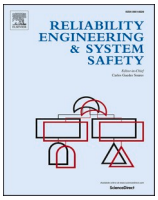
Reliability Engineering and System Safety 215 (2021) 107610

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress



Systems Approach to Creating Test Scenarios for Automated Driving Systems

Siddartha Khastgir [a,*], Simon Brewerton [b], John Thomas [c], Paul Jennings [a]

[a] WMG, University of Warwick, UK
[b] Aurrigo Driverless Technology, UK
[c] Massachusetts Institute of Technology, USA

ARTICLE INFO

Keywords:
Autonomous vehicles
STPA
Safety
Testing
test scenarios
Hazards

ABSTRACT

Increased safety has been advocated as one of the major benefits of the introduction of Automated Driving Systems (ADSs). Incorporation of ADSs in vehicles means that associated software has safety critical application, thus requiring exhaustive testing. To prove ADSs are safer than human drivers, some work has suggested that they will need to be driven for over 11 billion miles. The number of test miles driven is not, by itself, a meaningful metric for judging the safety of ADSs. Rather, the types of scenarios encountered by the ADSs during testing are critically important.

With a Hazard Based Testing approach, this paper proposes that the extent to which testing miles are *'smart miles'* that reflect hazard-based scenarios relevant to the way in which an ADS fails or handles hazards is a fundamental, if not pivotal, consideration for safety-assurance of ADSs. Using Systems Theoretic Process Analysis (STPA) method as a foundation, an extension to the STPA method has been developed to identify test scenarios. The approach has been applied to a real-world case study of a SAE Level 4 Low-Speed Automated Driving system (a.k.a. a shuttle). This paper, discusses the STPA analysis and a newly-developed test scenarios creation method derived from STPA.

1. Introduction

The last few decades have seen an increase in the amount of automation in safety critical systems in various industries e.g. aviation, manufacturing, automotive etc. In the automotive domain, the introduction of automation is driven by its many potential benefits like increased safety [9,19,52] among others. While introduction of automation has a potential to increase safety in various domains including automotive, they also add complexity especially in cyber-physical systems, requiring new risk assessment and safety verification methods for such systems [4,28,63]. The aviation industry approaches safety assessment by placing high safety integrity targets throughout the product development and use cycle [13]. However, compared to 6.5 million lines of code in a Boeing 787 airplane, current luxury cars (even without high levels of automation) have over 100 million lines of code [7,51]. The introduction of Advanced Driver Assistance Systems (ADASs) and Automated Driving Systems (ADSs) is further increasing the complexity manyfold [10].

One of the main challenges in evaluating risk and safety of complex systems with safety critical applications is that the knowledge of overall (system level) activity is poor [22]. Another trade-off for such complex systems is the balance between *"mission completion"*, *"mission abort"* and *"system or occupant survival"* [36,38]. Traditional methods view a complex system as a collection of independent components with linear relationships. However, one of the key features of complex systems is non-linearity and the dependent nature of causal links caused by feedback loops. Many real-world systems even display probabilistic dependence between sub-systems [59], thus requiring safety to be treated as an emergent property [44]. With complex systems (e.g. ADSs), accidents emerge due to the diverse interactions with a wide and open system [2]. These difficulties which emerge from traditional methods, like event tree analysis, fault tree analysis, and bow-tie diagrams, suggest the need for new approaches [8,12].

In the automotive domain, for ADSs, it is suggested that there will be a need to drive ADSs for more than 11 billion miles in order to prove that ADSs are safer than human drivers [24]. Moreover, even after 11 billion miles, such testing will *"only assure safety but not always ensure it"* [54], thus suggesting vehicle level testing or real world testing before start of

* Corresponding author:
E-mail address: S.Khastgir.1@warwick.ac.uk (S. Khastgir).

https://doi.org/10.1016/j.ress.2021.107610
Received 3 December 2019; Received in revised form 6 January 2021; Accepted 11 March 2021
Available online 16 March 2021

production (SOP) wouldn't be enough to prove safety of the automated driving systems [31,58]. While average customers are willing to pay up to $3500 for partial automation in their cars and $4900 for full automation [11], it is essential to provide them with true information about system limitations and capabilities to ensure safe and correct use and trust in the system [25].

One of the reasons behind suggesting the requirement of 11 billion miles to prove the safety of ADSs is that it would potentially reveal all possible *"black swan"* and *known unknown* scenarios or unexpected potential accidents. Identification of such scenarios remains an issue for test engineers and risk analysts dealing with complex systems [3]. Another factor is the lack of a standard set of validation metrics [56]. *"Black swan"* scenarios by their very definition may or may not be discovered even with 11 billion miles. This has also led to an increasing concern among the regulatory organisations about failures which have low probability and high consequences [50]. Therefore, rather than focussing on the number of miles, it is critically important to focus the research on the identification of the *"black swan" (a.k.a. unknown unknowns)* and *"known unknown"* scenarios.

Although requirements based testing captures the *"known knowns"* efficiently, the inability to ensure its completeness leads to the occurrence of *"unknown knowns", "known unknowns"* and the *"black swan"* scenarios. The latter three together could possibly be combined to form *"interesting"* scenarios, and to define such scenarios should be the goal for testers.

While it might be unfeasible to achieve 11 billion miles driven by ADSs, testing does have a role to play and it is important to understand not only the quantity of miles but also the quality of those miles in terms of achieving testing objectives. One might attempt a safety claim by driving 11 billion miles on a straight road on a sunny day in the middle of the desert. However, such testing may not be sufficient or even relevant for ADASs and ADSs to be deployed in environments where Vulnerable Road Users (VRUs) are present, or where it rains/snows heavily. Therefore, number of miles driven by itself is insufficient as a basis for a safety claim. The nature of scenarios comprising those miles (i.e. quality of those miles), becomes fundamental to an assessment of safety. Additionally, it is important to test the ADASs or ADSs in the Operational Design Domain (ODD) they have been designed for (e.g. a low-speed automated shuttle designed for urban environment should be tested in urban settings and not in the middle of a desert). While the aim of testing should be to increase the coverage of *"known known"* scenarios by better specification, one of the challenges of identifying *"black swan"* scenarios is their lack of correlation with time [58], suggesting their ad-hoc nature and their unpredictability.

Therefore, in order to ensure that the systems have a safe and a robust functionality, it is important to be able to define test scenarios which are able to: 1) trigger real-world use sequence 2) represent user input values 3) define and identify "all" operating conditions. However, lack of standardized methods for test scenario definition or classification, and the lack of international standards to define safety requirements for ADASs and ADSs, have led to a subjective interpretation of test scenarios and desired "safety" levels, particularly for ADASs and ADSs in vehicles [28].

1.1. Hazard Based Testing

It is suggested that for ADASs and ADSs, it is more important to identify *"how a systems fails (or misbehaves)"* as compared to *"how a system works"* [26]. Subsequently, in order to identify the smart miles, a Hazard Based Testing (HBT) approach was introduced to complement the traditional requirements based testing approach used in the automotive domain [26]. With a focus on failures (and misbehaviour), HBT has a potential to identify the *"black swan"* scenarios. The aim of using hazard based test scenarios is to increase the *"known knowns"* space and decrease the *"unknown unknowns"* space. A HBT approach has the following steps:

1. Identification of hazards
2. Creating test scenarios for the identified hazards
3. Pass criteria for the created test scenarios

Having determined that there is a need to identify hazards in order to conduct HBT, various hazard identification methods were explored. Hazard analysis methods help in identifying future causes of accidents for a system under analysis. There are various hazard analysis methods (e.g. Failure modes and effects analysis (FMEA) ([14]; Gary G. [18]), Fault Tree Analysis (FTA) [23,57], Event Tree Analysis (ETA), HAZOP [6] and STPA [34,35]. The limitations of the first four methods especially in ADASs and ADSs context led to the use of Systems Theoretic Process Analysis (STPA) as the hazard identification method for this work. The advantages of STPA over other hazard identification methods are discussed in section 2.

1.2. Research Question

Inspired by Hazard Based Testing, this paper aims to answer the following two research questions:

- How to identify hazards for ADASs and ADSs?
- How to create test scenarios for ADASs and ADSs for the identified hazards?

This paper is organized in six sections. Section 2 discusses the STPA method, section 3 introduces the extension to STPA to create test scenarios and corresponding pass criteria, section 4 illustrates the results from a real-world case study of the application of the proposed method, section 5 provides a discussion on the results and the paper concludes with a conclusion in section 6.

2. Systems Theoretic Process Analysis (STPA)

Systems Theoretic Process Analysis or STPA is a hazardous event identification method which is based on the accident causality model called STAMP (Systems Theoretic Accident Model and Processes) which in turn is grounded in systems theory and control theory [33,34]. It is designed to analyse safety in a socio-technical system with diverse interacting elements [34]. With foundations in a systems based approach, STPA identifies a broader range of hazards which may occur due to a variety of reasons including component failures, component interactions, human-error, human-automation interaction, software issues, incorrect requirements and even socio-technical and organisational factors. One of the key benefits of STPA is that it allows the person performing the analysis to identify the causal factors of the hazards and their corresponding requirements that if implemented, would prevent the hazard from occurring. Therefore, it supports the identification of preventive actions for a hazard and not just its downstream mitigation (as in other methods).

Unlike other methods like FMEA, FTA and ETA which are focussed on identifying downstream effects of failures or chain of events, STPA addresses failures as well as non-failures that lead to losses including interactions among components and systems operating exactly as designed and providing functions exactly as specified. One of the significant benefits of using STPA is its capacity to identify diverse causal factors (component failures, component interactions, requirements flaws, human-errors, design flaws, societal issues, organizational issues etc.) [16,34]. It is important to acknowledge that the quality of the STPA results is dependent on the quality of the information and the knowledge used to perform the analysis, as with any hazard analysis method [47].

