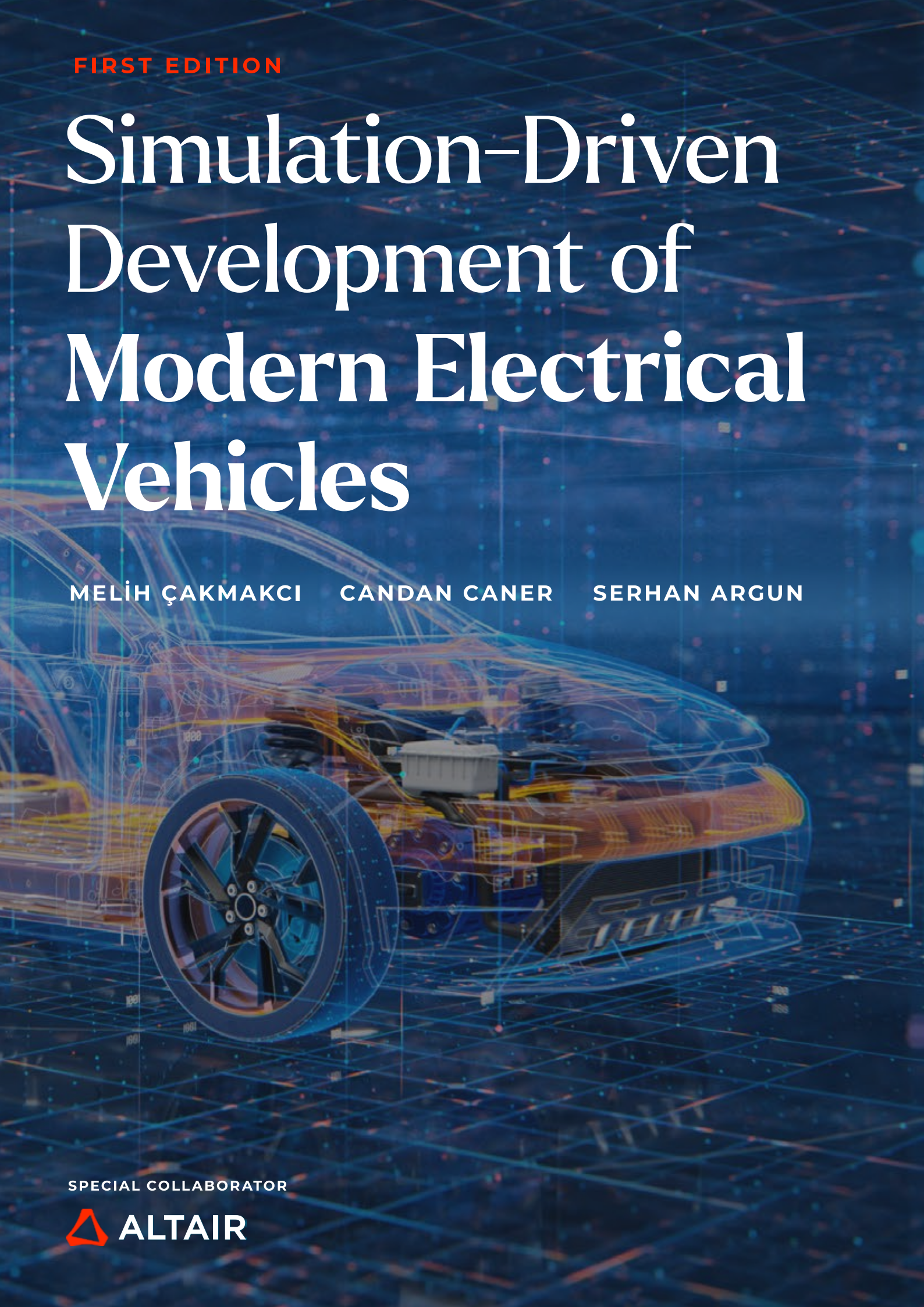
FIRST EDITION

Simulation-Driven Development of Modern Electrical Vehicles

MELİH ÇAKMAKCI CANDAN CANER SERHAN ARGUN



SPECIAL COLLABORATOR

ALTAIR

PDF FILE VERSION: 0.9.1

Simulation-Driven Development of Modern Electrical Vehicles

AUTHORS:

Melih ÇAKMAKCI

Bilkent University

Smart Mechanical Systems Laboratory

Candan CANER

Serhan ARGUN

SPECIAL COLLABORATOR:

Table of Contents

1 Introduction **9**
- 1.1 Background . . . 9
- 1.2 Motivation . . . 10
- 1.3 Scope . . . 11
- 1.4 References . . . 11

2 Vehicle Dynamics Overview **12**
- 2.1 Introduction . . . 12
- 2.2 Longitudinal Vehicle Motion . . . 14
  - 2.2.1 Brief Description . . . 14
  - 2.2.2 Descriptive Figure . . . 15
  - 2.2.3 Equations . . . 16
  - 2.2.4 Model Figures . . . 17
  - 2.2.5 Model Overview . . . 19
  - 2.2.6 Simulation Results . . . 28
  - 2.2.7 Try On Your Own . . . 28
  - 2.2.8 3-D Visualization and Analysis in RTV-Simulator with FMI . . . 28
- 2.3 Lateral Vehicle Motion . . . 30
  - 2.3.1 Brief Description . . . 30
  - 2.3.2 Descriptive Figure . . . 30
  - 2.3.3 Equations . . . 32
  - 2.3.4 Model Figures . . . 34
  - 2.3.5 Model Overview . . . 34
  - 2.3.6 Results . . . 36
  - 2.3.7 Try On Your Own . . . 36
  - 2.3.8 3-D Visualization and Analysis in RTV-Simulator with FMI . . . 37
- 2.4 Vertical Vehicle Motion . . . 37
  - 2.4.1 Brief Description . . . 37
  - 2.4.2 Descriptive Figure . . . 37
  - 2.4.3 Equations . . . 38
  - 2.4.4 Model Figures . . . 39
  - 2.4.5 Model Overview . . . 39
  - 2.4.6 Results . . . 41
  - 2.4.7 3-D Visualization and Analysis in Simulator . . . 42
- 2.5 Combined Vehicle Dynamics . . . 42
  - 2.5.1 Brief Description . . . 42
  - 2.5.2 Descriptive Figure . . . 42
  - 2.5.3 Equations . . . 43
  - 2.5.4 Model Figures . . . 44
  - 2.5.5 Model Overview . . . 45
  - 2.5.6 Simulation Results . . . 46
- 2.6 References . . . 46

3 Electric Powertrain Overview **47**
- 3.1 Introduction . . . 47
- 3.2 Electric Motors . . . 48

3.2.1 Brief Description . . . 48
3.2.2 Model Figure . . . 49
3.2.3 Model Overview . . . 50
3.2.4 Results . . . 53
3.3 E-Powertrain Elements . . . 56
3.3.1 Brief Description . . . 56
3.3.2 Model Figure . . . 56
3.3.3 Model Overview . . . 57
3.3.4 Results . . . 57
3.4 Battery . . . 58
3.4.1 Brief Description . . . 59
3.4.2 Model Figure . . . 59
3.4.3 Model Overview . . . 60
3.4.4 Results . . . 63
3.5 EV Full Simulation Model 1-D . . . 64
3.5.1 Brief Description . . . 64
3.5.2 Model Figure . . . 65
3.5.3 Results . . . 66
3.6 References . . . 67

4 Electric Vehicle Simulation Development **68**
4.1 Introduction . . . 68
4.2 Input for Multibody Simulation . . . 69
4.3 A typical case study of a multibody simulation . . . 70
4.3.1 Definition of the problem . . . 70
4.3.2 Build and Test the model . . . 70
4.3.3 Review the results . . . 71
4.3.4 Improve the model . . . 72
4.4 References . . . 75

5 Appendix A - Altair® Products Installation **76**
5.1 Installation of Visual Studio™ C++ Compiler . . . 78

6 Appendix B - Activate Quick Start **79**
6.1 View Menu Options . . . 80
6.2 Home and Block Icons . . . 81
6.3 Results & Scope . . . 84
6.4 Co-simulation with Activate and MotionSolve . . . 84

7 Appendix C - RTV Simulator™ Quick Start **86**
7.1 Navigation and Controls in RTV . . . 88

8 Appendix D - MotionVIEW™ Quick Start **89**
8.1 References . . . 92

List of Figures

1.1 EV global market share during 2010-2021 . . . 9
2.1 Vehicle Coordinate System vectors . . . 13
2.2 Heading, slip, and course angles of the vehicle . . . 14
2.3 Longitudinal motion vectors. . . . 14
2.4 Force balance equation vectors . . . 15
2.5 Forces and balances around the center-line . . . 17
2.6 Top Level (Superblock) and Interior View . . . 17
2.7 Longitudinal Simulation v2 model. . . . 18
2.8 Longitudinal Simulation Model (Version #3) . . . 18
2.9 Longitudinal Simulation Model (Version #1) . . . 20
2.10 Initial parameter values for the model . . . 21
2.11 Simulation parameters . . . 22
2.12 Longitudinal Simulation Model (Version #2) . . . 23
2.13 Example force signal pattern . . . 23
2.14 Initial parameter values for the model . . . 24
2.15 Longitudinal Simulation Model (Version #3) . . . 25
2.16 Example Force pattern for the traction delivered to the wheels . . . 26
2.17 Close-up for Longitudinal Model (Version#1) . . . 26
2.18 Front Wheel Superblock . . . 27
2.19 Rear Wheel Superblock . . . 27
2.20 Longitudinal Model (Version#3) - Simulation Output . . . 28
2.21 3-D Co-Simulation screenshot - Using the RTV-Simulator . . . 29
2.22 Generating the Functional Mock-Up Interface (FMU) model . . . 29
2.23 e-Book FMU Folder Contents . . . 30
2.24 Lateral dynamics is all about the steering system and the tire forces . . . 30
2.25 Lateral force vectors while steering . . . 31
2.26 Tire lateral force vs. Slip angle relationship . . . 32
2.27 Lateral force vectors during a steady-state turn . . . 33
2.28 Lateral Simulation v1 model . . . 34
2.29 Lateral Simulation v2 model . . . 34
2.30 Lateral Simulation v1 - Model Details . . . 35
2.31 Initial parameter values for the model . . . 35
2.32 Lateral Simulation Model (Version #2) . . . 36
2.33 Road surface and the perception of the driver . . . 37
2.34 Vertical dynamics equivalent diagram for one wheel . . . 37
2.35 Vertical Model v1 . . . 39
2.36 Vertical Model v2 . . . 39
2.37 Vertical Model v1 . . . 40
2.38 Vertical Simulation Model (Version #2) . . . 41
2.39 Vertical Simulation Model (Version #1) . . . 42
2.40 Full Simulation Model Figure #1 . . . 43
2.41 Full Simulation Model Figure #2 . . . 43
2.42 Full Vehicle Simulation Model . . . 45
2.43 Signal connections of sub-models in the Full Vehicle Model. . . . 45
2.44 Simulation Results for the Full Vehicle Dynamics . . . 46
3.1 Simulated Components of the EV . . . 47

