



SAFETY ASSURANCE FRAMEWORK FOR CONNECTED, AUTOMATED MOBILITY SYSTEMS

D7.3

Safety assurance framework demonstration instances

Project short name
SUNRISE

Project full name
Safety assurance framework for connected, automated mobility systems

Horizon Research and Innovation Actions | Project No.
101069573
Call HORIZON-CL5-2021-D6-01



Funded by
the European Union

ccam-sunrise-project.eu/

Dissemination level	Public (PU) - fully open
Work package	WP7: Use cases and framework demonstration instances development
Deliverable number	D7.3: Safety assurance framework demonstration instances
Deliverable responsible	Daniel Hassler, AVL
Status - Version	Final – V1.0
Submission date	28/08/2025
Keywords	Safety Assurance Framework, use cases, demonstration, validation, virtual testing, hybrid testing, physical testing, XiL testing, proving ground testing

Authors

Role	Name
Main author	Daniel Hassler (AVL)
Contributing authors	Sarp Kaya Yetkin (AVL), Anders Thorsén (RISE), Martin Skoglund (RISE), Ramana Reddy Avula (RISE), Amélie Reher (BAST), Bryan Bourauel (BAST), Mauro Da Lio (UNITN), Eloy Collado (IDI), Gisela Leiva (IDI), Thaddäus Menzel (IDI DE), Gabriel Villalonga (CVC), Mohammed Shabbir ALI (VEDECOM), Mohsen Alirezai (SISW), Jobst Beckmann (ika), Georg Stettinger (IFAG), Patrick Weissensteiner (IFAG), Jeroen Uittenbogaard (TNO), Francisco Navas (RESA), Guillaume Mercier (RSA), Anastasia Bolovinou (ICCS), Bernhard Hillbrand (VIF), Martin Kirchengast (VIF), Gerhard Benedikt Weiß (VIF)

Quality Control

	Name	Organisation	Date
Peer review 1	Francisco Navas / Guillaume Mercier	RESA / RSA	23/07/2025
Peer review 2	Stefan de Vries	IDIADA	14/08/2025

Version history

Version	Date	Author	Summary of changes
0.1	30/06/2025	Daniel Hassler	First draft version of document
0.2	08/08/2025	Sarp Kaya Yetkin	Second draft version of document
0.3	28/08/2025	Sarp Kaya Yetkin	Third draft version of document
0.4	28/08/2025	All authors	Fourth draft version of document
1.0	28/08/2025	All authors	Final version

Legal disclaimer

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them.

Copyright © SUNRISE Consortium, 2025.

TABLE OF CONTENTS

EXECUTIVE SUMMARY	23
1 INTRODUCTION	25
1.1 Project introduction	25
1.2 Purpose of deliverable	27
1.3 Intended audience	28
1.4 Deliverable structure and relation to other parts of project	29
1.5 Differences deliverable D7.2 and D7.3	30
2 VALIDATION OF THE SAFETY ASSURANCE FRAMEWORK	32
2.1 Overview of the SAF	32
2.2 Criteria for validation of the SAF	32
2.2.1 SUNRISE DF	33
2.2.2 Query & Concretise	33
2.2.3 Allocate	33
2.2.4 Execute	34
2.2.5 Coverage	34
2.2.6 Test Evaluate	34
2.2.7 Safety case	34
2.2.8 Decide	35
2.3 Covered SAF blocks by use cases	35
2.4 Feedback loop	36
3 USE CASE DEMONSTRATION	37
3.1 Use case 1.1: Urban AD validation – Perception testing	37
3.1.1 Use Case Overview	37
3.1.1.1 Covered aspects of the SAF	38

3.1.1.2	Safety case setup	39
3.1.2	Overview of tested scenarios	39
3.1.3	SAF Block demonstrations	43
3.1.3.1	SUNRISE DF	43
3.1.3.2	Query and Concretise	43
3.1.3.3	Allocate	49
3.1.3.4	Execute.....	51
3.1.3.5	Test Evaluate.....	66
3.1.4	Key take aways	71
3.1.5	Deviations from D7.2.....	72
3.2	Use case 1.2: Urban AD validation – Connected perception testing	73
3.2.1	Use Case Overview.....	73
3.2.1.1	Covered aspects of the SAF	73
3.2.1.2	Safety case setup	74
3.2.2	Overview of tested scenarios	75
3.2.3	SAF Block demonstrations	76
3.2.3.1	Query & Concretize	76
3.2.3.2	Allocate	79
3.2.3.3	Execute.....	79
3.2.3.4	Test Evaluate.....	86
3.2.3.5	Coverage	88
3.2.3.6	Decide	91
3.2.4	Key takeaways	91
3.2.5	Deviations from D7.2.....	91
3.3	VED Contribution for UC 1.2: Urban AD validation – Connected perception testing....	91
3.3.1	Covered aspects of the SAF	91
3.3.2	Overview of Tested Scenarios	92
3.3.2.1	Scenario 111 – Nominal Stop :	93

3.3.2.2	Scenario 121 – Resume on Green:.....	95
3.3.2.3	Scenario 134: ADS stops before and after the red light due to a delayed IMA message. 96	
3.3.2.4	Scenario 211: ADS stops for a priority obstacle vehicle approaching from the right.	99
3.3.2.5	Scenario 211 – Priority Obstacle at Intersection	99
3.3.2.6	Scenario 221 – Urgent Stop for Non-Priority Obstacle (Collision Case).....	100
3.3.2.7	Virtual Testing Framework:	102
3.3.2.8	Hybrid Testing Framework	105
3.3.3	SAF Block Demonstrations	106
3.3.3.1	Allocate	106
3.3.3.2	Execute.....	108
3.3.3.3	Coverage	109
3.3.3.4	Test Evaluate.....	117
3.3.3.5	Safety Argument.....	130
3.3.4	Key Takeaways	131
3.3.5	Deviations from D7.2.....	131
3.4	Use case 1.3: Urban AD validation – Cooperative perception testing	132
3.4.1	Use Case Overview.....	132
3.4.1.1	Covered aspects of the SAF	133
3.4.1.2	Safety case setup	134
3.4.2	Overview of tested scenarios	136
3.4.3	SAF Block demonstrations	138
3.4.3.1	Query & Concretize	138
3.4.3.2	Allocate	139
3.4.3.3	Execute.....	142
3.4.3.4	Test evaluate	149
3.4.3.5	Coverage	163
3.4.3.6	Decide	165

3.4.4	Key take aways	165
3.4.5	Deviations from D7.2.....	166
3.5	Use Case 2.1: Traffic jam AD validation – Safety assessment & Decision making....	167
3.5.1	Use Case Overview.....	167
3.5.1.1	Covered aspects of the SAF	167
3.5.1.2	Safety case setup.....	168
3.5.2	Overview of tested scenarios	168
3.5.3	SAF Block demonstrations	169
3.5.3.1	SUNRISE DF.....	169
3.5.3.2	Query	170
3.5.3.3	Concretise	172
3.5.3.4	Allocate	174
3.5.3.5	Execute.....	175
3.5.3.6	Test Evaluate.....	180
3.5.3.7	Safety Case	184
3.5.3.8	Decide	185
3.5.4	Key take aways	185
3.5.5	Deviations from D7.2.....	185
3.6	Use case 3.1: Highway AD validation – Map based perception & decision making...	186
3.6.1	Use Case Overview.....	186
3.6.1.1	Covered aspects of the SAF	186
3.6.1.2	Safety case setup.....	187
3.6.2	Overview of tested scenarios	188
3.6.3	SAF Block demonstrations	188
3.6.3.1	Query & Concretize	188
3.6.3.2	Allocate	189
3.6.3.3	Execute.....	190
3.6.3.4	Test Evaluate.....	194

3.6.3.5	Coverage	200
3.6.3.6	Decide	201
3.6.4	Key take aways	201
3.6.5	Deviations from D7.2.....	201
3.7	Use case 3.2: Highway AD validation – Cooperative perception & decision making & control.....	203
3.7.1	Use Case Overview.....	203
3.7.1.1	Covered aspects of the SAF	203
3.7.1.2	Safety case setup	204
3.7.2	Overview of tested scenarios	205
3.7.3	SAF Block demonstrations	205
3.7.3.1	Query & Concretize	205
3.7.3.2	Allocate	208
3.7.3.3	Execute.....	209
3.7.3.4	Coverage / Test Evaluate / Safety Case / Decide	213
3.7.4	Key take aways	224
	Deviations from D7.2.....	225
3.8	Use case 4.1: Freight vehicle automated parking validation – Truck low-speed perception & decision making	226
3.8.1	Use Case Overview.....	226
3.8.1.1	Covered aspects of the SAF	227
3.8.1.2	Safety case setup	228
3.8.2	Overview of tested scenarios	228
3.8.3	SAF Block demonstrations	229
3.8.3.1	Query and Concretize	230
3.8.3.2	Allocate	231
3.8.3.3	Execute.....	237
3.8.3.4	Test Evaluate.....	248

3.8.3.5	Safety Case	251
3.8.3.6	Decide	252
3.8.4	Key take aways	252
3.8.5	Deviations from D7.2.....	252
3.9	Use case 4.2: Freight vehicle automated parking validation – Truck low-speed connected perception cyber-security	253
3.9.1	Use Case Overview.....	253
3.9.1.1	Covered aspects of the SAF	254
3.9.1.2	Safety case setup	255
3.9.2	Overview of tested scenarios	255
3.9.3	SAF Block demonstrations	255
3.9.3.1	Query and concretize	255
3.9.3.2	Execute.....	255
3.9.3.3	Test Evaluate.....	258
3.9.4	Key take aways	259
3.9.5	Deviations from D7.2.....	259
4	CONCLUSIONS	260
5	REFERENCES	262

LIST OF FIGURES

Figure 1: Safety Assurance Framework stakeholders	26
Figure 2: Workplan of the SUNRISE Project	27
Figure 3: Overview of the SAF	32
Figure 4: SAF demonstrated blocks in this UC's demo (s).	39
Figure 5: A Pedestrian crossing the street behind a parked car	40
Figure 6: Car-to-Pedestrian Farside Adult [1]	40
Figure 7: Car-to-Pedestrian Nearside Adult [1]	41
Figure 8: Car-to-Pedestrian Turning Adult [1]	41
Figure 9: Right/Left turn junction scenarios	41
Figure 10: A parked car opens its door with oncoming traffic	42
Figure 11: A parked car is leaving a parking spot	42
Figure 12: Car following	42
Figure 13: Left turn scenario in different weather and lighting conditions	43
Figure 14: Simulation framework at Siemens using Simcenter HEEDS (as the test case manager) and Simcenter Prescan (as the simulation tool) (left figure), Simulation framework in D4.1 (right figure) ..	44
Figure 15: concrete scenarios with different weather and lighting condition in signalised and non-signalised intersection	44
Figure 16: The left column illustrates the open-loop method's simulation results, while the right column shows the corresponding closed-loop results. Due to the guided search via objective function minimisation, the open-loop approach needs significantly fewer simulations to find the most critical cases.	46
Figure 17: Comparison of two closed-loop variants (pattern search and surrogate optimisation) with the open-loop method for all 2D parameter variations. The first subplot shows the number of simulations of each approach. The second and third subplots show the minimum TTC and inter-vehicle-distance values found during the simulations. The final plot depicts the objective function value together with the evaluation result (F ... failed, P ... passed). For this scenario, surrogate optimisation finds scenarios with collisions in the least amount of time.	46
Figure 18: Two examples of 3D parameter variations for this scenario. The closed-loop method (right column) finds critical cases with fewer simulations than the open-loop method.	47
Figure 19: In the case of 3D variations, surrogate optimisation also finds scenarios with collisions in the least amount of time.	47
Figure 20: In this scenario, the open-loop method (left subplot) did not find any collision in contrast to the optimisation-based closed-loop technique (right subplot).	48
Figure 21: Comparison of surrogate optimisation and open-loop method for 2D variations in this scenario. The closed-loop approach needs fewer simulations and finds collisions where the open-loop method does not (light blue bars, corresponding to the figure above).	48
Figure 22: Closed-loop and open-loop result comparison for 4D variations. Again, the optimisation-based method achieves similar results with fewer simulation executions.	49
Figure 23: A section of the Excel file that was used to compare the requirements of the test case with the capabilities of the selected simulation framework.	51
Figure 24: Framework used by the CVC to prove the SAF framework for the selected scenarios. It includes the scenario generation and the AD stack used in the vehicle. Blue boxes represent modules run by the CARLA simulator. Red boxes are trained AI models. White boxes are coded scripts.	52

Figure 25: Left: Camera sample generated by the CARLA simulator. Right: Rendered image by the photorealistic AI.....	52
Figure 26: The radar power over range.	54
Figure 27: a) Interference between different sensors – same ramp duration for interferer and ego sensor b) Interference between different sensors. For both: The red arrow indicates the real target. .	54
Figure 28: a) Virtual ground truth, azimuth angle is 0 degree, b) Small deviation, c) Large deviation .	55
Figure 29: Importing real data to Prescan Critical Scenario Creation tool.....	55
Figure 30: Optimisation process using Prescan and HEEDS.....	56
Figure 31: Generated a dissimilar set of unknown unsafe scenarios.	56
Figure 32: Simulation topology of the framework implemented in Model.CONNECT.	57
Figure 33: Near-critical scenario with a right-turn manoeuvre at a T-junction. The bounding box of the other car is coloured orange when it is detected by the sensor; otherwise, it is coloured yellow.....	59
Figure 34: Time-series data of the right turn manoeuvre.....	60
Figure 35: Critical scenario with a right-turn manoeuvre at an X-junction and two other cars.	61
Figure 36: Time-series data of the 2nd scenario.	62
Figure 37: Critical scenario with a left-turn manoeuvre at a T-junction and another car.	63
Figure 38: Time-series data of the 3rd scenario.	64
Figure 39: RESA Autonomous driving prototype for proving ground testing.	64
Figure 40: AD software architecture.	65
Figure 41: Proving ground scenarios.	66
Figure 42: Top: The sensor model without sensor phenomena and HAD, Bottom: The sensor model with sensor phenomena and HAD.	69
Figure 43: Comparison between the output of the proposed system (in blue) and ground truth (in orange) for executed tests. From left to right: Car following, vehicle leaving the parking and parked vehicle with open door.	71
Figure 44: SAF demonstrated blocks in this UC's demo (s).	74
Figure 45: Use Case 1.2 includes four subcases.	76
Figure 46: Logical scenario for UC 1.2-B.....	77
Figure 47: Initial sampling for UC 1.2-B.....	78
Figure 48: ika's automated vehicle test platform karl.....	80
Figure 49: Arial view of the Aldenhoven Testing Center.....	80
Figure 50: Overview of UC1.2-A real world test.....	81
Figure 51: Digital view of UC1.2-A real world test.	81
Figure 52: Overview of UC1.2-B real world test.....	82
Figure 53: CCAM vehicle data of UC1.2-B real world test.....	83
Figure 54: Digital view of UC1.2-B real world test.	84
Figure 55: Overview of UC1.2-C real world test.	84
Figure 56: Digital view of UC1.2-C real world test.	85
Figure 57: Overview of UC1.2-D real world test.	85
Figure 58: Digital view of UC1.2-D real world test.	86
Figure 59: Surrogate model for UC 1.2-D.....	87
Figure 60: UC 1.2-B. First iteration in building a surrogate model.....	87
Figure 61: UC 1.2-B. Second iteration in building a surrogate model.	88

Figure 62: UC 1.2-D. The random sampling (left) identified 9 collisions with TTI>5 s. The LHS sampling found 11 for TTI>5 s. The horizontal black lines indicate successful evasive manoeuvres (from the initial TTI to an increased TTI by deceleration). Unfortunately, in some cases, the correction does not take place.	89
Figure 63: UC 1.2-B. Second iteration in building a surrogate model.	90
Figure 64: UC 1.2-B. Evaluation of unsafe regions. Top: for pedestrian at walking speed. Bottom, for running pedestrians.	90
Figure 65: Diagram illustrating the SAF components (allocation, execution, coverage, test evaluation) that are validated by VED for Use Case 1.2D.	92
Figure 66: Scenario 111 – Nominal Stop illustration.	93
Figure 67: Initial ego vehicle speed.	93
Figure 68: Distribution of sample of distance of the ego vehicle to the intersection.	94
Figure 69: Scenario 121 – Resume on Green illustration.	95
Figure 70: Distribution of samples of speed.	95
Figure 71: Scenario 134: ADS stops before a red light due to a delayed IMA message.	96
Figure 72: Scenario 134: ADS stops after the red light due to a delayed IMA message.	97
Figure 73: Scenario 211: Illustration of ADS stops for a priority obstacle vehicle approaching from the right.	99
Figure 74: Scenario 221 – Urgent Stop for Non-Priority Obstacle (Collision Case) illustration.	100
Figure 75: Virtual testing framework architecture using MATLAB and SIMULINK.	102
Figure 76: Intersection manager architecture.	103
Figure 77: Ego vehicle model in control simulator.	104
Figure 78: Hybrid testing framework.	105
Figure 79: UC 1.2 Allocate validation: final allocate structure diagram obtained highlighting the components allocated to different testing frameworks.	108
Figure 80: Traffic light violation for scenario 111 for orange light duration of 3s.	109
Figure 81: Traffic light violation for scenario 111 for orange light duration 5s.	110
Figure 82: Traffic light stop maximum deceleration.	111
Figure 83: Traffic light stop maximum jerk.	111
Figure 84: Traffic light state delay, acceleration, and jerk.	112
Figure 85: Intersection deceleration max.	113
Figure 86: Intersection maximum jerk.	114
Figure 87: Intersection non-priority obstacle scenario 221: maximum deceleration.	115
Figure 88: Intersection non-priority obstacle scenario 221: maximum jerk.	115
Figure 89: Scenario 221: Identification of critical scenario with possible collisions considering various initial speed V_{init} and distance to intersection.	116
Figure 90: Identification of critical scenarios for all the considered scenarios. These scenarios will be tested in hybrid setup.	117
Figure 91: CAM delivery ratio (%) by scenarios.	118
Figure 92: CAM average delay in seconds by scenarios.	118
Figure 93: Ego vehicle initial speed comparison for virtual testing and hybrid testing.	120
Figure 94: Ego vehicle remaining distance to intersection comparison for virtual testing and hybrid testing.	120
Figure 95: Speed adaptation decision making performance.	121

Figure 96: Decision making to stop the ego vehicle.	122
Figure 97: Robustness of decision-making (number of state changes).	123
Figure 98: Delay (s) of decision making.	124
Figure 99: Vehicle control accuracy (distance in meters to the traffic line or collision zone).	125
Figure 100: Vehicle control delay (s).	126
Figure 101: Collision with obstacle vehicle.	127
Figure 102: Comfort control: average acceleration on the sequence studied.	128
Figure 103: Comfort control: peak acceleration on the sequence studied.	128
Figure 104: KPI Results of the decision-making performance and the longitudinal vehicle control performance.	129
Figure 105: KPI results of the comfort vehicle control.	130
Figure 106: UC 1.3 visual sketch	133
Figure 107: SAF demonstrated blocks in this UC's demo (s). Up: ICCS SAF instance; Down: VED SAF instance.	134
Figure 108: Safety case building steps of UC1.3 based on SUNRISE SAF.	136
Figure 109: Concrete scenario variations in CARLA : day vs. night, adult vs. kid, fog, rain, darting out direction vertical vs. non vertical.	137
Figure 110: Virtual testing, CPM transmission from vehicle V1 and CPM reception by ego vehicle..	138
Figure 111: Allocation structure (initially presented as Figure 79 in Sunrise D3.3).	142
Figure 112: Snapshot from the two virtual testing environments : up, snapshot from ICCS CARLA-ROS custom BEV GUI; down, Snapshot from ICCS Msvan3t/CARLA co-simulation custom GUI. ...	144
Figure 113 : Virtual test environment architecture and GUI (Approach 2).....	145
Figure 114: Snapshot from ICCS hybrid setup (National Technical University of Athens premises and CARLA)	146
Figure 115: VED Hybrid testing framework for UC1.2A.....	147
Figure 116: SUMO/OMNET++ structure.....	148
Figure 117: Snapshot from VED hybrid setup (VED premises in Paris and SUMO/OMNET++)	148
Figure 118: Snapshot from the three virtual testing environments : a) Snapshot from OMNET++/SUMO VED tool b) Snapshot from ICCS CARLA custom BEV GUI c) Snapshot from ICCS Msvan3t/CARLA ICCS custom GUI	149
Figure 119: Local and CPM perception coverage of EGO for Vinit 30 km/h and pedD 15m.	151
Figure 120: Local and CPM perception coverage of EGO for Vinit 60 Km/h and pedD = 30m.	151
Figure 121: Number of CPM transmitted by the vehicle V1	152
Figure 122: Number of CPM received by EGO.....	153
Figure 123: PDR of EGO.	154
Figure 124: Mean latency of CPM.	155
Figure 125: EGO speed for Vinit 30 km/h and pedD 15m.	156
Figure 126: EGO speed for Vinit 60 km/h and pedD 30m.	157
Figure 127: EGO acceleration for Vinit 30 km/h and pedD 15m.....	158
Figure 128: EGO acceleration for Vinit 60 km/h and pedD 30m.....	159
Figure 129: EGO TTC for Vinit 30 km/h and pedD 15m.	160
Figure 130: EGO TTC for Vinit 60 km/h and pedD 30m.	161
Figure 131: EGO maximum jerk.	162

Figure 132: EGO collision outcomes.	163
Figure 133: Scenario library and scenario query panes within Safety PoolTM SCDB.	170
Figure 134: Test case selection pane in Streetwise SCDB.	171
Figure 135: Parameter distribution for an example test case in Streetwise SCDB.	172
Figure 136: Scenario Parameter Space in AVL Scenius.	173
Figure 137: Scenius Test Case Generation (Concretization).	174
Figure 138: Exemplary Cut-In scenario in three different test environments.....	176
Figure 139: Test case setup for proving ground testing.	178
Figure 140: Test case setup public road testing.	179
Figure 141: Four-way split screen from public road testing in a cut-in scenario.....	179
Figure 142: KPI results of cut-out scenarios.	181
Figure 143: Cut-out simulation result at 50 km/h.	182
Figure 144: Data plot from proving ground testing showcasing a cut-in scenario with ego-vehicle speed of 60 km/h and target-vehicle speed of 30 km/h.	183
Figure 145: Data plot from public road testing of an ALKS system with a leading target vehicle.	184
Figure 146: SAF demonstrated blocks in this UC's demo (s).	187
Figure 147: Test case setup (use case B). On the left is the simulation environment, featuring a sensor view in the top left and a map view in the top right. On the right is the proving ground setup.	188
Figure 148: Use Case 3.1 includes two subcases: A) testing speed adaptation (left) and B) testing speed choice and lane maintenance (right).	188
Figure 149: ODD-based allocation of two clusters of test cases to proving ground test instance (indicated in blue) and virtual simulations test instance (indicated in green color).	190
Figure 150: CAVKit rack used in the SUNRISE project.	192
Figure 151: Use Case 3.1: Ego Vehicle used.	193
Figure 152: Communication workflow between related systems used in UC 3.1.	194
Figure 153: Use Case 3.1 A: Contour plots showing the deceleration required to comply with the new speed limit.	195
Figure 154: Use Case 3.1 A: Contour plots illustrating velocity undershoot. Hard brakes, on the right, can cause significant undershoot, negatively impacting traffic flow. When the speed reduction is minor, the overshoot remains under 5 km/h (shown in the blue region).....	196
Figure 155: Use Case 3.1 B: The system reliably stays within the lane.	197
Figure 156: Use Case 3.1 B: The system uses lateral accelerations compatible with human choice in curves (left) and human combined accelerations (right) [8].	197
Figure 157: Use Case 3.1 A: Scatter plot showing the overshoot/undershoot at speed limit sign position.	198
Figure 158: Use Case 3.1 A: Scatter plot showing the overshoot for maximal VUT speed and undershoot for minimal VUT speed.	199
Figure 159: Use Case 3.1 B: The system reliably stays within the lane.	200
Figure 160: SAF demonstrated blocks in this UC's demo (s).	204
Figure 161: Use Case 3.2 includes three subcases.	205
Figure 162: Search interface within Safety PoolTM SCDB demonstrating the relevant tag query of OpenLabel tags.	208
Figure 163: Use Case 3.2: Cooperative Vehicle and Ego Vehicle used.	211
Figure 164: CAVKit Physical Architecture.....	212

Figure 165: Communication workflow between related systems used in UC 3.1 and UC 3.2.	212
Figure 166: Negotiation workflow between related platforms used in UC 3.2-C.	213
Figure 167: Use Case 3.2 2-C-1. Confidence levels associated with the collision class.	214
Figure 168: Use Case 3.2 2-C-1. Use of surrogate model to evaluate the collision risk for three behaviours of the cut-in vehicle, at two confidence levels and for two CCAM systems.	214
Figure 169: Use Case 3.2 2-C-1. Analysis of scenarios at the safe-unsafe boundary and clustering into different types of failure modes.	215
Figure 170: Use Case 3.2 2-C-1. Use of the surrogate model to recommend safe speed policies. ...	215
Figure 171: Use Case 3.2-C1. Scatter plot showing the minimal longitudinal free distance during cut-in.....	216
Figure 172: UC 3.2-C2. Successful merge trajectories for the cautious (top) and aggressive (bottom) settings of Ego 2.	217
Figure 173: UC 3.2-C2. Successful merge trajectories for the cautious (top) and aggressive (bottom) settings of Ego 2.	218
Figure 174: Use case 3.2-C-2. Results from proving ground execution at 120km/h.	218
Figure 175: UC 3.2-A. Distribution of the bumper-to-bumper distance at stop (top) and the contour plots across the logical scenario (the x-axis is the time gap when the first communication is received, and the y-axis is the difference in velocity. The red zones are the regions with distance below 2.8 meters- The minimum distance is indicated.....	220
Figure 176: UC 3.2-A. Distribution of Ego vehicle deceleration after analyzing the 1000 simulations. The large decelerations observed in some cases happen because, at the detection time, the cooperative vehicle might already be slower and/or already braking, and the communication delay might be too late.....	221
Figure 177: UC 3.2-A. Results from proving ground execution at 100km/h.	222
Figure 178: UC 3.2-B. Distribution of the bumper-to-bumper distance at stop (top) and the contour plots across the logical scenario (the x-axis is the time gap when the first communication is received, and the y-axis is the difference in velocity.	223
Figure 179: UC 3.2-B. Results from proving ground execution at 80km/h.	224
Figure 180: Illustration of the use case with a truck with semitrailer to dock at a logistic hub.....	226
Figure 181: SAF demonstrated blocks in this UC's demo (s).	227
Figure 182: Schematic view of the test scenario to be demonstrated.	229
Figure 183: Schematic view of the test set-up connected to the select SPIs.	230
Figure 184: Photo from a manual test drive with Chalmers Revere's full-size Volvo FH16 "Rhino" with semitrailer.....	230
Figure 185: GNSS trajectories from the truck perform repeated parking manoeuvres.	230
Figure 186: Concluded logical scenario with some possible geometrical distributions.	231
Figure 187: The chosen starting positions for the truck with semitrailer.....	231
Figure 188: Overview of the initial allocation process [5].....	232
Figure 189: Top module for ISO 34503 ODD in Pkl.	232
Figure 190: Part of the ODD Taxonomy from ISO 34 503 [16].....	233
Figure 191: Example: Definition of drivable area.	233
Figure 192: Instantiating an ODD from the template.	234
Figure 193: The rendered part of the JSON ODD visualized by PlantUML.....	234
Figure 194: The Pkl formalized ISO 34503 template extended with four test environment attributes, specified in both test case requirements and environment capabilities for valid comparison.	235

Figure 195: An example showing a subset of the test environment requirement parameters, out of 300 ODD configurable elements.	235
Figure 196: Example for inclusions of weather parameters available in CARLA in the Pkl description.	236
Figure 197: Example of METAFODD description of the CARLA test environment capability.	236
Figure 198: Automatic allocation evaluation.	237
Figure 199: Illustration of the building blocks of the WayWise framework.	237
Figure 200: Illustration of a tractor with a connected semitrailer and important angle.	238
Figure 201: Calculated movements of truck respectively trailer back wheels.	238
Figure 202: Schematic view of the simulation set-up to be used for sub-UC4.1.	239
Figure 203: Snapshot from simulation combining Carla and WayWise.	240
Figure 204: Plots of the rear-axle middle positions of the semitrailer for 50 simulations with the same starting position. The red dots show the requested trajectory and the other coloured lines illustrate the simulated positions.	241
Figure 205: Plots of the rear-axle middle positions of the semitrailer for 200 simulations with random starting positions. The red dots show the requested trajectory and the other coloured lines illustrate the simulated positions.	242
Figure 206: Photo of the 1:14 model truck with a semitrailer. On the roof of the truck is the GNSS antenna.	243
Figure 207: Schematic view of the software set-up with RISE model truck.	243
Figure 208: Photo of the physical test setup for running the model truck.	244
Figure 209: Schematic view of the important parts for the model truck physical test set-up.	245
Figure 210: Graph showing the monitored truck during the reverse driving operation.	245
Figure 211: The model truck with semitrailer positioned in the staging area.	246
Figure 212: Ten repeated tests run with the model truck starting from the middle of the staging area.	247
Figure 213: Test runs with the model truck with starting positions according to Figure 187.	247
Figure 214: Photo from a test run with Chalmers Volvo FH16 "Rhino".	248
Figure 215: The final position for the rear axis in the docking area for 10 runs with the 1:14 scale truck.	249
Figure 216: The distance measured by using ultrasonic sensors to the wall behind the trailer for 10 runs with the 1:14 scale truck from the same starting position.	249
Figure 217: The distance measured using ultrasonic sensors to a semitrailer parked alongside the docking area trailer for 10 runs with the 1:14 scale truck from the same starting position.	249
Figure 218: Monte-Carlo simulation of the trajectories for the rear wheel axis of the truck respectively semitrailer using the Python model. Rather large limits for the staging area is assumed.	250
Figure 219: A test scenario with sun elevation of 9° in a hardware-in-the-loop scale truck environment.	251
Figure 220: A test scenario with sun elevation of 6° in Carla simulator UE4 environment.	251
Figure 221: UC 4.2 functional representation.	253
Figure 222: SAF demonstrated blocks in this UC's demo (s).	254
Figure 223: Execute' block functional architecture.	256
Figure 224: Execute block snapshot showing the CARLA-msvan3t console hosting information about CPM exchange (purple window), the virtual RSU camera and lidar perception (annotated windows on the right side of the picture), the CARLA birds-eye-view in the center and the Waywiser GUI hosting information on (black window on the bottom).	257

Figure 225: Test evaluate block snapshot showing network and security-related KPIs applying specially on this UC.....	258
---	-----

LIST OF TABLES

Table 1: Overview of the validation activities per use case	36
Table 2: Overview of the SUNRISE use cases	37
Table 3: Contributions to the SAF validation per Partner in UC1.1	38
Table 4: Perception sensor settings for all scenarios.	58
Table 5: Perception (Precision, Recall, Accuracy, and F1-Score) and Pass/Fail (Collisions and completed route) metrics for the executed tests.	67
Table 6: Precision, Recall, Accuracy, and F1-Score metrics for the executed tests. (Part 1)	69
Table 7: Precision, Recall, Accuracy, and F1-Score metrics for the executed tests. (Part 2)	70
Table 8: Contributions to the SAF validation per Partner in UC1.2	73
Table 9: UC 1.2-A logical scenario parameters.	76
Table 10: UC 1.2-D logical scenario parameters.	78
Table 11: Allocate validation, scenario requirements and virtual testing framework capabilities comparison.	106
Table 12: Contributions to the SAF validation per Partner in UC1.3.	134
Table 13: Contributions to the SAF validation per Partner in UC1.3.	136
Table 14: Comparison of requirements and testing framework capabilities.	140
Table 15: Contributions to the SAF validation per Partner in UC2.1.	168
Table 16: Overview of use cases and allocated test environments.	175
Table 17: Overview of safety performance metrics for ALKS system.	180
Table 18: From UN-R 157 [6] showing minimum headway distances for ALKS.	181
Table 19: Contributions to the SAF validation per Partner in UC3.1	186
Table 20: UC 3.1-A logical scenario parameters	189
Table 21: UC 3.1-B logical scenario parameters	189
Table 22: Contributions to the SAF validation per Partner in UC3.2	203
Table 23: UC 3.2-A logical scenario parameters	206
Table 24: UC 3.2-B logical scenario parameters	206
Table 25: UC 3.2-C2 logical scenario parameters	207
Table 26: Logical scenario parameter ranges.	207
Table 27: Contributions to the SAF validation per Partner in UC4.1	227
Table 28: KPIs for different test environments	252
Table 29: Contributions to the SAF validation per Partner in UC4.2	254

ABBREVIATIONS AND ACRONYMS

Abbreviation	Meaning
ACC	Adaptive Cruise Control
AD	Automated Driving
ADS	Automated Driving System
AEB	Autonomous Emergency Braking
AI	Artificial Intelligence
ALKS	Automated Lane Keeping Systems
ASAM	Association for Standardisation of Automation and Measuring Systems
AVL	AVL List GmbH
BAST	Bundesanstalt für Straßenwesen (Federal Highway Research Institute)
CAM	Cooperative Awareness Message
CAF	Connected Automated Function
CARLA	Car Learning to Act (Open-Source Simulator)
CCAM	Connected, Cooperative, and Automated Mobility
CPM	Collective Perception Message
COTSATO	CONcretizing Test Scenarios and Associating Test Objectives
CPFA	Car-to-Pedestrian Farside Adult
CVC	Computer Vision Center
DDT	Dynamic Driving Task
DF	Data Framework
EC	European Commission
EGO	The vehicle under test (from Latin "ego" meaning "self")
ERTRAC	European Road Transport Research Advisory Council
ETSI	European Telecommunications Standards Institute

FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FOV/FoV	Field of View
GLOSA	Green Light Optimized Speed Advisory
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GUI	Graphical User Interface
HAD	Highly Automated Driving
HARA	Hazard Analysis and Risk Assessment
HD	High Definition
HiL	Hardware-in-the-Loop
HWP	Highway Pilot
ICA	Intersection Collision Avoidance
IMA	Intersection Moving Assist
IMU	Inertial Measurement Unit
IPG	IPG Automotive (Company Name)
ISMR	In-Service Monitoring and Reporting
ITS	Intelligent Transport Systems
ITS-G5	European standard for Intelligent Transport Systems
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LiDAR	Light Detection and Ranging
MAC	Media Access Control
MCM	Manoeuvre Coordination Message
METAFODD	Methodology for Test case Allocation based on a Formalized ODD
MQTT	Message Queuing Telemetry Transport
NATM	New Assessment/Test Method for Automated Driving

NCAP	New Car Assessment Program
OBU	On-Board Unit
ODD	Operational Design Domain
OMNeT++	Objective Modular Network Testbed in C++
OSI	Open Simulation Interface
PDF	Probability Density Function
PG	Proving Ground
PKL	Configuration Language
PR	Public Road
RESA	Renault Site Autonomous
ROS	Robot Operating System
ROS2	Robot Operating System 2
RSU	Road Side Unit
SAF	Safety Assurance Framework
SCDB	Scenario DataBase
SiL	Software-in-the-Loop
SOTIF	Safety Of The Intended Functionality
SPI	Safety Performance Indicator
SRI	Scenario Relatedness Index
SUT	System Under Test
TARA	Threat Analysis and Risk Assessment
TJC	Traffic Jam Chauffeur
TTC	Time-To-Collision
UC	Use Case
UNECE	United Nations Economic Commission for Europe
UNITN	University of Trento
V2V	Vehicle-to-Vehicle

V2X	Vehicle-to-Everything
VED	VEDECOM (Vehicle Environmental Data)
VIF	Virtual Vehicle
VSM	Vehicle Simulation Model
WP	Work Package

EXECUTIVE SUMMARY

Safety assurance of Cooperative, Connected, and Automated Mobility (CCAM) systems is a crucial factor for their successful adoption in society, yet it remains a significant challenge. It is generally acknowledged that for higher levels of automation, the validation of these systems by conventional test methods would be infeasible. Furthermore, certification initiatives worldwide struggle to define a harmonized safety assurance approach enabling massive deployment of CCAM systems.

The **SUNRISE** project develops and demonstrates a **CCAM Safety Assurance Framework (SAF)**. The overall objective of the SUNRISE project is to accelerate the large-scale and safe deployment of CCAM systems. In alignment with international twin projects and initiatives, the project aims to achieve this objective by providing a SAF consisting of three main components: a Method, a Toolchain and a Data Framework. The **Method** is established to support the SAF safety argumentation, and includes procedures for scenario selection, sub-space creation, dynamic allocation to test instances and a variety of metrics and rating procedures. The **Toolchain** contains a set of tools for safety assessment of CCAM systems, including approaches for virtual, hybrid and physical testing. The **Data Framework** provides online access, connection and harmonization of external Scenario Databases (SCDBs), allowing its users to perform query-based extraction of safety relevant scenarios, allocation of selected scenarios to a variety of test environments, and reception of the test results.

This deliverable presents the practical demonstration and validation of **the SUNRISE SAF** in eight representative use cases. Positioned within **Work Package 7**, this deliverable D7.3 substantiates the SAF's capability to support comprehensive safety validation of CCAM systems by applying its methods and tools across diverse domains—urban mobility, highway driving, and freight operations.

In the broader scope of the SUNRISE project, deliverable D7.3 plays a pivotal role in **demonstrating and validating the SAF**, one of the key outcomes designed to foster a **harmonized, traceable safety assurance approach**. The deliverable is strategically relevant for developers, regulators, and technology evaluators, as it outlines how safety assessment can be operationalized by interfacing virtual, hybrid, and physical testing environments with structured safety argumentation.

Deliverable D7.3 demonstrates how individual SAF components — **Scenario, Environment, and Safety Argument** blocks — are demonstrated and validated within the use cases. The demonstrators contribute toward several project milestones, including **feedback integration into SAF development**, evaluation of **test environments**, and deployment of novel safety assessment techniques (e.g., **concretization methods, coverage metrics, surrogate modeling**, and **hybrid testing environments**).

The **core of this deliverable** consists of:

- **Detailed demonstration of SAF blocks** through practical demonstration and validation in eight use cases including automated urban perception, decision-making at intersections, highway maneuvers, and low-speed freight parking scenarios.
- **Deployment of SAF functionalities across a variety of test environments:** virtual testing (e.g., CARLA, Simcenter Prescan, IPG CarMaker), proving ground trials (e.g., at Aldenhoven and Renault Valladolid sites), and hybrid setups integrating V2X hardware and simulated networks (e.g., VEDECOM's integration of OMNeT++, SUMO, and real vehicle components).
- **Demonstrations of SAF blocks including query & concretise, allocate, execute, coverage, test evaluate, safety case and decide,** guided by consistent ODD requirements, safety metrics, and standardized data formats (e.g., ASAM OpenX).

Key findings from the Use Case demonstration activities include:

- **Effectiveness of SAF's modular methodology** — even when applied in varied domains and system architectures.
- **Utility of optimization-based and machine learning methods** (e.g., surrogate modeling) to identify critical safety boundaries.
- **Robustness of SAF interfaces and workflows** for allocating scenarios efficiently between virtual and physical layers.
- **Evidence of enhanced safety transparency**, enabling reproducibility and regulatory relevance.

Through rigorous evaluation, partners assessed safety metrics, validating system behavior under nominal, edge, and failure-inducing conditions.

The outcomes of deliverable D7.3 feed directly into parallel running work packages, particularly in **refining interfaces and validation tools (WP3, WP4)** and informing **coverage metrics development (WP5)**. Documented deviations and identified challenges also guide SAF improvements and extensions.

Deliverable D7.3 successfully demonstrates the **end-to-end capabilities of the SUNRISE SAF**, offering concrete evidence of its applicability across varying CCAM systems. The comprehensive documentation of SAF block demonstrations significantly advances the standardization and adoption pathway for safety assurance in automated driving systems.

Multimedia documentation and demonstrator videos further support dissemination and stakeholder engagement via the Use Cases section of the SUNRISE Handbook (<https://ccam-sunrise-project.eu/use-cases/>).

1 INTRODUCTION

1.1 Project introduction

Safety assurance of Connected, Cooperative, and Automated Mobility (CCAM) systems is a crucial factor for their successful adoption in society, yet it remains a significant challenge. CCAM systems need to demonstrate reliability in all driving scenarios, requiring robust safety argumentation. It is acknowledged that for higher levels of automation, the validation of these systems by means of real test-drives would be infeasible. In consequence, a carefully designed mixture of physical and virtual testing has emerged as a promising approach, with the virtual part bearing more significant weight for cost efficiency reasons.

Worldwide, several initiatives have started to develop test and assessment methods for Automated Driving (AD) functions. These initiatives already transitioned from conventional validation to a scenario-based approach and combine different test instances (physical and virtual testing) to avoid the million-mile issue.

The initiatives mentioned above, provide new approaches to CCAM validation, and many expert groups formed by different stakeholders, are already working on CCAM systems' testing and quality assurance. Nevertheless, the lack of a common European validation framework and homogeneity regarding validation procedures to ensure safety of these complex systems, hampers the safe and large-scale deployment of CCAM solutions. In this landscape, the role of standards is paramount in establishing common ground and providing technical guidance. However, standardising the entire pipeline of CCAM validation and assurance is in its infancy, as many of the standards are under development or have been very recently published and still need time to be synchronised and established as common practice.

Scenario Databases (SCDBs) are another issue tackled by several initiatives and projects, that generally tends to silo solutions. A clear concrete approach should be used (at least at European level), dealing with scenarios of any possible variations, including the creation, editing, parameterisation, storing, exporting, importing, etc. in a universally agreed manner.

Furthermore, validation methods and testing procedures still lack appropriate safety assessment criteria to build a robust safety case. These must be set and be valid for the whole parameter space of scenarios. Another level of complexity is added, due to regional differences in traffic rules, signs, actors and situations.

Evolving from the achievements obtained in HEADSTART and taking other project initiatives as a baseline, it becomes necessary to move to the next level in the development and demonstration of a commonly accepted **SAF** for the safety validation of CCAM systems, including a broad portfolio of Use Cases (UCs) and comprehensive test and validation tools. This will be done in **SUNRISE**, which stands for **S**afety ass**U**ra**N**ce f**R**amework for connected, automated mobility **S**yst**E**ms.

The SAF is the main product of the SUNRISE project. As the following figure (Figure 1) indicates, it takes a central role, fulfilling the needs of different automotive stakeholders that all have their own interests in using it.

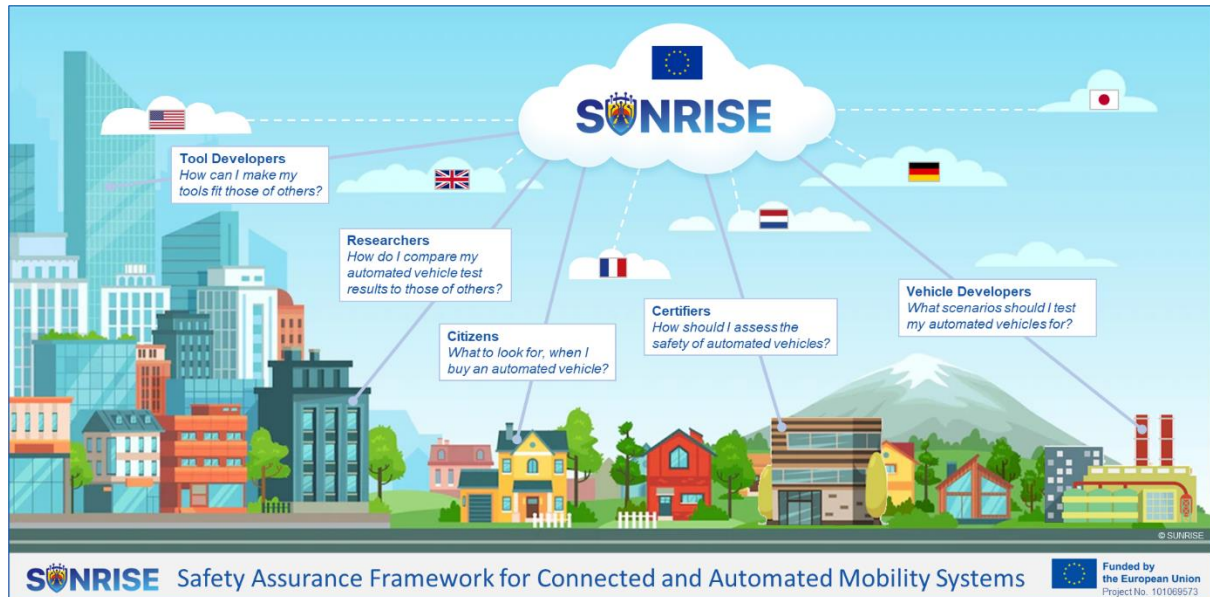


Figure 1: Safety Assurance Framework stakeholders

The **overall objective** of the SUNRISE project is to accelerate the safe deployment of innovative CCAM technologies and systems for passengers and goods by creating demonstrable and positive impact towards safety, specifically the EU's long-term goal of moving close to zero fatalities and serious injuries by 2050 (Vision Zero), and the resilience of (road) transport systems. The project aims to achieve this objective by providing a SAF consisting of three main components: a Method, a Toolchain and a Data Framework. The **Method** is established to support the SAF safety argumentation, and includes procedures for scenario selection, sub-space creation, dynamic allocation to test instances and a variety of metrics and rating procedures. The **Toolchain** contains a set of tools for safety assessment of CCAM systems, including approaches for virtual, hybrid and physical testing. The **Data Framework** provides online access, connection and harmonization of external Scenario Databases (SCDBs), allowing its users to perform query-based extraction of safety relevant scenarios, allocation of selected scenarios to a variety of test environments, and generation of the test results. The SAF will be put to the test by a series of **Use Cases demonstrations**, designed to identify and solve possible errors, gaps and improvements to the underlying methods, tools and data.

Following a common approach will be crucial for present and future activities regarding the testing and validation of CCAM systems, allowing to obtain results in a standardised way, to improve analysis and comparability, hence maximising the societal impact of the introduction of CCAM systems.

The following figure (Figure 2) shows the general workplan of the SUNRISE project.

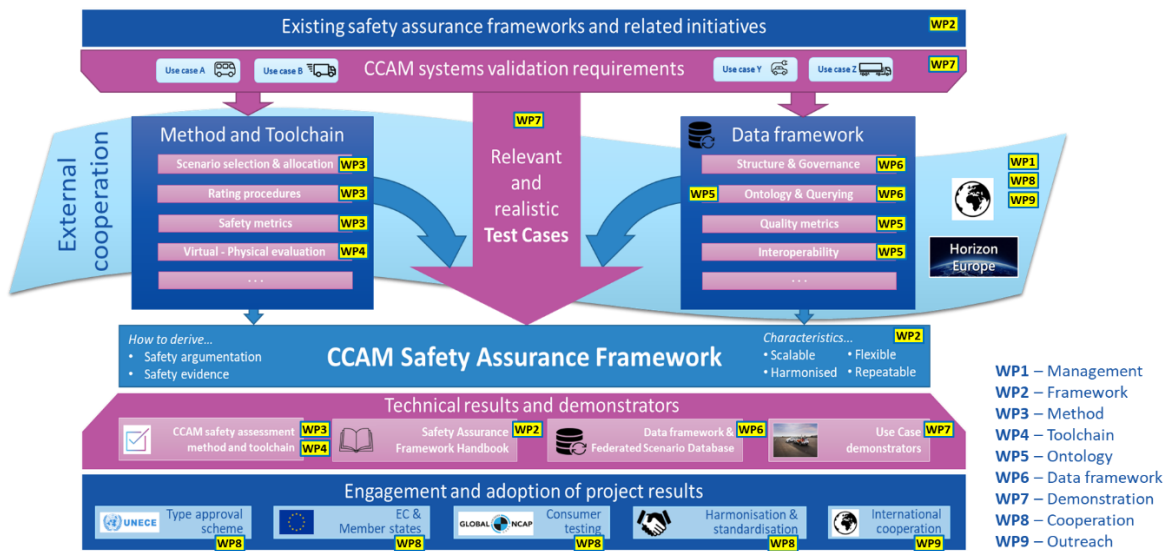


Figure 2: Workplan of the SUNRISE Project

1.2 Purpose of deliverable

The purpose of deliverable D7.3, “**Safety Assurance Framework Demonstration Instances**”, is to **implement, demonstrate, and validate** the **SUNRISE SAF** through a structured set of representative **CCAM use cases** across different **operational domains**. This deliverable operationalizes and experimentally evaluates individual SAF components—namely, Scenario, Environment, and Safety Argument blocks—under realistic conditions and across both virtual and physical test environments.

Deliverable D7.3 directly supports **SUNRISE project objective 1**, about the development of a **robust and harmonized methodology** for the safety assurance of **Connected, Cooperative, and Automated Mobility (CCAM)** systems. It addresses the complexity of assessing the safety of automated driving systems (ADS), particularly for higher levels of automation, where traditional on-road validation is no longer feasible or sufficient. By showcasing the SAF’s capability to support **scenario-based safety validation, allocation and execution of tests**, and evidence-based argumentation, deliverable D7.3 complements the overarching ambition of enabling scalable, traceable, and repeatable safety assessment of CCAM systems in line with European regulatory and industrial needs and expectations.

SUNRISE Project objective 7—“Demonstrate the framework in a representative set of use cases to prove the robustness, repeatability and versatility of the methodology through a rich simulation kits portfolio”—**is directly addressed by this deliverable, D7.3**, through the comprehensive selection, implementation, and evaluation of **eight diverse** Connected, Cooperative, and Automated Mobility (CCAM) **use cases** across **urban, highway, and freight domains**. Each use case operationalizes the SUNRISE Safety Assurance Framework (SAF) within multiple testing environments—including **virtual, hybrid, and physical setups**—and employs a wide array of simulation tools (e.g., CARLA, Simcenter Prescan, IPG

CarMaker, OMNeT++/SUMO, MATLAB/Simulink), proving the framework's adaptability and performance under varying conditions.

The expected impact of this work is multifaceted. First, this deliverable provides evidence of SAF's interoperability across diverse technology stacks, stakeholders, and application contexts. Second, it demonstrates how various **data-driven, knowledge-based, and optimization-based methods** can concretely be used for **scenario derivation** and **test case allocation**. Third, it outlines SAF's capacity to guide **hybrid testing strategies**, support **simulation-based edge case evaluation**, and structure argumentation for **safety case development**.

Several innovative aspects characterize this deliverable. Among them are the first-time **demonstration of modular SAF block implementations in a federated validation context**; the application of surrogate modeling and **hybrid query-concretization** mechanisms; the use of **standard ontologies** (e.g., **OpenLABEL**) and scenario description formats (e.g., **ASAM OpenX**); and the implementation of adaptive test environment allocation methods. Moreover, the bridging of virtual and hybrid testing environments (e.g., via OMNeT++/SUMO integrated with real RSUs and automated vehicles) elevates the level of realism and operational relevance of the demonstrations.

In terms of **scope**, the deliverable includes **eight fully implemented use cases** (UC1.1–UC4.2), targeting diverse **CCAM scenarios**, such as **urban perception, connected intersections, cooperative traffic jam and highway pilot functions, and automated freight parking**. Each use case covers a specific subset of the SAF blocks depending on partner capabilities and domain focus. However, regulatory compliance assessments (audits) and in-service monitoring are considered out of scope for this specific deliverable, although the SAF addresses these matters to a certain extent.

The results of task T7.3 (described in this deliverable) are intended to be directly embedded in **methodological (WP3), toolchain-related (WP4), and coverage analysis (WP5)** developments of SUNRISE. It provides practical insights for project partners to refine SAF components and their interfaces. Furthermore, the scenarios, validation findings, and methodological lessons offer a knowledge base for external stakeholders—such as **CCAM developers**, safety assessors, and regulators—seeking to integrate scenario-based safety assurance into their workflows. The **publicly accessible videos and documentation** provided via the **SUNRISE Handbook** serve as additional material for external application and validation.

1.3 Intended audience

The deliverable serves multiple stakeholders, including the SUNRISE project itself, as it presents the execution of demonstration instances of specified use cases (UCs) that demonstrate and validate the components, functionalities, and workflow of the SUNRISE Safety Assurance Framework (SAF). It is also of interest to all users of the SUNRISE SAF who seek deeper insight into the SAF's validation activities, with descriptions of which SAF blocks are validated by the individual use cases. Additionally, the intended audience explicitly includes the general public. This inclusion is aligned with the description of task T7.2 in the

Grant Agreement, which emphasizes the need to "make visible the novelties and technical outcomes of the project and enhance public awareness and acceptance for CCAM Validation and Verification (V&V)." Thus, the deliverable aims to communicate the project's advancements comprehensively to both technical stakeholders and the broader public to foster understanding and acceptance of CCAM safety assurance.

1.4 Deliverable structure and relation to other parts of project

This deliverable is structured to provide a thorough and systematic account of the demonstration and validation activities for the SUNRISE SAF. The document is organized into several main chapters that guide the reader from the broader context and objectives to technical details of SAF validation, specific use case demonstrations, key findings, and future implications.

The introductory chapter situates the deliverable within the overall SUNRISE project, outlining its objectives, intended audience, and, specifically in this section, how the structure of the document supports its dual role as both a technical report and a cross-work package integrator. This introduction ensures that readers of varying backgrounds—internal project partners and external stakeholders—can navigate the technical content with a clear understanding of its scope and relevance.

Chapter 2 details the **Validation of the Safety Assurance Framework**. It introduces the SAF's architecture and describes each main block, such as scenario creation, environment setup, and safety argumentation. Each of these blocks are mapped to defined validation criteria, and dependencies on interfaces, tooling (e.g., ASAM OpenX standards), and methodologies developed in other work packages are clearly stated. The logical progression in this section allows the reader to see how each SAF block functions independently *and* as part of the integrated framework, laying the groundwork for their application in practical demonstrations.

Chapter 3: Use Case Demonstrations presents eight use cases systematically, categorized by domain: urban, highway, and freight. For each use case, consistent subsections cover the overview, scenario definitions, technical implementation, SAF block validation, key takeaways, and deviations from originally planned activities. This uniform structure allows for direct comparison across use cases, highlighting not only coverage and methodological breadth but also the practical feedback integrated into ongoing SAF development. The demonstration activities draw heavily on inputs from earlier deliverables, such as scenario design, allocation algorithms, and toolchain integration, particularly outputs from deliverable D3.4, deliverable D3.5, deliverable D4.4, deliverable D4.5, deliverable D4.6, deliverable D5.2 and deliverable D5.3.

After the use cases, the deliverable presents the **Conclusions**, where overarching insights, limitations, and recommendations for further refinement of the SAF are summarized. This final section explicitly connects the demonstrator findings to recommended modifications to specific SAF blocks or other project results. It summarizes how the results directly informed the refinement of task activities in WP3, WP4, WP5 and WP6, as well as how feedback processes facilitated an iterative improvement cycle: a characteristic of the SUNRISE approach.

Throughout the deliverable, specific dependencies and inputs from prior project tasks and deliverables are systematically referenced—such as scenario allocation methodologies (from deliverable D3.3), test case management frameworks (from deliverables D4.1 and D4.3), and approaches for coverage analysis (from deliverables D5.3). In turn, the results and lessons from the present deliverable constitute a central knowledge base and feedback resource for future milestones, particularly for the enhancement of SAF interfaces, tool interoperability, and regulatory alignment. The established feedback loops, coordinated at both Work Package and Task levels, are documented to ensure that insights can be rapidly propagated to other domains of the project, thereby enhancing the responsiveness and robustness of SAF development.

In summary, the document is logically structured to facilitate traceability from project-wide objectives, through SAF design and validation, to applied demonstrations and their implications for overall project progress. It acts as an integrative nexus between multiple SUNRISE work packages (primarily WP3, WP4, WP5, WP6 and WP7), both consuming and producing significant methodological and empirical outputs that shape the development of the SUNRISE Safety Assurance Framework.

1.5 Differences deliverable D7.2 and D7.3

The main difference between deliverable D7.2 and D7.3 is that deliverable D7.2 is about planning and design, while D7.3 is about execution and delivery of working Proof-of-Concepts (PoCs). Further details about this difference are as follows:

- **Scope Expansion and Demonstration Depth:** Deliverable D7.3 significantly expands practical demonstration activities. While deliverable D7.2 focused on the planning, structure, and initial setup of SAF validation, deliverable D7.3 provides extensive documentation of test execution, results, and insights across all eight use cases (urban, highway, and freight). Deliverable D7.3 contains in-depth scenario definitions, test environment details, execution logs, evaluation metrics, and specific partner contributions, many of which were briefly planned or theorized in deliverable D7.2.
- **Explicit Documentation of Deviations:** For each use case, deliverable D7.3 provides a section called “Deviations from D7.2” section, reporting changes or modifications to the original plans. Examples include:
 - Some SAF blocks (e.g., Query & Concretise, Allocation) were not fully validated in real-world (proving ground) contexts because tests were limited to available scenarios and physical conditions, rather than strict allocations as foreseen in D7.2.
 - Certain simulation or sensor technologies evolved, leading to methodological updates (e.g., adoption of AI-enhanced rendering over planned static photorealistic rendering in UC1.1).

- Planned weather condition tests in proving ground environments could not be executed due to lack of prototype readiness or climatic constraints.
- **Methodological and Technical Refinements:** Deliverable D7.3 incorporates experience from preliminary validation and makes explicit reference to improved toolchains (e.g., scenario concretization methods, surrogate modeling, and hybrid testing frameworks) that were only described conceptually in deliverable D7.2. In deliverable D7.3, the processes and criteria for each SAF block (Scenario, Environment, Safety Argument) are tested, discussed, and, where relevant, altered based on practical feasibility and results.
- **Expanded Evaluation and Metrics:** Evaluation is considerably more quantitative and use-case specific. Deliverable D7.3 gives actual test outputs, comparison of proposed vs. implemented scenario sampling techniques, and traces the effectiveness of newly introduced surrogate modeling approaches or scenario allocation procedures.
- **Feedback Loop and Iterative Refinement:** The “Deliverable structure and relation to other parts of project” and “Feedback loop” sections in D7.3 describe more intense feedback processes between WP7 (demonstration) and other work packages (especially WP3, WP4, and WP5), demonstrating for the first time how insights and lessons from hands-on tests are already shaping refinements in current and future development cycles.
- **Documentation and Handbooks:** D7.3 newly introduces or expands references to multimedia documentation and public-facing knowledge-transfer materials (e.g., Handbook videos corresponding to each use case demonstration).
- **Clarification of Limitations and Future Work:** Unlike D7.2, D7.3 also emphasizes what could not be achieved yet or where further validation is required. This includes pending interface validation in mixed virtual/physical environments, incomplete coverage due to time/resource/weather constraints, and items deferred to upcoming deliverables.

2 VALIDATION OF THE SAFETY ASSURANCE FRAMEWORK

2.1 Overview of the SAF

The SUNRISE SAF is a harmonized, scalable system of methods, tools, and data for assessing the safety of CCAM systems. The framework is based on a multi-pillar approach, inspired by the UNECE New Assessment/Test Method for Automated Driving (NATM), and integrates Audit, Safety Performance Assurance, In-Service Monitoring and Reporting (ISMR), as well as the Input Layer as central elements. The goal is to ensure a comprehensive, end-to-end safety assessment from development through validation to in-service monitoring.

Figure 3 illustrates the SUNRISE SAF and its core components for the safety assurance of CCAM systems. The diagram visually represents the workflow and interactions among the main building blocks.

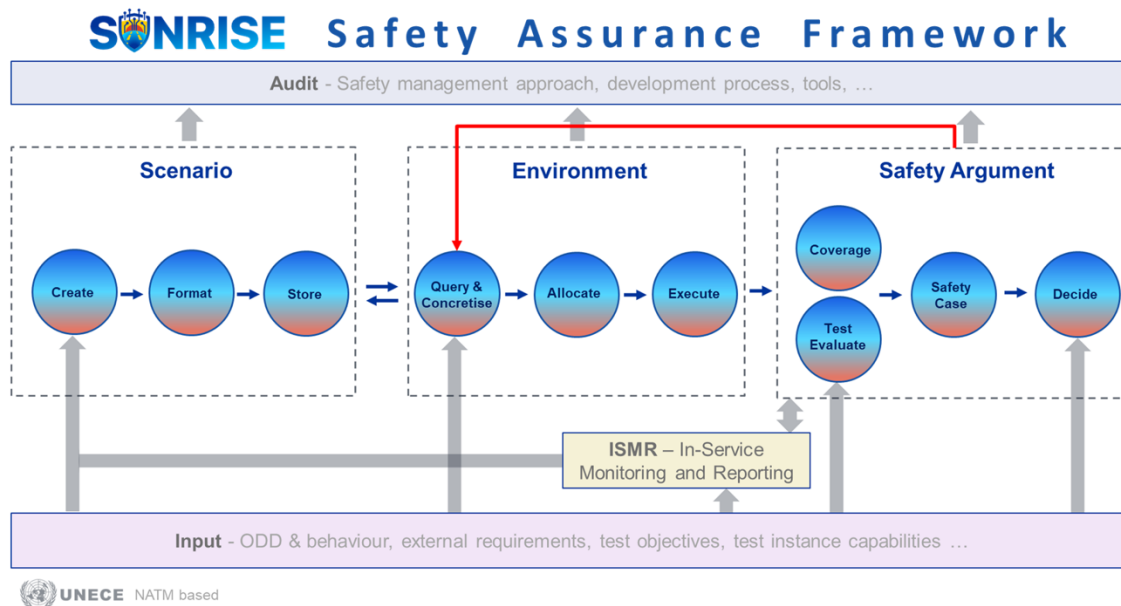


Figure 3: Overview of the SAF

Further details on the individual SAF blocks can be found in deliverable D2.3 [32].

2.2 Criteria for validation of the SAF

The SAF validation focuses on confirming that each block within the framework fulfils its intended role for safety assurance of CCAM systems. Validation is conducted through diverse, representative use cases, covering the full process from scenario generation to safety argumentation and decision-making. The following sections detail the validation criteria for each relevant block and interface of the SAF.

2.2.1 SUNRISE DF

The SUNRISE Data Framework (DF) serves as the federated interface, ensuring that scenario databases (SCDBs) can be accessed and harmonized for querying and concretization. The criteria for validating this framework include:

- **Interoperability:** Scenarios must be formatted according to agreed standards (e.g., ASAM OpenX formats), allowing seamless querying across federated databases.
- **Accessibility:** All connected SCDBs are required to support standardized query mechanisms from the DF, ensuring scenarios can be retrieved based on test objectives and requirements.
- **Data Integrity and Traceability:** The DF must reliably return accurate scenarios while maintaining records for provenance and changes.
- **Compliance:** SCDBs and the DF interface are checked against SUNRISE's defined requirements for scenario content, parameterization, and access rights.

2.2.2 Query & Concretise

This block is responsible for transforming high-level test requirements into executable scenarios. Validation criteria include:

- **Input Completeness:** Proper integration and use of input data such as Operational Design Domain (ODD), system requirements, behaviors, and external regulations.
- **Query Validity:** All queries to scenario storage must be well-formulated, drawing on a harmonized ontology (e.g., OpenLABEL) to ensure common understanding.
- **Scenario Quality:** Logical scenarios (with parameter ranges) must be correctly concretized into executable forms, incorporating relevant test objectives.
- **Auditability:** Validation includes the ability to review the process of concretization—audit steps must check that ODD descriptions, requirements, and variable assignments are all present and compliant with standards.
- **Feedback Integration:** The block must incorporate feedback from test analysis, enabling refinement of parameter combinations and ensuring coverage of edge cases.

2.2.3 Allocate

The allocation process assigns concretized scenarios to suitable test environments (simulation, Hardware-in-the-Loop, proving ground). Validation criteria are:

- **Matching Accuracy:** Test scenarios must be allocated according to the capabilities and constraints of the test environment, based on clear criteria such as required fidelity, safety needs, and resource availability.
- **Efficiency:** The allocation process should prioritize virtual (lower complexity) environments before moving to real-world testing and clearly document external influences and overrides.
- **Traceability:** Each allocation action must be fully documented to ensure transparency and reproducibility.

2.2.4 Execute

The execution block oversees running allocated scenarios in their respective environments. Validation criteria include:

- **Capability Assurance:** Confirmation that the selected test instance can execute the given test scenario, and that both simulation and physical test setups meet SUNRISE's requirements for reproducibility and reliability.
- **Data Collection:** Execution must ensure the capture of all relevant data for post-test analysis.
- **Completion & Quality:** All steps of the test execution process must be finished correctly, and the outcome must feed meaningfully into the evaluation process.
- **Feedback Loops:** The mechanism must enable necessary adaptations or adjustments in scenarios or objectives based on intermediate findings.

2.2.5 Coverage

Coverage assessment ensures that the scenario space tested is representative and sufficient. Criteria include:

- **Diversity:** The selected scenarios must cover a wide range of ODDs, behaviors, and environmental conditions.
- **Completeness:** Coverage analysis must check for gaps, redundancies, and sufficient representation of critical and failure scenarios.
- **Metrics & Indices:** Use of quantitative measures (e.g., Scenario Relatedness Index) to monitor and report scenario diversity and completeness.

2.2.6 Test Evaluate

The Test Evaluate block determines whether safety and performance criteria have been achieved based on test data. Validation includes:

- **Metric-Based Assessment:** Each test execution is assessed against clearly defined pass/fail criteria derived from system requirements and ODDs.
- **Traceable Evidence:** All evaluations must be traceable to the executed scenarios and underlying requirements.
- **Statistical Rigor:** Where relevant, statistical guarantees of safety must be established, quantifying any residual risk.

2.2.7 Safety case

The Safety Case is the final, integrated body of evidence and reasoned safety argumentation underpinning the SUNRISE SAF process. Its validation is central to regulatory acceptance and stakeholder confidence. Criteria include:

- **Structured, Modular Argumentation:** The Safety Case must present logically organized, well-structured arguments linking requirements, scenarios, test results, and safety criteria.

- **Traceability & Transparency:** All claims within the Safety Case are fully referenced to specific evidence—requirements, scenarios, tests, and evaluation outcomes—accessible for regulators and stakeholders.
- **Evidence Sufficiency & Rigor:** The Safety Case demonstrates sufficient evidence for all relevant ODDs, failure conditions, and hazards, presenting clear justifications for why the system is considered “safe enough.”
- **Auditability & Change Management:** All Safety Case elements are maintained under version control and are accessible for inspection, supporting audits and updates as technology, regulations, or operational experience evolves.
- **Stakeholder Acceptance:** The Safety Case must be constructed for clarity and completeness to ensure that it is understandable not only to technical reviewers but also to regulatory authorities and public stakeholders where required.
- **Continuous Updating:** Mechanisms must be in place to integrate new evidence from in-service monitoring (ISMR), incident analysis, or technology evolution, thus maintaining safety case validity throughout the CCAM system’s lifecycle.

2.2.8 Decide

The Decide function uses all available evidence to make an informed, justifiable approval or rejection decision:

- **Compliance Confirmation:** Decision-making is based on system compliance with UNECE, EC, and other relevant regulations and standards.
- **Evidence-Based Transparency:** All inputs, analyses, and rationales for decisions must be well-documented and accessible for audits or regulatory inspection.
- **Stakeholder Participation:** External influences, regulatory requirements, and stakeholder feedback are all incorporated before final decisions are made.

2.3 Covered SAF blocks by use cases

The table below gives an overview of which of the SAF blocks will be covered by each use case. Details on the validation of the SAF blocks of each Use Case are described in chapter 3. Within the scope of this deliverable, none of the use cases performed scenario extraction from federated scenario databases or exercised the dedicated "Scenario" architecture functionalities as described in the SAF. Instead, all logical and concrete scenarios for the demonstrations were handcrafted by experts or directly defined within the simulation frameworks, bypassing the specific scenario querying, storage, or database retrieval procedures that those subblocks are intended to validate. As a result, there were no practical activities pertaining to the "Scenario" part of the SAF for reporting during the evaluation of these use cases, which is why these subblocks were excluded from the table summarizing validation activities.

Table 1: Overview of the validation activities per use case

	UC1.1	UC1.2	UC1.3	UC2.1	UC3.1	UC3.2	UC4.1	UC4.2
SUNRISE DF				x				
Query & Concretise	x	x	x	x	x	x	x	x
Allocate	x	x	x	x	x	x	x	
Execute	x	x	x	x	x	x	x	x
Test Evaluate	x	x	x	x	x	x	x	x
Coverage		x	x		x	x		
Safety Case				x		x	x	
Decide		x	x	x	x	x	x	


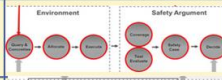

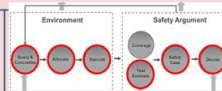
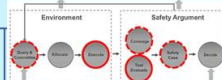



2.4 Feedback loop

The exchange of feedback took place specifically at the Work Package (WP) and Task (T) level, ensuring that the feedback from the use case demonstrations could be clearly assigned to the responsible development teams. In particular, the insights gained from WP7 T7.3, based on the eight demonstrated use cases, were directly incorporated into the relevant development work packages and considered in the further development of the SAF. Ongoing coordination was ensured through regular reviews of deliverables and constant communication between all involved teams. Furthermore, the responsible WP and Task Leads from the work packages and tasks relevant to the individual SAF blocks, were directly involved in WP7 and Task 7.3, facilitating effective knowledge transfer and seamless feedback. This cooperative approach made it possible to specifically incorporate practical experiences and challenges from the use cases into the validation and further development of the SAF.

3 USE CASE DEMONSTRATION

This chapter presents the execution, demonstration, and validation of the eight sub use cases, summarized in Table 2, highlighting the practical application of the SUNRISE SAF blocks.

Table 2: Overview of the SUNRISE use cases

UC Nr.	Name	SAF coverage	Environment	Partners
UC1 Urban AD				
UC1.1	Perception testing		PG virtual	CAF, IFAG, RSA, CVC, VIF, RESA, SISW
UC1.2	Connected perception testing		hybrid virtual	VED, UNITN, ika
UC1.3	Cooperative perception testing		hybrid virtual	ICCS, VED
UC2 Traffic jam AD				
UC2.1	Safety assessment & decision making		PR PG virtual	AVL, AVL TR, CAF, AVL HU, TNO, (UoW), BAST
UC3 Highway AD				
UC3.1	Map based perception & decision making		PG virtual	IDI, IDI DE, UNITN, UoW, ika
UC3.2	Cooperative perception & decision making & control		PG virtual	IDI, IDI DE, CRF, UNITN, UoW, ika
UC4 Freight vehicle automated parking				
UC4.1	Truck low-speed perception & decision making		PG virtual	RISE, CHAL
UC4.2	Truck low-speed connected perception cyber-security		virtual	ICCS, RISE

Simulations are conducted across various domains including:

- PG: Proving grounds for controlled and repeatable physical testing
- PR: Public roads for evaluating system behavior in real-world traffic and environmental conditions
- Hybrid testing environments that integrate physical hardware with virtual models for co-simulation
- Fully virtual testing platforms where scenarios are simulated entirely in software to enable safe, cost-effective, and early-stage validation

3.1 Use case 1.1: Urban AD validation – Perception testing

3.1.1 Use Case Overview

In Use Case 1.1, the SUNRISE SAF is tested for perception systems in urban environments. The aim here is to test these systems in complex urban intersections and scenarios, as well

as in adverse weather conditions. There are camera-based, LiDAR-based and radar-based perception systems.

There are six behaviours defined for UC1.1: car-following, outdoor parking with other vehicles entering or leaving parking spots, obstacle avoidance due to construction work, pedestrian and bicycle interactions, and mainly urban intersections, such as roundabouts and right/left turn junctions. Some of them with day/night conditions and/or adverse weather conditions like fog, rain and others.

Objective:

Enhance the existing methodologies for testing and validating CCAM functions in urban settings, with particular emphasis on intersections, which are high-risk areas for accidents, primarily due to distracted pedestrians and traffic signal violations.

3.1.1.1 Covered aspects of the SAF

Table 3 below shows the contributions to the SAF block validation per partner for UC1.1 and summarized in Figure 4.

Table 3: Contributions to the SAF validation per Partner in UC1.1

Partner/ Block	CVC	IFAG	RESA/RSA	SISW	VIF
SUNRISE DF					
Query & Concretise		x		x	x
Allocate	x	x			x
Execute	x	x	x	x	x
Test Evaluate	x	x	x		
Coverage					
Safety Case					
Decide					

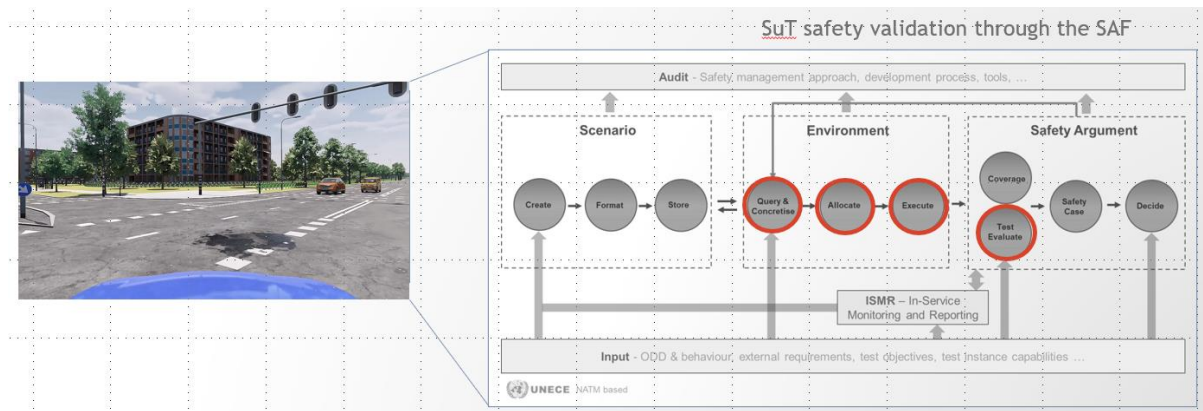


Figure 4: SAF demonstrated blocks in this UC's demo (s).

UC1.1 <https://www.youtube.com/watch?v=sZ45NUitwow&t=18s>

3.1.1.2 Safety case setup

This use case illustrates the safety test case by building safety evidence through microscopic evaluation on individual scenarios using binary pass-fail criteria, detection recall and precision, or quality of the perception system's tracking. For the safety argumentation, test scenario allocation, Co-simulation framework execution, Coverage-oriented testing including SOTIF analysis, Risk-oriented testing including high-risk and boundary scenarios identification and knowledge-based scenarios are considered.

3.1.2 Overview of tested scenarios

Several partners provided their own simulation frameworks that focused on different parts of the urban perception use case. Therefore, different urban scenarios were considered depending on the partner.

CVC was responsible for conducting virtual testing of camera-based perception systems. These tests were carried out using the CARLA driving simulator and a collection of photorealistic images. The scenarios used in CARLA were provided in OpenScenario 2.0 format, encompassing various urban scenarios such as roundabouts, object avoidance, and turns. One of these scenarios, where a pedestrian is crossing the street behind parked cars, can be seen in Figure 5.



Figure 5: A Pedestrian crossing the street behind a parked car

IFAG focused on three distinct Euro NCAP scenarios: (i) Car-to-Pedestrian Farside Adult (see Figure 6, (ii) Car-to-Pedestrian Nearside Adult (see Figure 7), and (iii) Car-to-Pedestrian Turning Adult 4 (see Figure 8), each of which presents unique challenges and opportunities for validation in urban environments. By concentrating on these scenarios, the aim was to gain a deeper understanding of how ADS-equipped vehicles interact with vulnerable road users, such as pedestrians, in a variety of situations.

Furthermore, IFAG has set up the demonstration instance as virtual testing using two different versions, distinguished by the fidelity of the sensor model, to showcase the differences arising from these two variants. Eventually, this highlights the importance of adequate perception system modelling, especially in urban environments.

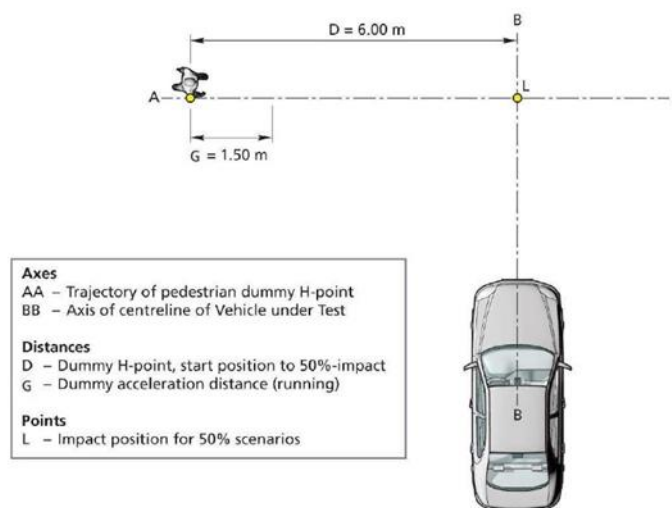


Figure 6: Car-to-Pedestrian Farside Adult [1]

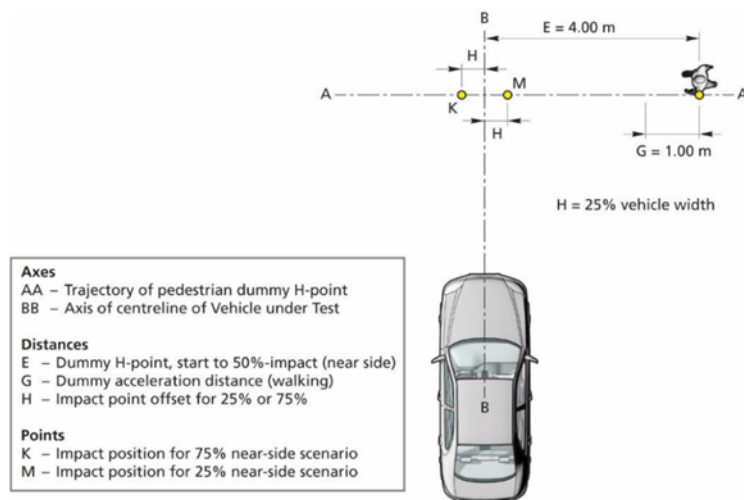


Figure 7: Car-to-Pedestrian Nearside Adult [1]

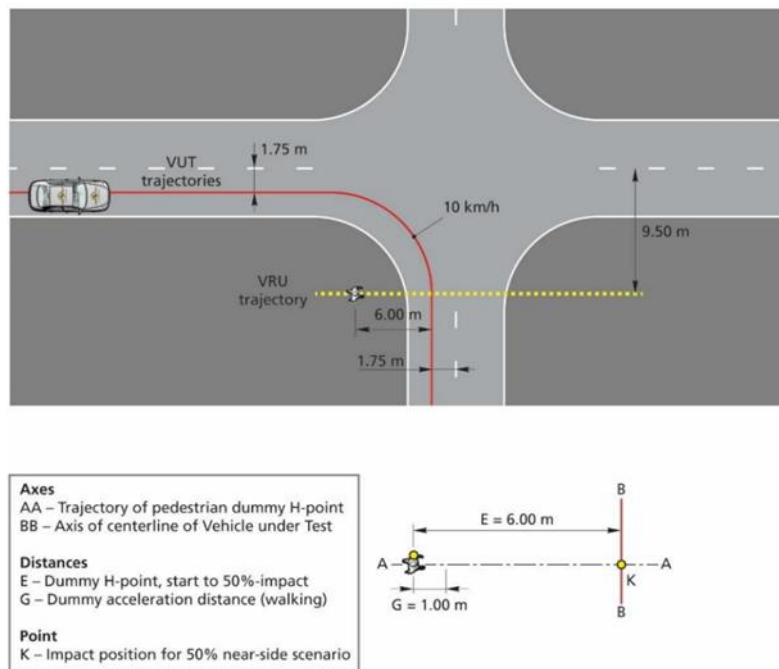


Figure 8: Car-to-Pedestrian Turning Adult [1]

Virtual Vehicle (VIF) focused on left turn and right turn junction scenarios (see Figure 9).

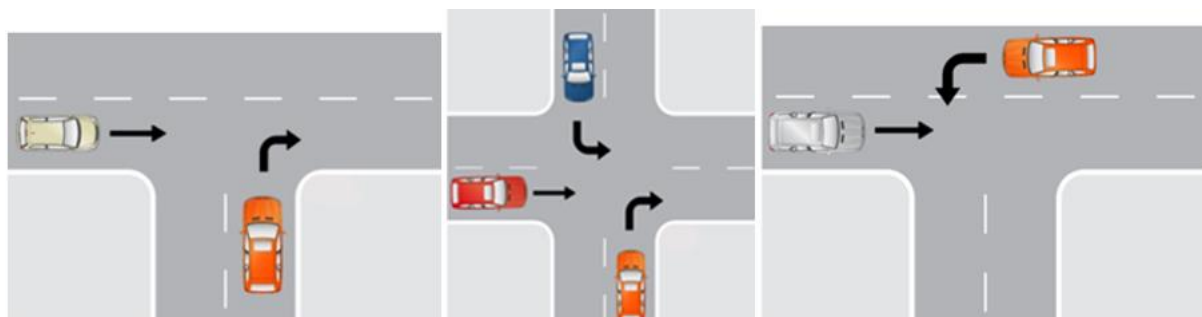


Figure 9: Right/Left turn junction scenarios

RESA/RSA is providing an autonomous driving prototype equipped with a LiDAR-based perception system. A proving ground mimicking an urban area with outdoor parking and junctions is in Renault's facilities in Valladolid (Spain). Several scenarios are considered. Figure 10 shows a scenario where a parked car opens its door with oncoming traffic. Figure 11 also shows an outdoor parking scenario with another vehicle leaving a parking spot. Figure 12 shows a car following scenario. Figure 5 scenario is also considered by RESA (Pedestrian crossing behind a parked vehicle).

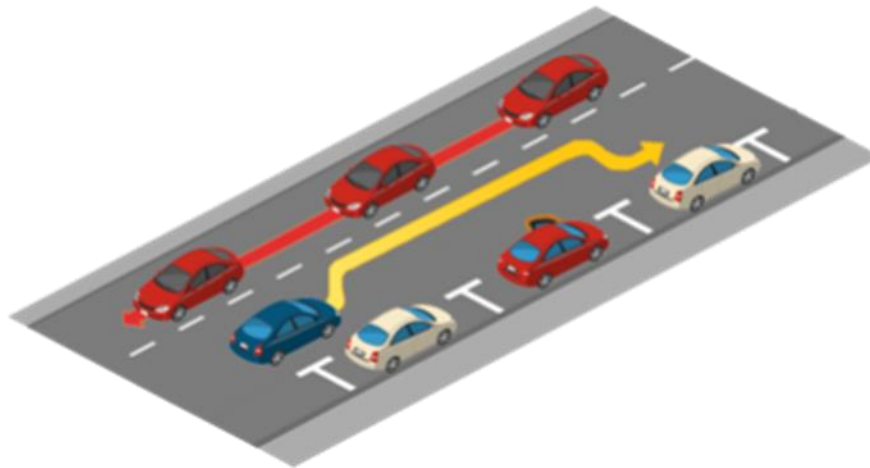


Figure 10: A parked car opens its door with oncoming traffic



Figure 11: A parked car is leaving a parking spot



Figure 12: Car following

SISW focused on the safety analysis of the Automated Emergency Braking (AEB) perception system in an intersection with different lighting and weather conditions using the SAF. The analysis includes execution of the system under test in known and unknown unsafe scenarios according to SOTIF standards. The test Scenario is a turning left at a signalised and non-

signalised junction, which is employed in the virtual environment of Simcenter Prescan (Figure 13).



Figure 13: Left turn scenario in different weather and lighting conditions

3.1.3 SAF Block demonstrations

This part provides a summary of the use case activities divided into the individual blocks of the Safety Assurance Framework.

3.1.3.1 SUNRISE DF

This use case did not query and extract scenarios from scenario databases; therefore, no activities in the “Scenario” part of the SAF can be reported.

3.1.3.2 Query and Concretise

IFAG:

IFAG used a knowledge-based approach for creating test cases. Concretely, Euro NCAP scenarios were created, which are based on expert knowledge and represent an important aspect to consider for supporting ADS validation. Specifically, input completeness (referring to the defined validation criteria) is ensured by having the necessary ODD elements included based on the given Euro NCAP scenario description. Furthermore, another validation criteria is utilised: scenario quality. Concretely, it showcases how the Euro NCAP scenario description (essentially logical scenarios) is correctly concretised into an executable form. This is also shown by executing these scenarios using virtual testing. Overall, this approach demonstrates the capability of handling knowledge-driven scenarios, addressing the needs of many vehicle safety bodies.

SISW:

For the safety argumentation, Siemens has considered scenarios according to the SOTIF standard. The SOTIF standard (Safety of the Intended Functionality) emphasises the importance of addressing both known and unknown unsafe scenarios.

For the known unsafe scenarios of SOTIF, Siemens first established the safety assurance framework using Siemens' Simcenter Prescan and HEEDS software in Figure 14. This framework begins with a logical scenario and, by varying scenario parameters such as speed and Operational Design Domain (ODD) parameters like weather and lighting conditions through optimisation techniques, it generates concrete scenarios which are critical that reveal

potential safety issues in Figure 15. This systematic approach ensures that known risks are thoroughly analysed and mitigated.

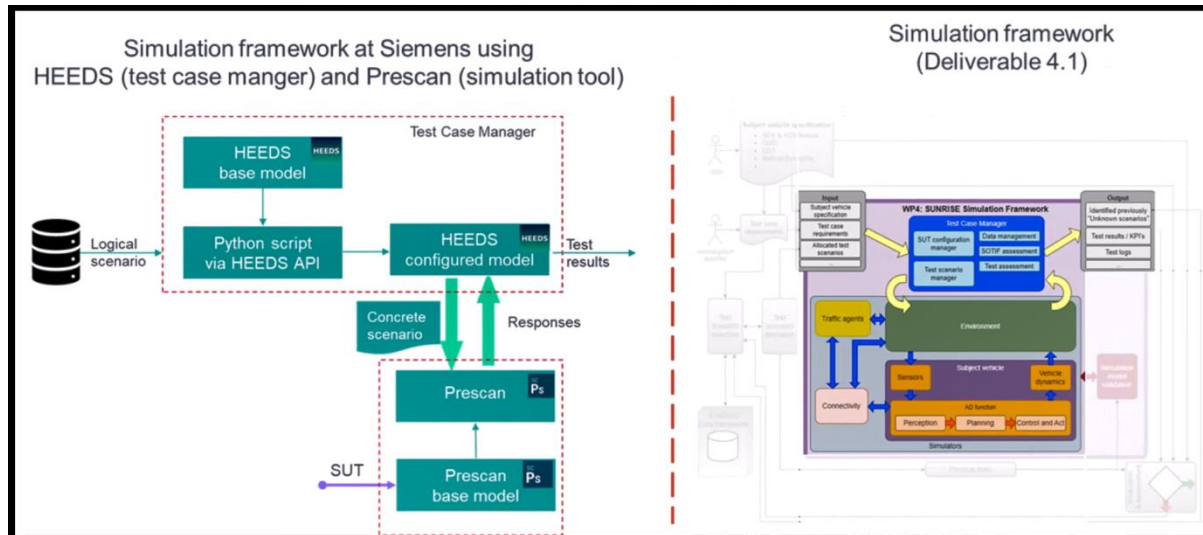


Figure 14: Simulation framework at Siemens using Simcenter HEEDS (as the test case manager) and Simcenter Prescan (as the simulation tool) (left figure), Simulation framework in D4.1 (right figure)



Figure 15: concrete scenarios with different weather and lighting condition in signalised and non-signalised intersection

VIF:

The SAF's query and concretise step instantiates concrete scenarios from logical ones by assigning feasible values to all free parameters in the scenario description. Each parameter is defined by its name, type (real-valued, integer, or from a finite set), and a permissible range, which may depend on the values of other parameters. In most practical applications, exploring

the entire feasible parameter space is computationally impractical, especially when parameters are real-valued.

Therefore, Sobol Sequences, which show good space-filling characteristics, are used to generate a representative scenario set. Since each scenario is generated independently, this method is referred to as open loop. A limitation of the open-loop approach is that it does not inherently prioritise critical or failing test cases. As a result, discovering such cases depends on the number of simulated concrete scenarios, the density of sampling in the parameter space, and an element of chance.

In contrast, the proposed closed-loop approach utilises optimisation to identify critical combinations of parameters within the feasible space. Starting from an initial guess, the simulation results are assessed using an objective function that quantitatively reflects a safety-related metric. This evaluation guides the selection of subsequent parameter combinations and steers the process toward failed test cases, based on the defined criteria. The chosen objective function is a weighted combination of minimum Time-To-Collision (TTC) and minimum distance between the ego and other actors.

