An Automated Vehicle Safety Concept Based on Runtime Restriction of the Operational Design Domain

Ian Colwell[1], Buu Phan[1], Shahwar Saleem[1], Rick Salay[1], Krzysztof Czarnecki[1]

***Abstract*— Automated vehicles need to operate safely in a wide range of environments and hazards. The complex systems that make up an automated vehicle must also ensure safety in the event of system failures. This paper proposes an approach and architectural design for achieving maximum functionality in the case of system failures. The Operational Design Domain (ODD) defines the domain over which the automated vehicle can operate safely. We propose modifying a runtime representation of the ODD based on current system capabilities. This enables the system to react with context-appropriate responses depending on the remaining degraded functionality. In addition to proposing an architectural design, we have implemented the approach to prove its viability. The proof of concept has shown promising directions for future work and moved our automated vehicle research platform closer to achieving level 4 automation.**

I. Introduction

Due to the inherent difficulty of solving the "self-driving car problem", Automated Driving Systems (ADS) require complex software and hardware system architectures [1][2]. These systems are expected to operate safely even in the event of system failures or hazardous external conditions such as poor weather. An ADS must be able to achieve a minimal risk condition (such as pulling to the side of the road) if it detects any issues with its own functionality or external conditions that prevent further safe operation. We propose a safety concept and architectural design that integrates functional degradation and functional boundary monitoring to maintain a runtime representation of the functional boundary based on current system capabilities.

The latest recommended practice from SAE [3] presents the need to monitor the Operational Design Domain (ODD) at runtime. The purpose of ODD monitoring is to determine whether or not the ADS is in a situation that it was designed to handle safely. The ODD can also be described as the "functional system boundary". For example, if the ODD contains only unsignalized intersections, the ADS must not enter intersections controlled by traffic lights. If ODD monitoring determines that the ADS has violated, or is about to violate, the bounds of the ODD, then it is expected to initiate a Dynamic Driving Task (DDT) fallback in order to achieve a minimal risk condition. A DDT fallback for a level 3 ADS is human intervention and a DDT fallback for a level 4 or 5 ADS is executed by the ADS itself.

For example (as shown in figure 1), if the side-facing long range sensing (long-range radar) required to turn onto a major road from a minor road (T intersection) fails before the intersection, the vehicle must not enter it and may need to pull over. There exists work on defining and monitoring the functional system boundary [4] [5] (we will use the term ODD in the sequel) and work on graceful degradation approaches [6] [7] for handling failures. However, these existing works do not address how the degraded functionality of the ADS affects the ODD or how the ODD monitoring system should handle degraded operation. We contribute the concept of *restricting the ODD based on current system capabilities*. We call this concept the Restricted Operational Domain (ROD), since it represents a restricted version of the ODD in which the ADS can still operate safely. The representation of the ROD allows the ADS to perform runtime monitoring of the ROD, and evaluate the domain of safe operation during changing system capabilities or faults. Continuing the T intersection example: if the right side facing long-range radar fails, the ODD is restricted to exclude left turn and straight through movements at T intersections since the ADS would have insufficient detection range for vehicles approaching from the right. In this degradation scenario a right turn may still be allowed, since approaching vehicles from the front or left can be detected and the remaining right side short-range sensing will detect pedestrians and other obstacles. This restriction allows the ADS to either continue safely and re-plan a route that avoids violating the ROD, or to perform a DDT fallback to a minimal risk condition if an intersection can not be avoided.

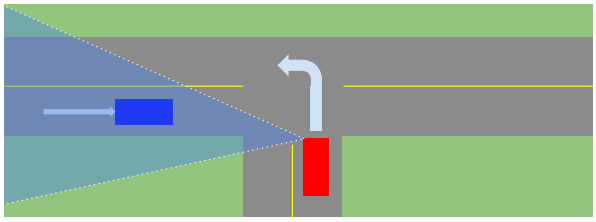

Fig. 1. An example scenario where side-facing sensing is required to navigate an intersection.

The main contributions of this paper are the concept of restricting the ODD at runtime based on current system capabilities and providing an architectural design that integrates both ROD monitoring and graceful degradation. Among other benefits, this integrated architectural design allows the vehicle to continue to operate within a safe domain and monitor the boundary of the safe domain during changing system capabilities. In addition, we implement a proof of concept on an automated driving research vehicle

[1]All authors are with the Waterloo Intelligent Systems Engineering Lab, University of Waterloo, Waterloo, ON N2L 3G1 Canada (email: icolwell, btphan, s25salee@uwaterloo.ca; rsalay, kczarnec@gsd.uwaterloo.ca)

to demonstrate the idea.

