Industrial Applications and Challenges for Verifying Reactive Embedded Software

Tom Bienmüller, SC^2 Summer School, MPI Saarbrücken, August 2017

Agenda

- Who am I?
- Who is BTC Embedded Systems?
- Formal Methods in Automotive Industry
- Breathing industrial life into formal methods: expectations, applications and challenges
  - Today: (model-based) floating-point embedded software component development
  - Tomorrow: model checkers in industrial applications – how to gain end user‘s confidence
  - Day after tomorrow: autonomous driving – “minds off”!

Who am I

- Studied Computer Science at University of Oldenburg; diploma thesis on multi-threaded CPU exception handling (1997)
- Worked at the group of Prof. Dr. Werner Damm in Oldenburg; received doctoral degree in 2003 on optimizations of model-checking procedures for reactive systems
- Since 2003 in BTC-Embedded Systems – "lifting formal methods to industrial level" regarding performance, applicability, quality, and usability
- Responsible for Product Development and Quality Assurance for "BTC EmbeddedPlatform" product
    - Application implementing Formal Specification, Formal Verification, Automatic Test Vector generation

Who is BTC Embedded Systems?

Company established in 1999

BTC-ES Headquarter in Oldenburg

- Offices in Stuttgart, Berlin and Paris
- BTC Japan Co., Ltd.
- BTC Embedded Systems Romania SRL
- BTC China
- > 100 Employees worldwide

Expert for automatic Testing, Verification and Validation of embedded Software and Systems

- Automotive Domain
- Safety Critical & High Quality
- Relevant Safety Standard ISO-26262

BTC – official Test-Partner of

- dSPACE for Matlab/Simulink/TargetLink
- IBM Rational for Rhapsody-UML

Oldenburg
Germany
Berlin
Germany
Timisoara
Romania
Paris
France

Tokyo
Japan
Shanghai
China

Agenda

- Who am I?
- Who is BTC Embedded Systems?
- Formal Methods in Automotive Industry
- Breathing industrial life into formal methods: expectations, applications and challenges
    - Today: (model-based) floating point embedded software component development
    - Tomorrow: model checkers in industrial applications – how to gain end user‘s confidence
    - Day after tomorrow: autonomous driving – “minds off”!

Formal Methods in Industry: Toyota Japan

Toyota Prius, Application of both automatic test vector generation for B2B-Testing and formal verification against formal requirements

Shinichi Abe, General Manager of Toyota HV System Control Development Division: *“To complete the new process the development teams then selected dSPACE’s production code generator TargetLink in combination with the test and validation tools EmbeddedTester and EmbeddedValidator by TargetLink’s strategic partner BTC Embedded Systems AG. Adoption of this proven toolchain further increased the overall development efficiency and in the end allowed Toyota to produce all of the final control software’s code in-house – from TargetLink.”*

Source: http://techon.nikkeibp.co.jp/atclen/news_en/15mk/070500684

Formal Methods in Industry: MAN Germany

Overall MBD-Development Process, Application of both automatic test vector generation for B2B-Testing and formal Verification against formal requirements

Stefan Teuchert, Head of the Department Software-Development and Base Technologies, MAN Nutzfahrzeuge AG (Munich): *“MAN Nutzfahrzeuge AG successfully uses EmbeddedTester and EmbeddedValidator as a standard Automatic Test and Validation Environment for the leading AutoCode Generator TargetLink in the Model Driven Development of series-production Power Train applications.*

The automatic test generation, execution, analysis and debug capabilities of EmbeddedTester is one important key to fulfill the high efficiency and quality levels of MAN Nutzfahrzeuge AG, under the permanent time-to-market pressure.”

Formal Methods in Industry – further Success Stories

- Ford
- Wabco
- DENSO
- PSA Peugeot Citroën
- Deutz
- Claas
- …
- See BTC-ES web-site

Agenda

- Who am I?
- Who is BTC Embedded Systems?
- Formal Methods in Automotive Industry
- Breathing industrial life into formal methods: expectations, applications and challenges
  - Today: (model-based) floating-point embedded software component development
  - Tomorrow: model checkers in industrial applications – how to gain end user‘s confidence
  - Day after tomorrow: autonomous driving – “minds off”!

