Ansys / DYNARDO | presented at the 17th Weimar Optimization and Stochastic Days 2020 |
Source: www.dynardo.de/en/library

A Simulation Tool Chain for Verification and Validation of L3 and Higher Level Autonomous Vehicles

Ralf Lampert, Senior Product Sales Manager, Ansys

17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020
Virtual Conference

Ansys

Agenda

- Introduction
- Case Studies
- Validation of Safety Critical Scenarios
- VRX Driving Simulator & optiSLang
- Automatic Controller Calibration with VRX Driving Simulator and optiSLang
- Q&A

Introduction of Autonomous Vehicles

17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020

Virtual Conference

Ansys

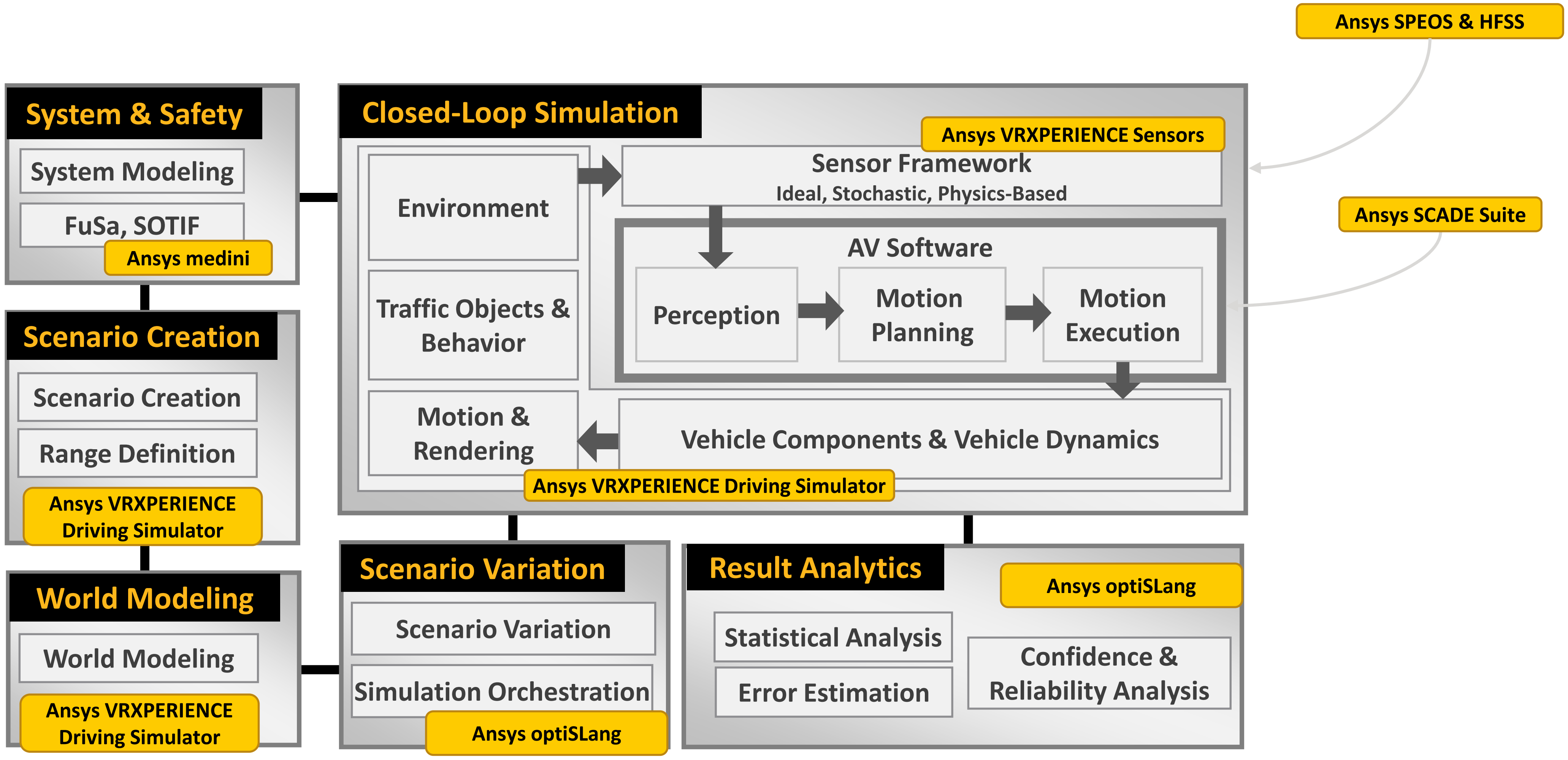
Why are Automotive companies (OEM & Supplier) choosing Ansys to deliver autonomous & driver assistance technologies?

Single largest engineering simulation company in the world

- Est. 50+ years
- 100+ offices
- 4000+ employees (many in R&D & Engineering)

- Market leaders for ADAS & electrification simulation
- $23bn market cap
- >$1bn in recent acquisitions

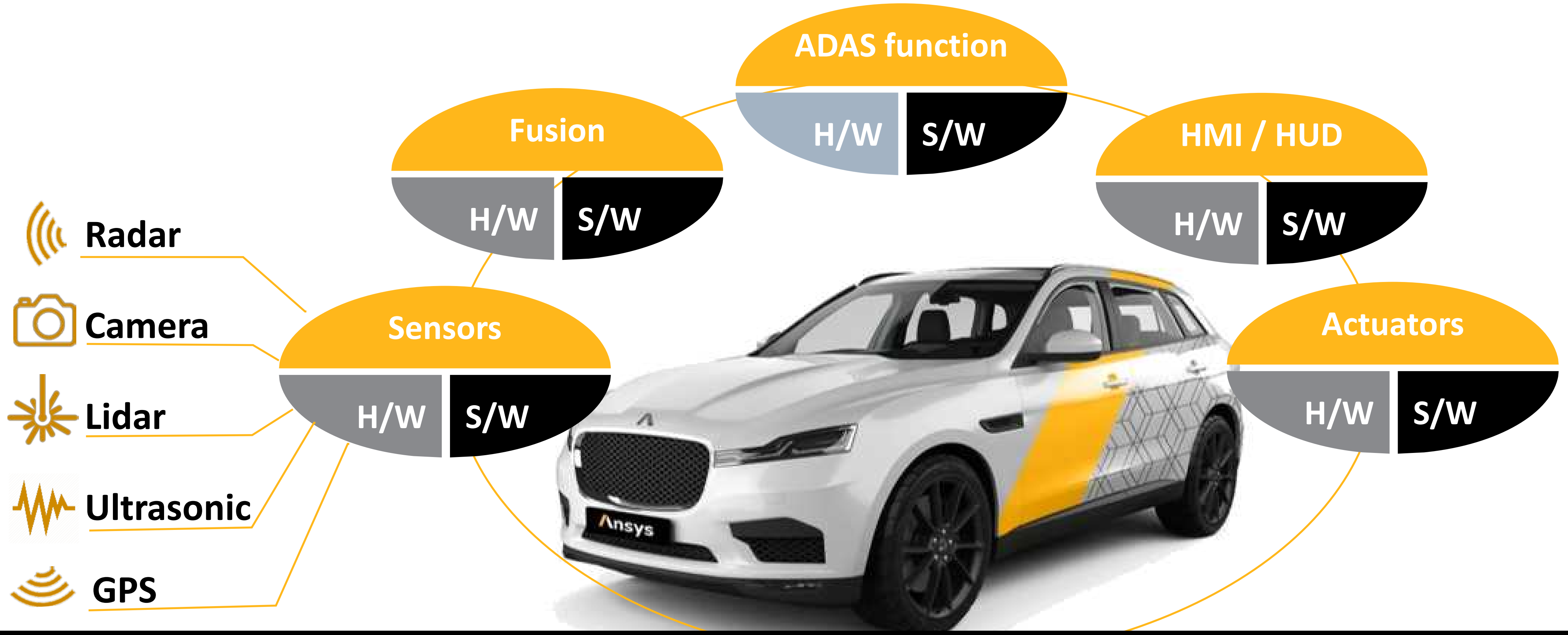
Autonomous Vehicles simulation platform view



Much wider breadth throughout the V-cycle

MIL or SIL on Workstation	Driver Simulator	HPC / Cloud	HIL
Preparation (scenario,...) Quick testing of control software (policy) MIL or SIL	Human in the loop testing AD L2/L3 - Driver take over and re-engagement Situational awareness	Campaign testing against Millions of scenarios Non regression testing	ECU control software testing Non regression testing Validation

Addresses all aspects of an ADAS system



L2 ADAS
AEB, ACC, LKA…

L3 AD
TJC, HC, …

Automatic Parking

Intelligent Lighting

Moving from L2 to L3-L4 requires a technological quantum leap

Driver	Feet off	Hands off	Eyes off	Mind off	...everywhere

No assistance (L 0)	Assistance (L 1)	Semi Automated (L 2)	Highly Automated (L 3)	Autonomous (L 4)	Autonomous (L 5)

Driver Only

Driver is responsible (Advanced Driver Assistance Systems - ADAS)

Machine is responsible (Autonomous Vehicles - AV)

Image Courtesy: BMW

Developing L3-L4 mainly requires mastering safety challenges

Case Studies

17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020

Virtual Conference

Ansys

Ansys - BMW Group Technology Partnership

“Ansys And BMW Group Partner To Jointly Create The Industry's First Simulation Tool Chain For Autonomous Driving”

New agreement drives development of autonomous driving technology for the BMW iNEXT, the next-generation autonomous vehicle
https://www.ansys.com/about-ansys/news-center/06-10-19-ansys-bmw-group-partner-jointly-create-simulation-tool-chain-autonomous-driving

- **Long term agreement**
- **Level 3 / 4**
- **iNext Launch 2021**

Ansys will assume exclusive rights to the simulation tool chain technology for commercialization to a wider market as part of Ansys Autonomy.

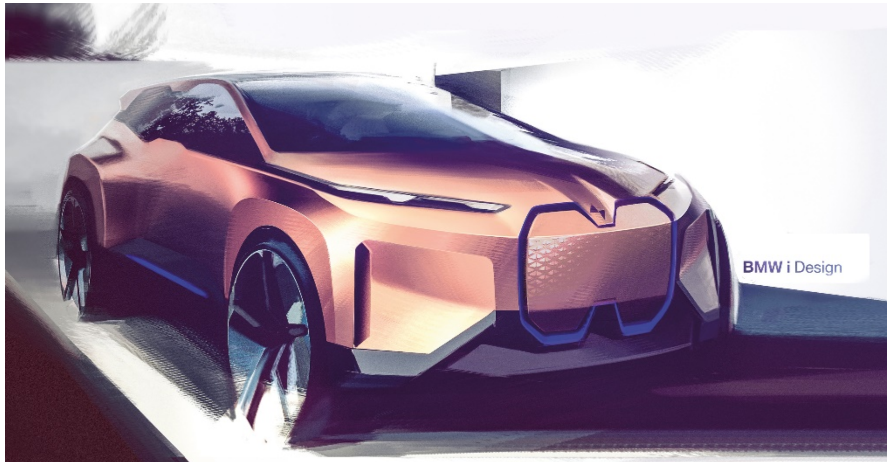
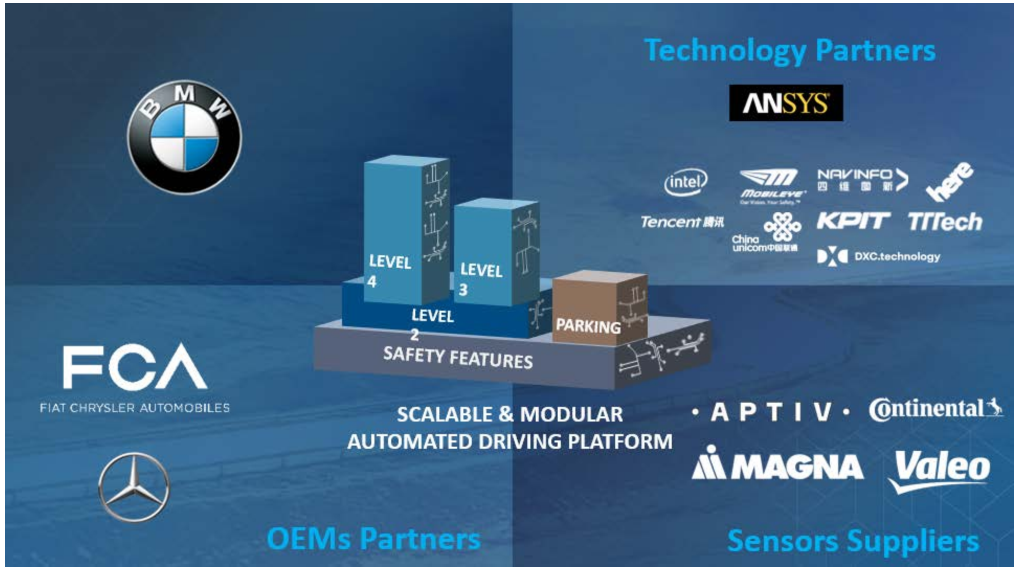


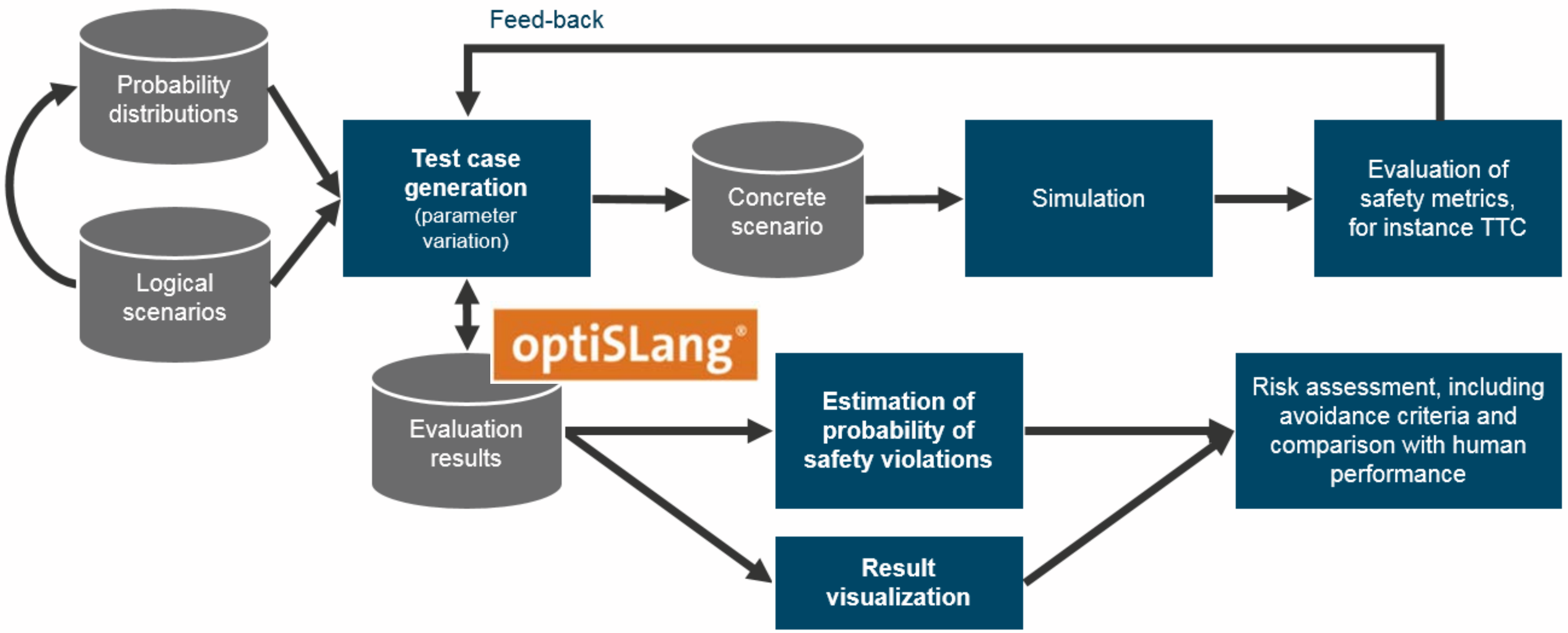
Image source: BMW Press Photos Website



ADAS L3 scenario based using reliability analysis

Daimler has implemented Ansys optiSLang
for **automation** of driving scenario-based evaluation

Result is a solid **workflow** considering robustness evaluation and reliability analysis
for parameterized driving scenarios in a way that is much more efficient than Monto-Carlo Sampling.