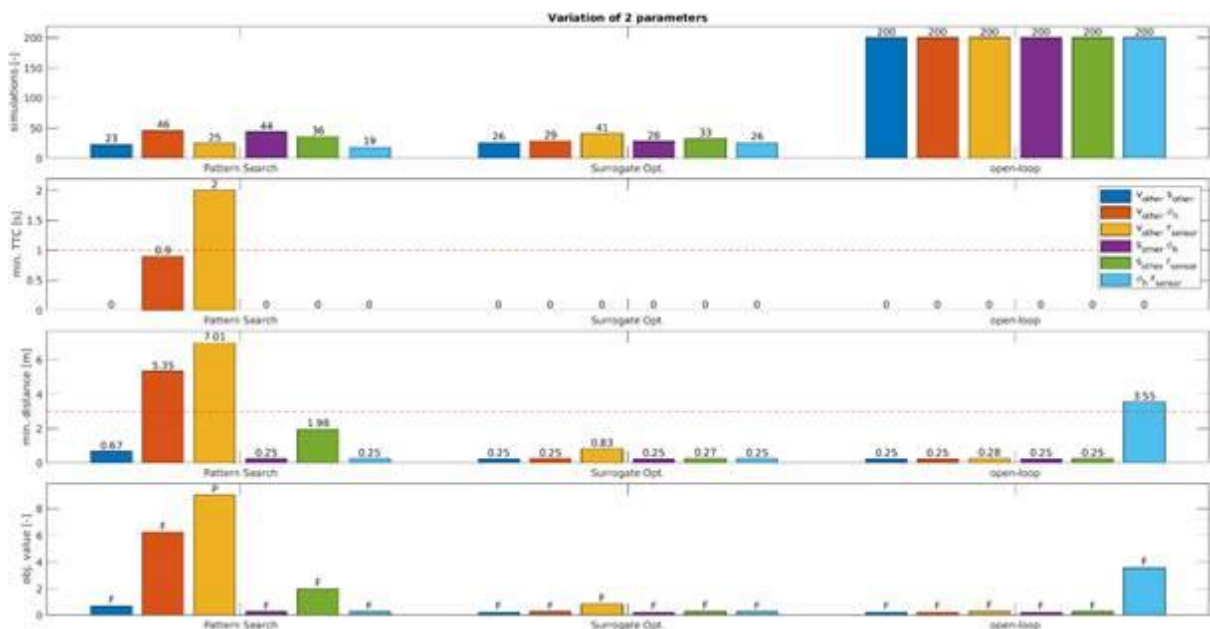
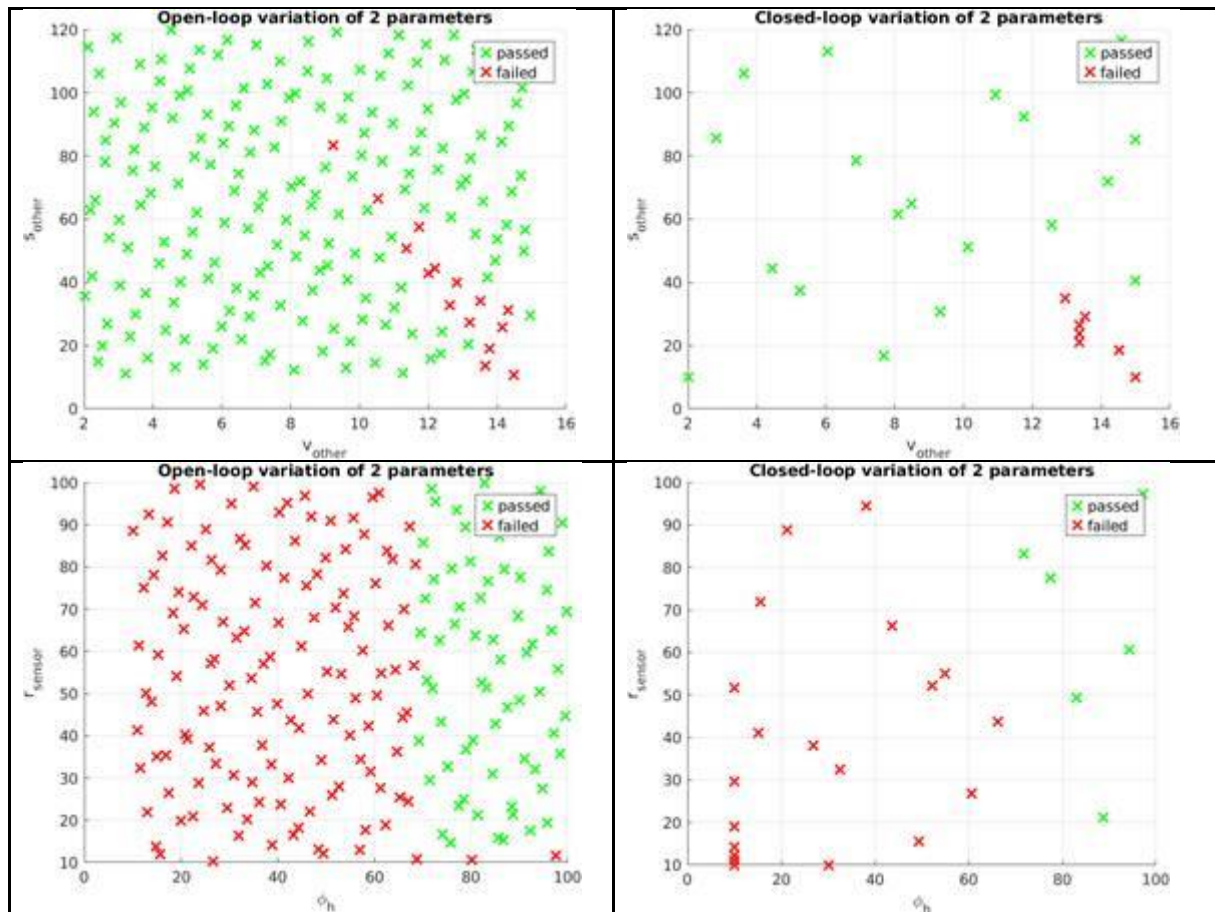
Further details on this methodology can be found in Sunrise D3.4 [2]. For all intersection scenarios considered the following parameters are varied by the open- and closed-loop methods:

- Other vehicle's starting position: S_{other}
- Other vehicle's velocity: V_{other}
- Ego vehicle sensor horizontal field-of-view: ϕ_h
- Ego vehicle sensor range: r_{sensor}

Meaningful default values are defined to facilitate doing two-, three-, and four-dimensional parameter variations. The default starting positions and velocities are selected such that the ego and other vehicles arrive at the intersection roughly at the same time.

Scenario (ego: right turn, other vehicle: from the left)

Figure 16 and Figure 18 illustrate some example concretisation results in 2D and 3D cartesian parameter space (default values are used for the remaining parameters). One can see that the closed-loop method requires fewer (time-consuming) simulations than the open-loop approach. Comparisons regarding runtimes and objective function values are illustrated in Figure 17 and Figure 19. Each colour represents one possible 2D or 3D parameter combination, which is varied by the respective method.



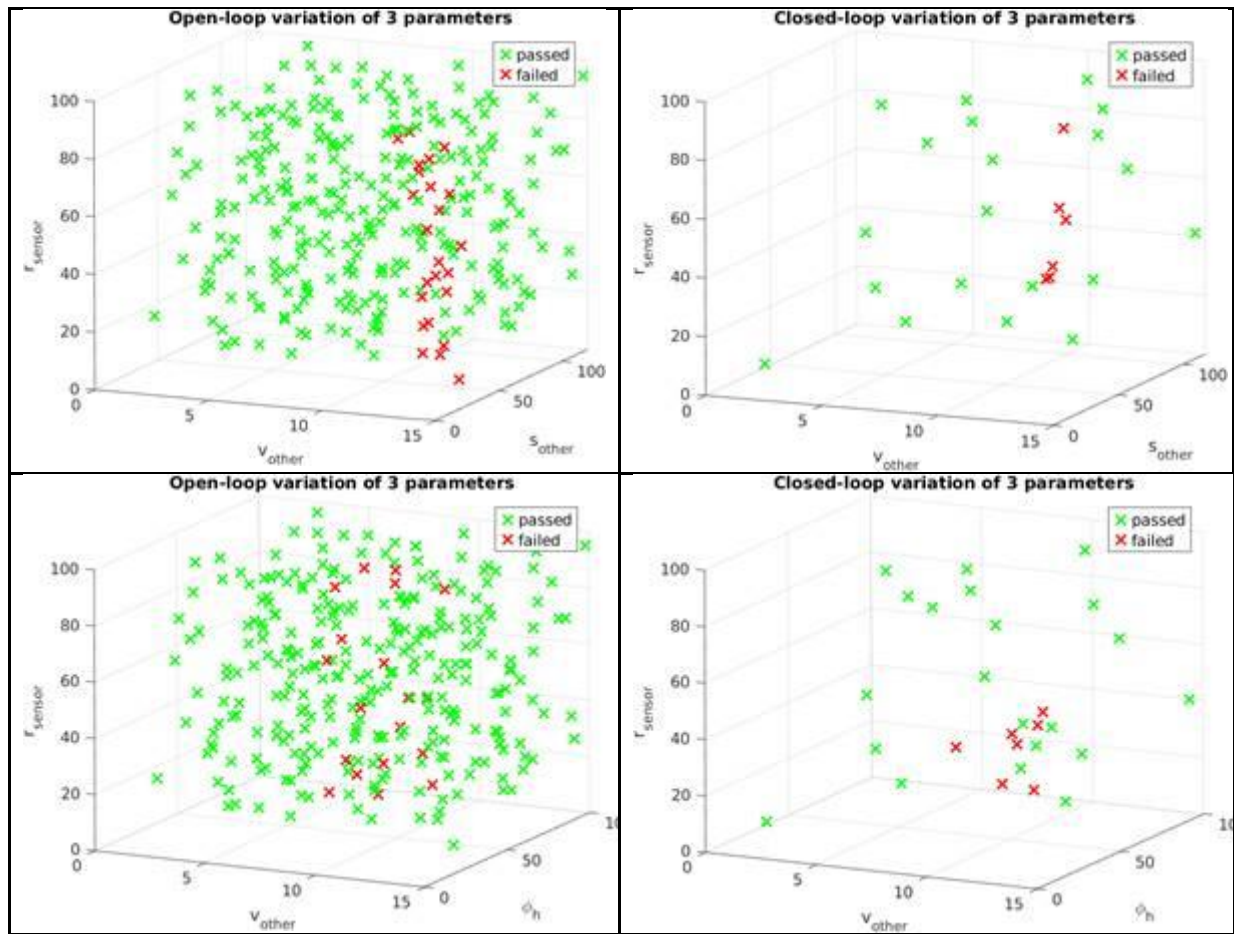


Figure 18: Two examples of 3D parameter variations for this scenario. The closed-loop method (right column) finds critical cases with fewer simulations than the open-loop method.

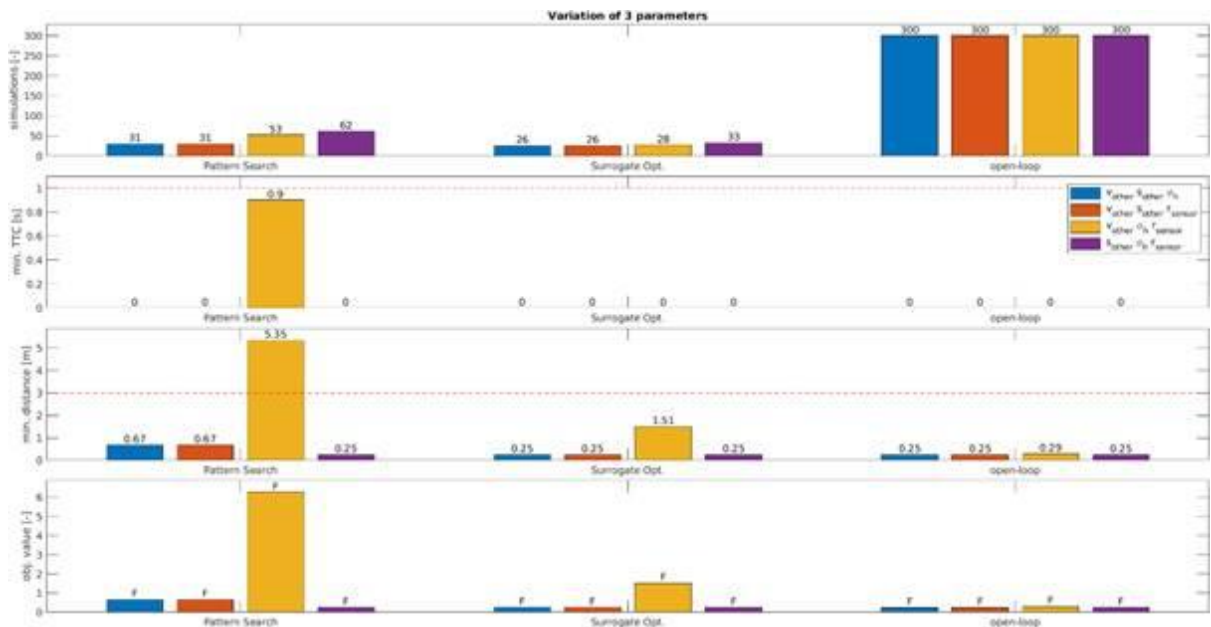


Figure 19: In the case of 3D variations, surrogate optimisation also finds scenarios with collisions in the least amount of time.

Scenario (ego: right turn, other vehicle: from the front) 2D variations

Figure 20 shows a scenario where the closed-loop method finds a failed test case, which remains hidden for the open-loop approach. Number of simulations and results are compared in Figure 21.

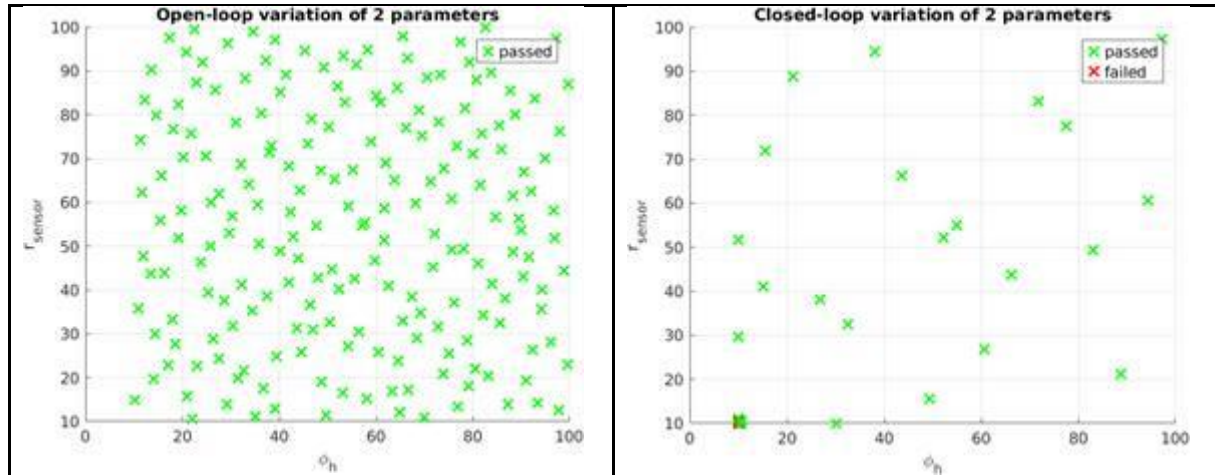


Figure 20: In this scenario, the open-loop method (left subplot) did not find any collision in contrast to the optimisation-based closed-loop technique (right subplot).

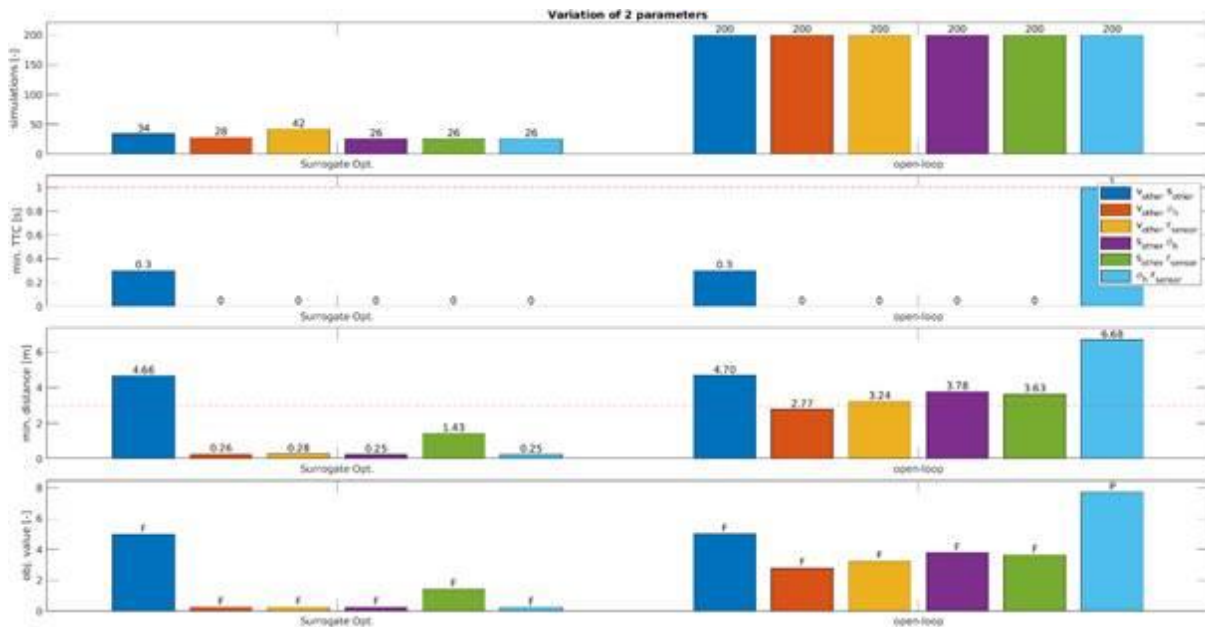


Figure 21: Comparison of surrogate optimisation and open-loop method for 2D variations in this scenario. The closed-loop approach needs fewer simulations and finds collisions where the open-loop method does not (light blue bars, corresponding to the figure above).

Scenario (ego: left turn, other vehicle: from the front) 4D variation

For the 4D case, no visualisation in the cartesian parameter space is available. Thus, only the comparison in Figure 22 is presented.

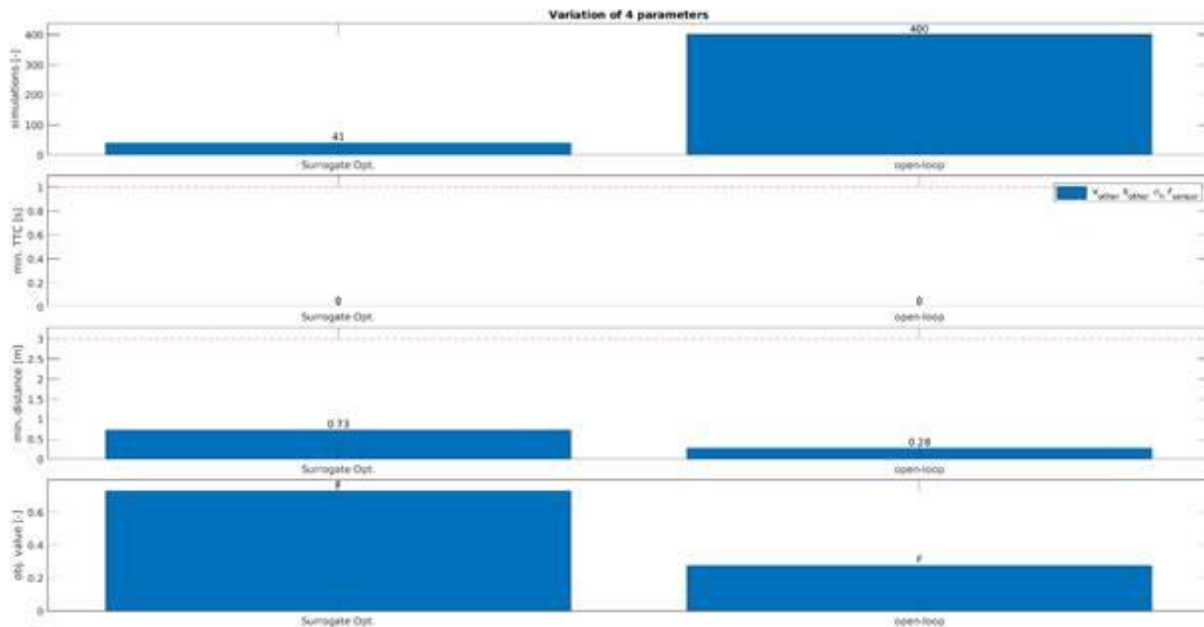


Figure 22: Closed-loop and open-loop result comparison for 4D variations. Again, the optimisation-based method achieves similar results with fewer simulation executions.

This work covers the third and fifth evaluation criteria of the Query & Concretise block.

3.1.3.3 Allocate

CVC:

The allocation process consists of two stages. First, it ensures that the requirements are fulfilled for the tests to be conducted. Then, the most appropriate test scenarios are selected, giving priority to virtual cases over real-world tests. However, since CVC can only perform virtual testing, only the first stage of the allocation process is executed in these demonstrations.

The test requirements are defined in D7.1. Not all requirements are considered in the CVC scenarios, as they focus exclusively on virtual environments and camera-based sensors for driving. Requirements related to LiDAR, radar sensors and physical tests are excluded for CVC demonstration.

Among the ODD requirements, cases such as adverse weather conditions are considered, but only in terms of their visual impact on camera sensors. Physical effects, such as reduced tire traction on wet roads, are not modelled and thus do not influence vehicle dynamics. Similarly, conditions like strong winds are excluded as they have no visual effect on the camera sensor.

IFAG:

IFAG used the initial allocation process described in Sunrise D3.3. Since IFAG only used virtual testing in this use case, the allocation process will not be used to distribute test cases between different test instances but to verify that a test case can be allocated to the available test instance. Furthermore, it showcases that knowledge-driven scenarios can be handled as

well. In doing so, the matching accuracy (a defined validation criteria) is proven as based on the capabilities and constraints of the test environment (only virtual testing in this case) a matching towards the higher fidelity virtual testing setup is executed. This can be seen in detail in the *execute* block.

VIF:

The purpose of the initial allocation is to assign the defined test cases to a test instance. A process has been defined for this purpose in the Sunrise D3.3, which shows and describes the necessary steps. This process first compares the requirements of the test case with the capabilities of the available test instances. This determines which of the test instances can be used to execute the test case. If several test instances are possible, it is then determined which would be the most efficient, whereby it is not the cost but rather the execution time that is important. In principle, simulations are preferable to real-world tests. In a further step, however, test cases can be reallocated, which is described in the Sunrise D3.5.

VIF wanted to show that this process works with its selected scenarios. However, as only one test instance (a simulation framework) was available to VIF, it was not possible to run through the entire process. Therefore, the focus was on the first part of the process, in which the requirements of the test case were compared with the capabilities of the simulation framework to ensure that the framework was suitable for executing this test case.

The structure and the criteria that are compared were defined in D3.3 and are based on the ISO 34503 standard. What has not been defined is the way in which the comparison is to be presented. VIF decided in favour of a comparison within an Excel file (see Figure 23). All the categories that are to be compared are listed in the left-hand column, with the arrows representing their position in the tree structure. The requirements of the respective test case are entered in the next column, and in the right-hand column, it can be selected whether the test instance can fulfil the category. If the test instance fulfils the requirements of the test case, the cell is coloured green. This makes it easy to recognise whether all requirements can be fulfilled or not.

Structure		Requirements	Capabilities
Main topic	Sub topic	Test case 1	Test instance 1
Scenery elements	Zone		
	> Geo-fenced area	no	No
	> Zone type	intersection	Yes
	> Regions /states	Austria	Yes
	Drivable area		
	> Surface	asphalt	Yes
	> Signs	no	Yes
	> Type	distributor road	Yes
	> Geometry	plane	Yes
	> Lane specification	3m	Yes
	> Edge	line markers	Yes
	Junctions		
	> Type	T-junction	Yes
	> Signalization	no	Yes
	> Additional attribute	no	No
	Basic structures		
	> Buildings	no	Yes
	> Streetlights	no	Yes
	> Vegetation	no	Yes
	> Other basic structures	no	Yes
	Special structures		
	> Bridges	no	Yes
	> Tunnels	no	Yes
	> Smart RSUs	no	No
	>> Sensing	no	No
	>> I2X	no	No
	> Other special structures	no	Yes
	Temporary drivable area structures		
	> Road works	no	Yes
	> Additional signage	no	No
	> Temporary modifications on other elements	no	Yes
Environment Conditions	Weather		
	> Ambient air temperature	20°C	Yes
	> Wind	calm	Yes
	> Rainfall	no	No
	> Snowfall	no	No
	Particles	no	No

Figure 23: A section of the Excel file that was used to compare the requirements of the test case with the capabilities of the selected simulation framework.

In the case of the selected test cases, it could be shown that the selected simulation framework fulfils all requirements to be able to execute the test cases. Therefore, this work covered the first evaluation criteria of the Allocation block.

3.1.3.4 Execute

CVC (Virtual Testing):

The execution part is responsible for running the test scenarios. Since CVC operates only in virtual environments, this module handles both the generation of the scenarios and the execution of the AD stack. An overview of this process is illustrated in Figure 24, which shows the components involved in the execution phase.

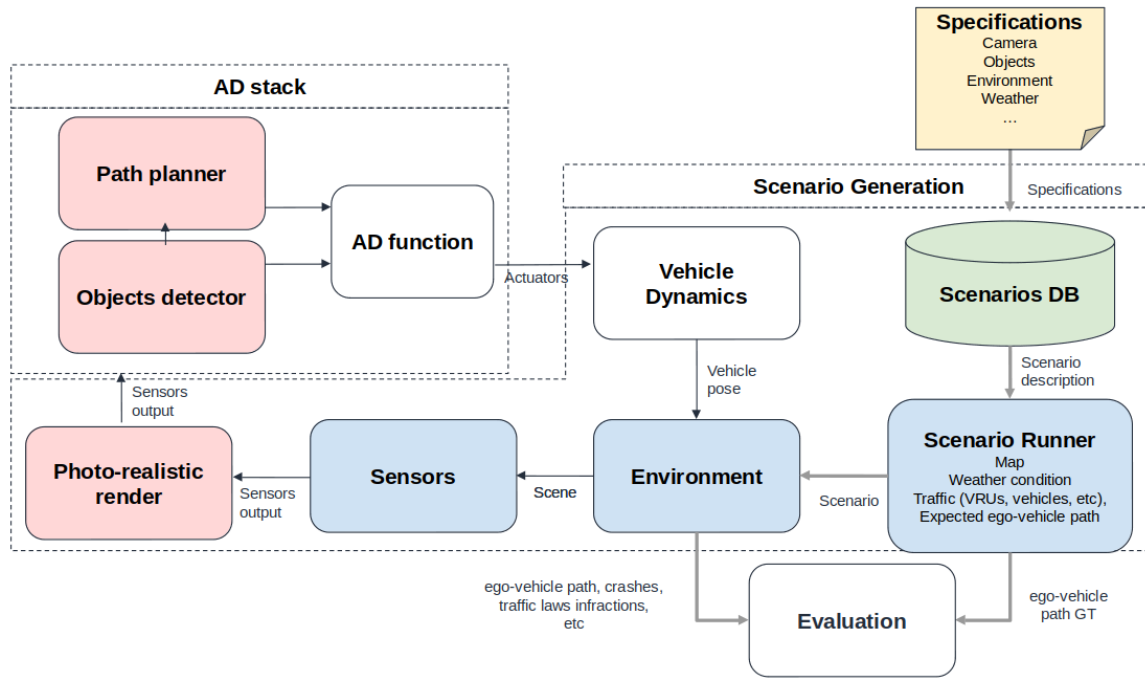


Figure 24: Framework used by the CVC to prove the SAF framework for the selected scenarios. It includes the scenario generation and the AD stack used in the vehicle. Blue boxes represent modules run by the CARLA simulator. Red boxes are trained AI models. White boxes are coded scripts.

Scenario generation is performed using the CARLA simulator, which creates both the environment and the behaviour of participant agents, such as vehicles, pedestrians, and cyclists. As CARLA's camera sensor simulations lack photorealism, an AI-based enhancement module is applied to improve the visual realism of the rendered images. Examples of the CARLA generated images and photorealistic AI outputs are shown in Figure 25. These enhanced images are used throughout the remaining execution process.



Figure 25: Left: Camera sample generated by the CARLA simulator. Right: Rendered image by the photorealistic AI.

Although the combination of CARLA and the AI enhancement module does not support real-time processing, the system operates in a synchronous mode to ensure that the AD stack can function at the required speed. Future work includes optimising the performance of the image enhancement pipeline to enable real-time scenario generation.

Figure 24 also presents the architecture of the AD stack. It comprises multiple AI models: one responsible for route planning and vehicle navigation through free space, and another focused on object detection, enabling actions such as braking and collision avoidance. The AD stack determines the driving path and braking behaviour based on outputs from these models. The perception results presented for UC1.1 specifically focus on the object detection module.

The CVC considered:

- 2 scenarios involving object avoidance in parking areas (pedestrians and open car doors),
- 5 scenarios involving object avoidance in urban settings (e.g., construction objects, overtaking cyclists, sudden braking by leading vehicles, and pedestrian crossings on straight roads and at turns),
- 3 roundabout scenarios (each covering a different exit the vehicle must take).

A detailed description of these scenarios can be found in Sunrise D7.2.

All scenarios executed by the CVC are simulated using the CARLA environment. For each scenario, the simulator records detailed data, including the exact positions of all objects, lane information, the ego vehicle's trajectory, and any collisions. The AD stack communicates with CARLA via ROS2, and its predictions are recorded using standard ROS2 tools. Using this ground truth data, the validation criteria defined in Sunrise D7.2 are applied.

IFAG (Virtual Testing):

In the previous T4.4, IFAG specified the simulation framework, using standardised interfaces such as ASAM OSI, the FMI 2.0 standard and FMU. In T7.2, the complete simulation framework, including the radar sensor model (as FMU, integrated into the framework using FMI), was shown already. In addition, an exemplary demonstration instance was shown. IFAG contributed to T7.3 by setting up the demonstration instances related to the demonstration of the radar sensor model, focusing on the development of visualisation concepts for the targeted phenomena of the high-fidelity radar sensor model. Concretely, the chosen effects, in line with the metrics and success criteria defined in T4.5, are phase noise, mixer non-linearity, and phase drift. These activities also include the integration of the high-fidelity radar sensor model into the simulation framework specified in T4.4.

In the following, the evaluation of each included sensor effect is briefly explained. This is followed by a comparison, using the chosen EuroNCAP scenarios, between two demonstration instances (one with and one without the inclusion of sensor model phenomena).

Phase Noise:

The chosen scenario for the verification of the phase noise phenomena includes 3 targets at approximately 40 meters of distance from the radar sensor. As Figure 26 shows, three distinct peaks, although the targets are all in very close proximity, are distinguishable. Furthermore, the estimated and true power levels for the targets are matching.

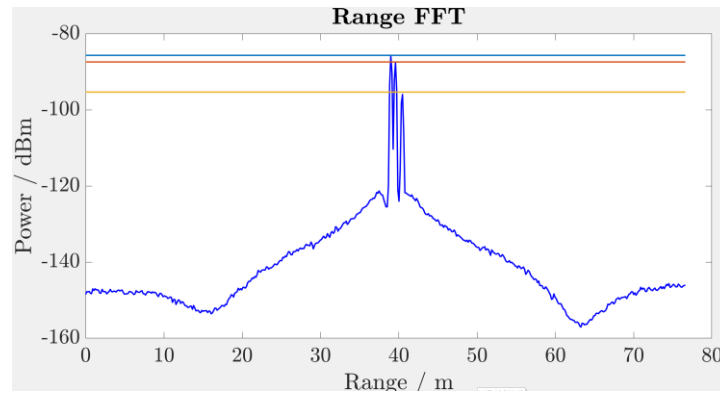


Figure 26: The radar power over range.

Interference:

For the same parameters of interferer and ego vehicles, a ghost target should appear because of the constant frequency difference between ego and interferer. Different parameters of interferer and ego should result in a uniform „noise floor“ because no constant frequency difference will appear. The analytical power and magnitude values should match with estimated target peak power and magnitude.

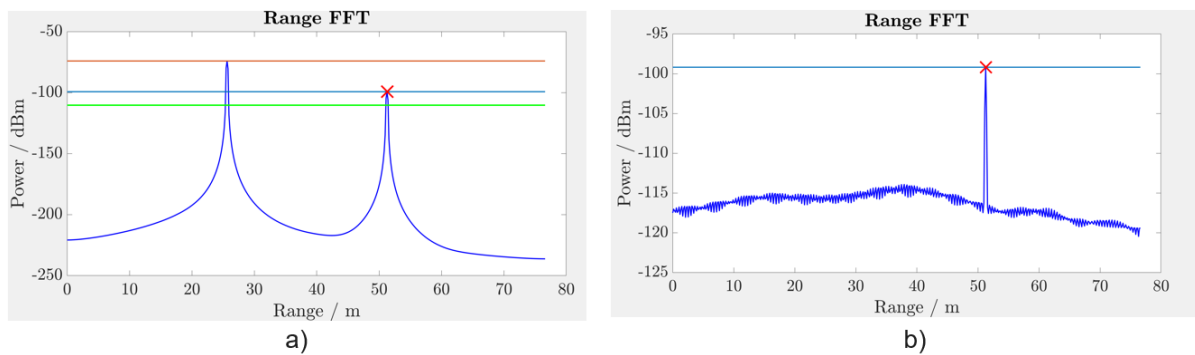


Figure 27: a) Interference between different sensors – same ramp duration for interferer and ego sensor b) Interference between different sensors. For both: The red arrow indicates the real target.

Overall, the same parameters of interferer and ego vehicles result in a ghost target in Figure 27. Different parameters of interferer and ego result in a uniform „noise floor“ but no ghost target. The analytical and estimated values for power and magnitude match.

Mixer non-linearity:

The mixer non-linearity is implemented as mathematical model. Concretely, a mixer voltage gain is considered. Furthermore, only the IM3 products $2\omega_{IF,1} - \omega_{IF,2}$ and $2\omega_{IF,2} - \omega_{IF,1}$ are modelled. Local oscillator leakage is not modelled.

Phase-drift:

Phase-drift describes the change of output phase of one TX channel, affecting also the phase balance, mainly over temperature. Phase-drift causes an angular estimation error as well as sensitivity/SNR degradation in the angular domain. Figure 28 shows the different deviations in the azimuth angle.

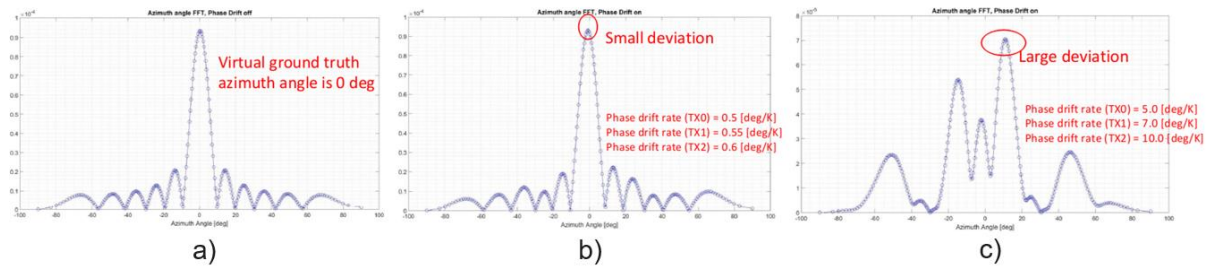


Figure 28: a) Virtual ground truth, azimuth angle is 0 degree, b) Small deviation, c) Large deviation

IFAG contributed to the execution block of the SAF by preparing the execution of the allocated test cases. The actual execution was done using the described virtual testing setup, which eventually enabled to demonstrate that the execution block of the SAF can handle EuroNCAP scenarios as well.

SISW (Virtual Testing):

In this block, the focus was on the SOTIF unknown unsafe scenarios. In this phase, real-world data from intersections is imported into the Prescan Critical Scenario Creation tool to extract actual road users' behaviours (see Figure 29).

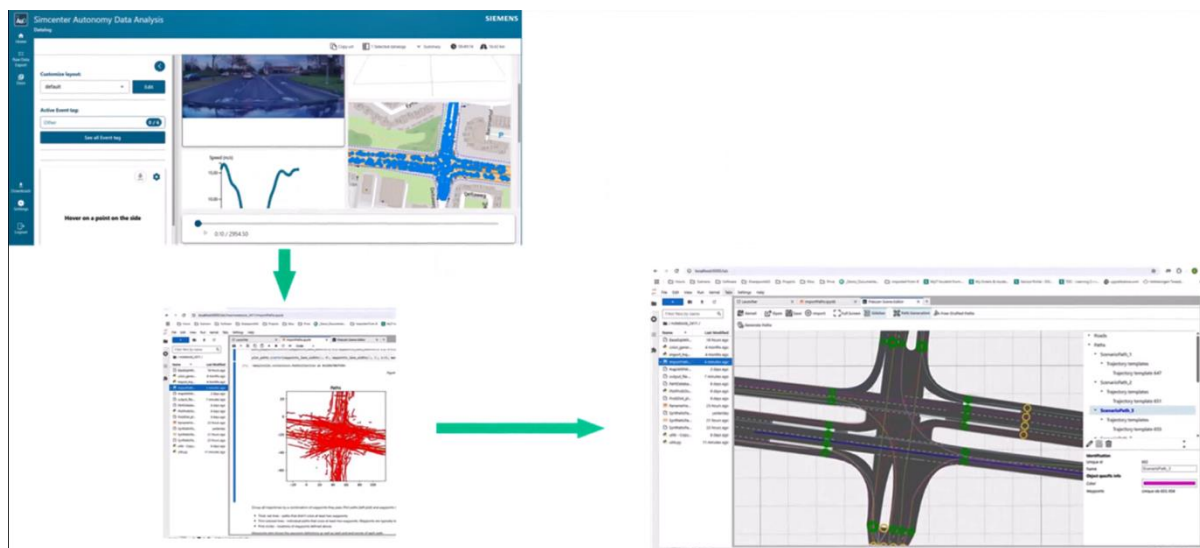


Figure 29: Importing real data to Prescan Critical Scenario Creation tool.

The Critical Scenario Creation optimisation algorithm assesses the severity of risks by employing proprietary indicators that measure criticality and novelty. These indicators provide a clear measure of the potential impact of these scenarios. By using novelty and criticality indicators, the critical scenario creation process can uncover previously unknown unsafe scenarios. Figure 30 demonstrates Execute process of Prescan Critical Scenario Creation tool which runs optimization process to find unknown unsafe scenarios.

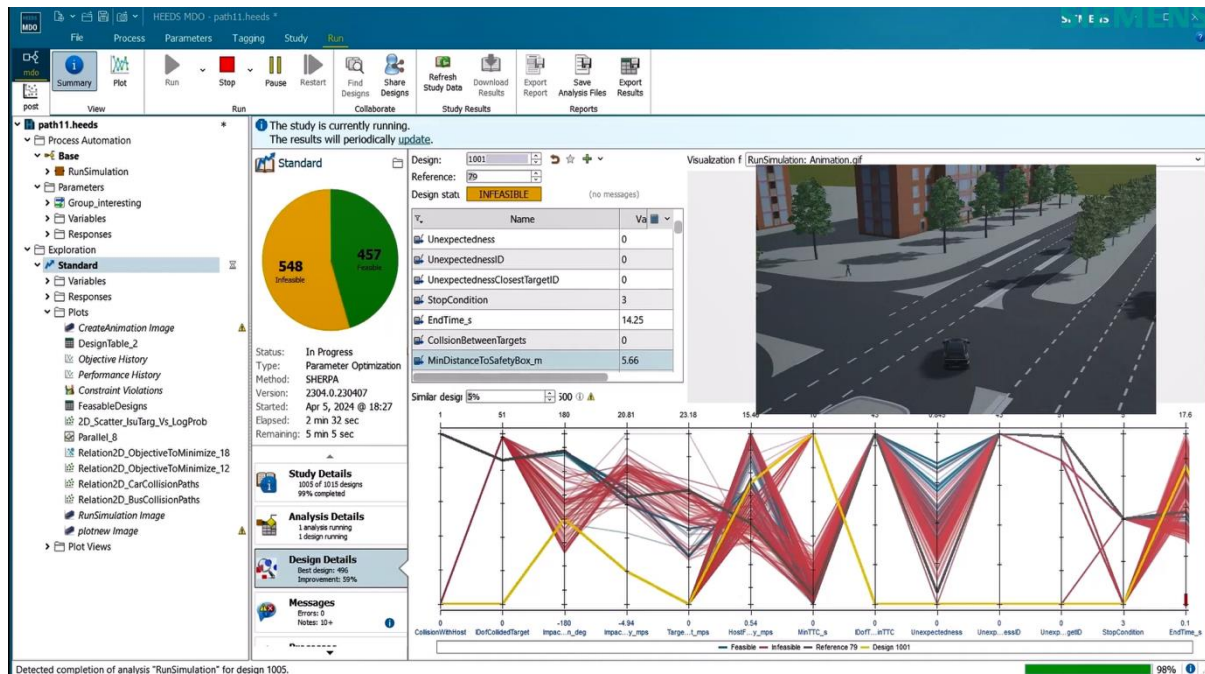


Figure 30: Optimisation process using Prescan and HEEDS.

These scenarios are further categorised using a dissimilarity metric, and the most critical cases are automatically selected for detailed analysis (Figure 31).

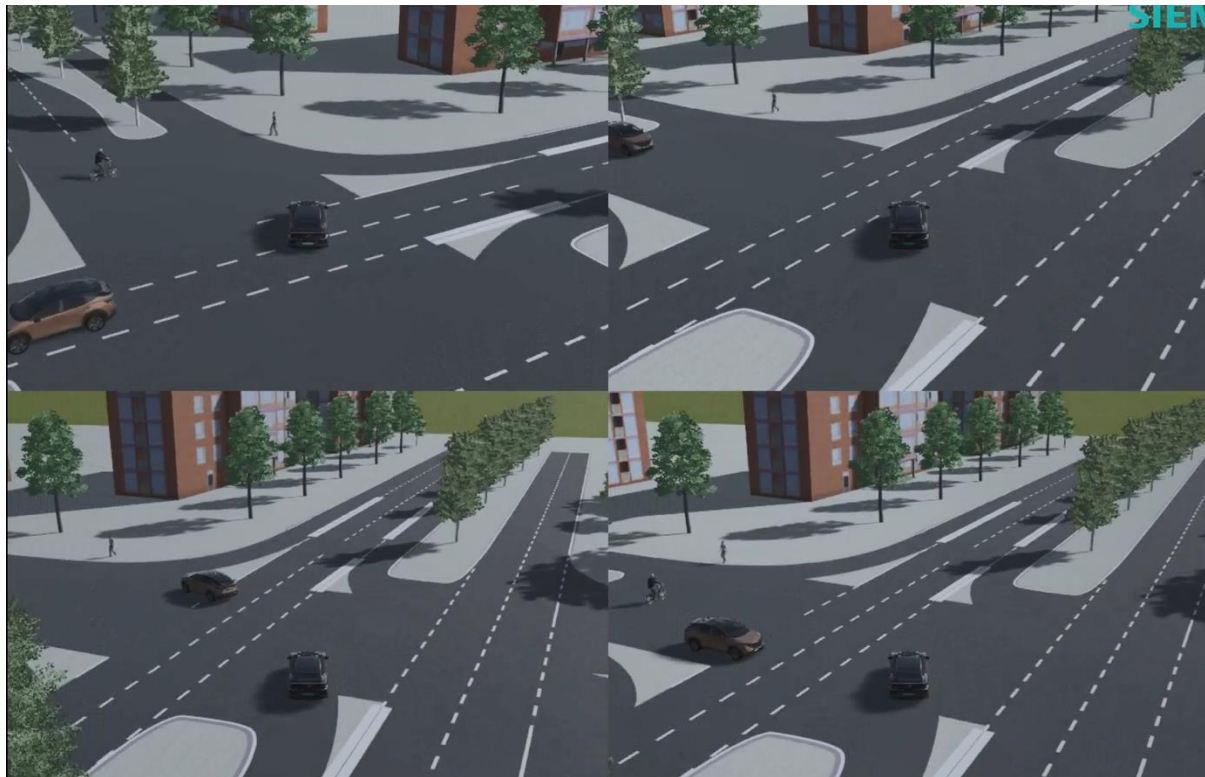


Figure 31: Generated a dissimilar set of unknown unsafe scenarios.

ViF (Virtual Testing):

ViF has developed a modular simulation framework designed to execute turn manoeuvres in an urban setting. A detailed description of the framework can be found in Sunrise D4.4 and Sunrise D7.2; therefore, only a brief overview is provided here.

Figure 32 illustrates the topology of the simulation framework. The co-simulation platform ModelConnect™ by AVL serves as both the co-simulation master and test case manager. An in-house developed Python module, RomPac, is used for both the AD function and vehicle dynamics. Additionally, sensor models, developed in-house, are integrated as Functional Mock-up Units (FMUs). The open-source tool esmini is employed for environment and traffic simulation.

Furthermore, the framework is built on ASAM OSI, OpenDRIVE, and OpenSCENARIO. The adoption of ASAM standards enables the execution of any scenario, provided it is in the correct format. The simulation results are saved as MCAP files and visualised using Lichtblick.

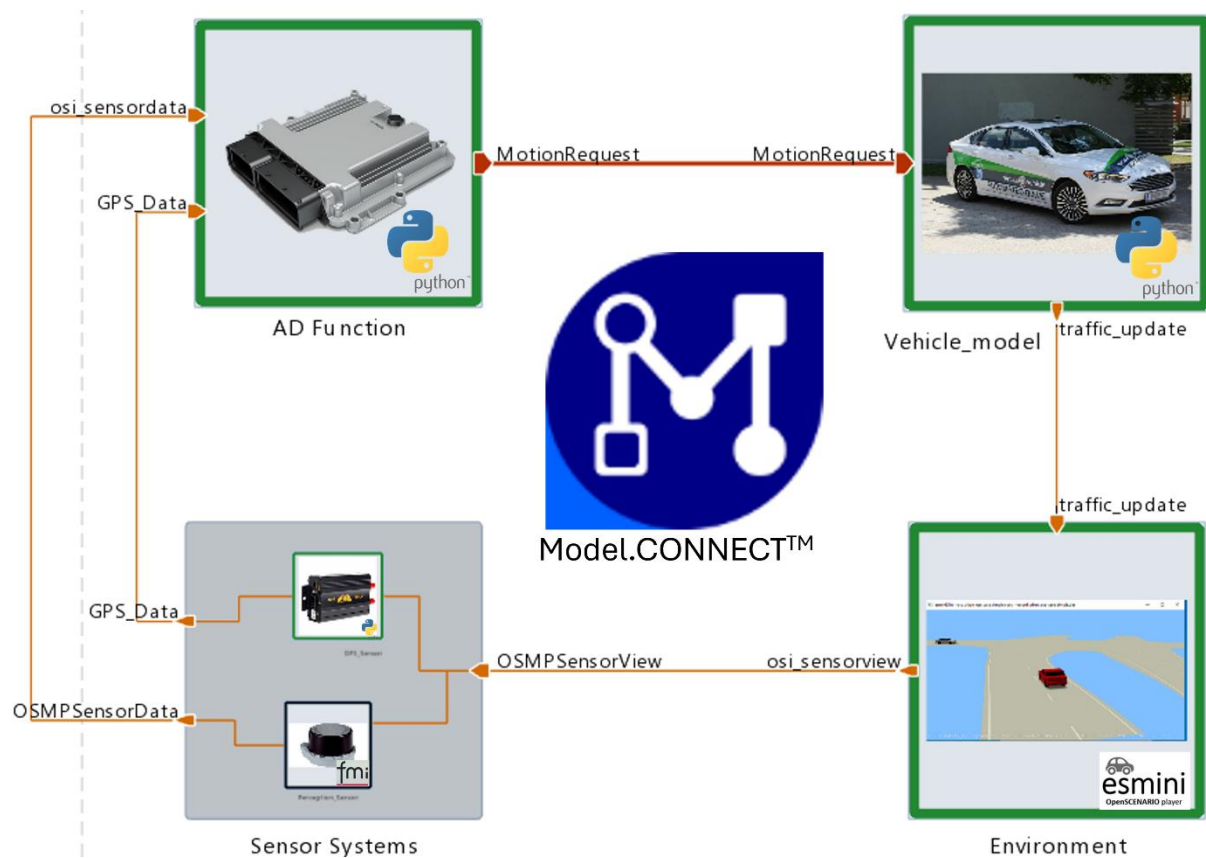


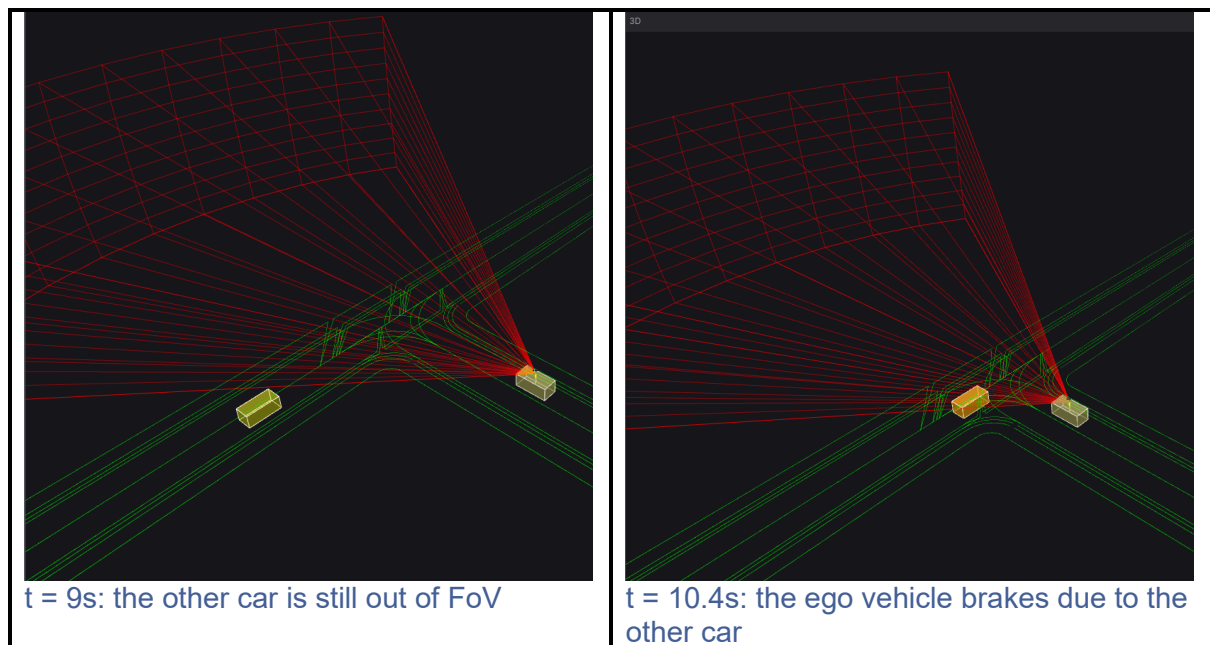
Figure 32: Simulation topology of the framework implemented in Model.CONNECT.

Figure 33 presents a 3D visualisation of the simulation result for the first scenario depicted in Figure 9. In this scenario, the ego vehicle, positioned at the bottom of the scene, is tasked with making a right turn. Its motion is governed by the implemented Automated Driving (AD) function, and the Field of View (FoV) of its perception sensor is illustrated in red.

Another vehicle approaches the intersection with a constant velocity, unaffected by the behaviour of the ego vehicle. The perception sensor settings used in this scenario correspond to the first row of Table 4. Notably, the horizontal FoV is deliberately restricted to just 65°, creating a (near) critical scenario to test the limits of perception and AD function. The Occlusion Threshold is not relevant in this case, as there is only one other vehicle and no additional obstacles present.

Table 4: Perception sensor settings for all scenarios.

	FoV Range	FoV Horizontal	FoV Vertical	Occlusion TH
	m	degree	degree	-
Scenario (ego: right turn, other vehicle: from the left)	100	65	20	1
Scenario (ego: right turn, other vehicles: from the left and front)	120	150	20	1
Scenario (ego: left turn, other vehicle: from the front)	70	120	20	1



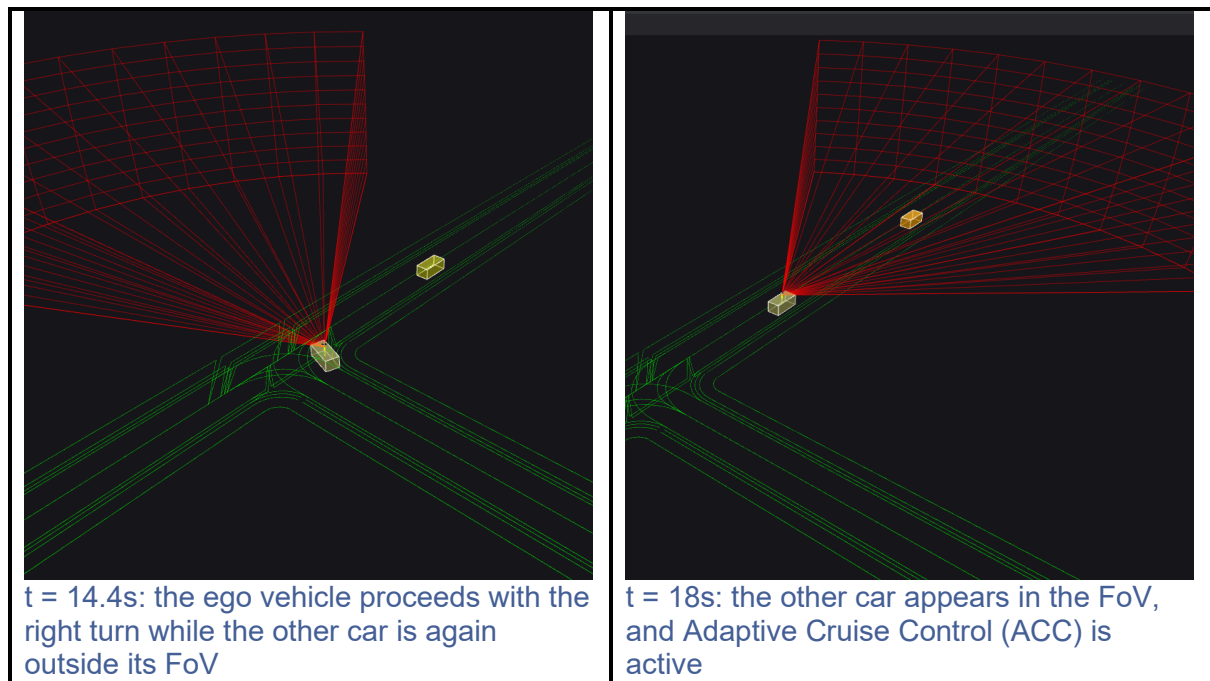


Figure 33: Near-critical scenario with a right-turn manoeuvre at a T-junction. The bounding box of the other car is coloured orange when it is detected by the sensor; otherwise, it is coloured yellow.

Figure 34 shows the most relevant time-series data for the scenario. The lower subplot of this figure illustrates the distance between the two vehicles over time, highlighting a critical threshold of 4 meters. At the 11-second mark, the distance comes close to this threshold but remains above it. As a result, the scenario is considered non-critical.

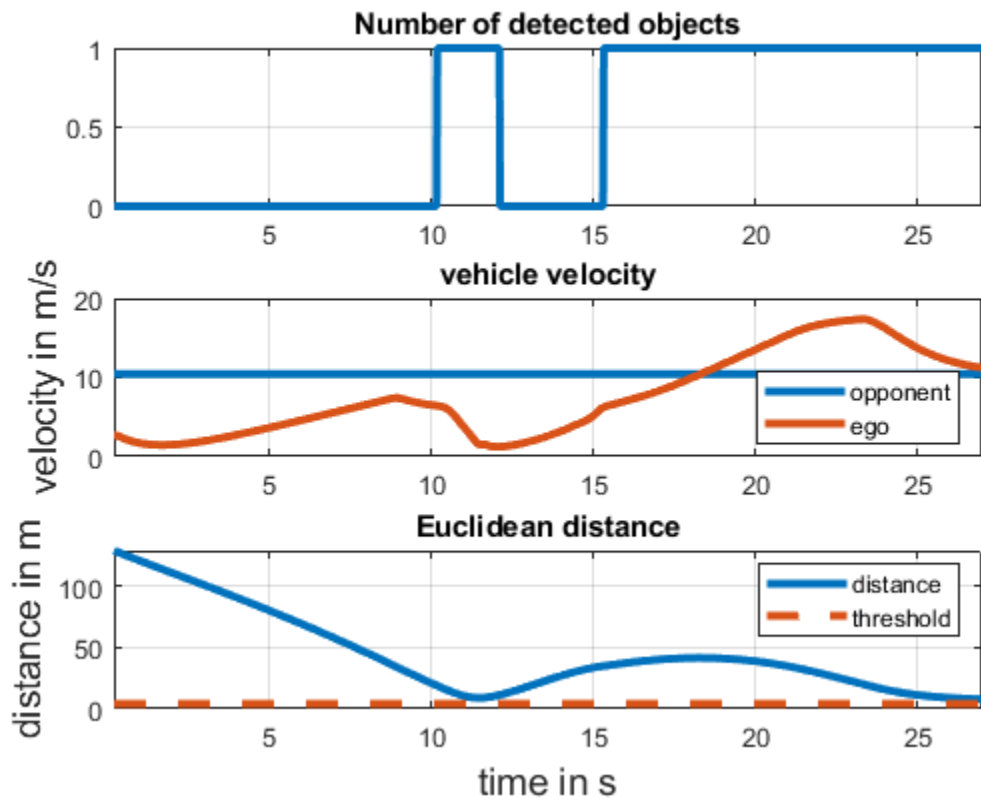


Figure 34: Time-series data of the right turn manoeuvre.

Figure 35 illustrates the simulation result of the second scenario, which features an X-junction and two other vehicles. The ego vehicle, positioned at the bottom of the scene, is again tasked with executing a right turn. One vehicle, located to the left, moves straight through the intersection at a constant velocity, while the vehicle opposite the ego vehicle is also performing a right turn. Both vehicles operate independently of the ego vehicle's behaviour.

The perception sensor settings for this scenario are specified in the second row of Table 4. In this configuration, the Occlusion Threshold (Occlusion TH) is set to 1, meaning that if an object is fully or partially occluded by another obstacle, it becomes invisible to the sensor. Due to this occlusion condition and the presence of multiple interacting vehicles, the scenario is potentially critical.

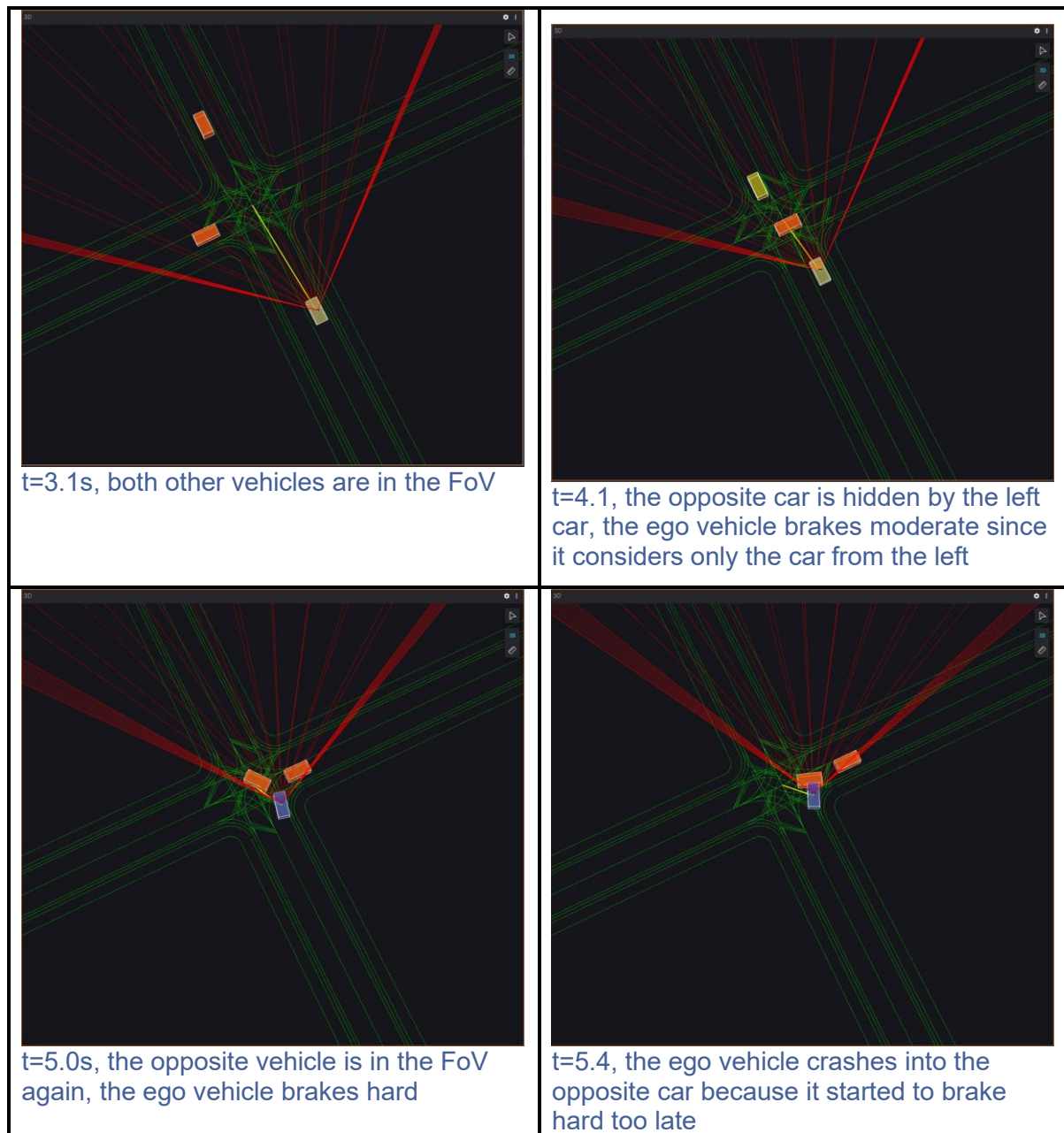


Figure 35: Critical scenario with a right-turn manoeuvre at an X-junction and two other cars.

Figure 36 presents the most relevant time-series data for the second scenario. In the upper subplot, it can be observed that the number of detected objects decreases from 2 to 1 between 3.7 seconds and 4.4 seconds, as one of the other vehicles becomes temporarily occluded and is no longer visible to the perception sensor

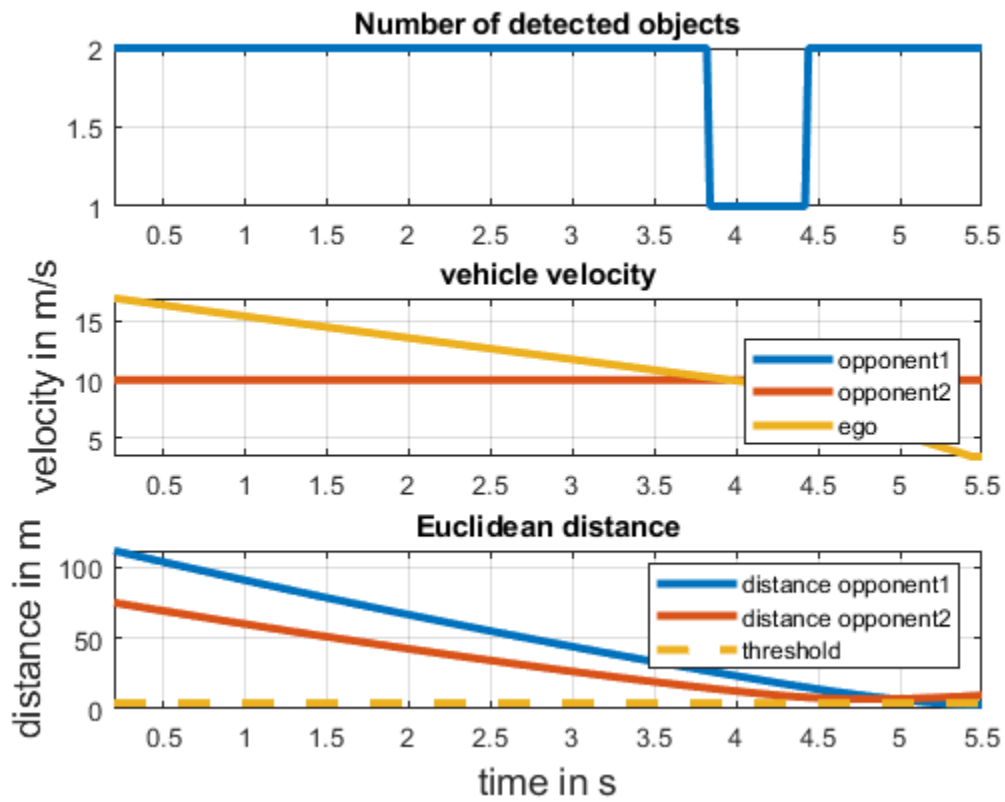


Figure 36: Time-series data of the 2nd scenario.

Figure 37 illustrates the simulation result of the third scenario, which takes place at a T-junction involving one other vehicle. The ego vehicle, positioned on the right side of the scene, is tasked with making a left turn. Meanwhile, another vehicle moves across the junction at a constant velocity, unaffected by the ego vehicle's behaviour.

The perception sensor settings used in this scenario are detailed in the third row of Table 4. Notably, a relatively short Field of View (FoV) range is intentionally selected for this setup, increasing the challenge for the perception system in detecting cross-traffic.

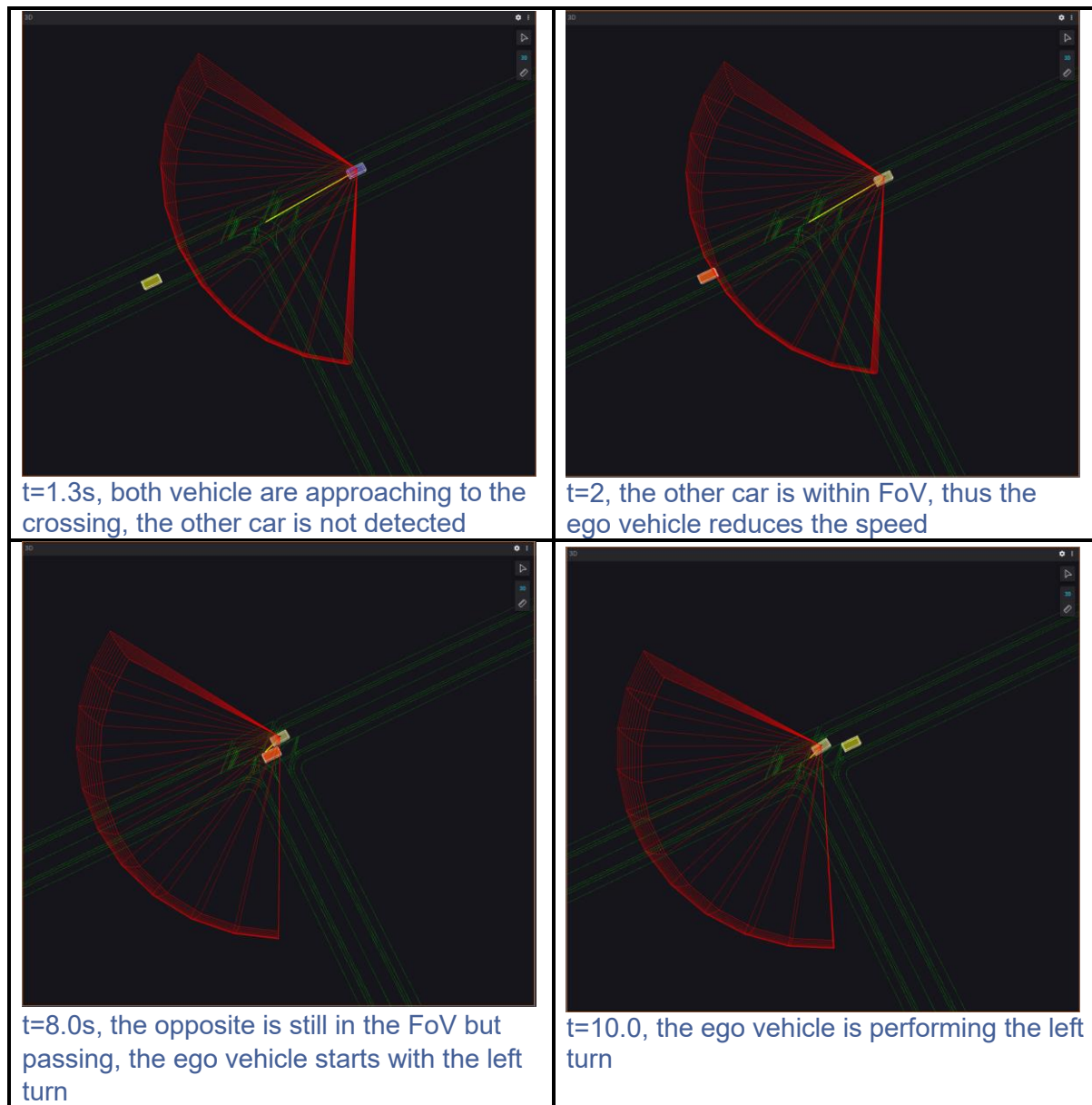


Figure 37: Critical scenario with a left-turn manoeuvre at a T-junction and another car.

Figure 38 presents the most relevant time-series data for the third scenario. In the lower subplot, it can be observed that the distance to the other vehicle briefly drops below the predefined threshold. However, as the other vehicle continues to pass by without interfering with the ego vehicle's path, the situation does not escalate into a critical scenario.

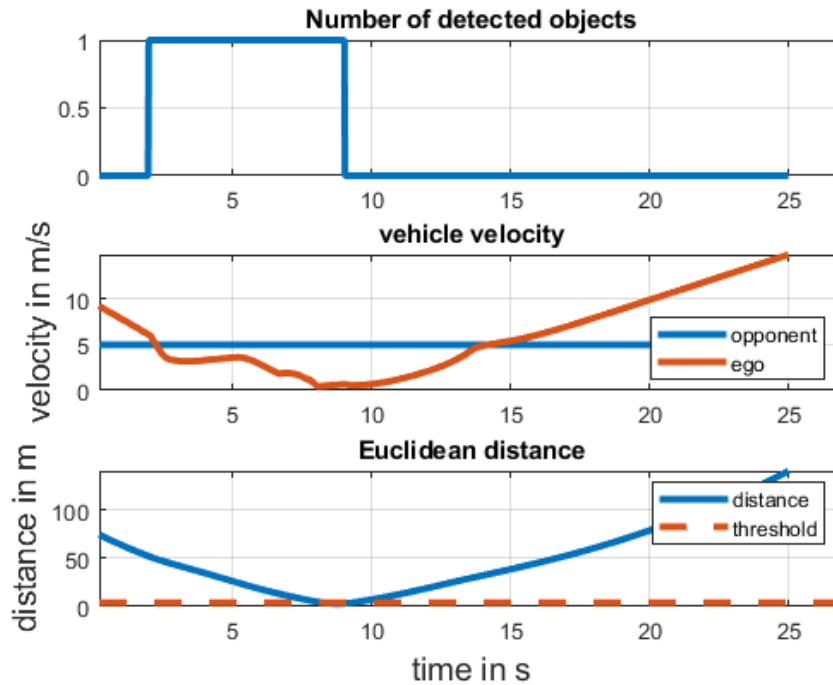


Figure 38: Time-series data of the 3rd scenario.

This work covers the second and fourth evaluation criteria of the Execute block.

RESA (Proving Ground Testing):

RESA is providing an autonomous driving prototype equipped with a LiDAR-based perception system. The vehicle's equipment and area of operation details are in Sunrise D7.1 [3]. The vehicle equipped with a Lidar can be seen in Figure 39.



Figure 39: RESA Autonomous driving prototype for proving ground testing.

The software architecture of the vehicle is the one in Figure 40. The focus is on the perception system. The data provided by the LiDAR is processed by four different modules to obtain the position, orientation, volume, and speed of all the obstacles surrounding the vehicle. The modules are Ground Segmentation, Static & Dynamic Object segmentation, Clustering &

Obstacle Detection, and Obstacle Tracking. The output of the perception system is then used by navigation and control to correctly react to the given scenario.

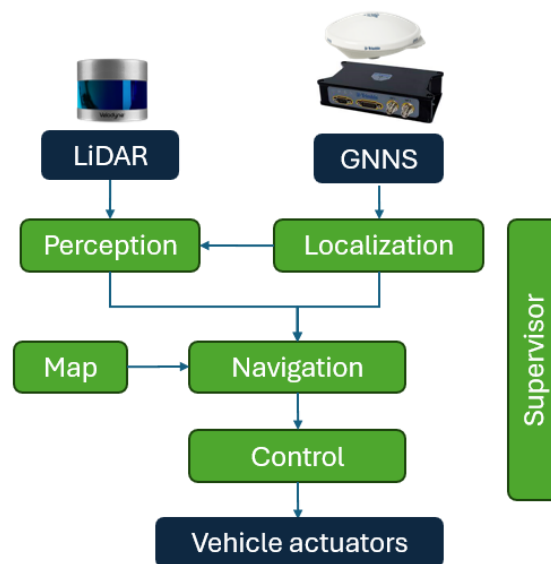


Figure 40: AD software architecture.

In order to execute the proposed scenario, the perception system is tested as part of the whole autonomous driving software pipeline. RESA's prototype is tested in car-following, outdoor parking with other vehicles leaving parking spots, with a parked vehicle with an open door, and obstacle avoidance due to a pedestrian. Testing conditions were sunny.

The prototype is equipped with GPS and IMU. The other vehicle (parked or car-following) is also equipped with GPS in order to be able to obtain precise positions of the ego and other vehicles, so the performance of the corresponding perception system can be evaluated.

To guarantee that the perception system is safe and reliable and can be implemented in the prototype without concerns, the performed experiments are aimed at evaluating all aspects of the proposed system: from robustness and reliability through indicators such as detection recall and precision, to the quality of the generated detections regarding obstacle distance, heading estimation, and speed prediction. Pass/Fail criteria is also measured, meaning the prototype manages obstacles in a correct way, either following a vehicle at a safe distance, or stopping when there is a static obstacle.

Execution of the pedestrian avoidance scenario is in SUNRISE's handbook (Check video in <https://ccam-sunrise-project.eu/uc1-1-perception-testing/>). Figure 41 shows all the other scenarios.



Figure 41: Proving ground scenarios.

3.1.3.5 Test Evaluate

Partners in this use case covered the SAF block “Test Evaluate” but not the other blocks of the Safety Argument part of the SAF.

CVC:

The test evaluation is done using the privileged information retrieved from the CARLA simulator. The pass/fail criteria are based on two conditions: whether the ego vehicle reaches the final waypoint and whether it avoids all collisions.

To evaluate the perception system, standard object detection metrics are used: Precision, Recall, Accuracy, and analysis tool-Score. These metrics are calculated only for objects within a 40-meter range, as it is enough to stop at the given speeds.

Table 5 presents the results. Although the perception metrics are not perfect, this does not directly impact the route completion rates or the number of collisions. This is because the planning AI can compensate for certain perception errors by relying on its predicted trajectory. Nonetheless, accurate object detection generally correlates with successful scenario completion.

Table 5: Perception (Precision, Recall, Accuracy, and F1-Score) and Pass/Fail (Collisions and completed route) metrics for the executed tests.

SCENARIOS	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)	Collisions	Completed route (%)
Parking area: open doors	92.90	97.06	98.36	99.12	0	52.53
Parking area: pedestrian	99.69	98.66	90.29	94.05	0	100
Urban area: work objects	92.29	97.82	90.94	94.72	0.94	51.14
Urban area: overtaking cyclists	97.47	99.57	97.16	98.37	0.25	94.30
Urban area: Following vehicle	98.14	98.88	96.96	97.68	0	100
Urban area: Pedestrian	98.99	98.88	97.85	98.54	0	100
Urban area: Turn +	99.41	99.94	99.36	99.63	0	100

Pedestrian						
Roundabout: Right exit	99.82	99.82	99.64	99.71	0	100
Roundabout: Second exit	95.85	98.65	94.50	95.11	0.2	95.68
Roundabout: Last exit	95.43	93.21	89.38	92.14	0.45	78.41

IFAG:

IFAG contributed by ensuring that the pass/fail criteria of this block include the criteria from EuroNCAP, demonstrating that this block is capable of handling these types of scenarios as well. Concretely, for the showcased demonstration instance, an actual collision was used as pass/fail criteria.

Concretely, two demonstration instances were compared: One without and one with the implemented sensor phenomena. Out of the relevant EuroNCAP scenarios, the AEB Pedestrian Test Car to Pedestrian Farside Adult (CPFA-50) is chosen. In this scenario, a car approaches an adult pedestrian crossing the street from the far side of the road.

As Figure 42 shows, the sensor model without the implemented phenomena leads to inaccurate detection, which is followed by a collision. However, with the sensor model with sensor phenomena, the detection at the same time instance is accurate and thus no collision occurs. This showcases the importance of including the relevant sensor effects in the model.

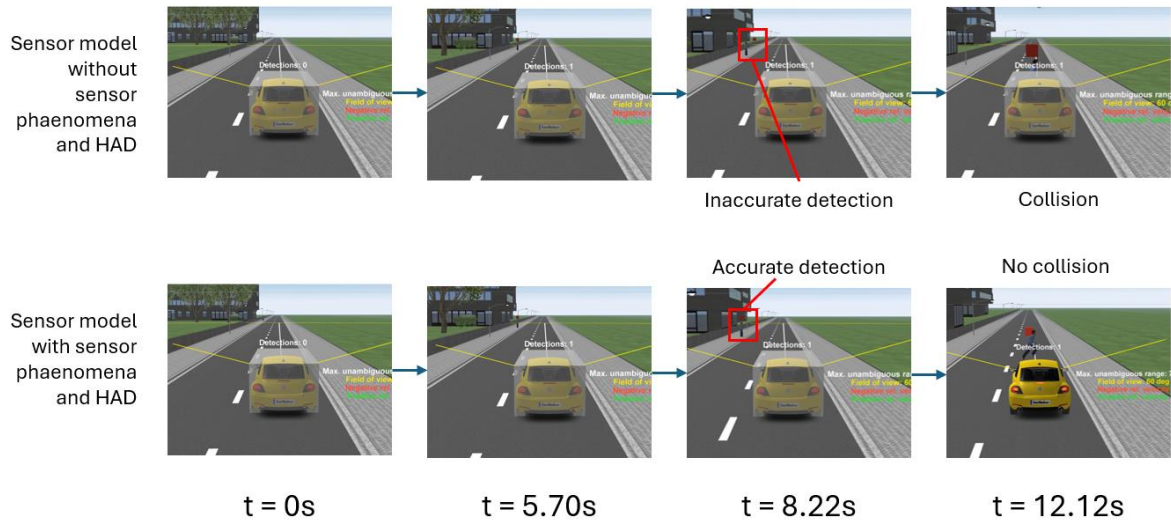


Figure 42: Top: The sensor model without sensor phenomena and HAD, Bottom: The sensor model with sensor phenomena and HAD.

RESA:

Experimental data was obtained in real environment, in which the vehicle navigates through Renault's site, while specific scenarios are reproduced with a controlled agent (preceding vehicle or pedestrian). Through the experiments, all data is recorded in rosbags, ROS' tool for recording data flowing through the software. The data is used to obtain the validation criteria defined in Sunrise D7.2.

Pass/Fail criteria are passed in all the proposed scenarios, as the prototype stops when there is a static obstacle or a pedestrian or safely follows a preceding vehicle.

To evaluate the perception system, standard object detection evaluation metrics have been used: Precision, Recall, Accuracy, and F1-Score. These metrics effectively indicate the system's overall performance and accuracy. They are derived from combinations of True Positives (correct detections), True Negatives (correct non-detections), False Positives (incorrect detections), and False Negatives (missing detections).

In Table 7, it can be observed that the proposed system achieves more than 85% in all metrics for all the considered tests. The car following test seems to be the one with better results, with an F1-Score of 100%. However, for the parked vehicle with open door sequence, even though the metrics obtained are still extremely favourable, the lowest metric being the recall with 85.02%.

Table 6: Precision, Recall, Accuracy, and F1-Score metrics for the executed tests. (Part 1)

SCENARIOS	TOTAL FRAMES	TP	TN	FP	FN
CAR FOLLOWING	587	587	0	0	0
LEAVING PARKING	267	123	141	0	3

OPEN DOOR	352	176	151	10	15
PEDESTRIAN	308	229	74	0	5

Table 7: Precision, Recall, Accuracy, and F1-Score metrics for the executed tests. (Part 2)

SCENARIOS	Precision (%)	Recall (%)	Accuracy (%)	F1 (%)
CAR FOLLOWING	100	100	100	100
LEAVING PARKING	100	97,62	98,88	98,8
OPEN DOOR	94,62	85,02	92,9	89,56
PEDESTRIAN	100	97,86	98,38	98,92

To further evaluate the perception system and the quality of its output, a vehicle equipped with a GNSS localisation system served as a reference obstacle (This cannot be performed for the pedestrian scenario). This setup allowed to establish a ground truth for the obstacle's real position, orientation, and speed, thereby enabling the collection of quantitative data crucial for validating any perception system.

To produce quantitative results from these experiments three variables have been evaluated: (1) the distance between the ego vehicle and the closest face of the Bounding Box of the vehicle equipped with the Ground Truth system, and the (2) heading and (3) speed estimated by the perception system for such vehicle/obstacle compared to those generated by the Ground Truth's.

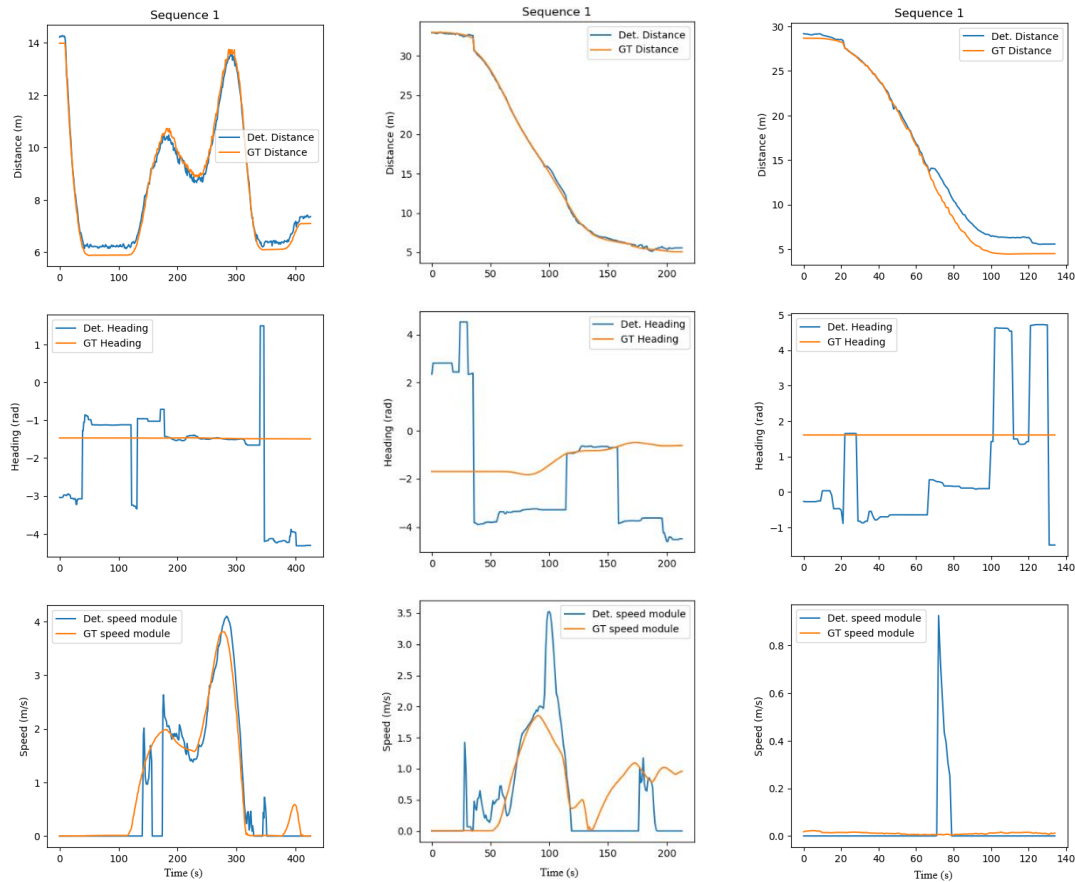


Figure 43: Comparison between the output of the proposed system (in blue) and ground truth (in orange) for executed tests. From left to right: Car following, vehicle leaving the parking and parked vehicle with open door.

As it can be observed in Figure 43, the proposed system is notably accurate in terms of distance and speed, being even difficult to distinguish Ground Truth from detection in some areas of the figure due to their similarity. While in the heading estimation, there is a significant error, this is due to the low speed of the obstacles, it is difficult to detect of an obstacle when it is static.

3.1.4 Key take aways

This use case evaluation demonstrated that the SUNRISE SAF provides substantial and practical guidance for the safety validation of perception systems.

Validation activities were conducted across a diverse set of environments, including simulation platforms and proving grounds.

To exploit the full potential of the SAF, hybrid results mixing simulation and proving ground tests would have been useful.

The major takeaway for SISW is that SAF provides a structural way to produce SOTIF known and unknown unsafe scenarios for the validation.

The major takeaway for IFAG is the fact that knowledge-driven scenarios, such as the mentioned EuroNCAP scenarios, can also be included in the overall SAF. Furthermore, when introducing the relevant sensor phenomena for virtual testing, the added benefits of introducing such phenomena can be clearly shown following the relevant aspects of the SAF. Hence, the SAF contributes towards ensuring that the required sub-systems of an ADS, such as radars, are considered properly as part of the overall safety validation, especially regarding the test method of virtual testing.

3.1.5 Deviations from D7.2

RESA:

Query & Concretise and Allocation of the SAF could not be validated as proving ground tests were conducted. Tests were conducted in the conditions of the tested day, and allocation makes sense when there is a mix between virtual and proving ground tests.

Decide was not validated as the results are not sufficient to determine if the whole perception system can be considered safe. The proposed system is considered safe for the given test scenarios, but more scenarios should be performed to ensure safety.

Test execution in adverse weather conditions was not conducted. This was due to the prototype not being ready on time, and by the time it was, most days experienced sunny weather.

CVC:

Sunrise D7.1 and Sunrise D7.2 proposed using a photorealistic renderer to extract metrics from static data generated by a photo-realistic simulator, producing different scenarios than those created in CARLA. Since then, AI-based approaches have advanced significantly and now offer superior realism in online images generated by the CARLA simulator. As a result, the CVC opted for this AI-based method, as it provides more reliable perception metrics when comparing AD stack performance in virtual versus real environments.

According to Sunrise D7.2, the CVC was expected to partially perform the Query & Concretise process due to its focus on virtual-only testing. However, the integration of the photorealistic renderer into CARLA took longer than anticipated. As a result, the Query & Concretise step, being the only component that could not be fully executed, was ultimately omitted.

3.2 Use case 1.2: Urban AD validation – Connected perception testing

3.2.1 Use Case Overview

UC1.2 tests the SUNRISE SAF on Urban intersection scenarios, with V2X communications that extend the in-vehicle perception system. There are four scenarios (See section 3.2.2). The system under test is an intelligent Connected Adaptive Cruise Control (C-ACC) with Green Light Optimised Speed Advisory (GLOSA) for optimal intersection crossing.