While STPA has been used widely in aviation [5,15,48], space industry [21], marine industry [45,60], military applications, railways [43], organisational [43], but its application in the automotive domain is fairly recent. This is partly because there are other established methods (e.g. FMEA, FTA, HAZOP etc.) with corresponding software

tools that are embedded in current safety processes. While there are a number of STPA software tools available, they are relatively new [42]. As ADASs and ADSs are complex systems in which hazards can occur due to a number of reasons (mentioned earlier), STPA offers the most complete set of hazards [41]. Thus, STPA was chosen as the method of choice for hazard identification. Hazard identification is the first step of hazard based testing.

STPA is a four step process. The first step is to define the purpose of the analysis. These include: 1) what kinds of losses will the analysis aim to prevent; 2) will STPA be applied only to traditional safety goals like preventing loss of human life or will it be applied more broadly to security, privacy, performance, and other system properties? The second step is to build a model of the system called a control structure, which captures functional relationships and interactions by modelling the system as a set of feedback control loops. The third step is to analyse control actions in the control structure to examine how they could lead to the losses defined in the first step, and create functional requirements and constraints for the system. The fourth step identifies loss scenarios to explain why unsafe control and other unsafe behaviours might occur in the system.

2.1. STPA Step 1: Defining the purpose of analysis

Step one involves defining the purpose of the analysis and defining the system (at a high level) that is to be analysed. It also involves identifying high level "losses" or accidents for the system, which need to be avoided along with potential system hazards. These losses may include loss of human life or loss of quality of service (e.g. longer trip journey, incomplete trip journey, etc.) or may include security, privacy and performance concerns. It should be noted that only the types of losses are required at this stage and not the number in each type, for example number of crashes is not required. Following the losses, corresponding system hazards are also identified (e.g. vehicle does not maintain safe distance from nearby objects).

For ADASs and ADSs, the Operational Design Domain (ODD) of the system is defined as "Operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics" [46]. The ODD could provide an input to STPA if the ODD has already been specified. Alternatively, the STPA results in later steps could be used to create or adjust the ODD as appropriate to mitigate loss scenarios.

2.2. STPA Step 2: Creating system control structure

In STPA step 2, a hierarchical control structure model of the system is created to capture the functional interactions and feedback control loops. A control structure generally consists of controllers, actuators, sensors, controlled processes, control actions, feedback and data inputs. The control structure is refined with various iterations as the system requirements are defined and detail is added about the system. A control structure for the SAE Level 4 Low-Speed Automated Driving (LSAD) system case study is illustrated in section 4.2.

2.3. STPA Step 3: Identification of UCAs

After the system level hazards and losses are identified (in step 1), and the system control structure is defined (in step 2), STPA step 3 can be performed. STPA step 3 identifies Unsafe Control Actions (UCAs) including the contexts in which control actions can lead to a hazard or a loss. Each control action (identified in the system control structure) is analysed to consider four general cases:

- Not providing a control action causes a loss
- Providing a control action causes a loss
- Providing a control action too late, too early or out of sequence causes a loss
- Control action stopped too soon or applied too long causes a loss

In order to efficiently capture the above analysis to identify UCAs, a simple table (Table 1) can be used. In the example in Table 1, the control action *"destination command"* (provided by the occupant to undertake a journey) is analysed for the four general cases mentioned earlier. It is important to note that a UCA is not just restricted to causing a physical accident. Consequences of UCAs include mission losses or inability to provide required quality of service (e.g. too long to complete a journey).

Each element in the table is evaluated against the system hazards defined in step 1 (section 2.1), to determine whether it should be classified as a UCA. For instance, in the example discussed in Table 1, if the occupant doesn't provide the destination command *"when the occupant doesn't want to undertake a journey"*, no loss is caused. On the other hand, when the context is changed to the occupant wanting to undertake a journey, not providing the destination command will cause a loss, and is hence classified as a UCA. Identification of UCAs in STPA is done on a worst case analysis. The analysis doesn't assume that the system will know how to respond safely. On the contrary, the opposite assumption is made and the aim of the analysis of UCAs is to identify requirements and solutions to either eliminate or mitigate the UCAs.

Each of the UCAs can then be separately analysed to create system safety requirements or identify missing requirements. Additionally, a UCA has a defined structure to it, i.e., controller, control action type, control action and context (Fig. 1). It is important to maintain this structure as the proposed extension for test scenario generation (section 3) makes use of this UCA structure. In Table 1, each UCA has been linked to its corresponding system hazard. Linking UCAs to their corresponding system hazards also provides a degree of traceability in the analysis.

2.4. STPA Step 4: Identifying why UCAs and hazards might occur (Loss scenarios)

After identifying Unsafe Control Actions (UCAs), STPA step 4 involves identification of the causal factors for the UCAs by analysing each control loop for each control action in the control structure created in step 2. Causal factors are not simply fault states as identified in FMEAs. In an FMEA a known failure is analysed to understand the downstream effect and the question asked is *"what failed"*. However, in STPA step 4, the question asked is *"why did it occur"*. This can be explained by the fact that STPA offers a preventive outlook to failures and non-failures whereas other methods like FMEAs look to mitigate the effect of the failure rather than to prevent it. For loss scenarios for every UCA, two types of reasons (causal factors) must be considered [35]: a) Type a: why would Unsafe Control Actions occur? b) Type b: why would control actions be improperly executed or not executed, leading to hazards?

Table 1
Example UCA table

Control Action	Not Providing causes a loss	Providing causes a loss	Too early, too late, out of sequence causes a loss	Stopped too soon or applied too long causes a loss
Destination command	*Occupant doesn't provide destination command when occupant needs to undertake a journey. - [H5] [..]*	*Occupant provides destination command when that is unachievable or presents a road hazard. - [H2, H3, H5] [..]*	*[..]*	*[..]*

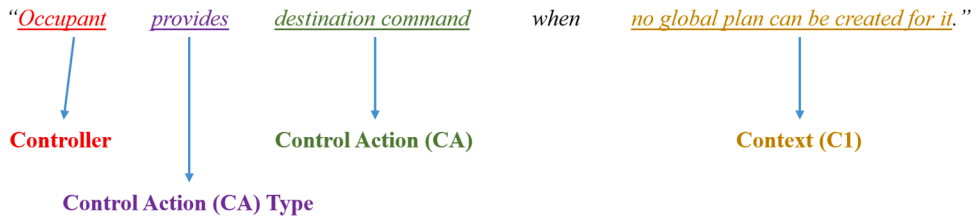


Fig. 1. Structure of an Unsafe Control Action (UCA)

Some examples of the factors considered include wrong controller process model, software design flaws, incorrect or missing requirements, missing/wrong feedback, hardware failures, contradictory control etc. A controller process model represents the controller's internal beliefs used to make decisions and may include beliefs about the process being controlled or other relevant aspects of the system or the environment [35]. Depending on the context, a software error or a hardware failure can cause either a type A or a type B scenario. A software fault in a type A scenario will be in the controller from which the control action (associated to the UCA) is being triggered (i.e. source controller). Software fault in a type B scenario will be associated with the controller within the actuator system which is receiving the control action from the source controller. The STPA Handbook [35], contains more details about the STPA process including the differences between type a and type b, and how these relate to software errors and hardware failures.

UCAs can occur due to controller beliefs that are represented by process models and acted on by the control algorithm. Thus, the first aspect to understand is the process model. Next, the reason(s) for the process models' beliefs are identified. Finally, the situations for the corresponding reasons are identified, or for control actions that are potentially not followed or executed improperly. Section 4.4 discusses the step 4 analysis for UCAs and identifies the loss scenarios.

3. Extending STPA to create test scenarios

In this section, a methodology to create test scenarios from the STPA output (step 1-4) is introduced. This involves identifying parameters for various elements of STPA. First, it is important to understand how a test scenario needs to be constructed, or in other words what are the components of a test scenario. It is suggested that a test scenario should consist of a world, actors and their behaviour [55]. However, for a test scenario to be usable for testing, it is essential to know what criteria must be met within a test scenario in order for a vehicle to receive a "passing" score for its performance or safety. Thus, a complete test scenario definition should also contain the *"pass criteria"* for the corresponding test scenario. With this understanding, a complete test scenario description will have four components:

1. Scenery
2. Environment
3. Dynamic elements
4. Pass criteria
5. Additional context

Scenery defines all geo-spatially stationary objects in the Operational Design Domain (ODD) of the vehicle [55], and includes attributes like road layouts, road furniture (e.g. barricades), traffic lights etc. Thus, scenery can be sub-categorised by various attributes (with their sub-attributes). Environment includes attributes such as weather, visibility, connectivity etc. Selection of the scenery and environment conditions to be used for testing is influenced by the ODD of the System-Under Test (SUT). While urban roads or city centres or motorway roads are part of scenery elements, if the ODD of a LSAD system includes urban roads and city centres, then the scenery parameters will be selected accordingly. However, it should be noted that the testing plan for the SUT also needs to include out-of-ODD situations or scenarios focussing on imminent ODD exit. For example, if the ODD of LSAD system only includes clear weather, SUT also needs to be tested in rainy weather to evaluate if the SUT can detect an out of ODD condition and respond appropriately.

All moving objects and actors in the world comprise the dynamic elements category [55]. Dynamic elements are sub-categorised by various parameters. Fig. 2 illustrates the top-level scenery attribute parametrisation. At the top-level, scenery can be classified into drivable area, junctions, zones, fixed road structures, temporary road structures and special structures attributes [62]. Fig. 3 illustrates the top-level dynamic elements parametrisation. At the top-level, dynamic elements can be classified into scripted traffic and non-scripted traffic. Scripted traffic refers to non-SUT agents which have pre-defined manoeuvres in the test scenario. Non-scripted traffic refers to agents which may be part of a traffic model or an actor model without predefined path.