3.2 Electric Motor's Interior Structure . . . 48
3.3 Example Electric Motor Simulation Software . . . 49
3.4 Electric Motor Model . . . 49
3.5 Interior of the Electric Motor Model . . . 50
3.6 Input Signals for EM Simulation . . . 50
3.7 EM Simulation Drive-Cycle Data Plot . . . 51
3.8 EM Simulation Input and Output Data Plot Section . . . 51
3.9 EM Model Fidelity Options . . . 51
3.10 EM simulation results calculation block . . . 52
3.11 EM Model Fidelity Option #1 . . . 52
3.12 EM Model Fidelity Option #2 . . . 53
3.13 EM Model Fidelity Option #3 . . . 53
3.14 EM Model Fidelity Selection . . . 54
3.15 EM simulation outputs vs. time graphs . . . 54
3.16 EM Simulation Output as Torque vs. Speed variation . . . 55
3.17 Torque vs. Speed data for Model Option #3 . . . 55
3.18 Efficiency plots for the highest fidelity model option . . . 55
3.19 E-Powertrain Superblock Inputs & Outputs . . . 56
3.20 E-Powertrain block diagram . . . 57
3.21 Inputs for the E-Powertrain Model Simulation . . . 58
3.22 Results of the E-Powertrain Simulation with the predefined energy, torque, and motor speed test values. . . . 58
3.23 Battery system is based on 2 consecutive subsystems. . . . 59
3.24 Inverter / Converter Superblock Inputs Outputs . . . 60
3.25 Battery Pack Superblock Inputs & Outputs . . . 60
3.26 Inverter / Converter Model Block Diagram. . . . 61
3.27 Battery Pack Subsystem's block diagram . . . 61
3.28 Coulomb Counting Method for Battery Pack . . . 62
3.29 Voltage LUT with Coulomb Counting Method for Battery Pack. . . . 62
3.30 Battery Management System block diagram . . . 63
3.31 Power Demand input data for Battery Pack simulation . . . 63
3.32 Battery Pack simulation outputs . . . 64
3.33 Battery Pack Simulation - SoC Image . . . 64
3.34 EV Full Simulation Model (1-D) . . . 65
3.35 Drivecycle data used in the Full EV Simulation. . . . 66
3.36 Battery Pack SoC vs. distance traveled . . . 66
3.37 Vehicle's desired speed vs. actual speed. . . . 66
3.38 Power output of the EM during the drive cycle. . . . 67
4.1 Input for multibody simulation . . . 69
4.2 1-D systems available in MotionSolve . . . 70
4.3 MotionView simulation model of the Vehicle . . . 71
4.4 A double lane change event . . . 71
4.5 Behavior of baseline model at 60 kmph . . . 71
4.6 Behavior of baseline model at 65 kmph . . . 72
4.7 Yaw angle of the vehicle . . . 73
4.8 YAW angle variation at two different speeds . . . 73
4.9 DOE study chart . . . 74

4.10 GRSM to improve the design of the vehicle . . . 74
4.11 CAD models and the results after GRSM change . . . 75
5.1 Altair Student License . . . 76
5.2 Altair Connect . . . 77
5.3 Download Panel . . . 77
5.4 HyperWorks Download Panel . . . 77
5.5 License location . . . 78
6.1 Activate GUI . . . 79
6.2 Project Browser . . . 80
6.3 Property Editor . . . 80
6.4 Menus tabs . . . 81
6.5 Simulation Parameters . . . 81
6.6 Model Initialization . . . 82
6.7 Diagram/Context Initialization . . . 82
6.8 The two ways to select blocks for Super Block Creation (L & R) . . . 83
6.9 Super Block generation: Super Block icon or right click . . . 83
6.10 A port block . . . 83
6.11 Ports inside a Super Block . . . 84
6.12 Scope Palette includes different Scope options . . . 84
6.13 Preferences panel . . . 85
7.1 RTV Simulator Folder and Files . . . 86
7.2 RTV Simulator - Default View . . . 87
7.3 RTV - Top Ribbon and Camera Selection . . . 87
7.4 Activate GUI . . . 87
7.5 Activate GUI . . . 88
8.1 .mdl file for MotionVIEW simulation . . . 89
8.2 Result Tables in HyperStudy . . . 90
8.3 Double Lane Change Event Activation for 3-D Simulation . . . 90
8.4 Changing Simulation Parameters in Event Editor . . . 91
8.5 Console Solver View for MotionVIEW Simulation . . . 91
8.6 Hyperworks Simulation Player . . . 92

List of Tables

1 Vehicle Coordinates Nomenclature 13
2 Forces and their description for the force balance equation 16
3 Port Definitions for the Activate Wheel Block Models 27
4 Forces and descriptions of the force balance equation 31
5 Constants and Masses for the 2-DOF Quarter Car Vertical Model 38
6 Coordinates, Constants, and Masses of the Full Car Model 43
7 Applications of multibody simulation in automotive industry 69

SECTION 1

Introduction

1.1 Background

Since the 2000s, the automotive industry started offering electrical vehicle (EV) options to the consumer market in more affordable and available numbers. With the use of lithium-ion batteries in production cars, companies started to see a good market response from consumers for EVs. Based on the report [1] of the International Energy Agency, EV car sales are around 8.57% as of 2021 globally, more than triple their market share from two years earlier.

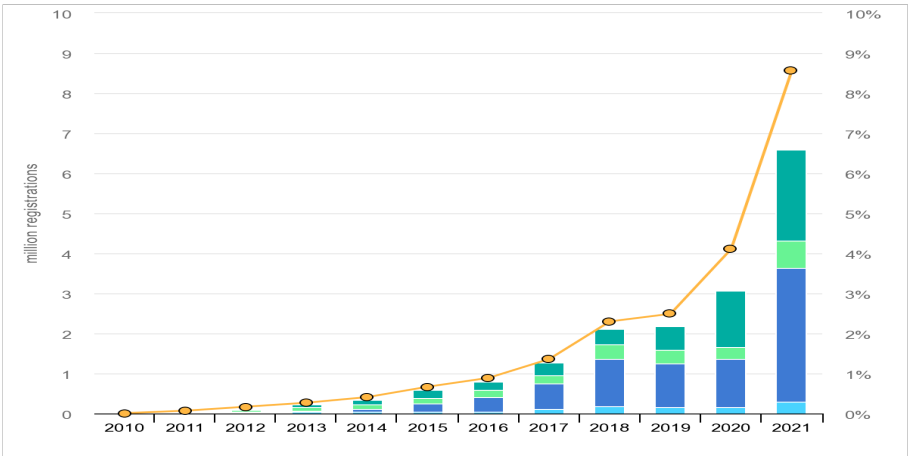


Figure 1.1: *EV global market share during 2010-2021. Global sales and sales market share of electric cars, 2010-2021 (The image is courtesy of IEA)*

Due to the increasing demand in the market, there is also an increase in the supply of EVs. According to a McKinsey report [2], in 2019 only, automakers launched 143 new electric vehicles (105 battery-operated EVs (BEVs) and 38 plug-in hybrid electric vehicles (PHEVs)).

Although the exterior body and the passenger compartment of the EVs do not change extensively, the majority of the powertrain components go through an update process. This is a massive step in the

automotive industry; hence there are different methodologies between manufacturers and OEMs.

From an educational and personnel training perspective, although the fundamental knowledge and essential learning tools for the components (such as batteries, energy harvesting systems, chargers and inverters, cooling systems, etc.) are already available, there are few comprehensive resources covering all the major components of an EV as an integrated powertrain solution. Furthermore, vehicle manufacturers typically use sophisticated CAD/CAE tools with cutting-edge simulation capabilities; however, these tools are not readily available to the public; hence they cannot be used for academic teaching or research in non-commercial scenarios.

Another trailing point is that there is not a ready-for-use entry-level EV simulation software supported with a mathematical background that is also open for modifications. Even though the fundamental simulations for Internal Combustion Engine (ICE) vehicle models are available, they can not be used and modified easily without extensive software modification effort.

Last but not least, a step-by-step guided tutorial for running and explaining how to customize visual vehicle simulations is also missing. Software manufacturers provide tutorials or example files. However, the topics are not interconnected with each other, and only limited aspects of the vehicle operation are covered.

1.2 Motivation

Based on the previous background section, the motivation of this book could be explained with the following items:

Why is this e-book created?

- First and with utmost importance, EVs are an emerging and popular topic in today's engineering world. There are wide but fragmented online resources covering the fundamentals of how EVs operate. The main motivation of this book is to provide comprehensive content explaining the primary components of EVs and their operation integrated with vehicle dynamics features for realistic simulations.
- We believe our e-book will be an essential source for academia, since;
  - Simulation and visualization parts will be provided as part of the content,
  - We will provide how to use/develop the given simulations,
  - Visual simulations with vehicle dynamics and EV characteristics will provide an in-depth analysis perspective even for entry-level professionals or students.
- For researchers, our e-book and its supplements provide open-source or re-usable content available for non-commercial usage.
- Topics such as Co-Simulation, and Functional Mock-Up Interface (FMU or FMI), which are heavily used and relied on in the automotive industry, are not explained much in the academic curriculum.

