Safety Validation of Autonomous Vehicles using Assertion-based Oracles

Chris Harper, Greg Chance, Abanoub Ghobrial, Saquib Alam, Kerstin Eder, Tony Pipe

***Abstract*—Safety and mission performance validation of autonomous vehicles is a major challenge. We describe a methodology for constructing and applying assertion checks, operating in simulation or in the real world, through the use of an assertion-based oracle. An assertion library has been derived from the UK Highway Code (UKHC), as an example of a legal code of practice. Transformation of UKHC rules into assertions can be achieved either by direct translation or by physical modelling, to yield formal logical expressions that can be monitored automatically. The approach is illustrated on the example of assertion checking for vehicle overtaking, using a geospatial information system in an SQL database for validation and performance assessment. We present initial simulation and realtime video analysis experiments that apply assertions relevant in a vehicle overtaking scenario together with a brief analysis of the safety and mission performance characteristics measured.**

***Index Terms*—Verification and Validation (V&V), Autonomous Driving, Autonomous Vehicles, Simulation, Testing**

I. Introduction

Verification and validation are important to earning trust and gaining confidence in the safety of autonomous systems (AS) such as autonomous vehicles (AVs). Safety validation of any system can (and should) be performed by a variety of measures, including analysis and inspection of designs and their implementation. In this paper we focus on validation by system testing, and in particular on using assertions within simulation and at runtime for a vehicle overtaking scenario.

There has been some recent work to make autonomous vehicles abide by highway traffic laws by formalising the rules in a language that autonomous systems can understand. This is challenging not only because driving rules are often vague and may be conflicting, but also because of the contextual and common sense knowledge required to interpret these rules into appropriate driving actions. Field testing, often the traditional approach to validating systems in their target environments, is likely to be prohibitively expensive for autonomous systems, if a high degree of coverage or statistical confidence is required. This has led to much interest and research into the use of driving simulators[1] for validation.

We specifically focus on methodologies that incorporate the classification, measurement and evaluation of a system's situated behaviour. One such approach is based on the specification of situated behavioural properties as *assertions*, logical expressions that can be applied to monitor the behaviour of the system under test and to report any property violations. These properties relate to the observable driving behaviour of road vehicles. They characterise what any vehicle operator, autonomous or human, may be expected or legally required to achieve on public roads. An oracle is defined as a process or mechanism to check the output of a test [20] and, in the case of AV safety validation, this test is the process to determine compliance with the legal driving codes for human drivers. We use the UK Highway Code [44] (UKHC), as exemplar diving conduct and thus can be used to create validation assertions for properties such as safety and mission performance.

We demonstrate how to systematically transform these rules into assertions. To achieve this, the driving rules must be translated from human-readable format to machine-readable expressions that can be automatically monitored. Although we select the UKHC as the reference of our oracle for pragmatic reasons, it could equally be replaced with other metrics pertaining to functionality or even social convention. We review some related work in Section II, in which other potential sources for assertion-based oracles have been identified.

We illustrate the assertion-based oracle with two examples of lane changing during an overtaking scenario. We adopt the terms scene, scenario and situation from [45] throughout this paper. A geospatial information system in an SQL database was used for validation of the assertions and also for performance assessment of both simulation and the operational monitoring system. Note that the same assertions and assertion checkers can be used for either case.

Safety assurance is emerging as a legislative practice and an essential aspect of AV verification and certification, e.g. see recommendations set out in the recent UK Law commission report which specifically mentions evaluation of AV behaviour against the rules of the road [42]. Assertion monitoring may be used as evidence to show functional safety compliance against national regulations or codes of practice, the general safety

Manuscript received ...; revised ...; accepted.... Date of publication ...; date of current version ....

This research has in part been funded by the ROBOPILOT and CAPRI projects. Both projects are part-funded by the Centre for Connected and Autonomous Vehicles (CCAV), delivered in partnership with Innovate UK under grant numbers 103703 (CAPRI) and 103288 (ROBOPILOT). This research was also supported in part by the UKRI Trustworthy Autonomous Systems Node in Functionality under grant number EP/V026518/1. Also special thanks to Séverin Lemaignan.

Chris Harper (e-mail: chris.harper@brl.ac.uk), Saquib Alam (e-mail: saquib764@gmail.com), and Tony Pipe (e-mail: tony.pipe@brl.ac.uk) are with the Bristol Robotics Lab, T Block, University of the West of England, Frenchay, Coldharbour Ln, Bristol, BS34 8QZ, United Kingdom.

Greg Chance (e-mail: greg.chance@bristol.ac.uk), Abanoub Ghobrial (e-mail: abanoub.ghobrial@bristol.ac.uk), and Kerstin Eder (e-mail: kerstin.eder@bristol.ac.uk) are with the Trustworthy Systems Lab, Department of Computer Science, University of Bristol, Merchant Ventures Building, Woodland Road, Bristol, BS8 1UB, United Kingdom.

Digital Object Identifier ....

[1]Examples include https://www.rfpro.com and http://carla.org.

argument being that if an AV satisfies all the rules that may be legally expected of a human driver, then its behaviour is comparable to that of human drivers and therefore should be acceptable, or at least legal. It should be noted that acceptance may therefore be somewhat context-specific, against the code of practice for each country, and what passes under one legal code may not be acceptable in another. We anticipate that there may need to be extensive harmonization of national driving codes to avoid having to re-certify vehicles for each new national market and improve the cost-effectiveness of certification processes, but this is outside of the paper's scope.

The need to measure the compliance of an AV's behaviour with legal codes of practice may go beyond the initial design validation or certification stage as performed in simulation and (to some extent) road tests. It may become necessary to measure such compliance during vehicle operation, for example to establish that the vehicle's behaviour is correct (or otherwise) in the event of an incident that requires subsequent investigation. If an AV is involved in an incident, but it can be shown by assertion-based assessment during operation that it was adhering to all relevant legal standards of driving behaviour, then this may have an impact on any liability or insurance-based compensation that may be due as a consequence of the incident. A detailed discussion of the rationale and requirements for explainability [38] of AV decision-making in post-incident investigation is beyond the scope of this paper.

This paper makes the following contributions. We present a novel technique for deriving, systematically formalising and encoding assertions from the UKHC and applying them in two modes: (i) within a Geospatial Information System (GIS) database linked to a vehicle simulator, and (ii) within a real-time monitoring system during vehicle operation. In the first mode of use, we encode the UKHC assertions as SQL queries and run them within a PostgreSQL database with a PostGIS extension.[2] In this mode, the assertions can make use of any data stream available, passed into the database records. In the second mode of use, the principal requirement is that data about the AV, other road users and the adjacent environment wrt. the test vehicle is sufficient to evaluate the assertions during operation. The same PostgreSQL-based database as used in the simulator can therefore be installed as embedded software, but is receiving its data directly from vehicle sensors instead of a simulator. This data stream need not be of a high fidelity, but rather should have "just enough" detail [26] to allow the assertions to be evaluated with good confidence.

In the following, related work is reviewed in Section II. Section III explains the structure of the PostgreSQL database and how it integrates with the underlying simulator or runtime monitoring environment. Section IV discusses the methodology of deriving assertions from the UKHC, and the principles of using an assertion-based oracle both in simulations and for operational analysis. Section V introduces the overtaking scenario we have used to demonstrate the application of assertion-based oracles and describes two case studies to illustrate the two options, simulation-based testing and operational monitoring. The results from the case studies are presented and analysed. In Section VI we discuss some important aspects of assertion-based monitoring, which have emerged from the experimental work, in particular the use of assertions for performance monitoring as well as for safety, and the observation that in many assertions the intention of ego-vehicle agents may also need to be monitored to demonstrate compliance with certain driving rules. Finally, in Section VII we draw conclusions and suggest directions for future work.

[2]Refer to https://www.postgresql.org/ and http://postgis.net.

II. Related Work

Challenges exist to formalise human-defined traffic rules that can then be used as the ground truth for an oracle to verify the correctness of a vehicle's behaviour. Prakken et al. [33] discusses some of these issues and argue in favour of the need for formalisation of traffic rules. Rizaldi et al. [35] use the formalisation as a means to resolve accountability issues in an event of an accident. To the best of our knowledge, most work [35, 14, 36, 4] advocates the use of Higher-Order Logic (HOL) to codify a wide range of traffic rules into a formal language. The authors of [35, 14, 36, 4] use Linear Temporal Logics (LTL) to construct a set of equations for various manoeuvres to ensure road safety. To construct these equations, a formal model of each lane and lane marking is required. The authors of [36] used *lanelets* [5] to build these lane models. While many works use the German or Vienna convention [13] on road traffic, in [4] a road junction rule from the UKHC is used to demonstrate their methodology. Some researchers [21, 16] have used temporal logic queries to formulate the HOL, using a variety of query languages like SQL, OQL, RQL, etc.

DeCastro et al. [9] guide an agent to learn probabilistic models, via a learning-based algorithm, constrained by safety contracts which are in essence traffic rules. However, the major challenges for using learning algorithms are completeness and verification of the solutions. Such systems are essentially a black box, and it is extremely difficult to validate their correctness. Another major problem is what reward or cost function should be used to train the model. One approach can be to have an oracle providing rewards for correct behaviour, but this leads back to developing traditional assertion-based rules using the relevant highway code.

The Responsibility Sensitive Safety (RSS) framework developed by MobilEye [39, 24] proposes mathematical models for mimicking the human subjective decision making as an effort to provide safe driving behaviour by AVs. On the other hand, legal codes such as the UKHC often define the interactions required with particular features of their respective road environments (for example, junctions, roundabouts, or crossings). Our work is intended to address this latter domain, although a complete database will need to incorporate sets of assertions of both categories.

