward automation. ADAS (advanced driver-assistance systems) aims to assist the driver or to automate some functionality in order to enhance safety and driving comfort, from assisting the driver during parking maneuvers to enhancing the vision during the night. Some of the systems already available in modern vehicles are:

- *ACC (adaptive cruise control)* allows maintaining the safety distance between the car and the vehicle in front.
- *FCW (forward collision warning)* uses camera and radar sensors to identify obstacles on the path of the vehicle.
- *LKAS (lane-keeping assist system)* in addition to warning the driver when the vehicles are going out of its lane also automatically steers the vehicle back to the center.
- *ISA (intelligent speed assistance)* helps the driver to maintain the speed, thanks to a software that recognizes the road signs indicating the limits or more simply by setting an own choice.

ADAS functions are mostly limited to assist the driver without replacing it, but the direction is to create more functions and integrate them in order to reach higher levels of autonomy. Several pilot assistance systems are already available in the market bringing the vehicles forward in the automation level 3 (partial automation, the driver has to be ready to take over at all times). Tesla keeps updating his software for the autopilot adding new functionalities. Besides keeping the lane, it can also overtake other vehicles on the highway and park the car autonomously. Volvo Cars has developed the Pilot Assist system that is able to drive automatically when certain conditions are met. However, the system demands that the driver does always hold her/his hands on the steering wheel.

In order to reach higher levels of autonomy, we need more sophisticated applications that make the most out of the sensors and communication channels available in connected vehicles. Significant developments in the information and communication technologies (ICT) have enabled the advancement of new safety applications. Bila et al. [11] categorize different safety-critical applications that have been developed for intelligent vehicles. *Vehicle detection* application aims to detect and track surrounding vehicles with respect to the host vehicle. Such applications are useful for collision avoidance but also for decision-making based on the movements of other vehicles. *Road detection* and *lane detection* are fundamental tasks to adequately steer the vehicle in "drivable" regions. *Pedestrian detection* is of critical importance for reducing the number of traffic accidents and road fatalities. *Drowsiness detection* is a feature that is already offered by some vehicle companies with the aim of reducing accidents caused by drowsy drivers. Once drowsiness is detected, the vehicle should take over and bring itself into a safe state by, for example, pulling over to the side of the road. More generally, *collision avoidance* is the main system that can include all of the above applications. It can either warn the driver or, in more autonomous systems, take control of the vehicle by steering and breaking.

2.2 Vehicle Architectures

Self-driving vehicular systems in robotic cars, automated trucks, and the like share similarities in their fundamental software architectures. The predominant design pattern in robotic vehicles participating in the 2004 and 2005 DARPA Grand Challenge [59] as well as the 2007 DARPA Urban Challenge [3, 4, 41, 48] was following the well-known *Observe-Orient-Decide-Act* (OODA) pattern, typically in a *pipes-and-filters*-based realization. In practice, these robotic vehicles have software modules dealing with:

- Gathering data from the sensors like cameras, radars, laser scanners, ultrasound sensors, GPS positioning sensors, and the like (*Observe*)
- Data fusion and analysis using sensor fusion, for example (*Orient*)
- Interpretation and decision-making (*Decide*)
- Acting by selecting appropriate driving trajectories to derive the set points for the control algorithm (*Act*)

These software modules get data streams as input, apply filters for data processing, and produce output data that in turn is serving as input data for subsequent software modules. This pattern is still dominant for other robotic applications [6, 7] and a well-accepted engineering approach [8].

Along with the pipes-and-filters pattern and a system architecture dating back to the DARPA Grand and Urban Challenges, software systems distributed across several physically separated computing nodes interconnected over networks were dominantly in use. Therefore, the typical software design that can be found in such systems is based on the *publish/subscribe* communication principle. While this pattern quickly enables a distributed software system whose distribution can even be dynamically adjusted at runtime following the recently emerging *microservices* design pattern [9, 25, 39]. However, the fundamental network design enabling such distributed systems is putting the fundamental software architecture design at risk considering the increasing demand for handling more and more data streams within self-driving vehicles originating from various sensors to redundantly cover 360° of a vehicle's surroundings, especially today, when CPUs are becoming more and more powerful with recent chips providing 32 cores that can be operated next to powerful specialized processing units like GPUs, FPGAs, or TPUs. Under such circumstances, a publish/subscribe network-oriented communication is unnecessarily introducing communication overhead for serializing and deserializing the messages to be exchanged on local systems. Consequently, a more centralized information exchange pattern is needed. The *blackboard* pattern might be the optimal pattern since it is able to cope with increasing data loads. This is an important aspect when several software agents running in parallel can independently and instantaneously access and process large amounts of data while meeting real-time requirements.

Behere and Törngren [5] have defined a functional reference architecture for autonomous driving. They divide the architectural components into three main categories: (1) perception, (2) decision and control, and (3) vehicle platform manipulation. All the components are also distributed in those belonging to the *driving intelligence* and to the *vehicle platform*. They recommend a clear separation between these two: the first should take high-level decision, while the latter is composed of low-level control loops and actuators that apply the decisions.

