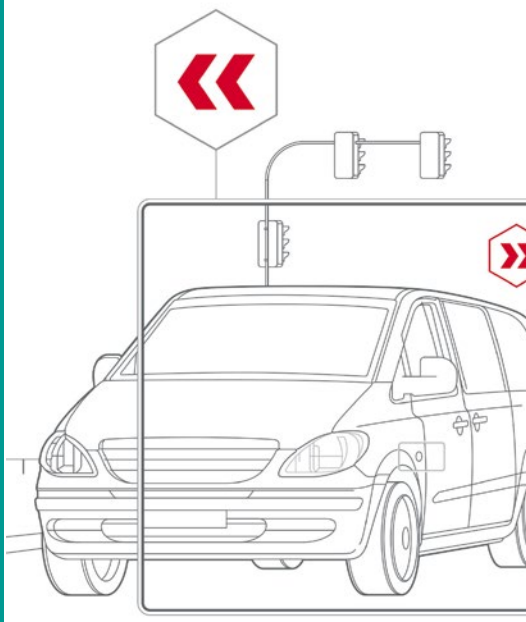
Validierung Automatisierter und Autonomer Fahrzeuge

Automatisierte und autonome Fahrsysteme kommen zunehmend zum Einsatz. Doch das Misstrauen in deren Zuverlässigkeit wächst. Die zugrundeliegenden Algorithmen sind schwer nachvollziehbar und damit intransparent. Herkömmliche Validierungen sind komplex, aufwendig und teuer. Zudem wird keine transparente Abdeckung bei Regressionsstrategien für Upgrades und Updates erreicht. Vector Consulting und das IAS der Universität Stuttgart zeigen in diesem Beitrag, dass klassische Validierungsverfahren durch kognitive Testmethoden ergänzt werden müssen.



AUTOREN

Christof Ebert
ist Geschäftsführer bei Vector Consulting in Stuttgart.

Michael Weyrich
ist Direktor des Instituts für Automatisierung und Softwaresysteme (IAS) der Universität Stuttgart.