What this e-book is offering?

- One-stop introductory document for understanding how an EV works and provides easy-to-use content with vehicle dynamics content.
- E-Book is prepared under the partnership of Altair Inc. It is based on Altair's model-based design (MBD) software called Activate, which is very intuitive and easy-to-use simulation software.
  - Quick start guides for the Altair software are available for first-time users;
  - Integrates and explains how to use different 3D simulation tools with Activate. 3D simulations are not only more attractive for students but also help analyze the vehicle's motion and in some cases the force/momentum relationships with a more in-depth perspective.
  - To achieve such an integration interactive to the users, explanatory content is provided for industry standards such as **Co-simulation and FMU.**

1.3 Scope

This e-book provides introductory background on how an EV operates and how its dynamic behavior can be simulated. Although the focus of this book is developing useful simulations rather than providing technical results, a considerable amount of technical information is also provided as background in each chapter. The chapters are presented with the following outline.

Each chapter,

- defines the most important parts of the vehicle physics (like drag force, traction, steering angle, torque) and EV parts (battery, battery management system, electrical motor, etc.);
- provides the basic analytical equations for vehicle dynamics and EV components (where available);
- provides example Activate simulation models, based on the analytical equations or system model/sample system data for the EV components;
- explains the models and their customizable parameters in depth;
- provides the expected run results and 3D animation screenshots (where available) based on the default parameters;
- assigns do-it-yourself-exercises, and a set of questions are directed to try and analyze different results;
- 1x Case Study in the final chapter to analyze the effect of the subsystems on the mileage of an electrical vehicle.

1.4 References

[1] IEA (2022), Electric cars fend off supply challenges to more than double global sales, IEA, Paris. https://www.iea.org/commentaries/electric-cars-fend-off-supply-challenges-to-more-than-double-global-sales.

[2] McKinsey (2020), McKinsey Electric Vehicle Index: Europe cushions a global plunge in EV sales, written collaboratively by members of McKinsey's Automotive and Assembly Practice: Thomas Gersdorf, Patrick Hertzke, Patrick Schaufuss, and Stephanie Schenk. https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/mckinsey-electric-vehicle-index-europe-cushions-a-global-plunge-in-ev-sales.

SECTION 2

Vehicle Dynamics Overview

2.1 Introduction

To answer the "How does a modern vehicle work?" question, a solid understanding of vehicle dynamics, i.e., physics rules apply to vehicles during motion, is needed. Therefore, before going into the details of the electrical propulsion system (also known as the electric powertrain system) in modern vehicles, we first introduce an overview of vehicle dynamics in this chapter.

Vehicle dynamics is a multidisciplinary engineering subject and involves components from propulsion, braking, aerodynamics, and tire dynamics. Different approaches exist to analyze some of these topics. Advanced technical analysis and design methods are given in references such as [6] in detail. In this chapter and throughout this book, we have selected the typical components suitable for our target audience. This chapter aims to provide a condensed summary to build crucial vehicle dynamics knowledge. The discussion will start with introducing a common motion framework called **Vehicle Coordinate System**. Next, we assemble the common theoretical principles around each coordinate direction. We select a bottom-up approach and progressively increase the detailing while using the components of the previous sub-chapters. In each sub-chapter, a basic mathematical model is introduced, and an analytical vehicle model is generated with a simulation example in Altair's Activate software. The results of the simulations are then analyzed. To better visualize the simulations, additional 3D simulator software will also be used.

The order in which each sub-chapter is presented can be listed as follows:

1. We will start with the primary vehicle direction called **Longitudinal Vehicle Motion**. We introduce the forces affecting the vehicle in this direction and their effect on the acceleration/deceleration, velocity, and distance of the vehicle, i.e., the dynamics in the longitudinal movement.

2. Then, we introduce the heading (including the yaw angle) movement called the Lateral Movement. The **Lateral Vehicle Dynamics** analysis includes components from the longitudinal vehicle motion part, as forward movement is required to change the heading angle.

3. Next, we introduce the vertical (including the pitch angle) movement plane and introduce the **Vertical Vehicle Dynamics**. Again, the vertical motion dynamics also include components from the

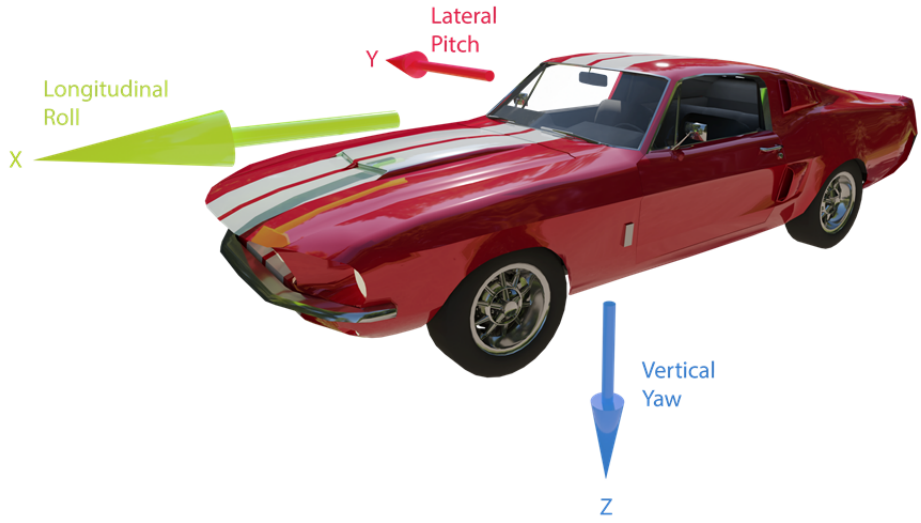


Figure 2.1: *Vehicle Coordinate System vectors*

Axis	Translational Velocity	Angular Displacement	Angular Velocity	Force Component	Moment Component
x	u (forward)	ϕ	p or ϕ' (roll)	F_x	M_x
y	v (lateral)	θ	q or θ' (pitch)	F_y	M_y
z	w (vertical)	ψ	r or ψ' (yaw)	F_z	M_z

Table 1: *Vehicle Coordinates Nomenclature*

longitudinal vehicle motion.

4. In the last **Combined Vehicle Dynamics** sub-section, we combine all of the previous separate Activate models and run a system-wide simulation to analyze the full vehicle motion. We also utilize a 3D simulation to simulate the vehicle and the road to make it more visually distinctive.

To better understand what longitudinal or lateral motion means, we will need to introduce the Vehicle Coordinate System (VCoS in short) as shown in Figure 2.1. VCoS is used to introduce standard coordinates and notations for describing vehicle dynamics. The following image describes the coordinate system defined by the Society of Automotive Engineers (SAE) standard:
The common VCS framework has the following definitions:
Pitch angle: the angle between the x-axis and the horizontal plane.
Roll angle: the angle between the y-axis and the horizontal plane.
Yaw angle: the angle between the x-axis and the X-axis of the inertial frame.

Fixed Coordinates and Vehicle Slip: In order to analyze the vehicle motion entirely, definitions for the course angle, the heading angle, and the sideslip angle of the vehicle, observed from a fixed reference point (i.e., Origin - O) should be understood. These definitions are given based on the earth-fixed axis system and an origin that lies in a reference ground plane. Since OXYZ is fixed on Earth, it does not turn with the vehicle. However, the car's heading rotates as it moves, slips, etc.

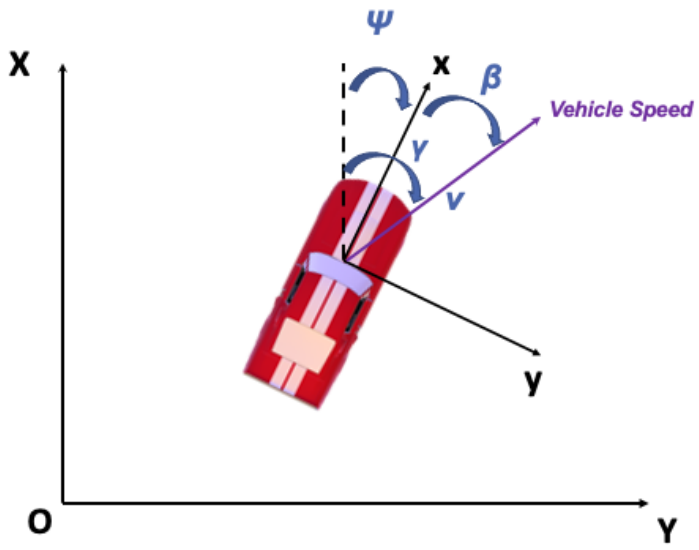


Figure 2.2: *Heading, slip, and course angles of the vehicle*

As shown in Figure 2.2, the angles are defined as; $\boldsymbol{\gamma} = \boldsymbol{\Psi} + \boldsymbol{\beta}$

where;

$\boldsymbol{\gamma}$ = Course angle (in degrees)

$\boldsymbol{\Psi}$ = Heading angle (in degrees)

$\boldsymbol{\beta}$ = Side slip angle (in degrees)

The course and heading angle of the vehicle are approximately equal during longitudinal motion unless the tires are excessively slipping in sideways, such as the drift action.

2.2 Longitudinal Vehicle Motion

2.2.1 Brief Description

The x axis is primarily horizontal in the vehicle plane of symmetry and points forward (heading) of the vehicle as shown in Figure 2.3 below. In the case of longitudinal motion analysis, it should be considered tha

Figure 2.3: *Longitudinal motion vectors.*

The forces acting on the car during forwarding movement are in the vertical (Z) direction and in the longitudinal (X) direction. When the driver applies force via engine (or brake) power, the force balance changes, resulting in acceleration (or deceleration) both in the longitudinal direction and also in the vertical direction (as a change in its **pitch angle**).

The longitudinal vehicle dynamics are mainly influenced by power generation (engine) and transfer (transmission, tires) devices. As the vehicle contacts the ground on its four wheels, there will be a force/moment interactions at the wheels. These interaction equations (explained in Section 2.2.3) helps calculate the wheel speed, vehicle speed, resulting acceleration/deceleration, and also the vehicle's pitch angle influenced by the acceleration or deceleration profile.

