UNIVERSITE BRETAGNE LOIRE / MATHSTIC

CENTRALE NANTES

Thèse de Doctorat

Yrvann EMZIVAT

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'École centrale de Nantes
sous le sceau de l'Université Bretagne Loire

École doctorale : Mathématiques et STIC

Discipline : Automatique, Productique et Robotique – Signal, Image, Vision – Télécommunications
Unité de recherche : Laboratoire des Sciences du Numérique de Nantes (LS2N) – UMR 6004

Soutenue le 30 mai 2018

Safety System Architecture for the Design of Dependable and Adaptable Autonomous Vehicles

JURY

Rapporteurs :	**M^me^ Claire PAGETTI**, Ingénieur de Recherche, ONERA
	M. Fawzi NASHASHIBI, Directeur de Recherche, INRIA
Examinateurs :	**M. Philippe BONNIFAIT**, Professeur des Universités, Université de Technologie de Compiègne
	M. Javier IBANEZ-GUZMAN, Expert véhicule autonome, Renault
Directeurs de thèse :	**M. Olivier H. ROUX**, Professeur des Universités, École Centrale de Nantes
	M. Philippe MARTINET, Directeur de Recherche, Inria Sophia Antipolis

Contents

List of Tables

List of Figures

Acronyms

ACC Adaptive Cruise Control

ADAS Advanced Driving Assistance Systems

ADS Automated Driving System

ADS-DV ADS-Dedicated Vehicle

ADS-EV ADS-Equipped Vehicle

ACC Adaptive Cruise Control

BEA Bureau d'Enquêtes et d'Analyses pour la Sécurité de l'Aviation Civile

DDT Dynamic Driving Task

DMV Department of Motor Vehicles

ETA Event Tree Analysis

FHA Fault Hazard Analysis

FMEA Failure Mode and Effects Analysis

FMECA Failure Modes, Effects and Criticality Analysis

FTA Fault Tree Analysis

HAZOP Hazards and Operability Analysis

ITS Intelligent Transportation Systems

LDWS Lane Departure Warning System

LKS Lane Keeping System

MC Markov Chain

MDP Markov Decision Process

NHTSA National Highway Traffic Safety Administration

NMVCCS National Motor Vehicle Crash Causation Survey

ODD Operational Design Domain

OEDR Object and Event Detection and Response

OICA Organisation Internationale des Constructeurs d'Automobiles

ONISR Observatoire National Interministériel de la Sécurité Routière

PHA Preliminary Hazard Analysis

PASCG Probabilistic Atomic State Class Graph

PSSCG Probabilistic Strong State Class Graph

PTPN Probabilistic Time Petri Nets

SAE Society of Automotive Engineers

SIL Safety Integrity Level

STAMP Systems-Theoretic Accident Model and Processes

TPN Time Petri Net

TTC Time to Collision

V2X Vehicle-to-Everything communication

Acknowledgement

I would like to thank Professor Olivier H. Roux for his continued support and kindness since the time I first engaged in research and for sharing his keen interest for everything formal during that time. I am grateful to Professor Philippe Martinet for the particularly interesting discussions we had on perception and computer vision. I am also appreciative of Doctor Javier Ibanez Guzman's supervision during my time at Renault's Research and Development facility. A special thank you to Doctor Didier Lime, Doctor Hervé Illy and Doctor Benoît Delahaye for their welcome involvement in this work.

I would like to extend my gratitude to Doctor Claire Pagetti for her thorough proof-reading and relevant feedback. I am also thankful for Doctor Fawzi Nashashibi's earnest and kind report. Finally, I express a sincere thank you to Professor Philippe Bonnifait, whose responsiveness and role as jury president contributed to make the thesis defence a very pleasant experience.

.

Why did the chicken cross the road?
It assumed autonomous driving was safe.

What was the last event in the crash causal chain?
Chicken error.

Chapter 1

Introduction

1.1 Motivation

1.1.1 Issues and Challenges of Sustainable Mobility

The cost of mobility on today's society is often presented as the rationale behind the development of innovative new technologies that aim to provide cleaner, safer and more efficient means of transportation while providing a better user experience overall. In this regard, intelligent transportation systems (ITS) have generated increased interest as of late, due to their potential to improve roadway safety, reduce traffic congestion, and meet the increasing needs for the mobility of people and goods. ITS consist of a wide variety of technologies and applications such as vehicle, traffic management and travel information systems. They do not embody intelligence as such, yet they aim to provide innovative services that enable better informed users to enhance their everyday experience by making a safe and efficient use of transport networks [6]. The importance of ITS lies in their potential to create a paradigm shift in transportation technology, by developing vehicles and infrastructures that are able to cooperate effectively and efficiently in an intelligent manner [7].

More specifically, motivation for the development of intelligent transportation systems rests on the following key figures:

1. According to OICA [1], the total number of motor vehicles on the planet crossed 1 billion in 2010, growing from 892 million in 2005 to 1.282 billion in 2015. In Europe alone, the vehicle fleet grew from 322 million to 388 million over the same period. Yet it is estimated that a car is parked 95% of the time.

1. The International Organisation of Motor Vehicle Manufacturers (Organisation Internationale des Constructeurs d'Automobiles) is an international trade association.

2. Only 45% of the French population has easy access to public transportation. At the same time, dense urban areas are saturated. On average, French people make 3.6 trips and spend 78 minutes in their car every day. Their behaviour is an indication of the disparity of territory servicing. Congestion has negative consequences on health as it induces stress, fatigue and increases air pollution.
3. While a variety of factors contribute to accidents, the last event in the crash causal chain is assigned to the driver 90% to 95% of the time [8]. Driver-related critical reasons underlying the alleged liability of the driver for a crash include speeding, drunk driving, distracted driving (such as texting) and other forms of reckless driving leading to the violation of traffic rules.
4. A weighted sample of 5 470 crashes was investigated [9] in the National Motor Vehicle Crash Causation Survey (NMVCCS) [2] by the U.S. Department of Transportation National Highway Traffic Safety Administration. Driver-related critical reasons were broadly classified into recognition errors, decision errors, performance errors and non-performance errors. Recognition error includes drivers' inattention, internal and external distractions and inadequate surveillance. They accounted for about 41% of the crashes. Decision errors such as driving too fast given the road and weather conditions, false assumption of others' actions, illegal manoeuvre and misjudgment of gap or others' speed accounted for about 33% of the crashes. In about 11% of the crashes, the critical reason was performance error such as overcompensation and poor directional control. Sleep was the most common critical reason among non-performance errors, accounting for 7% of the crashes.
5. Data shared by the ONISR [3] [8] determined that road traffic accidents claimed 3 477 lives and left 72 645 injured in France in 2016. This amounts to an estimated cost of 38.3 billion € [4]. On average, one person dies every minute somewhere in the world due to a car crash.

Driving automation has attracted much attention alongside intelligent transportation systems, fostered by the potential and promise of new public services addressing relevant transportation issues, mainly [10]:

- the improvement of road safety, as the alleged responsibility of human drivers accounts for the majority of all accidents,

2. The National Motor Vehicle Crash Causation Survey (NMVCCS), conducted from 2005 to 2007, was aimed at collecting on-scene information about the events and associated factors leading up to crashes involving light vehicles over a period of two and a half years.

3. The Observatoire National Interministériel de la Securité Routière (ONISR) is an organisation whose role is to collect, interpret and diffuse national and international statistics relating to road insecurity in France.

4. The evaluation methodology uses reference values defined in the document 'Instruction du 16 juin 2014 relative à l'évaluation des projets de transport' of the French government.

- the improvement of the commute experience, allowing part of the commute time to be re-allocated to tasks other than driving,
- the improvement of mobility for everyone, giving differently abled people access to a form of transportation that is adapted to their needs,
- the reduction of fuel consumption, making autonomous driving a green and eco-friendly alternative to more traditional means of transportation, and
- the reduction of traffic congestion, which contributes to most of the above.

Driving automation is a sensible solution to the aforementioned challenges. It has the potential to have a profound, far-reaching impact through a wide range of applications. For example, driving automation is expected to expand the scope of public transit to on-demand shared mobility. It provides opportunities for package delivery and garbage collection services as driving currently makes up for a significant amount of a worker's shift. The technology could also be an asset for the industry and the military, as truck platooning and autonomous military convoys are developed.

However, the development of driving automation is a challenge in itself. While the general population expects vehicles equipped with this technology to perform regular trips from home to work on a sustained basis, the deployment of truly autonomous vehicles at affordable prices for private use seems unlikely in the near future. In fact, car manufacturers envision a gradual transition towards full driving automation, by selling cars with increasingly sophisticated driver-assistance features first. This has led to the widespread misconception that driver assistance systems will gradually evolve into fully autonomous cars. Driver assistance systems, such as auto-parking, emergency braking and lane warning only operate for short periods of time in limited settings by responding to specific events. The sustained performance of the dynamic driving task by a dedicated system meant to rival and ultimately replace a human driver requires a substantial jump in technological capabilities that has yet to be achieved.

1.1.2 Driving Automation

Terms such as 'autonomous', 'self-driving', 'driverless' and 'unmanned' are often used inconsistently and confusingly to characterise driving automation systems and vehicles equipped with them. As such, the use of these terms can lead to confusion, misunderstanding, and diminished credibility. According to the Recommended Practice published by SAE International [3],

> "[The term *autonomous*] has been used for a long time in the robotics and artificial intelligence research communities to signify systems that have

Table 1.1 – Reference architecture for intelligent systems [1]

System	Cognitive Science	Description
Sensory processing	Perception: frontal lobe (attention), parietal lobe (tactile, heat, pain), occipital lobe (vision), temporal lobe (hearing)	Data and image filtering, abstraction processes by cognitive signal processing using clustering, self-organising artificial neural networks, reinforcement learning, etc., to recognise signal patterns, objects and events in the environment
World model	Coordination of sensations: middle cortex sections. Maps: lobe (precuneus) Knowledge: frontal lobe	World model can include static geometric models, visual models, memory organisation, abstract models of the present and past, dynamical modelling and parameter estimation, etc.
Behaviour Generation	Abstract planning: frontal lobe. Motor responses: cerebellum	Behaviour generation can be more complex than running a simple feedback/feedforward control algorithm. It can also include complex composition of switching and realtime adaptation of controllers supervised by symbolic computation
Value judgement	Decision making: frontal lobe. Sensual assessment: middle cortex regions	Value judgement can be much more than evaluating a fixed performance function. Goal oriented behaviour of agents means that the agent selects its own performance functions which can be constrained by behaviour rules and can require to make a compromise between partially conflicting goals

the ability and authority to make decisions independently and self-sufficiently. Over time, this usage was casually broadened to not only encompass decision making, but to represent the entire system functionality, thereby becoming synonymous with automated. This usage obscures the question of whether a so-called 'autonomous vehicle' depends on communication and/or cooperation with outside entities for important functionality (such as data acquisition and collection). Some driving automation systems may indeed be autonomous if they perform all of their functions independently and self-sufficiently, but if they depend on communication and/or cooperation with outside entities, they should be considered cooperative rather than autonomous. [...]

Additionally, in jurisprudence, autonomy refers to the capacity for self-governance. In this sense, also, 'autonomous' is a misnomer as applied to automated driving technology, because even the most advanced Automated Driving Systems are not 'self-governing.' Rather, they operate based on algorithms and otherwise obey the commands of users."

For the purpose of complying with the Recommended Practice, the popular term 'autonomous' is not used here to describe driving automation.

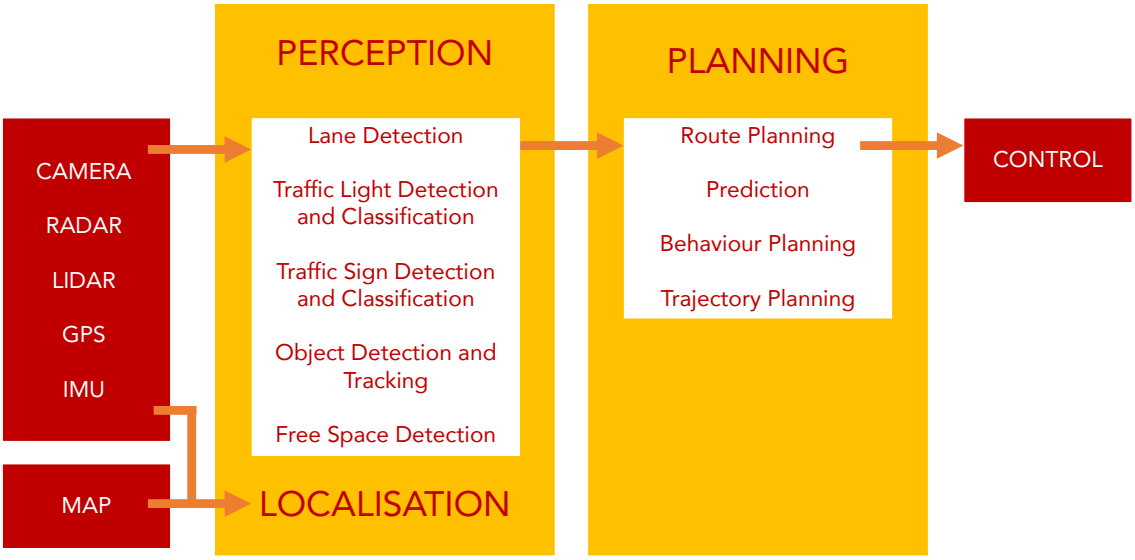


Figure 1.1 – General architecture of an Automated Driving System [2]

Functional System Architecture

A system architecture is a conceptual model that defines the structure and behaviour of a system. The general architecture of a cognitive system contains the main processing units of sensory processing, world model, behaviour generation and value judgement [11]. Table 1.1 points out parallel concepts in cognitive systems and their location in the human brain. The architecture of an Automated Driving System subsumes the real-time operational and tactical functions required to operate a vehicle in on-road traffic. These functions are sometimes defined in terms of ability and skill graphs [12]. For the most part, the functional architecture of an Automated Driving System stems from the natural idea of the Sense-Plan-Act (SPA) paradigm, though it is very rarely used in its most basic form. The system is often divided into a perception component, a behaviour and planning component and a component for vehicle control and actuation [13, 14]. The general architecture of a Driving Automation System is given in figure 1.1. Depending on the system, situation assessment is either part of the perception component, the planning component, or both [15]. Decision making is usually part of the behaviour and planning component, though it is sometimes merged with vehicle control and actuation to form the navigation component of the system. Localisation and map service can be included in the perception component, the navigation component, or be a component of its own [16].

In essence, the core of an Automated Driving System consists of two primary systems [17]:

1. a perception system, whose purpose is to interpret and understand the vehicle's surroundings, and
2. a navigation system, whose purpose is to guide the vehicle in the driving environment.

In the following, it is assumed that situation understanding is part of the perception system and that decision making is part of the navigation system.

The perception system detects dynamic and static obstacles, attempts to identify them and measures their speed. Exteroceptive sensors such as cameras, lidars and radars gather data about the environment and deliver it to a sensor fusion component for processing. The recovered features are used to build an explicit representation of the world as the system knows it.

The navigation system generally consists of a trajectory generation component, whose role is to generate a set of obstacle free trajectories, a decision making module, which picks the optimal manoeuvre or trajectory, and a control sub-system, which executes the chosen trajectory through the use of actuators.

System localisation is needed to determine on which road the vehicle is travelling through map-matching. The navigation map provides contextual information that is used for situation understanding and autonomous navigation. It stores information about the road geometry and topology. On a macroscale level, topological road network maps are used to augment perceived information with a priori map information. On a mesoscale level, lane level map information may be used to augment context modelling beyond the limited field of view of on-board sensor systems. On a microscale level, feature information may be used to provide additional landmarks or to stabilise lane tracking [16]. Proprioceptive sensors (vehicle movement, pitch, charging level of the battery, fuel tank) are used to obtain information about the vehicle and its internal state, while wireless communication (V2X) is sometimes used to obtain additional information about the environment.

1.1.3 Technical Challenges

High levels of dependability are required for full-scale fleet deployment. In theory, Automated Driving Systems can outperform human drivers for they possess a wider field of view and can endure higher workloads. They can perform multiple tasks concurrently with little to no effect on their general performance. They are not subject to human flaws that often lead to accidents such as drinking and driver fatigue. However, the true capabilities and behaviour of these systems remain unclear for the most part, which makes

any attempt at proving that they can be safe especially difficult. Moreover, the resolution of major technical challenges at the sub-system level has essentially taken precedence over the definition of an architecture that would establish that the whole system is sound and provably safe.

Perception and Situation Understanding

A key shortcoming to the development of a robust perception system lies in the lack of publicly available information pertaining to the performance of existing ones. The majority of published studies are conducted under optimal weather conditions, with no guarantee of robustness and very little in the way of repeatability. Yet weather conditions have a significant impact on the performance of these systems [18]. Light conditions (daylight, nighttime, dusk, dawn), cloud cover (clear, cloudy, sunny), precipitation (rain, hail, snow) and fog (patch fog, ground fog) are but a few relevant characteristics of the environment that need proper consideration when designing the perception system of an Automated Driving System. Sun glare and fog blind cameras. Smog generates ghost targets. Snow and heavy rain hide features of the environment that enable the system to distinguish the road (e.g. lane markings). In general, lidar is brittle to laser blockages, airborne precipitation and wet surfaces, whereas dark lighting conditions and glare mostly affect cameras by lowering image contrast. Radars display best robustness to all weather conditions, but suffer from low angular resolution and usually provide less information about object shape, size and classification than lidars or cameras. Beyond these preliminary, empirical observations, there is a lack of relevant metrics [19] allowing for proper assessment of the perception system's capabilities.

A standard methodology for the representation of the driving environment is needed but none has yet been developed in-depth. Ideally, the world model includes all of the actors' and observers' self-representations in the scene, as well as the relationships between them. However, such a representation is usually incomplete, incorrect and uncertain from one or several observers' point of view [20]. In fact, current systems fail to properly detect and classify important features in the environment and have trouble making sense of what is seen in general. Machine learning techniques are widely used and tend to be based on inductive training approaches, yet validating inductive reasoning is known to be inherently difficult. Most systems display an unsatisfactory classification rate of critical elements in the scene, such as cyclists, which often end up being confused with pedestrians. The classification of other key elements in the driving environment, such as scooters and unicycles, also requires further investigation. Furthermore, many expensive exteroceptive sensors come with proprietary black-box on-board processing whose characteristics are

not made readily available. As a result, a thorough understanding of their capabilities cannot be achieved easily.

Wireless communication (V2X) can provide an Automated Driving System with valuable information about its environment and tackle some of the aforementioned problems. However, V2X information can be uncertain, incorrect or intentionally misleading. This means that security measures will need to be developed to encompass system-level attacks and failures.

Maps and Localisation

Road networks are ever changing. Consequently, the embedded navigation map must be updated on a regular basis for it to be used reliably. It is estimated that 15% of road networks experience significant changes in developed countries in a single year. Yet building a map for navigation is a time-consuming process. By the time a new map becomes available, the information it provides may already be outdated. Although the contextual information contained in maps is assumed to be correct, they can contain structural, geometric and contextual faults [21].

Global Navigation Satellite Systems, such as GPS, GLONASS and Galileo, are widely used for their relatively cheap cost. However, they suffer from low accuracy and can only be used in open-sky environments. They cannot, for instance, be used reliably in tunnels. In urban areas where road density is high, GNSS error can be significant. Any lack of satellite coverage or availability can jeopardise safety. Inertial measurement units can be used in addition to GNSS, but they are expensive and suffer from accumulated error.

Decision Making and Navigation

In general, the rules underlying the decision making process of Automated Driving Systems are not made apparent. It is unclear how they will adapt, how they will learn from their mistakes and how close to human action their behaviour will or can be. Communicating one's intentions is an important part of driving and the Nissan IDS [22] concept car is one example of a system that includes various exterior lights and displays to partially convey its intentions. Yet research in this area remains rudimentary and Automated Driving Systems do not currently have any convincing means of communicating and cooperating efficiently with humans driving conventional vehicles. These systems are expected to abide by traffic rules, yet a complete, formal model of traffic regulations is not available. As a result, it seems current systems only obey to the most basic of rules.

A review of motion planning techniques can be found in [23]. Motion models have been developed for motion prediction and risk assessment in an attempt to predict how a given situation can evolve [24]. Physics-based motion models consider that the motion of vehicles only depends on the laws of physics. Yet they are limited to short-term motion prediction and are unable to anticipate any change in the motion of a vehicle caused by the execution of a particular manoeuvre. Manoeuvre-based motion models consider that the future motion of a vehicle also depends on the manoeuvre that the driver intends to perform. They provide a more reliable estimation of long-term motion and risk, but ignore the dependencies between the vehicles in the scene. Their reliance on prototype trajectories also prevents their adaptation to different road layouts. Interaction-aware motion models take into account the inter-dependencies between vehicles' manoeuvres. They are the most comprehensive models and contribute to a better understanding of the situation and a more reliable evaluation of risk. Yet they suffer from computational complexity and have not been sufficiently studied. Overall, motion prediction and risk assessment require further work that is admittedly needed for navigation.

It is currently impossible to put forward convincing arguments when attempting to establish when and how Automated Driving Systems will outperform human drivers. Even though driving automation is still in its earlier stages of development and strong technical limitations remain, the automotive industry must work towards making these systems as safe as possible. Reflecting on the safety implications of deploying them on public roads in the future is needed today [25].

1.2 Research Problem

1.2.1 Safety Challenges

A significant issue underlying the deployment of vehicles equipped with Automated Driving Systems lies in the legal issues of liability. It seems car manufacturers currently design their systems with the expectation that a human driver will be ready to take over whenever necessary, effectively making one liable in case of a crash if one fails to do so in a timely manner. Yet it is unclear how to prepare human drivers to take over efficiently and how to provide them with the information they will need at the right time to make the right decision. It has been hypothesised that high levels of automation can cause a deterioration of human situation awareness, which can become problematic if a driver is to retake control within a short period of time [26]. There is also much concern to be had regarding the way a human driver handles the system. A vehicle equipped with a driving automation system is a complex, safety-critical system that must be operated with care.

Some may deliberately choose to act against the intent and design of the system or simply test its limits.

In fact, it has been stated that drivers are generally unaware of the capabilities and limitations of advanced driving assistance systems (ADAS) that are already commonplace. A cluster analysis was used to classify drivers based on their understanding of the limitations associated with adaptive cruise control (ACC) [27]. Three cluster groups emerged: those who were aware, unaware, and unsure of ACC limitations. Further examination revealed that drivers who were unaware or unsure of these limitations exhibited potentially hazardous behaviour when compared to the aware group. Yet they reported high levels of trust in the system and were more willing to use it when tired or when driving on curvy roads. This may be problematic, for trust in a system based on inappropriate expectations can impact safety. Studies suggest that this could potentially be mitigated through extended use of ACC [28]. Still, proper understanding of system capabilities by the user remains crucial regardless of the level of automation being achieved.

1.2.2 Lessons Learned from Recent Crashes

In May 2016, a 2015 Tesla Model S collided with a tractor trailer crossing an uncontrolled intersection on a highway in Florida, resulting in fatal injuries of the driver. Neither the system nor the driver noticed the white side of the tractor trailer against the brightly lit sky. This caused the car to pass under the trailer with the bottom of the trailer impacting the windshield. Data indicates that the car was being operated in Autopilot mode and that the driver took no action to avoid the collision. The Autopilot is an advanced driver assistance system that requires the continual and full attention of a human driver to monitor the traffic environment and be prepared to take action to avoid crashes. However, it has been pointed out that while Tesla has provided information about system limitations in the owner's manual, it may not have been as specific as it should have [29].

The importance of clearly defining the capabilities of driving automation systems cannot be stressed enough. The work presented in [30] provides analysis of the data contained in accident reports filed to the California Department of Motor Vehicles (DMV) for accidents involving driving automation systems that are undergoing testing on the state's public roads. Reports were filed by five manufacturers between September 2014 and March 2017, out of the thirty currently holding permits for public testing in California. The DMV created two databases: the Autonomous Vehicle Disengagement Reports Database, which lends itself to statistical analysis and the Report of Traffic Accidents Involving Autonomous Vehicles database, which provides more descriptive and detailed

reports for 26 actual accidents. It was found, among other things, that driving automation systems systematically fail to properly detect and react to rear-end types of collision.

In fact, driving automation systems have trouble behaving in a truly cautionary manner and preempting potential danger the way a human driver would. In many scenarios, a knee-jerk reaction of hitting the breaks is not sufficient and a preemptive action is required instead. In November 2017, a shuttle with driving automation capabilities designed by Navya was involved in an accident in Las Vegas as it hit a semi-truck that was backing out of an alleyway [31]. It was stated that the system brought the vehicle to a stop as it should have and the truck driver was blamed for the accident as per traffic regulations. However, the passengers of the shuttle said the accident could have been avoided if the system had only reversed. It apparently failed to do so because the truck's trailer moved in a way that it did not anticipate. The system did not make use of the horn either, as common sense would dictate. More recently, in March 2018, a woman was killed in Arizona after she was struck by an Uber operated in autonomous mode. The vehicle was travelling at a speed of 40 mph and it was found the system showed no signs of slowing down prior to impact. This accident would seem to indicate that autonomous systems available today are not quite ready to meet safety requirements yet. Uber, Toyota and Nvidia halted self-driving tests in wake of the crash [32].

1.2.3 Lessons Learned from the Aviation Industry

Modern aircrafts are increasingly reliant on automation for safe and efficient operation. This has resulted in increased passenger comfort, improved flight path control and reduced weather minima. The coupling of systems monitoring displays and diagnostic assistance systems have also provided enhanced understanding of aircraft system states for pilots and maintenance staff. Other benefits include increased safety, increased consistency and reliability of service, increased interconnectivity between sectors, increased resilience of operation, reduced environmental impact and reduced costs.

However, it has been stated that while automation reduces workload and helps pilots by providing them with better information management, it also increases their workload when faced with a complex failure event and has the potential to cause significant incidents when misunderstood or mishandled. A relatively simple to understand failure information can actually hinder diagnosis and distract the crew from efficiently flying the aircraft. This may result in an aircraft developing an undesirable state from which it is difficult or impossible to recover using traditional hand flying techniques. Major accidents show that even expert pilots have trouble coping with failure information.

Air France Flight 447 (AF447) was a scheduled flight from Rio de Janeiro, Brazil to Paris, France, which crashed in June 2009 [33]. The BEA's [5] final report concluded that temporary inconsistencies between the airspeed measurements (likely due to the aircraft's pitot tubes being obstructed by ice crystals) caused the autopilot to disconnect. It was determined that the crew reacted incorrectly and ultimately caused the aircraft to enter an aerodynamic stall, from which it did not recover. It has been pointed out that the crew made inappropriate control inputs that destabilised the flight path, were late in identifying and correcting the deviation from the flight path, failed to recognise that the aircraft had stalled and consequently did not make inputs that would have made it possible to recover from the stall. The plane crashed into the Atlantic Ocean, killing all 228 passengers, air crew and cabin crew aboard the aircraft.

Automation can relieve pilots from repetitive and non-rewarding tasks for which humans are ill-suited. Yet it invariably changes their active involvement into a monitoring role, which humans are particularly poor at doing effectively for long periods of time. Unanticipated situations requiring manual override of automation are difficult to understand and manage. They can create a surprise or startle effect, and can induce peaks of workload and stress. Unless the crew has been correctly trained to handle such situations, flight deck workload levels can reach the point where crew co-operation becomes severely challenged. When automation fails or disconnects, an alarm is triggered, yet no course of action is presented to the crew. Flight crews may spend too much time trying to understand the origin, conditions or causes of one or multiple alarms, distracting them from other priority tasks such as actually flying the aircraft.

In the long run, basic manual and cognitive flying skills can decline, due to lack of practice. Yet operators often actively discourage flight crews from flying aircrafts manually and even limit the manual modes they may use, which exacerbates automation dependency. This term describes a situation in which pilots are only fully confident in their ability to control the trajectory of an aircraft when using the full functionality of the automated systems. This dependence usually stems from a combination of inadequate knowledge of the automated systems and a lack of manual flying. For example, pilots who invariably fly with auto-throttle engaged can quickly lose the habit of scanning speed indications.

Automation dependency ultimately hinders safety. Affected pilots are reluctant to voluntarily reduce the extent to which they use full automation capability to deal with any

5. The Bureau of Enquiry and Analysis for Civil Aviation Safety (Bureau d'Enquêtes et d'Analyses pour la Sécurité de l'Aviation Civile) is an agency of the French government responsible for investigating accidents and incidents and making safety recommendations based on what is learned from those investigations.

routine or abnormal situation. When the full automation capability is no longer available or is considered incapable of delivering the required aircraft control, pilots tend to seek to partially retain the use of automated systems rather than revert to wholly manual aircraft trajectory control. This often leads to a loss of situation awareness triggered by task saturation. Situation awareness is defined as the continuous extraction of environmental information, the integration of this information with previous knowledge to form a coherent mental picture, and the use of that picture in directing further perception and anticipating future events. For a pilot, situational awareness means having a mental picture of the existing inter-relationship of location, flight conditions, configuration and energy state of the aircraft as well as any other factors that could be about to affect its safety. Such a mental picture would also be necessary for a human driver to take over in a timely manner.

Advanced technology designed to reduce workload and improve situation awareness in the aviation industry has created new challenges, notably complacency, automation dependency and lack of understanding. One can easily anticipate a similar situation arising from driving automation. Hence it seems unwise to rely on a human driver to act as a safety net to offset the lack of maturity of automated driving systems. That is not to say, however, that human drivers should not ultimately remain in full control of the vehicle in hazardous situations.

Qantas Flight 72 (QF72) was a scheduled flight from Singapore Changi Airport to Perth Airport in October 2008 that made an emergency landing at Learmonth airport in Western Australia following an inflight accident featuring a pair of sudden pitch-down manoeuvres that severely injured many of the passengers and crew [34]. The Australian Transport Safety Bureau investigation found fault with one of the aircraft's three air data inertial reference units and a previously unknown software design limitation of the Airbus A330's fly-by-wire flight control primary computer. This limitation meant that in a very rare and specific situation, multiple spikes in angle of attack data from one of the air data inertial reference units could result in a pitch down command from the flight control primary computer. The pilots had to fight the automated system for it would not allow them to counter this action. The system was showing stall and over-speed warnings at the same time and making commands that were imperilling all on board [35]. This event serves as a cautionary tale as society accelerates towards a world of automation and artificial intelligence. The behaviour of automated systems needs to be made explicit so that these faults can be prevented or eliminated during the design phase.

Ensuring the safety of Automated Driving Systems requires a multi-disciplinary approach across all levels of functional hierarchy. Yet, there is insufficient data to do so

and the methods and tools currently used by the automotive industry are lacking. They do not take into account the propagation of uncertainties that come from the perception components or the reliability of the world model that is used during the decision making process. Ultimately, creating an end-to-end design and deployment process that integrates safety concerns into a unified approach is a major challenge of driving automation [36].

1.3 Thesis Hypothesis

The following facts have been brought to light and discussed in the previous sections:

▷ Designing an Automated Driving System is a challenging task that requires a substantial jump in technological capabilities that has yet to be achieved. From both a safety and a security perspective, the proposed solutions to tackle these technical challenges are insufficient:

1. Wireless communication can be used to augment perceived information, but an Automated Driving System must ultimately be able to perform its functions independently and self-sufficiently regardless, as cyber-attacks can impact the availability and integrity of the information.
2. Human drivers cannot be used reliably as a safety net to offset the lack of maturity of Automated Driving Systems, as automation changes their active involvement into a monitoring role and creates new challenges, such as complacency, automation dependency, lack of understanding and misuse. This is all the more important given the following item.

▷ While Automated Driving Systems have the potential to address relevant transportation issues, their true capabilities remain unclear for the most part. They currently lack common sense and have trouble behaving in a truly cautionary manner. As a result, their ability to outperform human drivers cannot be established.

1. The definition of an architecture that could help establish the inherent safety of driving automation systems has not been proposed or made publicly available.
2. There is a lack of research on the subject of system-based approaches to safety in the context of driving automation.
3. Strategies allowing the system to achieve a minimal risk condition after occurrence of a performance-relevant system failure have yet to be defined.

The hypothesis of the thesis is as follows:

The safety of Automated Driving Systems can and should systematically be established through the use of tools and methods that provide clarification

> regarding their ability to perceive the driving environment, their ability to reason and make sound decisions and their ability to adapt to their environment in a controlled manner, in driving scenarios that range from mundane to life threatening.

This work places emphasis on the design of dependable Automated Driving Systems. In particular, the thesis addresses the problem of designing a new ADS primary subsystem, whose role it is to monitor the state of the ADS, supervise its actions and respond as needed to guarantee the safety of its occupants and of others.

1.4 Thesis Structure

In the following, the problem of designing an autonomous shuttle on a road whose characteristics are known is considered. The safety of human passengers is a primary design consideration and involves both careful evaluation of the system's ability to perceive the driving environment and of its behaviour.

Perception is a complex task which usually involves using many different sensors, based on multiple sensing modalities. Ideally, they should be positioned and oriented in such a way that most points in the vehicle's surroundings are observed by at least one sensor of each type. Not all sensors of the same family (camera, lidar, radar) are interchangeable. For example, one sensor configuration could make use of a Velodyne 3-D lidar, mounted on a raised platform on the roof. It performs a 360° sweep at 15 Hz and provides a dense point cloud. However, the sensor is expensive and it would be impractical to add a second one to the sensor configuration in case the first one is subject to a failure. Moreover, it is impossible to mount it in such a way that there are no blind spots. One can use less expensive two-dimensional 2-D SICK lidars to fill those blind spots and to provide some measure of fault tolerance in case the Velodyne fails. A single planar lidar cannot reliably differentiate between obstacles and non-flat terrain however. One advantage is that multiple, inexpensive planar lidars can be mounted at a variety of heights to allow appreciable changes in reliable obstacle signature. These 2-D lidars provide point clouds that are less dense. Experiments show that while the Velodyne tracks 95% of all obstacles, SICKs alone only track 61% of obstacles, meaning that while they do provide some measure of fault tolerance and additional features, they cannot on their own replace the Velodyne 3-D lidar [37]. The union of the two subsystems yields a minor but measurable improvement of 1%.

A camera can be added to the system, such as the one built by Mobileye. Cameras are generally more proficient at object classification than lidars. The Mobileye camera

comes with proprietary software, which provides vehicle lane and pedestrian detection technology. Delphi also proposes proprietary blackbox on-board processing to generate tracks in the form of object bearing, range and radial speed with its radars. Such software can be useful for rapid deployment, but the lack of information makes safety analysis difficult. Ultimately, the evaluation of the perception task remains a difficult subject. In order to provide fault tolerance, more trust in the system's perception task is required. It is therefore important to evaluate the impact of sensor failure and loss of computing power on the system's performance to provide necessary redundancy when designing the system.

It is assumed that the shuttle can sustain the entire dynamic driving task in its operational design domain. Nowadays, one expects that a human driver will intervene and respond appropriately to a system failure, though humans are ill-suited to do so. This work aims at designing a safety system which will respond to such a failure instead. Three contributions are made.

- Fault-tolerance is provided for the perception system so that operation can be maintained in case of a sensor failure, a loss of processing power, and in case of deteriorating weather conditions.
- In case the failures are too severe, a fallback strategy is provided so that the vehicle can be brought to a safe stop.
- Finally, hazardous situations are considered to make the safety system proactive in the context of daily commuting.

The remainder of the thesis is divided into six chapters:

Chapter 2 lays the foundation of the thesis. The taxonomy and definitions for terms related to driving automation are introduced. A survey of existing literature related to safety engineering is then presented. Related work on safety, as applied to driving automation systems is also provided. Finally, gaps are identified and the basis for the rest of the thesis is developed.

Chapter 3 introduces a new model for the design of concurrent stochastic real-time systems. Probabilistic time Petri nets (PTPN) are an extension of time Petri nets in which the output of tokens is randomised. A theorem establishes the fact that the schedulers of the PTPN induce the same Markov chains as the adversaries of the Markov decision process, derived from the atomic state class graph construction. This formal model is used as a basis for the following chapters.

Chapter 4 focuses on the ability of an Automated Driving System to perceive the driving environment. A formal approach for the design of multi-sensor data fusion systems that support adaptive graceful degradation through the smart use of sensor modalities is presented. A coloured probabilistic time Petri net is used to model known algorithms in a multi-sensor fusion scheme. The specification of safety requirements in terms of confidence levels conditions the outcome of the reachability analysis. The characteristics of a credible solution are then provided to the embedded safety module as support for online reconfiguration and decision making tasks. An example showing how the proposed approach can be of value for the design of Automated Driving Systems is then provided.

Chapter 5 focuses on the ability of an Automated Driving System to reason and to make sound decisions. It outlines the necessity of defining dynamic driving task fallback strategies that can be performed by an Automated Driving System, if and when necessary. The proposed fallback strategy is aimed at level 4 ADS features designed to operate a vehicle on a road whose characteristics make any attempt at stopping hazardous. The transition stage, during which the strategy is triggered, consists in the replacement of missing vehicles and obstacles in the world model with ghost objects. An embedded visibility map is then used to retrieve the maximum distance at which the ADS-operated vehicle can be seen, when driving behind it. The speed profile underlying the fallback strategy meets a time to collision criterion of 4 s, which enables the avoidance and the mitigation of rear-end collisions.

Chapter 6 focuses on the ability of an Automated Driving System to adapt to its environment in a controlled manner. It outlines a method that aims to give these systems the means to identify hazardous areas in the road network so that they can change their behaviour accordingly. Intelligent agents are tasked with the evaluation of the system's performance during operation and record any violation of safety constraints during periodically recurring travels. If the ADS is confronted to several hazardous situations in a given area of the road network, specific rules are applied to reduce the severity and the likelihood of occurrence of similar situations during later trips.

Chapter 7 presents concluding remarks on the thesis, including potential areas of future work.

Chapter 2

Automated Driving Systems and Dependability

2.1 Introduction

This chapter presents the terminology and concepts around which this thesis is built. Terms related to driving automation and system dependability are presented first. A state of the art on safety for driving automation systems is then provided.

2.2 Driving Automation

As stated in the previous chapter, there is much to be gained by providing clarity and stability when communicating on the topic of driving automation. For example, the role of the human driver during driving automation system engagement requires clarification. While car manufacturers often present driving automation levels from a user-centric point of view, safety requires a more comprehensive description of the system's role and capabilities.

The three main documents that describe levels of driving automation are:

- ▷ The National Highway Traffic Safety Administration (NHTSA)'s 'Preliminary Statement of Policy Concerning Automated Vehicles' (May 2013) [38],
- ▷ The Federal Highway Research Institute's (Bundesanstalt für Strassenwesen, a.k.a. BASt) 'Legal consequences of an increase in vehicle automation' (July 2013) [39], and
- ▷ The SAE (Society of Automotive Engineers) International's J3016 'Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles' (issued in January 2014 and revised in September 2016) [3].

The International Organisation of Motor Vehicle Manufacturers (Organisation Internationale des Constructeurs d'Automobiles, a.k.a. OICA) adopted the BASt levels and aligned them with the J3016 taxonomy.

BASt and SAE levels differ fundamentally from the levels described by the National Highway Traffic Safety Administration. NHTSA's levels were intended to provide preliminary policy guidance to U.S. state and local governments contemplating legislation and regulation related to driving automation. As such, NHTSA's level descriptions are written in loosely descriptive terms using normative language. They do not provide the degree of definitional and functional clarity that is ultimately required to support the technical and policy discussions that lead to standards, norms and legal requirements. All definitions and examples used in this section have been extracted from the Recommended Practice provided by SAE International.

2.2.1 Driver Effort

The act of driving can be divided into three types of driver effort [40]:

- Strategic effort involves trip planning, such as deciding whether, when and where to go, how to travel, best routes to take, etc.
- Tactical effort involves manoeuvring the vehicle in traffic during a trip and includes deciding whether and when to overtake another vehicle or change lanes, selecting an appropriate speed, checking mirrors, etc.
- Operational effort involves split-second reactions that can be considered pre-cognitive or innate, such as making micro-corrections to steering, braking and accelerating to maintain lane position in traffic or to avoid a sudden obstacle or hazardous event in the vehicle's pathway.

Dynamic Driving Task (DDT) The term dynamic driving task is used to describe all of the real-time operational and tactical functions required to operate a vehicle in on-road traffic. It excludes the strategic functions such as trip scheduling and selection of waypoints (Fig. 2.1). It includes, without limitation, the following:

1. lateral vehicle motion control via steering,
2. longitudinal vehicle motion control via acceleration and deceleration,
3. monitoring the driving environment via object and event detection, recognition, classification and response preparation,
4. object and event response execution,

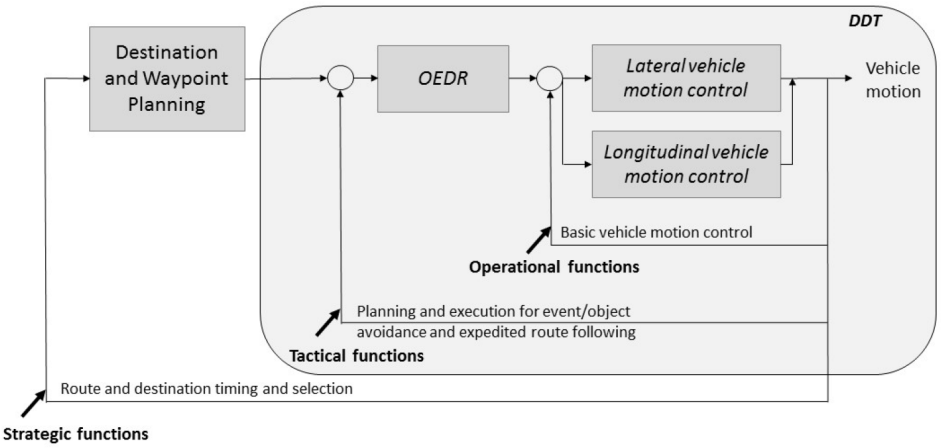


Figure 2.1 – Schematic view of driving task showing DDT portion [3]

5. manoeuvre planning, and
6. enhancing conspicuity via lighting, signalling and gesturing.

For simplification and to provide a useful shorthand term, subtasks (3) and (4) are referred to collectively as object and event detection and response (OEDR).

External events are situations in the driving environment that necessitate a response by a driver or driving automation system. A human driver or driving automation system is said to perform the DDT on a *sustained* basis if it performs part or all of the DDT both between and across external events, and if it responds to such events. Driving automation specifically refers to the performance of part or all of the DDT on a sustained basis.

2.2.2 Automated Driving Systems

Automated Driving System (ADS) The Recommended Practice provides a taxonomy for motor vehicle driving automation systems that perform part or all of the dynamic driving task (DDT) on a sustained basis. They range (Fig: 2.2) from no driving automation (level 0) to full driving automation (level 5).

For purposes of DDT performance, level 1 encompasses automation of part of the innermost loop (i.e., either lateral vehicle motion control functionality or longitudinal vehicle motion control functionality and limited OEDR associated with the given axis of vehicle motion control); level 2 encompasses automation of the innermost loop (lateral and longitudinal vehicle motion control and limited OEDR associated with vehicle motion

control), and levels 3-5 encompass automation of both inner loops (lateral and longitudinal vehicle motion control and complete OEDR). Note that DDT performance does not include strategic aspects of driving.