All the works discussed earlier are limited to detecting the occurrence of some undesirable events while driving. However, such events are not equal in severity [40]. Myers et al. [31] propose a framework to access autonomous driving systems using two types of scoring rules: prescriptive and

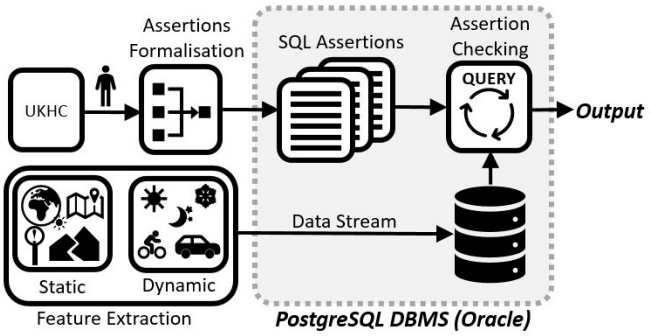


Fig. 1: Testbench for interpreting and monitoring assertions.

risk-based. To the best of our understanding, the prescriptive scoring mechanism captures the capability of the AV to follow traffic rules while the risk-based scoring system assesses the behaviour of the AV in rare events where undesirable outcomes are unavoidable, and it is expected of the AV to exhibit behaviour that would minimise the risk. Developing a generic system to score risks in different scenarios is challenging and requires a separate study of its own. Therefore, risk-based assessment is not part of this paper.

We use SQL to build the assertions, which is highly expressive [29] and can be integrated with powerful geospatial object handling technology (PostGIS). This provides a natural environment to write logical checks that can be performed on static and dynamic objects in a geographical context.

III. System Architecture for Assertion-based Monitoring

The evaluation of assertions requires an environment to capture or generate vehicle data, and apply the assertions to interpret the driving situation. This section presents the generic components and software architecture we have deployed to support assertion-based monitoring either in simulation or during operation.

A. Assertion Monitoring Testbench

Figure 1 shows the testbench architecture that was used to formalise and monitor assertions and interpret the diving situation. The testbench is used by starting with the reference used to assess the driving situation, in this case the UKHC. Following this, there is a human-interpreted stage (human icon) that converts human-readable rules into formal assertions (*Assertion Formalisation*) that are machine-readable. This forms one of two inputs to the *Oracle* which uses postgreSQL database technology to perform the automated assertion checking. The second input to the Oracle is the *Data Stream* of extracted features that is observing the driving scenario, which may be from simulation or from live sensor data. The *Feature Extraction* process will capture relevant information from the driving scenario, including *Static* elements such as the road network and *Dynamic* elements such as vehicle position and orientation. Records are captured from the *Data Stream* and stored on the PostgreSQL DBMS server at each available time step. Automated assertion checking happens within the DBMS, where *SQL Assertions* are stored which have been written to logically express the formalisations of the UKHC in the SQL language. At each time step the *Assertion Checking* process executes a query engine that evaluates the relevant assertions encoded with the assertion logic against the stored records and generates a pass or a fail verdict as an *Output*.

B. Geospatial Database for Assertion Checking

To develop assertions derived from UKHC rules, we need a formal language into which they can be translated for automatic checking. Two aspects are important to the choice. First, as we discuss later, the rules and guidelines in the UKHC are written for human readers not automatic systems. They consist of sets of discrete rules applicable to specific aspects of road user behaviour, which must be taken collectively, and not necessarily always in a particular order, to form the legally required profile of behaviour. This suggests that a declarative programming language, such as Prolog or SQL, may offer the required expressive power for encoding rules. Second, to process the data produced during driving, we seek languages that handle large data sets naturally, where the assertions constitute formal operations on datasets corresponding to the logical properties we extract from the UKHC rules. Taking these two requirements into consideration, SQL emerges as the most suitable programming language. SQL has been demonstrated formally [29] to be sufficiently expressive to be able to encode related problems such as state or graph reachability as well as all standard problems in relational algebra, albeit with the use of some of its later extensions such as recursive operators. Safety properties can be defined in terms of reachability [17, 30] or (in-)stability [19, 50] of desirable (goal) and undesirable (hazardous) situated states. For these reasons, we have selected SQL-based database technology as the framework for our assertion-based oracle.

The technical challenge then becomes how to compare the data captured against the assertions. We argue that a geospatial relational database is well suited for this purpose. The vehicle state can be easily stored and accessed. Additional insights, such as dynamic properties, can also be derived from this base information. Thus, we have implemented Assertion Checking as a PostgreSQL database with a PostGIS geospatial information system extension, which provides an extensive library of SQL-native spatial measurement functions. We chose these particular software products because they are open-source, and PostgreSQL implements powerful, high-performance table searching algorithms. Real applications will involve the use of large data sets, so the performance of the underlying database engine is an essential factor in practice.

The use of PostGIS introduces an extensive library of data types and method functions suited to measurement of basic physical (geometric) relationships. For the overtaking scenario in this paper, functions measuring distance between geometric objects (the bounding-box shapes of vehicles on the road), and partial or total containment of the vehicle within road lane geometries, are of particular use to assess vehicle positions during the overtaking manoeuvre. PostGIS functionality can also be used to generate more complex geometric dynamic

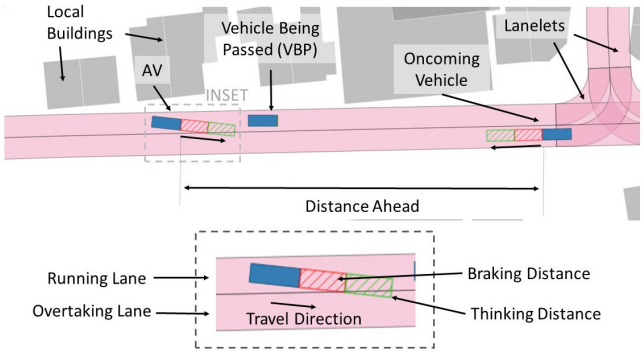


Fig. 2: Database view of the overtaking scenario.

data derived from the basic state of the vehicle or other agents. An example of this is shown in Figure 2, in which two polygons for *thinking distance* (inset, green hashed area) and *braking distance* (red hashed area), which are parameters defined in the UKHC (Rule 126), can be generated as a function of the basic forward velocity data of the vehicle. Figure 2 shows a projection of these derived geometries onto the associated street map. Thus, new abstractions can be developed by deriving new data sets from the original (captured) data. These can be manipulated effectively as a new variable (SQL attribute) appended to the original data set, and referenced by queries in the same way.

This is an essential principle of our methodology. Procedures and queries can be developed to generate abstractions directly related to the concepts expressed in the rules of the UKHC. Where such concepts are used in multiple rules, the corresponding abstractions form a library of common queries or operations, which can be incorporated into the DBMS.

To illustrate the functionality of the PostgreSQL DBMS server, Figure 2 shows the operating situation associated with the overtaking scenario described in Section V, and gives an example of a derived variable called *Safe Distance Ahead* (SDA), whose name is derived from terminology in the UKHC. The figure shows an urban scene depicting a section of road network divided into *lanelets* denoting legal division between driving lanes and any buildings in the area. In this example the AV is on the left of the figure overtaking a parked vehicle requiring a lane change. The distance to the oncoming vehicle is termed the Distance Ahead.

The assertion monitor performs a check (as defined in UKHC Rule 162) that a safe distance ahead exists between the AV and the nearest vehicle in an oncoming lane, labelled the 'Oncoming Vehicle' (OV) as the AV starts to overtake the 'Vehicle Being Passed' (VBP). The start time of the overtake is defined as the time that the AV first begins to enter the adjacent lane (Overtaking Lane) of the road, at which point it must commit to the manoeuvre or else abort and return to driving in the initial lane (the Running Lane).

In the case of assertions related to overtaking, it is necessary to determine whether the overtaking vehicle wholly or partly occupies a given road lane, since overtaking manoeuvres require it to move at least partially into an adjacent lane and back. Therefore, it is necessary for the database to include lane geometries (polygons), as shown in Figure 2, in order to allow checking of vehicle occupancy of lanes by testing for the overlap between vehicle and lane objects. The overtake manoeuvre ends when the vehicle is entirely within the running lane once more after having passed the VBP. Lane geometry objects (shape or bounding box polygons) are therefore an essential element of the assertions related to overtaking.

IV. Assertion Development and Application

In this section we first identify the types of assertions that may exist. We then discuss how to develop assertions from the natural language statements of the UKHC. We also present how to use assertions, both in simulation and during operation.

A. Types of Assertions

The assertions to encode UKHC rules tend to fall into one of four principal types. Except for invariants, assertions are defined with respect to some *assertion reference point* - a particular step of a captured data trace, defined by some reference condition which holds at a specified time step. The assertion is then defined as some spatio-temporal condition relative to the reference point, as shown in Figure 3.

The four types of assertions are:

- Invariant Condition: A condition that must be satisfied at all time steps within the captured data trace.
- Execution Condition: A condition that must be satisfied at the assertion reference point.
- Pre-condition: A spatio-temporal phenomenon that must exist in the steps preceding the assertion reference point.

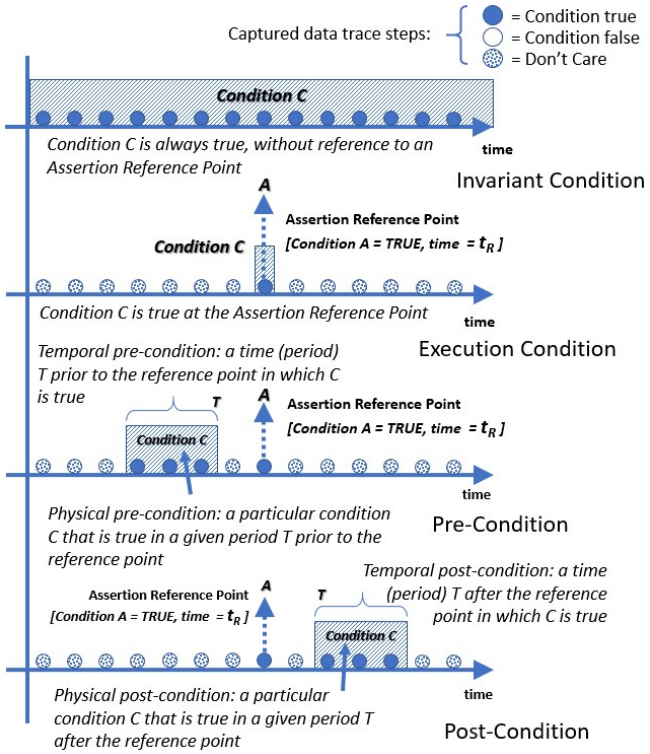


Fig. 3: Types of assertions.