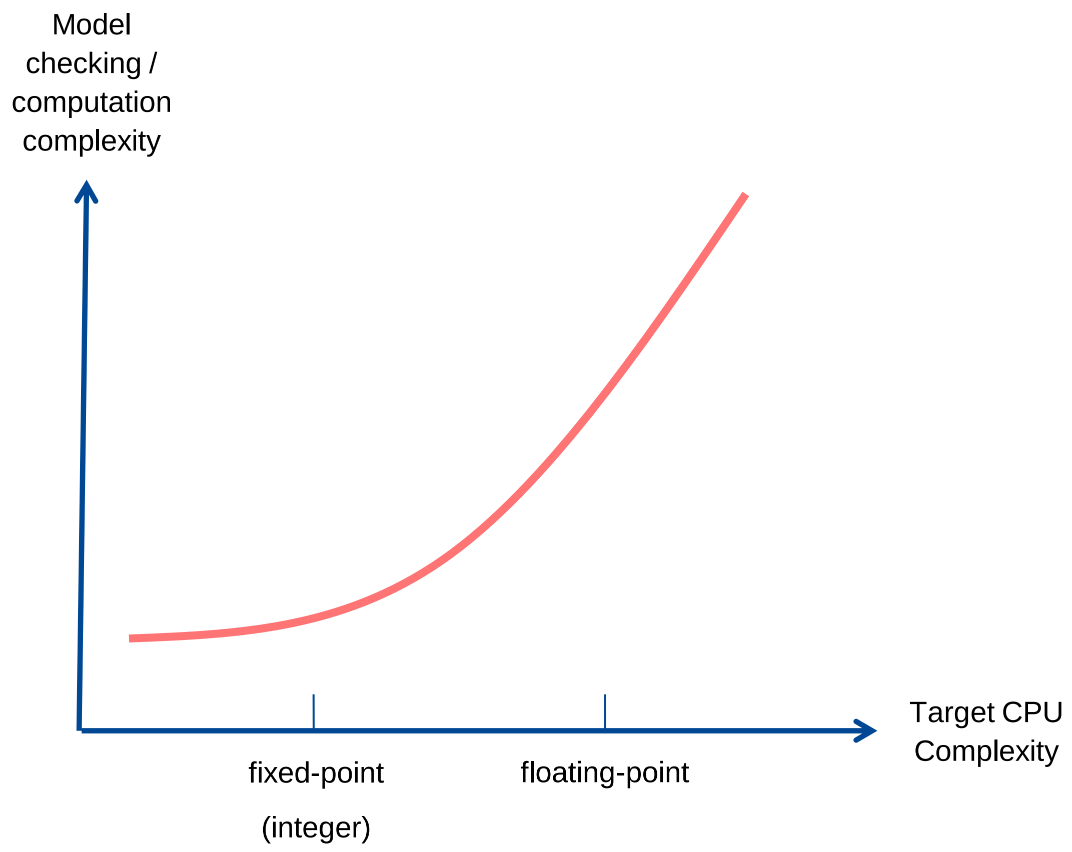
BTC-ES Motivation to join SC^2 – Network

Mission Statement "BTC Embedded Systems AG"

Our mission is to enable customers to increase product quality in a shortened design phase by introducing automatic test and verification technology to the model-based systems & software development process.

- We need to keep pace with increasing demands on formal methods and techniques from markets
  - **First SC^2 Motivation**: Regarding supported model class : IEEE-754
  - **Second SC^2 Motivation**: Regarding quality of / confidence in our tools for formal methods : Proof Certificates
  - **Third SC^2 Motivation**: Regarding potential combination of Symbolic Computation and Satisfiability Checking for verification of autonomous cars' systems