- A driving automation system refers to the hardware and software that are collectively capable of performing *part or all* of the DDT on a sustained basis. This term is used generically to describe any system capable of level 1-5 driving automation.
- An Automated Driving System refers to the hardware and software that are collectively capable of performing the *entire* DDT on a sustained basis. This term is used specifically to describe a level 3, 4 or 5 driving automation system [1].

An ADS-dedicated vehicle (ADS-DV) is a vehicle designed to be operated exclusively by a level 4 or level 5 ADS for all trips. This is the only category of ADS-operated vehicle that requires neither a conventional nor remote driver during routine operation. An ADS-DV might be designed without user interfaces, such as braking, accelerating, steering and transmission gear selection input devices designed to be operable by a human driver. An ADS-DV is a truly driverless vehicle [2].

The Recommended Practice recommends against using terms that make vehicles, rather than driving, the object of automation, because doing so tends to lead to confusion between vehicles that can be operated by a human driver or by an ADS and ADS-DVs, which are designed to be operated exclusively by an ADS. It also fails to distinguish other forms of vehicular automation that do not involve automating part or all of the DDT. Some vernacular usages associate autonomous specifically with full driving automation (level 5), while other usages apply it to all levels of driving automation, and some state legislation has defined it to correspond approximately to any ADS at or above level 3 (or to any vehicle equipped with such an ADS).

Operational Design Domain (ODD) The operational design domain refers to the specific conditions under which a given driving automation system or feature thereof is designed to function. An ODD may include geographic, roadway, environmental, traffic, speed, and temporal limitations. For example, a given ADS may be designed to operate

1. Given the similarity between the generic term "driving automation system" and the level 3-5 specific term ‘Automated Driving System’, the latter term is capitalised when spelled out and reduced to its acronym as much as possible, while the former is not.

2. The term ‘driverless vehicle’ is not used by SAE International because it has been, and continues to be widely misused to refer to any vehicle equipped with a driving automation system, even if that system is not capable of always performing the entire DDT and thus involves a human driver for part of a given trip.

Level	Name	Narrative definition	DDT		DDT fallback	ODD
			Sustained lateral and longitudinal vehicle motion control	OEDR		
Driver* performs part or all of the *DDT						
0	**No Driving Automation**	The performance by the *driver* of the entire *DDT*, even when enhanced by *active safety systems*.	*Driver*	*Driver*	*Driver*	n/a
1	**Driver Assistance**	The *sustained* and *ODD*-specific execution by a *driving automation system* of either the *lateral* or the *longitudinal vehicle motion control* subtask of the DDT (but not both simultaneously) with the expectation that the *driver* performs the remainder of the *DDT*.	*Driver* and *System*	*Driver*	*Driver*	Limited
2	**Partial Driving Automation**	The *sustained* and *ODD*-specific execution by a *driving automation system* of both the *lateral* and *longitudinal vehicle motion control* subtasks of the *DDT* with the expectation that the *driver* completes the *OEDR* subtask and *supervises* the *driving automation system*.	***System***	*Driver*	*Driver*	Limited
***ADS* ("*System*") performs the entire *DDT* (while engaged)**			*System*	***System***	*Fallback-ready user (becomes the driver during fallback)*	Limited
3	**Conditional Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire DDT with the expectation that the *DDT fallback-ready user* is *receptive* to *ADS*-issued *requests to intervene*, as well as to *DDT performance-relevant system failures* in other *vehicle* systems, and will respond appropriately.				
4	**High Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	***System***	Limited
5	**Full Driving Automation**	The *sustained* and unconditional (i.e., not *ODD*-specific) performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	*System*	**Unlimited**

Figure 2.2 – Summary of levels of driving automation [3]

only within a geographically-defined military base, only under 30 km/h, and only in daylight.

A driving mode is a type of vehicle operation with characteristic DDT requirements (expressway merging, high-speed cruising, low-speed traffic jam). An ODD may include one or more driving modes. For example, a given ADS may be designed to operate a vehicle only on fully access-controlled freeways and in low-speed traffic, high-speed traffic, or in both of these driving modes.

- Usage specification refers to a particular level of driving automation within a particular ODD. A driving automation system feature is a driving automation system's design-specific functionality at a specific level of driving automation within a particular ODD. A given driving automation system may have multiple features, each associated with a particular level of driving automation and ODD. Each feature satisfies a usage specification.
- A given vehicle may be equipped with a driving automation system that is capable of delivering multiple driving automation features that operate at different levels; thus, the level of driving automation exhibited in any given instance is determined by the feature(s) engaged. As such, the recommended usage for describing a vehicle with driving automation capability is 'level [1 or 2] driving automation system-equipped vehicle' or 'level [3, 4, or 5] ADS-equipped vehicle.' The recommended usage for describing a vehicle with an engaged system (vs. one that is merely available) is 'level [1 or 2] driving automation system-engaged vehicles' or 'level [3, 4, or 5] ADS-operated vehicle.'

Dynamic Driving Task (DDT) Fallback A DDT performance-relevant system failure is a malfunction in a driving automation system or in other vehicle systems that prevents the driving automation system from reliably sustaining DDT performance (either partially or completely). This definition applies to vehicle fault conditions and driving automation system failures that prevent a driving automation system from performing at full capability according to design intention. A minimal risk condition is a condition to which a user or an ADS may bring a vehicle in order to reduce the risk of a crash when a given trip cannot or should not be completed. The dynamic driving task fallback is the response by the user or by an ADS to either perform the DDT or achieve a minimal risk condition after occurrence of a DDT performance-relevant system failure(s) or upon ODD exit.

2.2.3 Human Drivers and Monitoring

The term user references the human role in driving automation. Sustained performance of part or all of the DDT by a driving automation system changes the user's role. The levels of driving automation are defined by reference to the specific role played by the three primary actors in performance of the DDT: the human driver, the driving automation system, and other vehicle systems and components. A human driver is a user who performs in real-time part or all of the DDT or DDT fallback for a particular vehicle. A distinction can be made between a conventional driver, who manually exercises in-vehicle braking, accelerating, steering and transmission gear selection input devices in order to operate a vehicle, and a remote driver, who is not seated in a position to manually exercise these tasks but remains able to operate the vehicle. The operation of a motor vehicle refers, collectively, to the activities performed by a human driver (with or without support from one or more level 1 or 2 driving automation features) or by an ADS (level 3-5) to perform the entire DDT for a given vehicle during a trip.

Monitoring Monitoring is a general term referencing a range of functions involving real-time human or machine sensing and processing of data used to operate a vehicle, or to support its operation. Receptivity is an aspect of consciousness characterised by a person's ability to reliably and appropriately focus his/her attention in response to a stimulus. Monitoring includes receptivity. The driver state or condition of being receptive to alerts or other indicators of a DDT performance-relevant system failure, as assumed in level 3, is not a form of monitoring. Supervision refers to driver activities, performed while operating a vehicle with an engaged level 1 or 2 driving automation system, to monitor the driving automation system's performance, respond to inappropriate actions taken by that system, and to otherwise complete the DDT.

- User monitoring refers to the activities or automated routines designed to assess whether and to what degree the user is performing the role specified for him/her. User monitoring in the context of driving automation is most likely to be deployed as a countermeasure for misuse or abuse (including over-reliance due to complacency) of a driving automation system, but may also be used for other purposes. User monitoring is primarily useful for levels 2 and 3, as below these levels evidence from the field has not identified significant incidence of misuse or abuse of driving automation technology, and above these levels the ADS is always capable of achieving a minimal risk condition automatically, so user misuse/abuse is not relevant.
- Driving environment monitoring refers to the activities or automated routines that accomplish real-time roadway environmental object and event detection, recognition, classification, and response preparation (excluding actual response), as needed

to operate a vehicle. When operating conventional vehicles that are not equipped with an engaged ADS, drivers visually sample the road scene sufficiently to competently perform the DDT while also performing secondary tasks that require short periods of eyes-off-road time (e.g., adjusting cabin comfort settings, scanning road signs, tuning a radio, etc.). Thus, monitoring the driving environment does not necessarily entail continuous eyes-on-road time by the driver.

- Vehicle performance monitoring refers to the activities or automated routines that accomplish real-time evaluation of the vehicle performance, and response preparation, as needed to operate a vehicle. While performing the DDT, level 4 and 5 ADSs monitor vehicle performance. However, for level 3 ADSs, as well as for level 1 and 2 driving automation systems, the human driver is assumed to be receptive to vehicle conditions that adversely affect performance of the DDT.
- Driving automation system performance monitoring refers to the activities or automated routines for evaluating whether the driving automation system is performing part or all of the DDT appropriately [3]. Recognising requests to intervene issued by a driving automation system is not a form of monitoring driving automation system performance, but rather a form of receptivity.

2.2.4 Comprehensive Examples

An automated intervention that is not sustained does not qualify as driving automation. Hence systems that provide momentary intervention in lateral or longitudinal vehicle motion control but do not perform any part of the DDT on a sustained basis (e.g., anti-lock brake systems, electronic stability control, automated emergency braking) are not classifiable (other than at level 0) under the J3016 taxonomy.

The following examples are intended to summarise and illustrate the SAE levels of driving automation:

Level 1 (Driver Assistance) refers to the sustained and ODD-specific execution by a driving automation system of either the lateral or the longitudinal vehicle motion control subtask of the DDT (but not both simultaneously) with the expectation that the driver performs the remainder of the DDT.

- A conventional driver is expected to verify that an engaged ACC system is maintaining an appropriate gap while following a preceding vehicle in a curve. If the

3. The term 'monitor driving automation system performance' should not be used instead of supervise, which includes both monitoring and responding as needed to perform the DDT.

system is not maintaining headway to a preceding vehicle, the driver must brake accordingly.

- If a level 1 adaptive cruise control (ACC) feature experiences a system failure that causes the feature to stop performing its intended function, the human driver must perform the DDT fallback by resuming performance of the complete DDT.

Level 2 (Partial Driving Automation) refers to the sustained and ODD-specific execution by a driving automation system of both the lateral and longitudinal vehicle motion control subtasks of the DDT with the expectation that the driver completes the OEDR subtask and supervises the driving automation system.

- A remote driver engaging a level 2 automated parking feature must monitor the pathway of the vehicle to ensure that it is free of pedestrians and obstacles.
- At level 1 and level 2, the human driver is expected to be receptive to evident vehicle system failures, such as a broken tie rod.

Level 3 (Conditional Driving Automation) refers to the sustained and ODD-specific performance by an ADS of the entire DDT with the expectation that the DDT fallback-ready user is receptive to ADS-issued requests to intervene, as well as to DDT performance-relevant system failures in other vehicle systems and will respond appropriately. At level 3, an ADS is capable of continuing to perform the DDT for at least several seconds after providing the fallback-ready user with a request to intervene. A DDT fallback-ready user is considered to be receptive to a request to intervene or to an evident vehicle system failure, whether or not the ADS issues a request to intervene as a result of such a vehicle system failure. He/she is then expected to achieve a minimal risk condition if he/she determines it to be necessary.

- Let us assume a level 3 ADS is performing the DDT in stop-and-go traffic when the left-front tie rod breaks. The DDT fallback-ready user feels that the vehicle has pulled dramatically to the left and must intervene in order to move the vehicle onto the road shoulder.
- If a vehicle with an engaged level 3 ADS experiences a broken tie rod, it causes the vehicle to handle very poorly giving the fallback-ready user ample kinaesthetic feedback indicating a vehicle malfunction necessitates intervention. The fallback-ready user must respond by resuming the DDT, turning on the hazard lamps, and pulling the vehicle onto the closest road shoulder, thereby achieving a minimal risk condition.
- If a level 3 ADS experiences a DDT performance-relevant system failure in one of its exteroceptive sensors, it can prevent it from reliably detecting objects in the

vehicle's pathway. If the ADS issues a request to intervene to the DDT fallback-ready user, the ADS must continue to perform the DDT, while reducing vehicle speed for several seconds to allow time for the DDT fallback-ready user to resume operation of the vehicle in an orderly manner.

- Let us assume a level 3 ADS is performing the DDT on a free-flowing highway when the left side mirror glass falls out of the housing. The DDT fallback-ready user, while receptive, is not expected to notice this failure because it is not apparent.

Level 4 (High Driving Automation) refers to the sustained and ODD-specific performance by an ADS of the entire DDT and DDT fallback without any expectation that a user will respond to a request to intervene.

- If a level 4 ADS feature designed to operate a vehicle at high speeds on freeways experiences a DDT performance-relevant system failure, it must automatically remove the vehicle from the active lane of traffic before coming to a stop.
- If a level 4 ADS feature designed to operate a vehicle at high speeds on freeways receives a request by a passenger to stop, it must automatically remove the vehicle from the active lane of traffic before coming to a stop.
- If a level 4 ADS experiences a DDT performance-relevant system failure in one of its computing modules, it must transition to DDT fallback by engaging a redundant computing module to achieve a minimal risk condition.
- Let us assume a level 4 ADS is engaged in stop-and-go traffic when a malfunctioning brake caliper causes the vehicle to pull to the left when the brakes are applied. The ADS must recognise this deviation, correct the vehicle's lateral position and transition to a limp-home mode until it achieves a minimal risk condition.
- A level 4 ADS feature designed to operate a vehicle in a high-speed convoy with small gaps between vehicles may delay relinquishing performance of the DDT to a user upon his or her request to resume driving until after the ADS has safely manoeuvred the vehicle out of the convoy, since human drivers may not be capable of safely operating a vehicle in a close-coupled convoy.

Level 5 (Full Driving Automation) refers to the sustained and unconditional (i.e. not ODD-specific) performance by an ADS of the entire DDT and DDT fallback without any expectation that a user will respond to a request to intervene.

In the following, the focus is set on level 4 and 5 equipped or dedicated Automated Driving Systems.

2.3 Dependability

2.3.1 Terminology

Dependability is a system property that enables its users to justify placing trust in the service that it provides [41]. The term is often used in quality control, yet its significance goes beyond the user's sole expectations. In essence, one must consider all that the system can be, without any restriction with regards to what one would like it to be. In particular, one would be deluded in assuming a system is dependable because of its mere compliance with requirements and specifications. A thorough understanding of the way the system behaves and interacts with its environment is needed before it can truly be referred to as a dependable system. This is precisely what safety engineering aspires to acquire, as it commits itself to foreseeing and foretelling hazardous situations.

However, safety analysis is a costly process that often leads to compromises, for the system must both be dependable and affordable. Risk assessment is used to evaluate, for any given undesirable event, the likelihood that a loss will occur and the severity of the potential loss. The choice of a unit of time and of a severity scale is not without significance as it must accommodate the point of view of those towards which the evaluation is targeted. In practice, a risk is deemed acceptable if one understands and tolerates the fact that the cost or difficulty of implementing an effective countermeasure for the identified vulnerability exceeds the expectation of loss.

There exists a tacit correspondence between a rare but severe event and a benign but frequent one, which is introduced through the notion of criticality, defined as the product of the severity and probability of an unwanted event. It is assumed that two potential risks with the same criticality are equally acceptable. The Farmer diagram (Fig: 2.3) is a frequency-severity diagram that illustrates this hypothesis, by combining events of similar criticality on hyperbola branches called isorisk curves. A criticality threshold defines the boundary between the domain of acceptable risks and the domain of inacceptable risks. In order to deal with high criticality risk, one can either attempt to reduce its frequency (prevention) or its severity (protection). Finally, a system is said to have a safe behaviour if the frequency of failure occurrences is maintained below a threshold given the severity of their consequences.

The dependability tree (Fig: 2.4) depicts attributes, impairments and means as components of dependability.

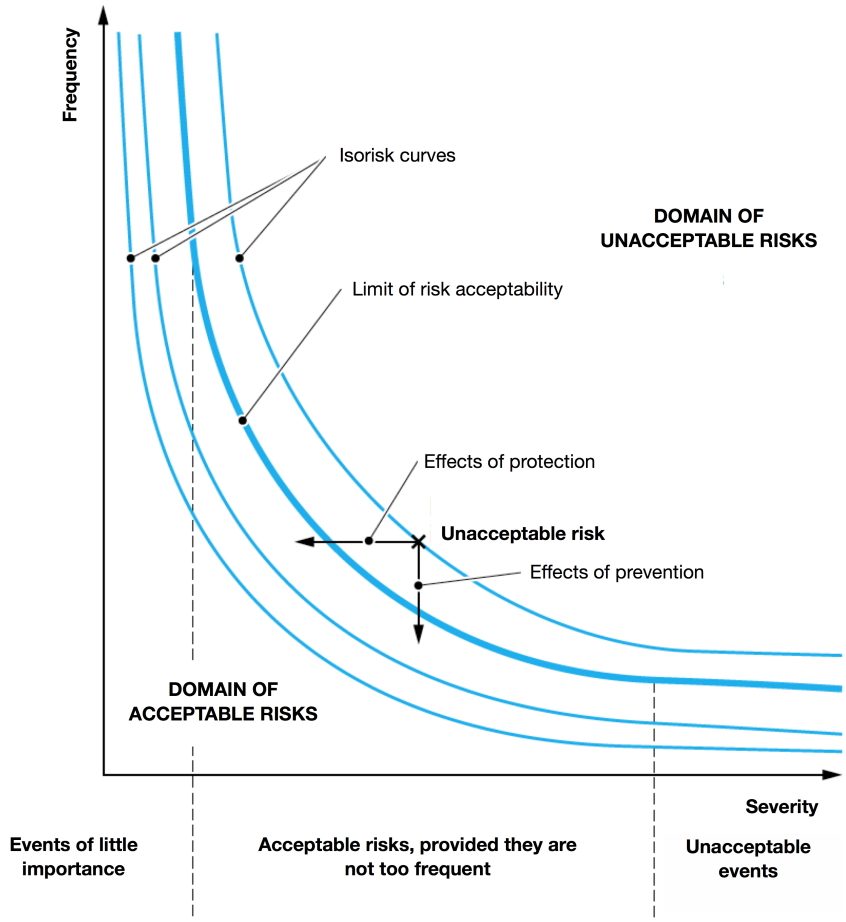


Figure 2.3 – Farmer diagram [4]

Attributes

Dependability is often presented as a measure of a system's availability, reliability, maintainability, confidentiality, integrity and safety. They define its primary attributes.

- Availability is a measure of the delivery of proper service with respect to the alternation of proper and improper service. Availability is defined [42] as the ability of an item to be in a state to perform a required function at a given instant of time or at any instant of time within a given time interval, assuming that the external resources, if required, are provided.
- Reliability is a measure of the continuous delivery of proper service (where service is delivered according to specified conditions) or equivalently of the time to failure. Reliability is defined as the ability of an item to perform a required function under given conditions for a given time interval.
- Safety is a measure of the continuous delivery of service free from occurrences of catastrophic consequences for the system's environment. Safety can be defined as

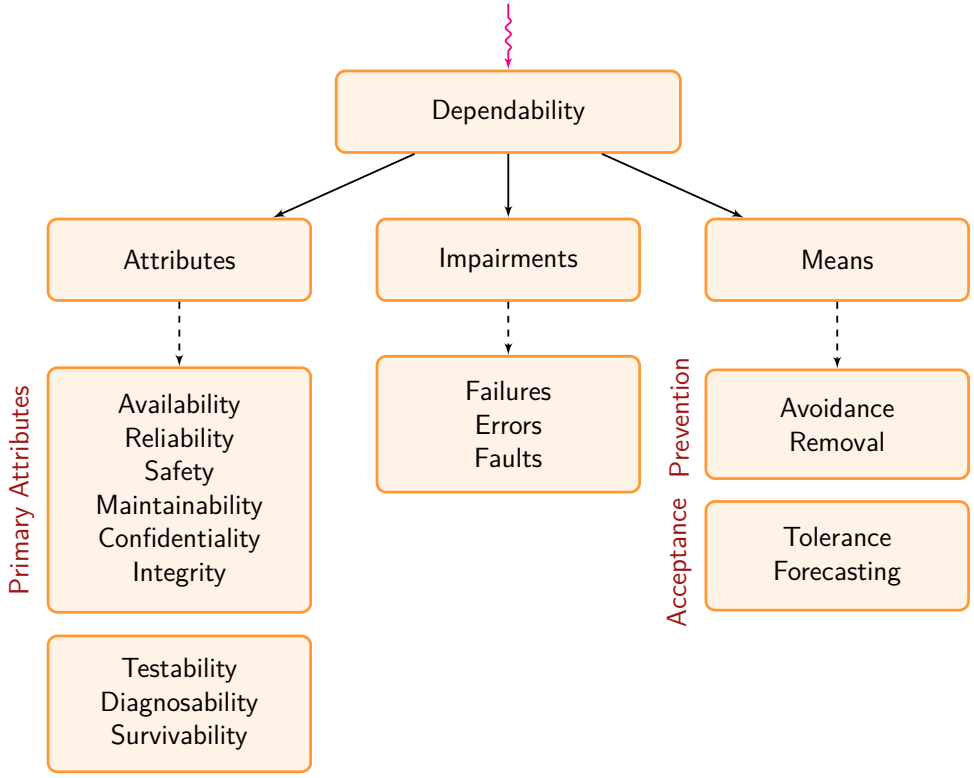


Figure 2.4 – The dependability tree

the absence of accidents (or mishaps), where an accident is defined as an unplanned event involving an unplanned and unacceptable loss (death, injury, illness, damage to or loss of property or environmental harm) [43]. System failures are defined in terms of system services while safety is defined in terms of external consequences.

- Maintainability is a measure of the ability of the system to undergo repairs and evolutions. Maintainability is defined as the probability that a given active maintenance action, for an item under given conditions of use can be carried out within a stated time interval, when the maintenance is performed under stated conditions and using stated procedures and resources.
- Confidentiality is a measure of the continuous delivery of service free from occurrences of unauthorised information disclosure.
- Integrity is a measure of the continuous delivery of service free from improper information alteration. Integrity involves maintaining the consistency, accuracy, and trustworthiness of data over its entire life cycle.
- Security is sometimes defined as the combination of confidentiality, integrity and availability. These three attributes are also known as the CIA triad [44], which is a model designed to guide policies for information security within an organisation.

The following attributes describe other properties that contribute to system depend-

ability:

- Testability is a measure of the ability of the system or a component of the system to provide information about its state and performance.
- Diagnosability is a measure of the ability of a system to display symptoms when impaired.
- Survivability is a measure of the ability of a system to proceed with its mission following human or environmental disturbances.

Impairments

Impairments to dependability encompass failures, faults and errors.

- Failures are the result of an unfulfilled system requirement. Any departure from the service that is required of a system constitutes a failure. Failures are attributed to underlying causes called errors.
- An error is that part of the system that has been damaged by a fault and can lead to a failure. A system can contain a latent error that only becomes effective later on, when it affects the service and the failure occurs.
- Faults can be software or hardware related, internal or external, permanent or temporary, malicious or not, accidental, intentional or the sheer result of incompetence. Mistakes in specifications or design, component failures, improper operation and environmental anomalies are examples of faults.

Means

There are many methods and techniques that one can use to make a system more dependable. In practice, one either mitigates the effects of faults or tries to prevent them altogether [45].

- Fault avoidance attempts to prevent fault occurrence or introduction by construction.
- Fault tolerance attempts to provide a service complying with specification in spite of faults by redundancy. Fault tolerance is concerned with detecting latent errors before they become effective and replacing the erroneous component with an error-free version [46].
- Fault removal attempts to minimise the presence of faults by verification.
- Fault forecasting attempts to estimate the presence, the creation and the consequences of faults by evaluation.

Common misconceptions

Safety vs. Security While both safety and security involve loss prevention, security deals with losses resulting from intentional actions caused by malevolent actors whereas safety handles losses due to unintentional actions caused by benevolent ones. The key difference lies in the motives of those actors that interact with the system.

Hazards vs. Failures Hazards refer to states of the system that will inevitably lead to an accident (or loss event) when certain conditions in the environment of the system are met. This definition is contingent on the stipulated system boundaries. The severity (or damage) of a hazard is assimilated to the most unfavourable state of the environment and thus pertains to the worst possible accident it could lead to. Danger is the probability of the hazard leading to an accident. In order to prioritise the effort used to deal with hazards within the engineering design space, most refer to the hazard level, which is a combination of the severity and the likelihood of occurrence of a hazard [47].

Failures occur when the specified requirements are not met, meaning the system or one of its component is unable to perform its intended function for a specified time, under specified environmental conditions. Hazards can be caused by failures but this is not always the case. Indeed, the fact that all system components function exactly as specified isn't proof that an accident cannot occur. Likewise, not all failures lead to hazards.

Safety vs. Reliability Safety is a system property whereas reliability is a component one. Reliability is defined as the probability that a component satisfies its specified behavioural requirements over time and under given conditions. Increasing reliability can decrease safety and vice versa. One increases reliability by preventing failures and increases safety by preventing hazards.

Verification vs. validation Verification refers to the evaluation of the compliance of a product, service or system with regards to a regulation, requirement, specification or imposed condition. Validation refers to the assurance that a product, service or system meets the needs of the customer.

2.3.2 Safety Engineering

Analysis techniques used in safety engineering can be split into two categories: qualitative methods and quantitative methods. Both approaches share the common goal of finding causal dependencies between a hazard on a system level and failures of individual

components. Qualitative approaches focus on the question 'What must go wrong, such that a system hazard may occur?', while quantitative methods aim at providing estimations about probabilities, rates and severity of consequences. The quality of these methods is contingent on the skill and expertise of the safety engineer that conducts them.

Traditional approaches

Traditional safety strategies can be inductive or deductive, quantitative or qualitative. Inductive methods are applied to determine what system states (usually failed states) are possible. In an inductive approach, one assumes some possible component condition or initiating event and tries to determine the corresponding effect on the overall system. Deductive methods are applied to determine how a given system state (usually a failed state) can occur. In a deductive system analysis, one postulates that the system itself has failed a certain way and attempts to find out what modes of system or component behaviour contribute to this failure. This subsection reviews some of the most common strategies used in safety engineering.

A preliminary hazard analysis (PHA) is often the first step of a safety analysis. This approach is performed in order to produce a list of potential hazards and accidental events that must be studied. Potential hazards and accidental events that may lead to an accident are identified and ranked according to their severity. Those deemed minor are disregarded. Required hazard controls and follow-up actions are then identified.

The failure mode and effects analysis (FMEA) is a prevalent inductive method in safety engineering for the identification of failures in a design, a manufacturing or assembly process, or a product or service. It was developed in the late 1940s to study problems that might arise from malfunctions in military systems. The analysis is qualitative and may be performed at either the functional or piece-part level. Failure modes are identified for each system component and the effects they have on the system as a whole are then determined. The failure modes and effects analysis also documents current knowledge and actions about the risks of failures, for use in continuous improvement.

The failure modes, effects and criticality analysis (FMECA) extends the FMEA by including a criticality analysis, which is used to chart the probability of failure modes against the severity of their consequences [48]. Failures are prioritised according to how serious their consequences are, how frequently they occur and how easily they can be detected. The purpose of the FMECA is to take actions to eliminate or reduce failures, starting with the highest-priority ones. The result highlights failure modes with relatively high probability and severity of consequences, allowing remedial effort to be directed where it will produce the greatest value [49]. The analysis is both quantitative and qualitative.

The fault hazard analysis (FHA) is a deductive method that can be used exclusively as a qualitative analysis. The FHA requires a detailed investigation of the subsystems to determine component failure modes, the causes of these failures, and the effects they have on the system and its operation. Safety is assured by identifying hazards and tracing them to the components. The fault hazard analysis requires that the system be divided into modules. The significant failure mechanisms that could occur and affect components are determined. The failure modes of individual components that would lead to the various possible failure mechanisms of the subsystem are then identified.

The event tree analysis (ETA) is an inductive, quantitative analysis that was developed in the 1980s. Sequences of events arising from a single hazard or event are considered and the seriousness of their outcome is described in terms of probabilities. They can easily be assessed if the probability of each alternative at the tree nodes can be determined.

The fault tree analysis (FTA) is a top-down, deductive analysis in which an undesired state of a system is analysed using Boolean logic in order to combine a series of lower-level events. This analysis method was originally developed to evaluate the Minuteman I Intercontinental Ballistic Missile (ICBM) Launch Control System [50]. The FTA is mainly used in the aerospace, nuclear power, chemical, pharmaceutical and petrochemical industries. This analysis is also used in software engineering for debugging purposes and is closely related to the cause-elimination technique used to detect bugs. At the top of the tree is a hazard event or a serious consequence. The paths represent different combinations of intermediate causes described by logical operators at the tree nodes. A minimal set of failures that can lead to the hazard event is called a minimal cut. A common requirement is that no single point of failure should lead to a hazard.

The hazards and operability analysis (HAZOP) is a qualitative, structured and systematic examination of a planned or existing process or operation. It aims at identifying and evaluating problems that may represent risks to personnel or equipment, or prevent efficient operation.

Other methods and approaches include the failure propagation and transformation notation (FPTN), the interface analysis (IA), the double failure matrix (DFM), the management oversight and risk tree analysis (MORT), the state machine hazard analysis (SMHA) and the task and human error analysis (THEA). Risk and safety modelling in civil aviation includes causal models for risk and safety assessment (fault tree analysis, common cause analysis, event tree analysis, bow-tie analysis, TOPAZ accident risk assessment methodology uses scenario analysis, bayesian belief networks), collision risk models (Reich-Marks model, Machol-Reich model, intersection models, geometric conflict models, generalised Reich model), human error models (hazard and operability method, human

error assessment and reduction techniques, technique for the retrospective analysis of cognitive errors, human error in ATM approach, human factor analysis and classification system) and third-party risk models [51].

System-oriented approaches

Safety engineering techniques often focus on component failures, while studies show that hazardous situations emerge from component interactions. In contrast to traditional methods, model-based techniques try to derive relationships between causes and consequences from some model of the system.

The most important accidents in the world of high-risk technologies are the result of the interaction of multiple failures. The operators are often blamed for these accidents, yet knowing what they should have done is only possible after the accident occurs. The most effective models go beyond assigning blame and try to help the engineers learn about all the factors involved [52]. Accident models play a critical role in accident investigation and analysis. Each level of the socio-technical system involved in risk management is usually studied separately by a particular academic discipline. [53] argues that risk management must be modelled by cross-disciplinary studies and that this requires a system-oriented approach based on functional abstraction rather than structural decomposition. The functional resonance accident model (FRAM) is an example of an accident model that uses a systemic description of emergent phenomena [54].

Each level of a system hierarchy presents a number of emergent properties. Emergent properties arise through interactions among the components at lower levels. The interacting tendency of inevitable failures is called the interactive complexity of the system [55]. Safety being a system property, it must be analysed for the system as a whole. Most traditional safety strategies focus on component failures while most complications are evidence of a design flaw. Therefore, traditional design-based mitigation at the component system level cannot be relied upon. The operational policy that governs the safe integration and collective use of these systems must be the focus of attention [56]. The term 'systems of systems' is sometimes used but it has been pointed out that using that term may be misleading [57].

The STAMP-based process analysis (STPA) is an example of a system-oriented approach. This hazard analysis technique is based on STAMP (Systems-Theoretic Accident Model and Processes). The system is viewed as a collection of interacting loops of control. Hazards for the system are identified and top-level system safety constraints are determined. A basic control structure is then defined. Each control action is assessed for

potential contribution to hazards in the control structure diagram. Inadequate control actions are used to refine system safety constraints. How these potentially hazardous control actions is then determined so that recommendations are updated if necessary [58].

2.3.3 Formal Methods

The term 'formal methods' applies to a class of rigorous mathematical techniques, notations and tools that provide a guarantee of system correctness. They enable developers to design systems that operate reliably despite their complexity. Formal methods provide both precise mathematical tools to represent a system and its requirements (formal specification) and a repertoire of proof techniques to analyse the correctness of those models (formal verification). They are rated as highly recommended for the specification of systems and components with the higher levels of SIL [59]. Formal methods provide rigorous design methods and automatic validation techniques to find subtle bugs that go undetected during testing [60].

Formal specification

Specification refers to the precise description of what a system does or should do. Specifications can be informal, formalised or formal. Informal specification refers to free form, natural language. Specifications are often written in natural language, which can lead to ambiguities, inaccuracies, inconsistencies, misunderstandings and contradictions. Formalised specifications have a standardised syntax which enables basic consistency and completeness checks, but their imprecise semantics makes them prone to error. Formal specifications provide a concise description of the properties and behaviour of a system. An abstraction of the system is used to ignore irrelevant or complex details. The verifiable transformation of an abstract formal specification into a concrete executable program is called program refinement. Formal specifications are the basis for mathematically verifying the equivalence between specification and implementation.

Formal specifications are the translation of non-mathematical descriptions into a formal specification language. They provide a rigorously defined syntax and rigorous semantics which eliminate imprecision and ambiguities. Specification languages are expressive languages that provide general proof methods. Formal specifications can be property oriented or model oriented. Property oriented formal specifications state the desired properties in a purely declarative way. They can be algebraic or axiomatic. Model oriented formal specifications provide ways of describing the system's behaviour. They refer to abstract models and state machines. Describing how things work is easier than imagining

how they can go wrong. Using models to describe components and formal analysis to explore and calculate properties of their interaction combines the strengths of man and machine [61].

Model oriented formal specifications include automata, Petri nets and process algebras. The simplest of those models is probably the finite state machine, which is a variety of automata. This model consists of states (represented by circles) and transitions (represented by arrows). Petri nets are a directed bipartite graph, in which nodes represent transitions and places, which makes them especially interesting for the description of distributed systems. Process algebras are a family of models used to describe concurrent systems. If every program written in language A can also be written in language B, we say that language B is at least as expressive as language A. Process algebras are expressive models but they do not have the wide applicability that a field like automata theory does. The choice of one model over another depends on the application that is being considered, the expressiveness of the model, the complexity of its implementation as well as the underlying decidability properties.

Formal verification

Formal verification refers to the production of a mathematical proof in order to prove the correctness of a system. Program analysis is the automatic static determination of dynamic run-time properties of programs. Unlike dynamic program analysis which involves running the program, static program analysis is performed without actually executing it. Automatic program verification requires a choice between precision and efficiency. Given a program and a specification, a program analyser will check if the program semantics satisfies the specification. In case of failure, the analyser will provide hints to understand the origin of the error.

Program correctness proofs associate logical inference rules with programming syntax, while model based approaches provide algorithms for correctness proofs. Formal program verification methods attempt to prove that every program execution is correct in every specified environment. They include deductive methods and model-checking [62]. Deductive methods use an abstraction of the program semantics in order to obtain minimal verification conditions to prove system correctness. A theorem prover or a proof assistant is then used to check the verification conditions. An inductive argument must be provided by the user in general. The verification proof can only be partially automated and debugging an unsuccessful proof can be a complex task. Model-checking is an automated technique that, given a finite-state model of a system and a formal property, systematically checks whether this property holds for a given state in that model. Model checking

considers systems that have a finite number of states or can be viewed as such by abstraction. A model of the program is given and a specification of the program is provided by the user using expressive temporal logics. A model checker is then used to check the specification by exhaustive exploration of the state space. As the number of state variables in the system increases, the size of the system state space grows exponentially [63]. This is known as the state explosion problem and it is the most important shortcoming of model checking. Model checking is an effective technique to expose potential design errors that can provide a significant increase in the level of confidence of a system design.

Formal methods can assume various forms and levels of rigour. [64] classifies formal methods into four levels of increasing rigour. Level 0 corresponds to current standard practice. Verification is a manual process and documents are written in natural language. Validation is based on testing driven by requirements and specifications. At level 1, some of the natural language used in requirements statements and specifications are replaced with notations and concepts from logic and discrete mathematics. At level 2, formalised specification languages are used with some mechanised support tools. At level 3, formal specification languages with comprehensive support environments, including mechanised theorem proving and proof checking are used.

Benefits and fallibilities of formal methods

The adequacy of formal methods depends on the nature and the criticality of the application, on the stage of the lifecycle in which formal methods are introduced, on the degree of formality employed, on the quality of the method and tools that support it and on the experience and skill of the practitioners.

Formal methods are not commonplace, not even in safety-critical applications. They fail to attract enthusiasm from practicing software engineers and programmers which leads a number of people to believe they have little practical utility. Mathematical notations are both concise and compact. Formal specifications can therefore be difficult to read because the majority of people are accustomed to traditional informal methods. Unskilled practitioners believe the activity tends to be focused on the formalism and not on the real problem at hand. Development costs increase for some companies because they need to invest money for the training of their staff in formal methods technology. [65] discusses seven myths about formal methods that make them one of the more contentious techniques in industrial software engineering. The widespread belief that formal methods delay the development process, that they are not supported by tools, that they forsake traditional engineering design methods, that they only apply to software, that they are not required, that they are not supported and that the proponents of formal methods apply them in

all aspects of system development is disproved. Yet formal methods do have a number of perceived limitations. There are many notations, methods and tools originating from the academia. However, they lack industrial strength in terms of tool stability, documentation and user support. The choice among so many different formal methods proposals is not an easy task for a company. On the other hand, there are very few technically sound methods and tools coming from the industry itself. Formal methods suffer from intrinsic computational complexity and calculation in formal logic is harder than most numerical mathematics. Most computational problems are NP-hard, many are super-exponential and some are undecidable. Correctness proofs are therefore resource-intensive [61]. The behaviour of the finished product depends on physical processes whose behaviour might not be modelled with complete accuracy. Formal requirements statements might not capture real-world expectations accurately and completely [66]. As a result, formal methods are not a guarantee of success and proof should not be considered an alternative to testing.

According to [67], 'a formal approach still remains the only way in which we can meaningfully talk about correct systems'. Formal methods reduce the ambiguity and imprecision of natural language specifications, meaning the reasoning can be checked by others. Requirements and specifications are unambiguous and errors due to misunderstandings are reduced as a result. The validation of requirements and specifications becomes easier. Formal methods are attractive because they often reveal subtle bugs in well-tested programs. This is due to the fact that dynamic analysis only considers the executions that are actually performed whereas static analysis considers all possible executions. They confront the discrete behaviour of computer systems by using discrete mathematics to model it [64]. Implementations based on formal specifications are usually easier than those based on informal ones. Formal methods have been demonstrated to result in systems of the highest integrity, when applied correctly [67]. They can guarantee the absence of certain faults. Correctness proofs are a powerful approach to verifying implemented software systems, especially for safety-critical properties. They enable faults to be detected earlier and with greater certainty than would otherwise be the case. Some methods of static analysis can construct a counter-example when a specification violation is detected. This capability can be inverted to provide automated test case generation.

Industrial applications of formal methods

Formal methods have been used successfully in a wide variety of applications such as aviation, railway systems, nuclear power plants, medical systems, ammunition control and embedded microprocessors [45]. [68] describes the state of the art in the industrial use of formal methods at the earlier stages of specification and design. The paper examines the

industrial application gathered in a review of sixty-two projects, the largest application domains being transport, the financial sector, defence, telecommunications and administration. It presents what appears to be a growing interest in the industrial applications and concludes with observations on the vitality of formal methods and the maturity of its underlying theory and supporting tools. Examples of industrial usage of formal methods include the design of a storm surge barrier control system, the scheduling and rescheduling of trains, the formalisation of the safety requirements for an ammunition control system replacement, the development of an ATC information system, the development of a radiation therapy machine control program, the improvement of the quality of a medical instruments control system, the development of a trustworthy network security device and the formal specification of the central control function display and information system facility at the London air traffic control center [66, 69]. Engineers at Amazon Web Services have also been using formal specification and model checking successfully to help solve difficult design problems in critical systems [70]. Experience has shown that the use of informal techniques in communication protocol development generally produces systems with errors and undesirable behaviours. On the other hand, formal methods have enabled the development of highly reliable and easily maintainable communication protocols [71].

2.4 Related Work

While the last few years have brought an increased interest in the safety challenges of driving automation, research on the subject remains but a very small fraction of the work that has been published in the field of intelligent transportation systems. This section reviews current work related to safety for different levels of driving automation.

A variety of driver assistance systems have been developed and made available by automotive manufacturers to automate mundane driving operations, reduce driver burden and thus reduce accidents. They include active safety systems, which are vehicle systems that sense and monitor conditions inside and outside of the vehicle for the purpose of identifying perceived present and potential dangers to the vehicle, occupants, or other road users. They automatically intervene to help avoid or mitigate potential collisions via various methods, including alerts to the driver, vehicle adjustments, or active control of the vehicle subsystems. Examples of such driver assistance systems include [72]:

- Collision avoidance systems [73], which automatically detect slower moving preceding vehicles and provide warning and brake assist to the driver,
- Adaptive cruise control systems [74], which are enhanced cruise control systems that enable preceding vehicles to be followed automatically at a safe distance,

- Lane departure warning systems [75],
- Lane keeping systems [76], which automate steering on straight roads,
- Vision enhancement and night vision systems,
- Driver condition monitoring systems [77], which detect and provide warning for driver drowsiness, and
- Safety event recorders, automatic collision and severity notification systems.

Adaptive cruise control (ACC) systems are an extension of standard cruise control systems. They essentially rely on radars to detect preceding vehicles traveling in the same lane on the highway. When following a slower moving vehicle, an ACC system automatically switches from speed control to spacing control and follows the preceding vehicle at a safe distance using automated throttle control. In the absence of preceding vehicles, the ACC vehicle travels at a user-set speed, much like a vehicle with a standard cruise control system. An ACC system does not depend on wireless communication or on cooperation from other vehicles on the highway. It only uses on-board sensors such as radar to accomplish the task of maintaining the desired spacing from the preceding vehicle. A formal model of a distributed control system in which every car is controlled by adaptive cruise control was proposed for cooperative systems [78]. Further research is needed to understand the influence of inter-vehicle spacing policies and control algorithms on traffic flow stability.

Three types of lateral systems have been developed in the automotive industry to address lane departure accidents: lane departure warning systems, lane keeping systems and yaw stability control systems. A lane departure warning system (LDWS) is a system that monitors the vehicle's position with respect to the lane and provides warning if the vehicle is about to leave the lane. A lane-keeping system (LKS) automatically controls the steering to keep the vehicle in its lane. Vehicle stability control systems prevent vehicles from spinning, drifting out and rolling over. Such stability control systems are often referred to as yaw control systems or electronic stability control systems. A yaw stability control system contributes to rollover stability by keeping the vehicle on its intended path. Active rollover prevention systems are being developed for sport utility vehicles and trucks. Three types of stability control systems have been proposed and developed for yaw control:

1. Differential braking systems rely on the ABS brake system of the vehicle to apply differential braking between the right and left wheels to control yaw moment,
2. Steer-by-wire systems modify the driver's steering angle input and add a correction steering angle to the wheels, and

3. Active torque distribution systems rely on active differentials and all wheel drive technology to independently control the drive torque distributed to each wheel and thus provide active control of both traction and yaw moment.

