

# Development of a MotionSolve integrated Driver Model



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



Altair Engineering  
**Mark Krueger**

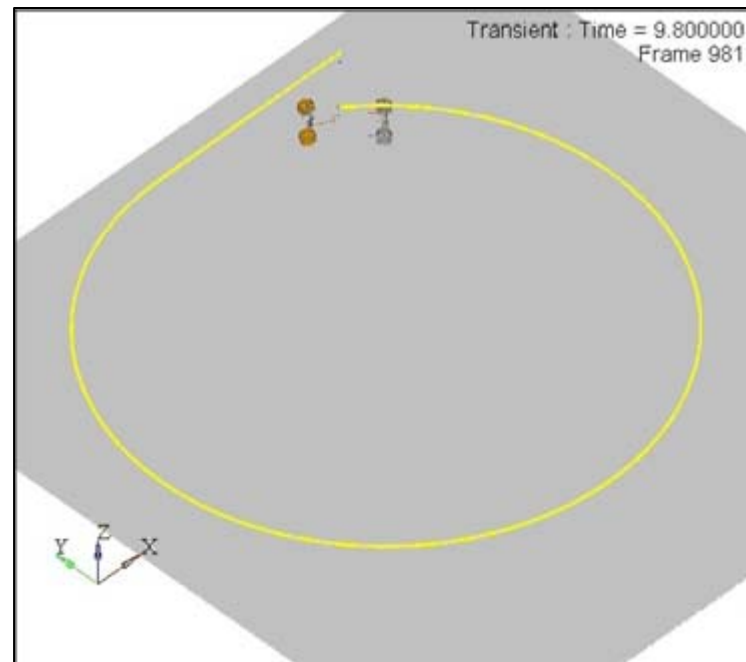


Institut für Intelligente Systemtechnologien Alpen-Adria Universität Klagenfurt  
Univ.-Prof. **M. Horn**



*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)

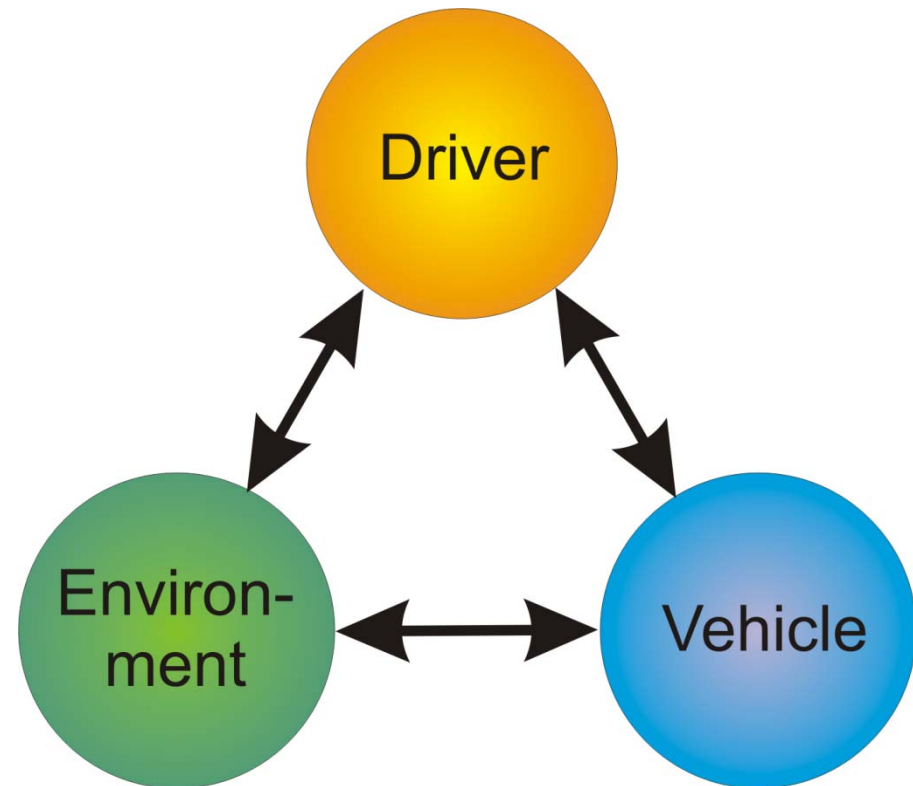
- Introduction
- Vehicle Model
- Driver Structure
- Driver Modules
- Controller Switching
- Cosimulation
- Results
- Conclusion



Source: Altair Engineering

## Vehicle Dynamics Simulation:

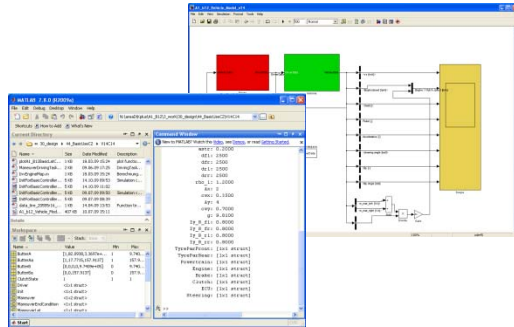
- Vehicle
- Environment (road, ... )
- Driver



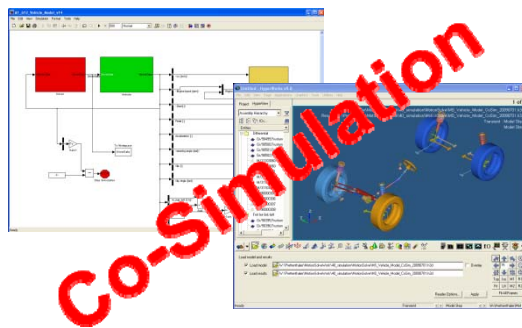
### Goal:

A robust driver model for Vehicle Dynamics Simulation on basis of a the multibody simulation (MBS) *Altair MotionSolve*