IEEE-754 Floating Point



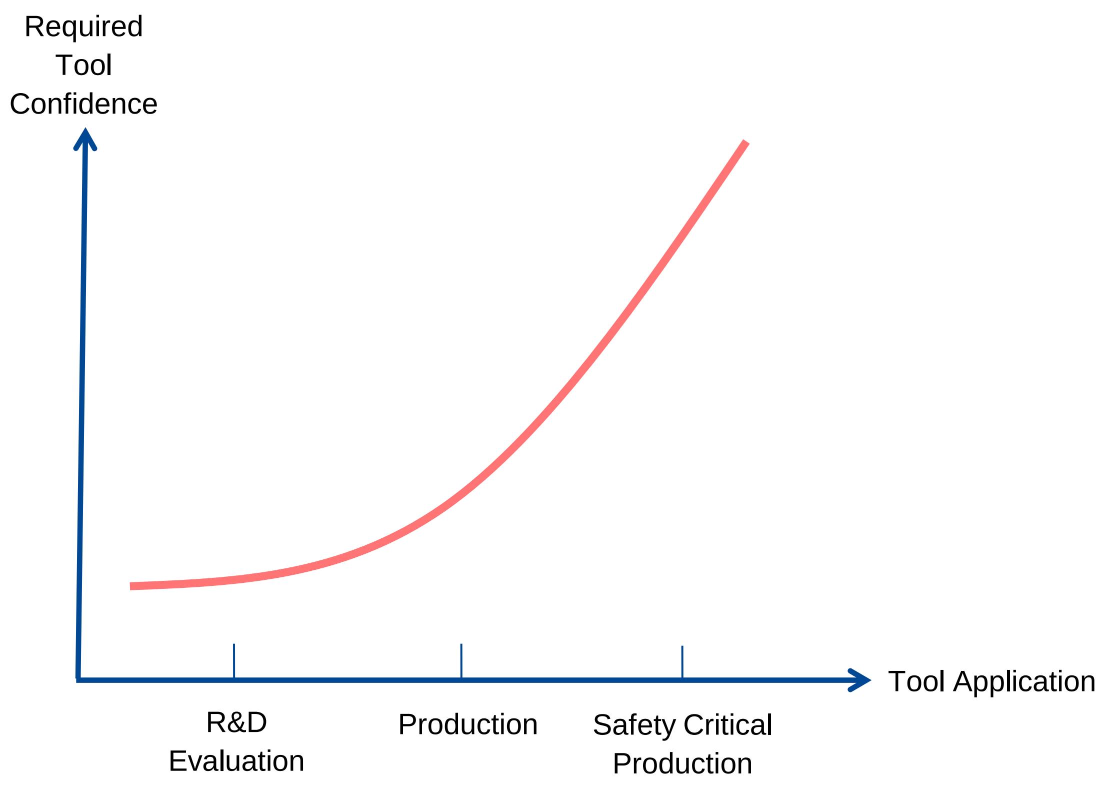
IEEE-754-based Floating Point Support – Motivation

- Embedded Software in Automotive domain still integer dominated
  - Fixed-Point approximation for real numbers
  - Model Checkers reducing formal verification to Boolean satisfiability work well for this type of models
- Reduced cost for CPUs containing floating-point-units
  - IEEE-754-based floating point approximation for real numbers
  - Almost all our customers (OEM, Supplier) have at least pilot projects
  - Model Checkers reducing formal verification to Boolean satisfiability do not always work sufficiently well for this type of models
    - Performance issue: bit blasting approach inefficient both in time and space
    - Approach "special solution for special problem": use SMT with native IEEE-754 support