Assuring the seamless and safe integration of multiple driver assistance systems is a major challenge for vehicle manufacturers. Moreover, different suppliers provide software modules for different control functionalities. A preliminary approach combining formal methods, control theory and correct-by-construction software synthesis was used as a preliminary approach to address this problem [79]. The ACC problem was formalised using a hybrid dynamical system and an LTL specification, allowing for the explicit computation of the control domain.

The frequency of crashes has been gradually declining worldwide thanks to the development of advanced driver assistance systems. Yet these systems only operate for short periods of time, in limited settings, by responding to specific events. They are meant to be used exclusively if the OEDR is performed by a human driver. As a result, advanced driver assistance systems are only meant to increase safety for level 2 driving automation systems and below. New approaches are needed for level 3 Automated Driving Systems and above.

Safety for level 3, 4 and 5 Automated Driving Systems

It has often been suggested that testing Automated Driving Systems in real or simulated traffic environments should provide sufficient safety guarantees, by making statistical comparisons to human driver performance. However, it was shown that the number of miles of driving needed to provide such statistical evidence amounts to hundreds of millions to hundreds of billions [80]. Moreover, the simulated traffic environment used for validation needs to model human behaviour and interaction properly, as an overly simplified driving environment would not challenge the ADS with satisfactory real-life scenarios. A framework was proposed for the simulation of realistic scenarios [81]. Recorded sensor data was used to create critical traffic scenarios for open-loop testing from observed, non-critical real-world situations [82]. Yet despite the importance of edge case testing [83], a lack of simulation tools designed specifically to test Automated Driving Systems remains.

ISO 26262 is a standard that regulates functional safety of road vehicles. It recommends the use of a hazard analysis method to identify hazardous events and to specify safety goals that mitigate them. The standard is currently insufficient in its current form for Automated Driving Systems, as was discussed in [84], which recommends more safety refinement steps to the standard's lifecycle. The importance of standardisation of safety

assurance was discussed in [85] and an interpretable model for safety assurance was provided. However, the formalisation is based on the notion of accident blame. As stated in the previous section, safety goes beyond blame assignment and attempts to identify all the factors involved to improve the system. Moreover, it is often conceded that society will only accept Automated Driving Systems if the fatality rate is reduced by three orders of magnitude compared to human drivers, leading to a probability of 10^{-9} fatalities caused by an accident per one hour of driving. Though this estimate takes some inspiration from aviation standards, more work is needed to justify this evaluation of public acceptability and the way this affects the perception of what safety means and the way it should be conducted.

Several papers were published on various subjects related to the safety of Automated Driving Systems recently. The STAMP model and STPA technique were used in a case study [86] for the identification of hazards and safety constraints. A correct-by-construction controller was proposed [87], by abstracting obstacle anticipation to a set of Boolean observations. A conceptual framework and meta-model for the design of a safety supervisor was also presented [88]. Formal verification was used to study the most common type of rear-end crashes [89], while [90] proposed an approach based on model predictive control for the development of collision free systems. A formal, prescriptive definition of safe distances was provided to serve as a specification for autonomous vehicle manufacturers [91]. [92] used formal verification for the certification of platoons to verify safety requirements of a model of the system. A moral component in ADS decision making preceding crashes was identified, leading to the development of ethical crashing algorithms [93]. It was noted that machine learning leads to major challenges when attempting to ensure functional safety [94] and that proving guarantees of correctness is a difficult problem in general [95]. A probabilistic extension of temporal logic [96] was proposed to specify correctness requirements in presence of uncertainty, while the use of Bayesian deep learning was suggested to propagate uncertainty throughout the entire system pipeline in [97].

2.5 Conclusion

This chapter presented an overview of the terminology for driving automation systems and safety as well as a review of related work on the safety challenges of driving automation. The following points were brought to light:

While recent studies have revealed an increased interest in the safety challenges of driving automation, research in this area remains insufficient. The focus has not yet

shifted from advanced driving assistance systems to Automated Driving Systems. Advanced driver assistance systems are meant to be used when the OEDR is performed by a human driver, for level 2 driving automation systems and below. Yet an Automated Driving System is expected to perform the entire dynamic driving task on a sustained basis, including the OEDR. Advanced driver assistance systems do not provide a solution to make these systems safer, meaning new approaches must be developed for level 3 Automated Driving Systems and above.

It has been pointed out that system dependability goes beyond its mere compliance with requirements and specifications. While there has been much discussion regarding the amount of testing needed to provide sufficient safety guarantees, there is much work to be done towards properly understanding ADS capabilities, behaviour and interaction with the driving environment. The amount of testing and simulation that must be performed does have its importance. However, making the architecture of an ADS explicit should be a priority. Recent studies focusing on the safety of Automated Driving Systems have commented on the lack of metrics allowing for proper assessment of system capabilities and performance. The need for proper simulation tools was pointed out, as was the importance of standardisation of safety assurance. Most studies adopted a system-oriented approach to safety, relying on some formal model of the system or part of the system.

The thesis aims at providing a framework in which the capabilities of an Automated Driving System are made clearer. More specifically, the focus is set on level 4 Automated Driving Systems, which must perform both the DDT and DDT fallback without any expectation that a user will respond to a request to intervene.

Chapter 3

Probabilistic Time Petri Nets

3.1 Introduction

Modelling tools are needed to represent Automated Driving Systems and their requirements. These tools must integrate concurrency, real-time constraints and probabilities. Designing such models is challenging for they require the development of new algorithms that combine both real-time and probabilistic verification techniques. Continuous-time Markov chains [98], continuous-time Markov decision processes [99], probabilistic timed automata [100], Markov automata [101] and stochastic timed automata [102] are but a few examples of models that were introduced with the intention of formally verifying probabilistic real-time systems. In particular, the product of probabilistic timed automata [103, 104] provides the medium for concurrency in a real-time constrained environment. Yet, none of the aforementioned formalisms are adapted to the modelling of systems that exhibit variables whose bounds cannot be inferred. In contrast, the blending of concurrency and of such dynamical bounds is inherent to Petri net models.

Petri nets were enhanced through the use of stochastic temporal parameters and exponential distributions of firing times in [105, 106] for the modelling of concurrent probabilistic real-time systems. Time Petri nets were extended by adding a probability density-function to the static firing interval of each non-deterministic transition [107]. Stochastic time Petri nets also generalise time Petri nets [108]. They make use of an extension of the state class graph [109] to account for stochastic information in state classes. While stochastic time Petri nets are a powerful formalism in terms of expressivity and conciseness, one can argue that the randomisation of transition rates is not necessarily required, while the randomisation of token generation might be needed. For example, a component failure in a gracefully degrading system can be linked to the firing of a transition whose rate is not necessarily subject to some random phenomenon, but whose outcome needs to be specified in terms of token generation. The extended stochastic Petri nets introduced

in [110] allow firing times to belong to an arbitrary distribution and output places to be randomised, but they still require stringent restrictions, including the randomisation of transition rates.

In this chapter, probabilistic time Petri nets (PTPN) are introduced as a new modelling formalism. By enhancing the forward incidence mapping of a classic time Petri net in such a way that transitions are mapped to a set of distributions of markings, the class of time Petri nets can be extended to a wider class of nets. The output arcs of a transition are effectively replaced with stacks of probabilistic hyperarcs. Each hyperarc contributes to the generation of tokens in output places of the transition. When a transition is fired in a PTPN, one hyperarc is chosen in each stack according to some probability distribution. A resulting marking emerges from this combination of choices. In fact, a time Petri net is a probabilistic time Petri net if the firing of any given transition almost surely leads to a certain marking.

The tools that are used for the analysis of time Petri nets can easily be adapted to this probabilistic setting. Here, the classic atomic state class graph construction [111] is used to isolate the properties of a PTPN into a finite Markov decision process (MDP). This MDP induces the same Markov chains as the semantics of the PTPN, up to isomorphism. As a result, this construction preserves the lower and upper bounds on the probability of reaching a given marking. The extensive set of tools that are used for the study of MDPs can therefore be employed to study the probabilistic real-time reachability problem in the context of PTPNs thoroughly. The construction put forward in this chapter is quite complex, for it is based on the atomic state class graph. Unfortunately, the simpler state class graph construction cannot be adapted to this probabilistic setting, as it does not preserve these lower and upper probability bounds.

3.2 Probabilistic Time Petri Nets

3.2.1 Preliminaries

The set of natural numbers is denoted $\mathbb{N}$, the set of rational numbers by $\mathbb{Q}$ and the set of real numbers by $\mathbb{R}$. 0 is considered to be an element of $\mathbb{N}$. Let $\mathbb{N}^* = \mathbb{N} \setminus \{0\}$. For $n \in \mathbb{N}$, let $[\![0, n]\!]$ denote the set $\{i \in \mathbb{N} \mid i \leq n\}$. The set of real intervals that have rational or infinite endpoints is denoted $\mathscr{I}(\mathbb{Q}_+)$. A *clock valuation* over a set T is a mapping $v : T \to \mathbb{R}_+$, where $\mathbb{R}_+$ denotes the set of non-negative real numbers. Let 0_T denote the clock valuation that assigns 0 to every clock in T. For $d \in \mathbb{R}_+$, let $v + d$ be the clock valuation that satisfies $(v + d)(t) = v(t) + d$ for every clock t in the domain of v.

For a given set X, let $\mathscr{P}(X)$ denote the power set of X. The *characteristic (or indicator)* function of $A \in \mathscr{P}(X)$, denoted $\chi_A : X \to \{0,1\}$, is defined as

$$\chi_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases}$$

Given two arbitrary sets E and F, let F^E denote the set of functions from E to F. When F is an ordered set, the partial order $\preceq$ on F^E is defined by $f \preceq g$ if $f(x) \leq g(x)$ for all $x \in E$.

A *discrete probability distribution* over a countable set X is a function $\mu : X \to [0,1]$ such that $\sum_{x \in X} \mu(x) = 1$. The *support* of a discrete probability distribution μ, denoted $Supp(\mu)$, is the preimage of the interval $]0,1]$ under μ. For an arbitrary set X, $\mathscr{D}ist_X$ is defined as the set of functions $\mu : X \to [0,1]$ such that $Supp(\mu)$ is a countable set and μ restricted to $Supp(\mu)$ is a discrete probability distribution. For $x_0 \in X$, let the discrete probability distribution denoted δ_{x_0} be the *Dirac measure* which assigns probability 1 to x_0:

$$\delta_{x_0}(x) = \chi_{\{x_0\}}(x) = \begin{cases} 1 & \text{if } x = x_0, \\ 0 & \text{if } x \neq x_0. \end{cases}$$

3.2.2 Probabilistic Time Petri Nets

This section introduces the syntax and the semantics of *probabilistic time Petri nets*. Intuitively, a probabilistic time Petri net is a time Petri net in which every non-deterministic choice involves the resolution of a probabilistic experiment. Such experiments are described explicitly by means of discrete probability distributions. In a typical time Petri net, these probability distributions are Dirac measures. In other words, any given state of a time Petri net has a successor that is uniquely determined by a chosen course of action. This is generally not the case for probabilistic time Petri nets, which extend the class of time Petri nets as a result.

Syntax of a probabilistic time Petri net.

Definition 1 (Probabilistic time Petri net (PTPN))**.** *A* probabilistic time Petri net *is a quintuple* $\mathcal{N} = (P, T, Pre, Post, I)$ *where*

- P *is a finite, non-empty set of* places*,*
- T *is a finite set of* transitions *such that* $T \cap P = \emptyset$*,*
- $Pre : T \to \mathbb{N}^P$ *is the backward incidence mapping,*

- $Post : T \to \mathscr{P}(\mathscr{D}ist_{\mathbb{N}^P})$ *is the forward incidence mapping, and*
- $I : T \to \mathscr{I}(\mathbb{Q}_+)$ *is a function assigning a* firing interval *to each transition.*

An element of $\mathbb{N}^P$ is called a *marking* of the net. A marking denotes a distribution of *tokens* in the places of the net. The forward incidence mapping $Post$ specifies a *finite* set of probability distributions of markings for every transition of the net. For a given transition t, it is assumed that the probability distributions in $Post(t)$ are associated with *independent* random variables. These random variables each contribute to the production of tokens in subsequent places when that transition is fired. Since the number of places in the net is finite, the support of each discrete probability distribution in $Post(t)$ is also finite.

A distribution in $Post(t)$ is represented graphically by a stack of *hyperarcs*. A hyperarc is labelled with a probability before it is split into a set of arcs that lead to a set of *output places*. These arcs contain information about the number of tokens that are generated in each one of these places when that hyperarc is selected.

Definition 2 (Marked probabilistic time Petri net)**.** *A* marked probabilistic time Petri net *is a sextuple* $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$ *where*

- $(P, T, Pre, Post, I)$ *is a probabilistic time Petri net, and*
- $\rho_{\mathcal{N}} \in \mathscr{D}ist_{\mathbb{N}^P}$ *is the* distribution of initial markings *of the net.*

The experiment that yields the initial marking of the net is only conducted once. Any marking belonging to the support of $\rho_{\mathcal{N}}$ is *an* initial marking of the net. The value $\rho_{\mathcal{N}}(M)$ specifies the probability that *the* initial marking of the net is indeed M.

Example 1. *In order to grasp the intuition behind the proposed model, let us consider the probabilistic time Petri net depicted in Fig. 3.1. Transition T_2 of the net displays two probability distributions. The first distribution either generates one token in P_1, three tokens in P_3 and one token in P_4 with probability a, or two tokens in P_4, one token in P_5 and one token in P_6 with probability b. No token is generated with probability $c = 1-a-b$. The second distribution generates one or two tokens in P_6 with probability p and $1-p$ respectively. Since these distributions are associated with independent random variables, it follows that the firing of T_2 leads to the consumption of one token in P_1 and two tokens in P_2, and the generation of two tokens in P_4, one token in P_5 and three tokens in P_6 with probability $b(1-p)$.*

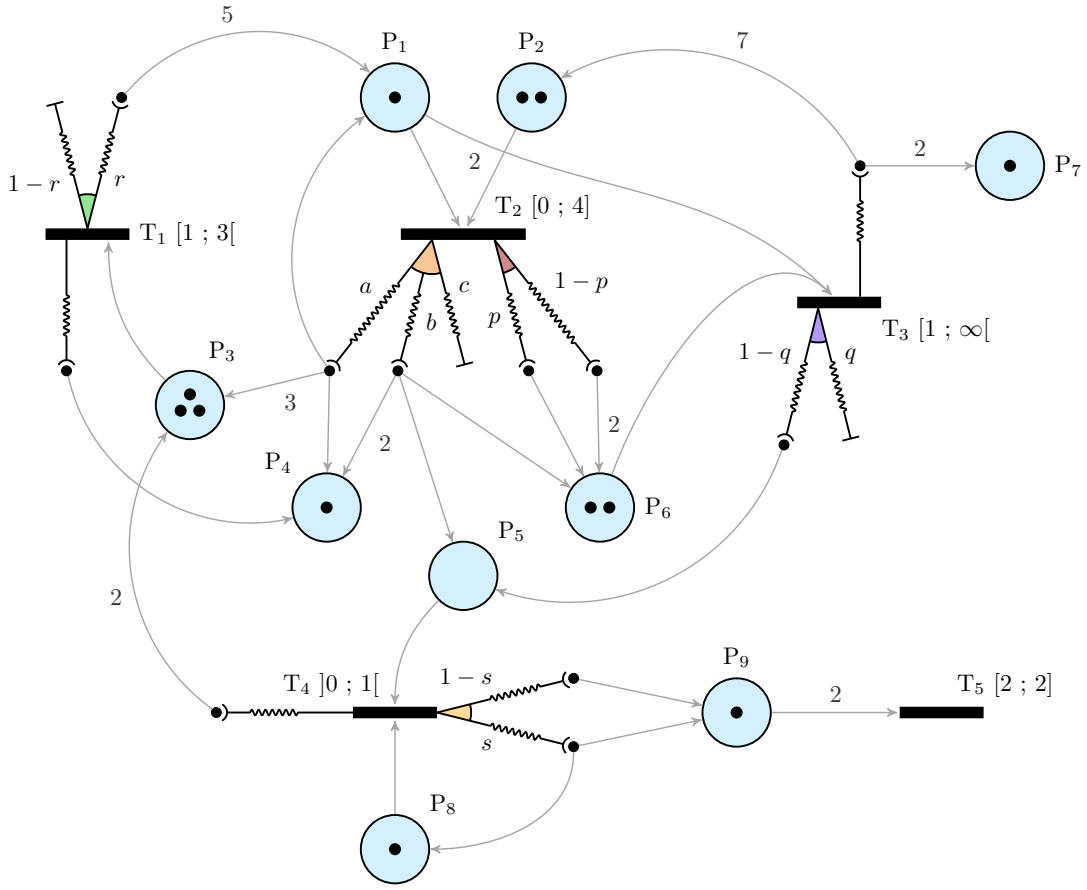


Figure 3.1 – A marked probabilistic time Petri net in its initial state, given by the distribution of initial markings $\rho_{\mathcal{N}} = \delta_{(1,2,3,1,0,2,1,1,1)}$

The following paragraph introduces the terminology of probabilistic time Petri nets as well as important notations. Let $\mathcal{N} = (P, T, Pre, Post, I)$ be a probabilistic time Petri net. A *state* of the net $\mathcal{N}$ is described by an ordered pair (M, v) in $\mathbb{N}^P \times \mathbb{R}_+^T$, where M is a marking of $\mathcal{N}$ and v is a clock valuation over the set of transitions T. In practice, clock valuations are only defined for transitions that are enabled.

- A transition $t \in T$ is said to be *enabled* by a given marking $M \in \mathbb{N}^P$ if M supplies t with at least as many tokens as required by the backward incidence mapping Pre. We define the set $\mathscr{E}(M)$ of transitions that are enabled by the marking M as

$$\mathscr{E}(M) = \{t \in T \mid M \succeq Pre(t)\}.$$

- A transition $t \in T$ is said to be *firable* from a given state (M, v) if the transition t is enabled by M and if its clock is assigned a value that lies within its firing interval. We define the set $\mathscr{F}(M, v)$ of transitions that are firable from the state (M, v) as

$$\mathscr{F}(M, v) = \{t \in \mathscr{E}(M) \mid v(t) \in I(t)\}.$$

- A time delay $d \in \mathbb{R}_+$ is said to be *compliant* with a given state (M, v) if every transition that is firable from $(M, v + d')$ for some time delay $d' \in [0, d]$ stays firable until $(M, v + d)$. We define the set $\mathscr{C}(M, v)$ of time delays that are compliant with the state (M, v) as

$$\mathscr{C}(M, v) = \{d \in \mathbb{R}_+ \mid \forall t \in T, t \notin \mathscr{F}(M, v + d) \Rightarrow \forall d' \in [0, d], t \notin \mathscr{F}(M, v + d')\}.$$

- An action $(d, t) \in \mathbb{R}_+ \times T$ is said to be *feasible* from a given state (M, v) if the time delay d leads the net to a state from which t is firable. We define the set $\Phi(M, v)$ of actions that are feasible from the state (M, v) as

$$\Phi(M, v) = \{(d, t) \in \mathbb{R}_+ \times T \mid d \in \mathscr{C}(M, v) \text{ and } t \in \mathscr{F}(M, v + d)\}.$$

When adopting a purely semantical standpoint, an element of the set T is best referred to as a *trial*, through the medium of an underlying probability distribution μ_t. Informally, a trial t induces the production of tokens in the net each time it is conducted, by providing alternatives that lead to one marking or another.

- Let $t \in T$ be a transition of $\mathcal{N}$. The discrete probability distributions in $Post(t)$ are associated with random variables that can take one of many values. By definition, these values are endowed with a non-zero probability. An *alternative* is a function f that chooses a value for each one of these random variables. Formally, the set $\mathscr{A}(t)$

of *alternatives* provided by the transition t is defined as follows:

$$\mathscr{A}(t) = \Big\{ f : Post(t) \to \mathbb{N}^P \mid \forall \mu \in Post(t), f(\mu) \in Supp(\mu) \Big\}.$$

- An *outcome* of a given trial t is a marking ω of $\mathcal{N}$ which results from the choices of some alternative in $\mathscr{A}(t)$. This marking ω accounts for the tokens that are to be generated in each output place of t. The set $\Omega(t)$ of outcomes of the trial t is defined as
$$\Omega(t) = \left\{ \omega \in \mathbb{N}^P \mid \exists f \in \mathscr{A}(t), \omega = \sum_{\mu \in Post(t)} f(\mu) \right\}.$$
- For a given outcome $\omega \in \Omega(t)$, the non-empty set $\mathscr{A}_\omega(t) \subseteq \mathscr{A}(t)$ of alternatives that lead to it is defined as
$$\mathscr{A}_\omega(t) = \left\{ f \in \mathscr{A}(t) \mid \omega = \sum_{\mu \in Post(t)} f(\mu) \right\}.$$

Let us now provide the formal definition of the probability distribution μ_t that governs a trial $t \in T$. Intuitively, the probability of reaching a given outcome $\omega \in \Omega(t)$ is the sum of the probabilities of all the alternatives leading to ω. Since the probability distributions in $Post(t)$ are assumed to be independent, the probability that an alternative is chosen is the product of the probabilities of all the independent choices that are made. Formally, μ_t is defined as follows:

Definition 3. *Let $(P, T, Pre, Post, I)$ be a probabilistic time Petri net. The discrete probability distribution that governs a trial $t \in T$ is a function $\mu_t : \Omega(t) \to [0, 1]$ that assigns probabilities to the outcomes of t as follows:*

$$\mu_t : \omega \to \sum_{f \in \mathscr{A}_\omega(t)} \left(\prod_{\mu \in Post(t)} \mu\Big(f(\mu)\Big) \right).$$

Lemma 1. *Let $(P, T, Pre, Post, I)$ be a probabilistic time Petri net. For a given trial $t \in T$, the function μ_t is a discrete probability distribution over $\Omega(t)$.*

Proof. Let us consider a trial t of $\mathcal{N}$.

- For each $f \in \mathscr{A}(t)$ and each $\mu \in Post(t)$, the marking $f(\mu)$ lies in $Supp(\mu)$, hence $\mu(f(\mu)) \in [0, 1]$ and $\mu_t(\omega) > 0$ for all $\omega \in \Omega(t)$.

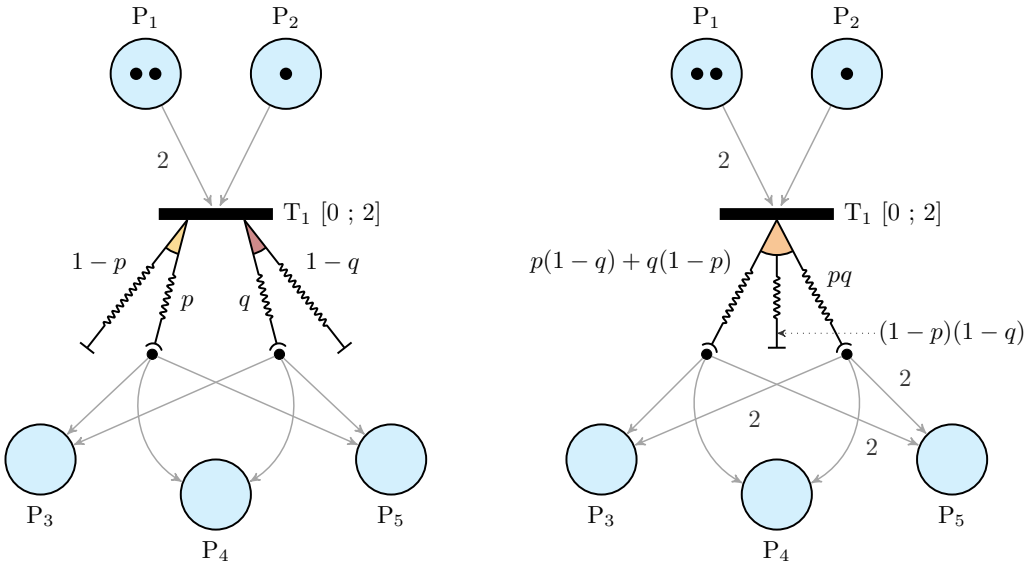


Figure 3.2 – Two syntactically different probabilistic time Petri nets that are equivalent from a semantical standpoint

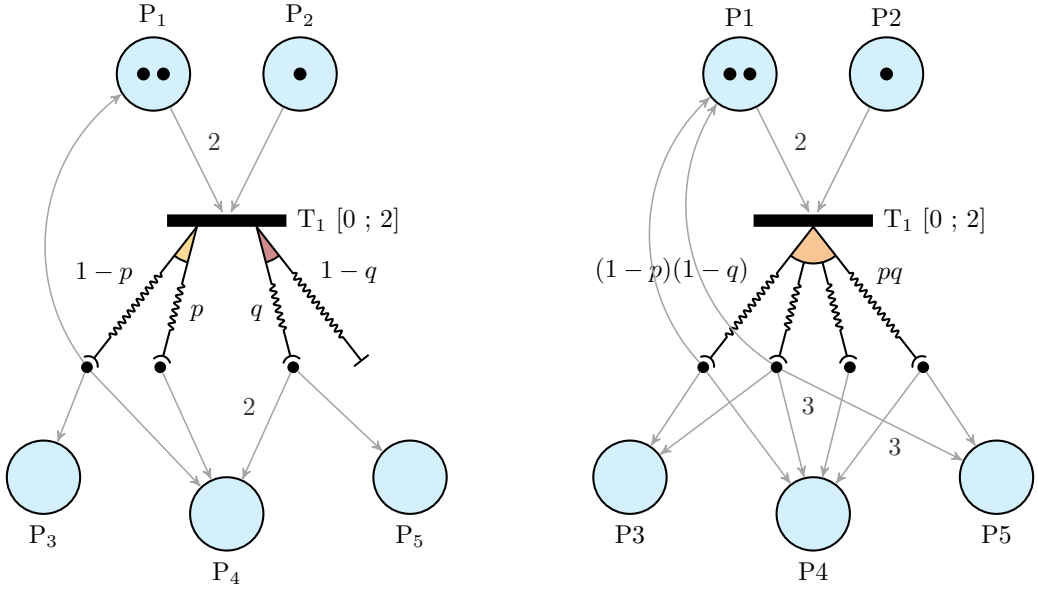


Figure 3.3 – Two other syntactically different probabilistic time Petri nets that are equivalent from a semantical standpoint

- Let us now show that

$$\sum_{\omega\in\Omega(t)} \mu_t(\omega) = 1.$$

— The family of subsets $\{\mathscr{A}_\omega(t)\}_{\omega\in\Omega(t)}$ of $\mathscr{A}(t)$ defines a partition of the set $\mathscr{A}(t)$. Therefore,

$$\sum_{\omega\in\Omega(t)} \left[\sum_{f\in\mathscr{A}_\omega(t)} \left(\prod_{\mu\in Post(t)} \mu\Big(f(\mu)\Big) \right) \right] = \sum_{f\in\mathscr{A}(t)} \left(\prod_{\mu\in Post(t)} \mu\Big(f(\mu)\Big) \right).$$

— An element of $\mathscr{A}(t)$ is characterised by its graph, which consists of $|Post(t)|$ *independent* choices. Each one of these choices corresponds to a probabilistic experiment by means of the discrete probability distributions that make up $Post(t)$. Consequently, the function

$$\begin{aligned} \mu_\times : \mathscr{A}(t) &\longrightarrow [0,1] \\ f &\longmapsto \prod_{\mu\in Post(t)} \mu\Big(f(\mu)\Big) \end{aligned}$$

is a discrete probability distribution over $\mathscr{A}(t)$ and

$$\sum_{\omega\in\Omega(t)} \mu_t(\omega) = \sum_{f\in\mathscr{A}(t)} \mu_\times(f) = 1$$

□

The probabilistic time Petri nets depicted in Fig. 3.2 have different structures. Yet they are equivalent from a semantical standpoint, since the discrete probability distribution μ_{T_1} is the same in both nets. Another example of such equivalence is given in Fig. 3.3. In fact, every probabilistic time Petri net can be canonicalised into a probabilistic time Petri net such that $Post(t)$ is a singleton for every transition t.

It is worth noting that a probabilistic time Petri net is equivalent to a time Petri net if $Supp(\mu_t)$ is a singleton for all trials t. A time Petri net can therefore be interpreted as a probabilistic time Petri net whose transitions yield a single combination of hyperarcs, or similarly, whose trials each lead to a single outcome.

Semantics of a probabilistic time Petri net.

A probabilistic time Petri net $\mathcal{N}$ has the following operational behaviour. The distribution $\rho_\mathcal{N}$ yields the initial marking M_0 of the net $\mathcal{N}$ and subsequently, the initial state $(M_0, 0_T)$ of $\mathcal{N}$. From a given state, either enabled transitions are fired or time is allowed

to flow. Doing so changes the state of the net. An enabled transition is firable if and only if its clock value lies within its firing interval. Furthermore, a time delay must always be compliant with the current state of the net. In other words, time can flow as long as otherwise enabled transitions are not disabled in the process. This behaviour is typical in the context of *strong time semantics* and conveys the notion of urgency. As such, the behaviour of a probabilistic time Petri net is similar to that of a classic time Petri net. Once a choice has been made, however, the next state is selected in a probabilistic manner. The difference therefore lies in the way the subsequent state of the net is computed once the non-determinism has been resolved.

- If a certain amount of time d is allowed to elapse, then the marking remains the same while the clock values of the enabled transitions are increased by that particular amount.
- If a certain transition t is fired, tokens are removed from the current marking according to the mapping $Pre(t)$ while the outcome of the trial t generates additional ones. Moreover, the clocks associated with the transition t and with any transition that has been disabled by the removal of the $\sum_{p \in P} Pre(t)(p)$ tokens are reset and disabled. Finally, the clocks associated with newly enabled transitions are activated. This includes those that were previously disabled.

The semantics of a probabilistic time Petri net is defined as a probabilistic timed transition system. Probabilistic timed transition systems can be considered an extension of Markov decision processes that account for the flow of time, leading to a potentially uncountable set of states. Formally:

Definition 4 (Probabilistic timed transition system (PTTS))**.** *A* probabilistic timed transition system *is a quadruple* (Q, ρ, T, W) *where*

- Q *is a set of* states*,*
- $\rho \in \mathcal{D}ist_Q$ *is the distribution of* initial states*,*
- T *is a set of* trials*, and*
- $W : Q \times (T \cup \mathbb{R}_+) \to \mathcal{D}ist_Q$ *is a (partial) probabilistic transition function.*

The semantics of marked probabilistic time Petri nets is formally defined in terms of probabilistic timed transition systems:

Definition 5 (Semantics of a marked probabilistic time Petri net)**.** *The semantics of a marked probabilistic time Petri net* $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$ *is a probabilistic timed transition system* $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ *where*

- $Q \subseteq \mathbb{N}^P \times \mathbb{R}_+^T$ *is the set of* states *of the net* $\mathcal{N}$,
- $\rho_{S_\mathcal{N}} : Q \to [0,1]$ *is the distribution of* initial states, *defined for* $(M,v) \in Q$ *by*

$$\rho_{S_\mathcal{N}}(M,v) = \rho_\mathcal{N}(M) \times \chi_{\{0_T\}}(v) \text{ , and}$$

- $W : Q \times (T \cup \mathbb{R}_+) \to \mathcal{D}ist_Q$ *is the (partial) piecewise* probabilistic transition function *that defines* continuous time transition relations *over* $Q \times \mathbb{R}_+$ *and* discrete transition relations *over* $Q \times T$.

1. *W is defined for $((M,v),d) \in Q \times \mathbb{R}_+$ if and only if the delay d is compliant with the state (M,v). In that case, let $W((M,v),d)$ be the Dirac measure $\delta_{(M,v')}$, where the clock valuation v' is defined for all transitions t' enabled by the marking M by*

$$v'(t') = v(t') + d.$$

2. *W is defined for $((M,v),t) \in Q \times T$ if and only if the transition t is firable from the state (M,v). In that case, let $W((M,v),t) = \tilde{\mu}_t$, where $\tilde{\mu}_t \in \mathcal{D}ist_Q$ is defined as follows:*
   - *Let $(M',v') \in Q$. The state (M',v') lies in $Supp(\tilde{\mu}_t)$ if and only if the two following conditions are met:*
     - *there exists an outcome $\omega_{M'} \in \Omega(t)$ such that*

$$M' = (M - Pre(t)) + \omega_{M'} \tag{3.1}$$

     - *the clock valuation v' is defined for all transitions t' enabled by the marking M' by*

$$v'(t') = v(t') \times (1 - \chi_t(t')) \times \chi_{\mathcal{E}(M-Pre(t))}(t') \tag{3.2}$$

   - *Suppose that $(M',v') \in Supp(\tilde{\mu}_t)$. We define the image of (M',v') by the formula*

$$\tilde{\mu}_t(M',v') = \mu_t(\omega_{M'}).$$

■

Figure 3.4 depicts two probabilistic time Petri nets and a fragment of their semantics. Clock valuations are not represented. Figure 3.5 depicts a probabilistic time Petri and

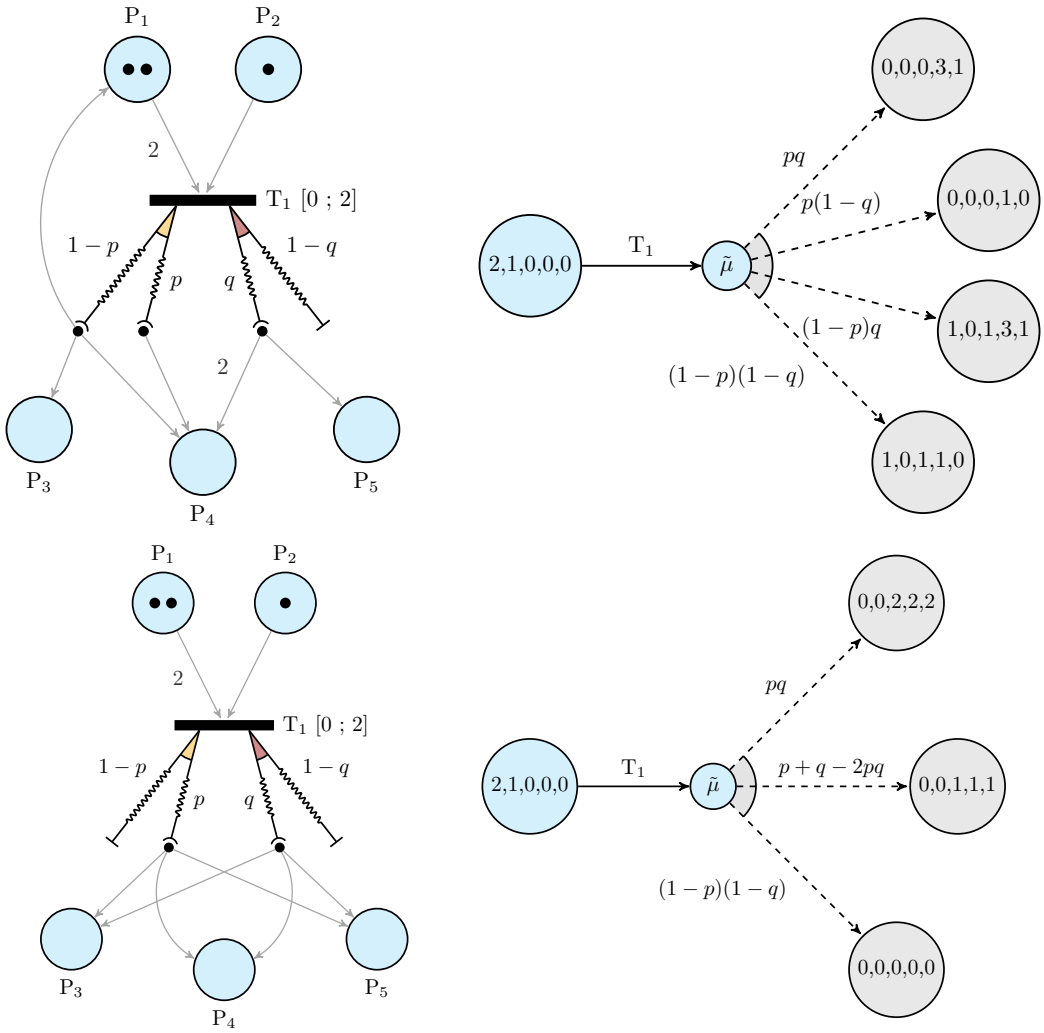


Figure 3.4 – Correspondence between the transitions of a probabilistic time Petri net and the trials of its semantics

one of its possible executions. The clock values of enabled transitions are represented in orange. The clock values of firable transitions are represented in green. As stated earlier, strong time semantics are used, meaning time can only flow as long as otherwise enabled transitions are not disabled in the process.

Example 2. *Let us consider the probabilistic time Petri net depicted in Fig. 3.6. This net represents the formal specification for the oral presentation of the article* Probabilistic Time Petri Nets*, whose contents are presented in this chapter. It was used as a substitute for the table of contents of the presentation at the 37th International Conference on Application and Theory of Petri Nets and Concurrency.*

An oral presentation typically lasts 20 to 30 minutes. At least 5 minutes must be left for questions from the audience. The chairman of the session is expected to cut the presentation short if the presentation exceeds 28 minutes. He/She can even force the

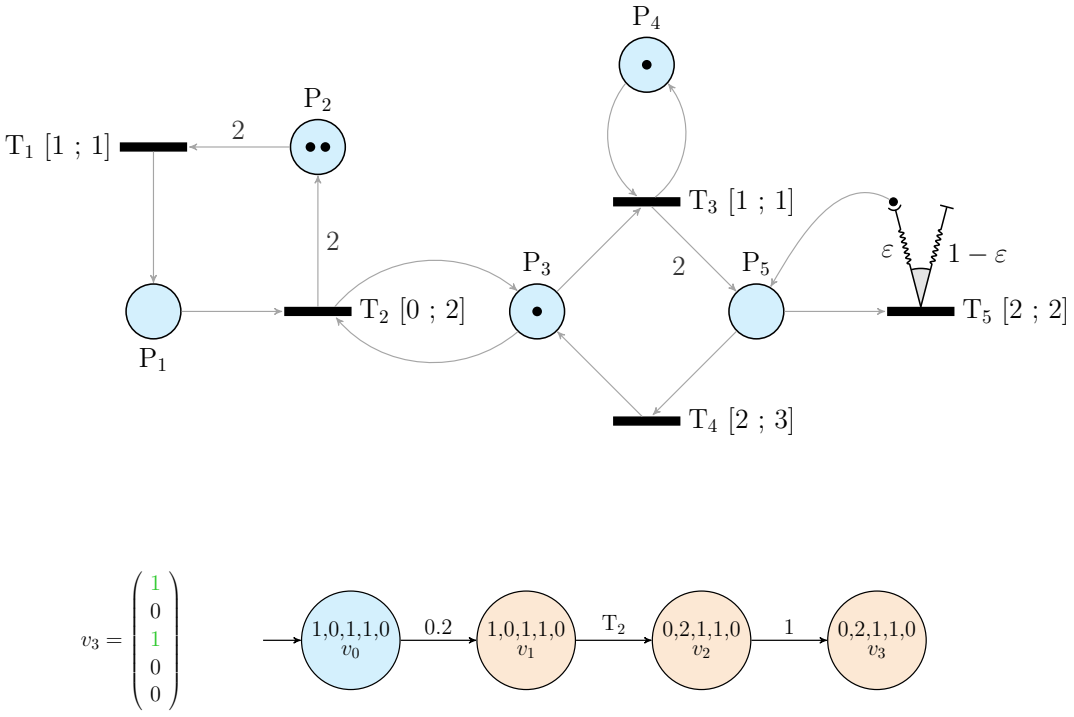


Figure 3.5 – A finite path in the probabilistic timed transition system of a probabilistic time Petri net

speaker to jump to the conclusion or simply stop the presentation at the 25 minute mark to leave some time for questions (with a fifty-fifty chance).

The chairman is represented by a place labelled 'Watchdog'. The watchdog timer is represented by a transition to which the firing interval $[25, 28]$ *is assigned. Such a model can be used to determine presentations that are acceptable, in the sense that the chairman will not deem it necessary to interfere. It can for instance be used to optimise the time spent on each section of the presentation.*

3.3 Probabilistic Systems

Probabilistic systems are systems that are subject to various phenomena of stochastic nature. The verification of probabilistic systems can be performed through quantitative or qualitative properties. Quantitative properties constrain the probability or expectation of certain events. For example, it can be required that the probability of delivering a message in an allotted time be at least 0.9. Qualitative properties assert that good events with

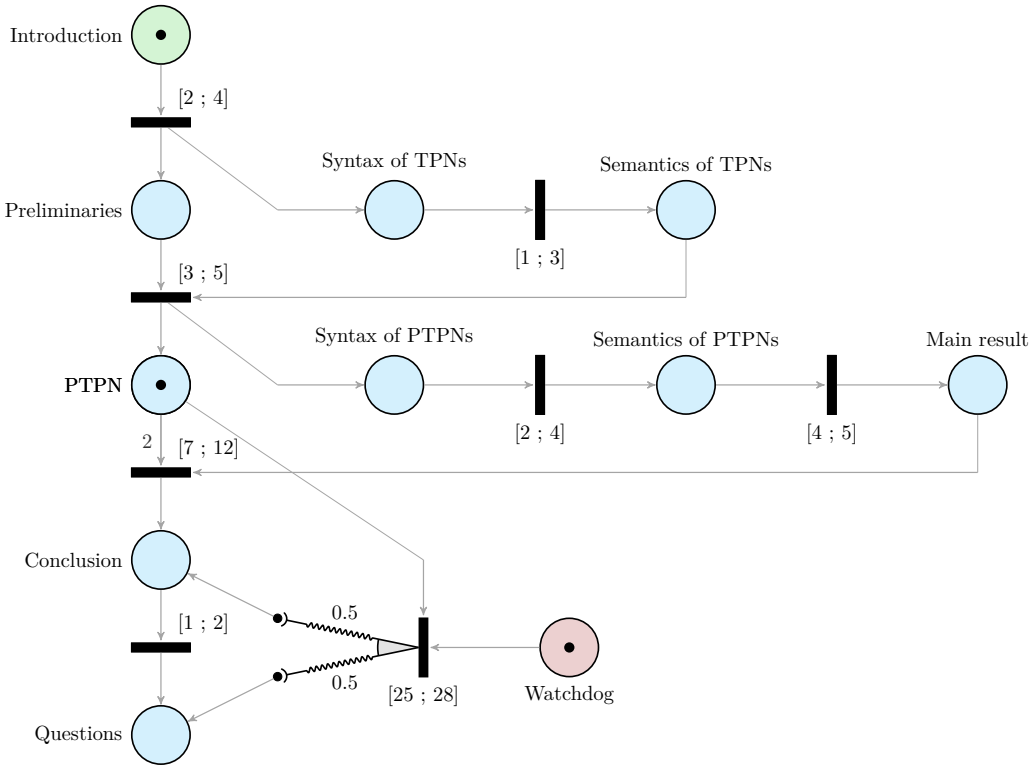


Figure 3.6 – Formal specification of the presentation of the article *Probabilistic Time Petri Nets* at the 37th International Conference on Application and Theory of Petri Nets and Concurrency

almost surely happen (with probability 1) and that bad events almost never occur (with probability 0. They typically include reachability, persistence and repeated reachability.