Objective:

The objectives of UC1.2 are to address current gaps in extended virtual perception through V2X cooperation, as well as ODD and scenario coverage, which include connectivity with vehicles and infrastructure (in this case, connected traffic lights

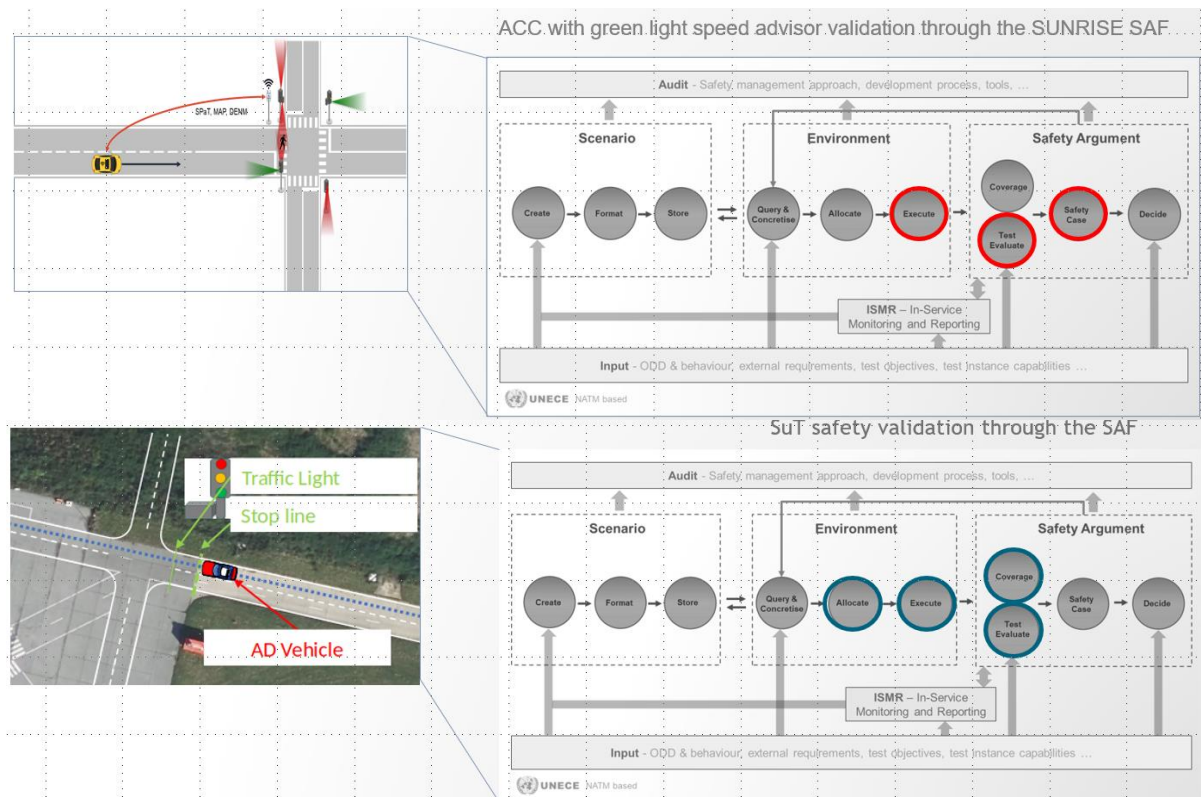
3.2.1.1 Covered aspects of the SAF

Table 8 and Figure 44 below, show the contributions to the SAF block validation per partner for UC1.2.

Table 8: Contributions to the SAF validation per Partner in UC1.2

Partner/ Block	ika	UNITN	VED
SUNRISE DF			
Query & Concretise	x**	x**	
Allocate	x		x
Execute	x	x	x
Test Evaluate	x	x	x
Coverage	x*	x*	x*
Safety Case		x	
Decide		x	x

* Only parameter space coverage. ** All scenarios are expert-defined. Only concretization methods were tested.



Handbook videos produced:

UC 1.2-B <https://www.youtube.com/watch?v=jzDMNKKIUgA&t=7s>

UC 1.2 <https://www.youtube.com/watch?v=12GBcRWN2jU>

UC 1.2-D <https://www.youtube.com/watch?v=dOK965QCCZc&t=1s>

3.2.1.2 Safety case setup

The objective is to apply the SUNRISE SAF to the safety assurance for a CCAM system in an urban environment, focusing on intersections where the majority of accidents occur due to distracted pedestrians or traffic light violations. The aim is to verify that the CCAM safety leverages connectivity. The information used can come from HD maps, sensors, and connectivity; however, connectivity is the primary focus. The following steps are carried out:

- Safety Goals and Key Performance Indicators:
- The primary indicator in simulations was collision avoidance (collision versus non-collision), which served as the key performance indicator. The second indicator was a red traffic light violation. A performance indicator was the deceleration of the vehicle, which must not pose rear-end collision risks.

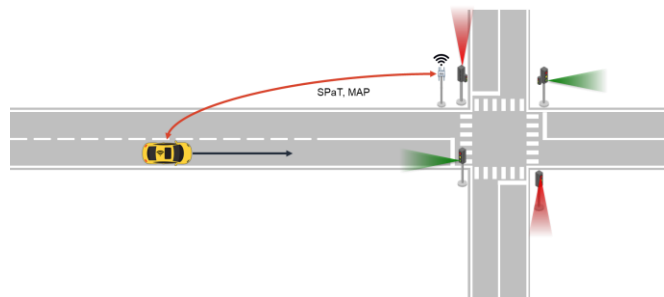
- Tests were conducted in different test environments, including simulations with IPG CarMaker at UNITN and simulations and Proving Ground tests at VED. This use case compares two sampling strategies: random versus Latin Hypercube Sampling (LHS).
- Simulation outcomes have been confirmed by physical testing on selected concrete scenarios by VED.

3.2.2 Overview of tested scenarios

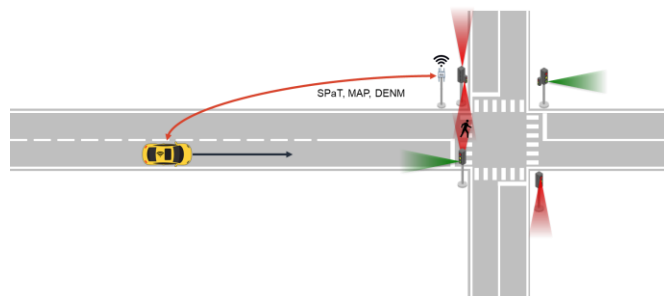
The use case comprises four subcases (Figure 45): A) approaching an urban intersection with Adaptive Cruise Control and Green Light Optimized Speed Advisory (ACC+GLOSA), B) similar to A but a pedestrian suddenly violates the intersection, C) like A but a pedestrian pushes the crossing request, resetting the traffic light phases, D) like A but a car violates the intersection. In all cases the Ego vehicle receives information from connectivity with the infrastructure (I2V). In case D it receives information also from the violating car (V2V). However, messages may be late.

Note that UC 1.2-C, beginning with the Ego Vehicle status at the time of the reset is identical to 1.2-A with the exception that the reset will, in general, allow sufficient duration for amber light. For this, we focused on the remaining Use Cases (conclusions for C can be drawn by adapting A).

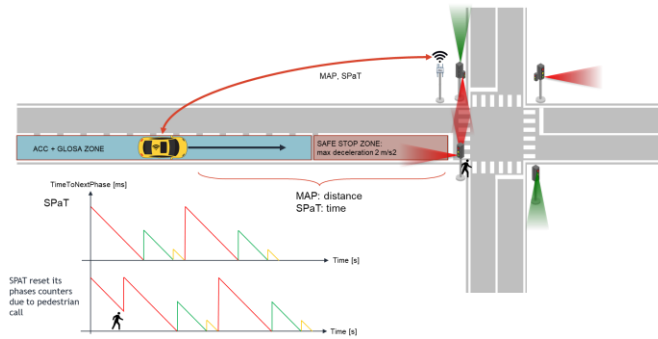
A) Adaptive Cruise Control and Green Light Optimized Speed Advisory (ACC+GLOSA).



B) Pedestrian violation.



C) Traffic light phases reset.



D) Vehicle violation.

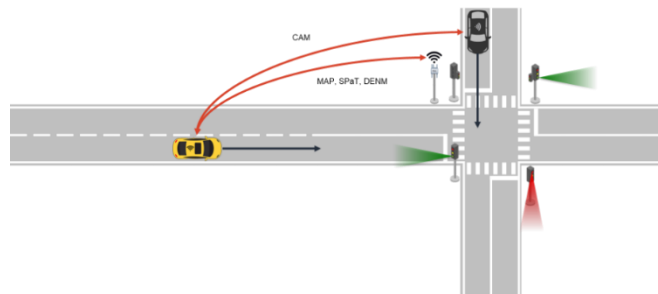


Figure 45: Use Case 1.2 includes four subcases.

3.2.3 SAF Block demonstrations

The logical scenarios were defined by experts. They are listed below.

The ODD is an urban intersection with Adaptive Cruise Control and Green Light Optimized Speed Advisory in Sunrise D7.1 Section 4.2.

3.2.3.1 Query & Concretize

Scenarios for the UC1.2 were created based on expert knowledge to fulfil the requirements set out in Sunrise D7.1. There was no query; however, we conducted tests on various sampling methods, evaluated coverage using surrogate models, reiterated sampling and execution, and concluded whether the system either passes or fails. In this way, many of the criteria listed in section 2.2 were used (see case-by-case annotations).

Approach 1 (UNITN/ika)

UC 1.2-A. The logical scenario has the following parameters as shown in Table 9. The initial distance is the distance where the first message informing about the traffic light phases is received. If it is too late, the ego vehicle will not be able to comply with the intersection. By varying the initial distance, one can test a system against late arrival of I2V communications (and find a safe speed for plausible lags).

Table 9: UC 1.2-A logical scenario parameters.

Speed limit	50 km/h
-------------	---------

Initial speed	30-50 km/h
Initial distance	100-0 m
Current phase	{green, yellow, red}
Time to next phase	0-30 s
Next phase	{green, yellow, red}
Time to 2 nd next phase	0-30 s
2 nd next phase	{green, yellow, red}
Time to 3 rd next phase	0-30 s

Use Case 1.2-A was extensively employed to test concretion approaches developed as part of the SUNRISE Methodology (Validation criteria Section 2.2.2, c and d). An initial set of concrete scenarios was determined by Latin Hypercube Sampling using 500 samples. In a second iteration, 200 new samples were generated.

UC 1.2-B. The logical scenario is shown in Figure 46. There are four parameters: Ego velocity, v (0-90 km/h), distance from pedestrian crossing q , pedestrian distance from the lane, p , and pedestrian average velocity, w (0-12 km/h). Note that, when sampling intersection scenarios, an expert consideration was that collisions cannot happen if the time of arrival at the intersection is greater than a convenient threshold (here 4 seconds). Hence, the TTI was sampled in the range 0-25 seconds, and the distances were derived.

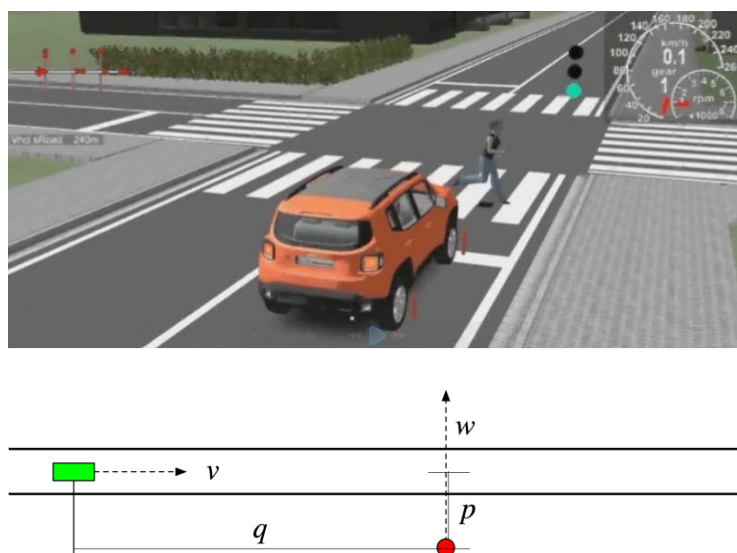


Figure 46: Logical scenario for UC 1.2-B.

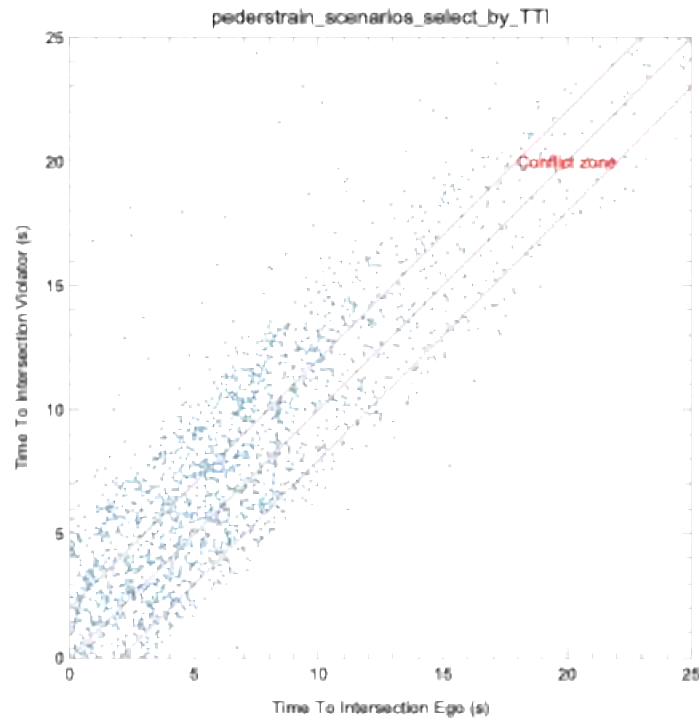


Figure 47: Initial sampling for UC 1.2-B.

UC 1.2-B was initially sampled with 1981 concrete scenarios with a TTI difference smaller than 4 seconds (Figure 47), selected from 50,000 random points. In a second iteration 572 new concrete scenarios were created near the boundary of the collision region using a surrogate model (see Safety Case Section).

UC 1.2-D. The logical scenario has the following parameters illustrated in Table 10. The speed limit is constant and the same for both vehicles (up to 90 km/h to include highway intersections). The requested cruising speed allows modelling transient velocity, including starting from the stop line at green onset. The traffic light is always green (other cases do not produce conflicts). The initial distance and speed determine the obstacle state when the first CAM message is received (at the start of simulations). The target obstacle speed models intentional variations of the obstacle's trajectory, which may include stopping in the intersection.

Table 10: UC 1.2-D logical scenario parameters.

Speed limit	90 km/h
Initial speed	0-90 km/h
Initial distance	0-300 m
Requested cruising speed	0-90 km/h
Current phase	green

Initial speed of the obstacle	0-90 km/h
Initial distance of the obstacle	0-300 m
Target obstacle speed	0-90 km/h

Use Case 1.2-D used concretization approaches based on LHS and random sampling, with 100 and 1000 samples (Validation criteria Section 2.2.2, c and d).

All sub-use cases were tested on the proving ground at Aldenhoven Testing Center. For each sub-use case a concrete scenario was chosen by experts to challenge the system, while not introducing unmanageable risk for the safety drivers in those scenarios. The query component of the SAF was not tested with the physical tests. The system tested was different from those simulated in the simulated tests.

3.2.3.2 Allocate

For scenario allocation to test environments, the initial allocation followed the process described in the Sunrise D3.3.

Approach 1 (UNITN/ika) was limited to simulations. In UC1.2-B and 1.2-D a second iteration with new samples was tested.

Approach 2 (VED)

3.2.3.3 Execute

- **Virtual Testing:**

Approach 1 (UNITN). The concrete scenarios were executed in IPG CarMaker (Sunrise D4.3, Sections 2.1.4, 2.2.4, 2.3.5, 2.4.2, 2.5.1, 2.6.1, 4.1.2), recording the complete logs and extracting the relevant information (Validation criteria Section 2.2.4, b). The KPIs used are found below, and they respond to the Validation criteria Section 2.2.6, a, b).

The virtual test environment could successfully run the scenario (Validation criteria Section 2.2.4, a, c). In case 1.2-B there was a second sampling-execution iteration (Validation criteria Section 2.2.4, d).

- **Proving Ground Testing:**

Approach 2 (ika)

Physical tests of all sub-use cases were carried out using the automated vehicle test platform of the RWTH's Institute of Automotive Engineering (ika). All test were conducted at the Aldenhoven Testing Center (ATC), a large proving ground near Aachen shown in Figure 49. This proving ground includes V2X capable infrastructure capable of sending the required messages needed to test UC1.2.



Figure 48: ika's automated vehicle test platform karl.

Tests were conducted using ika's test vehicle karl. The vehicle is shown in above Figure 48. The test vehicle is built upon the VW T7 Multivan platform. On the roof of the vehicle, 8 cameras and 5 lidars are mounted as additional sensors to perceive the vehicle's environment. In addition, 5 radar systems are mounted around the vehicle as well. The vehicle can be controlled both in longitudinal and lateral directions. The software to enable automated driving is developed in-house at ika. For the SUNRISE project, the software was extended to handle the different V2X formats required for the use case, including DENM, CAM, and GLOSA, transforming the vehicle into a true CCAM test platform.



Figure 49: Arial view of the Aldenhoven Testing Center.

Use case 1.2-A consists of a reaction of the vehicle to GLOSA messages sent to the vehicle by infrastructure, in this case a traffic light. The messages help the vehicle adjust its speed accordingly and prevent traffic light violations. The vehicle's ability to react to GLOSA messages, as defined in the UC1.2-A scenario, was tested at ATC, at an intersection capable of sending GLOSA messages. All other sub-use cases were tested at this same intersection. An overview of the intersection is shown in Figure 50. Figure 51 shows a digital snapshot of

the situation directly after the GLOSA message indicates that a shift to a red traffic light is imminent, with the planned vehicle trajectory shown as well. It can be observed that the vehicle replans the trajectory accordingly and comes to a stop at the traffic light.



Figure 50: Overview of UC1.2-A real world test.

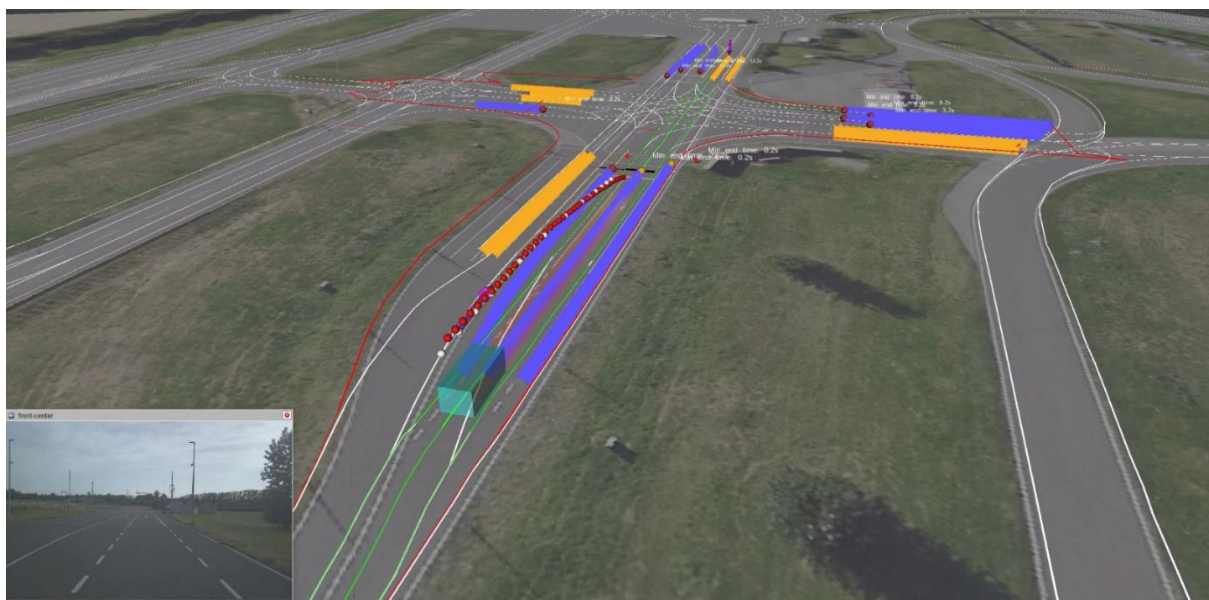


Figure 51: Digital view of UC1.2-A real world test.

Use Case 1.2-B consists of a distracted pedestrian entering an intersection illegally. The vehicle has to stop at the intersection to ensure a safe outcome in this situation. An overview of this situation is shown in Figure 52. The vehicle is warned of the pedestrian intrusion by a DENM message. In Figure 53, one can observe that the vehicle reacts to the DENM message with a slight delay and then proceeds to a full stop before the intersection. Figure 54 shows a digital snapshot of the moment at which the DENM was sent to the vehicle.



Figure 52: Overview of UC1.2-B real world test.

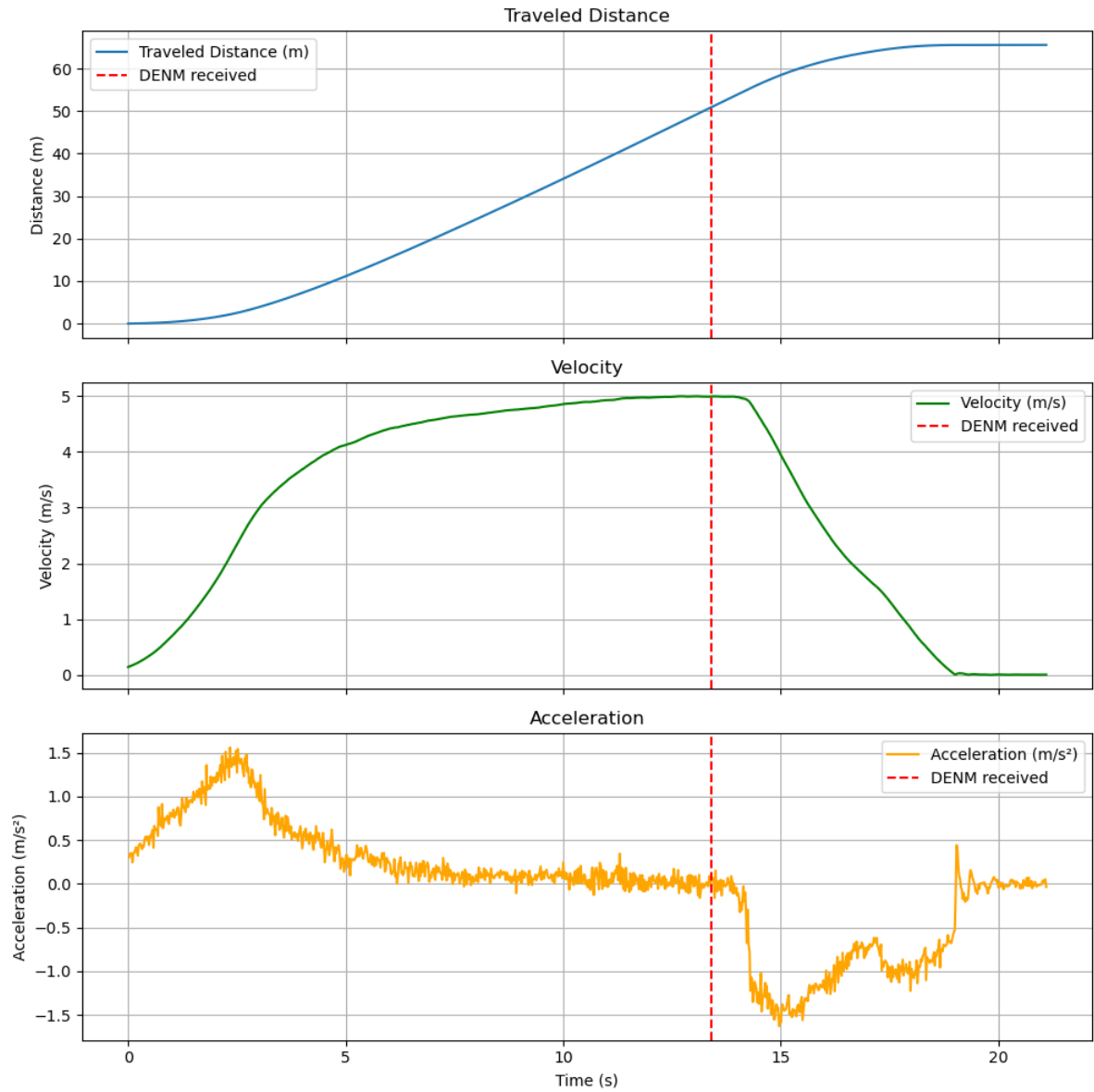


Figure 53: CCAM vehicle data of UC1.2-B real world test.

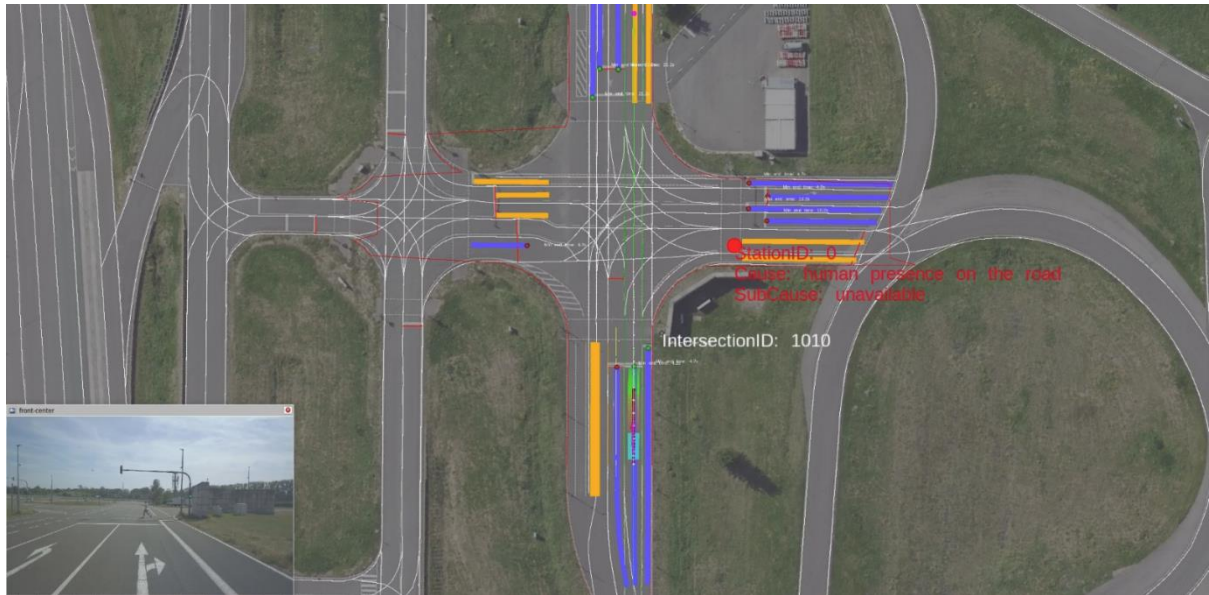


Figure 54: Digital view of UC1.2-B real world test.

Use Case 1.2-C tests the reaction of the vehicle to a reset of the traffic light cycle. In this case, the cycle is changed to red prematurely, and the vehicle has to react accordingly. Note that this change was triggered remotely by software and not by a pedestrian as shown in the use case description; however, this has no impact on the outcome of the scenario. An overview of the test can be seen in Figure 55. A digital snapshot of the situation at the time of the cycle change can be seen in Figure 56.



Figure 55: Overview of UC1.2-C real world test.

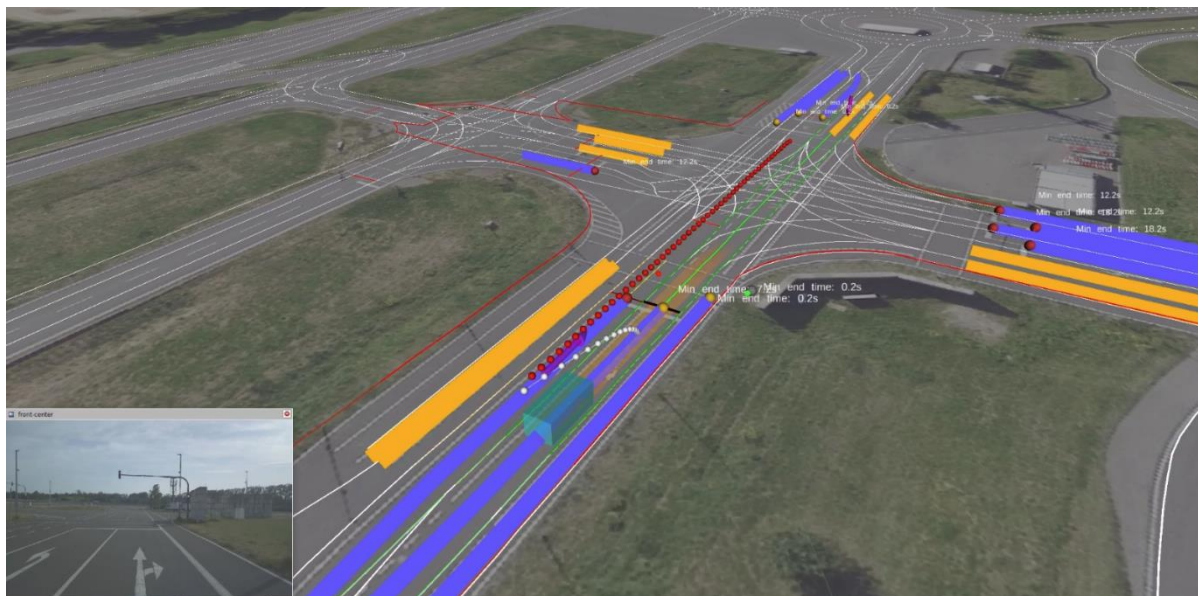


Figure 56: Digital view of UC1.2-C real world test.

Use Case 1.2-D tests the intrusion of a vehicle into the intersection while violating the traffic light. The CCAM vehicle has to react accordingly and prevent a collision. Note that the parameters cannot be selected as critically as in the simulated tests, in order to safeguard the safety drivers in both vehicles. An overview of the situation can be found in Figure 57. A digital view of the situation is shown in Figure 58.



Figure 57: Overview of UC1.2-D real world test.

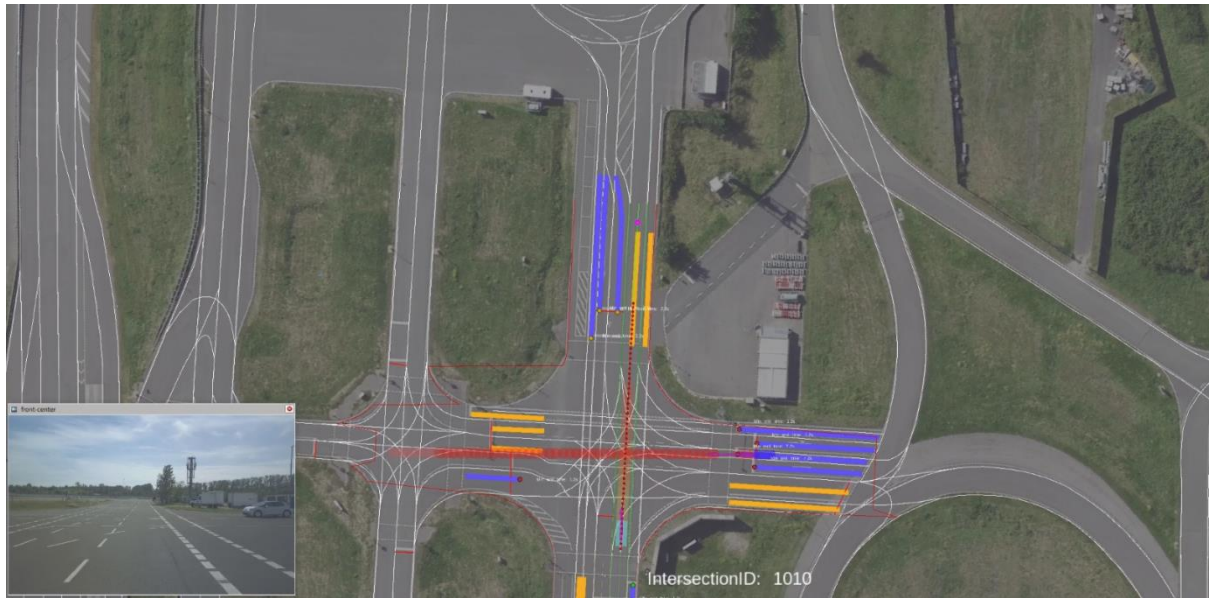


Figure 58: Digital view of UC1.2-D real world test.

3.2.3.4 Test Evaluate

Approach 1 (UNITN/ika)

Use Case 1.2-A and **1.2-D** were extensively employed to test concretion approaches developed as part of the SUNRISE Methodology (Validation criteria Section 2.2.2, c and d). An initial set of concrete scenarios was determined by Latin Hypercube Sampling. These initial scenarios were then evaluated in simulation. Based on the results, a Gaussian Process Classifier was used to generate a surrogate model of the evaluation metric for the scenarios (Validation criteria Section 2.2.2, e and 2.2.5, 2.2.6 c). An example visualization of this surrogate model can be seen in Figure 59 for sub-use case UC1.2-D.

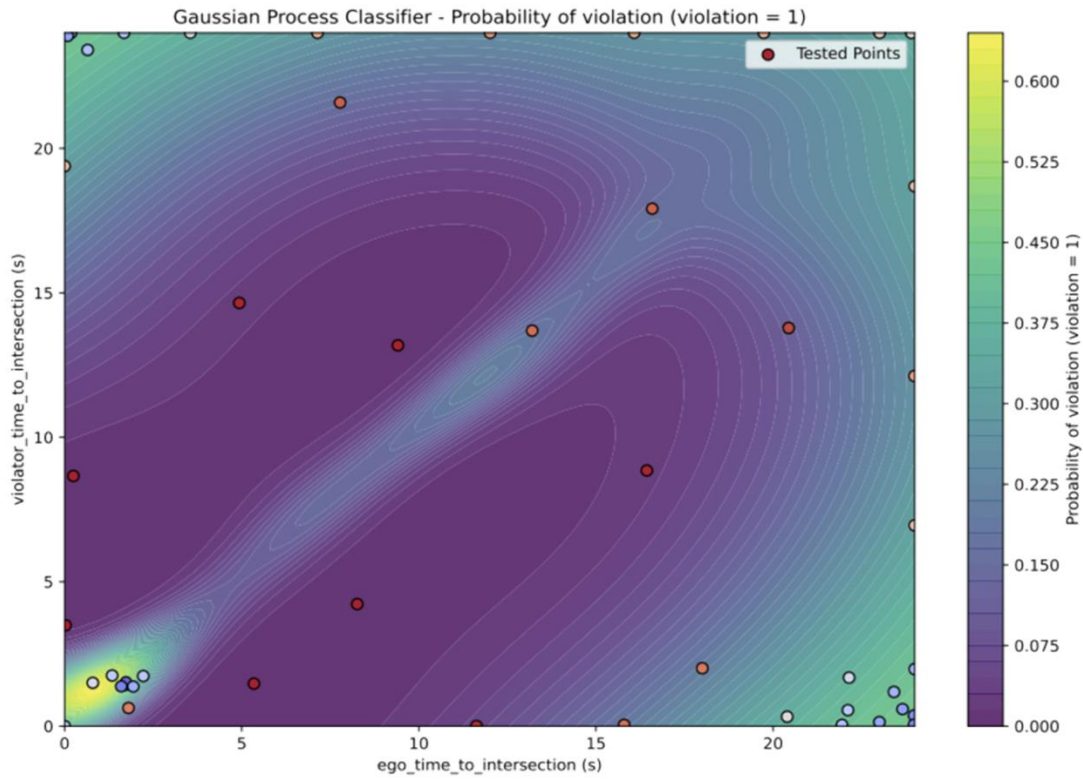


Figure 59: Surrogate model for UC 1.2-D.

Use Case 1.2-B was developed as follows. 1981 concrete scenarios with a TTI difference smaller than 4 seconds were selected from 50,000 random points and simulated in CarMaker. Results are shown Figure 60. (Validation criteria Section 2.2.2, c, d)

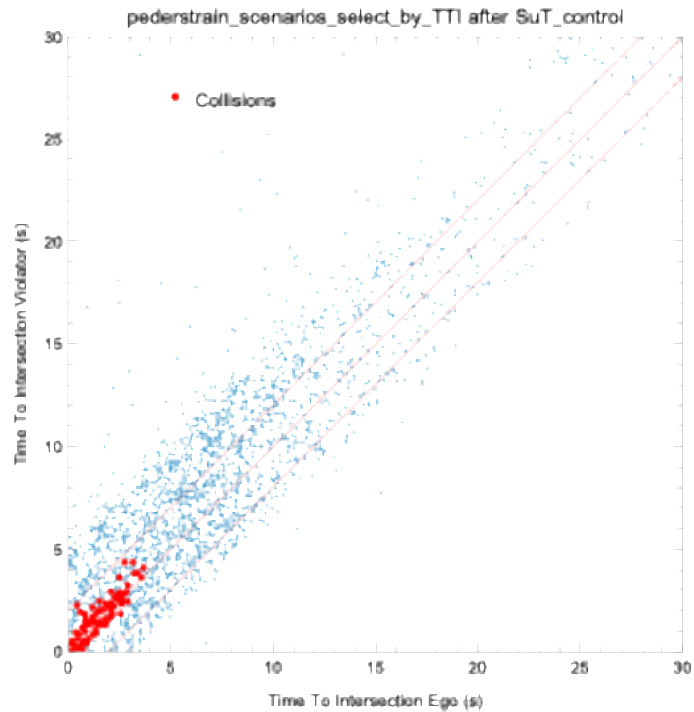


Figure 60: UC 1.2-B. First iteration in building a surrogate model.

A surrogate stochastic predictive neural model was trained (using the same methodology of UC3.2-C-1) and used to resample the logical scenario near the collision boundary (Validation criteria Section 2.2.2, e and 2.2.5). Then, 572 new concrete scenarios were created, as shown in Figure 61.

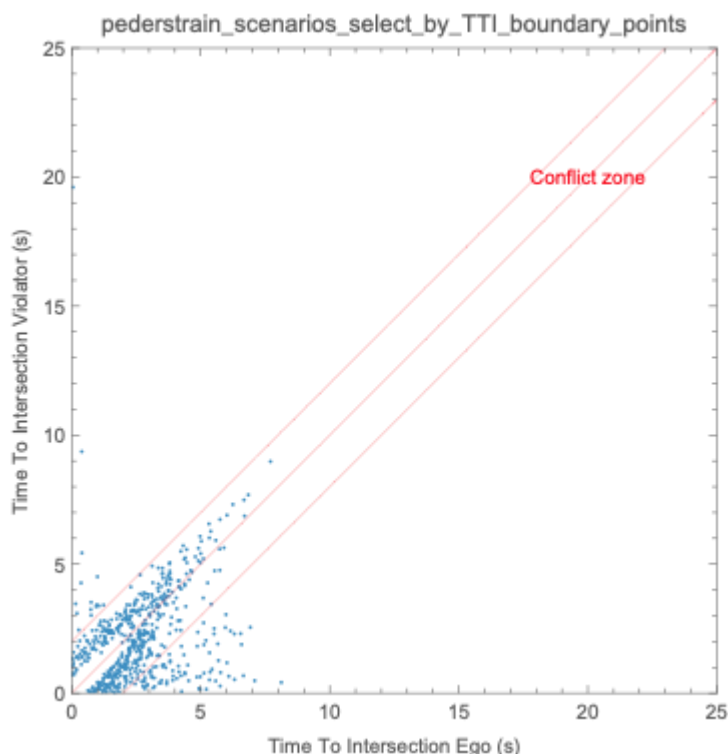


Figure 61: UC 1.2-B. Second iteration in building a surrogate model.

For all scenarios tested at ATC the system could successfully prevent collisions and traffic light violations, and thereby passed all tests based on the KPI's defined for the use case. Safety driver intervention was not needed for any scenario. Note that the system tested was different from the one used in simulated testing.

3.2.3.5 Coverage

Approach 1 (UNITN/IKA). Based on the surrogate model, in UC 1.2-A an optimization algorithm was used to determine additional sample points in regions with high uncertainty. These include points near the pass-fail boundary as well as points in regions of the parameter space that were not previously tested (Validation criteria Section 2.2.2, e and 2.2.5 a, b, c).

For UC 1.2-D, a comparison between random and LHS sampling was also carried out, Figure 62. The random sampling (left) identified 9 collisions with $TTI > 5$ s. The LHS sampling found 11 for $TTI > 5$ s (Validation criteria Section 2.2.2, e, 2.2.4 d and 2.2.5, 2.2.6 b). The horizontal black lines indicate successful evasive manoeuvres (from the initial TTI to an increased TTI by deceleration). Unfortunately, in some cases, the correction does not take place.

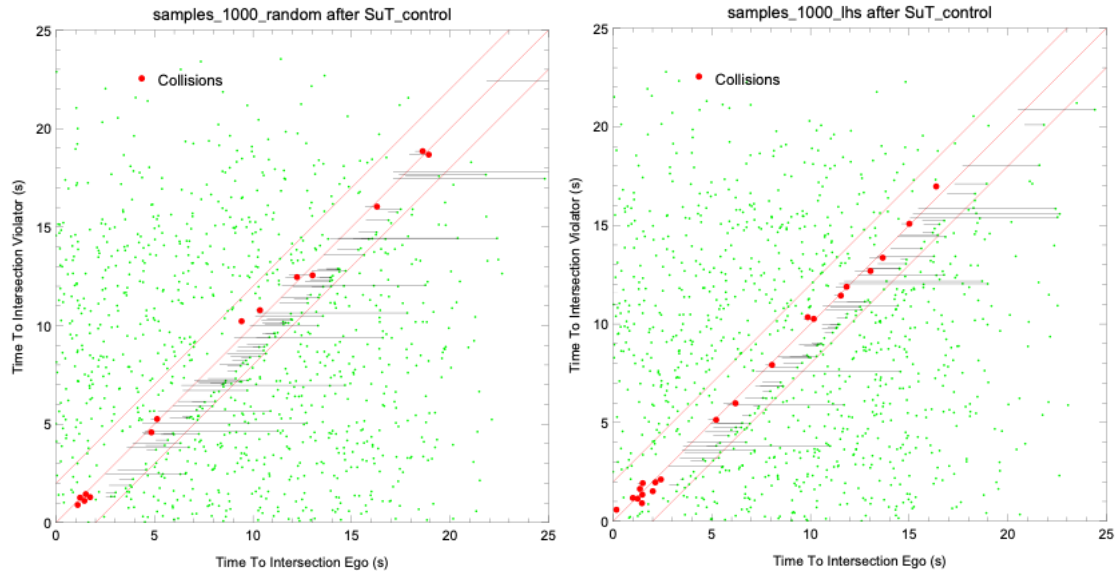


Figure 62: UC 1.2-D. The random sampling (left) identified 9 collisions with $TTI > 5$ s. The LHS sampling found 11 for $TTI > 5$ s. The horizontal black lines indicate successful evasive manoeuvres (from the initial TTI to an increased TTI by deceleration). Unfortunately, in some cases, the correction does not take place.

For UC 1.2-B, final surrogate model was trained on the 1981+572 balanced dataset. It provides fast predictions with confidence levels (Validation criteria Section 2.2.5 b). The model accuracy is shown in the confusion matrix below in Figure 63 (Validation criteria Section 2.2.6 b, c). The classifier is biased towards high recall (not missing collisions at the expense of false positives).

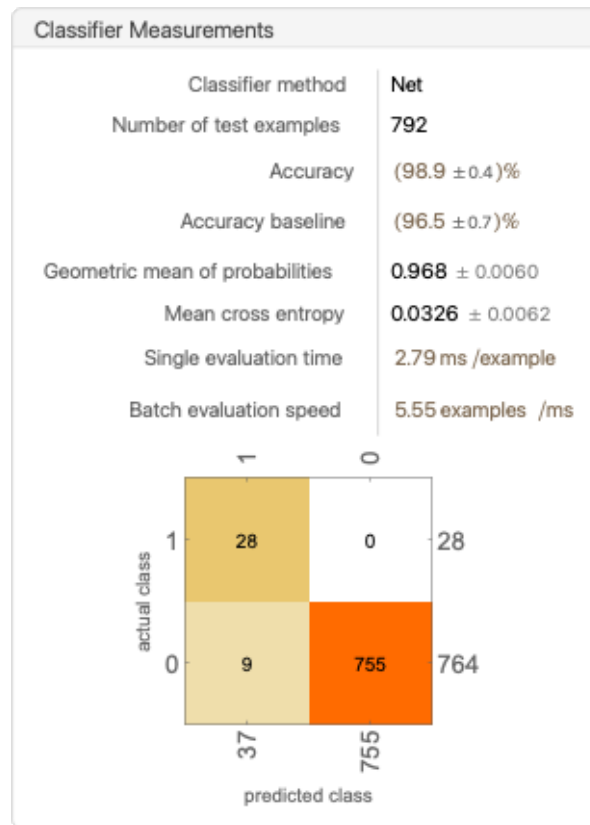


Figure 63: UC 1.2-B. Second iteration in building a surrogate model.

The surrogate model is used to identify unsafe conditions (Figure 64) for two risks: walking ($w < 5$ km/h, top) and running pedestrians (w up to 12 km/h).

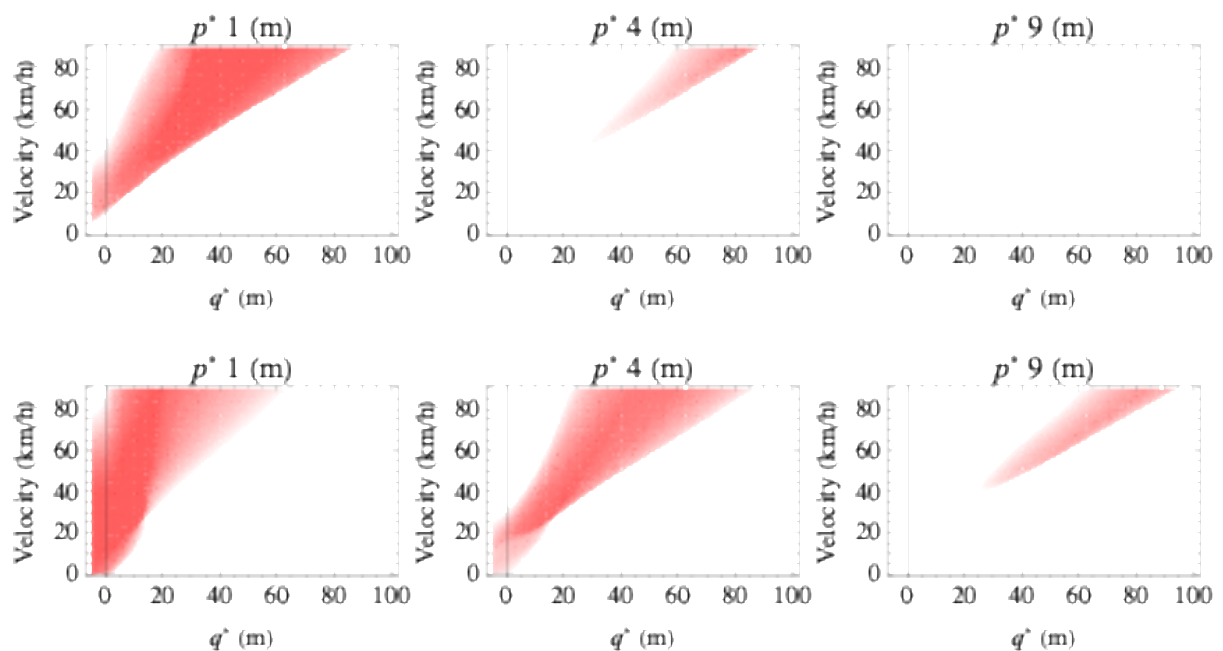


Figure 64: UC 1.2-B. Evaluation of unsafe regions. Top: for pedestrian at walking speed. Bottom, for running pedestrians.

3.2.3.6 Decide

Approach 1 (UNITN/IKA). No conclusion is given for UC 1.2-A.

The conclusion for UC1.2-B is as follows (Validation criteria Section 2.2.7, a-d): for pedestrians at walking speed there is a safe velocity (which depends on the pedestrian distance from the lane. However, for pedestrians running the safe speed exists only if the pedestrian starts running at a significant distance from the lane (which leave enough time to react).

The conclusion for UC 1.2-D can be drawn that **the system is not safe**. The reason is due to a defect in low level control that in some cases do not execute exactly the requested deceleration. This is confirmed by the surrogate model in Figure 62, which indicates non-zero collision risk for TTI > 5 seconds. (Validation criteria Section 2.2.7 c, 2.2.8 a: since there are collisions where there should not be, this is sufficient for the reject decision).

3.2.4 Key takeaways

A rejection decision is simpler than acceptance, because a single failure where no should happen is sufficient for rejection (UC 1.2-B, Approach 1).

Surrogate models, such as Approach 1 UC 1.2, are efficient tools for evaluating coverage and for near boundary resampling. They can also be statistically proved against additional test sets.

Surrogate models can be used to generate safe behaviors recommendations preventing to enter unsafe conditions.

Latin Hypercube Sampling performed slightly better than random sampling (UC1.2-D, Approach).

For intersection scenarios, sampling can be restricted to TTI differences smaller than 4-5 seconds.

3.2.5 Deviations from D7.2

Use Case 1.2-C was not executed because it can be reduced to 1.2-A as explained in Section 3.3.2.

3.3 VED Contribution for UC 1.2: Urban AD validation – Connected perception testing

3.3.1 Covered aspects of the SAF

VED Approach for Validation of UC 1.2

For the validation of **UC 1.2**, VED concentrates on **UC 1.2A** and **UC 1.2D** (as introduced in Section 3.2). The validation process covers four core **Safety Assurance Framework (SAF) blocks: allocation, execution, coverage, and test evaluation**, illustrated in Figure 65.

Allocation: VED maps the relevant elements of the system—**ADS functions, vehicle dynamics, connectivity, and environmental conditions**—to suitable **virtual and hybrid test setups**. This follows the methodology defined in Deliverable D3.3, ensuring that each element is allocated to the most appropriate testing environment.

Execution: Testing is carried out in two complementary ways. **Virtual simulations** are performed using **MATLAB/Simulink**, while **hybrid simulations** combine OMNeT++, Veins, and SUMO with real ADS components. This dual setup allows VED to replicate both controlled simulations and realistic hardware-in-the-loop conditions.

Coverage: VED ensures that the test campaigns include a wide range of scenarios, covering both **nominal operating conditions** and **edge cases**. This guarantees that ADS behavior is validated across diverse and challenging contexts.

Test Evaluation: Results from the tests are analyzed through **Key Performance Indicators (KPIs)**. By comparing outcomes from both virtual and hybrid setups, VED assesses the **consistency, robustness, and reliability** of ADS performance.

Through this structured approach, VED demonstrates how SAF components are applied to validate UC 1.2 in a rigorous and comprehensive manner.

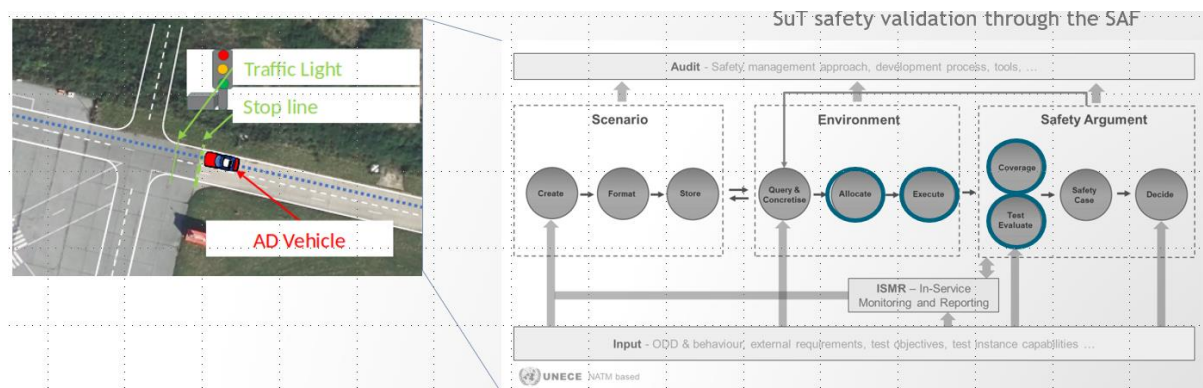


Figure 65: Diagram illustrating the SAF components (allocation, execution, coverage, test evaluation) that are validated by VED for Use Case 1.2D.

3.3.2 Overview of Tested Scenarios

VED focuses on UC1.2A and UC1.2D, as described earlier in section 3.2. From these logical use cases, we derive the following test scenarios:

From UC1.2A:

Scenario 111 – Nominal Stop: The ADS comes to a stop at a red traffic light.

Scenario 121 – Resume on Green: The ADS restarts and crosses the intersection once the traffic light turns green.

Scenario 134 – Delayed IMA Stop: The ADS stops before a red light due to a delayed Intersection Movement Assist (IMA) message.

From UC1.2D:

Scenario 211 – Stop for Priority Obstacle: The ADS stops for an obstacle vehicle approaching from the right, which has priority.

Scenario 221 – Urgent Stop for Non-Priority Obstacle: The ADS performs an emergency stop to avoid a non-priority obstacle vehicle coming from the left, but a collision occurs.

A detailed description of each scenario is provided below:

3.3.2.1 Scenario 111 – Nominal Stop :

The ADS detects a red traffic light and brings the vehicle to a safe stop before the intersection as illustrated in Figure 66.

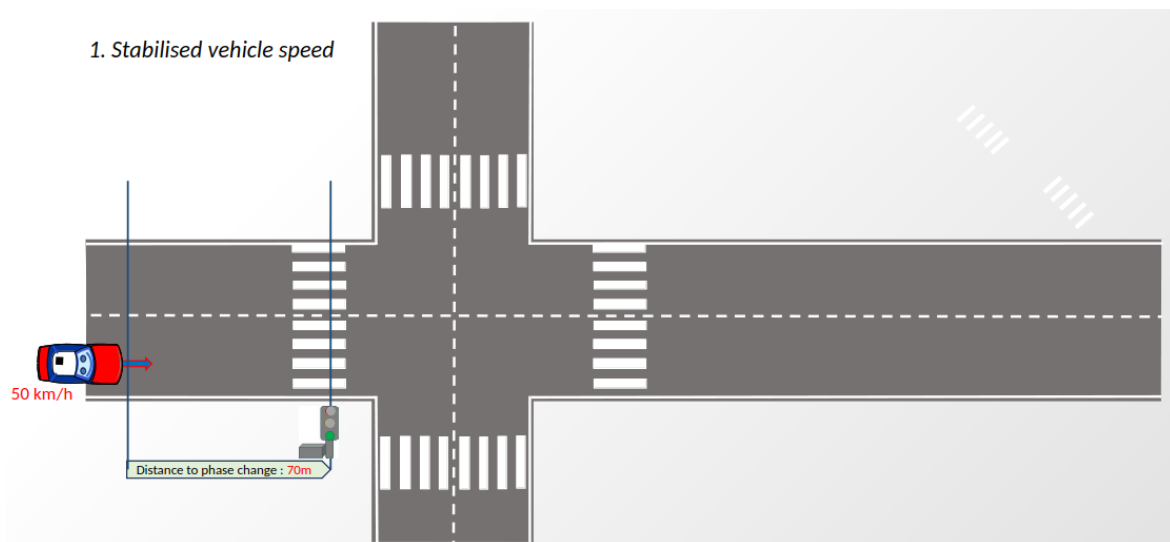


Figure 66: Scenario 111 – Nominal Stop illustration.

The objective of testing **Scenario 111** is to evaluate the ADS response when a traffic light changes from green to orange and then to red, based on varying initial vehicle speeds and distances from the traffic light at the moment the phase change occurs.

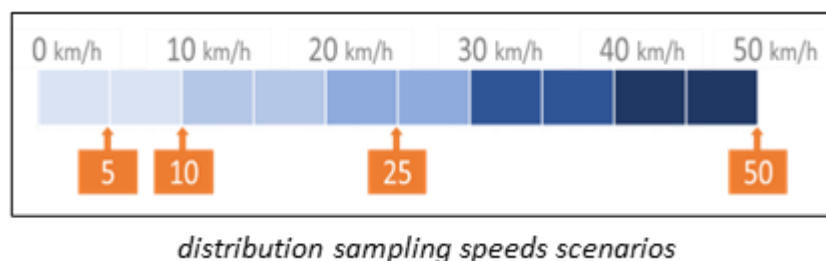


Figure 67: Initial ego vehicle speed.

Initial Vehicle Speeds:

High speed: 50 km/h

Medium speed: 25 km/h

Low speed: 10 km/h

Very low speed: 5 km/h

These reference speeds in the Figure 67, cover the full range of the vehicle's autonomous mode (0–50 km/h) and are chosen to assess how different speed profiles affect system behavior during calibration.

Distances to Traffic Light at Orange Phase Trigger:

Far: 70 m

Medium: 45 m

Close: 20 m

Zero: 0 m

Passed light: -3 m

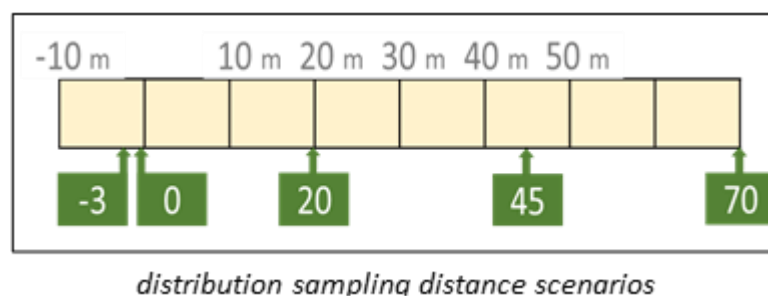


Figure 68: Distribution of sample of distance of the ego vehicle to the intersection.

These distances above and shown in Figure 68 reflect realistic urban driving conditions. The zero and negative distances are particularly useful for testing the ADS's ability to make appropriate stop decisions at low speeds, including edge cases where the vehicle is at or just beyond the stop line.

3.3.2.2 Scenario 121 – Resume on Green:

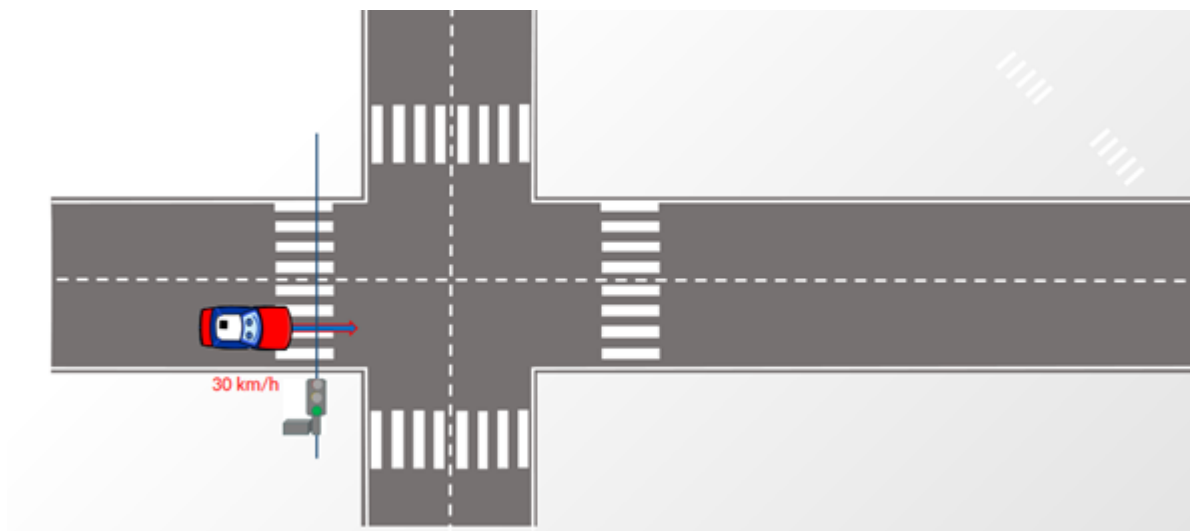


Figure 69: Scenario 121 – Resume on Green illustration.

In this scenario 121 illustrated in Figure 69, the ADS resumes motion and passes through an intersection when the traffic light turns green.

As the vehicle approaches a red light, the signal switches to green during braking—specifically when the vehicle's speed drops below a defined threshold. A single initial speed of **50 km/h** is used for all tests, as it is representative of relevant real-world driving conditions.

Initial Vehicle Speed:

High speed: 50 km/h

Green Phase Trigger Speed Thresholds shown in Figure 70:

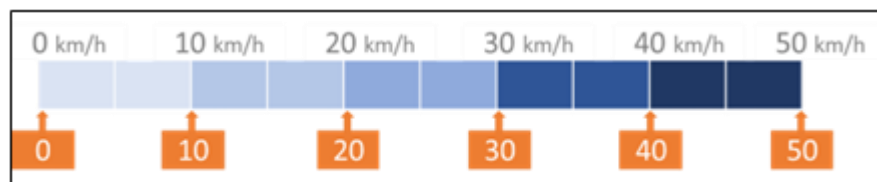
Very high threshold: 40 km/h

High threshold: 30 km/h

Medium threshold: 20 km/h

Low threshold: 10 km/h

Zero threshold: 0 km/h



distribution sampling speeds scenarios

Figure 70: Distribution of samples of speed.

This scenario evaluates how the ADS responds to green-light triggers at different braking points. The thresholds follow a 10 km/h step size, which is sufficient to capture meaningful differences in vehicle behavior. Finer granularity was deemed unnecessary, as it would not provide additional insights.

All parameters were validated through simulation to confirm their relevance and to ensure they produce measurable effects on the autonomous vehicle's response.

3.3.2.3 Scenario 134: ADS stops before and after the red light due to a delayed IMA message.

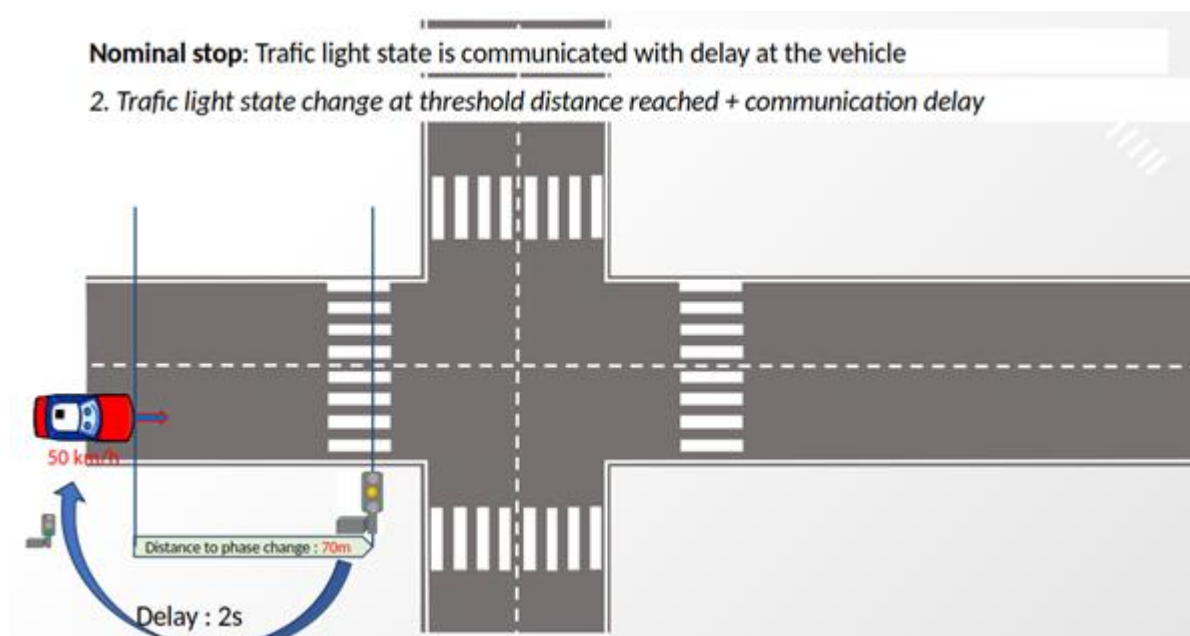


Figure 71: Scenario 134: ADS stops before a red light due to a delayed IMA message.

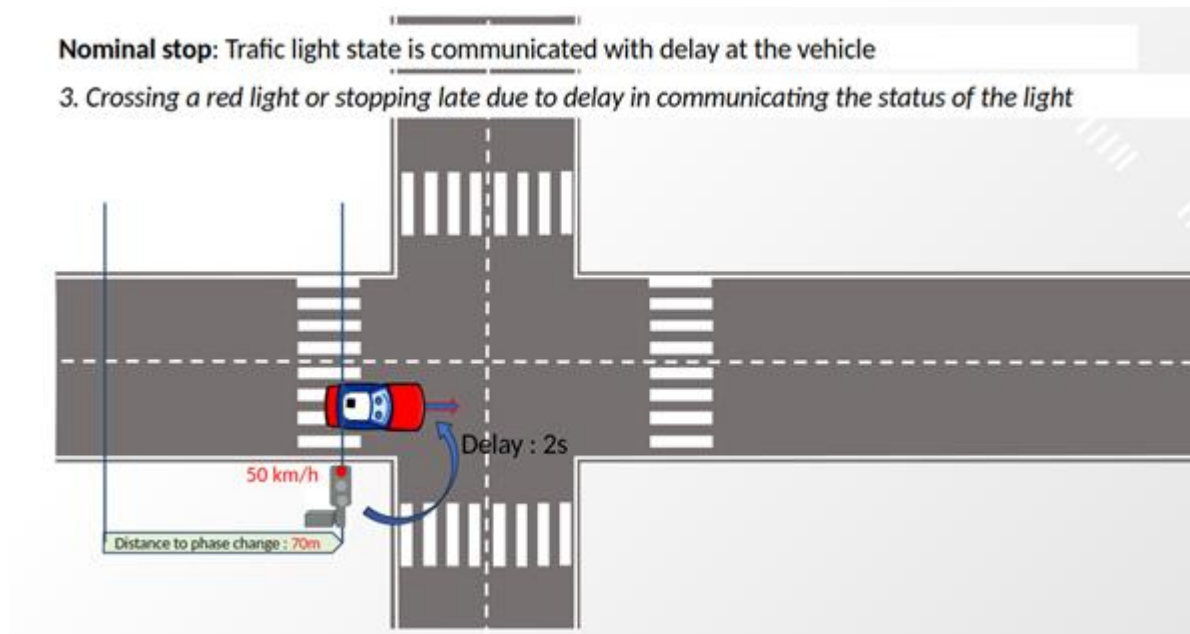


Figure 72: Scenario 134: ADS stops after the red light due to a delayed IMA message.

This scenario 134 illustrates in Figure 71 where ADS stops before red light and illustration in Figure 72 where the ADS stops after the red light. This is designed to evaluate how delays in receiving traffic light state information affect the behavior of the autonomous vehicle (AV), particularly during stop and resume phases.

Objective

To assess the impact of various communication delays in traffic light state updates on vehicle response, focusing on passenger comfort and control smoothness during braking and stopping.

Tested Cases

Two key use cases are considered:

- **Stopping at a red light** (nominal stop scenario)
- **Resuming after a green light**

Multiple simulation runs are performed to analyze how different delay durations influence the AV's control behavior.

Delay Sampling Strategy

Delays are chosen to represent different system layers, from software-level timing to full-vehicle dynamic response. The goal is not to exhaustively test all possible delays, but to identify one representative scenario for comparison with real-world vehicle tests.

Delay	Description
0 s	Ideal case, with no delay. Used as reference for baseline behavior.
0.1 s	Simulates timing issues within the AD software, e.g., slight desynchronization between modules (AD typically runs at 0.04 s cycles). Reflects connectivity-induced latency or software jitter.
0.3 s	Represents sensor processing or actuator latency (e.g., brake pressure buildup). Comparable to low-level physical system delays.
0.5 s	Corresponds to the response time of the full vehicle dynamics and longitudinal controllers. Captures limitations due to vehicle inertia or during emergency braking.
1 s	Reflects human-like reaction times and slower actuators (e.g., electric parking brake). Also models control system anticipation windows.
2 s	Represents planning and decision-making time horizons in high-level AV modules.

Selection of Test Scenario

Based on simulation outcomes, the **red light stop** scenario was selected over the green light resumption case. The stopping scenario exhibited greater sensitivity to delay, particularly in terms of deceleration behavior and passenger comfort, making it more effective for detecting discrepancies between simulated and real-world results.

Selected scenario parameters:

Trigger point: Red light activation at 30 meters before the intersection

Initial speed: 30 km/h

This setup aligns well with physical test track constraints. Alternative configurations (e.g., red light at 70 meters with 50 km/h) require a longer distance to allow stabilization in autonomous mode, which is not feasible within our current testing environment.

In the hybrid testing, the red light stop scenario at 30 km/h and 30 m distance will be executed using the predefined set of delay values. This will enable a comprehensive assessment of delay-induced effects on the AV's stopping behavior, control accuracy, and passenger comfort.

3.3.2.4 Scenario 211: ADS stops for a priority obstacle vehicle approaching from the right.

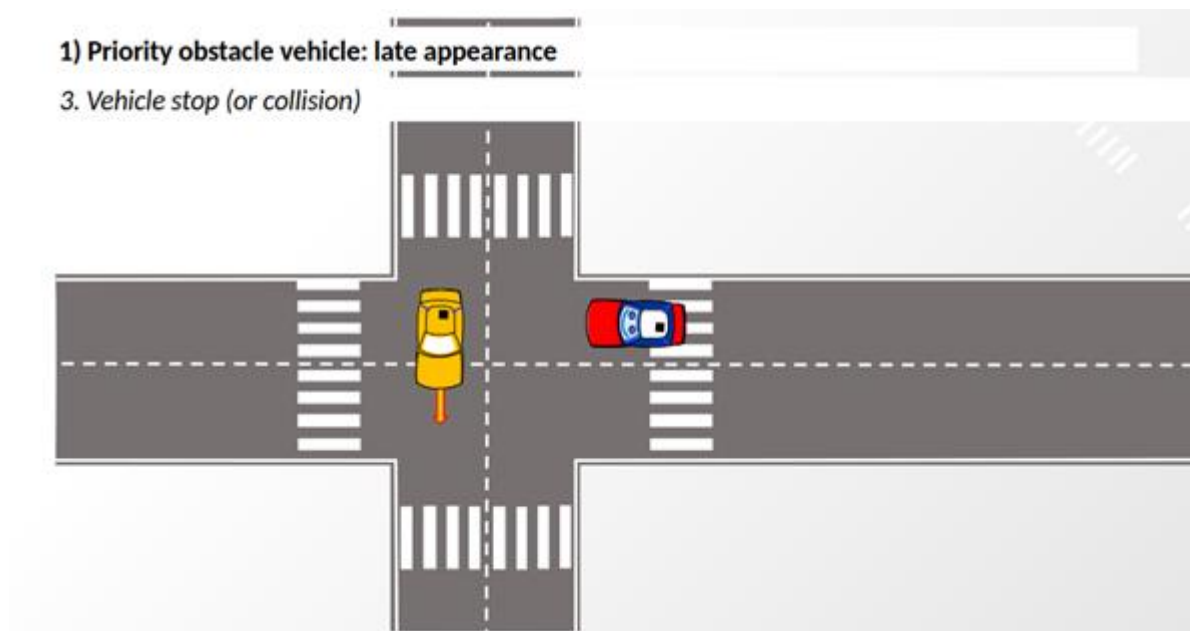


Figure 73: Scenario 211: Illustration of ADS stops for a priority obstacle vehicle approaching from the right.

3.3.2.5 Scenario 211 – Priority Obstacle at Intersection

This scenario 211 illustrated in Figure 73, evaluates the autonomous vehicle's response and passenger comfort when a **priority obstacle vehicle** appears at an intersection.

Test Parameters

Simulations are conducted with the ego vehicle at two distances from the intersection entry point:

10 meters

25 meters

The ego vehicle approaches at varying initial speeds:

10 km/h, 20 km/h, 30 km/h, 40 km/h, and 50 km/h

The priority obstacle vehicle appears **20 meters before** its own entry into the intersection, traveling at a constant speed of **30 km/h**—equivalent to entering the intersection approximately **2.5 seconds** after detection.

Objective

The goal is to assess how the ego vehicle reacts—both in terms of behavior and comfort—when a priority vehicle suddenly appears, across a range of approach speeds and distances.

Critical Timing Scenarios

The combination of speed and distance creates a range of critical intersection approach situations, where the **time to intersection** varies from:

0.72 seconds (10 m at 50 km/h) to **9 seconds** (25 m at 10 km/h)

Sampling is finer at lower speeds and shorter distances, as these configurations are more critical and sensitive.

Scenario Design

All speed and distance parameters were **tested and refined through simulation**.

Scenarios were carefully selected to ensure they have a **significant impact** on the AV's behavior.

The number of test cases is intentionally limited to focus on the **most representative and critical real-world situations**.

3.3.2.6 Scenario 221 – Urgent Stop for Non-Priority Obstacle (Collision Case)

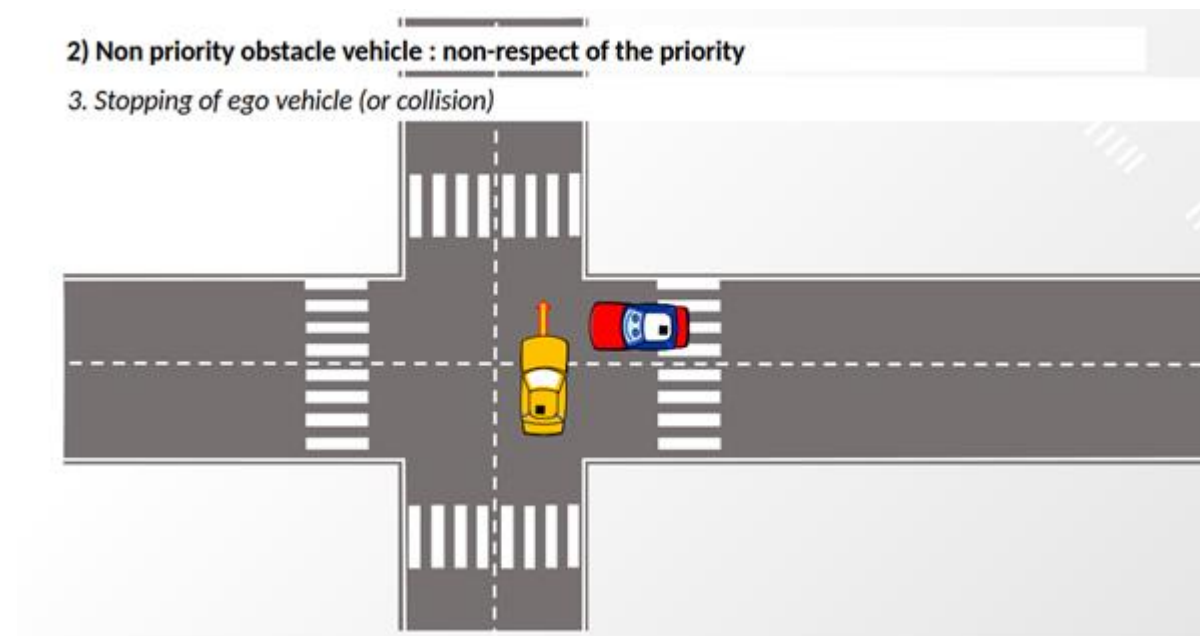


Figure 74: Scenario 221 – Urgent Stop for Non-Priority Obstacle (Collision Case) illustration

This scenario 221 illustrated in Figure 74 assesses the behavior of the autonomous driving system (ADS) when it must perform an urgent stop to avoid a **non-priority obstacle vehicle** approaching from the **left** at an intersection. Despite braking, the situation results in a **collision**, making it a critical test case for evaluating risk handling, reaction timing, and vehicle dynamics under pressure.

Objective

To evaluate how the ADS handles last-moment decision-making and emergency braking when faced with a non-priority vehicle that violates the intersection rules. The focus is on:

- Collision avoidance behavior
- Control system response time
- Passenger safety and comfort
- Failure case handling

Test Parameters

Simulations are run with the ego vehicle approaching the intersection at varying **distances** and **initial speeds**:

Distances from intersection:

- 5 meters**
- 10 meters**
- 25 meters**

Initial speeds:

- 10 km/h**
- 20 km/h**
- 30 km/h**
- 40 km/h**
- 50 km/h**

These combinations produce a wide range of time-to-intersection values, creating both avoidable and unavoidably late reaction cases.

Critical Timing Range

The **remaining time before intersection entry** spans:

- 1.8 seconds:** Extremely late scenario (5 m at 50 km/h), where a collision is inevitable
- 9 seconds:** Ample reaction time (25 m at 10 km/h), where avoidance is feasible

Sampling becomes **denser at lower speeds and shorter distances**, where decision complexity and risk are highest. This ensures detailed evaluation of system behavior in borderline or high-stress situations.

Scenario Design Strategy

All speed–distance combinations were **simulated, tested, and calibrated** to reflect realistic and impactful traffic situations.

Test cases are **limited but strategically selected** to maximize insight into ADS behavior. Emphasis is placed on scenarios where behavior divergence is most likely, enabling meaningful comparison between simulation and real-world outcomes.

These scenarios were tested in both virtual (MATLAB/Simulink) and hybrid (OMNeT++/Veins/SUMO with real ADS components) environments, which are described in the following.

3.3.2.7 Virtual Testing Framework:

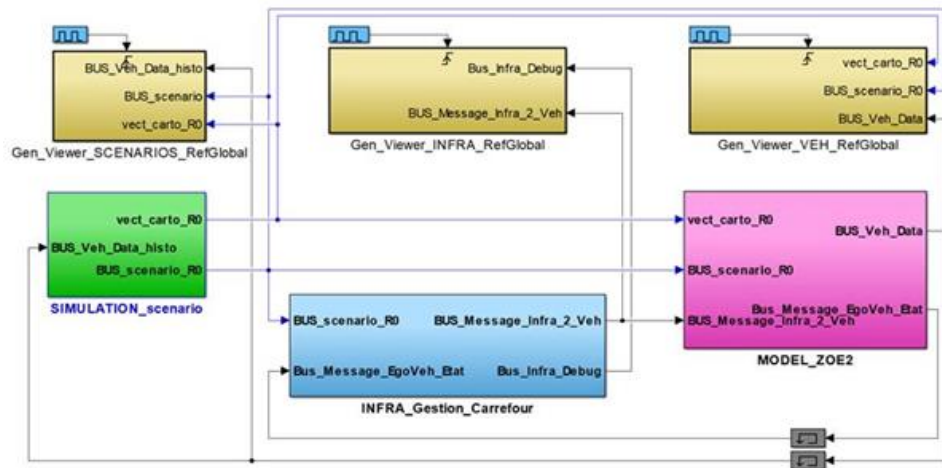


Figure 75: Virtual testing framework architecture using MATLAB and SIMULINK.

Development & Simulation Tools

Matlab/Simulink

Used for designing control and perception algorithms, modeling vehicle dynamics, and running system-level simulations.

RTMaps (Real-Time Multisensor Applications)

A middleware platform for real-time deployment of perception and control algorithms on both the physical vehicle and roadside infrastructure.

C/C++ Code Generation

Simulink models are converted into C/C++ code for portability:

S-Functions for integration in simulation environments

DLLs for deployment on real-time systems

Visualization Interfaces

Custom real-time 3D visualization tools enable scenario playback and monitoring from multiple viewpoints.

Main Components and Their Functions shown in *Figure 75*.

1. Real-Time Visualization

Ego Vehicle View: Visual feedback from the vehicle's onboard sensors and systems

Intersection Manager View: Displays decisions made by roadside control units

Simulator View: Global scenario overview for debugging and analysis

2. Scenario Generation

Dynamic creation and control of:

Obstacles (vehicles, pedestrians, etc.)

Traffic Lights (state changes, timing)

Ego Vehicle Behavior: Includes manual driving phase before switching to autonomous control

3. Intersection Manager

Hosts compiled **control algorithms** as dynamic libraries (DLLs)

Deployed on the **physical roadside unit (RSU)**

Coordinates intersection logic and communicates with the ego vehicle

4. Ego Vehicle

Incorporates a **vehicle dynamics model** for realistic motion behavior

Executes **autonomous driving algorithms** (compiled as DLLs)

Runs on the **onboard in-vehicle computer** for real-time operation

Intersection Manager – Enhancing Collision Risk Awareness

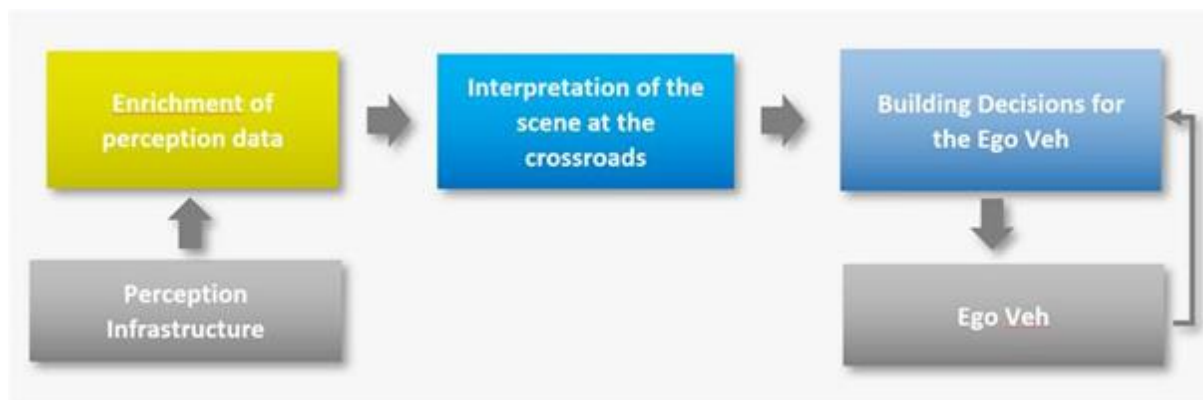


Figure 76: Intersection manager architecture.

The **Intersection Manager** in Figure 76 enhances collision risk awareness by providing the autonomous vehicle with an extended, infrastructure-based view of the intersection environment. Using fixed sensors, it avoids perception limitations caused by vehicle motion and benefits from a strategically elevated placement that offers a global perspective, often surpassing the vehicle's onboard perception. Its data processing pipeline includes two key stages: **perception enhancement**, where it estimates object speed, analyzes heading (even for seemingly static objects), and performs temporal tracking to manage brief occlusions; and **scene interpretation**, where it links detected objects to road topology (such as lanes, crosswalks, and intersections), applies traffic priority rules, and identifies collision risks specifically related to the ego vehicle's intended path.

Ego Vehicle Model – For AD Algorithm Validation

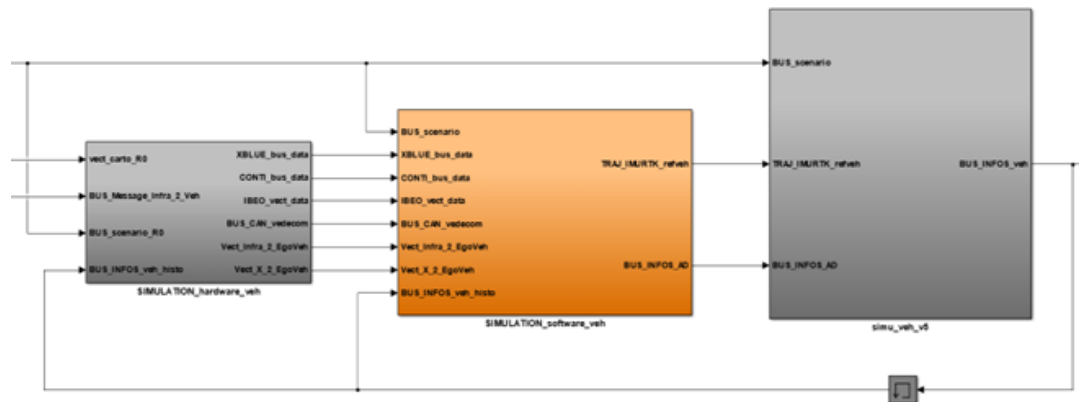


Figure 77: Ego vehicle model in control simulator.

The **Ego Vehicle Model** in Figure 77 is designed to validate VEDECOM's autonomous driving (AD) stack by simulating vehicle dynamics, manual driving behavior, and the full range of onboard sensor and actuator systems.

It incorporates detailed **sensor models**, including IMU and RTK for localization, a lane marking detection camera, an Ibeo lidar belt, and CAN-based signals like wheel speed, yaw rate, accelerometer data, and steering angle.

The **VEDECOM AD stack** integrated into the simulation includes topological map matching with elements such as traffic lights and risk zones, obstacle detection and tracking, as well as decision-making, planning, and control algorithms.

To ensure realistic responses, the model also simulates **driving and dynamics**, including a manual driver mode for transition testing, a low-level robotization controller, and a detailed Renault Zoé vehicle model that accounts for actuator latencies and control constraints such as torque and steering limits.

3.3.2.8 Hybrid Testing Framework

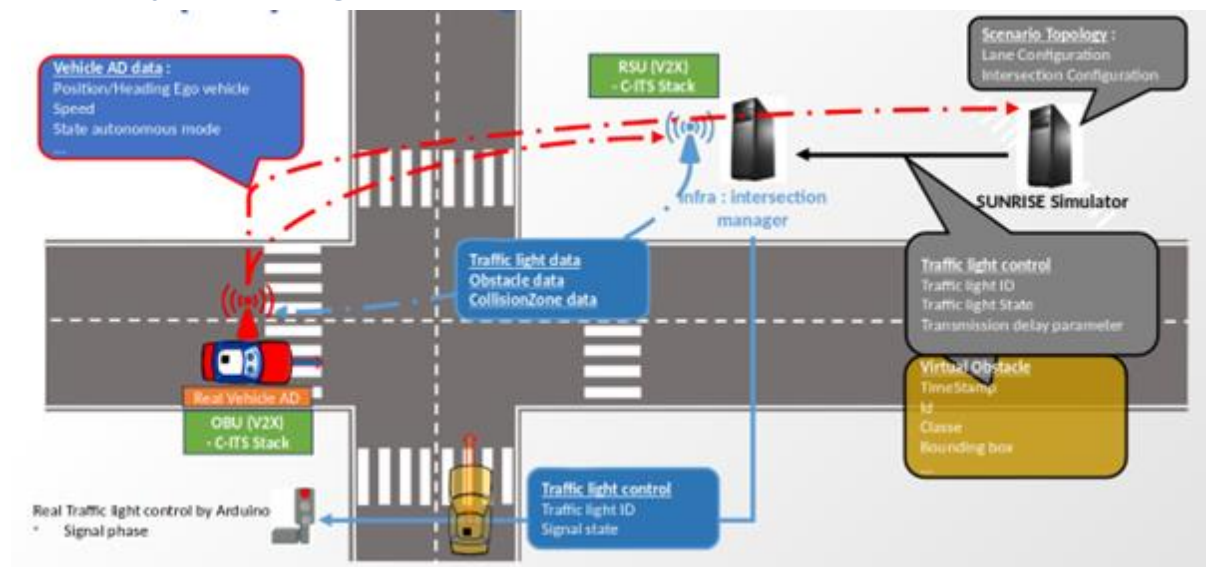


Figure 78: Hybrid testing framework.

Hybrid Testing Framework

The hybrid testing framework shown in Figure 78 combines advanced simulation tools with real-world components to validate cooperative autonomous driving scenarios under realistic conditions.

Development & Simulation Tools:

OMNeT++ serves as the core discrete-event simulation engine, enabling detailed modelling of network and system behaviors in C++.

Veins integrates with OMNeT++ and SUMO to simulate V2X communications, handling Cooperative Awareness Messages (CAM), Collective Perception Messages (CPM), and perception modeling.

SUMO provides realistic traffic and mobility simulation, modeling vehicle behaviors, traffic flow, and dynamic interactions.

An **MQTT Broker** acts as the middleware interface between simulated components and real-world systems, ensuring seamless data exchange and synchronization.

Real-World Components:

Testing is conducted on a **real test track** marked with standard road features to mimic urban intersections.

A **real autonomous vehicle** equipped with VEDECOM's AD stack enables real-time deployment and testing of autonomous decision-making and control.

A **physical traffic light**, controlled via Arduino, adds to the realism of intersection behavior.

V2X connectivity is provided by VEDECOM's ITSC communication stack, with real **OBU (On-Board Unit)** and **RSU (Road-Side Unit)** deployed to enable cooperative perception and decision-making.

Visualization Interfaces:

The **SUMO GUI** provides a live 2D representation of traffic dynamics, vehicle paths, and infrastructure status.

Custom real-time 3D viewers allow immersive visualization from multiple viewpoints, including the ego vehicle, the intersection manager, and a global overview of the entire simulation scenario.