M. Rasch (Daimler AG), Simulative validation of automated driver assistance systems using reliability analysis, *WOSD, Weimar, 2019*

Validation of safety critical scenarios with Reliability Analysis

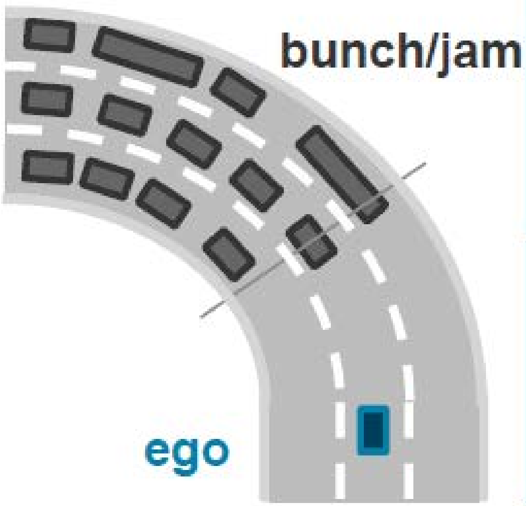
17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020

Virtual Conference

Ansys

Functional vs. Logic Scenarios



Functional scenarios	Logic scenarios
Basis road: highway in bend	**Basis road:** number of lanes [2..4] curve radius [0,6..0,9] kph
Stationary objects: -	**Stationary objects:** -
Movable objects: ego, jam; interaction: ego approaches end of jam	**Movable objects:** End of jam position[10..200] m jam speed [0..30] kph ego distance [50..300] m ego speed [80..130] kph
Environment: summer, rain	**Environment:** temperature [10..40] °C droplet size [20..100] µm rain amount [0,1..10] mm/h

© PEGASUS | VDA Technical Congress | April 6, 2017

How many miles …

Table 1. Examples of Miles and Years Needed to Demonstrate Autonomous Vehicle Reliability

Statistical Question		Benchmark Failure Rate		
	How many miles (years[a]) would autonomous vehicles have to be driven…	**(A) 1.09 fatalities per 100 million miles?**	**(B) 77 reported injuries per 100 million miles?**	**(C) 190 reported crashes per 100 million miles?**
	(1) without failure to demonstrate with 95% confidence that their failure rate is at most…	275 million miles (12.5 years)	3.9 million miles (2 months)	1.6 million miles (1 month)
	(2) to demonstrate with 95% confidence their failure rate to within 20% of the true rate of…	8.8 billion miles (400 years)	125 million miles (5.7 years)	51 million miles (2.3 years)
	(3) to demonstrate with 95% confidence and 80% power that their failure rate is 20% better than the human driver failure rate of…	11 billion miles (500 years)	161 million miles (7.3 years)	65 million miles (3 years)

[a] We assess the time it would take to compete the requisite miles with a fleet of 100 autonomous vehicles (larger than any known existing fleet) driving 24 hours a day, 365 days a year, at an average speed of 25 miles per hour.

Source: Nidhi Kalra, Susan M. Paddock: Driving to Safety, www.rand.org

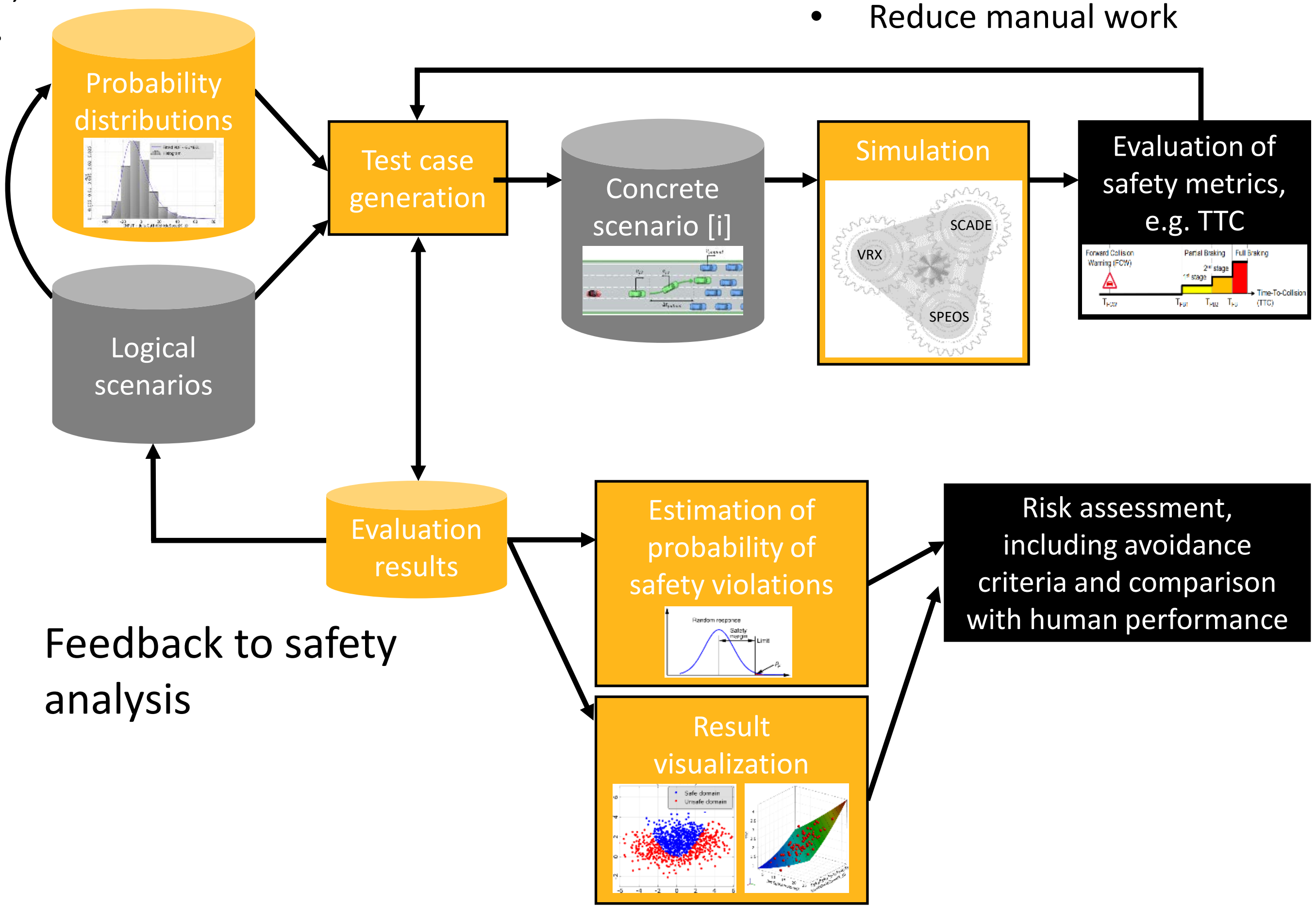
➡ **There is a crucial Need for Smart Reliability Methods for Vehicle Function Evaluation & Validation**

Vehicle function evaluation based on simulation of scenarios

Input from safety analysis including
- measurements,
- databases etc.

Reduce number of designs necessary for validation

Workflow generation & automation capability
- Combine capabilities of several tools,
- Standardize workflows &
- Reduce manual work



Quantification of probability of failure

Identification of safety critical inputs & input combinations

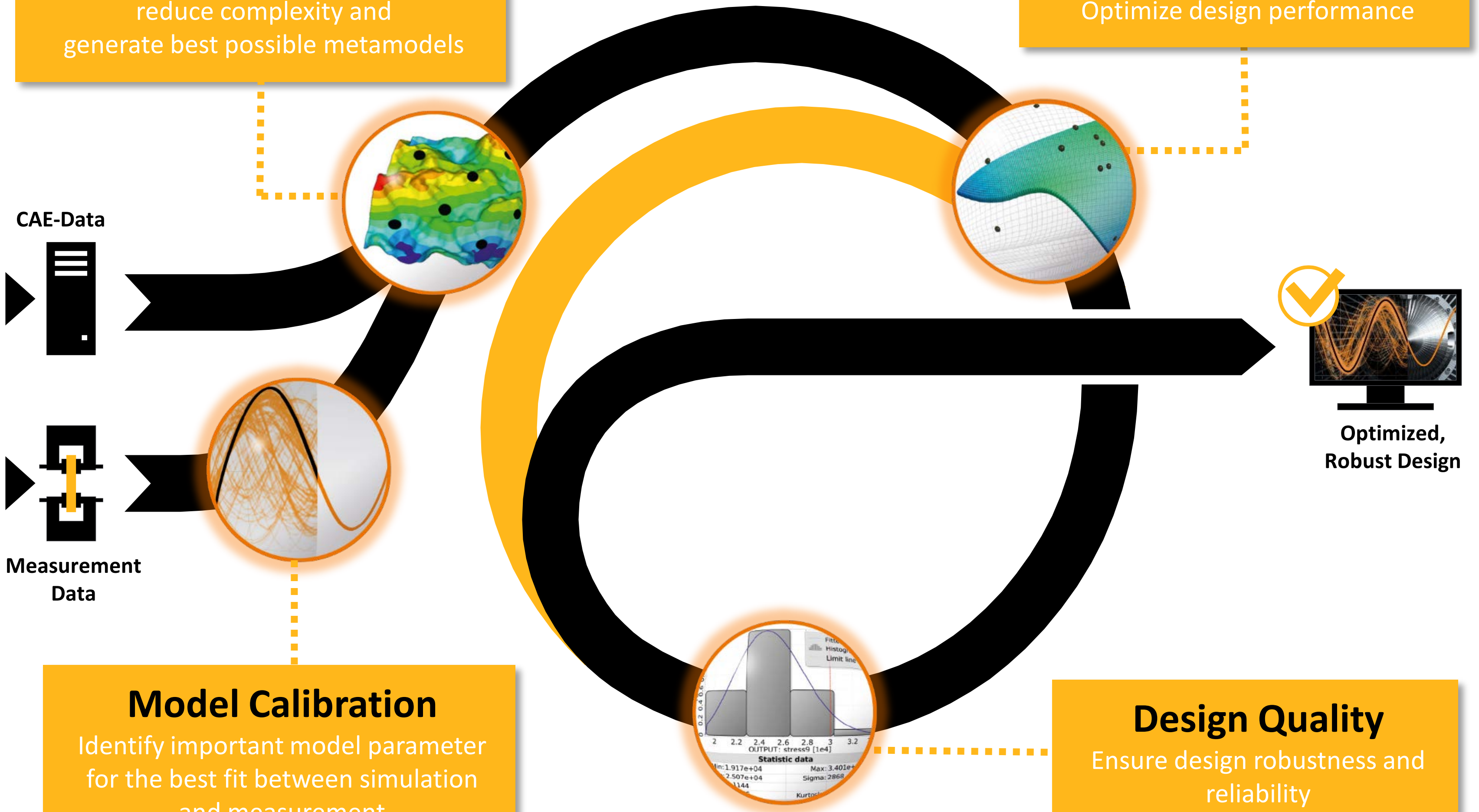
Identification of software malfunction

Comparison of performance between different software versions/ scenarios etc.

Robust Design Optimization Strategy

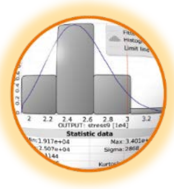
Design Understanding
Investigate parameter sensitivities, reduce complexity and generate best possible metamodels

Design Improvement
Optimize design performance



Model Calibration
Identify important model parameter for the best fit between simulation and measurement

Design Quality
Ensure design robustness and reliability



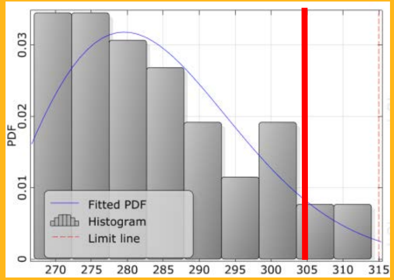
Robustness Evaluation

Ensure your product quality!