In order to model probabilistic systems, transition systems were enhanced with probabilities to model random phenomena. For example, Markov chains are transition systems in which every nondeterministic choice is replaced by a purely probabilistic one. They are the most popular model for the evaluation of performance and dependability of information-processing systems. Formally:

Definition 6 (Markov chain (MC))**.** *A* Markov chain *is a triple* $(S, \rho, \mathbf{P})$ *where*

- S *is a countable, non-empty set of* states *of the chain,*
- $\rho \in \mathcal{D}ist_S$ *is the distribution of initial states of the chain, and*
- $\mathbf{P} : S \to \mathcal{D}ist_S$ *is the (total) probabilistic transition function.*

M is said to be finite if S is finite.

Markov chains cannot model the interleaving behaviour of concurrent processes in an adequate manner. Markov decision processes (MDP) are used for this purpose instead.

Both nondeterministic and probabilistic choices coexist in an MDP. They are transition systems in which a nondeterministic choice of probability distributions exists in every state. Markov decision processes can be used to model concurrency between processes in randomised distributed algorithms by means of interleaving. Formally:

Definition 7 (Markov decision process (MDP))**.** *A* Markov decision process *is a quadruple* $(C, \rho, A, \mathbf{P})$ *where*

- C *is the (countable) set of states of the process,*
- $\rho \in \mathscr{D}ist_C$ *is the distribution of initial states of the process,*
- A *is the set of actions of the process, and*
- $\mathbf{P} : C \times A \to \mathscr{D}ist_C$ *is the (partial) probabilistic transition function.*

For a given state c *of a Markov decision process, we define the set* $\Sigma(c)$ *of actions that are enabled in the state* c *as*

$$\Sigma(c) = \{\alpha \in A \mid \mathbf{P}(c, \alpha) \text{ is defined}\}.$$

The assumption that $\Sigma(c) \neq \emptyset$ *for all* $c \in C$ *is a conventional requirement in the literature that is not specific to our setting.*

A Markov chain is a Markov decision process in which the choice of a probability distribution is uniquely determined in each state of the system. In both Markov chains and Markov decision processes, the probability distributions only depend on the current state on the system. The evolution of such probabilistic systems does not depend on its history but only on its current state. This is known as the memoryless property.

It is important to point out that the definitions used here are extracted from [112]. These definitions are used by the formal methods community for the verification of computer systems and networks through stochastic models and measurements. While Markov chains are often defined as sequences of random variables, a state-based view of probabilistic models is adopted here. The robotics community usually defines Markov decision processes with reward functions and a discount factor, while weakening the distribution of initial states specified here to a Dirac measure. The rationale behind the choices made by each one of these communities exceeds the scope of this document, yet a few elements can be highlighted. The use of reward functions is not always needed or even adapted to resolve non-deterministic choices in a Markov decision process. While the robotics community exclusively formulates policies in MDPs according to some optimisation of a cumulative function of random rewards, the formal methods community makes use of a

wide variety of modal logics to specify requirements. Some of these modal logics take reward functions into account, while others focus on atomic propositions and labelling functions instead. Furthermore, reward functions are objects that do not change the semantics of a Markov decision process. As a result, they are not inherent to the structure of MDPs but rather act as observation variables. For this reason, reward functions are not taken into account in the definition of Markov chains or Markov decision processes in this document.

3.4 The Probabilistic Real-Time Reachability Problem

A state is said to be *reachable* if there exists a sequence of transition relations that leads a probabilistic time Petri net from one of its initial states to that particular one. When considering a given system, one might want to express the fact that certain unwanted events are unlikely to happen when it operates. If that system is modelled as a probabilistic time Petri net, those unwanted events are formally represented by a certain set of states. Proving whether a given set of states can be reached with a certain probability or not is at the core of the *probabilistic real-time reachability problem for probabilistic time Petri nets*. Quantitative reachability properties assert that the probability of reaching certain unwanted states is sufficiently small and that the probability of achieving a certain desired system behaviour is above a given threshold.

We artificially introduce $(d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}})$ as the action performed when no transition is to be fired again. Let $d_{\mathcal{N}}$ be a real number that is strictly greater than the greatest real endpoint of any firing interval in $\{I(t) \mid t \in T\}$ and let $\mathrm{t}_{\mathcal{N}}$ be a fictitious trial that does not belong to the set T. Intuitively, $(d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}})$ is to be a feasible action whenever the firing intervals of the transitions enabled in the current state of the net have no upper bound.

Subsequently, the extended set $\tilde{\Phi}(M, v)$ of actions that are feasible from a given state (M, v) is defined by setting $\tilde{\Phi}(M, v) = \Phi(M, v) \cup \{(d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}})\}$ if $\mathscr{C}(M, v) = \mathbb{R}_+$, and $\tilde{\Phi}(M, v) = \Phi(M, v)$ otherwise. Let $\tilde{T} = T \cup \{\mathrm{t}_{\mathcal{N}}\}$ denote the *extended set of trials* and extend the domain of the partial piecewise probabilistic transition function W to take $(d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}})$ into account as follows:

$$W((M, v + d_{\mathcal{N}}), \mathrm{t}_{\mathcal{N}}) = \delta_{(M, v+d_{\mathcal{N}})}.$$

3.4.1 Paths and Schedulers

The possible evolution of a probabilistic time Petri net is described formally by a *path*. Reasoning about probabilities of sets of paths relies on the resolution of non-determinism, which is performed by a *scheduler*. The paths describe the potential computations that are obtained by resolving both the non-deterministic and probabilistic choices in the underlying probabilistic timed transition system. In other words, a path is a sequence of trials that are performed at certain dates. These trials carry the net over a set of states.

Definition 8 (Path in a probabilistic timed transition system)**.** *Let $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ be the semantics of a marked probabilistic time Petri net $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$.*

- *A finite* path *in the probabilistic timed transition system $S_{\mathcal{N}}$ is a finite sequence*

$$q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} \dots \xrightarrow{d_n,t_n} q_n$$

where $q_0 \in Supp(\rho_{S_{\mathcal{N}}})$, $n \in \mathbb{N}$ and for all $i \in [\![0, n-1]\!]$,

$$\begin{cases} q_i = (M_i, v_i) \in Q, \\ (d_{i+1}, t_{i+1}) \in \tilde{\Phi}(M_i, v_i), \\ q_{i+1} \in Supp(W((M_i, v_i + d_{i+1}), t_{i+1})). \end{cases}$$

The integer n is called the length *of the path. A finite path in $S_{\mathcal{N}}$ is an element of $Supp(\rho_{S_{\mathcal{N}}}) \times ((\mathbb{R}_+ \times T) \times Q)^*$. The set of finite paths in the probabilistic timed transition system $S_{\mathcal{N}}$ is denoted $Path^*_{(S_{\mathcal{N}})}$.*

- *An infinite* path *in the probabilistic timed transition system $S_{\mathcal{N}}$ is an infinite sequence*

$$q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} q_2 \xrightarrow{d_3,t_3} \dots$$

such that $q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} \dots \xrightarrow{d_n,t_n} q_n \in Path^*_{(S_{\mathcal{N}})}$ for all $n \in \mathbb{N}$.
An infinite path in $S_{\mathcal{N}}$ is an element of $(Q \times (\mathbb{R}_+ \times T))^\infty$. The set of infinite paths in the probabilistic timed transition system $S_{\mathcal{N}}$ is denoted $Path^\infty_{(S_{\mathcal{N}})}$.

The resolution of all non-deterministic choices in a probabilistic time Petri net is described formally by a scheduler. A scheduler chooses a feasible action $\tilde{\Phi}(M, v)$ in any state (M, v) of the net, but does not have any influence on the probability that one marking or another will be reached once that action has been chosen.

Definition 9 (Scheduler for a probabilistic timed transition system)**.** *Let $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ be the semantics of a marked probabilistic timed transition system $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$.*

For a given finite path $\pi = q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} \dots \xrightarrow{d_n,t_n} q_n$ *in* $S_{\mathcal{N}}$*, let* $last(\pi)$ *denote the state* q_n*.*

- *A* scheduler *for the probabilistic timed transition system* $S_{\mathcal{N}}$ *is a (total) function* $\mathfrak{S} : Path^*_{(S_{\mathcal{N}})} \to (\mathbb{R}_+ \times T) \cup \left\{(d_{\mathcal{N}}, t_{\mathcal{N}})\right\}$ *such that for all finite paths* π *in* $S_{\mathcal{N}}$

$$\begin{cases} \mathscr{C}(last(\pi)) \neq \mathbb{R}_+ \Rightarrow \mathfrak{S}(\pi) \in \Phi(last(\pi)), \\ \mathscr{C}(last(\pi)) = \mathbb{R}_+ \Rightarrow \mathfrak{S}(\pi) \in \tilde{\Phi}(last(\pi)). \end{cases}$$

A finite or infinite path $\pi = q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} \dots$ *of* $S_{\mathcal{N}}$ *is called a* $\mathfrak{S}$-path *if* $\mathfrak{S}(\pi_{|i}) = (d_{i+1}, t_{i+1})$ *for all prefixes* $\pi_{|i}$ *(the path* $\pi_{|i}$ *denotes the finite prefix of* π *of length* i*). Let* $Path^*_{\mathfrak{S}}$ *denote the (countable) set of finite* $\mathfrak{S}$*-paths.*

The behaviour of a probabilistic time Petri net that is subject to a scheduler $\mathfrak{S}$ can be formalised by a *Markov chain* [112]. Intuitively, this Markov chain unfolds the net into as many trees as there are elements in $Supp(\rho_{\mathcal{N}})$.

Definition 10 (Markov chain of a PTPN induced by a scheduler)**.** *Let* $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ *be the semantics of a marked probabilistic time Petri net* $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$*.*

A scheduler $\mathfrak{S}$ *of* $S_{\mathcal{N}}$ *induces a Markov chain* $\mathscr{M}_{\mathfrak{S}} = (Path^*_{\mathfrak{S}}, \rho_{\mathfrak{S}}, \mathbf{P}_{\mathfrak{S}})$ *where*

- $\rho_{\mathfrak{S}} \in \mathscr{D}ist_{Path^*_{\mathfrak{S}}}$ *is the distribution of initial paths of the chain. Its support is equal to the finite paths in* $S_{\mathcal{N}}$ *of length* 0 *that are also initial states of the probabilistic timed transition system* $S_{\mathcal{N}}$*.*

  For all $(M_0, 0_T) \in Supp(\rho_{\mathfrak{S}})$, $\rho_{\mathfrak{S}}((M_0, 0_T)) = \rho_{S_{\mathcal{N}}}(M_0, 0_T) = \rho_{\mathcal{N}}(M_0)$.
- $\mathbf{P}_{\mathfrak{S}} : Path^*_{\mathfrak{S}} \to \mathscr{D}ist_{Path^*_{\mathfrak{S}}}$ *is the (total) probabilistic transition function of* $\mathscr{M}_{\mathfrak{S}}$*. For* $\lambda \in Path^*_{\mathfrak{S}}$*, the support of* $\mathbf{P}_{\mathfrak{S}}(\lambda)$ *is the set of* $\mathfrak{S}$*-paths of the form* $\pi \xrightarrow{\mathfrak{S}(d,t)} q$*. For* $(\pi, q) \in Path^*_{\mathfrak{S}} \times Q$*,*

$$\mathbf{P}_{\mathfrak{S}}(\pi)(\pi \xrightarrow{\mathfrak{S}(d,t)} q) = \tilde{\mu}_t(q).$$

Example 3. *Let us consider the marked probabilistic time Petri net depicted in Fig. 3.7, whose initial marking is given by* $\rho_{\mathcal{N}} = \delta_{(1,0,0,1,0,0,0)}$*. Since the enabled transition* T_4 *is not firable before date* 2*, all paths in* $\mathcal{N}_1$ *start with the resolution of the trial* T_1*, which either generates a token in* P_2 *or in* P_3*. Every scheduler must first choose when to fire that transition. Depending on the outcome of the trial* T_1*, a scheduler is not presented with the same opportunities. Let us consider a scheduler* $\mathfrak{S}_1$ *that chooses to fire* T_1 *immediately. If a token ends up in* P_2*, then* $\mathfrak{S}_1$ *is constrained by the deterministic trial* T_3 *which ends up being performed at date 1. If a token ends up in* P_3*, then* $\mathfrak{S}_1$ *must let time flow before performing either* T_2 *or* T_4*.*

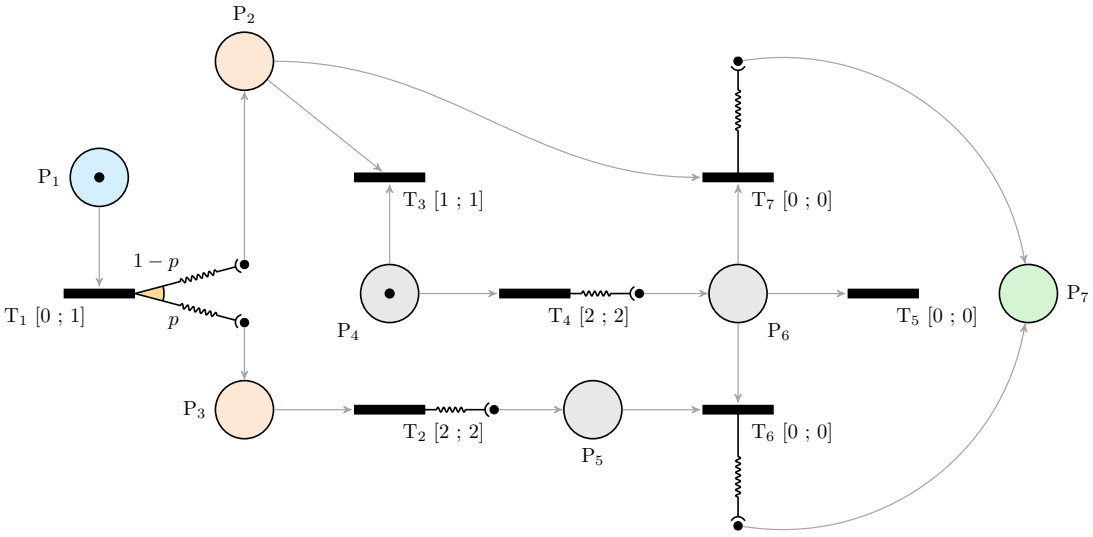


Figure 3.7 – The probabilistic time Petri net $\mathcal{N}_1$

Assume the goal is to reach place P_7. The target set consists of every marking M for which $M(P_7) > 0$. Figure 3.8 depicts the choices scheduler $\mathfrak{S}_1$ makes as it resolves all non-determinism before reaching P_7 with probability p. While scheduler $\mathfrak{S}_1$ does exhibit a path leading to a target marking, a thorough study of the likelihood of reaching those particular markings is needed.

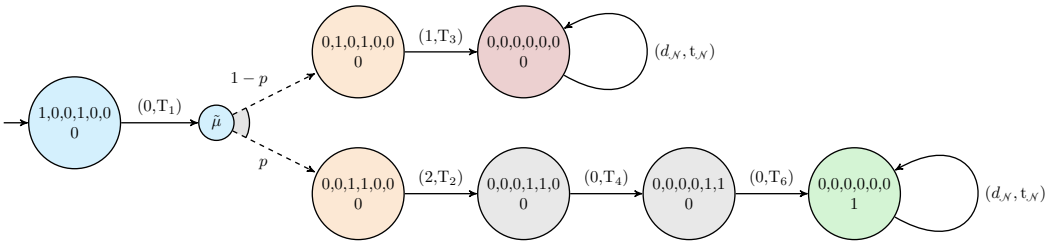


Figure 3.8 – Abridged representation of the scheduler $\mathfrak{S}_1$

Intuitively, the deterministic trials T_3 and T_5 must be avoided at all costs if P_7 is to be reached. This stems from the fact that these trials eliminate the tokens that are needed to fire T_6 or T_7. To avoid T_3, the trial T_1 must necessarily be resolved at date 1 and no sooner than that. To avoid T_5, the trial T_1 must necessarily be resolved at date 0, without delay. Schedulers that do not fire T_1 at date 0 or at date 1 never reach P_7. Therefore, the minimum probability of reaching P_7 is 0. Since a scheduler has no influence over the outcome of T_1, it has no way of knowing if firing T_1 at date 0 or at date 1 is best. As a result, the probability of reaching P_7 can be no greater than $max(p, 1-p)$.

The probabilistic real-time reachability problem consists in the establishment of these lower and upper probability bounds. The whole set of schedulers of a probabilistic time Petri net is considered in order to compute these bounds, as they cover every possible resolution of non-determinism. This corresponds to a worst-case analysis.

3.4.2 Paths and Adversaries

Since a probabilistic time Petri net evolves in a dense-time environment, there are usually infinitely many schedulers as soon as a single firing interval is a proper interval. To compute the lower and upper probability bounds by ranging over all schedulers, it is natural to proceed to a grouping of states that leads to the formation of *state classes*. The needed information is captured in a finite graph, called a state class graph, which can then be used to apply formal verification techniques. The state space of the net thus takes the form of a Markov decision process.

For probabilistic time Petri nets, the paths in a Markov decision process resolve both probabilistic and non-deterministic choices.

Definition 11 (Path in a Markov decision process)**.** *Let $\mathscr{M} = (C, \rho, A, \mathbf{P})$ be a Markov decision process.*

- *A finite* path *in the Markov decision process $\mathscr{M}$ is a finite sequence*

$$c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} \dots \xrightarrow{\alpha_n} c_n$$

where for all $i \in [\![0, n-1]\!]$,

$$\begin{cases} c_i \in C, \\ \alpha_{i+1} \in \Sigma(c_i), \\ c_{i+1} \in Supp(\mathbf{P}(c_i, \alpha_{i+1})). \end{cases}$$

The integer n is called the length *of the path. A finite path in $\mathscr{M}$ is an element of $Supp(\rho) \times (A \times C)^*$. The set of finite paths in the Markov decision process $\mathscr{M}$ is denoted $Path^*_{(\mathscr{M})}$.*

- *An infinite* path *in the Markov decision process $\mathscr{M}$ is an infinite sequence*

$$c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} c_2 \xrightarrow{\alpha_3} \dots$$

where $c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} \dots \xrightarrow{\alpha_n} c_n \in Path^*_{(\mathscr{M})}$ for all $n \in \mathbb{N}$.

An infinite path in $\mathscr{M}$ is an element of $(C \times A)^\infty$. The set of infinite paths in the Markov decision process $\mathscr{M}$ is denoted $Path^\infty_{(\mathscr{M})}$.

An adversary of a Markov decision process fulfils the same function a scheduler does for a probabilistic time Petri net.

Definition 12 (Adversary of a Markov decision process)**.** *Let $\mathscr{M} = (C, \rho, A, \mathbf{P})$ be a Markov decision process.*

For a given finite path $\sigma = c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n} c_n$ in $\mathscr{M}$, let $last(\sigma)$ denote the state c_n.

- *An* adversary *of the Markov decision process $\mathscr{M}$ is a (total) function $\Lambda : Path^*_{(\mathscr{M})} \to A$ such that for all finite paths $\sigma = c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n} c_n$ in $Path^*_{(\mathscr{M})}$*

$$\Lambda(\sigma) \in \Sigma(last(\sigma)).$$

A finite or infinite path $\sigma = c_0 \xrightarrow{\alpha_1} c_1 \xrightarrow{\alpha_2} \ldots \xrightarrow{\alpha_n} c_n$ of $\mathscr{M}$ is called a Λ-path *if $\Lambda(\sigma_{|i}) = \alpha_{i+1}$ for all prefixes $\sigma_{|i}$ of σ (the path $\sigma_{|i}$ denotes the finite prefix of σ of length i). Let $Path^*_\Lambda$ denote the (countable) set of finite Λ-paths.*

- *An adversary Λ of the Markov decision process $\mathscr{M}$ induces a Markov chain $\mathscr{M}_\Lambda = (Path^*_\Lambda, \rho_\Lambda, \mathbf{P}_\Lambda)$ where*
  - *ρ_Λ is the distribution of initial paths of the chain. Its support is equal to the finite paths in $\mathscr{M}$ of length 0 that are also initial states of the process. For all $c \in Supp(\rho_\Lambda)$, $\rho_\Lambda(c) = \rho(c)$.*
  - *$\mathbf{P}_\Lambda : Path^*_\Lambda \to \mathscr{D}ist_{Path^*_\Lambda}$ is the (total) probabilistic transition function of $\mathscr{M}_\Lambda$. For $\sigma \in Path^*_\Lambda$, the support of $\mathbf{P}_\Lambda(\sigma)$ is the set of Λ-paths of the form $\sigma \xrightarrow{\Lambda(\sigma)} c$. For $(\sigma, c) \in Path^*_\Lambda \times C$,*

$$\mathbf{P}_\Lambda(\sigma)(\sigma \xrightarrow{\Lambda(\sigma)} c) = \mathbf{P}(last(\sigma), \Lambda(\sigma))(c).$$

3.4.3 The Probabilistic Strong State Class Graph

Time Petri nets typically generate an infinite state space. The *linear state class graph* was introduced in [108] and [109] in order to capture linear time temporal properties of time Petri nets in a finite graph. Intuitively, each class is an element of $\mathbb{N}^P \times \mathscr{P}(\mathbb{R}^T)$ which captures all the states that are reachable from an initial state class by firing schedules of a given support. Since there are generally infinitely many supports, state classes are then

considered modulo some equivalence relation. The graph thus becomes *finite* if the net is *bounded*.

The probabilistic strong state class graph extends the construction methods that are proposed in the literature to account for the probabilistic nature of PTPNs. The following definition introduces *strong state classes* for probabilistic time Petri nets and details how the successor of a class is obtained when firing a given transition.

Definition 13 (Strong state classes)**.** *Let* $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ *be the semantics of a marked probabilistic time Petri net* $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$*. The set of* strong state classes *is defined as follows:*

1. *For a given transition* t *of the net* $\mathcal{N}$*, the set* $\Delta(t)$ *of* decoupled trials *of* t *is defined as* $\Delta(t) = \left\{ t_\omega \in \mathcal{D}ist_{\mathbb{N}^P} \mid \exists \omega \in Supp(\mu_t), t_\omega = \delta_\omega \right\}$ *and denote by* $T_\Delta = \bigcup_{t \in T} \Delta(t)$ *the set of decoupled trials in* $S_{\mathcal{N}}$*.*
2. *For a given initial state* $q_0 \in Supp(\rho_{S_{\mathcal{N}}})$*, let* $C_{q_0} = \bigcup_{\tau \in T_\Delta^*} c_\tau$ *be a cover of* Q*, defined inductively by* $c_\varepsilon = \{q_0\}$ *and*

$$c_{\tau t_\omega} = \left\{ \begin{array}{l} (M', v') \in \mathbb{N}^P \times \mathbb{R}_+^T \mid \exists (M, v) \in c_\tau, \exists (d, t) \in \Phi(M, v), t_\omega \in \Delta(t) \\ \quad and\ \forall t' \in T, v'(t') = (v(t') + d) \times (1 - \chi_t(t')) \times \chi_{\mathcal{E}(M - Pre(t))}(t') \\ \quad and\ M' = (M - Pre(t)) + \omega \end{array} \right\}$$

   The classes $c_{\tau t_\omega}$ *are the* successors *of the state class* c_τ*.*
3. *The cover* C_{q_0} *denotes the set of nodes of the tree that is generated by* q_0*. All of the trees that are generated by an initial state of the net must be taken into account. Finally, let*

$$C = \bigcup_{q_0 \in Supp(\rho_{S_{\mathcal{N}}})} C_{q_0}.$$

Let $c \in C$ be a state class in which the shared marking is M. A transition $t \in \mathcal{E}(M)$ is said to be *firable* from the state class c if there exists a state $q \in c$ such that t is firable from q. The *probabilistic strong state class graph* (which remains *finite* if the net is *bounded*) is defined as follows:

Definition 14 (Probabilistic strong state class graph)**.** *Let* $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ *be the semantics of a marked probabilistic time Petri net* $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$*. The* probabilistic strong state class graph *of the net* $\mathcal{N}$ *is a Markov decision process* $\mathcal{M} = (C, \rho, \tilde{T}, \mathbf{P})$ *where*

- C *is the set of* strong state classes*,*

- $\rho : C \to [0,1]$ *is the distribution of initial classes of the graph. The support of* ρ *is equal to the set of singletons* $\{q_0\}$*, where* $q_0 \in Supp(\rho_{S_{\mathcal{N}}})$*. For all* $(M_0, 0_T) \in Supp(\rho_{S_{\mathcal{N}}})$*,*

$$\rho\Big(\Big\{(M_0, 0_T)\Big\}\Big) = \rho_{S_{\mathcal{N}}}(M_0, 0_T) = \rho_{\mathcal{N}}(M_0).$$

- $\mathbf{P} : C \times \tilde{T} \to \mathscr{D}ist_C$ *is the (partial) transition probability function.*
  1. $\mathbf{P}$ *is defined for* $(c,t) \in C \times T$ *if and only if the transition* t *is firable from the state class* c*. In that case, let* $\mathbf{P}(c,t) = \widehat{\mu}_t$*, where* $\widehat{\mu}_t \in \mathscr{D}ist_C$ *is defined as follows:*
     — *Let* $c' \in C$*. The class* c' *lies in* $Supp(\widehat{\mu}_t)$ *if and only if* c' *is the successor of* c *for some decoupled transition* $t_\omega \in \Delta(t)$*.*
     — *Suppose that* $c' \in Supp(\widehat{\mu}_t)$*. We define the image of* c' *by the formula*

$$\widehat{\mu}_t(c') = \mu_t(\omega).$$

  2. $\mathbf{P}$ *is defined for* $(c,t) \in C \times \{t_{\mathcal{N}}\}$ *if and only if* $\mathscr{C}(q) = \mathbb{R}_+$ *for some* $q \in c$*. In that case,* $\mathscr{C}(q) = \mathbb{R}_+$ *for all* $q \in c$ *since all the states in* c *have the same marking. The image of* $(c, t_{\mathcal{N}})$ *is defined by the formula*

$$\mathbf{P}(c, t_{\mathcal{N}}) = \delta_c.$$

■

The probabilistic strong state class graph of the probabilistic time Petri net $\mathcal{N}_1$ is represented in Fig. 3.9. Each class is represented by a node, which is labelled with the marking that all states share in that particular class. Here, strong state classes are considered modulo an equivalence relation $\equiv$ that asserts that two classes are equivalent if they denote the same set of states. For the sake of clarity, time domains are not represented.

Let us now consider the adversary Λ_1 of the probabilistic state class graph of $\mathcal{N}_1$, depicted in Fig. 3.10. Depending on the outcome of the trial T_1, it either performs the untimed sequence of actions $\mathrm{T}_1 \to \mathrm{T}_4 \to \mathrm{T}_7$ or the untimed sequence $\mathrm{T}_1 \to \mathrm{T}_2 \to \mathrm{T}_4 \to \mathrm{T}_6$ before reaching a target state. However, there is no scheduler for $\mathcal{N}_1$ that can perform both of these paths since the path $\mathrm{T}_1 \to \mathrm{T}_4 \to \mathrm{T}_7$ can only be performed when T_1 is fired at date 1 while the path $\mathrm{T}_1 \to \mathrm{T}_2 \to \mathrm{T}_4 \to \mathrm{T}_6$ can only be performed when T_1 is fired at date 0. As a result, the probabilistic strong state class graph potentially generates duplicitous adversaries, which display a probability of reaching target states greater than that of any scheduler.

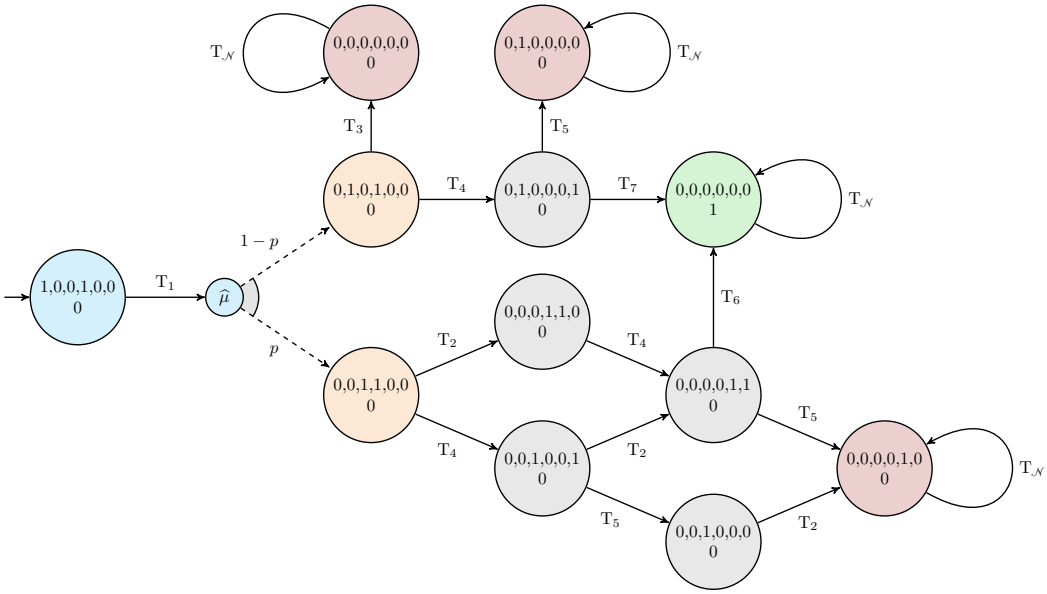


Figure 3.9 – The probabilistic strong state class graph of the probabilistic time Petri net $\mathcal{N}_1$

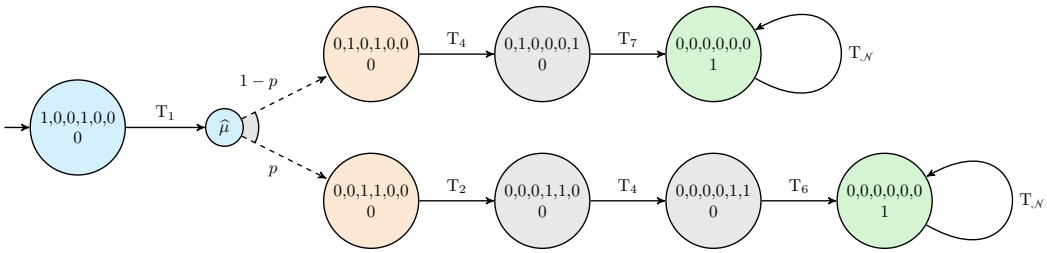


Figure 3.10 – Abridged representation of the duplicitous adversary Λ_1

3.4.4 The Probabilistic Atomic State Class Graph

The reason why the probabilistic strong state class graph fails to produce proper adversaries lies in the way time and probabilities are intertwined in probabilistic time Petri nets. A graph that better captures the effect the firing date of probabilistic trials has on future actions is needed in order to solve the probabilistic real-time reachability problem.

Berthomieu and Vernadat introduced the atomic state class graph for time Petri nets in order to preserve their branching time temporal properties in a finite graph [111]. The construction of this graph can be adapted for probabilistic time Petri nets to preserve the adversaries that are needed. Let us consider the following properties of interest for state class graphs:

- (EE) For all classes $c, c' \in \mathbb{N}^P \times \mathscr{P}(\mathbb{R}^T)$ and for all $t \in \Sigma(c)$,

$$c \xrightarrow{t} c' \in Path^*_{(\mathscr{M})} \iff \exists q \in c, \exists q' \in c', \exists d \in \mathbb{R}_+, \begin{cases} (d,t) \in \Phi(q) \\ q \xrightarrow{(d,t)} q' \in Path^*_{(S_{\mathcal{N}})} \end{cases}$$

- (AE) For all classes $c, c' \in \mathbb{N}^P \times \mathscr{P}(\mathbb{R}^T)$ and for all $t \in \Sigma(c)$,

$$c \xrightarrow{t} c' \in Path^*_{(\mathscr{M})} \implies \forall q \in c, \exists q' \in c', \exists d \in \mathbb{R}_+, \begin{cases} (d,t) \in \Phi(q) \\ q \xrightarrow{(d,t)} q' \in Path^*_{(S_{\mathcal{N}})} \end{cases}$$

State class graphs typically satisfy property (EE) and so does the probabilistic strong state class graph. The *probabilistic atomic state class graph* introduced in this section is built from the probabilistic strong state class graph, by refining its classes into *atomic* ones. An atomic class is a state class in which each state has a successor in each of the successors of that class. Intuitively, each atomic class captures all the states that are reachable from an initial state by firing schedules of a given support *during certain time windows*.

The algorithm that details how to split strong state classes into atomic ones can be found in [111]. Splitting a class c replaces it with a pair of classes which both inherit the predecessors of c and the successors of c that they can still reach. This technically causes multiple hyperarcs leaving the predecessors of c to have the same label. However, each one of these hyperarcs is implicitly augmented with a time interval. This time window corresponds to the set of delays that enforce property (EE) in each one of the states it leads to. Since no time delay is shared among those hyperarcs, any ambiguity is lifted.

This stable refinement enforces property (AE) in the probabilistic atomic state class graph, which once again takes the form of a Markov decision process $\mathscr{M}_A = (C_A, \rho_A, \tilde{T}, \mathbf{P}_A)$. However, this graph is usually significantly bigger than the probabilistic state class graph from which is it built. The probabilistic atomic state class graph of the probabilistic time Petri net $\mathcal{N}_1$ is represented in Fig. 3.11.

Theorem 1. *Let $\mathcal{N} = (P, T, Pre, Post, I, \rho_{\mathcal{N}})$ be a bounded marked probabilistic time Petri net, let $S_{\mathcal{N}} = (Q, \rho_{S_{\mathcal{N}}}, T, W)$ be its semantics and let $\mathscr{M}_A = (C_A, \rho_A, \tilde{T}, \mathbf{P}_A)$ be the probabilistic atomic state class graph of $\mathcal{N}$.*

1. *Let Λ be an adversary of the Markov decision process $\mathscr{M}_A$. There exists a scheduler for the probabilistic timed transition system $S_{\mathcal{N}}$ that induces the same Markov chain as Λ up to isomorphism.*

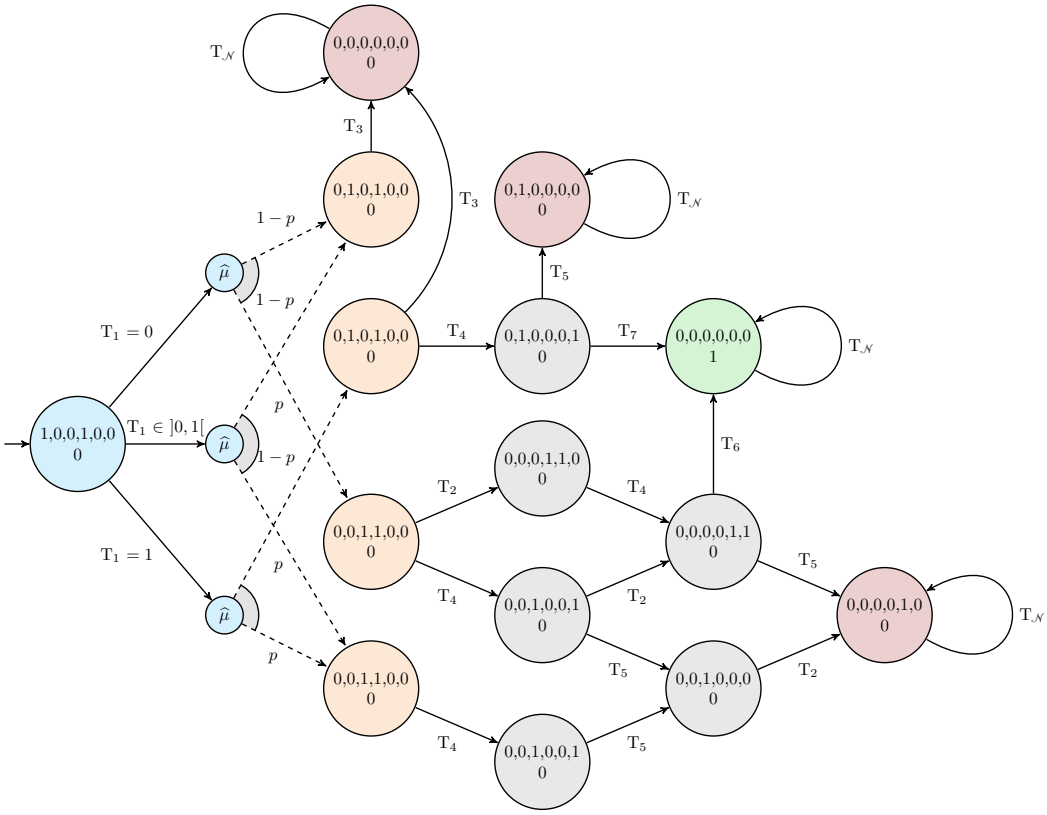


Figure 3.11 – The probabilistic atomic state class graph of the PTPN $\mathcal{N}_1$

2. *Conversely, let $\mathfrak{S}$ be a scheduler for the probabilistic timed transition system $S_{\mathcal{N}}$. There exists an adversary of the Markov decision process $\mathscr{M}_A$ that induces the same Markov chain as $\mathfrak{S}$ up to isomorphism.*

Proof. 1. For a given adversary Λ of the Markov decision process $\mathscr{M}_A$, let us define a scheduler $\mathfrak{S} : Path^*_{(S_{\mathcal{N}})} \to (\mathbb{R}_+ \times T) \cup \{(d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}})\}$ for the probabilistic timed transition system $S_{\mathcal{N}}$ as follows:

- Let $\pi = q_0 \xrightarrow{d_1,t_1} \dots \xrightarrow{d_n,t_n} q_n \in Path^*_{(S_{\mathcal{N}})}$ be a finite path in the probabilistic timed transition system $S_{\mathcal{N}}$. According to property (EE), there exists a path $\sigma = c_0 \xrightarrow{t_1} \dots \xrightarrow{t_n} c_n \in Path^*_{(\mathscr{M}_A)}$ in the probabilistic atomic state class graph $\mathscr{M}_A$ such that $q_i \in c_i$ for all $i \in [\![0, n]\!]$. Let $q_n = (M_n, v_n)$.
  - If $\Lambda(\sigma) = \mathrm{t}_{\mathcal{N}}$, then $\mathrm{t}_{\mathcal{N}} \in \Sigma(c_n)$. It follows that $\mathscr{C}(q) = \mathbb{R}_+$ for all $q \in c_n$. In particular, $\mathscr{C}(q_n) = \mathbb{R}_+$. We can thus set

$$\mathfrak{S}(\pi) = (d_{\mathcal{N}}, \mathrm{t}_{\mathcal{N}}).$$

  - If $\Lambda(\sigma) \neq \mathrm{t}_{\mathcal{N}}$, then property (AE) guarantees the existence of a delay $d_{n+1} \in \mathscr{C}(M_n, v_n)$ such that $\Lambda(\sigma) \in \mathscr{F}(M_n, v_n + d_{n+1})$. In that case,

$(d_{n+1}, \Lambda(\sigma)) \in \Phi(q_n)$. We can thus set

$$\mathfrak{S}(\pi) = (d_{n+1}, \Lambda(\sigma)).$$

- We will now demonstrate that the Markov chain $\mathscr{M}_{\mathfrak{S}} = (Path^*_{\mathfrak{S}}, \rho_{\mathfrak{S}}, \mathbf{P}_{\mathfrak{S}})$ induced by $\mathfrak{S}$ and the Markov chain $\mathscr{M}_{\Lambda} = (Path^*_{\Lambda}, \rho_{\Lambda}, \mathbf{P}_{\Lambda})$ induced by Λ are the same, up to isomorphism. To do so, we will introduce a bijection ζ that maps the nodes of the chain $\mathscr{M}_{\mathfrak{S}}$ to the nodes of the chain $\mathscr{M}_{\Lambda}$. We will then show that this mapping is a graph isomorphism by proving that it preserves the structure of $\mathscr{M}_{\mathfrak{S}}$, as defined by the distribution of initial paths $\rho_{\mathfrak{S}}$ and the probabilistic transition function $\mathbf{P}_{\mathfrak{S}}$, in the chain $\mathscr{M}_{\Lambda}$.

  (a) Let us define the canonical bijection ζ between the $\mathfrak{S}$-paths of $Path^*_{\mathfrak{S}}$ and the Λ-paths of $Path^*_{\Lambda}$ as follows:
  Let $\pi = q_0 \xrightarrow{d_1,t_1} \dots \xrightarrow{d_n,t_n} q_n$ be a $\mathfrak{S}$-path. According to property EE, there exists a path $\sigma = c_0 \xrightarrow{t_1} \dots \xrightarrow{t_n} c_n \in Path^*_{(\mathscr{M}_A)}$ in the probabilistic atomic state class graph $\mathscr{M}_A$ that verifies $q_i \in c_i$ for all $i \in [\![0, n]\!]$. Since every prefix $\pi_{|i}$ of π is a $\mathfrak{S}$-path, it follows that $\Lambda(\sigma_{|i}) = t_{i+1}$ for all prefixes $\sigma_{|i}$ of σ, by definition of the scheduler $\mathfrak{S}$. This implies that σ is a Λ-path and that such a path is unique for any given π. Let

  $$\zeta : Path^*_{\mathfrak{S}} \to Path^*_{\Lambda}$$

  be the function that maps each finite $\mathfrak{S}$-path π to its correponding Λ-path σ.

  — Let us prove that ζ is injective. Let $\pi = q_0 \xrightarrow{d_1,t_1} \dots \xrightarrow{d_n,t_n} q_n$ and $\pi' = q'_0 \xrightarrow{d'_1,t'_1} \dots \xrightarrow{d'_n,t'_n} q'_n$ be two Λ-paths, such that $\zeta(\pi) = \zeta(\pi')$. Let $\zeta(\pi) = c_0 \xrightarrow{t_1} \dots \xrightarrow{t_n} c_n$. Since π' is a $\mathfrak{S}$-path and $q'_i \in c_i$ for all $i \in [\![0, n]\!]$, it follows that $(d'_{i+1}, t'_{i+1}) = \mathfrak{S}(\pi_{|i}) = (d_{i+1}, t_{i+1})$ for all $i \in [\![0, n-1]\!]$ and $q_i = q'_i$ for all $i \in [\![0, n]\!]$. Consequently, the equality $\pi = \pi'$ holds. This proves that the function ζ is injective.

  — Let us prove that ζ is surjective. Let $\sigma = c_0 \xrightarrow{t_1} \dots \xrightarrow{t_n} c_n$ be a Λ-path. The only $\mathfrak{S}$-path $\pi = q_0 \xrightarrow{d_1,t_1} \dots \xrightarrow{d_n,t_n} q_n$ that verifies $q_i \in c_i$ for all $i \in [\![0, n]\!]$ also verifies $\zeta(\pi) = \sigma$. Consequently, the function ζ is surjective.

  (b) For every $q_0 = (M_0, 0_T) \in Supp(\rho_{\mathfrak{S}})$, $\zeta(q_0) = \{q_0\} \in Supp(\rho_{\Lambda})$.