This integrated approach ensures that algorithm development, simulation, and physical validation work together to accelerate the testing and deployment of cooperative automated driving solutions.

3.3.3 SAF Block Demonstrations

3.3.3.1 Allocate

The validation of the 'Allocate' block within the SAF begins with a thorough analysis of the test scenario's core requirements. These span across environmental complexity, expected vehicle behaviors, communication demands, and the intricacies of perception tasks. Each requirement is then evaluated against the capabilities provided by our hybrid testing framework in Table 11.

Simulated testing is appropriate for aspects such as the static road network, traffic infrastructure, and typical behaviors of non-subject vehicles and pedestrians. Hybrid testing is reserved for components requiring higher fidelity or real-time constraints—such as the ego vehicle's AD stack, V2X communications (e.g., CAM/CPM/SPATEM), and sensor/perception systems. Real components are incorporated when physical hardware or real infrastructure is essential to ensure the accuracy and integrity of the validation process.

Table 11: Allocate validation, scenario requirements and virtual testing framework capabilities comparison.

Requirement	Capability	Remarks
Urban road network, traffic light	Matlab/Simulink simulates the roads network and traffic agents	Match
Dynamic elements	Matlab/Simulink simulates dynamic traffic agents	Match
Subject vehicle AD function.	Matlab/Simulink simulates the AD functions.	Partially met → XiL
V2X communication	Not simulated. Direct information transfer.	Test in XiL framework

(Spatem, CAM CPM)		
Perception	Not simulated. Direct information obtained from the simulator	Test in XiL framework

Through this comparison, we determine the most suitable testing method for each scenario component—virtual simulation, hybrid (e.g., XiL), or real-world execution. This process results in an allocation structure diagram Figure 79 that organizes each element based on how it should be tested.

In this diagram:

Blue indicates elements tested in a fully simulated environment.

Gold represents components validated using hybrid testing methods like Software-in-the-Loop (SiL) or Hardware-in-the-Loop (HiL).

Red highlights real-world components integrated into the test setup.

This allocation strategy ensures each system element is tested with the right level of realism, enabling a safe, efficient, and reliable validation process that supports the overall safety assurance goals.

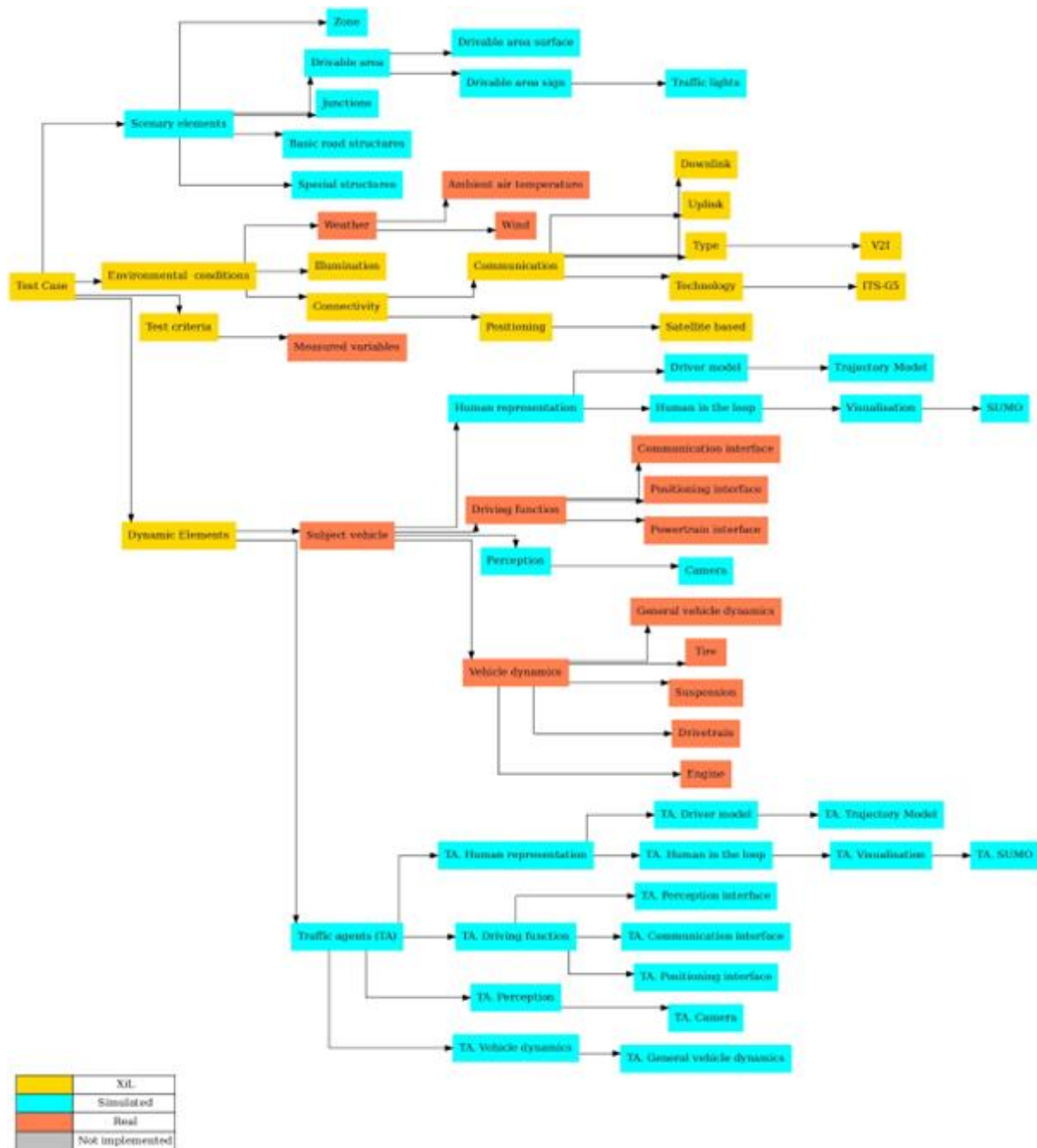


Figure 79: UC 1.2 Allocate validation: final allocate structure diagram obtained highlighting the components allocated to different testing frameworks.

3.3.3.2 Execute

We validated the *Execute* block through testing with the hybrid setup. A demonstration video of this validation is available in the Sunrise Handbook and can be accessed here:

https://www.youtube.com/watch?v=12GBcRWN2jU&t=1s&ab_channel=ERTICO-ITSEurope.

Below, we present the results of the *Coverage Validation* and *Test Evaluation* phases.

3.3.3.3 Coverage

For Scenario 111, which represents a nominal stop use case tested in a virtual environment, we analyzed the vehicle's decision-making during amber light phases.

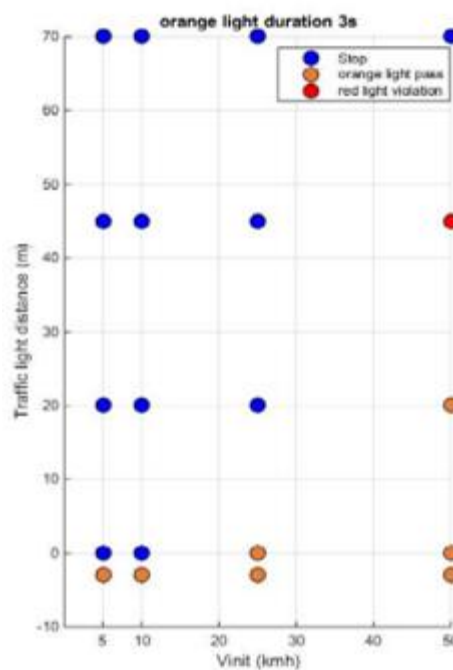


Figure 80: Traffic light violation for scenario 111 for orange light duration of 3s.

Figure 80 illustrates the vehicle's stop/go decisions during a 3-second amber light. While most actions align with expected behavior, one instance of red-light violation (crossing 0.2 seconds after the light turned red) highlights the impact of tuning the decision model for a peri-urban environment, where amber durations are typically 5 seconds.

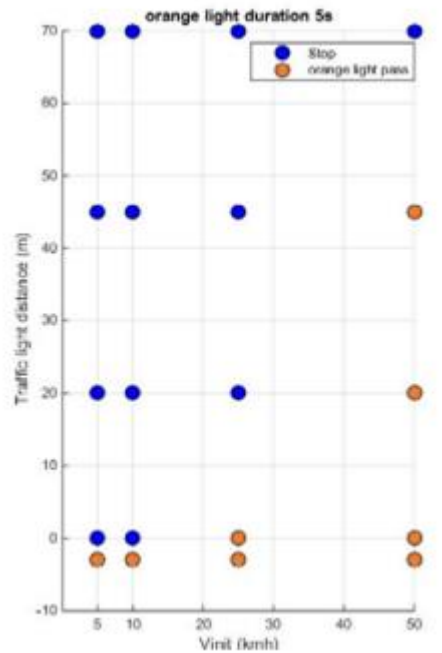


Figure 81: Traffic light violation for scenario 111 for orange light duration 5s.

Figure 81 presents the vehicle's behavior under a 5-second amber phase. Here, all decisions to proceed before the transition to red, reflecting a more robust compliance with traffic rules under adjusted timing conditions.

Our experimental focus was on cases where the vehicle chose to decelerate and stop. This subset of data—represented by the blue points in the graphs—was analyzed to assess the comfort and smoothness of the stop, a key factor in evaluating passenger experience and control logic performance.

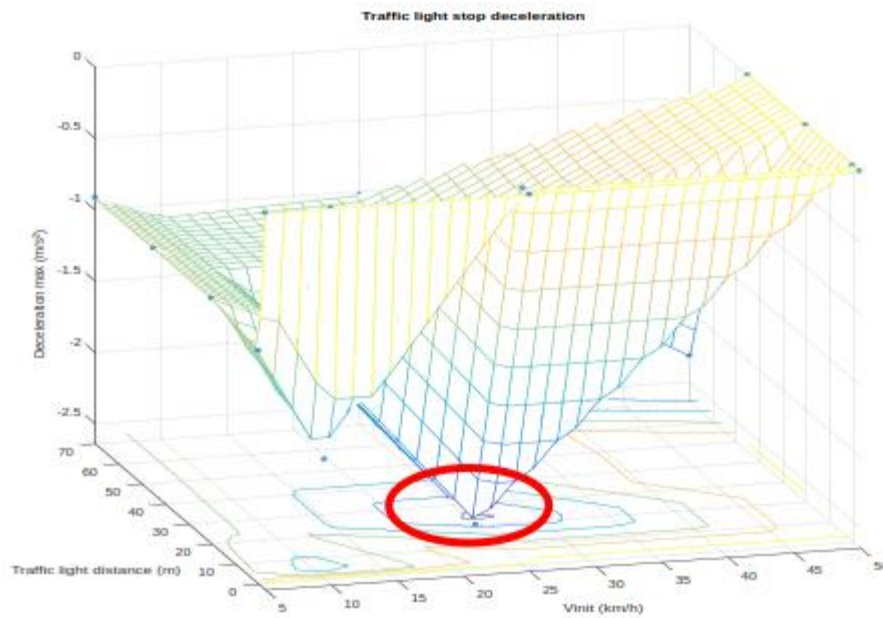


Figure 82: Traffic light stop maximum deceleration.

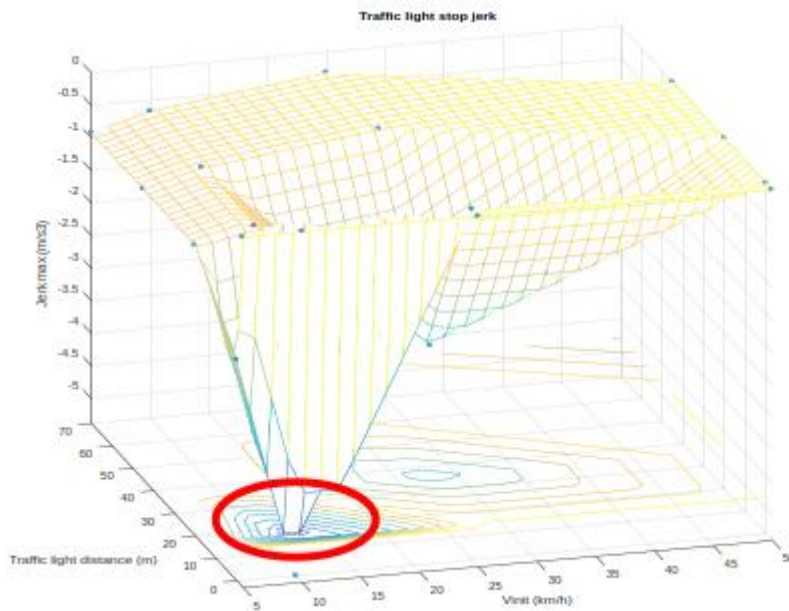


Figure 83: Traffic light stop maximum jerk.

Figure 82 and Figure 83 illustrate the peak deceleration and jerk experienced by the Ego Vehicle during braking across all MATLAB simulations. These metrics are key indicators of braking intensity and ride comfort.

Among the tested scenarios, the one selected for real-world validation involves a traffic light changing as the vehicle is exactly at the stop line, traveling at 10 km/h. While this case does not result in the highest deceleration, it produces the highest peak jerk—highlighting it as the most uncomfortable stop. This makes it a critical edge case for assessing the braking comfort and the vehicle’s behavior when forced to make a last-moment decision to stop.

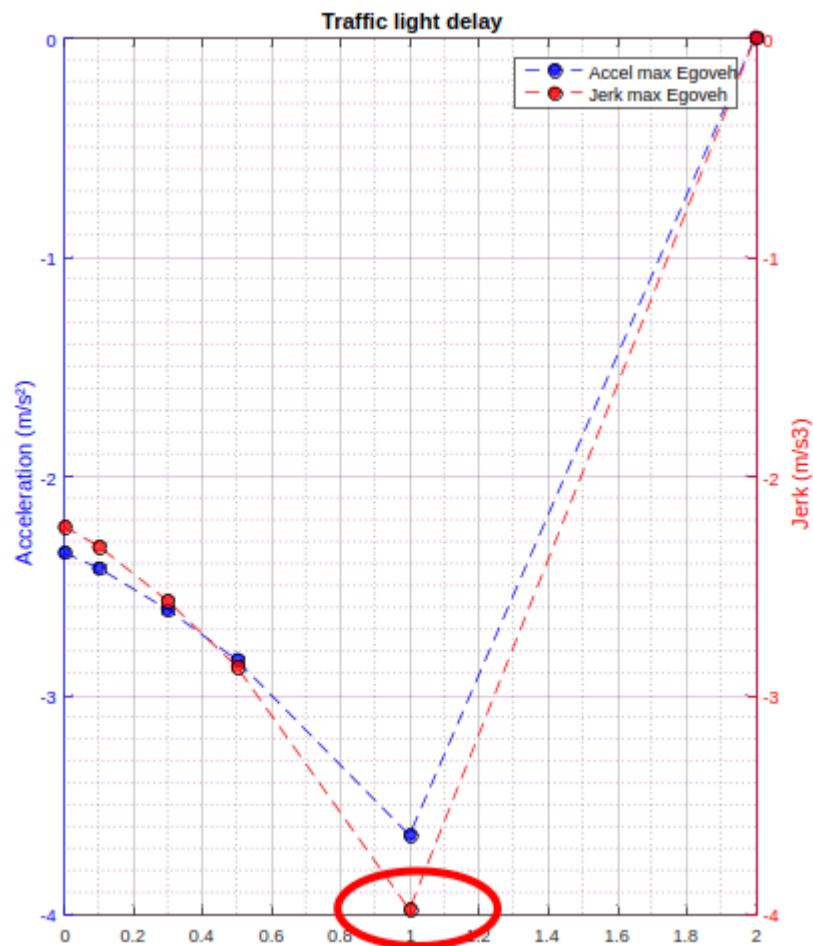


Figure 84: Traffic light state delay, acceleration, and jerk.

Maximal jerk experienced by EgoVeh passengers:

Figure 84 illustrates the maximum acceleration and jerk experienced by passengers in the Ego Vehicle during scenarios where the vehicle stops at a red light with delayed communication of the traffic light phase. The variations observed across different delay intervals reveal how communication latency impacts braking behavior and passenger comfort.

Among these, the scenario with a 1-second delay in receiving the traffic light phase was identified as the most critical for evaluating comfort, as it results in a sharp increase in jerk while still allowing the vehicle to stop in time. This case was selected for real-world

experimentation. Conversely, the 2-second delay scenario was excluded from further analysis, as the vehicle fails to stop before the light, rendering it unsuitable for comfort evaluation.

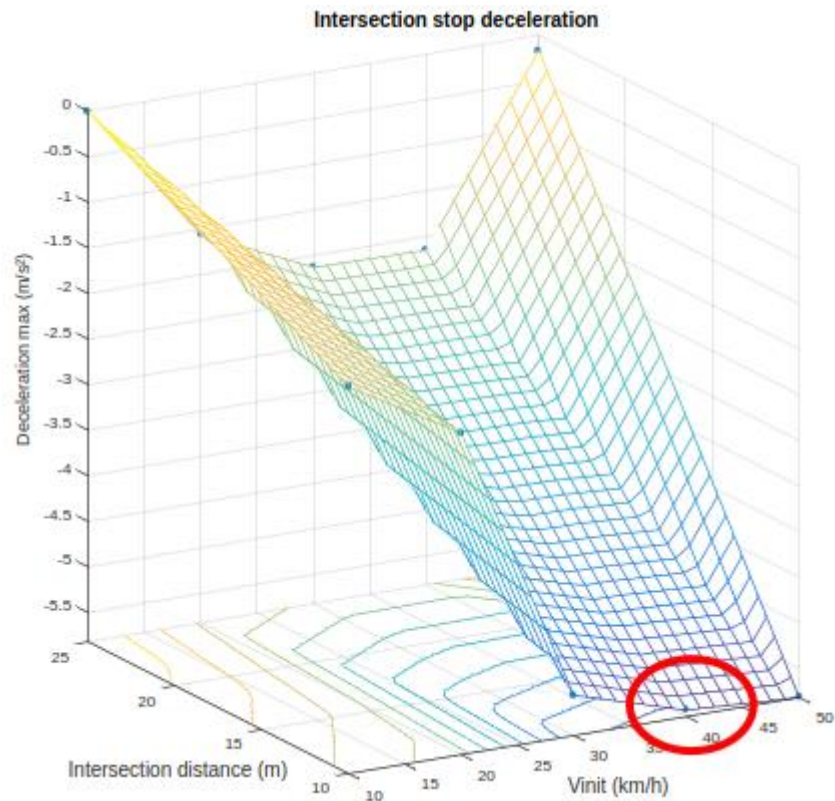


Figure 85: Intersection deceleration max.

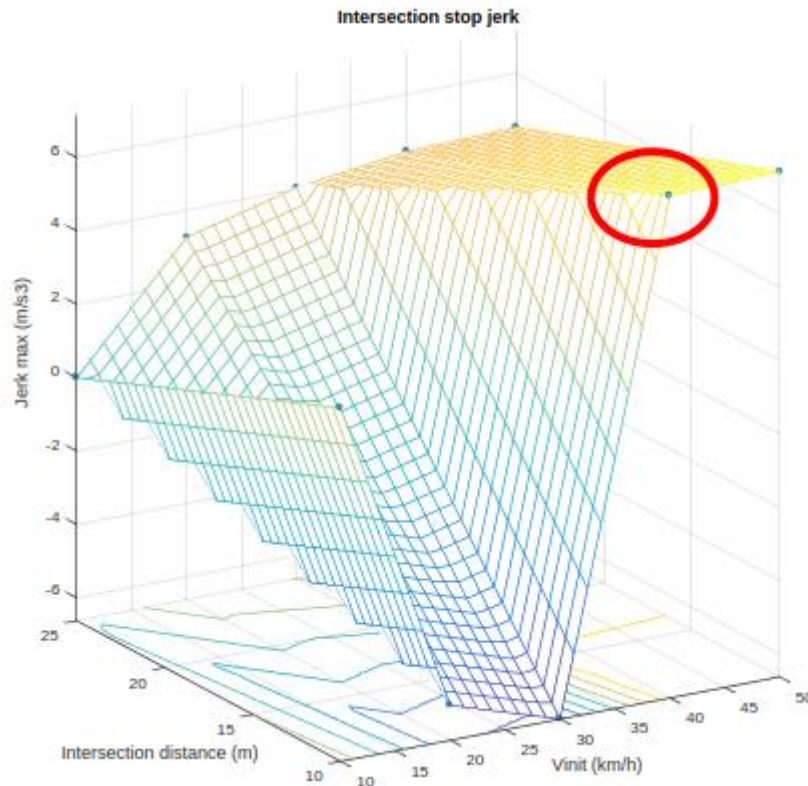


Figure 86: Intersection maximum jerk.

Deceleration and Maximum Jerk Experienced by Passengers of the Ego Vehicle

Figure 85 and Figure 86 present the maximum accelerations and jerks experienced by passengers during a stop at an intersection triggered by the sudden appearance of a priority obstacle. Among the various simulated cases, the most critical in terms of discomfort occurs when the Ego vehicle is 10 meters from the intersection and traveling at 50 km/h. However, for practical reasons, the scenario chosen for real-world experimentation involves the same distance but a speed of 40 km/h. This is because the peak deceleration (-5.75 vs -5.82 m/s^2) and jerk (7.06 vs 7.14 m/s^3) are nearly identical in both cases, while 40 km/h is more feasible to achieve consistently under autonomous driving conditions on our test track.

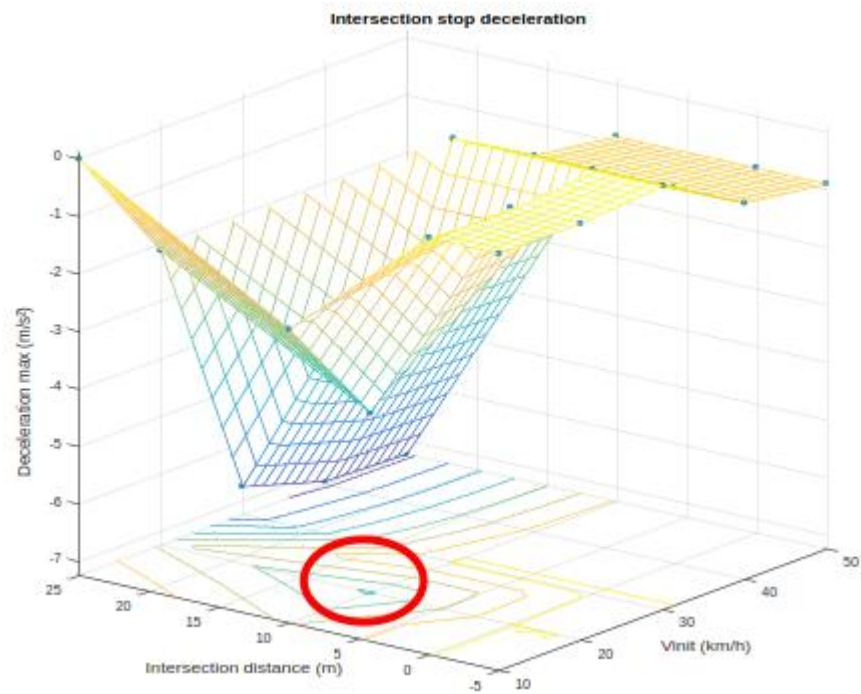


Figure 87: Intersection non-priority obstacle scenario 221: maximum deceleration.

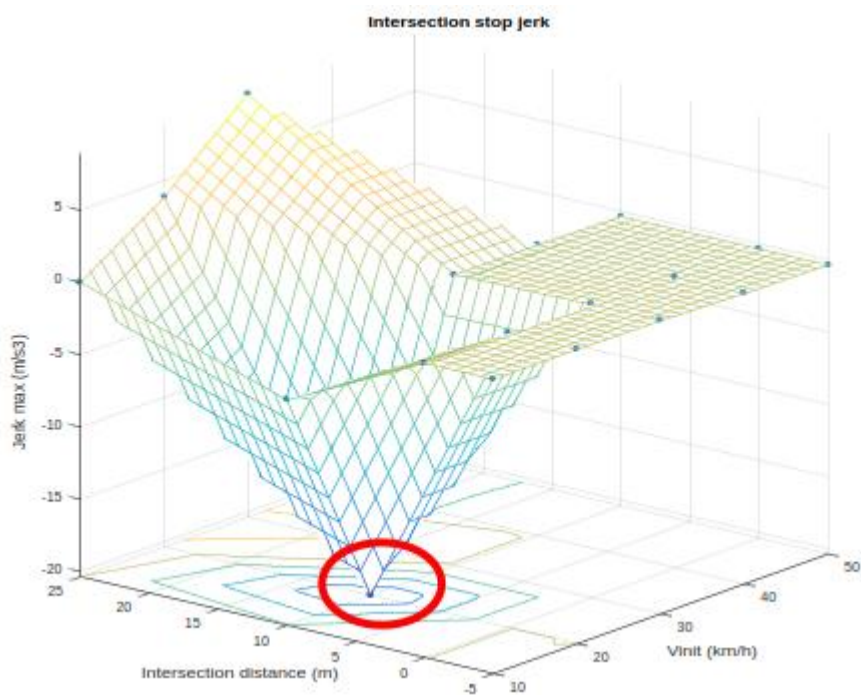


Figure 88: Intersection non-priority obstacle scenario 221: maximum jerk.

Maximal Jerk Experienced by Ego Vehicle Passengers in scenario 221

Figure 87 and Figure 88 depict the maximum acceleration and jerk experienced by the vehicle during a stop at an intersection caused by the late engagement of a non-priority vehicle. The jerk values in particular reflect sharp, short-lived peaks—calculated over brief time intervals ranging from 0.08 to 0.16 seconds—resulting in notably high magnitudes.

For experimental validation, the selected scenario involves a collision case: the Ego vehicle is traveling at 20 km/h, and a non-priority vehicle suddenly enters the intersection when the Ego is 10 meters away. This situation leads to the highest instantaneous jerk observed in the simulations, making it especially relevant for evaluating extreme braking responses and discomfort due to sudden intrusions.

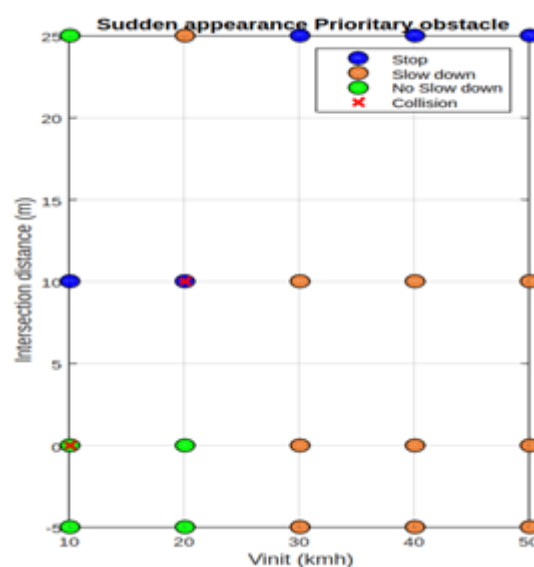


Figure 89: Scenario 221: Identification of critical scenario with possible collisions considering various initial speed V_{init} and distance to intersection.

In Figure 89, scenario 221 is simulated under varying initial speeds and distances to the intersection for the Ego vehicle. The results are categorized to visualize the vehicle's behavior under different conditions:

Blue points indicate cases where the Ego vehicle successfully comes to a stop before the intersection.

Orange points represent cases where the vehicle slows down but does not fully stop.

Green points correspond to situations where the Ego vehicle proceeds through the intersection without needing to brake significantly.

X-marked points highlight scenarios where the Ego vehicle passes through the intersection but a **collision occurs**, identifying these as **critical scenarios** from a safety perspective.

This analysis helps isolate the boundary conditions where the vehicle transitions from safe operation to unsafe outcomes, providing valuable input for designing conservative yet efficient autonomous decision-making strategies.

Scenario	N°	Realisation / Situation	Triggering Event	scenarios to be tester	Nb scenarios
1 Traffic Light Crossing					3
1.1 Nominal stop	1.1.1	Triggering orange light then red light	Ego vehicle at 0m from the light / Ego vehicle at 10 km/h	1 critical scenario identified	1
1.2 Restart	1.2.1	Triggering green light from red	Ego vehicle at 20 km/h	1 critical scenario identified	1
1.3 Traffic light failuer	1.3.4	Delay (1s) on light status (passing through red)	Vehicle at 30 km/h, distance to light <= 30 m	1 critical scenario identified	1
2 Intersection Crossing					2
2.1 Obstacle prioritaire	2.1.1	Late appearance of a priority obstade (from the right), creating a conflict situation	Remaining distance to intersection for ego vehicle: 10 m with initial speed EgoVeh: 40 km/h	1 critical scenario identified	1
2.2 Obstacles non prioritaire	2.2.1	A non-priority obstacle (from the left) enters the intersection at the arrival of the ego vehicle	Remaining distance to intersection for ego vehicle: 10 m with initial speed EgoVeh: 20 km/h	1 critical scenario identified	1

Figure 90: Identification of critical scenarios for all the considered scenarios. These scenarios will be tested in hybrid setup.

Figure 90 provides a summary of the **critical scenarios** identified across all the simulation cases considered for testing. These selected scenarios are then evaluated in the **hybrid setup**, and the corresponding results and analyses are presented in the subsequent **Test Evaluate Validate** block.

This process completes the validation of the **Coverage block** within the SAF, ensuring that all edge and high-risk cases have been systematically addressed.

3.3.3.4 Test Evaluate

We present the results of two key connectivity metrics:

CAM Delivery Ratio (%)

CAM Latency (s)

In the hybrid setup, the real vehicle driving on the test track transmits Cooperative Awareness Messages (CAMs) at regular intervals of 0.1 seconds. These CAMs are received by the virtual simulator to update the state of the virtual EGO vehicle.

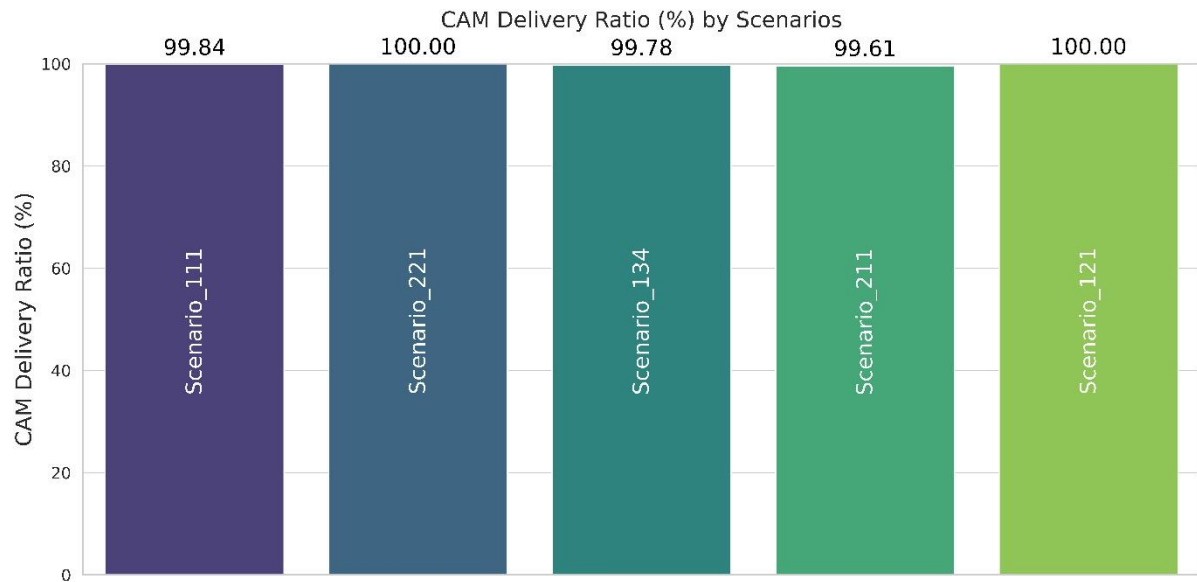


Figure 91: CAM delivery ratio (%) by scenarios.

The CAM delivery ratio is nearly **100%** across all tested scenarios as shown in Figure 91. This indicates that **almost all CAMs transmitted by the real ego vehicle are successfully received by the virtual ego vehicle.**

This result is expected since:

Only the ego vehicle transmits CAMs.

There are no concurrent transmissions or message collisions.

There is no packet loss in the tested scenarios.

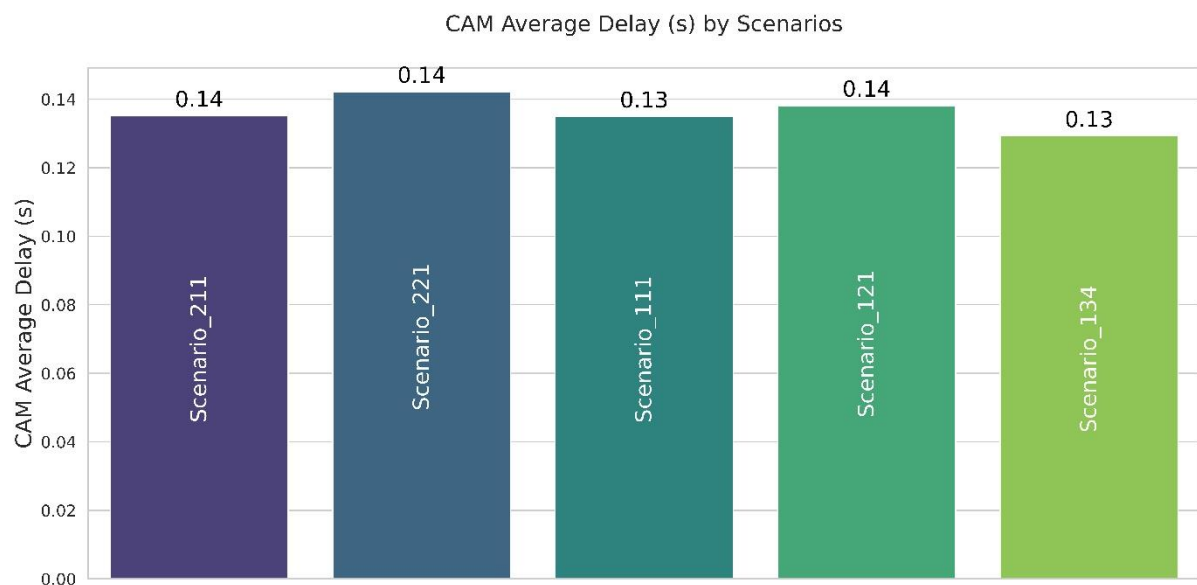


Figure 92: CAM average delay in seconds by scenarios.

The CAM average delay measures the time between CAM transmission by the real ego vehicle and its reception by the virtual ego vehicle.

As shown in Figure 92, the maximum observed delay is 140 milliseconds.

This delay includes:

The communication delay introduced by the network connecting the simulator and the real vehicle.

Despite this, the delay remains within acceptable limits for the Automated Driving System (ADS) under test.

Next, we compare key performance indicators (KPIs) between the simulation environment and the hybrid (hardware-in-the-loop) test setup. One fundamental KPI is the **initial speed of the Ego Vehicle at the moment the scenario is triggered**, as it directly influences vehicle behavior and braking dynamics.

Figure 93 illustrates this comparison across all selected scenarios. The data shows that the initial speeds in the hybrid tests closely align with those in the simulations, with deviations consistently staying below **2 km/h**. This high level of consistency confirms that the scenarios are being replicated under nearly identical conditions in both environments.

The small discrepancies observed are attributed to practical constraints in the real-world test setup, such as limitations in **track length**, minor **surface irregularities**, or **sensor latency**, which may slightly affect vehicle acceleration and speed. Nevertheless, the close match demonstrates the reliability of the hybrid testbed in reproducing simulated conditions, thereby validating the starting point for further KPI comparisons and safety evaluations.

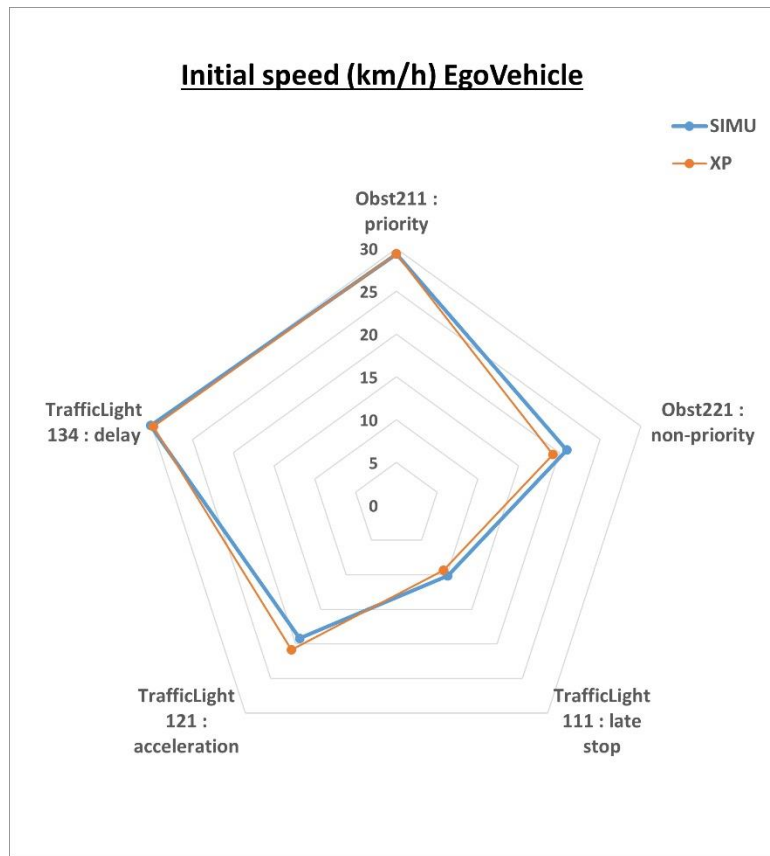


Figure 93: Ego vehicle initial speed comparison for virtual testing and hybrid testing.

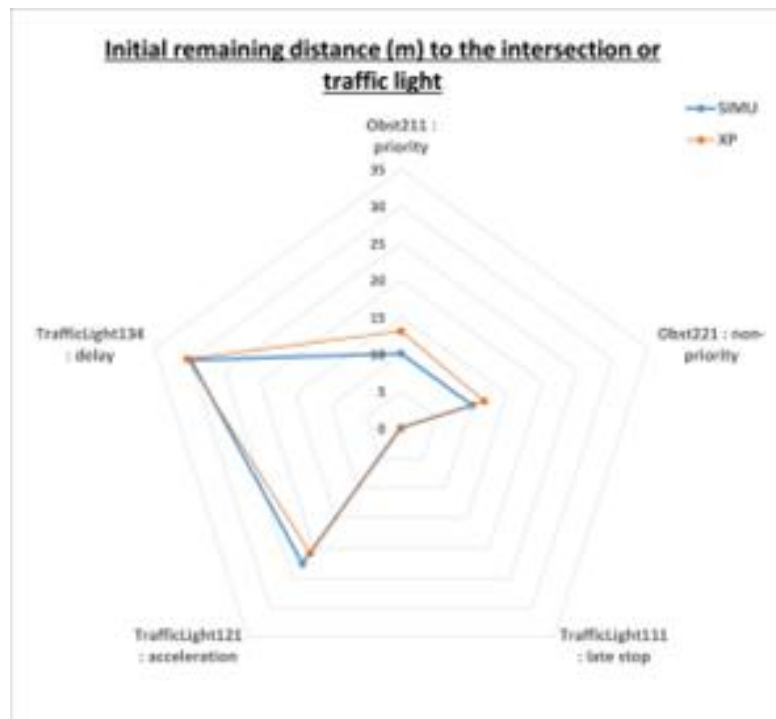


Figure 94: Ego vehicle remaining distance to intersection comparison for virtual testing and hybrid testing.

The second critical parameter we analyze is the **distance between the Ego Vehicle and a predefined reference point at the moment the scenario is triggered**. This reference point differs depending on the scenario type—for example, it is the **traffic light projection** in scenarios 111, 121, and 134, while for scenarios 211 and 221, it is the **entry point of the intersection**.

Figure 94 presents a comparison of this distance between the simulation and hybrid test results. Overall, the values align closely, confirming that the scenario timing is consistently replicated in both environments. However, one notable exception is observed in **scenario 211 (priority obstacle)**, where the hybrid tests show a **3-meter shorter distance** at the trigger point compared to simulation.

This discrepancy is attributed to a **premature trigger event** in the Sunrise simulator during the hybrid setup, likely caused by a timing offset in obstacle appearance or sensor detection logic. Despite this variation, the results still reflect a high degree of alignment, supporting the validity of the hybrid setup for realistic scenario reproduction and further safety assessments.

Analysis of Decision-Making performance of the ADS

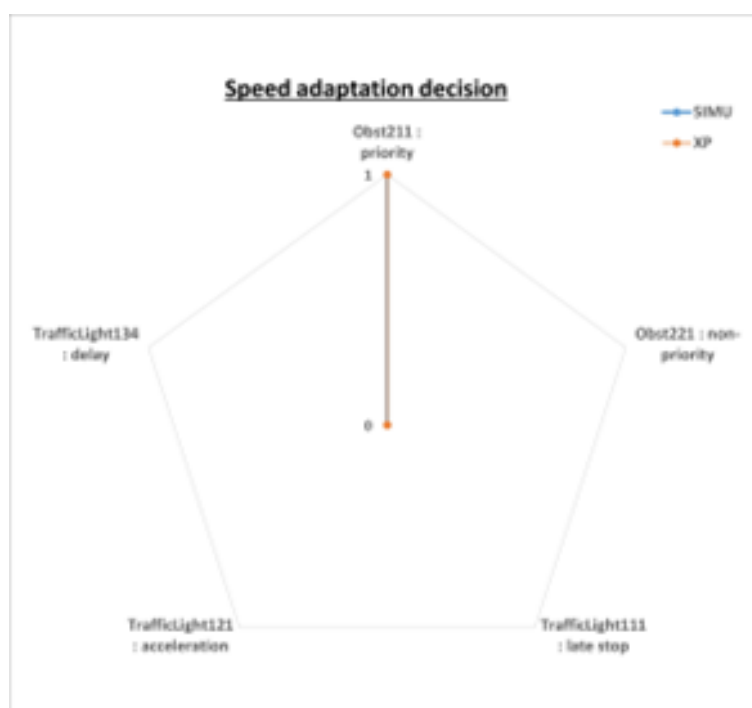


Figure 95: Speed adaptation decision making performance.

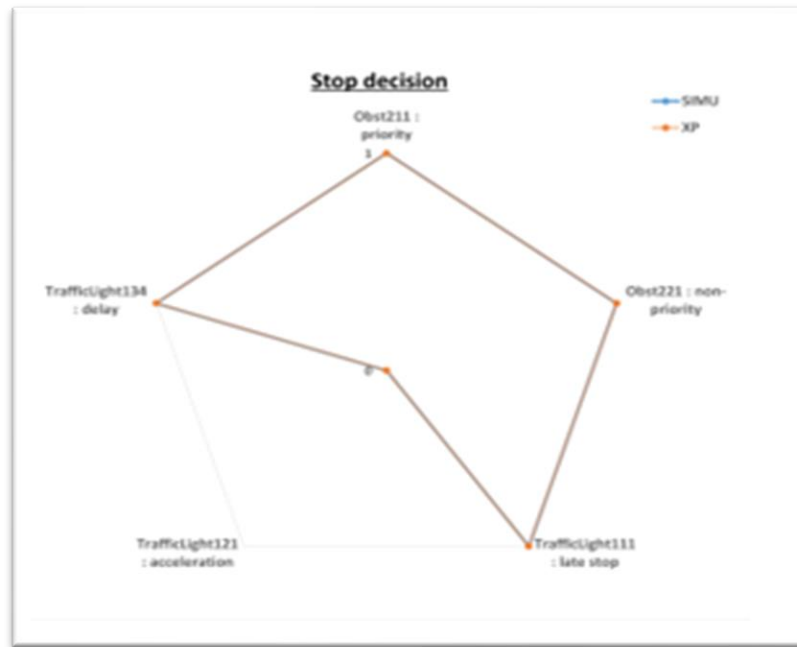


Figure 96: Decision making to stop the ego vehicle.

The Autonomous Driving (AD) system consistently makes the correct decision to stop in all tested scenarios, with the **only exception being the restart scenario (121)**, illustrated in Figure 95. In this case, the vehicle appropriately resumes movement after the traffic light turns green, as expected from the system's logic.

Across all other scenarios, stop decisions made by the AD system are reliably reproduced in both simulation and hybrid tests, demonstrating **consistent behavior across environments**.

An exception to stopping is observed in the **priority obstacle scenario (211)**, shown in Figure 96. Here, instead of a full stop, the vehicle intelligently **reduces its speed from 40 km/h to 30 km/h**, allowing it to safely navigate the intersection.

For **slower scenarios**—specifically scenario 221 and all traffic light cases where the ego vehicle's speed is **under 30 km/h**—no additional speed adaptation is necessary. The vehicle behavior remains stable and predictable.

Overall, these results confirm that the **AD system's speed adaptation strategy and stop/go decisions are aligned** between the simulated environment and the hybrid (real-world) testing setup, reinforcing the reliability of the simulation framework and its capability to replicate real-world responses accurately.

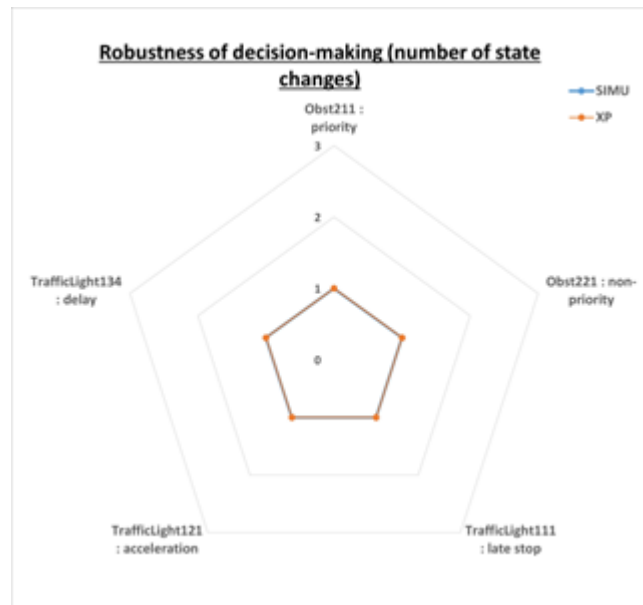


Figure 97: Robustness of decision-making (number of state changes).

Decision-making robustness is assessed by tracking how many times the AD system changes its stop/go decision during a scenario. In all evaluated scenarios—both in simulation and hybrid tests—the **decision is made only once**, with no reversals or fluctuations. This indicates a **high level of consistency and robustness** in the decision-making process.

Specifically, in the **restart scenario (121)**, the system demonstrates the expected behavior: it does **not reissue a stop command** after the traffic light turns green, confirming the correctness of the transition logic.

As shown in Figure 97, there is **no observable discrepancy** between the simulated and real-world outcomes in terms of decision transitions. This confirms that the AD system's decision-making logic remains **stable and reliable across different testing environments**, an important criterion for safety-critical applications.

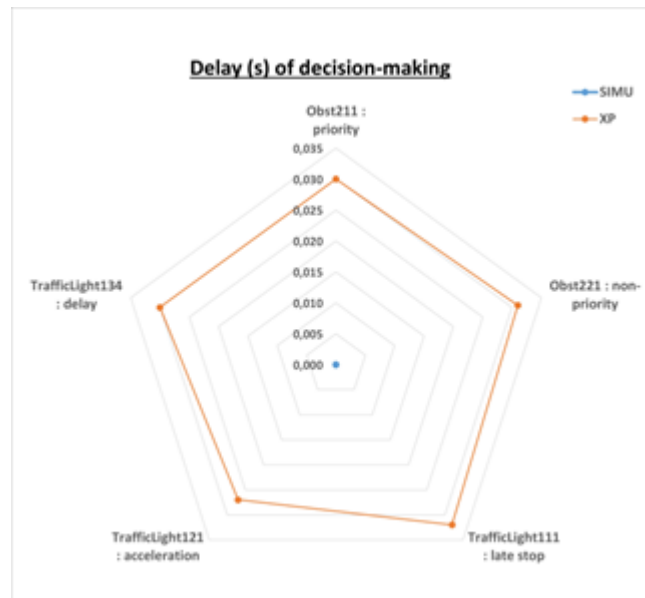


Figure 98: Delay (s) of decision making.

As illustrated in Figure 98, the **decision delays**—measured as the time between trigger detection and execution of the stop/go decision—are **zero in simulation**, but range between **27 to 32 milliseconds** in the real-world hybrid tests.

This difference stems from the execution environments:

In **simulation**, MATLAB runs in a **deterministic and tightly synchronized** time framework, allowing decisions to be executed instantaneously with no measurable delay.

In contrast, the **real tests** run on **RTMaps**, a real-time platform that distributes tasks across **multiple threads** without strict global synchronization. This results in slight but consistent desynchronization, leading to the observed ~30 ms delay.

Despite this, the delay is **well within the system's 40 ms control cycle**, meaning it has **no significant impact** on the vehicle's dynamics or the correctness of its decisions. Thus, while measurable, the delay remains **negligible from a functional and safety perspective**, affirming the system's real-time performance.

Evaluation of the performance of the control of ADS

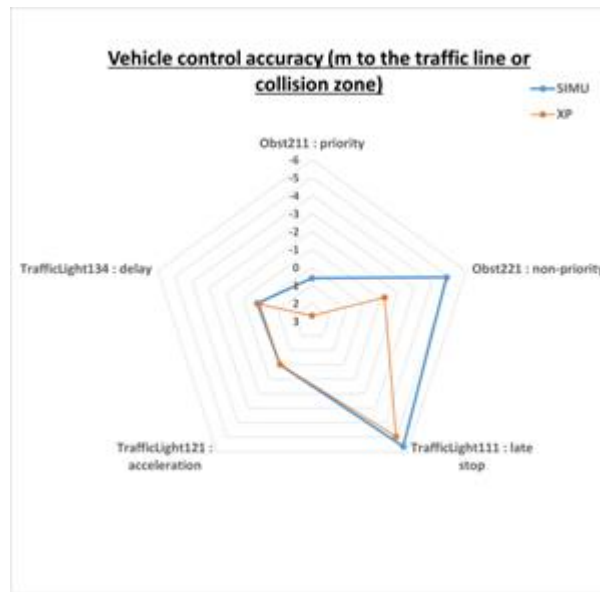


Figure 99: Vehicle control accuracy (distance in meters to the traffic line or collision zone).

Control precision is evaluated by measuring how closely the Ego Vehicle stops relative to a predefined **reference point**, such as the traffic light line or a potential **collision zone**. This is done by calculating the remaining distance between the reference point and the front bumper of the vehicle at the time of stopping.

In **non-critical, nominal stop scenarios**, the system demonstrates **high control accuracy**. For instance, in scenario 134 (a nominal traffic light stop), the Ego Vehicle stops within **0.19 meters in simulation** and **0.095 meters in real-world tests**, showing excellent alignment between both testing environments.

As shown in Figure 99, across all **traffic light scenarios** (111, 121, 134), control behavior in simulation and real tests **closely match**, indicating a reliable and predictable stopping capability in structured, rule-based situations.

However, in **more complex scenarios involving obstacles**, discrepancies grow larger. In scenario 211 (priority obstacle), the stopping deviation reaches **2.1 meters**, while in scenario 221 (non-priority obstacle), it increases further to **3.7 meters**. These larger deviations are expected due to the **increased difficulty of anticipating the precise stop point** in dynamic or ambiguous contexts, where the Ego Vehicle must interpret less deterministic cues.

Overall, while **control precision is high in structured scenarios**, the increase in stopping error in obstacle cases highlights areas where **further calibration or prediction handling** may enhance system responsiveness and safety margins.

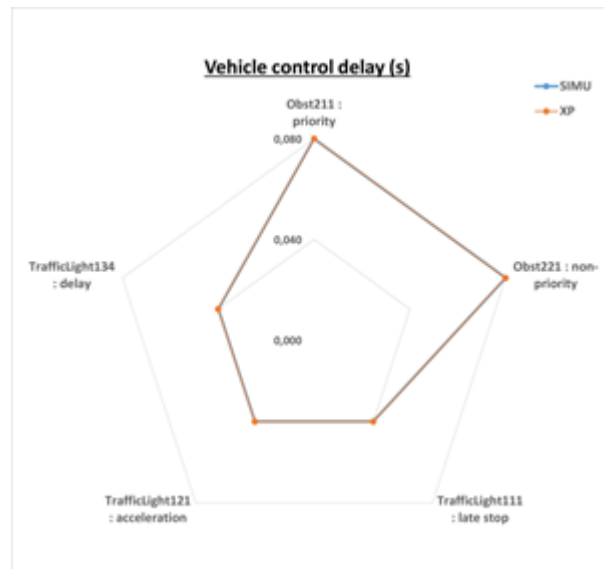


Figure 100: Vehicle control delay (s).

Vehicle control delay—defined as the time between event detection (e.g., a red light or obstacle) and the issuance of a command to the vehicle actuators—is observed to be **consistent across both simulation and real-world tests**, demonstrating the reliability of the control algorithm’s timing logic in Figure 100.

In **traffic light stop scenarios**, the control delay is intentionally fixed at **40 ms**. This ensures a **smooth deceleration profile** by aligning control updates with the system’s 40 ms cycle, providing predictable and stable speed transitions.

For **intersection stops**, an additional **40 ms delay** is introduced to allow for **decision confirmation**, bringing the total control delay to **80 ms**. This added buffer accounts for the increased complexity and safety-critical nature of unregulated or ambiguous intersection scenarios.

However, **Sunrise simulation results** suggest that this confirmation step may be **unnecessary**, as the system consistently demonstrates **robust and confident decision-making**. This opens the possibility for **optimization** by reducing or eliminating the confirmation delay in future versions, particularly in scenarios where decision consistency is already proven to be high.

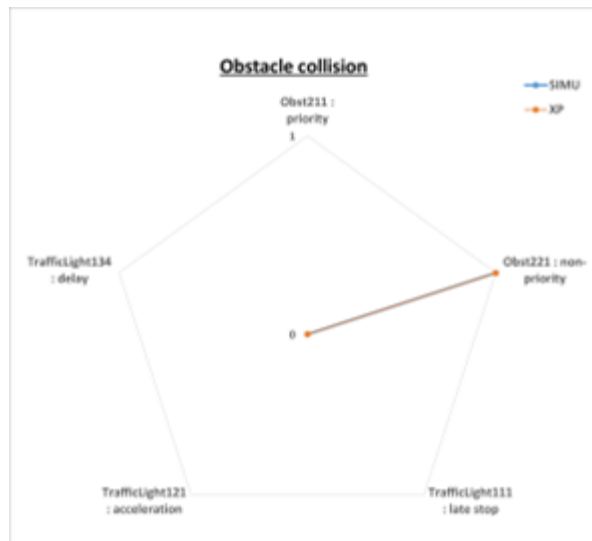


Figure 101: Collision with obstacle vehicle.

Experimental results summarized in Figure 101 reveal a single collision event in **scenario 221**, where a **non-priority obstacle** enters the intersection, and the autonomous vehicle **accelerates late** after an initial stop. This rare case—occurring only in a **real-world test**—serves as a **validation of the longitudinal control** logic under edge-case conditions.

The obstacle’s **uncertain trajectory** as it moves near the Ego Vehicle complicates the timing of collision detection and decision-making. Both **simulation and real test** outcomes confirm the collision, indicating that the perception and control modules are working consistently but are challenged by ambiguity in obstacle behavior.

Importantly, after the collision, the Ego Vehicle **activates its hazard lights** and **remains stationary**, demonstrating that the **post-collision behavior** prioritizes safety. This response ensures the vehicle does not proceed into further risk, even once the obstacle clears the intersection, thereby validating safety fallback mechanisms in the control architecture.

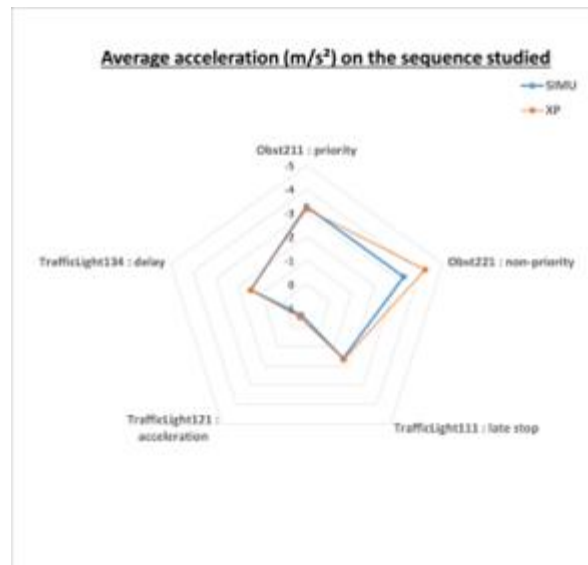


Figure 102: Comfort control: average acceleration on the sequence studied.

Global deceleration during braking and acceleration during the re-acceleration phase—particularly in **scenario 121**—serve as indicators of overall system performance during stop and restart events. As shown in Figure 102, the behavior is closely aligned between **simulation and real-world tests** for all **traffic light scenarios (111, 121, 134)** and the **priority obstacle stop (211)**. The only significant deviation occurs in the **non-priority obstacle stop scenario (221)**, where a difference of approximately **1 m/s²** is observed. This discrepancy is attributed to previously noted **variations in obstacle trigger distances** between the simulation and real test environments, reinforcing the importance of accurate environment modeling for control evaluation.

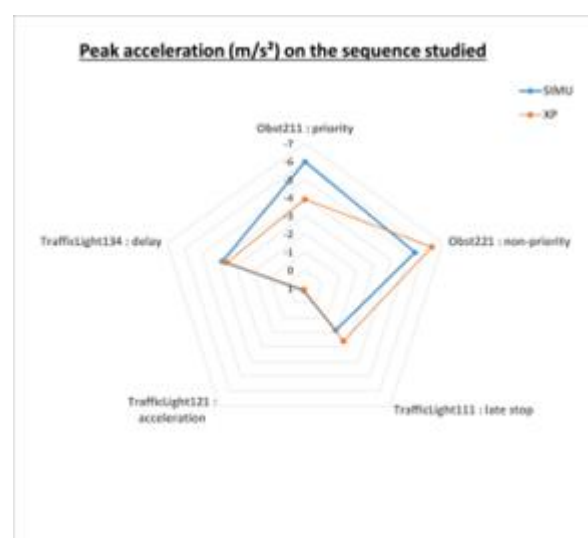


Figure 103: Comfort control: peak acceleration on the sequence studied.

As shown in Figure 103, maximum decelerations during traffic light scenarios generally align well between simulation and real-world tests. However, in high-dynamic situations at low speeds (<10 km/h), simulations slightly underestimate deceleration, leading to longer stopping distances compared to real tests. Despite this minor deviation, the simulation overall provides a reliable representation of real vehicle behavior in these scenarios.

Types of KPI	KPI	Scenario : 111	Scenario : 121	Scenario : 134	Scenario : 211	Scenario : 221	KPI criteria	Remarks
		Traffic light : Late stop	Traffic light : Acceleration	Traffic light : Delay	Obstacle : priority	Obstacle : NON-priority		
Decision making : Longitudinal control	Decision making : stop/restart	yes	yes	yes	yes	yes	yes : Expected decision no : Unexpected decision	
	Decision making : speed adaptation	no	yes	no	yes	no	yes : Expected decision no : Unexpected decision	
	Robustness : (number of state changes)	1	1	1	1	1	1 : Robust decision > 1 : No-robust decision	
	Delay (s)	0,032	0,027	0,030	0,030	0,031	<= 0,04 s : Without delay < 0,2 s : Acceptable delay < 0,3 s : Significant delay > 0,3 s : Unacceptable delay	
Performance : Longitudinal control	Accuracy : vehicle control (m)	-4,963	Unmeasurable	-0,095	2,701	-1,240	-0,3m < Dist < 1m : High precision -1m < Dist < 2m : Acceptable precision -5,47m < Dist < 3m : Low precision Dist < -5,47m ou Dist > 3m : Unacceptable precision	vehicle length 5,47 m
	Delay : vehicle control (s)	0,040	0,040	0,040	0,080	0,080	= 0,0 s : Without delay < 0,120 s : Acceptable delay < 0,300 s : Significant delay > 0,300 s : Unacceptable delay	

Figure 104: KPI Results of the decision-making performance and the longitudinal vehicle control performance.

Figure 104 provides a summary of the KPIs obtained from the validation tests, with color coding used to indicate compliance with predefined acceptance criteria. All **decision-making KPIs** appear in **green**, confirming that the system's decision logic performs **within the expected range**. In contrast, some **longitudinal control performance KPIs**—notably in the **traffic light late stop**, **scenario 211**, and **scenario 221**—show **lower precision**. These discrepancies are primarily attributed to **V2X communication delays** observed in the **hybrid testing setup**, rather than issues with the decision-making process itself, which remains **consistently within acceptable bounds**.

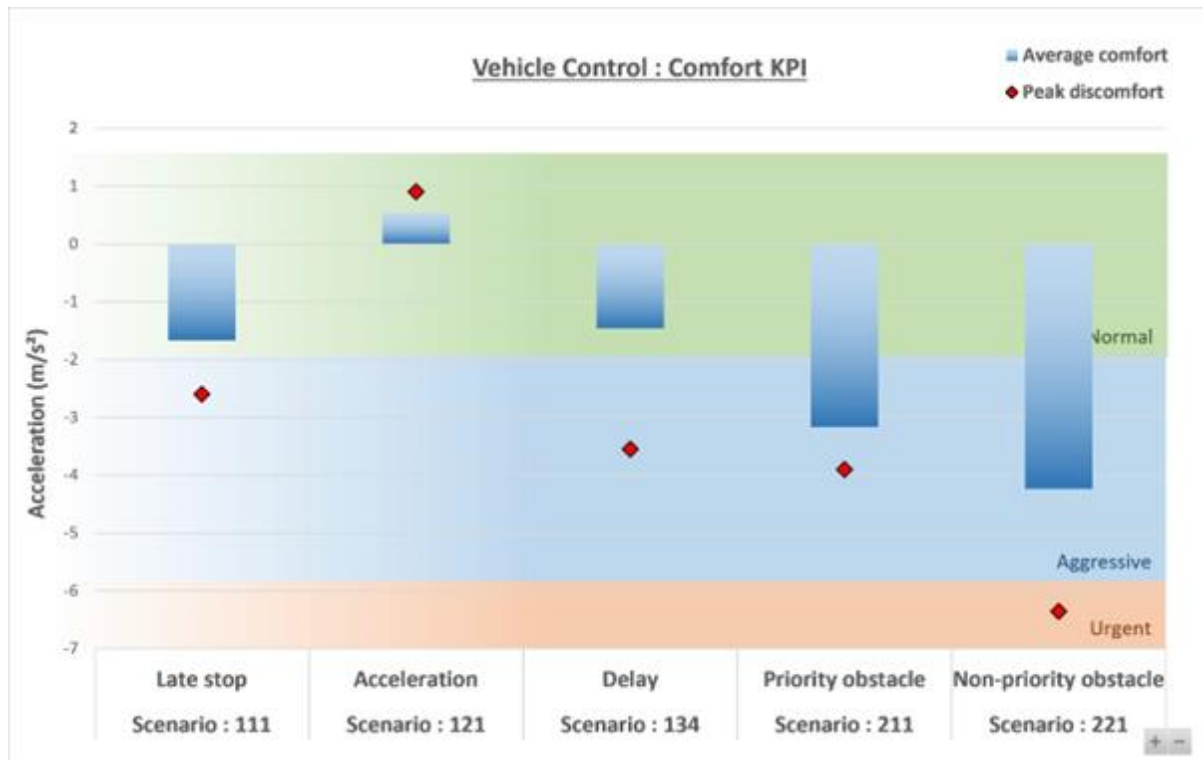


Figure 105: KPI results of the comfort vehicle control.

Figure 105 illustrates the vehicle control comfort KPIs obtained from the hybrid testing. For scenarios 111, 121, and 134, the average acceleration and deceleration values remain within the normal comfort range, consistent with the absence of sudden or unexpected events.

In contrast, scenarios 211 and 221 exhibit comfort levels that fall into the aggressive range, due to the sudden appearance of an obstacle vehicle which forces the ego vehicle to decelerate rapidly. Additionally, the maximum jerk—indicating a sharp change in acceleration—is observed in scenario 224, where a non-priority obstacle appears abruptly in front of the ego vehicle. This highlights the increased control demand and passenger discomfort in highly dynamic or unpredictable traffic scenarios.

3.3.3.5 Safety Argument

The safety argument is built on scenario coverage, pass/fail evaluation, and binary safety case outcomes. Key points include:

Scenario Coverage: All five scenarios were tested in both virtual and hybrid setups, covering nominal and edge cases. The selection of critical scenarios was based on peak jerk values, indicating potential safety risks.

Pass/Fail Evaluation: The ADS made correct stop decisions in all scenarios, with Scenario 121 showing appropriate resumption on green. Scenario 221 resulted in a collision due to delayed acceleration, but the vehicles stopped safely and activated hazard lights.

Binary Safety Case: The ADS was deemed safe in Scenarios 111, 121, 134, and 211, with consistent behavior across virtual and hybrid tests. Scenario 221 highlighted a failure case, informing future improvements.

KPIs: Ego vehicle speeds at trigger points matched closely (deviations <2 km/h), trigger distances were consistent (except a 3-meter deviation in Scenario 211), and decision-making delays (27–32 ms) were below the 40 ms control cycle. Connectivity showed a 100% CAM delivery ratio with a 140 ms average delay.

3.3.4 Key Takeaways

The demonstration of Use Case 1.2 provided valuable insights into the application of the SAF for urban AD validation:

Robust Scenario Coverage: The combination of virtual and hybrid tests ensured comprehensive coverage of nominal and edge cases, validating the ADS's ability to handle complex urban intersections.

Consistent Performance: Virtual and hybrid tests showed close alignment in ego vehicle speeds (deviations <2 km/h) and trigger distances, with minor differences due to real-world factors like track length and event timing.

Connectivity Reliability: V2X communication achieved a near-100% CAM delivery ratio with a 140 ms average delay, though end-to-end connectivity delays (200 ms) suggest room for improvement.

Control and Comfort: Decision-making and longitudinal control KPIs were within acceptable ranges, but longitudinal control accuracy was slightly low, and comfort KPIs indicated aggressive braking in Scenario 221 (-6 m/s^2 at 20 km/h).

Identified Improvements: Future enhancements include reducing connectivity delays (targeting 40–80 ms), improving low-speed braking models, and streamlining decision-making to reduce reaction delays.

3.3.5 Deviations from D7.2

Compared to the commitments in D7.2, we have not covered the decide block validation.

3.4 Use case 1.3: Urban AD validation – Cooperative perception testing

3.4.1 Use Case Overview

In Use Case 1.3, the SUNRISE SAF is applied on a cooperative perception and decision-making CCAM system assumed to be deployed in urban intersection scenarios. The SUT is the cooperative perception AD subsystem of an urban chauffer ADS which is capable of sending/receiving and processing rich V2V information data mainly consisting of object-level data perceived from other connected vehicles.

The Use Case aims to demonstrate how the SUNRISE safety argumentation is derived for urban collective perception (CP), focusing on the co-simulation aspects and the safety case building when occlusions and vehicle to vehicle (V2V) connectivity aspects are present.

There are three functional scenarios defined in Sunrise D7.1 assessing the cooperative perception system performance in Use Case 1.3. The first is the “darting-out pedestrian” scenario (1.3 A), the second one is the “urban junction” scenario (1.3 B), and the third one is the “urban roundabout” scenario (1.3 C). UC1.3 A, presented in

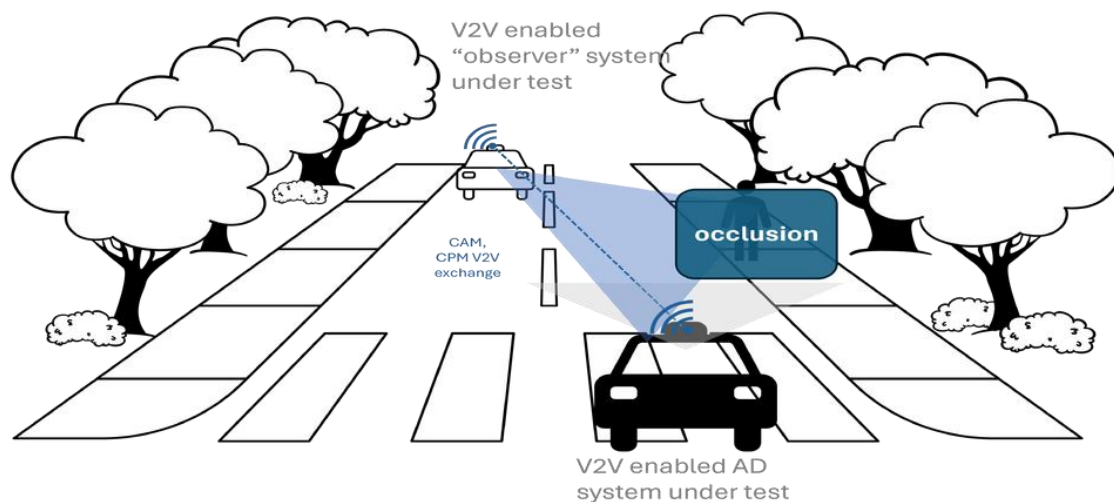


Figure 106, was selected for the SAF evaluation and the demos.

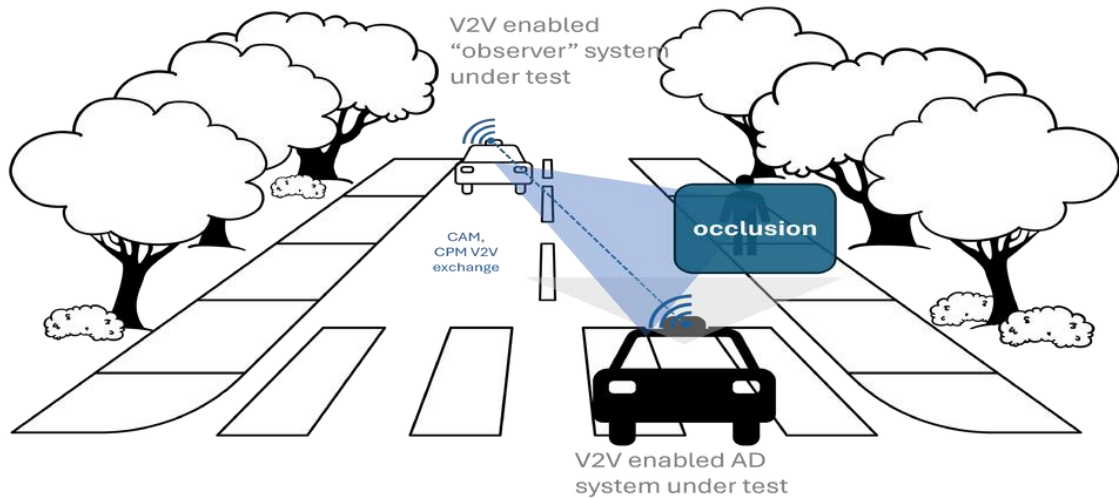


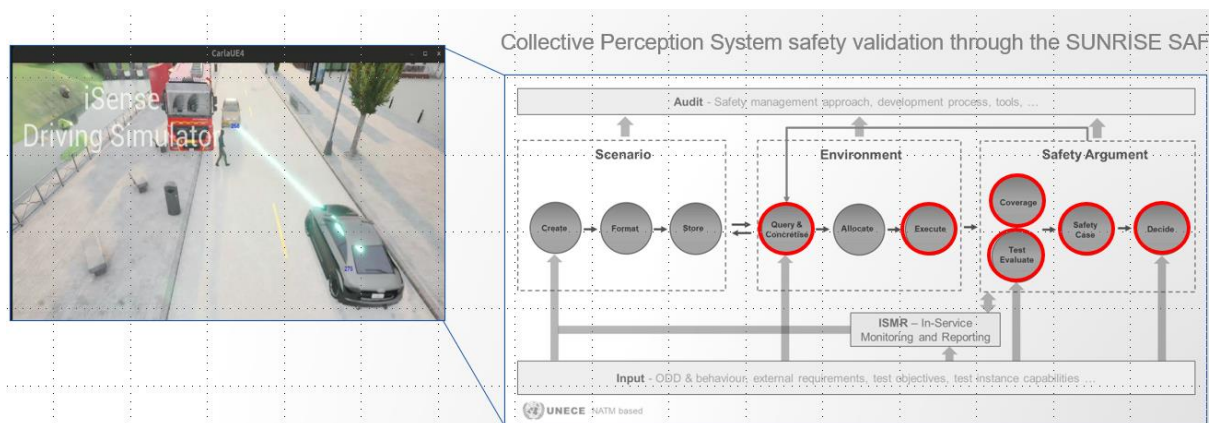
Figure 106: UC 1.3 visual sketch

Objective:

- Safety case building of the perception layer of CCAM system in urban non-line-of-sight (NLOS) scenarios with V2X connectivity.
- Combine virtual with hybrid testing.

3.4.1.1 Covered aspects of the SAF

Six sub-blocks of the SAF contributing to the safety case building have been applied and demonstrated by the two teams of ICCS and VED as presented Figure 107, namely: the scenario Concretization, the scenario Allocation, the simulation framework Execution (part of SAF environment block) as well as the test outcome Evaluation, Safety Case and the Scenario Coverage (part of the SAF Safety argument block). The Table 12 below shows the contributions to the SAF block validation per partner for UC1.3.



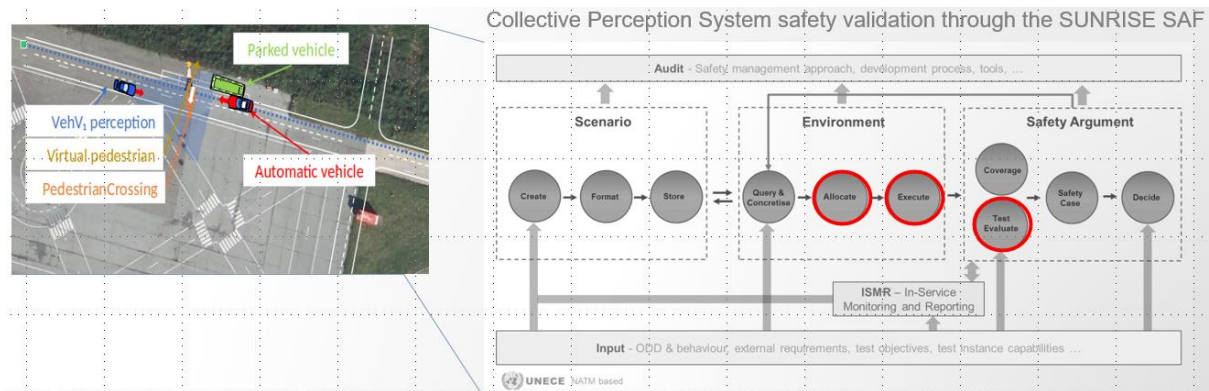


Figure 107: SAF demonstrated blocks in this UC's demo (s). Up: ICCS SAF instance; Down: VED SAF instance.

Table 12: Contributions to the SAF validation per Partner in UC1.3.

Partner/ Block	ICCS	VED
SUNRISE DF		
Query & Concretise	x	
Allocate		x
Execute	x	x
Test Evaluate	x	x
Coverage	x	
Safety Case	x	
Decide	x	

Two tutorial demo videos were produced and included in the SUNRISE Handbook, focusing more on the “darting-out pedestrian” scenario (1.3 A), their links are provided hereafter:

<https://youtu.be/i1vV8lq7Vvs>

<https://youtu.be/ZE93V7sfVzM>.

3.4.1.2 Safety case setup

The Cooperative Perception System under test for this demo included two connected vehicles both equipped with on-board Perception units and able to exchange CPMs either directly or

via a virtual Road Side Unit (RSU). Tested operational design domain was urban pedestrian crossings in straight road segments with parked vehicles (the parked vehicle is assumed to play the role of the object creating the occlusion for the SUT). To support the safety case building, a mixed proving ground and virtual co-simulation environment is implemented where the test scenario of a darting out pedestrian including both adults and children has been tested in many scenario variations.

The approach in Safety case building is characterized by: (a) A Modular testing approach where both independent subsystems' testing as well as end-to-end system testing is considered in scenario-based safety evidence gathering. This leads to two types of safety evaluation using appropriate metrics each time, one for ADS cooperative perception function and one for ADS control function, which in turn leads to different set of critical scenarios for each of these two subsystems; (b) Evidence credibility checks are considered in both virtual and hybrid testing settings, namely analysis of test scenario repeatability during execution and analysis of system under test behaviour in virtual versus hybrid setup.

The safety argumentation building of this UC includes 6 key steps (Figure 108):

Step 0: it's where the system under test and the safety case objectives are defined, including system's under test expected behaviour and pass-fail criteria.

Step 1: it's where the safety metrics are defined, in this case also considering dedicated metrics for cooperative perception sub-system of the system under test.

Step 2: it's where the scenario space is explored and where automated smart concretization algorithms are implemented targeting at maximum scenario space coverage on one hand and critical scenario generation on the other hand (only supported by ICCS).

Step 3 and 4: it's where the set of generated concrete scenarios are executed in virtual custom test environment, enabling V2V-augmented perception testing and metrics logging. Few of the test scenarios that are classified as "outcome uncertain" from step 6 are also executed in hybrid test environment with one real vehicle and two virtual agents, one vehicle and one pedestrian.

Step 5: it's where scenario coverage is estimated after all test scenarios have been executed or classified by the algorithm of step 2 (only supported by ICCS).

Step 6: it's where safety violations are analyzed and combined coverage from virtual and hybrid environments is calculated.

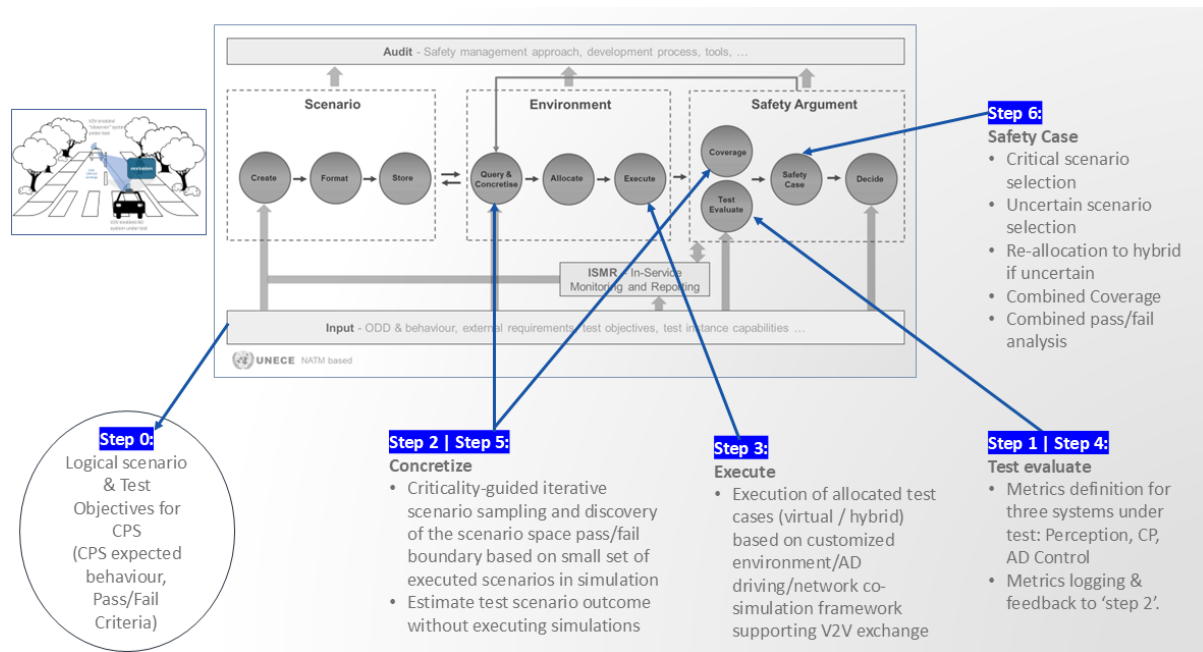


Figure 108: Safety case building steps of UC1.3 based on SUNRISE SAF

3.4.2 Overview of tested scenarios

The selected logical scenario for both ICCS and VED SAF instance evaluation was the “darting-out pedestrian” scenario (1.3 A).

Approach A

A set of concrete scenarios was produced based on smart sampling (see section ‘query and concretize’ below) of the following parameter space:

Table 13: Contributions to the SAF validation per Partner in UC1.3.

Scenario space Parameters	ego speed	pedestrian speed	distance ego_ped
Parameters range ; sampling step for SuT A: Perception and Control	[30, 50] ; 1 (km/h)	[0.5, 2] ; 0.1 (km/h)	[18, 50] ; 2 (m)
Parameters range ; sampling step for SuT B: Cooperative Perception	[36-42]; 1 (km/h)	[0.5. 2]; 0.1 (km/h)	[18, 50]; 2 (m)

Other scenario variations were also executed in order to apply ODD/DDT variations during the SAF ‘Execute’ block evaluation but not used in the safety argumentation of this demo. The **Error! Reference source not found.** below presents four variations of the same scenario

created in CARLA by configuring the darting out agent's trajectory or the environmental conditions.



Figure 109: Concrete scenario variations in CARLA : day vs. night, adult vs. kid, fog, rain, darting out direction vertical vs. non vertical.

Approach B

Validation was based on a combination of virtual and hybrid test environment where, in order to validate the virtual setup, connectivity metrics from the hybrid setup (real HW and V2X communications) were compared against their counterparts in the virtual testing environment.

As shown in Figure 110, the test scenario includes V2X-based Cooperative Perception Message (CPM) exchange between vehicles. The cooperative behaviour unfolds as follows:

- When the moving vehicle detects the pedestrian through its onboard perception system, it generates a **CPM** containing information about the detected object (pedestrian).
- This CPM is broadcast, visually represented in the simulation by **orange rings** emanating from the sender vehicle.
- The EGO vehicle receives the CPM, shown with **white rings** indicating incoming messages.
- Upon receiving the CPM containing the pedestrian's position, the EGO vehicle responds by slowing down appropriately.
- This enables the pedestrian to safely cross the zebra crossing.
- After the pedestrian crosses, both the EGO and the moving vehicle resume normal operation.

Note: Even when the pedestrian is no longer visible to the EGO vehicle due to the obstruction (e.g., the parked car), the EGO vehicle still receives the CPM from the observer vehicle which maintains line-of-sight. This reinforces the benefit of cooperative perception in enhancing situational awareness and improving safety in complex urban scenarios.

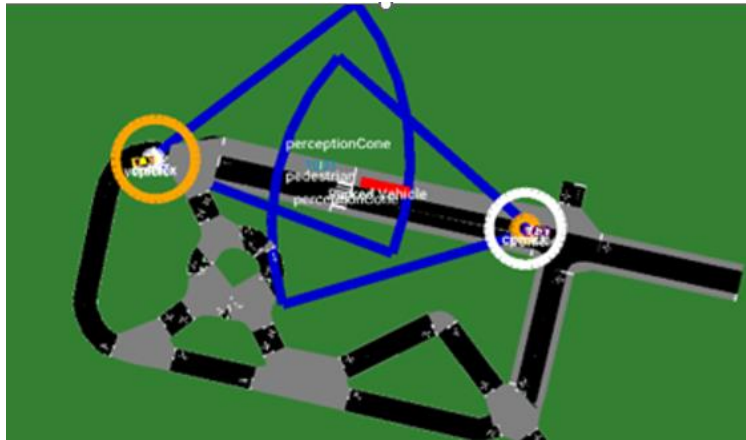


Figure 110: Virtual testing, CPM transmission from vehicle V1 and CPM reception by ego vehicle

The visual elements presented in Figure 110 are:

- Road infrastructure: A typical urban environment is modeled, including a road network with adjacent sidewalks, a designated pedestrian crossing (zebra crossing), and side car parking.
- Traffic agents:
 - ✓ The EGO vehicle (white) under test, equipped with V2X and perception systems.
 - ✓ A moving vehicle (yellow), participating in cooperative awareness.
 - ✓ A pedestrian (yellow, ID: 1001), representing a vulnerable road user.
 - ✓ A parked vehicle (red), obstructing the pedestrian's line of sight.
- On-board (camera-based) perception FoV is presented by the blue cone.

3.4.3 SAF Block demonstrations

3.4.3.1 Query & Concretize

Two scenario concretization approaches have been implemented by two different teams.

Note: No database querying was performed for this UC. The logical scenarios selected for testing were handcrafted by experts based also on non-light of sight scenarios from the literature and pedestrian darting out scenarios from EuroNCAP.

Approach A (ICCS): Starting from a logical scenario described by a given parameter space with defined parameter ranges, a smart sampling method and scenario (pass/fail) outcome prediction is proposed based on a machine learning Gaussian Process-based algorithm. The proposed method allows for i) estimation of the pass/ fail probability of unseen concrete scenarios without the need for execution/simulation, and ii) generation of scenarios close to the pass/fail boundary, which are the hardest to predict their outcomes (high uncertainty). Different pass/fail criteria are used depending on the SUT (Perception, Collective Perception, AD_controller) generating different set of 'critical' scenarios close to the pass/fail boundary. Uncertainty of pass/fail outcome (each test scenario outcome is recorded along with a pass/fail confidence value assigned to it), is then used in order to take decisions on specific set of scenarios to be allocated in higher-fidelity testing environments, i.e. hybrid testing.

Concretization and critical scenario retrieval is validated through simulation execution. Method is described in Sunrise D3.4 while its metrics are described in Sunrise D5.3.

Approach B (VED): Use of hand-crafted concrete scenarios and application of a ready-to-use tool from T3.3 to concretize scenarios using some criteria. Concretization is validated through simulation execution and verifying that each concrete scenario is successfully executed and leads to slightly different results/KPIs.

SAF block validation based on sec 2.2

Validation criteria and how they are met was first described in Sunrise D7.2 [4] and complementary information is provided below:

- Input Completeness: Proper integration and use of input data such as Operational Design Domain (ODD), system requirements, behaviors, and external regulations.
- Selected knowledge-based test logical scenario was based on EuroNCAP.
- Query Validity: All queries to scenario storage must be well-formulated, drawing on a harmonized ontology (e.g., OpenLABEL) to ensure common understanding.
- N/A since no query was performed.
- Scenario Quality: Logical scenarios (with parameter ranges) must be correctly concretized into executable forms, incorporating relevant test objectives.
- Parameter-based concretization mechanism has been implemented in alignment with the general guidelines of the ISO34505 which described prioritization of boundary and critical scenarios.
- Auditability: Validation includes the ability to review the process of concretization—audit steps must check that ODD descriptions, requirements, and variable assignments are all present and compliant with standards.
- Covered by the parameter-based concretization mechanism.
- Feedback Integration: The block must incorporate feedback from test analysis, enabling refinement of parameter combinations and ensuring coverage of edge cases.
- Edge case scenarios were identified based on the joint coverage/test evaluate results (see coverage sub-block).

3.4.3.2 Allocate

For scenario **allocation** to test environments both approaches make use of the initial allocation of the test cases to the test instances with the process described in the deliverable D3.3. All tests that can be performed using simulations will be allocated to virtual testing. However, during the process of safety case building (see ‘safety case’ sub-block), a few test scenarios will be re-allocated to physical or hybrid testing according to the principle of dynamic re-allocation supported by the SAF. Selection of these re-allocated scenarios is based either on the limitations of the virtual setup to model certain aspects (e.g. connectivity delays), or the scenario outcome ‘Test evaluate’ results and their proximity to the pass/fail boundary.

To validate the **Allocate** block of the SAF in the context of **UC1.3A**, we follow a structured approach presented in D3.4. Main steps followed are:

- **Requirements identification**

We begin by identifying the critical test requirements derived from the UC1.3A scenario. These requirements reflect the functional, behavioral, and communication aspects necessary to evaluate the cooperative perception system under typical urban driving conditions.

- **Test scenario allocation**

Each requirement is then mapped to the capabilities of the two available testing frameworks: **Virtual Testing** (fully simulated environment) and **Hybrid Testing** (combination of simulated environment and real components). This step ensures that the chosen test environments can support the validation of each identified requirement with appropriate fidelity. The resulting Requirement vs. Capability Allocation Table, presented in Table 14, provides a clear overview of which requirements are supported by each testing framework. This allocation step ensures coverage of the necessary elements for safety assurance.