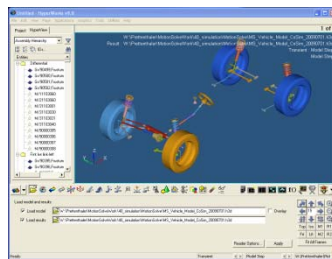
Time



Matlab/Simulink  
Development and Basic Tests

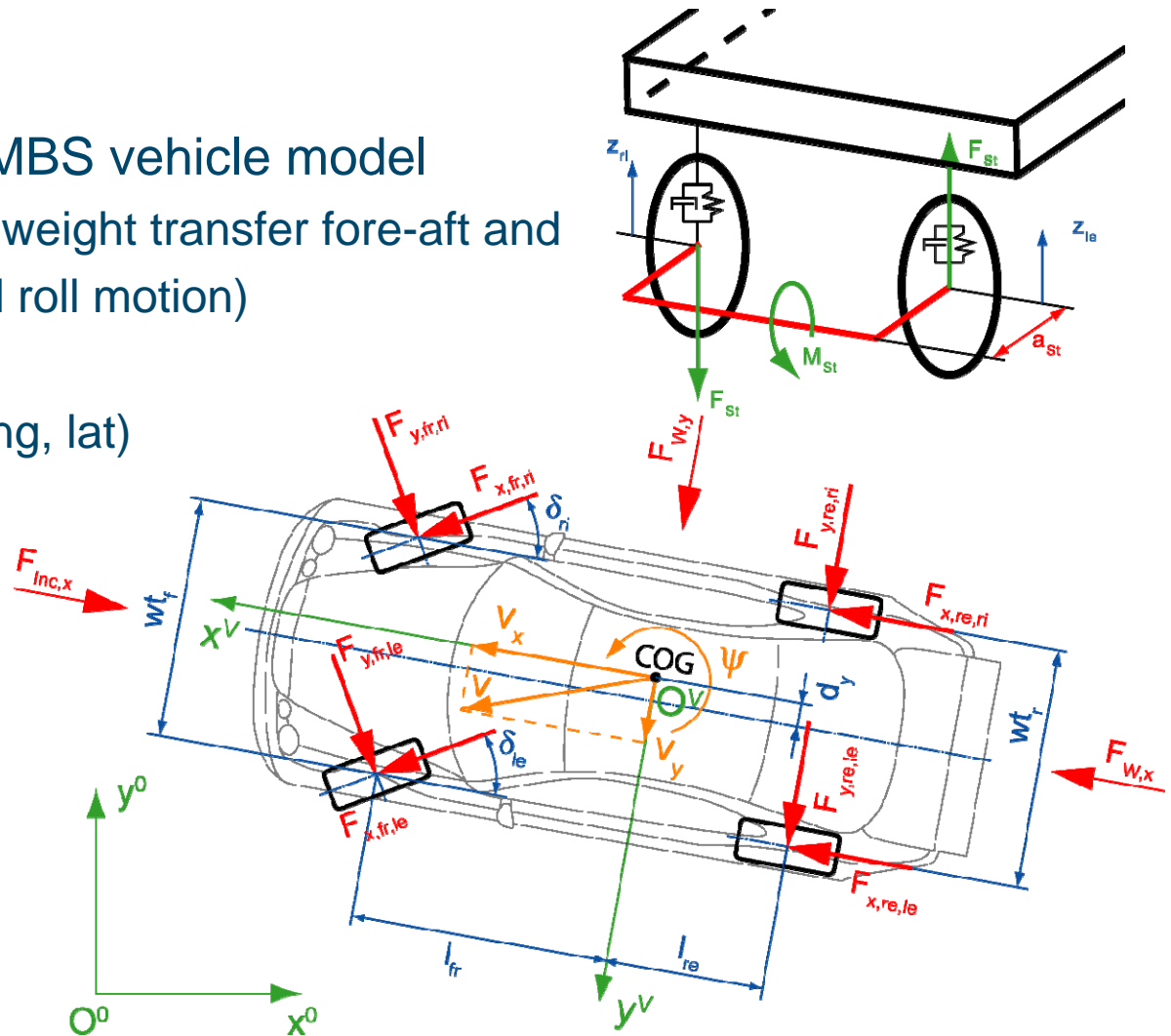
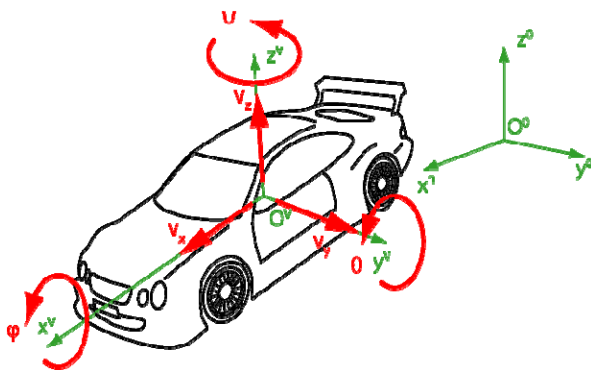


Co-Simulation MotionSolve/Simulink  
Tests



Integration into Motionsolve

- 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



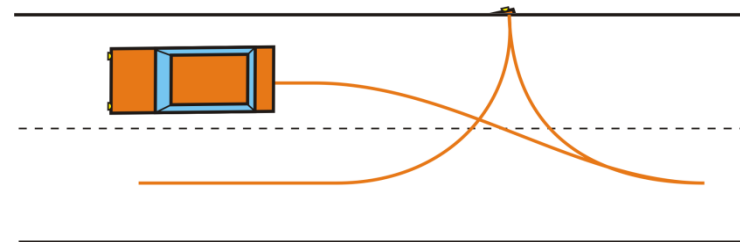
## Driver - Requirements derived from Use Cases