Section II reviews the importance of ODD monitoring and summarizes existing work related to fault-tolerant and graceful degradation methods for automated vehicles. The following sections introduce the key concepts behind the approach (III) and presents an architectural design to apply the approach on an automated vehicle (IV). In Section V we share the experiences and results of applying the architectural design to an automated driving research vehicle as a proof of concept. Finally, Section VI discusses the remaining challenges and future work to address these challenges.

II. Background and Related Work

The computing systems inside an ADS have already approached a level of complexity that warrants the application of autonomic computing principles [8], such as self-awareness and self-adaptation [9],[10]. This paper addresses part of this need for autonomic computing principles by proposing an integrated concept for self-awareness and self-adaptation that maintains a runtime representation of its potentially restricted ODD and adapts its behaviour to remain within this boundary. The following subsections summarize the definition and monitoring of the ODD as well as the related works regarding fault tolerance methods for automated vehicles.

A. Operational Design Domain (ODD)

SAE J3016 defines the ODD as "The specific conditions under which a given driving automation system or feature thereof is designed to function, including, but not limited to, driving modes." [3]. Examples of conditions include geographical areas, road types, traffic conditions, and maximum speed of the subject vehicle. The main purpose of the ODD is to precisely specify the domain in which the ADS can *safely* perform the DDT. The work of Geyer et al. [11] provides an ontology for a structured representation of the ODD. K. Czarnecki's work describes this ontology using a detailed "Operational World Model" [12], which can be used as a reference for creating an ODD. In general, the ODD is useful for the following tasks:

- *Design Process*: Defining the ODD helps identify what scenarios the ADS must handle. System-wide and functional requirements can then be defined alongside the ODD.
- *Testing and verification*: The ODD can be sampled to generate test cases with varying levels of detail for unit testing or integration testing via simulation.
- *Online monitoring*: The ODD can be instantiated as a runtime object to be measured and validated during operation. This is also known as functional boundary monitoring [4].

B. ODD Monitoring

SAE levels 1 to 4 of driving automation define an ODD for a limited range of ADS functionality. While the SAE definition for level 5 automation specifies an "unlimited ODD", a level 5 ADS must offer the same mobility that an average human driver can provide. For the purposes of this paper, we assume that level 5 automation still requires some form of ODD monitoring for determining when the ADS is outside of its designed functional boundary, which may also be beyond what a human driver could safely handle. Levels 3 and 4 require the system to monitor the ODD and respond to ODD violations by requesting manual control from a fallback-ready user (level 3) or by automatically performing the DDT fallback (level 4). In our more general interpretation of ODD monitoring, a level 5 ADS must also monitor its ODD and be able to perform fallback if it was to exit the ODD, such as in a sudden severe weather event. The purpose of the DDT fallback is to achieve a "minimal risk condition" or "safe state", which depends on the situation [13]. An example of a DDT fallback to a minimal risk condition is a pullover manoeuvre to a stopped position on the side of the road. If an ADS is expected to detect whether it has left the ODD, then it must be able to monitor the ODD at runtime. Wittmann et al. highlighted the need for monitoring the functional boundary in order to ensure safe operation [5]. Their use of the term "functional boundary" is the same concept as the ODD since a domain can be defined in terms of its boundary. They specify a functional boundary in terms of five subspaces consisting of static environment, traffic dynamics, environmental conditions, state of the subject vehicle, and passenger actions. The work of Horwick et al. provides a robust architecture for performing functional boundary (ODD) monitoring and DDT fallback execution [4]. While these existing works clearly outline the need and use of ODD monitoring, system faults are treated as events that induce immediate DDT fallbacks if unhandled by redundancy. Our contribution is to introduce the ROD as a restriction of the ODD based on current system capabilities, so that the system can continue operating safely in the ROD or perform a DDT fallback in the case of ROD violations.

C. Fault Tolerance in Automated Vehicles

Typical safety-critical fault-tolerance strategies involve redundancy of system components, whether it be running processes, hardware components, communication pathways, or all of the above [14]. These techniques guarantee a high level of system dependability, but often come at a significant cost increase due to redundancy measures. While some level of redundancy will always be required for safety assurance, what if the ADS could continue to operate safely after suffering system faults?

Graceful degradation allows a system to continue operating with reduced functionality when parts of the system fail. Automated vehicles can benefit greatly from graceful degradation strategies since they operate in environments with changing requirements (highway vs. busy urban street). They also use a diverse suite of sensors, which suggests they could operate without every single sensor fully functioning at all times. In terms of safety, graceful degradation is essential for ensuring automated DDT fallbacks in the case of system failures. Several fault-tolerant safety and monitoring systems for automated vehicles exist.

The work by Reschka et al. on fault tolerance for automated vehicles presents a surveillance and safety system that influences vehicle control and driving manoeuvrers in response to system failures [6]. Specific performance criteria (such as localization accuracy, sensor coverage, etc.) are maintained by surveilling the system. These performance criteria then lead to intentional degradation actions used to limit driving manoeuvres and adjust driving parameters.