2.2.2 Descriptive Figure

If we look into the force balance in the longitudinal directions in detail, we notice that some force vectors, moment arm lengths, and angles are important for consideration. These vectors and related constants are described with the below figure (Figure 2.4) using the descriptions in Table 2). Detailed discussions are also available [1] and [2].

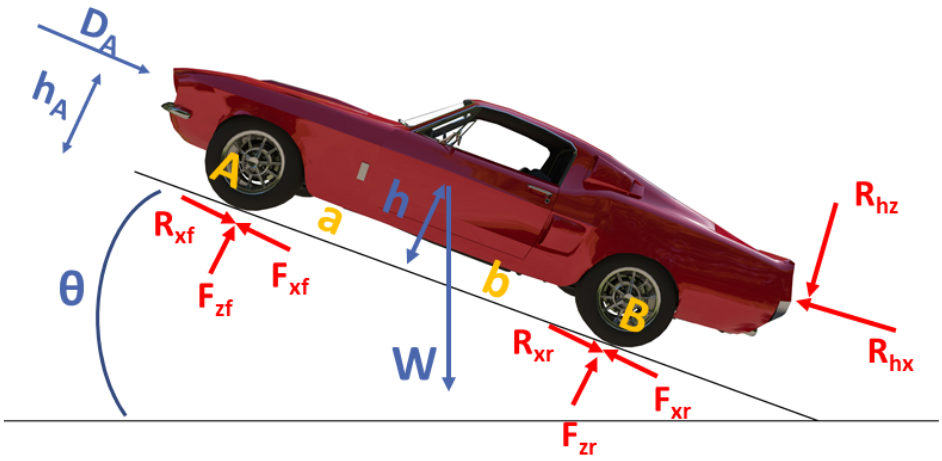


Figure 2.4: *The forces and the dimensions of the vehicle are used in the force balance equations*

This description in Figure 2.4 aims to study the force, that accelerates (or decelerates) the vehicle in the x direction.

Abbrev	Constant /Force Vector Name	Description
g	Gravity constant	—-
D_A	Aerodynamic drag force	The force occurs due to the aerodynamic repulsion. Can be considered as the resistive force due to wind or air displacement.
h_A	Height of the aerodynamic drag force	This is the average height of the effective aerodynamic drag force
λ	Road's slope (degrees)	The road's inclination angle

R_{xf} and R_{xr}	Rolling resistance force for front and rear	Energy dissipation due to wheel's rotation and deflection.
F_{xf} and F_{xr}	Tractive Force for front and rear	This is the force generated with the engine or brakes.
F_{zf} and F_{zr}	Tire normal forces for front and rear	Vertical force applied from the ground to the wheel.
W	Weight of the vehicle (= mg)	—
h	Height of the CoG	The weight center of the vehicle in its actual state
A, B	Tire contact points	Contact points of the tire, where the moment balance is calculated for the force balance calculation in the vertical (z) direction
a, b	Moment arm lengths for front rear	These lengths are used in the calculation of the moment due to various forces acting on the wheels.
R_{hx} and R_{hz}	Draw bar force in the longitudinal (x) and in the vertical (z) direction	Also known as the hitch force
a_x	Motion component	Acceleration of the vehicle in the x direction.

Table 2: *Forces and their description for the force balance equation*

These forces can be considered in two parts: The tractive forces (i.e., F_{Xf} and F_{Xr}) are the inputs of the force balance equation and generally initiate the motion, The rest of the forces are disturbances to the system's balance. The aerodynamic drag (D_A), the rolling resistance force, and the draw bar force are examples of disturbance to the force balance that designers have little control over for analysis in real applications.

2.2.3 Equations

In the forward (i.e., longitudinal) motion analysis, the vehicle body is considered only to have move in 2 different axes. These are the vertical axis (i.e. Z-axis), and the forward (i.e., longitudinal or X) axis.
The system considered by the inclusion of the vehicle body and the tires will be considered to be in static equilibrium in steady state motion conditions. To analyze the movements in the vertical axis, the force balance equation is used. As there will not be a net vertical acceleration, the equivalent force in the vertical direction should be zero. The forces that act in the vertical (Z) direction are given as:

$$0 = W cos\theta - F_{zf} - F_{zr} + R_{hz} \tag{1}$$

If we neglect the draw bar force (i.e., R_h), this equation can be used to calculate the static load of the car.

During acceleration or deceleration, due to the spring effect, the load of the car is split between the front and rear tires, as can be seen in the following figure

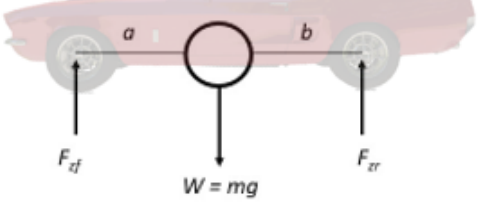


Figure 2.5: *Forces and balances around the center-line*

The force balance equation in the X direction is given as the following:

$$ma_x = (W/g)a_x = F_{xr} + F_{xf} - W sin\theta - R_{xr} - R_{xf} - D_A + R_{hx} \tag{2}$$

The aerodynamic drag force (D_A) is calculated using the below formula:

$$D_A = \frac{\rho\, C_d\, A\, V^2}{2} \tag{3}$$

where ρ is the density of the surrounding air, A is the cross-section of the vehicle (a design feature), C_d is a constant drag-coefficient number, which is also a design feature, and finally, V as the actual speed of the vehicle.

The Rolling Resistance Force (R_{xf} or R_{xr}) is calculated with the following equation [3]:

$$(R_{xf} \text{ or } R_{xr}) = C_{rr}\, N \tag{4}$$

where C_{rr} is the dimensionless rolling resistance coefficient, and N is the normal force, the force perpendicular to the surface on which the wheel is rolling.

To calculate the amount of normal force change between the front and the rear axle during acceleration and deceleration, we will need to calculate the load transfer using the following formula:

$$\text{load transfer} = W\left(\frac{a_x h}{gL}\right) = \frac{ma_x h}{L} \tag{5}$$

2.2.4 Model Figures

In this Chapter, readers are provided with three examples of Activate model files for simulating the longitudinal movement of the car, as in this order:

- Longitudinal Simulation Version #1 - Force Balance Model Only (Figure 2.6)
- Longitudinal Simulation Version #2 - Force Balance + Load Transfer Model (Figure 2.7)
- Longitudinal Simulation Version #3 - Force Balance + Load Transfer + Wheel Dynamics Model (Figure 2.8)

As can be noted with the naming convention, models have been presented in a progressive development schema. They are also named with increasing version numbers to represent this progressive development.

Like every example model given in this book, the models have a cover page (i.e., top-level superblock) introducing the model and its purpose, as in the below-left figure. Users are encouraged to double-click on the top-level superblock(s) and investigate the algorithm and the building blocks underneath, as in the below-right figure.

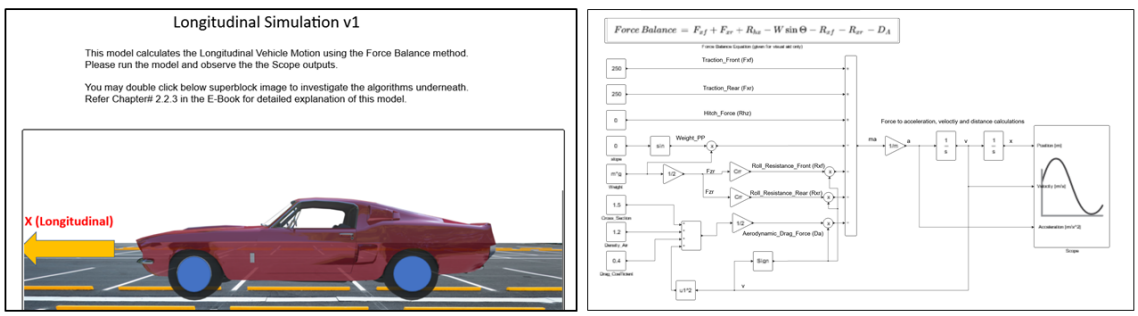


Figure 2.6: *Top Level (Superblock) and Interior View for the Longitudinal Simulation v1 model. On the left is the top-level superblock. On the right the interior of the top-level superblock is given.*

....