### 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

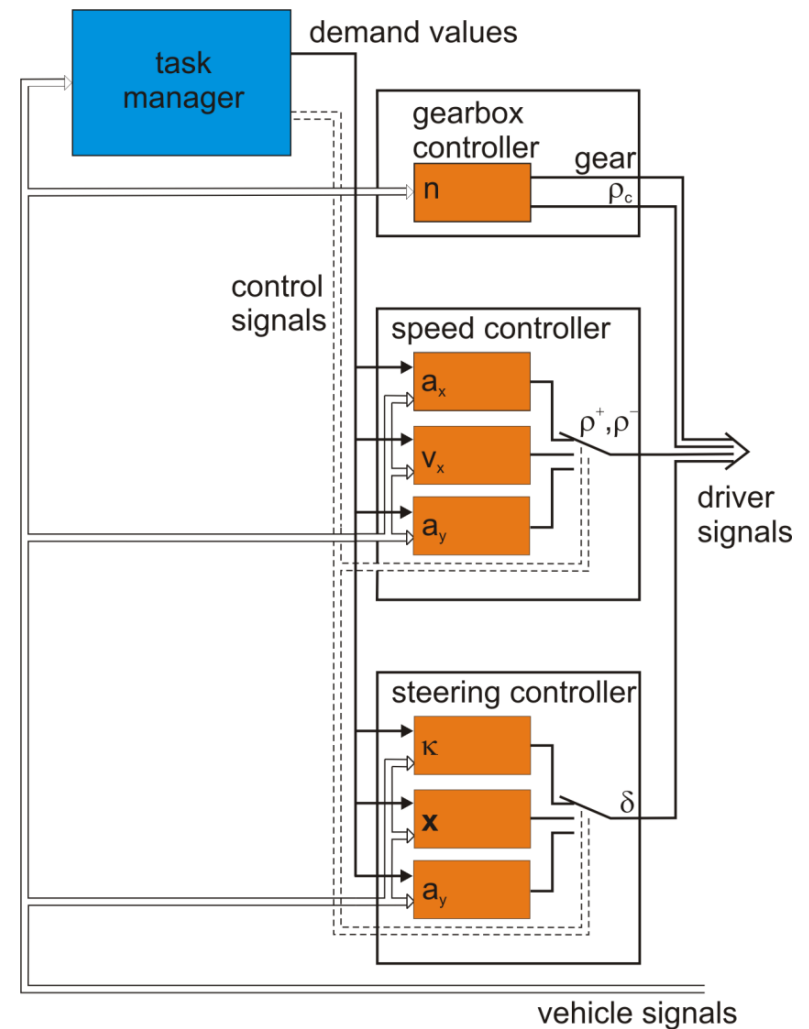
→ required control variables

## Controller

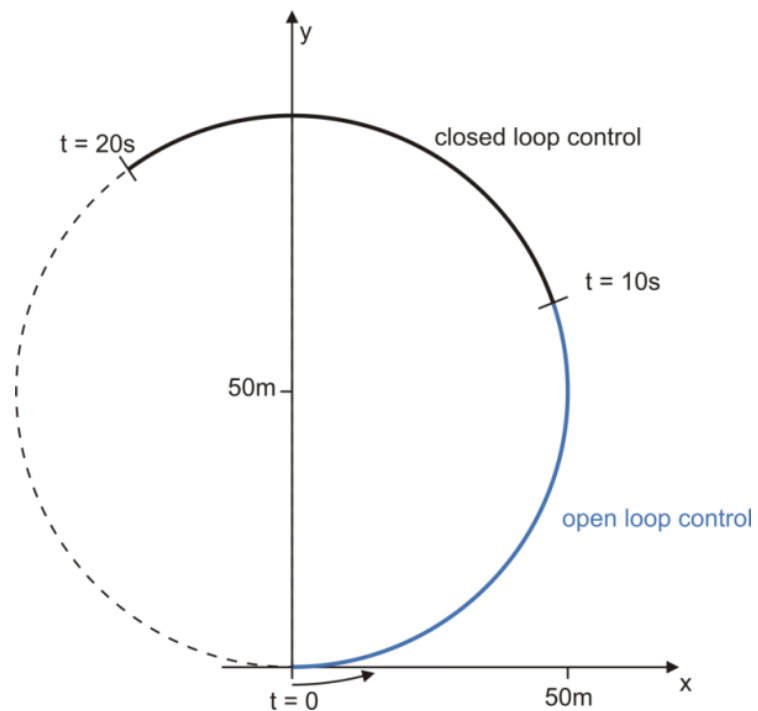
- speed controller
  - (longitudinal) acceleration
  - velocity
  - lateral acceleration
- steering controller
  - curvature
  - path
  - lateral acceleration
- gear box controller

## Controller switching

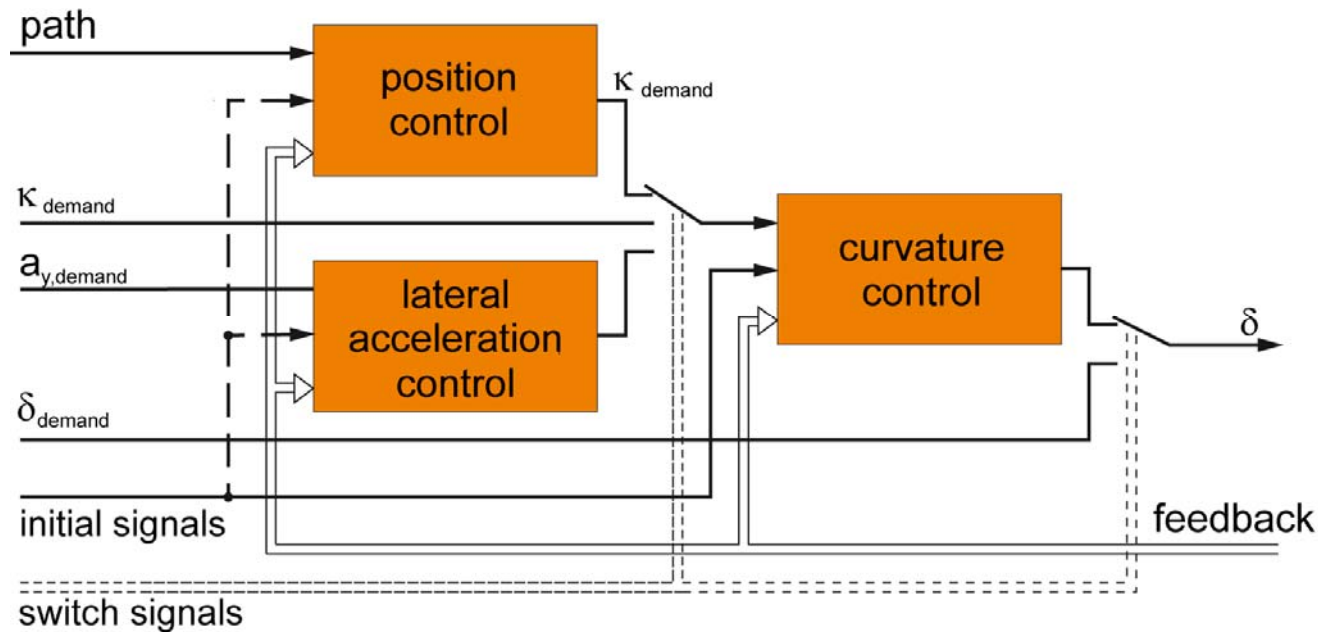
- Use Case: steady state cornering
  - steering controller