- Post-condition: A spatio-temporal phenomenon that must exist in the steps following the assertion reference point.

Pre- and post-conditions can be one of two sub-types:

- *Temporal*: A time period preceding or following the assertion reference point, within which a specified condition holds. This is the temporal component of the overall spatio-temporal condition, e.g.:
  - *Temporal pre-condition*: as a vehicle crosses a highway exit lane boundary (reference point) checking that the vehicle's indicators have been set continuously in a preceding period (e.g. 5s) (i.e. checking for unduly late or intermittent indications).
  - *Temporal post-condition*: a highway exit assertion may check that if a vehicle starts indicating to leave at the exit lane, then it does enter the exit lane within a given time period (i.e. checking for unnecessarily early indications).
- *Physical*: A physical (e.g. spatial or state) condition that must be satisfied at a specified time(s) before or after the assertion reference point. This is the physical component of the overall spatio-temporal condition, e.g.:
  - *Physical pre-condition*: a check that, as it passes the exit lane boundary, a vehicle's indicators were set at the last highway exit sign before the exit (i.e. checking for omitted indications).
  - *Physical post-condition*: a check that if a vehicle is indicating to leave in the approach to an exit lane, then it does in fact leave the lane (i.e. this is a check for false or spurious indications).

B. Assertion Selection

Identifying which assertions are relevant in a given situation is important to manage the performance of assertion checking and to ensure the output is meaningful. For example, assertions for highway driving are not applicable in urban or rural road environments. Similarly, different assertions checking for mutually exclusive conditions will be guaranteed to generate errors unless the inappropriate assertions are filtered out. For example, checking that an AV is braking smoothly will inherently fail in a situation where it is required to accelerate. If assertions of this kind are not selected correctly, they will generate nuisance failure results.

Filtering assertions can also improve oracle throughput when they are applied in a time-critical mode of operation, e.g. during simulation (see Section IV-E). If the assertion type can be embedded in its script (for example, in its header), then the assertion checking process in the testbench (see Figure 1) can be set to check only assertions of a given type.

Discrimination of assertions can also be achieved by specifying 'trigger conditions' based on the type of *Operating Design Domain (ODD)* in which the assertion is relevant. [3]

[3]This is similar to typical practice in computer hardware verification [41], where an assertion may take the form "under <environment conditions> if <trigger> then <expectation>", where <*environment conditions*> may be "we are not in RESET mode" and <*trigger*> may be "we receive a request" and <*expectation*> may be "an acknowledge signal is driven high sometime in the X next cycles".

C. Human-Readable Driving Rules

The UKHC comprises several hundred rules and guidelines, applicable to different categories of road user (e.g. pedestrians, cyclists, cars, trucks and buses) and covering a number of different aspects of road use. The UKHC rules are written entirely for human readers, and seek to offer guidance in common sense terms that any reader should understand clearly. For example, two rules of central interest to this paper relate to overtaking manoeuvres:

Rule 162 **Before overtaking** you should make sure

- the road is sufficiently clear ahead
- road users are not beginning to overtake you
- there is a suitable gap in front of the road user you plan to overtake.

Rule 163 **Overtake only** when it is safe and legal to do so. You should:

- not get too close to the vehicle you intend to overtake
- use your mirrors, signal when it is safe to do so, take a quick sideways glance if necessary into the blind spot area and then start to move out
- not assume that you can simply follow a vehicle ahead which is overtaking; there may only be enough room for one vehicle
- *move quickly past the vehicle* you are overtaking, once you have started to overtake. *Allow plenty of room. Move back to the left as soon as you can but do not cut in*
- take extra care at night and in poor visibility when it is harder to judge speed and distance
- give way to oncoming vehicles before passing parked vehicles or other obstructions on your side of the road
- only overtake on the left if the vehicle in front is signalling to turn right and there is room to do so
- stay in your lane if traffic is moving slowly in queues. If the queue on your right is moving more slowly than you are, you may pass on the left
- give motorcyclists, cyclists and horse riders at least as much room as you would when overtaking a car (see Rules 211 to 215).

This example illustrates several characteristics found frequently in UKHC rules:

i Rules are often expressed in a second-person tense, offering advice directly to the reader. Converting them into assertions that are in effect performed (measured) from an external third-person perspective requires changes that may transform the logic of the rule to some extent.

ii The logical sense of the safety property expression (i.e. whether the rule expresses a pass or a fail condition) varies depending on the concept being expressed. Since the general guidance for direct conversion of rules is to keep as close as possible to the top level natural language, the logical sense of the assertions will follow suit. This contrasts with typical practice for their use in conventional computer programming, where the standard practice is to encode an assertion such that a pass condition permits a program to continue running normally, and a fail causes an exception to be triggered.

iii Being written at a natural human-readable level of abstraction, the rules and guidelines rely greatly on the reader having sufficient background knowledge and capability to resolve their generalized constraints to the point where actions can be selected that satisfy them. This abstraction also interferes with the ability to validate the assertions by any means other than manual design review; formalization of the assertions may require all the hidden complexity to be reintroduced.

iv Many clauses of UKHC rules offer advice to drivers about their internal decision making, often in the form of constraints. The third bullet-item of Rule 163 is a typical example. These clauses cannot be measured externally without some form of communication by the driving agent (for example, an AV) of the results of its decision-making processes. To date we have not attempted to establish assertions of this kind, as there are no standards or conventions by which an AV might explain and communicate its actions. However, we note that this is an interesting direction of future work (see Section VI).

D. Deriving Assertions

Deriving assertions from the human-readable UKHC requires that it be interpreted in a consistent manner. We present two approaches for developing assertions from the UKHC: direct translation into logic, and the use of modelling.

Method 1: Direct translation into logic

This approach involves converting the natural language text into a logical predicate. We developed a procedure containing the following steps:

i) Identify and extract a given UKHC rule clause.

- A separate assertion will be required for every distinct subject-verb-object clause in the text; a given UKHC rule could easily require on the order of 5-10 separate assertions to be fully covered;
- the clause must define a *testable* requirement - some UKHC rules are (at least at present) not testable. [We discuss this issue further in Section VI.]

ii) Write a natural language hypothesis that captures the safety property of the rule.

- A useful way to do this is to phrase the natural language hypothesis interrogatively, i.e. as a question.
- Many rules are written imperatively, as orders or advice for road users to follow. The hypothesis is straightforward: were the conditions of the instruction(s) satisfied?

iii) Write a logical statement that reflects accurately the assertion hypothesis

- Note that the perspective of the hypothesis may need to be transformed to a 'third-person' (external) viewpoint.
- Write the assertion logic as closely as possible to the language of the assertion hypothesis, keeping the same high-level terms.

iv) Verify assertion logic by manual design review.

- As previously mentioned, pragmatically this is the only method available. But if the assertion logic is directly comparable to the natural language of the assertion hypothesis, this step should be reasonably self-evident.

As an example of the direct translation method, consider the following clause from UKHC Rule 258. The procedure could be applied as follows:

i) UKHC rule clause:

> ***Red flashing lights.*** *If red lights flash on a signal and a red 'X' is showing, you MUST NOT drive in the lane shown as closed beyond the signal.*

ii) Assertion Hypothesis:

> The hypothesis question is: *Does the AV pass a 'Lane Closed' overhead gantry signal?*

iii) Assertion Statement (Predicate logic):

The assertion question concept of passing a sign then becomes a test of whether the AV bounding-box geometry overlaps with a boundary line between the posts of an overhead gantry object, which must lie across the roadway. This can be converted into an SQL query (SELECT statement) returning all time steps where the bounding box shape of a vehicle overlaps the line between the posts of an overhead gantry and the overhead sign of that gantry reads "Lane Closed" for the road lane occupied by the vehicle.

Key concepts of the assertion (for example, boundary lines of overhead gantry signals) become predicates in the assertion logic, and can be implemented as SQL procedural functions stored in the PostgreSQL database.

We anticipate that typically assertion queries will make use of PostGIS relational operators and functions for object geometries, whereas the library functions will typically use PostGIS geometry constructors to set up the geometric objects for the assertions as variables derived from the captured simulation data. A hierarchical organization of queries and procedural functions (from Assertions to CAV DBMS library functions to PostGIS functions to captured simulation data) emerges from this approach.

Method 2: Model-based Analysis

While many UKHC rules can be translated directly into logical conditions or constraints expressible as simple Boolean predicates, some describe more complex situations that may require more explicit modelling to identify the assertion hypothesis and logical expression. We require no particular constraints on the modelling methodology used, except that it needs to produce a well formed logical expression deriving the essential parameter of the assertion hypothesis from the data stream.

As a worked example, we present an assertion we developed from Rule 162, which we applied experimentally. The first two steps of this method are the same as for Method 1: identify the UKHC clause and the associated assertion hypothesis:

i) UKHC rule clause:

> *Before overtaking you should make sure the road is sufficiently clear ahead.*

ii) Assertion Hypothesis:

> The hypothesis question is:
> *Is the distance between the AV and OV sufficiently large to complete the overtake manoeuvre, at the*

moment the AV starts it (i.e. as it crosses the road centre line into the overtaking lane)?

While the UKHC rule clause seems straightforward, the natural language actually encapsulates a complex judgement that a driver must make at the moment they begin an overtake manoeuvre. The driver must judge whether the distance between their vehicle and any oncoming vehicle in the adjacent road lane is sufficient to allow the manoeuvre to be completed safely. This is contingent on the (relative) speeds of the driver's vehicle (the AV), VBP and OV, line-of-sight distances to limits of view or obscuring obstacles such as bends in the road, blind summits, or road-side buildings, as well as the style of manoeuvre that the driver intends to perform - an 'aggressive' overtake manoeuvre can probably be completed in less distance than a 'relaxed' one [9, 43].[4] There is a complex relationship between all the elements of this assertion that is quantitative in nature and hence an algebraic formula must be developed that expresses the assertion hypothesis. Such expressions must be developed by modelling the situation.