The scenery and the dynamic elements for the test scenario are selected from the library based on the defined Operational Design Domain (ODD) of the SUT (in STPA step 1). The scenery and the dynamic elements together form the base world for the test scenario and their corresponding parameters depending upon the ODD form the base parameters for a test scenario. The pass criterion defines the set of conditions (internal to SUT or external to SUT) for which the test scenario will be considered as a pass. Additional context refers to the context element of the UCA (STPA step 3) and the causal factors for *"Potential control action not followed / How could this happen"* in STPA step four.

3.1. Additional context test scenario parameters

As mentioned in section 2.3, an Unsafe Control Action (UCA) is structured into four components. As an example, let us consider the UCA illustrated in Fig. 4 – *"GPP doesn't provide waypoint design area when there are obstacles on the path and pod is moving"*. Here GPP refers to *"Global Path Planner"* and *"pod"* refers to a low-speed automated driving system equipped vehicle, i.e. shuttle.

Test scenario parameters are identified for the scenery, environment, dynamic elements (both as per the ODD defined in STPA step one) and the additional context. It should be noted that additional context parameters have a higher priority over base parameters and the assumptions made in base parameters. The additional context comes from both STPA step three (from the UCA) and STPA step four. One of the parameters identified for the additional context is the *"context (C1)"* of a UCA (Fig. 4). In the example in Fig. 4, the content element is *"…when there are obstacles on the path and pod is moving"*. The corresponding parametrisation of *"pod is moving"* and *"overlapping distance of obstacle and the pod path"* is done with respect to pod trajectory and pod velocity. Overlapping distance, trajectory and velocity form the additional context parameters for the test scenario.

In another UCA example (Fig. 5) concerning LPP (Local Path Planner), the *"context element (C1)"* of the UCA is: *"pod is moving and is around a bend"*. The corresponding parameter for *"around a bend"* is the base map, which is a scenery element of the scenario. In this case, the base map (radius of curvature) is a STPA specific parameter for the test scenario. The other context elements like in Fig. 4, are the pod trajectory

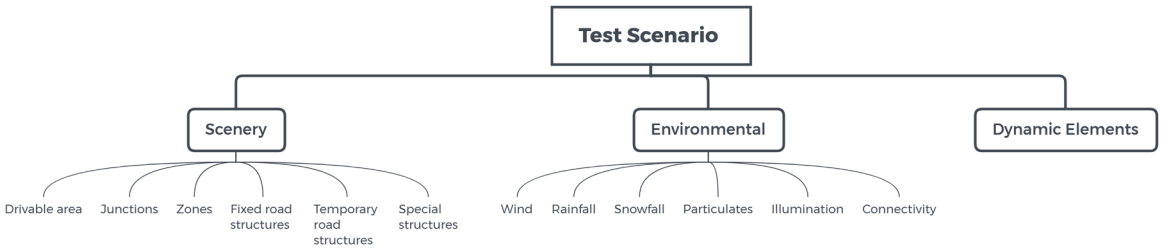


Fig. 2. Test scenario attributes for parametrisation (top level)

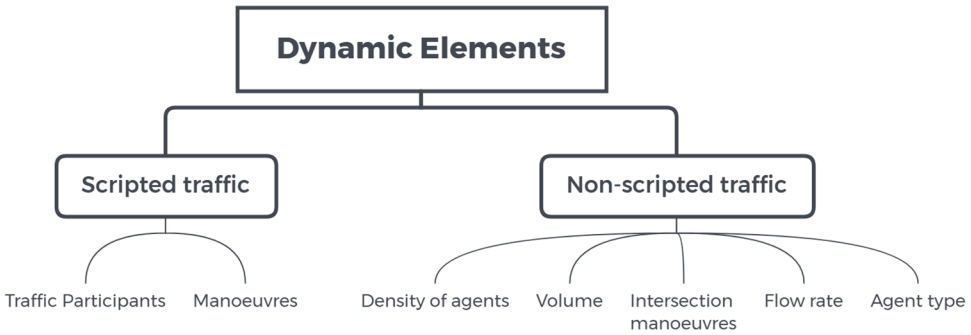


Fig. 3. Dynamic element attributes for parametrisation (top level)

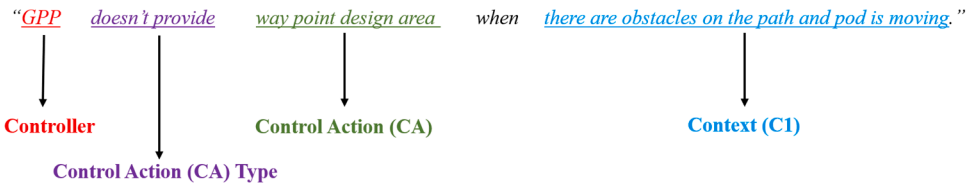


Fig. 4. Example 1: Unsafe Control Action from STPA of Low-Speed Automated Driving system [UCA# 14a]

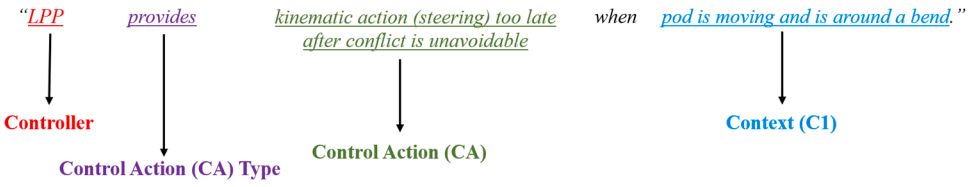


Fig. 5. Example 2: Unsafe Control Action from STPA of Low-Speed Automated Driving system [UCA# 8b]

and pod velocity which refer to the *"pod is moving"* context. The difference between an STPA specific parameter and a base parameter is that in the execution of test cases for the test scenario, the STPA specific parameters will have a higher resolution when parameter values are chosen and base parameters will have a lower resolution (or random selection depending on the defined ODD) during test scenario execution.

The second part of the "additional context" for the test scenario comes from STPA step four analysis. An example step 4 analysis is discussed below for UCA# 15b1.

UCA# 15b1: *Global Path Planning (GPP) doesn't provide way-points path command when destination command is present. – [L2, L3]*

Process Model believes (B1):

- GPP believes that a path is not possible for the given destination, current pose and base map inputs. (15b1.B1.1)
- GPP believes that destination command is not present. (15b1.B1.2)
- Etc.

Process Model believes that because (B2): For the process model belief 15b1.B1.1, potential reasons for the belief include:

- GPP believes that because it can't resolve a path in the current base map. (15b1.B2.1)
- GPP believes that because destination is out of bounds as per the current base map. (15b1.B2.2)
- Etc.

Causal Factors (B3): Causal factors leading to loss scenarios include identifying why would a control action not be followed or how could the reason for the process model belief occur. For process model belief 15b1.B2.1, the causal factors include:

- This could occur due to incorrect current pose provided to GPP. This could be because localization provides pose which doesn't match the ground truth. This could be because all sensor feeds are delayed in time leading to a low covariance error as they are coherent (suggesting the pose is correct).
- This could occur due to incorrect base map in the pod.

- Etc.

For the second part of the additional context for the test scenario, the *"Potential control action not followed"* or *"how the situation could happen"* B3 element of STPA step four is parametrised. E.g. if the *"how"* element is: *"sensor feed was delayed"* [UCA# 15b1], then the parametrisation of *"type of sensor feed"* and *"delay time"* is done for the test scenario.

3.2. Pass / fail criteria for the test scenario

The pass criteria for the test scenario are identified from STPA step four. One aim of the STPA process is to identify safety requirements to prevent a UCA from happening (if possible). Consider a control loop (with controller with process model and control algorithm, controlled process, actuator and a sensor). For a UCA to occur, the process model of the controller could represent a belief that makes the control action it is directing appear to be safe (when actually it is unsafe, i.e. a UCA). Let us call this belief as B1. If B1 were not true, the controller would not direct the original control action (i.e., original UCA). Therefore, one of the pass criterion for the test scenario would be defined as the negation of the belief B1 (process model belief), i.e., B1′, as identified in STPA step four. Secondly, the process model has the belief (B1) because of some reasons. Let us call these reasons causing the process model belief B1 as B2. Once again, if these reasons B2 were not true, B1 would not be true (when treated recursively) and subsequently, the controller would not direct the UCA. Thus, the second pass criterion for the test scenario is the negation of the reasons for the process model belief, i.e., B2′. Thus, the two pass criteria for the test scenario coming out of STPA step four for each UCA are B1′ and B2′. Therefore, the pass criteria for the test scenarios is the negation of the *"process model belief"* and negation of the *"reason for the process model belief"*. It is possible that there could be multiple process model beliefs causing the UCA and multiple corresponding reasons for those beliefs. In such a situation, there will be more than one test scenario for a single UCA and each test scenario will have its corresponding pass criteria. Table 2 illustrates the pass criteria based on UCA# 15b1.

Table 2
Pass criteria for test scenario based on process model belief and reasons for the process model belief

Unsafe Control Action (UCA)	Process Model: belief and its reasons	Pass criteria
Global Path Planning (GPP) doesn't provide way-points path command when destination command is present. – [H2, H3, H5]	B1 *GPP believes that a path is not possible for the given destination, current pose and base map inputs.*	B1' *GPP SHALL believe that a path is possible for the given destination, current pose and base map inputs*
	B2 *GPP believes that because it can't resolve a path in the current base map.*	B2' *GPP SHALL believe that because it CAN resolve a path in the current base map.*

The test scenario description (based on UCA# 15b1) can also be written in the following notation (Table 3):

The test scenario described in Table 3, involves adding delay in various sensor(s) feed(s). These test scenarios evaluate if the covariance error detects the mismatch between the ground truth and the current pose calculated based on the sensor feed (which has been delayed). The number of test scenarios (in Table 3) is calculated using the combination formula of selecting *"k (= 1 - 4)"* parameters from a given set of *"n (n = 4)"* number of parameters. As each test scenario has its corresponding "pass criterion", the number of test scenarios are doubled due to two pass criteria to obtain the total number of test scenarios.