- **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

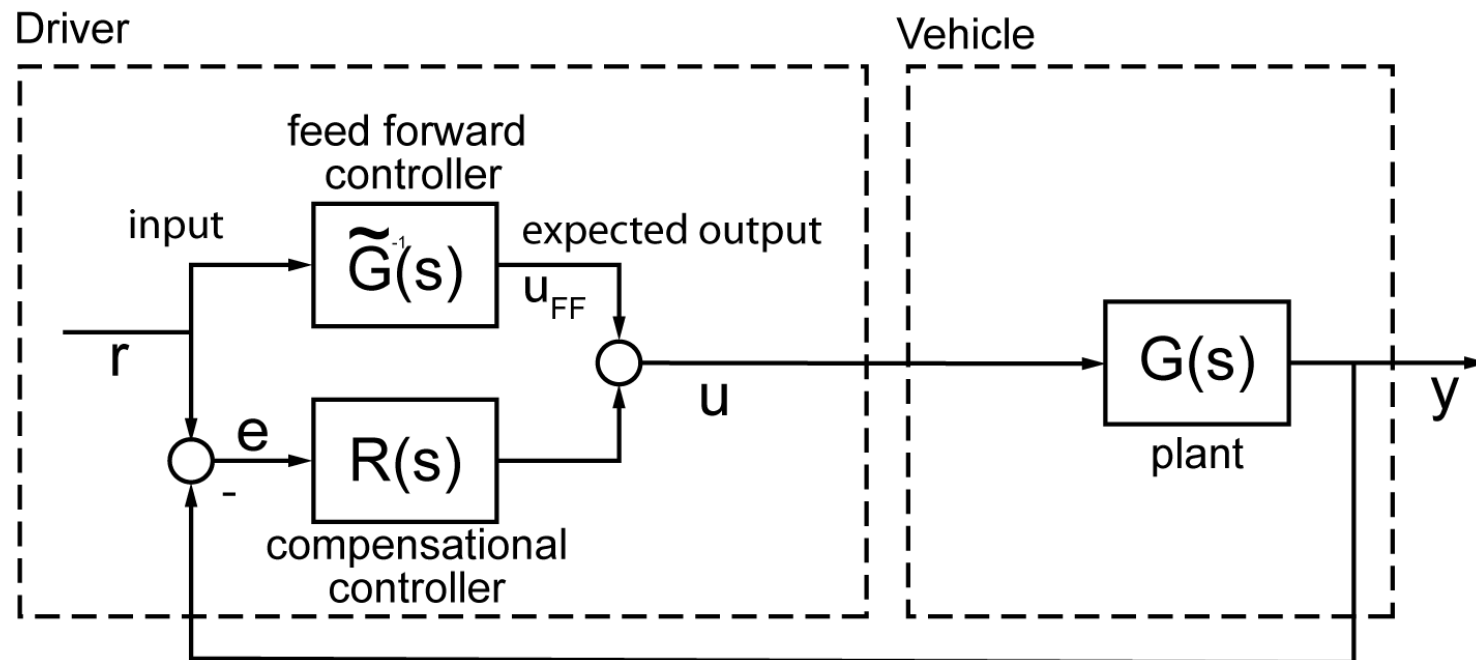
1st Driving Task: open loop control of the steering angle  $\delta$

2nd Driving Task: closed loop control of the vehicle position (path)

