Real-Time Simulation Environment for Autonomous Vehicles in Highly Dynamic Driving Scenarios

Thomas Herrmann, Technical University of Munich
Michael Lüthy, Speedgoat GmbH

MathWorks Automotive Conference 2019, Stuttgart
About Roborace

Roborace is a development platform for algorithms powering self-driving cars

History

➤ Third FormulaE series → a support series called Roborace added
➤ Races on Formula E tracks
➤ Provides the first racing series for autonomous vehicles
➤ Teams taking part only develop the software for the provided autonomous Robocars
➤ The Human + Machine Challenge in 2018 in Berlin

Future

➤ 2019: 4 x competitions planned (Monteblanco, Berlin, …)
➤ Several cars on one track
Video: Human + Machine Challenge 2018 in Berlin
Roborace: TUM Team Structure

Prof. Dr.-Ing. Markus Lienkamp  
Chair of Automotive Technology

Prof. Dr.-Ing. Boris Lohmann  
Chair of Automatic Control

Johannes Betz  
Alexander Heilmeier  
Alexander Wischnewski  
Tim Stahl

Leonhard Hermansdorfer  
Thomas Herrmann  
Felix Nobis
Roborace: Motivation for TUM

Know How:
- Artificial intelligence algorithms
- Sensor fusion
- Control
- Automotive technology

Research:
- PhD thesis
- Publications
- Student thesis

Road relevant research:
- Real traffic scenarios
- Static and dynamic objects
- Different road quality and road surfaces

Teaching:
- New lectures
Roborace & TUM: Milestones

- Software development started in January 2018
- Focus → reliable algorithms and rapid prototyping
- DevBot uses LIDAR and GPS to build the map in advance and localize
- Achieved 150 kph at ~80% of the maximum friction level fully autonomously
- Collaboration with Speedgoat to enable Real-Time HIL set-up
Roborace: Vehicles

Development car Devbot
- Based on an LMP chassis
- Can be driven manually

Racing car Robocar
- Developed from scratch for fully autonomous racing
- Nearly all internal components are equivalent to DevBot

Both vehicles equipped with
- 5x LiDAR, 6x Camera, 2x RADAR, 17x Ultrasonic
- Optical Speed Sensor, Vehicle Dynamic Sensors
- 4x Electric motors
- 62 kWh Battery
Roborace: ECU Hardware Set-Up

- **Hardware**: Nvidia Drive PX2
- **Software**: Perception → Planning → Control
- **Software Language**: C++, ROS, Python, MATLAB/Simulink
- **Interface**: ØMQ, UDP, Ethernet

Mobile real-time target machine
Roborace: ECU Software Set-Up

**Offline (PC)**
- LiDAR + GPS
- Mapping
- Map Processing
- Time Optimal Path
- Energy Strategy

**NVIDIA Drive PX2**
- Precomputed Map
- Friction Estimation
- LiDAR Localization
- Behavior State Machine
- Global Race Trajectory
- Local Trajectory Generation

**Speedgoat**
- Odometry
- Sensor Fusion
- Trajectory Tracking Controller & Vehicle Dynamics Control

**Legend**
- ROS
- Raw UDP
- Internal
- Future Feature

---

MathWorks Automotive Conference 2019, Stuttgart
Testing & Validation

Due to limited testing time on track other means are required to test and validate functions such as:

» The inter-component communication

» Behavior on
  » Static/dynamic obstacles
  » Safety/emergency stops
  » Sensor failures

» Identify performance potentials regarding
  » Processor loads
  » Parametrization of algorithms with realistic noise

» Train startup-logic and evaluate working procedures
Software Module Testing

Perception Unit (NVIDIA Drive PX2)
- Use pre-recorded LIDAR data to test
- Check mapping and localization algorithms
- Check interfaces using NVIDIA Drive PX2