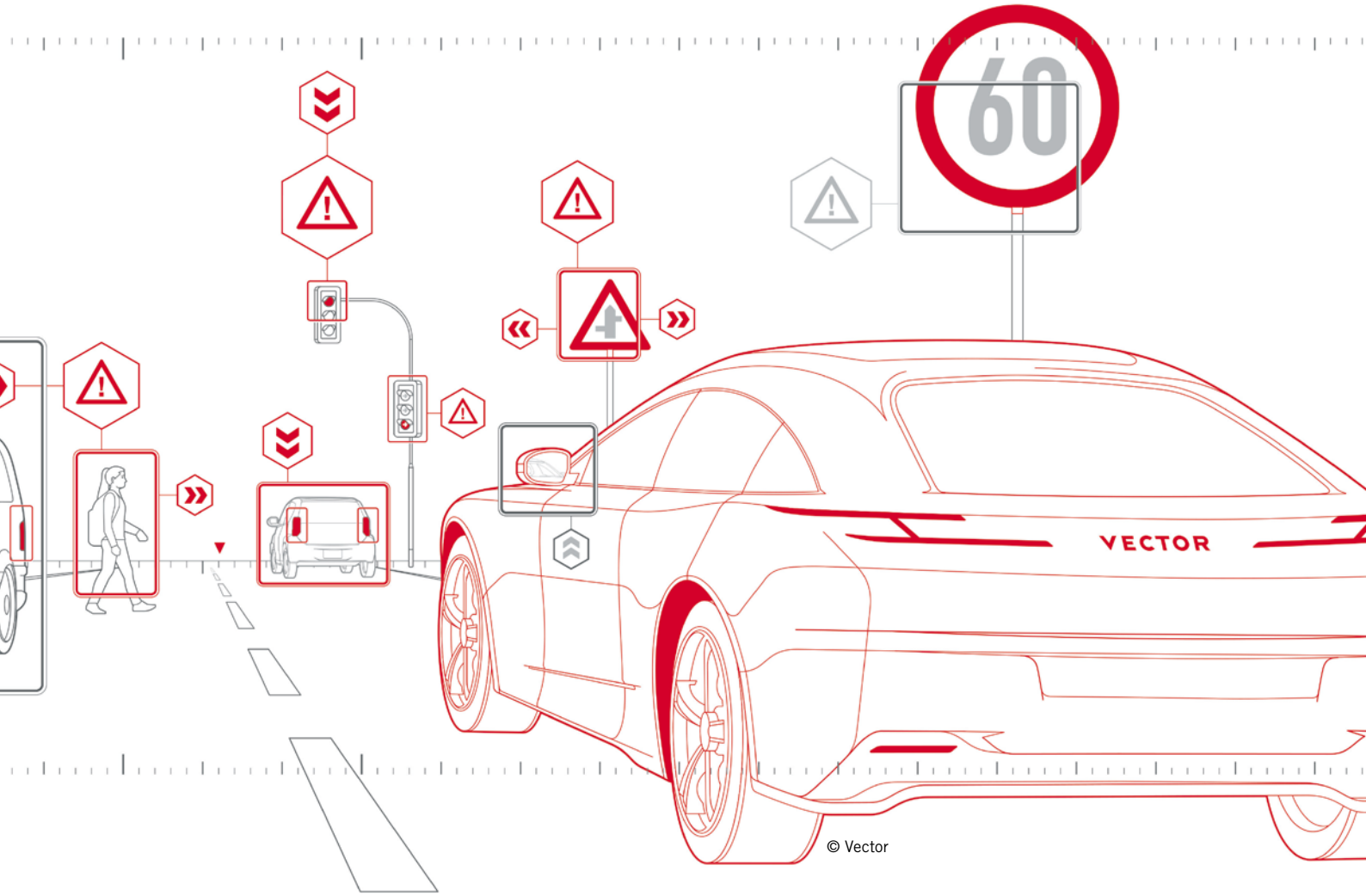
AUTONOME FAHRZEUGSYSTEME

Unsere Gesellschaft hat sich an automatisierte Systeme gewöhnt. Schon heute erkennen wir nicht mehr immer, ob wir es mit einer Maschine oder einen Menschen zu tun haben, was der ultimative Beweis für den Turing-Test ist. Das Potenzial automatisierter und autonomer Fahrsysteme ist gewaltig: Zum Beispiel wird der Einsatz autonomer Fahrzeuge bis zu 90 % der Unfälle beseitigen und bis zu 50 % der Pendelzeit pro Benutzer und Tag reduzieren [1].

BILD 1 zeigt die fünf Schritte von der Automatisierung zur Autonomie, wie sie auch aus dem menschlichen Lernen bekannt sind. Diese Schritte veranschaulichen den Weg eines einfachen und „unterstützten Verhaltens“ in Bezug auf die Erfassung und Steuerung auf niedriger Ebene hin zu „vollständigen kognitiven Systemen“ mit einem sehr hohen Grad an Autonomie.

Gleichzeitig regt sich gegenüber automatisch getroffenen Entscheidungen zunehmend Mistrauen. Insbesondere in Deutschland ist eine Diskussion entstanden, die den Einsatz künstlicher Intelligenz hinterfragt und ethische Fragestellungen adressiert. Eine aktuelle Umfrage zeigt, dass die Ablehnung automatisierter und autonomer Systeme steigt [1,2]. Diese Ablehnung wird weiter zunehmen, wenn es den Herstellern nicht gelingt, Transparenz und ein garantiertes Verhalten bei Steueralgorithmen, Regeln und Lernfunktionen für relevante Szenarien zu schaffen. Eine unzureichende Testabdeckung mag im experimentellen Stadium noch vertretbar sein, führt jedoch unter Umständen zu großen gesellschaftlichen Widerständen. Emotionale Ablehnung blockiert die weitere Entwicklung mit dem Risiko, dass auch in diesem Feld die USA und China die Führung übernehmen.

Entscheidungen autonomer Systeme sind aufgrund der Komplexität und Vielschichtigkeit oft nicht transparent nachvollziehbar und erklärbar. Automatisierte und autonome Fahrzeuge



© Vector

benötigen offensichtlich eine völlig neue Strategie zur Validierung, da aufgrund der Komplexität und der Kosten weder vereinzelte Funktionstests noch Brute-Force-Tests ausreichen.

Dieser Beitrag führt in das Thema Validierung und Zertifizierung sowie die generelle Freigabe (Homologisierung) von autonomen Fahrzeugen und deren Komponenten ein. Er gibt Einblicke in die Validierung autonomer Systeme, wie sie in der Automatisierungstechnik und Robotik zum Einsatz kommen. Er ist auch eine Übersicht zu Verfahren zur Verifikation und Validierung autonomer Fahrzeuge, skizziert aktuelle Werkzeuge und zeigt die Evolution hin zu KI-basierten Techniken für Einflussanalyse von ständigen Änderungen auf.

VALIDIERUNG AUTONOMER FAHRZEUGSYSTEME

Autonome Fahrzeugsysteme haben komplexe Wechselwirkungen mit der realen Welt. Das birgt viele Fragen hinsichtlich ihrer Validierung: Wie definiert man Zuverlässigkeit? Wie kann man die Entscheidungsfindung zurückverfolgen und danach beurteilen? Wie beaufsichtigt man? Oder, wie definiert man eine Haftung im Fehlerfall? **BILD 2** gibt einen Überblick über aktuelle Validierungstechnologien für autonome Fahrzeugsysteme. Hierbei wird horizontal die Transparenz der Validierung unterschieden. Black Box bedeutet, dass der Tester keinen Einblick in die getestete Software und Daten hat oder braucht, während White Box eine transparente Abdeckung