IEEE-754-based Floating Point Support – SAT vs. SMT

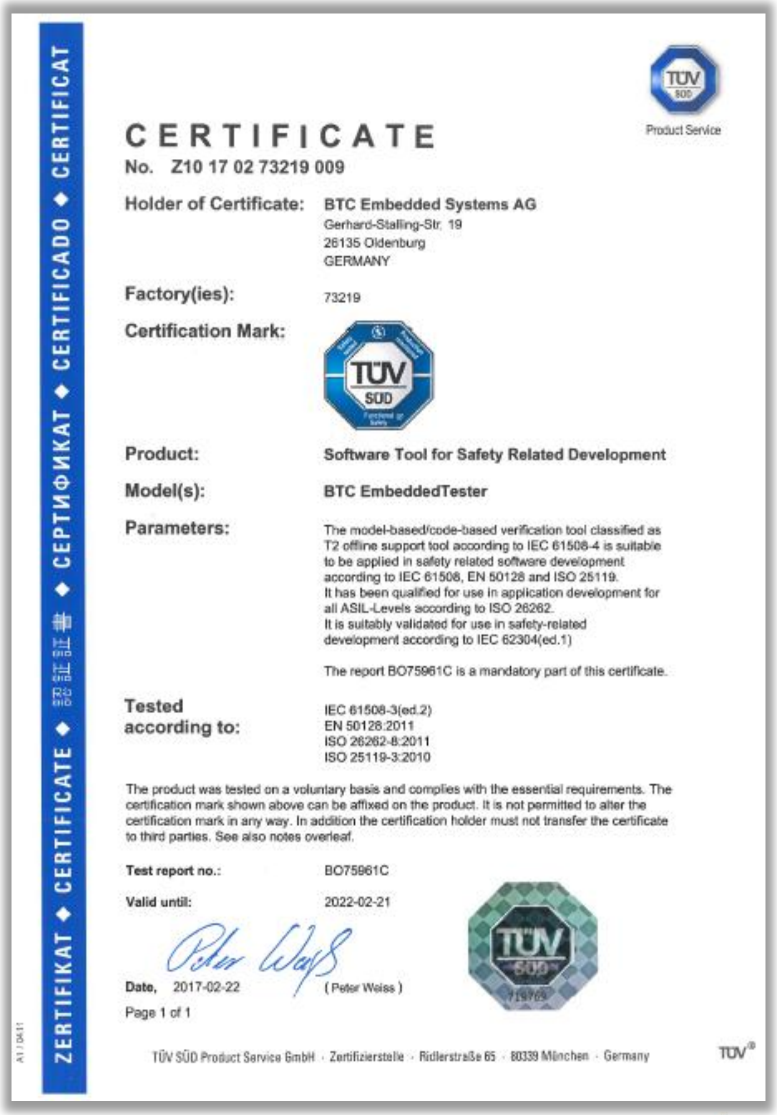
- SAT: floating-point Numbers and Operations reduced to bit level
  - fine-granular and bit-exact approach; induces complexity in space and in time (“bit blasting”)
- SMT: floating-point Numbers and Operations can be handled on arithmetic level
  - more abstract, thus efficient in space and more freedom in optimizations → may lead to efficient theory algorithms
- gap becomes more important for complex floating-point operations like mathematical functions:
  - 1.4 == exp(x) with double x in [-104.0, 89.0]
  - for SAT in our product: providing implementation of exp() reducing to standard operations → 700 LOC → cbmc solving time ~5min
  - Experimental for SMT: native support → iSAT3 solving time << 1sec (but on real arithmetic, no floating-point yet!)

Confidence in Model Checkers



Certified Software Confidence – ISO Requirements

- In ISO-26262, automating process steps by software tools requires “tool qualification”
  - “fit for purpose”
- Depending on addressed Safety Integrity Level, different measures to give evidence for “Confidence in the Tool”
- BTC-ES received tool qualification for Back-to-Back-Testing in 2010
- In 2017, use case “Formal Verification” shall be qualified



ZERTIFIKAT ◆ CERTIFICATE ◆ CERTIFICADO ◆ CERTIFICAT ◆ СЕРТИФИКАТ ◆ CERTIFICADO ◆ CERTIFICAT

Product Service

CERTIFICATE

No. Z10 17 02 73219 009

Holder of Certificate:	BTC Embedded Systems AG Gerhard-Stalling-Str. 19 26135 Oldenburg GERMANY
Factory(ies):	73219
Certification Mark:	
Product:	Software Tool for Safety Related Development
Model(s):	BTC EmbeddedTester
Parameters:	The model-based/code-based verification tool classified as T2 offline support tool according to IEC 61508-4 is suitable to be applied in safety related software development according to IEC 61508, EN 50128 and ISO 25119. It has been qualified for use in application development for all ASIL-Levels according to ISO 26262. It is suitably validated for use in safety-related development according to IEC 62304(ed.1) The report BO75961C is a mandatory part of this certificate.
Tested according to:	IEC 61508-3(ed.2) EN 50128:2011 ISO 26262-8:2011 ISO 25119-3:2010

The product was tested on a voluntary basis and complies with the essential requirements. The certification mark shown above can be affixed on the product. It is not permitted to alter the certification mark in any way. In addition the certification holder must not transfer the certificate to third parties. See also notes overleaf.