Two Swedish projects, called NGEA and NGEA2, are investigating the challenges and providing recommendations for the electrical and software architecture of the next generation of vehicles. The projects are coordinated by Volvo Cars and involve some universities, including Chalmers, and research centers in Sweden and many suppliers of the OEM, including Autoliv, Arccore, Combitech, Cybercom, Knowit, Prevas, ÅF-Technology, Semcom, and Qamcom. The projects released an initial version of an architecture framework for Volvo Cars [45]. According to the ISO/IEC/IEEE 42010:2011 standard [27], an architecture framework prefabricated knowledge structure, identified by architecture viewpoints, used to organize an architecture description into architecture views. The framework is based on the conceptual foundations provided by the standard [27] and currently focuses on three new viewpoints that need to be taken into account for future architectural decisions: continuous integration and deployment, ecosystem and transparency, and the car as a constituent of a system of systems. Another natural viewpoint that will be part of the framework is the viewpoint about autonomous driving.

A self-driving car should be able to deliver safe driving in all planned conditions, that is, under conditions and use cases where the autonomous functionality is available. In this regard, it is of vital importance that a self-driving car correctly reacts to all usual and unusual behavior of other road users and traffic conditions. Moreover, the system should be reliable and fault-tolerant considering the hardware. A well-known fault-tolerant architecture that is widely used in the avionics systems is the redundant architecture [5, 26]. In a simple setup, all the components, including perception, decision and control, and actuators, are duplicated, and, therefore, the system becomes more tolerant to the software and hardware failures. Moreover, one way to handle high-ASIL (Automotive Safety Integrity level) components can be realized using ASIL decomposition via a combination of a monitor/actuator architecture and redundancy [32].

3 Autonomous Driving: Trends and Future Direction

Autonomous vehicles have been considered science fiction only a few years ago. Technological advancements together with increases of computational power and a huge amount of available data are making autonomous driving less of a vision and more of a reality in the near future.

Artificial intelligence will play a big role in recognizing situations and, ultimately, in making optimal decisions as explained in Sect. 3.1. Furthermore, autonomous vehicles must adapt their behavior at runtime (Sect. 3.2) and continuously evolve (Sect. 3.3). In the remaining of this section, we discuss the trends and future directions of autonomous driving.

3.1 Artificial Intelligence

An autonomous vehicle can be seen as a rational agent: it perceives information from the environment, computes a decision, and, finally, applies actions to its environment [49]. Artificial intelligence is the ultimate concept that will allow machines to perform any sort of task autonomously and in any situation. Under the umbrella of artificial intelligence, applications such as machine learning (ML) are being increasingly used to implement solutions in different fields, one of which is self-driving vehicles. Techniques such as deep learning have recently become popular thanks to their outstanding results in a variety of different tasks, such as image recognition and decision-making.

Machine learning (ML) techniques can be used with fuzzy information; it can deal with uncertainty using a huge amount of data coming from the vehicle's sensors. They build models based on the data and later use the same models to make predictions. For example, the information captured by the camera and lidar sensors is collected. In the next step, a semantic representation of the raw data is produced with techniques such as sensor fusion. Later, machine learning techniques are used to classify such data into surrounding objects, supporting vehicles with their detection and collision avoidance.

Fully autonomous vehicles require different machine learning techniques to be applied to several components of the vehicle. Recently, deep neural networks (DNNs) have been adopted as one of the main techniques to solve several important tasks related to the perception part of autonomous driving. This includes, for example, object detection and classification, road detection, and semantic segmentation of images. DNNs are powerful methods and can learn complex representations from raw data and hence eliminate the need for hand-crafted features, which is difficult for other machine learning methods. In order to build an autonomous agent (i.e., an entity that is able to make decisions based on what it observes), there are three main tasks to be taken into consideration: perception, prediction, and planning. Machine learning can be applied in all of them [19].

1. *Perception*: Perception is the main area where deep learning is used at the moment. It consists in extracting semantic knowledge from raw data coming from vehicle's sensors. In autonomous vehicles, it can be found, for example, in object detection and tracking, pedestrian detection, road signs recognition, and location and mapping, supporting functions for intelligent vehicles, such as prediction of friction, destination, and traffic flow [15, 18, 36, 40].
   Identifying objects in the environment can be achieved with advanced deep learning algorithms. Convolutional neural networks (CNNs) [34] are considered one of the best deep learning models to classify images. Since 2012, ImageNet has launched a competition where different groups try to set the new state of the art for classification and detection of millions of images. In 2016 Google has proposed GoogLeNet v4 [57], an algorithm that achieves 3.08% error rate in image classification (ImageNet has reported the human error rate to be 5.1%).
2. *Prediction*: In order to build models that can predict the future states of the environment, recurrent neural networks [47] can be used. Furthermore,

supervised learning can be used to predict the near future based on the current state [53]. Convolutional neural networks can be used to directly predict the steering angle of the vehicle from raw input signals such as the pixel of the front-facing camera [12].

3. *Planning*: Perception and prediction are combined to plan a sequence of actions. Frameworks such as reinforcement learning serve such a purpose. It can be used with both deep neural networks and recurrent neural networks. Reinforcement learning [56] involves an agent that acts on an environment by choosing predefined actions with the goal of maximizing a numerical reward. The environment issues these rewards to the agent for every action it receives. The agent keeps track of the rewards and uses them to improve its decision-making policy. Another paradigm to teach a system how to perform tasks is *learning from demonstration* [1], where policies are learned from a teacher. Imitation learning, or behavioral cloning, directly copies the teacher and can be achieved by supervised learning alone. Apprenticeship learning instead tries to infer the goal of the teacher that once formalized as a reward function can be used by reinforcement learning.