Latin Hypercube Sampling

	NORMAL	1
	NORMAL	20
	NORMAL	0.02
	NORMAL	10

Output parameter variation



Input parameter importance

Sensitivity vs Robustness vs Reliability Analysis

Analysis	Design Space	Application	Main purpose and outcome	Probability of failure
Sensitivity	"**Optimization space**" with equal distribution of input parameters (all controllable)	Early in design phase; software function **testing** during development	**Exploration** of the parameter variation space, Identification of software malfunction, consistency checks	not of interest here
Robustness	"**Robustness space**" with particular distribution of input parameters (controllable and not controllable)	**Later** in design phase; software function **validation**	Estimate **variation range** of output parameters, Quantification of **probability of failure**, Identification of critical input parameters	is **usually higher** (10^{-6} and higher)
Reliability	"**Robustness space**" with **precise** distribution of input parameters (controllable and not controllable), Definition of a **failure limit** is crucial	**Later** in design phase, software function **validation**	Quantification of **probability of failure**, Identification of critical input parameters	is usually **very low** (10^{-6} and lower)

Sensitivity vs Robustness vs Reliability Analysis

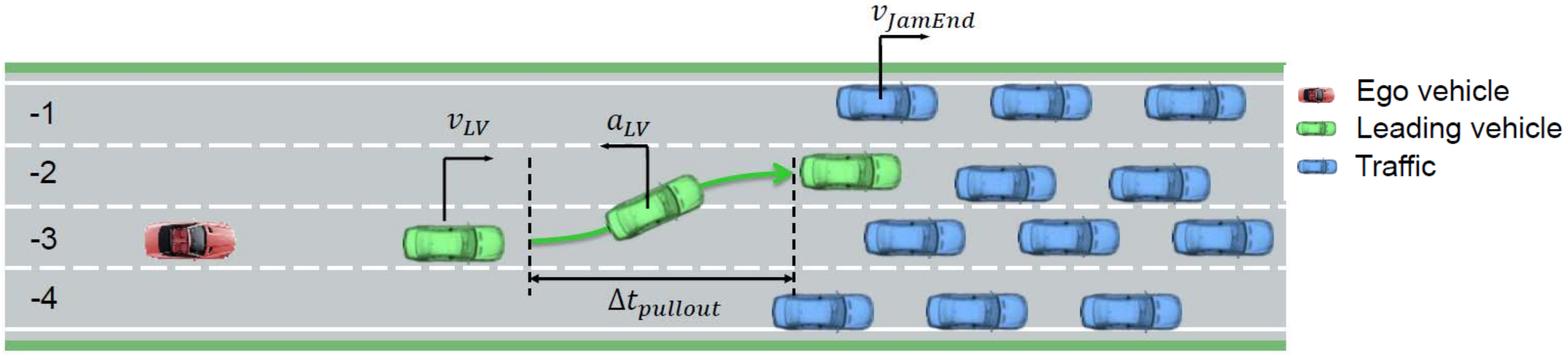
Analysis	Design Space	Application	Main purpose and outcome	Probability of failure
Sensitivity	"**Optimization space**" with equal distribution of input parameters (all controllable)	Early in design phase; software function **testing** during development	**Exploration** of the parameter variation space, Identification of software malfunction, consistency checks	not of interest here
Robustness	"**Robustness space**" with particular distribution of input parameters (controllable and not controllable)	**Later** in design phase; software function **validation**	Estimate **variation range** of output parameters, Quantification of **probability of failure**, Identification of critical input parameters	is **usually higher** (10^{-6} and higher)
Reliability	"**Robustness space**" with **precise** distribution of input parameters (controllable and not controllable), Definition of a **failure limit** is crucial	**Later** in design phase, software function **validation**	Quantification of **probability of failure**, Identification of critical input parameters	is usually **very low** (10^{-6} and lower)

Ansys

Example of scenario based evaluation

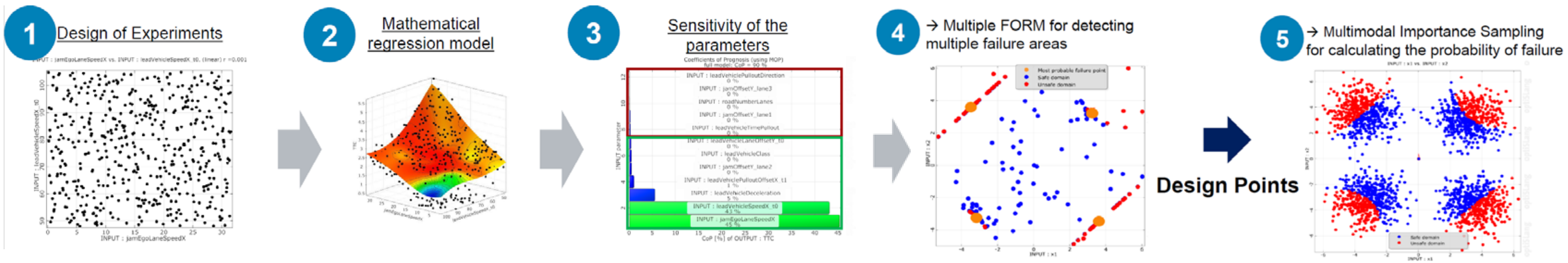
Reduce the number of simulation by a factor of 1000

"jam-end" functional scenario → logical scenarios with 13 parameters → require 39.420.000 concrete scenarios using Monte-Carlo approach



Adaptive Sampling or ISPUD reduce the required concrete scenarios to obtain similar results in terms of probability of failure

➔ here: 28,500 simulation runs versus 39.420.000 simulations



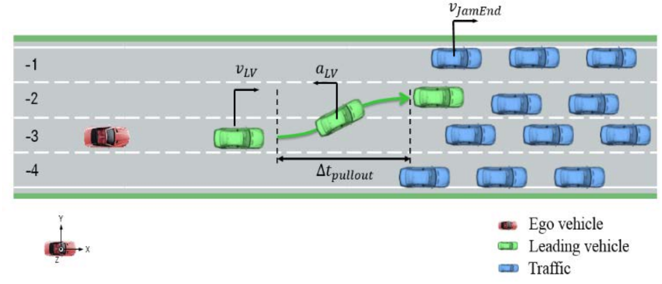
M. Rasch (Daimler AG), Simulative validation of automated driver assistance systems using reliability analysis, *WOSD, Weimar, 2019*

Ansys

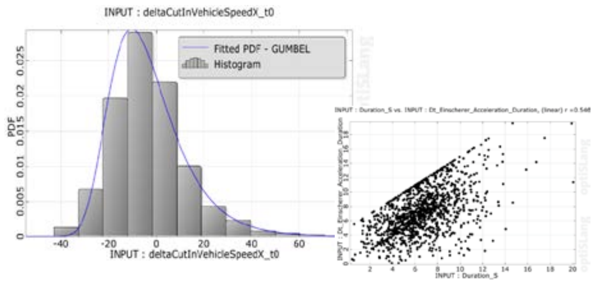
Scenario based evaluation (details)

Automated workflow in Ansys Autonomy platform

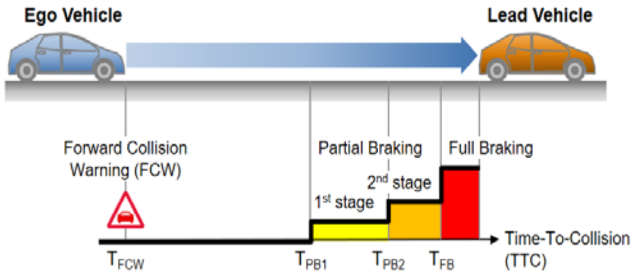
1. Define scenario and its parametric



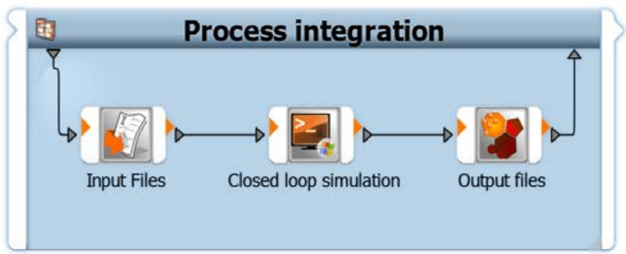
2. Derive parameter scatter and correlation



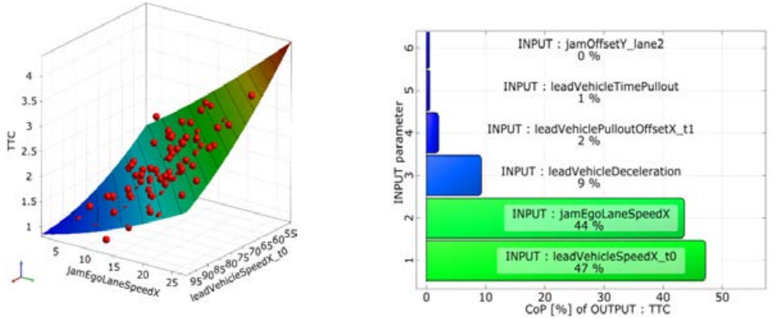
3. Define criticality by means of available KPIs, e.g. TTC



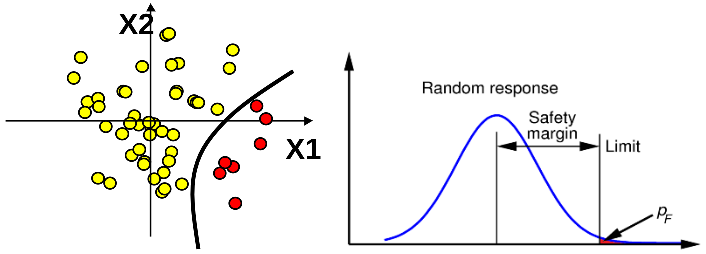
4. Automate simulation runs



5. Get parameter importance by robustness analysis



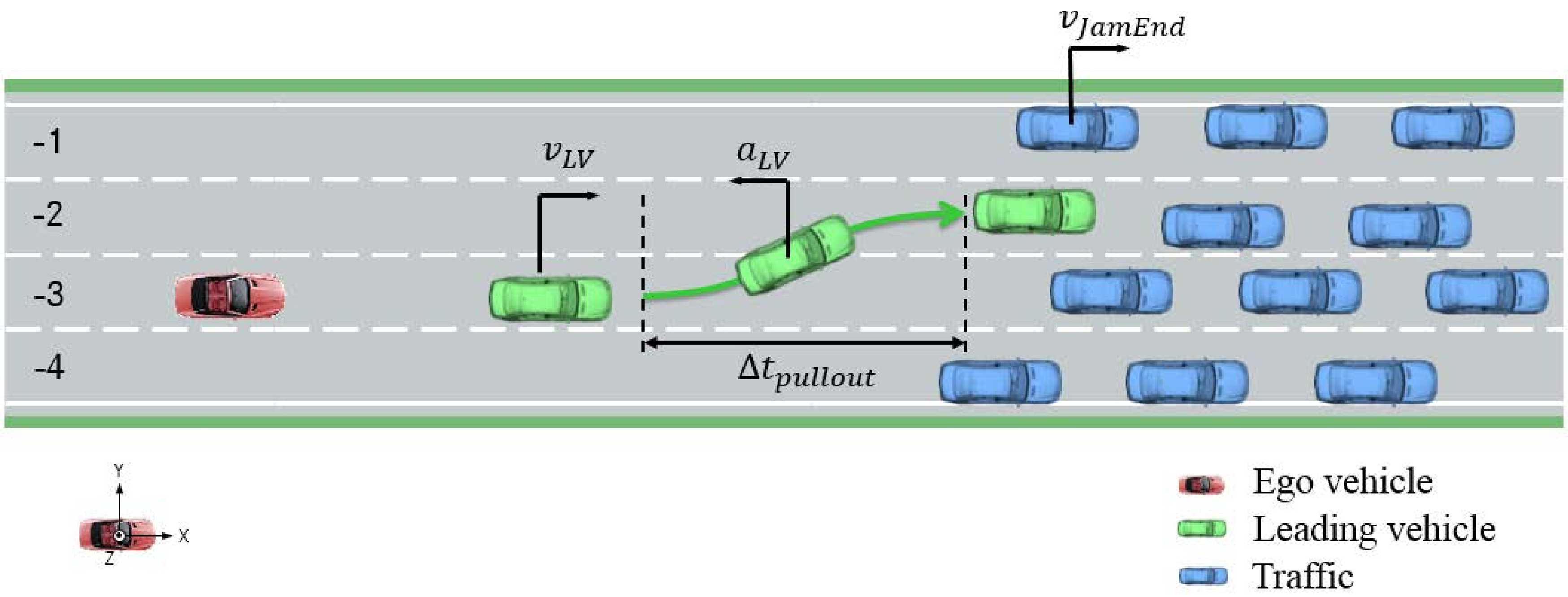
6. Uncertainty quantification by reliability analysis



Jam End Scenario: Random Parameters

- 7 stochastic parameters describing e.g. jam end speed, lead vehicle speed & deceleration, lead vehicle class, pullout direction & time

	Name	PDF	Type	Distribution parameter	Mean	Std. Dev.
1	jam_end_speed		TRUNCATEDNORMAL	20; 10; 0; 50	20.5078	9.34424
2	lead_vehicle_class		DISCRETE	1; 0.2; 2; 0.3; 3; 0.5	2.3	0.781025
3	lead_vehicle_deceleration		BETA	2; 8; 2.5; 2	5.33333	1.27128
4	lead_vehicle_pullout_direction		DISCRETE	-1; 0.5; 1; 0.5	0	1
5	lead_vehicle_pullout_offset		UNIFORM	0; 1	0.5	0.288675
6	lead_vehicle_pullout_time		LOGNORMAL	-0.804719; 0.472381	0.5	0.25
7	lead_vehicle_speed		TRUNCATEDNORMAL	80; 10; 50; 110	80	9.86578



Jam end scenario variation with optiSLang

- Collaborative work possible because all information is everywhere available, here: input parametrization in Postprocessing

Parametrization

Parameter | Responses | Criteria

	Activation	Name	Parameter type	Reference value	Constant	PDF	Type	Mean	Std. Dev.	CoV
1	☑	jam_end_speed	Stochastic	15	☐		TRUNCATEDN...	20.5078	9.34424	45.5643 %
2	☐	lead_vehicle_class	Stochastic	2	☑ filtered		DISCRETE	2.3	0.781025	33.9576 %
3	☑	lead_vehicle_deceleration	Stochastic	8	☐		BETA	5.33333	1.27128	23.8366 %
4	☐	lead_vehicle_pullout_direction	Stochastic	1	☑ filtered		DISCRETE	0	1	100 %
5	☐	lead_vehicle_pullout_offset	Stochastic	0.5	☑ filtered		UNIFORM	0.5	0.288675	57.735 %
6	☐	lead_vehicle_pullout_time	Stochastic	0.5	☑ filtered		LOGNORMAL	0.5	0.25	50 %
7	☑	lead_vehicle_speed	Stochastic	90	☐		TRUNCATEDN...	80	9.86578	12.3322 %

- Determine concrete critical scenarios, export & visualize them

Designtable

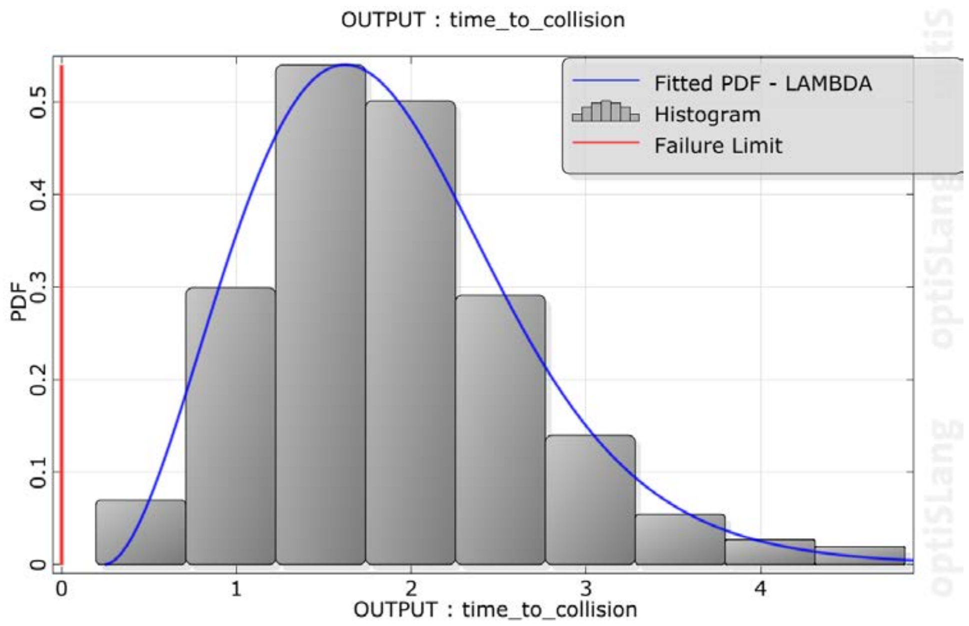
	tivati	Id	Feasible	plicat	Status	Style	jam_end_speed	ead_vehicle_clas:	vehicle_decelera	ehicle_pullout_dir	vehicle_pullout_c	vehicle_pullout_	ead_vehicle_spee	time_to_collision
4	☑	0.383	true		Succeeded		34.7105	2	6.9517	1	0.577	0.381848	56.0592	4.56883
5	☑	0.323	true		Succeeded		44.9888	2	5.91616	1	0.075	0.509663	71.2884	4.34155
6	☑	0.478	true		Succeeded		37.8098	3	5.93865	-1	0.193	0.504686	61.5152	4.22531
7	☑	0.497	true		Succeeded		39.81	2	5.66379	1	0.175	0.228066	65.1043	4.18682
8	☑	0.210	true		Succeeded		35.8067	2	7.06036	1	0.667	0.393382	64.6331	4.0885
9	☑	0.44	true		Succeeded		36.7246	3	6.20611	-1	0.999	0.37517	65.6853	4.01766
10	☑	0.26	true		Succeeded		42.5442	1	5.74507	1	0.567	0.55558	74.3366	3.9571