Test report no.:	BO75961C
Valid until:	2022-02-21

Date, 2017-02-22 (Peter Weiss)

Page 1 of 1

719769

A1 / 04.11

TÜV SÜD Product Service GmbH · Zertifizierstelle · Ridlerstraße 65 · 80339 München · Germany

Quality Expectation for Model Checkers

- End user’s perception:
  - Phrase 1: “Everybody knows that each software contains bugs”
  - Phrase 2: “Model Checkers detect all bugs which are in a software”
- End user’s conclusion:
  - “When we apply model checkers, our software does not contain bugs”
- But Model Checker’s are made of software … so they contain bugs
  - A “false negative” is no problem – can verify by simulation
  - But what about a “false positive”?
    - ***A “false positive” is a big problem … the model checker gave the wrong ultimate answer “no bug found”!***

Model Checkers in Software Production

- More and more applications of formal verification techniques in automotive embedded software production development
- Semantic Bugs in formal verification tools have tremendous effect on end-user's processes, product quality and reputation
  - Users trust and rely a lot on such tools. Very high reliability expected.
  - Single bugs may lead to huge additional cost when iterations are needed, not counting issues occurring in the field
  - Loss of reputation for users and tool vendors even for single bugs
  - … even though everybody agrees that no software is free of bugs, formal verification tools need to be (almost) 100% reliable
- Customers already requested ISO-certification of BTC-ES formal verification tools. This requires to achieve "high tool confidence level"

What can we do to achieve close to 100% confidence in formal Verification Software Tools? Each individual bug counts!