4. Applying proposed test scenario generation method to a Low-Speed Automated Driving System: A Real-World Case Study

In order to evaluate the applicability of the proposed method to create test scenarios, the method was applied to a real-world automated driving system as a case study. In this case-study, Aurrigo Driverless Technology's fully automated Low-Speed Automated Driving (LSAD) system similar to Fig. 6, was the system under consideration or the system under test (SUT). As discussed in section 1.1, in order to apply the proposed method to create test scenarios using Hazard Based Testing, first STPA of the LSAD system was conducted.

4.1. STPA Step 1: LSAD system

The first step involves defining the system, the losses and corresponding system hazards. The Aurrigo LSAD system is an SAE Level 4 system, i.e., it is fully autonomous in a dedicated Operational Design Domain (ODD). The dedicated ODD for the LSAD system was its predefined routes in an urban environment. Additionally, the larger mobility system (of which each LSAD system was a part) consisted of a dispatcher, web-server and a fleet supervisor. The LSAD system equipped vehicle had electric propulsion with brake-by-wire and steer-by-wire functionality. It had a diverse range of sensors including multiple LiDAR and Cameras as a part of its sensor suite. For the Aurrigo LSAD system the following were identified as losses:

1 Collision with objects outside the vehicle or damage to vehicle (L1)
2 Not completing the journey with passenger and cargo (L2)
3 Time of journey being too long, i.e., service target not met (L3)
4 Loss of life or serious injury to people (L4)

The corresponding system hazards were identified as:

Table 3
Test Scenario based on UCA# 12a depicted in Table 2

Scenery	Urban areas
Dynamic elements	Pedestrians, vehicles (random selection)
Context	Obstacle position Type of sensor feed Sensor feed delay time Base map
Pass criteria (PC) for Test Scenario	**PC1:** GPP SHALL believe that a path is possible for the given destination, current pose and base map inputs. **PC2:** GPP SHALL believe that because it CAN resolve a path in the current base map.
Number of Test Scenario Parameters	4
Number of Test Scenarios	$({}^4C_1 + {}^4C_2 + {}^4C_3 + {}^4C_4 = 15)$ x 2 (number of pass criteria) = 14 or $(2^k - 1)$ x 2, where k is the number of parameters (for corresponding B1` and B2`)

Fig. 6. Low-Speed Automated Driving (LSAD) system (a.k.a. pod) (Image courtesy: Aurrigo Driverless Technology)

1 Vehicle does not maintain safe distance from nearby objects (H1) – Linked to L1
2 Vehicle enters dangerous area/region (H2) – Linked to L1
3 Vehicle exceeds safe operating envelope for environment (speed, lateral/longitudinal forces) (H3) – Linked to L1, L2, L3
4 Vehicle occupants exposed to harmful effects and/or health hazards (e.g. fire, excessive temperature, inability to escape, door closes on passengers, etc.) (H4) – Linked to L4
5 Vehicle does not follow an efficient, complete path to destination (H5) – Linked to L2, L3

Loss 2 and Loss 3 were defined as losses as the Aurrigo LSAD system is a part of a larger mobility service system. An incomplete journey or if a journey takes too long would lead to loss of customer satisfaction which will ultimately lead to loss of revenue for the business (mobility service). This is an important aspect of STPA as it can capture causal factors for social-technical losses also.

4.2. STPA Step 2: LSAD system

STPA step 2 involves creating a control structure of the system under test (SUT), i.e., Aurrigo's LSAD system. One of the most important aspect of the control structure development is the identification of the interactions (control actions and feedback) between the subsystems. The LSAD control structure can be classified into five different components (Fig. 7). The first component consists of the "world" in which the LSAD is being deployed. The second component is the "raw sensing, i.e. sensors" of the world in which the LSAD is being deployed. The third component is the "autonomous control system" of the LSAD. The forth component is the "LSAD actuation system" which responds to the autonomous control system to make the LSAD to move in the world. The fifth component is the "human input, i.e. customer" to the LSAD.

In order to refine the control structure within the LSAD (a.k.a. pod) system, first the various sub-systems needed to be identified. These subsystems are: customer, sensors, daily authorizer, world, Autonomous Control System (ACS) and LSAD actuation (Fig. 8). The ACS is comprised of a remote operator, authorizer, localization, obstacle detection classifier, global path planner, local path planner and a kinematic model. LSAD actuation is comprised of vehicle management system, steering sensor estimator, braking pressure estimator and a motor torque estimator.

The control actions and feedback interactions between the LSAD subsystems are captured in Fig. 8. In Fig. 8, red arrows indicate control actions and green arrows indicate feedbacks. The STPA analysis carried out as a part of this research involved the sensors, fleet supervisor, world and the autonomous control systems subsystems. The analysis discussed in this section is for a part of the autonomous control system as depicted in the highlighted region in Fig. 9.

4.3. STPA Step 3: LSAD system

STPA step three involves analysing each control action to identify Unsafe Control Actions (UCAs). For the autonomous control system part of the LSAD system, analysis of 12 control actions (in Fig. 8) for each of the four general cases discussed in section 2.3 resulted in 70 UCAs. Some of the control actions resulted in more than one UCA for a particular general case. For the purpose of this paper (and due to confidentiality reasons), only the UCAs for the control actions (highlighted in Fig. 8) are discussed. Some of the identified UCAs are illustrated in Table 4. Table 4 captures the analysis of the control actions (e.g. destination command, way-points path command, way-point design area command and requested kinematic command) in the four general cases discussed in section 2.3. The columns represent one type of the mentioned general cases. It should be noted that depending on the context element of the UCA, multiple UCAs may be identified for the same general cases (for example two UCAs have been mentioned for the control action *"way-points path command"* for the general case *"providing causes loss"*.).