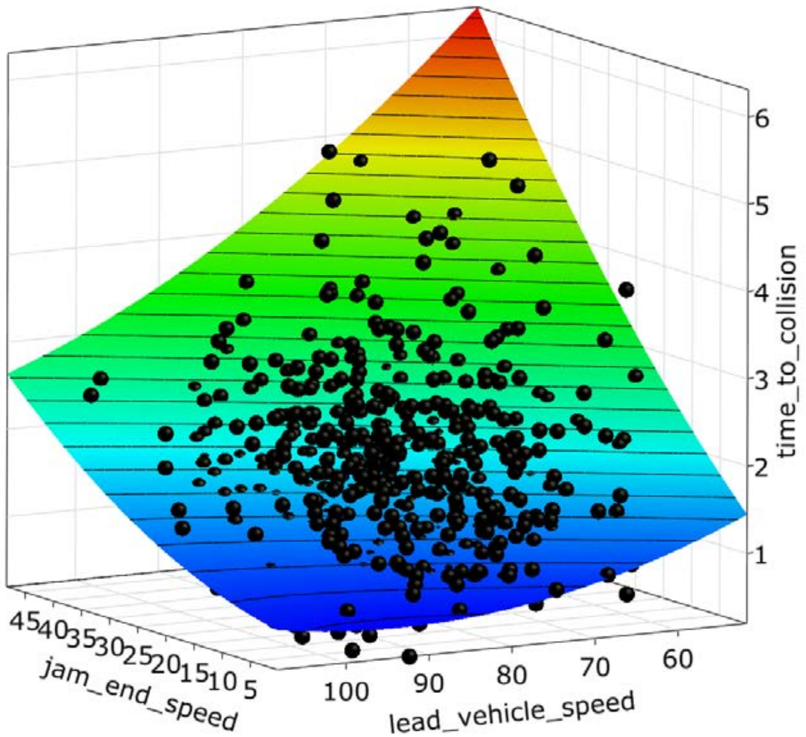
17th Weimar Optimization and Stochastic Days 2020
June 25 – 26, 2020
Virtual Conference

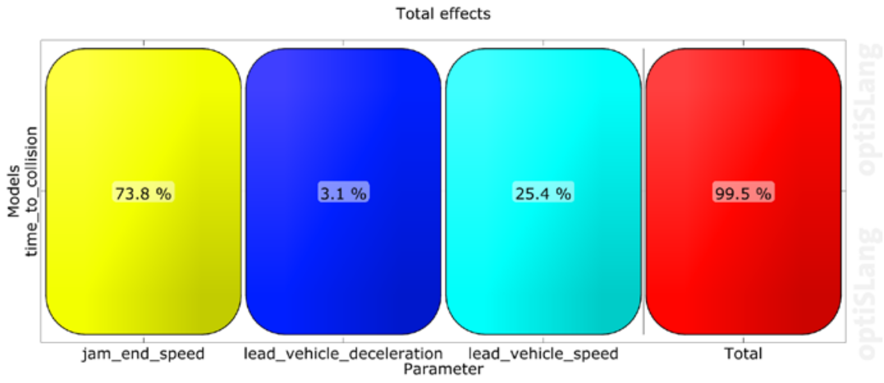
Jam end scenario variation with optiSLang

Reliability analysis:

- 3 most important parameters can explain 99.5% of variation
- TTC can be represented by lambda distribution very well
- However, extrapolation into region TTC <= 0 is not confident



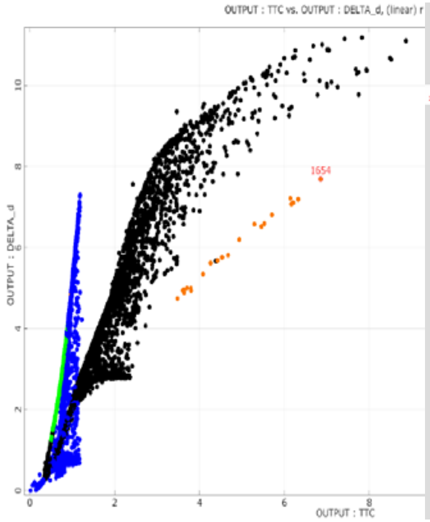


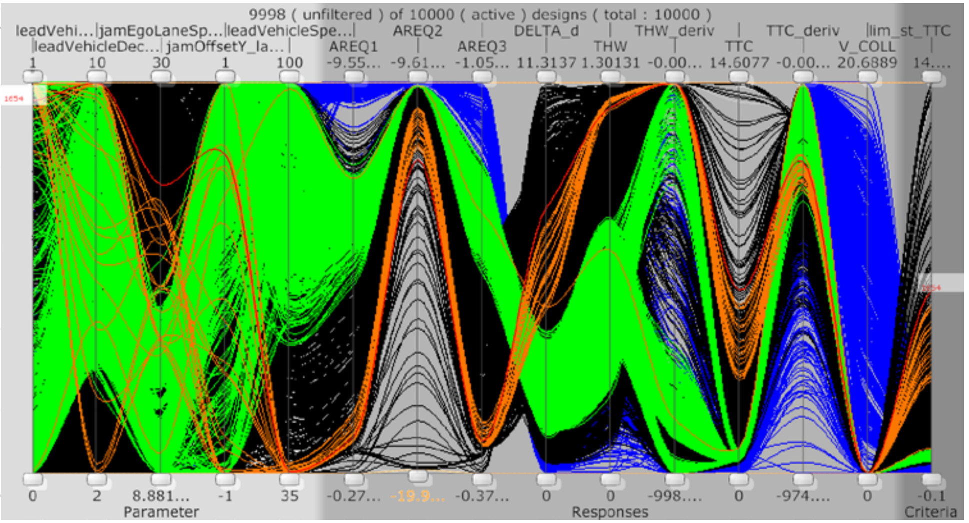


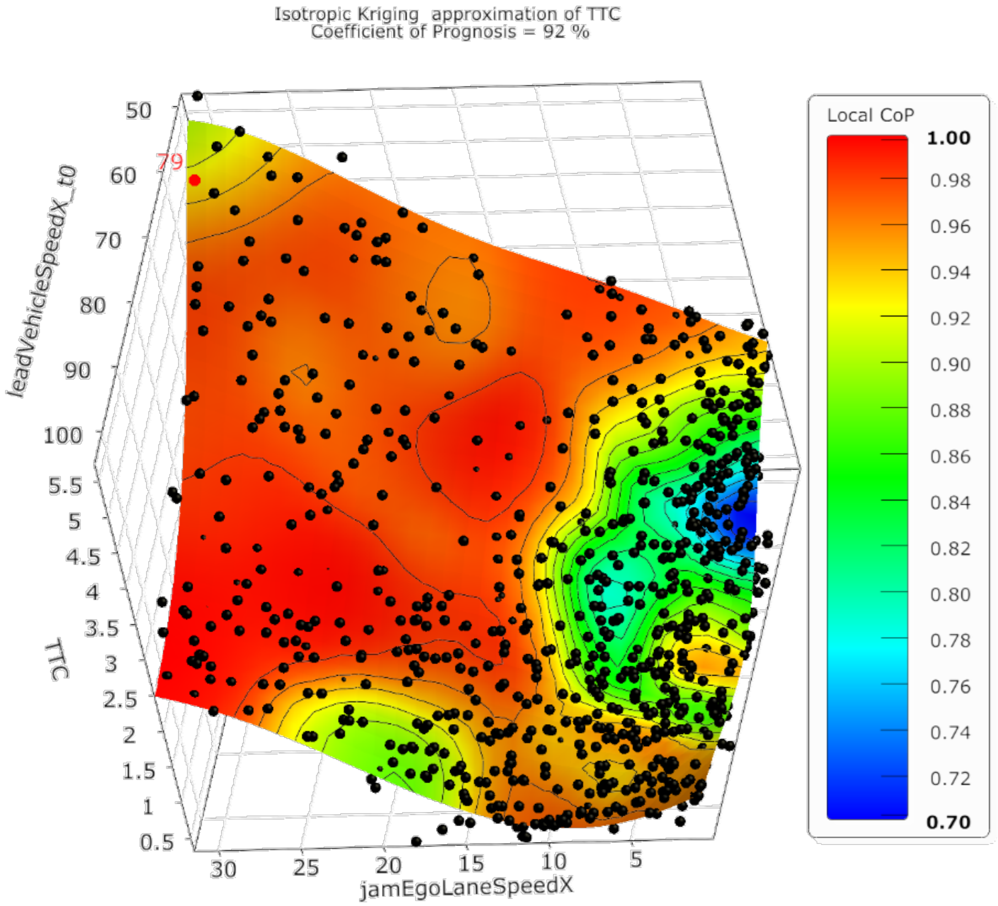
Ansys

Detection of software malfunction

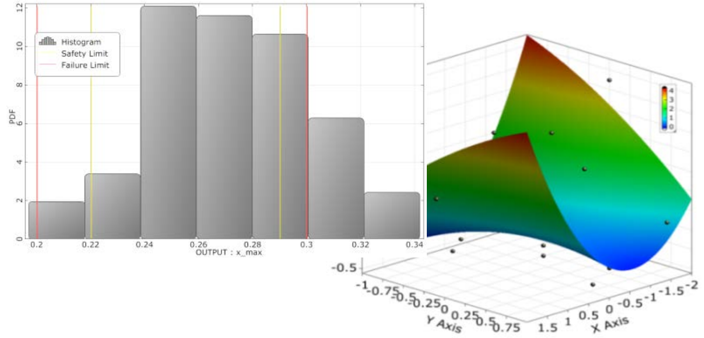
- Partially low local Coefficient of Prognosis (CoP)
- Assumption special physical and control mechanisms in these regions
- Some output parameters are used for the steering and therefore have impact on other output parameters
- Analysis provided excellent indication which parameters are used for steering



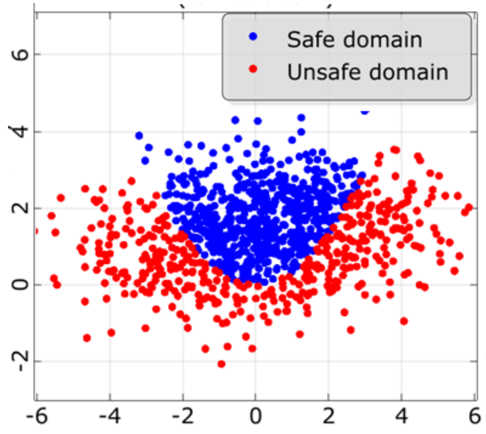




Safety Assessment



Robustness Analysis



Reliability Analysis

- Check variation of inputs & responses
- Check plausibility in MOP to proof simulation model
- Eventually reduce parameter number
- 200 – 500 samples
- Check different safety limits
- Stop if failure probability is large

- Define specific failure criterion
- Perform reliability analysis (Importance Sampling) until defined accuracy is reached
- 10000 – 20000 samples
- In case of fulfilled safety requirement: proof the result with different approach

Jam end scenario variation with optiSLang

- Definition of the limit state function to be analysed for a concrete scenario with the defined input parameter variation range

Parametrization

Parameter | Responses | Criteria

Criteria

Name	Type	Expression	Criterion	Limit	Evaluated expression
lim_st_time_to_collision	Limit state	time_to_collision	≥	0	0.644133 ≥ 0
new					

Create new

Variable | Objective | Constraint | Limit state

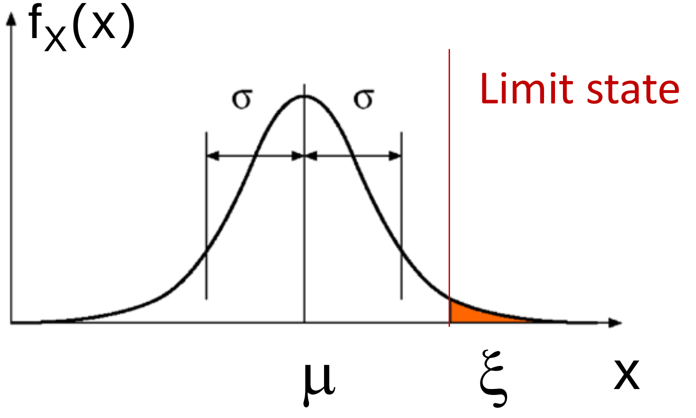
Instant visualization | Import criteria from file | Apply

Parametrization

Parameter | Responses | Criteria

	Activation	Name	Parameter type	Reference value	Constant	PDF	Type	Mean	Std. Dev.	CoV
1	☑	jam_end_speed	Stochastic	15	☐		TRUNCATEDN...	20.5078	9.34424	45.5643 %
2	☑	lead_vehicle_class	Stochastic	2	☐		DISCRETE	2.3	0.781025	33.9576 %
3	☑	lead_vehicle_deceleration	Stochastic	8	☐		BETA	5.33333	1.27128	23.8366 %
4	☑	lead_vehicle_pullout_direction	Stochastic	1	☐		DISCRETE	0	1	100 %
5	☑	lead_vehicle_pullout_offset	Stochastic	0.5	☐		UNIFORM	0.5	0.288675	57.735 %
6	☑	lead_vehicle_pullout_time	Stochastic	0.5	☐		LOGNORMAL	0.5	0.25	50 %
7	☑	lead_vehicle_speed	Stochastic	90	☐		TRUNCATEDN...	80	9.86578	12.3322 %