In the case of this Rule 162 example, it was necessary to develop a complete model of the overtaking manoeuvre in order to produce an expression for the *Safe Distance Ahead* (SDA), which calculates how far away the Oncoming Vehicle needs to be from the AV for an overtaking manoeuvre to be conducted without risk of collision. The analysis decomposed the overtaking manoeuvre into several stages: Pulling Out (into the oncoming lane), Passing the VBP, and Cutting In (to the original running lane). A formula was developed for the overall distance required to complete the manoeuvre without violating safety constraints, such as impingement upon vehicle thinking or braking distance (see Figure 2) or clearance distances around the VBP. A diagram of the full analysis model, and the derivation of the formula for Safe Distance Ahead, is provided in Appendix A.

The analysis identifies several key parameters that characterise the overtaking manoeuvre. These include:

- *Stopping distance* (SD) is defined as the sum of the 'thinking distance' and 'braking distance' of the vehicle (assuming it has a human driver). A table of the typical stopping distances for travelling speeds (v) between 20 mph - 70 mph is provided with Rule 126 in the UKHC. Equation 1 is a regression formula derived from this table to estimate the stopping distance (m) given the vehicle speed (mph). The coefficients are: $a = 0.300$, $b = 0.058$, $c = -0.011$ and $d = 0.015$.

$$\text{SD} = \underbrace{av}_{\substack{\text{Thinking}\\ \text{Distance}}} + \underbrace{b + cv + dv^2}_{\substack{\text{Braking}\\ \text{Distance}}} \tag{1}$$

- *Danger Space* (DS) is an area projected forward in the direction of travel with a length equal to the stopping distance and a width equal to that of the vehicle. The overall formula for SDA incorporates the DS of the OV. SD varies with vehicle speed, making the DS a function of speed, therefore intuitively as the speed increases the DS ahead of the vehicle will project further forward.
- *Pull-out Clearance* distance is the minimum safe separation that must be maintained as the AV pulls out into the oncoming lane (and carries its own constraints (and assertions) as defined in the first bullet-point of Rule 163.
- *Pull-out Angle* (β) is the steering angle taken by the AV as it moves into the oncoming lane.
- The *Cut-in Clearance* distance is the minimum required separation distance between AV and VBP when the AV begins to cut back into the running lane.
- *Cut-in Angle* (θ) is the steering angle taken during the cut-in move.

Taken as an ensemble, the last four parameters in the above list define a *driving profile* for the manoeuvre, which characterise a level of urgency or aggression associated with the manoeuvre. An 'aggressive' profile might be taken as one which has a low pull-out clearance, high pull-out angle, low cut-in clearance and high cut-in angle, indicating a 'sharp and close' manoeuvre taken by the AV around the VBP. A more 'relaxed' driving profile may have larger clearances and lower steering angles. Various profiles of manoeuvre with different characteristic values are assessed in the experimental work discussed later. It should be noted that other UKHC rules place (at least qualitative) restrictions on these parameters to ensure safe driving behaviour (see the fourth bullet-point of Rule 163.

As with the problem of direct translation of the UKHC rules, the approach aimed to develop a model using the concepts of the natural language of the rule rather than by other parameters, for example time-to-collision, which has been found in human factors studies to be the most significant parameter characterising overtaking manoeuvres [27, 7]. Hence the model developed for the Safe Distance Ahead measurement differs to many that have been developed for analysis of overtaking manoeuvres, which aim to capture the decision making of the driver rather than the characteristics of the legal code of behaviour that is essentially an independent observer's perspective on the scenario.

1) Geospatial Database for Assertion Checking: Analysis of the overtaking manoeuvre problem fits neatly into the geospatial database framework for assertion checking. The vehicles can be represented as shape objects using the PostGIS system extension which allows for logical tests and measurement metrics to be efficiently calculated between shapes. An Entity Relation Diagram (ERD), Figure 4, shows the variables within the database and the relationships between them [5].

For each log file there exists an environment table containing map information, which in this case is based on the openDrive[6] and lanelet formats. From the lanelet data we

[4]The internal plans and intentions of an AV may not be available to the external validation system, so the definition of the assertions may need to make default assumptions about its intentions, or make a 'worst case' test that checks against the most conservative case that can be assumed. For example, using the case of Overtaking, the assertion could assume that the AV was making a 'relaxed' overtake manoeuvre, in which case an assertion might register a safety violation for anything less than large separation distances between the AV and oncoming vehicle at the start of the manoeuvre, even though the manoeuvre could probably be completed safely if a more urgent profile were performed.

[5]See https://en.wikipedia.org/wiki/Entity-relationship_model for an explanation of *crow's foot* notation.

[6]https://www.asam.net/standards/detail/opendrive/

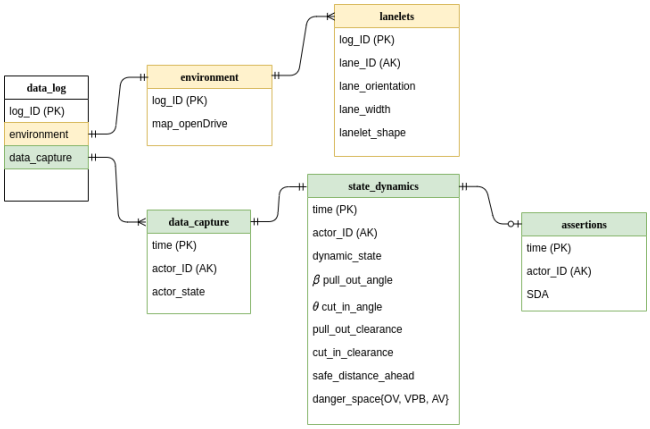


Fig. 4: PostgreSQL database Entity Relationship Diagram.

derive a table containing useful entries such as lane width and orientation for the environment. Data captured from the simulation (or real-time monitor), contains a state vector for the actors in $data\_capture.actor\_state$ which contains position and orientation information. From this captured data table we can calculate dynamic state information, that is information changing over time (e.g. velocity, acceleration) or with vehicle speed (e.g. braking distance).

Further insight to the driving situation can be derived when combining actor state information with information about the environment. For example, driving lane orientation ($lanelets.lane\_orientation$) can be used to calculate the vehicle heading angle relative to the driving lane, to infer the cut-in or pull-out angles ($state\_dynamics.pull\_out\_angle$) required for the Safe Distance Ahead calculation. The final $assertions$ table is where the Safe Distance Ahead output can be written and stored as a Boolean pass/fail result.

E. Modes of Use of the Assertion-checking Subsystem

Assertion-checking databases can be used as a test oracle in a variety of ways, both during simulation and in runtime monitoring:

Modes of use during simulation:

- *Simulation Control (Halting)*: Assertions can be checked as the simulation runs, and a simulation could be halted after a major safety violation if there is little value in continuing with a simulation beyond that point.
- *Operator Warnings*: As above, assertions can be checked as the simulation runs, but instead of halting the operator can be warned of the occurrence, allowing them to exercise their judgement on whether to continue.
  We anticipate that this may be the best policy for minor safety violations or events such as collisions which may lead to secondary effects like non-deterministic behaviour in the simulator [6], where the log file data might be degraded or unrepeatable following the event. The simulation may need to be halted, but pragmatically we believe this decision is probably best left for test engineers to decide. However, the use of assertions to provide operator warnings could be a beneficial mode of use.
- *Simulation Log Annotation*: for negligible safety assertions failures, or for non-safety checks such as service performance monitoring, assertions could be checked as the simulation runs. But, instead of halting or providing warnings, assertion failures could simply result in annotations being added to the simulation log data, indicating that an assertion failed at a given time step but otherwise having no other externally observable effect. The log can then be analysed retrospectively after the simulation run has finished.
- *Retrospective analysis*: Instead of checking assertions as the simulation runs, the captured data can be recorded and passed to the assertion checking database after simulation has finished, to analyse an AV's performance retrospectively. To date this is the mode in which we have been using the assertion-checking facility.

The simulation experiments with assertions presented in Section V were all run retrospectively, as they were a simple demonstration of concept rather than full implementation of the above modes of use.

Modes of use during runtime monitoring:

- *Warnings and Alarms*: Assertion checks can be used to raise warnings or alarms to human vehicle operators, to draw their attention or request them to take control over the vehicle if safety violations occur. Additionally these warnings can be used to indirectly assess the performance of subsystems during operation to provide development feedback in future versions of the system.
- *Triggering safety functions*: Detection of assertion violations can be used to initiate automated pre-planned risk mitigation plans as a safety function, where such occurrences present an immediate and sufficiently high risk that such a reaction is warranted [28]. Having a ground truth reference to judge on monitored data at runtime can be challenging. Assertion failures can provide a reliable verdict to execute appropriate risk mitigation plans.
- *Self-adaptation*: using assertion checking during runtime can be combined with measurements of key and safety performance indicators (KPIs, SPIs) [25] or other scoring functions to help the AV to actively learn and reliably adapt their behaviour during operation [22].
- *Accident investigation or insurance risk assessment*: Assertion-checking subsystems can be used to annotate the internal data logs of an AV to help identify potential causes of an accident in any post-incident investigation (similar to flight data recorders in aircraft), or to measure the occurrence rate of UKHC driving rule infractions for insurance risk adjustment purposes. However, there are ethical concerns associated with such applications, as discussed in [48].

V. Experimental Case Studies

As an aid to developing the methods and techniques described previously, we present worked examples of assertions and measurements applied both to simulation data and real-time video footage. The experiments were early stages of work

in developing the methodology, and were not developed by all the methods described in this paper, nor possess all the features described herein, but they did guide the development of the methodology and we believe that they do prove the general concept and utility of the approach. The use of methods such as modelling, and the need for features such as the use of standard procedures/library functions, all emerged as direct consequences of these experiments.

We chose two case studies inspired from a challenging overtaking scenario; the first aims at demonstrating the use of assertions in simulation testing, and the second shows how assertions can be used as SPIs to assess the safety performance of an AV or other road user at runtime.