3.2 Self-adaptive Systems

An interesting direction that is emerging is self-adaptive systems, that is, systems that are able to adapt their behavior at runtime without human intervention [16, 17, 50] in response to changes in the environment or in their internal state. An autonomous vehicle can be framed as a self-adaptive system: it gathers knowledge from the environment, contextualizes it considering also its internal state, and adapts itself to achieve its goals. This process is performed continuously at runtime.

All self-adaptive systems implement some sort of feedback loop that drives their adaptations [13]. This basic mechanism for adaptation has been applied for years in control engineering; it consists of four main activities: collect, analyze, decide, and act. A well-known reference model for describing the adaptation processes is the MAPE-K loop (consisting of the parts model, analyze, plan, execute, and the knowledge base) [29]. Furthermore, the system is collecting new data from the environment, learns from it, and continuously improves. An example of such a system is the never-ending language learning [14].

Because of the complexity of such systems, maintaining them is a problem similar to large-scale distributed systems. To manage their complexity, they should maintain themselves through a variety of different scenarios [10]. They should self-optimize, continuously seeking for ways to improve its actions on the environment. Self-protect from malicious attack or an error in one system should not propagate to other systems. Self-heal, after detecting that a fault or error has occurred, should recover autonomously. Ultimately, in order to achieve all these self-properties, autonomous vehicles should reach self-awareness, but it is hard to reach it without a breakthrough in artificial intelligence.

Self-adaptive systems and online learning are promising concepts but are still far from practical application in the automotive industry. Collecting data at runtime, self-adaption, and continuous evolution are a fascinating concept. When dealing with safety-critical systems, the adaptation of the system must be certified as *safe* before it can be applied. A more realistic scenario is that automotive companies collect data from their customer, and based on this data, they adapt their method and software which will be pushed back to the vehicles once it has been tested and proved safe.

However, with the intensive use of machine learning techniques, it is hard to test the software and be sure that it will act always as intended. Methods such as *runtime monitoring* can help to prevent the system from violating its requirements. On the one side, we can let the autonomous vehicle reconfigure itself at runtime following high-level policies that specify *what* to achieve and let the system determine *how*. On the other side, a monitor can help to check that its decisions do not violate any predefined invariant of the system [37].

3.3 Continuous Software Engineering

A vehicle is a mechatronic system composed of hardware, software, and mechanical parts. The development of such complex systems has traditionally followed the V-model. This process follows a V-shape starting from high-level requirements of each component going down to the implementation and, then, up again to the verification of the integrated components. The size of the software in the vehicle has grown tremendously in recent years, and for this reason, the development cycle has moved from a rigid V-model to more flexible agile methodologies where the iterations are much more frequent, thus enabling a fast feedback. Moreover, as anticipated above, OEMs are becoming software companies, and they would like to support continuous evolution of software, even after the vehicles are already on the road, through continuous integration and continuous deployment (CI&D) practices. CI&D practices promise to shorten integration, delivery, and feedback cycles [54]. These techniques have been applied by pure software companies, such as Facebook, and now automotive companies have a strong motivation to embrace organization-wide continuous integration and delivery practices to yield improvements in flexibility and cycle time despite the challenges [30]. Challenges include tooling, which works well in a V-model context but that might cause friction in a continuous engineering environment, especially because it facilitates silos thinking and does not support cross-functional aspects [30]. The same paper recommends also to establish suitable transparency between the actors in the automotive value chain, to avoid bottlenecks, as well as information overload, and to address the need for scalability and cross-functional collaboration.

Moreover, construction of shared plant models in a particular value chain as a continuous delivery target can be valuable in future research. Model-driven engineering can bring great benefits in the design and development of automotive systems [20]. In continuous integration and delivery, models need to be integrated on the system level, thus bringing together knowledge about application software, ECU hardware and basic software, and mechanics. This might allow domain experts to be directly involved in the development of the software and have running prototypes from early stages of development. Furthermore, simulation environments enable to perform tests on the system before the mechanical and software components from the suppliers are available, for example, using executable Simulink models of the actual ECU of the vehicle and testing them in a Model-in-the-Loop (MIL) environment where the other models (so-called plant) are used for simulating the surroundings of the ECU.

Continuous evolution of a vehicle should additionally exploit the knowledge collected by other vehicles. In this sense, assuming that mistakes are inevitable also for autonomous vehicles, not every vehicle has to make the same mistake to learn from it. Once the software has been fixed, all other vehicles will be updated based on this knowledge and no other vehicles will do the same mistake again. As Tesla says: *as more real-world miles accumulate and the software logic accounts for increasingly rare events, the probability of injury will keep decreasing*.[6]

3.4 User Aspects

For the lower levels of automation, the driver is responsible for the driving task and takes all risk. For fully automated vehicles, the vehicle itself is responsible for that task and does not require the driver to interact with the system. For the levels in between, the driver is expected to interact with the vehicle. For level 1 and level 2, the vehicle is supporting the driver. If it fails in any of its tasks, the driver is still responsible for the monitoring of the driving task and its own safety. Hence, the vehicle can be only seen as a support and not the final point of decisions. For levels 3–5, the vehicle itself executes the driving task as well as monitors the driving environment. For level 3, the driver is still expected to be the fallback option, in case the vehicle fails in any situation. Then the driver is expected to take over the driving task.