Two complementary Approaches

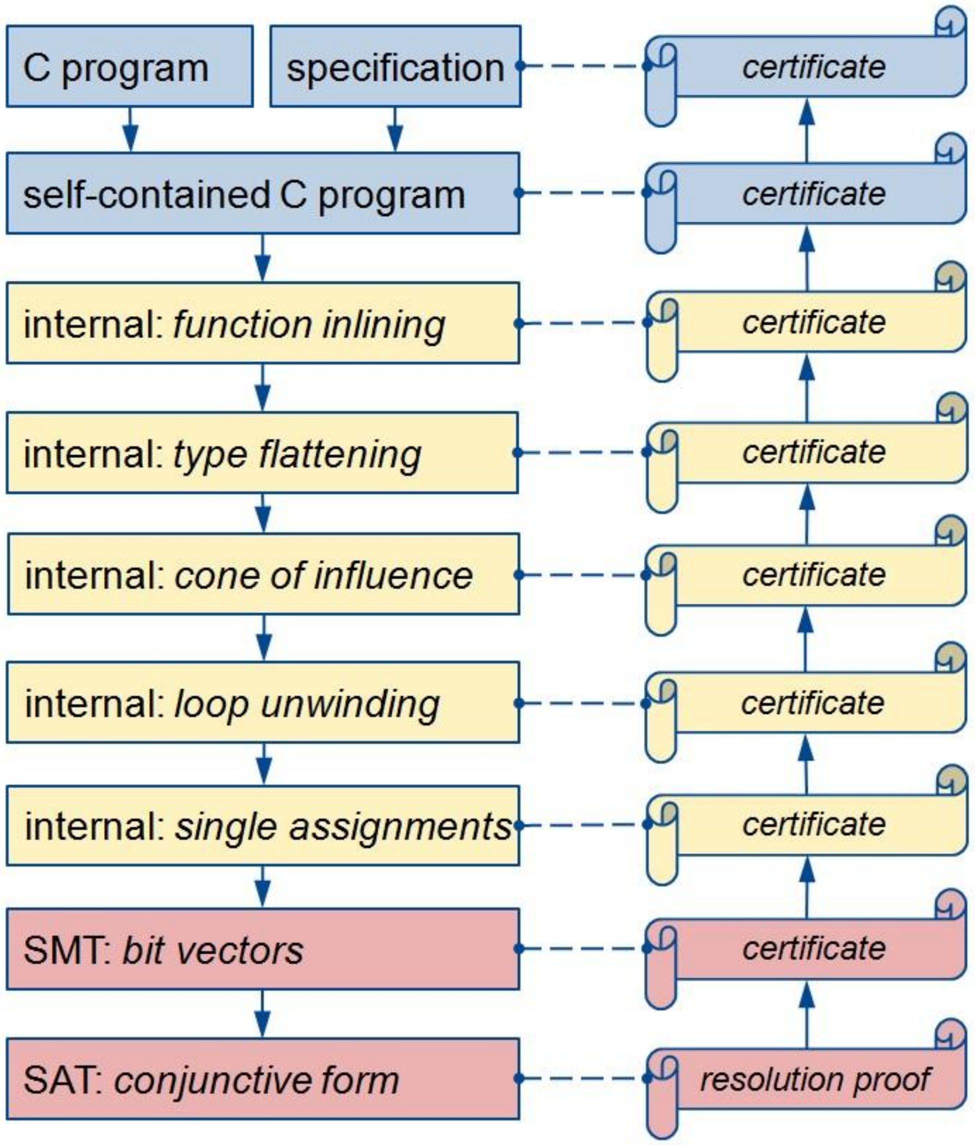
- Offline Quality Assurance : traditional QA during Software Development incl. highly sophisticated testing environment
  - Huge amount of (customer) models
  - Back-to-back test between different model checkers when possible
  - ...
- Online Quality Assurance : check the results of formal verification Tasks online within end-user's environment
  - Easy and established for counterexamples / witnesses : probe by post-simulation against end-user's model/design
  - Hard and not established for certifications / true- / unreachable- Results
- Goal: get Confidence by "99% offline QA and 1% online QA"

BTC-ES requires support by academic experts for online QA of "the specification holds"-results