Ability and skill graphs have been suggested as an approach for achieving graceful degradation in automated vehicles [7]. The runtime skill graph provides a detailed representation of the current abilities of the system, which can be used to re-configure the system or support decision making of the ADS. However, no relation is made to the functional boundary or ODD monitoring.

The existing works present ways to improve the safety of an ADS in the case of system faults. However, they do not address how system faults affect the ODD or how to integrate their approach with an ODD monitoring system.

III. Proposed Approach

The main purpose of the proposed approach is to produce a runtime representation of the domain in which the ADS can currently operate safely, regardless of functional degradation. In order to achieve this, we must answer the question "How should the ODD be restricted if a given subsystem is degraded?". This question is addressed by creating a mapping between degraded subsystem functionality and ODD restrictions. The following is needed in order to create this mapping:

1) An ODD definition that can be monitored at runtime.
2) Predefined subsystem Degraded Operation Modes (DOM).

Item 1 will be addressed in the following section and item 2 will be discussed, after a detailed explanation of the ROD, in section III-C.

A. Traceability Between Functional Requirements and the ODD Definition

Ideally, the ODD should be defined at the very beginning of the design process. The initial ODD would begin from a high level perspective and be further detailed and refined as the design of the ADS progresses. For example, Figure 2 shows an ISO 26262 inspired design process, and highlights phases where the ODD is continually refined [15]. An example of ODD refinement during the design process is starting with an ODD of "Driving within a specific city", and then refining it to the types of roads that exist within this city, or the precise geographical boundary of this city. In the end, the ODD should accurately represent the domain in which the ADS can safely perform the DDT.

In order to restrict the ODD in the case of degraded operation, it is essential to establish the link between ODD elements and system functionality requirements or specifications. As soon as a rough functional architecture is defined in the design process, this linking between ODD elements and system functionality can be made. Ideally, this linking should be done using a top-down approach in parallel with the definition of the ODD as previously mentioned. However, an existing system could be analyzed in a bottom-up fashion to determine a safe ODD based on implemented functionality and components. An example of top-down would be: first defining that the ODD contains other road users, such as motorcycles, and then linking this ODD element to a functional requirement of the perception subsystem that requires the detection of motorcycles. An example of bottom-up would be analyzing the functionality of an existing perception subsystem (and possibly other subsystems) and determining that the ODD can safely include motorcycles, since the analyzed subsystems have the functionality to detect and interact with motorcycles. During a real design process, the ODD and linking to subsystem functionality will likely be established using both top-down and bottom-up approaches as the design is further refined. This combination of top-down and bottom-up aligns with the "V" portion of the design process shown in Figure 2 where the design is iteratively improved based on verification and testing results.

Regardless of the approach, it is key to establish and maintain traceability between subsystem specifications and ODD elements.

For example, a system-level ODD capability of "left turn at T intersection" would define what the perception system must be able to detect, as shown in Figure 3.

ODD monitoring is required for levels 3 and higher. This means the ODD defines not only the functional requirements of subsystem components responsible for performing the DDT, but also the functional requirements of the subsystem components responsible for ODD monitoring. For example, if the ODD is constrained to only clear weather, then the ADS must be responsible for detecting the current weather condition in order to determine whether or not the ADS is within the ODD.

In the following section, we discuss how to use this traceability between subsystem functional requirements and ODD elements to restrict the ODD based on the degraded functionality of the ADS. This restricted ODD is a domain within which the degraded ADS can still function safely.

B. The Restricted Operational Domain (ROD)

We propose a new concept called "Restricted Operational Domain" (ROD), which is defined as:

> The specific conditions under which a given driving automation system or feature thereof is *currently able* to function, including, but not limited to, driving modes.

While the ODD is considered static throughout the operation of the ADS, the ROD may change given the degraded/restricted operation mode of the vehicle. In general the ODD can be seen as a set of constraints used to define the domain over which the system is designed to operate. The ROD constraints are a superset of the ODD constraints because more constraints are added to the domain based on the degraded functionality of the ADS. In terms of functionality,