This is one of the reasons why user-related aspects are gaining increased importance in autonomous driving. In 2013 the Asiana Flight 214 crashed at San Francisco International Airport.[7] The CBS News aviation and safety expert Capt. Chesley “Sully” Sullenberger hypothesized that “the pilots may have not fully understood the workings of the automation and that they assumed that the auto-thrust was controlling the speed when in fact it was not.” Even though this example comes from avionics, it clearly shows that the interplay between humans and

[6]https://www.tesla.com/blog/tragic-loss.

[7]http://www.cbsnews.com/news/asiana-crash-hearing-role-of-pilots-automation-to-be-questioned/.

machines is not easy, and it is not enough to just assume that the driver will need to have the hands on the steering wheel and will take control if something will go wrong. Humans have to be aware of the current situation, monitor the vehicle, and be aware of system limitations [38]. Humans are not good at remaining vigilant in the long run while monitoring the system, and this is partly due to under-stimulation. Moreover, unlike airplanes, we cannot assume to have professional drivers in cars, and most probably in the future, as a side effect of automation, we might even have drivers with degraded performance.

While for the lower levels of automation functional aspects are very important, with increased automation, nonfunctional aspects play an important role as well. Recent work has been focusing on aspects like driver taking over the steering wheel after automated driving and driver handing back the steering wheel [60], the different factors that play a role for the time that it takes to take over the driving task [61], and the factors besides the take-over time that plays a role in the take-over quality [62]. Furthermore, six types of transitions between the driver and vehicle were defined: (1) optional driver-initiated driver-in-control, (2) mandatory driver-initiated driver-in-control, (3) optional driver-initiated automation-in-control, (4) mandatory driver-initiated automation-in-control, (5) automation-initiated driver-in-control, and (6) automation-initiated automation-in- control [35].

However, many questions remain open and more research is needed in order to provide an answer. Examples of questions are:

- Which human behavior model to consider while developing systems? We cannot develop systems according to how humans behave under ideal circumstances since humans are not always taking decisions rationally. Often we make decisions based on context, feedback, culture, stress, workload, perception, expectations, training, feelings, etc.
- Will users behave the same over time?
- Can a driver do something wrong? Should the system try to correct human behaviors, or should we assume that humans are always correct?
- Do the drivers have the needed information in order to actively make the right decisions? In various modes of automation, user involvement decreases, and this can put the driver out-of-the-loop.
- How to establish a sympathetic cooperation between the driver and the vehicle, by making sure that it covers all situations and all different (possibly unpredictable) behaviors?

4 Verification of Autonomous Driving: Challenges for Guaranteeing Safety

The main challenges for deploying autonomous vehicles are raised by the implicit nature of the vehicle: its connectivity to the external world and its self-adaptability situations in the environment. This openness allows vehicles to exchange information with other vehicles (vehicle-to-vehicle (V2V) communication) or to the environment (vehicle-to-infrastructure (V2I) communication). Vehicles should be able to adapt to a different scenario in the environment without human intervention. Such scenarios are usually defined at design time, but one of the challenges is to deal with the uncertainty of the environment as explained in Sect. 4.2. Vehicles become then part of an open system, whose boundaries might be subject to dynamic changes.

On one side, the vehicle autonomy depends on the information it receives from its sensors. Information, in this case, rely on controllable devices, and it is possible to compute their reliability and other quality attributes. On the other side, the self-adaptation capabilities can be triggered also by information coming from other vehicles or infrastructure. Information, in this case, is coming from devices that could be completely uncontrollable, and it is often difficult to compute their reliability. This is completely different from the classical control of vehicles.

Autonomous vehicles bring enormous benefits to the society in terms of reducing traffic, saving fuel consumption, and especially increasing the overall safety. However, at the same time, they open new challenges that need to be addressed before we can see them driving around our cities. In the following, we describe some of the major challenges for guaranteeing safety in autonomous vehicles as pointed out also by Koopman [33] and Schmittner [51].

4.1 Safety Standards Are Not Ready for Autonomous Vehicles

Safety standards in the automotive domain, such as ISO26262, have been used to address the safety requirements of the system throughout its entire life cycle. The standard sets out detailed requirements for processes, methods, and tools used during the design and development of the electrical and electronic system of the vehicle. The requirements related to functional safety represent a challenge for the designers, who must integrate them from the earliest stages of the development process.

The life cycle begins with the description of an item, identifying its functions and interfaces. It follows the hazard analysis and risk assessment, which determine the formulation of a safety goal to be achieved by the item. An Automotive Safety Integrity Level (ASIL) is associated with the item estimating severity, probability, and controllability of hazards, based on item functional behavior. The evaluation of the ASIL is divided into four levels, from ASIL A (the lowest degree of hazard) to ASIL D (the highest degree hazard). For the safety functions with a relatively low degree of criticality, the most suitable level is probably ASIL A. On the other hand, for safety functions with a high degree of criticality, the ASIL C level may be necessary or ultimately ASIL D.

Verification and validation of the system requirements are done according to a V-model that ties each requirement to a different kind of testing. Furthermore, in an environment where systems continuously evolve, such standards may fail to identify all safety-related issues of the system. While there are clear regulations for the safety aspects regarding isolated car system, there is a lack of safety requirements for systems composed of connected cars. Furthermore, in order to be safe, the vehicle must be secure from attacks.

Traditionally, the software in the car was mainly involved in improving the vehicle's internal functions. The vehicle has been considered a closed system with no communication with the external environment. In a connected world bounding the vehicle communication with the environment is no longer possible. The attention is now moving toward connecting the car with the outside world.