  — The distribution of initial paths $\rho_{\mathfrak{S}}$ of the Markov chain induced by the scheduler $\mathfrak{S}$ is defined (Def. 10) from the distribution of initial

markings $\rho_{S_{\mathcal{N}}}$ of the probabilistic timed transition system $S_{\mathcal{N}}$ (Def. 5) as follows:

$$\rho_{\mathfrak{S}}(q_0) = \rho_{S_{\mathcal{N}}}(q_0) = \rho_{\mathcal{N}}(M_0).$$

— The distribution of initial markings ρ_Λ of the Markov chain induced by the adversary Λ is defined (Def. 12) from the distribution of initial classes ρ_A of the atomic state class graph, which is the same as the distribution of initial classes of the strong state class graph (Def. 14):

$$\rho_\Lambda(\{q_0\}) = \rho_A(\{q_0\}) = \rho_{\mathcal{N}}(M_0).$$

Therefore,

$$\rho_{\mathfrak{S}}(q_0) = \rho_\Lambda(\zeta(q_0)).$$

(c) Let π be a Λ-path and $\sigma = \zeta(\pi)$ be the $\mathfrak{S}$-path that is canonically associated with π. Let $q = (M, v) \in \mathbb{N}^P \times \mathbb{R}_+^T$ be a state of $\mathcal{N}$ such that $\pi \xrightarrow{\mathfrak{S}(\pi)} q$ is a $\mathfrak{S}$-path and let $\sigma \xrightarrow{\Lambda(\sigma)} c = \zeta(\pi \xrightarrow{\mathfrak{S}(\pi)} q)$.

— By definition of the partial probabilistic transition function $\mathbf{P}_{\mathfrak{S}}$ (Def. 10) of the scheduler $\mathfrak{S}$ and by definition of the probability distribution of states $\tilde{\mu}_t$ (Def. 5),

$$\begin{aligned}\mathbf{P}_{\mathfrak{S}}(\pi)(\pi \xrightarrow{\mathfrak{S}(\pi)} q) &= \tilde{\mu}_t(q) \\ &= \mu_t(\omega_M).\end{aligned}$$

— By definition of the partial probabilistic transition functions $\mathbf{P}_\Lambda$ of the adversary Λ and $\mathbf{P}_A$ of the atomic state class graph (Def. 12) and by definition of the probability distribution of classes $\widehat{\mu}_t$ (Def. 14),

$$\begin{aligned}\mathbf{P}_\Lambda(\sigma)(\sigma \xrightarrow{\Lambda(\sigma)} c) &= \mathbf{P}_A(last(\sigma), \Lambda(\sigma))(c) \\ &= \widehat{\mu}_t(c) \\ &= \mu_t(\omega_M).\end{aligned}$$

Therefore,

$$\mathbf{P}_{\mathfrak{S}}(\pi)(\pi \xrightarrow{\mathfrak{S}(\pi)} q) = \mathbf{P}_\Lambda(\zeta(\pi))(\zeta(\pi \xrightarrow{\mathfrak{S}(\pi)} q)).$$

This proves that ζ is edge-preserving from a probabilistic standpoint. It follows that ζ is a Markov chain isomorphism. The forest of trees generated by the Markov chains $\mathscr{M}_{\mathfrak{S}}$ and $\mathscr{M}_\Lambda$ are therefore identical up to isomorphism.

2. Conversely, let $\mathfrak{S}$ be a scheduler for $S_{\mathcal{N}}$. Let us define an adversary $\Lambda : Path^*_{(\mathscr{M}_A)} \to$

$\tilde{T}$ of the probabilistic atomic state class graph as follows:

- Let $\sigma = c_0 \xrightarrow{t_1} c_1 \xrightarrow{t_2} \dots \xrightarrow{t_n} c_n \in Path^*_{(\mathscr{M}_A)}$ be a finite path in the probabilistic atomic state class graph $\mathscr{M}_A$. According to property (EE), there exists a path $\pi = q_0 \xrightarrow{d_1,t_1} q_1 \xrightarrow{d_2,t_2} \dots \xrightarrow{d_n,t_n} q_n \in Path^*_{(S_{\mathscr{N}})}$ in the probabilistic timed transition system $S_{\mathscr{N}}$ such that $q_i \in c_i$ for all $i \in [\![0, n]\!]$.
  - If $\mathfrak{S}(\pi) = (d_{\mathscr{N}}, \mathrm{t}_{\mathscr{N}})$, then $\mathscr{C}(q_n) = \mathbb{R}_+$. Therefore $\mathbf{P}_A$ is defined for $(c_n, \mathrm{t}_{\mathscr{N}})$ and $\mathrm{t}_{\mathscr{N}} \in \Sigma(c_n)$ as a result. We can thus set
  $$\Lambda(\sigma) = \mathrm{t}_{\mathscr{N}}.$$
  - If $\mathfrak{S}(\pi) = (d_{n+1}, t_{n+1}) \in \Phi(q_n)$, then the sequence $\sigma.t_{n+1}$ is the support of a path of $Path^*_{(S_{\mathscr{N}})}$. The transition $t_{n+1} \in T$ is therefore firable from the state class c_n and $\mathbf{P}_A$ is defined for (c_n, t_{n+1}). In that case, $t_{n+1} \in \Sigma(c_n)$. Let
  $$\Lambda(\sigma) = t_{n+1}.$$
  Choosing t_{n+1} in $\Sigma(c_n)$ can be ambiguous since there is potentially more than one output hyperarc of c_n that is labelled with t_{n+1} as a result of the splitting process. However, since each one of these hyperarcs is implicitly augmented with a time window that correponds to the set of delays that are allowed before the firing of t_{n+1}, the chosen arc can only be the one that allows a time delay of d_{n+1}.

- The graph isomorphism introduced in 1. can be used to prove that the Markov chain $\mathscr{M}_{\mathfrak{S}} = (Path^*_{\mathfrak{S}}, \rho_{\mathfrak{S}}, \mathbf{P}_{\mathfrak{S}})$ induced by $\mathfrak{S}$ and the Markov chain $\mathscr{M}_\Lambda = (Path^*_\Lambda, \rho_\Lambda, \mathbf{P}_\Lambda)$ induced by Λ are the same, up to isomorphism.

□

As a result of theorem 1, the probabilistic real-time reachability problem can be solved by computing the probability of reaching a target state for every adversary of the probabilistic atomic state class graph (with the tools commonly used for Markov decision processes). For example, it can easily be shown that the sought probability bounds for reaching P7 in the net $\mathscr{N}_1$ (Fig. 3.7) are indeed 0 and $max(p, 1-p)$, by considering all the adversaries of its probabilistic atomic state class graph (Fig. 3.11). In fact, an array of algorithms can now be used to prove that the net verifies the following properties:

- **reachability**: the net $\mathscr{N}_1$ can reach P_7 with probability (at least) 0.5,
- **inevitability**: the net $\mathscr{N}_1$ inevitably leaves P_1 with probability 1,

- **time bounded reachability**: the net $\mathcal{N}_1$ can reach P_7 within two time units with probability 0.5,
- **bounded response**: the net $\mathcal{N}_1$ inevitably reaches P_5 or P_7 within two time units with probability 1 after reaching the marking $(0, 0, 1, 1, 0, 0, 0)$.

3.5 Conclusion

A new formalism was introduced for the modelling of concurrent probabilistic real-time systems. This new model extends time Petri nets by enhancing the forward incidence mapping with sets of probability distributions. Probabilistic time Petri nets natively integrate time, concurrency and probabilities. In the spirit of probabilistic timed automata [113], any random phenomenon has been limited to the discrete behaviour of a time Petri net. Time and concurrency are still resolved in a non-deterministic manner. The atomic state class graph construction of TPNs can be adapted for this new model to obtain a Markov decision process that induces the same Markov chains as the semantics of the PTPN. Therefore, the use of a wide range of tools for the analysis of PTPN is made available. Unfortunately, the simpler non-atomic state class graph construction cannot be adapted in a similar manner. This formalism is used for modelling purposes in the following chapters.

Chapter 4

A Formal Approach for the Design of a Dependable Perception System that Supports Graceful Degradation

4.1 Introduction

An Automated Driving System must respond appropriately to a component or system failure and to deteriorating environmental conditions. It must position itself correctly and in good time, take proper observation, react promptly and properly to hazards, adjust speed and comply with road traffic control. These necessities provide incentive for the development of a safety module that understands the system's capabilities and forces it into a suitable degraded mode whenever necessary. In particular, the quality of sensors and perception algorithms must be properly assessed, as it conditions the system's ability to perceive the driving environment. An ADS acting on erroneous or incomplete data is prone to making decisions that can lead to hazardous situations. Yet sensors are affected by changing weather and lighting conditions. The concern for predictable behaviour, safe operation and mission success motivates the need to design multi-sensor data fusion systems from a safety perspective.

Efforts towards building a flexible safety module as support for online reconfiguration of Automated Driving Systems are few. While conceptual frameworks that provide adaptive graceful degradation do exist [114], the development of a standard evaluation framework to assess the performance of data fusion algorithms requires further investigation. A fuzzy rule based strategy was proposed [115] in an attempt to evaluate the dependability of a set of embedded sensors. A real-time, multi-sensor architecture for the fusion of delayed observations was later presented [116] for the design and implementation of data fusion applications. The handling of sensor uncertainty in systems with

formal specifications has also been studied in recent years. Probabilistic model checking techniques were used to compute the probability with which an automatically synthesised controller satisfies a set of high-level specifications [117]. Linear temporal logic was used to reduce a stochastic control problem to a deterministic short path problem [118]. Yet in spite of these preliminary efforts, the particular problem of ADS perception system design remains largely unexplored.

A comprehensive review of ADS perception technology was recently published [119]. It provides useful information about the advantages, disadvantages, limits and ideal applications of specific sensors. Safety issues linked to ADS software architecture are classified as largely unaddressed by the authors. Single cameras are mainly used for lane detection and obstacle detection and classification. These sensors provide a wide field of view while maintaining good resolution. They perform poorly in suboptimal weather conditions however, and algorithms usually require a lot of computing power. Stereo-vision provides the added benefit of depth perception and is used for 3D mapping. As with single cameras, velocity information still needs to be computed. Lidars are used for obstacle detection, 3D mapping and lane detection. They provide direct distance measurements, large field of views with high resolution at medium range [120]. Some even have internal mechanisms to limit the impact of poor weather conditions. They perform poorly in certain weather conditions (rain, fog, snow) and do not perform as well as vision for object classification. Radars are used for obstacle detection. They provide high accuracy, even in poor weather conditions. They suffer from poor static object and pedestrian detection, poor classification and small field of views for long range radars. Sonars are mainly used for near obstacle detection. They provide direct distance measurement at close range and can operate in fog and snow. They suffer from poor angular resolution and poor detection beyond 2 m. Beyond these preliminary, empirical observations, it can be difficult to determine which combination of sensors is best suited for a given application. Sensors used for advanced driver-assistance systems can be found in [121].

In this chapter, a formal framework is presented for the design of multi-sensor data fusion systems. The aim is two-fold. First, a tool for the design of a perception system that inherently supports adaptive graceful degradation is presented. Second, the rules that are to be applied by the underlying safety module during operation are generated. They enable efficient re-allocation of resources when a sensor or a processing board failure occurs. By specifying the desired properties of the perception system in a formal language, the extensive set of tools of the formal methods community can be used to synthesise one that meets these requirements. Its characteristics are expressed in a language that the safety module can understand. This allows for adaptive graceful degradation and efficient online reconfiguration.

Automated fusion is part of ongoing research in data fusion. It has led to similar attempts within formal logic and category theory frameworks. For example, an outline of a formalisation of classes of information fusion systems was proposed in terms of category theory and formal languages [122]. A formal approach was also presented [123] to tackle the problem of synthesising algorithms that satisfy templates for information fusion. The approach that is proposed here relies on the construction of a high-level probabilistic time Petri net containing inhibitor arcs and read arcs. The model integrates concurrency, real-time constraints and probabilities. Petri nets have been successfully used for the study of real-time scheduling problems [124] and for the design of automated manufacturing systems, both in their simple and high-level form [125]. While simple Petri nets are a powerful formalism in terms of expressivity and conciseness, we argue that they can still lead to cluttered models when the modelled objects carry attributes. Probabilistic time Petri nets [126] were defined in the previous chapter. This class of nets is extended here by allowing tokens to have arbitrarily defined attributes. The resulting high-level model is more compact while being semantically equivalent to its simple counterpart.

4.2 Perception

4.2.1 System Performance

Fault tolerance consists in the estimation and minimisation of faults. Assume a processing board running a pedestrian detection algorithm experiences a failure. If the ADS is driving in an urban area, the presence of pedestrians is very likely. Hence the system must run the pedestrian detection algorithm in a degraded mode on another live processing board and perhaps even exert stringent constraints on the maximum speed of the vehicle. Similarly, if a radar for blind spot detection does not work properly, another sensor must be used instead until the vehicle can safely be brought to a stop. These examples show the importance of graceful degradation application, as it enables the system to recover from a failure with less resources. Designing the perception system of an ADS with adaptive graceful degradation in mind requires some form of system monitoring as well as redundancy, to maintain minimal system performance requirements. This means that some definition of system performance must first be provided. This is an especially difficult topic to address in the context of driving automation for what it means for an ADS to properly perceive its environment bears no easy answer.

Most studies on the subject of ADS perception focus on object detection, recognition and classification. The performance of classifiers is generally given through the use of a confusion matrix. Tables of confusion are also used to report the number of false positives, false negatives, true positives and true negatives, allowing for a more detailed analysis

than the mere proportion of correct classification. Other metrics can be derived from a confusion matrix, such as prevalence, accuracy, precision, recall, fall-out, specificity, false omission rate and F_1 score. However, comparing multiple, independent studies is close to impossible as there is much variability in terms of hardware, software and the database each system was tested on. These differences are not always made apparent, as either the environment, the hardware or the sensor fusion scheme lacks precise description. Moreover, the conditions in which individual algorithms or the whole perception system were tested are rarely detailed in a thorough manner. Consequently, it is difficult to interpret and give meaning to the results that have been obtained in terms of ADS capabilities, with respect to some operational design domain. It is important to note that the vehicle's speed has an impact on ADS perception capabilities. As a result, performance evaluation of algorithms on data obtained by a stationary ADS may not be relevant for the validation of an ADS driving on a highway.

The KITTI vision benchmark suite [127] proposes datasets obtained while driving in the city of Karlsruhe, in rural areas and on highways. The sensor setup consists of an inertial navigation system (OXTS RT 3003), a laser scanner (Velodyne HDL-64E), two grayscale cameras (Point Grey Flea 2 FL2-14S3M-C), two colour cameras (Point Grey Flea 2 FL2-14S3C-C) and four varifocal lenses (Edmund Optics NT59-917). The laser scanner has a vertical resolution of 64 and spins at 10 frames per second, capturing approximately 100k points per cycle. This platform was used to develop benchmarks for the tasks of stereo, optical flow, visual odometry and 3D object detection. Preliminary experiments showed that methods that ranked high on previously established benchmarks performed below average when being moved outside the laboratory to the real world. While the evaluation framework allows for some comparison between different algorithms, it is difficult to state which algorithm is better in general since the amount and impact of overfitting and optimisation cannot be determined. The suite does not provide a semantic segmentation benchmark yet, though some semantic labels have been annotated by some users. While the benchmark suite is a commendable first step that attempts to provide an evaluation framework for the computer vision community, it is not sufficient in its current form to obtain safety guarantees for ADS deployment.

Situation Understanding

Object detection, recognition and classification is a prerequisite for scene representation. However, perception goes beyond the mere description of a scene, where each entity is defined by its type and state. A scene refers to the description of a collection of road entities, including the ADS-equipped vehicle. It must be understood by the ADS to

Figure 4.1 – Scenario: Roundabout exit (1/2)

determine how the situation it is confronted to affects and constrains its navigation capabilities. Understanding the intent, relationship, behaviour and relevance of the entities surrounding the ADS is essential. The modelling and evaluation of contextual, spatio-temporal situation awareness has been partially addressed towards that goal, by studying the task of on-road object importance ranking from video [128]. Spatio-temporal object and scene cues were analysed for the task of object importance classification. However, scene description still remains the main focus of current research.

An ontology is an explicit specification of conceptualisation [129]. In essence, an ontology is a formal description that provides the means to perceive and understand the concepts that come into play in a situation. Ontologies have been used to gather information about surrounding entities for the purpose of defining a semantic description of a situation. For example, the framework proposed in [130] provides structure and gives meaning to a scene. However, in its current state, it does not take uncertainties into account and human behaviour that goes against regulations. More importantly, what needs to be perceived and what needs to be understood for an ADS to operate safely has yet to be determined.

Consider the scene depicted in figure 4.1. In this particular situation, it is essential that the ADS detect and properly classify the red vehicle. However, this is insufficient. The ADS must have the ability to realise that the intentions of the red car's driver are uncertain. He/she may exit the roundabout, despite being incorrectly positioned. If the driver initiates a left turn, the event must be perceived so that the uncertainty is lifted. The red car is no longer of relevance for decision making purposes. Before the

Figure 4.2 – Scenario: Roundabout exit (2/2) front view

Figure 4.3 – Scenario: Roundabout exit (2/2) back view

ADS-equipped vehicle makes a right turn, it must be able to determine the presence of pedestrians on or near the pedestrian crossing so that it can adapt its speed accordingly. The ADS must have a basic understanding of how human drivers exit roundabouts. They often overtake near the exit, meaning objects on the left side of the vehicle are actually more important than objects on its right. Indeed, this is the case here (Fig: 4.2), even though the grey car's right flashing light remains active. The ADS must then be able to detect the presence of a stationary truck on the lane it is in and be prepared to overtake said truck. Monitoring other vehicles in the left lane and behind it enables the ADS to determine when to overtake (Fig: 4.3). This shows the importance of identifying intentions and uncertainties as well as defining the capabilities of the perception system with respect to its reasoning processes, so that it has the means to gather the information it seeks.

Three levels of perception can be defined. The most basic level consists of object detection, recognition and classification for scene representation. The second level consists in describing the situation, by giving structure and meaning to the scene. Finally, the third level consists in the determination of the relative relevance of objects in the scene for the purpose of safe navigation. In the current safety framework, the perception system is assumed to be correct if the ADS has the ability to detect, recognise and classify objects *and* events for response preparation. In other words, the system is working as intended as long as any object or event that is incorrectly detected, recognised or classified bears no effect on the safe behaviour of the ADS. In the following, it is assumed that for a given operational design domain or trip, what the ADS must perceive has been determined. Ideally and to efficiently define degraded modes, some metric should be defined for every combination of ODD and sensor fusion scheme. One would then be able to determine where an ADS can operate safely and where it cannot.

4.3 Role of the Safety System in the Perception Task

4.3.1 Safety System

Let us consider the problem of designing an ADS that starts from point A and whose mission consists in reaching a certain location B. The goal is to provide the means for it to take proper observation in the situations that it may encounter along the way. The vehicle may be confronted to a variety of road types, described by a number of lanes, their classification (highway, street), their surface quality (dirt, concrete), their curvature and by the presence of intersections, interchanges and other features, such as railroad crossings, bridges and tunnels. It may cross an urban or suburban area, encounter parks, playgrounds, schools, hospitals or animal preserves. In practice, such information is stored in the navigation map, but it can contain errors, with respect to the road geometry for

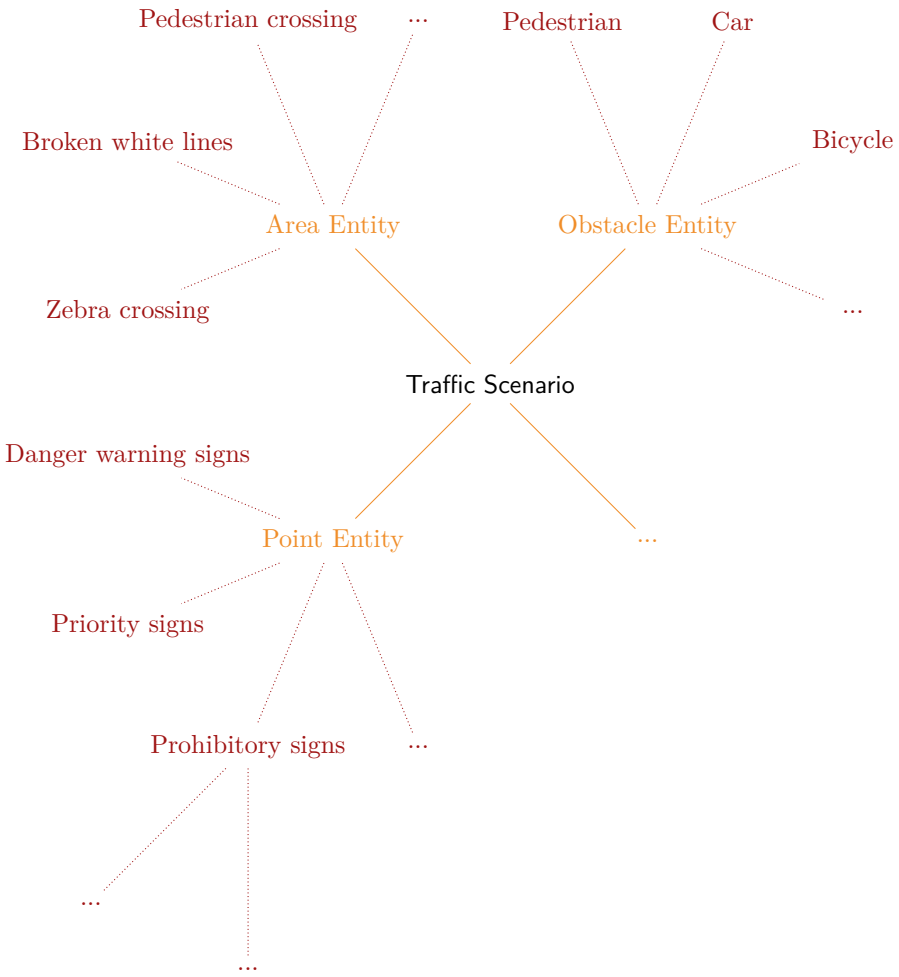


Figure 4.4 – Ontology-based scene representation [5]

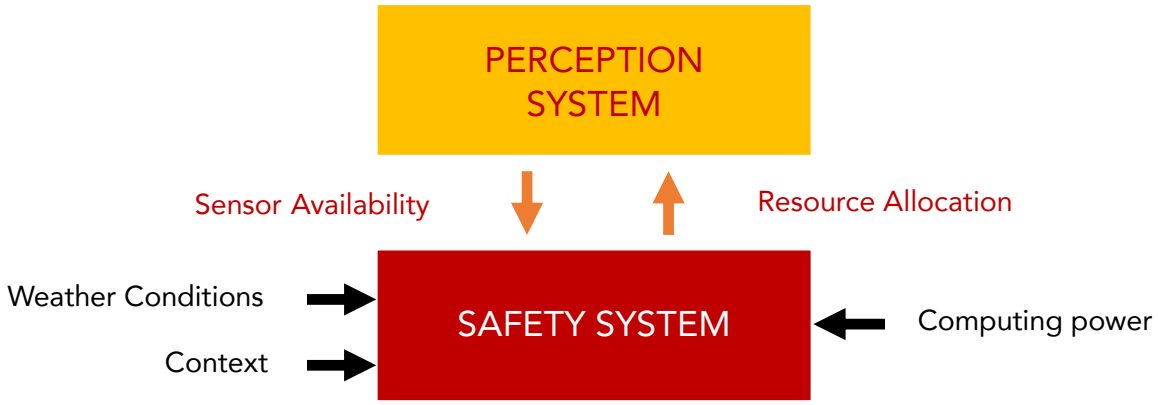


Figure 4.5 – The embedded safety system

example. However, the goal of the safety module is not to confront the information that is extracted from the navigation map with the information that is provided by the perception system during operation.

It is assumed here that the ADS operates in an environment whose characteristics are known. Studying these characteristics enables us to determine what is needed from the perception system. The relevant ones are compiled into a hierarchical tree structure (Fig. 4.4) in such a way that each node is an instance of the class that its parent node represents. A confidence value or vector is affected to each node of the tree to represent the level of performance that the system must achieve. These confidence values are context-dependent. Indeed, while the loss of pedestrian detection may not have great consequences when driving on a highway, this is not the case for an autonomous vehicle driving in an urban area. Confidence values can be expressed in a number of ways (percentages, probabilities, discrete levels) and are an indication of the performance of the system with regards to detection, classification and tracking tasks.

Although the proposed architecture gives way to data integrity and reliability analysis, it is not the main focus of the study. During operation, the safety module judges the perception system and continually evaluates its capabilities by monitoring sensor availability and environmental conditions (Fig. 4.5). The safety module can trigger different configurations using different sensors to recover from a failure. However, if every possible configuration leads to poor performances, the safety module is notified and will either force the vehicle to a stop or ask a human driver to take over.

The role of the safety module also encompasses the allocation of computing power in case of a processing board failure [114]. This means it must force the system into a degraded mode that maintains a sufficient level of performance, with but a few of the

most relevant algorithms running. It is therefore important to know how sensor failures, degrading weather conditions and computing power loss affect the capabilities of the system. Knowing that a sensor has failed is irrelevant if the ADS is clueless about the way it affects its capabilities. This motivates the need to design the perception system of an ADS from a safety perspective.

4.3.2 Multi-sensor data fusion

Fusion is the integration of information from multiple sources in order to produce specific and comprehensive unified data about an entity. The automotive industry is targeted towards a large-scale, cost-sensitive market, which is why the use of low-cost sensors in a multi-level fusion scheme is an interesting prospect for the improvement of accuracy and robustness. In principle, one can expect more specific inferences from the fusion of multi-sensor data over single source data. In practice however, the fusion may actually produce worse results than could be obtained with the most appropriate sensor available.

Observational data may be combined or fused at a variety of levels [131]. Low-level fusion (or *raw data fusion*) involves the combination of raw sensor data and is only possible if they are commensurate. Intermediate-level fusion (or *feature level fusion*) involves the extraction of representative features from sensor data. High-level fusion (or *decision level fusion*) involves the fusion of sensor information, after each sensor has made a preliminary determination of an entity's location, attributes and identity.

Three types of sensor configurations can be distinguished [132]. In a *complementary* configuration, the sensors are combined in order to give a more complete image of the phenomenon under observation. It extends spatial coverage, temporal coverage and improves detection. In a *competitive* configuration, each sensor delivers independent measurements of the same property. Such a configuration is used for the design of fault-tolerant and robust systems. It provides improved system reliability and enhanced spatial resolution. In a *cooperative* configuration, the sensors are combined in order to derive information that would not be available from a single sensor. It leads to increased dimensionality and enhanced spatial coverage.

The imperfection of data is a fundamental problem of data fusion systems. It has led to various mathematical theories. An approach based on credibility was used to model sensor information while an occupancy grid framework was designed to manage different sources of uncertainty [133]. A probabilistic approach capable of dealing with uncertainties when modelling the environment as well as detecting and tracking dynamic objects was proposed [134] for the improvement of safety. The Dempster-Shafer theory,

also known as the theory of belief functions, is a generalisation of the Bayesian theory of subjective probability. It was successfully applied in a vehicle detecting system to increase detection accuracy [135]. Road-matching methods designed to support real-time navigation functions using belief theory were also proposed [136].

Data imperfection is but one of a number of issues that make data fusion a challenging task. Data correlation, data inconsistency and disparateness of data form are challenging problems that must also be investigated when designing Automated Driving Systems for autonomous driving.

4.3.3 Coloured Probabilistic Time Petri Nets

The proposed approach relies on the construction of a high-level probabilistic time Petri net that includes read arcs and inhibitor arcs. The general idea is to build a graph that describes every possible combination of sensors and algorithms into a fusion scheme. Each path in the graph corresponds to a possible scheduling of known algorithms. Confidence values are added along the paths to guide the choice of an appropriate fusion scheme.

Simple and coloured Petri nets have the same computational power. In a simple Petri net, tokens are indistinguishable. Since every logical combination of algorithms is considered for the fusion of information at various levels, it is best to resort to a coloured model in which individual tokens can be identified. Else the formal model ends up being cluttered. Coloured placed are used here to regroup tokens that represent data of the same type. For example, point clouds provided by 2-D lidars are stored in a single place. Tokens in such a place can carry meta-data that enables to distinguish which point cloud was generated using which algorithms. Fortunately, each high-level probabilistic time Petri net can be translated into a simple probabilistic time Petri net that has the same behaviour. This enables the use of model-checking tools such as Romeo [137] for the verification of properties.

Finally, read arcs and inhibitor arcs [138] are added to the aforementioned model. These arcs link a place to a transition. An inhibitor arc imposes the precondition that the transition may only fire when the place is empty. A read arc imposes the precondition that the transition may only fire when the place is not empty. Read arcs and inhibitor arcs are typically used to allow or prevent a transition from firing. The resulting model presents a number of advantages. It natively integrates time, concurrency and probabilities, yet it still benefits from the concision and expressive power of Petri nets. Firing intervals allow for the modelling of computation time, while the randomisation of tokens allows for

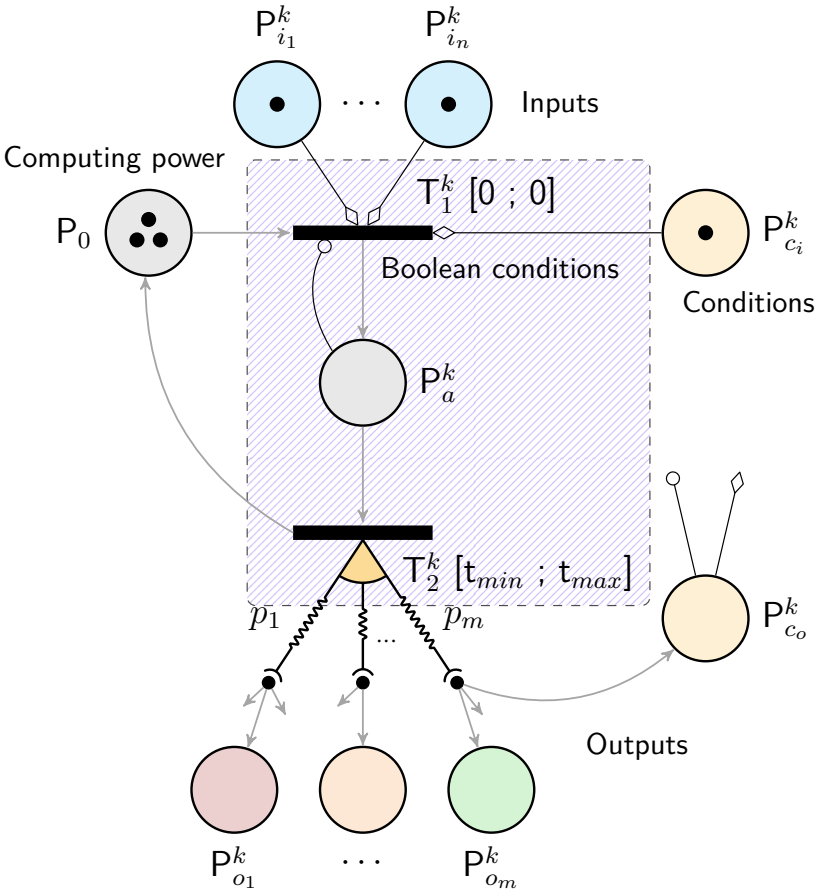


Figure 4.6 – Generic model of an algorithm a_k

the modelling of data reliability. Reachability analysis then provides an exhaustive list of sensor fusion schemes and their characteristics.

4.3.4 The Proposed Model

Formal Model of an Algorithm

A pattern for the modelling of algorithms that can be used in a fusion scheme is defined (Fig. 4.6). It encompasses algorithms for signal processing, pattern processing, feature extraction, sensor data fusion, feature fusion, decision fusion, voting and other algorithms that can be used in order to provide the vehicle with information about the relevant features that have been selected in subsection 4.3.1. Two examples are given. The first one shows the influence of weather conditions on the quality of the data provided by the sensor itself (Fig. 4.7). In the proposed example, it is assumed that some study showed that the performance decreased by a factor of q, while another showed the performance decreased by a factor of q'. If both experiments were conducted in contexts which ressemble the context our system is intended to drive in, both are given a 0.5 probability of being correct. The second one shows a simple algorithm representing proprietary software, such

as the one provided by Mobileye for cameras, by Delphi for radars and by Ibeo for lidars (Fig. 4.8). Depending on the weather conditions, the input data can be enhanced with a value $q < 1$.

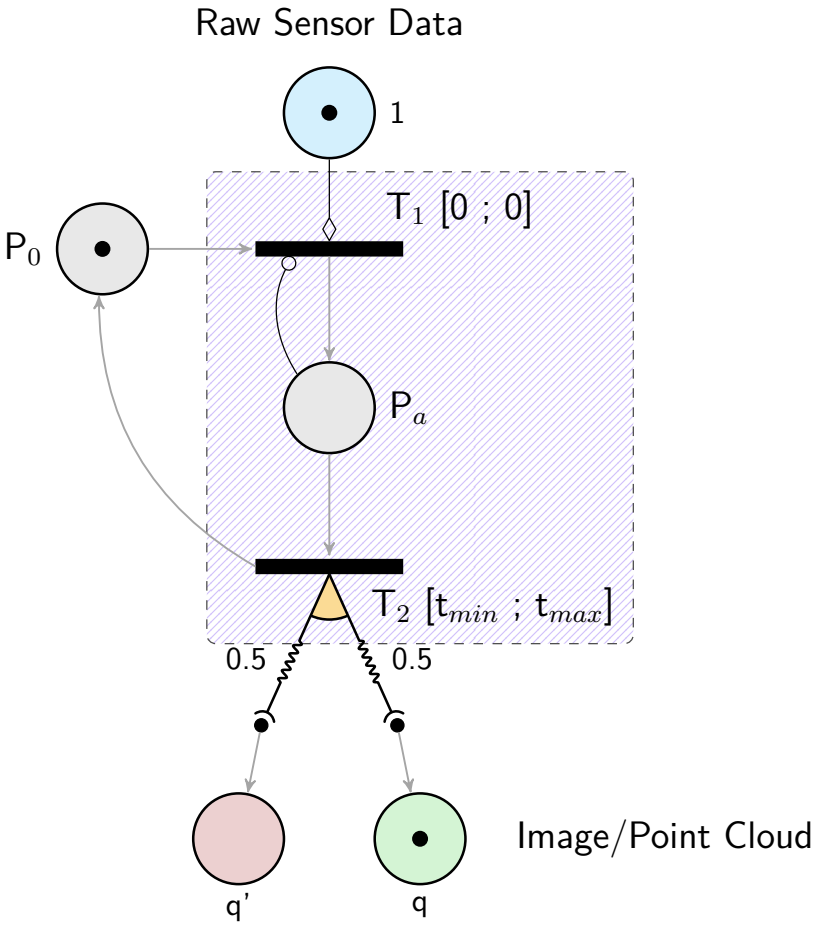


Figure 4.7 – Example showing the influence of weather conditions on the quality of the raw data generated by a sensor

The tokens in P_0 represent the available computing power in the system. When T_1^k is fired, one token is removed from P_0 and placed into P_a^k. This illustrates the fact that the algorithm is running. The inhibitor arc that links P_a^k to T_1^k expresses the fact that no more resources are allocated to the task when it is running. As stated earlier, strong time semantics are used. Resources are allocated whenever an algorithm is ready to run, which is why the timing constraints of T_1^k are $[0, 0]$. The algorithm is expected to run for a certain period of time, represented by the timing constraints of T_2^k.

An algorithm only runs if certain conditions are met. For example, it may not be possible to process data from a sensor that has failed. A feature level fusion algorithm cannot be used when a single piece of data is available. This is expressed by the presence

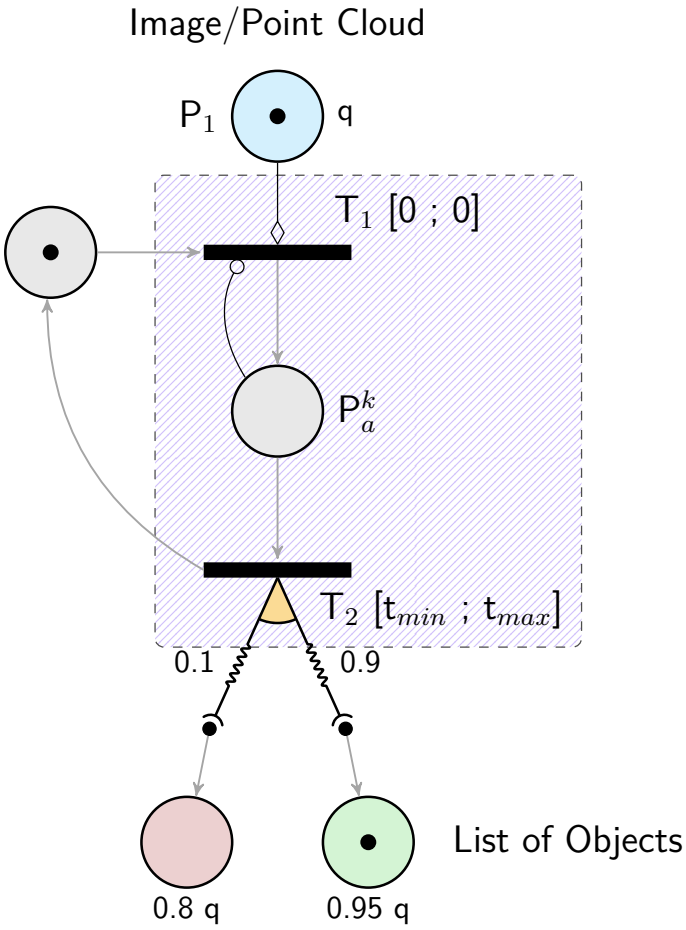


Figure 4.8 – Example showing proprietary software that generates a list of objects from images or point clouds

of read-arcs that link $P_{c_i}^k$ and $P_{i_1}^k, ..., P_{i_n}^k$ to T_1^k. These tokens only act as firing conditions and are not removed when T_1^k is fired. $P_{o_1}^k, ..., P_{o_m}^k$ represent the output of the algorithm. If the inputs and outputs of the algorithm have the same dimension, then it is possible to merge some P_o^k with some P_i^k. A discrete probability distribution of coloured markings is used to represent the fact that the output has a certain probability of being of a certain quality ($\sum_{1 \leq j \leq m} p_j = 1$). It can also be used to express the fact that the algorithm did not generate any output.

Once the algorithm terminates, the resource is removed from P_a^k and made available in P_0 once again. A given algorithm can allow or prevent other algorithms from running. This is represented by output arcs leading to place $P_{c_o}^k$. For example, if the algorithm a_k only runs once, a token is generated in a place that links back to T_1^k with an inhibitor arc. Additional parameters can be used to provide boolean conditions for the firing of transitions. The complete model includes every algorithm that constitutes a candidate in the design of a multi-sensor fusion scheme [139].

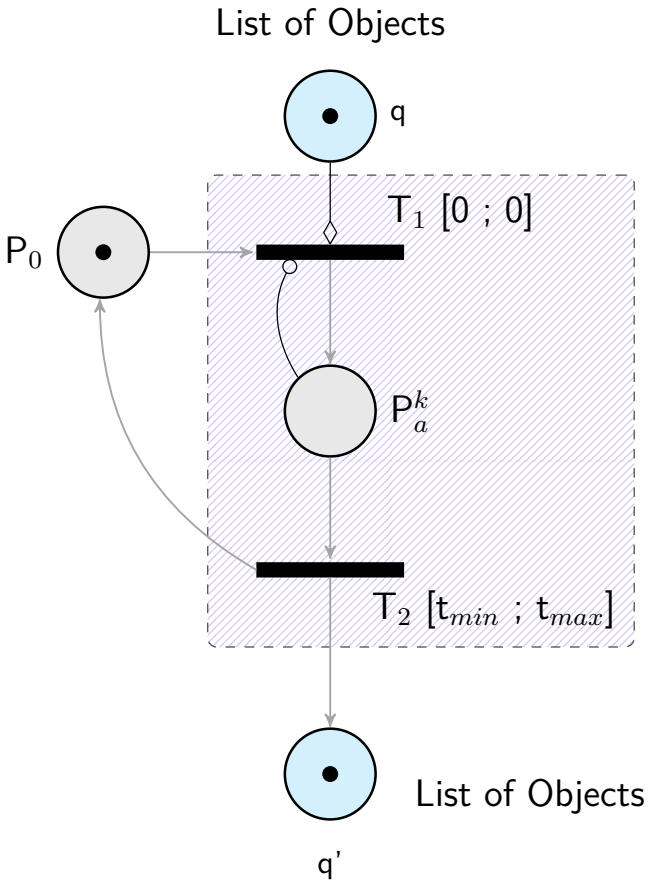


Figure 4.9 – Example showing the fusion of two lists of objects

Confidence Levels

The performance of a data fusion system is dependent on the quality of the input data and the efficiency of the fusion scheme. However, there is no standard and well-established evaluation framework to assess the performance of data fusion algorithms as of yet [131], despite attempts towards defining benchmarking procedures for intelligent robots [140]. The degree of confidence in the data can be described in terms of attributes such as reliability and credibility [141]. The literature work on measures of performance is rather extensive and includes a wide variety of measures. Capturing these dimensions in a comprehensive, unified model can be difficult, as there are trade-offs between these competing aspects. A fair indicator needs to be adapted to the given context or situation.

In the following, the performance of the system is represented by a confidence level, which can take any value between 0 and 1. More precisely, a confidence value is assigned to each node of the tree described in Fig. 4.4. These values represent what the perception system must achieve. Each one of these nodes is represented by a coloured place in the proposed Petri net model. The presence of a token in such a place means that some information about that feature has been generated. The presence of multiple tokens

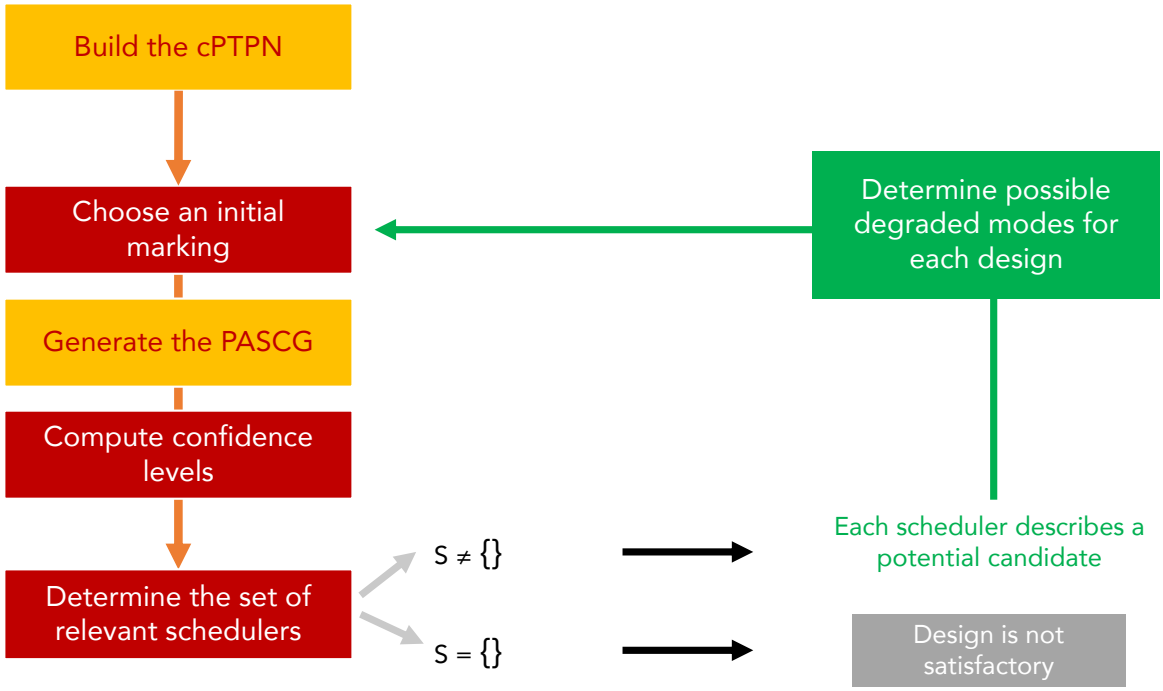


Figure 4.10 – The proposed approach

means that the corresponding information has been generated through different means during the fusion process. The information can be conflicting or not.