A. Experimental Scenario

To demonstrate the functionality of the assertion based safety validation process, we chose to consider an overtaking scenario similar to Scenario 6 of the driving situations in the NHTSA[7] pre-crash typology [46] that were published in the CARLA Challenge[8] The corresponding UKHC rules [44] that apply to this scenario are Rules 162-169. Specifically, the scenario involves overtaking on a single-lane road where the test vehicle must perform a lane change to avoid a parked or broken down vehicle. The vehicle must leave sufficient gap to the vehicle it is overtaking and also not endanger any vehicles that are oncoming.

In the simulation and the real-time video analysis experiments, we chose to assess clauses in two different UKHC overtaking rules. The simulation study developed an assertion for the first bullet-item of Rule 162 (as discussed in Section IV-C), developing an assertion check to determine whether a Safe Distance Ahead existed for an AV performing an overtake manoeuvre on a single-lane road, as it began to cross the road centre line at the start of the manoeuvre. The real-time safety validation study developed video analysis measurements that would support assertion checks against the fourth bullet-item of Rule 163 (see the italicised clause in Section IV-C), while the measurements were not developed fully into assertions in SQL code, the study does capture the information essential to the assertion hypothesis for that UKHC clause (the Cut-in Clearance distance as discussed in Section IV-D), and would be generated as a derived variable in the database using a standard procedure function as discussed previously.

B. Simulation case study

In the simulation case study we developed an assertion check for Safe Distance Ahead at the start of overtaking, using the expression for SDA discussed earlier in Section IV-D. We ran the simulation using a real road map segment (of a road in Bristol, UK, part of an AV test route being considered for the ROBOPILOT project[9]).

1) Environment model: The map segment used for the simulation case study is shown in Figure 2. For the sake of simplicity we ran the simulation on one of the straight sections of road, which avoided having to take considerations such as obscuring objects or limited lines of sight into account. The road layout model was developed using the Road Runner software package, from an OpenDrive environmental model developed from Open Street Map (OSM) data obtainable freely from UK government sources (Ordnance Survey).[10]

As discussed in Section III-B, lane polygons are required for efficient measurement of vehicle lateral movement within, and partial occupancy of, road lanes; in this simulator case study we used a lane partitioning concept termed *lanelets* [5], which allows exactly this level of description of the scene. We used an open-source software conversion package [3] to generate the lanelet data from the openDrive [12] model, dividing the roads into individual lanes and lanelets. This data was imported into the PostgreSQL database as a table of lane segment polygon objects ($lanelets.lanelet\_shape$ in 4) covering the whole road layout used for the study. Additionally, information about lane boundaries and orientations, required to identify road orientation and running lane directions, was appended to this data. Figure 2 shows an example of the lanelet shape object overlaid onto a road network. The blue boxes represent the dynamic actors, which in our case are the vehicles involved in the overtaking scenario. All of the above world map information was pre-loaded statically into the database, for use by the SQL query engine running the assertions queries.

2) Overtaking Profiles: The intent of the Safe Distance Ahead assertion we developed was to assess whether sufficient distance ahead of the simulated AV was available at the initial decision point of the manoeuvre for it to complete an overtaking manoeuvre safely. However, in practice, sufficiency of the distance ahead is dependent in part on the driving profile that the AV controller has selected to perform (see Section IV-D). But, if the safety validation exercise is to be performed as an independent 'black-box' measurement of AV behaviour (as is often the case in many practical development projects), the internal plans of the AV may be unavailable to the system testers, and we conducted our experiment in such a manner as to reflect such a situation.

The simulated AV in the experiment was programmed to follow a preset trajectory, which was not used to influence the choice of parameters in the Safe Distance Ahead assertion expression. We developed three different variations of the SDA assertion, each calculating the safe distance ahead using the formula developed from the modelling analysis but which assumed three different driving profiles corresponding to three different levels of urgency of overtake manoeuvre (called 'relaxed', 'hurried' and 'aggressive'). The levels of urgency were based loosely on similar concepts developed in other Human Factors research [7, 47], which had equated level or degree of urgency with *Time To Collision* (TTC). These sources identified that a TTC of approximately two seconds corresponds approximately to a medium degree of

[7]https://www.nhtsa.gov/

[8]https://carlachallenge.org/

[9]See URL: http://www.bristol.ac.uk/tsl

[10]https://os.openstreetmap.org/

urgency, which we have labelled 'hurried'. These three variants of the SDA assertion were then applied to the simulation data produced by the preset AV controller to see whether which SDA situation that occurred in each simulation run was acceptable for which driving profile.

3) Simulator system details: The simulator used for the case study was a minimal version of the generic simulator testbench architecture described in Section III-A. The system used the CARLA simulation software [11], an open-source autonomous driving simulation environment. Dynamic state data generated by the simulator was passed to a PostgreSQL database as shown in Figure 1, with the SQL assertion script being applied in retrospective analysis mode as discussed in Section IV-E.

The paths followed by each simulated vehicle (AV, VBP, OV - see Figure 2) were preset to follow a series of way-points. Using fixed vehicle paths facilitates evaluation of the assertion, as it made the test repeatable[11] and allowed us to see how different characteristic parameters affected the assertion checking (see Section V-B5 for details).

CARLA generates dynamic state information about the simulation actors, which is made available via an API, and Python scripts were written to transfer data from the simulator into the PostgreSQL database on a step-by-step basis.

4) Simulation runs: Three simulation runs were executed to generate simulation data traces for assessment by assertion checking. In each case, the trajectory of the AV remained unchanged, overtaking the stationary VBP and cutting back into the running lane with the same pull-out and cut-in clearance distances each time. The OV trajectory simply moved along the oncoming lane at constant speed, but the starting position of the Oncoming Vehicle was varied so as to generate three different scenarios of i) a safe overtake manoeuvre, ii) a near miss and iii) a dangerous overtake manoeuvre resulting in a collision. The simulation data was logged and transferred into the assertion database as three distinct subsets of samples within the main captured data tables. These were then analysed retrospectively after the end of the simulation run, using assertions designed to assess whether (and how well) the AV performed overtake manoeuvres of differing profiles (as discussed in Section IV-D). These results are discussed below.

5) Simulation Assertion Checking Results: In this section we discuss the outcomes of the simulation safety validation analysis, and the implications on the methodology of developing and using assertions.

The three simulation runs described previously produced simulation data logs that were subjected to assertion checking to ensure that a safe distance ahead existed as the AV began to cross the road centre line into the oncoming lane at the start of the overtake manoeuvre. However, there are several factors conditioning the test results that need to be taken into account. Comparison of the three variations of test against the simulated scenarios (safe, near miss and collision) produces nine distinct tests.

The fixed overtaking manoeuvre trajectory that had been pre-set for the AV vehicle approximated to a medium level of urgency, in terms of the pull-out and cut-in clearance distances achieved (about 2s time to collision at the simulated speeds of the vehicles). This meant that for a set of tests having a range of urgency levels centred on the medium urgency case, one should get a sliding scale of test case passes and failures as the test case variants are applied to decreasing initial separation distances between AV and OV in the three simulation logs. The results of the tests are provided in Table I, which shows the required separation distances at the start of each manoeuvre (determined by the modelling analysis formula described in Section IV-D), the Distance Ahead that was achieved in each simulation run, and thence the pass/fail assertion results of comparing the two distances in each case.

Scenario	**Achieved Distance Ahead (m)**	**Distance Ahead required by profile (m)** 'Relaxed' 101.39	'Hurried' 63.73	'Aggressive' 40.02
Safe	76.43	FAIL	PASS	PASS
Near miss	58.33	FAIL	FAIL	PASS
Collision	35.63	FAIL	FAIL	FAIL

TABLE I: Pass/Fail Assertion results.

The table shows that it would have been inadvisable to perform a 'relaxed' overtake in any of the three scenarios presented, as this would have resulted in a near miss or a collision. If a 'hurried' overtake is attempted (which actually corresponds to the behaviour of the AV in the scenarios), this policy was acceptable for the 'Safe' scenario, but inadvisable for the 'Near Miss' and 'Collision' scenarios (as their descriptive names imply). And if the AV had been programmed to overtake aggressively in these scenarios, then only in the case of the Collision scenario circumstances would the Distance Ahead have been short enough for the manoeuvre to be inadvisable; in the other two cases the overtake could have been achieved successfully (although aggressive overtaking may result in other safety assertions being violated).

C. Runtime case study

In the runtime case study we developed an assertion monitoring for checking safe stopping distance during the overtake. To demonstrate the use of assertions at runtime monitoring of an AV, in the absence of a fully functioning AV controller, we used manual video analysis in combination with a pre-trained object classifier to extract data from a dashboard video camera of a dangerous overtaking vehicle.[12] The aim of the assertions applied to runtime data was to assess whether the AV risks colliding with any of the other vehicles involved in the scenario during the overtake and in turn take an informative decision on aborting or continuing the overtake. Such situations are very likely to happen if road layouts are more complex, vehicles

[11]Test repeatability is needed to ensure trust in the simulated assertion results and also to ensure software bugs are found and fixed efficiently. Repeatability relies on deterministic simulation code and hardware. If the *simulation variance* is within tolerance then verification coverage results are stable and if an assertion fails the test can be re-run to provide the same output as described in [6].

[12]Link to video used in analysis: https://youtu.be/Gzi4X7WZkRI.

change speed or if for any other reason the SDA calculation at the start of the manoeuvre gives misleading estimates.

1) Danger Space (DS) Assessment: The overtake manoeuvre can be split into three distinct stages as was described in the modelling analysis (Section IV-D): pulling out, passing VBP and cutting in. During the different stages of the overtake, non of the vehicles should be present in the DS of another vehicle nor should the DS of one of the vehicles overlap with the DS of another. To satisfy this requirement, six assertions are needed to be checked at the different stages of the overtake. The DS subscript refers to the vehicle danger space.

The first four assertions are applicable through out the three stages of the overtake, these are:
i) the VBP should not be in the AV danger space at anytime during the overtake,
ii) the OV should not be in the DS_{AV},
iii) the AV should not be in the DS_{OV},
iv) there should be no intersection between the DS_{AV} and DS_{OV}.