position control and curvature control



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

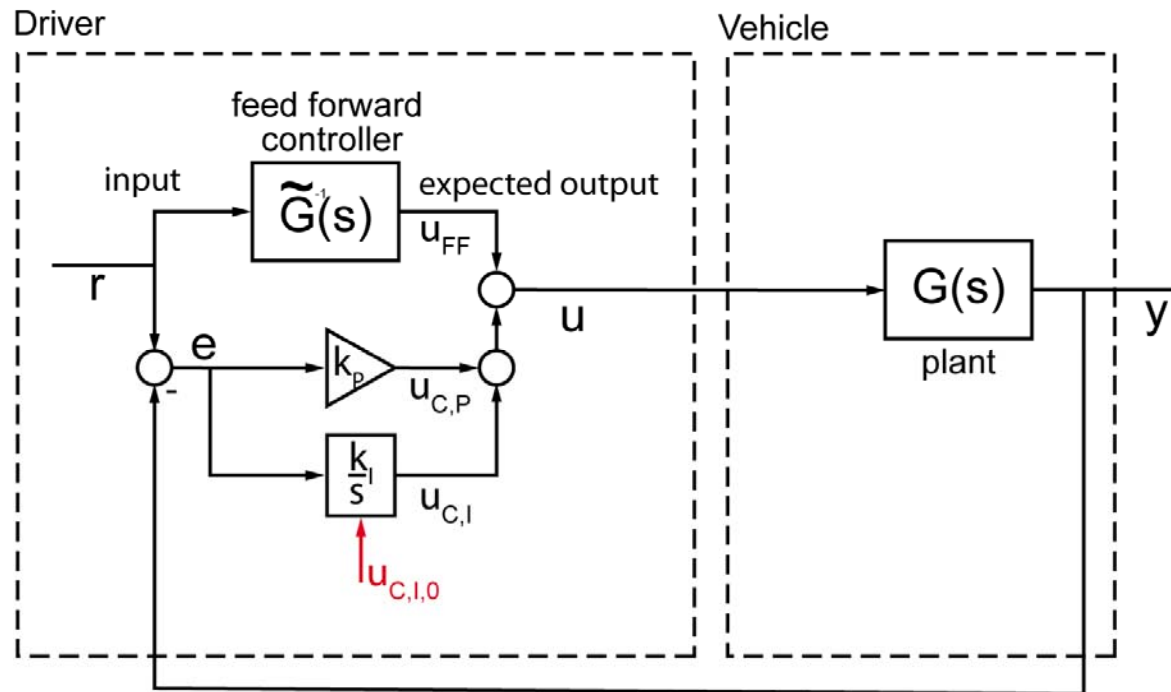


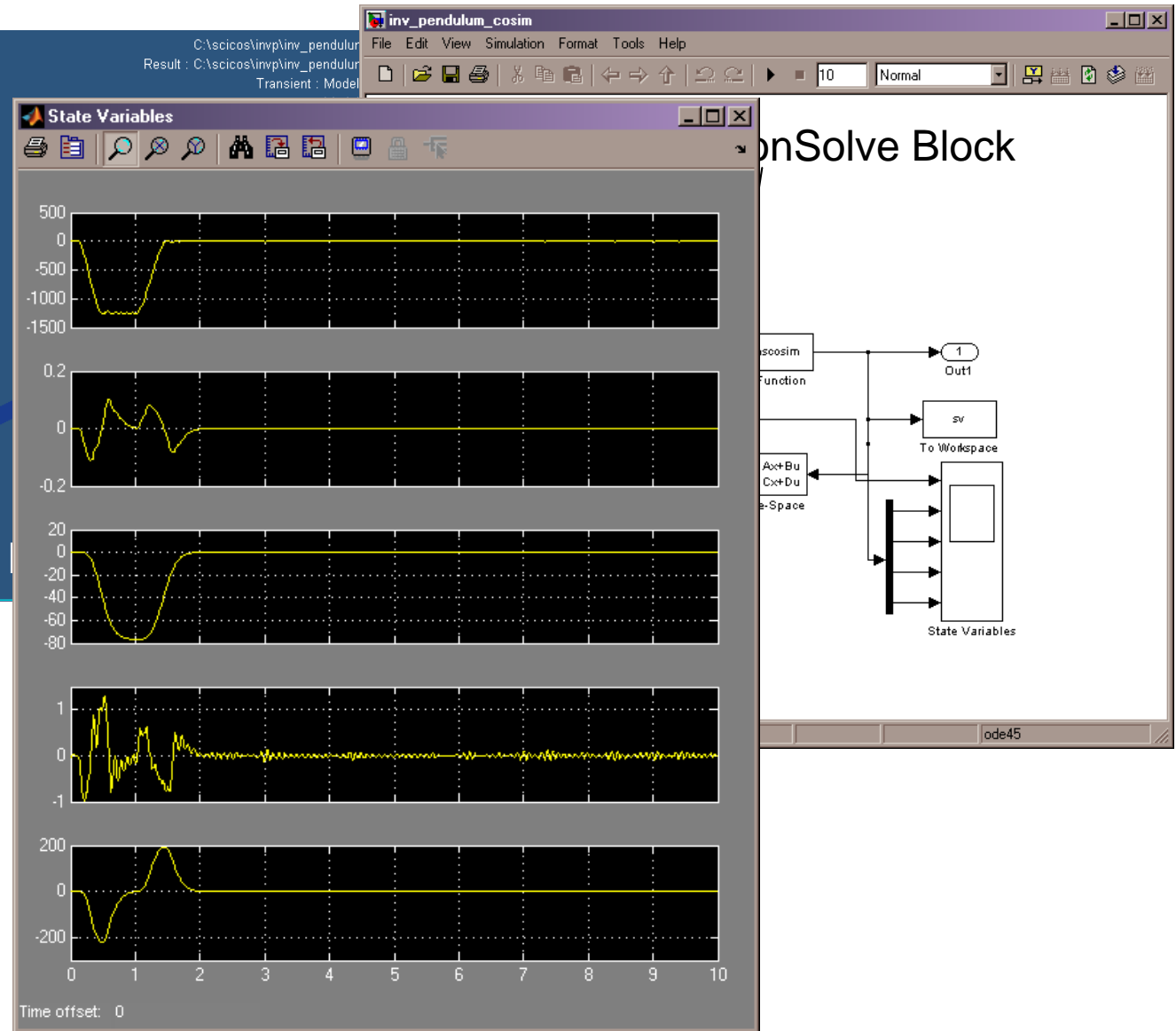
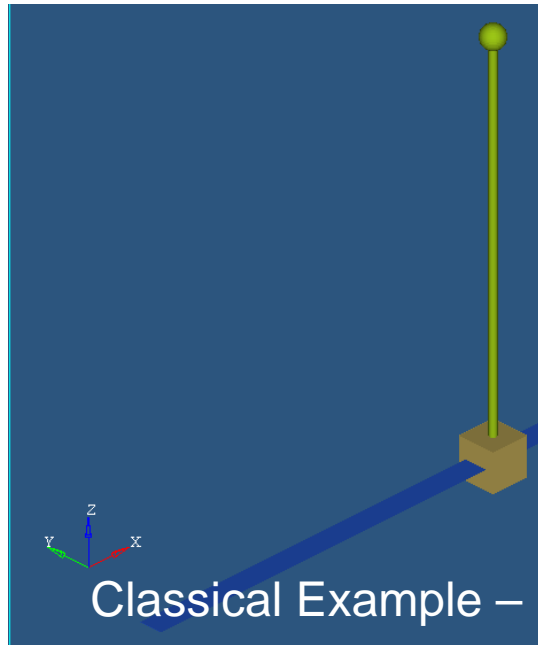
Principle :