In practice, a confidence measure is given to the information carried by each token. The transitions of the net are enhanced with a set of rules that are used to compute the level of confidence of the output tokens based on the confidence given to the input information. The performance of the system with regards to one item is deemed insufficient if no configuration of sensors can achieve it. In other words, no path in the state space of the net leads to a coloured marking for which a token in the place of interest has a high enough confidence level, with high enough probability.

The determination of possible candidates

A state of the net is said to be reachable if there exists a sequence of transition relations that leads from one of its initial states to that particular one. Proving that a given set of states can be reached with a certain probability is at the core of the probabilistic real-time reachability problem for probabilistic time Petri nets as shown in the previous chapter. Quantitative reachability properties provide statements regarding the fact that the probability of achieving a certain desired system behaviour is above a given threshold while the probability of reaching certain unwanted states is sufficiently small.

In practice, the probabilistic atomic state class graph (PASCG) of the net is built to capture every possible behaviour of a perception system in a finite and compact graph, which takes the form of a Markov decision process. In order to determine a perception system that performs adequately under optimum conditions, only the classes of the graph that have a confidence vector that is greater than what must be achieved are considered. The paths leading to these classes describe a set of algorithms that have been used to achieve that level of confidence. The schedulers whose paths lead to these classes with a probability that is higher than some given threshold are the ones used to determine what sensors and what algorithms to use.

In order to evaluate the capabilities of the resulting perception system when failures occur, a token is removed from one of the places that represent the availability of a sensor. The probabilistic atomic state class graph of the resulting marked Petri net is then computed once more. Similarly, to evaluate the capabilities of the chosen perception system when weather conditions deteriorate, the rules associated to the transitions for the computation of trust levels are modified. Thus the safety module can handle a continuous spectrum of conditions. If the resulting fusion scheme is satisfactory, then a suitable degraded mode has been found. The rules that are to be used by the safety module during operation are extracted from this information.

The desired property of the perception system can be formulated in natural language as follows: “Given these possible weather conditions and these embedded sensors, can the ADS perform adequately if less than n of these sensors fail and less than m of these processors fail?" This property can be expressed in a more formal manner: “Is there a path in the probabilistic atomic state class graph of the net that leads to a coloured marking with a probability greater than p_{ad}, such that the resulting confidence value is greater than the minimum confidence value to achieve, when the initial marking is such that up to n tokens are removed from the places that describe sensor availability, m tokens are removed from place P_0, and any combination of rules representing changing weather is chosen for the computation of trust levels?” Such a property can be expressed with modal logic and verified formally. The approach can be summarised by the graph that is represented in Fig. 4.10.

4.3.5 Experimental Results

Problem formulation

The objective of this subsection is to provide an example showing how the proposed approach can be of value for the design of an Automated Driving System. Let us study the

Figure 4.11 – Map and route planning

feasibility of designing an ADS-equipped shuttle running from the Renault bus station in Guyancourt to the railway station near the Palace of Versailles. The web mapping service Google Maps proposes two itineraries (Fig. 4.11) whose characteristics are given in Table 4.1.

The blue itinerary (denoted A) is the most direct route. It is made up of three sections. Buses and bicycles are to be expected, yet trees on both sides of the middle section impair visibility. The grey itinerary (denoted B) provides a less complex environment but it features high-speed roads. Both of these itineraries pose safety challenges.

Part of the study consists in evaluating the possibility of using one of these routes for the current application, prior to building a prototype. The proposed approach aims to provide a preliminary assessment of the level of autonomy [142] that can be expected of an ADS-equipped vehicle on these roads, given current technological capabilities. It also guides the selection of sensors and algorithms and generates reconfiguration rules for the embedded safety module automatically.

Available data

The lack of relevant, publicly available information prevents the tool from being thoroughly fed with context-dependent data. For this reason, the scope of the study is limited

Table 4.1 – Characteristics of the generated itineraries

	Itinerary A (Blue)	Itinerary B (Grey)
Distance	6.9 km	12.3 km
Estimated trip duration	12 min	15 min
Section 0	Private road (50km/h)	
Section 1	Street (50km/h)	Street (70km/h)
Section 2	Dual Carriageway (70km/h)	Highway (110 km/h)
Section 3	Urban area (50 km/h)	Urban area (50 km/h)

to the following ADS capabilities in this example:

- Free space determination, lane tracking, detection of lane markings [143, 144]
- Detection, classification and tracking of obstacles (pedestrians, cars, bicycles and buses), detection and recognition of speed limit signs [37, 145]

These capabilities are determined in a variety of weather conditions: cloudy/wet, sunny, night, snow/rain and fog [18, 146, 147]. The performance measurements taken from the literature are context-dependent and cannot be used as is in our setting. It is assumed that the performance loss due to this discrepancy can be modelled according to some probability distribution, though each system should be tested for each condition to obtain proper results. Since continuous probability distributions are incompatible with probabilistic time Petri nets, discrete uniform distributions can be used instead. Further work is needed to determine what level of performance a *safe* ADS must achieve. Here, for the purpose of illustrating the approach, a fusion scheme is deemed satisfactory if the ADS:

- accomplishes a 95% (resp. 70%) obstacle detection, classification and tracking rate with a probability greater than 0.9 in nominal (resp. degraded) mode
- misclassifies less than 1% (resp. 10%) of detected objects with a probability greater than 0.9 in nominal (resp. degraded) mode

Though the available data provides much needed performance measures, there is actually very little information about the computation time of individual algorithms. As a result, the experiment was conducted with an untimed cPTPN. Comparative studies [145] still provide a fair indication of the relative computation cost of different algorithms.

Table 4.2 – Size of the model

	Places	Transitions
Simple PTPN	158	76
Coloured PTPN	104	76

Affecting confidence levels to sensors and algorithms

Performance measures are often given for the whole perception system of a given ADS. The performance that was achieved with but a subset of sensors and algorithms of the system is rarely provided. To tackle this problem, a weight $\lambda_i \in [0, 1]$ is artificially affected to each experiment i here, according to the similarity between the context the system was tested in and our setting. A weight $w_i^j \in [0, 1]$ is then affected to each sensor or algorithm j as a measure of its contribution to the performance level of the system. The more sensors and algorithms are used and the more the weight is depreciated. It is important to note that this is an artificial way of comparing different studies in order to feed the model and illustrate its capabilities.

Let us consider an ADS capability k (e.g. pedestrian classification). The confidence level affected to a sensor and algorithm combination j for that particular capability is defined as the weighted average $\sum_i (\lambda_i w_i^j).p_i(k)/\sum_i (\lambda_i w_i^j)$, where $p_i(k)$ is the performance of the whole system that was used in experiment i.

The conducted experiments feature sensors commonly used in the context of autonomous driving. The complete model includes several cameras, lidars and radars featured in the literature. The use of many low-cost sensors provides a few benefits over a few expensive high-performance ones, such as system robustness (by avoiding any single point of failure) and a larger field of view. Some solutions work well for tracking vehicles in the context of highway driving but are not general enough for high-curvature roads. These aspects are neglected in this example. The size of the resulting coloured model is displayed in Table 4.2. The simple form of the model was implemented in the Romeo tool [137]. The probabilistic atomic state class graph takes 2.7 seconds to compute on a 2.3 GHz Intel i7 processor with 16 Gb of RAM on average. This takes into account the time taken to read and write files.

Preliminary results

The first step of the study involves following the method described in Fig. 4.10, assuming that every modelled sensor is available, that the weather conditions are optimal and

Table 4.3 – Expected performance of the system

	Correct tracking				Correct classification			
Sensor failures	Ø	C_2	R	L	Ø	C_2	R	L
Cloudy & Wet	0.96	0.96	0.95	0.9	0.86	0.86	0.83	0.95
Sunny	0.97	0.97	0.96	0.86	0.87	0.87	0.85	0.95
Night	0.96	0.96	0.96	0.68	0.87	0.87	0.85	0.62
Snow & Rain	0.97	0.97	0.96	0.72	0.74	0.74	0.7	0.92

that there is no shortage of computing power. The algorithm determined that no combination of modelled sensors and algorithms conforms to the safety requirements for section 2 of itinerary A. This can mean one of two things. Either valuable data pertaining to the performance of existing systems is missing or a key shortcoming of autonomous driving technologies has been identified. The data fed to the tool [18] suggests poor performance from existing systems with regards to the successful classification of cyclists, which is a safety requirement for this road section. This particular statement provides much needed justification when making strategic decisions regarding the deployment of ADS-equipped vehicles. Either itinerary A is deemed impractical for autonomous driving or it can be approved for research purposes, provided a human driver takes over in section 2 of the course whenever necessary. Designing a level 4 ADS-equipped shuttle operating through this road section seems currently unachievable given current data.

Design solution Ignoring current limitations regarding the lack of acceptable bicycle classification, the approach provided several solutions for obstacle detection, classification and tracking in optimal weather conditions. For example, it proposed the systems used in [18] and [37] as possible candidates for the choice of sensors and algorithms. It also proposed new sensor fusion schemes that were not fed to the tool, such as a lidar-based obstacle detection system with a Velodyne HDL-64 (denoted L) combined with a vision-based detection system using a Ladybug camera (denoted C_1). The expected performance for obstacle detection, classification and tracking rate is 96% with a probability of 0.9 in nominal mode. However, it is important to note that this suggestion is based on strong hypotheses.

Neither the lidar-based or vision-based systems can be used on their own. If no single-point of failure is allowed, at least one other sensor must be added to the system. The tool provides a possible solution, which consists in the addition of a radar (denoted R) and of another Ladybug camera (denoted C_2). The resulting system is expected to perform well enough (with a probability of 0.9) with up to one sensor failure in a number of weather conditions (Table 4.3). The confidence values indicate that vision-based systems

can provide a reliable alternative to lidar-based tracking. However, if the lidar system fails, then the ADS must not be used during nighttime. These results suggest a possible attempt at designing a level 3 ADS-equipped shuttle with some level 4 ADS functions (for sections A1, A3 and itinerary B) given the hypothesis that were made. The role of the safety module during operation would be to monitor the perception system, switch to the degraded algorithms in case of a failure and request assistance from a human driver in section A_2. The system would need to be built and tested to demonstrate such performances.

4.4 Conclusion

A formal framework for the design of multi-sensor fusion systems was proposed in this chapter. The approach is based on the construction of a coloured probabilistic time Petri net. This formalism is adapted to the modelling of data reliability and computation time for it natively integrates concurrency, time and probabilities. This work is motivated by the need for rigorous tools from which the capabilities of Automated Driving Systems can be inferred. The effect of failures, weather conditions and computing power loss are studied offline. The proposed method automatically generates the parameters that are needed for real-time online reconfiguration of the perception system.

The advantages of such an approach are two-fold. First, it provides a common language for safety engineers and scientists working in the field of robotic perception. It can be used to identify the limitations of current systems, to keep track of the progress that has been made and to make strategic decisions with respect to the deployment of ADS-equipped vehicles. It also provides valuable input to guide the choice of sensors and algorithms prior to the production of a prototype. Second, it offers a flexible architecture that supports the addition of new data. The more data is gathered and the better the inferences of the tool become. The aggregation of data within the tool is expected to provide a comprehensive understanding of the advantages of one fusion scheme over another. The approach provides the necessary information to elaborate a safety module automatically. When enough information becomes available, it can also be used to optimise the system's operation and to determine the amount of computing power that is necessary.

An example was provided to show how such an approach can be of value for the design of autonomous vehicles. A sensor fusion scheme was proposed and confidence values were given to determine what level of autonomy can be expected. However, a few shortcomings still need to be addressed. The success of the approach rests on the availability of data pertaining to the performance of existing systems. This is necessary to make a proper, complete model with relevant performance and confidence levels. Here, they

were artificially computed to generate additional data, for the sole purpose of illustrating the proposed approach. Proprietary black-box modules impede progress towards a better understanding of ADS capabilities. While the tool provides guidance for the design of a perception system, it does not provide guarantees. The system must be built and tested to properly validate its capabilities. Moreover, information regarding the computation time of algorithms is rarely provided in the literature, though it is an important aspect of safety-critical real-time embedded systems.

The ultimate goal is to develop a framework in which various design solutions can be determined and formally analysed by a computer. While the construction of the probabilistic atomic state class graph is fairly straightforward given the structure of the model, it is possible that the time needed to compute a solution becomes prohibitive, once enough information is gathered. The framework was specifically applied to the perception system as it appears to be one of the key shortcomings to the development of ADS technology. In the following chapters, it is assumed that for a chosen trip, a perception system with the required capabilities has been generated.

Chapter 5

Dynamic Driving Task Fallback for an Automated Driving System whose Ability to Monitor the Driving Environment has been Compromised

5.1 Introduction

The perception module provides an Automated Driving System (ADS) with the ability to monitor the driving environment through the accomplishment of real-time roadway environmental object and event detection, recognition, classification, and response preparation, as needed to operate a vehicle. In the event of a failure that renders it inoperable, a level 3 ADS issues a timely request to intervene and disengages soon after, while a level 4 or 5 ADS must make do with its localisation module and communication capabilities to perform the dynamic driving task (DDT) fallback if a human driver does not respond appropriately to such a request. The ADS is said to experience a DDT performance-relevant system failure and is expected to achieve a minimal risk condition in order to reduce the risk of a crash, as the trip cannot and should not be completed. At the onset of adverse weather conditions whose impact on the capabilities of the system is deemed significant, the ADS is also expected to perform the DDT-fallback to achieve a minimal risk condition.

Conditional driving automation assumes a DDT fallback-ready user is able to operate the vehicle and is receptive to ADS-issued requests to intervene. However, while a driving automation system can significantly reduce a human driver's workload [148], automation can also cause drowsiness and hypo-vigilance [149]. Research was conducted to determine when the user's attention should be directed back to the driving task [150]. Yet the

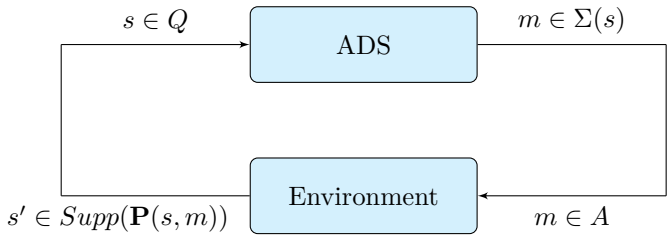


Figure 5.1 – Interaction between the vehicle and its environment

take-over process remains complex and needs to be studied further [151]. Moreover, some drivers are unaware of the capabilities and limitations of driving automation systems. Adaptive cruise control (ACC) is but one system that is being misused in the case of partial driving automation, for users expect it to perform effectively in situations when it actually cannot [28]. The safety benefits of lower levels of driving automation are compromised if a human driver does not supervise the system correctly. Thus, it can be argued that the value of a level 3 ADS is contentious and that the DDT-fallback ultimately constitutes a burden for the driver with whom all responsibility lies.

For the most part, a fallback strategy entails bringing the vehicle to a stop within its current travel path automatically, but it can also pertain to a more extensive manoeuvre designed to remove the vehicle from an active lane of traffic. In some cases however, the vehicle should not be brought to a stop. For example, stopping the vehicle in a tunnel or on a motorway that does not feature a hard shoulder is dangerous and should be avoided if possible. Automatic braking and swerving was investigated for collision avoidance systems with the rationale that evasive manoeuvres can be applied later than braking manoeuvres [152]. However, the approach relies on a functional radar sensor system to determine inter-vehicle distance values. A fallback strategy is proposed in this chapter. It is aimed at level 4 ADS features, designed to operate a vehicle on a road whose characteristics make any attempt at stopping the vehicle hazardous. This implies that the approach also applies to level 5 ADS-operated vehicles and to ADS-dedicated vehicles, for SAE levels indicate minimum rather than maximum ADS capabilities.

5.2 Role of the Safety System in the Decision Making Task

A trip can be interpreted as a game between two players: the ADS and the environment (Fig. 5.1). The ADS chooses a course of action and the environment responds by leading the vehicle to one situation or another. The goal of the ADS is to compute a course of

action by evaluating possible outcomes. Most approaches present safety solely through the prevention and mitigation of collisions. Uncertainty and disturbances have been modelled in [153] in order to study evasive manoeuvres. Other aspects of safety have also been considered and studied. For example, the intention of drivers were compared with what is expected of them in order to assess risk at road intersections [154]. However, most studies ignore the reaction of other entities to the actions made by the system. Yet in order to choose the best course of action, an ADS must have the means to predict how its environment will react. Other approaches include a real-time motion planning architecture for agile driving automation systems [155] and a model based controller synthesis for the safety of driving automation systems in a convoy [156]. Multiple criteria decision making was used to select the most appropriate driving manoeuvres from a set of feasible ones in [157].

An ADS must comply with traffic rules, as well as other commonly accepted rules that go beyond mere regulations, in certain scenarios. Here, safety requirements are assumed to be shared among intelligent agents [158]. An intelligent agent can be defined as an autonomous computational entity that executes certain tasks in such a way that it optimises some given performance measure [159]. A review of intelligent systems software can be found in [160]. A systematic review on artificial agent methodologies that can be applied to ADS control systems is provided in [161]. These supervisors forbid certain manoeuvres if they are deemed hazardous. In particular, one agent is tasked with the choice of a degraded mode and the reconfiguration of the perception system, as discussed in the previous chapter. The confidence levels are a measure of the system's ability to perceive the driving environment. If the system suffers from a severe failure or if environmental conditions become hazardous, the ADS self-confidence can be such that no degraded mode is deemed satisfactory. In that case, the system no longer tries to accomplish its mission and transitions from its prior fail-operational behaviour to a fail-safe mode (Fig. **??**).

5.3 Dynamic Driving Task Fallback

5.3.1 Problem Statement

The objective of this section is to evaluate the behaviour of an Automated Driving System experiencing a severe DDT performance-relevant system failure affecting its perception module, when the system is devoid of a proper fallback strategy. In the previous chapter, the safety system was merely used to allocate and provide resources to the system during operation. It is assumed here that the safety system has evaluated that there

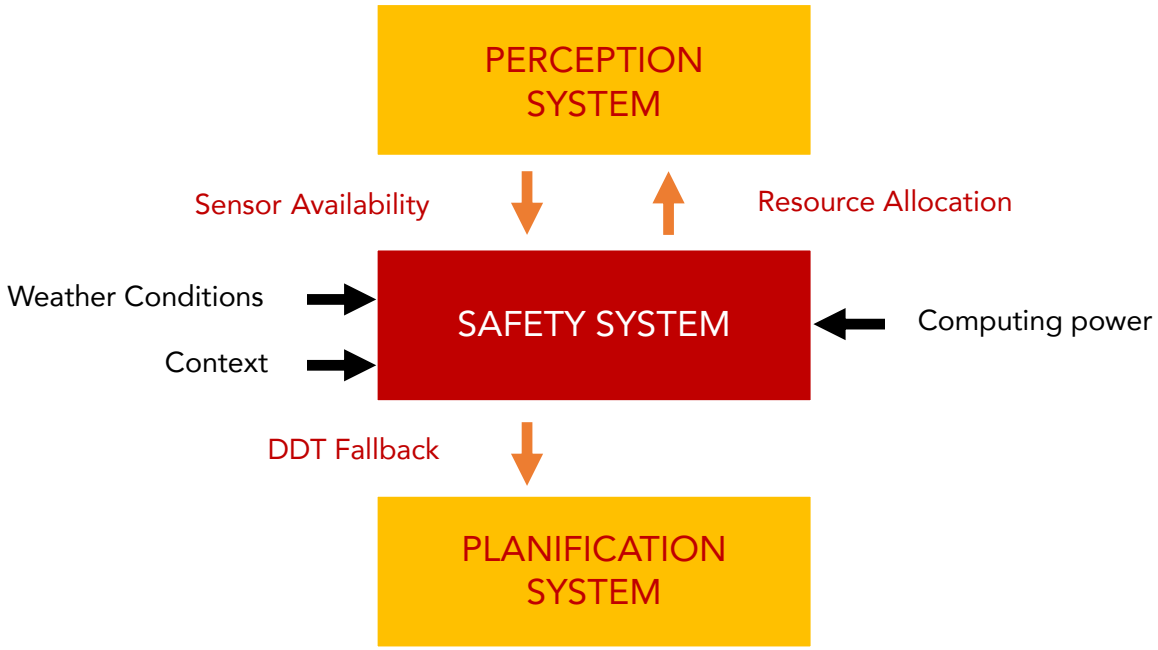


Figure 5.2 – The embedded safety system

are no available degraded modes and that a safe-operational behaviour can no longer be guaranteed. It takes an active role in trying to bring the vehicle to a stop.

Scenario

A dual carriageway with two lanes in each direction was created within the driving simulation software SCANeR studio to model a 2 km long road subsection, known as the Dampierre Road (D91). This busy road connects the city of Versailles to the activity Centre of Guyancourt in France (fig. 5.3). The Dampierre Road crosses the state forest of Versailles. Trees on both sides of the road impair visibility. The region is also subject to heavy rain and fog.

The modelled road consists of three subsections. The speed limit of the middle subsection, which features two sharp turns, is set to 50 $km.h^{-1}$. The speed limit is set to 70 $km.h^{-1}$ for the other two. Lanes in opposite directions have different elevations in the middle subsection. Buses take the Dampierre Road. They sometimes encroach on both lanes when they exceed the speed limit. Bicycles are also to be expected, for the road features a cycle lane in both directions. The road takes under two minutes to travel when traffic is flowing smoothly.

The proposed scenario consists of four vehicles initially positioned in the right lane.

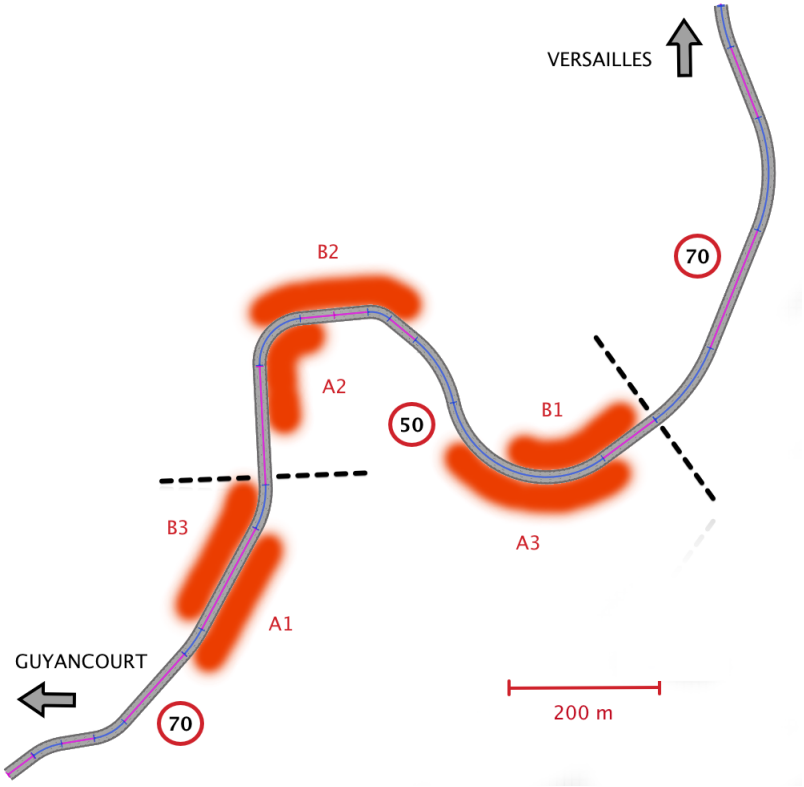


Figure 5.3 – Characteristics of the Dampierre Road (D91)

Table 5.1 – Characteristics of the vehicles

Identifier	A	B	C	D
Vehicle type	Car	ADS-EV	Bus	Car
Initial position	0	1	2	3
Max speed ($km.h^{-1}$)	90	75	90	90
Max acceleration ($m.s^{-2}$)	5	5	1.5	4
Max deceleration ($m.s^{-2}$)	9	9	7.5	9

Their characteristics are detailed in Table 5.1. The ADS-equipped vehicle (ADS-EV) is initially in second position. In the simulation software, a simple bicycle model is used to model vehicle dynamics. All vehicles comply with road traffic regulations.

The Automated Driving System is assumed to lose its ability to monitor the driving environment as a result of the DDT performance-relevant system failure affecting its perception module. This implies that the ADS cannot accomplish real-time roadway environmental object and event detection. Despite being unable to function optimally, the ADS remains capable of following the right lane during failures, thanks to its localisation module.

Both the Guyancourt-Versailles and the Versailles-Guyancourt trips were performed

Table 5.2 – Characteristics of the zones

Trip	Guyancourt - Versailles			Versailles - Guyancourt		
Zone	A_1	A_2	A_3	B_1	B_2	B_3
Distance (m)	210	180	235	190	190	185
Duration (s)	11.5	12.4	17.7	12.9	14.8	9.5

as part of the simulation scenario. The perception module of the Automated Driving System was subject to the DDT performance-relevant system failure on three occasions during each trip. These failures lasted throughout the areas that are represented in figure 5.3. Road geometry, visibility and speed limits were taken into consideration to define these zones. Their characteristics are given in Table 5.2. The duration of each failure corresponds to the time it takes to travel the zone at the maximum speed limit. It includes the reaction time of the ADS.

Separate experiments were conducted for each trip:

- For the Guyancourt-Versailles trip (scenario S_1), the speed profile of vehicle A was set according to the graph displayed in figure 5.4. Real trips were performed on the Dampierre Road in order to define a typical speed profile for vehicle A, when traffic is light.
- For the Versailles-Guyancourt trip (scenario S_2), vehicle A was positioned in front of the ADS-equipped vehicle with a given speed v_i (ranging from 0 to 50 km.h^{-1}), at a distance of 20 m, whenever the ADS recovered from a failure.

In both cases, data pertaining to the dynamics of all four vehicles was recorded and the behaviour of the ADS was evaluated.

Analysis

Figures 5.5 and 5.6 show the behaviour of the ADS-operated vehicle in scenario S_1 and scenario S_2 respectively. The speed profile of a vehicle that drives at the maximum speed limit is represented by a curve labelled v_{Ref} for each trip. Scenario S_1 illustrates the fact that an ADS that lacks a proper fallback strategy runs the risk of crashing into the vehicle in front of it *during the failure*. After 81 seconds of simulation time, a collision between vehicle B, travelling at 50 km.h^{-1} and vehicle A, travelling at 25 km.h^{-1} occurs. When the perception module of the ADS-operated vehicle fails, the Automated Driving System acts as if there are no obstacles in the scene, which explains why the speed profile of vehicle B matches the curve labelled v_{Ref} in these areas, regardless of the last known

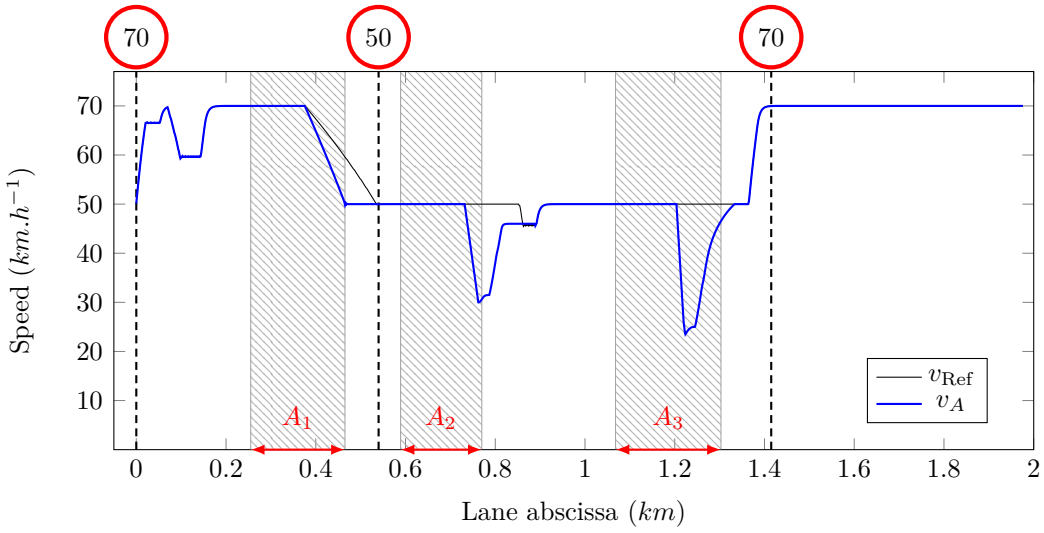


Figure 5.4 – Speed profile of vehicle A in scenario S_1

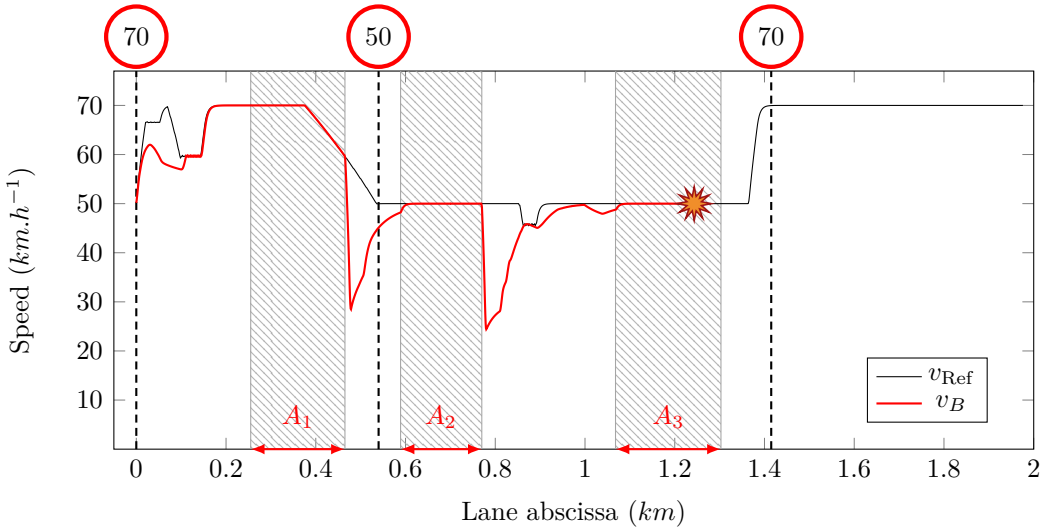


Figure 5.5 – Speed profile of vehicle B in scenario S_1

position of vehicle A. The longer the duration of the failure, the greater the risk of a collision between vehicles A and B. This also shows how an ADS would react if obstacles were missing in the world model as a result of sensor failures or occlusions, and if no action is taken to replace the missing information.

Scenario S_2 illustrates the fact that the behaviour of the ADS *at recovery* can lead to hazardous situations. For $v_A = 20$ km.h^{-1}, the sudden deceleration following the recovery of failure B_1 led vehicle D to overtake vehicles C and B while vehicle A was accelerating once again. The subsequent emergency break helped to avoid the collision between vehicle B and vehicle D but the sudden speed variation created yet another hazardous situation with vehicle C, which was forced to brake as well. Figure 5.7 provides the inter-vehicle distances separating the ADS-equipped vehicle from the other vehicles in this scenario.

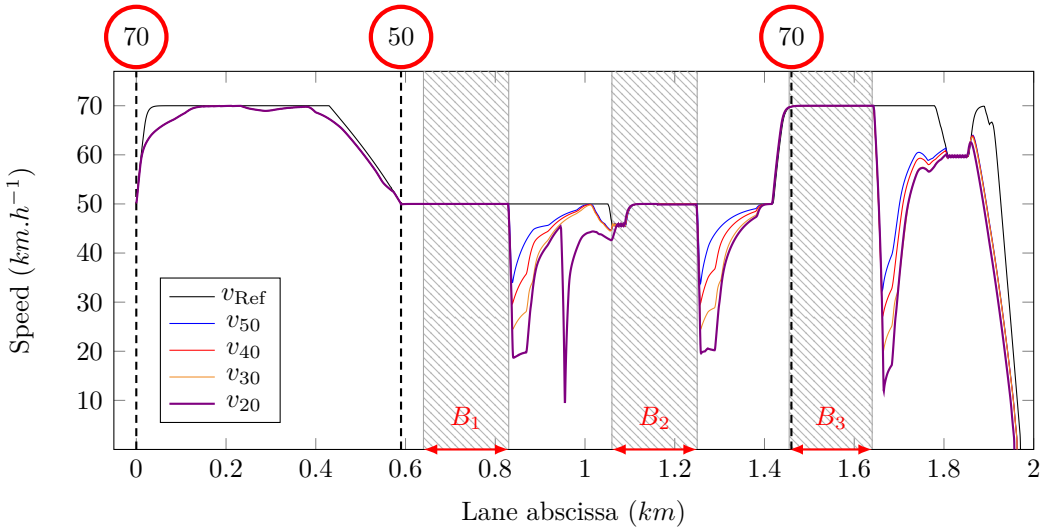


Figure 5.6 – Speed profile of vehicle B in scenario S_2

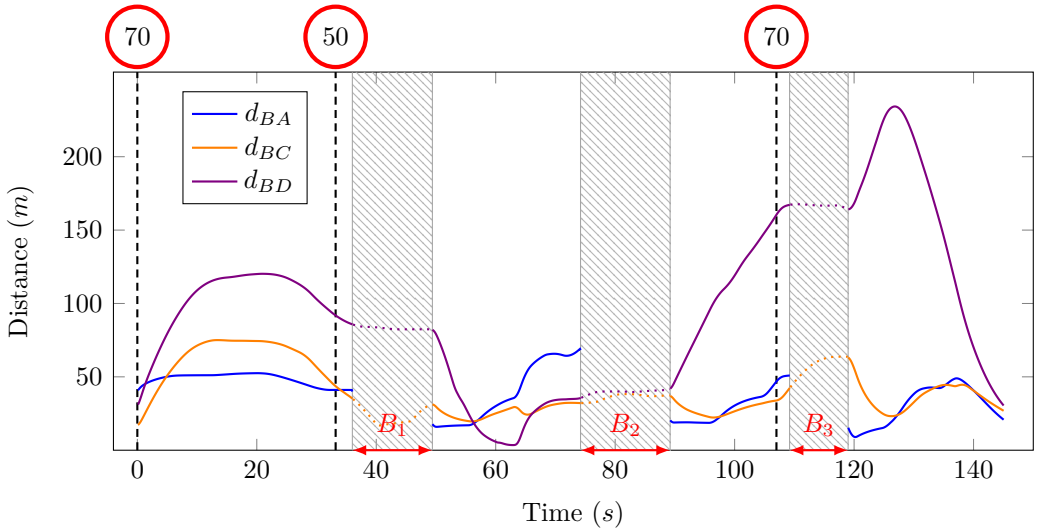


Figure 5.7 – Inter-vehicle distances when v_A is set to 20 km.h^{-1} in scenario S_2

The absence of a DDT fallback can lead to highly hazardous situations, including rear-end collisions at full speed and multi-vehicle crashes with severe consequences. It is necessary for the system to adapt gracefully to the failure in order to avoid hazardous recoveries. In order to design a level 4 ADS-equipped vehicle that can operate on the Dampierre Road, a proper fallback strategy must be defined in the event of such a failure.

5.3.2 The Proposed Fallback Strategy

A fallback strategy usually entails automatically bringing the vehicle to a stop within its current travel path, but can also pertain to a more extensive manoeuvre designed to remove the vehicle from an active lane of traffic. However, the road under consideration does not feature a hard shoulder. An emergency stop would be highly hazardous given the road geometry, the road logic, the variety of travellers and limited visibility. In the following, any attempt to stop on the road is considered hazardous and is therefore prohibited.

Assumptions

This subsection highlights the assumptions that are made in the proposed approach. It is assumed that the Automated Driving System's ability to monitor the driving environment has been compromised. Such a condition is attributable to failures and to hazardous weather conditions. Different types of sensors typically react to the same environmental condition in diverse ways. While common practices addressing hardware failures tend to over-provision resources, an Automated Driving System targeted towards large-scale cost-sensitive markets is more likely to feature a variety of sensor modalities.

A fallback strategy is needed in case such attempts at improving system dependability prove insufficient. To this end, a framework that gives the automated system the means to acknowledge its inability to monitor the driving environment is required. The purpose of this study is not to evaluate the validity of the request to intervene by the automated system. Therefore, it neither issues false alarms nor categorises critical situations as uncritical. In the following, the perception module is considered unreliable and cannot be used to elaborate a fallback strategy.

Nowadays, most Automated Driving Systems rely on a DDT fallback-ready user to achieve a minimal risk condition. However, a human driver may not respond to a request to intervene appropriately and in good time, which is why a fallback strategy is to be defined regardless of the presence of a human driver and his ability to intervene. The dynamic driving task fallback must also be defined for ADS-dedicated vehicles (designed to be operated exclusively by a level 4 or level 5 ADS for all trips).

The sharing of information between vehicles or with the infrastructure allows the ADS to perceive its environment beyond the limits of the on-board sensors of its perception module. This can be useful if the position and speed of other vehicles can be obtained with a good precision and in good time. However, communication poses security threats, confidentiality issues and is subject to delays in response time. In the following, communication is not used to elaborate a fallback strategy.

The localisation module is a driving automation system whose purpose is to determine the position of the vehicle with respect to an embedded map. The map contains relevant information about the road typography (number of lanes, intersubsections, etc). In the following, both the map and the localisation module of the ADS are assumed to be perfect. The ADS-operated vehicle can therefore position itself correctly in the right lane, regardless of any failure affecting the perception module.

In practice however, sensors can be shared between the perception module and the localisation module. For example, cameras are used by the perception module to detect pedestrians and by the localisation module to detect lane markings. As a result, both failures and hazardous weather conditions affecting the cameras can prevent the localisation module from functioning optimally.

The conflicting hypotheses of an unreliable perception module and of perfect vehicle localisation can be rephrased as follows. The combination of hazardous weather conditions and failures affecting the Automated Driving System are severe enough that the perception module cannot monitor the driving environment reliably. Yet they are not severe to the point the ADS can no longer reach the end of the road under consideration by following the right lane. It is assumed here that the Automated Driving System takes advantage of the multi-modality that its localisation module provides.

Vehicles drive on the right side of the road. When the failure occurs, the ADS-operated vehicle is either positioned on the right lane or is assumed to have the ability to immediately change lane to position itself there. The visibility range of other vehicles is determined by the curvature and elevation of the road. The chosen road adhesion parameters match those of a dry road in the simulation software. Finally, we assume that the ADS is able to make use of the full range of vehicle dynamics.

Active safety systems

An *active safety system* senses and monitors conditions inside and outside of the vehicle in order to identify potential dangers to the vehicle, occupants, and/or other road users [142]. In the following, the existence of an active safety system that can detect the

loss of the perception capabilities of the vehicle is assumed. This loss can be the result of a failure of the perception module as a whole, as well as the failure of key sensors or the presence of major inconsistencies in the generated world model.

The rationale behind the proposed fallback strategy rests on the following remarks:

- Since the vehicle is prohibited from stopping on the considered road, as it is highly hazardous, the goal of the ADS is to reach the end of the road while mitigating collisions or avoiding them.
- Since the ADS has lost its ability to monitor the driving environment, the ADS-operated vehicle is to be constrained to the right lane to avoid complex and hazardous situations. Moreover, the localisation module is assumed to be perfect, so the ADS can easily follow the lane. The vehicle must be slow enough so that rear-end collisions with vehicles in front are avoided or mitigated. It must also be fast enough so that other vehicles are not startled by its behaviour when it appears in their field of view.

Speed profile meeting a TTC-criterion

The speed profile that is to be adopted by the ADS-operated vehicle during failures depends on a time to collision criterion. The general idea is to drive as slowly as possible while giving following vehicles the time to slow down or attempt evasive manoeuvres such as overtaking. Figure 5.8 shows, for each position x_1 of the ADS-equipped vehicle, the maximum distance $D^v_{max}(x_1)$ at which it can be seen, by vehicles driving behind it. In order to mitigate the severity of collisions with leading and following vehicles, the minimum and maximum speed limits V_{min} and V_{max} are set to 20 km.h^{-1} and 35 km.h^{-1} respectively for the Automated Driving System, on all three subsections of the road. V_{max} influences the severity and likelihood of a rear-end collision with a vehicle in front of the ADS-operated vehicle, while V_{min} mostly influences the time it takes to reach the end of the road. These parameters must be chosen according to the type of road and the maximum speed limit.