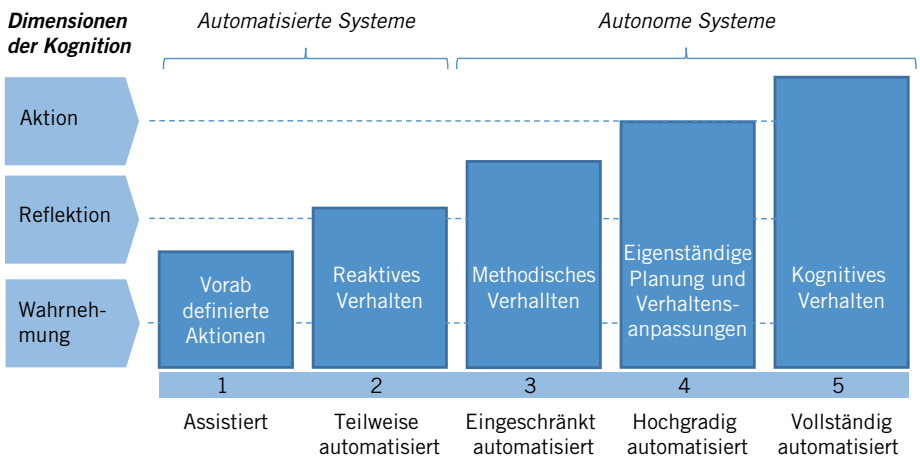


BILD 1 In fünf Stufen vom Assistenten zum autonomen System (© EbertIWeyrich)

Methode	Eigenschaften	Werkzeuge und technologien	Abdeckung	Regressions-Strategie	Stärken	Schwächen	Wirksamkeit	Effizienz
Modellierungs- und Simulationsumgebungen mit SIL, HIL, MIL	Statisch und dynamisch	Modellprüfer, z.B. Matlab, dSPACE, Vector VT, NovaCarts, Vires, PreScan	0	Beeinträchtigte Funktionen und Szenarien wiederholen (geringe Effizienz)	> Reduziert die Validierungskosten. > Entkoppelt Hardware- und Softwareentwicklung	> Viel Aufwand für eine hohe Abdeckung > Zu sehr Komponenten-orientiert > Testet nur für bekannte Szenarien > Intransparenz der Szenario-Bibliotheken	0	0
Funktionaler Test	Dynamisch, alle Funktionen	Modellierungstool für die funktionale Abstraktion mit Unit-Test-Tools (z.B. JUnit, PHPUnit), dedizierte Testumgebungen für die Stub-Generierung	0	Wiederholen der Funktionstestfälle für betroffene Funktionen	> Testet alle KI-Aspekte: Wahrnehmung, Entscheidungsfindung und ergriffene Maßnahmen. > Überprüft alle funktionalen Anforderungen	> Zu sehr Komponenten-orientiert > Nicht ausreichend, um das gesamte System zu validieren.	0	+
Integrationstest	Dynamisch	Testsuiten, Testmanagement, kombinatorische Tools wie AETG, Citrus etc.	0	Testfälle neu generieren	> Testet die Integration von Komponenten.	> Große Anzahl von Schnittstellen: Einige Links können leicht übersehen werden > Fehlerlokalisierung ist schwierig	+	+
Fehlerinjektion	Statisch, zur Restfehler-Abschätzung	Testumgebung und Fehlermodellierung, z.B. beSTORM, Sicherheitsinnovation	–	Ausgewählte spezifische Fehler (Muster) ausführen	> Schätzung der verbleibenden Fehler und der Abdeckung. > Zeigt kritische Bereiche, um sie zu verbessern	> Benötigen konkretes Verständnis der zugrunde liegenden Systemarchitektur und des darauf aufbauenden Verhaltens	–	–
Negative Anforderungen mit Misuse, Abuse, Confuse Fällen	Statisch,speziell für Sicherheit, Benutzerfreundlichkeit	Direkt modelliert und mit RE-Werkzeugen verfolgt, z.B. DOORS, Visure, PTC, PREEvision, Enterprise Architect, HP ALM	0	Wiederverwendung von situativen negativen Fällen	> Gut für Negativ-Szenarien, die vermieden werden sollen > Formalisiert nichtfunktionale Anforderungen. > Stärkt die Systemsicherheit	> Fehlende Systematik > Keine Strategie zur Abdeckung > Die Testfälle decken nicht unbedingt alle möglichen negativen Fälle ab	+	+
FMEA, FTA	Statisch, speziell für sicherheitskritische Systeme	FMEA-Arbeitsblätter, Komponentenabstraktionen, Wiederverwendung mit Bibliotheken	0	Wiederholen der Prüfungen für die geänderten Komponenten	> Für Sicherheitsanalysen gut etabliert, z.B. Angriffsbaum > Ermöglicht Korrektur von Schnittstellenfehlern	> Braucht viel Expertenwissen > Arbeitsintensiv	+	+
Experimente, empirische Teststrategien	Empirische Testgenerierung für Belastungstest, Leistung, Wärme usw.	Experiment-Spezifische Testtools wie Parasoft DTP, EggPlant, Thermal imager etc.	+	Wiederholen der Teststrategien für geänderte Funktionen	> Relativ einfach, die Testfälle zu gruppieren. > Deckt ein breites Spektrum an Systemen ab > Realitätsnahe Lasttests	> Gute Experimente brauchen viel Erfahrung > Arbeitsintensiv > Sehr wenig oder keine Automatisierung für gezielte Experimente	+	0
Spezifischer Test der Qualitäts-Anforderungen, z. B. Pen Testing, Fuzzing	Dynamisch, speziell für Qualitätsanforderungen	Spezielle Testwerkzeuge, z.B. automatische Fuzzing-Erweiterungen wie CANoe, OWASP ZAP, Vega usw.	–	Wiederholen der Tests für betroffene Szenarien und Komponenten	> Für Sicherheitsanalysen gut etabliert > Wirksame Methoden, damit das System die bekannten Qualitätsanforderungen erfüllt	> Nicht ausreichend für die Validierung der vollständigen Systemsicherheit	0	+
Brute-Force Simulation in der realen Welt, mit realistischen Szenarien	Dynamisch zur Sicherstellung der Situationsabdeckung	Aufzeichnen und Wiedergeben von aktuellen Szenariobibliotheken mit Datenloggern von verschiedenen Sensorsystemen, z. Tecnomatix, CarMaker, EB Assist, CANape	0	Wiederholung von Szenarien bei betroffenen Komponenten, z.B. anderer Sensor	> Der realen Welt am nächsten > Sehr effektiv > Überprüft alle Systeme auf einmal im Zusammenspiel > Bibliotheken zur Wiederverwendung mit allen relevanten Sensordaten (Camera, Radar, Lidar etc.) > Standardisiertes Format und Tagging der Szenarien	> Hoher Aufwand, um alle relevanten Szenarien zu erfassen und die Echtzeitdaten zu analysieren > Unklare Abdeckung > Die meisten Testfälle sind redundant > Intransparente Situationsabdeckung	+	–
Intelligente Validierung, z.B. kognitives Testen	Dynamische Testgenerierung und -auswahl je nach Situation und Umgebung	Maschinelles LernenFrameworks, wieTensorflow, Apache Spark usw. Offene Datensätze, wie nuScenes	+	Generierte Testfälle aus der Abhängigkeitsdatenbank wiederverwenden	> Kann Transparenz in der Abdeckung schaffen > Berücksichtigt automatisch Abhängigkeiten von externen Umgebungen und internen Funktionen > Automatisiert einen Großteil des Testverfahrens > Standardisiert das Szenario-Speicherformat und Tagging > Austausch von Testszenarien über die Abstraktionsebenen des V-Modells hinweg	> Hoher Aufwand beim Einrichten einer AI-basierten Testumgebung > Benötigt viel Rechenleistung > Wachsende Disziplin, d.h. nicht viele Methoden und Tools verfügbar	+	+