For the eight control actions identified in Fig. 9, 51 Unsafe Control

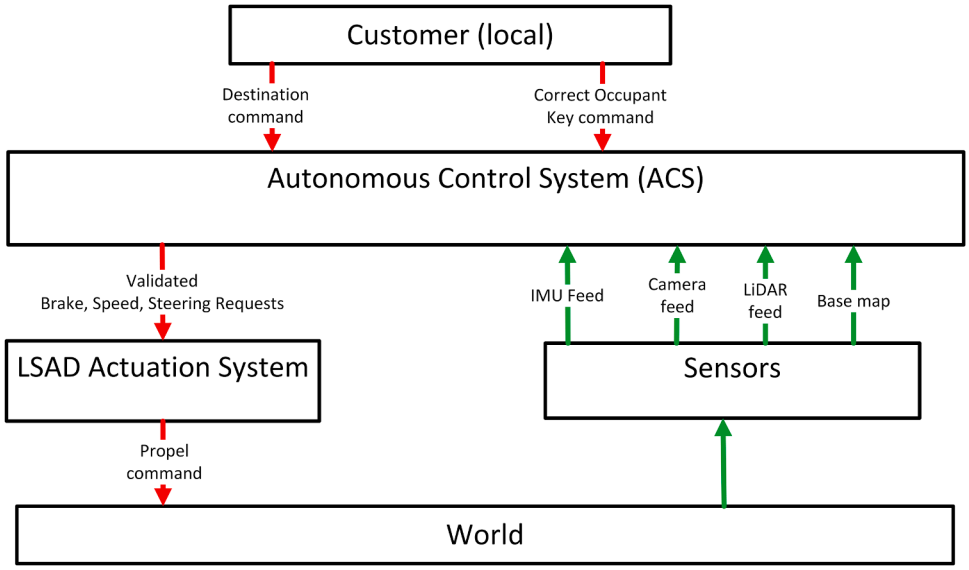


Fig. 7. High-level control structure of LSAD

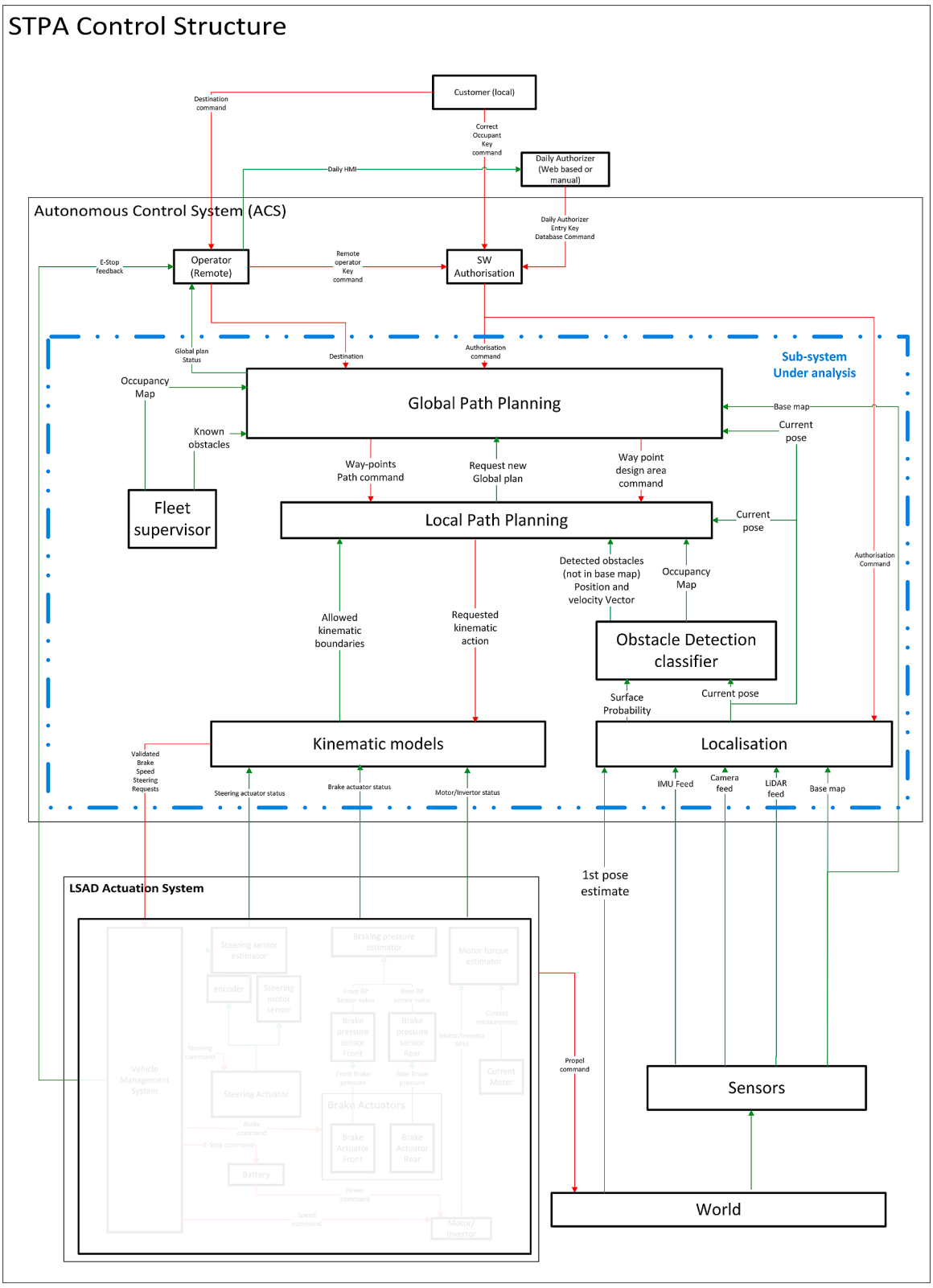


Fig. 8. STPA control structure for LSAD system

Actions were identified, with eleven UCAs being associated with a single control action – *"requested kinematic command"*. Other control actions like *"way point design area"* had four UCAs associated with them, while control actions *"destination"* and *"validated brake request"* had two and 10 UCAs associated with them respectively.

4.4. STPA Step 4: LSAD system

After identifying the UCAs (section 4.3), the reasons for the occurrence of the UCAs were identified as part of STPA step four. In this section, loss scenarios are identified and analysis for some of the UCAs have been illustrated. A loss scenario describes the causal factors that

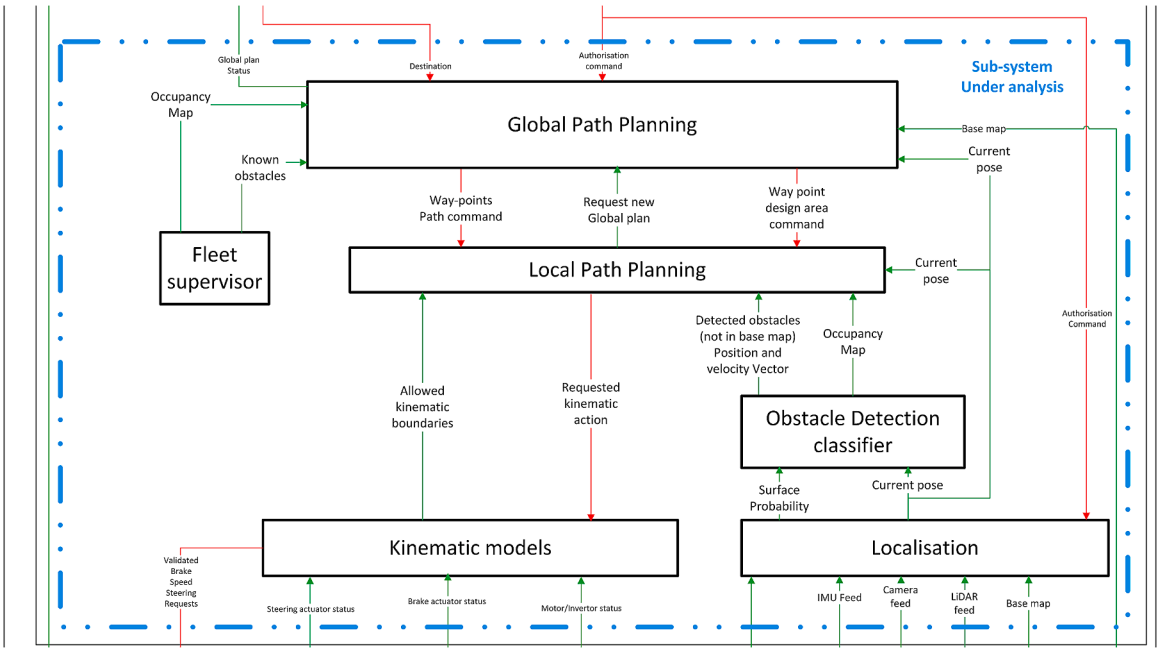


Fig. 9. Highlighted Region of the LSAD system control structure

can lead to the unsafe control actions and to hazards [35]. As discussed in section 2.4, in order to identify loss scenarios that involve process models, the *"process model belief"* and the *"reasons for the process model belief"* are identified.

4.4.1. Loss Scenarios for UCA# 15a

UCA# 15a: Local Path Planning (LPP) doesn't provide kinematic action (braking) when there is a valid local path and the pod is moving and there is an obstacle in front. – [H1, H2, H4, H5]

Process Model believes (B1): Three different process model beliefs to trigger the UCA to happen:

- LPP believes that obstacles are not in vehicle trajectory. (15aB1.1)
- LPP believes that there is a safe distance between current position of the vehicle and the obstacle (which is on vehicle trajectory). (15aB1.2)
- LPP believes that it is providing the kinematic action (braking). (15aB1.3)
- Etc.

Process Model believes that because (B2): For the process model belief 15a.B1.1, a potential reason for the belief could be:

- LPP believes that because the OD Classifier doesn't provide detected obstacles vector when obstacle is in vehicle trajectory and the Covariance Error is low (i.e., sensor data is coherent).
- Etc.

Causal Factors (B3): Possible causal factors for B2 (for 15a.B1.1) to be true:

- This could be because historical data of the pose and the surface probability shows no collision. Low Covariance Error could be because all sensor feeds are delayed in time leading to a low covariance error as they are coherent.
- This could be because OD Classifier believes that surface probability has not been present long enough for it to classify as an obstacle. This could be because of sudden changes in pose (acceleration or deceleration) or sudden appearance of surfaces (due to acceleration or deceleration of other entities).
- This could be because OD Classifier believes that the surface probability has no obstacles. This could be because the obstacles are occluded in some way.

For the process model belief 15a.B1.2, a potential reason for the belief (B2) could be:

- LPP believes that because the resolution of current pose provided by localisation (incorrect) and the detected obstacles vector OD Classifier suggests so.
- Etc.

Causal Factors (B3): Possible causal factors for B2 (for 15a.B1.2) to be true:

- This could be because localization provides pose which doesn't match the ground truth. This could be due to a featureless environment or repeating features or base map being incorrect.

4.4.2. Loss Scenarios for UCA# 15b1

UCA# 15b1: *LPP provides kinematic action (acceleration) when there is no valid local plan. – [H1, H2, H4, H5]*

Process Model believes (B1): Two different process model beliefs to trigger the UCA to happen:

- *LPP believes that it is not providing the kinematic action (acceleration).* (15b1B1.1)
- *LPP believes that the local plan is valid.* (15b1B1.2)
- *Etc.*

Process Model believes that because (B2): For the process model belief 15b1.B1.1, a potential reason for the belief could be:

- LPP believes that because LPP is not publishing the kinematic action (acceleration).
- Etc.

Table 4
UCA table with some of the UCAs identified for the LSAD system

Control Action	Not Providing causes a loss	Providing causes a loss	Too early, too late, out of sequence causes a loss	Stopped too soon or applied too long causes a loss
Destination command	[UCA# 4a] *Occupant doesn't provide destination command when occupant needs to undertake a journey. - [H5] [..]*	[UCA# 4b] *Occupant provides destination command when that is unachievable or presents a road hazard. - [H2, H3, H5] [..]*	*[..]*	*[..]*
Way-points Path command	[UCA# 13a] *Global Path Planning (GPP) doesn't provide way-points path command when destination command is present. – [H2, H3, H5] [..]*	[UCA# 13b.1] *GPP provide incorrect way-points path command when destination command is present. – [H2, H3, H5]* [UCA# 13a] *GPP provides new plan when no new plan is requested. – [H2, H3, H5] [..]*	*[..]*	*[..]*
Way-points design area command	[UCA# 14a] *Global Path Planning (GPP) doesn't provide design area when there are obstacles on the path and the pod is moving. – [H1, H2, H5] [..]*	[UCA# 14b] *Global Path Planning (GPP) provides large design area when there are obstacles on the path and when design area is larger than driveable area. – [H1, H2, H4, H5] [..]*	[UCA# 14c] *Global Path Planning (GPP) provides larger design area too early before the actual drivable area has increased and there is an obstacle in front and pod is moving. – [H1, H2, H4, H5] [..]*	[UCA# 14d] *Global Path Planning (GPP) stops providing design area too soon before vehicle has stopped and there is obstacle along the vehicle trajectory. – [H1, H2, H4, H5] [..]*
Requested kinematic command	[UCA# 15a] *Local Path Planning (LPP) doesn't provide kinematic action (braking) when there is a valid local path and the pod is moving and there is an obstacle in front. – [H1, H2, H4, H5] [..]*	[UCA# 15b1] *LPP provides kinematic action (acceleration) when there is no valid local plan. – [H1, H2,, H4, H5]* [UCA# 15b2] *LPP provides kinematic action when vehicle is not following the correct local plan. – [H1, H2, H3, H5] [..]*	[UCA# 15c1] *LPP provides kinematic action (brak ing) too late after conflict is unavoidable when there is an obstacle in front and pod is moving. – [H1, H2, H3, H4, H5] [..]*	[UCA# 15d1] *LPP stops kinematic action (steering) too soon before turn is complete when turning around a bend. – [H1, H2, H4, H5] [..]*

Causal Factors (B3): Possible causal factors for B2 (for 15b1.B1.1) to be true:

- This could be because the kinematic model is subscribing to a wrong message from LPP instead of the kinematic action (acceleration) message. This could be because of change in pointers or message IDs or communication error.
- This could be because the LPP algorithm is faulty and it is indeed publishing the kinematic action (acceleration). This could be due to a missing feedback to the LPP algorithm about the message status.