Idea on how to address Online-QA – Proof Certificates

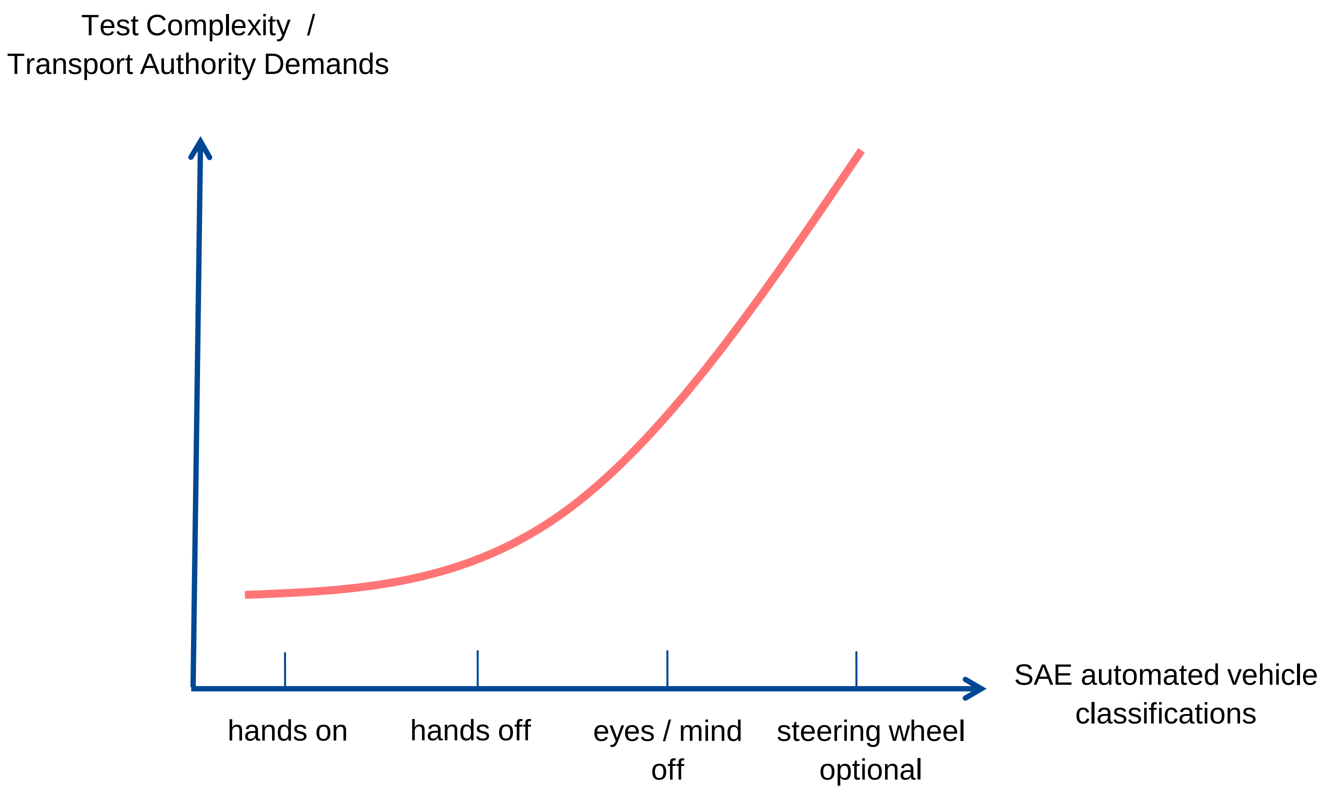
- Find reliable (approximated) solutions to the online-detection of wrong true-results
- Additional requirements to potential solutions
  - Needs to be embedded in workflow without introducing additional user interaction
  - Ideally, obtain "Proof Certificates" on the level of the input language to the verification process (production C-code; see next slide)
  - Certification technology shall not increase verification time too much
    - Take user-acceptance into account
- One straight forward approach to online QA: why don't we just simply apply a second model checker to gain trust?
  - Requires to have second implementation, which vendor does not have
  - Requires "doubled" time and space, what user's environment does not have
  - And we believe one can do better …