TABELLE 1 Validierungstechnologien für autonome Fahrzeugsysteme (© EbertIWeyrich)

schafft. Einfaches Beispiel hierfür ist der funktionale Test. Als White-Box-Ansatz mit Modified Condition/Decision Coverage (MC/DC) oder C1-Abdeckung zeigt er, welcher Code und welche Daten berührt wurden. Als Black-Box-Ansatz, anhand von Szenarien ausgewählt, erlaubt er die Abdeckung von Funktionen wie es für Cybersecurity und Regressionstests wichtig ist. Beides ist in der Verifikation relevant. Die vertikale Achse klassifiziert die Validierungstechniken anhand ihrer Automatisierbarkeit, um beispielsweise Regressionsstrategien für Software-Updates und -Upgrades vereinfachen zu können.

TABELLE 1 erläutert Validierungstechnologien für autonome Fahrzeugsysteme. Die dort genannten Werkzeuge sind allerdings eher als Impuls, denn als vollständige Liste oder gar Empfehlung zu sehen. Jedes Unternehmen implementiert heute seine eigene Methodik und Entwicklungsumgebung. Zu oft sieht manambitionierte Entwicklungsteams, komplexe Werkzeugketten, aber keine greifbare nachhaltige

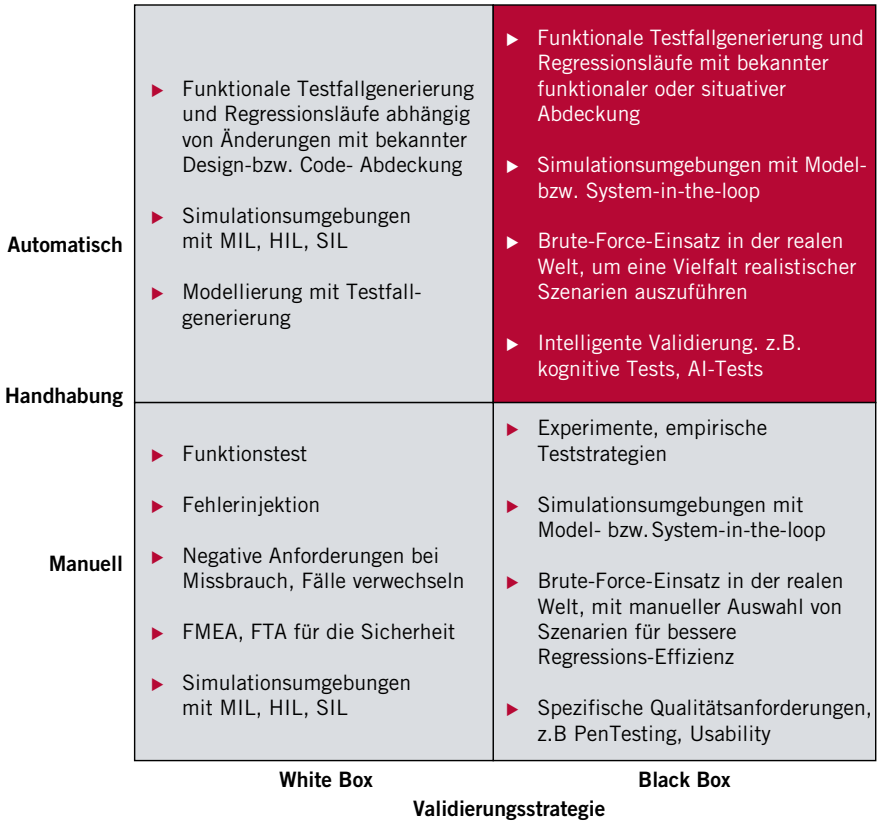


BILD 2 Strategien zur Validierung autonomer Fahrzeuge (© EbertIWeyrich)