Jam end scenario variation with optiSLang

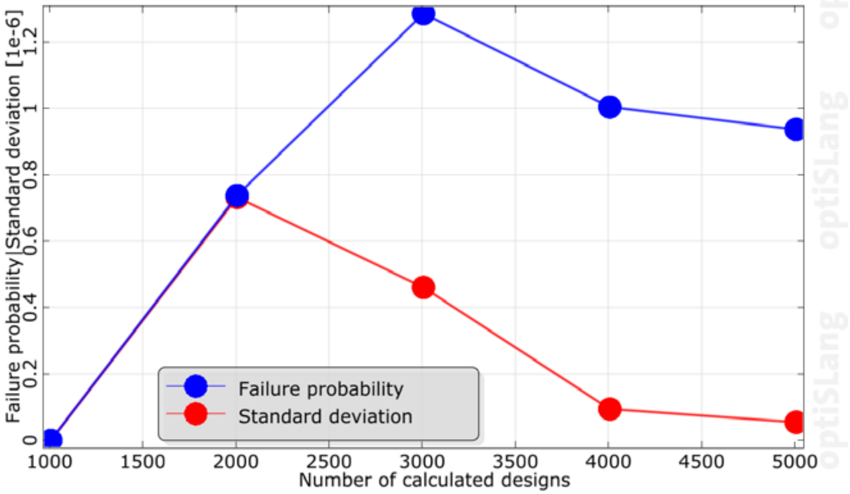


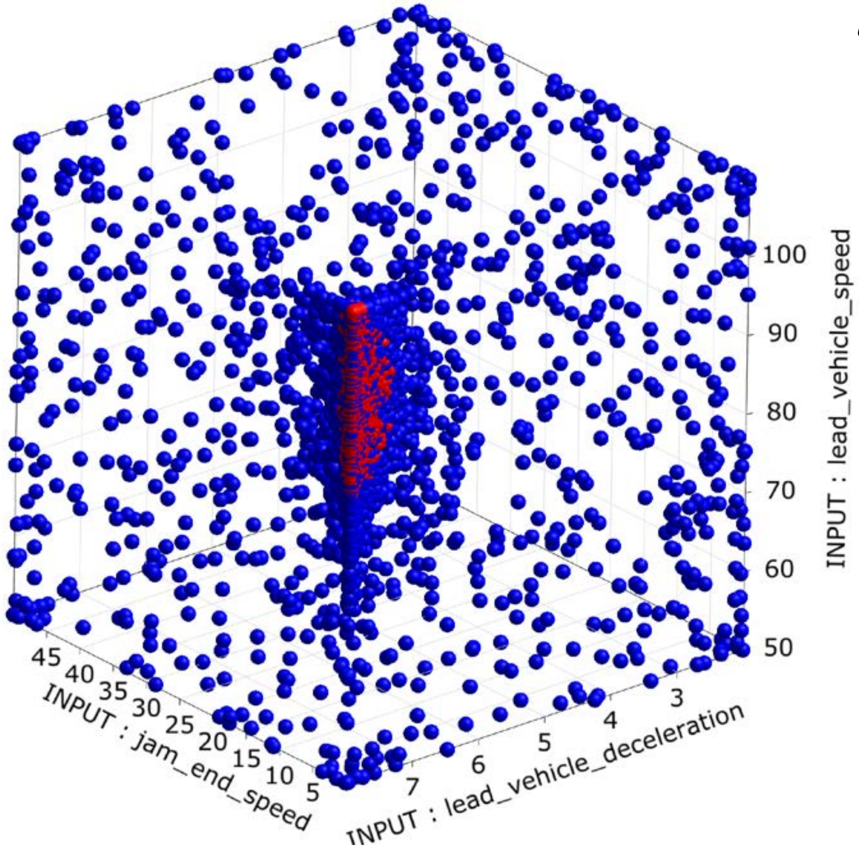
Reliability analysis: Adaptive Sampling

- Time to collision as limit state, TTC <= 0.0 s
- 10% accurracy with 5 iterations each having 1000 samples

Probability of Failure : 9.36376e-7
Standard deviation error : 5.48267e-8
Reliability Index : 4.76669

Number of designs
Total : 5000
Safe domain : 3219
Unsafe domain : 1781





Ansys

17th Weimar Optimization and Stochastic Days 2020
June 25 – 26, 2020
Virtual Conference

VRX Driving Simulator & optiSLang

17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020

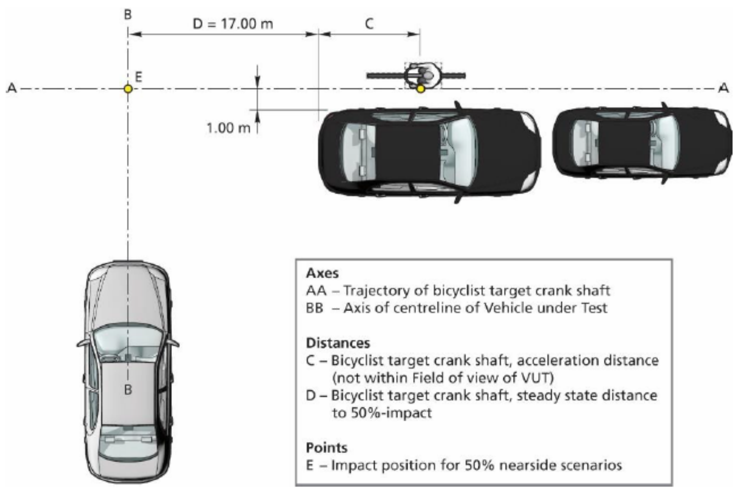
Virtual Conference

Ansys

Scenario: Car-to-Bicyclist Nearside Adult 50% – CBNA 50

Description:
A collision in which a vehicle travels forwards towards an bicyclist crossing its path cycling from the **nearside** and the frontal structure of the vehicle strikes the bicyclist when no braking action is applied.

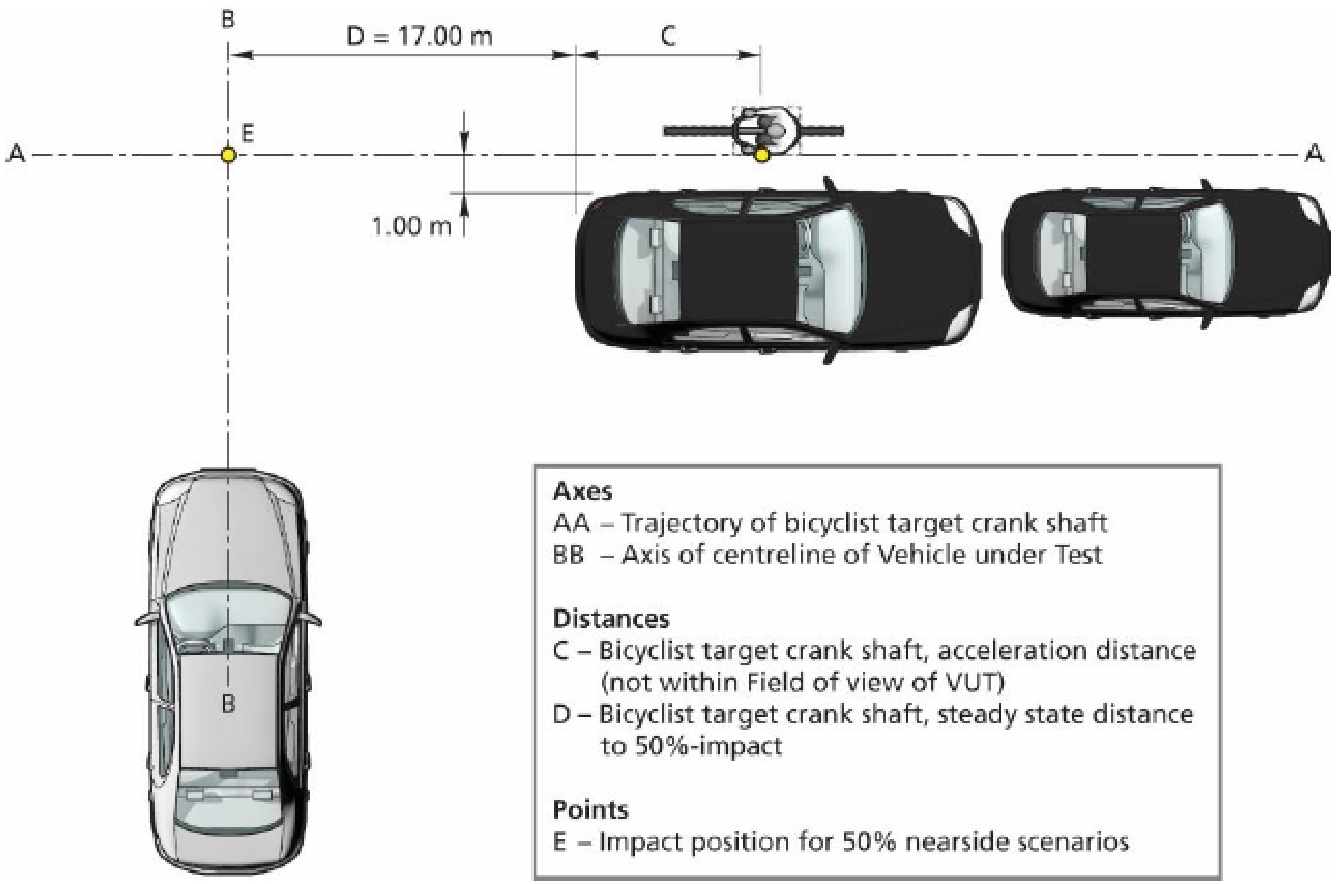
Scenario Name	CBNA
Type of test	AEB
VUT speed [km/h]	10 - 60
VUT direction	Forward
Target speed [km/h]	15
Impact location [%]	50
Lighting condition	Day



Optimization Task

Parameter

- VUT speed
- Collision threshold
- Default Breaking Force
- Detection Range
- EBT visual
- FOV Horizontal
- Speed threshold



Axes
AA – Trajectory of bicyclist target crank shaft
BB – Axis of centreline of Vehicle under Test

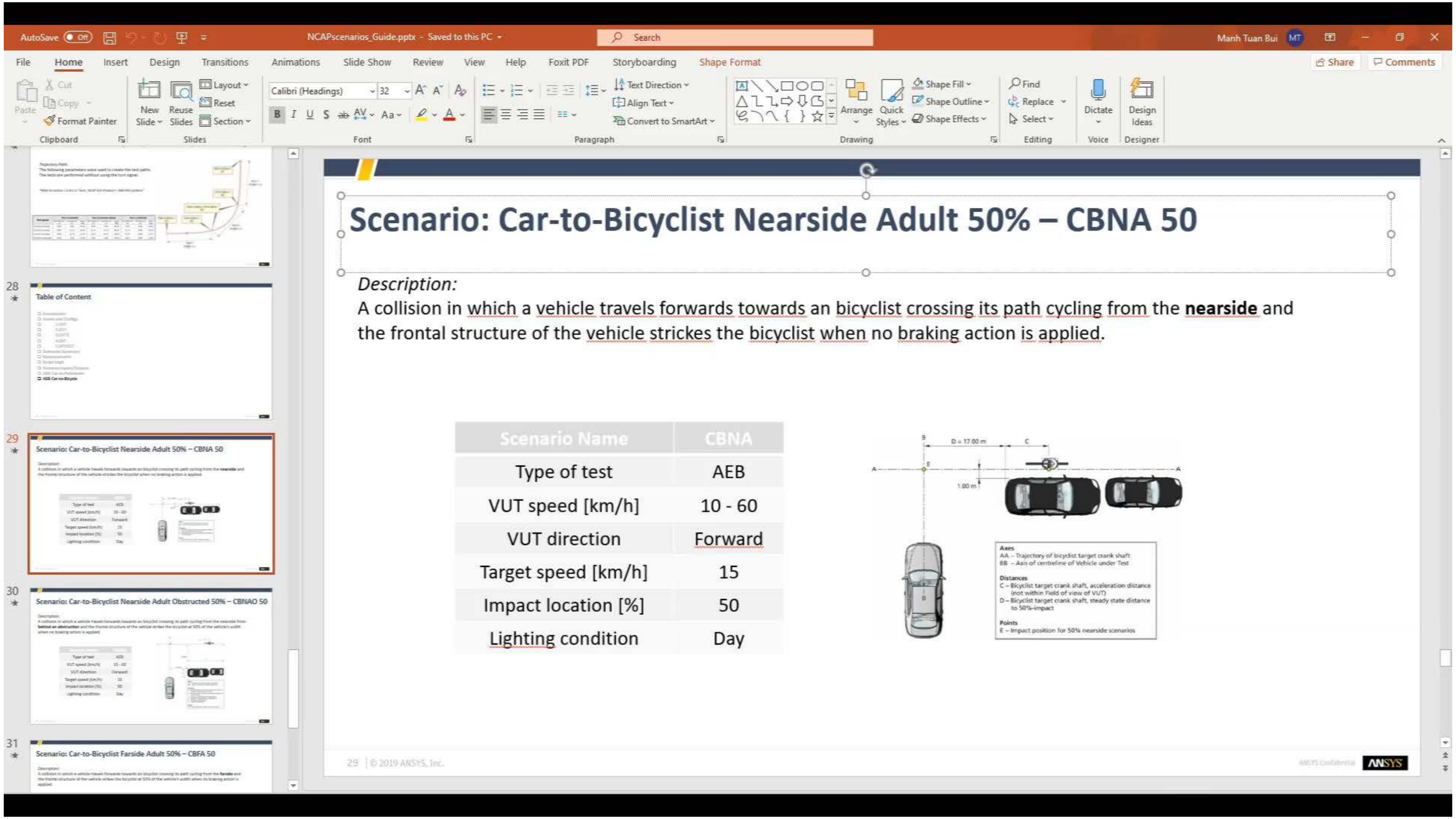
Distances
C – Bicyclist target crank shaft, acceleration distance (not within Field of view of VUT)
D – Bicyclist target crank shaft, steady state distance to 50%-impact