Control Unit (later: Mobile Real-Time Target Machine by Speedgoat)
- Control software simulated in Simulink
- Same interfaces to vehicle model as in real car
- Driven by replaying recorded trajectories
Software Module Testing

Software-in-the-Loop
Communication realized via same interface as in real car (UDP)
Limitations & Challenges of Current Testing Methods

› Testing on track:
  › Track-time is very limited
  › Covering all safety relevant scenarios difficult
  › Testing algorithms in safe environment is key
  › Inconvenient debugging

› Testing using recorded data from previous runs
  › Interaction between vehicle and changes in the algorithms not testable
  › Limited variety of test scenarios implementable

› Transfer of research to generic AD- and ADAS application development
Using a HIL System to Simulate the Vehicle & Environment

- Closed-loop testing in virtual environment
  - Test interaction between the real ECU and vehicle already in the lab
  - Test safety relevant functions in the virtual environment first
  - Generate dynamic sensor realistic feedback to test algorithms

- Real-time HIL system required that can
  - Run vehicle & low-level ECU models in real-time & deterministic
  - Communicate with the ECU using the same interfaces
  - Simulate the environment and various sensor feedbacks to interact with ECU

- Ideally: spend as little time as possible on model adaption and code deployment
- Solution: Speedgoat real-time systems & Simulink Real-Time!
About Speedgoat

- Real-time solutions designed specifically to work with Simulink
- A MathWorks associate company
- Incorporated in 2006 by former MathWorks employees in Switzerland
- Headquarter in Switzerland with subsidiaries in the USA and Germany
- Real-time core team of around 100 people within MathWorks and Speedgoat
- Working closely with the entire MathWorks organization worldwide and its over 4000 employees
How it Works
Real-Time Simulation and Testing

Simulink Real-Time by MathWorks
- Code Generation (C/VHDL)
- Toolboxes/Blockset Support
- Real-Time Instrumentation
- Simulink Test

Real-Time Target Machines by Speedgoat
- I/O modules / protocol support
- FPGA-based solutions
- Speedgoat driver library
- Complete hardware-in-the-loop rigs

Simulink Real-Time Kernel
Fast-Track from Desktop to Real-Time Simulation and Testing
Automatically create your real-time application from Simulink

1. Automatic C or VHDL Code Generation
2. Compile and Synthesize
3. Download and Ready to Run

Target machine with multicore CPU, FPGAs, and I/O

"With limited resources as a start-up, the time to get things working is really key. Speedgoat excelled for us." -- ClearMotion
Speedgoat Real-Time Systems Ideal for Automotive Applications

▷ Different real-time systems to meet your requirements:
  ▷ Mobile: Performance System ideal for in-vehicle use
  ▷ Performance: High Performance for most demanding applications
  ▷ Complete Racks: Customized to your specific requirements!

MathWorks Automotive Conference 2019, Stuttgart
Speedgoat Real-Time Systems: Ideal for Automotive Applications

- I/O connectivity and communication protocols support:
  - Vehicle communication (CAN, CAN-FD, LIN, J1939, FlexRay, PSI5 and many more)
  - HIL testing platform for battery management systems
  - ....

- High-performance FPGAs for most demanding applications
  - HDL coder support
  - FPGA code module support (CAM, CRANK, PWM and many more)

- Seamless integration of Simulink toolboxes and blocksets
  - Vehicle Dynamics Blockset
  - Simscape
  - Powertrain Blockset
  - ....
Speedgoat Real-Time Systems: Fast Track Your Development!