BILD 1 ImageCaption

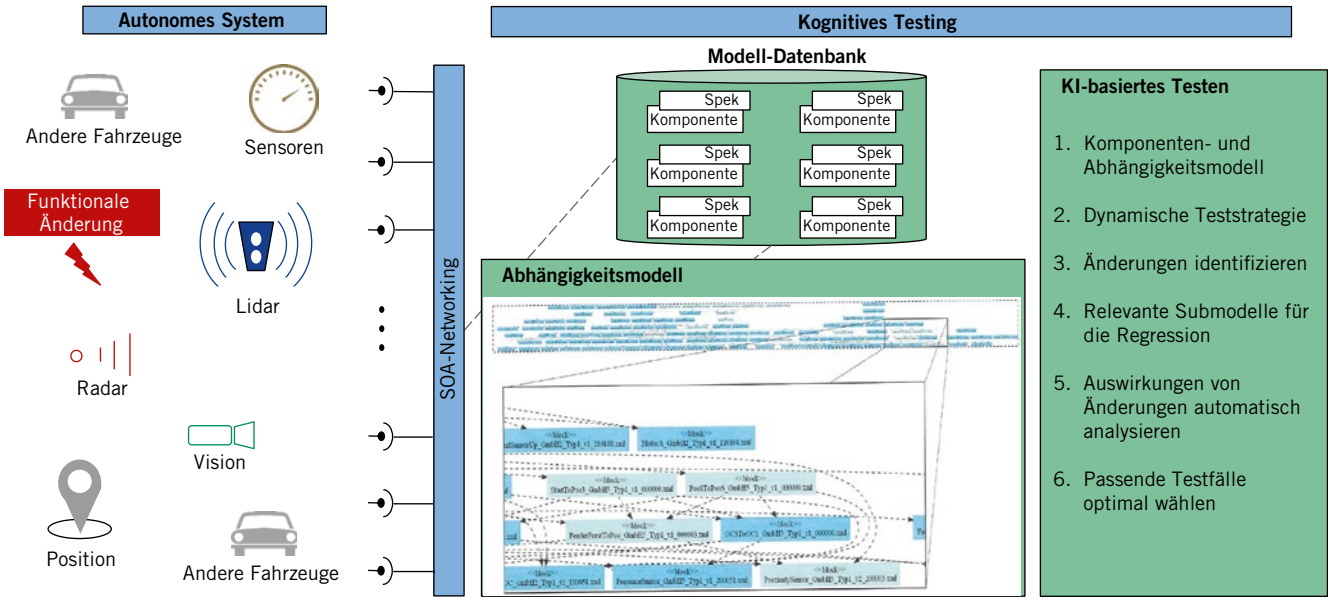


BILD 3 Kognitives Testen für autonome Fahrzeugsysteme (© EbertIWeyrich)

Teststrategie. Autonome Systeme und Fahrzeuge werden derzeit primär mit zuvor aufgezeichneten Szenarien getestet. Dazu werden auf Komponenten- und Systemebene möglichst realistische Umgebungen in ihrer sensorischen Information gespeichert und als Testfälle im zu testenden System abgefahren. Auf Systemebene werden zudem große Mengen von Szenarien in der tatsächlichen Umgebung (geschützt) „durchfahren". Zudem kommen emulierte Szenarien zum Einsatz, zum Beispiel Videosequenzen oder Signalsimulation von Sensoren, die einem Kamerasystem gezeigt werden, und die das autonome System oder eine zugehörige Komponente dann anstelle echter Umgebungsdaten bewertet.

Das Dilemma: Man kann eine beliebig große Menge von realitätsnahen Szenarien erproben, ohne genau darstellen zu können, ob das zu testende Systeme sich wirklich korrekt verhält. Der interne Steuerungsalgorithmus ist nicht transparent, gelernte und getestete Situationen können sich mit nicht getesteten und damit nicht gelernten Situationen überlappen. Die steigende Zahl von Unfällen autonomer Fahrzeuge mag als Beispiel dienen. Wenn das Fahrzeug beispielsweise gelernt hat, dass eine weiße Wolke im Hintergrund kein Hindernis darstellt, wird es bei einem tatsächlichen Hindernis, das wie eine Wolke aussieht, einen Unfall haben – wie geschehen. Auf dieser unsicheren Basis möchten weder Gesetzgeber noch Versicherer oder Anwender ein solches Fahrzeug freigeben, versichern oder nutzen. Derzeit stagnieren allerdings die Zulassungen solcher Systeme genau an diesem Punkt, da die eingebaute KI nicht transparent und regressionsfähig validiert und damit homologiert werden kann.