Time to collision (TTC) is defined as the time required for two vehicles to collide if they continue at their present speed and on the same path [162]:

$$TTC = \frac{\Delta L}{v_2 - v_1} = \frac{x_2 - x_1}{v_2 - v_1},$$

where ΔL is the distance between both vehicles, x_1 and x_2 are the lane abscissas of the vehicles, v_1 is the speed of the leading vehicle and v_2 is the speed of the following vehicle. The value of TTC is usually used in the decision making process that leads to

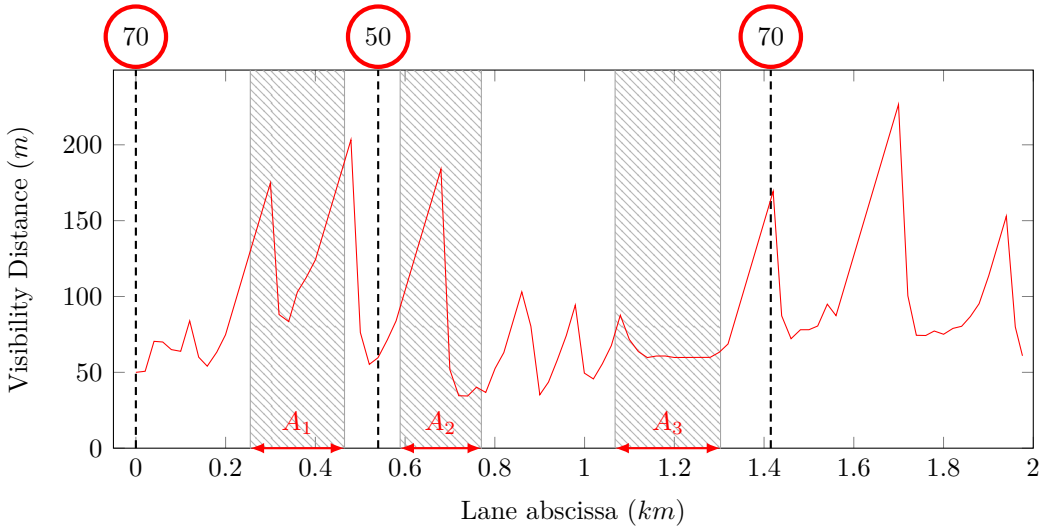


Figure 5.8 – Visibility distances on the Guyancourt-Versailles trip

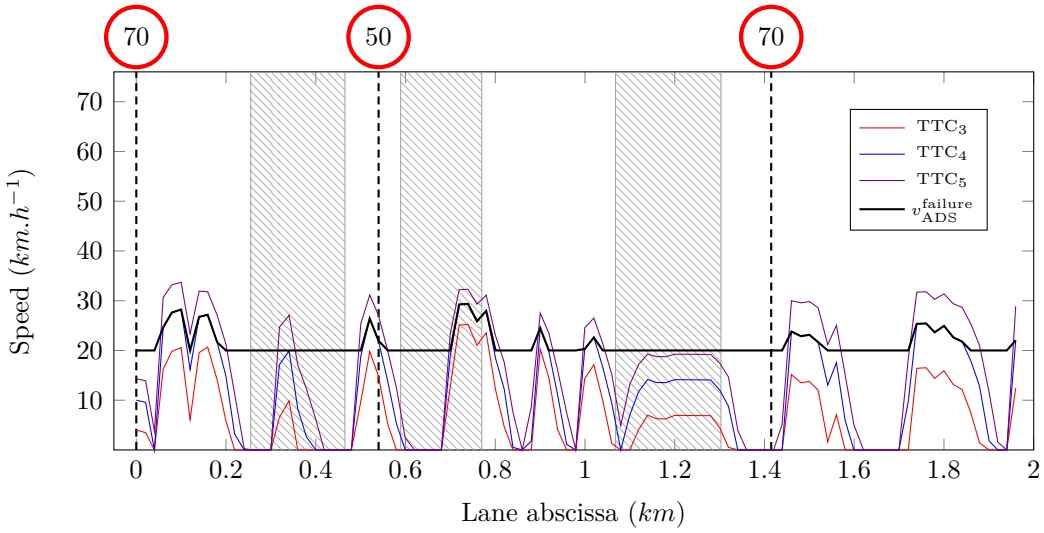


Figure 5.9 – Speed profile that is to be adopted by the ADS-operated vehicle in the event of a failure

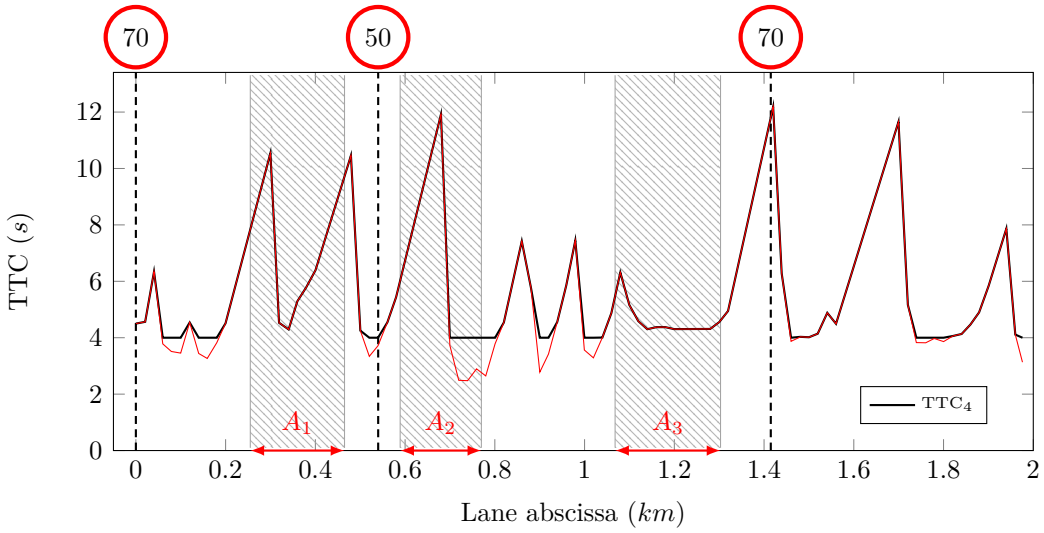


Figure 5.10 – Evolution of the Time To Collision parameter when adopting the proposed speed profile

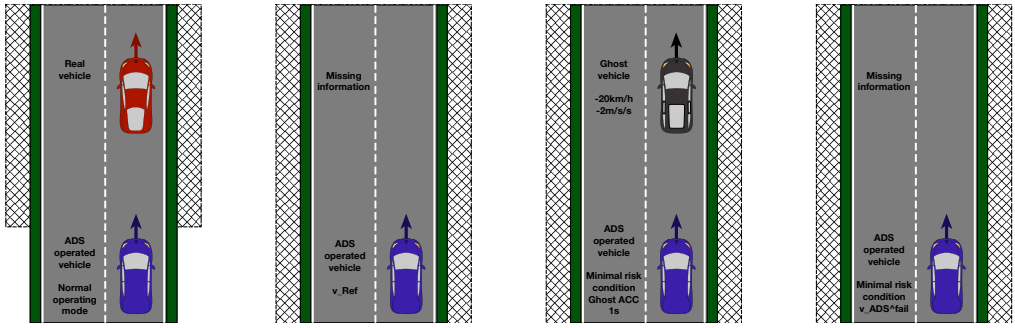
the activation of driver assistance systems for driving automation systems. Studies have shown that it is a fitting measure of the severity of conflicts on the road. This metric was successfully used for collision risk prediction using a grid-based approach [163] and for situation identification [164] in the context of autonomous driving. The lowest value of TTC during a situation involving two vehicles on a collision course is denoted TTC_{min} and indicates the severity of the encounter. The lower the value of TTC_{min}, the higher the risk of a collision. Situations are considered critical when the minimum TTC is less than 1.5 s. TTC_{min} appears to be independent of approach speed [162].

The value of TTC at the onset of braking, denoted TTC_{brake}, represents the available manoeuvring space when the evasive action starts. In the following, the reaction time of other drivers is set to 1.5 s and TTC_{brake} is set to 2.5 s, leading to a TTC-criterion of 4 s. Assuming $\Delta L = D^v_{max}$ and $v_2 = v_{\text{Ref}}$, the minimal risk condition is defined for each lane abscissa x as the speed that guarantees the following TTC-criterion:

$$v^{failure}_{ADS}(x) = min[max(v^{TTC_4}_{ADS}(x), V_{min}), V_{max}],$$

where $v^{TTC_4}_{ADS}(x) = v_{\text{Ref}}(x - D^v_{max}(x)) - \frac{D^v_{max}(x)}{TTC_4}$.

The minimum speed that the ADS-operated vehicle needs to achieve a TTC-criterion of 3, 4 and 5 s, is represented in figure 5.9 along with the speed profile that verifies the minimal risk condition. As expected, a higher TTC-criterion implies a higher minimum speed must be achieved. As shown in figure 5.10, the 4 second TTC criterion can be met without the vehicle exceeding 29.4 km.h^{-1}. The chosen speed profile $v^{failure}_{ADS}$ ensures:



(a) Situation prior to the failure (b) No fallback strategy (c) Ghost method (d) TTC-criterion

Figure 5.11 – Behaviour of the Automated Driving System based on its interpretation of the driving environment

- The ADS-operated vehicle leaves the road in under 6 minutes.
- The mitigation of rear-end collisions with vehicles in front of the ADS-operated vehicle, as its speed never exceeds 30 $km.h^{-1}$.
- The avoidance of severe rear-end collisions with vehicles following the ADS-operated vehicle, as the TTC-criterion of 4 seconds is met for the whole duration of the trip.

5.3.3 Experimental Results

Simulations show that both the collision in scenario S_1 and the hazardous situation in scenario S_2 are avoided when the ADS performs the DDT-fallback by enforcing the proposed speed profile during failures. However, the ADS-operated vehicle shows sudden speed variations that lead to close calls such as the one presented in scenario S_2 in other situations, since the ADS toggles the DDT-fallback as soon as the failure is detected. This occurs almost exclusively when simulating intermittent faults.

The transition stage, during which the fallback strategy is activated, must be given proper consideration in order to tackle the remaining number of close calls. This is especially challenging as the nature and duration of the failure usually cannot be determined. If the fault is intermittent, applying the proposed speed profile as soon as a failure is detected leads to a succession of sudden speed variations as the vehicle switches between its normal operating mode and the speed profile underlying the fallback strategy.

The last generated world model prior to the failure of the perception module can be used to finely-shade the transition stage. Missing obstacles can be replaced in the world model with ghosts of the vehicles that were last perceived (fig. 5.11). Such an approach is only valid for a short period of time, as each vehicle on the road continually adapts

Table 5.3 – Experimental results

Fallback strategy	None	TTC_4	TTC_4 + Ghost
Close calls	10	12	3
Collisions (front)	10	4	4
Collisions (back)	10	0	0

its behaviour to new events that cannot be predicted. In the following, it is assumed valid for a time duration of 1 s. When the failure occurs, the ADS adjusts the speed of the vehicle to maintain a safe distance from the ghost of the vehicle that is directly ahead. The initial speed of the ghost vehicle is reduced by 20 km.h^{-1} compared to the real vehicle and its acceleration is reduced by 2 m.s^{-2}. This forces the ADS to initiate a preventive braking procedure and enables the vehicle to smoothly transition to the TTC compliant speed profile. After 1 s, the ADS terminates the ghost ACC-procedure and enforces the proposed speed profile $v_{ADS}^{failure}$. This approach provides a smoother speed profile and removes the sudden speed variations for failures that last less than 3 seconds entirely, as well as intermittent faults of that length.

The setup proposed for scenario S_1 was used to elaborate 10 situations featuring a close call and 20 situations featuring rear-end collisions for the Guyancourt-Versailles trip, by modifying the speed profile and behaviour of vehicles A and C. These situations were submitted to an ADS that achieves the minimal risk condition as defined previously, with and without the transition stage adaptation. Results are given in table 6.2. As expected, the 4 s TTC criterion enables the avoidance of collisions with vehicles behind the ADS-operated vehicle, while the addition of the ghost method mitigates the number of close calls. Depending on the traffic conditions and the areas where the failures occur, some collisions with vehicles in front of the ADS-operated vehicle cannot be avoided. For each of the four recorded collisions, the speed difference between the ADS-operated vehicle and vehicle A was below 15 km.h^{-1}.

5.4 Conclusion

A fallback strategy was proposed for an Automated Driving System whose ability to monitor the driving environment had been compromised. It was assumed that bringing the vehicle to a stop was hazardous. The approach favours rear-end collisions at low speed with vehicles in front of the ADS-operated vehicle against risking more serious hazards by bringing the vehicle to a stop. It enables the vehicle to reach a road where it can safely stop, as the ADS keeps the vehicle on the move. However, this can be a matter of

discussion as it raises the question of responsibility and ethical decision making [165].

The approach relies on the detection of hazardous weather conditions and failures by an active safety system. It can be adapted to a variety of operational design domains, as long as the backwards visibility mappings are included in the embedded map. In practice, sensors are shared between the perception module and the localisation module. This can prevent the approach from being used as is by some Automated Driving Systems, unless the vehicle can be positioned precisely in its lane. Taking advantage of multi-modality is recommended as it can enable an ADS to reliably determine its position for a longer period of time.

The ADS was supposed devoid of communication capabilities as they are prone to security threats and confidentiality issues. However, communicating with the infrastructure and with other vehicles can be useful to obtain information about their position, their speed and the state of traffic. The hypothesis of a truly autonomous vehicle was favoured here. To further reduce the risk of collisions with vehicles in front of the ADS-equipped vehicle, special hazard flashers can be activated in case of such failures. Vehicles in front of the ADS-equipped vehicle are expected to yield when detecting such signals, while human drivers tailgating the ADS-operated vehicle are expected to slow down.

In the provided example, it was assumed that the ADS-operated vehicle could position itself in the right lane. The focus was set on defining a speed profile that allowed the ADS to mitigate risk, provided it could stay in its lane. Future work includes studying a wider variety of scenarios in which steering and braking are used for safety purposes as well. Giving an ADS the ability to accurately predict the consequences of its actions is necessary in order for it to make informed decisions. The proposed ghost method can be interpreted as predictive motion planning, for the ADS makes use of the last known data it deems accurate about its environment to transition more smoothly to its target speed profile. The predicted characteristics of the ghost objects could be refined in terms of intent. Finally, the approach relies on the localisation system to properly follow its lane. Its success is therefore limited to the performance of that system.

Chapter 6

Adaptability of Automated Driving Systems to the Hazardous Nature of Road Networks

6.1 Introduction

In chapter 4, the safety system was given the means to reconfigure the perception system and force it into a suitable degraded mode whenever possible. In chapter 5, it was given the means to preempt the decision making task of the navigation system in the event the ADS became unable to monitor the driving environment by enforcing a given fallback strategy. The proposed architecture can be further enhanced by allowing the system to learn from its experience and to adapt. In this chapter, the safety system is given more influence over the decision making task. It is given the means to alter the behaviour of the ADS in certain areas of the road network by studying previous trips. The study is restricted to daily commuting.

6.2 Role of the Safety System in the Learning Task

Designing an ADS that operates safely in a wide variety of scenarios is an ambitious task. Due to system design limitations, an ADS will ultimately be confronted to hazardous situations that it was not designed to deal with. These situations may be the result of the inherent hazardous nature of the road network. However, it can be difficult to determine which roads are hazardous and which are not. Some roads provide poor visibility while

others have a poorly designed layout. One typically acquires this kind of information with experience, by driving on said roads. Therefore, it seems an ADS must be given the means to detect and adapt to such environments. This can be done by giving an agent the ability to learn from the mistakes the ADS made during past trips. While a human driver may initially behave in a dangerous or inconsistent manner on a road whose logic or geometry is unusual, when traffic signs are occluded or when they seemingly convey contradictory information, human drivers who travel in such areas on a regular basis are able to adapt their behaviour in a way that reduces risk. Automated Driving Systems are currently unable to do so, for they lack the necessary learning capabilities. The aim of this study is to give an ADS the means to identify such areas in its operational design domain during periodically recurring travels so that it can adapt its behaviour, as a human driver would. The emphasis lies on safety, which refers to losses due to unintentional actions caused by benevolent actors [166].

Hazard perception refers to one's situation awareness capabilities for the detection of dangerous situations in the traffic environment [167]. The recognition of dangerous situations has been studied within a cooperative group of autonomous vehicles in [168], but the approach is dependant on a highly available communication system. Other inventions rely on communication to broadcast information about the fact that accidents have occurred. For example, [169] proposes a dangerous road subsection analysis method to warn vehicles entering a dangerous road section, while [170] presents a module that can retrieve information about the hazardous nature of a road by communicating with a server. It is assumed here that the vehicle does not rely on its communication capabilities to identify hazardous areas, for they are prone to security and safety issues.

A method and system for screening potential traffic safety hazard road sections was investigated in [171]. Identifying hazardous locations can be carried via statistical models or by incorporating the spatial configuration by means of a local indicator of spatial association [172]. To the best of our knowledge, the identification of hazardous road locations [173] is contingent on the availability of traffic accident databases, such as the ones provided in [174] and [175]. However, there is no evidence that points towards those databases covering the entirety of the hazardous areas in the road network. Moreover, new hazardous areas are bound to emerge as the road network grows and changes. Even roadworks are prone to making roads hazardous for extended periods of time. After their completion, old lane markings sometimes remain visible and can be seen alongside new ones. This can puzzle human drivers as they try to figure out what the logic of the road

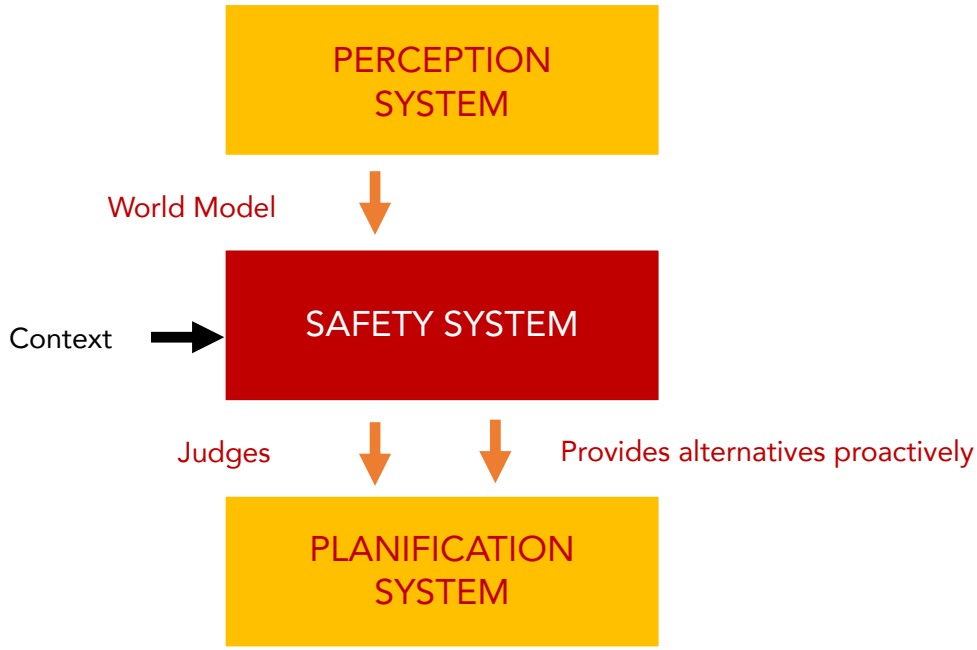


Figure 6.1 – The embedded safety system

is. In the approach that is proposed here, the Automated Driving System determines the location of the hazardous areas in its operational design domain by itself. The ADS also rectifies its behaviour and develops strategies to avoid finding itself in hazardous situations during later trips.

6.3 The Proposed Architecture

6.3.1 System Architecture

The safety system now features two additional components (fig. 6.1):

- A *recording component*, whose role is to observe and record any violation of safety constraints.
- A *rectification component*, whose role is to analyse previously encountered hazardous situations in order to adapt the behaviour of the ADS during later trips.

In order to avoid issues related to user confidentiality, the monitoring component of the safety module does not operate at all times. Moreover, storing unnecessary information

might clutter memory. The rectification module operates whenever the ADS drives on a road where hazardous situations have been previously recorded.

Human machine interface

The user defines a list of destinations called points of interest through a human machine interface, by supplying the system with their address or GPS coordinates and giving them a label, such as 'work' or 'home'. It is hypothesised that trips between two points of interest are instances of periodically recurring travels, which is why the recording component is activated during such trips. In practice, the user chooses a destination (by selecting the corresponding point of interest in the list). It is for the system to determine that the vehicle is initially located in the vicinity of a point of interest. For example, the vehicle might not be parked at the same location every evening, yet the ADS must be able to determine that the ongoing trip falls within the framework of daily commuting to work and that the monitoring component must be activated. The monitoring component can also be activated during non-recurring travels. For example, if the ADS travels on a road subsection that is usually taken during daily commuting or if the ADS travels in the vicinity of a point of interest, then the monitoring component is activated for good measure.

6.3.2 Operational behaviour of the Automated Driving System

Recording component

An intelligent agent is an autonomous entity that directs its activity towards achieving goals. It observes through sensors and acts upon an environment using actuators. Intelligent agents may also learn or use knowledge to achieve their goals. The decision making process can be interpreted as a periodic deliberation task among a set of agents, each of which supervises a set of conditions. Safety agents ensure that the ADS complies with the highway code (safety distances, maximum speed) and acts upon any indication that it is not behaving in a safe or correct manner (abrupt and repeated speed variations). Non-critical agents are responsible for the optimisation of travel time and user comfort. The decision is said to be consonant if the chosen trajectory is expected to verify all the safety constraints.

Besides their role in the decision making process, the agents are tasked with the evaluation of the Automated Driving System's performance during operation (fig. 6.1).

Figure 6.2 – Distribution of hazardous situations in the navigation map, after multiple trips

A broken safety condition expresses a discrepancy between the confidence that the safety module placed in a chosen course of action and the subsequent hazardous situation, which occurred during its execution. In the following, it is assumed that the behaviour of the ADS is correct. In other words, the decision making process always leads to a consonant decision. An unfulfilled safety condition can thus be the result of the actions of a human driver or ADS that behaves in an unexpected or dangerous way or the indication of an inherent hazardous driving environment. It can also be linked to the transition phase of an event in traffic. For example, an overtaking manoeuvre that involves a vehicle moving in between the ADS-equipped vehicle in question and the vehicle in front of it might lead to a temporary violation of the safety distances.

Each safety agent is responsible for the verification of one safety condition. If the condition does not hold, a beacon is added to the map. The constraint violation c extends over a continuous area called a λ_c zone. A λ_c *zone* is characterised by the position of its endpoints, the time it took to cross, a list of relevant values that characterise the fact that the constraint did not hold along the zone, and other variables that describe the severity or magnitude of the constraint violation. For example, failing to maintain safety distances with the vehicle in front of the ADS-equipped vehicle will result in the creation of a list of relevant values containing the inter-vehicle distance measured at each verification period. Other useful information, such as time to collision values, are recorded as well. The agent also records relevant contextual data (time of the incident, season, weather and distribution of obstacles in the scene during and prior to the problem).

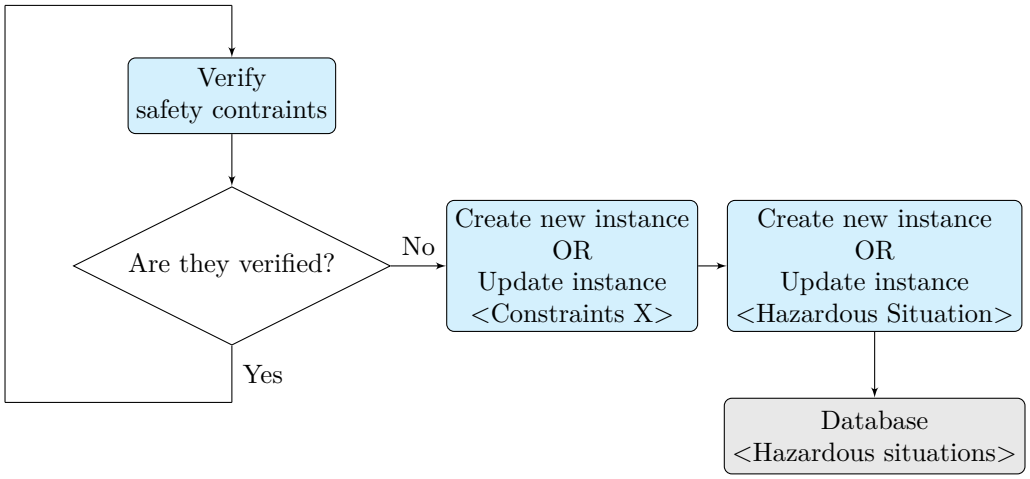


Figure 6.3 – Recording component

A hazardous situation involves at least one broken constraint and can be associated with multiple unfulfilled safety conditions. Therefore, it can either coincide with a single zone or it can encompass multiple, overlapping ones. Figure 6.2 shows a possible distribution of hazardous situations in the map after multiple trips. A recorded hazardous situation is represented by a dot whose radius corresponds to the severity of the encounter.

The perception module is tasked with tracking each obstacle in the scene and records the tactical decisions (intentions and manoeuvres) made by the dynamic obstacles during the last ten seconds. This gives the ADS situation awareness capabilities and helps it understand how the behaviour of each obstacle in the scene has led to the hazardous situation in the first place. Each hazardous situation has a type, which defines the problem. For example, 'safety distances are not verified (broken constraint) with a vehicle coming from road A after it positions itself in front of the ADS-equipped vehicle on road B (situation)' is a type of hazardous situation.

If a safety constraint is broken for the first time during the trip, then an instance of the class of objects 'Zone' and an instance of the class of objects 'Hazardous Situation' are created (fig. 6.3). The distribution of obstacles in the scene and their past actions are recorded, alongside relevant information about the time of day, weather and traffic density. During each subsequent constraint verification period, the agents check if the safety conditions hold. The order of magnitude of the verification period is that of the refresh rate of the world model. Each λ_c zone that is associated with a broken safety constraint that persists is updated. If a broken safety condition is verified once more,

then the zone is saved by the recording component. If another condition is broken, then another zone is created.

A time duration is defined to discriminate hazardous situations from isolated problems, such as transient events in traffic. As soon as the safety constraints are verified once again, the hazardous situation instance is updated and archived. If it only lasted for a short period of time however, the instance is deleted as it means the ADS managed to recover from the problem rapidly. If a constraint is broken at a future date, then a new instance of the class of objects 'Hazardous Situation' is created with the correponding zone or zones. After several trips, a set of lists of hazardous situations is gathered. Each list is associated to one road subsection and sorted according to the direction of traffic.

Rectification component

The embedded navigation map is divided into areas that define a cover of the road network (as defined in mathematics). The cover is initially defined according to the road logic (intersection, roundabout, continuous road subsection). It is then further refined to encompass specific features of the environment (schools, hospitals, speed limit). The size of the refined areas depends on the range up to which such information remains relevant. As a result, a continuous road that joins two intersections can be broken down into multiple, potentially overlapping areas. Figure 6.2 shows an example of such a cover. Areas are not to be mistaken with the previously defined λ_c zones. The cover of the road network in areas is defined a priori when the ADS is built, while hazardous situations and zones are recorded by the ADS during operation.

An inherent hazardous driving environment is expected to generate more hazardous situations on a given road subsection or area after multiple trips than one time encounters with random drivers that behave in a dangerous manner. Therefore, the ADS attempts to rectify its behaviour in a given area of a trip if a minimum number of hazardous situations have been recorded in that area during previous trips and during a certain time period. This number (for example 'five hazardous situations in a given area in one year' or 'two hazardous situations in a given area after one hundred trips') can be adjusted depending on the type of area and the type of hazard.

During operation, once the path between two points of interest has been computed, the safety module checks if it crosses an area that exhibits a high number of hazardous

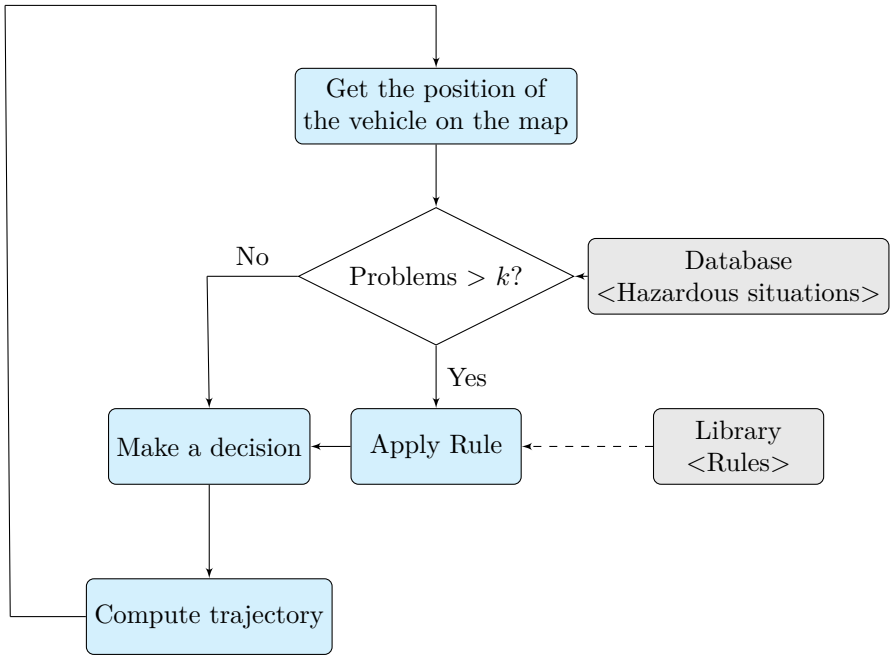


Figure 6.4 – Rectification component

situations. If that is the case, it will invoke a rule from a library of fallback strategies (slow down, set local speed limit to a lower value, change lane, yield, forbid overtaking, etc), with respect to the type of hazardous situations that were encountered (fig. 6.4). For example, if the ADS-equipped vehicle fails to maintain safety distances for a prolonged period of time despite aggressive braking, multiple times, near an area where two roads meet and if the situation awareness function of the recording component determines that the hazardous situation is related to the late detection of a vehicle that positions itself in front of the ADS-equipped vehicle, then the ADS will slow down before it reaches the intersection or change lane ahead of it. If multiple hazardous situations are linked to an overtaking attempt in a given area, then overtaking can be prohibited on that road subsection.

There may be several applicable rules for a given hazardous situation. These rules are sorted a priori in a certain order, according to certain features of the environment. For example, changing lane may have a higher priority near a highway ramp, while slowing down may have a higher priority near a school or hospital. A given rule can be applied in multiple ways. For example, braking can be done according to various deceleration profiles. The ADS chooses a given profile according to the number of recorded hazardous situations, their severity (given by the list of relevant values) and their distribution in the

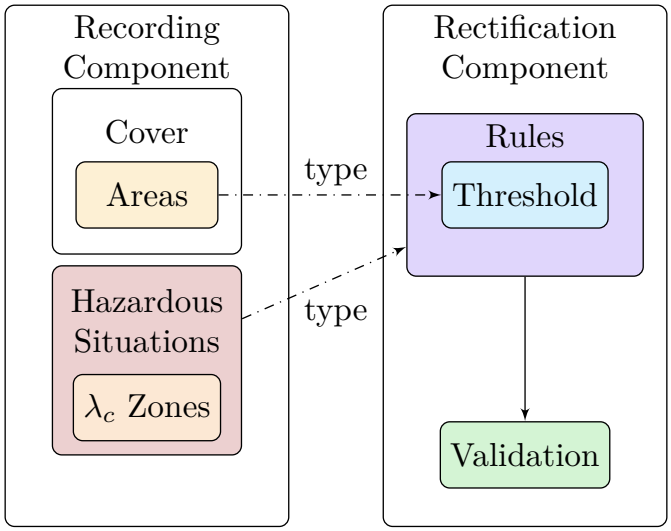


Figure 6.5 – Inner workings of the safety module

area.

A neighbourhood of the ADS-equipped vehicle is defined in order to measure the density of previously recorded hazardous situations in the vicinity of the vehicle. The ADS will attempt to rectify its behaviour if a minimum number of hazardous situations have been recorded in its neighbourhood. During operation, the size of the neighbourhood can vary according to the speed of the vehicle, the type of road, the distance to the next intersection, the curvature of the road, etc. The neighbourhood of the vehicle can contain several recorded hazardous situations that are of different types. A rule that applies to all of the previously encountered problems may be chosen over a rule that only applies to the most critical one.

Performance criterion The architecture of the safety module is presented in figure 6.5. The performance criterion reflects the difference between the number of hazardous situations that occur before and after the new strategy has been adopted. If the number of hazardous situations decreases significantly, then the adopted strategy is deemed satisfactory. Otherwise, the strategy is either reinforced or replaced by a lower priority rule.

The hazardous nature of an area of the road network is contingent on a number of factors. For example, the presence of patches of black ice depends on the weather and the season. A road can become more or less dangerous depending on the traffic density and related driving behaviours. Occluded traffic signs can make some roads even more

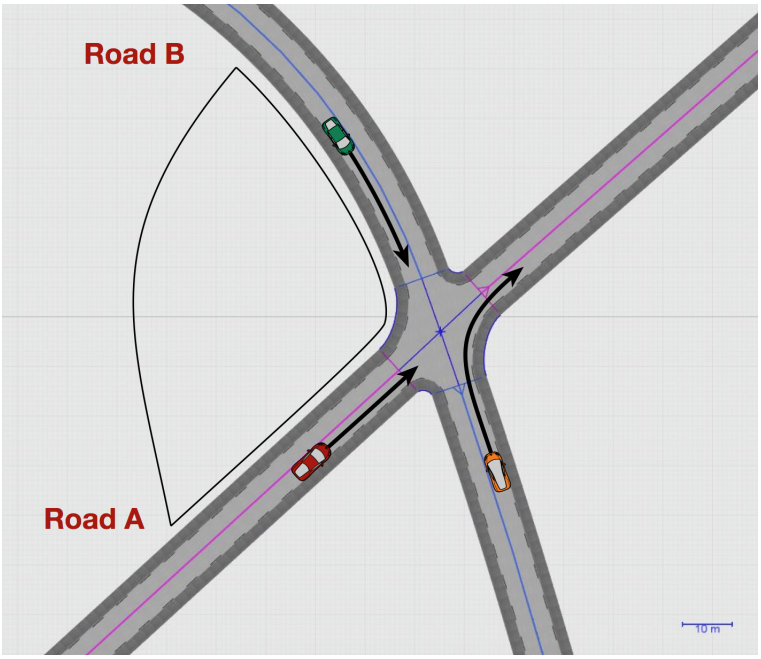


Figure 6.6 – Case study A: Poor visibility at an intersection

hazardous during nighttime. Recording relevant contextual data allows the vehicle to add context to the occurrence of hazardous situations. Such data can also be used to check if the application of the selected rule is meaningful.

6.3.3 Case studies

This subsection illustrates the practicality of the proposed approach by explicitly defining the thought process of the ADS in two simulated examples. For the sake of clarity, it is assumed *for each scenario* that the hazardous situations were encountered in one area.

Hazards due to poor visibility near an intersection

Let us consider an Automated Driving System that travels on a priority road (denoted road A), which only has two lanes (one in both directions). The ADS is regularly confronted to hazardous situations near an intersection, which is represented in figure 6.6. Safety distances are not verified with vehicles coming from road B as they refuse to yield and either cross the intersection at high speed or position themselves in front of the ADS-equipped vehicle on road A. This corresponds to two different types of hazardous situations, as defined earlier. It is assumed here that these situations are due to poor visibility and a missing or occluded road sign on road B. The ADS has no knowledge of

this fact and cannot possibly determine the origin of the problem. However, the ADS is able to infer the late detection of other vehicles at this intersection thanks to its situation awareness capabilities.

The corresponding scenario was created within the driving simulation software SCANeR studio as it could not be tested with a real autonomous vehicle due to the risk of damage to the prototype. Since the traffic module of the simulation software prevents such a scenario from occurring under normal conditions, the speed profiles of the vehicles driving on road B is defined artificially in order to bypass it. The speed limit is set to 50 km.h^{-1} on both roads. On average, when the other vehicles are detected, their speed is close to 45 km.h^{-1}, while the speed of the ADS-equipped vehicle nears 40 km.h^{-1}. Four vehicles were added to the simulation environment. They were given the behaviour of an ADS devoid of a safety module on road A and the behaviour of a driver that is unaware of the presence of the intersection on road B. Simulations ran for 12 hours. There were 18 collisions and 71 close calls. Most collisions are the consequence of both vehicles initiating an emergency break and colliding in the central part of the intersection.

The addition of the safety module in the proposed framework implies that a cover of the road network has been defined. In particular, the intersection is included in one of the areas of the cover. The ADS-equipped vehicle activates the rectification component after being confronted to several hazardous situations. The only applicable rule here is 'slowing down'. Time to collision (TTC) is defined as the time required for two vehicles to collide if they continue at their present speed and on the same path [162]:

$$TTC = \frac{\Delta L}{V_2 - V_1} = \frac{x_2 - x_1}{V_2 - V_1}, \tag{6.1}$$

where ΔL is the distance between both vehicles, x_1 and x_2 are the lane abscissas of the vehicles, V_1 is the speed of the leading vehicle and V_2 is the speed of the following vehicle. Studies have shown that it is a fitting measure of the severity of conflicts on the road. TTC cannot be applied directly as the paths of the ADS-equipped vehicle and of the other vehicles are different. This notion can still be adapted if a vehicle positions itself in front of the ADS-equipped vehicle on road A. Assuming the speed of the vehicle is 45 km.h^{-1} when it becomes visible to the ADS-equipped vehicle, it can be determined with equation (6.1) that setting the speed of the ADS-equipped vehicle to 32 km.h^{-1} before that point guarantees a TTC criterion of 4 s can be achieved. If its speed is slightly lower than 45 km.h^{-1} however or if it keeps traveling on road B, then the proposed strategy is not

Table 6.1 – Experimental results

Strategy	None	32 km.h^{-1}	23 km.h^{-1}
Close calls (nb.h^{-1})	5.9	3.1	0.12
Collisions (nb.h^{-1})	1.5	0.8	0.01

satisfactory. Results are given per hour of simulation (Table 6.1).

If the system is devoid of any means of elaborating a new strategy to take into account the other type of hazardous situation, the roads that lead to this intersection are prohibited by the safety module. Here, the chosen rule can be strengthened further to reduce risk. Table 6.1 shows that limiting the speed of the ADS-equipped vehicle to 23 km.h^{-1} prior to reaching the intersection allows the ADS to reduce risk significantly and can be used as a fallback strategy by the ADS.

Hazards near slip roads on a highway

Let us consider an Automated Driving System that travels on a highway. The ADS is confronted to hazardous situations near a slip road (fig. 6.7) as human drivers attempt to exit the highway, despite initially traveling on the left lane. Safety distances are not verified with respect to these vehicles, which results in aggressive braking. Such a scenario can be the result of poor signalling on the highway subsection that leads to this particular exit or to poor driving habits from users in this particular geographic area. Once again, the ADS is unable to infer the cause of the situation it is confronted to.

The corresponding scenario was created within SCANeR studio. The speed limit is set to 130 km.h^{-1} on the highway. Vehicles were forced to attempt a lane change in order to reach the exit. The ADS drives near the slip road every other minute. Results are reported in Table 6.2.

In the proposed framework, the slip road is included in one area, as part of the cover of the road network. After being confronted to the same hazardous situation multiple times, the rectification component of the safety module is activated by the ADS. Two rules can be applied here: ‘changing lane’ prior to the slip road and ‘slowing down’. Given the context (driving on a highway), changing lane is given higher priority than slowing down (as would otherwise be the case near a school or hospital). When the vehicle should

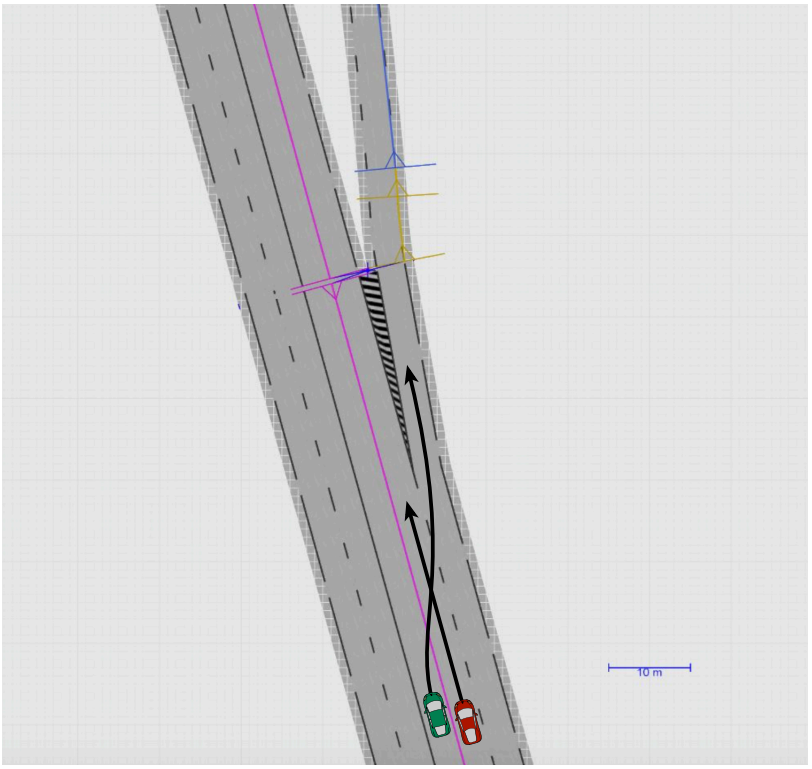


Figure 6.7 – Case study B: Hazardous situations near an exit ramp

Table 6.2 – Experimental results

Strategy	None	Lane change
Close calls (nb.h^{-1})	2.2	0.01
Collisions (nb.h^{-1})	1.6	0

attempt to change lanes is the only remaining decision to make. By changing lanes 600 m before reaching the slip road, the Automated Driving System has more than 15 seconds to position itself. This strategy greatly reduces risk as shown in Table 6.2. The only recorded close call is the result of the ADS being unable to change lane as too many vehicles attempted to exit the highway. Once the slip road has been left behind, the ADS can position itself in the right lane once again.

6.4 Conclusion

A system that allows an Automated Driving System (ADS) to adapt its behaviour to inherent hazardous areas of the road network during periodically recurring travels was presented. The system attempts to mimic the chain of thought of a human driver who learns from his past experience to reduce risk. While the system was presented from a safety perspective, the method can also be adapted to improve the driving style of an Automated Driving System. Communication was proscribed as the safety module should not rely solely on the availability of such means. However, the data gathered by all the ADS-equipped vehicles could be retrieved to elaborate a hazardous behaviour map that is specific to Automated Driving Systems. The strategies of an ADS could then be shared with other vehicles that travel on inherent hazardous areas of the driving environment for the first time.

The goal of the proposed approach was to provide an ADS with situation awareness and learning capabilities to increase safety. This approach can be used to enable an ADS to adapt to situations that were not considered during its design. It could also be used to determine how an ADS should behave in certain locations. The information that is gathered by the ADS is stored in the embedded map and update as it finds itself in trouble. The proposed approach does not take uncertainties into account. The perception system was assumed to be perfect. Future work includes taking into account uncertainties in the world model (such as the velocity and the position of objets in the environment). Other layers can be added to the situation awareness capabilities of the vehicle, such as the estimation of the intention of other drivers. Finally, further study of the classification of hazards in a given area with respect to relevant contextual hazardous information (such as the weather, time of day, season and traffic) is to be conducted as it should give the ADS more flexibility when adopting one fallback strategy over another.