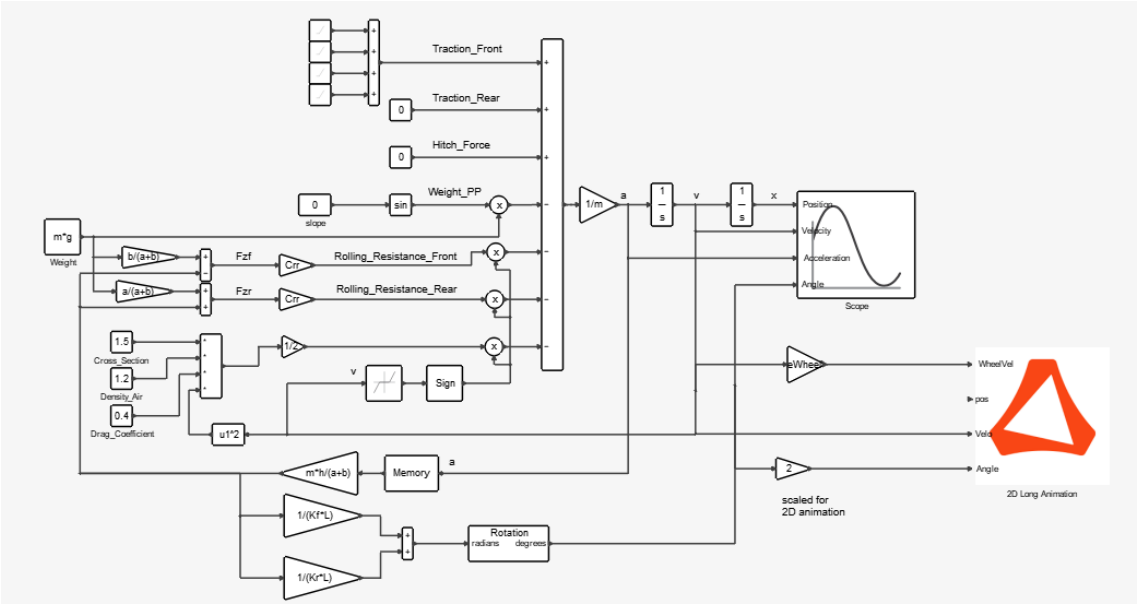


Figure 2.7: *Longitudinal Simulation v2 model.*

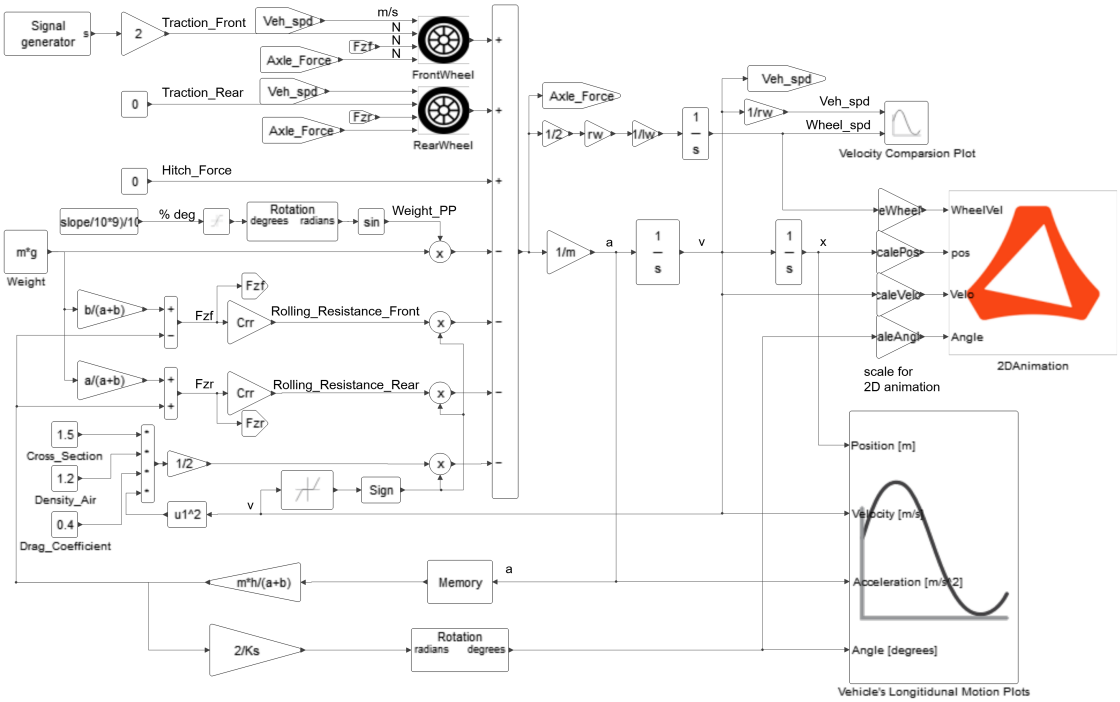


Figure 2.8: *Longitudinal Simulation Model (Version #3). This model is based on* 5 *different parts. Each part is separated with a different color code. More info is available in the next Model Overview section.*

Models have a common structure. Each model has the force inputs on the left-hand side, calculations in the middle, and the visualization (Scope and/or 2D animations) on the right-hand side. We also provide the relevant mathematical equation and the sketch of the force vectors in the top section of the models. Detailed information for each model will be presented in the next section.

2.2.5 Model Overview

This section will provide more information about the models and provide the relationship between the models and the equations.

The v_1 model is created for analyzing a simplified force balance equation to understand the acceleration, and velocity profiles of a vehicle without the load transfer and wheel dynamic effects. In a nutshell, this example is the simplest form of force vs vehicle motion. Further information on this model is given in the subsequent sub-chapter of 2.2.5.1.

The v_2 model (in Section 2.2.5.2) adds the axle load calculations, which include the load transfer calculations of the given equation X. In this version, the behavior of the car becomes physically more realistic; hence we also add a 2D animation representing the movement of the car.

In the last version of the example (Section 2.2.5.3), we will add the last component, which is the wheel dynamics and their effect on the motion. As we will re-use this model in the final vehicle model, we will spend some time explaining all the building blocks separately.

2.2.5.1 Longitudinal Simulation Version#1

Use Cases of this Model: The longitudinal vehicle models represent the simplest form of vehicle motion and will be helpful for you to understand the simplest Force vs. Acceleration relationship of the vehicle while omitting other vehicle dynamics parameters. The v1 version is an exclusively analytical model and it can also be used to solve basic Newtonian Physic motion questions as well.

Action Item: Please locate and open the simulation file named Longitudinal Simulation v1.scm in the Altair Activate software.

Further Reading: Please check *Appendix A) Getting Started with Activate* for a quick introduction to the software. There are plenty of resources available on Altair University portal [4] as well.

The Longitudinal Simulation Version #1 is our first simulation model, and we will be sharing detailed information on all of the basic blocks. The model is created to calculate the following Total Force equation.

$$\text{Total Force} = F_{xf} + F_{xr} + R_{hz} - W\,sin\theta - R_{xf} - R_{xr} - D_A \tag{6}$$

Please note that this equation takes into account the X coordinates only. Hence, the inputs and outputs of the model are only related to these coordinates.

For documentation purposes, a screenshot of the model is given below, and some portions of the image are segmented with color codes and numbers. Please note that this coloring is applied via an external image processing software, and colored boxes are not available in the Activate software.

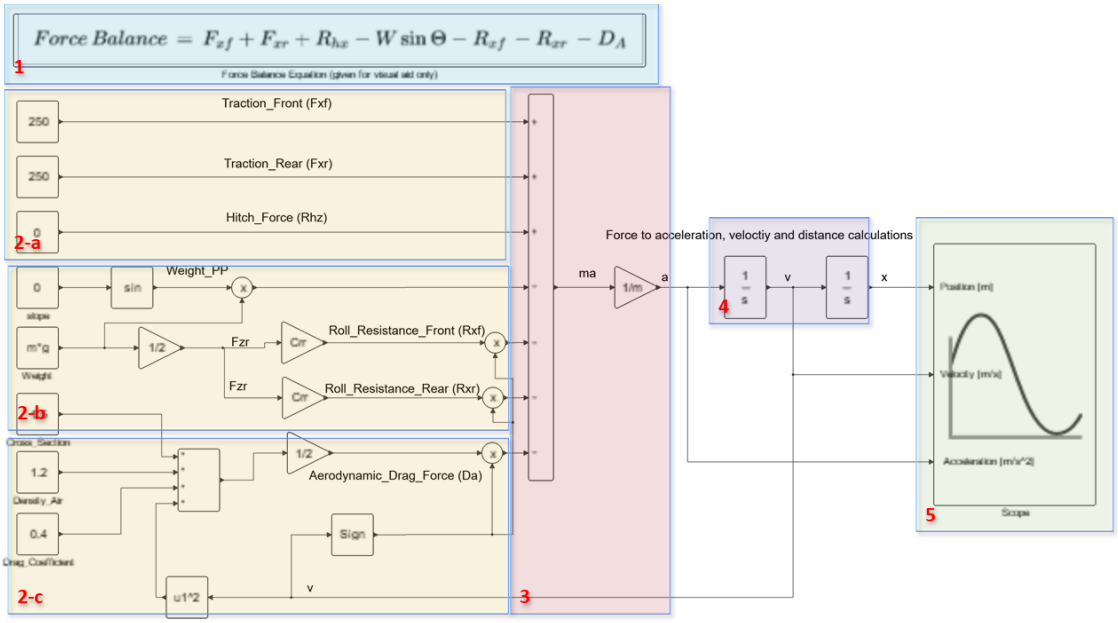


Figure 2.9: *Longitudinal Simulation Model (Version #1). This model is based on 5 different segments. Each part is separated with a different color code.*

The Segment **#1** (blue box) of the model is for visual aid only. Similar visuals are placed on most of the models given in this book. The equation(s), the angles, the vector information, etc. are placed in this visual to show how "this model" is structured quickly.

The Segment **#2-a, b, c** (orange boxes) represent the force values on the right-hand side of the above Force Balance equation. The details are as follows:

- The Segment **#2-a** represents the positive forces including the Front and Rear tractive (ie. engine) forces called F_{xf} and F_{xr} and the hitch force (R_{hx}). Please note that our model (with default values) has an all-wheel-drive configuration with an input of 250 Newtons of tractive forces each and has 0 (zero) Newton hitch force. Readers can change these forces to see the difference in the outcome, which has a linear relationship.

- The Segment **#2-b** represents the positive forces including the negative forces except for the aerodynamic drag. 3 weight-related forces exist in this segment. These are the perpendicular weight (Weight$_{PP} = mgsin\theta$) and the front and rear rolling resistance forces (R_{xf}, R_{xr}).

  Please note that the C_{rr} is given as a parameter that can be quickly changed using the Activate's model initialization property window, as can be seen in the below figure.