Vehicles are no longer single monolithic systems but they form a connected system of systems [46]. A malfunction in one of the system can propagate to other systems. However, the standard assumes that every other system is functioning correctly, viewing hazards and malfunctions only of a single vehicle level. This obviously cannot be always the case in a system of system scenario.

From a security point of view, vehicles are opened at their boundaries, and this increases their attack surface. Security is a new factor to consider when engineering the system. The standard does not include security as a risk factor, hence not considering security breaches as a possible cause of hazards. Furthermore, although it does not assume misuse of the system, it does not consider people with malicious intentions (e.g., hackers). The deployment of vehicular networks is rapidly approaching, and their success and safety will depend on good security solutions; this will also raise privacy concerns that need to be addressed [44].

4.2 Uncertainty Is Everywhere

Self-driving vehicles will have to operate in all sorts of situations. They will be requested to operate in unpredictable and uncontrollable environments that are dominated by uncertainty. Sources of uncertainty are the environment around the vehicle, the availability of the resources that the vehicle can access at a given time, or the difficulty of predicting the other vehicle's behavior [2, 21, 24]. Uncertainty can require the system to dynamically evolve its goals and adapt itself while it is running.

As autonomous vehicles have to handle all faults or exceptions, the requirements for each scenario must be taken into consideration. The problem here is that the requirements are not fully known. The requirements should model all possible situations and unforeseeable events (e.g., extreme weather conditions, wrong traffic signs, or different kinds of animals). Enumerating all such requirements is impractical if not impossible [55].

4.3 The Use of Machine Learning

Different types of machine learning (ML) techniques are being used in various parts of autonomous vehicles, for example, to classify the detection of objects from sensors data. Such components require a set of training data which has to be independent of the validation data to avoid overfitting. One main problem with ML methods is that they are optimized for average cost function and they do not guarantee for corner cases. Challenges in this area are comprised of the fact that when using methods such as neural networks, it is difficult for humans to understand the rules that have been learned by simply looking at its weights. This is one of the hot research areas at the moment, and researchers are investigating different ways to visualize and understand the logic behind the learned neural networks in solving various tasks. Brute force testing is a widely used technique to validate the network resulting in an expensive and not always the best validation method. Furthermore, because the neural network learns the rules from a training set, if certain data is missing or wrongly correlated in the training data, this can cause the network to fail to cause safety hazards. In other words, if there is a special case that the system has not experienced, it cannot correctly predict such case; this is known as the black swan problem [42]. Hence, it is hard to detect and isolate bugs where the behavior is not expressed through traditional lines of codes but entrusted to a neural network. The network would need to be retrained with the risk to *unlearn* correct behaviors. This also motivates the safe AD architectures, where the safety of the complete system can be guaranteed.

Integrating machine learning components with traditional software is a challenge that risks leading to technical debt [52]. Technical debt indicates the long-term cost that arises when implementing a quick solution that works in the short run. Besides having all the maintainability problems of traditional software, machine learning components have special issues. Design principles, such as separation of concerns and strict abstraction boundaries, might fail to be applied in machine learning software as it uses signals from a variety of components and it has dependencies on external data. Some of the problems that the use of machine learning can cause are:

- The use of machine learning erodes boundaries and diminishes encapsulation and modular design, leading to *CACE principle*: changing anything changes everything. This causes great difficulty to maintain the code, isolate changes, and make improvements.
- Unstable data dependencies are caused when input data is changing behavior over time. This can be *implicit*, that is, a machine learning system that is being updated over time; the output of such a system is used as input to another machine learning system. For example, let us consider the case in which an input signal was previously miscalibrated. The model consuming it likely fits these miscalibrations, and a silent update that corrects the signal will have sudden ramifications for the model. In order to avoid problematic situations, a suggestion is to freeze the earlier models before using their output in a new ML system.
- The amount of needed supporting code to use generic ML packages results in glue code system. In such conditions, trying other methods or making improvements become very expensive.
- The absence of good abstraction to support ML systems (similar to what exists for database technology) makes it easy to blur the lines between components.

4.4 Validation Process Is Not Clear

A common way to assess safety in autonomous vehicles is through extensive testing, by test-driving the vehicles and evaluating the vehicle's performance. The more the vehicles drive in autonomous mode, the more experience they gather, and this will result in continuously improving the system. Companies like Waymo[8] advertises that their fleet of autonomous vehicles has driven 2.5 million miles accumulating the equivalent of 400 years human driving experience. Although such numbers seem impressive, in order to demonstrate their reliability, an autonomous vehicle might need to drive for hundreds of millions or in some cases billions of miles [28]. New methods to establish the safety of autonomous vehicles are needed. Big data analysis becomes very important in this regard; statistical signal processing and ML methods have to be developed to analyze the large amounts of data that is collected from the test vehicles or customer vehicles. Examples of such analysis includes (1) the detection of the use cases for which the sensors or the complete system provide poor performance, (2) anomaly detection in the sensor data [58], and (3) the creation of realistic simulation frameworks through the use of sensor models where the logged sensor data are used to improve the quality of the simulations performed in a cluster of computers. To be able to do so, sensor comparison frameworks [23] are required to be able to compare the data obtained from two different sensors, where one of the sensors could have, for example, significantly higher accuracy and can be used as a reference sensor.

Furthermore, the use of probabilistic models (as in the object detection) and stochastic algorithms (as in planning) poses new challenges in the validation process. Having a probabilistic system passing the test once does not guarantee that the same test will succeed every time. Testing becomes difficult for two reasons. The first reason is that due to the non-repeatability of such algorithms, it can be difficult to exercise a particular corner case. The second reason is that it is difficult to evaluate whether a result is correct or not in the case where there are multiple correct behaviors for each test case.