Table 14: Comparison of requirements and testing framework capabilities.

Requirement	Virtual Testing	Hybrid Testing
Simulation of realistic traffic scenarios (e.g., pedestrian crossings, parked vehicles)	Supported	Supported
Modelling of occlusion effects (e.g., pedestrian hidden behind obstacles)	Supported	Supported (via virtual detection and CPM relay)
Emulation of sensor perception (e.g., field of view, detection cone)	Supported	Partially Supported (via virtual detection)
CAM/CPM generation and compliance with ETSI TR 103 562	Supported	Supported (real and virtual stack via MQTT)
Communication modelling (range, latency, reliability)	Supported (simulated channel models)	Partially Supported (depends on actual network conditions)
Real-time integration with physical Ego vehicle	Not Supported	Supported (via MQTT-based V2X stack interface)
Accurate logging of message exchange and vehicle reaction timing	Supported	Supported (via synchronized logs from simulator and OBU)

Ego vehicle behavioral response to CPMs (e.g., stop, slow down)	Supported (simulated decision logic)	Supported
Visualization of scenario dynamics and communication events	Supported (SUMO + OMNeT++)	Supported (SUMO for virtual; physical observation + OBU)
Testing of V2X stack interoperability and protocol compliance (e.g., ITS-G5, MQTT)	Supported	Supported (real software stack, protocol interface)

Using the above comparison, we obtain the allocation structure diagram of Figure 111, where the following notation is used:

- **Simulated Components (Blue):** All road network elements, pedestrian/vehicle models, their dynamics, and all traffic agents (TA) are implemented in simulation using tools like SUMO and Veins.
- **XIL Components (Gold):** Components such as V2X communication and positioning are handled via an MQTT interface linking real vehicles to the simulator. This allows virtual and real entities to share CAMs and CPMs in real time.
- **Real Components (Red):** All physical vehicle elements (engine, sensors, OBU, communication interfaces) and environmental parameters (lighting, weather, temperature) are implemented and tested in real-world conditions at the test track.

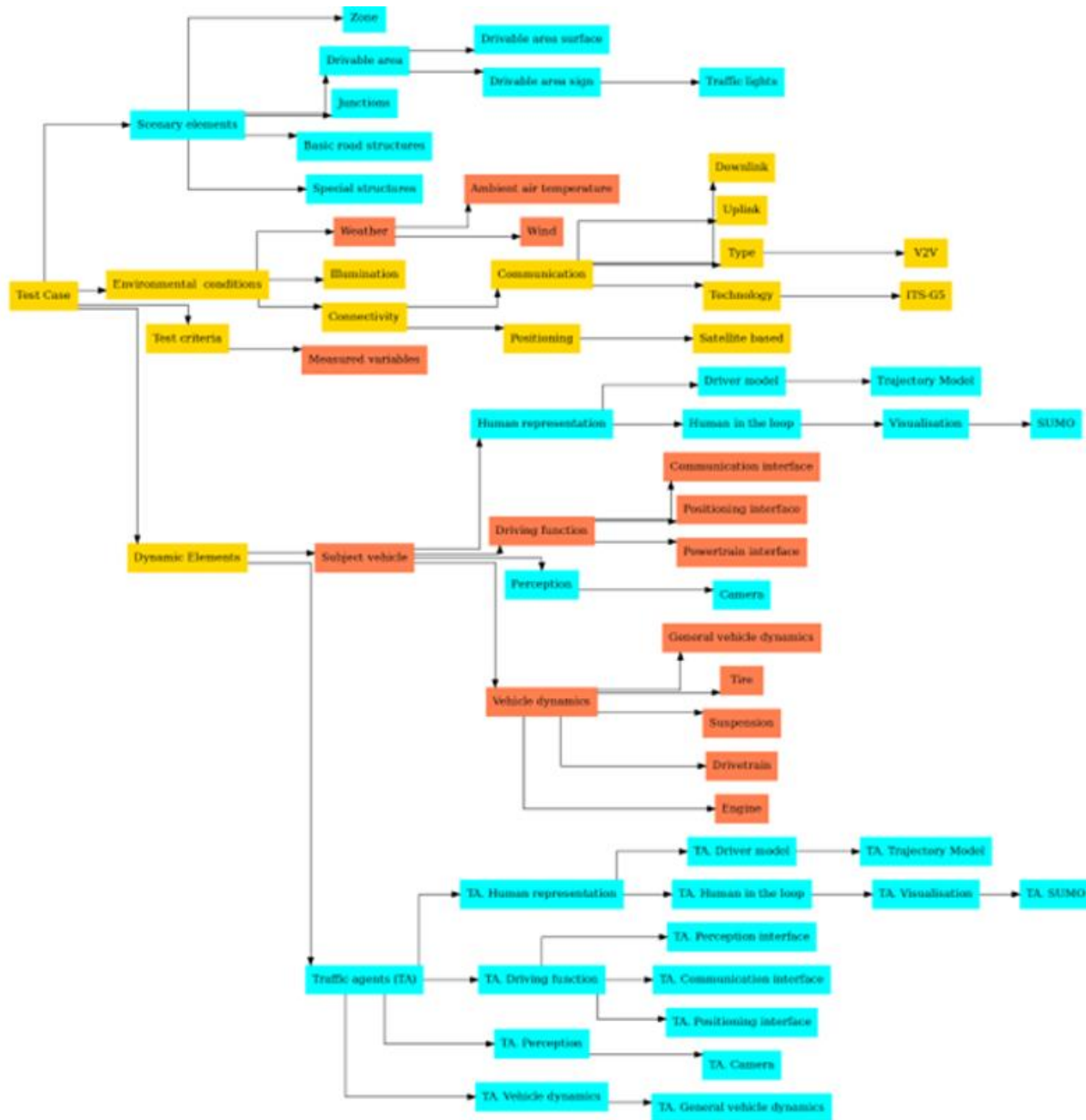


Figure 111: Allocation structure (initially presented as Figure 79 in Sunrise D3.3).

SAF block validation based on sec 2.2

Validation criteria and how they are met was first described in Sunrise D7.2. Since the exact allocation process from Sunrise D3.3 was followed, the validation criteria for this block are all considered met.

3.4.3.3 Execute

UC1.3 consists of 2 partners testing different CP systems. The CP system integrates external object-level data received via V2X communications (ETSI CAM, CPM) from nearby vehicles or a virtual RSU. Two simulation frameworks (OMNET++/SUMO & MSVan3t/CARLA) and two proving grounds are available for test execution (both virtual and hybrid setups supported). The two teams supported testing in both hybrid (one real agent) and virtual test environments while execution validation was performed by comparing results from proving ground and

virtual test environments. Testing frameworks deployed for **execution** of allocated test scenarios are described in Sunrise D4.4 and Sunrise D4.6.

SAF block validation based on sec 2.2

The primary objective of this validation is to ensure that test scenario execution results meet the input requirements of the 'Safety Argument' super-block that follow. Validation criteria and how they are met was first described in D7.2 [4] and complementary information is provided below:

- Capability Assurance: Confirmation that the selected test instance can execute the given test scenario, and that both simulation and physical test setups meet SUNRISE's requirements for reproducibility and reliability.
- SUNRISE detailed scenario allocation process was followed as described by D3.3 [5].
- Collective Perception system under test was put in test in both virtual and PG settings in order to validate its performance in the two settings.
- It was found (ICCS study) that perception of the observer vehicle was somewhat optimistic in the simulation environment leading to CPMs earlier exchange. That led to weighting higher the results from hybrid testing especially for time-critical scenarios.
- It was found (VED study) that hybrid simulation including a network simulator supported realistic delays/latencies with respect to the real world setting.
- Reproducibility tests in CARLA/MsVan3t virtual test environment using 'synchronous' event triggering mode in scenario execution were performed. Only when the results satisfied reproducibility criteria, did the execution proceed.
- Data Collection: Execution must ensure the capture of all relevant data for post-test analysis.
- CPM, Scenario and KPIs data Logging mechanisms were developed by both teams for both virtual and hybrid test environments.
- Completion & Quality: All steps of the test execution process must be finished correctly, and the outcome must feed meaningfully into the evaluation process.
- KPIs were fed into the Test evaluate and Coverage blocks.
- Evidence credibility checks are considered in both virtual and hybrid testing settings, namely analysis of system under test behaviour in virtual versus hybrid setup.
- Feedback Loops: The mechanism must enable necessary adaptations or adjustments in scenarios or objectives based on intermediate findings.
- Scenario adaptations were not needed. Test objectives add specifically the agent's velocities were indeed adapted based on scenario space exploration as an intermediate result of the joint concretization and coverage block, leading to prioritization in testing in hybrid environment of only critical and boundary scenarios.

Virtual test environment

Approach 1

The same set of scenarios were executed in simulation in order to perform scenario space exploration and log the defined KPIs (see Test evaluate section). As shown in Figure 112, two different co-simulation environments were supported one without a network simulator (emulated CPMs) and one with integrating a network simulator too (CPMs generated by a simulation model configured with certain characteristics for transmission).

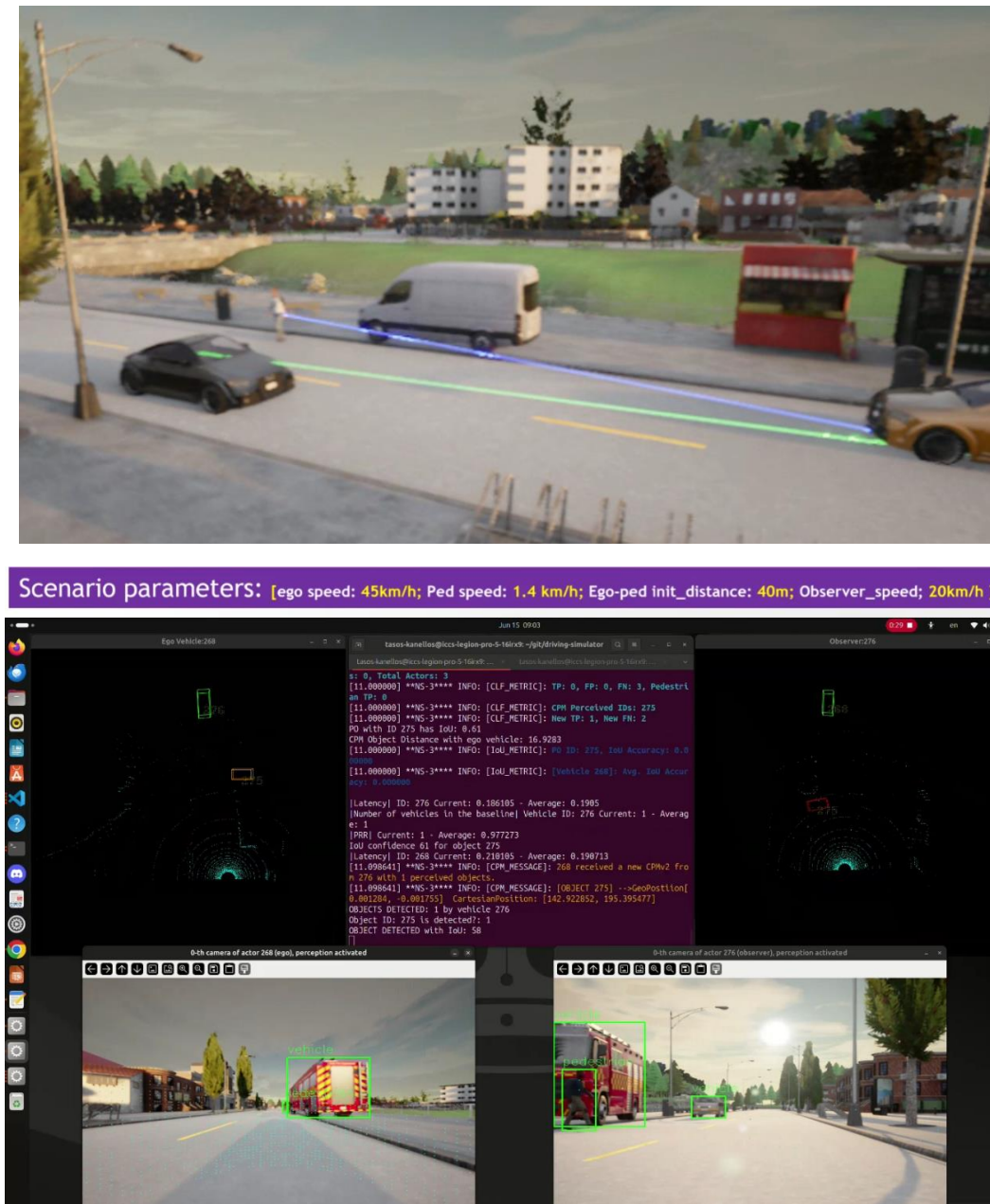


Figure 112: Snapshot from the two virtual testing environments : up, snapshot from ICCS CARLA-ROS custom BEV GUI; down, Snapshot from ICCS Msvan3t/CARLA co-simulation custom GUI.

Approach 2

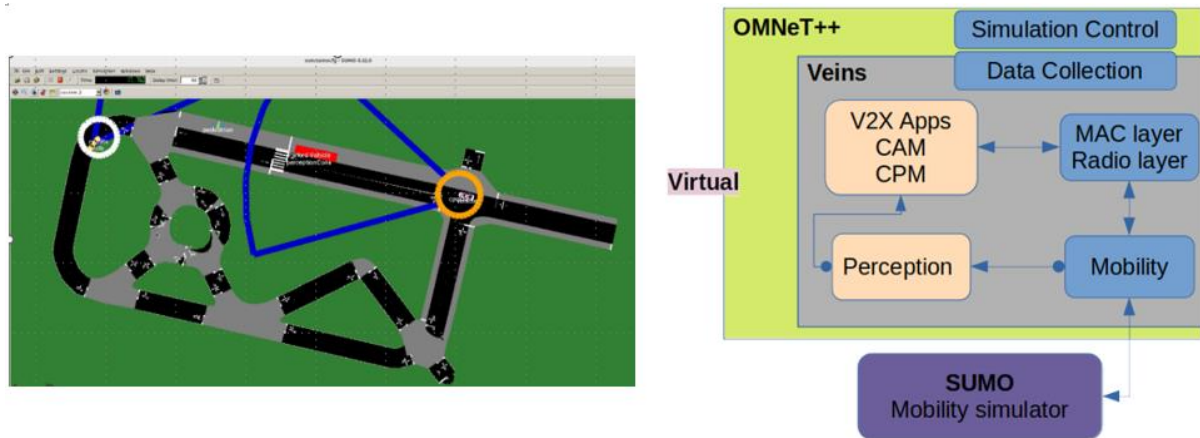


Figure 113 : Virtual test environment architecture and GUI (Approach 2)

The virtual testing framework depicted in Figure 113 integrates multiple simulation tools to evaluate cooperative perception in connected vehicle environments. This framework combines OMNeT++, Veins, and SUMO to create a realistic and modular setup for modeling both perception and communication mechanisms in CCAM scenarios.

At the core of this framework is OMNeT++, a discrete event simulator, where the Veins framework is embedded to simulate Vehicle-to-Everything (V2X) communication. Within Veins, various layers are modeled including the MAC and radio layers, mobility layer, and application layers supporting CAM (Cooperative Awareness Messages) and CPM (Collective Perception Messages), compliant with ETSI TR 103 562. The perception module within Veins enables each vehicle to simulate a 2D sensor model, allowing detection of dynamic objects (like pedestrians) and considering static occlusions such as parked vehicles. When objects are detected within a vehicle's perception cone, CPMs are broadcasted at 10 Hz, mimicking real-world V2V data sharing.

The SUMO (Simulation of Urban MObility) mobility simulator provides the road network and vehicle dynamics, and it is visually represented in the lower half of the diagram. It simulates the positions and movements of all mobile entities, including the ego vehicle, other vehicles, pedestrians, and parked cars. SUMO's GUI also visualizes elements such as perception cones, detected objects, CPM transmission/reception rings, and all entities participating in the scenario.

A specific scenario illustrated in the image involves an ego vehicle; another vehicle labeled v_1, a pedestrian, and a parked vehicle partially occluding the pedestrian from the ego vehicle's field of view. As the simulation runs, v_1 detects the pedestrian using its perception module and transmits a CPM. The ego vehicle, being within the communication range, receives the CPM and updates its local awareness, even though it cannot directly see the pedestrian. Based on this enhanced awareness, the ego vehicle makes an informed decision to slow down or stop, effectively mitigating the collision risk posed by the visual occlusion.

Hybrid test environment

Approach A

A vehicle-in-the-loop test environment was implemented where data from the ego in the test track were transmitted to the cloud-based CARLA simulation environment in real-time during a hybrid scenario execution. As shown in Figure 114, a cloud-based CP testing dashboard was implemented where information from emulated CPMs (all agents except from the ego are simulated in the virtual environment) and the real movement of the vehicle in the test track can be presented during scenario execution.



Figure 114: Snapshot from ICCS hybrid setup (National Technical University of Athens premises and CARLA)

Approach B

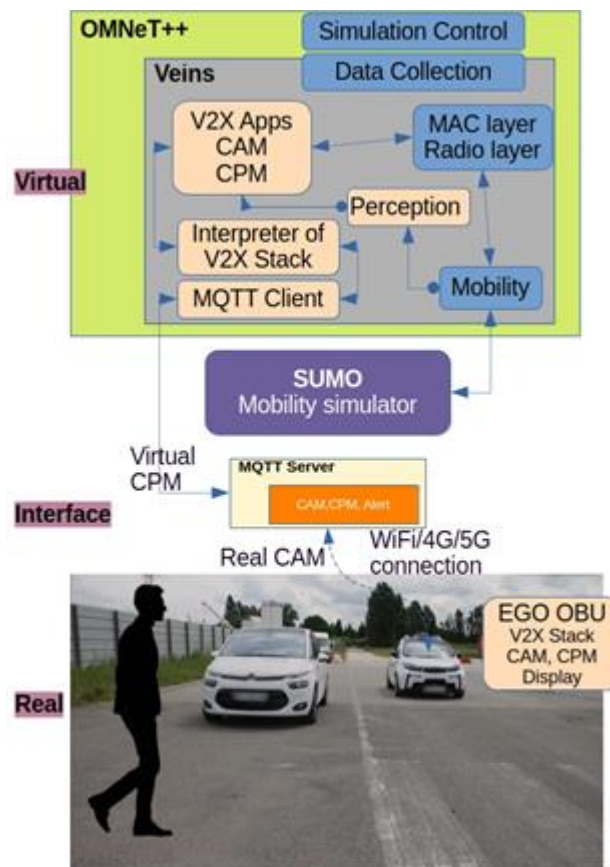


Figure 115: VED Hybrid testing framework for UC1.2A

The **hybrid testing framework** shown in Figure 115 developed by **VED** for **UC 1.3A – Darting-Out Pedestrian** enables real-time **real-virtual integration** to evaluate cooperative perception and decision-making in complex urban environments. This setup connects a **real Automated Driving (AD) vehicle** with a **virtual simulation environment** composed of **OMNeT++**, **Veins**, and **SUMO**, using an **MQTT-based interface** to exchange V2X messages in real time.

In the **virtual domain**, the framework relies on **OMNeT++** as the event-driven simulator, running the **Veins** framework (Figure 116), which includes a modular V2X stack. This stack features **V2X applications** for generating and handling **CAMs** and **CPMs** according to **ETSI TR 103 562**. The **2D perception model** within Veins simulates the vehicle's sensor view, including occlusions from static obstacles like parked vehicles. When a virtual vehicle detects an object (e.g., a pedestrian), it broadcasts **CPMs at 10 Hz**, allowing nearby vehicles to share situational awareness. The **mobility layer**, synchronized with **SUMO**, ensures accurate simulation of all dynamic agents and road elements.

A key feature of this framework is the **MQTT interface**, which bridges the **virtual world** with the **real environment**. An **MQTT server** facilitates communication between the virtual simulation and a **real Ego vehicle** operating on a test track. The **Ego On-Board Unit (OBU)**, equipped with a complete **V2X stack**, receives **virtual CPMs** and transmits **real CAMs** over **Wi-Fi/4G/5G**, enabling live data exchange and coordinated behavior across both domains.

The **scenario** involves a **virtual vehicle (v_1)** detecting a **pedestrian** who is occluded from the **real Ego vehicle** by a **parked vehicle**. Upon detection, **v_1** sends a **CPM** through the virtual V2X stack. This **virtual CPM** is transmitted via the MQTT interface to the Ego OBU. On receiving this message, the **Ego vehicle updates its awareness** of the hidden pedestrian, despite the occlusion. It then reacts—**slowing down or stopping**—to prevent a potential collision, demonstrating **cooperative behavior** through **V2V communication** (Figure 117 and Figure 119).

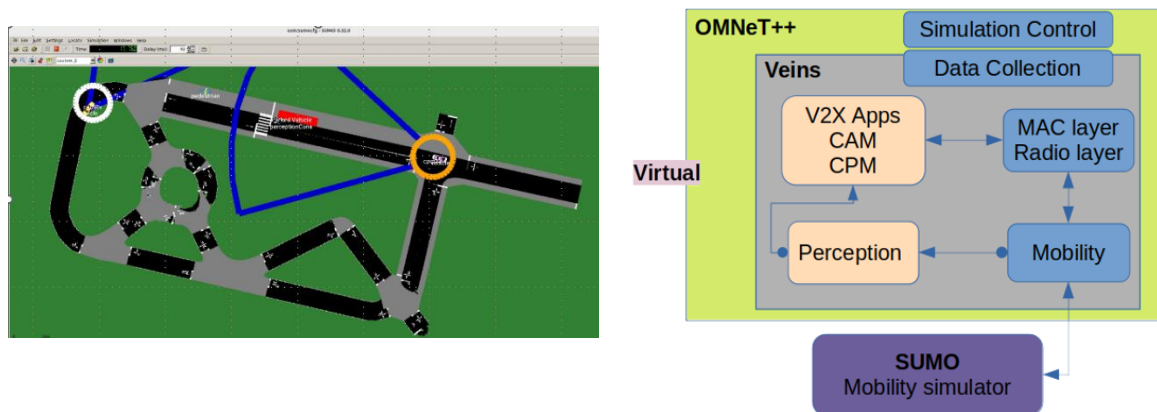


Figure 116: SUMO/OMNET++ structure

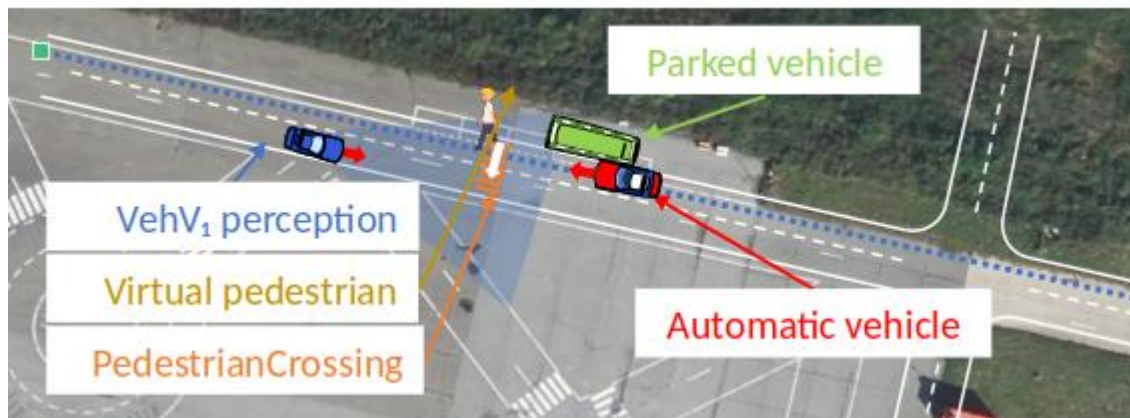


Figure 117: Snapshot from VED hybrid setup (VED premises in Paris and SUMO/OMNET++)

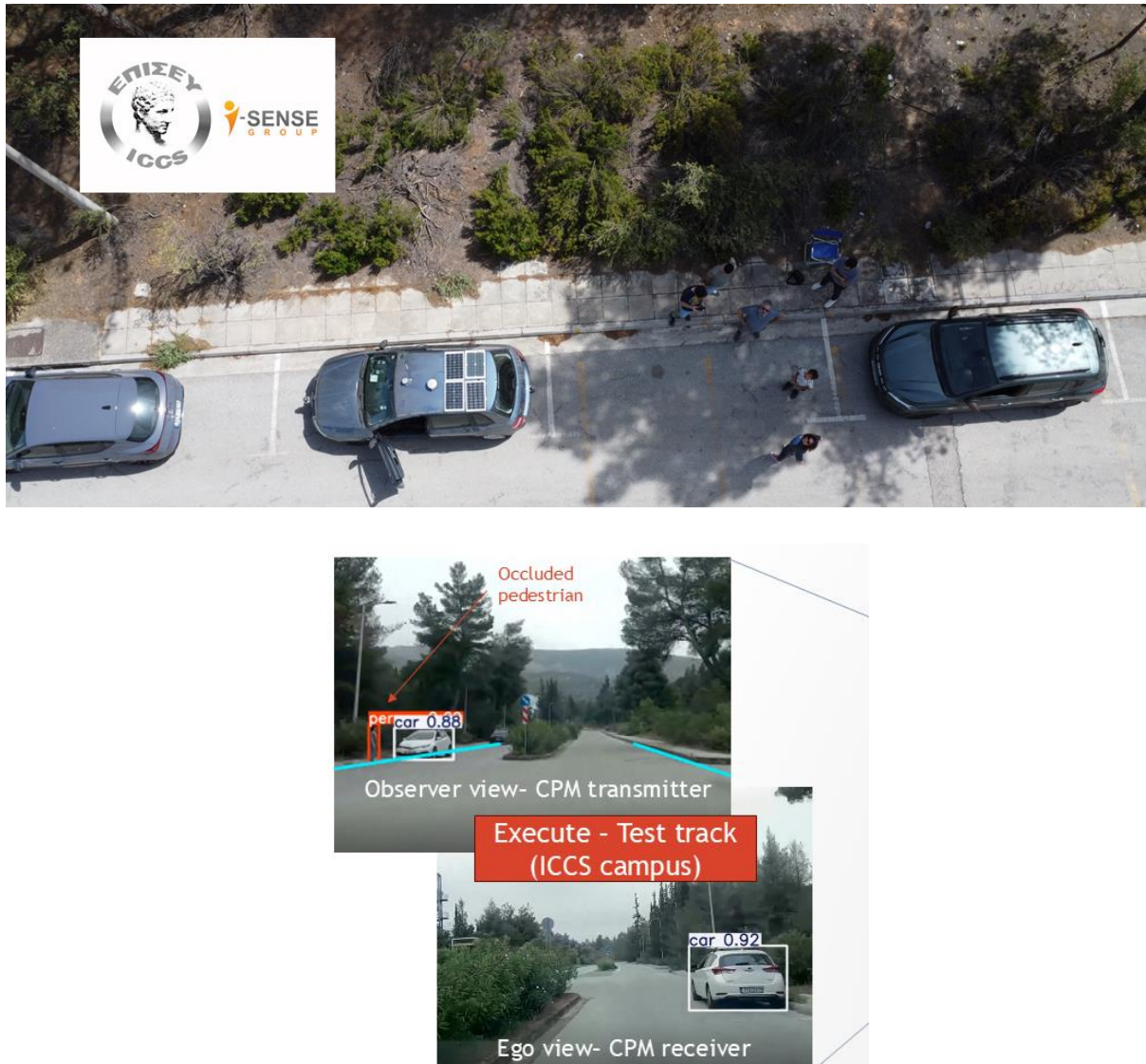


Figure 118: Snapshot from the three virtual testing environments : a) Snapshot from OMNET++/SUMO VED tool
b) Snapshot from ICCS CARLA custom BEV GUI c) Snapshot from ICCS Msvan3t/CARLA ICCS custom GUI

3.4.3.4 Test evaluate

The pass/fail criteria defined were based on popular metrics employed in the AD control and perception testing literature (like TTC and object detection accuracy and confidence respectively) with the addition of connectivity-related metrics that capture the effects of network parameters in the context of this UC, where AD driving and network are co-simulated or tested in a hybrid setup (metrics were described in Sunrise D5.3).

Test scenarios allocated to a hybrid testing environment (XiL) have been validated against their counterparts in simulation. Virtual and hybrid (Vehicle-in-the-Loop) testing results were compared based on the same set of KPIs.

Results from approach 2

In this section, we validate the test evaluate block of the SAF. To do this we identify and implement KPIs (Key Performance Indicators) that evaluate both the system performance and safety-critical behaviors.

To conduct the evaluation, the simulation environment was configured with the following parameters:

- Perception Range: 70 meters
- Camera Field of View: 60 degrees
- CPM (Cooperative Perception Message) Transmission Interval: 0.1 seconds
- Initial Speed of the Ego Vehicle:
- $V_{init} = \{30, 40, 50, 60, 70\} \text{ km/h}$
- Pedestrian Distance (pedD): The pedestrian begins to cross the road at distances ranging from 10 to 30 meters in front of the ego vehicle, in steps of 5 meters. The pedestrian moves at a constant speed of 3 m/s.
- Total Simulation Duration: 30 seconds

The vehicle control logic and overall system behavior are regulated by the following safety-related constraints:

- $\text{SAFE_DISTANCE_PEDESTRIAN} = 3.0 \text{ meters}$
- $\text{MAX_ACCELERATION} = 2.6 \text{ m/s}^2$
- $\text{DESIRED_TIME_GAP} = 1.5 \text{ seconds}$
- $\text{TTC_THRESHOLD} = 2.0 \text{ seconds}$ (Time-To-Collision threshold for triggering speed adjustment)

These parameters are integrated into the decision-making module of the system to ensure timely and appropriate reactions to the sudden appearance of pedestrians. They also serve as foundational settings for validating the safety performance of the overall framework.

Various KPIs were used to quantify perception, CPM exchange, safety and mobility as described below:

○ Perception-Related KPIs

LocalPerceptionCoverage Over Time: Measures the binary detection (0 or 1) of the pedestrian by EGO's and V1's local sensors over time, starting from $t=0$. It indicates the effectiveness of onboard camera sensor in detecting the pedestrian before cooperative data is received (Figure 119 and Figure 120).

CPMPerceptionCoverage Over Time: Tracks the binary detection (0 or 1) conveyed via CPMs, starting from $t=0$, reflecting the success of cooperative perception in sharing pedestrian data between V1 and EGO (Figure 119 and Figure 121). A content of CPM is shown in Figure 121.

Combined Local and CPM Perception Coverage Over Time (EGO, VinitEgo=30 km/h, pedD=15 m)

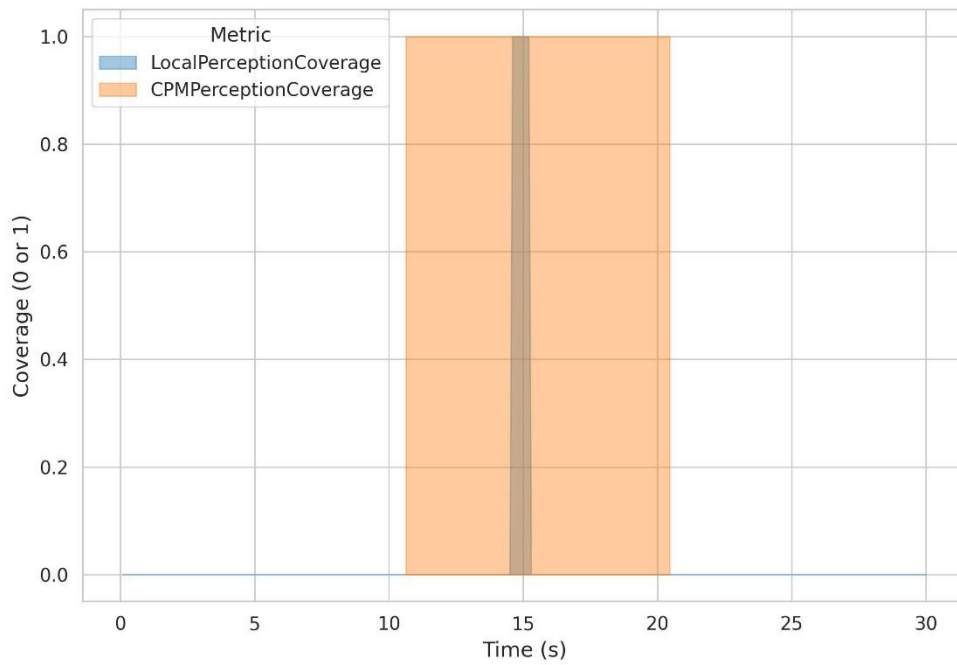


Figure 119: Local and CPM perception coverage of EGO for Vinit 30 km/h and pedD 15m.

Combined Local and CPM Perception Coverage Over Time (EGO, VinitEgo=60 km/h, pedD=30 m)

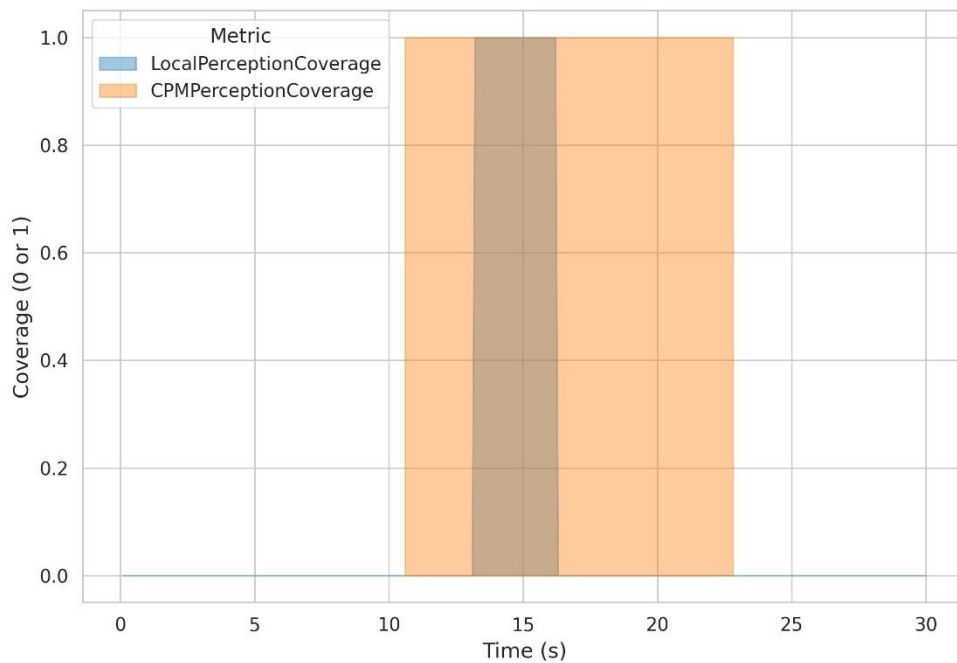


Figure 120: Local and CPM perception coverage of EGO for Vinit 60 Km/h and pedD = 30m.

From Figure 119 and Figure 120, we first observe that the pedestrian information is first obtained by the EGO from the received CPM information. The local perception captures the

pedestrian later. For higher speed and pedD values in Figure 119, the local perception information is little earlier than in Figure 120. However, in both cases the CPM perceptio is earlier. If cpm perception is not present then only with local perception may not give enough time for the EGO to slow down and avoid the collision with the pedestrian.

- **CPM exchange - Connectivity KPIs**

CPM Tx by Vehicle V1: Quantifies the number of CPMs transmitted by V1 (generated CPMs), indicating the frequency of cooperative data sharing (Figure 121). As the pedD increases from 10 m to 30 m the number of transmitted CPMs are increased because for higher pedD the pedestrian starts earlier and stays under the perception of V1 for longer time. And the CPMs are transmitted only if there is perception information to transmit. The number of CPMs transmitted are mostly independent of the Vinit.

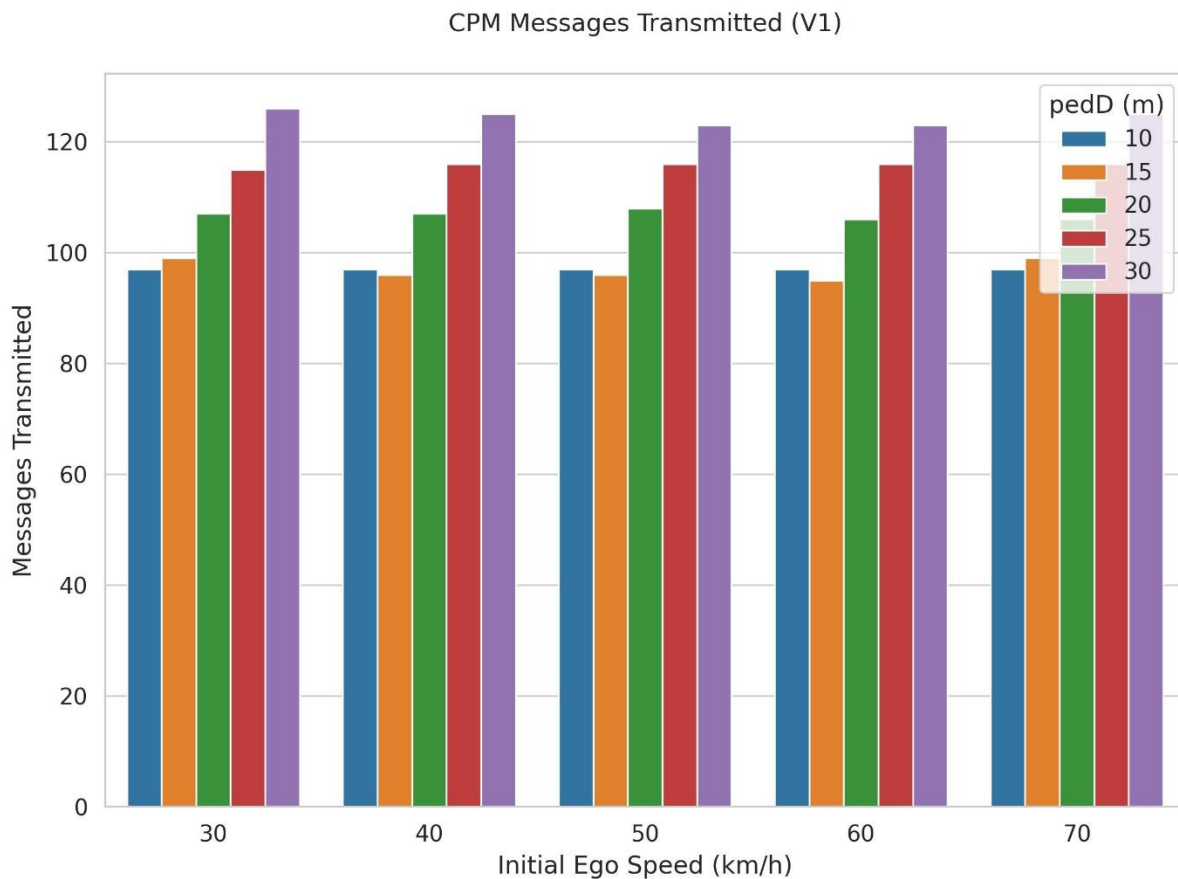


Figure 121: Number of CPM transmitted by the vehicle V1

CPM Rx by Vehicle EGO: Measures the number of CPMs received by EGO (received CPMs), reflecting communication reliability (Figure 122). The number of CPMs received by the EGO are same as the number of CPMs transmitted by V1, which can be verified by comparing the Figure 119 and Figure 120. This is because only V1 transmits the CPMs and there are no interference and collision of packets. Therefore, all the CPMs are received demonstrating very good connectivity and reliable connectivity. However, when EGO also transmits CPMs then there are some CPMs lost due to packet collisions.

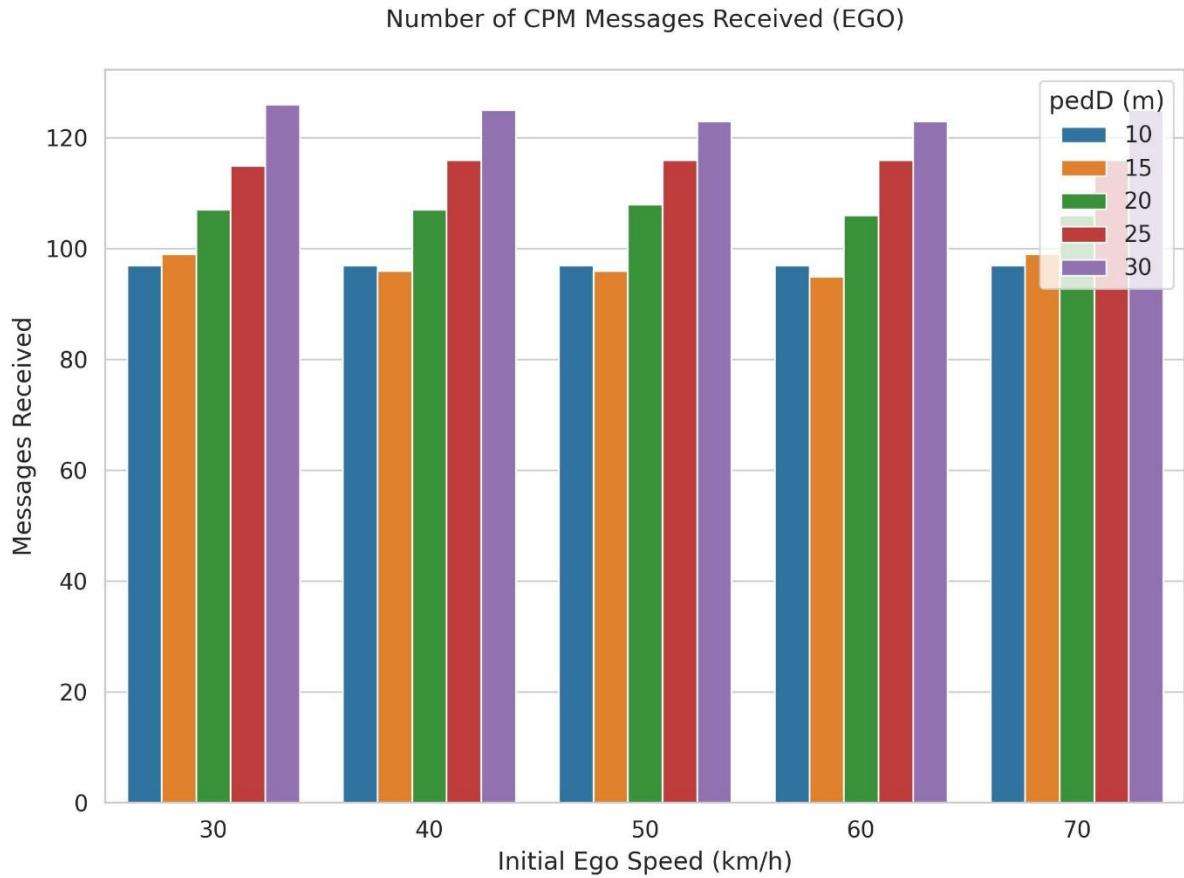


Figure 122: Number of CPM received by EGO.

CPM PDR of EGO: Calculates the Packet Delivery Ratio as the ratio of received CPMs by EGO to generated CPMs by V1 when , assessing the reliability of CPM connectivity (Figure 123). Since, from Figure 119 and Figure 120 the transmitted and received CPMs are equal the PDR of EGO is 1 showing 100% successful CPMs delivered.

In the hybrid tests, where CPMs transmitted by the virtual vehicle V1 are received by the real ego vehicle on the test track via the network, a CPM delivery ratio of **99.71%** was achieved.

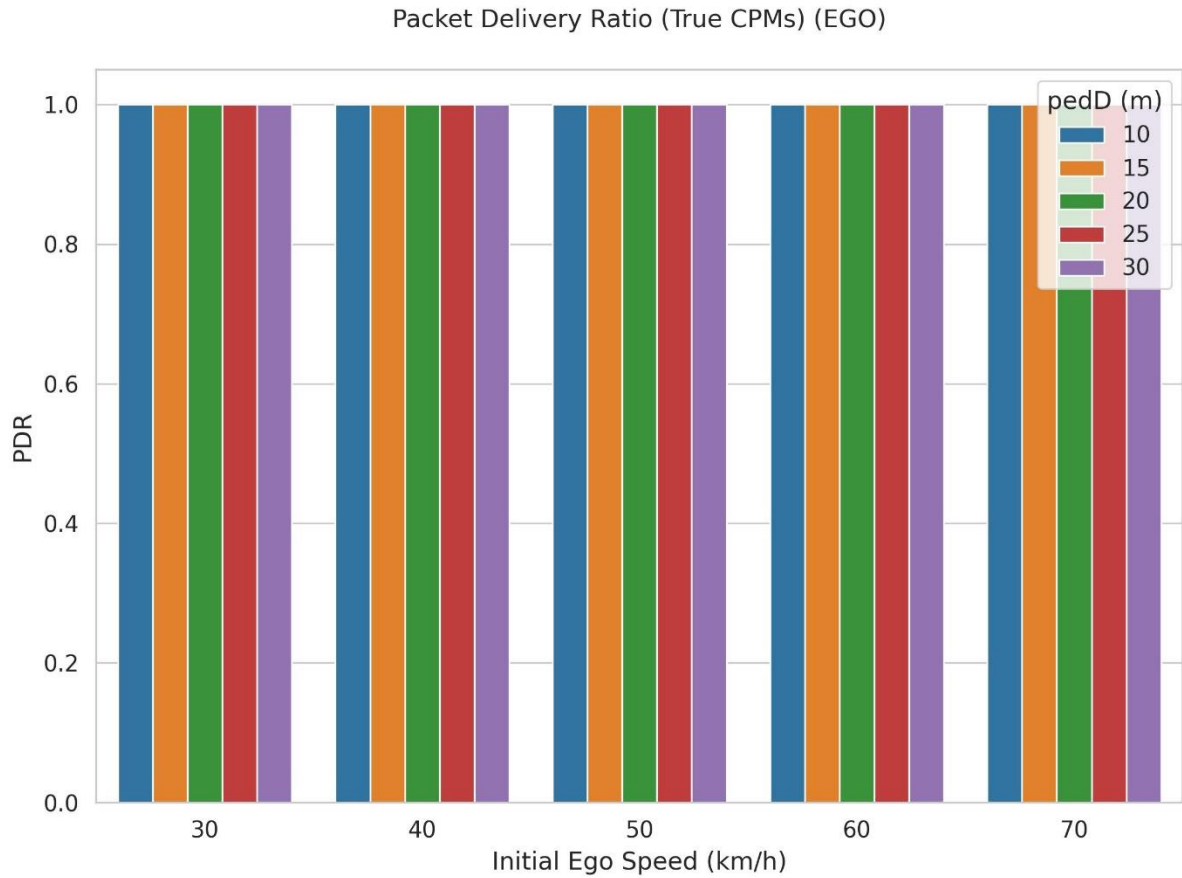


Figure 123: PDR of EGO.

Mean CPM Latency: Computes the average CPM latency across the simulation, indicating overall communication efficiency (Figure 124). The average CPMs latency is around 7.5 milliseconds, which is ideal. However, when EGO also transmits the CPMs then the latency can increase due to contentions at the radio level.

In the hybrid tests, the maximum observed latency was approximately **60 milliseconds**.

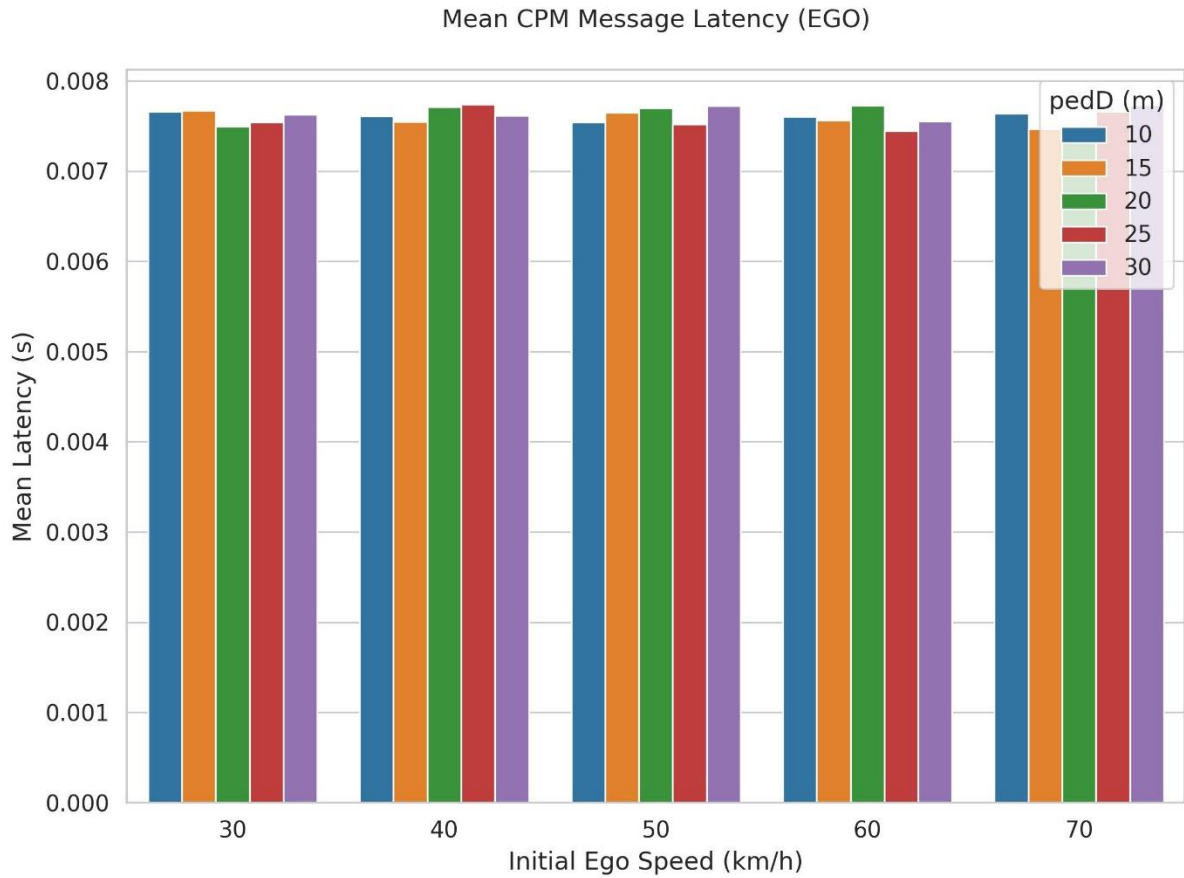


Figure 124: Mean latency of CPM.

○ Mobility KPIs

Speed Variation Over Time: Tracks EGO's and V1's speed (m/s) over time, starting from $t=0$, showing how vehicles adjust speed in response to pedestrian detection (Figure 125 and Figure 126). In Figure 121, which is for Vinit 30 km/h and pedD 15m, the speed of EGO decreases from time 0 and regains normal speed until 7 seconds. Then, the speed of EGO decreases from 16 seconds because of the presence of the pedestrian information from the CPMs. Demonstrating the effectiveness of the CPM in darting-out the pedestrian. Then the speed comes to normal once the pedestrian is passed.

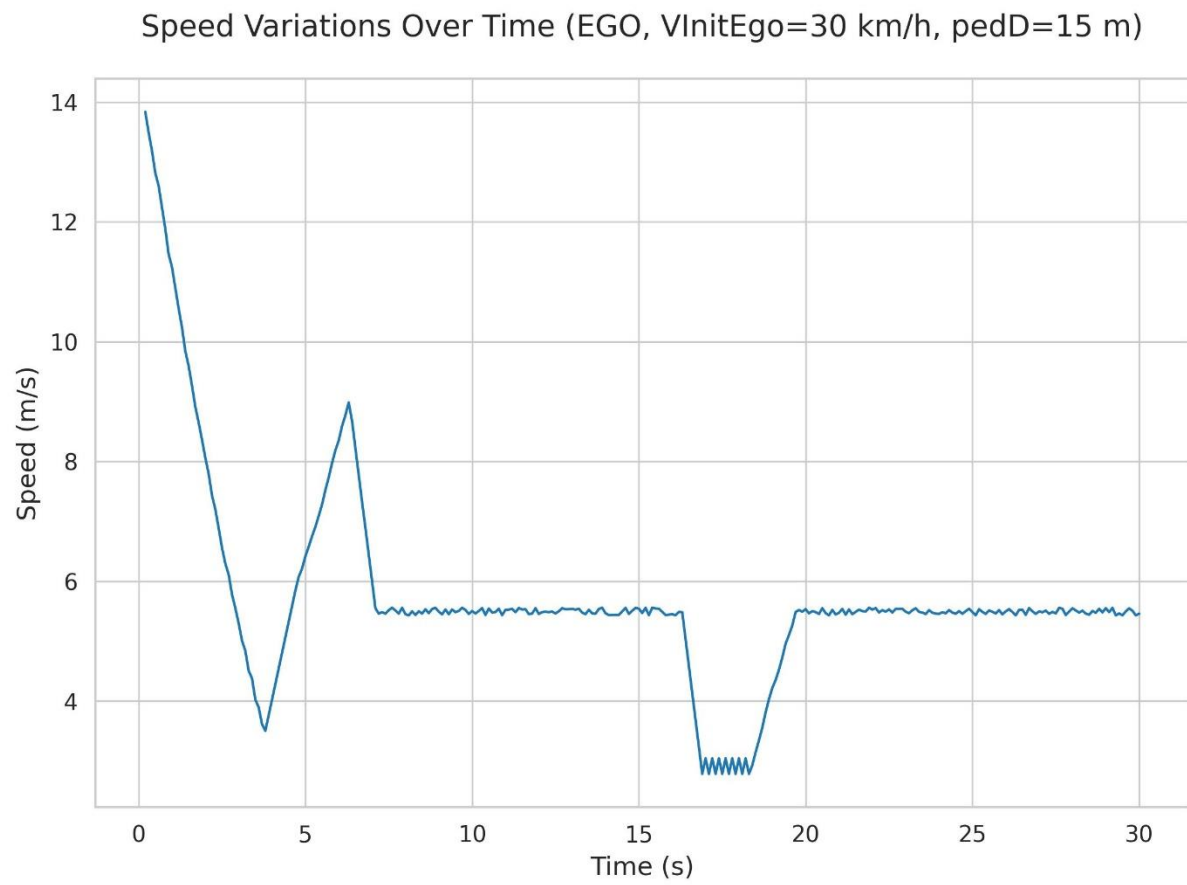


Figure 125: EGO speed for V_{init} 30 km/h and $pedD$ 15m.

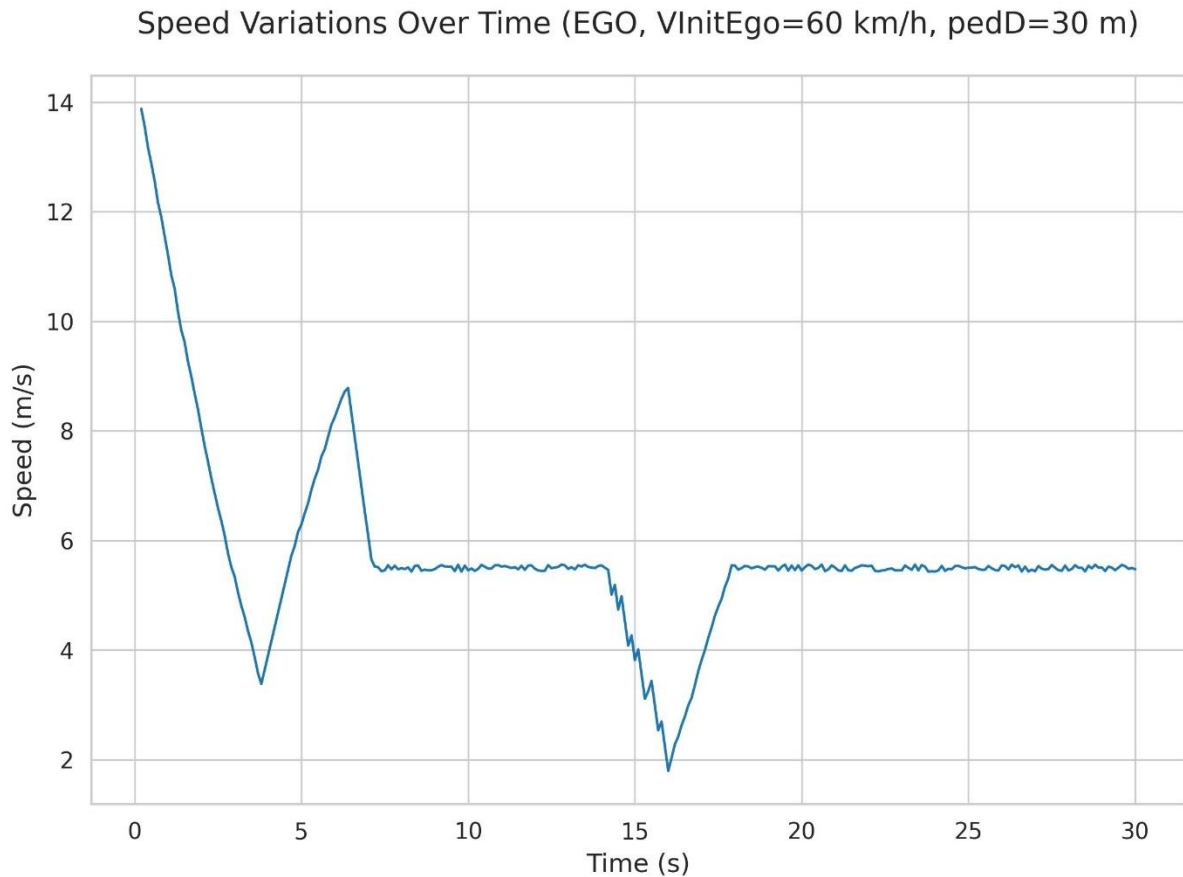


Figure 126: EGO speed for Vinit 60 km/h and pedD 30m.

Similarly, in Figure 126, which is for Vinit 60 km/h and pedD 30m, the speed of EGO decreases from time 0 and regains normal speed until 7 seconds. Then, the speed of EGO decreases from 14 seconds because of the presence of the pedestrian information from the CPMs. Demonstrating the effectiveness of the CPM in darting-out the pedestrian. In this case, the speed changes occur earlier because the pedestrian appears earlier in front of the EGO. Then the speed comes to normal once the pedestrian is passed.

Acceleration Over Time: Measures acceleration (m/s^2) over time, starting from $t=0$, indicating braking or acceleration behaviour (Figure 127 and Figure 128). In these figures, the changes of accelerations are in consistent with the changes of the speed in Figure 125 and Figure 126.

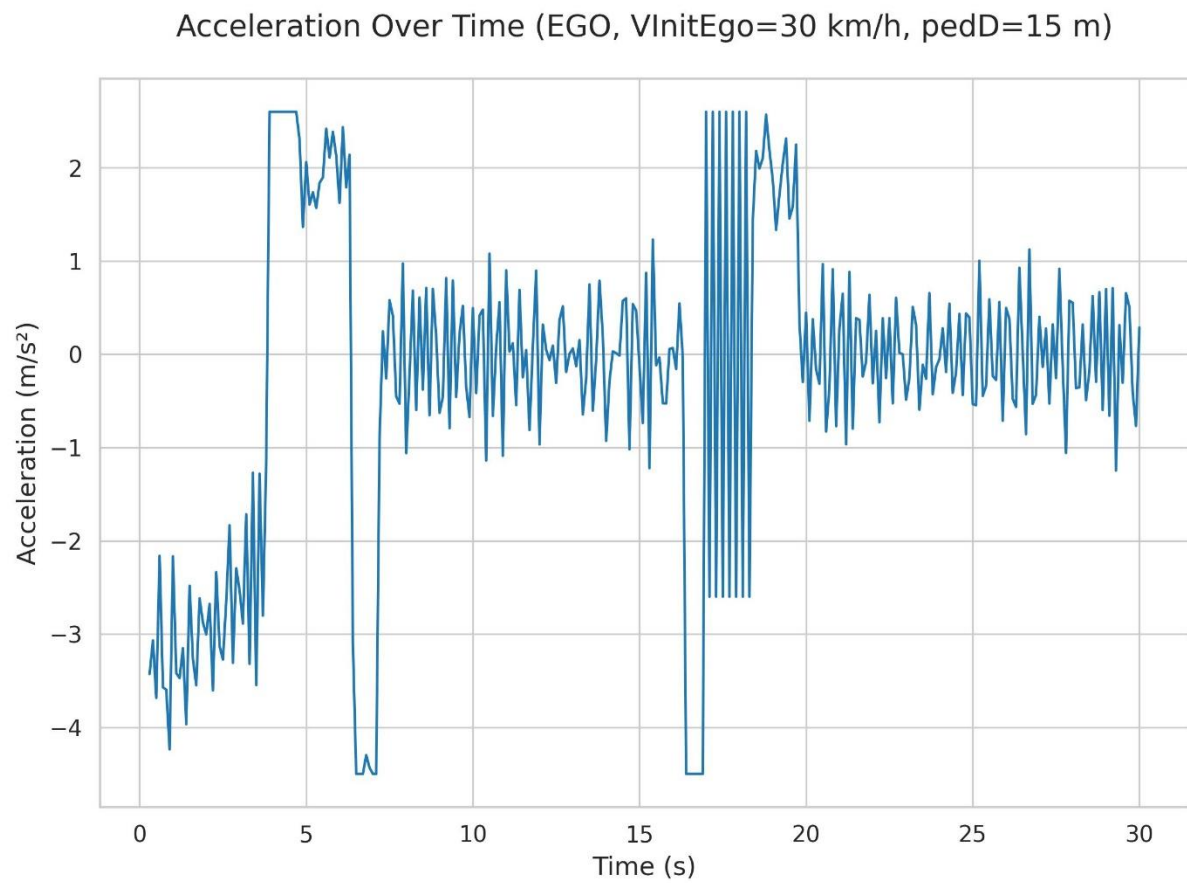


Figure 127: EGO acceleration for Vinit 30 km/h and pedD 15m.

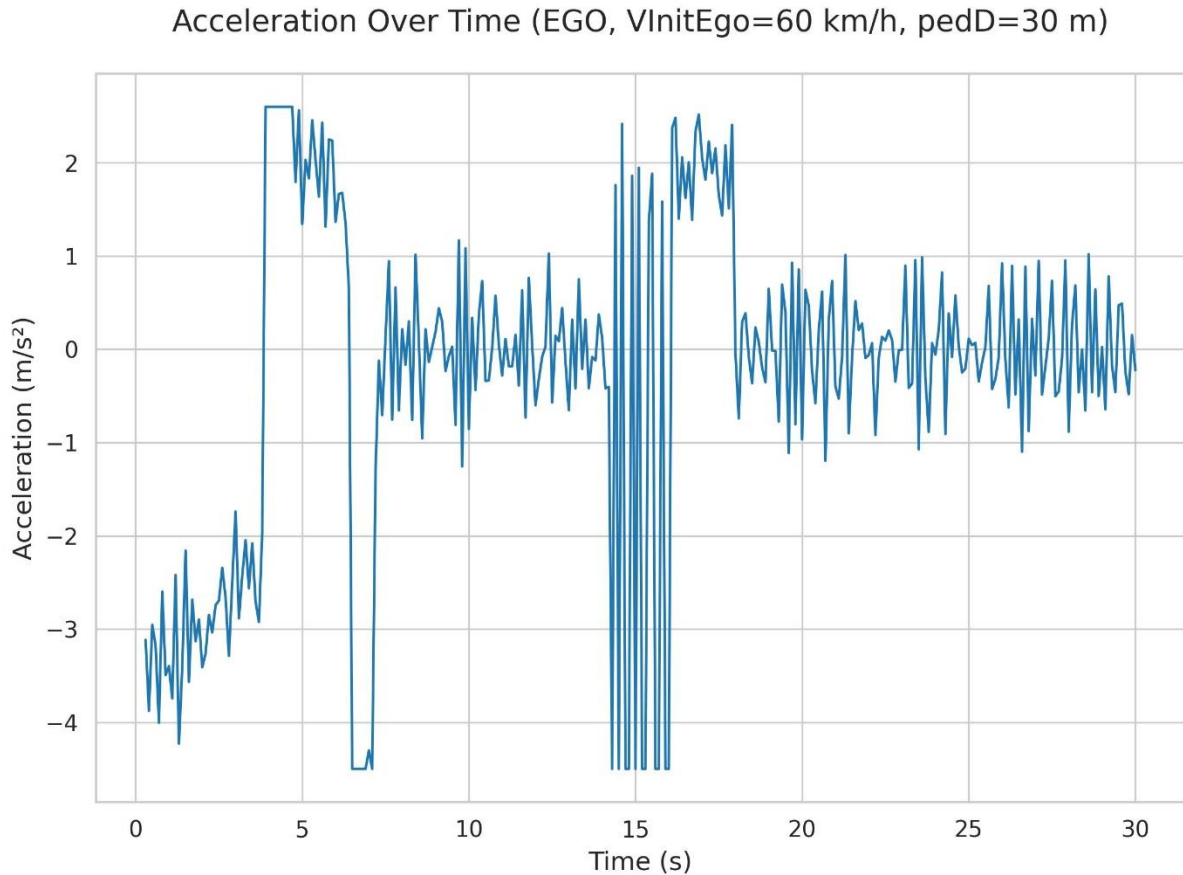


Figure 128: EGO acceleration for Vinit 60 km/h and pedD 30m.

○ Safety KPIs

TTC Over Time: Monitors TimeToCollision (seconds) with the pedestrian over time, starting from $t=0$, assessing collision risk (Figure 129 and Figure 131). In these figures, the TTC shows decreases as the EGO approaches the pedestrian. Then TTC increases because the EGO speed decreases because of the CPM information. Demonstrating the effectiveness of the CPM connectivity in avoiding the collision with the pedestrian. Then the TTC gradually decreases because the EGO speed increases. Then, the TTC gradually increases because the EGO and pedestrian move away from each other.

In Figure 125, the TTC changes happen during 16s to 19s which is consistent with the speed changes in Figure 121. Similarly, in Figure 129, the TTC changes during 14s to 18s, which is consistent with the speed changes in Figure 130.

Time to Collision (TTC) Over Time (EGO, $V_{initEgo}=30$ km/h, $pedD=15$ m)

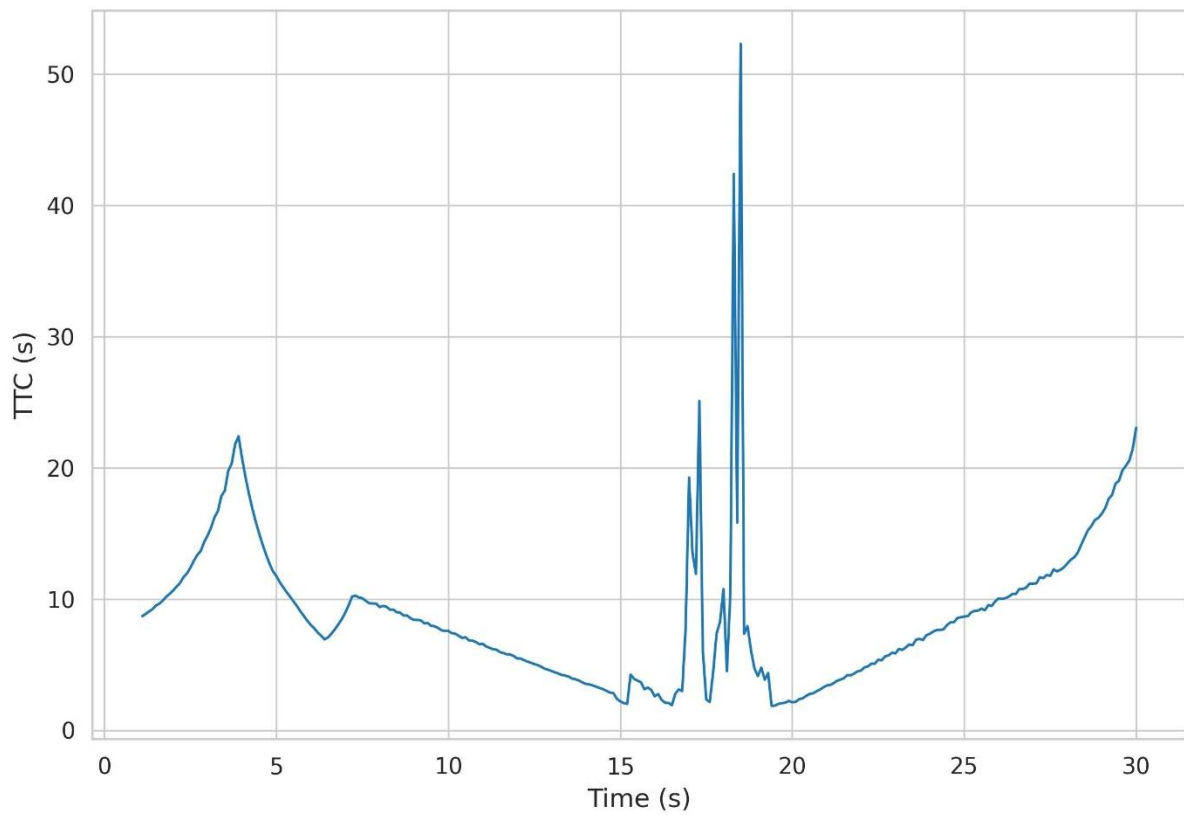


Figure 129: EGO TTC for V_{init} 30 km/h and $pedD$ 15m.

Time to Collision (TTC) Over Time (EGO, VInitEgo=60 km/h, pedD=30 m)

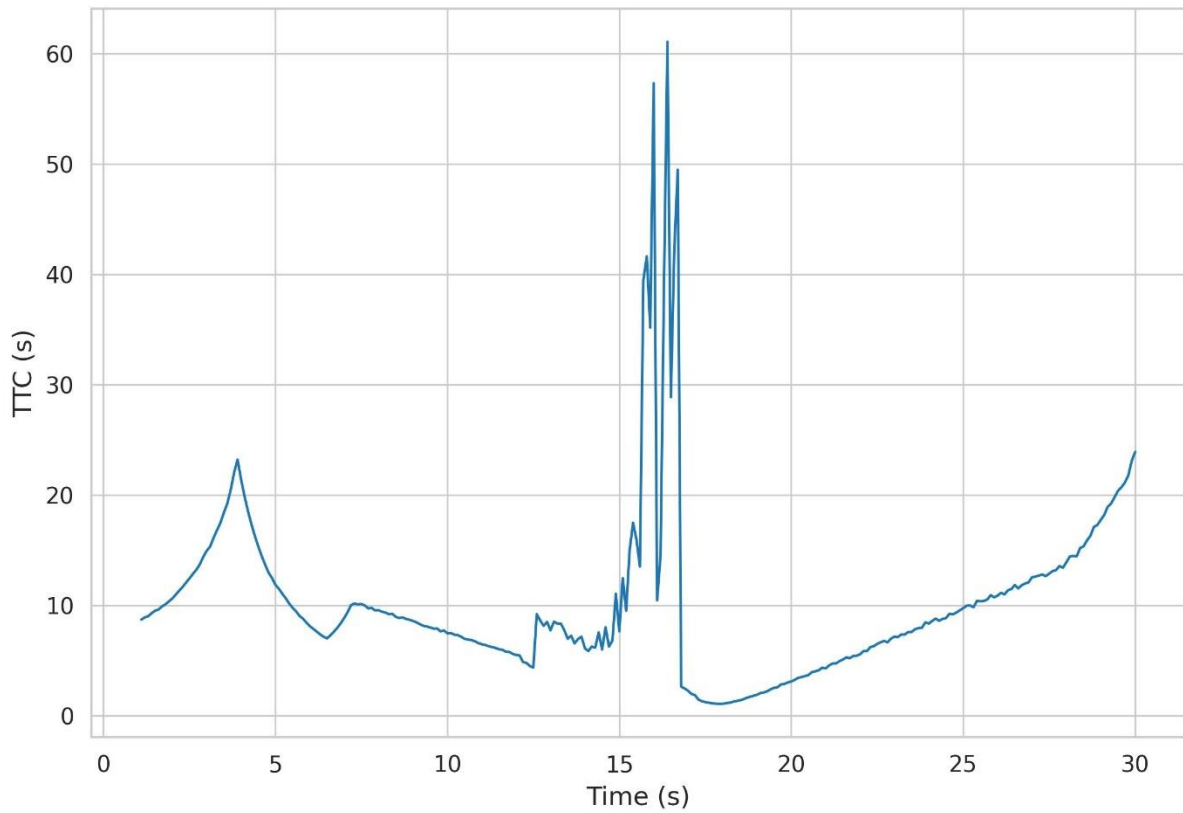


Figure 130: EGO TTC for Vinit 60 km/h and pedD 30m.

Max Jerk: Records the maximum absolute jerk value, indicating the worst-case ride comfort impact (Figure 131). The max jerk is around 70 m/s³. For pedD 10m, the max jerk is lower because the pedestrian appears behind the EGO because it does not have enough time to come in front the EGO and it passes. For the case of Vinit 40 m/s and pedD 20m, the EGO does not receive CPM on time and the pedestrian passes without collision.

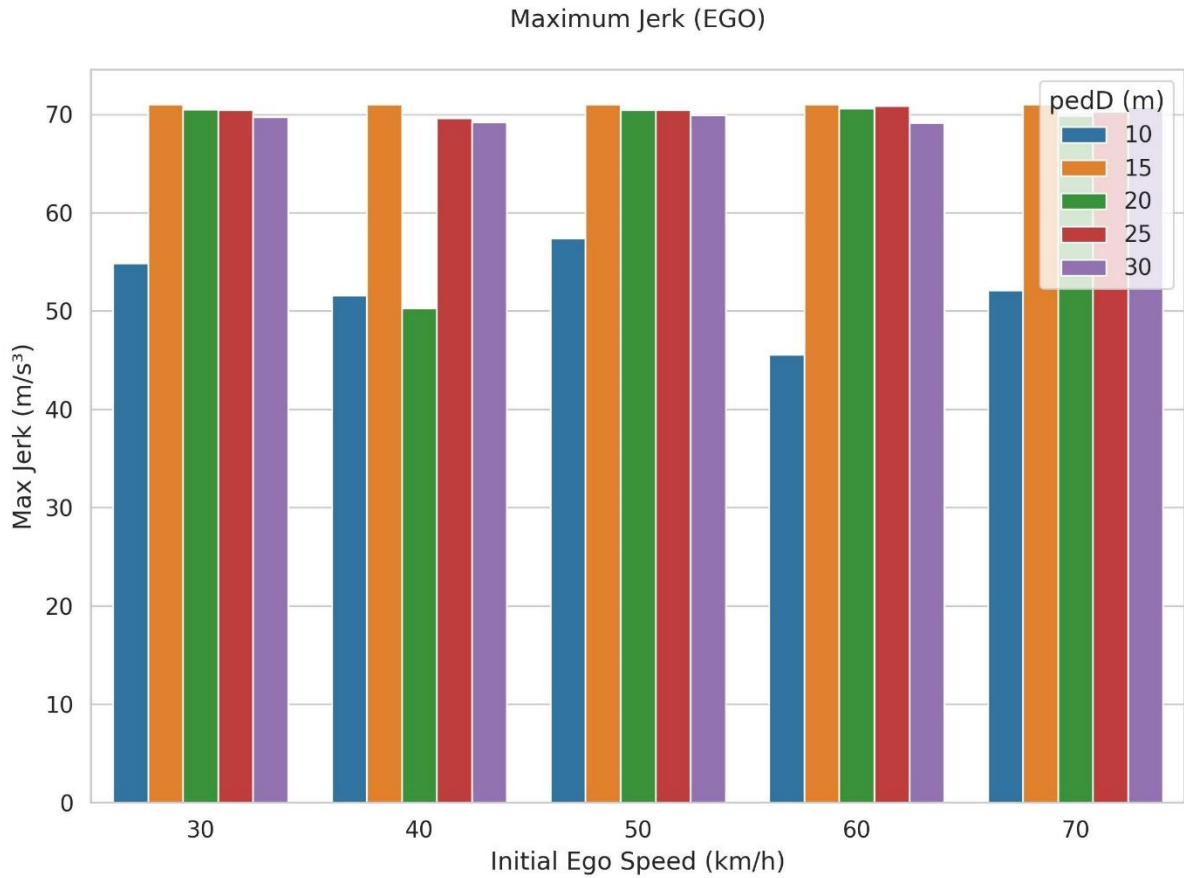


Figure 131: EGO maximum jerk.

Collision outcomes: For each value of V_{init} and $pedD$, we ran the simulation and determine if there is a collision. In Figure 132, the resulting collision/no-collision scenarios are shown. For V_{init} 40 m/s, 50 m/s, and 60 m/s; and $pedD$ 15m the collisions occur. Because the pedestrian comes directly in front of the EGO and the EGO does not get enough time to slow down and avoid the collision. However, for $pedD$ lower values, the pedestrian appears after the EGO has passed, so there are no collisions. For higher values of $pedD$ the CPMs information helps to avoid the collisions.

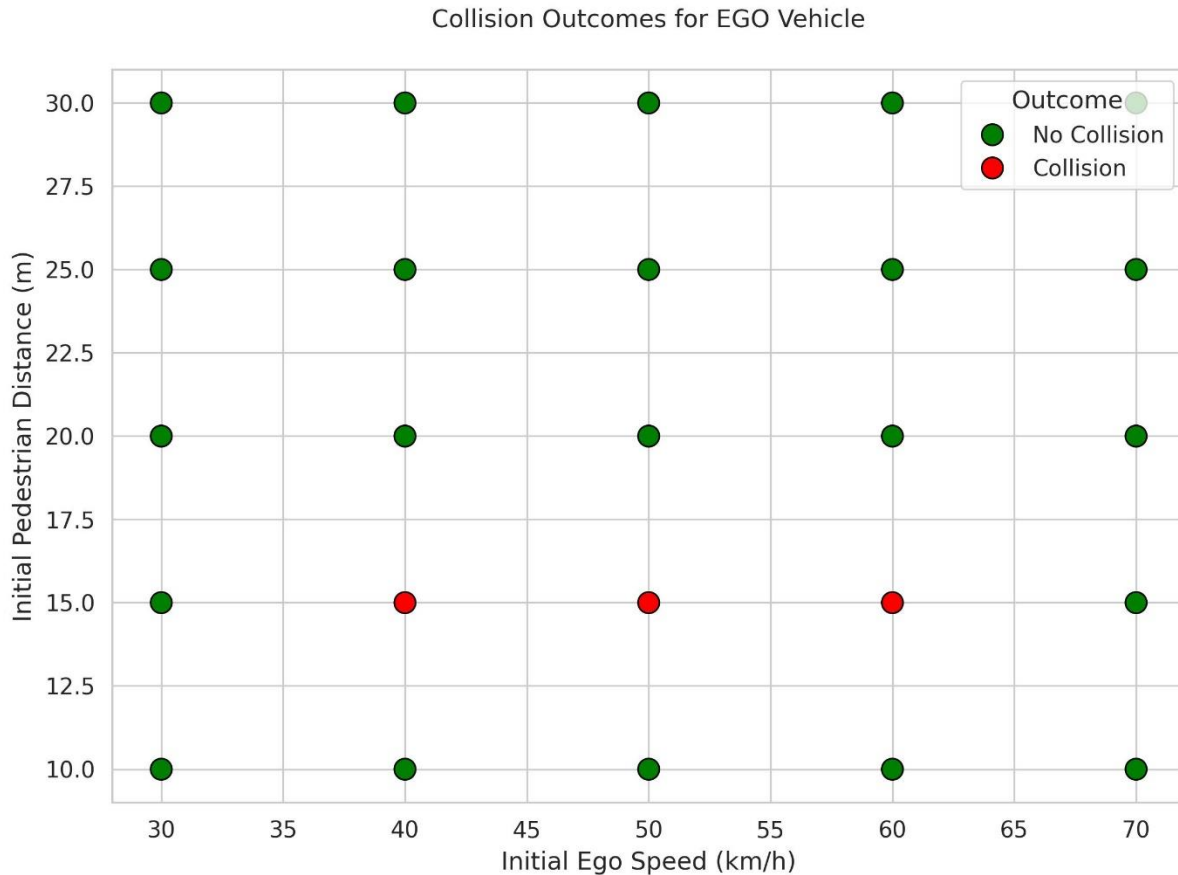


Figure 132: EGO collision outcomes.

These KPIs collectively validate the system's ability to detect the pedestrian, communicate effectively, adjust dynamics, and ensure safety.

SAF block validation based on sec 2.2

Validation criteria and how they are met was first described in Sunrise D7.2 and complementary information is provided below:

- **Metric-Based Assessment:** Each test execution is assessed against clearly defined pass/fail criteria derived from test objectives. Three types of metrics are used to quantify collective perception, AV control and connectivity performance.
- **Traceable Evidence:** All evaluations are traceable to the executed scenarios.
- **Statistical Rigor:** based on high number of concrete scenarios tested (see coverage section), statistical guarantees of safety can be established at least for the scenarios that are not close to the pass/fail boundary.

3.4.3.5 Coverage

Scenario space coverage on the logical scenario level is assessed by using the results of the critical scenario generation probabilistic process developed in T3.3 by ICCS (Sunrise D3.4. High scenario space coverage in the virtual setup and possible test scenario re-allocation is achieved through a Gaussian Process-based concretization method that classifies the entire scenario space into ‘pass’, fail’ and ‘boundary’ regions. The proposed method allows for estimation (with certain confidence) of the pass/ fail probability of unseen concrete scenarios without the need for execution/simulation and guides scenario generation towards new scenarios that are close to the pass/fail boundary, which are the hardest to predict their outcomes (high uncertainty). The coverage validation is performed by executing random scenarios in the estimated pass, fail and boundary regions and evaluating them in simulation to verify pass or fail decision. The method is described in detail in D3.4.

ODD coverage analysis is also performed by counting the set of attributes covered through virtual testing, following the process developed in Sunrise D3.4. Finally, as a joint task between ‘Coverage’ and ‘Safety case building’ blocks (see ‘Safety Case’ sub-block), **safety case requirements’ coverage** is partially addressed by analyzing the output of critical scenario generation when different pass/fail criteria apply (perception vs. control – focused metrics).

Safety case building aims at safety case requirements’ coverage and follows a modular approach where both independent subsystems’ testing as well as end-to-end system testing is considered. It includes, in alignment with the general guidelines of the ISO34505, a) scenario space coverage results’ analysis based on the results from ‘Coverage’ sub-block b) the analysis of critical scenario generation when different pass/fail criteria apply (perception vs. control – focused metrics) b) critical scenario prioritization based on criticality and pass/fail classification uncertainty c) dynamic re-allocation of selected prioritized test scenarios from virtual to hybrid test environment and d) a method to combine metrics from different SuT subsystems and results coming from virtual and hybrid test environment based on the analysis provided in Sunrise D4.6 . *Please note that in the context of this UC, the ultimate goal that is to combine results from different test environments to argue ‘macroscopic’ safety margin violation for one logical scenario is only partially covered since the complete analysis that would include scenarios complexity and exposure as extra metrics to these of criticality and coverage is here missing.* It also includes analysis of safety case artifacts, namely the analysis of test scenario repeatability during execution and system under test reaction times in virtual and hybrid setup to understand the sim2real gap.

SAF block validation based on sec 2.2

- Diversity: The selected scenarios must cover a wide range of ODDs, behaviors, and environmental conditions.
- Automatic concretization mechanism has been implemented for changing agents’ behaviour, while different ODDs could be configured manually via an .ini file.
- Completeness: Coverage analysis must check for gaps, redundancies, and sufficient representation of critical and failure scenarios.

- Scenario space coverage bigger than 96% sufficiently covers the scenario space with failures while boundary scenario identification supports representation of interesting scenarios to be prioritized for higher fidelity testing.
- Metrics & Indices: Use of quantitative measures (e.g., Scenario Relatedness Index) to monitor and report scenario diversity and completeness.
- Metrics for scenario criticality and pass/fail uncertainty were used.

3.4.3.6 Decide

The evaluation of Use Case 1.3 highlights the effectiveness of cooperative perception in enhancing urban AD safety. The safety argument asserts that cooperative perception enables EGO to detect the pedestrian earlier and adjust its dynamics to avoid collisions. Key evidence and key takeaways includes:

- Perception: High CPMPerceptionCoverage indicate reliable pedestrian detection (Figure 119 and Figure 120).
- CPM Connectivity: High PDR of EGO in Figure 119, with low CPMLatency in Figure 119, ensure timely data exchange.
- Mobility: Smooth speed (Figure 125 and Figure 126) and acceleration profiles show effective braking.
- Safety: Higher TimeToCollision in Figure 129 and Figure 130 confirms maneuvers of slowing down to avoid collision. The MaxJerk indicates the discomfort experienced the EGO passenger.

These results underscore the importance of cooperative perception in challenging urban scenarios, with detailed insights provided by individual plots for each KPI and parameter combination.

However: Based on the process of the safety case building on this specific logical scenario (darting out pedestrian) and paying special attention to the discrepancies between the virtual and hybrid CPM generation as reported by ICCS study, the decision on the collective perception system under test is that more testing is needed and we cannot deduct safety based on one logical scenario.

SAF block validation based on sec 2.2

Validation criteria for this block are considered partially met since only evidence-based transparency is supported by this proof of concept which we do not consider mature enough to be considered for compliance checks.

3.4.4 Key take aways

- Co-simulation setup integrating network aspects made possible

- SUNRISE's simulation-first approach used to derive test scenario allocation with hybrid components: Test scenario allocation and re-allocation if needed in higher fidelity test environment.
- Gaussian process works well for criticality-driven scenario sampling and pass/fail boundary discovery in scenario spaces of small dimensions
- Safety evidence: Microscopic evaluation (on individual scenarios) of pass/fail results
- Risk-oriented testing: Boundary scenarios identification
- Test scenario prioritization (based on scenario criticality, exposure and complexity as per ISO 34505)
- Modular approach: Independent subsystems & End-to-end system testing (Open-Loop / Closed-Loop)
- Verification of simulation results through Hybrid testing
- Safety case pass/fail binary result

3.4.5 Deviations from D7.2

None. All SAF blocks supported as planned. Additionally, proving ground testing was also included although not planned from Sunrise D7.2.

3.5 Use Case 2.1: Traffic jam AD validation – Safety assessment & Decision making

3.5.1 Use Case Overview

The Traffic Jam AD Validation (TJC) use case focuses on the validation of an Automated Driving (AD) function in dense traffic conditions, primarily targeting UN-R 157 Automated Lane Keeping Systems (ALKS) on motorways or motorway-like roads [6]. This use case builds on the ERTRAC Connected, Cooperative and Automated Mobility Roadmap (2022), specifically addressing UC ID 2.1. It aims to establish a hybrid validation framework that integrates both virtual simulations and physical testing, enabling efficient test case generation, scenario-based model development, and the creation of assessment metrics that reflect realistic driving behaviors under traffic jam conditions.

The AD function under test, manages the full driving task including both longitudinal and lateral dynamics of the ego vehicle during congested traffic with the human driver in the role as fallback ready user able to take over the driving task within 10 seconds [6]. The AD function performs acceleration, deceleration, braking, and steering maneuvers while ensuring compliance with key constraints such as maximum allowed speed within the operational speed, safe following distance, and lane centering. The system is also expected to execute critical tasks like emergency braking and pedestrian avoidance, highlighting its capability to function reliably without constant driver supervision during its operational speed in high-density traffic scenarios.

A wide range of representative scenarios are covered within the validation process to ensure robustness and functional completeness, including: TJC speed limit adaptation to the vehicle ahead, cut-in and cut-out handling, blocked lane by stationary object, start moving after full stop, regular and emergency deceleration, pedestrian detection and avoidance, as well as stable lane keeping while navigating curves. The use case not only aims to verify the system's functional safety and performance but also contributes to the broader goal of establishing a scalable and repeatable validation methodology for next-generation automated driving systems.

Objective:

UC2.1 focuses on validating AD behavior by optimizing the workflow from scenario creation (Query & Concretise), execution (Execute), evaluation (Test Evaluation), to ensure robust collision avoidance against challenges like cut-ins and VRUs.

3.5.1.1 Covered aspects of the SAF

The table below shows the contributions to the SAF block validation per partner for UC2.1.

Table 15: Contributions to the SAF validation per Partner in UC2.1.