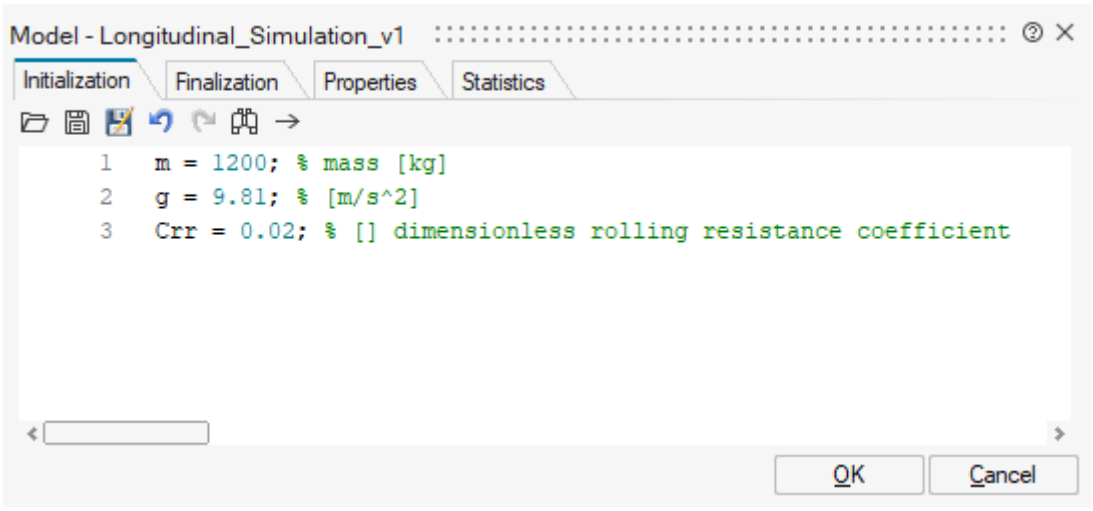


Figure 2.10: *Initial parameter values for the model. Parameters can be quickly changed using Activate's model initialization window.*

– The Segment **#2-c** represents the last negative force, the aerodynamic drag force (or drag in short). The Drag force is calculated using the 4 different inputs as can be seen in the Segment. From top to bottom, it is dependent on the cross-section of the vehicle (a design feature), the density of the surrounding air, a constant drag-coefficient number -which is also a design feature- and finally the actual speed of the vehicle.

  Please note that all negative forces in the Segment#2 are multiplied by the sign of the vehicle's speed. So that they are counteracting to the opposite of the vehicle's speed, i.e., if the speed is positive (in the X direction), then the negative forces will have the negative sign, but if the speed is negative (with respect to the X direction), then the negative forces will have the positive sign.

The Segment **#3** (red box) is the summation of all the forces involved. After the summation, we divide the net force by the mass of the vehicle, which provides the net acceleration amount.

In the Segment **#4** (purple box) we derive the velocity and the distance traveled. This is achieved by applying an integral to the previously calculated acceleration value -which gives the instantaneous velocity- and then integrating the velocity value to achieve the distance traveled.

In the last Segment #5 (green box); we use a Scope block to plot the instantaneous values over the period of the simulation, which is set to 100 seconds by default.

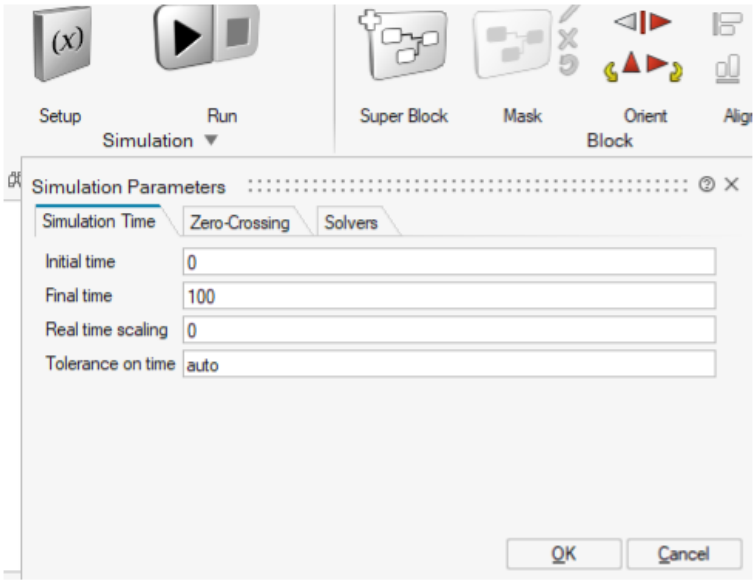


Figure 2.11: *The simulation time is set to 100 seconds by default. It can be changed via the **Setup (x)** button available on the toolbar.*

2.2.5.2. Longitudinal Simulation Version #2

Next, the version of our model calculates the overall Axle load, which covers the effect of the axial loads in the longitudinal calculations. This would help us to see the rolling movement of the vehicle.

Use Cases of this Model: The longitudinal vehicle models represent the simplest form of vehicle motion and will be helpful for you to understand the simplest Force vs. Acceleration relationship of the vehicle while omitting other vehicle dynamics parameters. The v2 version includes the weight transfer calculation due to acceleration or deceleration. Automatic weight control systems used in the ballast tanks of the vessels also have similar calculations to counter-act the porpoising motion for the comfort of the passengers or for the protection of their cargo.

Action Item: Please locate and open the simulation file named Longitudinal Simulation v2.scm in the Altair Activate software.
The load transfer is calculated with the following equation;

$$\text{load transfer} = W\left(\frac{a_x h}{gL}\right) = \frac{m a_x h}{L} \tag{7}$$

In this v2 model, in principle, we will change all the weight-related calculations using the outcome of the above load transfer calculation, which is calculated in the Segment **3-b**. More info is available below.
Similar to the previous (v1) model, we have also color-coded and segmented the model file for documentation purposes.

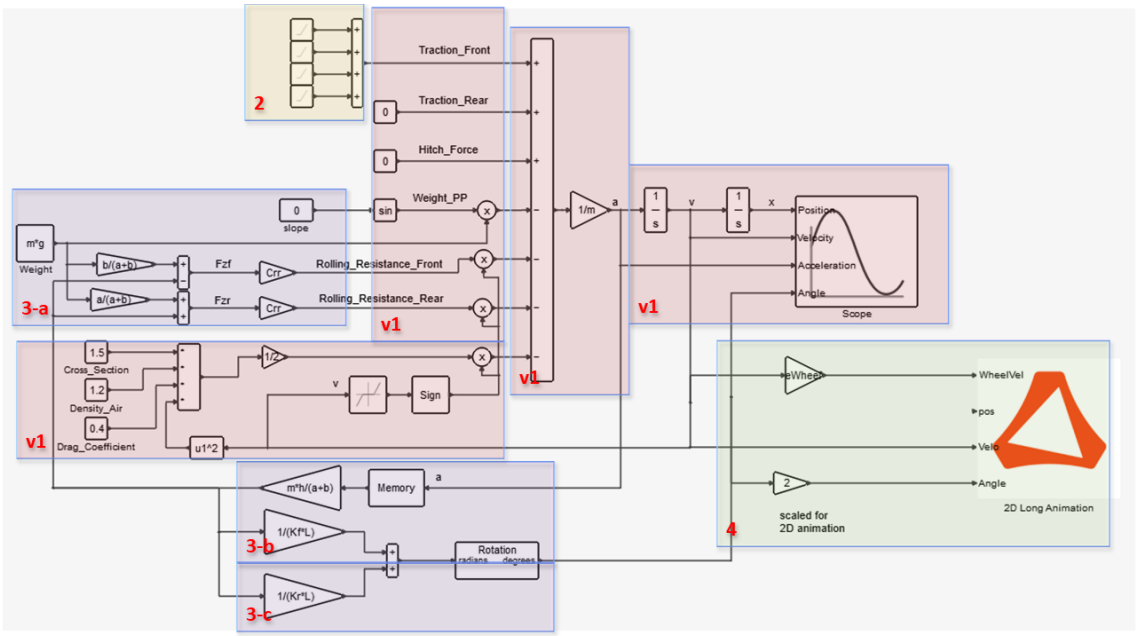


Figure 2.12: *Longitudinal Simulation Model (Version #2). This model is based on 3 new segments when compared with Version #1. Each part is separated with a different color code.*

This time we have highlighted the un-changed blocks with red color and added the "v_1" text to the bottom-left corners. We have three new/updated segments, which are described below.
The Segment **#1** (not shown in the image) is for visual aid only. Compared to Version#1, we have added an image of the load transfer equation only.

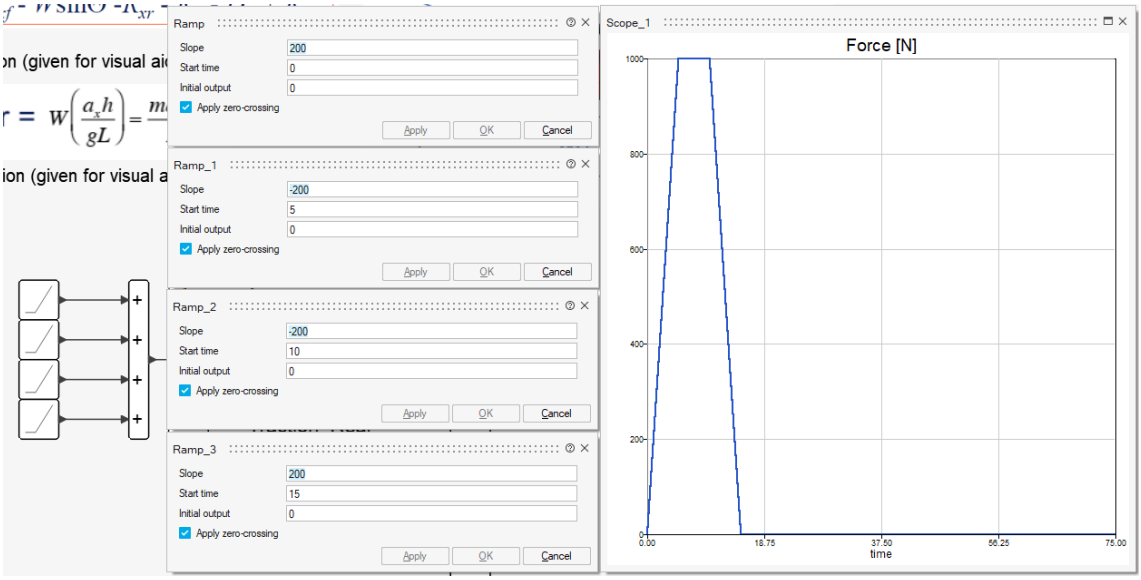


Figure 2.13: *Force signal pattern based on the summation of equivalent-timed ramp patterns (left). The resulting signal is an acceleration, cruise, and deceleration staircase pattern (right image).*

The Segment **#2** (yellow box) is used to apply a force pattern using four separate linear ramp sections as 0→200→-200→200 Newton forces. Please refer to the above image on how the force pattern is created and what the equivalent time signal looks like.