4.5 Nontechnical Challenges

The issues of liability in the event of an accident, the insurance coverage, and the moral dilemmas are all aspects which constitute nontechnical challenges. As the levels of automation increase, the liability in case of accident shifts from being totally a responsibility of the driver (levels 1 and 2) to the OEMs of the intelligent vehicle (levels 4 and 5). Laws are emerging today in order to approve the tests of self-driving cars on the road,[9] but laws and regulations that can allow autonomous vehicles to drive on the road are still missing.

[8]https://waymo.com.

[9]https://www.engadget.com/2017/05/12/germany-self-driving-car-test-laws/.

Another challenge of autonomous driving is played on trust: convincing people to trust the technology remains the most difficult obstacle to tackle. There are also ethical issues to be taken into consideration. When an accident happens, the outcome might have been already decided months in advance by someone else. It is still not clear who will take the decisions in the society among the programmers, the policymakers, the government, or other stakeholders. These are surely not easy decisions to make, and the decision should involve different stakeholders with different expertise going beyond the technical ones.

Even choosing an ethical framework that *minimize harm* could lead to morally arguable issues. For example, consider a scenario where a vehicle has to take the decision of either turning left or right in order to save its passengers. By turning left, he would hit a motorcyclist with the helmet on and by turning right a motorcyclist without any helmet. By following the principle of *minimizing harm*, the vehicle should choose to hit the motorcyclist with the helmet penalizing him for being responsible.

5 Conclusions

Autonomous vehicles are increasingly attracting attention, and they are investigated from several different points of views. In this chapter, we have presented an overview of the topic from several perspectives. First, we have categorized autonomous vehicles according to their levels of automation, and have described the ecosystem composed of companies and organizations working on them. Subsequently, we have analyzed the state of the art of the vehicle functionalities and how these functionalities are integrated with the overall vehicle architecture. Afterward, we have discussed the current trends including machine learning, self-adaptive systems, and continuous software engineering and how these trends fit in the automotive industry while taking into consideration the humans that will interact with the vehicle. Finally, we have pointed out some of the major challenges we believe need to be addressed in order to guarantee the safety of autonomous vehicles due to the fact that they will operate in unknown and uncontrollable environments.

References

1. Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Auton Syst 57(5):469–483
2. Autili M, Cortellessa V, Di Ruscio D, Inverardi P, Pelliccione P, Tivoli M (2011) Eagle: engineering software in the ubiquitous globe by leveraging uncertainty. In: Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on foundations of software engineering, ESEC/FSE '11. ACM, New York, NY, pp 488–491. https://doi.org/10.1145/2025113.2025199

3. Bacha A, Bauman C, Faruque R, Fleming M, Terwelp C, Reinholtz C, Hong D, Wicks A, Alberi T, Anderson D, Cacciola S, Currier P, Dalton A, Farmer J, Hurdus J, Kimmel S, King P, Taylor A, Covern DV, Webster M (2008) Odin: team VictorTango's entry in the DARPA urban challenge. J Field Robot 25(9):467–492. https://doi.org/10.1002/rob.v25:8
4. Baker CR, Dolan JM (2008) Traffic interaction in the urban challenge: putting boss on its best behavior. In: IROS, pp 1752–1758
5. Behere S, Törngren M (2016) A functional reference architecture for autonomous driving. Inf Softw Technol 73:136–150
6. Berger C, Dukaczewski M (2014) Comparison of architectural design decisions for resource-constrained self-driving cars – a multiple case-study. In: Plödereder E, Grunske L, Schneider E, Ull D (eds) Proceedings of the INFORMATIK 2014, Gesellschaft für Informatik e.V. (GI), Stuttgart, pp 2157–2168. http://subs.emis.de/LNI/Proceedings/Proceedings232/2157.pdf
7. Berger C, Rumpe B (2012) Autonomous driving – 5 years after the urban challenge: the anticipatory vehicle as a cyber-physical system. In: Goltz U, Magnor M, Appelrath HJ, Matthies HK, Balke WT, Wolf L (eds) Proceedings of the INFORMATIK 2012, Braunschweig, pp 789–798. https://arxiv.org/pdf/1409.0413v1.pdf
8. Berger C, Rumpe B (2012) Engineering autonomous driving software. In: Rouff C, Hinchey M (eds) Experience from the DARPA urban challenge. Springer, London, pp 243–271. https://doi.org/10.1007/978-0-85729-772-3_10
9. Berger C, Nguyen B, Benderius O (2017) Containerized development and microservices for self-driving vehicles: experiences & best practices. In: Proceedings of the third international workshop on automotive software architectures (WASA), p 6
10. Berns A, Ghosh S (2009) Dissecting self-* properties. In: Third IEEE international conference on self-adaptive and self-organizing systems, 2009. SASO'09. IEEE, Piscataway, pp 10–19
11. Bila C, Sivrikaya F, Khan MA, Albayrak S (2017) Vehicles of the future: a survey of research on safety issues. IEEE Trans Intell Transp Syst 18(5):1046–1065
12. Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, Jackel LD, Monfort M, Muller U, Zhang J, et al (2016) End to end learning for self-driving cars. arXiv preprint arXiv:160407316
13. Brun Y, Serugendo GDM, Gacek C, Giese H, Kienle HM, Litoiu M, Müller HA, Pezzè M, Shaw M (2009) Engineering self-adaptive systems through feedback loops. In: Software engineering for self-adaptive systems, vol 5525. Springer, Berlin, pp 48–70
14. Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka ER Jr, Mitchell TM (2010) Toward an architecture for never-ending language learning. In: AAAI, vol 5, p 3
15. Chavez-Garcia RO, Aycard O (2016) Multiple sensor fusion and classification for moving object detection and tracking. IEEE Trans Intell Transp Syst 17(2):525–534
16. Cheng BH, Giese H, Inverardi P, Magee J, de Lemos R, Andersson J, Becker B, Bencomo N, Brun Y, Cukic B, et al (2008) Software engineering for self-adaptive systems: a research road map. In: Dagstuhl seminar proceedings, Schloss Dagstuhl-Leibniz-Zentrum für Informatik
17. de Lemos R et al (2013) Software engineering for self-adaptive systems: a second research roadmap. In: Software engineering for self-adaptive systems II. Springer, Berlin, pp 1–32
18. Dollar P, Wojek C, Schiele B, Perona P (2012) Pedestrian detection: an evaluation of the state of the art. IEEE Trans Pattern Anal Mach Intell 34(4):743–761
19. El Sallab A, Abdou M, Perot E (2017) Deep reinforcement learning framework for autonomous driving. In: Electronic imaging. Autonomous vehicles and machines
20. Eliasson U, Heldal R, Lantz J, Berger C (2014) Agile model-driven engineering in mechatronic systems-an industrial case study. In: International conference on model driven engineering languages and systems. Springer, Cham, pp 433–449
21. Esfahani N, Malek S (2013) Uncertainty in self-adaptive software systems. Springer, Berlin, pp 214–238. https://doi.org/10.1007/978-3-642-35813-5_9
22. Fleming B (2014) An overview of advances in automotive electronics [automotive electronics]. IEEE Veh Technol Mag 9(1):4–9. https://doi.org/10.1109/MVT.2013.2295285