There are two key SPIs applicable for the last stage of the overtake (cutting in):
v) the AV should not enter the DS_{VBP} and
vi) there should be no intersection between the DS_{AV} and the DS_{VBP}.

2) Video analysis: We used the following approach to approximate the processing of the real-time data stream for the runtime case study. First, a pre-trained object classifier, YOLOv3 [34], was used to provide locations of vehicles in each frame and how much area each vehicle occupies of the frame. The separation distances between the AV and the other vehicles are the SPIs at the various stages of the overtaking manoeuvre. The longitudinal distance (s) of a vehicle along the road from the AV can be estimated using Equation 2:

$$s = \frac{c \times W}{w} \tag{2}$$

where c is a constant capturing the focal length of the camera and is found empirically, W is the actual width of a vehicle in meters, and w is the width of the box detected by YOLOv3 in pixels. Second, the lateral distance d of the AV from the middle white line is estimated using Equation 3:

$$d = \frac{w_{L(real)}}{w_{L(pixels)}} \times d_{pixels} \tag{3}$$

where $w_{L(real)}$ is the real lane width measured in meters, $w_{L(pixels)}$ is the lane width measured in pixels and d_{pixels} is the lateral distance of the AV from the middle white line in pixels. d_{pixels} is determined by simply finding where the middle white line is compared to the middle of the frame. The VBP and OV are assumed to have a fixed lateral distance from the white line separating lanes. Third, the pull out β and cut in θ angles of the AV are calculated using the change in lateral distance and change in the longitudinal distance measured. Fourth, as we were not able to measure vehicle speed directly from the video, we based the DS calculation on the worst case scenario; that is all vehicles were traveling at the speed limit of the road. The speed limit of the single carriage way in the video was found to be the national speed limit. According to Rule 124 in the UKHC the national speed limit for cars (AV and OV) is 60mph, and 50mph for goods vehicles (VBP). Using the above steps the real time video was converted to a 2D driving scenario, where the danger space assertions listed in section V-C1 can be checked.

	Overtaking manoeuvre stages		
Assertion	**Pulling out**	**Passing VBP**	**Cutting in**
VBP not in DS_{AV}	PASS	PASS	FAIL
OV not in DS_{AV}	PASS	FAIL	FAIL
AV not in DS_{OV}	PASS	FAIL	FAIL
No $\text{DS}_{AV} \cap \text{DS}_{OV}$	PASS	FAIL	FAIL
AV not in DS_{VBP}	N/A	N/A	PASS
No $\text{DS}_{AV} \cap \text{DS}_{VBP}$	N/A	N/A	FAIL

TABLE II: Pass/Fail danger space assertion checks results.

3) Runtime Validation Results: Table II shows the results for the six assertions introduced in section V-C1 at the three different stages of the overtake. At pulling out, the assertion checks suggest that it was safe to continue the overtake manoeuvre as non of the vehicles were present in another vehicles danger space and there was no overlap in any of the danger spaces. When passing the VBP the assertion checks indicate that the overtake should have been aborted or if it was aborted then it should have been aborted earlier during passing the VBP stage as both the AV and the OV have entered each others danger spaces. Finally at the last stage of the overtake, the assertion checks suggest that the AV has attempted to cut in before it has completely overtaken the VBP, whilst still failing the assertions relating to the OV. However, the AV has passed the assertion of not being in the DS_{VBP}, which suggests, given all of the other assertions have failed, that it has not overtaken the VBP before cutting in.

VI. Discussion

A. Assertion Checking to measure Performance

While we have applied the assertion checking methodology to measure the safety properties of vehicles in simulation and at runtime, the methodology can also be extended to validate other important system properties. This may include performance based metrics such as smooth driving (minimal jerk), accuracy of control (such as parking or stopping precisely, e.g. at a bus stop), and decisive behaviour at junctions or roundabouts, sometimes termed *liveness* [23]. In particular, the mission performance of a vehicle, for example, can be measured in journey time or efficiency of movement through traffic, which may affect not only passenger expectations but also the commercial viability of such an autonomous system.

Performance (achievement of task goals) and safety (avoidance of task hazards) are distinct behavioural properties of a system; achievement of one does not necessarily entail the other. A task controller must achieve satisfactory compliance with both properties if a dependable system is to be developed. In some scenarios, such as during overtaking, their may exist a competition within the controller to satisfy both the achievement of goals and the avoidance of hazards.

For example, an autonomous taxi may need to perform a minimum number of fare-paying journeys in a day in order

Distance Ahead Condition	**AV Does Not Overtake**	**AV Overtakes**
Case 1	safe	unsafe (assertion failure)
Case 2	safe but not taking opportunities	safe and taking opportunities
Case 3	safe but incurring significant delay	safe & expected behaviour

TABLE III: Performance classifications for deciding to overtake based on the distance ahead to the Oncoming Vehicle.

to remain commercially viable. Performance assertions can measure whether suitable opportunities for manoeuvres such as overtaking were taken efficiently, without incurring delays or driving sub-optimally for the given road network i.e. using all available space. However, it must still drive safely as it conducts each journey, so in situations that may induce journey delays, such as high traffic congestion, optimum performance and safety may not be mutually achievable. Assertions can measure how the AV trades off these two properties in the execution of its task(s).

An analysis for overtaking performance can be observed in the simulation results presented in Section V-B by comparing the required SDA to the actual distance ahead of the Oncoming Vehicle as listed in Table I. Considering this example from the point of view of performance, the AV must not violate the safety property but must overtake within a reasonable window of opportunity (δ), i.e. taking any presented opportunity to overtake if safe to do so and if appropriate to the road network.

The size of this *opportunity window* will be dependent on a myriad of different contextual and preferential conditions such as the driving style, vehicle capability, passenger comfort, etc.

We analyse the overtaking scenario in Figure 2, considering performance as well as safety, using the required SDA and opportunity window to define three specific cases:

- Case 1: $d_{OV} < SDA$
- Case 2: $SDA \leq d_{OV} \leq SDA + \delta$
- Case 3: $d_{OV} > SDA + \delta$

Here, d_{OV} is the distance to the Oncoming Vehicle, SDA is the required Safe Distance Ahead based on the model given in Appendix VIII and δ is the optimal window of opportunity for the vehicle to perform the manoeuvre. In each of these cases the AV could decide to overtake or hold back and the performance outcomes for each of these are shown in Table III. For case 1, the distance to the Oncoming Vehicle is less than SDA, so the only safe action is not to overtake and indeed to attempt to do so in this condition would result in failure of the SDA safety assertion. Safe driving behaviour whilst making timely use of the overtaking opportunities presented to the vehicle will see the vehicle progress through a road network most efficiently. This sentiment is captured in case 2, where the vehicle is taking opportunities to overtake whilst maintaining a safe distance to the oncoming vehicle. Deciding not to overtake in case 2 would not take advantage of opportunities to progress through traffic and will incur journey time penalties. In case 3, the distance to the Oncoming Vehicle is greater than SDA plus the window of opportunity, where the vehicle has unhindered opportunity, i.e. no currently oncoming traffic. In this case not overtaking (although completely safe) will incur significant and unnecessary delay to the journey and can be considered lower performance. Overtaking in case 3 would be the expected behaviour of the AV.

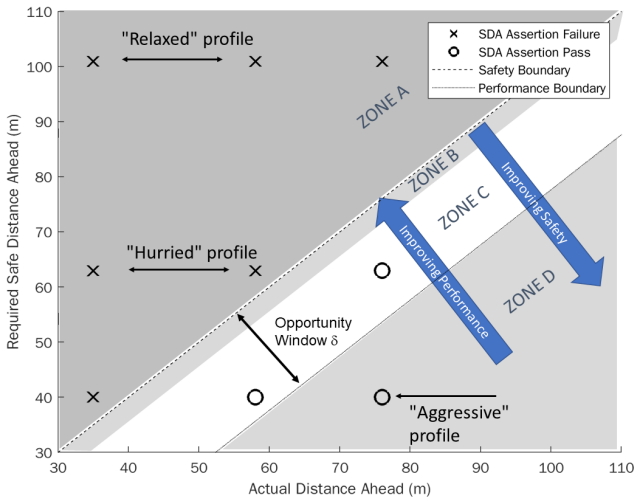


Fig. 5: Simulation results for performance.

The previous simulation results from the perspective of performance are shown in Figure 5. Here δ is set to be equivalent to TTC=$2.5s$ [37, 7, 47] for the purposes of illustrating this concept. The required SDA is shown as a function of the actual distance ahead for the three overtaking profiles. Test cases that fail the safety assertion are shown as cross markers, all being within *Zone A* which is an area where the SDA assertion will be violated. At the edge of this area, defining unsafe driving behaviour, is a *safety boundary* (dashed line) being the edge of Zone A. Beyond the safety boundary exists the opportunity window that extends to the limit of the *performance boundary* (dotted line), and covers *Zone B* and *Zone C*. Zone B represents a conceptual *safety margin* or *buffer* between optimal performance and the safety boundary, and in reality this may be the limit of a near-miss condition, e.g. TTC=$0.5s$. Reward should be given for an adaptive controller that makes use of Zone C but does not enter Zone B, i.e. executes optimal overtaking given opportunity δ but does not cause a near-miss event. *Zone D* can be considered safer than the other zones, i.e. the vehicle is giving lots of distance to the Oncoming Vehicle, but may be unnecessarily incurring delays and hence considered lower performance. An improvement could be made by moving from Zone D to Zone C without violating any safety property.

There is therefore a trade-off between safety and performance for optimal driving behaviour, where performance is improved by moving towards Zone B (but not passing the safety boundary), and safety is improved by moving towards Zone D (see large arrows in Figure 5). While we have observed this conceptual trade-off between safety and performance in our analysis, we also see evidence of this in the UKHC rules themselves. An example of this is exemplified in the 4th bullet point in Rule 163 (see Section IV-C) where the overtaking rule states “Move quickly past the vehicle” and

"Move back to the left as soon as you can...", which clearly suggests that performance will play a part in correct driving behaviour. There is also evidence of this trade-off between performance and safety, while the AV should complete the overtake maneuver "as soon as you can" (higher performance) but "do not cut in" (higher safety) we see these competing demands resolve in optimal driving behaviour analogous to that described by Zone C in Figure 5. This analysis considers only the overtaking scenario and in reality there will be many competing demands on the decision making process of a real controller.