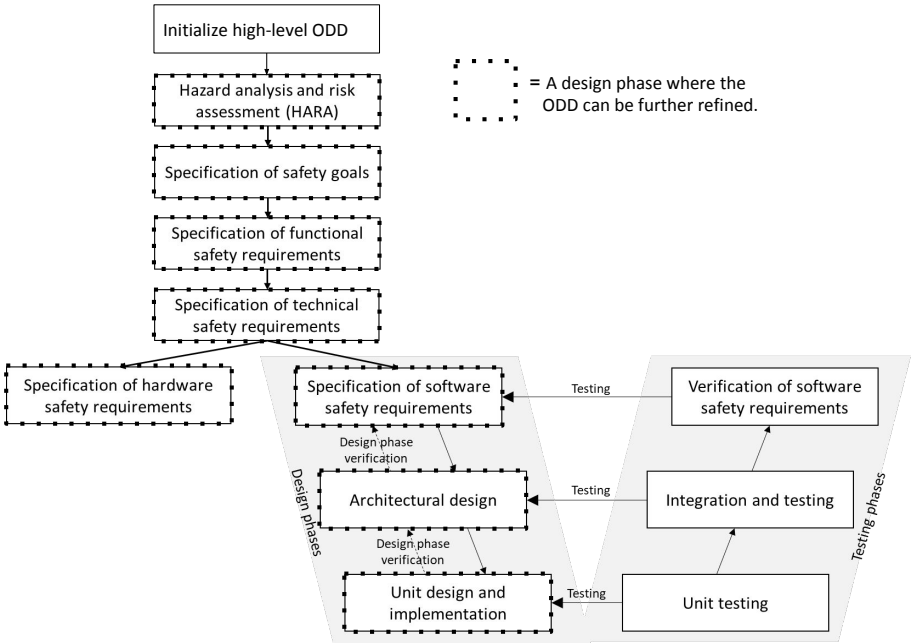

Fig. 2. The integration of the ODD definition into the design process.

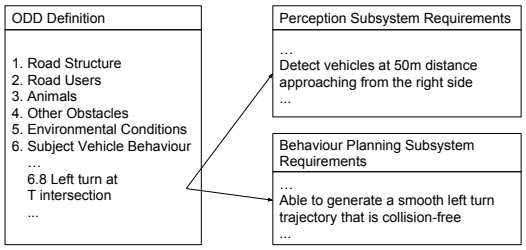

Fig. 3. The mapping between ODD elements and high-level subsystem requirements.

the ROD would allow a subset of the functionality that the ODD would allow as shown in Figure 4. When the system is fully functional the ROD is equivalent to the ODD.

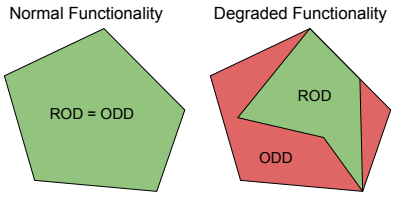

Fig. 4. The Restricted Operational Domain represented as an arbitrary 2D area.

Take the following failure case for example: A beam on the roof-mounted 360 degree LIDAR fails, creating a gap in the produced point cloud. The failed beam is in the middle of the range, at the height of other vehicles on the road. This failure affects three different subsystems that depend on the LIDAR data. The perception subsystem can not detect vehicles as far as it normally can, so the ROD constrains the subject vehicle speed so that the stopping distance is within the vehicle detection range. The localization subsystem is still able to perform scan matching, but the performance has been reduced and so the ROD constrains the speed of the subject vehicle to improve scan matches. This limits the type of roads to those that are under the constrained speed limit. The occupancy grid subsystem is not seriously affected as it can still create a representative grid based on the remaining beams. If more or different beams were to fail, the resulting restrictions would be different depending on the unique functionality of each subsystem. For example, losing one of the lower beams may have no effect on vehicle detection or localization, but could severely limit the ability of the occupancy grid subsystem to detect small obstacles on the road, such as curbs. If the subject vehicle was travelling at high speed on a highway while the failure occurred, the DDT fallback would need to occur since the vehicle immediately exited its ROD (due to a violation of the speed constraint). If the vehicle was on a road with a minimum speed below the speed constraint, the system would continue and re-plan a route that stays within the ROD or, if no such route is possible, would have to prepare for exiting the ROD similarly to preparing to exit the ODD.

C. System Degradation and the ROD

In order for the ROD to represent what domain the ADS can currently handle safely, we require a monitoring system capable of detecting the current internal state of the ADS functionality. System failures and functional degradation must be identified and measured. We define the Degraded Operation Mode (DOM) as:

> A sub-optimal functional operating mode of a system or subsystem.

A degraded operation mode (other than fully degraded) guarantees that some level of functionality remains. Fully degraded implies the subsystem is offline or failed. Some subsystems degrade in discrete modes, and some subsystems could degrade in a continuous fashion (such as confidence in calculated output data).

A *system impairment* is any effect on a component of the system that impacts the optimal performance of that component. Faults such as hardware or software component failures would be considered impairments. Other issues such as *confounding factors* are also considered impairments. Confounding factors are situations or events that complicate a task but are not considered failures or faults. Examples of confounding factors are snowfall interference in a LIDAR scan or obstructions to cameras. System impairments lead to functional degradation unless they are already handled by traditional fault tolerance techniques (e.g, redundancy and compensation). In summary, degraded operation occurs when the system experiences some kind of impairment.