23. Florbäck J, Tornberg L, Mohammadiha N (2016) Offline object matching and evaluation process for verification of autonomous driving. In: International conference on intelligent transportation systems (ITSC), pp 107–112. https://doi.org/10.1109/ITSC.2016.7795539
24. Garlan D (2010) Software engineering in an uncertain world. In: Proceedings of the FSE/SDP workshop on future of software engineering research, FoSER '10. ACM, New York, NY, pp 125–128. https://doi.org/10.1145/1882362.1882389
25. Giaimo F, Berger C (2017) Design criteria to architect continuous experimentation for self-driving vehicles. In: Proceedings of the international conference on software architecture (ICSA), pp 203–210. http://arxiv.org/abs/1705.05170
26. Hammett R (2001) Design by extrapolation: an evaluation of fault-tolerant avionics. In: 20th DASC. 20th Digital avionics systems conference (Cat. No.01CH37219), vol 1, pp 1C5/1–1C5/12. https://doi.org/10.1109/DASC.2001.963314
27. ISO/IEC (2011) ISO/IEC/IEEE 42010:2011 Systems and software engineering – architecture description. https://www.iso.org/standard/50508.html
28. Kalra N, Paddock SM (2016) Driving to safety: how many miles of driving would it take to demonstrate autonomous vehicle reliability? Transp Res A Policy Pract 94:182–193
29. Kephart JO, Chess DM (2003) The vision of autonomic computing. Computer 36(1):41–50
30. Knauss E, Pelliccione P, Heldal R, Ågren M, Hellman S, Maniette D (2016) Continuous integration beyond the team: a tooling perspective on challenges in the automotive industry. In: Proceedings of ESEM '16. ACM, New York. https://doi.org/10.1145/2961111.2962639
31. Knauss A, Schroeder J, Berger C, Eriksson H (2017) Paving the roadway for safety of automated vehicles: an empirical study on testing challenges. In: Proceedings of intelligent vehicle symposium (IV)
32. Koopman P, Wagner M (2016) Challenges in autonomous vehicle testing and validation. Technical report, Carnegie Mellon University; Edge Case Research LLC
33. Koopman P, Wagner M (2016) Challenges in autonomous vehicle testing and validation. SAE Int J Transp Saf 4(1):15–24
34. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
35. Lu Z, Happee R, Cabrall C, Kyriakidis M, de Winter J (2016) Human factors of transitions in automated driving: a general framework and literature survey. Transp Res F 43:183–198
36. Maldonado-Bascon S, Lafuente-Arroyo S, Gil-Jimenez P, Gomez-Moreno H, López-Ferreras F (2007) Road-sign detection and recognition based on support vector machines. IEEE Trans Intell Transp Syst 8(2):264–278
37. Mallozzi P (2017) Combining machine-learning with invariants assurance techniques for autonomous systems. In: Proceedings of the 39th international conference on software engineering companion. IEEE, Piscataway, pp 485–486
38. Martens M, van den Beukel A (2013) The road to automated driving: dual mode and human factors considerations. In: Proceedings of conference on intelligent transportation systems, pp 2262–2267
39. Masek P, Thulin M, Andrade H, Berger C, Benderius O (2016) Systematic evaluation of sandboxed software deployment for real-time software on the example of a self-driving heavy vehicle. In: Proceedings of the 19th IEEE intelligent transportation systems conference (ITSC), pp 2398–2403. https://doi.org/10.1109/ITSC.2016.7795942, http://ieeexplore.ieee.org/abstract/document/7795942/
40. Montemerlo M, Thrun S, Koller D, Wegbreit B, et al (2002) Fastslam: a factored solution to the simultaneous localization and mapping problem. In: Aaai/iaai, pp 593–598
41. Montemerlo M, Becker J, Bhat S, Dahlkamp H, Dolgov D, Ettinger S, Haehnel D, Hilden T, Hoffmann G, Huhnke B, Johnston D, Klumpp S, Langer D, Levandowski A, Levinson J, Marcil J, Orenstein D, Paefgen J, Penny I, Petrovskaya A, Pflueger M, Stanek G, Stavens D, Vogt A, Thrun S (2008) Junior: the Stanford entry in the urban challenge. J Field Robot 25(9):569–597. https://doi.org/10.1002/rob.v25:9, http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6681152
42. Nassim NT (2007) The black swan: the impact of the highly improbable. Random House, New York