Die Validierung autonomer Systeme beginnt mit den Anforderungen. Ein spektakulärer Unfall ereignete sich, als ein automatisiertes Fahrzeug weiter auf der Straße fuhr, als sein Fahrer einen Herzinfarkt erlitt und das Auto nicht mehr überwachen konnte. Innerhalb weniger Sekunden überfuhr das Fahrzeug eine Mutter und ein Kind, um nicht mit einem Baum zusammenzustoßen. Offensichtlich waren die Anforderungen an das ADAS unzureichend, zumal in diesem Fall ein Antizipieren der Handlungsfolgen unzureichend erfolgte.

SOTIF (Safety of the Intended Functionality) ist ein wesentlicher Mechanismus, um die Methodik des Requirements Engineering sowohl auf moderne Softwaresysteme als auch auf die Herausforderungen automatischer Funktionen anzupassen [3,4]. Moderne Softwaresysteme, egal ob Basissoftware wie Autosar oder auch die Applikationen und Datensätze sind hochgradig dynamisch. Schon allein die Anforderungen der Cybersecurity erfordern ständige Upgrades mittels Over-the-Air(OTA)-Schnittstellen. Benutzer erwarten vom Infotainment eine ähnliche Anpassungsfähigkeit, beispielsweise mit frei konfigurierbaren Oberflächen, wie von Smartphones bekannt. Kontinuierliche Updates der Software bieten solch ständig neue Funktionen und damit Benutzererlebnisse. Kognitive Systeme passen sich an die Umgebung an und aktualisieren ihre Regeln ständig. Die zugrundeliegende Software ist also hochgradig adaptiv und muss entsprechend getestet werden [5,6].

Die Zulassung autonomer Fahrzeuge benötigt daher eine regressive Validierung, das heißt einen Test, der nach einer Veränderung der Steuerungsalgorithmen eine neuerliche Überprüfung durchführt und die Funktion sicherstellt. Damit sind sowohl in der Entwicklung, der Erprobung als auch im Einsatz zuverlässig Aussagen zur Sicherheit möglich, selbst wenn sich das System adaptiert, also verändert. Obwohl immer noch relevant, reichen traditionelle Validierungsmethoden nicht aus, um die wachsende Komplexität autonomer Autos vollständig zu testen. Maschinelles Lernen mit situativen Anpassungen sowie Software-Updates und -Upgrades erfordern neuartige Regressionsstrategien.

Hier kommen intelligente Validierungstechniken ins Spiel, die gezielt Randbedingungen manipulieren und mit kognitiven Tests situativ anpassen, **BILD 3**. Dies reduziert die fehlerhafte Auswahl und Ableitung von Testfällen aus Multi-Sensor-Sequenzen und entsprechenden Bibliotheken. Außerdem spart man enorme Mengen an Zeit, die nicht mehr

investiert werden muss, um neue Testfälle in kritischen Situationen abzuleiten und in vorhandene Bibliotheken zu integrieren. Insbesondere aber erlauben Tests basierend auf neuen Regressionsstrategien eine hohe Transparenz in Bezug auf die Abdeckung von Szenarien und eine Möglichkeit der Rückverfolgung in den Algorithmen. Somit werden Validierungsmethoden transparent und der Testprozess nachvollziehbar. Diese Testverfahren sind daher äußerst relevant und werden zukünftig in den kommenden Entwicklungsschritten zum autonomen Verhalten benötigt.

KOGNITIVES TESTEN

Algorithmische Transparenz benötigt neuartige Ansätze der Erkennung beim Testen, sogenanntes kognitives Testen, das Szenarien aufgrund definierter Randbedingungen und explizit nachvollziehbaren Abhängigkeiten zu sich adaptiv ändernder Software auswählt. Kognitive Testverfahren bauen auf einer Datenbasis auf, die Szenarien und Störungen transparent darstellt, so dass ein Soll-Verhalten für kritische Situationen, Randbedingungen, etc. definiert wird. Im Signalweg werden aus den Szenarien Signale für die Schnittstellen des autonomen Systems oder seiner Komponenten generiert. Taucht beispielsweise ein spielendes Kind plötzlich vor einem Fahrzeug auf, so wird die Reaktion des Gesamtsystems oder die Aktion seiner Komponenten, beispielsweise seiner Lenkung, getestet. Diese Signale können Simulationen für Kamera- und Radarsensoren sein, aber auch Kommunikationssignale, wie Car-to-X, Restbussimulation und Darstellung von Störungen.

Durch Parametrisierung können Sonderfälle, wie verschiedene Lichtverhältnisse, dargestellt werden. Aus dem Verhalten des zu testenden Systems werden Ist-Regeln extrahiert, die mit dem erwarteten Sollverhalten verglichen werden. Dabei wird durch den Abgleich automatisch extrahierter Ist-Regeln mit bekannten und akzeptierten Soll-Regeln ermittelt, wie das zu testende System sich im speziellen Szenario verhalten sollte. Die Soll-Regeln werden aus Gesetzen, Erfahrungen, menschlicher Expertise, Vorgaben von Ethikkommissionen aber auch aus Simulationen abgeleitet. Sie sollen transparent und dadurch einer menschlichen Prüfung zugänglich sein. Aus dem Verhalten des zu prüfenden autonomen Systems werden Regeln extrahiert, um das angelernte, in impliziten Regeln oder Neuronen-Verknüpfungen abgelegte, intransparente Verhalten transparent zu machen. Diese nun transparenten aber durchaus unscharfen (Fuzzy-)Regeln werden mit den Soll-Regeln im Verhalten verglichen. Die Validierung und Zertifizierung erfolgt anhand der Regelabweichungen [5,7,8,9].