For the process model belief 15b1.B1.2, a potential reason for the belief (B2) could be:

- LPP believes that because it can resolve the local path plan on to the base map.
- Etc.

Causal Factors (B3): Possible causal factors for B2 (for 15b1.B1.2) to be true:

- This could be because the way-point path command is invalid. This could be due to a wrong base-map providing incorrect ground truth.
- This could be because LPP is receiving a set of coherent inputs as the covariance error is low. This could be because all sensor feeds are delayed in time leading to a low covariance error as they are coherent.
- This could be because the detected obstacles command is invalid. This could be due to an incorrect pose calculation.
- This could be because the way-point design area command is invalid as it doesn't describe the route constraints correctly in real time.

4.5. Creating test scenarios and test scenario parameters

The "loss scenarios" identified in section 4.4 will now be used as inputs to identify test scenarios and test scenario parameters for the subject under test. It must be noted that loss scenarios are distinct from test scenarios. As discussed in section 3, a test scenario in an ADS context is the description of a driving situation that includes the actors, scenery, environment, objectives and sequences of events, along with the evaluation criteria. A loss scenario may involve a UCA and explains the causes of hazards and losses. Loss scenarios may not have all the attributes of a test scenario.

Once the UCAs and their corresponding causal factors are identified, the next step involves parametrisation for test scenario creation. As discussed in section 3, a test scenario is comprised of scenery elements, environment elements, dynamic elements, context and pass criteria. The base parameters are selected depending upon the ODD specification of the LSAD. In this case study, the LSAD's ODD included urban areas with pre-determined routes only.

Parametrisation for test scenario generation involves parametrisation of the context element of the UCA and the parametrisation of the *"Potential control action not followed / How could this happen"* element (i.e. additional context). In this section, some of the UCAs identified in Table 4 with their corresponding causal reasons identified in section 4.4 are parametrised for test scenario creation. Additionally, the corresponding pass criteria are also identified by negating the process model belief and reasons for the belief as identified in section 4.4. Table 5 captures the test scenario parameters and their pass criteria. The test scenario representation of the Table 5 could also be made similar to Table 3.

Based on the parameters identified for each of the test scenarios, test cases can be created by assigning values to each of the parameters. While the parameter value selection process is out of scope of this research, some of the potential approaches are discussed in section 5. Applying the proposed extension method (discussed in section 3) to STPA of the LSAD system (Fig. 9), for the six control actions, over 250 parameters were identified which lead to the creation of over 3000 test scenarios (using the expression in Table 3). While 3000 test scenarios may seem a large number for manufacturers to test, it represents a significant reduction from the set of all theoretically possible test scenarios (with or without any loss or safety concern) down to the most critical test scenarios that describe exactly how the SUT can cause losses and how it must respond to prevent them. In this sense, the STPA inspired test scenario generation was found to be very effective in identifying the specific test scenarios that are the most critical to test. In addition, the 3000 test scenarios can

Table 5
Generating Test Scenarios for the LSAD system

STPA Results			Test Scenario Generation			
UCA selected for generating test scenarios	***Context from B3 (to be used for Test Scenarios)***	***Context from UCA (to be used for Test Scenarios)***	***Test scenario parameters based on UCA context***	***Test Scenario parameters based on B3***	***Pass Criterion 1***	***Pass Criterion 2***
	…This could be because sensor feed is delayed in time.			*Delay time Type of sensor delayed*	*OD Classifier shall not believe that the obstacle is not in vehicle trajectory (i.e., pose and surface probability don't collide).*	*Covariance Error shall not be low.*
[UCA# 15a] *Local Path Planning (LPP) doesn't provide kinematic action (braking) when there is a valid local path and the pod is moving and there is an obstacle in front. – [H1, H2, H4, H5]*	*… This could be because of sudden changes in pose (acceleration or deceleration) or sudden appearance of surfaces (due to acceleration or deceleration of other entities).*	*when there is a valid local path and the pod is moving and there is an obstacle in front*	*Obstacle position Velocity*	*Acceleration rate of SV Deceleration rate of SV Acceleration rate of other actors Deceleration rate of other actors*	*OD Classifier shall not believe that surface probability has not been present long enough for it to classify as an obstacle.*	*OD Classifier shall not believe that because that the obstacle is initially hard to discriminate.*
	… This could be because the obstacles is occluded in some way.			*Amount of occlusion*	*OD Classifier shall not believe that the surface probability has no obstacles.*	*OD Classifier shall not believe that because no change in surface probability or the calculated vectors don't collide.*
	… This could be because it is featureless or repeating features or base map incorrect.			*Amount of features in the world. Amount of difference in base map.*		

potentially be executed in variety of environments (e.g. simulation, test tracks or real-world testing including field operational tests). The feasibility of using such a large dataset and their selection for execution across test environments in discussed further in section 5.

It is important to highlight that this paper does not make any argument claiming that 3000 scenarios or any specific number of test scenarios are required for proving safety of an automated driving system. On the contrary, we argue that the test scenarios produced via the proposed STPA extension method, will uncover actual weakness or flaws in the system, instead of using a random scenario generation method or using a Monte-Carlo style approach where the hope is that throwing enough test cases is going to catch the problems.

5. Discussion

For real-world systems, there is a need to create the knowledge of the true capabilities and limitations of ADASs and ADSs. This knowledge can be created by evaluating such systems in various test scenarios to establish their capability to tackle them, leading to a state of *"informed safety"* [25]. However complex systems like ADASs and ADSs require new testing methods and ways to identify test scenarios [27]. Hazard Based Testing (HBT) has been proposed for ADASs and ADSs as a method to identify *"how a system fails (or misbehaves)"* to identify test scenarios. Hazard identification is the first stage of HBT. There are various methods for hazard identification (e.g. STPA, FMEA, FTA, ETA, HAZOP etc.). In this paper, Systems Theoretic Process Analysis (STPA) method has been chosen as most appropriate for complex systems like ADASs and ADSs.

For creating test scenarios from hazards and loss scenarios, an extension to STPA is presented in this paper. In order to create test scenarios: two levels of parametrisation are performed. Firstly, base parametrisation is conducted to create the scenery and the dynamic elements of a test scenario. Secondly, parametrisation of the elements of STPA output is done to create hazard based test scenario parameters. The proposed extension to STPA was aided by the existing structure of the Unsafe Control Actions (UCAs) identified by STPA step 3. The UCA structure ("controller, control action type, control action and context"), aided the parametrisation by highlighting the "context" element clearly.

In order to evaluate the applicability of the proposed method, it was then applied to a real-world system (low-speed automated driving system). In order to conduct the STPA of the LSAD system, the LSAD manufacturer – Aurrigo provided the system knowledge and expertise. Through a systems engineering approach adopted by STPA, the method was able to identify many of the situations which would have been missed by some of the traditional methods, thus reducing the *"unknown unknowns"* space by increasing the *"known knowns"* space. Additionally, *"known unknowns"* are often interpreted as the risks that we are aware of, in other words the risky scenarios that we already recognize but may have some uncertainty about the consequences. STPA will capture these risky scenarios that we already know about. But the key benefit from STPA is its ability to also capture new scenarios that nobody has thought of before or that nobody had the insight to identify as a risky scenario in the first place—the *"unknown unknowns"*. Thus, hazard-based test scenarios via STPA and its proposed extension complements test scenarios identified via requirement based testing approach, increasing the *"known knowns"*.

STPA process also enables the identification of system safety requirements from both step 3 (via UCAs) and step 4 (via causal factors) in order to prevent the hazards [1,40]. For a safe operation of the ADS, there would be a need to prevent or detect the causal factors identified in step 4, which will provide further safety requirements for the system. Along with safety requirements, the STPA process also highlights a set of assumptions of the system. Monitoring the validity of these assumptions

indicate the potential for an accident, e.g. monitoring the ODD of the ADS [32].

The application of the proposed method for a SAE level 4 automation LSAD system led to the creation of over 3000 test scenarios. While this may be considered a large set for execution in real-world testing, there is a need to understand which test scenarios (out of over 3000 identified) should be executed in real-world or test track testing. There is a wide consensus in industry and research community about the major role played by simulation based testing in the evaluation of ADSs [53]. This is due to potential reduction in the resource investment for test track and real-world testing. Thus, an important area of future research includes creating a methodology to select test scenarios across different test environments (e.g. simulation, test tracks and real-world testing). One potential approach includes using the identified hazard-based test scenarios in a simulation environment to identify the values of test scenario parameters for which the system fails. If one can identify parameter value combinations for which the system will fail, only those combinations of parameter values could be used for test track and real-world testing, leading to a reduction in the test scenario space. Therefore, an area of future research would involve creating an algorithm to select parameter values for the test scenario parameters which violated the pass criteria. Various white box and black box methods can be used for this type of parameter selection. One of the promising techniques is Bayesian Optimisation (BO), which converts the parameter selection problem into an optimisation problem [17]. Alternatively, other approaches like using constrained random testing may also be used for parameter identification [29] Additionally, each individual test scenario may not require a unique test execution run as multiple test scenarios could be evaluated using the same test execution run.