Points
E – Impact position for 50% nearside scenarios

Objective functions

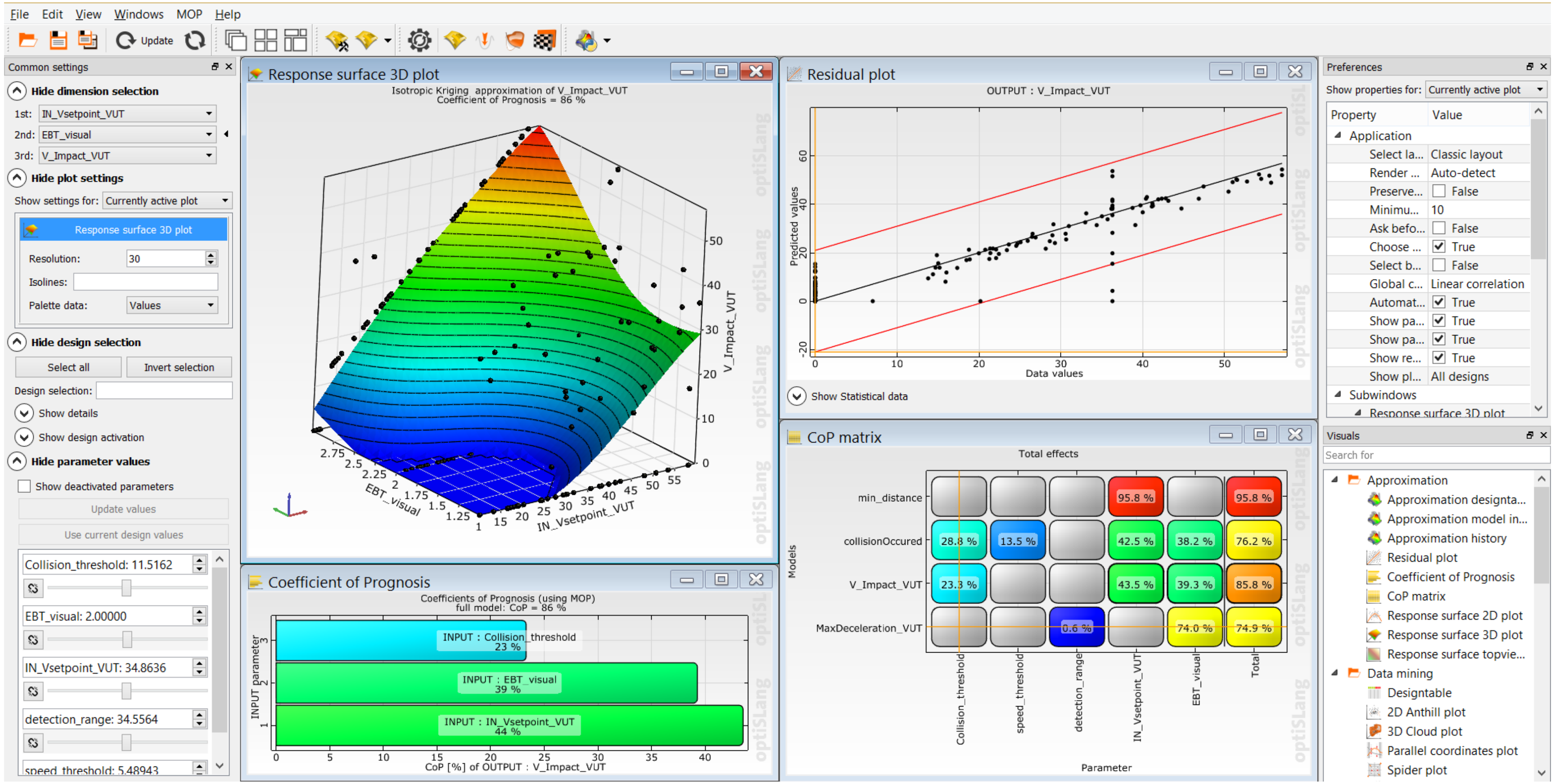
- distance to collision > 2m
- deceleration < 3 m/s²

VRX Driving Simulator & optiSLang: Example scenario (CBNA50)



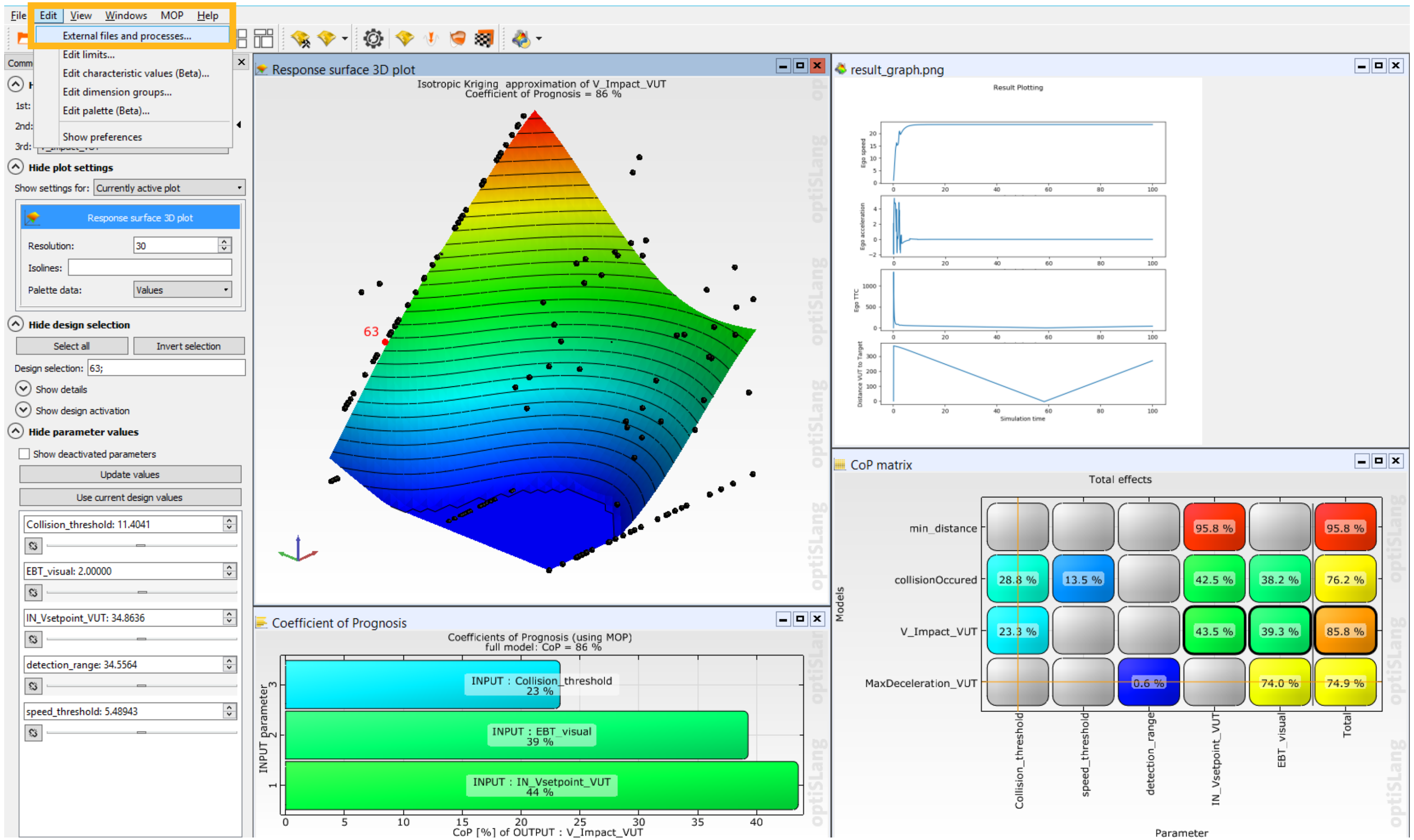
VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

- Interactive Postprocessing for data analysis



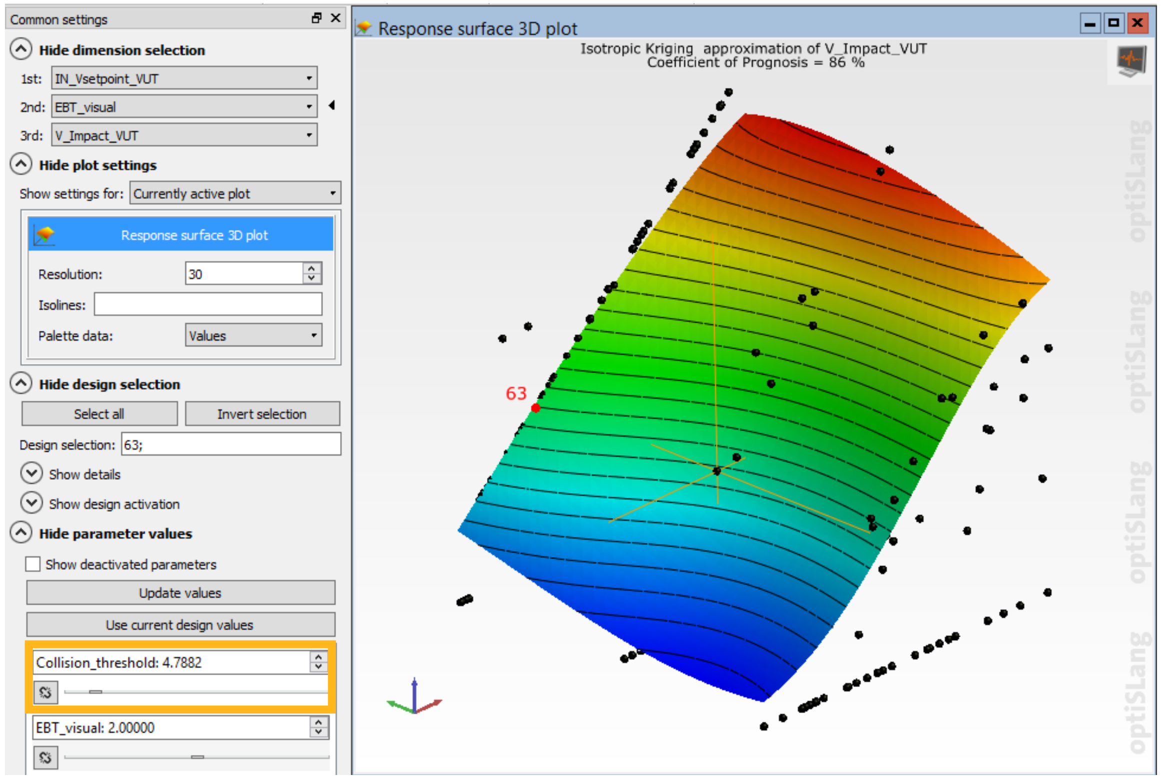
VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

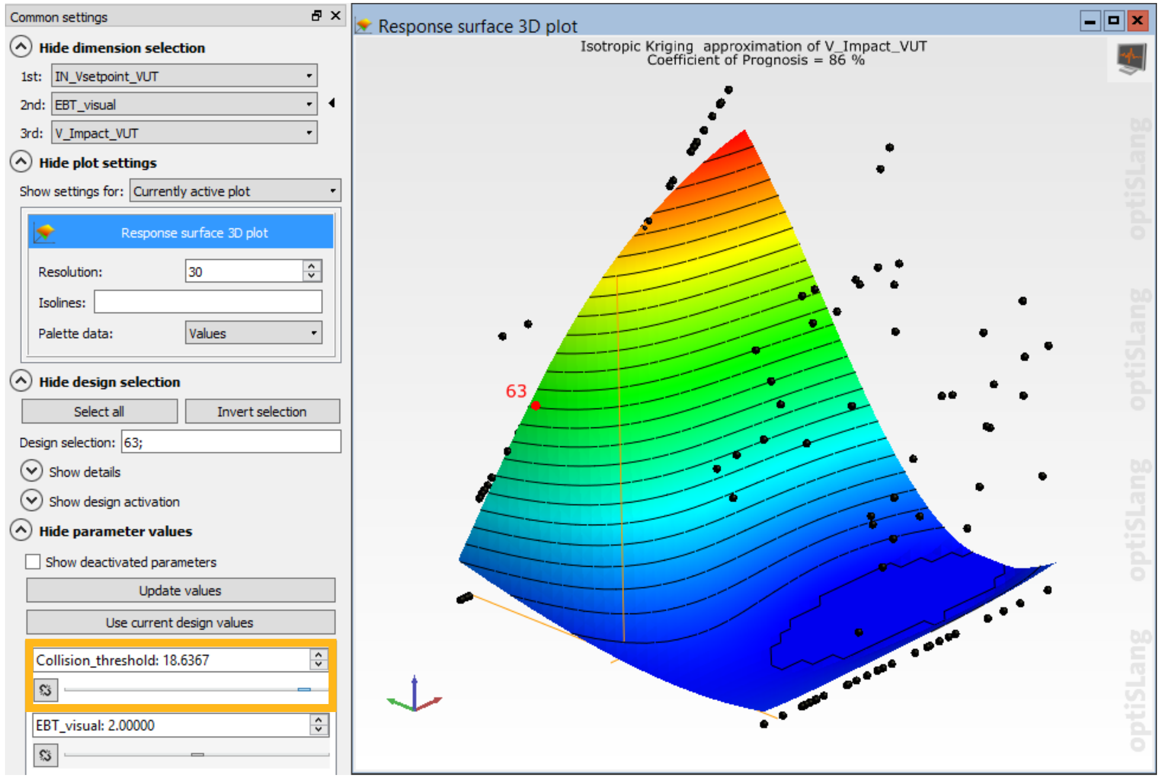
- Implement Images



VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

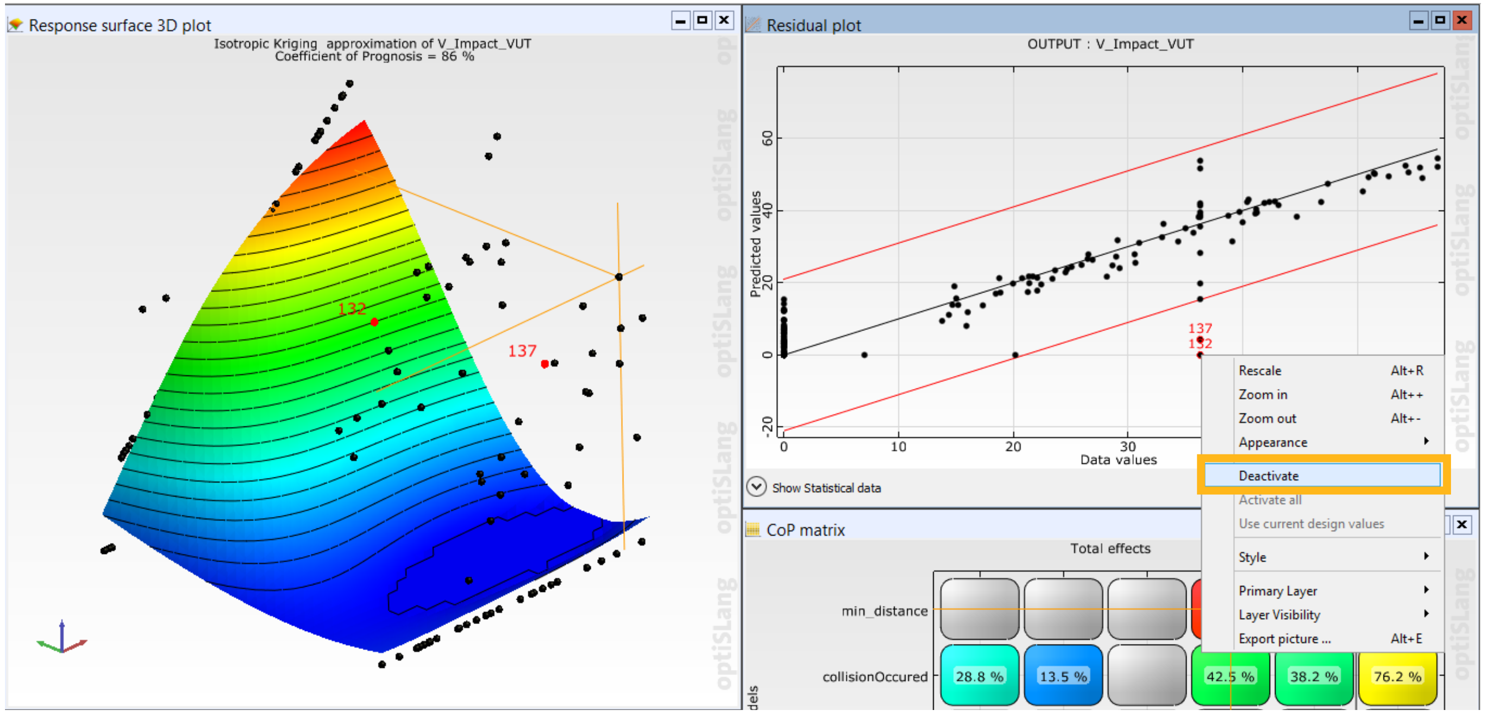
- Visualize the impact of the other parameters not illustrated in the 3D plot