BILD 4 gibt einen Überblick über das derzeit für vernetze Komponenten von autonomen Fahrzeugsystemen eingesetzte kognitive Testen. Anders als bei Brute Force werden die Abhängigkeiten zwischen White Box und Black Box berücksichtigt und somit Effizienz und Effektivität in Einklang gebracht. Automotive Funktionen bestehen aus dem Zusammenspiel vieler Komponenten, wie Steuerungen, Sensoren und Aktoren, die im System verteilt sind. In einem verteilten Gesamtsystem können unerwünschtes Verhalten und grundlegende Fehlfunktionen entstehen, weil an einer Stelle eine Softwareänderung erfolgte, die auf andere Komponenten durchschlägt. Daraus ergeben sich zahlreiche Fragestellungen: Wie kann die Funktion eines Systems sichergestellt werden, wenn Änderungen in den Teilkomponenten erfolgen? Wie kann die Sicherheit und das zuverlässige Verhalten garantiert werden, falls im Betrieb Softwareänderungen an einzelnen Komponenten erfolgen?

Eine Testzertifizierung erfordert ein tiefes Verständnis darüber, welchen Effekt eine Änderung im Gesamtsystem auslösen kann, obwohl diese lediglich in einer Komponente erfolgte. Wie lassen sich die Konsequenzen aufgrund der Interaktion der einzelnen Module abschätzen, ohne dass das Gesamtsystem vollständig neu getestet werden muss? Das von den Autoren entwickelte Verfahren setzt KI ein, um die Konsequenzen einer lokalen Softwareänderung an eine Komponente interpretierbar und für das Gesamtsystem abschätzbar zu machen. Auf dieser Basis kann dann die Gesamtfunktionalität zusammenhängend und auch regressiv überprüft und geeignete Testzertifikate ausgestellt werden.

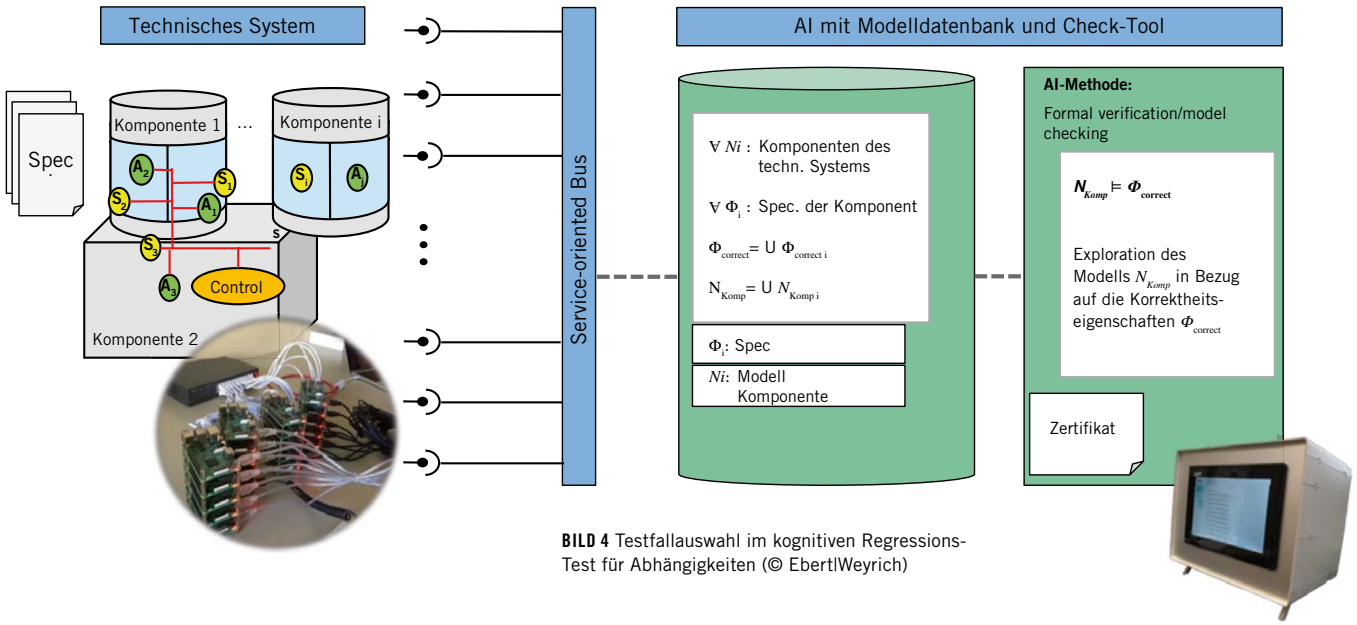


BILD 4 Testfallauswahl im kognitiven Regressions-Test für Abhängigkeiten (© Ebert|Weyrich)