B. Agent Intentions during Assertion Checking

The initial work presented in this paper has raised a number of consequent issues that require further investigation. One significant topic is the extent to which assertions require access to states of internal decision making within the autonomous vehicle being monitored. Many UKHC rules are advisories to potentially hazardous situations, and to actively perceive, deliberate, and take special or particular action to avoid harmful events. An excerpt from Rule 146 of the UKHC states

> *Try to anticipate what pedestrians and cyclists might do. If pedestrians, particularly children, are looking the other way, they may step out into the road without seeing you.*

which specifically asks to perceive and anticipate future developments of the situation. If we want to validate the correctness of an AV's behaviour in respect of these rules we must have the capability to observe the decisions made by the AV or its control system as they are made.

This case highlights the need for *explainable AI*, a topic gaining increasing attention in the AI, robotics and autonomous systems community. The opinion is widely held [10, 32, 49] that it may be essential for any such system employed in a safety related application to be able to explain its decision making. In many AI technologies the complexity and opacity of internal representations of perceptions, decisions, etc., combined with the non-deterministic nature of complex operational situations in real world environments means that they may not be repeatable in post-incident analysis without the system being able to report the precise situation that it perceived, and the planning or action-selection decisions that it made as a consequence. This requires some form of communication protocol or interface from the decision-making centres of the system to the outside world, so that the decisions can be observed and recorded for later analysis. This is a significant motivation behind the use of an assertion-based oracle subsystem as a runtime analysis facility, discussed in Section IV-E.

At present, we are not in a position to perform assertion checking for this kind of safety property as an independent validation exercise . There is no externally objective way to observe whether an AV agent is actually performing deliberative reasoning, planning or action selection consistent with rules such as Rule 146, unless it communicates explicitly its planning and decision-making through an interface, whose information can be read by the oracle for checking. A standard protocol for reporting decision-making information would need to be developed and at time of writing no such standard for doing this has been agreed. And given the commercially sensitive nature of such information, we do not anticipate that manufacturers would be willing to design systems that volunteer such information without legal requirements to do so and a standard protocol to use.

In the absence of any such explanation capabilities or facilities, it is still possible to achieve some degree of validation by performing checks derived from the text of the rule, which attempt to challenge the hypothesis expressed or implied by the UKHC rule or rule clause (*e.g. falsification testing*) of the safety property. For example, in the testing phase of an AV, to exercise the last bullet item of Rule 146 above one could construct a test case, perhaps similar in nature to those defined in [15], where simulated 'children agents' (that model the physical appearance of children as well as some aspects of their behaviour) step out into a road in front of the AV agent, having appeared either to be looking away from an AV agent, towards it, or some mix of the two focuses of attention. Such a test suite could then test the capability of the AV control system to evaluate the potential behaviour of children to determine what driving action can be taken successfully and safely. This can test the 'end-to-end' situated behaviour of the vehicle and thereby evaluate the cumulative effect of decision making, but as evidence to support a safety case this is less effective than explicit confirmation (by recording) of the AV's decision process that would come from Explainable AI.

VII. Conclusion

The results obtained from both the simulation and the runtime video analysis case studies show the general feasibility and utility of assertion-based safety validation testing, with the assertions being derived from regulatory documentation written in natural language and intended for human drivers rather than written specifically to be usable by computer. The results provide our first indication that human-oriented natural language rules of driving are an effective oracle for assessment of artificial systems.

The case studies were carried out at an early stage of development so we did not employ every element of the methodology that has emerged. Nevertheless, we believe that enough of the process has been demonstrated to show that the full methodology is viable. We also present the idea of using UKHC-based assertion checking methodology as a method to infer autonomous vehicle performance and have demonstrated this with an overtaking scenario using simulation results. We have demonstrated that a common UKHC assertion-based oracle is equally usable in runtime monitoring as it is in simulation.

We are considering numerous other directions in which to continue this work such as extending the sources of information for safe driving, e.g. German driving regulations [35], and models such as the RSS model [39, 24]. We are also researching the use of the testbench to perform falsification testing [8, 1] and *situation coverage testing* [2] and how other techniques such as *Environmental Survey Hazard Analysis* [18]

can be used to support the tool chain framework. We also aim to investigate assertion checking to provide reward function scores as a critic in reinforcement learning for continuous improvement of the AVs balance between safety and liveness during operation.

VIII. Acknowledgments

This research has in part been funded by the ROBOPILOT and CAPRI projects. Both projects are part-funded by the Centre for Connected and Autonomous Vehicles (CCAV), delivered in partnership with Innovate UK under grant numbers 103703 (CAPRI) and 103288 (ROBOPILOT). This research was also supported in part by the UKRI Trustworthy Autonomous Systems Node in Functionality under grant number EP/V026518/1. Also special thanks to Séverin Lemaignan.

References

[1] T. Akazaki, Y. Kumazawa, and I. Hasuo. "Causality-Aided Falsification". In: *Electronic Proceedings in Theoretical Computer Science* 257 (Sept. 2017), pp. 3–18. URL: http://dx.doi.org/10.4204/EPTCS.257.2.

[2] R. Alexander, H. R. Hawkins, and A. J. Rae. "Situation coverage–a coverage criterion for testing autonomous robots". In: (2015).

[3] M. Althoff, S. Urban, and M. Koschi. "Automatic conversion of road networks from opendrive to lanelets". In: *2018 IEEE International Conference on Service Operations and Logistics, and Informatics SOLI*. IEEE. 2018, pp. 157–162.

[4] G. Alves, L. Dennis, et al. "Formalisation of the Rules of the Road for embedding into an Autonomous Vehicle Agent". In: (2018).

[5] P. Bender, J. Ziegler, and C. Stiller. "Lanelets: Efficient map representation for autonomous driving". In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE. 2014, pp. 420–425.

[6] G. Chance et al. "On Determinism of Game Engines used for Simulation-based Autonomous Vehicle Verification". In: *arXiv preprint arXiv:2104.06262* (2021).

[7] R. Chen, K. D. Kusano, and H. C. Gabler. "Driver behavior during overtaking maneuvers from the 100-car naturalistic driving study". In: *Traffic injury prevention* 16.sup2 (2015), S176–S181.

[8] A. Corso et al. "A survey of algorithms for black-box safety validation". In: *arXiv preprint arXiv:2005.02979* (2020).

[9] J. DeCastro et al. "Counterexample-guided safety contracts for autonomous driving". In: *International Workshop on the Algorithmic Foundations of Robotics*. Springer. 2018, pp. 939–955.

[10] A. Deeks. "The judicial demand for explainable artificial intelligence". In: *Columbia Law Review* 119.7 (2019), pp. 1829–1850.

[11] A. Dosovitskiy et al. "CARLA: An open urban driving simulator". In: *Conference on robot learning*. PMLR. 2017, pp. 1–16.

[12] M. Dupuis, M. Strobl, and H. Grezlikowski. "OpenDRIVE 2010 and Beyond–Status and Future of the de facto Standard for the Description of Road Networks". In: *Proc. of the Driving Simulation Conference Europe*. 2010, pp. 231–242.

[13] Economic Commission for Europe Inland Transport Committee. *Convention On Road Traffic*. 1968.

[14] K. Esterle, L. Gressenbuch, and A. Knoll. "Formalizing traffic rules for machine interpretability". In: *2020 IEEE 3rd Connected and Automated Vehicles Symposium CAVS*. IEEE. 2020, pp. 1–7.

[15] European New Car Assessment Programme. *Euro NCAP Vulnerable Road User VRU Test Protocol, v-3.0.3*. Accessed: 2021-10-20.

[16] M. Gueffaz, S. Rampacek, and C. Nicolle. "Temporal logic to query semantic graphs using the model checking method". In: *Journal of Software* 7.7 (2012), http–www.

[17] J. Guiochet. "Trusting robots: Contributions to dependable autonomous collaborative robotic systems". PhD thesis. Université de Toulouse 3 Paul Sabatier, 2015.

[18] C. Harper and P. Caleb-Solly. "Towards an Ontological Framework for Environmental Survey Hazard Analysis of Autonomous Systems." In: *SafeAI AAAI*. 2021.

[19] C. J. Harper and A. F. Winfield. "A methodology for provably stable behaviour-based intelligent control". In: *Robotics and Autonomous Systems* 54.1 (2006), pp. 52–73.

[20] W. E. Howden. "Theoretical and empirical studies of program testing". In: *IEEE Transactions on Software Engineering* 4 (1978), pp. 293–298.

[21] S. Huang and R. Cleaveland. "Temporal-logic query checking over finite data streams". In: *International Conference on Formal Methods for Industrial Critical Systems*. Springer. 2020, pp. 252–271.

[22] D. Kang et al. "Model assertions for monitoring and improving ML models". In: *arXiv preprint arXiv:2003.01668* (2020).

[23] K.-D. Kim and P. R. Kumar. "An MPC-based approach to provable system-wide safety and liveness of autonomous ground traffic". In: *IEEE Transactions on Automatic Control* 59.12 (2014), pp. 3341–3356.

[24] P. Koopman, B. Osyk, and J. Weast. "Autonomous vehicles meet the physical world: Rss, variability, uncertainty, and proving safety". In: *arXiv preprint arXiv:1911.01207* (2019).

[25] P. Koopman and M. Wagner. "Positive Trust Balance for Self-driving Car Deployment". In: *International Conference on Computer Safety, Reliability, and Security*. Springer. 2020, pp. 351–357.

[26] P. Koopman and M. Wagner. *Toward a framework for highly automated vehicle safety validation*. Tech. rep. SAE Technical Paper, 2018.

[27] J. Lenard, R. Welsh, and R. Danton. "Time-to-collision analysis of pedestrian and pedal-cycle accidents for the development of autonomous emergency braking systems". In: *Accident Analysis & Prevention* 115 (2018), pp. 128–136.