- 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

$$u_{C,I,0} = u - \tilde{u}_{FF} - \tilde{u}_{C,I}$$

- Set the initial conditions of the integrator



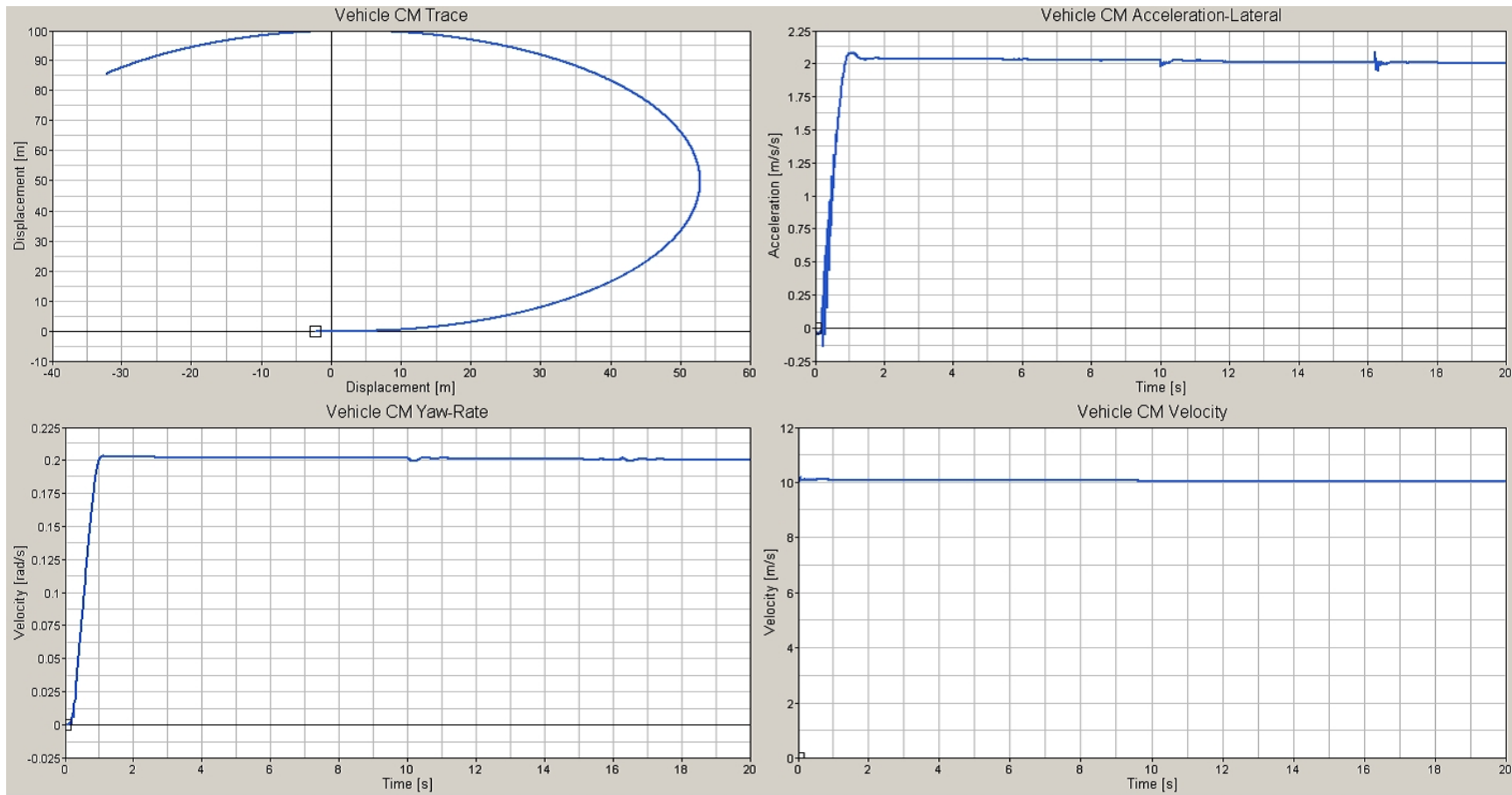


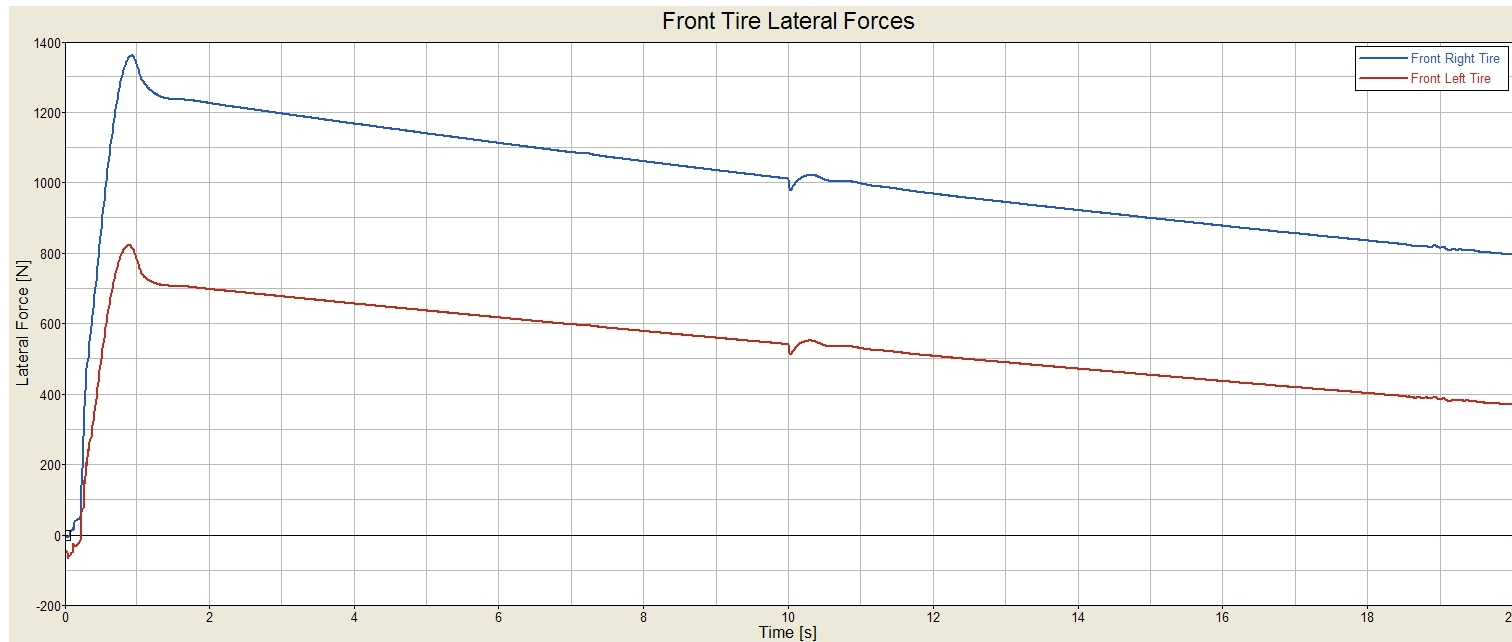
```
C:\qa\depot_xml\md\BusSuspension2PMIMO\BusSuspension2PMIMOmix.xml *
  expr                               = "FZ(30103035,30104042)"
</>
<Control_PlantInput
  id                                  = "30100100"
  is_active                           = "TRUE"
  num_element                         = "2"
  variable_id_list                    = "30100400, 30100500"
  sampling_period                     = "0.01"
  offset_time                         = "0.0"
</>
<Control_PlantInput
  id                                  = "30100300"
  is_active                           = "TRUE"
  num_element                         = "2"
  variable_id_list                    = "30100800, 30100900"
  sampling_period                     = "0.0"
  offset_time                         = "0.0"
</>
<Control_PlantOutput
  id                                  = "30100400"
  is_active                           = "TRUE"
  num_element                         = "2"
  variable_id_list                    = "30100600, 30100700"
  sampling_period                     = "0.0"
  offset_time                         = "0.0"
</>
<Control_PlantOutput
  id                                  = "30100200"
  is_active                           = "TRUE"
  num_element                         = "2"
  variable_id_list                    = "30100200, 30100300"
  sampling_period                     = "0.01"
  offset_time                         = "0.0"
</>
```

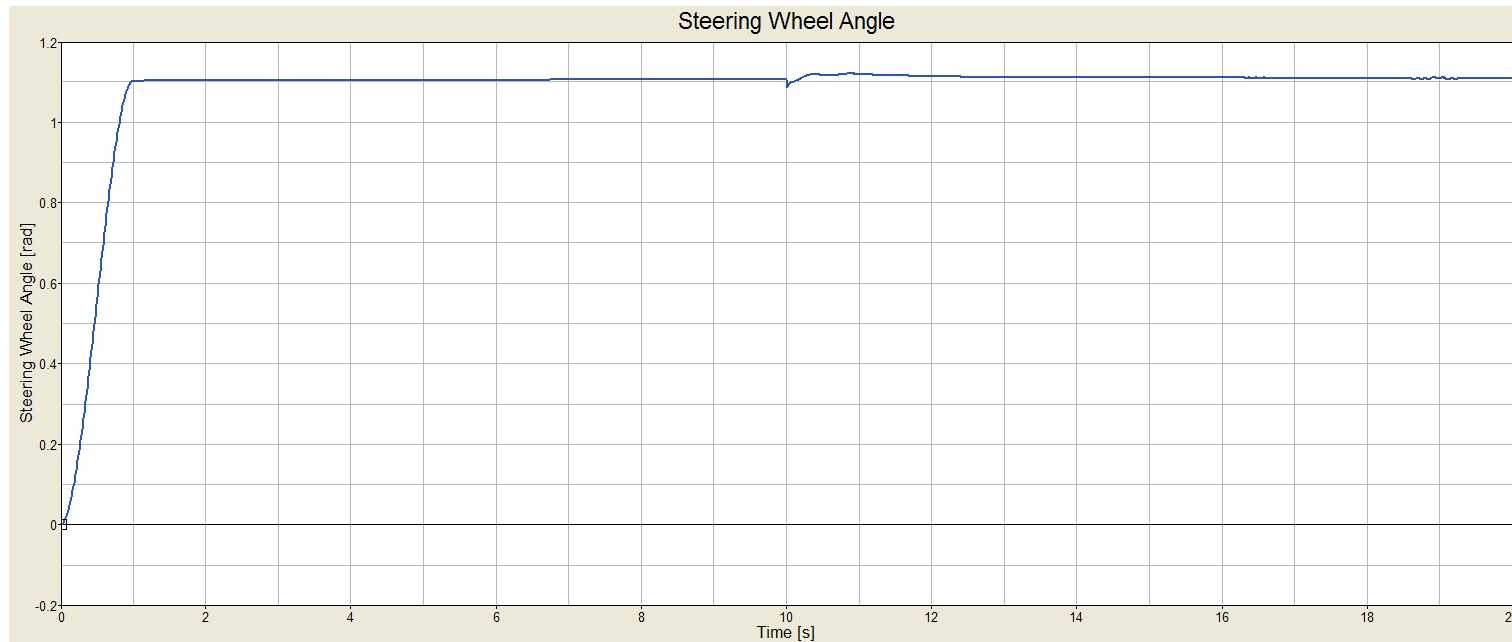
- Supports multiple input/output ports
- User may disable/enable ports via "is\_active" flag
- Supports either continuous or discrete sampling