Eine Schlüsselfrage ist, auf welche Weise KI den Validierungsprozess transparent und nachvollziehbar, das heißt homologisierbar, unterstützen kann. Die Autoren erprobten hierzu KI-Ansätze, die von regelbasierten Systemen, Fuzzy-Logik, Bayes-Netzen bis zu den Ansätzen mit mehreren neuronalen Netzen reichen. Dabei können verschiedene Ebenen von Validierungstests unterschieden werden, zum Beispiel die Systemebene, die Komponenten oder die Module. Somit wird der Validierungsprozess eines autonomen Systems zwar vielschichtig und detailreich, aber transparenter. Das Potenzial für kognitives Testen ist vielfältig: So gibt es auf der Systemebene Fragen, welche Testfälle in welchem Umfang ausgeführt werden müssen. Dies bedeutet, dass eine intelligente Validierung erforderlich ist, die bei der Auswahl oder Erstellung von Testfällen für die Validierung hilft.

AUSBLICK

Angesichts der wachsenden Bedeutung, und damit verbunden der Besorgnis von Nutzern und politischen Entscheidungsträgern hinsichtlich der Auswirkungen autonomer Systeme auf unser Leben und unsere Gesellschaft, müssen Softwareingenieure sicherstellen, dass autonome Funktionen und Systeme hinreichend gut und richtig funktionieren. Um Vertrauen aufzubauen, wird im Vergleich zu von Menschen betriebenen Systemen eine mindestens eine Größenordnung bessere Qualität des technischen Systems erwartet.

Dieser Aufbau von Vertrauen ist eng verknüpft mit Fragen der Validierung. Solche Validierungen hängen jedoch von vielen Faktoren ab. Autonome Fahrzeugsysteme sorgen für Effizienz und Sicherheit, da sie den Bediener von ermüdenden und fehlerträchtigen manuellen Aufgaben entlasten. Die Frage „Können wir autonomen Fahrzeugen vertrauen?“ wird in den kommenden Jahren zunehmend gestellt werden. Das Vertrauen der Öffentlichkeit in autonome Fahrzeugsysteme hängt stark von der algorithmischen Transparenz und der kontinuierlichen Validierung ab. Ein durch Softwarefehler entstandener Unfall wird heute intensiver diskutiert als die vielen Unfälle durch Alkoholeinfluss. Auf der anderen Seite zeigen aktuelle Softwarefehler mit Todesfolgen in der Luftfahrt auch eine gewisse „Gewöhnung“. Die Zahl der Fluggäste nimmt wegen Abstürzen nicht ab, da jeder weiß, dass Flugzeuge insgesamt sicher entwickelt sind.

Diese Lernkurve der Akzeptanz ist bei allen autonomen Systemen zu erkennen, historisch beispielsweise bei Smartphones, Bots mit automatischer Sprachverarbeitung und in sozialen Netzen. Eine zunehmend informierte Gesellschaft akzeptiert, dass Software zwar nie fehlerfrei ist, also ein Restrisiko vorhanden ist, aber trotzdem viele Vorteile im Vergleich zur Vergangenheit bestehen.

Alan Turing, der Vater der KI, bemerkte klug: „Wir können nur eine kurze Entfernung vor uns sehen, aber wir können dort viel sehen, was noch getan werden muss“. Dies gilt für die gerade begonnene Transformation zu autonomen Systemen in unserem Leben. Intelligente und transparente Validierung wird dabei eine entscheidende Rolle spielen.

LITERATUR

[1] Gao, P.; Kaas, H.-W.; Mohr, D.; Wee, D.: Automotive revolution: Perspective towards 2030. McKinsey, 2016.
[2] Heerwagen, M.: Rechtlich Ausgebremst. In: ATZelektronik (2018), Nr. 6, S. 8-13
[3] Ebert, C.: Systematisches Requirements Engineering. dPunkt, 6. Aufl., 2019.
[4] ISO: Road vehicles—Safety of the indented functionality, International Organization for Standardization. ISO 21448, 2019.
[5] Santori, M.; Hall, D. A.: Tackling the test challenge of next generation ADAS vehicle architecture. National Instruments, 2016. Online: http://download.ni.com/evaluation/automotive/Next_Generation_ADAS_Vehicle_Architectures.pdf, aufgerufen: 24.6.2019
[6] Rodriguez, M.; Piattini, M.; Ebert, C.: Software verification and validation technologies and tools. In: IEEE Software (2019), Vol. 36, No. 2, S. 13-24.
[7] Ebert, C.: Rule-based fuzzy classification for software quality control. In: Fuzzy Sets Systems (1994), Vol. 63, No. 3, S. 349-358.
[8] Zeller, A.; Weyrich, M.: Composition of modular models for verification of distributed automation systems. In: Proceedings of the 28th International Conference Flexible Automation and Intelligent Manufacturing (FAIM2018), Columbus, USA, 2018, S. 870-877.
[9] Shalev-Shwartz, S. et. al.: On a Formal Model of Safety and Scalable Self-Driving Cars. Intel. Online: www.mobileye.com/responsibility-sensitive-safety, aufgerufen: 24.6.2019.

READ THE ENGLISH E-MAGAZINE
Test now for 30 days free of charge:
www.ATZelectronics-worldwide.com