[28] M. Leucker and C. Schallhart. "A brief account of runtime verification". In: *The Journal of Logic and Algebraic Programming* 78.5 (2009), pp. 293–303.
[29] L. Libkin. "Expressive power of SQL". In: *Theoretical Computer Science* 296.3 (2003), pp. 379–404.
[30] L. Masson. "Safety monitoring for autonomous systems: interactive elicitation of safety rules". PhD thesis. Université Paul Sabatier-Toulouse III, 2019.
[31] R. Myers and Z. Saigol. "Pass-fail criteria for scenario-based testing of automated driving systems". In: *arXiv preprint arXiv:2005.09417* (2020).
[32] S. OSullivan et al. "Legal, regulatory, and ethical frameworks for development of standards in artificial intelligence AI and autonomous robotic surgery". In: *The International Journal of Medical Robotics and Computer Assisted Surgery* 15.1 (2019), e1968.
[33] H. Prakken. "On the problem of making autonomous vehicles conform to traffic law". In: *Artificial Intelligence and Law* 25.3 (2017), pp. 341–363.
[34] J. Redmon and A. Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018).
[35] A. Rizaldi and M. Althoff. "Formalising traffic rules for accountability of autonomous vehicles". In: *2015 IEEE 18th international conference on intelligent transportation systems*. IEEE. 2015, pp. 1658–1665.
[36] A. Rizaldi et al. "Formalising and monitoring traffic rules for autonomous vehicles in Isabelle/HOL". In: *International conference on integrated formal methods*. Springer. 2017, pp. 50–66.
[37] C. J. Robbins, H. A. Allen, and P. Chapman. "Comparing drivers' gap acceptance for cars and motorcycles at junctions using an adaptive staircase methodology". In: *Transportation research part F: traffic psychology and behaviour* 58 (2018), pp. 944–954.
[38] A. Rosenfeld and A. Richardson. "Explainability in human–agent systems". In: *Autonomous Agents and Multi-Agent Systems* 33.6 (2019), pp. 673–705.
[39] S. Shalev-Shwartz, S. Shammah, and A. Shashua. "On a formal model of safe and scalable self-driving cars". In: *arXiv preprint arXiv:1708.06374* (2017).
[40] A. Sinha et al. "A Crash Injury Model Involving Autonomous Vehicle: Investigating of Crash and Disengagement Reports". In: *Sustainability* 13.14 (2021), p. 7938.
[41] Y. Tao. "An introduction to assertion-based verification". In: *2009 IEEE 8th International Conference on ASIC*. IEEE. 2009, pp. 1318–1323.
[42] The Law Commission. *Automated Vehicles: joint report*. LC 204, Accessed: 2022-02-16. Jan. 2022.
[43] P. Tkachenko et al. "On-line maneuver identification in highway traffic using elastic template matching". In: *IFAC-PapersOnLine* 51.15 (2018), pp. 557–562.
[44] UK Driving Standards Agency. *The Official Highway Code*. Her Majestys Stationery Office, 2012.
[45] S. Ulbrich et al. "Defining and substantiating the terms scene, situation, and scenario for automated driving". In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE. 2015, pp. 982–988.
[46] US Department of Transportation. *PreCrash Scenario Typology for Crash Avoidance Research*. https://www.nhtsa.gov/sites/nhtsa.gov. Accessed: 2021-10-20.
[47] X. Wang, M. Yang, and D. Hurwitz. "Analysis of cut-in behavior based on naturalistic driving data". In: *Accident Analysis & Prevention* 124 (2019), pp. 127–137.
[48] A. F. Winfield and M. Jirotka. "The case for an ethical black box". In: *Annual Conference Towards Autonomous Robotic Systems*. Springer. 2017, pp. 262–273.
[49] R. H. Wortham. *Transparency for Robots and Autonomous Systems: Fundamentals, technologies and applications*. Institution of Engineering and Technology, 2020.
[50] B. Xue et al. "Safety Verification for Random Ordinary Differential Equations". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.11 (2020), pp. 4090–4101.

Chris Harper is a Research Fellow in Robotics Safety & Control at Bristol Robotics Laboratory, researching methods and technologies for safety assurance of autonomous systems, machine learning and AI. For over 20 years Dr Harper has worked in both industry and academia as a safety assurance specialist with experience in system/software hazard and safety analysis, safety cases and safety management, and has also been involved in development of international industry safety standards for robotics. His research interests include assurance methods, high integrity design, safety validation, and safety arguments/cases of autonomous systems.

Greg Chance received the B.Sc. (Hons.) and Ph.D. degrees in physics from the University of Bath, in 2001 and 2005, respectively. He was previously at the Bristol Robotics Lab and is currently a Senior Research Associate with the Trustworthy Systems Laboratory, University of Bristol. His current research interests are simulation-based verification for autonomous systems and cybersecurity. He has ten years industrial experience researching for Oxford Instruments, and BAE Systems. Dr Chance is a Chartered Engineer and a member of the Institute of Physics and holds a patent for a novel plasma control method.

Abanoub Ghobrial received an MEng in mechanical engineering from the University of Bristol in 2018. He is currently pursuing a Ph.D. degree in computer science at the University of Bristol and is a part-time Research Associate with the Trustworthy Systems Lab, where he was a full-time Research Associate from 2018 to 2020. His current research interests are techniques to allow self-managing of autonomous safety-critical systems via continual learning during operation and the development of simulation-based verification techniques for autonomous systems.

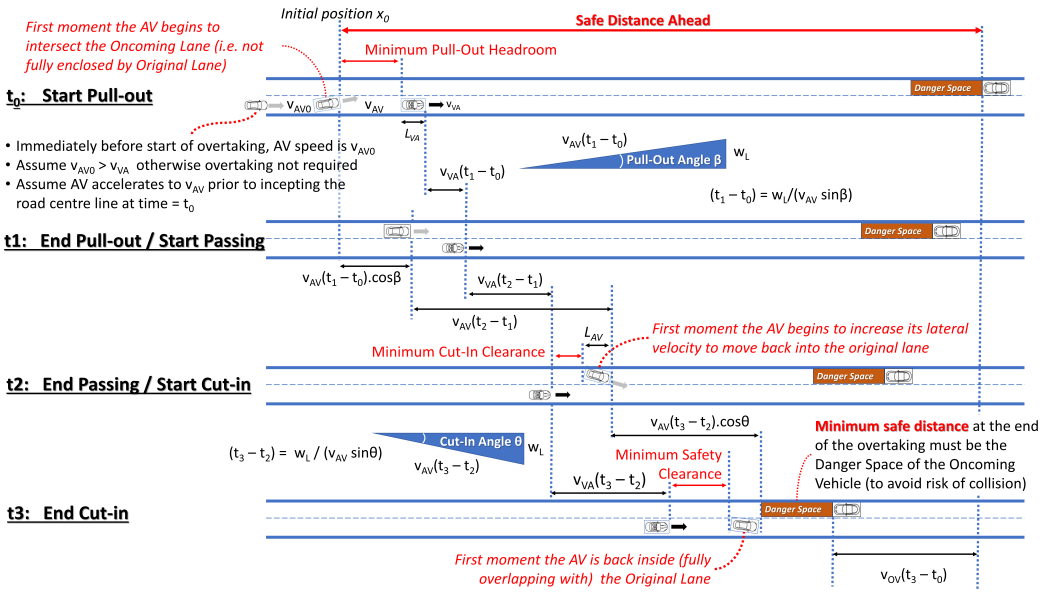
Saquib Alam received a B. Tech in engineering physics from the Indian Institute of Technology, Mumbai in 2015 and MSc in control systems and robotics from KTH Royal Institute of Technology, Stockholm in 2020 with a specialisation in autonomous control. Since then he has worked as a research engineer at the Bristol Robotics lab building a cloud testbench for autonomous vehicle verification using simulation.

Tony Pipe received his BSc (Hons.) degree in Electronic Engineering from Warwick in 1979 and, after an eight-year period in industry that included running a small company, received his PhD in Robotics from the University of the West of England (UWE) in 1997. Since 2010 he has been a Professor of Robotics and Autonomous Systems at UWE. He was a founding member of the Bristol Robotics Laboratory (BRL), and acted as one of its two Deputy Directors from its inception in 2006 until 2020. His research interests focus on safe and correct operation whilst in close-proximity to human beings. He has co-authored over 200 international refereed publications, supervised 35 PhDs, and created a BRL income of £10M since 2013.

Kerstin Eder is Professor of Computer Science and heads the Trustworthy Systems Laboratory at the University of Bristol, UK. She also leads the Verification and Validation for Safety in Robots research theme at the Bristol Robotics Laboratory. Prof. Eder's most recent contributions include agent-based testing, how to use assertions and theorem proving to verify control system designs as well as novel energy modeling and static resource analysis techniques to predict energy consumption of software. She holds a PhD in Computational Logic and an MSc in Artificial Intelligence from the University of Bristol, UK, in addition to an MEng in Informatics (Dipl. Inf.) from the Technical University Dresden, Germany. In 2007 she was awarded an "Excellence in Engineering" prize from the Royal Academy of Engineering, UK.

APPENDIX A

Figure 6 presents a model of an overtaking manoeuvre, showing the derivation of a formula for calculating Safe Distance Ahead, which was used in the simulation study results presented in Section V-B.



$$\text{Safe Distance Ahead} = \frac{w_L}{\tan\beta} + v_{AV}\left[\frac{k+(v_{VBP}-v_{AV}\cos\beta)\left(\frac{w_L}{v_{AV}\sin\beta}\right)}{(v_{AV}-v_{VBP})}\right] + \frac{w_L}{\tan\theta} + DS(v_{OV}) + v_{OV}(t_3 - t_0)$$

where DS = Danger Space (Thinking + Braking distances from UKHC Rule 126)
k = **Minimum Pull-Out Headroom** + L_{VA} + **Minimum Cut-In Clearance** + L_{AV}

Fig. 6: Schematic of model-based analysis of overtaking manoeuvre.