Proof Certificates – Conceptual Sketch



Verifying Autonomously Driving Cars –

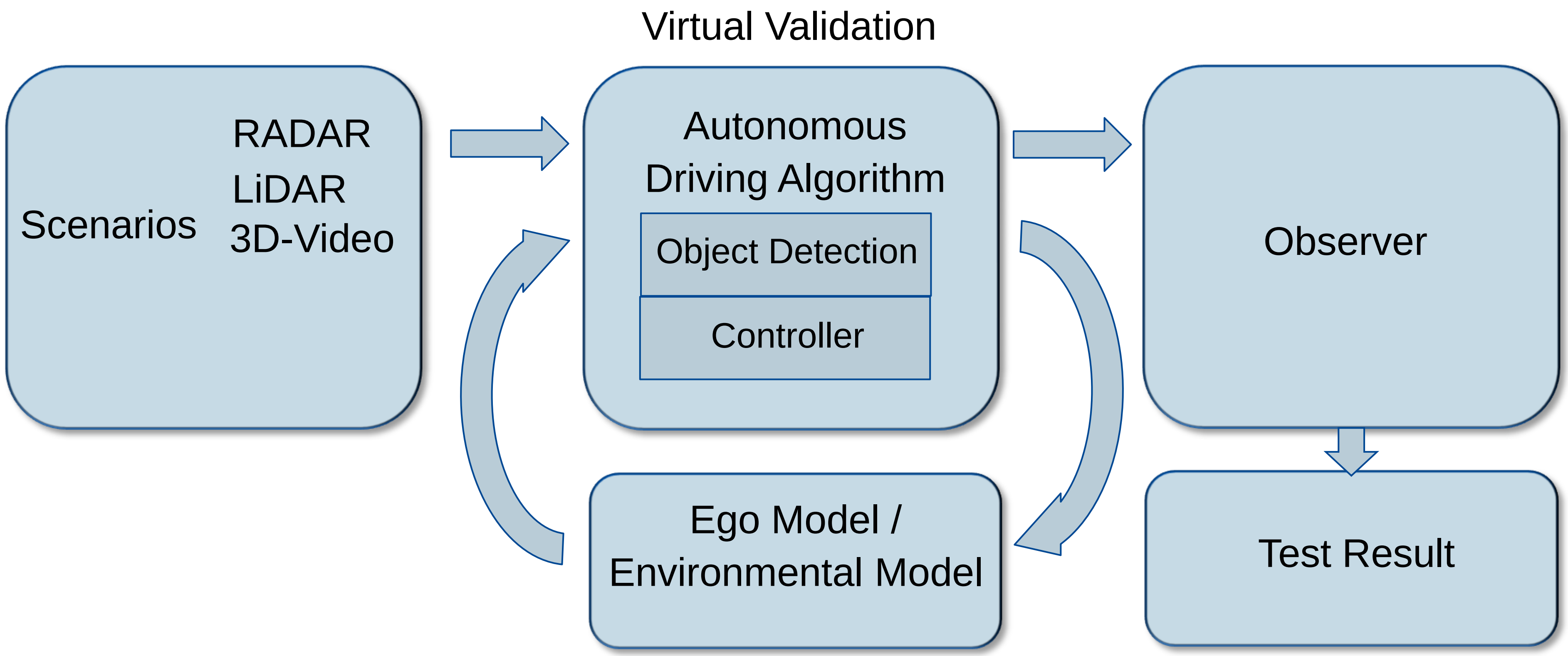
Combining Symbolic Computation with Satisfiability-Checking?



Autonomous Driving – Observations

- Getting the same confidence in an autonomously driving car's function by traditional testing methods would require to go over 80-times to the sun and back (following Hermann Winner, expert for car technology)
  - More that 13 billion kilometers
- This will not be feasible. Hence, changed approach:
  - based on virtual validation and "intelligent" definition of driving scenarios plus parallel observation of safety goals, traffic rules etc
  - Requires dedicated coverage criterion to get test-exit criteria
    - Requires to convince transport authority / standardization authorities
  - get some statistical argument enabling to derive needed sufficient amount of driving kilometers (bringing the need to "really drive" down to a reasonable number again)
- artificial intelligence meets embedded software development
  - Stage 1 (e.g., highway driving): object detection uses AI algorithms, controller developed the traditional way
  - Stage 2 (e.g., urban driving): also controller contains AI-components?

General Test Architecture (Stage 1)



- “Object Detection” is based on AI-algorithms (“Deep Learning”)
- Controller is traditionally developed without any AI aspects
  - This ensures same reaction upon same driving situation

Challenges: Verification of AD-Algorithm

- Formal verification of safety goals would require to deal with discrete controller algorithm, but also with continuous systems like the ego car or the environment (other cars, human beings etc)
  - Combine Satisfiability Checking (→ controller) with Symbolic Computation (ego-car, environmental behavior)?
- Verification by “sufficient testing”
  - One approach for SC^2: generate test cases automatically using verification technology (using the same combination of SC*SC as mentioned above)
  - Requires further ingredients like stochastic argument for test exit criterion
- When Controller is implemented also using AI algorithms (“Stage 2”): reuse symbolic computation algorithms for trained artificial intelligence implementations?

Summary

- Formal Methods reached real (safety) embedded software production
- Providing usable tools requires
  - dealing with increased demand comes along with more sophisticated Target CPUs (→ IEEE-754 Floating-Point)
  - End user‘s having confidence in the correctness of tools‘ output (→ proof-certificates; tool qualification)
- Innovations such as autonomous driving requires to re-think traditional verification and validation strategies
  - ***Today’s industrial testing and formal verification approaches will not always be feasible!***
  - Artificial intelligence meets safety critical systems development