Partner/ Block	AVL	BASt	CAF	TNO	UoW
SUNRISE DF				x	x
Query & Concretise	x			x	x
Allocate	x	x			
Execute	x	x	x		
Test Evaluate	x	x			
Coverage					
Safety Case					
Decide		x			

UC 2.1 https://youtu.be/jS_r-44K5GM?t=2

3.5.1.2 Safety case setup

3.5.2 Overview of tested scenarios

TNO and the **University of Warwick (UoW)** performed scenario generation for use case 2.1. This generation was based on the key functional scenarios, terms use cases below in **Error! Reference source not found.** These 12 functional scenarios are specified originally in Sunrise D7.1. These scenarios were identified as key test cases for traffic jam AD by expert judgement rather than by following any specific or automated scenario generation methodology. These scenarios were created in using Streetwise (TNO) and defined in UoWs Scenario Description Language (SDL) level 2 (UoW) and translated into ASAM OpenSCENARIO. For execution in simulation environments, ASAM OpenSCENARIO (XOSC) and ASAM OpenDRIVE (XODR) files are generated for these scenarios. The scenarios are defined as logical scenarios, using ranges for any key parameters.

The scenarios were executed from both scenario databases in simulation so that a comparison could be drawn between the results of both through the same execution frameworks, as detailed in 3.4.3.2.

3.5.3 SAF Block demonstrations

3.5.3.1 SUNRISE DF

The scenarios, which are summarised in **Error! Reference source not found.**¹⁶, have been crafted as a scenario abstraction adjustment from the functional scenarios defined in Deliverable D7.1 [3]. The scenarios are stored in ASAM OpenScenario and ASAM OpenDrive, satisfying the interoperability requirement of section 2.2.1 for the validation of the SAF (page 33). The following section summarises the scenarios which were accessed via the SUNRISE DF for this use case. Scenario A is crafted to test the TJC's ability to adapt to a new speed limit and a slowing lead; core variables captured as ranges in the logical variation were the speed of the ego and lead vehicles, and the distance to the speed sign. Scenario B tested the TJC's ability to react to the changing speed of a lead vehicle, whose speed changed during the scenario; key variables were the vehicles' speeds, the differential of these values, and the initial distance separating them. Scenario C tests the TJC's ability to respond to a cut-in driver with lead vehicles ahead; the key variables for this scenario include the initial positions, velocities, and the triggering distance for the initiation of the manoeuvre. Scenario D addresses the situation where a lead vehicle in the ego lane changes lanes, potentially revealing new objects or gaps ahead; variables include lateral position and timing of the lane change, and distance headway. Scenario E considers a blocked lane of travel due to a stationary object or road hazard; key parameters include the position of the obstruction, and the ego vehicle's speed and deceleration rate. Scenario F captures the restart of traffic from a standstill, with the ego vehicle following a previously stationary target vehicle; relevant variables include the gap to the lead vehicle and following distance. Scenario G simulates a decelerating lead vehicle detected within system range where no emergency manoeuvre is required; key variables include deceleration rate, and initiation of the stop. Scenario H introduces a crossing pedestrian that does not require evasive action; parameters involve pedestrian trajectory and time-to-arrival margin. Scenario I intensifies this by simulating an imminent collision with a rapidly decelerating lead vehicle, requiring emergency intervention; critical parameters include deceleration rate, and trigger thresholds for emergency braking. Scenario J extends this further by simulating the aftermath of such an emergency stop where the risk has subsided, assessing the system's capacity to resume control; variables include stopping distance, residual motion, and clearance time. Scenario K repeats the crossing scenario but includes a sudden reappearance of the pedestrian, demanding an emergency response; key factors include occlusion timing, visibility delay, re-entry timing, and collision likelihood. Lastly, Scenario L addresses lateral control by testing lane keeping while navigating a curve; variables include curvature radius, initial starting distance and lateral offset.

These scenarios, and variations thereof are stored in both Safety Pool™ SCDB and Streetwise DB. Through inclusion in the Sunrise project efforts have been made towards API connection with the centralised DF to meet the requirements of 2.21 in terms of accessibility and compliance.

3.5.3.2 Query

This use case demonstrated the query portion of the SAF. The scenarios were uploaded both the Safety Pool™ and Streetwise scenario database. Since the Data Framework was still underdevelopment, scenarios were retrieved from individual SCDBs directly.

With in the Safety Pool™ scenario database appropriate ODD and behaviour labels were utilised in the search which retrieved the 11 logical scenarios. The ODD and behaviour labels were sourced from deliverables D7.1 and D7.2, combining ODD, system requirements and behaviours which are traceable through documentation, satisfying requirement 2.2.2 a of the SAF validation requirements. This work demonstrates the successful storage and query of relevant logical scenarios using scenario tags, OpenLABEL based query and API based retrieval. These logical scenarios were provided to partners for concretisation and subsequent execution.

The Figure 133 below exemplifies this process of query within Safety Pool™ scenario database. The query is demonstrated in the lower half of the frame is represented by OpenLabel tags, which have been combined to represent appropriate search which filtered only the relevant scenarios for this use case. This demonstrates the fulfilment of the requirements of 2.2.1 b) and c), and 2.2.2 a) and b) of the SAF validation.

The screenshot displays the 'Private Scenario Library' interface within the Safety Pool™ Database. The top navigation bar includes 'Home', 'Scenarios', 'Test Suites', 'Testbeds', 'Users', and 'Roles'. The main content area shows a list of scenarios under the 'SUNRISE Demo' library. The list includes three scenarios: '2_1A-AdaptingSpeedToNewSpeedLimit', '2_1B-AdaptingSpeedToLeadVehicle', and '2_1C-TargetVehicleCutIn', all with version 1.0 and status 'Active'. Below the list, a search bar shows 'UseCaseDemo' and 'Libraries: SUNRISE Demo'. A green banner indicates 'Your search found 11 scenarios - click here to view'. The bottom section, titled 'Tags', shows a hierarchical structure for 'ODD' (Dynamic Elements, Environmental Conditions, Scenery) and 'Behaviours' (Communication, Motion). The 'Search Criteria' pane on the right shows a complex query using OpenLabel tags, including conditions for 'Maximum Allowable Speed', 'Vehicle', 'Lane change', 'Decelerate', and 'Drive'.

Sc...	Description	Version	Status	View
106840	2_1A-AdaptingSpeedToNewSpeedLimit	1.0	Active	View
106841	2_1B-AdaptingSpeedToLeadVehicle	1.0	Active	View
106842	2_1C-TargetVehicleCutIn	1.0	Active	View

Figure 133: Scenario library and scenario query panes within Safety Pool™ SCDB.

For the Streetwise database the logical scenarios are translated to scenario categories available including the appropriated tags. Since the Streetwise database does not store the actual scenario, but uses a parameterized approach, the parameter distribution for this scenario categories can be computed. In the next step the desired ranges for the parameters that make up the scenario are selected. Finally, a Monte Carlo approach is used to generate the actual concrete scenarios which are in this case outputted. In the figures below (s. Figure 134, Figure 136) the User interface to generate the concrete scenarios can be seen followed by a visual representation of the 1D plot of the different parameter distributions.

streetwise

Logout

STREETWISE TEST CASE SELECTION

VEHICLE TYPE
CategoryM_PassengerCar

ROAD TYPE
Highway

GEOGRAPHIC AREA
☐ Western_Europe
☒ EU_WestCentral

TAGS FOR SCENARIO CATEGORY SELECTION

☐ RoadUserType_Vehicle
☐ RoadUserType_CategoryM_PassengerCar
☐ RoadUserType_CategoryM_Bus
☐ RoadUserType_CategoryM_Minibus
☐ RoadUserType_CategoryN_LCV
☐ RoadUserType_CategoryN_LGV

SCENARIO CATEGORY

Name	Exposure (n/h)	Observations
<input checked="" type="checkbox"/> Lead vehicle decelerating	7.4	7205
<input type="checkbox"/> Vehicle overtaking ego vehicle	0.7	715
<input type="checkbox"/> Cut-in in front of ego vehicle	3.2	3084
<input type="checkbox"/> Ego vehicle performing lane change with vehicle behind	1.3	1307
<input type="checkbox"/> Lead vehicle cruising	7.0	6826
<input type="checkbox"/> Ego vehicle approaching slower lead vehicle	2.5	2445
<input type="checkbox"/> Ego vehicle driving in lane without lead vehicle	8.0	7781
<input type="checkbox"/> Cut-out in front of ego vehicle	3.9	3760
<input type="checkbox"/> Lead vehicle accelerating	7.0	6832
<input type="checkbox"/> Ego merging into an occupied lane	0.4	375

SCENARIO DETAILS
Lead vehicle decelerating

DESCRIPTION

Dynamic actors:

- Ego: Driving in lane.
Driving forward.
- Target: Driving in same lane as ego.
Lead vehicle, driving forward, performs a deceleration activity.

Static environment:

- Main environment: Motorway
- Infrastructure: Not specified
- Region: EU west central

Environmental conditions:

- Light: Not specified
- Weather: Not specified

TAGS

- RoadType_Highway
- OSMType_PrincipleRoad_Motorway
- Region_EU_WestCentral
- RoadUserType_CategoryM_PassengerCar

PARAMETERS

- ego initial longitudinal velocity [m/s]
- target initial relative longitudinal velocity [m/s]
- target longitudinal velocity decrease during deceleration activity [m/s]
- target mean deceleration during deceleration activity [m/s²]

Figure 134: Test case selection pane in Streetwise SCDB.

STREETWISE TEST CASE SELECTION

PARAMETER DISTRIBUTIONS Lead vehicle decelerating

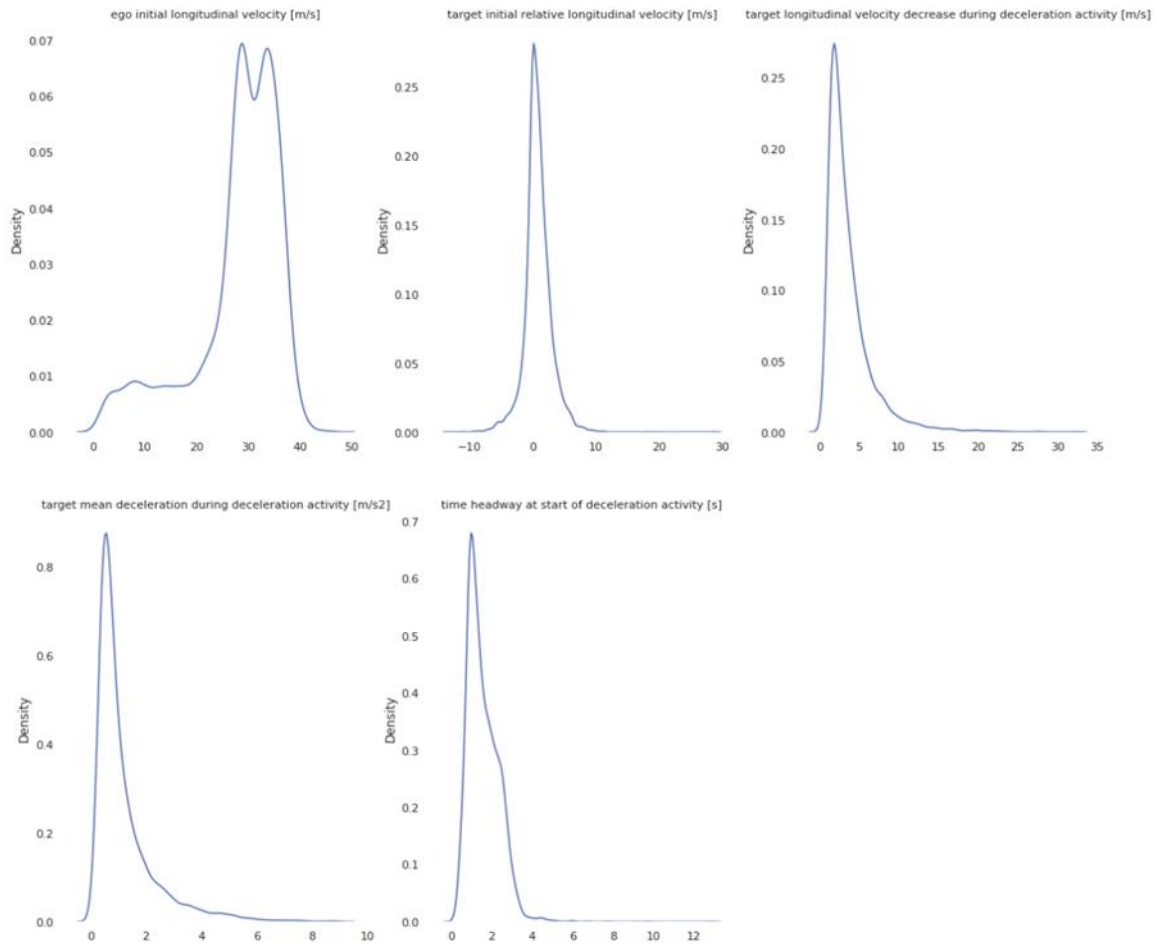


Figure 135: Parameter distribution for an example test case in Streetwise SCDB.

3.5.3.3 Concretise

Scenarios are retrieved in ASAM OpenSCENARIO (.xosc) format and they can be imported to AVL Scenius SCDB for concretization, execution and evaluation. Each scenario that has parameters are considered as a logical scenario. Within the logical scenario model, the parameter space can be constructed by setting minimum-maximum values and step sizes as shown in Figure 137.

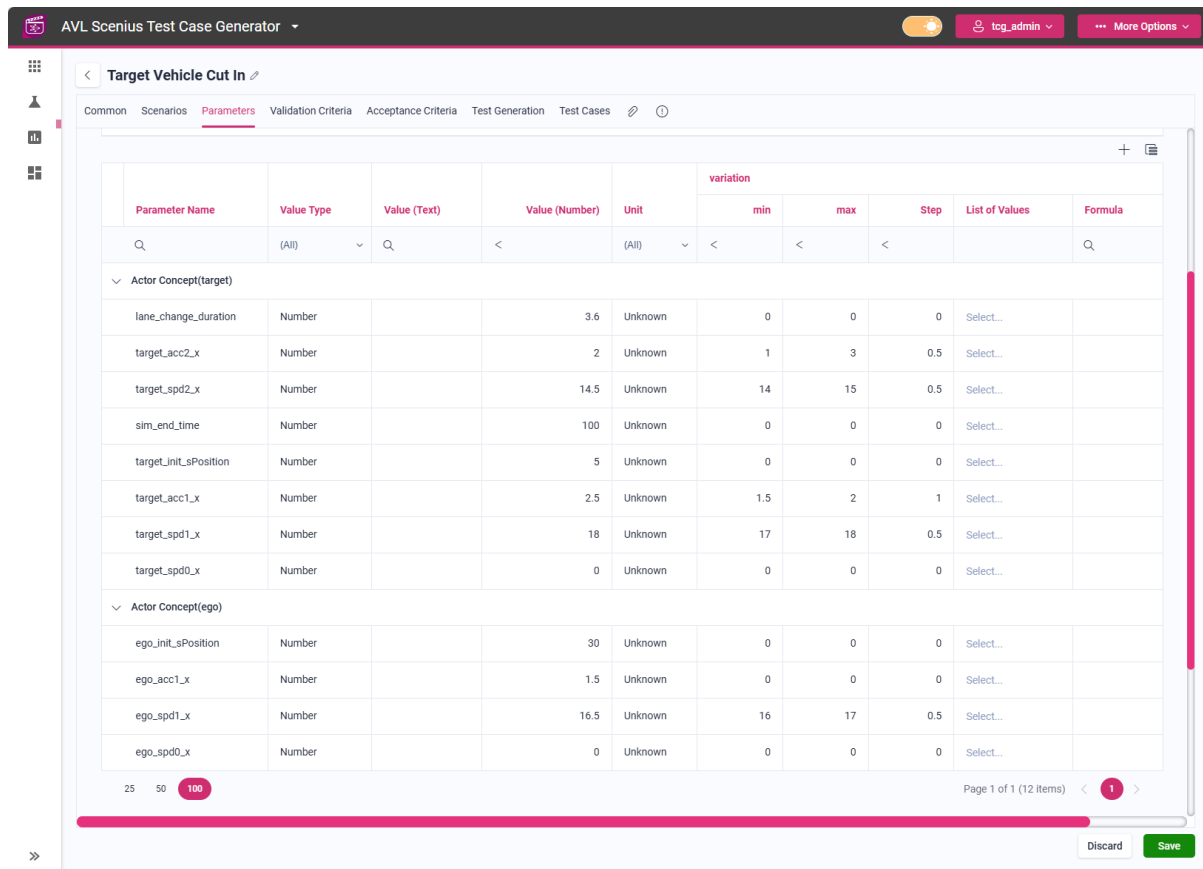


Figure 136: Scenario Parameter Space in AVL Scenius.

After configuring the parameter space, concretization be initiated with different test case generation methods which can be seen below in Figure 137. This Test Case generator generates the concrete scenario artifacts in ASAM OpenSCENARIO format, which are then ready for execution in supported simulation environment.

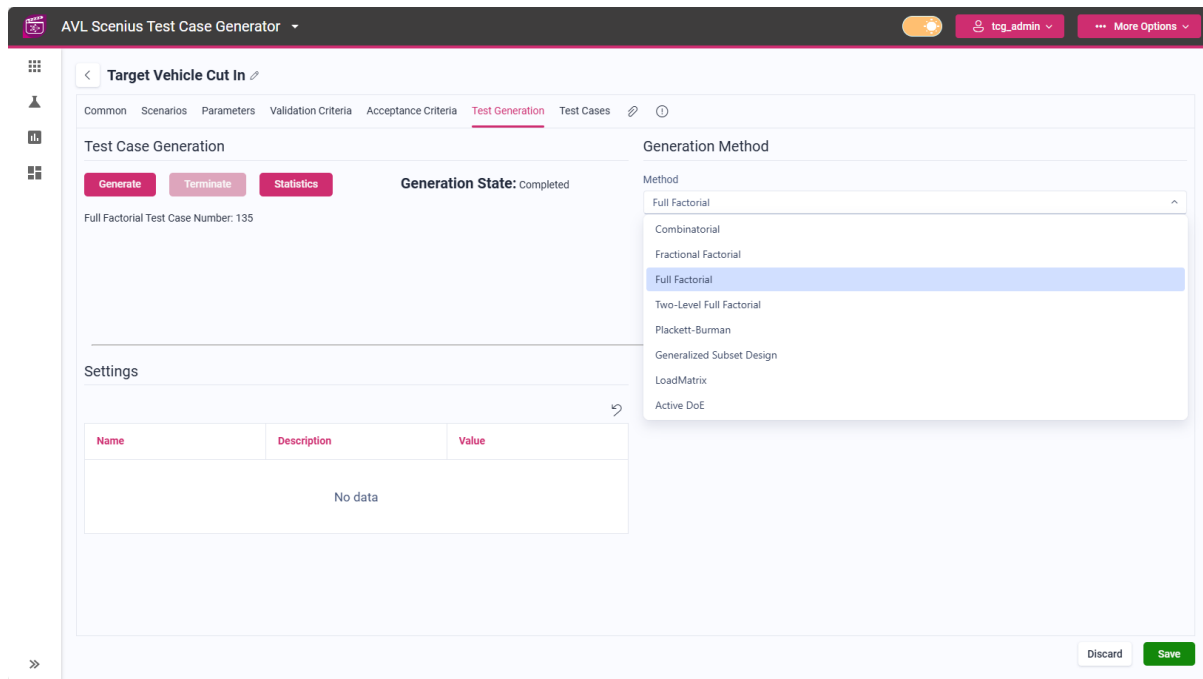


Figure 137: Scenius Test Case Generation (Concretization).







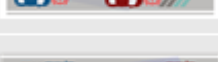

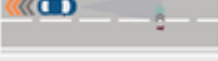



3.5.3.4 Allocate

After the test scenarios were queried and concretised, test cases are systematically allocated to the most appropriate testing environment based on defined parameters of the Operational Design Domain (ODD), with particular focus on vehicle speed and acceleration. This allocation methodology follows the method outlined in Sunrise D3.3.

According to this methodology, all test cases targeting the evaluation of decision logic are initially assigned to the simulation environment. However, scenarios with a strong emphasis on system validation—identified through expert domain knowledge—are assigned to controlled proving grounds or public road testing, depending on the required fidelity and realism.

A structured overview of the use cases is presented in the accompanying Table 16, which includes schematic representations and textual descriptions. The final three columns of the table indicate the allocated validation environment—proving ground, public roads or simulation.

Table 16: Overview of use cases and allocated test environments.

Use Case	Figure	Description	Proving Ground	Public Road	Simulation
A		TJC speed limit adaptation to new speed limit and distance to vehicle ahead		X	
B		TJC speed limit adaptation to vehicle ahead		X	X
C		A target vehicle from an adjacent lane cuts-in to ego vehicle's lane	X	X	X
D		A lead vehicle in ego vehicle's lane of travel cuts out to another lane	X	X	X
E		A stationary road user appears or there's a blocked lane of travel.	X		X
F		Ego vehicle is in standstill behind a stationary target vehicle, target vehicle starts moving			X
G		Ego vehicle is traveling on its own lane, a lead vehicle in ego vehicle's lane of travel inside the feature detection range decelerates, emergency manoeuvre is not necessary		X	X
H		Ego vehicle is traveling on its own lane, an unobstructed pedestrian crossing the carriageway appears in the front, emergency manoeuvre is not necessary	X		X
I		Ego vehicle is traveling on its own lane, a lead vehicle in ego vehicle's lane of travel inside the feature detection range decelerates resulting in an imminent collision risk, emergency manoeuvre is deemed necessary			X
J		Ego vehicle has come to a standstill after an emergency manoeuvre, the imminent collision risk has disappeared			X
K		Ego vehicle is traveling on its own lane, an unobstructed pedestrian crossing the carriageway or a stationary obstacle appears in the front, resulting in an imminent collision risk, emergency manoeuvre is deemed necessary	X		X
L		Lane keeping while entering a curve			X

3.5.3.5 Execute

Following the allocation of test cases to appropriate environments, the execution phase serves to evaluate the system under test (SUT) in both virtual and physical conditions. This phase is critical to assess the system's behavior and performance across a range of operational scenarios.

Three primary environments are employed in this use case for test execution:

1. **Simulation:** Simulation provides a safe, repeatable, and cost-efficient means to examine edge cases and complex scenarios. Within this virtual environment, the system can be assessed for its ability to detect and respond to vehicles, lane markings, and other traffic elements, all under controlled and reproducible conditions.
2. **Proving Ground:** The proving ground offers a controlled test track environment that enables high-precision testing with real production vehicles. It eliminates the variability associated with public traffic, allowing for consistent collection of vehicle data. This environment is particularly valuable for validating system performance under repeatable conditions, thereby enhancing reliability and safety.
3. **Public Road Testing:** In this phase, the system is exposed to the full complexity and dynamic nature of real-world traffic. Public road testing serves as the final and most comprehensive validation step, ensuring that the system performs safely and robustly in everyday scenarios while interacting with other road users and infrastructure.

Collectively, these three environments form a comprehensive validation strategy that spans the entire development pipeline—from early simulation-based evaluation to real-world deployment readiness.

An exemplary use case involving a cut-in scenario, tested across all three environments, is shown in Figure 138.



Figure 138: Exemplary Cut-In scenario in three different test environments.

Virtual Testing

The simulation focuses on distance-keeping and lane-centering functions.

At the core of the setup is a co-simulation platform, AVL Model.Connect, which manages communication and synchronization between multiple simulation tools. This platform enables a unified test environment where each component of the ADAS stack can be evaluated collectively.

The simulation toolchain includes four main modules:

- **CARLA:** A high-fidelity simulator used for 3D visualization and environment interaction. CARLA generates realistic urban and highway scenarios, simulating sensor inputs and dynamic actors necessary for perception and visualization of ADAS behavior.
- **Esmini:** Used for scenario playback and visualization. It utilizes OpenSCENARIO files to define and reproduce complex driving situations, allowing for consistent testing and evaluation of system performance across various use cases.
- **AVL VSM:** Responsible for vehicle dynamics modeling. VSM provides accurate simulation of vehicle behavior, ensuring that control algorithms are validated against realistic vehicle responses to environmental and system changes.
- **ROS (Robot Operating System):** Functions as the platform for development and execution of control algorithms, such as lane-centering and adaptive cruise control. ROS also facilitates integration by managing communication between the co-simulation platform and the control modules.

By synchronizing these components, the simulation environment is modular, scalable, and realistic. This setup supports iterative development, regression testing, and performance evaluation of ADAS controllers prior to deployment in real vehicles.

Proving Ground Testing

A proving ground is an instrumented and controlled environment designed to enable high-fidelity testing of complex traffic situations. The core of the experimental setup for proving ground testing is the Vehicle Under Test (VUT), which is equipped with Differential GNSS (D-GNSS) and Inertial Measurement Unit (IMU) sensors as shown in Figure 139. The VUT is connected to a WiFi mesh network and interacts with a base vehicle, which is responsible for coordinating the test execution and managing the platform system to propel the movable targets. The propulsion platform system—also networked via WiFi and equipped with high-precision positioning—controls movable elements such as pedestrian and vehicle dummies. This setup facilitates the safe and repeatable execution of critical test scenarios, including pedestrian crossings, cut-in and cut-out maneuvers, and lane blockages.

Due to the high degree of precision and the absence of uncontrolled external influences, the proving ground allows for systematic evaluation of system behavior under near-realistic conditions. This controlled testing environment supports the thorough validation of perception algorithms and decision-making logic in automated driving systems.

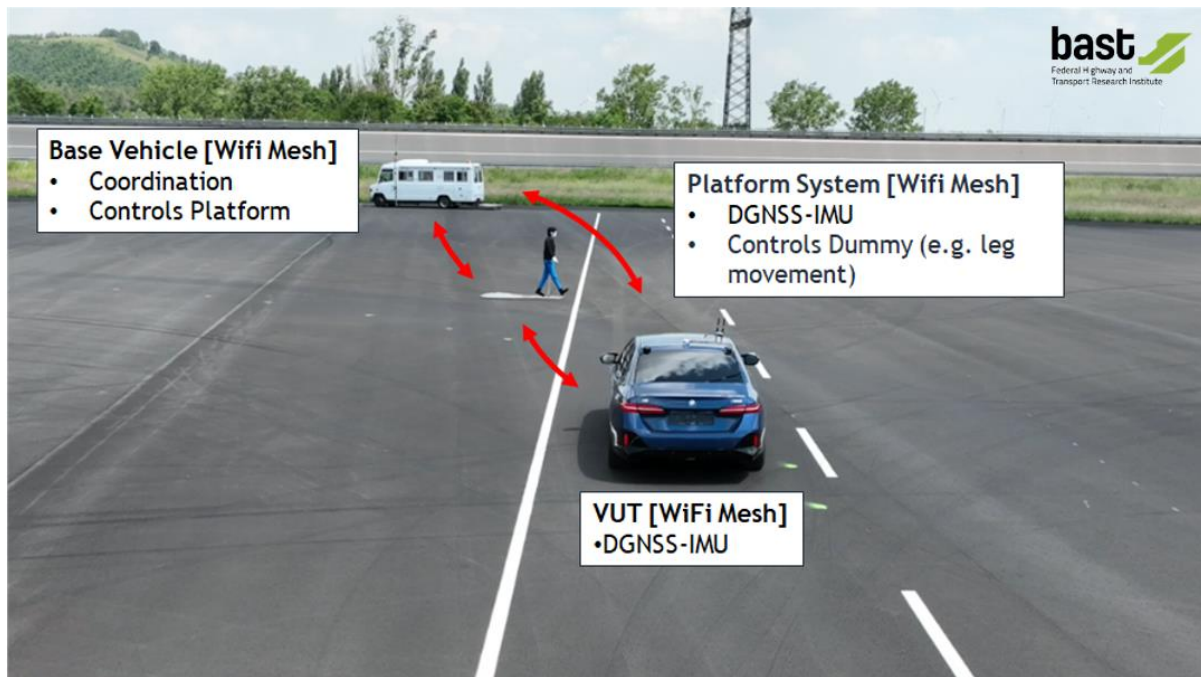


Figure 139: Test case setup for proving ground testing.

Public Road Testing

In this phase of the SAF block validation, the emphasis lies on real-world implementation, conducted under live traffic conditions on public roads. The Vehicle Under Test (VUT, also called ego-vehicle) is exposed to actual traffic scenarios, interacting with other vehicles, pedestrians, and existing infrastructure.

BAST's experimental setup for public road testing was designed for black-box testing (SUNRISE D4.6) of an SAE Level 3 production vehicle operating on German highways illustrated in Figure 140. It incorporates comprehensive data acquisition systems, including data loggers, GPS and WiFi antennas, and Inertial Measurement Units (IMUs), to monitor and record the vehicle behavior. A secondary vehicle, equipped with a similar instrumentation suite, was deployed to ensure and conduct realistic traffic interactions and scenario conditions.

The system architecture supports synchronised communication between the ego- and the target-vehicle using Differential GNSS, Ethernet, and WiFi. This configuration ensures consistent and reliable data exchange throughout each test drive.

Validating the system in real traffic environments allows for the assessment of its actual performance under dynamic and often unpredictable driving conditions. This approach is critical to verify that the system operates both safely and reliably under real-world circumstances.

To monitor potential human interaction during the execution of the use cases, an eye-tracking system and finger-mounted buttons were employed to capture the driver's attention and intention to intervene. Combined with vehicle data and video recordings, this setup enables a comprehensive reconstruction and analysis of any hypothetical human intervention, making it fully traceable and comprehensible.

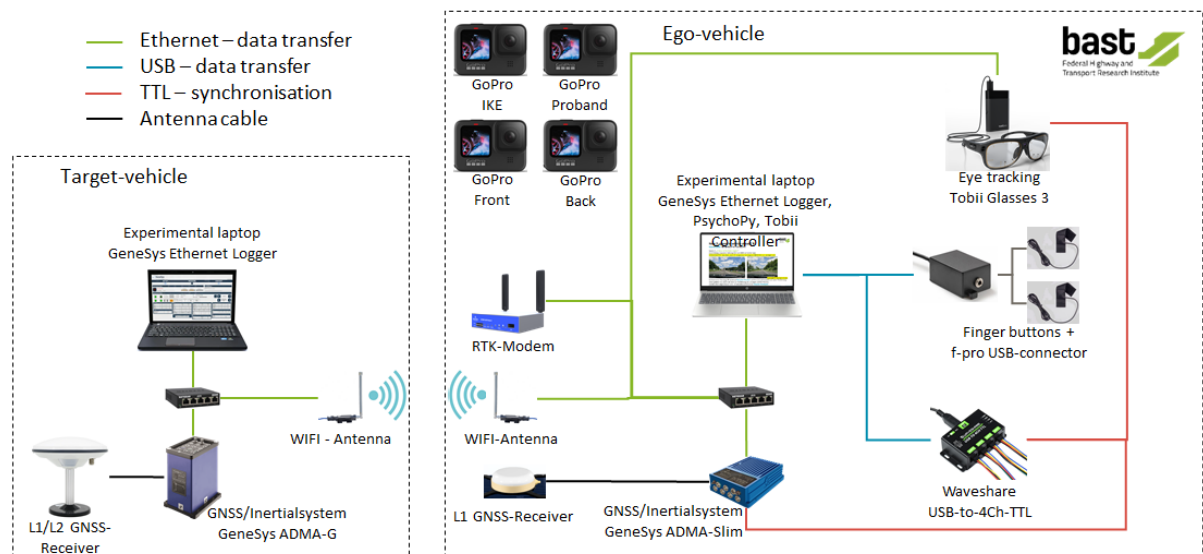


Figure 140: Test case setup public road testing.

The snapshot shown in Figure 141 was captured from a video which represents a four-way split-screen display, illustrating a cut-in scenario during one test drive on public roads. In this example, an SAE Level 3 automated driving system was active with full responsibility for the dynamic driving task (DDT). In the scenario, another vehicle overtakes the test vehicle and merges into its lane ahead of it. The ego vehicle's automated system reacts by decelerating and adapting its speed in response to the changing traffic conditions.

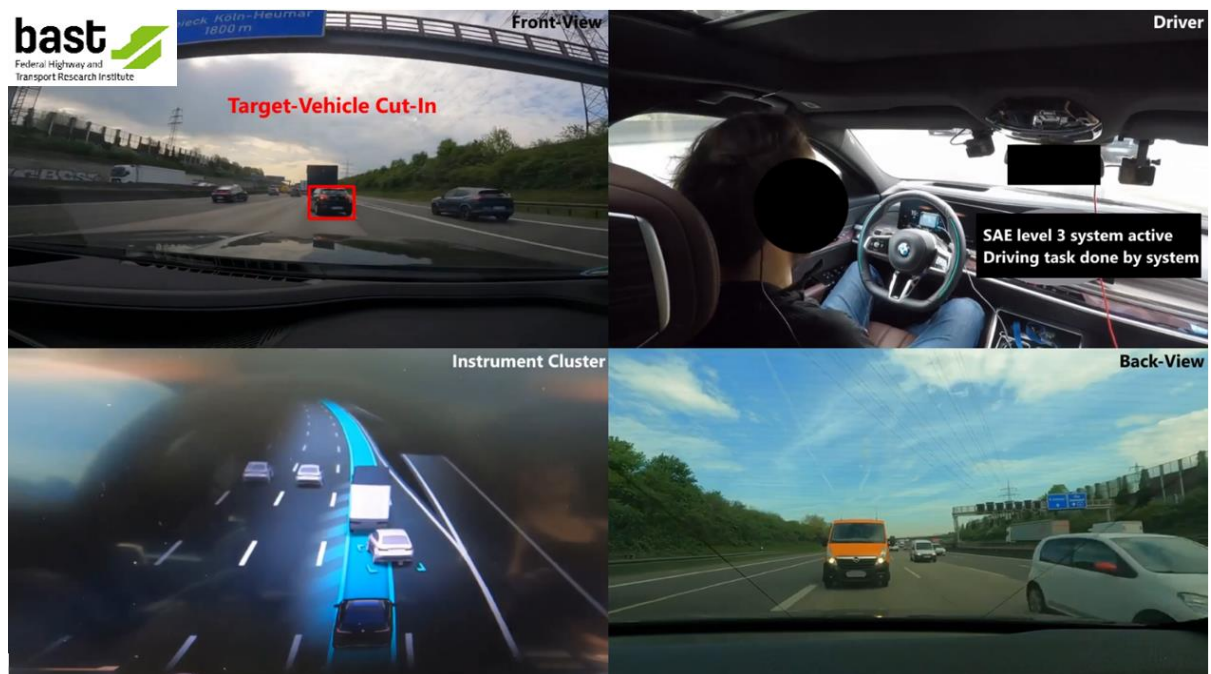


Figure 141: Four-way split screen from public road testing in a cut-in scenario.

3.5.3.6 Test Evaluate

Metrics

This section outlines the validation procedure for the SAF (Safety Assurance Framework), with a focus on the 'Test Evaluate' block. The evaluation centers around a set of key performance indicators (KPIs), which are used to assess the behavior of an Automated Lane Keeping System (ALKS) under test conditions. So, this is a Metric-Based Assessment as defined under validation criteria in chapter 2.2.6.

The primary metrics are used for evaluation for use case 2.1 include:

- Longitudinal acceleration of the ego vehicle
- Relative distances between the ego vehicle and surrounding traffic participants
- Time-to-Collision (TTC) values
- Impact assessment, i.e., whether a collision occurred during the test

These KPIs are derived from the requirements specified in UN Regulation No. 157 [6], which defines limits for acceptable performance. This includes thresholds for acceleration, jerk, minimum TTC, and system reaction time, all of which are critical to evaluating the safety performance of ALKS in Table 17.

Table 17: Overview of safety performance metrics for ALKS system.

KPI Name	Unit	Value (Limit)	Regulation/Standard
TTC_min	s	$TTC_{LaneIntrusion} > v_{rel} / (2 \times 6 \text{ m/s}^2) + 0.35 \text{ s}$	UNR - 157
Headway_dist_min	m	s. table (values in between interpolated)	UNR - 157
Ego_accel_x_cofm	m/s ²	-4	UNR - 157
Ego_accel_x_max	m/s ²	-6	UNR - 157
Ego_accel_y	m/s ²	1	UNR - 157
Ego_jerk_x	m/s ³	12.65	UNR - 157
reaction time of ALKS	s	0.75	UNR - 157

Collision_occurrence	bool	1/0	-
Take over request	bool	1/0	-

Table 18: From UN-R 157 [6] showing minimum headway distances for ALKS.

Present speed of the ALKS vehicle	Minimum time gap		Minimum following distance	Minimum time gap	Minimum following distance
	M_1/N_1		M_1/N_1	$M_2/M_3 // N_2/N_3$	$M_2/M_3 // N_2/N_3$
(km/h)	(m/s)	(s)	(m)	(s)	(m)
7.2	2.0	1.0	2.0	1.2	2.4
10	2.78	1.1	3.1	1.4	3.9
20	5.56	1.2	6.7	1.6	8.9
30	8.33	1.3	10.8	1.8	15.0
40	11.11	1.4	15.6	2.0	22.2
50	13.89	1.5	20.8	2.2	30.6
60	16.67	1.6	26.7	2.4	40.0

Simulation

In proposed simulation environment above, cut-in and cut-out scenarios were executed. Total test case numbers are xx and 162 for cut-in and cut-out respectively. In Figure 142, KPIs such as minimum TTC, collision occurrence (crash), maximum jerk, maximum comfort longitudinal deceleration and headway distance violation are all in between limits which are defined in UN-R 157 [6].

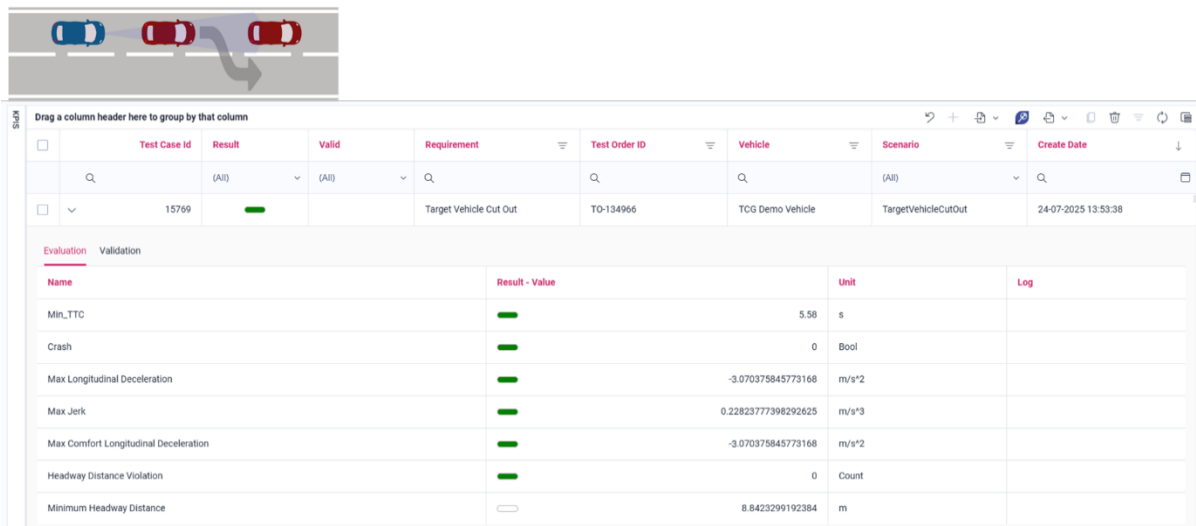


Figure 142: KPI results of cut-out scenarios.

In Figure 143 one of the executed test scenario results are illustrated. The target vehicle has always been considered as the vehicle in front of the ego vehicle. Ego and vehicle in front called as target accelerates from 0 to 60 km/h while actor vehicle, which is in front of target vehicle driving with 50 km/h. As can be seen, the cut-out manoeuvre took place at the 12-second of the simulation, and the ego vehicle suddenly detected the vehicle traveling at 50 km/h in front of the vehicle it was following at the same speed and adapted its speed to the new vehicle resulting with maximum deceleration of lower than comfort value of 4 m/s² deceleration rate and avoid collision.

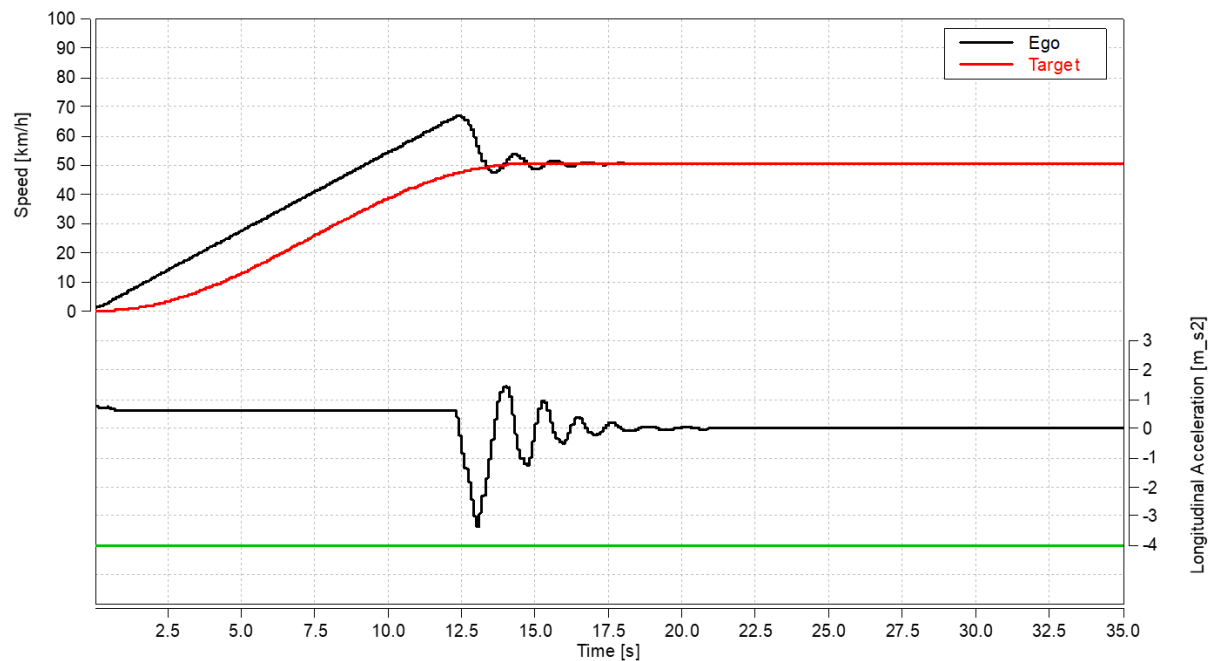


Figure 143: Cut-out simulation result at 50 km/h.

Proving Ground

The data plot in Figure 145, illustrates one exemplary cut-in scenario conducted on a proving ground, where the ego vehicle travels at 60 km/h and the target vehicle at 30 km/h. The recorded data visualises the evolution of speed, longitudinal acceleration, and relative lateral distance over time. The system successfully avoided a collision by adjusting its velocity to match the conditions of the cut-in event, demonstrating compliant behaviour with respect to the safety thresholds.

It can be seen that the deceleration does not overshoot the comfort value of -4 m/s^2 and the impact was avoided so we can mark this exemplary test as a pass.

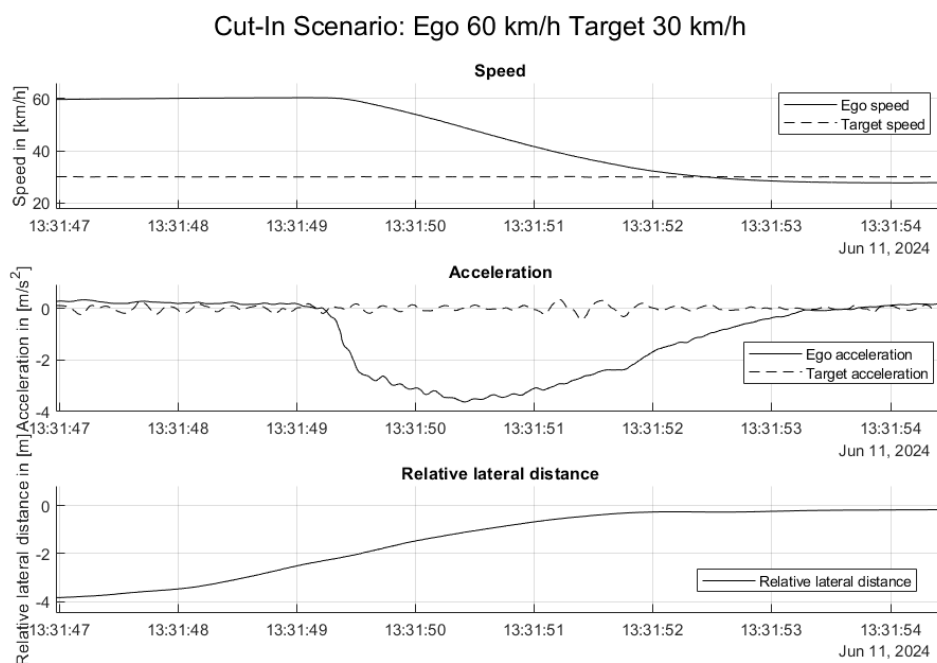


Figure 144: Data plot from proving ground testing showcasing a cut-in scenario with ego-vehicle speed of 60 km/h and target-vehicle speed of 30 km/h.

Public Road

In Figure 146, an exemplary data plot can be observed from the test drive on public roads with a production vehicle with an ALKS SAE level 3 system. The graphs illustrate the dynamics of a vehicle's driving behaviour over a short period. In the top graph, the ego vehicle's speed closely follows the target speed for most of the time, though a gradual decrease in speed occurs, followed by a sharp acceleration toward the end. The middle graph shows the ego and target accelerations, with the ego vehicle displaying more frequent and stronger fluctuations, indicating active adjustments to maintain distance and speed. The bottom graph presents the relative longitudinal distance between the ego and target vehicles, which fluctuates over time, showing a general increase, suggesting the ego vehicle was falling behind the target, likely due to reduced speed.

Throughout the observation period, the parameters defined by UN Regulation No. 157 are consistently met. There are no signs of atypical deceleration or abrupt braking behaviour, and the headway distance remains within acceptable limits, ensuring safe and smooth adaptive cruise control performance as part of the ALKS functionality.

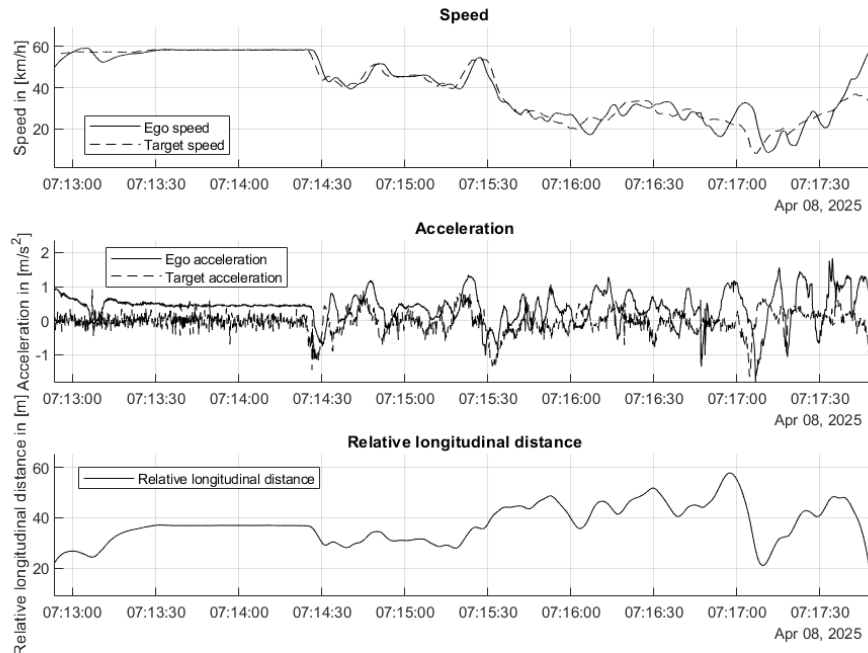


Figure 145: Data plot from public road testing of an ALKS system with a leading target vehicle.

3.5.3.7 Safety Case

In this stage, all previously generated outputs are consolidated to support the final decision within the Safety Argument Framework. This assessment draws on results from the Test Evaluation block; the Coverage block was not demonstrated in this particular use case. A thorough review of the available data was undertaken. Although evaluated coverage was not included in the present scenario, evidence from the test results was considered sufficient to confirm that the system satisfied the defined assessment criteria.

The Safety Case is the final, integrated body of evidence and reasoned safety argument underpinning the SUNRISE SAF process, and its validation is vital for regulatory acceptance and stakeholder confidence. It presents structured, modular arguments linking requirements, scenarios, test results and safety criteria. All claims are fully traceable to specific evidence, ensuring transparency for regulators and stakeholders. The documented evidence demonstrates sufficiency and rigour across all relevant ODDs, potential failure conditions and hazards, providing clear justification as to why the system is deemed safe enough.

All Safety Case elements are maintained under version control to ensure auditability and to enable updates as technology, regulations or operational experience evolve. Its structure is designed for stakeholder acceptance, ensuring it is comprehensible to technical reviewers, regulatory authorities and, where appropriate, public stakeholders. Continuous updating mechanisms allow the integration of new evidence from in-service monitoring, incident analysis and system evolution, thereby maintaining Safety Case validity throughout the CCAM system's lifecycle.

For formal auditing purposes, such as type-approval assessments, the SUNRISE Interactive Handbook provides applicable guidelines to validate the SAF process in a structured and transparent manner.

3.5.3.8 Decide

During testing with a real production vehicle equipped with an ALKS system, no irregularities were observed – which was not unexpected, given that the vehicle was a genuine series-production model. The tests primarily served to demonstrate that the SAF could be effectively applied to such a system. In line with the validation principles, the assessment confirmed compliance with UNECE, EC, and other relevant regulations and standards (UN-R 157 [6]). All inputs, analyses, and justifications for decisions were documented and made available for potential audits or regulatory inspection. Furthermore, the process incorporated relevant external influences, regulatory requirements, and stakeholder feedback before final conclusions were drawn. This integrated validation approach, covering all functional blocks and interfaces, was essential in demonstrating the SAF's capability, reliability, and regulatory compliance within the complete CCAM safety assurance process.

3.5.4 Key take aways

This use case evaluation demonstrated that the SUNRISE SAF provides substantial and practical guidance for the safety validation of Automated Driving Systems (ADS).

Validation activities were conducted across a diverse set of environments, including simulation platforms, proving grounds, and public roads. These setups enabled consistent and robust testing under repeatable conditions, reinforcing the framework's applicability across various testing modalities.

By performing black-box testing of an SAE Level 3 system integrated into a state-of-the-art vehicle, the framework's scalability and versatility were effectively illustrated from the standpoint of both regulatory authorities and consumer testing organisations.

3.5.5 Deviations from D7.2

No Deviations from D7.2.

3.6 Use case 3.1: Highway AD validation – Map based perception & decision making

3.6.1 Use Case Overview

Within this UC 3.1 the SUNRISE SAF is validated by the safety assurance of a highway pilot (AD function). The perception of the AD function is extended with map-based information about upcoming curvature or speed limit information. Two scenarios (see section 3.5.2) were implemented to assess the safety of the system. The highway pilot controls the lateral motion of the vehicle (lane keeping) as well as the longitudinal control of the vehicle (ACC – keeping or reaching a safe speed).

Objective:

Use the SAF to show how the safety and efficiency of map-based highway Automated Driving Systems can be demonstrated successfully.

3.6.1.1 Covered aspects of the SAF

The table below shows the contributions to the SAF block validation per partner for UC3.1 and summarized in Figure 146.

Table 19: Contributions to the SAF validation per Partner in UC3.1

Partner/ Block	IDI	ika	UNITN	UoW
SUNRISE DF				x
Query & Concretise		x	x	
Allocate	X	x	x	
Execute	X	x	x	
Test Evaluate	X		x	
Coverage	X	x*	x	
Safety Case				
Decide	X		x	

**only parameter space coverage*

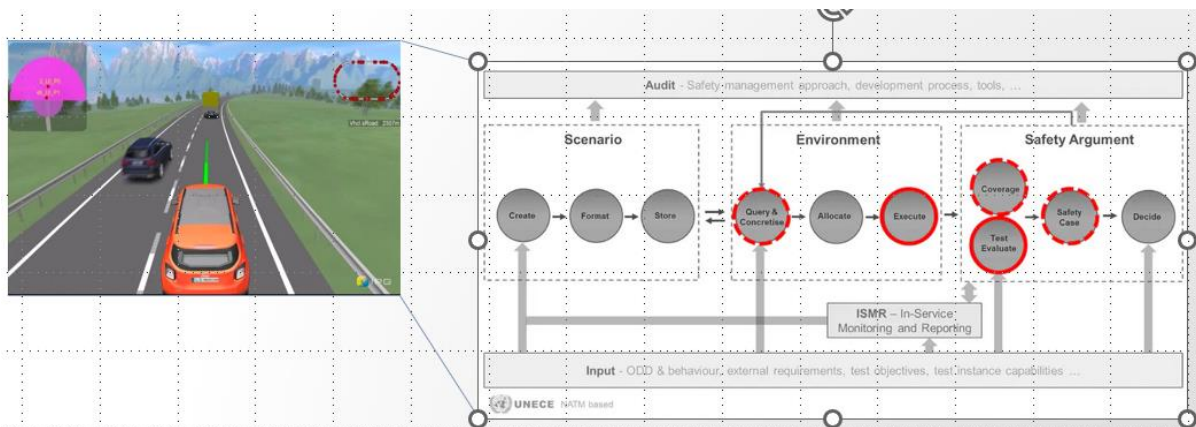


Figure 146: SAF demonstrated blocks in this UC's demo (s).

Handbook videos produced:

UC3.1-A https://www.youtube.com/watch?v=0R1LRYRj_6I&t=8s

UC3.1 (UNITN) <https://www.youtube.com/watch?v=kjk1kSLu6fw>

3.6.1.2 Safety case setup

The objective is to demonstrate how the SUNRISE SAF is applied for safety assurance of a map-based perception and decision-making CCAM system. The goal is to verify that the CCAM plans safe paths based on HD map data information. The information considered can originate from HD maps, sensors, and connectivity; however, the primary focus is on obtaining timely HD map data. The following steps are performed:

- Safety Goals and Key Performance Indicators: For lateral control, the vehicle must stay within its lane. For longitudinal control, it should use deceleration rates typical of human drivers to avoid rear-end collision risks, with acceptable¹ undershoot and overshoot. In curves, the speed must be similar to human speed choice for comfort considerations (D3.4 Section 6), but also because a vehicle moving differently than humans might cause misinterpretations by the human drivers.
- Tests performed in different test environments (simulations with IPG CarMaker at UNITN, simulations and Proving Ground tests (Figure 147).
- In simulations, the scenario space is covered by using a random sampling. For UC 3.1 A (adaptation to speed limits), 500 concrete scenarios were simulated. For UC 3.1 B (speed adaptation in curves), 622 concrete scenarios were studied with two consecutive curves ahead.

¹ A rational approach to define acceptability thresholds for undershoot and overshoot would call for string stability analysis (no worse than human and possibly better). For the purpose of demonstrating the application of the SAF, a conventional threshold of 5 km/h is assumed, without claiming it is the optimal threshold.

- Simulation outcomes have been confirmed by physical testing on selected concrete scenarios.

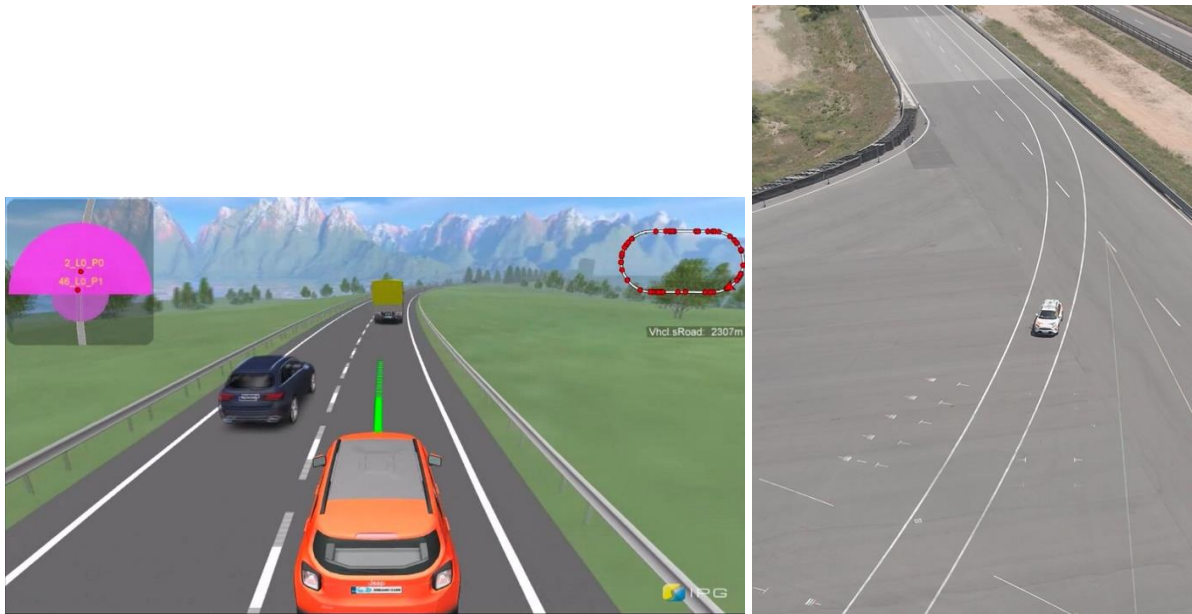


Figure 147: Test case setup (use case B). On the left is the simulation environment, featuring a sensor view in the top left and a map view in the top right. On the right is the proving ground setup.

3.6.2 Overview of tested scenarios

The Use Case comprises two subcases (Figure 148): A) testing adaptation to new speed limits through safe deceleration with minimal overshoot or undershoot, and B) testing speed adaptation, speed selection, and lane keeping.

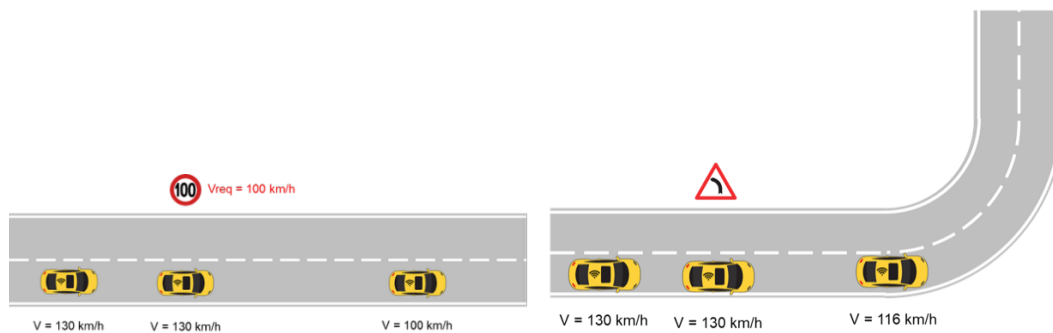


Figure 148: Use Case 3.1 includes two subcases: A) testing speed adaptation (left) and B) testing speed choice and lane maintenance (right).

3.6.3 SAF Block demonstrations

3.6.3.1 Query & Concretize

The ODD is defined in D7.1, Sections 6 and 6.2 (i.e., Highway, ACC enhanced with map-based perception)

The scenarios within this use case were used as a demonstration of the SAFs' querying process. They were uploaded to the public database in the form of Safety PoolTM. A query was crafted that utilised ODD tags and behaviours that were extracted from the original functional scenario definitions in Sunrise D7.1. This query resulted in the retrieval of the 2 relevant logical scenarios for the use case, a demonstration of the library and query views can be seen in Figure 133. The two logical scenarios were slightly revised by experts.

Once the logical scenarios were pulled from the various scenario databases, they were concretized using random sampling.

For UC 3.1-A, the logical scenario has three parameters as follows in Table 20. This was sampled with 500 random concrete scenarios.

Table 20: UC 3.1-A logical scenario parameters

Ego initial speed range	90-130 km/h
Speed limit displayed on sign	70-130 km/h
Starting distance to speed sign	100-300 m

For UC 3.1-B, the logical scenario includes two curves in opposite directions, with the inversion of curvature challenging the system, and it has 5 parameters listed in Table 21. The two curves are preceded by a straight road. The simulations begin with the Ego vehicle placed 15 s before the first curve (i.e. the distance travelled in 15 seconds at the initial speed). This scenario was sampled with 622 random concrete points.

Table 21: UC 3.1-B logical scenario parameters

Ego initial speed range	50-130 km/h
First curve direction change (degree)	0-180 °
First curve curvature (1/m)	0-0.001 m ⁻¹
Second curve direction change (degree)	0-180 °
Second curve curvature (1/m)	0-0.001 m ⁻¹

3.6.3.2 Allocate

The allocation process followed the ODD-based structure from D3.3 for initial allocation. Mainly the capabilities (test throughput and fidelity) as part of the test criteria were used to allocate some of the test cases (chosen by experts) to proving ground test instance, but they could also be assessed by virtual simulations. All other test cases were allocated to virtual

simulations due to the soft test case requirements on the fidelity. The main reason why all test cases could be performed on virtual simulation test instances is that the focus is not on ensuring the safety of a real ADS, but on validating the SAF. Figure 149 shows a simplified example from UC 3.2 (the original ODD structure covering all parameters is too large) for such an allocation process making use of the ODD structure for allocation presented in D3.3.

Scenery Elements			Environment conditions		Dynamic Elements			Test Criteria		Test instance feasible to use	
Drivable area			Connectivity		Subject vehicle / Traffic agents			Capability criteria		Simulation	Proving Ground
Type	Geometry	Lane specification	Communication	Positioning	Motor vehicles			Test throughput capability	Test environment fidelity capability		
					Human representation		Vehicle Dynamics				
					Driver or Trajectory follower	Real human representation	General				
Highway	straight / low curvature	Lane markings	V2V	OpenDRIVE or GNSS	Yes	No	speed 60-130	High	Low	Yes	No
Highway	straight	Lane markings	V2V	OpenDRIVE or GNSS	No	Yes	speed 80-100-120	Low	High	No	Yes

Figure 149: ODD-based allocation of two clusters of test cases to proving ground test instance (indicated in blue) and virtual simulations test instance (indicated in green color).

There are no big differences between the upper cluster of test cases allocated to virtual simulations (argumentation marked in green) and the lower cluster of test cases allocated to proving ground test instance (argumentation in blue) except for the capability criteria (test throughput and fidelity) and the concluded fidelity of the human representation.

Although the upper cluster of test cases was allocated to virtual simulations it would be feasible to re-allocate some of the test cases to higher fidelity test instance such as proving ground. But not many of them otherwise the requirement for high test throughput capability would get a strong requirement and can be not fulfilled by proving ground testing. However, the re-allocation method from D3.5 was not in the scope of the use case and the method was not available before the use case was applied. But the experiences from the use case with the initial allocation process were reported to the re-allocation process and are covered by D3.5.

As a final remark in D3.5 slight modifications of the initial allocation process are included (e.g., the SUT requirements), but within the use case none of the modifications had to be applied (e.g. all for the allocation process relevant SUT requirements were covered by the test case requirements and all test instances were Simulink compliant where the models were implemented).

3.6.3.3 Execute

- **Virtual Testing:**

Approach 1 (UNITN). The concrete scenarios were executed in IPG CarMaker (D4.3, Sections 2.1.4, 2.2.4, 2.3.5, 2.4.2, 2.5.1, 2.6.1, 4.3.1, corresponding to Validation Criteria in Section 2.2.4a, c), recording the complete logs and extracting the longitudinal velocity, deceleration, speed in curves, and lateral distance from the lane

edges (Validation criteria in Section 2.2.4 b). The chosen KPIs are derived from expert knowledge about the UC (2.2.4 b and 2.2.6 a).

Approach 2 (IDI). The concrete scenarios were executed in MATHWORKS Automated Driving toolbox, recording the complete logs and extracting the longitudinal velocity, deceleration, speed in curves, and lateral distance from the lane centre (Validation criteria in Section 2.2.4 b). The chosen KPIs are derived from expert knowledge about the UC (2.2.4 b and 2.2.6 a).

- **Proving Ground Testing:**

Approach 1 (IDI). The objective of this task was to evaluate how High-Definition (HD) map data and external information could enhance the capabilities of a CCAM system in highway scenarios. The focus was on the Highway Pilot (HWP) function and its ability to anticipate upcoming events, such as speed limits or sharp curves, that are not yet visible to the vehicle's onboard sensors. The test cases were executed at IDIADA proving grounds, which were mapped and integrated in the functionality using the OpenDRIVE standard. During operation, the HWP extended its trajectory planning capabilities by integrating lane markings detected by the perception system with HD map data, enabling a more accurate and anticipatory driving behaviour. RTK test device was used to extract ground truth data for validating and reporting the enhanced response.

Proving Ground Testing

This section describes the development and implementation methodology of the Highway Pilot (HWP) system on the test vehicle. In the first place, an overview of the vehicle instrumentation and the overall system architecture, detailing the components integrated into the platform, including sensors, perception modules, and computing units.

The Highway Pilot system was deployed on the vehicle platform referred to as CAVKit. This platform consists of various systems fully integrated into a Toyota RAV4, including the perception, planning and control systems, which are responsible for perceiving the operational environment and controlling the CAVKit's trajectory.

To integrate these systems into the vehicle, an instrumentation rack was installed in the trunk. This rack is powered directly by the vehicle's electrical system and is capable of supplying power to all components without requiring any external power source.

All systems within the CAVKit are interconnected via Ethernet, CAN, or USB interfaces as shown in Figure 150.

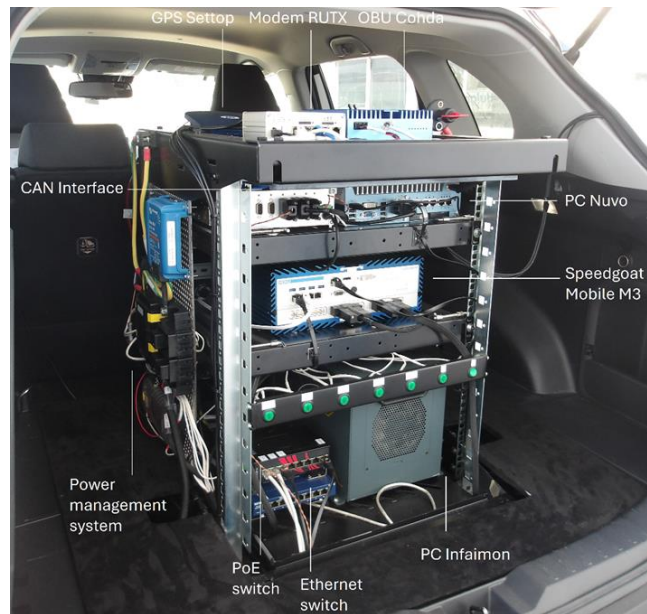


Figure 150: CAVKit rack used in the SUNRISE project.

In terms of onboard instrumentation, the ego vehicle was equipped with a Speedgoat real-time target machine (Mobile version), which acted as the central processing unit of the system. This unit was used to deploy the onboard decision-making and control algorithms.

For perception and data acquisition, a variety of sensors were integrated to enable accurate monitoring of the vehicle's surroundings and to support localization tasks. These sensors included cameras (3 cameras that output raw images and a MobilEye smart camera, which output the objects and lines detections), lidar, and a radar.

For positioning and motion estimation, the setup included an Inertial Measurement Unit (IMU), a Global Positioning System (GPS), and a Real Time Kinematic (RTK) module. The RTK module was used to extract ground truth data, which was fundamental for evaluating the performance of the functionalities during the analysis of the test results.

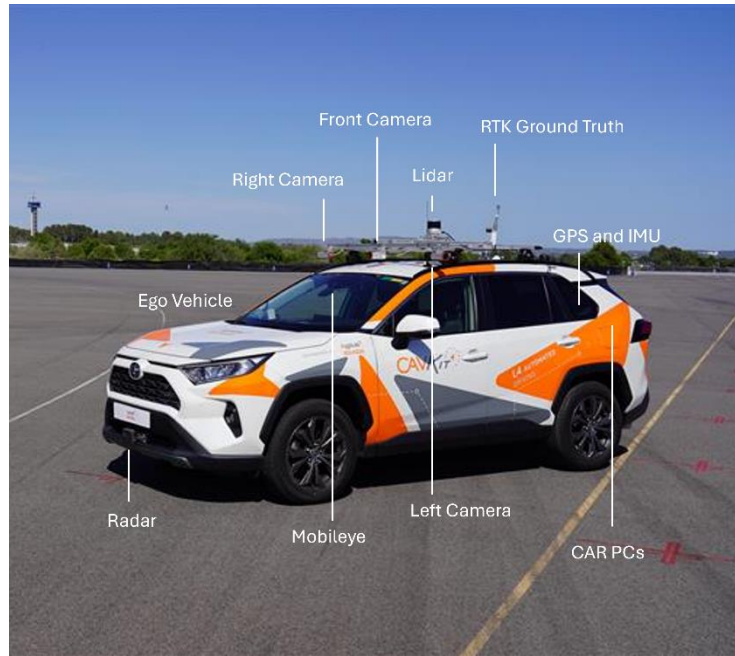


Figure 151: Use Case 3.1: Ego Vehicle used.

The original vehicle was a Toyota RAV-4 in Figure 151, so TME as partner of the project was contacted to provide technical support on how to access the correct CAN-bus terminals and pins, to be able to handle the longitudinal and lateral control of the vehicle. Based on the vehicle architecture, several signals had to be identified and overridden by the algorithm developed by IDIADA. This included also information related to failsafe logic and to the Electronic Power Steering (EPS) unit. Several exchange loops between TME and IDIADA took place, including experimental tests by IDIADA to define the limit values of the control parameters, such as acceleration or jerk limits.

For this use case, the interaction workflow involved the perception and control systems, as well as the interception of the vehicle's Controller Area Network (CAN) bus. This setup ensured that control responses were correctly transmitted to the vehicle actuators.

Figure 152 illustrates the main system interactions supporting the execution algorithms developed for UC 3.1.

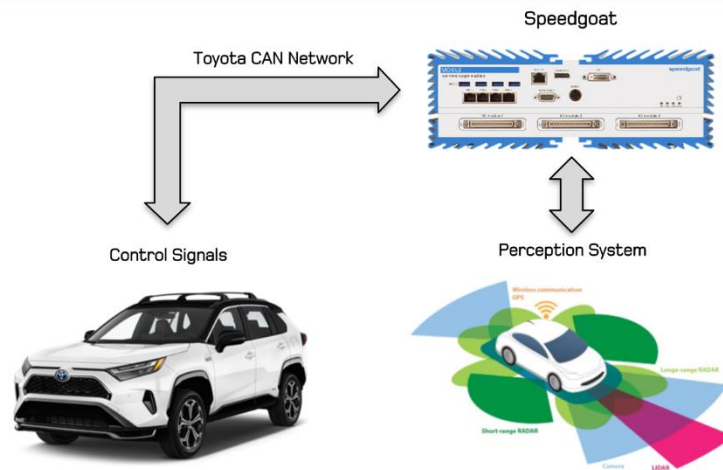


Figure 152: Communication workflow between related systems used in UC 3.1.

The experimental setup consisted of a fully equipped Automated Driving (AD) platform traversing a road segment. To isolate the impact of map data on system performance, the vehicle's perception was only used to detect lane markings, while all other relevant environmental information (specifically the coordinates of speed limit sign and sharp curve) was obtained from the map.

In **UC 3.1-A**, the objective was to enable early vehicle adaptation in response to a reduction in the speed limit indicated by a road sign. The vehicle was positioned using a high-precision GNSS module, which allowed accurate localization on the map via standard projection techniques. In particular, the Highway Planning module had prior knowledge of the upcoming speed limit change, allowing it to adjust the vehicle's trajectory smoothly, without requiring abrupt manoeuvres.

In **UC 3.1-B**, the aim was to further enrich the performance of the Automated Driving (AD) platform not only using explicit information of the road elements but also considering the road profile to adjust the parameters of the Highway Pilot algorithm. A comfort constraint was imposed on the lateral acceleration of the vehicle, which translates as a limit in the longitudinal speed in curvilinear segments. In this way, the road curvature was used to adapt the maximum speed of the planning algorithm to be compliant with the comfort constraints of the vehicle.

In both scenarios, the vehicle responded as expected, adapting its speed based on the map data. These scenarios illustrate the advantages of incorporating map-based information into the development of AD algorithms, as it enables the system to anticipate upcoming events and perform earlier and smoother adaptations. Particularly, in situations where complete environmental data cannot be reliably obtained from onboard sensors.

3.6.3.4 Test Evaluate

Approach 1 (UNITN)

The safety argument begins with evaluating the KPI listed in the Safety Goals (Section 3.6.1.2), i.e.:

The System must use deceleration values typical of human drivers to avoid rear-end collision risks (Validation Criteria 2.2.6 a-b).

The system should produce acceptable velocity undershoot and overshoot (Validation Criteria 2.2.6 a-b).

The vehicle must stay within its lane (Validation Criteria 2.2.6 a-b).

In curves, the system must use speed values compatible with the human curve speed choice and combined longitudinal/lateral accelerations typical of human driving (Validation Criteria 2.2.6 a-b).

For **UC 3.1-A**, and criterion 1, simulation results are summarized in the following chart, which assesses the decelerations produced for speed limit adaptation based on different distances (x-axis) and velocity variations (y-axis) required to meet the speed limit. Due to the scenario's low complexity, this image provides a thorough overview of the CCAM system's performance across the logical scenario (Validation Criteria: 2.2.6 c). Concrete scenarios are marked with dots.

The chart (Figure 153) gives the contour plot of the maximum deceleration KPI. The contour corresponding to -4 m/s^2 is a rounded value of the maximum deceleration used by humans in stop maneuvers [7].

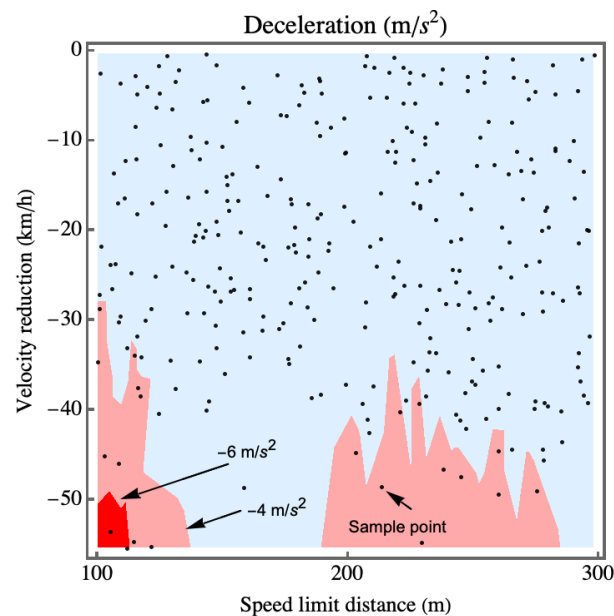


Figure 153: Use Case 3.1 A: Contour plots showing the deceleration required to comply with the new speed limit.

Concerning the velocity undershoot (criterion 2), which might penalize traffic flow (Sunrise D3.4 Section 6), the following chart (Figure 154) summarizes the findings. Significant undershoots occur after velocity reductions exceeding 30 km/h, and they are correlated with high decelerations. They can negatively impact traffic flow. When the speed reduction is minor, the overshoot remains under 5 km/h (shown in the blue region). In particular the undershoot restricted to the hypothesis of speed limit steps up to -20 km/h is -1.03 km/h (to assess

whether this value is acceptable, the effect on string stability should be studied, as mentioned earlier).

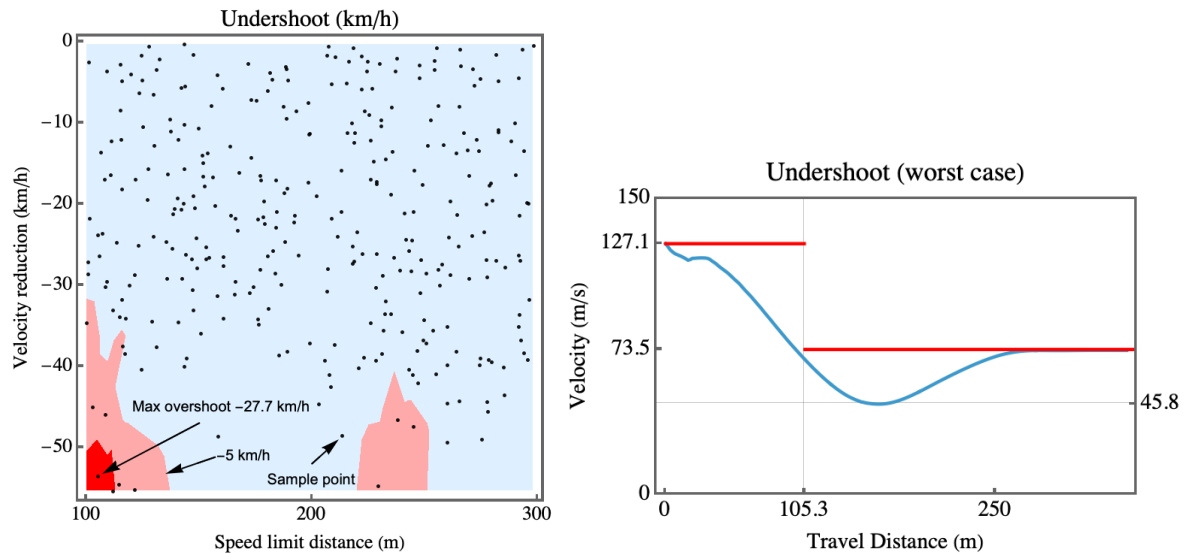


Figure 154: Use Case 3.1 A: Contour plots illustrating velocity undershoot. Hard bakes, on the right, can cause significant undershoot, negatively impacting traffic flow. When the speed reduction is minor, the overshoot remains under 5 km/h (shown in the blue region).

The overshoots were found to be related to the errors in speed tracking and are always small. The worst case was +0.3 km/h.

For **UC 3.1-B**, criterion 3, simulation results are summarized in the following charts (Figure 155 and Figure 156). The chart (Figure 155) shows where the minimum distance to the lane edge is reached, plotted against the curvature of the first (x-axis) and second curves (y-axis) (Validation criteria 2.2.5 b, 2.2.6 a, c).

The system remains in the lane (Validation criteria 2.2.6 c) with a minimum clearance of 0.39 m to the left and 0.46 m to the right. The results are slightly asymmetric because the sequence of curves is (straight, left then right). To decide whether these clearance values are acceptable the following approach should be used (Validation criteria 2.2.7 a-c, e):

Evaluate user acceptance. People might dislike small gaps, especially if there is a physical obstacle beyond the edge (may it be another vehicle, a guard rail, etc.). The accepted clearance is expected to vary according to speed, curvature and environment.

Evaluate the distribution of lateral control errors and set a threshold corresponding to very high confidence. Lateral deviations are expected to vary with the system low level control, the vehicle type and chassis dynamics, environmental conditions, (e.g., wind gusts), speed, slope and curvature.

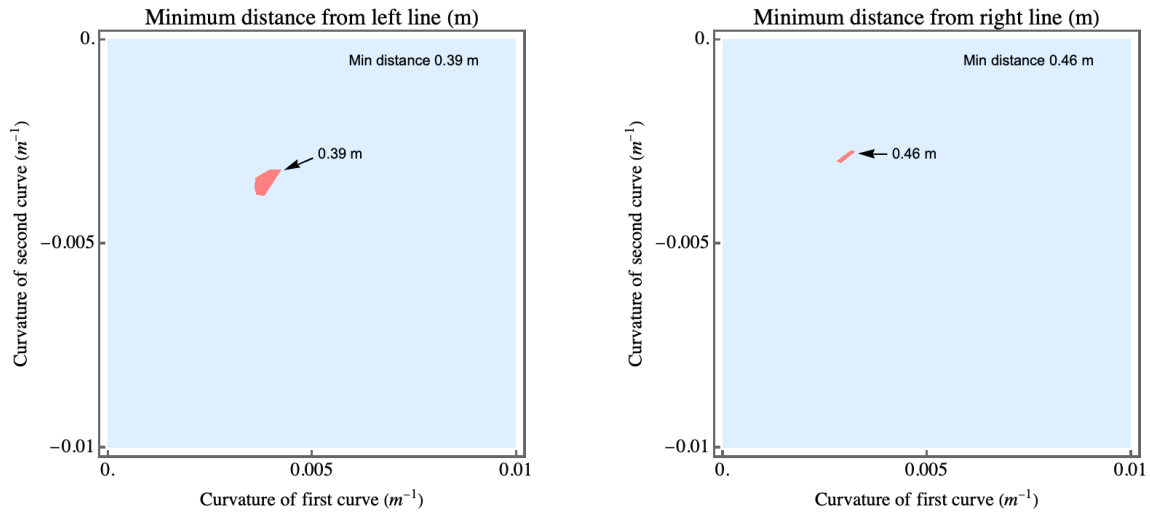


Figure 155: Use Case 3.1 B: The system reliably stays within the lane.

Regarding the selection of speed in curves (criterion 4), the following chart compares the lateral acceleration to the average human choice (Validation criteria 2.2.7 e), according to [8] (left), and shows the g-g diagram (Figure 156) that indicates accelerations within the human naturalistic range [8] (right).

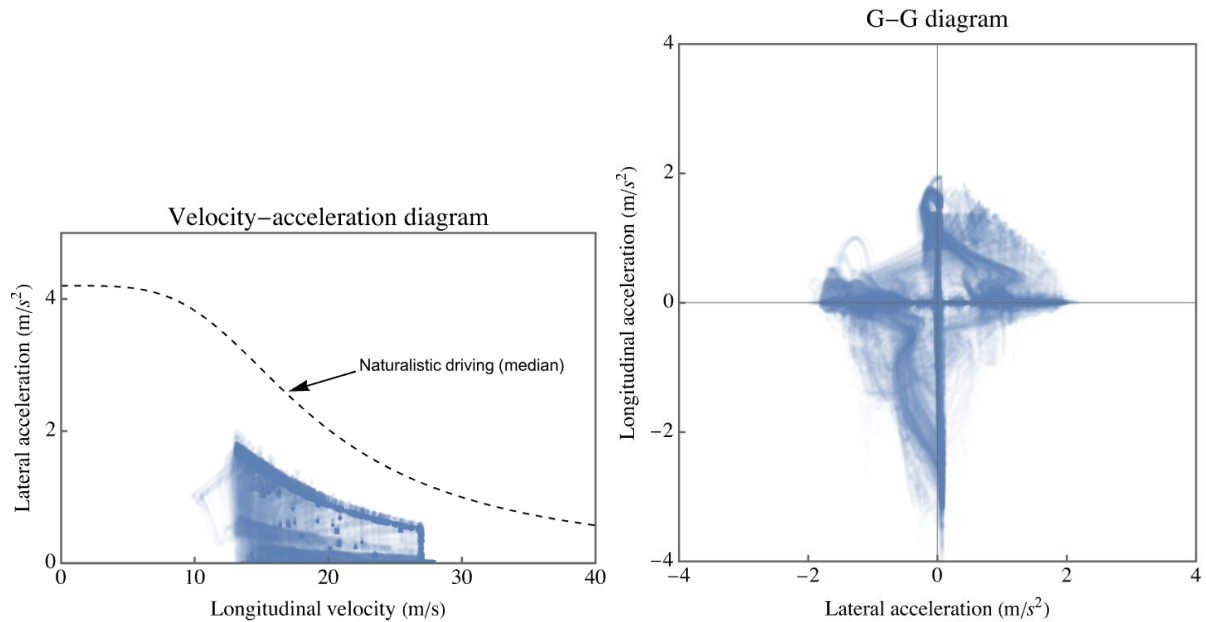


Figure 156: Use Case 3.1 B: The system uses lateral accelerations compatible with human choice in curves (left) and human combined accelerations (right) [8].

Approach 2 (IDI)

The safety argument begins with evaluating the KPIs listed in the Safety Goals (Section 3.6.1.2), i.e.:

The System must use safe deceleration values to avoid rear-end collision risks (Validation Criteria 2.2.6 a-b).

The system should produce acceptable velocity undershoot and overshoot (Validation Criteria 2.2.6 a-b).

The vehicle must stay within its lane (Validation Criteria 2.2.6 a-b).

In curves, the system must use speed values belonging to a maximal lateral acceleration of 3m/s^2 or below (Validation Criteria 2.2.6 a-b).

For **UC 3.1-A**, and criterion 1, simulation results are summarized in the following chart (Figure 157), which assesses the overshoot or undershoot at speed limit sign position based on different distances (x-axis) and speed variations (y-axis). As one can see in the upper area where the indicated speed limit is higher than the VUT speed the vehicle accelerates comfortably which results in undershoots as expected. In the lower part where the speed limit sign indicates a speed lower than ego the system shows an overshoot of up to 40 km/h in very extreme cases. This means the system keeps a comfortable safe braking for the following traffic and must not achieve the posted speed limit at the speed limit sign position if the data is available very late. However, in this use case the HWP perception is extended by map data information where the needed speed limit information should be available at least 300 m before the speed limit which would result in no overshoot.

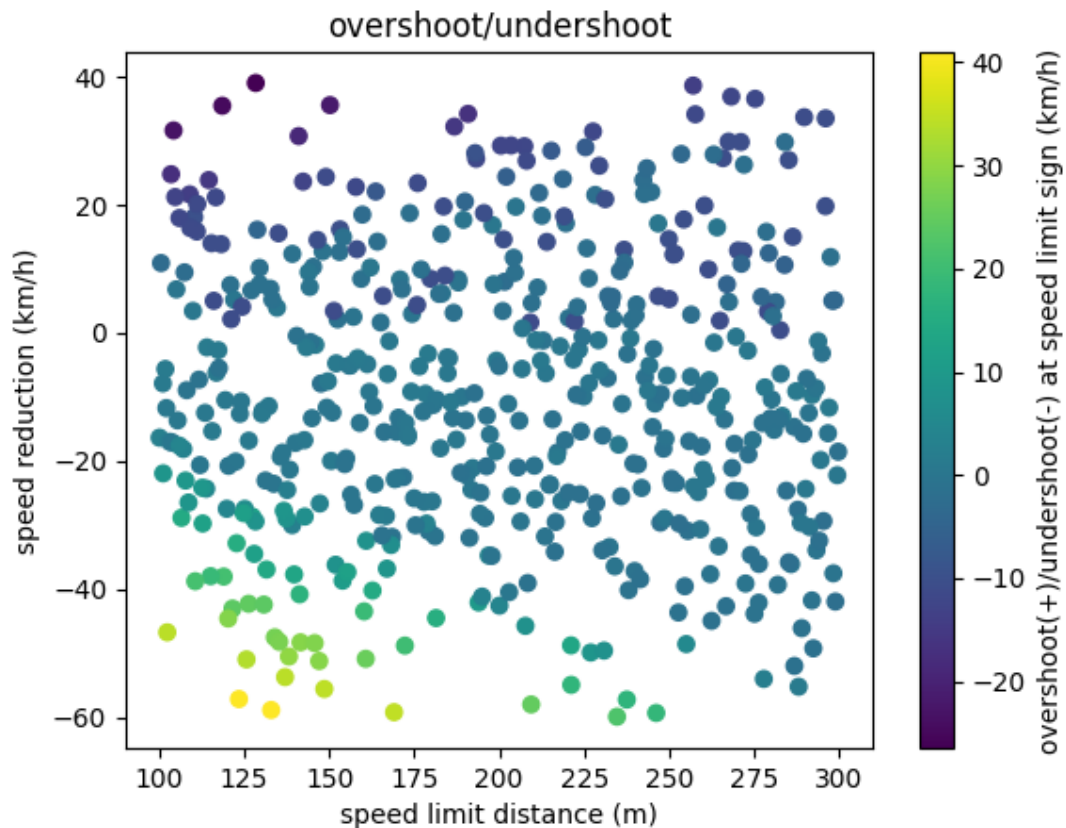


Figure 157: Use Case 3.1 A: Scatter plot showing the overshoot/undershoot at speed limit sign position.

The next scatter plot (Figure 158) shows that in case of speed reduction where the posted speed limit is lower than the VUT speed there is no relevant over- or undershoot. In the upper area where the posted speed limit is higher than the VUT speed the VUT has an overshoot of up to 5 km/h and an undershoot of up to -12 km/h. The undershoots could be either related to

inaccuracies of the SUT close to target speed or could be related to stability issues of the combination of used trajectory following and vehicle models or all together. Too short simulation time could also be the reason for these undershoots. The exact reasons would need further investigation into future work.

