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

Ralf Lampert, Senior Product Sales Manager, 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
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
Much wider breadth throughout the V-cycle

MIL or SIL on Workstation
- Preparation (scenario, ...)
- Quick testing of control software (policy) MIL or SIL

Driver Simulator
- Human in the loop testing
- AD L2/L3 - Driver take over and re-engagement
- Situational awareness

HPC / Cloud
- Campaign testing against Millions of scenarios
- Non regression testing

HIL
- ECU control software testing
- Non regression testing
- Validation
Addresses all aspects of an ADAS system
Moving from L2 to L3-L4 requires a technological quantum leap.

Developing L3-L4 mainly requires mastering safety challenges.
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.
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.
Validation of safety critical scenarios with Reliability Analysis
## Functional vs. Logic Scenarios

<table>
<thead>
<tr>
<th>Functional scenarios</th>
<th>Logic scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basis road:</strong></td>
<td><strong>Basis road:</strong></td>
</tr>
<tr>
<td>highway in bend</td>
<td>number of lanes</td>
</tr>
<tr>
<td></td>
<td>curve radius</td>
</tr>
<tr>
<td><strong>Stationary objects:</strong></td>
<td>[2..4]</td>
</tr>
<tr>
<td></td>
<td>[0.6..0.9] kph</td>
</tr>
<tr>
<td><strong>Movable objects:</strong></td>
<td><strong>Movable objects:</strong></td>
</tr>
<tr>
<td>ego, jam; interaction: ego approaches end of jam</td>
<td>End of jam position[10..200] m</td>
</tr>
<tr>
<td></td>
<td>jam speed</td>
</tr>
<tr>
<td></td>
<td>[0..30] kph</td>
</tr>
<tr>
<td></td>
<td>ego distance</td>
</tr>
<tr>
<td></td>
<td>[50..300] m</td>
</tr>
<tr>
<td></td>
<td>ego speed</td>
</tr>
<tr>
<td></td>
<td>[80..130] kph</td>
</tr>
<tr>
<td><strong>Environment:</strong></td>
<td><strong>Environment:</strong></td>
</tr>
<tr>
<td>summer, rain</td>
<td>temperature</td>
</tr>
<tr>
<td></td>
<td>[10..40] °C</td>
</tr>
<tr>
<td></td>
<td>droplet size</td>
</tr>
<tr>
<td></td>
<td>[20..100] μm</td>
</tr>
<tr>
<td></td>
<td>rain amount</td>
</tr>
<tr>
<td></td>
<td>[0..1..10] mm/h</td>
</tr>
</tbody>
</table>

© PEGASUS | VDA Technical Congress | April 6, 2017
How many miles ...

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

<table>
<thead>
<tr>
<th>Statistical Question</th>
<th>Benchmark Failure Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td><em><em>How many miles (years</em>) would autonomous vehicles have to be driven...</em>*</td>
<td><strong>(A) 1.09 fatalities per 100 million miles?</strong></td>
</tr>
<tr>
<td>(1) without failure to demonstrate with 95% confidence that their failure rate is at most...</td>
<td>275 million miles (12.5 years)</td>
</tr>
<tr>
<td>(2) to demonstrate with 95% confidence their failure rate to within 20% of the true rate of...</td>
<td>8.8 billion miles (400 years)</td>
</tr>
<tr>
<td>(3) to demonstrate with 95% confidence and 80% power that their failure rate is 20% better than the human driver failure rate of...</td>
<td>11 billion miles (500 years)</td>
</tr>
</tbody>
</table>

* 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

Probability distributions

Test case generation

Logical scenarios

Concrete scenario [i]

Simulation

Evaluation of safety metrics, e.g. TTC

Evaluation results

Estimation of probability of safety violations

Risk assessment, including avoidance criteria and comparison with human performance

Concrete scenario [i]

Result visualization

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.