Each subsystem has at least two operation modes: A normal operation mode and one that represents fully degraded or failed. Partially degraded functionality represents a DOM that lies somewhere in between the fully operational mode and the fully DOM.

The DOMs of the subsystems are monitored at runtime and used to update the ROD. In order to do this, a mapping has to be created that links DOMs to the ROD constraints that must be applied to the ROD based on the occurrence of that DOM. This mapping ensures that the ROD always represents a safe domain based on current functionality, regardless of the amount of system impairments or combinations thereof. An example of this mapping would be constraining the set of manoeuvres to exclude left turns at T intersections from the ROD if the right side facing vehicle detection system were to fail, as shown in Figure 3.

The main benefit of keeping a runtime representation of the ROD is that the ROD can be monitored for violations the exact same way as the ODD. This means if a system impairment occurs, the degraded ADS may still be able to continue safe operation within the ROD. Also, since ROD monitoring is occurring, the system will be able to engage a DDT fallback in the case of system impairments that result in a ROD that is violated or about to be violated by the actual driving environment (indicating unsafe or imminently unsafe operation).

IV. Proposed Architectural Design

A benefit of the ROD approach to system degradation/restriction is that we do not need to pre-define system-wide degradation modes and responses. We only need to define how a DOM in a subsystem maps to specific elements of the ROD. Once the ROD is updated, the subsystems are responsible for modifying their functionality to attempt to stay within the ROD. The subsystems need to be aware of ROD elements that are relevant to their operation and adjust accordingly. For example, the trajectory planner would need to know the ROD element that represents the subject vehicle maximum speed.

The proposed architectural design for enabling the ROD approach consists of a supervisory layer and a system layer, as shown in Figure 5. The system layer is responsible for executing the Dynamic Driving Task (DDT), and the supervisory layer is responsible for monitoring the system and responding to impairments by interacting with the system layer.

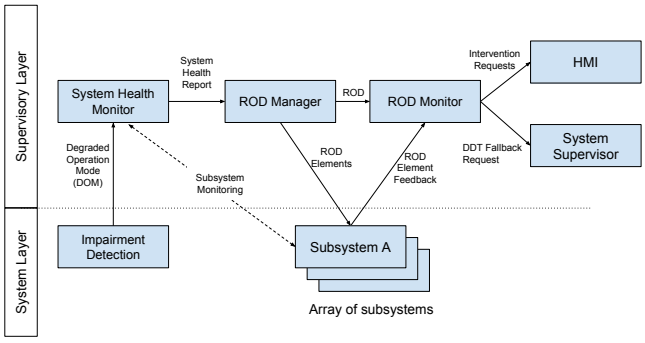

Fig. 5. The proposed architectural design.

The supervisory layer contains the core elements required to realize the ROD approach to system degradation. Each functional block of the supervisory layer can also be considered a subsystem on its own. A description of each of the subsystems in the supervisory layer follows.

A. System Supervisor

The System Supervisor maintains the current system state and manages requests for state changes. Some examples of system states would be "Manual", "Automated Drive", or "Emergency Pullover". The System Supervisor is also responsible for coordinating the DDT fallback based on the level of remaining functionality. Here we assume that there is redundancy in the system layer so that the ADS can always perform a DDT fallback. However, the design of this fallback system is outside the scope of this paper.

B. System Health Monitor

The System Health Monitor is responsible for maintaining a complete representation of the current system health known as the *System Health Report*. The system health report is an amalgamation of all the DOMs of the subsystems. At least three different strategies can be used to obtain the current DOM for each subsystem:

- *Self-monitoring*: A subsystem is responsible for monitoring its own DOM and sending these updates to the System Health Monitor, which records it and combines it into the system health report.
- *Distributed Monitoring*: A stand-alone subsystem is designed specifically for monitoring and evaluating the current DOM of one or several other subsystems. In this case, the monitoring subsystem would report the DOM of one or more subsystems to the System Health

Monitor, while leaving the subsystem under observation completely free of any diagnostic functionality.
- *Centralized Monitoring*: The System Health Monitor is responsible for monitoring a subsystem and generating the subsystem's DOM all on its own.

It is important to note that a System Health Monitor in production would most likely use a combination of these strategies to maintain the system health report. For example, if self-monitoring is used and the subsystem itself fails then there would be no way for the System Health Monitor to know that the subsystem is offline. This problem can be avoided by having the System Health Monitor perform centralized monitoring in the form of simple heartbeat checking, while more detailed diagnostics can be obtained by distributed or self-monitoring.

C. ROD Manager

The ROD Manager maintains the current ROD based on the system health report. The ROD Manager uses the mappings between subsystem DOMs and ROD elements (as mentioned in section III-C) to keep the ROD up to date given any combination of subsystem DOMs.