The quality of the test scenarios is based on the quality of the method (i.e. extended - STPA) used for test scenario generation. In other words, the test scenario quality is judged based on their coverage over the causes of losses. It is not based on the number of tests alone, but by examining how well the individual test scenario characteristics align with the actual causes of losses. Studies undertaking comparisons of various safety analysis methods have already shown that STPA has more coverage in identifying system flaws as compared to other approaches [20,41].

Future research may include formalising the *"context"* element of the UCA (in step 3) and the causal factors (in step 4). Formalisation would assist in automation of the test scenario parameter identification processes and may lead to automatically populating test scenarios parameters in Table 5. Furthermore, future research to address the dynamic nature of complex systems like ADSs due to Over-The-Air (OTA) updates, sensor ageing or degradation, software ageing etc. is another promising research area [30,39,49]. The dynamic changes may potentially be in the system itself or in its operating environment, with performance quality, failure robustness, mission completion capability being the objectives that may be optimised [37,61]. By monitoring the system assumptions, an STPA of a dynamic system can be accommodated either by iterations in the control structure (step 2) followed by step 3 and step 4 of the affected parts or by step 4 analysis of causal factors. The former will be true for system changes due to OTA, while the latter would deal with sensor and software ageing.

6. Conclusion

The literature has suggested that to test an ADS in order to assess its safety and prove that it is 20% better than human driven vehicle, it needs to be driven for over 11 billion miles [24]. Hazard Based Testing (HBT) and the tradition of sociotechnical systems instead suggest that the number of miles driven, by itself is not a meaningful metric for judging confidence in ADAS or ADS. Rather the types of scenarios encountered by an ADS during testing are critically important as the focuses needs to be on *"how a system fails (or misbehaves)"*. The nature of scenarios is fundamental to an assessment of safety.

The concept of Hazard Based Testing (HBT) involves creating hazard based test scenarios. To identify hazards, Systems Theoretic Process Analysis (STPA) was used as it has been demonstrated to identify hazards that might be missed by other hazard identification methods like FMEA, FTA, HAZOP, ETA etc. especially for complex systems involving human-automation interaction. Grounded in systems engineering and controls engineering, STPA is a four step process which considers safety as a control problem in which control flaws can lead to an accident/loss. STPA identifies Unsafe Control Actions (UCA) and their causal factors. This paper proposes an extension to STPA to create test scenarios for each of the UCAs identified as a part of the STPA method. The proposed method also identifies pass criteria for the test scenarios. The proposed test scenario consists of 1) scenery 2) environment 3) dynamic elements 4) additional context and 5) pass criteria. The scenery, environment and dynamic elements are selected according to the Operational Design Domain (ODD) of the vehicle. Additional context is selected from the STPA output (context element in step 3 and causal factors in step 4). STPA specific parameters have a higher priority over ODD parameters and override them. The proposed method was applied to a real-world case study of a Low-Speed Automated Driving (LSAD) system. The STPA analysis of a part of the Autonomous Control System of the LSAD system with eight control actions yielded 51 Unsafe Control Actions. This corresponded to the creation of over 3000 test scenarios with over 250 parameters associated with them.

This paper doesn't make any argument for the number of test scenarios needed for proving safety of an ADS. The test scenarios produced via the proposed STPA - extension method, are a targeted approach to uncover actual weakness and flaws in the ADS, as compared to random (or constrained random) scenario generation where the hope is to stimulate the system with as many test scenarios as possible to catch flaws. Thus, making the proposed extended - STPA based method more efficient in uncovering system flaws.

Future work can explore techniques for grouping similar scenarios, and identifying the most critical ones to be tested. This would complement the systems-based framework described here, which has been constructed to offer a valid, robust, and efficient hazard-based approach for evaluating ADS safety.

Authors Statement Name	Contribution
Siddartha Khastgir	Conceptualization, Methodology, Formal analysis, Investigation, Data Curation, Writing – Original Draft, Writing - Review & Editing, Visualisation,
Simon Brewerton	Formal analysis, Investigation, Writing - Review & Editing, Project administration, Funding acquisition
John Thomas	Validation, Data curation, Writing - Original Draft, Writing - Review & Editing
Paul Jennings	Conceptualization, Methodology, Validation, Writing - Original Draft, Resources, Writing - Review & Editing, Project administration, Funding acquisition

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The work presented in this paper has been carried under the EPSRC Grant (Grant EP/K011618/1), INTACT project – part-funded by Centre for Connect and Autonomous Vehicles (CCAV), HM Govt. grant (Grant reference 102587) and the UKRI Future Leaders Fellowship (Grant MR/S035176/1). The authors would also like to thank Mr Gunwant Dhadyalla for his contributions to the management of the project. Additionally, the authors would like to thank the WMG centre of HVM Catapult and WMG, University of Warwick, UK, for providing the

necessary infrastructure for conducting this study. WMG hosts one of the seven centres that together comprise the High Value Manufacturing Catapult in the UK. The authors would also like to thank three anonymous reviewers and the editor for their detailed comments on previous versions of the paper, which has helped considerably to improve the quality of the paper.