43. Panahandeh G, Ek E, Mohammadiha N (2017) Road friction estimation for connected vehicles using supervised machine learning. In: IEEE intelligent vehicles symposium (IV)
44. Parno B, Perrig A (2005) Challenges in securing vehicular networks. In: Workshop on hot topics in networks (HotNets-IV)
45. Pelliccione P, Knauss E, Heldal R, Ågren SM, Mallozzi P, Alminger A, Borgentun D (2017) Automotive architecture framework: the experience of Volvo cars. J Syst Archit 77:83–100. https://doi.org/10.1016/j.sysarc.2017.02.005, http://www.sciencedirect.com/science/article/pii/S1383762117300954
46. Pelliccione P, Kobetski A, Larsson T, Aramrattan M, Aderum T, Ågren M, Jonsson G, Heldal R, Bergenhem C, Thorsén A (2017) Architecting cars as constituents of a system of systems. In: Software-intensive systems-of-systems. ACM, New York
47. Pinheiro P, Collobert R (2014) Recurrent convolutional neural networks for scene labeling. In: International conference on machine learning, pp 82–90
48. Rauskolb FW, Berger K, Lipski C, Magnor M, Cornelsen K, Effertz J, Form T, Graefe F, Ohl S, Schumacher W, Wille JM, Hecker P, Nothdurft T, Doering M, Homeier K, Morgenroth J, Wolf L, Basarke C, Berger C, Gülke T, Klose F, Rumpe B (2008) Caroline: an autonomously driving vehicle for urban environments. J Field Robot 25(9):674–724. https://doi.org/10.1002/rob.20254
49. Russell S, Norvig P, Intelligence A (1995) A modern approach. Artificial Intelligence Prentice-Hall, Englewood Cliffs, pp 25–27
50. Salehie M, Tahvildari L (2009) Self-adaptive software: landscape and research challenges. ACM Trans Auton Adapt Syst 4(2):14
51. Schmittner C, Ma Z, Gruber T (2014) Standardization challenges for safety and security of connected, automated and intelligent vehicles. In: 2014 International conference on connected vehicles and expo (ICCVE), pp 941–942
52. Sculley D, Holt G, Golovin D, Davydov E, Phillips T, Ebner D, Chaudhary V, Young M, Crespo JF, Dennison D (2015) Hidden technical debt in machine learning systems. In: Advances in neural information processing systems, pp 2503–2511
53. Shalev-Shwartz S, Ben-Zrihem N, Cohen A, Shashua A (2016) Long-term planning by short-term prediction. arXiv preprint arXiv:160201580
54. Ståhl D, Bosch J (2014) Modeling continuous integration practice differences in industry software development. J Syst Softw 87:48–59. https://doi.org/10.1016/j.jss.2013.08.032
55. Sutcliffe A, Sawyer P (2013) Requirements elicitation: towards the unknown unknowns. In: 2013 21st IEEE international requirements engineering conference (RE). IEEE, Piscataway, pp 92–104
56. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction, vol 1. MIT Press, Cambridge
57. Szegedy C, Ioffe S, Vanhoucke V, Alemi A (2016) Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv preprint arXiv:160207261
58. Tashvir A, Sjöberg J, Mohammadiha N (2017) Sensor error prediction and anomaly detection using neural networks. In: The first Swedish symposium on deep learning (SSDL)
59. Thrun S, Montemerlo M, Dahlkamp H, Stavens D, Aron A, Diebel J, Fong P, Gale J, Halpenny M, Hoffmann G, Lau K, Oakley C, Palatucci M, Pratt V, Stang P, Strohband S, Dupont C, Jendrossek LE, Koelen C, Markey C, Rummel C, van Niekerk J, Jensen E, Alessandrini P, Bradski G, Davies B, Ettinger S, Kaehler A, Nefian A, Mahoney P (2006) Stanley: the robot that won the DARPA grand challenge. J Field Robot 23(9):661–692. https://doi.org/10.1002/rob.20147
60. Wintersbeger P, Green P, Riener A (2017) Am I driving or are you or are we both? A taxonomy for handover and handback in automated driving. In: Proceedings of driving assessment conference
61. Zeeb K, Buchner A, Schrauf M (2015) What determines the take-over time? An integrated model approach of driver take-over after automated driving. Accid Anal Prev 78:212–221
62. Zeeb K, Buchner A, Schrauf M (2016) Is take-over time all that matters? The impact of visual-cognitive load on driver take-over quality after conditionally automated driving. Accid Anal Prev 92:230–239