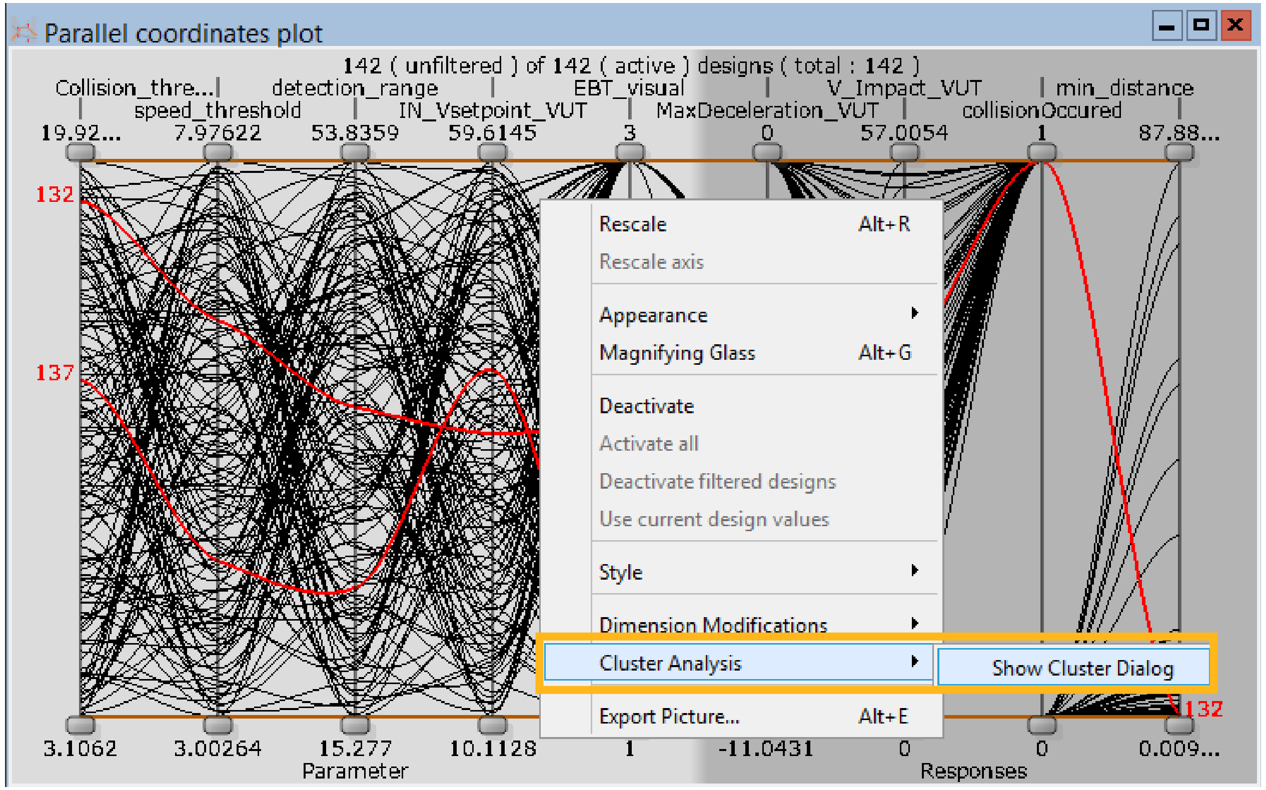
VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

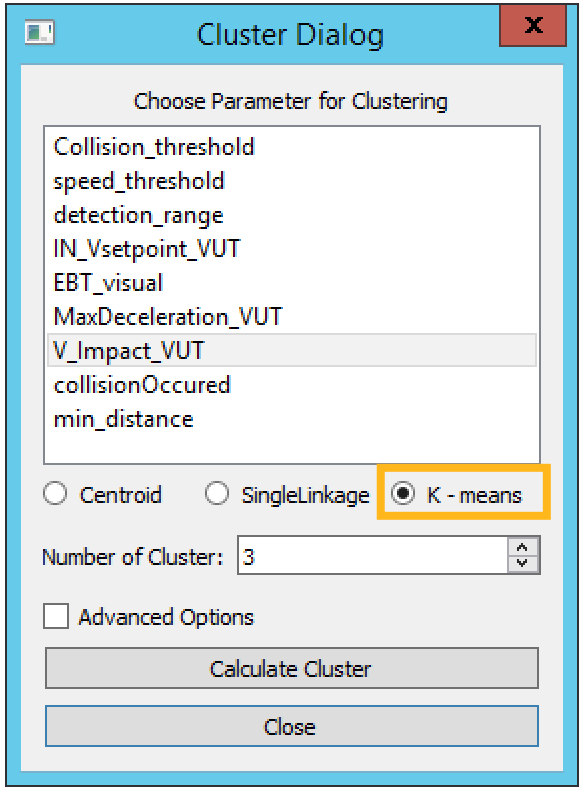
- Outlier detection & MOP generation



VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

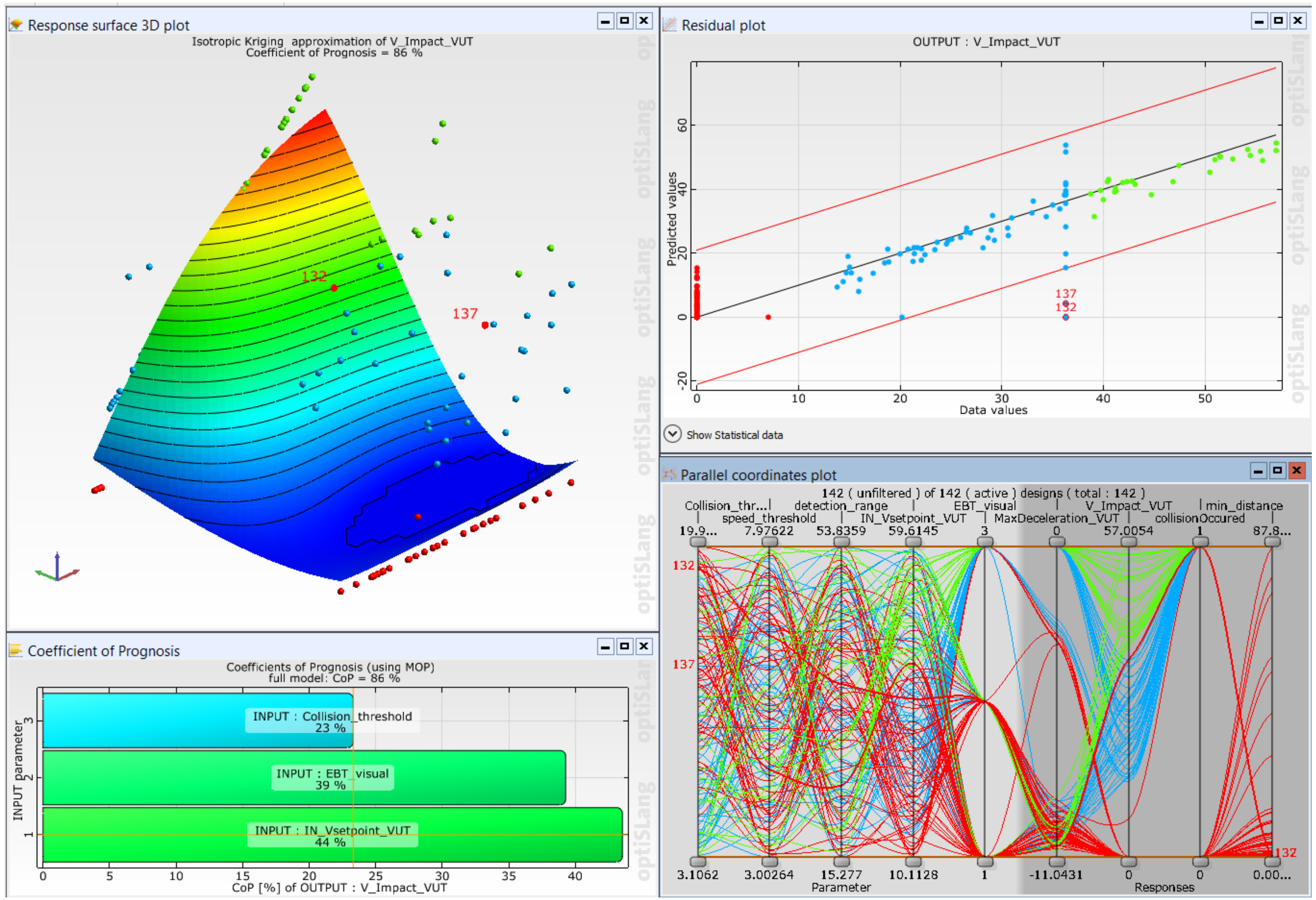
- Cluster analysis to detect correlations between input-output, output-output





VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

- Understand your design



Automatic Controller Calibration with VRX Driving Simulator and optiSLang

17th Weimar Optimization and Stochastic Days 2020

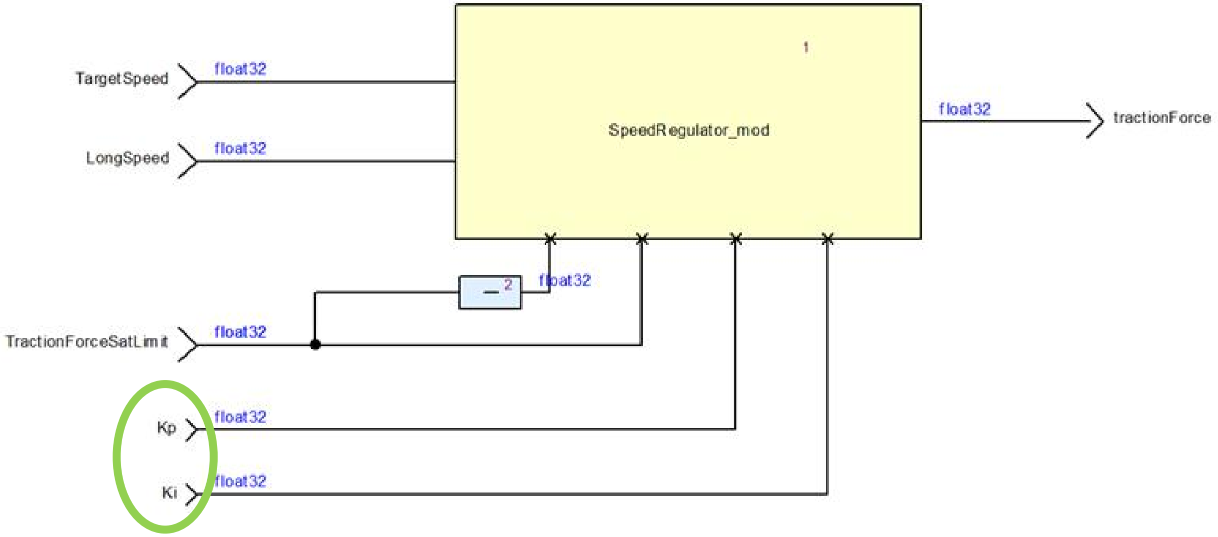
June 25 – 26, 2020

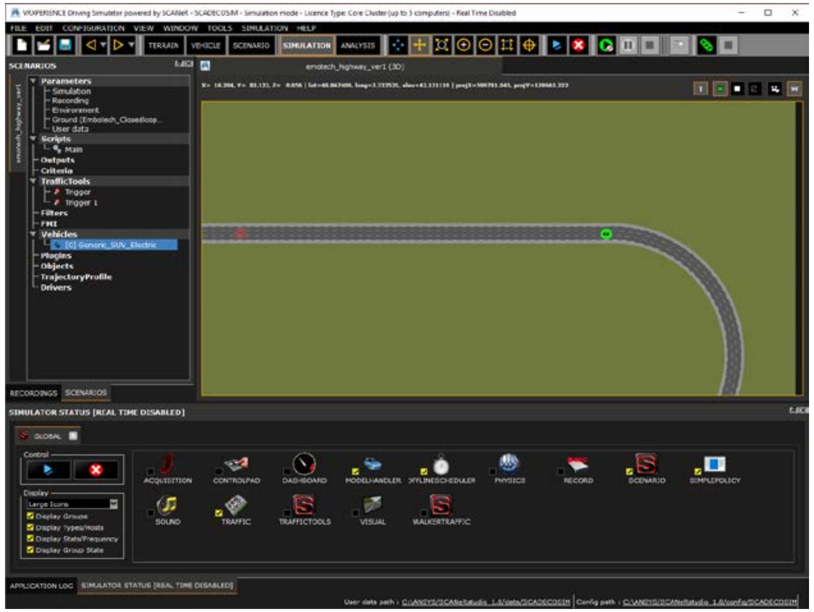
Virtual Conference

Ansys

Identify Calibration Parameters

- We begin with the original demo:
  - SCADE defines a deterministic model of the software controller
  - SCANeR defines the driving scenario with the controller in the loop
- We then select all SCADE variables to be used as calibration parameters