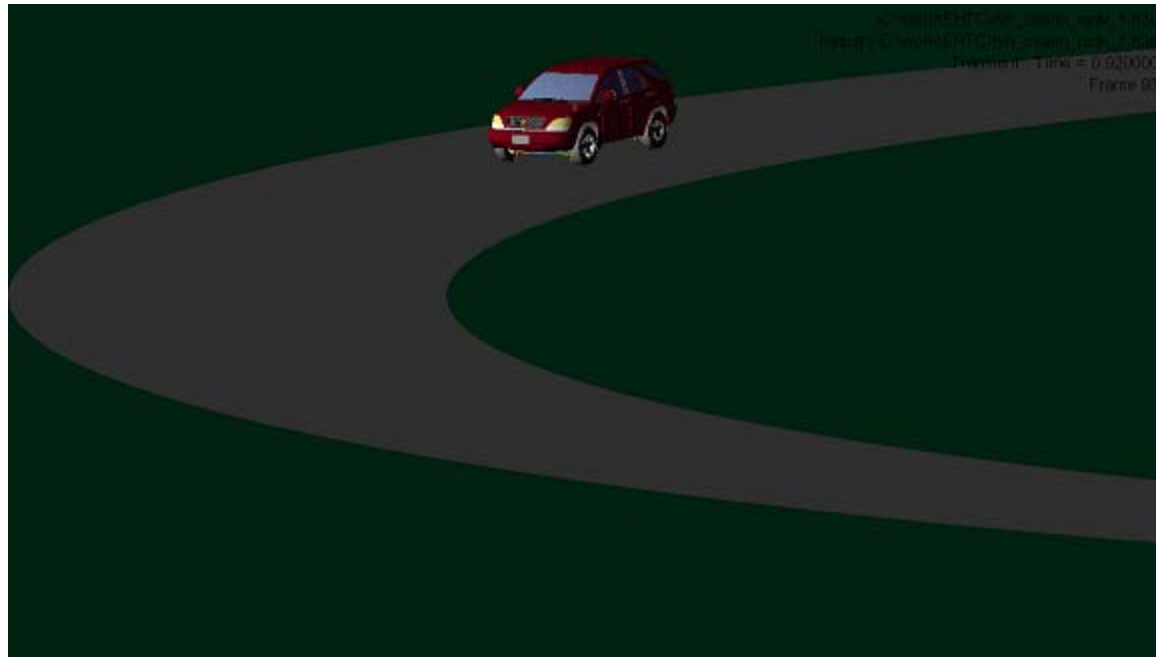
- MotionSolve and Simulink, run in parallel, each on their own thread
- Communication is via shared memory
- Input and output data is buffered
  - MotionSolve interpolates or extrapolates data in time as needed
  - Solvers are held within one integration time step of each other using PThread locks
  - 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