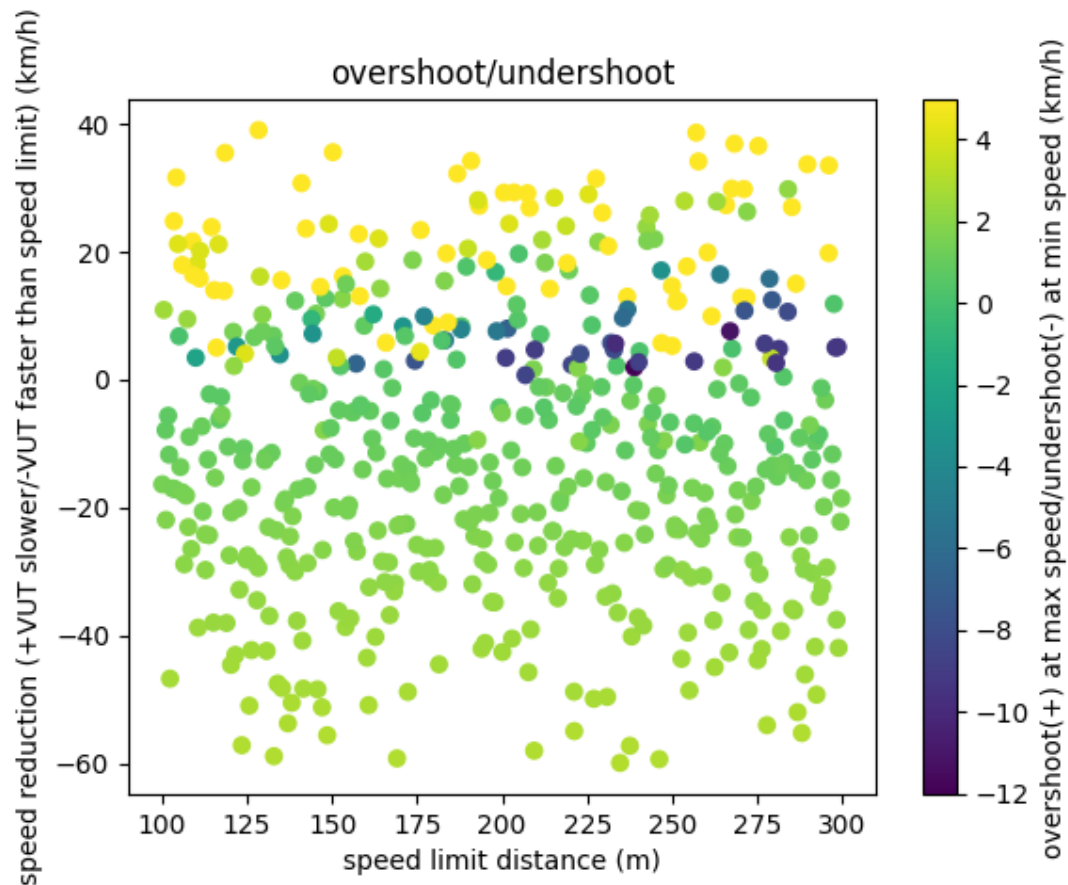


Figure 158: Use Case 3.1 A: Scatter plot showing the overshoot for maximal VUT speed and undershoot for minimal VUT speed.

For **UC 3.1-B**, simulation results are summarized in the following chart (Figure 159). The chart shows where the maximum distance to the lane centre line is reached, plotted against the curvature (x-axis) and largest VUT speed in curve (y-axis) (Validation criteria 2.2.5 b, 2.2.6 a, c).

The system remains in the lane (Validation criteria 2.2.6 c) with a maximum deviation from its lane centre line of 0.5 m. These deviations from lane centre are still safe especially for a prototype system as it is, because the vehicle does not leave its lane but might be not very comfortable for the driver of the VUT but also the surrounding traffic on the neighbour lanes.

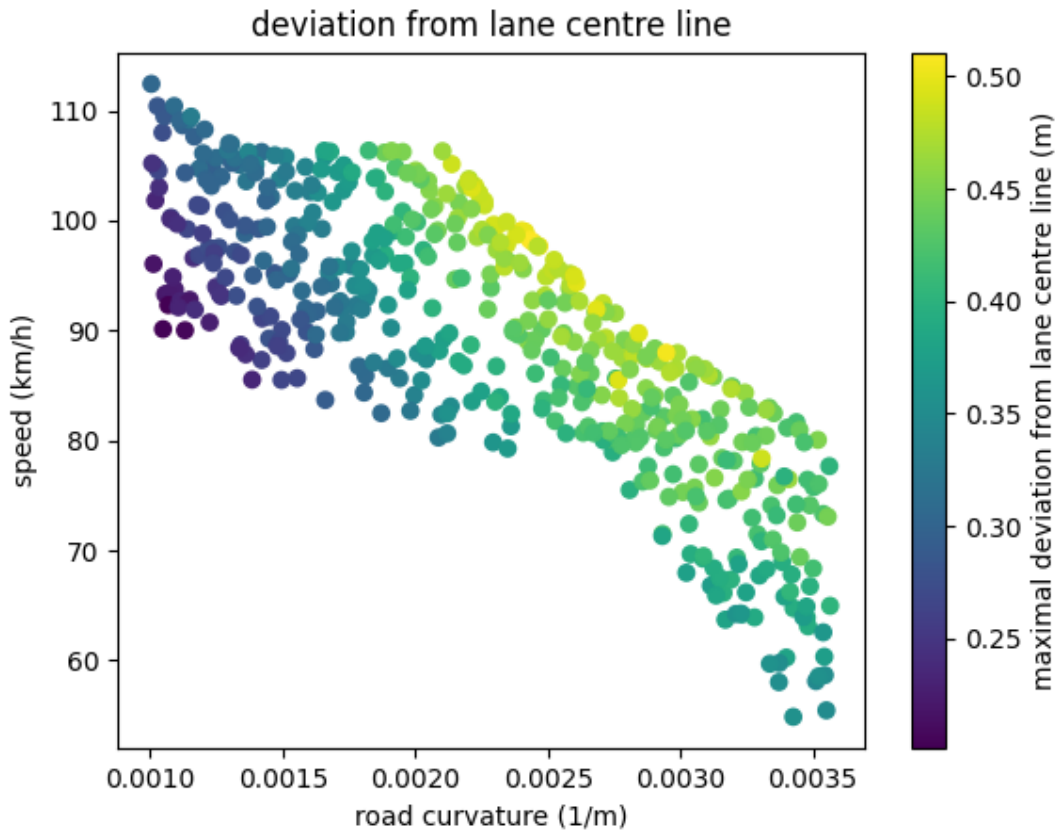


Figure 159: Use Case 3.1 B: The system reliably stays within the lane.

3.6.3.5 Coverage

Approach 1 (UNITN)

For UC 3.1-A. The coverage argument is based on the observation that, if the speed limit decreases by 20 km/h (which is usually the case), spaced at least 100 meters, the deceleration occurs well within the blue zone. The boundary at -4 m/s^2 , which could indicate a potentially dangerous deceleration exceeding human expectations, is distant (the requested deceleration restricted to the hypothesis of speed limit steps up to -20 km/h is -2.5 m/s^2).

All realistic scenarios involving a speed reduction up to -20 km/h are “internal points” as per Sunrise D3.4 Section 4.1, so additional sampling of the space to find the exact boundary is unnecessary. (Validation criteria 2.2.5 b)

Note that the requested deceleration, restricted to the condition of speed limit steps up to -20 km/h (i.e., -2.5 m/s^2), is within the maneuvering capacity of any vehicle, even in the case of rain (it might not be the case of ice with improper tires, though).

A similar argument holds for UC 3.2-B, albeit the sample points are not shown in its charts.

Approach 2 (IDI)

For UC 3.1-A and UC 3.1-B the coverage argument is based on the observation that the optimised LHS sampling (D3.4) covers the parameter space of the scenarios very well which can be also seen on the results of UC 3.1-A.

The coverage of the parameter space in UC 3.1-B is very similar as in UC 3.1-A but the low coverage of the results is related to the SUT behaviour and its target speed. This means the higher the curvature the lower the chosen speed of the system due to its target lateral acceleration of less than 3 m/s². Additionally, the target speed of the SUT is 120 km/h which results in higher maximal speeds in curves close to the lateral acceleration of 3 m/s² even the initial speed was below.

3.6.3.6 Decide

Regarding Scenario 3.1-A, the conclusion is that **the CCAM system meets the goals** outlined in section 3.5.3.2. based on the elements shown in Test Evaluate and Coverage

Regarding Scenario 3.1-B, the conclusion is the same: the CCAM system meets the goals outlined in section 3.5.3.2. However, one might argue that the vehicle uses reduced velocity in curves (approximately 0.8 times human average) which might pose some risk in the interactions with human drivers.

3.6.4 Key take aways

The SUNRISE SAF framework successfully evaluated the safety of a system in two core functions necessary for a Highway Autopilot.

Aggregated test evaluation and coverage are carried out with the simplified, yet effective, method described in Sunrise D3.4 Section 4.1.

Safety argumentation consists of evaluating case-specific KPI. The thresholds for pass/fail have been assumed to demonstrate the SAF. The procedures for defining the exact threshold values are given.

3.6.5 Deviations from D7.2

The deviations from D7.2 are divided into the SAF blocks:

Query & Concretize

The scenarios provided by UoW were adapted by experts for the needs of the UC and were exported from the database Safety Pool and not from SUNRISE DF since the framework was not ready when the scenarios were needed.

Allocate

The initial allocation of the SUNRISE SAF was covered and the methodology applied but no re-allocation was done.

Safety Argument/Decide

The added SAF block “Safety Argument” was not covered by this UC which is also not part of D7.2, therefore the Decide block was applied purely based on the “Coverage” block results and not based on the results of the “Safety Argument” block.

3.7 Use case 3.2: Highway AD validation – Cooperative perception & decision making & control

3.7.1 Use Case Overview

Within this UC 3.2 the SUNRISE SAF is validated by the safety assurance of a highway pilot (AD function). The perception of the AD function is extended with connectivity information about decelerating vehicles ahead or cutting-in vehicles from behind. Three scenarios (see section 3.6.2) were implemented to assess the safety of the system. The highway pilot controls the lateral motion of the vehicle (lane keeping but also evasive emergency manoeuvres) as well as the longitudinal control of the vehicle (ACC – keeping or reaching a safe speed and safe distance to vehicles ahead).

Objective:

By applying the SAF, demonstrate how safety and surrounding awareness can be improved on highways by including cooperative V2X communication with surrounding vehicles.

3.7.1.1 Covered aspects of the SAF

The table below shows the contributions to the SAF block validation per partner for UC3.2.

Table 22: Contributions to the SAF validation per Partner in UC3.2

Partner/ Block	IDI	ika	UNITN	UoW
SUNRISE DF				x
Query & Concretise		x	x	
Allocate	x	x	x	
Execute	x	x	x	-
Test Evaluate	x		x	-
Coverage	x	x*	x	-
Safety Case			x	
Decide	x	-	x	-

*only parameter space coverage

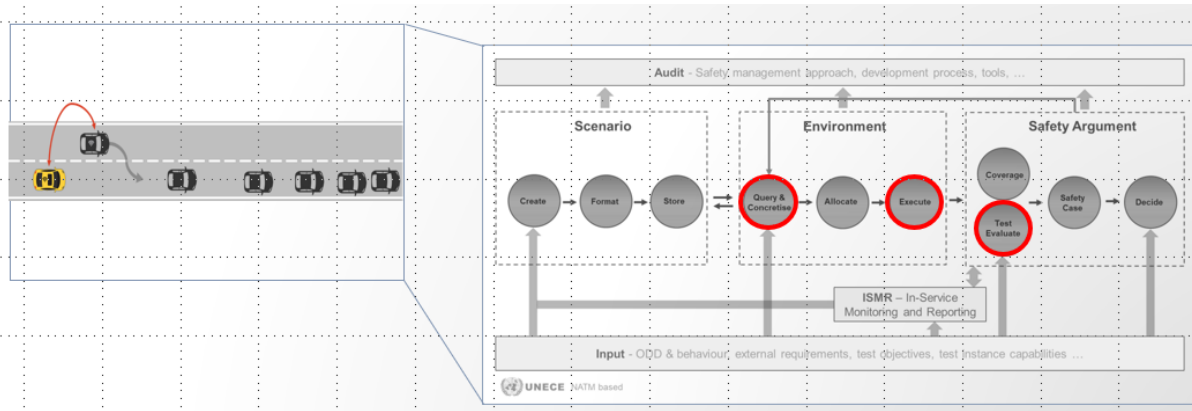


Figure 160: SAF demonstrated blocks in this UC's demo (s).

Handbook videos produced:

UC3.2-A <https://www.youtube.com/watch?v=SgN2qmvWnc>

UC 3.2 <https://www.youtube.com/watch?v=OE0e6Row2Bw&t=2s>

UC 3.2 C 1 (UNITN) <https://www.youtube.com/watch?v=mXv2oE2uHjA&t=1s>

UC 3.2 C 2 (UNITN) <https://www.youtube.com/watch?v=PGhAXYGYmVQ&t=1s>

3.7.1.2 Safety case setup

The objective is to apply the SUNRISE SAF to the safety assurance for a CCAM system in a highway scenario. The aim is to verify that the CCAM safety benefits from connectivity. The information used can come from sensors, and connectivity; however, connectivity is the primary focus. The following steps are carried out:

- Safety Goals and Key Performance Indicators:
- For the cut-in sub-use case variant 1 (UC3.2 C1), the primary indicator in simulations was collision avoidance (collision versus non-collision), which served as the key performance indicator. A surrogate indicator (near-miss) was also used to classify situations with reduced clearance. A performance indicator (the fraction of successful merges) was instead used in variant 2 of this UC (UC3.2 C2).
- For the cooperative ACC sub-use cases (UC3.2 A and B), bumper-to-bumper distance and maximum deceleration were also used as performance and surrogate metrics.
- Tests were conducted in different test environments, including simulations with IPG CarMaker at UNITN and simulations and Proving Ground tests at IDIADA. In UC 3.2 C1, simulations in IPG CarMaker were performed for three different CCAM systems: a traditional motion planner, a cognitive agent, and a simple generalized intelligent driver model. In UC 3.2 C2, only the cognitive agent was used, but with two different levels of aggressiveness.
- In simulations, the scenario space was sampled randomly. For UC 3.2 A (cooperative ACC in standard conditions), 1000 concrete scenarios were simulated. For UC 3.2 B (cooperative ACC in challenging conditions and cut-out), 592 concrete scenarios were studied. For UC 3.2 C1, 5,000 scenarios involving forced cut-in without negotiation were used; for UC 3.2 C2, 1000 concrete scenarios were used.
- Simulation outcomes have been confirmed by physical testing on selected concrete scenarios.

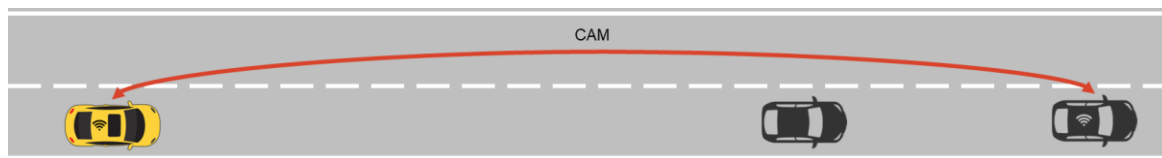
3.7.2 Overview of tested scenarios

The Use Case consists of three subcases: A) ACC detects the presence of a vehicle through communication, B) similar to A but with the cooperative vehicle possibly moving at low speed or experiencing high deceleration while the vehicle ahead exits, and C) the cooperative vehicle performs a cut-in.

For simulations, in case C, there are two variants: C1, where the cooperative vehicle simply communicates its intentions before executing the cut-in, and C2, where the cooperative vehicle negotiates the merge, only cutting in once the gap is sufficiently safe. In this latter case, the decision about whether the gap is safe enough has two thresholds: one representing a prudent cut-in and another representing an aggressive cut-in.

UC3.2 C1 was the first developed. It corresponds to a published paper [9].

A) Cooperative ACC.



B) Cooperative ACC with challenging vehicle and cut-out.



C) Cut-in of a cooperative vehicle (C1 with declared intention, C2 with negotiation).

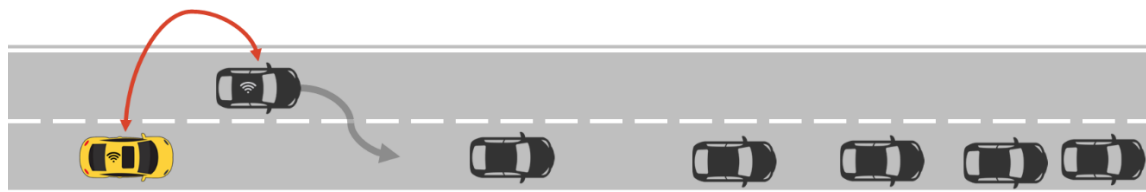


Figure 161: Use Case 3.2 includes three subcases.

3.7.3 SAF Block demonstrations

3.7.3.1 Query & Concretize

The ODD is defined in D7.1, Sections 6 and 6.3 (i.e., Highway Autopilot with V2V communication)

There are four logical scenarios. UC3.2 A, B and C-2, were defined by querying SafetyPool and were then slightly revised by experts. UC 3.2 C-1 was determined by experts in an earlier stage of the project and corresponds to a variation of the ALKS scenario .

For **UC 3.2 A**, the logical scenario has five parameters as follows. It was sampled with 1000 random concrete scenarios.

In this scenario, Ego follows a non-cooperative vehicle at 1-2 s time gap. Suddenly, it receives a CAM message from cooperative vehicle 1-8 s further ahead and tries to adapt to it. The goal is to find under which conditions it succeeds.

Table 23: UC 3.2-A logical scenario parameters

Parameter name	Parameter range
Ego initial speed range	60-130 km/h
Co-op vehicle initial speed range	60-Ego km/h
Co-op to Non Co-op initial distance (time gap)	1-8 s
Non Co-op vehicle initial speed range	Same as Ego
Non Co-op vehicle initial distance (time gap)	1-2 s
Co-op vehicle deceleration range	0-4 m/s ²

For **UC 3.2 B**, the logical scenario has five parameters as follows. It was sampled with 500 random concrete scenarios.

This scenario is similar to UC 3.2-A but involves more extreme conditions for the Co-op vehicle. The Non-Coop vehicle cuts out to prevent collisions.

Table 24: UC 3.2-B logical scenario parameters

Parameter name	Parameter range
Ego initial speed range	60-130 kph
Co-op vehicle initial speed range	0-Ego kph
Co-op to Non Co-op initial distance (time gap)	1-8 s
Non Co-op vehicle initial speed range	Same as Ego
Non Co-op vehicle distance when initiating cut out	1-2 s
Co-op vehicle deceleration range	3-10 m/s ²

For UC **3.2-C2**, the logical scenario is shown below. Note that in this scenario, both the Ego vehicle and cooperative vehicle are controlled by a CCAM system each. Therefore, the parameters of the cut-in manoeuvre are defined by the system rather than the scenarios.

Table 25: UC 3.2-C2 logical scenario parameters

Parameter name	Parameter range
Ego initial speed range	60-130 kph
Co-op vehicle initial speed range	Ego +/- 10 kph
Non Co-op vehicle initial speed range	Same as Ego
Non Co-op vehicle distance when initiating cut in	1-4 s
Co-op vehicle distance when initiating cut in	0-distance Non Co-op
Co-op vehicle deceleration range	(*)
Co-op vehicle distance duration of cut in	(*)

(*) values controlled by the CCAM system of the Cooperative vehicle.

For UC **3.2-C1**, the logical scenario has five parameters as shown below [9].

Table 26: Logical scenario parameter ranges

Variable		Range	
		min	max
v0	m/s	10	25
v1 - v0	m/s	-5	7
T	S	1.5	7.5
a	m/s ²	-6	3
d	m	-20	40

The scenarios within this use case were used as a demonstration of the SAFs querying process. They were uploaded to public database in the form of Safety Pool™. A query was crafted which utilised ODD tags and behaviours which were extracted from the original functional scenario definitions in deliverable D7.1, which can be seen in Figure 162. This query resulted in the retrieval of the 3 relevant logical scenarios for the use case.

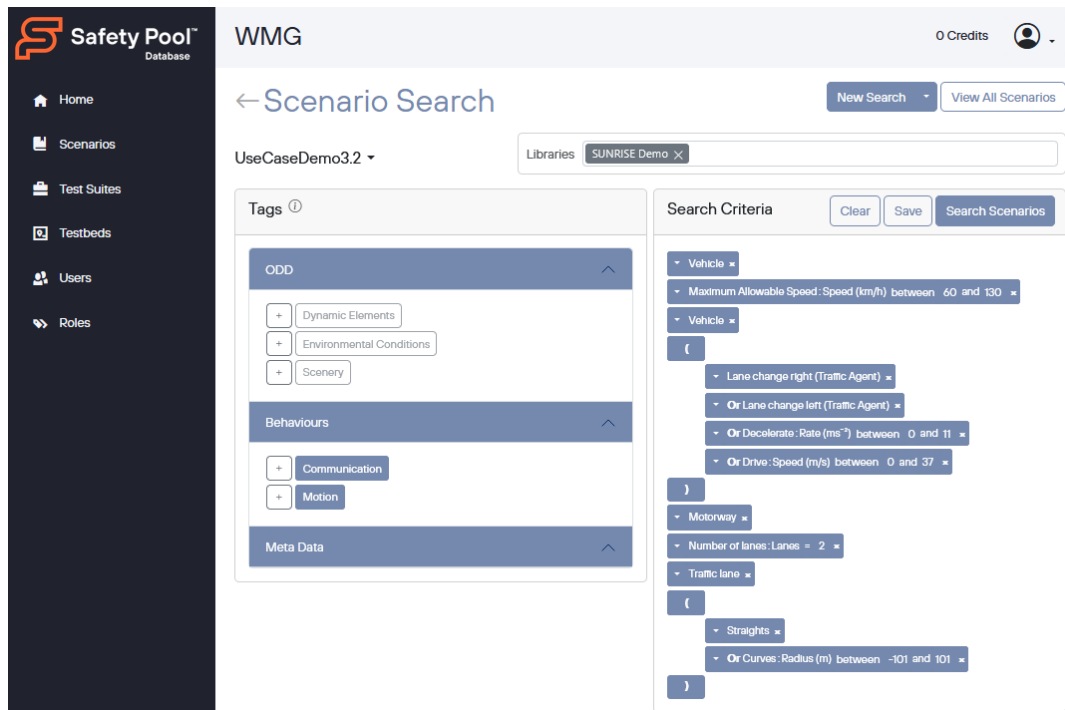


Figure 162: Search interface within Safety Pool™ SCDB demonstrating the relevant tag query of OpenLabel tags.

Once the logical scenarios were pulled from the various scenario databases they were concretized using Latin Hypercube Sampling (LHS). LHS is a stratified sampling method, it divides each input variable's range into equally probable intervals, ensuring better coverage of the space than pure random sampling. Each value (or "bin") in every dimension is sampled exactly once, avoiding clustering or gaps. Compared to simple random sampling, LHS can achieve similar accuracy with fewer samples. Further details of these methods use can be found in D3.4 [2].

This use case demonstrated an iterative sampling approach, using a surrogate model to improve sampling. The surrogate model is a representation of the underlying evaluation metric. Optimization approaches such as Bayesian Optimization can then be used to efficiently explore the space to find additional samples that reduce the uncertainty of the surrogate model.

3.7.3.2 Allocate

The Allocation has followed the ODD-based Allocation Methodology from D3.3 for initial allocation. Since the allocation process was the same as for UC 3.1 it will be not explained here once again in details. An example is provided in section UC 3.1 Allocate paragraph. The most important difference to UC 3.1 is the addition of the V2V communication which has no influence on the allocation process because the real systems are available on proving grounds implemented in the prototype vehicles but also as models in the virtual simulation test instances. And keeping the demonstration of the SAF in focus the test case requirements on the fidelity of these models are low and therefore all test cases can be performed on virtual simulation test instances and do therefore not influence the initial allocation process.

As in UC 3.1 a potential re-allocation of test cases would be feasible to do but the methodology was not ready before the use case was finished and was not in the scope of the use case which was used for SAF demonstration. However, there were no additional findings compared to UC 3.1 applying the initial allocation method and since the UC 3.1 experiences are covered by D3.5 and the re-allocation method, also this UC should be covered by D3.5 for a potential re-allocation.

3.7.3.3 Execute

In the Execute block, the test cases defined during the Allocate phase are carried out within the designated testing environments. This section presents the methodology used to validate the Highway Automated Driving (AD) functionalities, with a focus on UC 3.2.

Two testing environments were employed to enable an exhaustive validation process, ensuring the ability to perform complex scenarios in a safe manner.

- **Virtual Testing:**

Approach 1 (UNITN). The concrete scenarios were executed in IPG CarMaker (D4.3, Sections 2.1.4, 2.2.4, 2.3.5, 2.4.2, 2.5.1, 2.6.1, 4.3.2) corresponding to Validation Criteria in Section 2.2.4 a, c), recording the complete logs and extracting the longitudinal velocity, deceleration, nearest position, collisions, orientation and lateral position. (Validation criteria in Section 2.2.4 b). The chosen KPI are derived from expert knowledge about the UC (2.2.4 b and 2.2.6 a).

Approach 2 (IDI). The concrete scenarios were executed in MathWorks Automated Driving toolbox corresponding to Validation Criteria in Section 2.2.4 a, c), recording the complete logs and extracting the longitudinal velocity, deceleration, nearest position, collisions, orientation and lateral position. (Validation criteria in Section 2.2.4 b). The chosen KPIs are derived from expert knowledge about the UC (2.2.4 b and 2.2.6 a).

- **Proving Ground Testing:**

Approach 1 (IDI). The defined scenarios were executed in a controlled and secure environment that simulated highway driving conditions. The Highway Pilot (HWP) function under test was deployed and validated in the prototype vehicle (CAVKit) by means of the variants described in the UC 3.2, including the connected and cooperative driving manoeuvres. Specifically, cooperative Adaptive Cruise Control (C-ACC) for both approaching and cut-out scenarios and a cooperative cut-in manoeuvre.

Proving Ground Testing

This section focuses on the validation of the implementation of the V2V communication in a HWP system, highlighting key considerations for its integration. A detailed explanation of the HWP setup, including instrumentation, system architecture, and development, is provided in Section 3.6.3.3 of this document.

Depending on the use case variation, different communication approaches were employed.

In the **UC 3.2-A**, the scenario involved a non-cooperative vehicle in between the ego and cooperative vehicles. The ego vehicle exchanged Cooperative Awareness Messages (CAM) with the cooperative vehicle to monitor its speed, position, and other relevant parameters. The non-cooperative vehicle, which was not connected to the communication network, travelled at a higher speed than the cooperative vehicle.

When the cooperative vehicle began to brake, the non-cooperative vehicle, lacking prior awareness of the deceleration, was forced to brake more abruptly upon approaching it. In contrast, the ego vehicle, having received CAM updates from the cooperative vehicle, was able to anticipate the braking event and adjust its speed accordingly.

This communication enabled smooth and safe longitudinal control by allowing the ego vehicle to actively respond to changes in traffic dynamics and maintain a safe following distance. In the absence of such situations, the ACC system would have braked at a later point in time, increasing the risk of collision and reducing comfort due to abrupt deceleration.

UC 3.2-B employed a similar communication approach in a comparable scenario, with the addition of a cut-out (lane change) by the non-cooperative vehicle instead of braking. Despite lacking a direct line of sight to the cooperative vehicle due to the presence of the non-cooperative vehicle in between, the ego vehicle was able to track its position and motion state through the continuous exchange of CAM messages. This real-time information allowed the ego vehicle to anticipate traffic dynamics ahead. As a result, although the non-cooperative vehicle performed the cut-out manoeuvre in response to the slower speed of the cooperative vehicle ahead, the ego vehicle was able to maintain a safe distance to the cooperative vehicle without the need for a high deceleration.

In the **UC 3.2-C2**, the negotiation of a cut-in manoeuvre was initiated by the cooperative vehicle, which signaled its intent using the corresponding blinker. This process was supported by the exchange of both CAM and Manoeuvre Coordination Messages (MCM). MCM messages were transmitted throughout the manoeuvre, beginning with the cooperative vehicle's declaration of intent. The initial request aimed to create a gap in the traffic flow, providing sufficient space for the cooperative vehicle to safely perform the cut-in. During the execution phase, continuous MCM exchanges ensured cooperative and coordinated negotiation between the involved vehicles. CAM messages complemented this process by providing real-time updates on each vehicle's dynamic state. Finally, dedicated MCM messages were used to indicate the successful completion of the manoeuvre, thereby closing the coordination and negotiation process.

For the execution of this use case, two additional vehicles were employed alongside the one described in Section X. One of these was a cooperative vehicle equipped with connectivity and localization capabilities, while the other acted as a support vehicle (non-cooperative) to enable the validation of the functionality in the described test scenarios. The two relevant vehicles, the ego and the connected/cooperative, are shown in Figure 163.



Figure 163: Use Case 3.2: Cooperative Vehicle and Ego Vehicle used.

The cooperative vehicle, actively involved in the test scenarios, was equipped with a dedicated connectivity and localization stack. To complement the setup described in Section X, a V2X communication system was also equipped in the CAVKit platform, becoming one of the main components integrated into the Toyota RAV4. The purpose of this communication system was to enable interaction with other vehicles and road infrastructure.

The overall physical architecture of the CAVKit is shown in Figure 164. The platform supports C-V2X communication, allowing it to send and receive Cooperative Awareness Messages (CAM) and Manoeuvre Coordination Messages (MCM) from other vehicles. Additionally, the 5G modem hosted within the instrumentation rack provides high-speed Wi-Fi connectivity to both the vehicle's occupants and onboard systems.

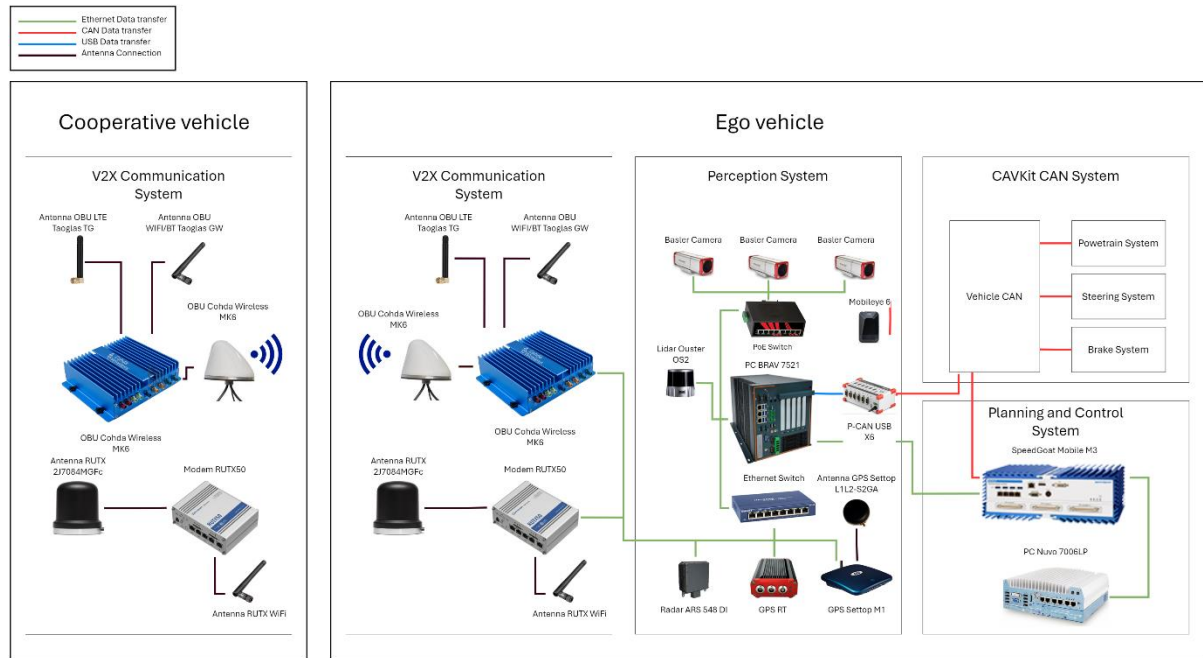


Figure 164: CAVKit Physical Architecture.

In terms of the overall concept developed to validate the functionality in the use case, the Speedgoat system received input from both the perception layer and the connectivity modules, enabling accurate response of the connected HWP system during the execution of the defined scenarios. Bidirectional communication messages were transmitted via the User Datagram Protocol (UDP), ensuring coordinated operation of the on-board devices. These interactions facilitated V2V communication and complemented the sensor-based data processed by the data fusion algorithms, already explained in Section X.

The interaction workflow between the different systems used for all variants of the UC 3.2 is illustrated in Figure 165.

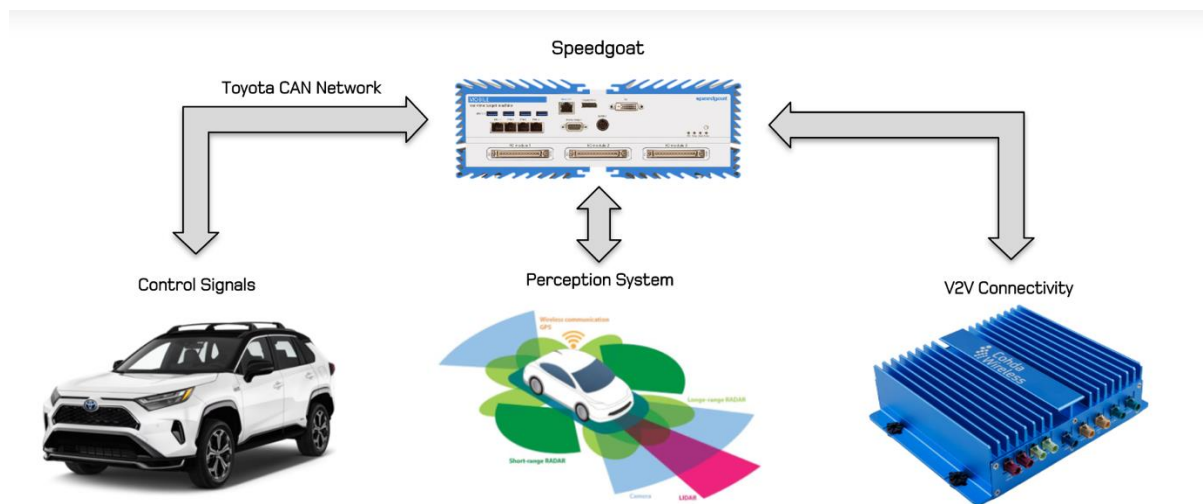


Figure 165: Communication workflow between related systems used in UC 3.1 and UC 3.2.

Regarding the UC 3.2-C-2, both vehicles involved in the scenario are required to talk each other to negotiate and agree the cut-in manoeuvre, in order to avoid risky situations when executing that move. From the V2X point of view, such communication has been implemented using standard MCMs (Manoeuvre Coordination Messages), defined by the work-in-progress ETSI standard (ETSI TS 103 561).

The general sequence diagram of such communication between the vehicles is:

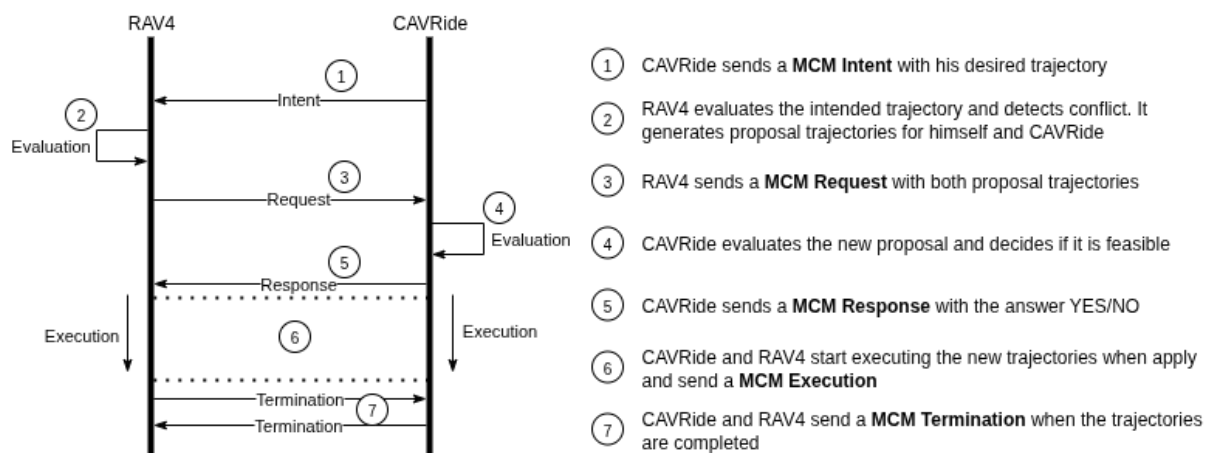


Figure 166: Negotiation workflow between related platforms used in UC 3.2-C.

In this use case, 5 types of MCM are used: Intent, Request, Response, Execution and Termination. All of them are composed by three mandatory containers: header + basicContainer + MCMcontainer.

3.7.3.4 Coverage / Test Evaluate / Safety Case / Decide

The final component of the SUNRISE Safety Assurance Framework is the Safety Argument, which consolidates and evaluates the evidence collected throughout the testing activities.

We present the sub use cases in the chronological order of their execution.

UC 3.2-C1, Approach 1 (UNITN)

For UC 3.2-C1, paper [10] presents the construction of a surrogate model via bootstrapping aggregation of 100 neural predictors (similar to the methods in D3.4, except that bootstrapping aggregation of neural learners is employed instead of Gaussian Processes) (Validation Criteria 2.2.4 a-c. 2.2.5 b-c). The surrogate model was trained with 5000 concrete scenarios and validated² on 1613 new and independent concrete scenarios. Refer to the indicated paper for details (Validation Criteria 2.2.6 c).

² The term “validated” refers to validation of the surrogate model, which is accomplished by evaluating the trained model predictions on an independent test set, assessing the classifier’s performance (e.g., prediction errors in terms of false positives, false negatives, accuracy, etc.).

The surrogate model predicts collision, near-miss, and safe conditions with confidence levels, as shown for example in Figure 167. (Validation Criteria 2.2.6 a-b).

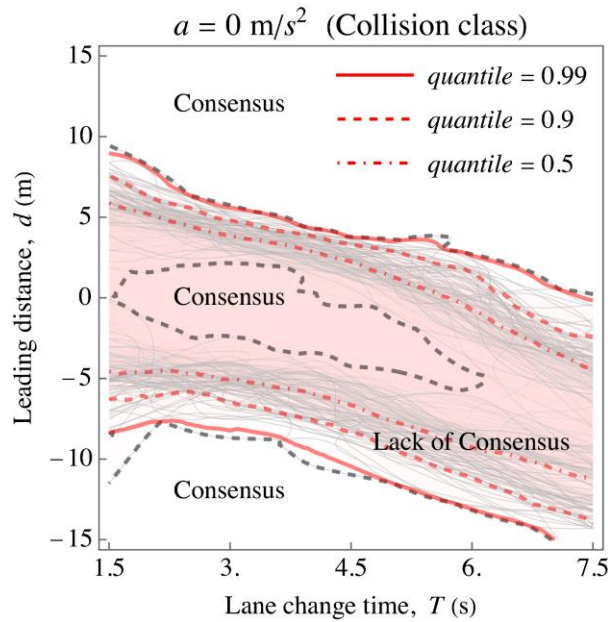


Figure 167: Use Case 3.2 2-C-1. Confidence levels associated with the collision class.

Using the surrogate model, various safety arguments are developed (Validation Criteria 2.2.7 a):

First, the risk for three different types of cut-in behaviours is evaluated for two different CCAM systems as shown in Figure 168. (Validation Criteria 2.2.7 b-d).

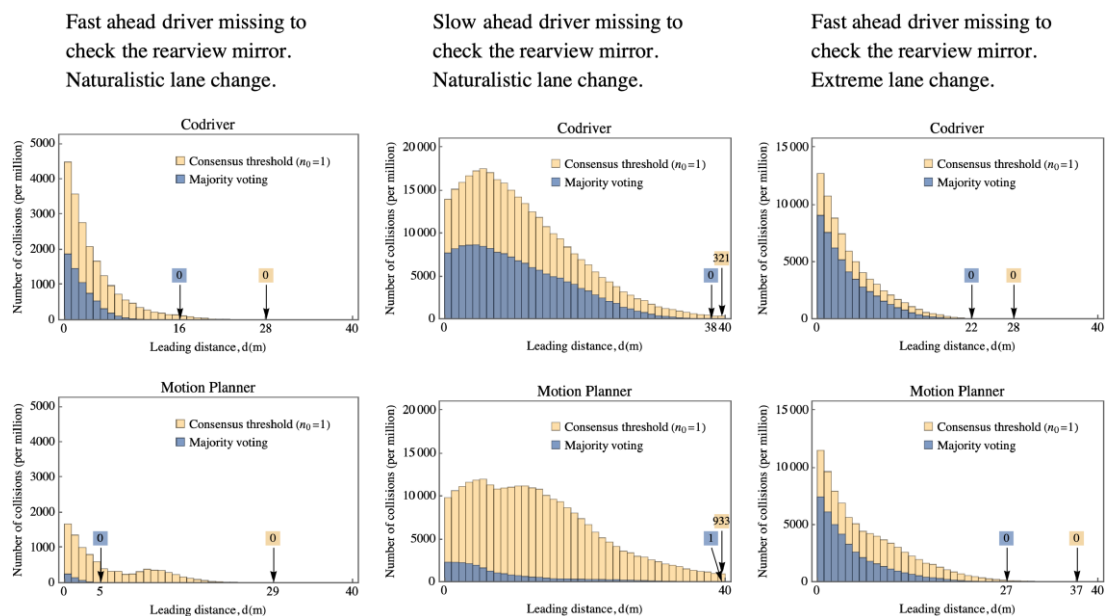


Figure 168: Use Case 3.2 2-C-1. Use of surrogate model to evaluate the collision risk for three behaviours of the cut-in vehicle, at two confidence levels and for two CCAM systems.

Second, different types of failure modes are discovered by sampling the scenario at the edge of the safe-unsafe region (Sunrise D3.4). Figure 169 shows the resulting critical trajectories obtained from 2,000 samples at the boundary and grouped into 19 clusters (only 6 are shown). (Validation Criteria 2.2.7 b-d).

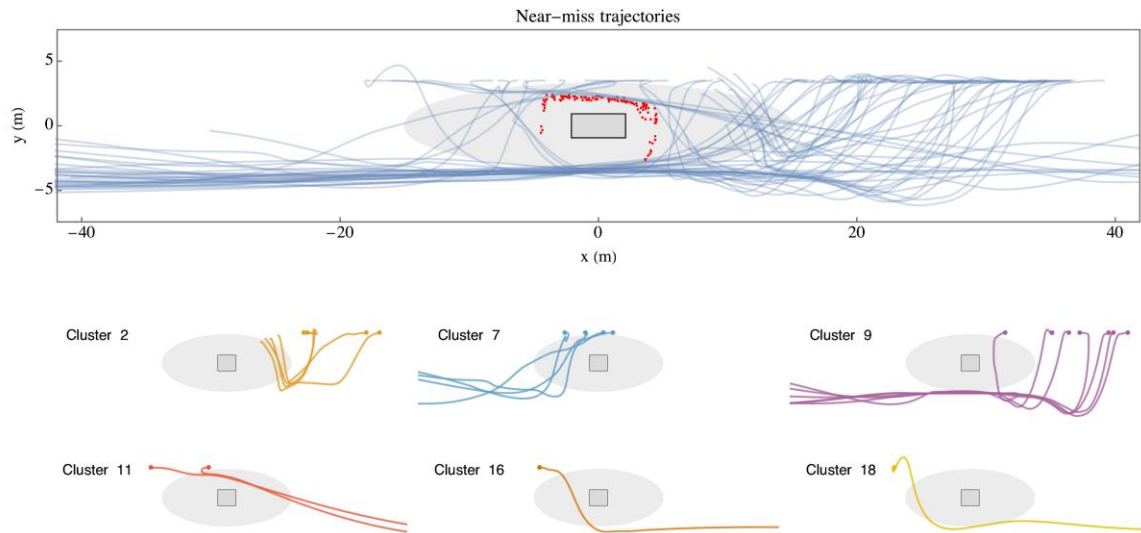


Figure 169: Use Case 3.2 2-C-1. Analysis of scenarios at the safe-unsafe boundary and clustering into different types of failure modes.

Third, the surrogate model can be used to derive safe speed recommendations, i.e., velocities that enable the system to remain in the collision-free subspace with a given confidence, as shown in Figure 170. (Validation Criteria 2.2.7 e-f).

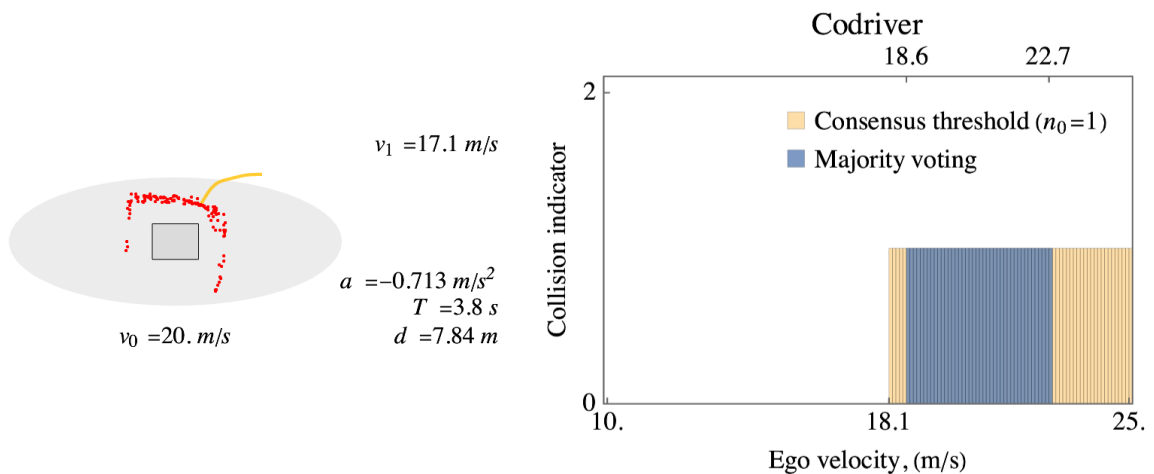


Figure 170: Use Case 3.2 2-C-1. Use of the surrogate model to recommend safe speed policies.

UC 3.2-C1 Approach 2 (IDI)

The cut-in vehicle communicates its trajectory and executes it, leaving Ego responsible for avoiding collisions. The following chart (Figure 171) shows if the cut-in vehicle is faster than the VUT a TTC value of 2 s and more is kept by the SUT which corresponds to safe driving

behaviour. At the centre of the scatter plot the cut-in vehicle is not much slower than the VUT or has a similar speed, means these results represent the comfort braking of the SUT. In the lower left corner of the scatter plot the cut-in vehicle is much slower than the VUT and the SUT performs an emergency braking but still no collisions happened (the minimal free longitudinal distance occurred was around 10 metres) and only in 4 extreme cases a TTC value of 2 s was undercut.

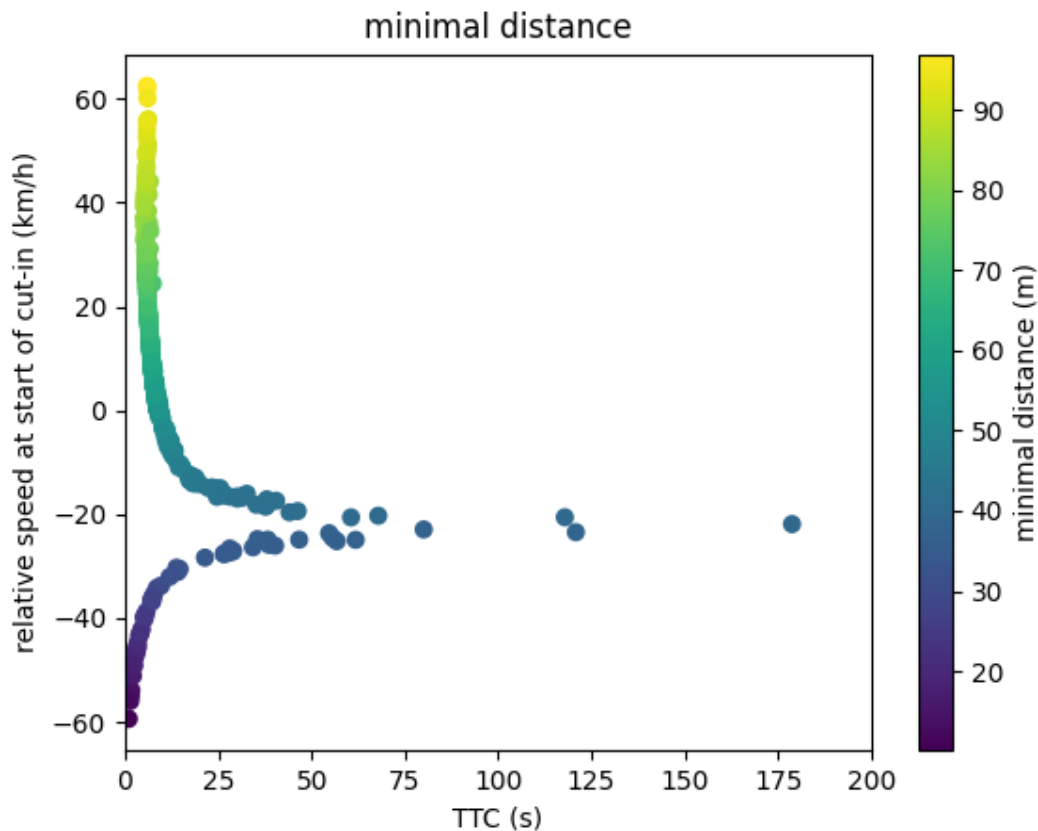


Figure 171: Use Case 3.2-C1. Scatter plot showing the minimal longitudinal free distance during cut-in.

In conclusion, the SUT behaves safe because it keeps a safe TTC of 2 s in nearly all simulated cases and prevents potential collisions.

UC 3.2-C2 Approach 1 (UNITN / IDI)

Use Case 3.2-C2 differs from C1 in how cooperation occurs. In C1, the cut-in vehicle communicates its trajectory and executes it, leaving Ego responsible for avoiding collisions. In C2, the cut-in vehicle is controlled by another instance of the CCAM system. After communicating the merge request, the cut-in vehicle evaluates whether the gap is safe before merging.

The events follow this logic:

- Ego 2 asks Ego 1 to open a gap.

- Ego 1 agrees and reduces its speed.
- Ego 2 then decides how and when to merge into the opening.

Two parametrisations are used for the Cooperative vehicle: cautious (accepts 1-second time gaps) and aggressive (accepts 0.5-second time gaps). A total of 1000 simulations were used (500 each). (Validation Criteria 2.2.4 a-c).

Since the Cooperative vehicle looks for a sufficient gap before merging, no collisions were observed (Validation Criteria 2.2.6 a). Nonetheless, merge success varies: the cautious driver achieved a success rate of 42% (210 out of 500 scenarios), while the aggressive driver achieved 97% (485 out of 500 scenarios).

Note that although there are no collisions in either case, the systems are not equally safe (Validation Criteria 2.2.7 a). In fact, when using a surrogate metric like time headway, the aggressive setting often operates at 0.5 seconds time headway, which carries a higher risk but allows for faster merging. The following images illustrate the paths taken by both versions of Ego.

The merge trajectories are shown in Figure 172. Notably, the merge manoeuvre is completed significantly earlier with an aggressive setting. Figure 168 shows the time to cross the lane in the two settings.

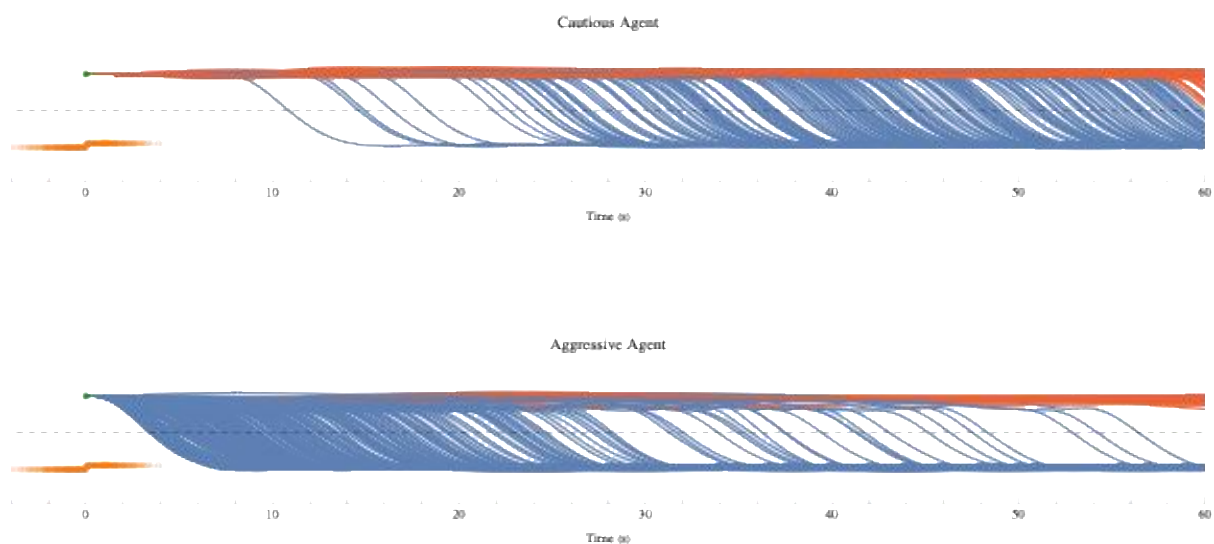


Figure 172: UC 3.2-C2. Successful merge trajectories for the cautious (top) and aggressive (bottom) settings of Ego 2.

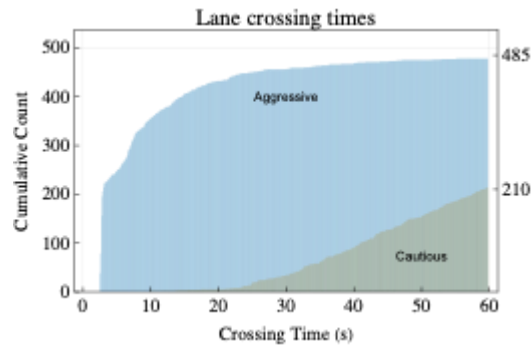
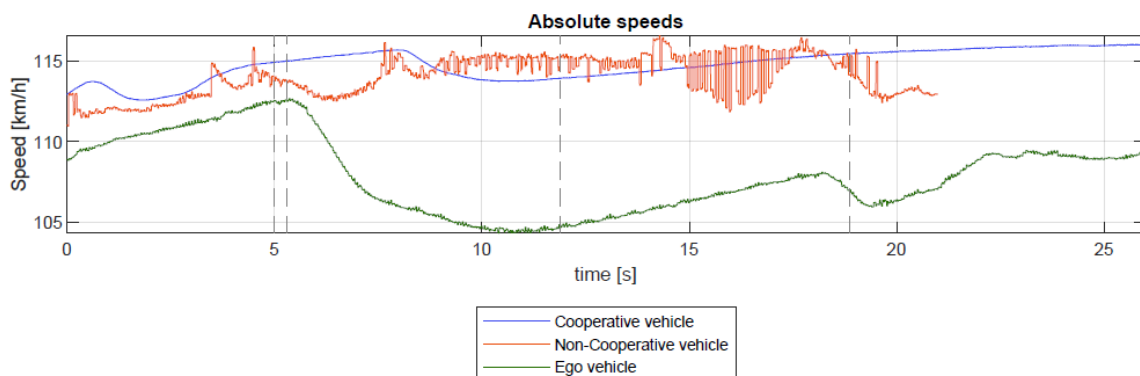


Figure 173: UC 3.2-C2. Successful merge trajectories for the cautious (top) and aggressive (bottom) settings of Ego 2.

To validate the simulation results according to the SAF, 3 tests at different speeds (80 km/h, 100 km/h and 120 km/h) have been carried out on proving grounds, with the negotiation between vehicles and the cut-in manoeuvre with a short distance as referred in the above-mentioned aggressive pattern (Validation Criteria 2.2.7 c).

Figure 174 shows the time series evolution of the 3 vehicles during the execution of the scenario at highest tested speed. The table in the figure shows the KPIs for the key moments of the execution and it reflects a starting gap of -2.9m between the subject vehicle's front bumper and the cooperative vehicle's rear bumper that increases up to 8.5m at the start of the lane change manoeuvre and up to 21m when the cooperative vehicle enters in the ego vehicle's path, due to the ego vehicle's speed adaptation after the negotiation. Thus, it is possible to validate the results obtained in the simulations and to determine that the operation is safe in such a situation.



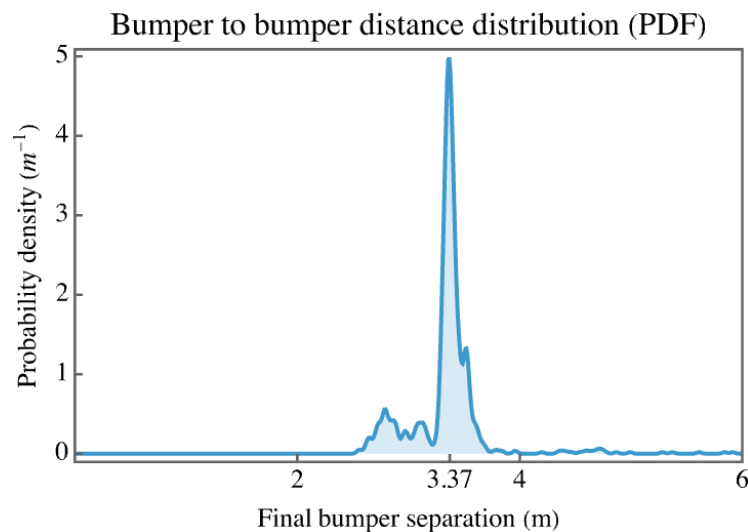
	ManoeuvreStart	EgoBrake	CoopCutIn	CoopInEgoPath
Time [s]	5	5.303	11.88	18.86
Gap to Coop [m]	-2.924	-2.87	8.482	21.17
Gap to Non-Coop [m]	49.58	49.89	64.36	81.3
Coop spd [km/h]	114.9	115	113.9	115.5
Non-Coop spd [km/h]	113.8	113.8	114.6	114.1
Ego spd [km/h]	112.5	112.6	104.8	106.9
Lat dist Coop [m]	-3.827	-3.86	-3.499	-0.896

Figure 174: Use case 3.2-C-2. Results from proving ground execution at 120km/h.

UC 3.2-A **Approach 1** (UNITN / IDI)

UC 3.2-A counts 1000 concrete scenarios. A first focus is on collision risk (Validation Criteria 2.2.4 b). Figure 175 shows the distribution of the bumper-to-bumper distance at stop (top) and the contour plots across the logical scenario (the x-axis is the time gap when the first communication is received, and the y-axis is the difference in velocity³).

The worst case (within the logical scenario parameter range) corresponds to a minimum distance of 2.55 m. Hence, the system is collision free (Validation Criteria 2.2.7 a).



³ In the logical scenario, the cooperative vehicle is always slower than ego (faster cases do not pose risks), and it has only negative acceleration.

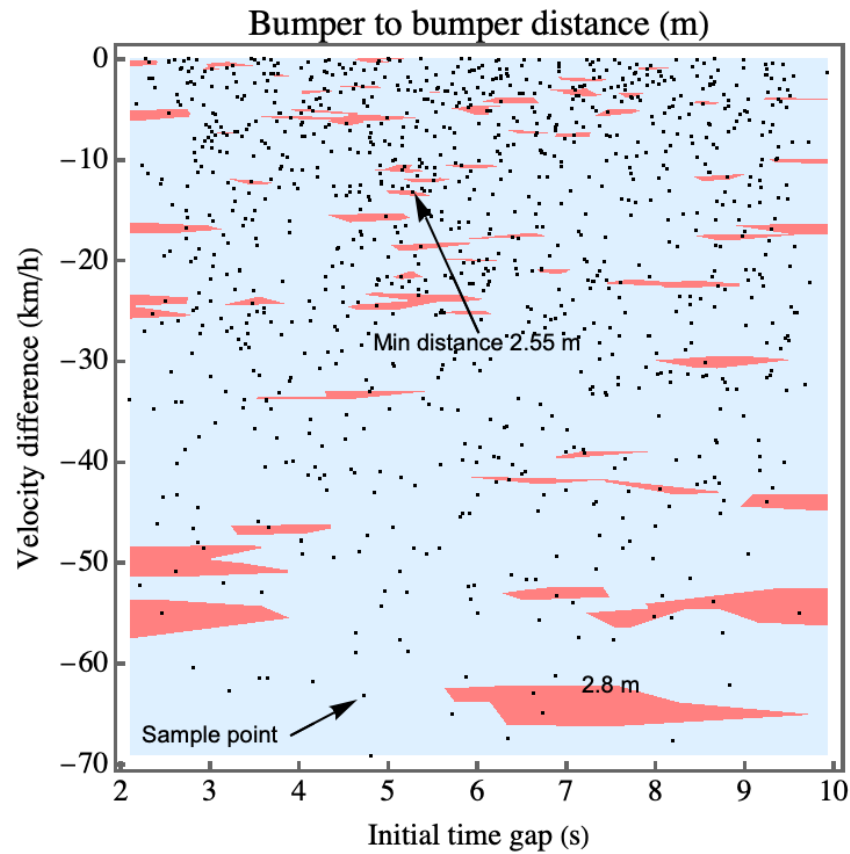


Figure 175: UC 3.2-A. Distribution of the bumper-to-bumper distance at stop (top) and the contour plots across the logical scenario (the x-axis is the time gap when the first communication is received, and the y-axis is the difference in velocity. The red zones are the regions with distance below 2.8 meters- The minimum distance is indicated.

Figure 176 shows the distribution of Ego's decelerations. In this case, the deceleration can be as high as -7 m/s^2 (posing rear-end collision risks), even though the cooperative vehicle's deceleration range (in the logical scenario) is limited to -4 m/s^2 . This happens because, at the detection time, the cooperative vehicle might already be slower and/or already braking, and the communication delay might be too late.

This situation is hence potentially unsafe. In certain cases (late communication from an already slow vehicle), a large deceleration is required to avoid collision. The system is collision-free, but not comfortable in some cases.

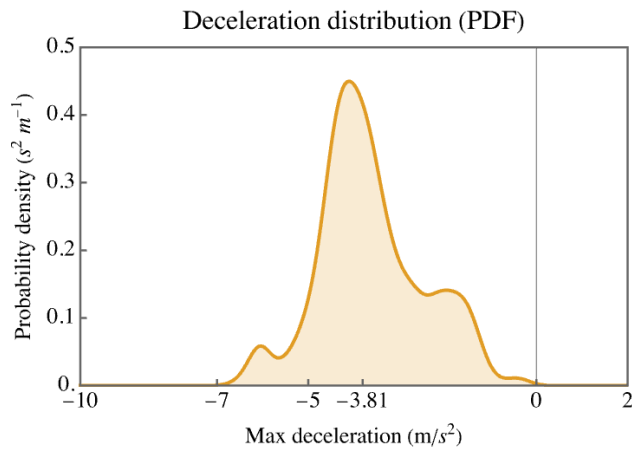
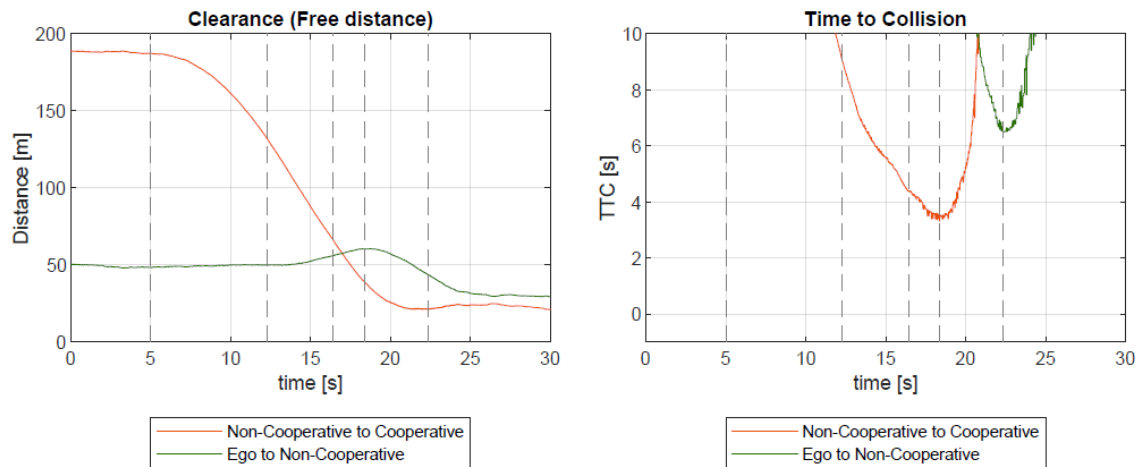


Figure 176: UC 3.2-A. Distribution of Ego vehicle deceleration after analyzing the 1000 simulations. The large decelerations observed in some cases happen because, at the detection time, the cooperative vehicle might already be slower and/or already braking, and the communication delay might be too late.

To validate the simulation results according to the SAF, 3 tests at different speeds (80 km/h, 100 km/h and 120 km/h) have been carried out on proving grounds (Validation Criteria 2.2.7 c). In all situations, the ego vehicle shows an anticipatory reaction to the cooperative vehicle braking with respect to the non-cooperative in between. This behaviour can be observed in the Figure 177, which exhibits the execution of this use case at 100 km/h. The ego vehicle starts braking (because of the occluded cooperative vehicle) 4 seconds before the non-cooperative, allowing to maintain a TTC higher than 6s at all times during the scenario execution. Moreover, all this is achieved by using a maximum deceleration level of 2 m/s^2 , which can be considered as comfortable and safe behaviour.

For this reason and based on the test runs in the various test environments, the CCAM system can be considered safe in the evaluated scenario, although in some cases it requires uncomfortable decelerations to avoid collisions. albeit in some cases it needs uncomfortable avoid collisions.



	CoopBrake	EgoBrake	NonCBrake	MinTTCNonC	MinTTCEgo
Time [s]	5	12.26	16.43	18.39	22.34
TTC Non-Coop [s]	>30	9.058	4.375	3.316	16.8
TTC Ego [s]	>30	>30	20.23	>30	6.497
Gap Non-Coop [m]	187.2	132.3	66.06	38.97	21.53
Gap Ego [m]	48.45	49.84	55.83	60.31	43.75
Coop spd [km/h]	95.94	43.49	36.36	36.04	36.04
Non-Coop spd [km/h]	95.46	96.08	90.71	78.35	31.42
Ego spd [km/h]	94.88	96.38	80.78	72.43	55.66

Figure 177: UC 3.2-A. Results from proving ground execution at 100km/h.

UC 3.2 B Approach 1 (UNITN / IDI)

This Use Case is a challenging version of the previous one (It follows the same validation criteria of UC 3.2-A). The challenge arose from the fact that the cooperative vehicle might be nearly stopped at the time of detection (second parameter in the logical scenario) and/or might be hard braking (last parameter of the logical scenario).

Figure 178 shows the bumper-to-bumper distance, which in several cases may be a negative value (indicating a collision). Collisions are shown in red colour on the chart at the right. The critical factor is the initial velocity difference. If it were less than 20 km/h there would be no collision, even if the cooperative vehicle decelerates hard (that is because Ego has similar braking capacity).

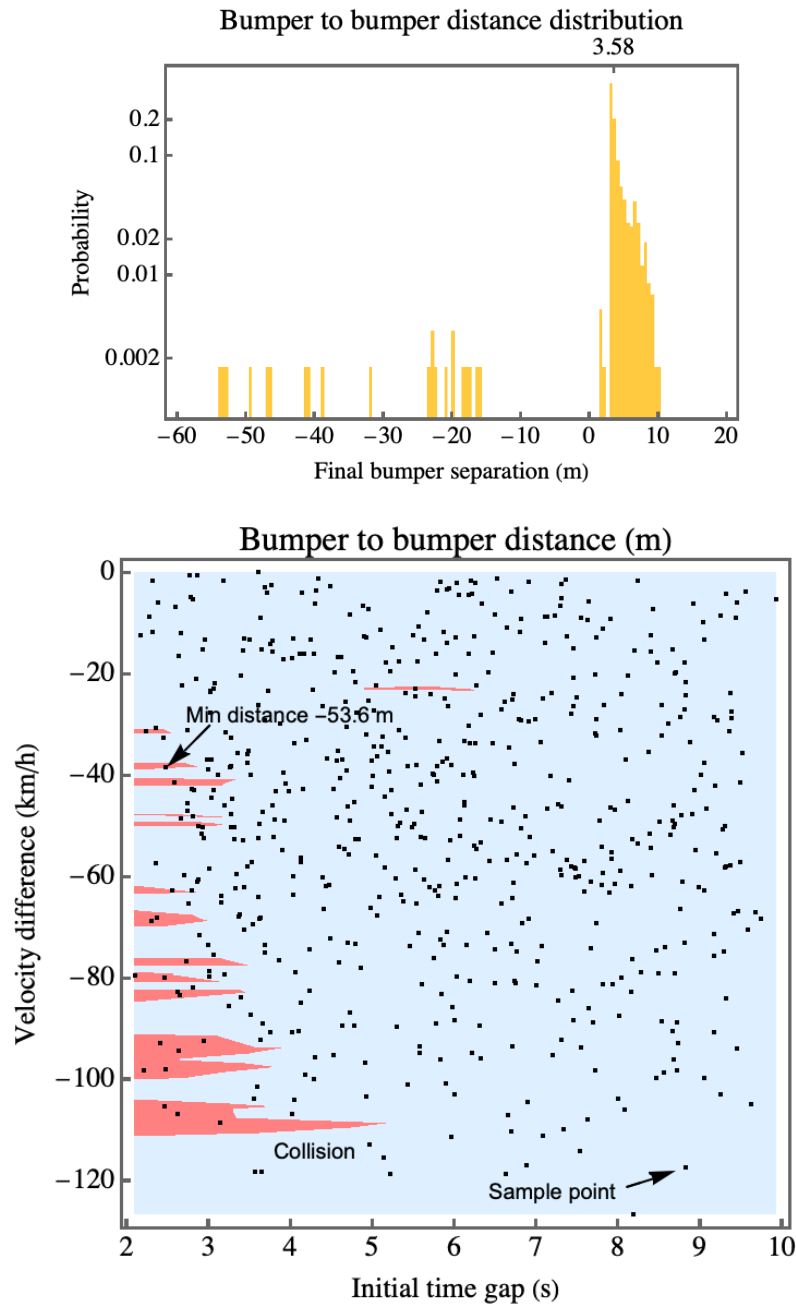
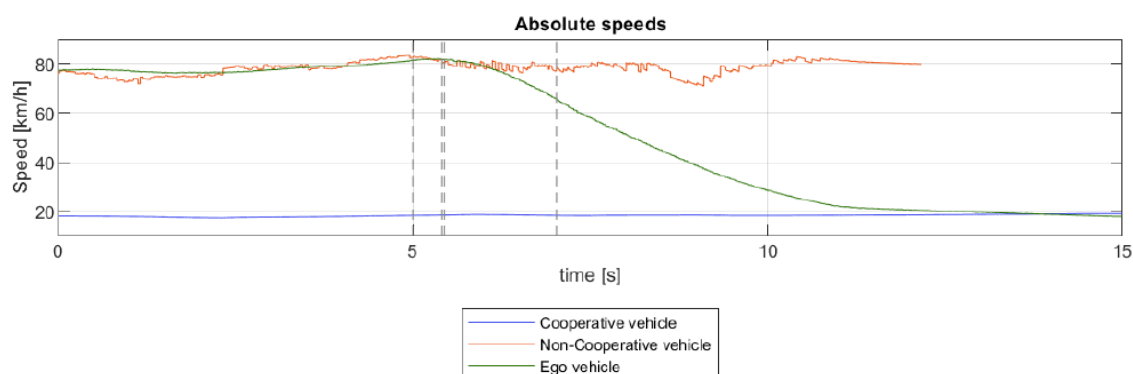


Figure 178: UC 3.2-B. Distribution of the bumper-to-bumper distance at stop (top) and the contour plots across the logical scenario (the x-axis is the time gap when the first communication is received, and the y-axis is the difference in velocity).

In conclusion, we deem that the system is safe only if the obstacle is detected (communication received) when the velocity difference is small. The system does not prevent collisions with stationary obstacles if they are detected with insufficient anticipation.

To validate the simulation results according to the SAF, 3 tests at different speeds (80 km/h, 100 km/h and 120 km/h) have been carried out on proving grounds. In all situations, the ego vehicle shows an anticipatory reaction and starts braking at the same time or before than the cut-out manoeuvre is performed by the non-cooperative vehicle. This behaviour can be observed in Figure 179, which exhibits the execution of this use case at 80 km/h. The ego vehicle starts braking at the same time the non-cooperative departs from the ego's path (no

overlap) allowing to maintain a TTC with respect to the slow-moving cooperative vehicle higher than 2.5s at all times during the scenario execution. This is achieved by using a maximum deceleration level of 4 m/s², which can be considered as a safe behaviour with 60 km/h of velocity difference.



	StartLnChg	CutOutNoOvrLap	EgoBrake	minTTCEgo2Coop
Time [s]	5	5.414	5.448	7.013
TTC Non-Coop [s]	1.697	1.338	1.308	-0.337
TTC Ego [s]	4.095	3.652	3.625	2.877
Gap Non-Coop [m]	30.31	22.96	22.43	-5.632
Gap Ego [m]	71.54	64.04	63.51	37.91
Coop spd [km/h]	18.47	18.61	18.65	18.5
Non-Coop spd [km/h]	82.76	80.4	80.37	78.66
Ego spd [km/h]	81.36	81.75	81.72	65.94

Figure 179: UC 3.2-B. Results from proving ground execution at 80km/h.

3.7.4 Key take aways

This UC presents two different logical scenarios for case C, representing two extreme cases of a continuous spectrum. IDI carried out a physical test for a case within that spectrum, but closer to C2. Overall, the proving round test qualitatively matches the simulation, considering that the cut in the vehicle was human driven instead of being driven by another SuT instance.

UC 3.2-C1 shows multiple iterations with resampling, which follow analysis with a surrogate model.

Coverage is justified with the surrogate model in case C, and with simpler (boundary points) arguments for cases A and B.

The surrogate model in case C1 is exploited to derive safe speed recommendations to keep away from critical scenarios.

The surrogate model in case C1 is exploited to resample the logical scenarios at the boundary of collisions and to find failure modes.

Deviations from D7.2

The deviations from D7.2 are divided into the SAF blocks:

Query & Concretize

The scenarios provided by UoW were adapted by experts for the needs of the UC and were exported from the database Safety Pool and not from SUNRISE DF since the framework was not ready when the scenarios were needed.

Allocate

The initial allocation of the SUNRISE SAF was covered and the methodology applied but no re-allocation was done.

3.8 Use case 4.1: Freight vehicle automated parking validation – Truck low-speed perception & decision making

3.8.1 Use Case Overview

Use case 4.1 (UC4.1) focus on the reverse parking of a truck with a semitrailer at a logistic hub as illustrated in Figure 180. From validating specific aspects of the SAF, the use case is attractive due to the controlled environment, limited range of possible scenarios, and low speed. It is a sub-use case of confined area use cases like “Automated trucks for operation in logistics terminals, quarries and construction sites” and “Trucks in hub-to-hub operation between terminals” [11]. These confined areas use cases, with perimeter protections and a reduced risk of unauthorized entry, offer the advantage of a well-defined ODD. These environments and highway use cases are expected to be among the first to support deploying highly automated heavy vehicle functions, aspects that make them essential to SAF validation efforts.

Objective:

The objective is to demonstrate how the SUNRISE SAF is applied for safety assurance of a reverse backing function focusing on its performance within defined operational conditions and adherence to safety requirements.



Figure 180: Illustration of the use case with a truck with semitrailer to dock at a logistic hub.

3.8.1.1 Covered aspects of the SAF

The table below shows the contributions to the SAF block validation per partner for UC4.1.

Table 27: Contributions to the SAF validation per Partner in UC4.1

Partner/ Block	CHAL	RISE
SUNRISE DF		
Query & Concretise	X	X
Allocate		X
Execute	X	X
Test Evaluate	X	X
Coverage		
Safety Case		X
Decide	X	X

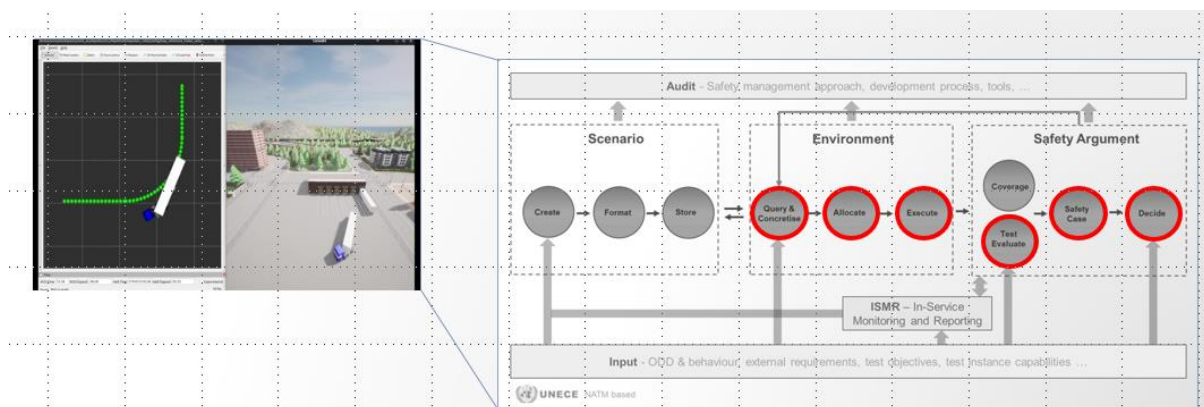


Figure 181: SAF demonstrated blocks in this UC's demo (s).

UC 4.1 https://youtu.be/Jy0QHD_BNpc?t=1

3.8.1.2 Safety case setup

The objective is to demonstrate how SUNRISE SAF is applied for safety assurance of a reverse backing function focusing on its performance within defined operational conditions and adherence to safety requirements. The following key steps are performed to contribute to building the safety case:

- Safety Goals and Key Performance Indicators defined out of a hazard analysis and risk assessment.
- Automatic test instance allocation through an approach describing the ODD and the test instance capabilities in a machine reading format using the Pkl language.
- Tests performed in different test environments.
- The scenario space is covered by using exhaustive search over distinct parameters.
- Simulation results verified through physical tests.
- The test results are evaluated in the defined Key Performance Indicator
- Five Safety Performance Indicators are defined for the safety case.
- The Key Performance Indicators are evaluated versus decision criteria.

. The black-box testing is performed in three differing test environments:

- Full-size truck
- 1:14 scale truck
- Simulations

3.8.2 Overview of tested scenarios

The scenario to be demonstrated is UC 4.1 is shown in Figure 182 and is based on the indicative test scenario described in Sunrise D7.1 [3]. The truck starts from the staging area and drives backwards until it parks in the docking area. The logical scenario consists of all possible positions and orientations (and possibly other dynamic states) that a truck with a trailer could assume within a square staging area, with all environmental parameters (including lighting) set to baseline conditions.

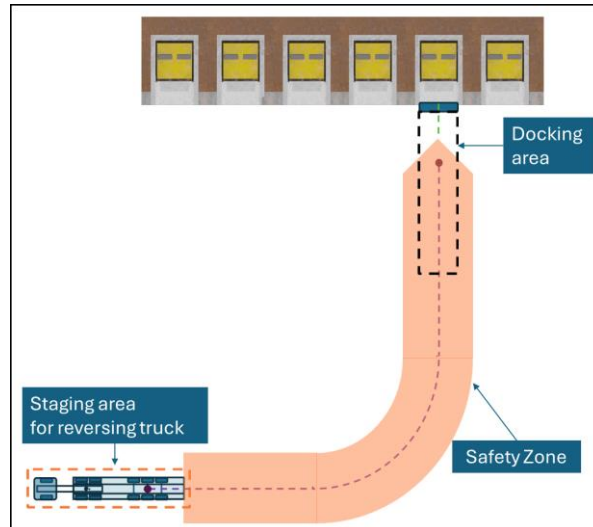


Figure 182: Schematic view of the test scenario to be demonstrated.

3.8.3 SAF Block demonstrations

As part of the input to the SAF process a hazard analysis and risk assessment (HARA) [12,13] was performed and Safety Goals (SGs). The indicative test scenario described in SUNRISE deliverable D7.1 [3] was used as bases for the HARA and following Safety Goals (SGs) were defined:

- SG1: Vehicle shall not collide.
- SG2: Vehicle shall not operate if conditions are not fulfilled.

Based on the SGs, following Key Performance Indicators (KPIs) were then defined:

- KPI1: Docking precision.
- KPI2: Safety Zone infractions.
- KPI3: ODD conditions for object detection.

These are shown in Figure 183 and shortly summarized in the following bullets.

- KPI 1: Docking precision
KPI 1 evaluates the docking precision with the semitruck repeatedly starting from the same position. The light condition of the ODD is daylight.
- KPI 2: Safety Zone infractions
For KPI 2 a safety zone is added in which the truck is expected to move. The starting position is expected to vary, and the test evaluates whether the truck manages to stay inside the safety zone. The light condition of the ODD is considered optimal for the sensors, i.e. no functional insufficiencies.

The complexity of the test can be increased by extending the ODD to include worse light conditions and evaluating how the semitruck manages to stay in the safety zone while dimming the light.

- **KPI 3 ODD conditions**

The intention with KPI 3 is to further evaluate variations in ODD conditions by introducing human beings to the test site and to further evaluate the environmental conditions.

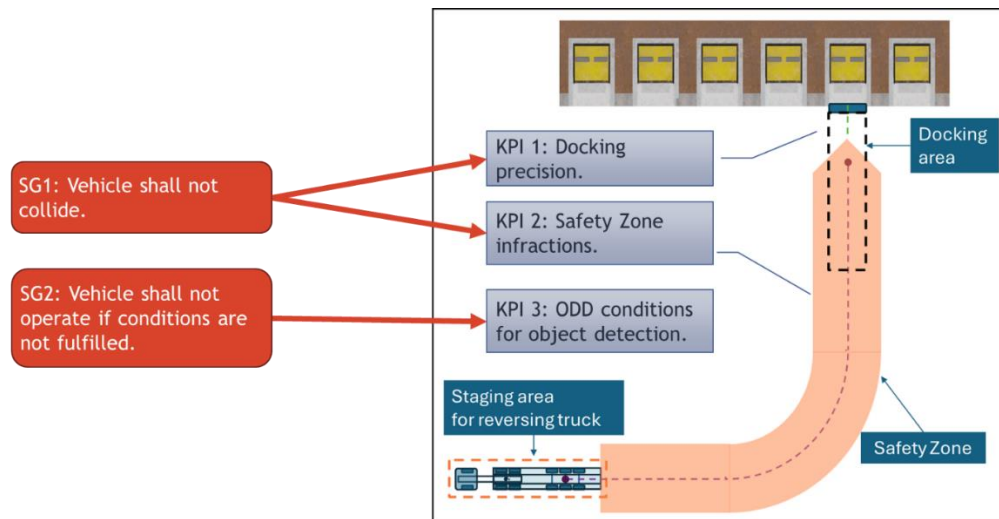


Figure 183: Schematic view of the test set-up connected to the select SPIs.

3.8.3.1 Query and Concretize

In the context of UC4.1 no relevant scenario database was available for the involved partners. Instead, scenarios were collected through experiments with Chalmers Revere's full-size Volvo FH16 "Rhino" with semitrailer. A photo from tests is shown in Figure 184 and recorded GNSS trajectories from the truck performing repeated parking manoeuvres in Figure 185.



Figure 184: Photo from a manual test drive with Chalmers Revere's full-size Volvo FH16 "Rhino" with semitrailer

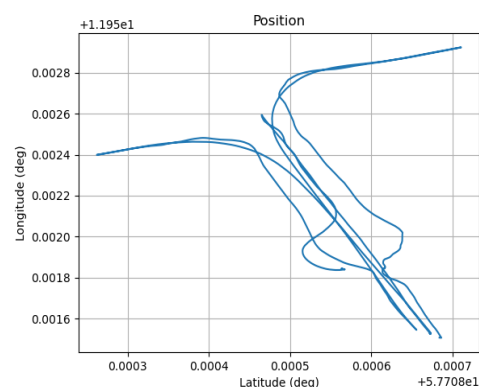


Figure 185: GNSS trajectories from the truck perform repeated parking manoeuvres.

A logical scenario was defined based on the experiments with Chalmers Revere's full-size Volvo FH16 "Rhino" combined with information from an experienced driver. The resulting logical scenario is shown in Figure 186 also showing some possible geometrical parameter distributions.

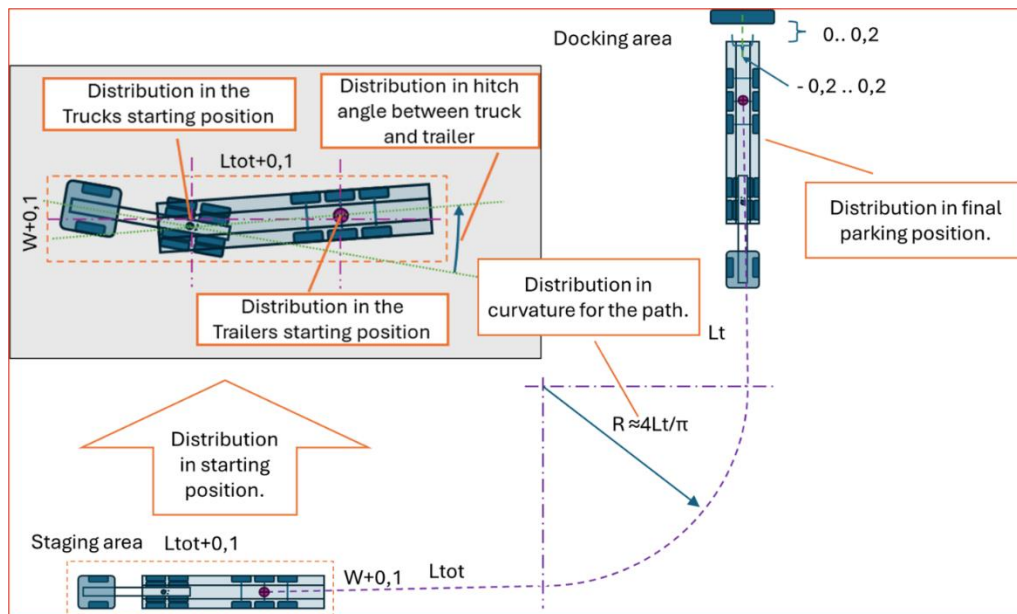


Figure 186: Concluded logical scenario with some possible geometrical distributions.

For concrete scenarios selection the choice was made to focus on the distribution in the starting position of the truck and the trailer. Start position selection are based on a form of combinatorial testing [2] choosing the nominal with the truck in line with the semitruck in the middle of the staging and the extreme positions in the corners of the staging area as shown in Figure 187.

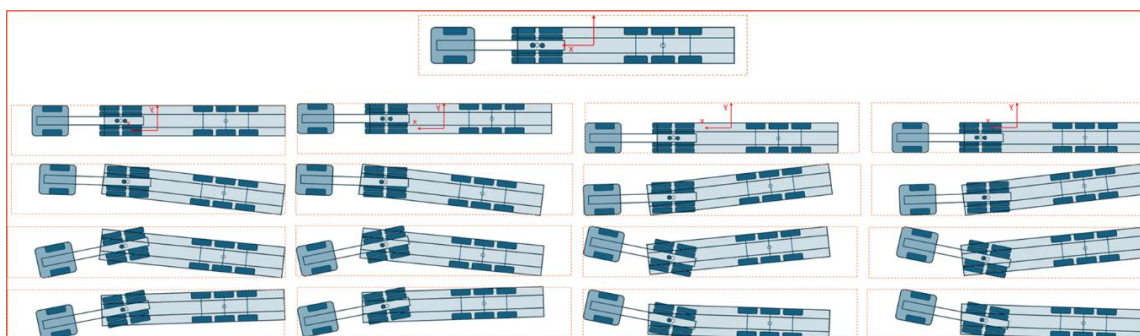


Figure 187: The chosen starting positions for the truck with semitrailer.