References

[1] Allison CK, Revell KM, Sears R, Stanton NA. Systems Theoretic Accident Model and Process (STAMP) safety modelling applied to an aircraft rapid decompression event. Saf. Sci. 2017;98:159–66. https://doi.org/10.1016/j.ssci.2017.06.011.
[2] Årstad, I., Aven, T., 2017. Managing major accident risk : Concerns about complacency and complexity in practice 91, 114–121. https://doi.org/10.1016/j.ssci.2016.08.004.
[3] Bjerga T, Aven T, Zio E. Uncertainty treatment in risk analysis of complex systems: The cases of STAMP and FRAM. Reliab. Eng. Syst. Saf. 2016;156:203–9. https://doi.org/10.1016/j.ress.2016.08.004.
[4] Bolbot V, Theotokatos G, Bujorianu LM, Boulougouris E, Vassalos D. Vulnerabilities and safety assurance methods in Cyber-Physical Systems: A comprehensive review. Reliab. Eng. Syst. Saf. 2019;182:179–93. https://doi.org/10.1016/j.ress.2018.09.004.
[5] Castilho DS, Urbina LMS, de Andrade D. STPA for continuous controls: A flight testing study of aircraft crosswind takeoffs. Saf. Sci. 2018;108:129–39. https://doi.org/10.1016/j.ssci.2018.04.013.
[6] CENELEC. Hazard and operability studies (HAZOP studies). Application guide - EN 61882; 2016.
[7] Charette RN. This Car Runs on Code. IEEE Spectr 2009.
[8] Chen C, Reniers G, Khakzad N. Integrating safety and security resources to protect chemical industrial parks from man-made domino effects : A dynamic graph approach. Reliab. Eng. Syst. Saf. 2019:1–13. https://doi.org/10.1016/j.ress.2019.04.023.
[9] Cicchino JB. Effectiveness of forward collision warning and autonomous emergency braking systems in reducing front-to-rear crash rates. Accid. Anal. Prev. 2017;99:142–52. https://doi.org/10.1016/j.aap.2016.11.009.
[10] Cuer R, Piétrac L, Niel E, Diallo S, Minoiu-Enache N, Dang-Van-Nhan C. A formal framework for the safe design of the Autonomous Driving supervision. Reliab. Eng. Syst. Saf. 2018;174:29–40. https://doi.org/10.1016/j.ress.2018.01.014.
[11] Daziano RA, Sarrias M, Leard B. Are consumers willing to pay to let cars drive for them ? Analyzing response to autonomous vehicles. Transp. Res. Part C 2017;78:150–64. https://doi.org/10.1016/j.trc.2017.03.003.
[12] Denney E, Pai G, Whiteside I. The role of safety architectures in aviation safety cases. Reliab. Eng. Syst. Saf. 2019;191:106502. https://doi.org/10.1016/j.ress.2019.106502.
[13] Dodd I, Habli I. Safety certification of airborne software : An empirical study. Reliab. Eng. Syst. Saf. 2012;98:7–23. https://doi.org/10.1016/j.ress.2011.09.007.
[14] Duckworth HA, Moore RA. Social responsibility: Failure mode effects and analysis. Soc. Responsib. Fail. Mode Eff. Anal. 2010:1–185. https://doi.org/10.1201/EBK1439803721.
[15] Fleming CH, Spencer M, Thomas J, Leveson N, Wilkinson C. Safety assurance in NextGen and complex transportation systems. Saf. Sci. 2013;55:173–87. https://doi.org/10.1016/j.ssci.2012.12.005.
[16] France ME. Engineering for Humans : A New Extension to STPA. MIT. 2017.
[17] Gangopadhyay B, Khastgir S, Dey S, Dasgupta P, Montana G, Jennings P. Identification of Test Cases for Automated Driving Systems Using Bayesian Optimization. In: Proc. of the IEEE Intelligent Trasnportation Systems Conference 2019; 2019. Auckland.
[18] Kelm Gary G. Failure Modes and Effects Analysis (FMEA). Critical Items List (CIL), and Fault Tree Analysis (FTA). 2010.
[19] Guériau M, Billot R, El Faouzi NE, Monteil J, Armetta F, Hassas S. How to assess the benefits of connected vehicles? A simulation framework for the design of cooperative traffic management strategies. Transp. Res. Part C Emerg. Technol. 2016;67:266–79. https://doi.org/10.1016/j.trc.2016.01.020.
[20] Ishimatsu T, Leveson N, Thomas J, Katahira M, Miyamoto Y, Nakao H. Modeling and hazard analysis using STPA. In: Proc. of the 4th IAASS Conference, Making Safety Matter; 2010.
[21] Ishimatsu T, Leveson NG, Thomas JP, Fleming CH, Katahira M, Miyamoto Y, Ujiie R, Nakao H, Hoshino N. Hazard analysis of complex spacecraft using systems-theoretic process analysis. J. Spacecr. Rockets 2014;51:509–22. https://doi.org/10.2514/1.A32449.
[22] Jensen A, Aven T. A new definition of complexity in a risk analysis setting. Reliab. Eng. Syst. Saf. 2018;171:169–73. https://doi.org/10.1016/j.ress.2017.11.018.
[23] Kaiser B, Liggesmeyer P, Mäckel O. A New Component Concept for Fault Trees. Proc. 8th Aust. Work. Saf. Crit. Syst. Softw. 2003;33:37–46.
[24] Kalra N, Paddock SM. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transp. Res. Part A Policy Pract. 2016;94:182–93. https://doi.org/10.1016/j.tra.2016.09.010.
[25] Khastgir S, Birrell S, Dhadyalla G, Jennings P. Calibrating trust through knowledge: Introducing the concept of informed safety for automation in vehicles. Transp. Res. Part C Emerg. Technol. 2018;96:290–303. https://doi.org/10.1016/j.trc.2018.07.001.
[26] Khastgir S, Birrell S, Dhadyalla G, Jennings P. The Science of Testing: An Automotive Perspective. SAE Technical Paper. 2018. https://doi.org/10.4271/2018-01-1070. 2018-01-1070.
[27] Khastgir S, Birrell S, Dhadyalla G, Jennings P. Identifying a gap in existing validation methodologies for intelligent automotive systems: Introducing the 3xD simulator. In: Proc. of the IEEE Intelligent Vehicles Symposium 2015; 2015. p. 648–53.
[28] Khastgir S, Birrell S, Dhadyalla G, Sivencrona H, Jennings P. Towards increased reliability by objectification of Hazard Analysis and Risk Assessment (HARA) of automated automotive systems. Saf. Sci. 2017;99:166–77. https://doi.org/10.1016/j.ssci.2017.03.024.
[29] Khastgir, S., Dhadyalla, G., Birrell, S., Redmond, S., Addinall, R., Jennings, P., 2017 b. Test Scenario Generation for Driving Simulators Using Constrained Randomization Technique, in: SAE Technical Paper# 2017-01-1672. https://doi.org/10.4271/2017-01-1672.
[30] Khastgir S, Sivencrona H, Dhadyalla G, Billing P, Birrell S, Jennings P. Introducing ASIL inspired Dynamic Tactical Safety Decision Framework for Automated Vehicles. In: Proc. of the IEEE Conference on Intelligent Transportation Systems, Proceedings (ITSC) 2017; 2017. p. 1–6. https://doi.org/10.1109/ITSC.2017.8317868.
[31] Koopman P, Wagner M. Challenges in Autonomous Vehicle Testing and Validation. SAE Int. J. Transp. Saf. 2016;4. https://doi.org/10.4271/2016-01-0128. 2016-01–0128.
[32] Leveson N. A systems approach to risk management through leading safety indicators. Reliab. Eng. Syst. Saf. 2015;136:17–34. https://doi.org/10.1016/j.ress.2014.10.008.
[33] Leveson N. A new accident model for engineering safer systems. Saf. Sci. 2004;42:237–70. https://doi.org/10.1016/S0925-7535(03)00047-X.
[34] Leveson NG. Engineering a Safer World - Systems Thinking Applied to Safety. The MIT Press; 2012.
[35] Leveson NG, Thomas JP. STPA Handbook. 2018. https://doi.org/10.2143/JECS.64.3.2961411.
[36] Levitin G, Finkelstein M, Dai Y. Mission abort policy optimization for series systems with overlapping primary and rescue subsystems operating in a random environment. Reliab. Eng. Syst. Saf. 2020;193:106590. https://doi.org/10.1016/j.ress.2019.106590.
[37] Levitin G, Xing L, Dai Y. Reliability versus expected mission cost and uncompleted work in heterogeneous warm standby multiphase systems. IEEE Trans. Syst. Man, Cybern. Syst. 2017;47:462–73. https://doi.org/10.1109/TSMC.2015.2505643.
[38] Levitin G, Xing L, Luo L. Influence of failure propagation on mission abort policy in heterogeneous warm standby systems. Reliab. Eng. Syst. Saf. 2019;183:29–38. https://doi.org/10.1016/j.ress.2018.11.006.
[39] Levitin G, Xing L, Xiang Y. Cost minimization of real-time mission for software systems with rejuvenation. Reliab. Eng. Syst. Saf. 2020;193:106593. https://doi.org/10.1016/j.ress.2019.106593.
[40] Mahajan HS, Bradley T, Pasricha S. Application of systems theoretic process analysis to a lane keeping assist system. Reliab. Eng. Syst. Saf. 2017;167:1339–51. https://doi.org/10.1016/j.ress.2017.05.037.
[41] Martínez RS. System Theoretic Process Analysis of Electric Power Steering for Automotive Applications. Massachusetts Institute of Technology (MIT; 2015.
[42] MIT, 2020. Partnership for Systems Approaches to Safety and Security (PSASS) - STAMP Tools [WWW Document]. URL http://psas.scripts.mit.edu/home/2016-2/ (accessed 5.1.20).
[43] Read GJM, Naweed A, Salmon PM. Complexity on the rails: A systems-based approach to understanding safety management in rail transport. Reliab. Eng. Syst. Saf. 2019;188:352–65. https://doi.org/10.1016/j.ress.2019.03.038.
[44] Roed-larsen S, Stoop J. Modern accident investigation – Four major challenges. Saf. Sci. 2012;50:1392–7. https://doi.org/10.1016/j.ssci.2011.03.005.
[45] Rokseth B, Utne IB, Vinnem JE. Deriving verification objectives and scenarios for maritime systems using the systems-theoretic process analysis. Reliab. Eng. Syst. Saf. 2018;169:18–31. https://doi.org/10.1016/j.ress.2017.07.015.
[46] SAE, 2018. Surface Vehicle Recommended Practice: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016). https://doi.org/10.4271/2012-01-0107.
[47] Salmon PM, Cornelissen M, Trotter MJ. Systems-based accident analysis methods : A comparison of Accimap, HFACS, and STAMP. Saf. Sci. 2012;50:1158–70. https://doi.org/10.1016/j.ssci.2011.11.009.
[48] Schmid D, Vollrath M, Stanton NA. The System Theoretic Accident Modelling and Process (STAMP) of medical pilot knock-out events: Pilot incapacitation and homicide-suicide. Saf. Sci. 2018;110:58–71. https://doi.org/10.1016/j.ssci.2018.07.015.
[49] Shavit M, Gryc A, Miucic R. Firmware update over the Air (FOTA) for automotive industry. SAE Technical Paper. 2007. https://doi.org/10.4271/2007-01-3523. 2007-01-3523.
[50] Stewart MG, Netherton MD. A probabilistic risk-acceptance model for assessing blast and fragmentation safety hazards. Reliab. Eng. Syst. Saf. 2019;191:106492. https://doi.org/10.1016/j.ress.2019.05.004.
[51] Strandberg K, Olovsson T, Jonsson E. Securing the Connected Car: A Security-Enhancement Methodology. IEEE Veh. Technol. Mag. 2018;13:56–65. https://doi.org/10.1109/MVT.2017.2758179.
[52] Tingvall C. The Zero Vision: A Road Transport System Free from Serious Health Losses. Transp. Traffic Saf. Heal. New Mobil. 1997:37–57.
[53] Transport Systems Catapult, 2018. Regulating and Accelerating Development Of Highly Automated And Autonomous Vehicles Through Simulation And Modelling.
[54] Transport Systems Catapult. Taxonomy of Scenarios for Automated Driving. 2017.

[55] Ulbrich S, Menzel T, Reschka A, Schuldt F, Maurer M. Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving. In: Proc. of the 2015 IEEE 18th 18th International Conference on Intelligent Transportation Systems; 2015. https://doi.org/10.1109/ITSC.2015.164.
[56] Vanslette K, Tohme T, Youcef-Toumi K. A general model validation and testing tool. Reliab. Eng. Syst. Saf. 2020;195:106684. https://doi.org/10.1016/j.ress.2019.106684.
[57] Vesely WE, Roberts NH. Fault Tree Handbook. 1981.
[58] Wachenfeld, W., Winner, H., 2017. The New Role of Road Testing for the Safety Validation of Automated Vehicles, in: Automated Driving. pp. 419–435. https://doi.org/10.1007/978-3-319-31895-0_17.
[59] Wang Y, Xing L, Wang H, Coit DW. System reliability modeling considering correlated probabilistic competing failures. IEEE Trans. Reliab. 2018;67:416–31. https://doi.org/10.1109/TR.2017.2716183.
[60] Wróbel K, Montewka J, Kujala P. Towards the development of a system-theoretic model for safety assessment of autonomous merchant vessels. Reliab. Eng. Syst. Saf. 2018;178:209–24. https://doi.org/10.1016/j.ress.2018.05.019.
[61] Xing L, Levitin G, Wang C, Dai Y. Reliability of systems subject to failures with dependent propagation effect. IEEE Trans. Syst. Man, Cybern. Part ASystems Humans 2013;43:277–90. https://doi.org/10.1109/TSMCA.2012.2197199.
[62] Zhang X, Khastgir S, Jennings P. Scenario Description Language for Automated Driving Systems: A Two Level Abstraction Approach. In: Proc. of the 2020 IEEE International Conference on Systems, Man and Cybernetics (SMC); 2020.
[63] Zio E. The Future of Risk Assessment. Reliab. Eng. Syst. Saf. 2018. https://doi.org/10.1016/j.ress.2018.04.020.