Feedback to safety analysis
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

©2020 ANSYS, Inc.
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

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

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Design Space</th>
<th>Application</th>
<th>Main purpose and outcome</th>
<th>Probability of failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>“Optimization space” with equal distribution of input parameters (all controllable)</td>
<td>Early in design phase; software function <strong>testing</strong> during development</td>
<td>Exploration of the parameter variation space, Identification of software malfunction, consistency checks</td>
<td>not of interest here</td>
</tr>
<tr>
<td>Robustness</td>
<td>“Robustness space” with particular distribution of input parameters (controllable and not controllable)</td>
<td>Later in design phase; software function <strong>validation</strong></td>
<td>Estimate <strong>variation range</strong> of output parameters, Quantification of <strong>probability of failure</strong>, Identification of critical input parameters</td>
<td>is usually <strong>higher</strong> (10⁻⁶ and higher)</td>
</tr>
<tr>
<td>Reliability</td>
<td>“Robustness space” with <strong>precise</strong> distribution of input parameters (controllable and not controllable), Definition of a <strong>failure limit</strong> is crucial</td>
<td>Later in design phase, software function <strong>validation</strong></td>
<td>Quantification of <strong>probability of failure</strong>, Identification of critical input parameters</td>
<td>is usually <strong>very low</strong> (10⁻⁶ and lower)</td>
</tr>
</tbody>
</table>
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
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
Collaborative work possible because all information is everywhere available, here: input parametrization in Postprocessing

Determine concrete critical scenarios, export & visualize them
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
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

- 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

Reliability Analysis

- 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
Jam end scenario variation with optiSLang

Reliability analysis: Adaptive Sampling

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

Probability of Failure: 9.36376e-7
Standard deviation error: 5.148257e-8
Reliability Index: 4.76569

Number of designs
- Total: 5000
- Safe domain: 3219
- Unsafe domain: 1781
VRX Driving Simulator & optiSLang

17th Weimar Optimization and Stochastic Days 2020

June 25–26, 2020

Virtual Conference
**Scenario: Car-to-Bicyclist Nearsidside Adult 50% – CBNA 50**

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

<table>
<thead>
<tr>
<th>Scenario Name</th>
<th>CBNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of test</td>
<td>AEB</td>
</tr>
<tr>
<td>VUT speed [km/h]</td>
<td>10 - 60</td>
</tr>
<tr>
<td>VUT direction</td>
<td>Forward</td>
</tr>
<tr>
<td>Target speed [km/h]</td>
<td>15</td>
</tr>
<tr>
<td>Impact location [%]</td>
<td>50</td>
</tr>
<tr>
<td>Lighting condition</td>
<td>Day</td>
</tr>
</tbody>
</table>

---

![Diagram showing a car and a bicyclist in a collision scenario.](attachment:image.png)

©2020 ANSYS, Inc.
Optimization Task

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

Objective functions
- distance to collision > 2m
- deceleration < 3 m/s²
Scenario: Car-to-Bicyclist Nearside Adult 50% – CBNA 50

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

<table>
<thead>
<tr>
<th>Scenario Name</th>
<th>CBNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of test</td>
<td>AEB</td>
</tr>
<tr>
<td>VUT speed [km/h]</td>
<td>10 - 60</td>
</tr>
<tr>
<td>VUT direction</td>
<td>Forward</td>
</tr>
<tr>
<td>Target speed [km/h]</td>
<td>15</td>
</tr>
<tr>
<td>Impact location [%]</td>
<td>50</td>
</tr>
<tr>
<td>Lighting condition</td>
<td>Day</td>
</tr>
</tbody>
</table>
VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

• Interactive Postprocessing for data analysis
• Implement Images
- Visualize the impact of the other parameters not illustrated in the 3D plot
VRX Driving Simulator & optiSLang: Example scenario (CBNA50)

- Outlier detection & MOP generation
- Cluster analysis to detect correlations between input-output, output-output
- 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
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
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 calls took a few hours
Automatic Controller Calibration

Using manually tuned Cal values
Thanks !