D. ROD Monitor

The ROD Monitor is responsible for determining whether or not the ADS has exited the ROD. Naturally, it also serves as the ODD monitor. The ROD Monitor must obtain certain information about the system in real time in order to verify that the vehicle and environment state has not violated the ROD constraints or is about to violate the constraints.

The ROD Monitor is similar to the System Health Monitor in the sense that its functionality could be spread out using self-monitoring, distributed monitoring, or central monitoring. While distributed or central monitoring makes the most sense for reliability reasons (a subsystem performing self-monitoring could itself fail, which could leave some ROD violations unmonitored), self-monitoring could be applied to give additional and more detailed feedback for unavoidable upcoming ROD violations. For example if the maximum subject vehicle speed was limited to 60 km/h, the planning subsystem/s could detect that the planned route eventually traverses high speed roads that will violate the ROD. The ADS can use this advanced ROD violation information to plan a new route, plan an optimal automated DTT fallback, or at the very least give advanced warning for smoother transition to a manual DDT fallback or the eventual minimal risk condition.

If the ROD Monitor detects an immediate violation, then it notifies the System Supervisor to initiate a DDT fallback. If an unavoidable upcoming ROD violation is detected by the ROD Monitor, then an intervention request is made to the fallback-ready user via the HMI systems as per SAE level 3 [3]. Unanswered intervention requests forces the ADS to perform the DDT fallback automatically (levels 4 and 5).

V. Proof of Concept

The ROD approach is implemented on the level 3 automated driving research platform currently in use at the University of Waterloo named the "autonomoose". The existing platform depends on cameras and lidar for its perception of dynamic objects, therefore the goal is to make the system tolerant to camera impairments (both camera failures and camera obstructions) using the proposed architectural design.

A perception impairment monitor detects obstructions to the camera feeds used for perception. If the camera becomes partially or fully obstructed, then the DOM of the camera is reported as fully degraded to the System Health Monitor. This is an example of distributed monitoring, as previously mentioned in section IV-B, since a stand-alone subsystem (perception impairment monitor) is reporting the DOM of another subsystem (perception).

Figure 6 shows the system that exists on the autonomoose research platform. It consists of the supervisory layer as outlined in Figure 5 and the newly created perception impairment monitor. The mission planner contains ROD element checking, so that it can respond to the changing ROD element that describes the types of roads that the ADS can currently traverse safely.

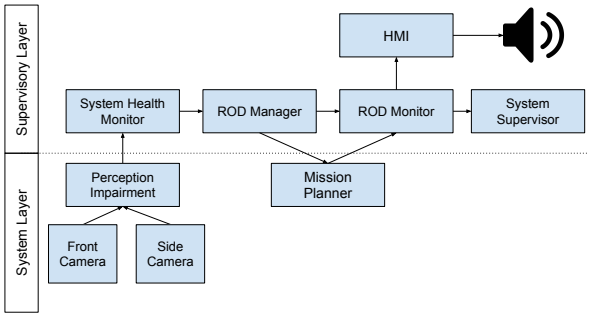

Fig. 6. The implemented supervisory layer and impairment detection.

The implemented supervisory layer responds to camera impairments in either the front or side facing camera. This setup allows the system to intelligently react in a context-aware fashion to camera impairments. The functionality of the implemented supervisory layer is explained using the following impairment scenarios:

1. **Front facing camera impairment:** The perception impairment monitor detects the impairment and updates the DOM of the camera subsystem. The System Health Monitor receives the DOM and updates the system health report. The ROD Manager becomes aware of the degraded state of the camera subsystem via the system health report and uses its mapping, implemented as a set of rules to update the ROD elements. In this case, the "road types" ROD element is modified to exclude all public roads, since the system requirements for safely traversing a public road can not be achieved without the front facing camera (and therefore forward perception) system. The mission planner determines that the car is currently traversing a road type that is not allowed and issues a ROD violated message to the ROD

Monitor. The ROD Monitor issues a request to intervene via the HMI, and the fallback-ready user takes control of the vehicle completing the DDT fallback.

2. **Side facing camera impairment:** A side facing camera impairment is detected in the same way as the front facing camera impairment; however, the system response differs. Since the side facing camera is not absolutely necessary for lane keeping on an unobstructed two-lane road segment, the "road types" ROD element is updated to remove intersections from the allowed road types. Thus, the system response will differ depending on the location of the vehicle. In the case where the vehicle is currently on a public local road segment ("local road" meaning two-lane road, 50 km/h maximum), the system issues a warning to the driver that the camera is impaired, but the system is still able to perform the DDT safely. In the case where the vehicle is approaching or within an intersection, the system issues a request to intervene, since side-facing perception is required to safely navigate intersections.