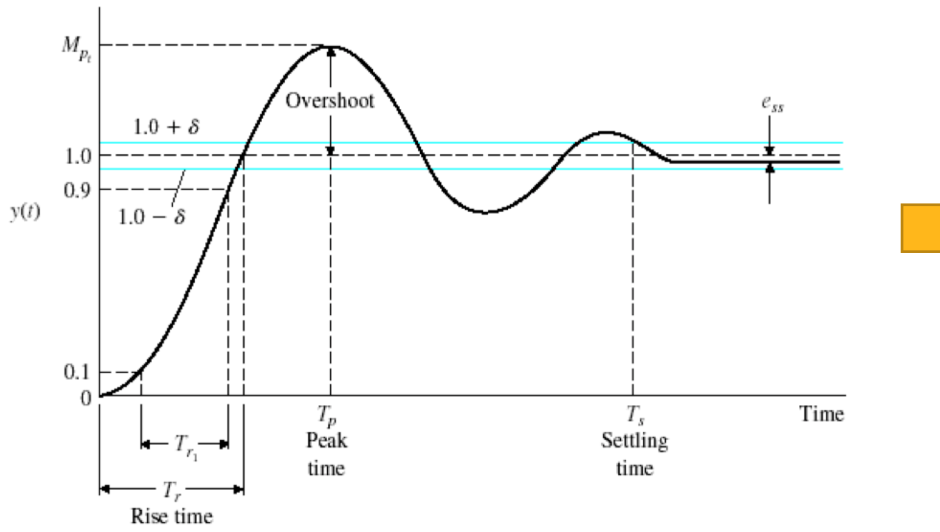


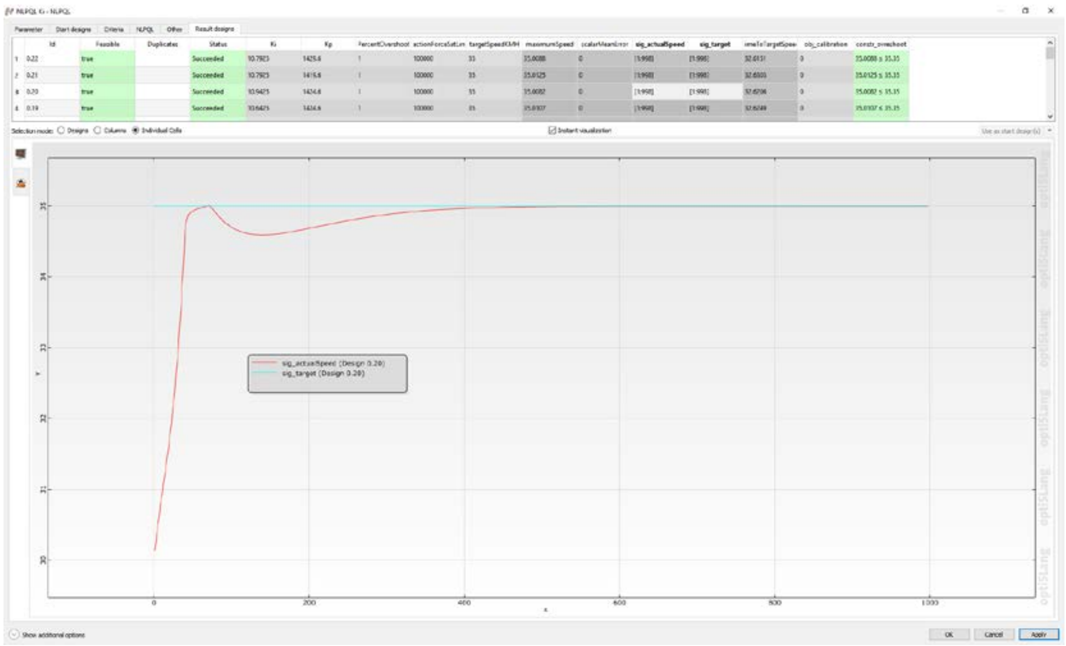


17th Weimar Optimization and Stochastic Days 2020
June 25 – 26, 2020
Virtual Conference

Define Criteria for System Response

- The controls engineer defines the desired characteristics for system response

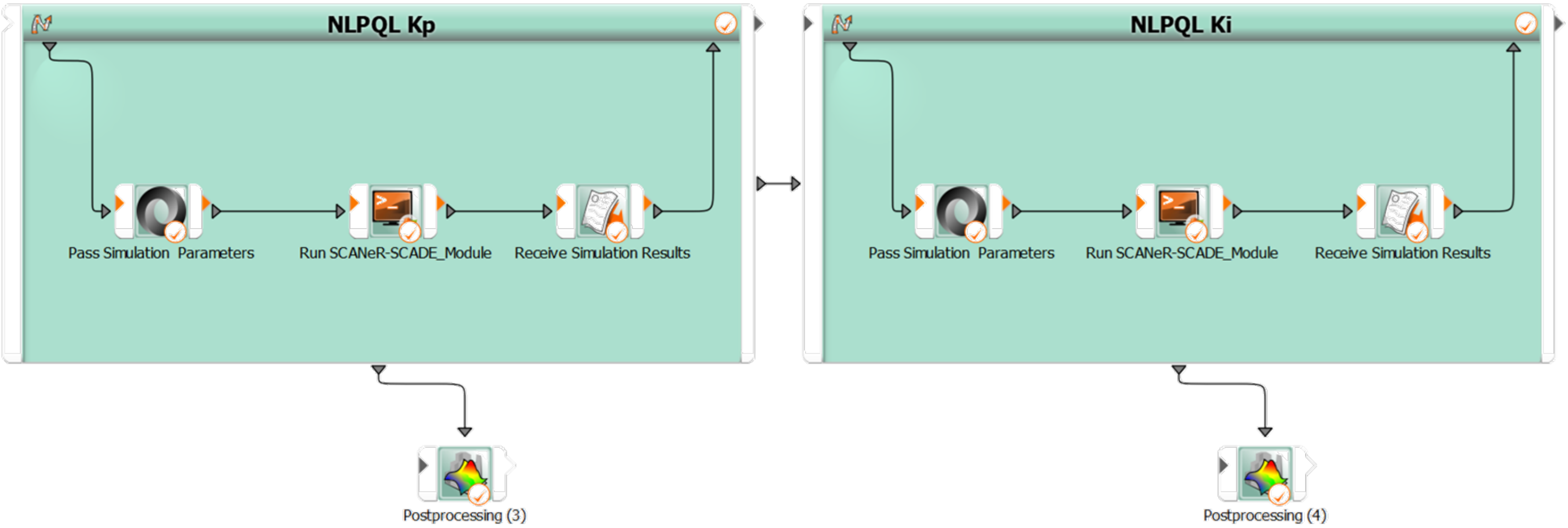




- These criteria are entered into optiSLang

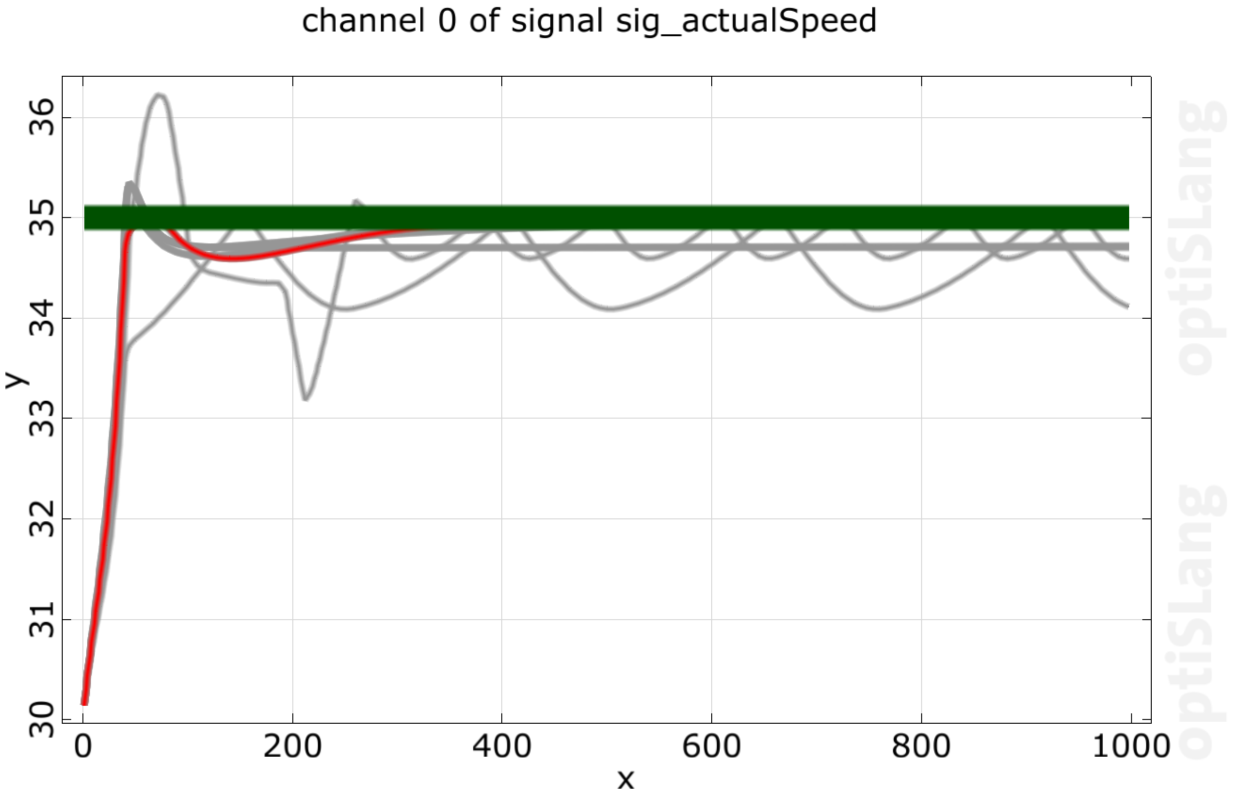
Define the Optimization Strategy

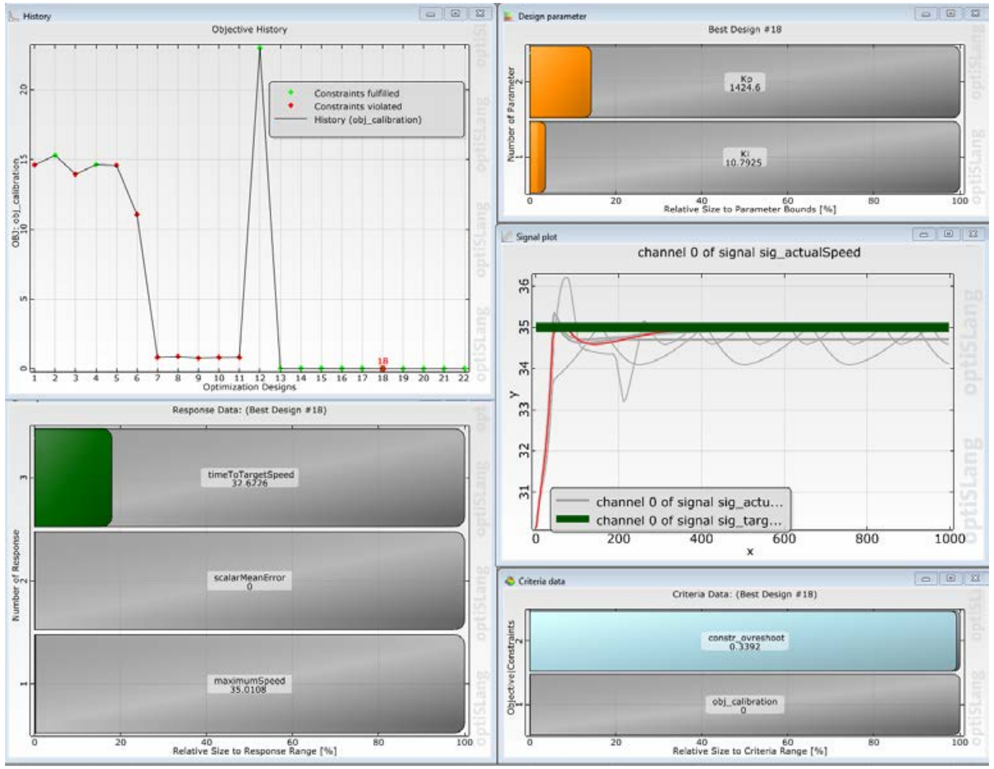
- optiSLang runs the driving scenarios in succession to compute optimal calibrations
  - The driving scenarios are set up to run in batch mode with “Drag-and-drop” integration
  - optiSLang provides a robust feature set of optimization methods to choose from
  - Our strategy in this example uses standard best practices to tune Kp first and then Ki



Execute the auto-tuning

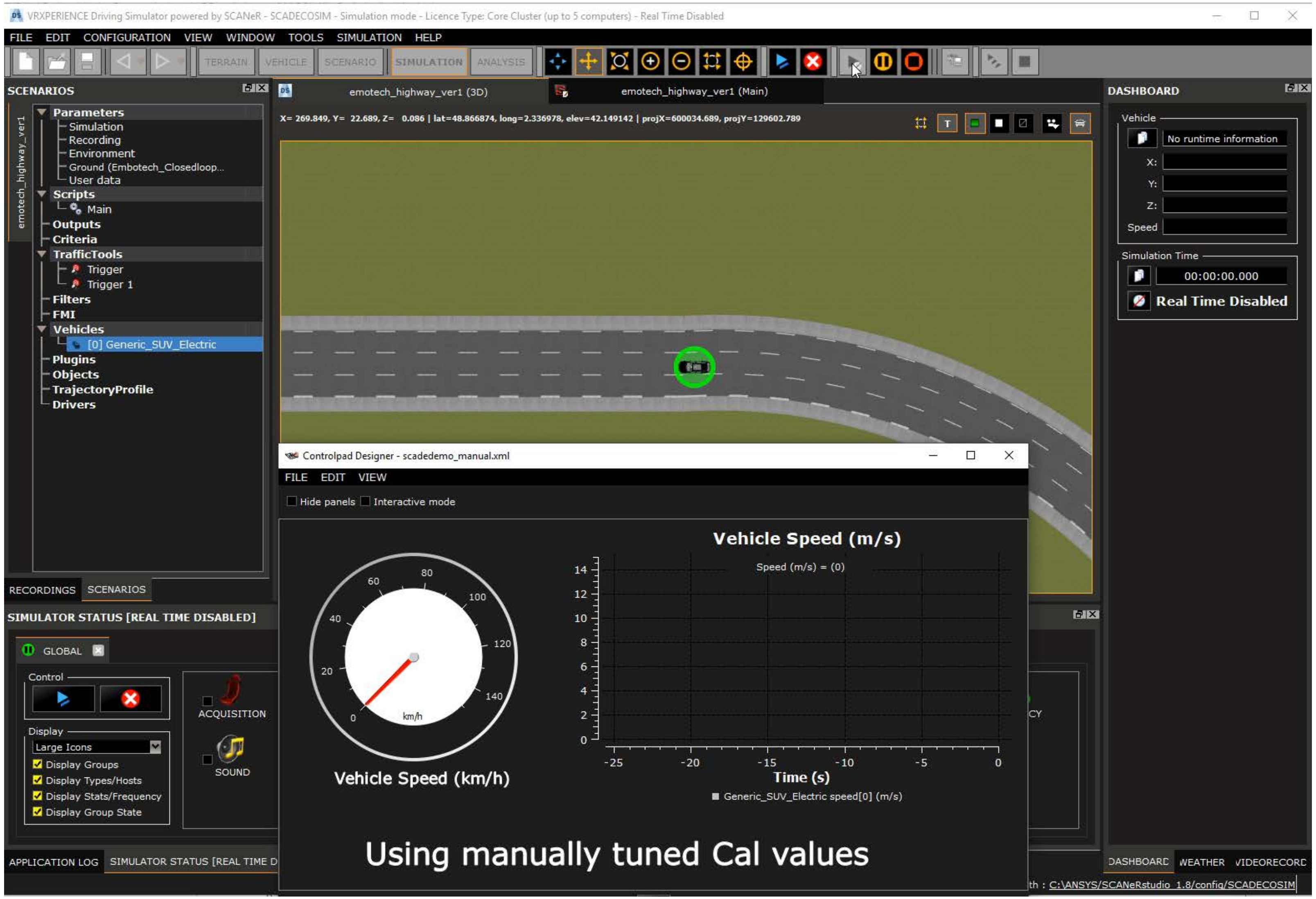
- optiSLang exercises the optimization strategy





- The best design is confirmed after 22 runs (total execution time ~= 10 minutes)
- More complex control laws scale up very well: an example with 10 cals took a few hours

Automatic Controller Calibration



Q&A

17th Weimar Optimization and Stochastic Days 2020

June 25 – 26, 2020
Virtual Conference

Ansys

Thanks !