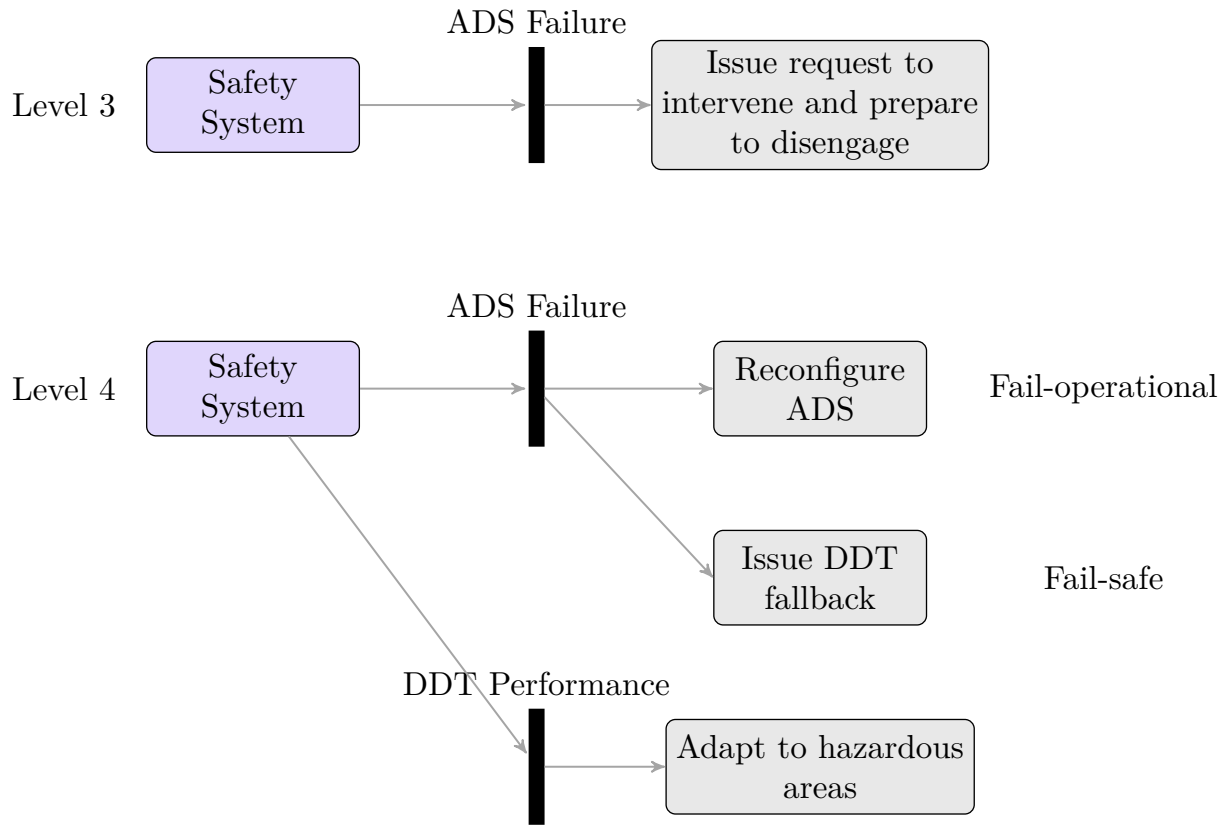
Chapter 7

Conclusion

7.1 Synthesis

Designing an Automated Driving System is a challenging task that requires a substantial jump in technological capabilities. Human drivers cannot be used reliably as a safety net to offset the lack of maturity of Automated Driving Systems, as automation changes their active involvement into a monitoring role and creates new challenges, such as complacency, automation dependency, lack of understanding and misuse. The definition of an architecture that could help establish the inherent safety of driving automation systems was proposed. The aim was to develop tools and methods that could provide insight into the capabilities of Automated Driving Systems as a preliminary attempt to estimate their dependability. In particular, the thesis addressed the problem of designing a new ADS primary subsystem, whose role it is to monitor the state of the ADS, supervise its actions and respond as needed to guarantee the safety of its occupants and of others (Fig. 7.1).

Probabilistic time Petri nets (PTPN) were presented as a new model for the design of concurrent stochastic real-time systems. This model extends time Petri nets by allowing the output of tokens to be randomised. It was used as a basis for the design of multi-sensor data fusion systems that support adaptive graceful degradation through the smart use of sensor modalities. The safety system was then given the means to reconfigure the perception system in real-time and force it into a suitable degraded mode whenever possible and necessary. A DDT fallback strategy was devised in the event the safety system determined the ADS's ability to monitor the driving environment had been compromised. This allowed the safety system to preempt the decision making task. More specifically, the proposed fallback strategy was aimed at level 4 ADS features designed to operate a vehicle



.

Figure 7.1 – From a level 3 ADS-operated vehicle to a level 4 ADS-dedicated vehicle

on a road whose characteristics made any attempt at stopping hazardous. The proposed architecture was further enhanced by allowing the system to learn from its experience and to adapt to its environment in a controlled manner. A method that aimed to give the system the means to identify hazardous areas in the road network was also outlined.

7.2 Publications

Part of this thesis was published, to some extent, in the following papers and led to a patent.

- Y. Emzivat, B. Delahaye, D. Lime and O. H. Roux, Probabilistic Time Petri Nets, In Fabrice Kordon and Daniel Moldt, editors, *The 37th International Conference on Application and Theory of Petri Nets and Concurrency (Petri Nets 2016*, volume 9698 of *Lecture Notes in Computer Science*, pages 261–280, Torún, Poland, June 2016, Springer
- Y. Emzivat, J. Ibanez-Guzman, H. Illy, P. Martinet and O. H. Roux, A Formal Approach for the Design of a Dependable Perception System for Autonomous Vehicles. In *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC 2018)*, Maui, USA, November 2018
- Y. Emzivat, J. Ibanez-Guzman, P. Martinet and O. H. Roux, Dynamic Driving Task Fallback for an Automated Driving System whose Ability to Monitor the Driving Environment has been Compromised. In *IEEE Intelligent Vehicles Symposium (IV 2017)*, Redondo Beach, USA. June 2017
- Y. Emzivat, J. Ibanez-Guzman, P. Martinet and O. H. Roux, Adaptability of Automated Driving Systems to the Hazardous Nature of the Road Network. In *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC 2017)*, Yokohama, Japan, October 2017
- Y. Emzivat, J. Ibanez-Guzman, P. Martinet and O. H. Roux. Procédé de détection d'endroits traîtres du réseau routier pour adapter le comportement du véhicule autonome lors de trajets répétés - Patent n°17-59003 (pending)

7.3 Perspectives

Despite the contributions made in this thesis, the development of a proper safety system for Automated Driving Systems requires more work.

Human Driver Interferences The focus of this work was set on providing the ADS with capabilities that enables it to increase safety. The user's reaction to the system's behaviour has been neglected. In practice, a lack of trust in the system's capabilities can lead the user to interfere with the system while it is enforcing a fallback strategy. It is important that the system communicate its intentions and its internal state to those inside and outside the vehicle both. While the user should ultimately remain in full control of the vehicle, allowing the system to prevent the user from interfering at specific times should be considered carefully and studied.

ADS Behaviour This work focused on providing the means to deal with failures and hazardous weather conditions through online reconfiguration and the definition of dynamic driving task fallback strategies. Yet clarifying the behaviour of an ADS in the absence of failures is also important. How the system should behave in a given environment and in a given situation should be defined better in general. More work is required to determine how the system should deal with conflicting information from its perception components.

Perception system A formal framework for the design of the perception system was provided. More specifically, the focus was set on the real-time aspects. More refined metrics are needed to properly compare and evaluate the systems that the algorithm proposes. It is important to understand where a system works and where it does not. With more data, machine learning could be used to determine a model of the parameters that make certain systems work where they do. However, the environments in which systems are tested would need to be better described than they are today in the literature. Computation time also need to be provided. Better testability and diagnosability are needed to improve ADS dependability in general. The success of the proposed approach rests on the availability of data pertaining to the performance of existing systems. More data is needed to refine the work that has been conducted here.

Probabilistic time Petri nets Probabilistic time Petri nets were used as a basis for the design of dependable perception systems. It was shown that the atomic state class

graph construction could be adapted, allowing for a number of properties to be checked on this new model. However, this leads to a graph that can be much bigger than the strong state class graph adaptation. Future work needs to be conducted to determine if a more efficient construction is possible, by refining only certain strong state classes and enforcing rule AE only where needed. This model has not yet been studied in the context of parametric model-checking.

Data availability The success of the proposed approach is dependant on the availability of data pertaining to existing systems. It is important that a common language is developed to describe the driving environment, that standard metrics are used to compare the performance of existing systems and that thorough information about computation time and performance is given for each of them.

Glossary

Automated Driving System (ADS) An Automated Driving System refers to the hardware and software that are collectively capable of performing the *entire* DDT on a sustained basis. This term is used specifically to describe a level 3, 4 or 5 driving automation system.

ADS-dedicated vehicle (ADS-DV) An ADS-dedicated vehicle (ADS-DV) is a vehicle designed to be operated exclusively by a level 4 or level 5 ADS for all trips.

Dynamic Driving Task (DDT) The term dynamic driving task is used to describe all of the real-time operational and tactical functions required to operate a vehicle in on-road traffic. It excludes the strategic functions such as trip scheduling and selection of waypoints.

Dynamic Driving Task Fallback The dynamic driving task fallback is the response by the user or by an ADS to either perform the DDT or achieve a minimal risk condition after occurrence of a DDT performance-relevant system failure(s) or upon ODD exit.

Monitoring Monitoring is a general term referencing a range of functions involving real-time human or machine sensing and processing of data used to operate a vehicle, or to support its operation.

Operational Design Domain (ODD) The operational design domain refers to the specific conditions under which a given driving automation system or feature thereof is designed to function. An ODD may include geographic, roadway, environmental, traffic, speed, and temporal limitations.

Receptivity Receptivity is an aspect of consciousness characterised by a person's ability to reliably and appropriately focus his/her attention in response to a stimulus. Monitoring includes receptivity.

Supervision Supervision refers to driver activities, performed while operating a vehicle with an engaged level 1 or 2 driving automation system, to monitor the driving automation system's performance, respond to inappropriate actions taken by that system, and to otherwise complete the DDT.

User The term user references the human role in driving automation.

Bibliography

[1] S. M. Veres, N. K. Lincoln, and L. Molnar, "Control engineering of autonomous cognitive vehicles - a practical tutorial," *Cognitive Rational Agents*, pp. 1–31, 2011. 7, 20

[2] "https://medium.com/udacity/how-the-udacity-self-driving-car-works-575365270a40." 9, 21

[3] *Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. Technical Report J3016-201609.* SAE International, 2016. 9, 19, 35, 37, 39

[4] Y. Mortureux, "La sûreté de fonctionnement: méthodes pour maîtriser les risques," *Techniques de l'Ingénieur*, pp. 1–17, 2001. 9, 46

[5] X. Geng, H. Liang, B. Yu, P. Zhao, L. He, and R. Huang, "A scenario-adaptive driving behavior prediction approach to urban autonomous driving," *Applied Sciences*, vol. 7, no. 4, p. 426, 2017. 9, 100

[6] *Directive 2010/40/EU of the European Parliament and Council of 7 July 2010 on the framework for the deployment of Intelligent Transport Systems in the field of road transport and for interfaces with other modes of transport.* Official Journal of the European Union, 2010. 17

[7] G. Dimitrakopoulos and P. Demestichas, "Intelligent Transportation Systems," *IEEE Vehicular Technology Magazine*, vol. 5, pp. 77–84, March 2010. 17

[8] *La sécurité routière en France. Bilan de l'accidentalité de l'année 2016.* Observatoire National Interministériel de la Sécurité Routière (ONISR), 2017. 18

[9] S. Singh, *Critical reasons for crashes investigated in the National Motor Vehicle Crash Causation Survey.* U.S. Department of Transportation. National Highway Traffic Safety Administration (NHTSA), February 2015. 18

[10] *34 plans Nouvelle France industrielle – Véhicule autonome.* Government of the French Republic, 2014. 18

[11] A. Meystel and J. S. Albus, *Intelligent Systems: Architecture, Design and Control*. Wiley Interscience, 2001. 21

[12] A. Reschka, G. Bagschik, S. Ulbrich, M. Nolte, and M. Maurer, "Ability and skill grpahs for system modeling, online monitoring and decision support for vehicle guidance systems," in *IEEE Intelligent Vehicles Symposium*, pp. 933–939, 2015. 21

[13] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang Jr., "Perception, planning, control and coordination for autonomous vehicles," *Machines*, vol. 5, no. 1, 2017. 21

[14] S. Behere and M. Törngren, "A functional architecture for autonomous driving," in *Proceedings of the First International Workshop on Automotive Software Architecture*, pp. 3–10, 2015. 21

[15] E. D. Dickmanns, "Dynamic vision for perception and contorl of motion," *Springer International Publishing*, 2007. 21

[16] S. Ulbrich, A. Reschka, S. Ernst, G. Bagschik, F. Dierkes, M. Nolte, and M. Maurer, "Towards a functional system architecture for automated vehicles," *arXiv preprint arXiv:1703.08557*, 2017. 21, 22

[17] A. Eskandarian, *Handbook of intelligent vehicles*. Springer London, 2012. 21

[18] P. Radecki, M. Campbell, and K. Matzen, "All Weather Perception: Joint Data Association, Tracking, and Classification for Autonomous Ground Vehicles," *arXiv preprint arXiv:1605.02196*, 2016. 23, 111, 113

[19] J. S. Albus, "Features of intelligence required by unmanned ground vehicles," *NIST Special Publication*, pp. 243–252, 2001. 23

[20] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation and scenario for automated driving," in *IEEE Intelligent Transportation Systems Conference, ITSC 2015, Las Palmas de Gran Canaria, Canary Islands, Spain, 15-18 September, 2015*, pp. 982–988, 2015. 23

[21] C. Zinoune, *Autonomous integrity monitoring of navigation maps on board intelligent vehicles*. PhD thesis, UTC Compiègne, 2014. 24

[22] Nissan Global, *Nissan IDS Concept: Nissan's vision for the future of EVs an autonomous driving*, 2015. 24

[23] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016. 25

[24] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion predicion and risk assessment for intelligent vehicles," *ROBOMECH Journal, Springer International Publishing*, vol. 1, pp. 1–14, 2014. 25

[25] N. Kalra, *Challenges and approaches to realizing autonomous vehicle safety*. RAND Corporation, 2017. 25

[26] J. C. de Winter, R. Happee, M. H. Martens, and N. A. Stanton, "Effects of adaptive cruise control and highly automated driving on workload and situation awareness: a review of the empirical evidence," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 27, Part B, pp. 196–217, 2014. 25

[27] A. Kesting, M. Treiber, M. Schönhof, and D. Helbing, "Adaptive cruise control design for active congestion avoidance," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 6, pp. 668–683, 2008. 26

[28] D. A. Dickie and L. N. Boyle, "Drivers' understanding of adaptive cruise control limitations," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 53, pp. 1806–1810, Sage Publications, 2009. 26, 118

[29] U.S. Department of Transportation National Highway Traffic Safety Administration (NHTSA), *ODI Resume on the subject of Automatic Vehicle Control Systems. Investigation PE 16-007*, 2016. 26

[30] F. M. Favarò, N. Nader, S. O. Eurich, M. Tripp, and N. Varadaraju, "Examining accident reports involving autonomous vehicles in California," *PLoS one*, vol. 12, pp. 1–20, 09 2017. 26

[31] "https://www.theguardian.com/technology/2017/nov/09/self-driving-bus-crashes-two-hours-after-las-vegas-launch-truck-autonomous-vehicle." 27

[32] "https://www.nytimes.com/2018/03/26/technology/arizona-uber-cars.html." 27

[33] "www.lefigaro.fr/atualite-france/af-447.php." 28

[34] L. Pike, N. Wegmann, S. Niller, and A. Goodloe, "Copilot: monitoring embedded systems," *Innovations in Systems and Software Engineering*, vol. 9, no. 4, pp. 235–255, 2013. 29

[35] "www.canberratimes.com.au/good-weekend/the-untold-story-of-qf72-what-happens-when-psycho-automation-leaves-pilots-powerless-20170510-gw26ae." 29

[36] P. Koopman and M. Wagner, "Autonomous vehicle safety: An interdisciplinary challenge," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 90–96, 2017. 30

[37] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, and J. Williams, "A perception-driven autonomous urban vehicle," *Journal of Field Robotics*, vol. 25, no. 10, pp. 727–774, 2008. 31, 111, 113

[38] *Preliminary statement of policy concerning automated vehicles.* National Highway Traffic Safety Administration, 2013. 35

[39] T. M. Gasser, C. Arzt, M. Ayoubi, A. Bartels, L. Bürkle, J. Eier, F. Flemisch, D. Häcker, T. Hesse, and W. Huber, *Legal consequences of an increase in vehicle automation. Report F.* Die Bundestanstalt für Straßenwesen (BASt), 2009. 35

[40] J. A. Michon, "A critical view of driver behavior models: What do we know, what should we do," *Human behavior and traffic safety*, pp. 485–520, 1985. 36

[41] J.-C. Laprie, "Dependability of computer systems: concepts, limits, improvements," in *Sixth International Symposium on Software Reliability Engineering*, pp. 2–11, IEEE, 1995. 45

[42] *Terms and definitions related to quality of service and network performance including dependability.* ITU-T, 1994. 46

[43] N. G. Leveson, "Applying systems thinking to analyze and learn from events," *Safety Science*, vol. 49, no. 1, pp. 55–64, 2011. 47

[44] Y. Cherdantseva and J. Hilton, "A reference model of information assurance and security," in *Eighth International Conference on Availability, Reliability and Security (ARES)*, pp. 546–555, IEEE, 2013. 47

[45] J. Bowen and V. Stavridou, "Safety-critical systems, formal methods and standards," *Software Engineering Journal*, vol. 8, no. 4, pp. 189–209, 1993. 48, 56

[46] B. Lussier, *Tolérance aux fautes dans les systèmes autonomes.* PhD thesis, Institut National Polytechnique de Toulouse-INPT, 2007. 48

[47] N. G. Leveson, *Safeware: System Safety and Computers*. Addison-Wesley, 1995. ISBN 0-201-11972-2. 49

[48] N. Narayana, N. Raju, K. Chaithanya, P. R. Reddy, and M. Rajesh, "Failure mode, effects and criticality analysis of 85T dumpers in open cast mines," *International Journal of Innovative Research and Development*, vol. 1, no. 5, pp. 142–153, 2012. 50

[49] D. Lawson, "Failure mode, effect and criticality analysis," in *Electronic Systems Effectiveness and Life Cycle Costing*, pp. 55–74, Springer, 1983. 50

[50] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, *Fault tree handbook*. DTIC Document, 1981. 51

[51] F. Netjasov and M. Janic, "A review of research on risk and safety modelling in civil aviation," *Journal of Air Transport Management*, vol. 14, no. 4, pp. 213–220, 2008. 52

[52] N. Leveson, "A new accident model for engineering safer systems," *Safety science*, vol. 42, no. 4, pp. 237–270, 2004. 52

[53] J. Rasmussen, "Risk management in a dynamic society: a modelling problem," *Safety science*, vol. 27, no. 2, pp. 183–213, 1997. 52

[54] E. Hollnagel and O. Goteman, "The functional resonance accident model," *Proceedings of Cognitive System Engineering in Process Plant*, vol. 2004, pp. 155–161, 2004. 52

[55] C. Perrow, *Normal accidents: Living with high risk technologies*. Princeton University Press, 2011. 52

[56] M. Hall-May, *Ensuring Safety of Systems of Systems*. PhD thesis, Citeseer, 2007. 52

[57] N. Leveson, "The drawbacks in using the term 'system of systems'," *Biomedical Instrumentation & Technology*, vol. 47, no. 2, pp. 115–118, 2013. 52

[58] T. Ishimatsu, N. G. Leveson, J. Thomas, M. Katahira, Y. Miyamoto, and H. Nakao, "Modeling and hazard analysis using STPA," *Conference of the International Association for the Advancement of Space Safety*, 2010. 53

[59] S. Bacherini, A. Fantechi, M. Tempestini, and N. Zingoni, "A story about formal methods adoption by a railway signaling manufacturer," in *FM 2006: Formal Methods*, pp. 179–189, Springer, 2006. 53

[60] P. Bouyer, *Modèles et algorithmes pour la vérification des systèmes temporisés*. PhD thesis, École Normale Supérieure de Cachan, 2002. 53

[61] J. M. Rushby, "Automated formal methods enter the mainstream.," *Communications of the Computer Society of India*, vol. 13, no. 5, pp. 650–660, 2007. 54, 56

[62] P. Cousot, "Abstract interpretation based formal methods and future challenges," in *Informatics*, pp. 138–156, Springer, 2001. 54

[63] E. M. Clarke, W. Klieber, M. Nováček, and P. Zuliani, "Model checking and the state explosion problem," in *Tools for Practical Software Verification*, pp. 1–30, Springer, 2012. 55

[64] J. Rushby, *Formal methods and the certification of critical systems*. Technical Report SRI-CSL-93-7, SRI International, 1993. 55, 56

[65] J. P. Bowen and M. G. Hinchey, "Seven more myths of formal methods," *IEEE Software*, vol. 12, no. 4, pp. 34–41, 1995. 55

[66] S. Liu, V. Stavridou, and B. Dutertre, "The practice of formal methods in safety-critical systems," *Journal of Systems and Software*, vol. 28, no. 1, pp. 77–87, 1995. 56, 57

[67] M. G. Hinchey, "Confessions of a formal methodist," in *Proceedings of the seventh Australian workshop conference on Safety critical systems and software*, pp. 17–20, Australian Computer Society, Inc., 2003. 56

[68] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: Practice and experience," *ACM Computing Surveys (CSUR)*, vol. 41, no. 4, p. 19, 2009. 56

[69] J. P. Bowen and M. G. Hinchey, "The use of industrial-strength formal methods," in *The Twenty-First Annual International Computer Software and Applications Conference (COMPSAC)*, pp. 332–337, IEEE, 1997. 57

[70] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, and M. Deardeuff, "Use of Formal Methods at Amazon Web Services," 2014. 57

[71] D. Sidhu, A. Chung, and T. P. Blumer, "Experience with formal methods in protocol development," *ACM SIGCOMM Computer Communication Review*, vol. 21, no. 2, pp. 81–101, 1991. 57

[72] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011. 57

[73] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997. 57

[74] A. Vahidi and A. Eskandarian, "Research advances in intelligent collision avoidance and adaptive cruise control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, pp. 143–153, 2003. 57

[75] K. Suzuki and H. Jansson, "An analysis of driver's steering behaviour during auditory or haptic warnings for the designing of lane departure warning system," *JSAE Review*, vol. 24, no. 1, pp. 65–70, 2003. 58

[76] S. Ishida and J. E. Gayko, "Development, evaluation and introduction of a lane keeping assistance system," *IEEE Intelligent Vehicles Symposium*, 2004. 58

[77] T. Brandt, R. Stemmer, and A. Rakotoniarainy, "Affordable visual driver monitoring system for fatigue and monotony," *IEEE International Conference on Systems, Mans and Cybernetics*, vol. 7, pp. 6451–6456, 2004. 58

[78] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," *International Symposium on Formal Methods*, vol. 6664, pp. 42–56, 2011. 58

[79] P. Nilsson, O. Hussien, Y. Chen, A. Balkan, M. Rungger, A. Ames, J. Grizzle, N. Ozay, H. Peng, and P. Tabuada, "Preliminary results on correct-by-construction control software synthesis for adaptive cruise control," in *53rd Annual Conference on Decision and Control (CDC)*, pp. 816–823, IEEE, 2014. 59

[80] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomosu vehicle reliability," *Transportation Research Part A: Policy and Practice*, 2016. 59

[81] M. R. Zofka, S. Klemm, F. Kuhnt, T. Schamm, and J. M. Zöllner, "Testing and validating high level components for automated driving: simulation framework for traffic scenarios," in *Intelligent Vehicles Symposium*, pp. 144–150, IEEE, 2016. 59

[82] M. R. Zofka, F. Kuhnt, R. Kohlhaas, C. Rist, T. Schamm, and J. M. Zöllner, "Data-driven simulation and parametrization of traffic scenarios for the development of advanced driver assistance systems," in *Information Fusion (Fusion)*, pp. 1422–1428, IEEE, 2015. 59

[83] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal Transportation Safety*, pp. 15–24, 2016. 59

[84] C. Bergenhem, R. Johansson, A. Söderberg, J. Nilsson, J. Tryggvesson, M. Törngren, and S. Ursing, "How to reach complete safety requirement refinement for autonomous vehicles," *Critical Automotive Applications: Robustness & Safety*, 2015. 59

[85] S. Shalev-Schwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *arXiv preprint arXiv:1708.06374*, 2017. 60

[86] S. Alvarez, Y. Page, and F. Guarnieri, "Analyse des risques pour la conception d'un systeème de conduite automatisèe avec stamp," *2A-Risques liés aux nouveaux usages-architectures robustes 1*, 2016. 60

[87] B. Johnson, F. Havlak, H. Kress-Grazit, and M. Campbell, "Experimental evaluation and formal analysis of high-level tasks with dynamic obstacle anticipation on a full sized autonomous vehicle," *Journal of Field Robotics*, vol. 34, no. 5, pp. 897–911, 2017. 60

[88] P. Feth, D. Schneider, and R. Adler, "A conceptual safety supervisor deinifion and evaluation framework for autonomous systems," *International Conference on Computer Safety, Reliability and Security*, vol. 10488, pp. 135–148, 2017. 60

[89] C. Fan, B. Qi, and S. Mitra, "Road to safe autonomy with data and formal reasoning," *arXiv preprint arXiv:1704.06406*, 2017. 60

[90] K.-D. Kim and P. R. Kumar, "An MPC-based approach to provable system-wide safety and liveness of autonomous ground traffic," *IEEE Control Systems Society*, 2014. 60

[91] A. Rizaldi, F. Immler, and M. Althoff, "A formally verified checker of the safe distance traffic rules for autonomous vehicles," *NASA Formal Methods Symposium*, vol. 9690, pp. 175–190, 2016. 60

[92] M. Kamali, L. A. Dennis, O. McAree, M. Fisher, and S. M. Veres, "Formal verification of autonomous vehicle platooning," *Science of Computer Programming*, vol. 148, pp. 88–106, 2017. 60

[93] N. Goodall, "Ethical decision making during automated vehicle crashes," *Transportation Research Record: Journal of the Transportation Research Board*, 2014. 60

[94] S. Shalev-Schwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016. 60

[95] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Towards proving the adversarial robustness of deep neural networks," *arXiv preprint arXiv:1709.02802*, 2017. 60

[96] S. Jha and V. Raman, "Automated synthesis of safe autonomous vehicle control under perception uncertainty," *NASA Formal Methods*, vol. 9690, pp. 117–132, 2016. 60

[97] R. MacAllister, Y. Gal, M. van der Wilk, A. Shah, R. Cipolla, and A. Weller, "Concrete problems for autonomous vehicle safety: advantages of Bayesian deep learning," *Internation Joint Conferences on Artificial Intelligence*, 2017. 60

[98] W. J. Stewart, *Introduction to the numerical solutions of Markov chains*. Princeton Univ. Press, 1994. 63

[99] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, Inc., 1994. 63

[100] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston, "Automatic verification of real-time systems with discrete probability distributions," *Theoretical Computer Science*, vol. 282, no. 1, pp. 101–150, 2002. 63

[101] C. Eisentraut, H. Hermanns, and L. Zhang, "On probabilistic automata in continuous time," in *Logic in Computer Science (LICS), 2010 25th Annual IEEE Symposium on*, pp. 342–351, IEEE, 2010. 63

[102] N. Bertrand, P. Bouyer, Th. Brihaye, Q. Menet, C. Baier, M. Größer, and M. Jurdziński, "Stochastic timed automata," *Logical Methods in Computer Science*, vol. 10, no. 4:6, 2014. 63

[103] M. Kwiatkowska, G. Norman, and J. Sproston, "Probabilistic model checking of deadline properties in the IEEE 1394 FireWire root contention protocol," *Formal Aspects of Computing*, vol. 14, no. 3, pp. 295–318, 2003. 63

[104] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: Verification of probabilistic real-time systems," in *Computer aided verification*, vol. 6806 of *LNCS*, pp. 585–591, 2011. 63

[105] M. A. Marsan, G. Conte, and G. Balbo, "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems," *ACM Trans. Comput. Syst.*, vol. 2, no. 2, pp. 93–122, 1984. 63

[106] M. K. Molloy, "Discrete time stochastic Petri nets," *IEEE Trans. Software Eng.*, vol. 11, no. 4, pp. 417–423, 1985. 63

[107] E. Vicario, L. Sassoli, and L. Carnevali, "Using stochastic state classes in quantitative evaluation of dense-time reactive systems," *IEEE Trans. Software Eng.*, vol. 35, no. 5, pp. 703–719, 2009. 63

[108] B. Berthomieu and M. Diaz, "Modeling and verification of time dependent systems using time Petri nets," *IEEE trans. on Soft. Eng.*, no. 3, pp. 259–273, 1991. 63, 83

[109] B. Berthomieu and M. Menasche, "An enumerative approach for analyzing time Petri nets," in *Information Processing: proceedings of the IFIP congress 1983* (R. E. A. Mason, ed.), vol. 9 of *IFIP congress series*, pp. 41–46, 1983. 63, 83

[110] J. B. Dugan, K. S. Trivedi, R. M. Geist, and V. F. Nicola, "Extended stochastic Petri nets: Applications and analysis.," tech. rep., DTIC Document, 1984. 64

[111] B. Berthomieu and F. Vernadat, "State class constructions for branching analysis of time Petri nets," in *TACAS'2003*, vol. 2619 of *LNCS*, pp. 442–457, Springer, 2003. 64, 86, 87

[112] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008. 77, 80

[113] G. Norman, D. Parker, and J. Sproston, "Model checking for probabilistic timed automata," *Formal Methods in System Design*, vol. 43, no. 2, pp. 164–190, 2013. 92

[114] J. Kim, R. Rajkumar, and M. Jochim, "Towards dependable autonomous driving vehicles: a system-level approach," *ACM SIGBED Review*, vol. 10, no. 1, pp. 29–32, 2013. 93, 101

[115] M. Gao and M. Zhou, "Control strategy selection for autonomous vehicles in a dynamic environment," in *International Conference on Systems, Man and Cybernetics*, pp. 1651–1656, 2005. 93

[116] C. Tessier, C. Cariou, C. Debain, F. Chausse, R. Chapuis, and C. Rousset, "A real-time, multi-sensor architecture for fusion of delayed observations: application to vehicle localization," in *Intelligent Transportation Systems Conference*, pp. 1316–1321, 2006. 93

[117] B. Johnson and H. Kress-Gazit, "Probabilistic analysis of correctness of high-level robot behaviour with sensor error," *Robotics: Science and Systems*, 2011. 94

[118] J. Fu, N. Atanasov, U. Topcu, and G. J. Pappas, "Optimal temporal logic planning in probabilistic semantic maps," *International Conference on Robotics and Automation*, 2016. 94

[119] J. Van Brummelen, M. O'Brien, D. Gruyer, and H. Najjaran, "Autonomous vehicle perception: The technology of today and tomorrow," *Transportation Research Part C: Emerging Technologies*, 2018. 94

[120] R. H. Rasshofer and K. Gresser, "Automotive radar and lidar systems for next generation driver assistance functions," *Advances in Radio Science*, vol. 3, pp. 205–209, 2005. 94

[121] D. Gruyer, V. Magnier, K. Hamdi, L. Claussman, O. Orfila, and A. Rakotonirainy, "Perception, information processing and modeling: Critical stages for autonomous driving applications," *Annual Reviews in Control*, 2017. 94

[122] M. M. Kokar, J. A. Tomasik, and J. Weyman, "Formalizing classes of information fusion systems," *Information Fusion*, vol. 5, no. 3, pp. 189–202, 2004. 95

[123] M. M. Kokar, K. Baclawski, and H. Gao, "Category theory-based synthesis of a higher-level fusion algorithm: An example," in *Information Fusion*, pp. 1–8, 2006. 95

[124] Y. Qiao, N. Wu, and Z. MengChu, "Petri net-based real-time scheduling of time-constrained single-arm cluster tools with activity time variation," in *International Conference on Robotics and Automation*, pp. 5056–5061, 2012. 95

[125] H. Hu, Y. Yang, Y. Liu, and C. Chen, "Supervisor design and simplification for automated manufacturing systems using colored Petri nets," in *International Conference on Robotics and Automation*, pp. 3826–3832, 2015. 95

[126] Y. Emzivat, B. Delahaye, D. Lime, and O. H. Roux, "Probabilistic time Petri nets," in *Application and Theory of Petri Nets and Concurrency*, pp. 261–280, 2016. 95

[127] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition*, 2012. 96

[128] E. Ohn-Bar and M. M. Trivedi, "Are all objects equal? Deep spatio-temporal importance prediction in driving videos," *Pattern Recognition*, pp. 425–436, 2017. 97

[129] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," *International Journal of Human-Computer Studies*, pp. 907–928, 1995. 97

[130] A. Armand, *Situation Understanding and Risk Assessment Framework for Preventive Driver Assistance*. PhD thesis, Université Paris-Saclay, 2016. 97

[131] B. Khaleghi, A. M. Khamis, F. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013. 102, 107

[132] H. F. Durrant-Whyte, "Sensor models and multisensor integration," *International Journal of Robotics Research*, vol. 7, no. 6, pp. 97–113, 1988. 102

[133] J. Moras, V. Cherfaoui, and P. Bonnifait, "Credibilist occupancy grids for vehicle perception in dynamic environments," in *International Conference on Robotics and Automation*, pp. 84–89, 2011. 102

[134] C. Laugier, I. E. Paromtchik, M. Perrollaz, M. Yong, J.-D. Yoder, C. Tay, K. Kekhnacha, and A. Nègre, "Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety," *Intelligent Transport Systems Magazine*, vol. 3, no. 4, pp. 4–19, 2011. 102

[135] W. Zhao, T. Fang, and Y. Jiang, "Data fusion using improved Dempster-Shafer evidence theory for vehicle detection," in *Fuzzy Systems and Knowledge Discovery*, pp. 487–491, 2007. 103

[136] M. E. El Najjar and P. Bonnifait, "A road-matching method for precise vehicle localization using belief theory and Kalman filtering," *Autonomous Robots*, vol. 19, no. 2, pp. 173–191, 2005. 103

[137] D. Lime, O. H. Roux, C. Seidner, and L.-M. Traonouez, "Romeo: A parametric model-checker for Petri nets with stopwatches," in *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 54–57, 2009. 103, 112

[138] B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. H. Roux, "The expressive power of time Petri nets," *Theoretical Computer Science*, vol. 474, pp. 1–20, 2013. 103

[139] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013. 106

[140] F. Bonsignorio and A. P. del Pobil, "Toward replicable and measurable robotics research," *Robotics & Automation Magazine*, vol. 22, no. 3, pp. 32–35, 2015. 107

[141] L. Cholvy, "Information evaluation in fusion: a case study," in *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pp. 993–1000, 2004. 107

[142] SAE International, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," 2016. 110, 126

[143] F. Moutarde, A. Bargeton, A. Herbin, and L. Chanussot, "Robust on-vehicle real-time visual detection of American and European speed limit signs," in *Intelligent Vehicles Symposium*, pp. 1122–1126, 2007. 111

[144] A. S. Huang, "Lane estimation for autonomous vehicles using vision and lidar," *Massachusetts Institute of Technology Cambridge*, 2010. 111

[145] D. Prajapati and H. J. Galiyawala, "A review on moving object detection and tracking," *International Journal of Computational Intelligence Research*, vol. 5, no. 3, pp. 168–175, 2015. 111

[146] D. Pomerleau, "Visibility estimation from a moving vehicle using the ralph vision system," in *Intelligent Transportation Systems Conference*, pp. 906–911, 1997. 111

[147] R. Rasshofer, M. Spies, and H. Spies, "Influences of weather phenomena on automotive laser radar systems," *Advances in Radio Science*, vol. 9, no. B. 2, pp. 49–60, 2011. 111

[148] J. C. de Winter, R. Happee, M. H. Martens, and N. A. Stanton, "Effects of adaptive cruise control and highly automated driving on workload and situation awareness: A review of the empirical evidence," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 27, Part B, pp. 196 – 217, 2014. Vehicle Automation and Driver Behaviour. 117

[149] P. Thiffault and J. Bergeron, "Monotony of road environment and driver fatigue: a simulator study," *Accident Analysis & Prevention*, vol. 35, no. 3, pp. 381 – 391, 2003. 117

[150] C. Gold, D. Damböck, L. Lorenz, and K. Bengler, "Take over! How long does it take to get the driver back into the loop?," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 57, pp. 1938–1942, SAGE Publications, 2013. 117

[151] M. Bahram, M. Aeberhard, and D. Wollherr, "Please take over! an analysis and strategy for a driver take over request during autonomous driving," in *Intelligent Vehicles Symposium (IV)*, pp. 913–919, IEEE, 2015. 118

[152] C. Ackermann, R. Isermann, S. Min, and C. Kim, "Collision avoidance with automatic braking and swerving," *IFAC*, vol. 47, no. 3, 2014. 118

[153] M. Althoff, D. Althoff, D. Wollherr, and M. Buss, "Safety verification of autonomous vehicles for coordinated evasive maneuvers," in *Intelligent Vehicles Symposium* (IEEE, ed.), 2010. 119

[154] S. Lefevre, C. Laugier, and J. I. Guzman, "Risk assessment at road intersections: Comparing intention and expectation," in *Intelligent Vehicles Symposium*, pp. 165–171, IEEE, 2012. 119

[155] E. Frazzoli, M. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," in *American Control Conference,*, vol. 1, pp. 43–49, IEEE, 2001. 119

[156] J. Park and Ü. Özgüner, "Model based controller synthesis using reachability analysis that guarantees the safety of autonomous vehicles in a convoy," in *International Conference on Vehicular Electronics and Safety*, pp. 134–139, IEEE, 2012. 119

[157] A. Furda and L. B. Vlacic, "Enabling safe autonomous driving in real-world city traffic using multiple criteria decision making," *Intelligent Transportation Systems Magazine*, 2011. 119

[158] P. Stone and M. M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, 2000. 119

[159] G. Weiss, *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999. 119

[160] L. N. Long, S. D. Hanford, O. Janrathitikarn, G. L. Sinsley, and J. A. Miller, "A review of intelligent systems software for autonomous vehicles," in *2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications, CISDA 2007, Honolulu, Hawaii, April 1-5, 2007*, 2007. 119

[161] S. M. Veres, L. Molnar, N. K. Lincoln, and C. P. Morice, "Autonomous vehicle control systems—a review of decision making," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 2, pp. 155–195, 2011. 119

[162] R. van der Horst and J. Hogema, *Time-to-collision and collision avoidance systems*. 1993. 127, 129, 143

[163] L. Rummelhard, A. Nègre, M. Perrollaz, and C. Laugier, "Probabilistic grid-based collision risk prediction for driving application," in *Experimental Robotics*, pp. 821–834, Springer, 2016. 129

[164] W. Wachenfeld, P. Junietz, R. Wenzel, and H. Winner, "The worst-time-to-collision metric for situation identification," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pp. 729–734, IEEE, 2016. 129

[165] N. Goodall, "Ethical decision making during automated vehicle crashes," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2424, pp. 58–65, 2014. 132

[166] N. G. Leveson, *Safeware: System Safety and Computers.* New York, NY, USA: ACM, 1995. 134

[167] M. S. Horswill and F. P. McKenna, *A cognitive approach to situation awareness: Theory and application*, ch. Drivers' hazard perception ability: Situation awareness on the road, pp. 155–175. Ashgate Publishing, Ltd., 2004. 134

[168] T. Batz, K. Watson, and J. Beyerer, "Recognition of dangerous situations within a cooperative group of vehicles," in *Intelligent Vehicles Symposium, 2009 IEEE*, pp. 907–912, IEEE, 2009. 134

[169] "Dangerous road section analysis method and dangerous road section analysis device," 2015. CN 201310731159. 134

[170] "System for correcting and providing real-time dangerous road-section information based on device in the car and its method," 2013. KR 20140119471. 134

[171] "Sections of road traffic safety hazard screening methods and systems," 2012. CN 201210426533. 134

[172] E. Moons, T. Brijs, and G. Wets, "Identifying hazardous road locations: hot spots versus hot zones," in *Transactions on Computational Science VI*, pp. 288–300, Springer, 2009. 134

[173] M. Bíl, R. Andrášik, and Z. Janoška, "Identification of hazardous road locations of traffic accidents by means of kernel density estimation and cluster significance evaluation," *Accident Analysis & Prevention*, vol. 55, pp. 265–273, 2013. 134

[174] I. Yamada and J.-C. Thill, "Comparison of planar and network k-functions in traffic accident analysis," *Journal of Transport Geography*, vol. 12, no. 2, pp. 149–158, 2004. 134

[175] I. B. Gundogdu, "Applying linear analysis methods to gis-supported procedures for preventing traffic accidents: Case study of konya," *Safety Science*, vol. 48, no. 6, pp. 763–769, 2010. 134

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

CENTRALE NANTES

Thèse de Doctorat

Yrvann EMZIVAT

Architecture d'un Organe de Survie pour la Conception de Véhicules Autonomes Agiles et Sûrs

Safety System Architecture for the Design of Dependable and Adaptable Autonomous Vehicles

Résumé

L'automatisation de la tâche de conduite s'inscrit dans un contexte de développement d'une mobilité durable. Il s'agit d'une solution prometteuse qui pourrait bien contribuer à la création de nouveaux moyens de transports sûrs et respectueux de l'environnement. Cependant, concevoir un véhicule autonome reste un défi majeur. C'est pour cette raison qu'il incombe aujourd'hui à l'utilisateur de reprendre le contrôle du véhicule à chaque fois que les circonstances s'y prêtent. Il semble pourtant peu judicieux de confier une telle responsabilité à un être humain dont l'implication dans la tâche de conduite est amoindrie. Les travaux menés dans le cadre de cette thèse portent sur la conception de véhicules autonomes habiles et sûrs. C'est plus précisément le développement d'un système capable de gérer de potentielles défaillances seul, d'élaborer des stratégies de repli par lui-même et de s'adapter à un environnement complexe qui est abordé ici.

Mots clés

véhicules autonomes, sûreté de fonctionnement, architecture, modes dégradés, échappatoire, capacités d'adaptation.

Abstract

Driving automation is often presented as a viable solution to the prevailing challenges of sustainable mobility. It has the potential to create a paradigm shift in transportation technology, by providing a medium for cleaner, safer and more efficient means of transportation, while providing a better user experience overall. However, designing a dependable Automated Driving System is a challenge in itself. Current systems lack common sense and have trouble behaving in a truly cautionary manner, which is why a fallback-ready user is expected to take over in the event of a performance-relevant system failure affecting the dynamic driving task. Yet it seems unwise to rely on human drivers to act as a safety net for the purpose of offsetting the lack of maturity of Automated Driving Systems, for automation changes their active involvement into a monitoring role and creates new challenges, such as complacency, automation dependency, lack of understanding and misuse. This work places emphasis on the design of dependable and adaptable Automated Driving Systems. In particular, the thesis addresses the problem of designing a new ADS primary subsystem, whose role it is to monitor the state of the ADS, supervise its actions and respond as needed to guarantee the safety of its occupants and of others.

Key Words

autonomous vehicles, safety, architecture, graceful degradation, fallback strategy, adaptability

L'UNIVERSITÉ NANTES ANGERS LE MANS