A video demonstration of the previously mentioned scenarios is available online [16]. This implementation is only a proof of concept that relies on intervention requests to a fallback-ready user; however, it paves the way for automated DDT fallback manoeuvres. Expanding the functionality of the supervisory layer and planning subsystems (to perform automated DDT fallbacks) will enable level 4 autonomy.

VI. Remaining Challenges and Future Work

While our proof of concept achieved its goal, we identified some challenges along the way.

Traceability between subsystem functionality and ODD: It may not always be easy to establish clear traceability between ODD elements and system functionality (or vice versa). Some elements of the ODD may be enabled by multiple subsystems working together. Therefore, if a link between an ODD element and every required subsystem is missed, it could cause the system to fail to restrict that element in the case of an impairment. It is also possible that subsystem functionality may impose very specific limitations to the ODD. For example, since the autonomoose is a research platform, it is common for subsystems to be developed according to research objectives rather than being derived from system level requirements in a top-down fashion. In this case, a subsystem may have very specific or limited operating conditions. Translating these specific conditions into ODD elements can be difficult without a highly detailed representation of the ODD.

Future work could address this challenge by using a highly detailed and standardized ontology of the ODD defined by a structured language, such as the Web Ontology Language (OWL), as suggested by [17]. Mappings between DOMs and ODD elements could then be defined using a structured language that integrates with the OWL representation of the ODD. This structured representation would also be used at runtime for ODD monitoring without the need to modify it.

Safety validation is another important consideration when determining traceability between DOMs and ODD element restrictions. The proposed approach depends upon the assumption that the DOM to ROD element mappings are valid. This assumption guarantees that as long as the DOM is detected and the correct mapping is applied, the ROD will accurately represent a domain in which the degraded ADS can operate safely within. The challenge lies within the safety analysis and verification of these mappings. Each mapping will need to be evaluated and verified to ensure that the ODD restrictions being applied do indeed result in a safe ROD given the degraded functionality. The safety verification process for all mappings could potentially increase upfront costs in the overall design process. However, if this process can ensure safe operation under system degradation, it provides the opportunity to reduce other methods of safety assurance. For example, the reduction of hardware redundancy measures could be highly profitable when mass-producing vehicles.

Future work could explore the impact of the proposed approach on the overall design process and propose concrete methods for evaluating the safety of the ADS within the ROD under various DOMs. In addition to exploring the verification of single mappings, the verification of combinations of mappings will also need to be addressed. Theoretically, if the DOM to ROD element mappings are valid, the ROD resulting from a combination of degraded subsystems should be safe, since the resulting ROD contains all the combined restrictions which would exclude any unsafe scenarios resulting from a combination of degraded functionality. However, this remains to be proven and is considered a topic for future work.

DOMs of interdependent subsystems: The definition of DOMs for subsystems that depend on other degraded subsystems could be non-trivial. For example, if subsystem A were to depend on the output of subsystem B and subsystem B became degraded, how should subsystem A respond? It is possible that subsystem A can not function without a fully functional subsystem B. If subsystem B were to become degraded, so should subsystem A. A subsystem could depend on multiple other subsystems all with their own DOMs. Each of these other subsystems provide different types of data that, when degraded, have different impacts on the resulting performance of the subsystem that depends on them.

Future work could explore using logics models to represent the dependence of subsystems on each other under multiple DOMs. This model could be stored in the System Health Monitor, so that the DOMs of dependent subsystems are updated when their dependencies degrade. Another direction could explore the use of similar concepts as introduced in [7] to evaluate the subsystem DOMs at runtime. These concepts introduce a graph based method to represent the interdependence of system abilities. The graph is used at runtime to determine the level of capability (or DOM, arguably) based on measured performance metrics such as reaction time.

Architectural Considerations: It is important to consider the architectural implications of the proposed supervisory layer. Consider the following example: The localization

solution begins to deteriorate (perhaps due to a failed wheel-speed sensor) resulting in higher covariances in the state estimate. Should the localization subsystem enter a DOM in response to the poor localization solution, should the state covariance be passed along to other subsystems along with the state estimate, or both? If the localization subsystem were to enter a DOM, the ROD Manager could update the ROD in response to this degradation. The subsystems that depend on the localization data could then adjust their operation based on the restrictions in the ROD. For example, the controller might have to respond to a new speed limit restriction from the updated ROD. In this case, the response to this impairment occurs via the supervisory layer after the System Health Monitor updates the system health report and after the ROD Manager updates the ROD. This information path could be too slow depending on the criticality of the impairment. The other option of passing the state covariance along with the state estimate would result in a more direct path to the required response. The controller could immediately slow down based on the state covariance. However, if other subsystems need to be aware of current restrictions (such as speed limitations), then they would also need to subscribe to this state covariance information or to a speed limit from the controller. For example, the mission planner needs to know the current speed restriction in order to re-plan a route that satisfies this restriction. Without the supervisory layer and structured ROD, the system architecture could become complicated with large amounts of subsystem connections used only for coordinating responses to system restrictions. In the end, it is likely that a combination of architectural approaches is necessary. The subsystems that directly rely on state estimates can receive the covariance along with the estimate, and the supervisory layer can also update the ROD to allow other subsystems (such as the mission planner), that might not normally use the state estimate, to adjust to the restrictions of the ROD. Further analysis of these architectural trade-offs and their impact on critical failure response latencies is considered future work.