3.8.3.2 Allocate

The initial allocation process in SUNRISE has been described in D3.3 [5] and an overview of the process is shown in Figure 188. The three green boxes describe "actions". To support the "compare test case requirements to test capabilities" in the top green box, a method has been developed to evaluate the suitability of test case allocation to test environments by using

a ODD formalization with key testing attributes. The method is denoted “Methodology for Test case Allocation based on a Formalized ODD” (METAFODD) [14].

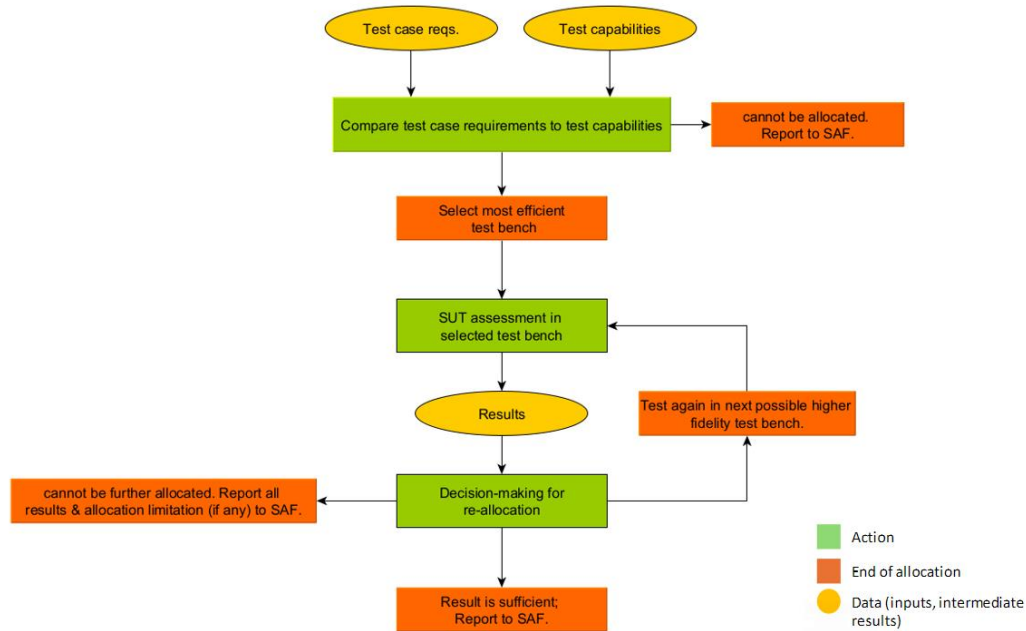


Figure 188: Overview of the initial allocation process [5].

As a first step, an ODD formalizing in a machine-readable format is developed [15] aligning with the taxonomy from ISO 34503 [16]. To formalization uses the Pkl [17] configuration language. The constructed ODD templates are available in [18] with the top module shown in Figure 189. It should be noted that the templates are reusable, and all hierarchies are preserved.

```

1. # Note: Template for ISO34503 ODD
2.
3. @ModuleInfo { minPklVersion = "0.25.1" }
4. module ODD.ODD_template.pkl
5.
6. import "dyn_template.pkl"
7. import "env_template.pkl"
8. import "scen_template.pkl"
9.
10. class odd {
11.     scenery : scen_template.scenery
12.     environment: env_template.env
13.     dynamic : dyn_template.dynamic_elements
14. }

```

Figure 189: Top module for ISO 34503 ODD in Pkl.

Part of the ODD taxonomy following ISO 34503 is illustrated in Figure 190 with the extract of the formalized ODD template for the ODD attribute *Drivable Area* shown in Figure 191. Typed objects are derived from templates, permitting only amendment by overriding existing properties rather than the creation of new ones. The ODD templates are available in [18].

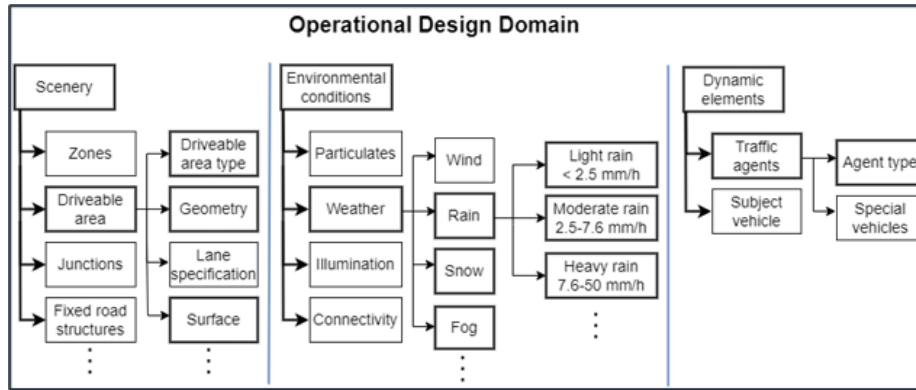


Figure 190: Part of the ODD Taxonomy from ISO 34 503 [16].

```

1. # Note: Excerpt from the file scen_template.pkl
2.
3. const speed_limit_global = 30.0
4.
5. typealias Direction_of_travel = "right_hand_travel" | "left_hand_travel"
6.
7. class Lane_dimensions {
8.     // Define properties related to lane dimensions
9.     lane_dimension : Float (isBetween(2.7, 3.2)) = 2.7 // meters
10. }
11.
12. class Drivable_area_lane_specification {
13.     lane_dimensions: Lane_dimensions
14.     lane_markings: Lane_markings
15.     lane_type: Lane_type
16.     direction_of_travel: Direction_of_travel
17.     speed_limit : Float (isBetween(0, speed_limit_global))
18.     lane_usage : Boolean = true
19.
20. }

```

Figure 191: Example: Definition of drivable area.

Figure 192 shows an example of instating an ODD, odd1, from the template. Note that there are checks that no properties are outside given limits. For visibility, it is also possible export the configuration into a static format like JSON or YAML and convert it into a tree structure diagram with the markup framework PlantUML [19]. An extract of a JSON ODD visualized using PlantUML is shown in Figure 193.


```

# Note: From the file ODD1_test.pkl importing
2. # and configuring ODD_template.pkl
3.
4. import "ODD_template.pkl"
5. odd1 : ODD_template.odd = new {
6.     scenery {
7.         zone {
8.             region_or_state = "Sweden"
9.         }
10.        drivable_area {
11.            drivable_area_lane_specification {
12.                direction_of_travel = "right_hand_travel"
13.                speed_limit = 15.0
14.                lane_usage = true
15.            }
16.        }
17.    }
18. }

```

Figure 192: Instantiating an ODD from the template.

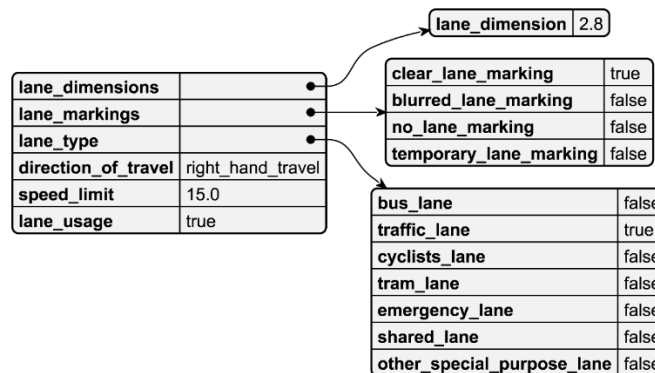


Figure 193: The rendered part of the JSON ODD visualized by PlantUML.

For the automatic test allocation, the ODD Pkl templates are extended with the four additional test environment attributes that are described in SUNRISE D3.3 [5]:

- **Safety Hazard Mitigation Capability:** The capability to minimize potential hazards.
- **Test Complexity Capability:** The degree of complexity involved in testing.
- **Test Environment Fidelity Capability:** The accuracy with which test models replicate real-world conditions.
- **SUT Fidelity Capability:** A metric that assesses the abstraction between a model and its intended production implementation.

These attributes are included in both METAFODD's test case definition, which represents the requirements, and in METAFODD's test environment capabilities description, which represents the provider. Same extended template is used for both test case definition and test environment capabilities description to ensure validation comparability. The four test attributes are subdivided into low, medium, and high levels, reflecting incremental capability. This extension can be represented in Pkl as an addition to the ODD, as illustrated in Figure 194. Low generally indicates minimal emphasis or significant abstraction, medium corresponds to partial coverage or moderate complexity, and high denotes thorough hazard management or near complete fidelity. The constructed templates in Pkl are a in [20].

```

1. #ModuleInfo { minPklVersion = "0.25.1" }
2. module ODD.ODD_template.pkl
3.
4. import "dyn_template.pkl"
5. import "env_sun_ext_template.pkl"
6. import "scen_template.pkl"
7.
8. open class odd {
9.     scenery: scen_template.scenery
10.    environment: env_sun_ext_template.environment
11.    dynamic: dyn_template.dynamic_elements
12. }
13.
14. class ext_odd extends odd {
15.     # 1 Low, 2 Medium , 3 High
16.     Safety_Hazard_Mitigation: Int (isBetween(1,3))
17.     Test_Complexity: Int (isBetween(1,3))
18.     Test_Environment_Fidelity: Int (isBetween(1,3))
19.     SUT_Fidelity: Int (isBetween(1,3))
20. }

```

Figure 194: The Pkl formalized ISO 34503 template extended with four test environment attributes, specified in both test case requirements and environment capabilities for valid comparison.

Figure 195 visualise a subset of the test environment requirement parameters using the template shown in Figure 194 configured as a test environment requirement. The visualisation, which allows easy review of human readers, is done by exporting to YAML format and then converting to graphics using PlantUML [19].

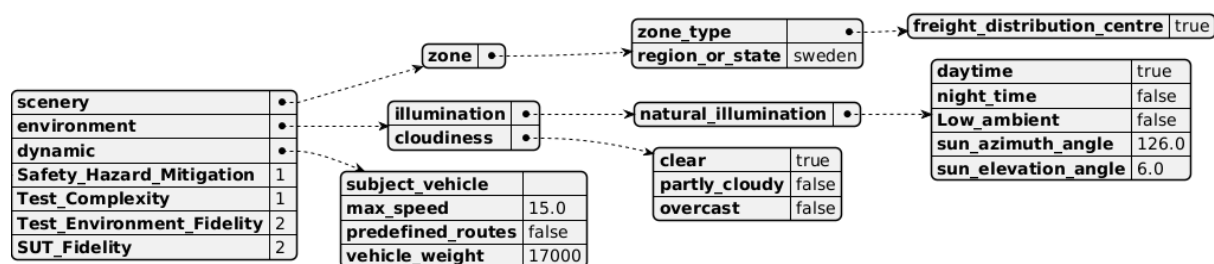


Figure 195: An example showing a subset of the test environment requirement parameters, out of 300 ODD configurable elements.

To exemplify the methodology Figure 196 shows a subset of how configurable weather parameters in Carla to be included in the test environment capabilities. An example of a METAFODD test environment description is shown in Figure 197. It can be seen how the SUT Fidelity is extended with a conditional expression to incorporate a check on sun azimuth angle, ensuring the value lies within $126.0^\circ \pm 10.0^\circ$ corresponding to capture the glare caveat for oblique angles.

```

1. #include <WeatherParameters.h>
2. WeatherParameters (
3. ...
4. float in_cloudiness
5. float in_sun_azimuth_angle,
6. float in_sun_altitude_angle,
7. #Same as sun_elevation_angle in ODD definition
8. ...)

```

Figure 196: Example for inclusions of weather parameters available in CARLA in the Pkl description.

```

1. import "ODD_template.pkl"
2.
3. odd_cap_carla: ODD_template.ext_odd = new {
4.   scenery {
5.     zone {
6.       region_or_state = "sweden"
7.
8.       # Max capability
9.       sun_azimuth_angle = 360.0
10.      sun_elevation_angle = 90.0
11.    }
12.  }
13. }
14.
15. Safety_Hazard_Mitigation = 3
16. Test_Complexity = 3
17. Test_Environment_Fidelity = 2
18. # Glare caveat for oblique angles
19. # When the risk of glare SUT_Fidelity = low
20. SUT_Fidelity = (if (
21.   (odd_req.environment.illumination.natural_illumination
22.     .sun_azimuth_angle >= 116.0)
23.   & & (odd_req.environment.illumination.natural_illumination
24.     .sun_azimuth_angle <= 136.0)
25.   & & (odd_req.environment.illumination.natural_illumination
26.     .sun_elevation_angle <= 10.0)
27. ) 1 else 2)
28. }

```

Figure 197: Example of METAFODD description of the CARLA test environment capability.

A validation method called *genericCompare* is implemented using the reflection property of Pkl [17]. The property enables a program to introspect its own structure and access metadata, including class definitions and their associated methods, fields, and properties. This functionality facilitates the construction of a recursive algorithm that performs a systematic, element-wise comparison of all terminal nodes. Within this framework, string and boolean values are evaluated for strict equality, whereas numeric types—such as integers and floating-point numbers—are compared using either equality or a “less than” relation, depending on the context.

Defined test criteria can now be integrated using the template shown in Figure 194. Comparison of test requirements, as shown in Figure 195, with environment capabilities, such as those in Figure 197 are now enabled using the *genericCompare* function in Figure 198. Integrated, these elements form a prototype methodology for automatically allocating test cases to suitable environments.

```

1. # The ext_ODD_test contains test requirements and test capabilities.
2. # Also a method of generic comparisons that evaluate those conditions.
3. ...
4. Within_CARLAs_Capabilities = genCompare.apply(odd_cap_carla, odd_req)
5. Within_Scaletruck_Capabilities = genCompare.apply(odd_cap_scale, odd_req)
6. ...
7. C: \pk1\ODD_allocate > ./pk1 eval .\ext_ODD_test.pkl
8. ...
9. # Result
10. Within_CARLAs_Capabilites = false
11. Within_Scaletruck_Capabilites = true

```

Figure 198: Automatic allocation evaluation.

In the three last rows in Figure 198, it is written as the result “Within_CARLAs_Capabilites = false” and “Within_Scaletruck_Capabilites = true” indicating that for the given example when the sun azimuth angle lies within $126.0^\circ \pm 10.0^\circ$ simulations using Carla will not handle it while the scale truck will handle it.

3.8.3.3 Execute

No production automated backing function was available for involved SUNRISE partners. Instead, for demonstrating purpose an automated backing function was implemented using the WayWise framework with WayWise [21], WayWiseR [22], and Control Tower [23], the open-source rapid prototyping framework for connected, autonomous vehicles developed by RISE. A schematic overview of the parts of the WayWise framework is shown in Figure 199. An advantage is that the WayWise framework implementation can be used for both virtual testing in Carla and in the 1:14 scale truck used for physical testing.

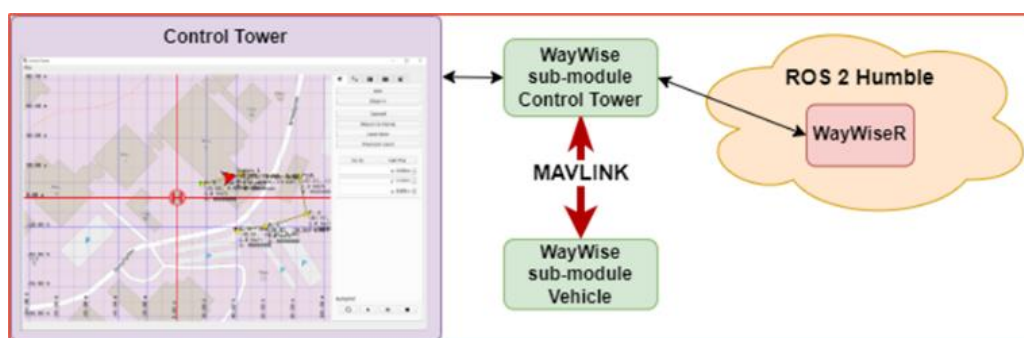


Figure 199: Illustration of the building blocks of the WayWise framework.

As described in SUNRISE D7.2 [4] there is work related to automated or assistant backing of truck–trailer systems in the literature. Commonly they are for low speed, which is reasonable for UC4.1, assuming a simplified linear bicycle model is sufficient, approximating the wheels on an axis to one wheel in the middle and approximating multiple wheel axis to a single axis in the front and back. Further, no slip is assumed. Commonly, the vehicle position control is achieved through feedback of the hitch angle (see Figure 200) based on a linearized system

approximation. Common variants of the pure pursuit algorithm are used [24] for path tracking. For UC4.1, a similar algorithm using the Lyapunov controller [25,26] was identified to be suitable for the backing function.

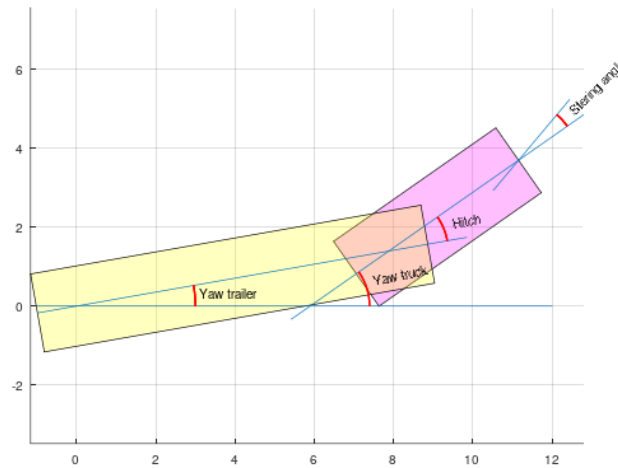


Figure 200: Illustration of a tractor with a connected semitrailer and important angle.

The mathematical model of the backing algorithms together with a simplified kinematic model of the tractor – semitrailer has been implemented using Python. Figure 201 shows the calculated trajectories for the back wheel and the trailer in **blue** respectively back wheel of the truck in **yellow** while following the course in **red**. The **green** circle is the calculated turning radius of the trailer, and the **magenta** circle is the look ahead distance.

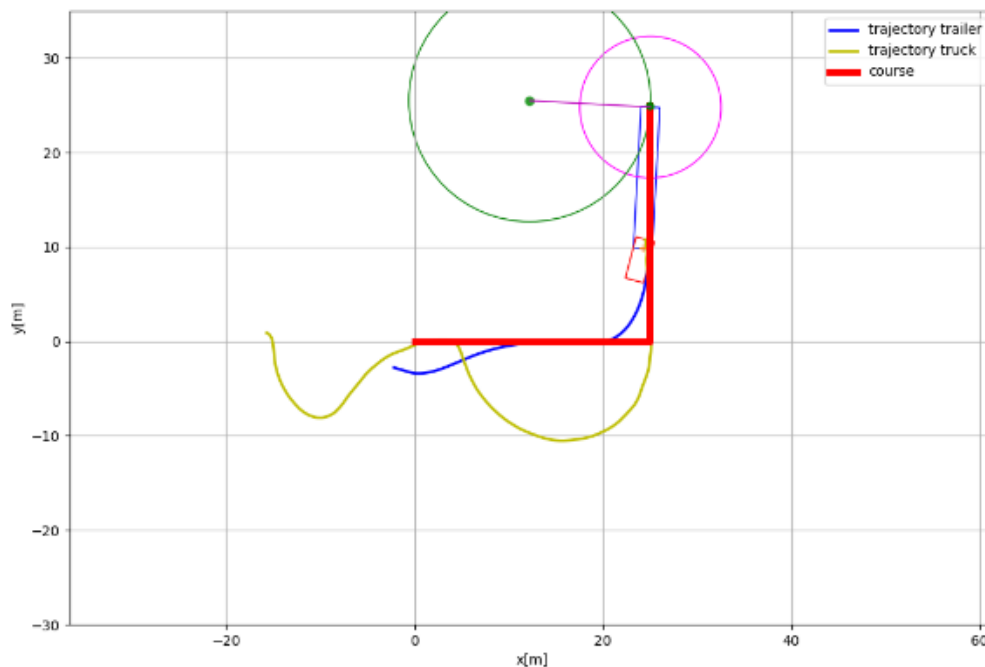


Figure 201: Calculated movements of truck respectively trailer back wheels.

- **Virtual Testing:**

The virtual testing is carried out by RISE using Carla v0.9.15 [27] combined with the WayWise framework. For the bridge between Carla and WayWiseR a fork of the Carla ROS-bridge [28] was used modified to support RISE set-up using ROS2 Humble [29]. A schematic illustration of the simulation set-up to be used is shown in Figure 202. In addition, these are complemented with software functionality for “Test orchestra” and “Data logging”.

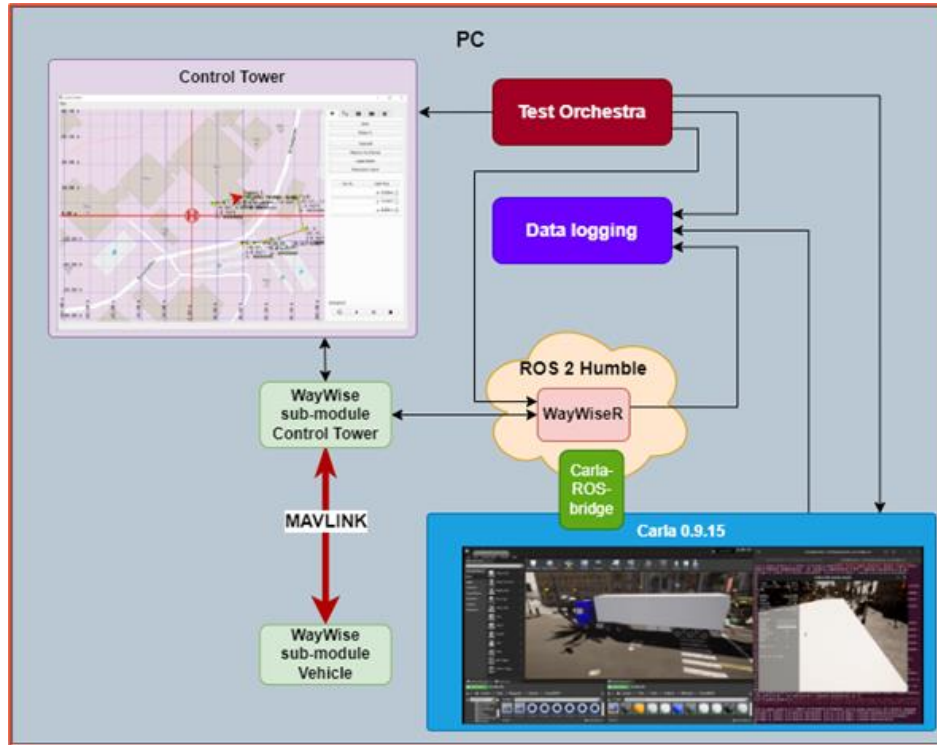


Figure 202: Schematic view of the simulation set-up to be used for sub-UC4.1.

For the simulation in Carla, the environment is reused from one of the included Towns with some modifications. A snapshot of the view with Carla and Rviz in the combined Carla-WayWise simulation is shown in Figure 203.

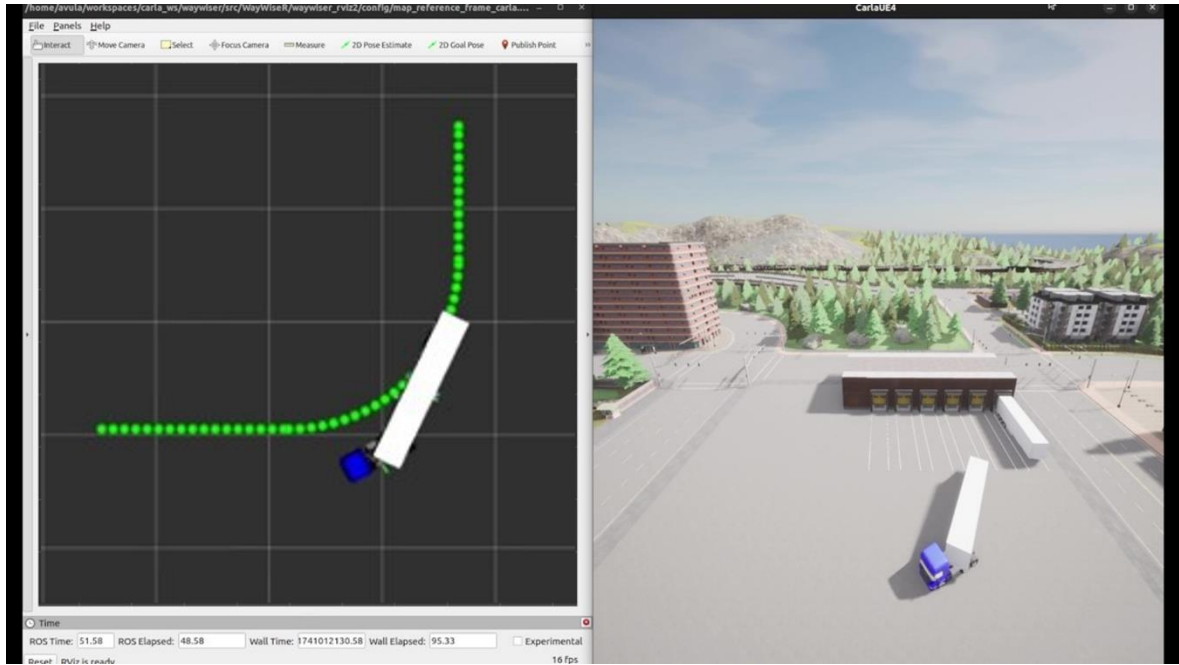


Figure 203: Snapshot from simulation combining Carla and WayWise.

Results from 50 repeated simulations with the truck with semitrailer starting from the middle of the staging area, i.e., the top subfigure in Figure 185, are shown in Figure 204. Additionally, results from 200 simulations with random positions, including the extreme positions from Figure 187 are shown in Figure 205. In both cases the simulated positions follow close to the requested trajectory.

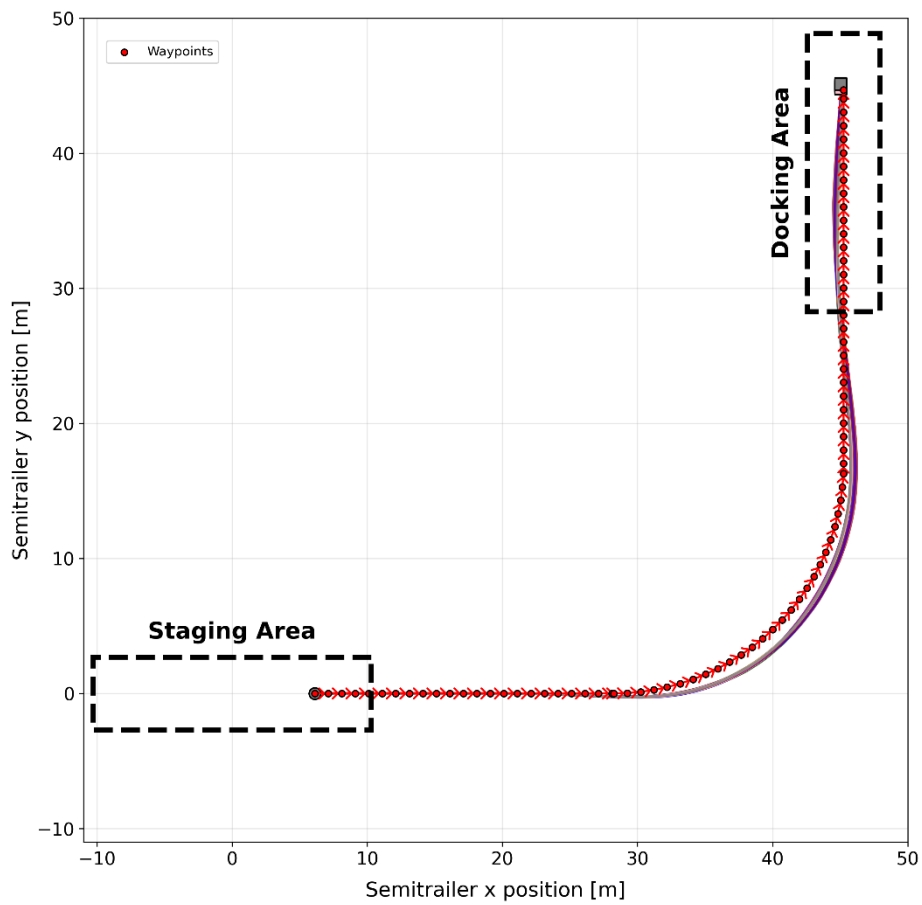


Figure 204: Plots of the rear-axle middle positions of the semitrailer for 50 simulations with the same starting position. The red dots show the requested trajectory and the other coloured lines illustrate the simulated positions.

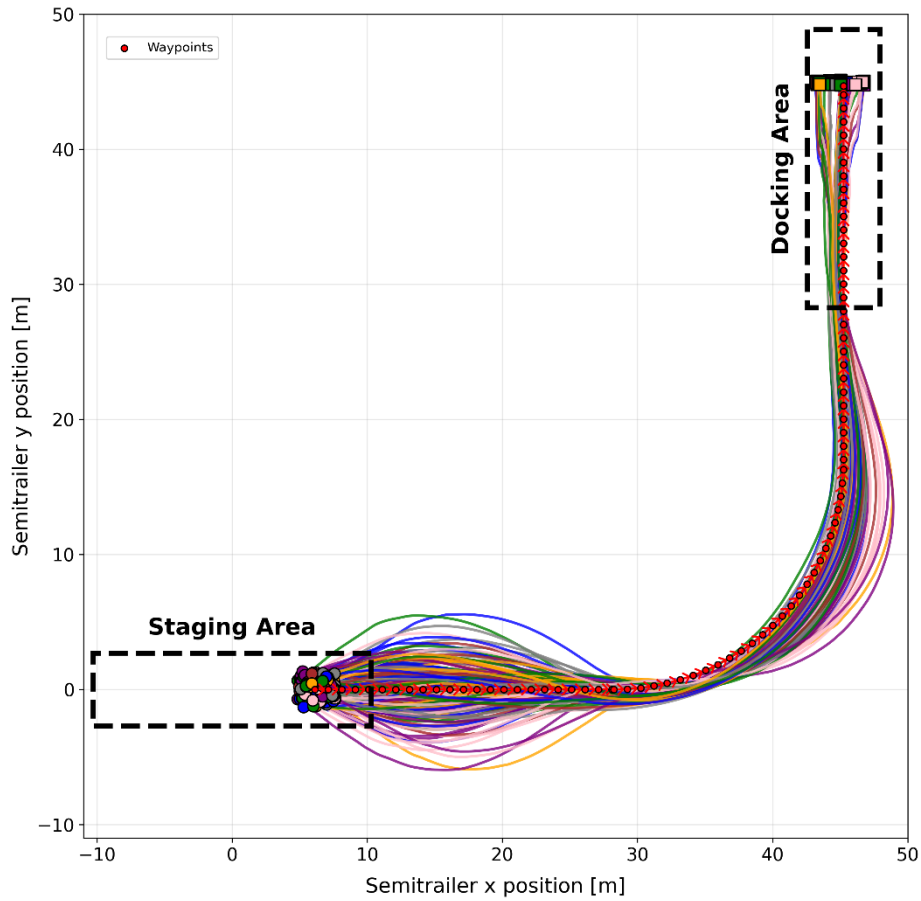


Figure 205: Plots of the rear-axle middle positions of the semitrailer for 200 simulations with random starting positions. The red dots show the requested trajectory and the other coloured lines illustrate the simulated positions.

- **Proving Ground Testing:**

Most of the physical testing is intended to be performed by RISE using a model truck, while a minor part of testing (or demonstration) is done by Chalmers using a full-size truck.

Testing using the RISE model truck

The RISE model truck is based on a Tamiya Scania R620 model truck in scale 1:14 combined with a Tamiya semitrailer model on the same scale. A photo of the truck with semitrailer is shown in Figure 206.



Figure 206: Photo of the 1:14 model truck with a semitrailer. On the roof of the truck is the GNSS antenna.

The model has been modified to use the WayWise [21] prototyping library to control the truck. A schematic view of the architecture is shown in Figure 207, and the intention is to minimize the differences in the control software between the simulation and the physical tests. Due to limited space and capacity in the onboard computer, parts of the functionality are placed in the Control Tower operated on a PC.

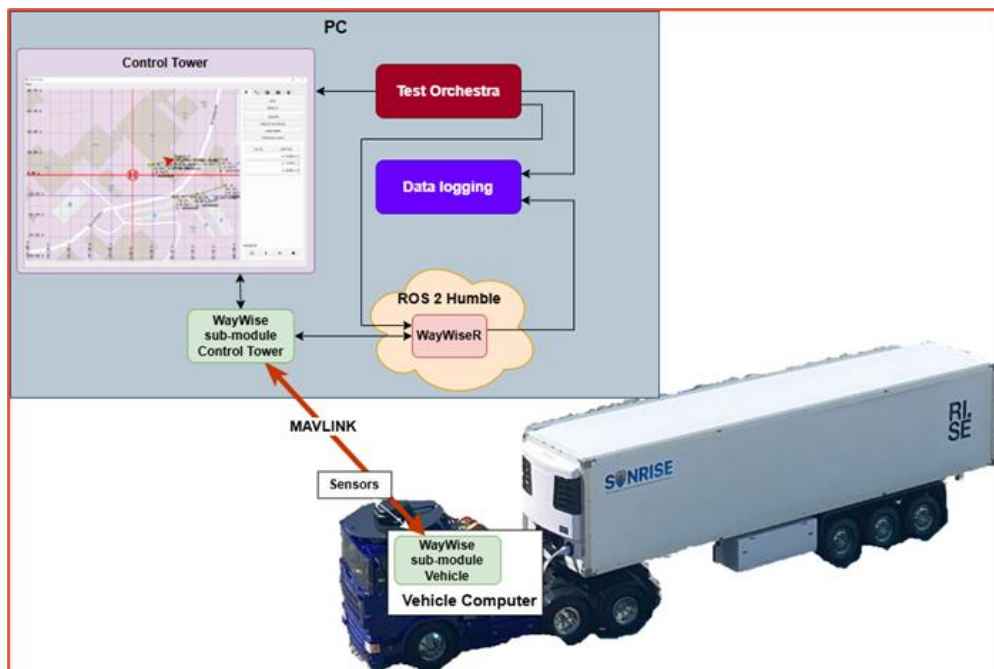


Figure 207: Schematic view of the software set-up with RISE model truck.

A photo of the test setup for the model truck is shown in Figure 208. The location is at the RISE area in Borås. The truck is parked close to the scenery used as the wall of the logistic

hub. The green rectangle is the staging area. On the right side is a GoPro camera used for video recording seen. Not seen in the picture is the laptop used to control the truck.



Figure 208: Photo of the physical test setup for running the model truck.

The photo in Figure 208 is complemented with the information in Figure 209 illustrating the staging area, safety zone, and docking area.

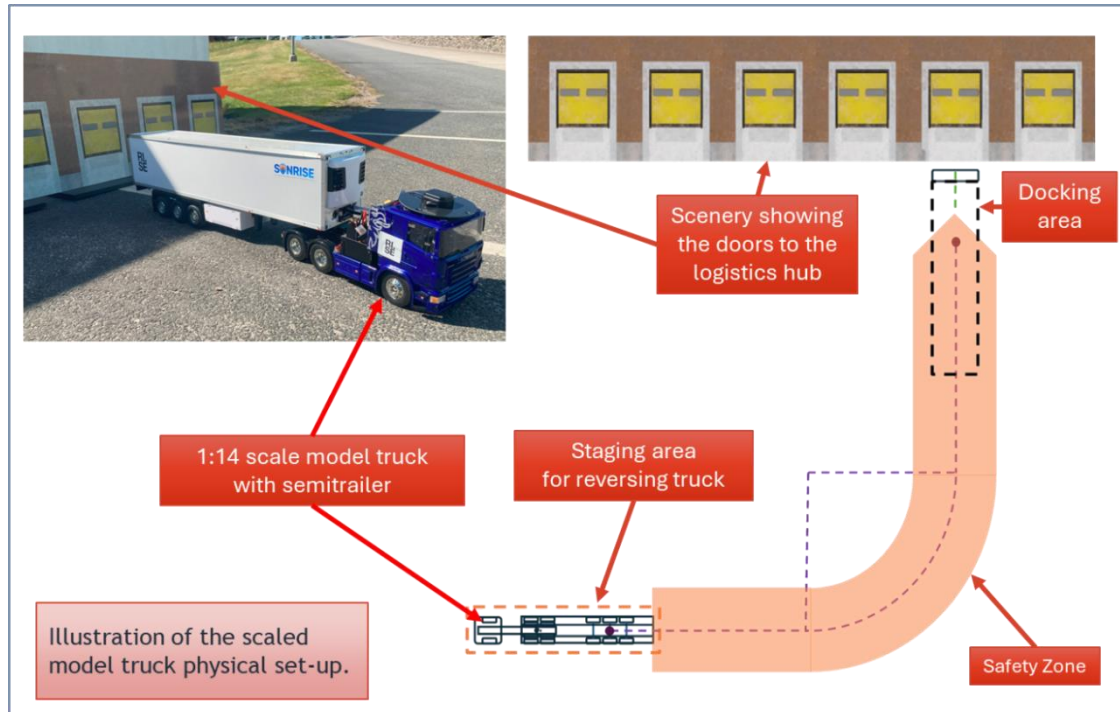


Figure 209: Schematic view of the important parts for the model truck physical test set-up.

Test were run corresponding to all the concrete scenarios defined by the positions in the staging area as illustrated in Figure 187. After starting the test, the truck was monitored from the computer running the Control Tower. An example of visualization of the monitored data is shown in Figure 210. All relevant data are logged during the tests.

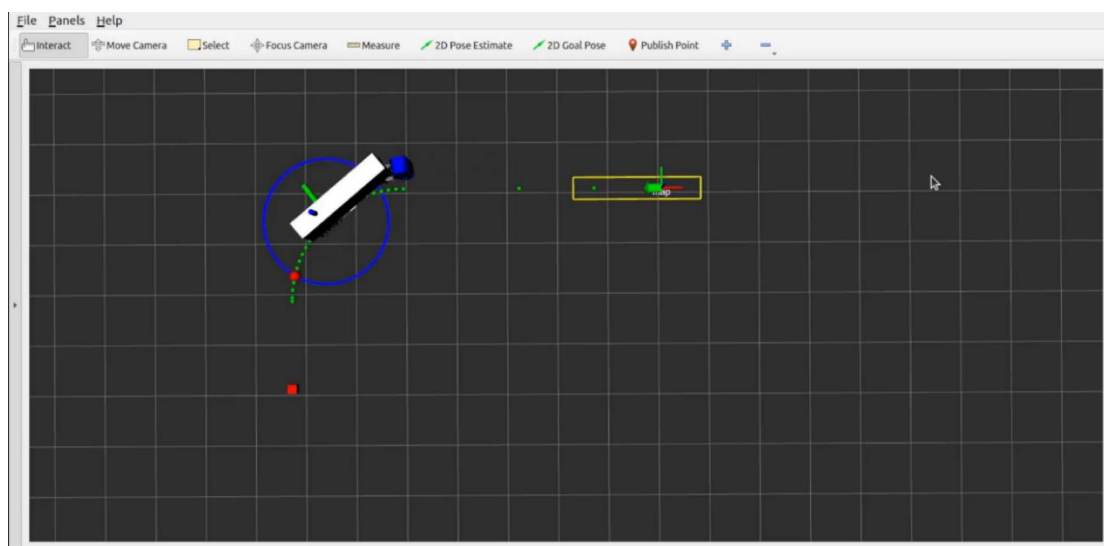


Figure 210: Graph showing the monitored truck during the reverse driving operation.

The photo in Figure 211 shows the model truck parked in the staging area corresponding to one of the starting positions shown in Figure 187. An important aspect to take into account is that the truck relies on the GNSS position for navigation along the intended trajectory. To not lost GNSS accuracy it is necessary to drive the truck into wanted position in the staging area and afterwards lift the trailer to correct position. If the truck is moved by lifting it the GNSS

position accuracy is lost. Unfortunately, there are limitations in how precise the truck can be parked which will have impact on the repeatability and accuracy of the tests with the model truck.

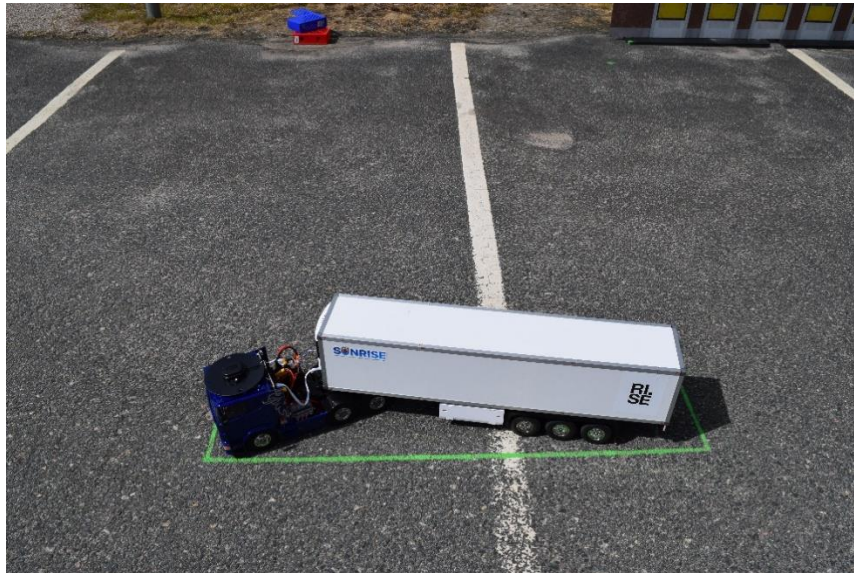


Figure 211: The model truck with semitrailer positioned in the staging area.

Tests were run corresponding to KPI 1 and KPI 2:

- Figure 212 corresponds to KPI 1 showing 10 repeated tests with the truck starting from the middle of the staging area. It can be seen that the intended trajectory is followed with rather small variations, especially considering the uncertainty in the starting position. Some shift is seen to the left in the docking area.
- Figure 213 shows 17 tests with starting positions in the staging area according to Figure 187. In this case much more variations are seen in the resulting trajectories though it partly may be due to inaccurate starting positions. In addition, it may be due to lost GNSS accuracy during the tests as that sometimes was seen.

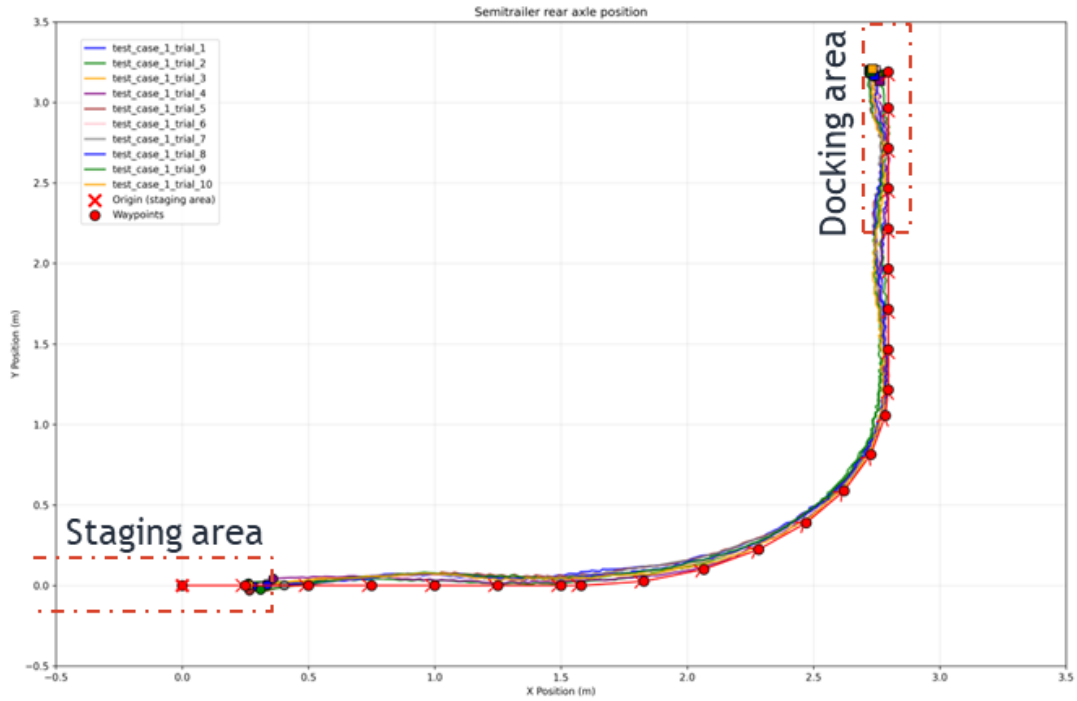


Figure 212: Ten repeated tests run with the model truck starting from the middle of the staging area

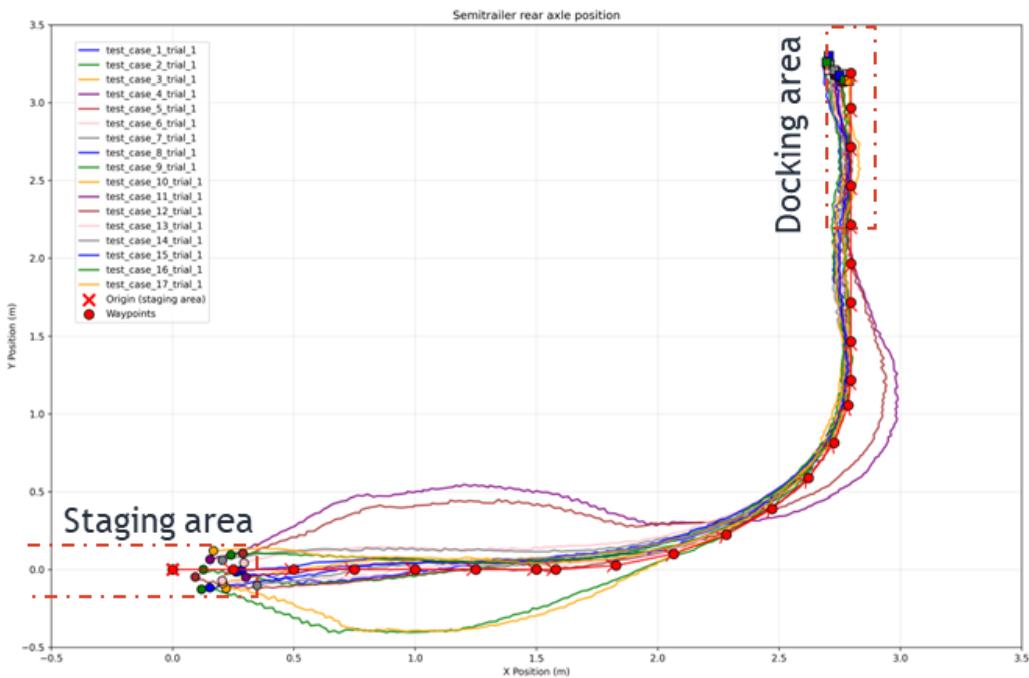


Figure 213: Test runs with the model truck with starting positions according to Figure 187.

Testing using Chalmers full-size truck

Chalmers has access to a Volvo truck and a semitrailer that will be used for SUNRISE. However, it has only been feasible to do a few tests runs in the scope of the SUNRISE project. Further, as no production-ready backing function is available in the Chalmers truck, the tests were performed using a human driver. A large effort is needed to adapt the backing function to the full-size truck, and the economic and safety risks are also

considered too high to use the demonstrator backing function developed for the RISE model truck. A photo from a test with Chalmers Volvo FH16 “Rhino” is shown in Figure 214.



Figure 214: Photo from a test run with Chalmers Volvo FH16 “Rhino”.

- **Public Road Testing:**

No public road testing was done as part of SUNRISE UC4.1.

3.8.3.4 Test Evaluate

The validity of each test is examined versus logs and the relevance of the results to the defined key performance indicators.

- KPI 1: Docking precision:

KPI 1 evaluates the docking precision with the semitruck repeatedly starting from the same position. The light condition of the ODD is daylight.

In Figure 205, the rear-end position for the semitrailer is shown for 50 simulation runs from the middle of the staging area with a full-size truck in Carla. Low distribution between is seen between the different runs.

Figure 215 to Figure 217 show 10 runs with the scaled model truck with start in the middle of the staging area:

- Figure 215 shows the final position for the rear axis in the docking area. A distribution is seen corresponding to an offset of about 8 cm. A major contributor may be the uncertainty in the RTK-GNSS position.
- Figure 216 and Figure 217 show the distance measured using ultrasonic sensors to the wall behind the trailer and to a semitrailer parked alongside the docking area. The distribution is a few centimetres and may be due to the uncertainty in RTK-GNSS positioning used by the model truck.

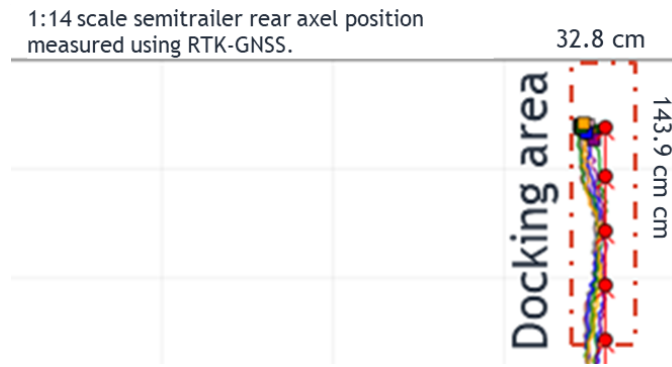


Figure 215: The final position for the rear axis in the docking area for 10 runs with the 1:14 scale truck.

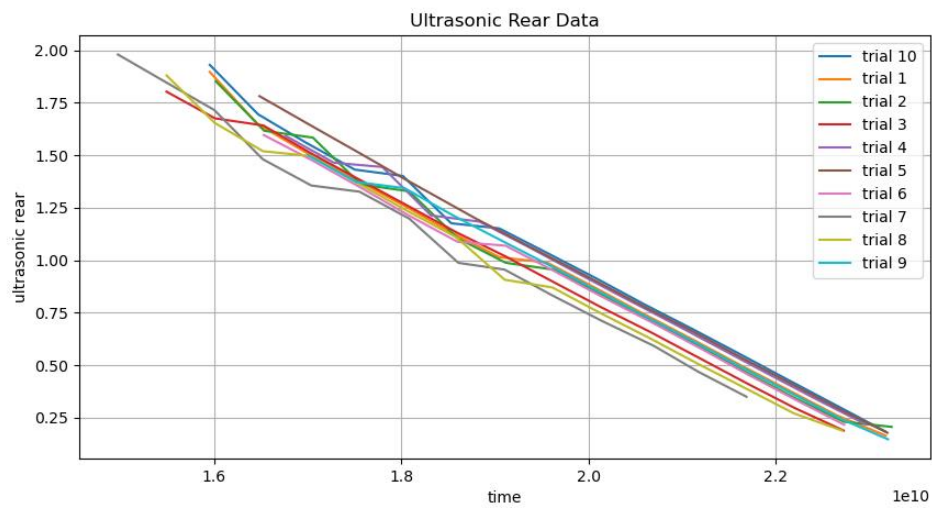


Figure 216: The distance measured by using ultrasonic sensors to the wall behind the trailer for 10 runs with the 1:14 scale truck from the same starting position.

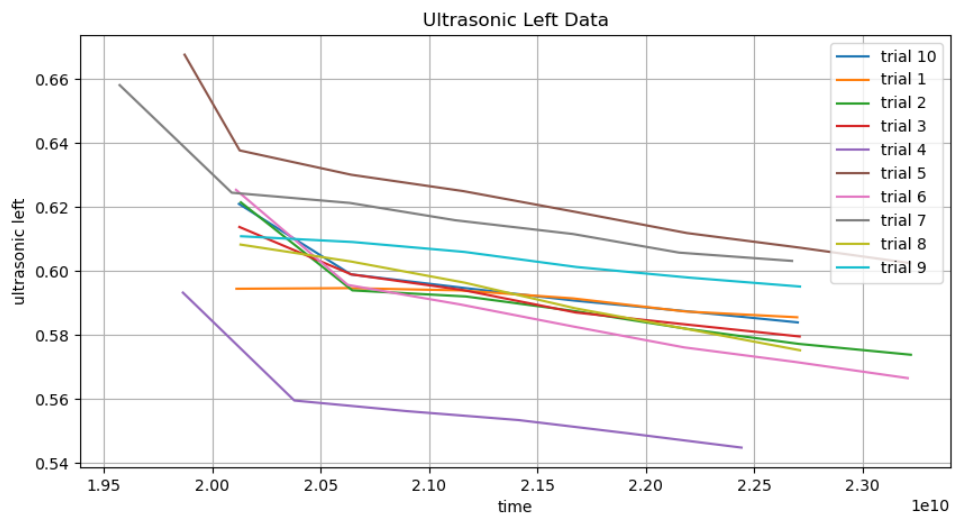


Figure 217: The distance measured using ultrasonic sensors to a semitrailer parked alongside the docking area trailer for 10 runs with the 1:14 scale truck from the same starting position.

- KPI 2: Safety Zone infractions

For KPI 2, a safety zone is added inside which the truck with semitrailer is expected to stay during the reverse parking manoeuvre. The starting position may be anywhere inside the staging area and the test evaluates whether the truck manages to stay inside the safety zone or not. The light condition of the ODD is considered optimal for the sensors, i.e., no functional insufficiencies. The complexity of the test can be increased if the ODD is extending to include worse light conditions, e.g., while dimming the light.

Figure 218 shows a Monte-Carlo simulation of the mathematic model implemented in python. This indicates that the truck should follow the intended path rather well even for a large staging area.

However, the tests shown in Figure 213 with the scaled truck on the right shows rather large deviation for some of the tests that should have infringed the safety zone. The uncertainty in the RTK-GNSS position may be a major contributor as it sometime has reduced accuracy that is significant in for the scale truck.

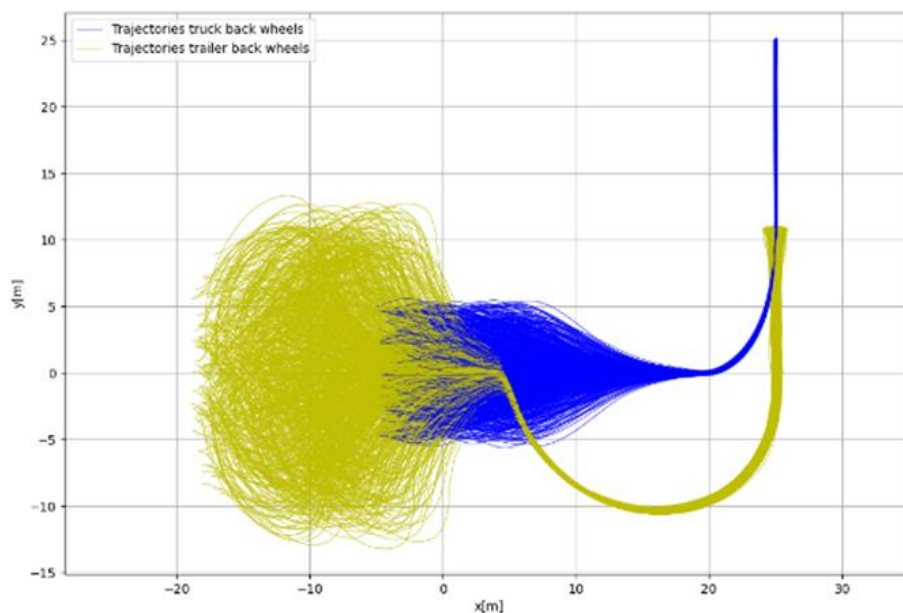


Figure 218: Monte-Carlo simulation of the trajectories for the rear wheel axis of the truck respectively semitrailer using the Python model. Rather large limits for the staging area is assumed.

- KPI 3 ODD conditions for object detection

The intention with KPI 3 is to further evaluate variations in ODD conditions by introducing human beings to the test site and to further evaluate the environmental conditions.

For KPI 3 the tests have been limited to verifying the previous mentioned example with glare from the METAFODD description (see Figure 219 and Figure 220).

- Figure 219 shows illustrate how glare looks on a physical tests site.

- Figure 220 shows a similar view in Carla's UE4 environment.
- The evaluation shows that in this situation glare must be tested in a high-fidelity environment and that the available Carla based simulation environment cannot produce reliable results.



Figure 219: A test scenario with sun elevation of 9° in a hardware-in-the-loop scale truck environment.

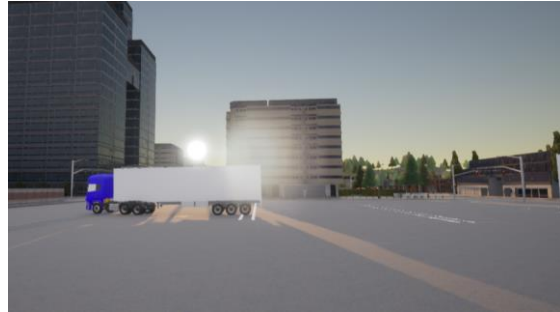


Figure 220: A test scenario with sun elevation of 6° in Carla simulator UE4 environment

3.8.3.5 Safety Case

The safety argument or assurance contribution block was addressed by implementing and evaluating safety performance indicators (SPIs) to assess the system's ability to achieve relevant safety objectives in a confined logistic environment.

- SPI 1: Docking precision (= KPI 1)
This SPI assessed whether the implementation can consistently achieve adequate docking accuracy, supporting the claim that the system performance can reach an acceptable threshold for safe operation under standard conditions.
- SPI 2: Safety Zone infractions (= KPI 2)
This SPI assess whether the implementation can safely manoeuvre inside the safety zone, supporting the claim that the system performance can reach an acceptable threshold for safe operation under standard conditions.
- SPI 3: Structured ODD expansion (based on KPI 3)
The process for test space expansion was structured to introduce additional environmental parameters, such as lighting and the presence of humans, in a controlled manner. This supported the evaluation of how environmental variation influences both actuation and perception functions.
- SPI 4: Test space coverage
The validation approach ensured that the test set-up could address all conceivable arrangements and starting positions within the staging area, indicating that the methodology is capable of systematically exploring the relevant test space.

- SPI 5: Toolchain flexibility

The chosen methodology allowed for the stepwise adaptation of validation activities as new complexity factors were introduced, enabling a measured approach to increasing scenario difficulty.

3.8.3.6 Decide

The decision criteria (DC) can be applied after the test-completing criteria (CC) are fulfilled as shown in Table 28:

- KPI 1 DC is that the truck with semitrailer stops in the defined docking area
- CC some proof (a combination from different environments) that all valid initial conditions permuted over all valid ODD parameters conform to this.
- KPI 2 no safety zone infractions for the same test space as above
- KPI 3 Object detection performance is above the acceptable threshold, e.g., >99%, over all valid object and ODD parameters. Or the decomposed performance requirement placed on the sensor in a multiple-sensor solution.

In addition, sufficient test space coverage according to SPI 4 must have been achieved.

Table 28: KPIs for different test environments

Key Performance Indicator	Unit	Target	Simulation	Proving Ground Scale truck	Real World
KPI 1: Docking precision	m	Max. ?	x	x	
KPI 2: Safety Zone infractions	#	0	X	x	
KPI 3: ODD conditions for object detection	lx	2000 lx	X	x	

3.8.4 Key take aways

- UC4.1 successfully demonstrates a practical application of the SAF, defining and applying clear safety test objectives.
- Successful validation of Environment and Safety Argument Blocks in both simulation and physical testing.
- Supported the construction and assessment of a safety case.

3.8.5 Deviations from D7.2

The deviations from SUNRISE deliverable D7.2 [4] primarily concern the demonstration plans which sometimes have needed to be simplified. In addition, the KPI 3 evaluation has been purely theoretical without physical experiments.

3.9 Use case 4.2: Freight vehicle automated parking validation – Truck low-speed connected perception cybersecurity

3.9.1 Use Case Overview

In Use Case 4.2, the SUNRISE SAF is tested for truck low-speed connected perception testing under cyber-attack. The SUT is a connected perception AD subsystem that is compromised by cyber-security threats. The main aim is to combine in a simulation environment several aspects simultaneously (physical environment, perception, V2X connectivity, Collective Perception Message, CPM, compromise by cyber-attack) and study the effects of physical or remotely executed cyber threats on collective environment awareness.

There are two functional scenarios defined from T7.1 for assessing the connected perception cyber-security performance in use case 4.2. The first is the “distorted camera input” scenario, whereas the second one is the “CPM message attacked” scenario.

UC4.2 builds on top of the CCAM system of UC4.1 and the perception system of UC1.3 and adds the feature of a compromised CPM message by a remote cyber-attacker. The main objective is to combine in virtual simulation several aspects simultaneously (environment, perception, V2X connectivity, cyber-attacks) and study the effects of a remotely executed cyber-attack on collective environment awareness, in this case a CPM message exchanged between a V2I node installed on an infrastructure camera and the truck, as shown in Figure 221.

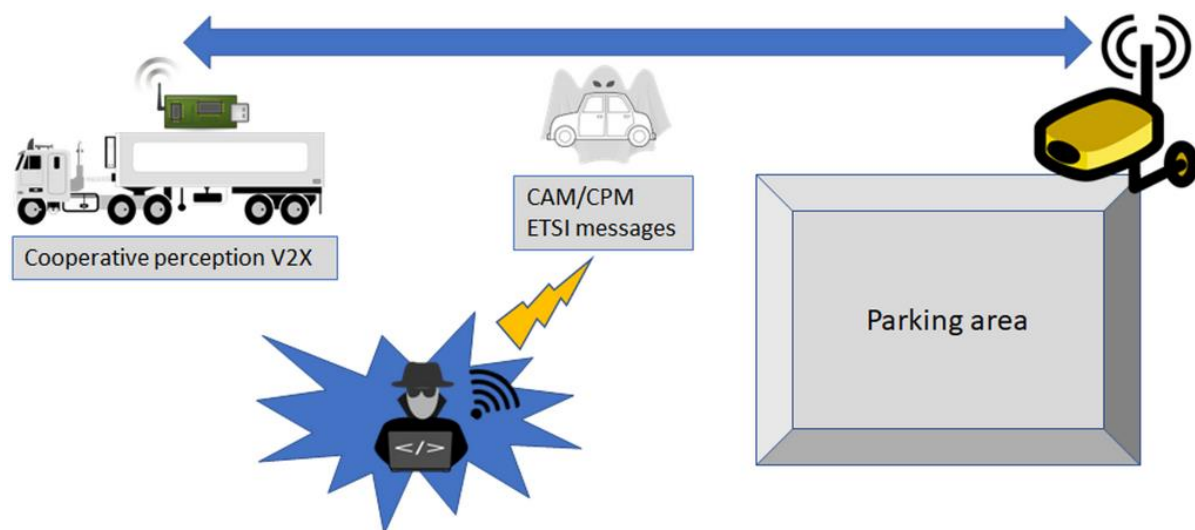


Figure 221: UC 4.2 functional representation

Objective :

In virtual simulation, combine several aspects simultaneously (environment, perception, V2X connectivity, cyber- attacks) and study the effects of remotely executed cyber attacks on collective environment awareness.

3.9.1.1 Covered aspects of the SAF

Three sub-blocks of the SAF contributing to the safety case building have been applied and demonstrated as presented in Figure 222, namely: the scenario Concretization, the simulation framework Execution as well as the test outcome Evaluation through dedicated safety KPIs. Figure 222 below shows the contributions to the SAF block validation per partner for this UC.

Table 29: Contributions to the SAF validation per Partner in UC4.2

Partner/ Block	ICCS	RISE
SUNRISE DF		
Query & Concretise	x	
Allocate		
Execute	x	x
Test Evaluate	x	x
Coverage		
Safety Case		
Decide		

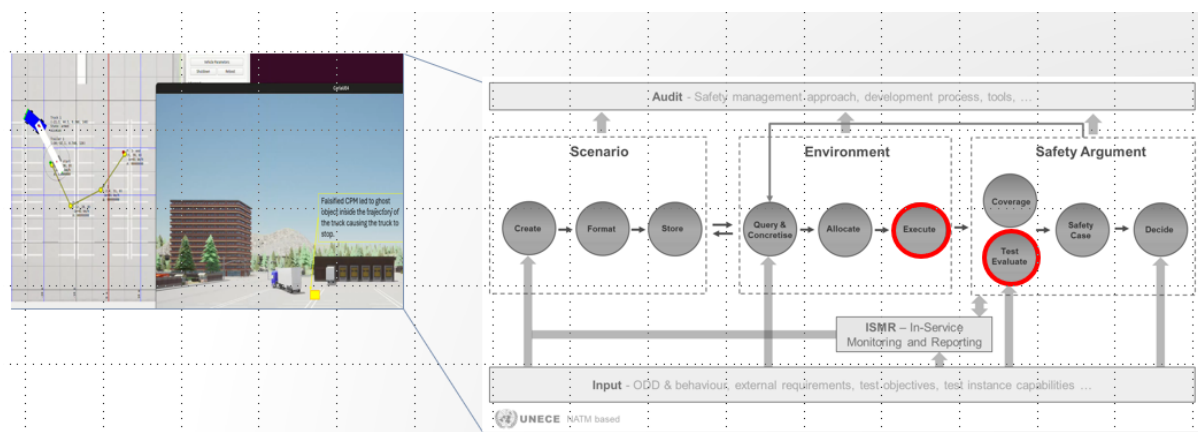


Figure 222: SAF demonstrated blocks in this UC's demo (s).

Tutorial video demonstrating the SAF applicability for this UC will be uploaded at the SUNRISE Handbook [webpage](#) before the end of the project.

3.9.1.2 Safety case setup

Two partners spoof the Road Side Unit (RSU) camera and falsify Cooperative Perception Messages (CPMs) while controlling a virtual truck in a parking manoeuvre in a CARLA-ns3 co- simulation environment. A virtual camera-based RSU module is assumed to provide information about the scene through CPMs (Collective Perception Message as standardized by ETSI). Considered pass/fail criteria are similar to the truck parking system of UC4.1 however additional KPIs related to the effects of the cyber attack have been defined as described in sec.3.9.3.3.

3.9.2 Overview of tested scenarios

The test scenario was one of the reverse parking scenario applied in UC4.1 augmented by the event of a falsified CPM reception by the truck leading to interruption of the parking maneuver.

3.9.3 SAF Block demonstrations

3.9.3.1 Query and concretize

The concrete scenario was hand-crafted using one of the scenarios applied in UC4.1 testing. An extra triggering condition was added as part of the network-CARLA co-simulation to emulate the cyber attack through the creation of a falsified CPM message containing a 'ghost' object through the following steps:

- A predefined route is assigned to the truck-and-semitrailer using RI.SE's Control Tower module;
- As the vehicle approaches the docking station, the ghost object function is triggered; the RSU sends a CPM indicating the sudden presence of a vehicle obstructing the path.

This UC did not focus on covering multiple test scenarios as this is already addressed by the work in UC1.3 (Collective Perception testing).

3.9.3.2 Execute

- Virtual Testing

A driving and network CARLA-ns3 co-simulation environment was developed for testing as presented in Figure 223. For this purpose, the open source msvan3t framework⁴ used in UC 1.3 was adapted by ICCS through the following steps:

- Extended the ICCS pipeline to enable virtual RSU camera deployment and support their communication via CPMs;
- Added and attached LiDAR sensors to both RSUs and the semitrailer to provide localization support for CPM detected objects;
- ROS2 integration in msvan3t framework in order to enable emergency stop when needed (ghost object incident);
- Carla ROS bridge acts as tick master > ROS bridge code adjusted to wait for update of the Local Dynamic Map (LDM) updates before giving each tick;
- Implemented ghost object injection for a predefined number of simulation ticks.

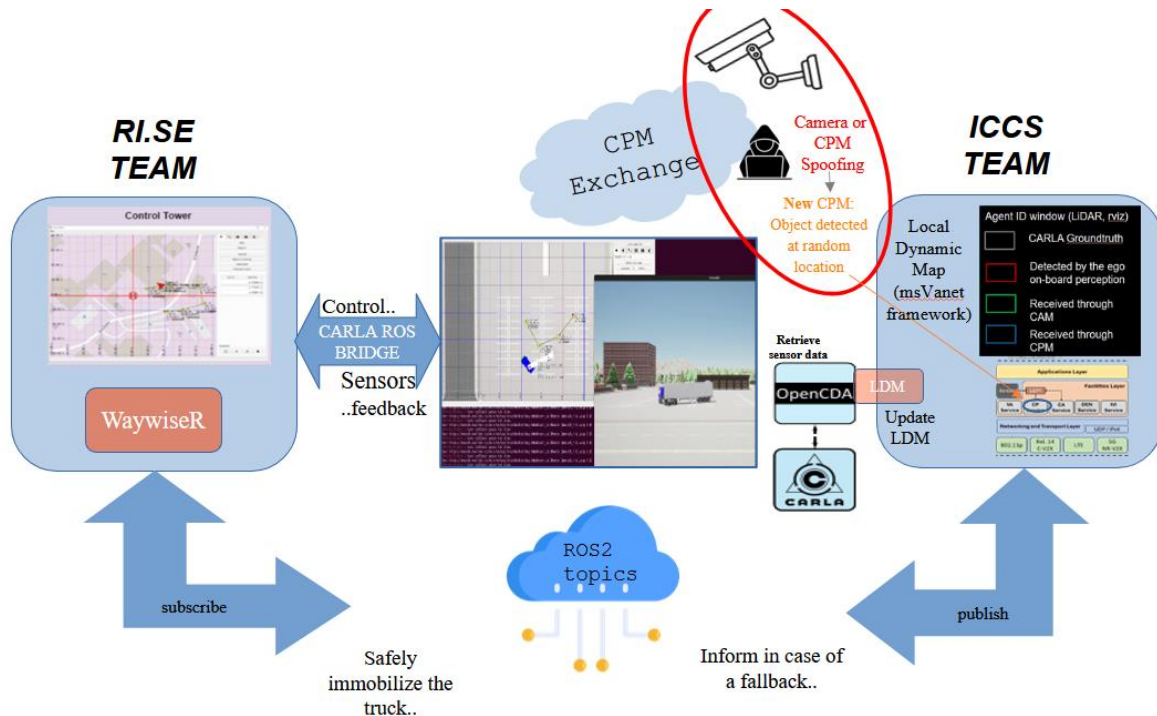


Figure 223: Execute' block functional architecture

Running the CARLA-V2X simulation includes the following steps:

- 1) Simulation begins with CARLA ROS bridge spawning the truck-and-semitrailer and its associated sensors, enabling full control within the CARLA world
- 2) A virtual RSU is spawned via the ICCS pipeline, and msvan3t's Connectivity Manager is applied to it

⁴ <https://ieeexplore.ieee.org/document/10449533>

- 3) The msvan3t framework initiates, allowing CPM and CAM message exchange between the RSU and the truck-and-semitrailer
- 4) A predefined route is assigned to the truck-and-semitrailer using RI.SE's Control Tower module
- 5) As the vehicle approaches the dock station, the ghost object function is triggered; the RSU sends a CPM indicating the sudden presence of a vehicle obstructing the path
- 6) An emergency stop is triggered, leading to the immobilization of the truck-and-semitrailer and ensuring compliance with Safety Goal 1 (UC4.1)

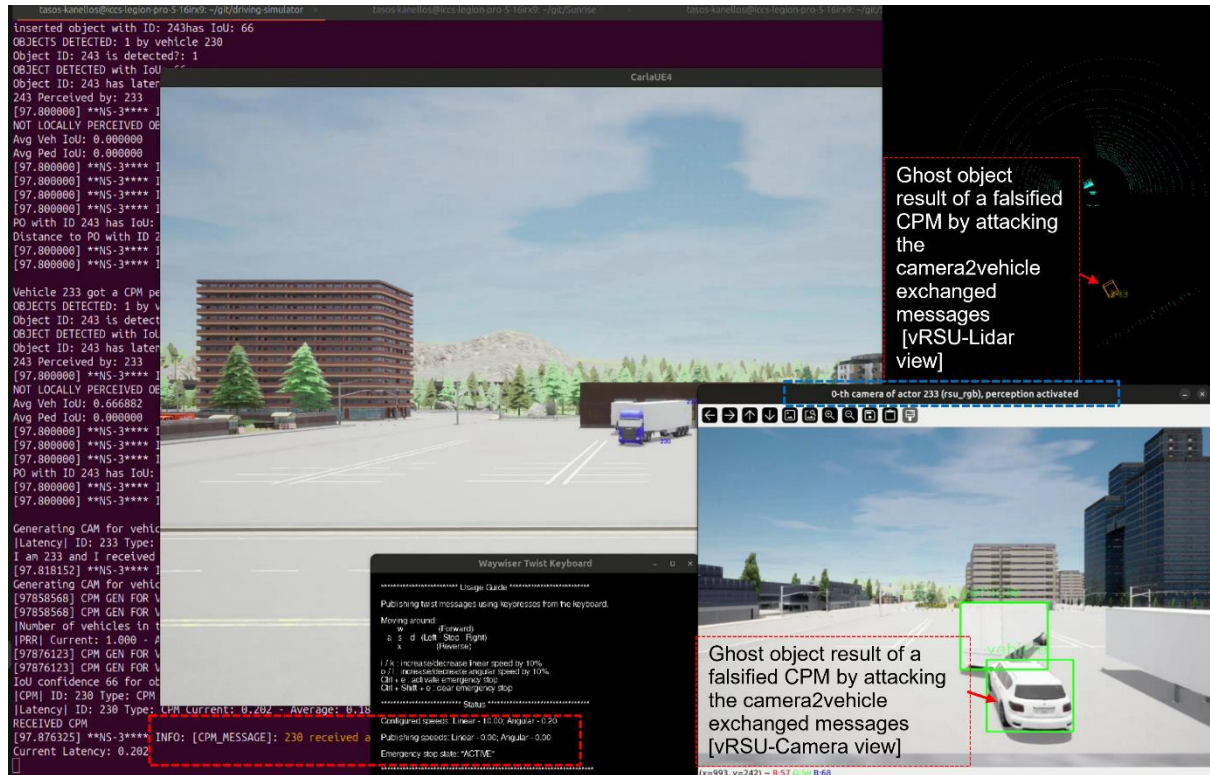


Figure 224: Execute block snapshot showing the CARLA-msvan3t console hosting information about CPM exchange (purple window), the virtual RSU camera and lidar perception (annotated windows on the right side of the picture), the CARLA birds-eye-view in the center and the Waywiser GUI hosting information on (black window on the bottom)

Note: Although in this SAF demonstration we mainly focused on the execute part and test evaluate part and not on scenario space coverage, CARLA-ns3 co-simulation framework as implemented by this UC, allows to parameterize network/message exchange characteristics⁵ and hence creating variations of the same concrete scenario for different CAM/CPM propagation loss rates among any connected agents. Therefore, the work performed here (CARLA-V2X co-simulation) is considered an important step towards studying in simulation these types of scenarios and we believe it will be a valuable tool for TARA experts, enabling them to build, visualise, and analyse the attacker's behaviour and operations to better understand the potential impacts of cyberattacks. This will in turn allow for quantification of different qualitative parameters set currently by experts when performing TARA analysis making attack feasibility assessment more objective and not solely based on experts' opinions and experience.

- Proving Ground Testing: N/a.

⁵<https://www.nsnam.org/docs/models/html/propagation.html?highlight=threegppv2urbanpropagationlossmodel#vehicular-channel-condition-models>

- Public Road Testing: N/a.

3.9.3.3 Test Evaluate

The result of the 'execute' block, focusing on the cyber-attack effects, have been quantified through two network-related metrics and two new security-related KPIs, namely:

- Average latency and packet receipt ratio (CAM,CPM) for all agents in the scene and per agent (network related metrics);
- Time offset between the CPM message reception and the immobilization of the truck-and-semitrailer (Time to Full Stop);
- Latency of the CPM message (measured in ideal conditions where the Propagation Delay model depends only on the distance between the source and the receiver excluding any interference effects, assuming a fixed propagation speed equal to the speed of light).

```
Ego vehicle has stopped. Elapsed time: 1.555 seconds

CPM sent with latency: 0.202 milliseconds

Initiating graceful shutdown: OpenCDAClient::GetManagedActorById() failed with error: Socket closed
[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average PRR: 0.994083

[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average latency (ms): 0.185089

[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average latency of vehicle 230 (ms): 0.183862

[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average PRR of vehicle 230 (%): 0.995633

[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average latency of vehicle 233 (ms): 0.18768

[0.000000] **NS-3**** INFO: [V2V_CONFIG]: Average PRR of vehicle 233 (%): 0.990826
```

Figure 225: Test evaluate block snapshot showing network and security-related KPIs applying specially on this UC

The results from this block can be used by a hypothetical subsequent SAF component performing safety case building based on quantifying the window of opportunity of an attacker that uses short-range wireless communications to attack similarly to the recent work of [30].

Note: In this UC a real-world attack is assumed but without directly attacking the vehicle under test but the infrastructure or the V2X message. It is acknowledged that these kinds of attacks are neither solely a cybersecurity nor a SOTIF topic. This should be considered as a cross-domain safety/security topic that requires a combined threat and risk analysis by the AV/CCAM engineers. The risk stemming from objects reported by external sources is considered by the recent cybersecurity standards. It is important that mitigations are evaluated, both inside and outside the vehicle [31]. CCAM systems cybersecurity extends to systems outside the CAV and hence it is a more complex task since typical cybersecurity measures like integrity checks or proofing the authenticity as the attacked environment is not cyber-physically connected to the considered system.

3.9.4 Key take aways

‘Execute’ sub-block:

- New scenario execution mechanism to support cyber-attack triggering event
- Co-simulation setup integrating virtual RSU and CPM spoofing made possible in CARLA:
 - a) custom camera sensor spoofing (exploiting light mechanisms offered by CARLA simulator) to interfere with the quality of the raw sensor data
 - b) ghost object spoofing on the virtual scene (this second type of cyber-attack is what we demonstrate through the tutorial demo video)

‘Test evaluate’ sub-block:

- New cyber-security related KPIs are proposed following the guidelines from ISO/SAE 21434 and ISO/TS 5083 on joint safety-security evaluation.

3.9.5 Deviations from D7.2

None.

4 CONCLUSIONS

This deliverable, D7.3 “Safety Assurance Framework Demonstration Instances,” presents a comprehensive and pragmatic demonstration and validation of the **SUNRISE Safety Assurance Framework (SAF)** through the execution of eight diverse Cooperative, Connected, and Automated Mobility (CCAM) use cases. These use cases cover a broad range of operational domains, including urban, highway, and freight transport, as described in detail in **Chapter 3**. Each use case implemented and assessed multiple SAF blocks across **virtual, hybrid, and physical testing environments**, substantiating the framework’s modularity, scalability, and applicability.

Aligned with project **objective 1**—the development of a robust, harmonised methodology for CCAM safety assurance—this deliverable showcases how the SAF’s blocks support end-to-end safety validation processes. The practical demonstrations verify the ability to transform logical scenarios into concrete test cases (Query & Concretise block), allocate scenarios to appropriate test environments based on their capabilities (Allocate block), execute tests reliably (Execute block), evaluate system performance against defined safety criteria (Test Evaluate block), and contribute to safety case argumentation (Safety Case and Decide blocks), as outlined in Chapters 2 and 3.

Furthermore, in support of project **objective 7**—to demonstrate the SAF across a rich portfolio of use cases—D7.3 successfully applies the SAF’s methods and tools in varied realistic CCAM contexts. This includes using advanced techniques such as surrogate modeling for boundary exploration, hybrid virtual-physical testing for high-fidelity validation, and standardisation compliance via ASAM OpenX and OpenLABEL. The demonstrations confirm the SAF’s versatility and provide extensive feedback that has been used for refinement of WP3 (Methods), WP4 (Toolchains), and WP5 (Coverage Analysis), consistent with the feedback loop described in Section 2.4 and Section 1.4. The partnered approach ensures the SAF can address different system architectures and domain requirements (see Chapter 3).

The demonstration results presented in this deliverable, confirm that the SAF enables **evidence-based, traceable safety argumentation**, supporting consistent test planning and execution aligned with safety goals—without reliance on exhaustive real-world driving tests alone, which is infeasible for higher levels of automated driving (Section 1.1). The practical use and assessment of scenario databases, toolchains, safety metrics, and hybrid testing frameworks across use cases, strongly suggests the SAF is a viable foundation for building regulatory-acceptable safety cases as discussed in Section 2.2.7 and demonstrated throughout Chapter 3.

While the demonstrations confirm significant progress, this deliverable also identifies areas requiring further research and development, including enhanced synchronization across testing environments for real-time interoperability, refinement of coverage quantification methods in high-dimensional scenario spaces, and evaluations under adversarial or uncommon road user behaviours.

Key actionable recommendations emerging from this work include:

- Early integration of closed-loop **concretization techniques** to efficiently target risk-critical test scenarios (Sections 3.1.3.2 and 3.2.3.1).
- Expansion of SAF usage in **hybrid testing environments** that combine simulations with physical components for enhanced fidelity (Sections 3.4 and 3.7).
- Embedding formal, quantitative **coverage metrics** into safety case reasoning to **minimize test redundancy** and **improve assurance** (Sections 3.2.3.5 and 3.4.3.5).
- Ongoing **alignment with established ontologies and standards**, such as OpenLABEL and ASAM OpenSCENARIO 2.0, to support interoperability and regulatory acceptance (Sections 3.5.3.1 and 3.6.3.1).

These recommendations, derived from detailed experimentation and analysis documented in this deliverable, provide a solid foundation for stakeholders, including developers, regulators, and certification bodies—to advance harmonized CCAM safety evaluation methods (Section 1.2).

In conclusion, Deliverable D7.3 demonstrates that the SUNRISE SAF is a **mature and operationalizable framework** capable of supporting scenario-based safety validation across diverse CCAM systems and testing environments. By bridging theoretical safety assurance principles with rigorous multi-domain demonstrations (Chapters 2 and 3), it markedly advances the goal of a unified European safety validation process for CCAM systems. However, stakeholders are advised to interpret these findings with due consideration of current limitations and ongoing developmental efforts.

5 REFERENCES

- [1] ENCAP (2024). Euro NCAP AEB LSS VRU Test Protocol v4.5.1. Retrieved on Aug. 28, 2025 from <https://www.euroncap.com/media/80156/euro-ncap-aeb-lss-vru-test-protocol-v451.pdf>
- [2] SUNRISE (2025). D3.4 Report on Subspace Creation Methodology.
- [3] SUNRISE (2023). D7.1 Report on CCAM Use Cases Validation Requirements.
- [4] SUNRISE (2024). D7.2 Report on Safety Assurance Framework Demonstration Instances Design. Retrieved on Aug. 28, 2025 from <https://ccam-sunrise-project.eu/deliverable/d7-2-safety-assurance-framework-demonstration-instances-design/>
- [5] SUNRISE (2025). D3.3 Report on the Initial Allocation of Scenarios to Test Instances.
- [6] UNECE (2023). Regulation No. 157 - Automated Lane Keeping Systems (ALKS). Retrieved on Aug. 28, 2025 from <https://unece.org/transport/documents/2023/03/standards/un-regulation-157-amend4>
- [7] Da Lio, M., Mazzalai, A., Gurney, K., & Saroldi, A. (2018). Biologically guided driver modeling: The stop behavior of human car drivers. *IEEE Transactions on Intelligent Transportation Systems*, 19(8), 2454–2469.
- [8] Bosetti, P., Da Lio, M., & Saroldi, A. (2014). On the human control of vehicles: An experimental study of acceleration. *European Transport Research Review*, 6(2), 157–170. <https://doi.org/10.1007/s12544-013-0120-2>
- [9] Cherubini, A., Papini, G. P. R., Plebe, A., Piazza, M., & Da Lio, M. (2024). Bootstrapped neural models for predicting self-driving vehicle collisions with quantified confidence: Offline and online applications. *IEEE Transactions on Intelligent Vehicles*. <https://doi.org/10.1109/TIV.2024.3512786>
- [10] Cherubini, A., Papini, G. P. R., Plebe, A., Piazza, M., & Da Lio, M. (2024). Bootstrapped neural models for predicting self-driving vehicle collisions with quantified confidence: Offline and online applications. *IEEE Transactions on Intelligent Vehicles*. <https://doi.org/10.1109/TIV.2024.3512786>
- [11] ERTRAC (2024). Connected, Cooperative and Automated Mobility Roadmap. ERTRAC. Retrieved on Sep. 24, 2024 from <https://www.ertrac.org/wp-content/uploads/2023/12/ERTRAC-CCAM-Roadmap-Chapter-2-Update-2024.pdf>
- [12] ISO (2018). ISO 26262: Road Vehicles – Functional Safety. Geneva: ISO.
- [13] ISO (2022). ISO 21448: Road Vehicles – Safety of the Intended Functionality. Geneva: ISO.
- [14] Skoglund, M., Warg, F., Thorsén, A., Punnekkat, S., & Hansson, H. (2026). Methodology for test case allocation based on a formalized ODD. In M. Törngren, B. Gallina, E. Schoitsch, E. Troubitsyna, & F. Bitsch (Eds.), *Computer Safety, Reliability, and Security. SAFECOMP 2025 Workshops (Lecture Notes in Computer Science, vol. 15955, pp. 61–72)*. Cham: Springer. https://doi.org/10.1007/978-3-032-02018-5_5

- [15] Skoglund, M., Warg, F., Thorsén, A., Hansson, H., & Punnekkat, S. (2025). Formalizing operational design domains with the pkl language. Paper presented at the IEEE Intelligent Vehicles Symposium (IV), Seoul, South Korea, June 2025, pp. 1482–1489. <https://doi.org/10.1109/IV64158.2025.11097576>
- [16] ISO (2023). ISO 34503: Road Vehicles – Test Scenarios for Automated Driving Systems – Specification for Operational Design Domain. Geneva: ISO.
- [17] Apple (n.d.). Pkl :: Pkl Docs. Retrieved on Aug. 28, 2025 from <https://pkl-lang.org/>
- [18] Skoglund, M. (n.d.). ODD in Pkl. Retrieved on Aug. 28, 2025 from <https://github.com/Marskse/ODD>
- [19] PlantUML (n.d.). Open-source tool that uses simple textual descriptions to draw UML diagrams. Retrieved on Aug. 27, 2025 from <https://plantuml.com/>
- [20] Skoglund, M. (n.d.). Baseline test case allocation using an ODD. Retrieved on Aug. 28, 2025 from https://github.com/Marskse/ODD_ext
- [21] RISE Dependable Transport Systems (n.d.). WayWise. Retrieved on Aug. 28, 2025 from <https://github.com/RISE-Dependable-Transport-Systems/WayWise>
- [22] RISE Dependable Transport Systems (n.d.). WayWiseR. Retrieved on Aug. 28, 2025 from <https://github.com/RISE-Dependable-Transport-Systems/WayWiseR>
- [23] RISE Dependable Transport Systems (n.d.). ControlTower. Retrieved on Aug. 28, 2025 from <https://github.com/RISE-Dependable-Transport-Systems/ControlTower>
- [24] Coulter, R. C. (1992). Implementation of the Pure Pursuit Path Tracking Algorithm. Pittsburgh, PA: Carnegie Mellon University. Retrieved on Aug. 28, 2025 from <https://apps.dtic.mil/sti/pdfs/ADA255524.pdf>
- [25] Elhassan, A. (2015). Autonomous driving system for reversing an articulated vehicle. Master Thesis, KTH Royal Institute of Technology. Retrieved on Oct. 24, 2023 from <https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-175373>
- [26] Kvarnfors, K. (2019). Motion Planning for Parking a Truck and Trailer System. Master Thesis, KTH Royal Institute of Technology. Retrieved on Apr. 15, 2024 from <https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-264501>
- [27] Dosovitskiy, A., Ros, G., Codevilla, F., López, A. M., & Koltun, V. (2017). CARLA: An open urban driving simulator. Paper presented at the Conference on Robot Learning.
- [28] RISE Dependable Transport Systems (n.d.). Carla ROS Bridge. Retrieved on Aug. 28, 2025 from <https://github.com/RISE-Dependable-Transport-Systems/carla-ros-bridge>
- [29] Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot Operating System 2: Design, architecture, and uses in the wild. Science Robotics, 7(66), eabm6074. <https://doi.org/10.1126/scirobotics.abm6074>
- [30] (2025). [Article in press]. Retrieved on Aug. 28, 2025 from <https://www.sciencedirect.com/science/article/pii/S016740482500238X>
- [31] ISO/SAE (2021). ISO/SAE 21434: Road Vehicles – Cybersecurity Engineering. Geneva: ISO & SAE International.

[32] SUNRISE Project Consortium. (2024). *D2.3 Final SUNRISE Safety Assurance Framework*. Horizon Research and Innovation Actions, Project No. 101069573. Submitted August 2025.