Lots of customers already rely on Speedgoat systems to fast track their development efforts regarding:

- Automated Driving  ADAS
- Electrification
- Battery Management Systems
- Engine & Drivetrains
- Drive-By-Wire
Visit Us at our booth "To be defined"
Vehicle ECU

**Code Integration Workflow - HIL**

*Vehicle ECU*

**Vehicle & Environment Simulation**

- **NVIDIA DRIVE PX2**
  - Perception & Planning

- Trajectories, real-time UDP

- **Speedgoat Mobile**
  - Sensor Fusion
  - Control

- CAN

- **Speedgoat Performance**
  - Vehicle Dynamics
  - & Rest Bus Simulation

- **Vehicle Position, Motion State, real-time UDP**

- **GPU Server**
  - Rendering, Visualization
  - & Sensor Simulation

Camera & LIDAR-Data, UDP

MathWorks Automotive Conference 2019, Stuttgart
Code Integration Workflow - HIL

NVIDIA DRIVE PX2
Perception & Planning

Vehicle ECU

Trajectories, real-time UDP

Speedgoat Mobile
Sensor Fusion
Control

Speedgoat Performance
Vehicle Dynamics & Rest Bus Simulation

Vehicle Position, Motion State, real-time UDP

GPU Server
Rendering, Visualization & Sensor Simulation

Camera & LIDAR-Data, UDP

Vehicle & Environment Simulation

CAN
Communication Between Simulink and Unreal Engine Using the Vehicle Dynamics Blockset

- **Input:**
  - position \((x, y)\)-coordinates & heading \(\Psi\) from vehicle-simulation or manual steering mode

- **Output:**
  - Camera stream
  - LIDAR-stream

- Vehicle Dynamics Blockset allows to interface with Unreal-Engine easily
Code Integration Workflow - HIL

**Vehicle ECU**

**NVIDIA DRIVE PX2**
Perception & Planning

Trajectories, real-time UDP

**Speedgoat Mobile**
Sensor Fusion Control

CAN

**Vehicle & Environment Simulation**

**Speedgoat Performance**
Vehicle Dynamics & Rest Bus Simulation

Vehicle Position, Motion State, real-time UDP

**GPU Server**
Rendering, Visualization & Sensor Simulation

Camera & LIDAR-Data, UDP
Unreal-Engine: Creating Tracks with Recorded Data

» Re-building real-tracks in virtual world fast and easy
Unreal-Engine: Virtual Car & Virtual Sensors

Sensor Models from MathWorks allow realistic closed-loop testing

GPU Server

Unreal-Engine
- Virtual Vehicle
- Virtual Track
- Sensor Position

LIDAR Sensor Model

Simulink
- Interface to Unreal-Engine
- ROS Interface

UDP Interface

NVIDIA Drive PX2
- Perception Algorithm

MathWorks Automotive Conference 2019, Stuttgart
Code Integration Workflow - HIL

Vehicle Perception & Control

Vehicle Simulation

Environment Simulation / Rendering
Conclusion

- Presented HIL set-up
  - Allows extensive testing & training in-house
  - Seamless Workflow provided by using Speedgoat Systems with Simulink Real-Time
    - Minimizes time spent for configuration
    - Maximizes time spent on algorithmic development

Next Steps Situation
- Preparing for race events in Spain and Germany
  - Leverage HIL set-up to train & test algorithms extensively before going to track

Future
- Open-source more of our software modules ([link](#))
- Make HiL accessible to community
- More features in software stack for autonomous driving applications
- Integrate Speedgoat Baseline on self-developed 1:10 model cars
Backup
Mathworks Software in the Project

• Function development
  • Speed control
  • Path tracking control
  • Sensor fusion
  • Vehicle state machine
  • More to come..

• Simulation
  • Software in the loop simulation
  • Self-designed vehicle dynamics
  • Real-Time testing

• Project organization
  • Simulink Project
  • Referenced models
  • Data dictionaries
  • Simulink Test
  • Data analysis
  • Merge tools (git)
Software Architecture – General Design

TUM Roborace Software
Functional Architecture

OFFLINE (LAPTOP)
- ROS Bag (LIDAR + GPS)
  - Mapping
  - Manual Map Postprocessing
  - Minimum curvature optimization
  - Velocity Profile Generation