Other directions for future work could explore using the context-aware nature of the architecture to dynamically re-configure the system based on current requirements. For example, if the vehicle is traversing a highway it could shut down certain sensors that are not required for highway driving. The mechanism for turning these sensors back on when leaving the highway could be activated by the existing mechanism for identifying an upcoming exit of the ROD.

VII. Conclusion

This work identified the need to integrate self-awareness and fault tolerance aspects into an ODD monitoring system to maintain a runtime representation of the ODD. To address this integration need, we introduced the concept of restricting the runtime ODD based on subsystem degradation modes. The proposed approach allows an ADS to continue to operate within a safe domain and monitor the boundary of the safe domain during changing system capabilities and faults. The proof of concept demonstrated the viability of the proposed approach, but also exposed some remaining challenges. Some of these challenges include the definition and safety validation of mappings between DOMs and ROD elements, the formulation of DOMs for interdependent subsystems, and system latencies induced by the proposed supervisory layer. Future work could address these challenges by investigating the use of a structured language to define DOM to ROD mappings, integration of existing concepts for determining complex subsystem DOMs, and analyzing the architectural trade-offs associated with the proposed supervisory layer.

VIII. Acknowledgements

The authors would like to thank Sean Sedwards, Steven Waslander, Thomas Fuhrman, and Ramesh S. for their insightful comments.

References

[1] P. Koopman and M. Wagner, "Transportation CPS safety challenges," in *NSF Workshop on Transportation CyberPhysical Systems*, 2014.

[2] Ö. Ş. Taş, F. Kuhnt, J. M. Zöllner, and C. Stiller, "Functional system architectures towards fully automated driving," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*. IEEE, 2016, pp. 304–309.

[3] *SURFACE VEHICLE RECOMMENDED PRACTICE Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, SAE J3016, 2016.

[4] M. Hörwick and K.-H. Siedersberger, "Strategy and architecture of a safety concept for fully automatic and autonomous driving assistance systems," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 955–960.

[5] D. Wittmann, C. Wang, and M. Lienkamp, "Definition and identification of system boundaries of highly automated driving," in *7. Tagung Fahrerassistenz*, 2015.

[6] A. Reschka, J. R. Böhmer, T. Nothdurft, P. Hecker, B. Lichte, and M. Maurer, "A surveillance and safety system based on performance criteria and functional degradation for an autonomous vehicle," in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE, 2012, pp. 237–242.

[7] A. Reschka, G. Bagschik, S. Ulbrich, M. Nolte, and M. Maurer, "Ability and skill graphs for system modeling, online monitoring, and decision support for vehicle guidance systems," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 933–939.

[8] A. G. Ganek and T. A. Corbi, "The dawning of the autonomic computing era," *IBM systems Journal*, vol. 42, no. 1, pp. 5–18, 2003.

[9] J. Schlatow, M. Moostl, R. Ernst, M. Nolte, I. Jatzkowski, M. Maurer, C. Herber, and A. Herkersdorf, "Self-awareness in autonomous automotive systems," in *2017 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2017, pp. 1050–1055.

[10] A. Reschka, G. Bagschik, and M. Maurer, "Towards a system-wide functional safety concept for automated road vehicles," in *Automotive Systems Engineering II*. Springer, 2018, pp. 123–145.

[11] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weißgerber, K. Bengler, R. Bruder, *et al.*, "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183–189, 2013.

[12] K. Czarnecki, "Operational World Model Ontology for Automated Driving Systems. Parts I and II." available at researchgate.net, November 2017.

[13] A. Reschka and M. Maurer, "Conditions for a safe state of automated road vehicles," *it-Information Technology*, vol. 57, no. 4, pp. 215–222, 2015.

[14] A. Kohn, R. Schneider, A. Vilela, A. Roger, and U. Dannebaum, "Architectural concepts for fail-operational automotive systems," SAE Technical Paper, Tech. Rep., 2016.

[15] *Road vehicles – Functional safety*, ISO 26262, 2011.

[16] "Autonomoose Public Drive - December 2017." [Online]. Available: https://youtu.be/i7S-JZYdb74?t=396

[17] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," *arXiv preprint arXiv:1704.01006*, 2017.