- Cosimulation Simulink / MotionSolve
- Switching point at  $t=10$





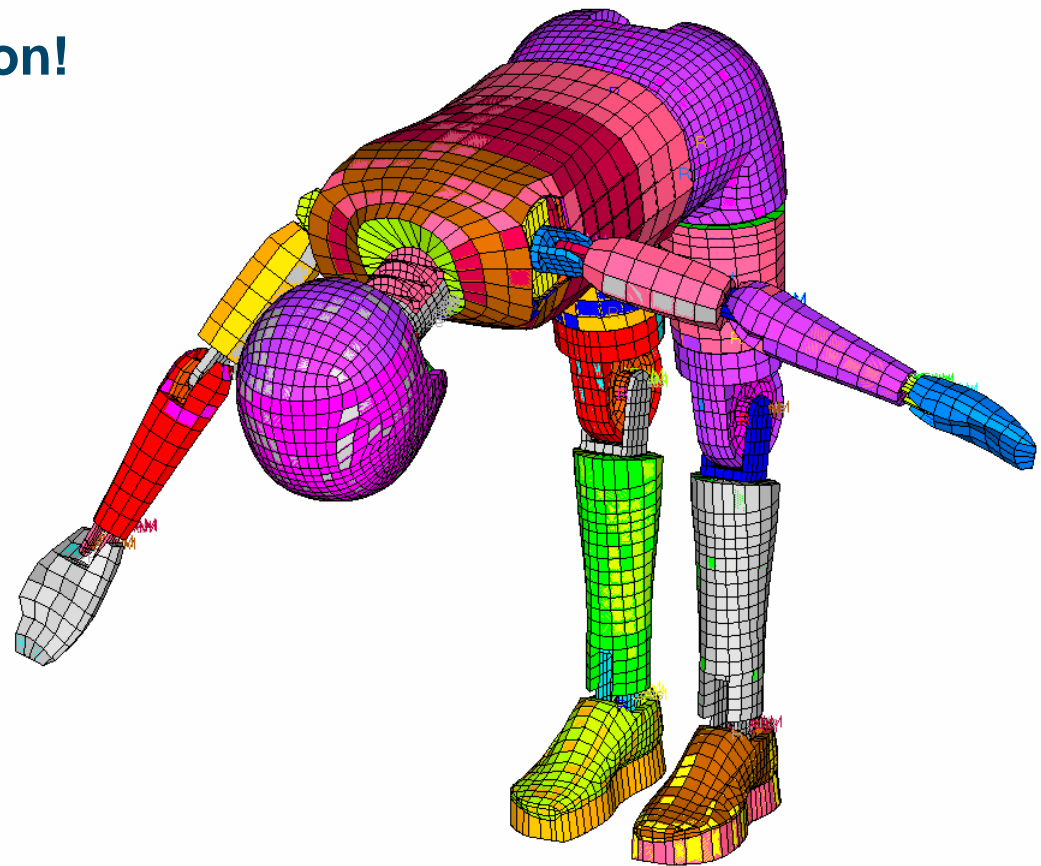




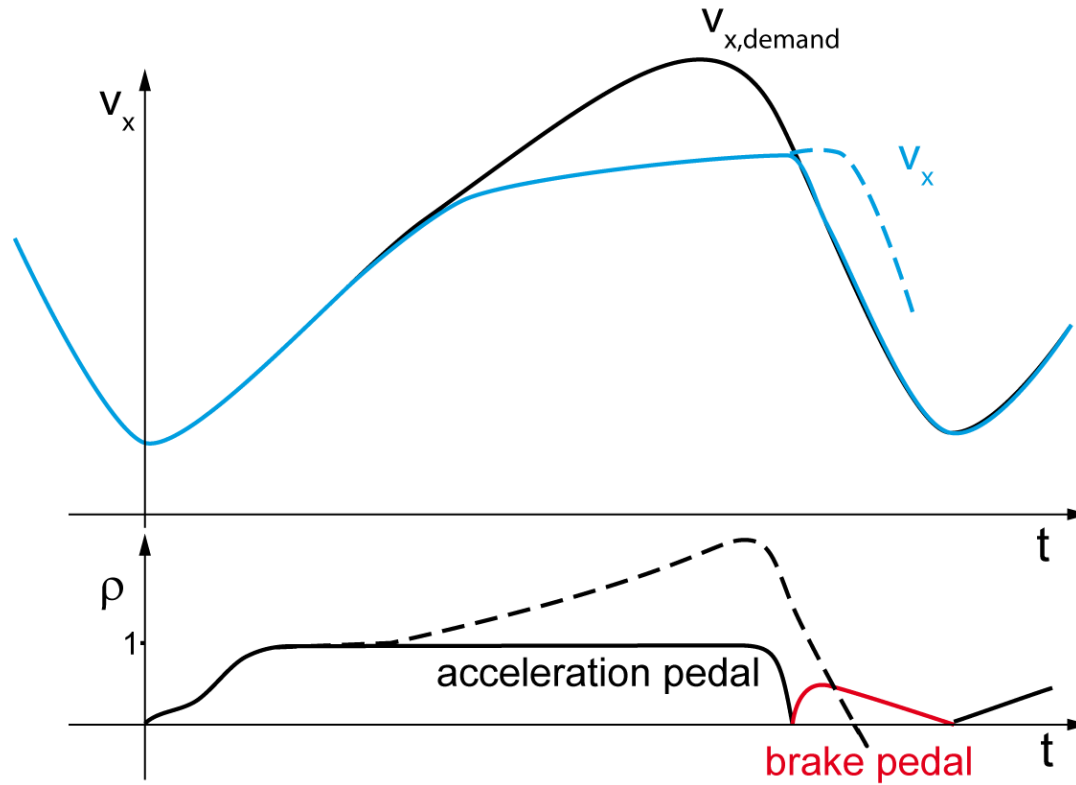
Source: Altair Engineering

- 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



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:

$$\delta_C = \frac{m a_y}{l_f + l_r} \left( \frac{l_r}{cs_{fr}} - \frac{l_f}{cs_{re}} \right)$$

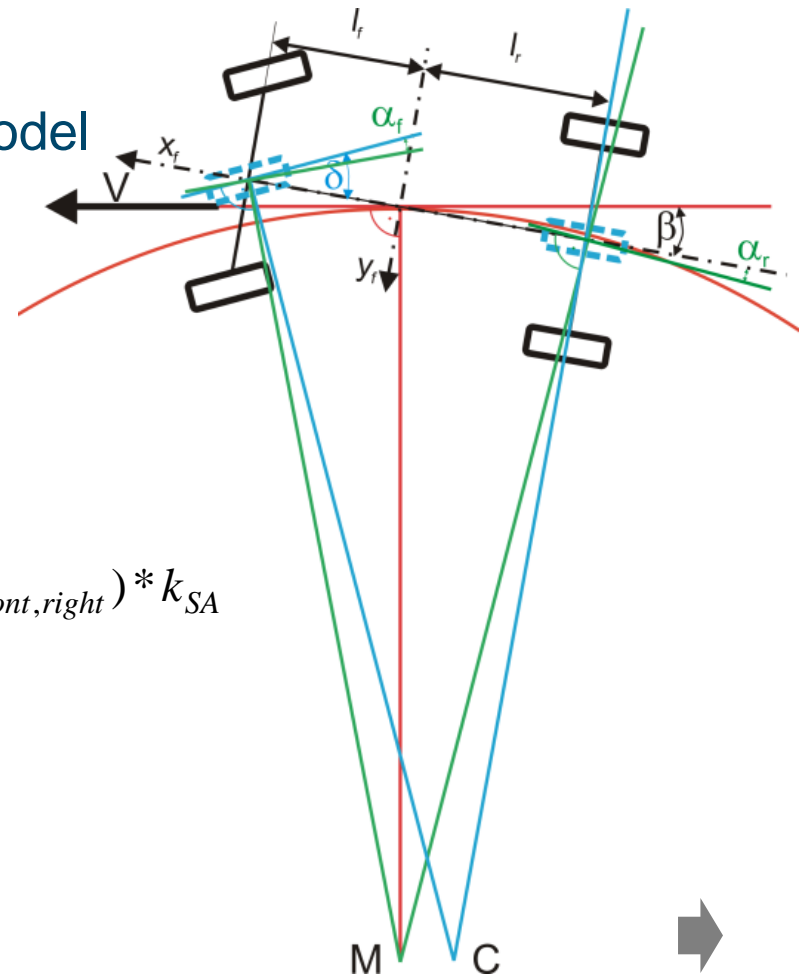
$$\delta_C = 0.5(\alpha_{rear,left} + \alpha_{rear,righ} - \alpha_{front,left} - \alpha_{front,right}) * k_{SA}$$

- Feed forward steering angle

$$\delta_{FF} = \delta_A + \delta_C$$

- Steering angle

$$\delta = \delta_{FF} + \delta_{PI}$$



Input: path

Output: curvature: [1/m]

$$\mathbf{x}_{Tr}(s) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \begin{bmatrix} m \\ m \\ m \end{bmatrix}$$

### Circle

- Vehicle position  $\mathbf{x}_{vehicle}$
- Direction / orientation of the vehicle
- Preview position  $\mathbf{x}_{preview}$
- Demand curvature = Curvature of the circle

