Development of a MotionSolve integrated Driver Model

Kompetenzzentrum - Das virtuelle Fahrzeug, Forschungsgesellschaft mbH
Martin Rudigier
Christian Prettenthaler

Altair Engineering
Mark Krueger

Institut für Intelligente Systemtechnologien Alpen-Adria Universität Klagenfurt
Univ.-Prof. M. Horn

K2 / K plus Competence Center - Initiated by the Federal Ministry of Transport, Innovation & Technology (BMVIT) and the Federal Ministry of Economics & Labour (BMWA). Funded by FFG, Land Steiermark and Steirische Wirtschaftsförderung (SFG)
Content

• Introduction
• Vehicle Model
• Driver Structure
• Driver Modules
• Controller Switching
• Cosimulation
• Results
• Conclusion

Source: Altair Engineering
Vehicle Dynamics Simulation:

- Vehicle
- Environment (road, …)
- Driver

Goal:
A robust driver model for Vehicle Dynamics Simulation on basis of a the multibody simulation (MBS) Altair MotionSolve
Matlab/Simulink Development and Basic Tests

Co-Simulation MotionSolve/Simulink Tests

Integration into Motionsolve
Matlab/Simulink Vehicle Model

• Controller Design
• Basic tests
• Similar behavior to the MBS vehicle model
  • Two-track-model with weight transfer fore-aft and side-to-side (pitch and roll motion)
  • Tire model
  • Aerodynamic drag (long, lat)
  • Drivetrain / engine
  • Inclination resistance
Driver - Requirements derived from Use Cases

**Basic Use Cases**
- Straight-line acceleration with gear shifting
- Steady-State cornering
- Brake-in-turn
- Tracking of path and speed

**Advanced Use Cases**
- Drive in reverse
- Starting from stand-still
- Three-Point-Turn
Analysis of the Use Cases

→ required control variables

Controller

• speed controller
  • (longitudinal) acceleration
  • velocity
  • lateral acceleration
• steering controller
  • curvature
  • path
  • lateral acceleration
• gear box controller

Controller switching

• Use Case: steady state cornering
  • steering controller
Steady state cornering

- Vehicle set onto road at $t=0$
- Open loop control $\rightarrow$ steering angle
  - Steady-State initial conditions
- Steering controller takes over control at $t=10$ (closed loop)
  - Not disturbing steady state
  - External disturbances
Structure of the steering controller:

Use Case: Steady-State Cornering
1st Driving Task: open loop control of the steering angle $\delta$
2nd Driving Task: closed loop control of the vehicle position (path) position control and curvature control
Controller concept

- Feed forward controller:
  → inverse plant
- Compensational controller:
  → PI-controller with Anti-Windup structure

![Controller concept diagram](image-url)
Controller switching

Principle:
- Calculate the contribution of the feed forward controller $u_{FF}$
- Calculate the P-contribution of the PI-controller $u_{C,P}$
- Calculate the initial conditions of the integrator

\[ u_{C,I,O} = u - \tilde{u}_{FF} - \tilde{u}_{C,I} \]
- Set the initial conditions of the integrator

[Diagram of controller switching]
MotionSolve/Simulink Co-Simulation

Classical Example – Inverse Pendulum

MotionSolve Block

State Variables

Time offset: 0
MotionSolve Input/Output Elements

- Supports multiple input/output ports
- User may disable/enable ports via “is_active” flag
- Supports either continuous or discrete sampling
MotionSolve/Simulink Co-Simulation Strategy

- MotionSolve and Simulink, run in parallel, each on their own thread
- Communication is via shared memory
- Input and output data is buffered
  - MotionSolve interpolates or extrapolates data in time as needed
  - Solvers are held within one integration time step of each other using PThread locks
  - User may choose zero-order, first-order or second-order hold
- Relationship is reciprocal, neither solver is master or slave
- Strategy results in faster and more robust co-simulations
Results - Velocity

- Cosimulation Simulink / MotionSolve
- Switching point at t=10
Results - Tyre Forces

Front Tire Lateral Forces

- Front Right Tire
- Front Left Tire

Lateral Force [N]

Time [s]
Results - Steering Wheel Angle
Results - Path of the Vehicle

Source: Altair Engineering
Conclusion

- Concept and structure of a driver model
- Specification of the general framework for controllers and controller switching
- Principle and implementation of controller switching
- Cosimulation works!!
- Results
Thank you for your attention!
Backup
Anti-Windup
Steering control - curvature

Input: curvature [1/m]
Output: steering wheel [rad]

Feed forward control: linear one track model
• Ackermann steering angle:
  \[
  \delta_A = (l_{re} + l_{fr}) \cdot \kappa_{demand}
  \]
• Compensation steering angle:
  \[
  \delta_C = \frac{m a_y}{l_f + l_r} \left( \frac{l_r}{c s_{fr}} - \frac{l_f}{c s_{re}} \right)
  \]
  \[
  \delta_C = 0.5(\alpha_{rear,left} + \alpha_{rear,right} - \alpha_{front,left} - \alpha_{front,right}) \cdot k_{SA}
  \]
• Feed forward steering angle
  \[
  \delta_{FF} = \delta_A + \delta_C
  \]
• Steering angle
  \[
  \delta = \delta_{FF} + \delta_{Pl}
  \]
Steering controller - position

Input: path
Output: curvature: [1/m]

\[ x_{Tr}(s) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \begin{bmatrix} m \\ m \end{bmatrix} \]

Circle
- Vehicle position \( x_{\text{vehicle}} \)
- Direction / orientation of the vehicle
- Preview position \( x_{\text{preview}} \)
- Demand curvature = Curvature of the circle