Automated simulation of scenarios to guide the development of a crosswind stabilization function

Klaus-Dieter Hilf, Jochen Rauh (Daimler AG)
Ingo Matheis, Jakob Mauss (QTronic GmbH)

6th IFAC Symposium Advances in Automotive Control
July 12-14, 2010, Munich, Germany
**Motivation**

Ever growing complexity of automotive controllers

How to validate and test?
- do more road tests?
- write more test scripts?

This does not scale well
Code size grows faster
New processes needed

**Idea**

- increase degree of automation
- generate and evaluate useful test cases automatically

---

- After initial coding you can expect one bug per 20 lines of code
- After thorough unit testing you can expect 1 bug per 1000 lines of code in the final release
  - 1 line ~5 bytes, so 1 bug per ~5KB

<table>
<thead>
<tr>
<th>Application</th>
<th>Code Size</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steering Angle Sensor</td>
<td>32KB</td>
<td>7 Bugs</td>
</tr>
<tr>
<td>Low-end Sensor Cluster</td>
<td>128KB</td>
<td>26 Bugs</td>
</tr>
<tr>
<td>Airbag Controller</td>
<td>256KB</td>
<td>52 Bugs</td>
</tr>
<tr>
<td>EPS Controller</td>
<td>512KB</td>
<td>104 Bugs</td>
</tr>
<tr>
<td>Central Chassis Controller</td>
<td>1.5MB</td>
<td>308 Bugs</td>
</tr>
</tbody>
</table>

source: presentation by Hans Adlkofer, Infineon, 2009
Outline

- Motivation
- Principle of the scenario generator
- Validation of the crosswind stabilization function
  - the function
  - the validation setup
- Validation results and Conclusion
TestWeaver: Scenario Generator

Idea

• intelligent generation of 1000s of differing test scenarios
• active attempt to:
  - maximize state coverage
  - drive the system in “difficult” situations

Benefit

• high coverage
• low efforts for test specification

Testing = playing against (simulated) system
TestWeaver - Test Generation Strategy

- control input
- component fault
- wind and road model
- vehicle model
- control SW
- simulation: MiL, SiL or HiL

TestWeaver

- reactive scenario generation
  - each scenario depends on history of generated scenarios
  - all cases can be reproduced

- change sub-optimal scenarios to generate worst-cases
- drive the system in states that were not covered before

- inputs u
- outputs y
- discrete state
- alarm

- reached state
- alarm state

- discrete state space

- state DB

- test report
TestWeaver - Test Generation Strategy

Diagram:

- Root
- Fault 1
- Fault 2
- Input a
- Input b
- Input c
- Time

Events:
- T0
- Ok
- Fault 1
- Fault 2
- Alarm!
The Crosswind Stabilization Function (CSF)

change of wheel load imposed by ABC (incl. CSF)
resulting lateral tire force
yaw to compensate the wind
System model: vehicle with CSF

Matlab/Simulink

fast feedback

developer

control SW

vehicle model

Silver

virtual vehicle

CASCaDE vehicle simulation
System model: Road and wind

superposition of two bezier splines
control points generated "on the fly"
Automated validation

Objective: Validation of the CSF safeguard function
- activate CSF only in case of strong crosswind
- active for at most 4 seconds
- not activated by uneveness of road, steering actions or wheel slip

control SW
vehicle model
wind and road model

TestWeaver instruments

developer

TestWeaver

drive the virtual vehicle
generate road and wind
search for bad scenarios

test report

Silver
Validation results and Conclusion

Crosswind stabilisation

- generated and analyzed 100,000 different driving scenarios, each 45 sec. within 3 weeks
- Iterative improvement of the safeguarding function. See paper for details.
- Systematic fault injection, to validate fault tolerance

Conclusion

- The presented approach seems extremely well suited for the validation of complex automotive controllers
- Main benefit:
  - high test coverage
  - with low work effort