NVIDIA PX2
- Precomputed Map
- LIDAR Sensor Interface
- LIDAR Localization
- Behavior State Machine
- Global Race Trajectory
- Local Trajectory Generation
- Trajectory Tracking Controller

Speedgoat
- Odometry (Kistler)
- Sensor Fusion
- Vehicle Dynamics Control

Communication Legend
- ROS
- Raw UDP
- Internal

DCU & Vehicle via CAN
Software Functionality – Localization

- Application of AMCL (Adaptive Monte Carlo Localization)
- Odometry based on optical velocity sensor
- Fusion with GPS in a second step
Software Functionality – Control

- Feedback control of lateral deviation and velocity error
- Tuning is difficult for high speeds $> 130$ kph
- No integral action in lateral feedback path to prevent difficulties in chicanes
Tracking and Prediction of Traffic Participants

Problem/Motivation

- Many traffic collisions arise through human error
- Make the car aware of its surroundings through sensor technologies

Goals

- Efficiently detect, track and predict the behavior of traffic participants (cars, bicycles, pedestrians)
- Enable the generation of a trajectory for the ego vehicle which takes into account the knowledge of its surroundings to avoid accidents

Approach

- Estimate movement possibilities for different object classes through classification
- Evaluate machine learning approaches for detection and prediction
Simulation – Hardware from Speedgoat

- Mobile real-time target machine → Fast control dynamics, used vehicle ECU
- Performance real-time target machine → Fast sensor sample rates, no delay and complex scenarios possible

MathWorks Automotive Conference 2019, Stuttgart
Safety Assessment of an Autonomous Roborace Race Vehicle

Problem/Motivation

› Artificial intelligence (AI) for safety critical tasks in autonomous driving (e.g. trajectory generation)
› Due to the non-deterministic nature of AI, a proper and safe function can not be guaranteed

Goals

› Safety assessment of trajectories generated by non-deterministic approaches
› Application on an autonomous race vehicle (Roborace)

Approach

› Determination of features indicating a safe planned trajectory
› Development of a deterministic safety supervisor for race vehicles
› Implementation and evaluation of the supervisor on an Roborace vehicle
Estimation and Prediction of Tire Road Friction Potential

Problem/Motivation
- Tire road friction potential has a major impact on vehicle safety and on vehicle performance and thus lap time in racing scenarios
- The tire road friction potential depends on a huge variety of influencing factors
- Knowledge of friction potential is essential for racing and autonomous driving

Goals
- Precise estimation of tire road friction potential and generation of a grid map
- Continuous updating of the 'road friction map'
- Prediction of tire road friction potential for short and long term (track section ahead and racetrack during whole race)

Approach
- Established methods relying on vehicle sensors, vehicle dynamic model, tire model, …
- Utilizing machine learning techniques with focus on
  - camera based friction estimation / prediction
  - Tire model-less tire force calculation
  - friction estimation based on acoustics
Safe Learning Control for Autonomous Vehicles

Motivation

• Classic control approaches depend heavily on model quality
• Difficult to deal with tire wear, low level component changes and changing environment conditions

→ Works well until 70-80% of the available friction

Goals

• Develop an algorithm which is capable of approaching the handling limit slowly and with care
• Guarantee safety and intuitive tuning of the algorithm

Approach

• Humans use a combination of experience and gut-feeling
• Use Machine Learning methods in combination with classic control theory to mimic this behavior
• Model the human safety and performance assessment for feedback (trajectory planning and learning style)
Energy Management Strategy for Autonomous Electric Cars

Motivation
- Amount of stored energy in battery electric vehicles is limited
- Technical limitations of electric vehicles’ components are reached soon due to highly dynamic driving scenarios

Goal
- Development of an Energy-Management-Strategy that optimizes the race trajectory and controls the power flow within the electric race car in real-time to optimize lap times for a whole race

Approach
- Deduction of controlling policies resulting from machine learning methods and comparison with conventional ones
- Development of an efficient operational strategy for an autonomous electric race car