The Segment **#3-a, 3-b, 3-c** (purple boxes) is the main part where the load transfer affects the force balance and roll angle. The details are as follows:

– The Segment **#3-a** uses the transferred load (which is calculated in Segment (#3-b) and combines it with the perpendicular weight to calculate the front and rear rolling resistance forces (R_{xf}, R_{xr}) under the effect of the load transfer.

  The memory block in this part is used to introduce an initial condition of "zero" acceleration to the load transfer calculations. Removing this block would create an algebraic loop, which prevents the model from running. Users are encouraged to Disable/On/Off this block to see the difference and why this block is needed.

– The Segment **#3-b** is added to calculate the equivalent load using the preliminary given load transfer formula.

– The Segment **#3-c** is used to calculate the pitch angle that happens due to acceleration or braking events. The transfer of load from the rear to the front axle that takes place during a braking event will cause the vehicle body to rotate about its lateral axis. This pitching motion also results in a change in the height of the vehicle's center of gravity. The calculation uses the spring constant (K_s), and it is given with the following formula:

$$\text{Pitch Angle(in radians)} = 2W_t / K_s \tag{8}$$

  Then we will use the radians to degree conversion block to convert for the next animation section.

– The Segment **#4** (green box) is used to visualize the movement of the car in a 2D animation window available in Activate. The 2D Long Animation block is created for visualizing the force vector and also shows the changes in the pitch angle of the vehicle. More information is given in the Results section.

  Similar to the previous model (also for the future models as well), we have some new initial state parameters used for calculations. New parameters are separated into a new section of the parameter initialization page, as seen below.

Model - Longitudinal_Simulation_v2

Initialization | Finalization | Properties | Statistics

```
% Calculation parameters used in the v1 model
m = 1200;  % mass of the vehicle [kg]
g = 9.81;  % Gravitational constant [m/s^2]
Crr = 0.02; % [] dimensionless rolling resistance coefficient

% Calculation parameters added in the v2 model
a = 2.1;   % Front axle distance from the Center of Gravity [m]
L = 3.7;    % Overall length of the car [m]
b = L-a;   % Rear axle distance from the Center of Gravity [m]
h = 0.4;   % Center of Gravity Height (m)
Kf = 4e3;  % Front Suspension Linear Spring Constant
Kr = 4e3;  % Rear Suspension Linear Spring Constant

% Below parameters are used in the 2D animation only
scaleWheelVel = 1/3;  % Scale for animating the wheel's rotation vs. the vehicle speed.
rw = 0.2;  % Effective Wheel Radius [m]
slope = 0; % the slope of the road. Kept 0 in this model, will be changed in v3.
```

Figure 2.14: *Initial parameter values for the model. Previous v1 parameters are in the top section, and new parameters of the v2 model are in the middle. There are also animation-related parameters at the bottom part.*

2.2.5.3. Longitudinal Simulation Version #3

In the final version, our model includes the wheel dynamics in the calculations. This will introduce an

important concept named the wheel slip, and we will compare the vehicle's speed versus the speed of the wheels which is the basis of various tire models that calculate the traction force on the wheel.

Use Cases of this Model: The longitudinal vehicle models represent the simplest form of vehicle motion and will be helpful for you to understand the simplest Force vs. Acceleration relationship of the vehicle while omitting other vehicle dynamics parameters. The v3 version includes the tire slip. Drift racing is all about how much the tire slips.

Action Item: Please locate and open the simulation file named Longitudinal Simulation v3.scm in the Altair Activate software.

The v_3 model has more additions over the v_2 model. In this last version, we add the axial load (based on the overall forces) and the wheel dynamics to the calculations. We will explain the changes in the below figure.

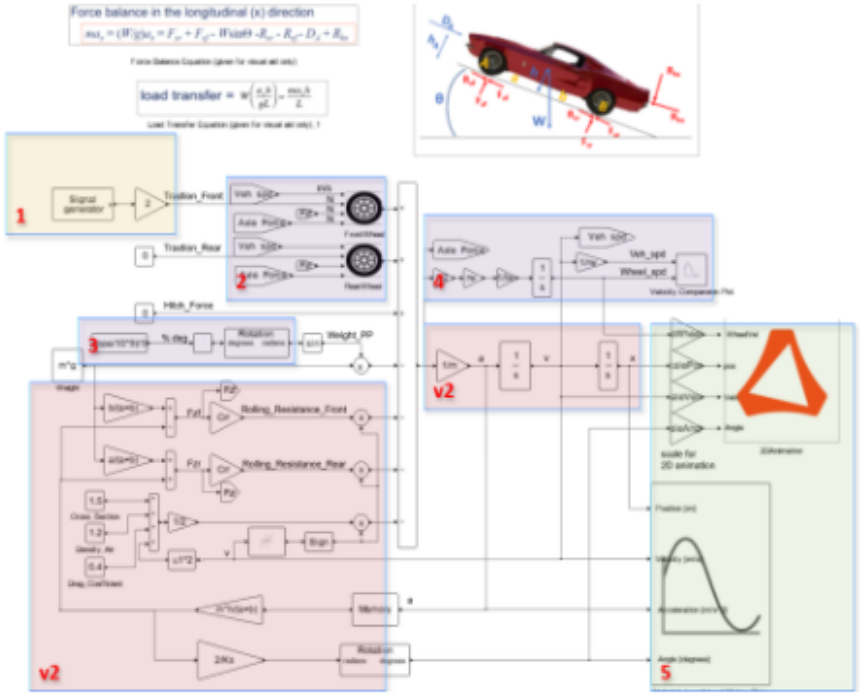


Figure 2.15: *Longitudinal Simulation Model (Version #3). This model is based on 5 new segments when compared with Version #2.*

The Segment **#1** (yellow box) is used to apply a force pattern, but this time using the Signal generator block. The default force pattern of the example is plotted in the below figure. Users are encouraged to try different force profiles. Please refer to the below image on what the force signal looks like. Please also note that the simulation time is now extended up to 100 seconds to cover the generated profile.

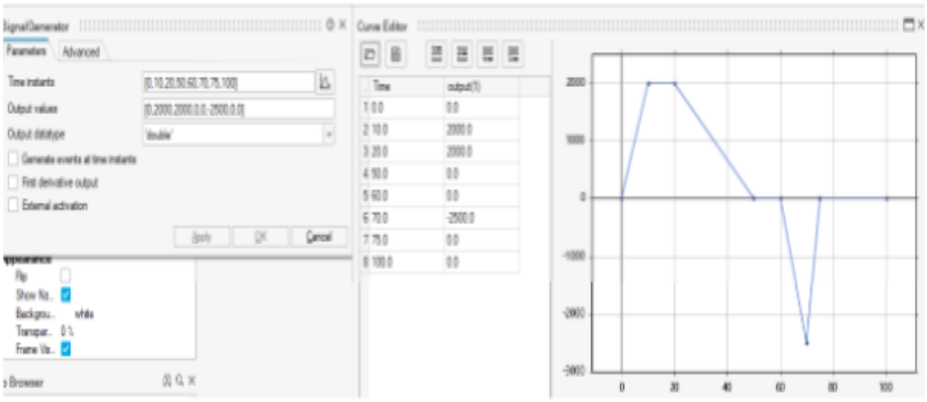


Figure 2.16: *Example Force pattern for the traction delivered to the wheels*

This force is the equivalent torque that is generated from the engine and modulated in the transmission and the final drive gears or the brakes.

The example model is prepared for a front traction vehicle, so the traction force is applied only to the front wheels. The brake power is also delivered to only one axle of the vehicle.

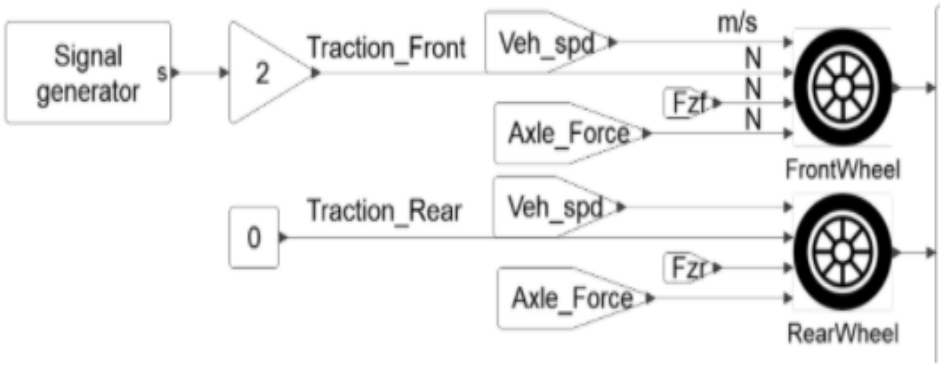


Figure 2.17: *Close-up for Longitudinal Model (Version#1) - Traction Force and Wheel Dynamics Calculations*

Next, we have two similar super blocks used for calculating each wheel's dynamics, named Front Wheel and Rear Wheel. Each block calculates the Wheel's dynamics with respect to the forces applied. The input and output ports of these blocks are identical and here is how they are named:

Signal Direction	Port No	Signal Name	Definition
Input	1	Veh_spd	Vehicle's Instantaneous Speed that is used for calculations
Input	2	Traction Front and Traction_Rear	Traction force applied to the wheels (depending on the vehicle's traction type)
Input	3	F_{zf} and F_{zr}	Tire normal forces for front and rear wheels
Input	4	Axle Force	An iterative variable holding the axle force
Output	1	F_{xf} and F_{xr}	Traction force limited front and rear wheel tires dynamics

Table 3: *Input & Output Port Definitions for the Activate Wheel Block Models (applicable to both Front Wheel and Rear Wheel blocks)*

The interior algorithm for each block is a saturation logic, so it is better to share the screenshot of each Wheel block and explain them hereunder:

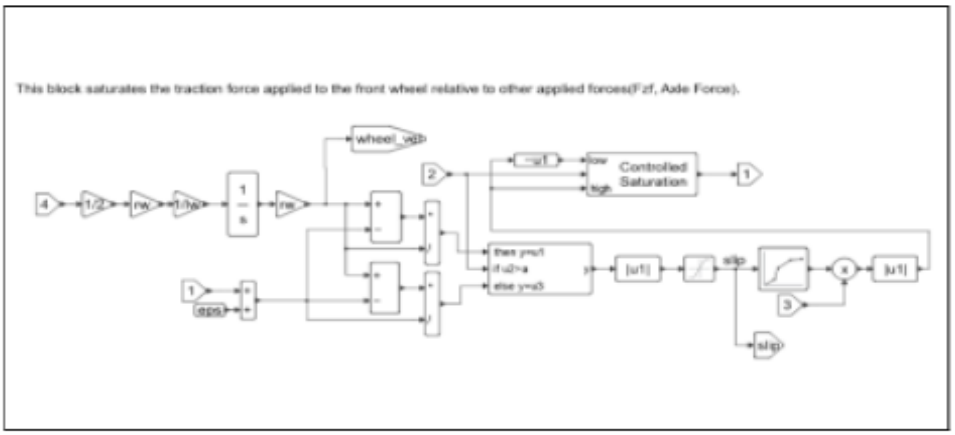


Figure 2.18: *Front Wheel Superblock*

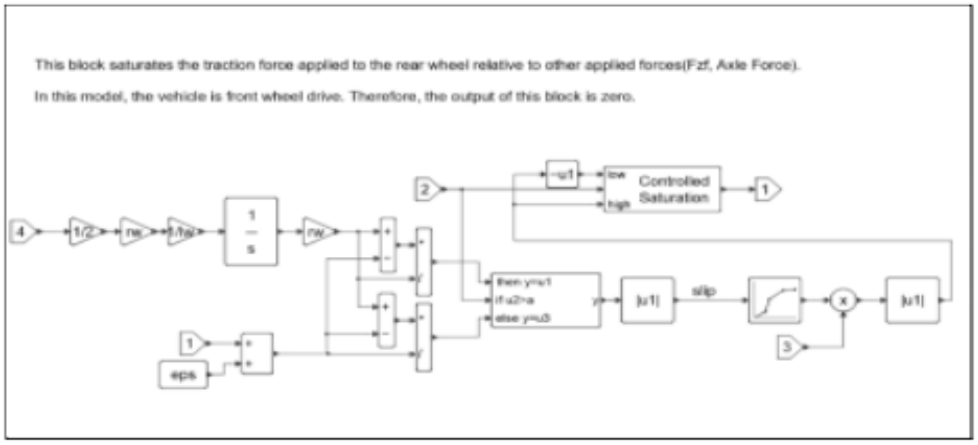


Figure 2.19: *Rear Wheel Superblock*

As the model is run, it will automatically launch a simple 2D animation, and draw some plots in the background. Users are encouraged to repeat the Run and observe how the velocity and motion plots

change synchronously with the 2D animation. The results of the simulation and their meanings are given in the next section.

2.2.6 Simulation Results

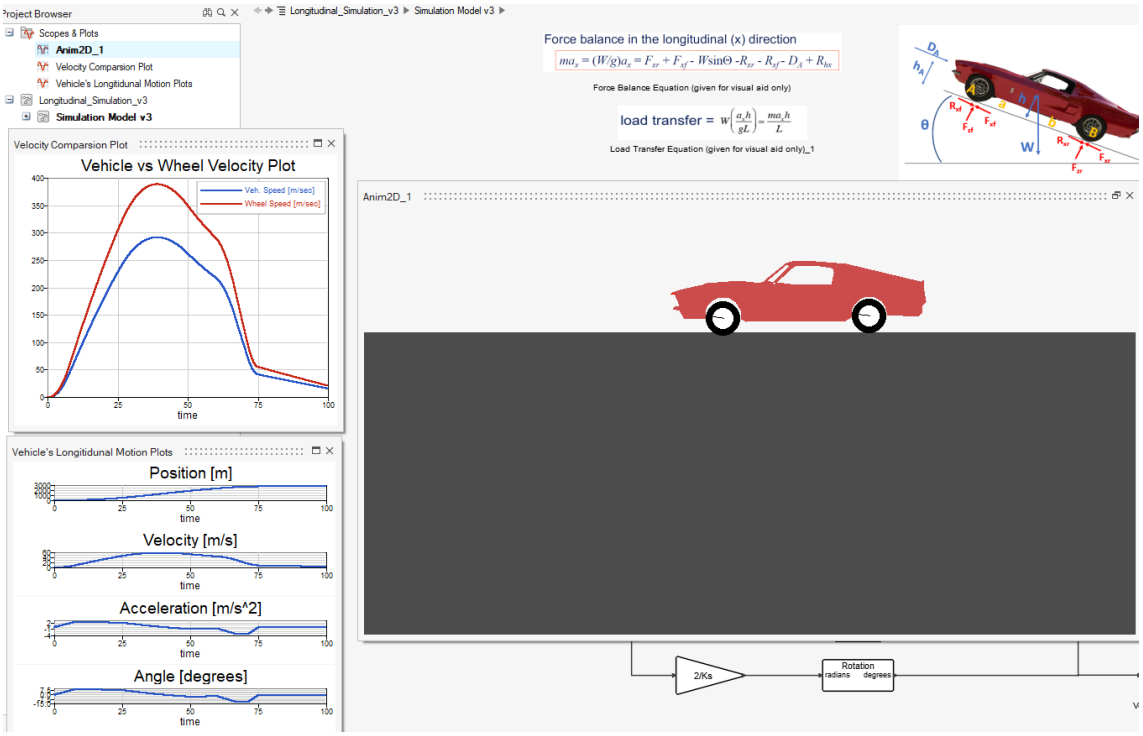


Figure 2.20: *Longitudinal Model (Version#3) - Simulation Output*

2.2.7 Try On Your Own

Please run the simulation file provided in this section. To see different slip results, you can change the Force profile.

Challenge: Obtain 20% slip, which is the best scenario for tire force also used in ABS and traction control studies.

2.2.8 3-D Visualization and Analysis in RTV-Simulator with FMI

Modern computers are quite capable of running rich 3-D animations including games, 3D simulations, etc. A popular way of observing/presenting the outputs of the engineering models -including the vehicle dynamics models of this book- is to use a particular purpose simulator software. It is also possible to show calculations, and results of the model(s) in the simulator. This can be achieved with different methods, but in this section, we would like to present one of the most popular engineering methods called **Functional Mock-up Interface** (FMI or FMU for short) exchange.

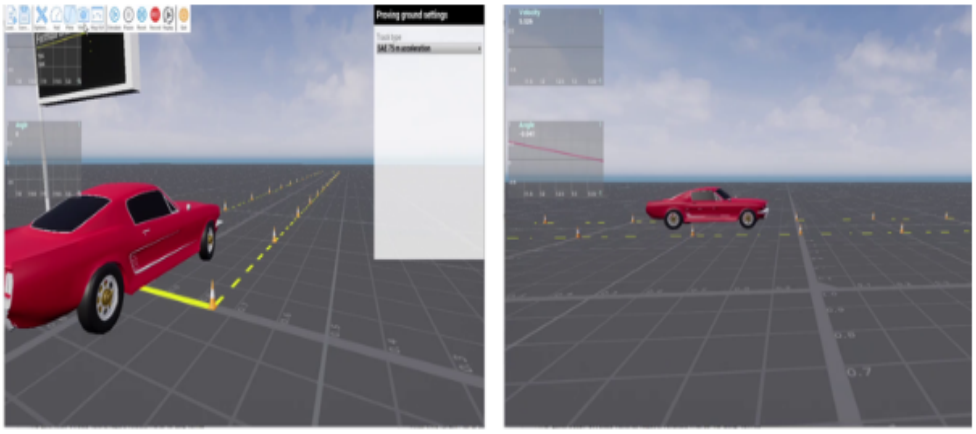


Figure 2.21: *3-D Co-Simulation screenshot - Using the RTV-Simulator. The Angle and Velocity values are plotted thanks to the FMU exported model of Activate.*

Activate has the built-in feature to generate an FMI export file. Any Activate superblock (or model file) could be exported into an FMI exchange file and then this file can be used in any FMI-compliant software such as the RTV-Simulator. RTV-Simulator is a custom

The FMI export operation can be initiated either from the Tools menu or with the right-click context menu as shown in the figure.

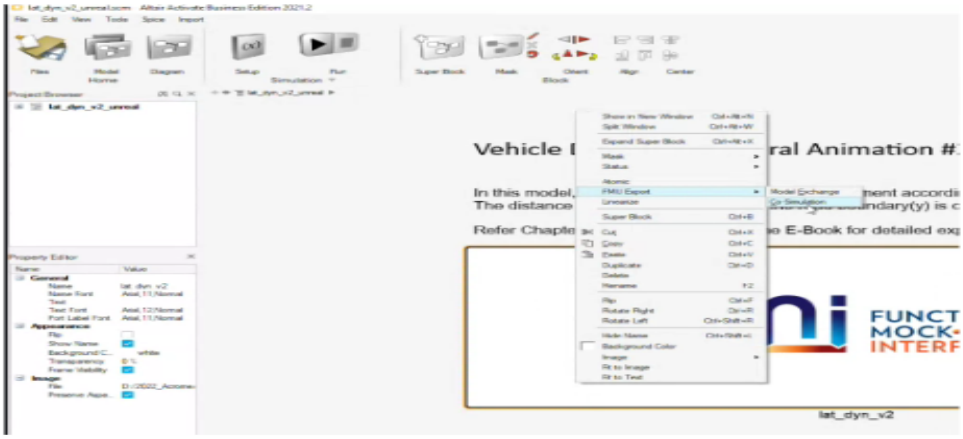


Figure 2.22: *Generating the Functional Mock-Up Interface (FMU) model*

Detailed explanation of the preparation and export process is explained in the Appendix-B section of this e-Book. Please note that for simplicity, we provided a particular version of the Activate model files, which can be directly used to re-export the FMU files. This is necessary as FMU files need input and output ports.

Figure 2.23: *e-Book FMU Folder Contents*

2.3 Lateral Vehicle Motion

2.3.1 Brief Description

The lateral motion is the movement of the vehicle in its lateral plane, i.e. the change of its yaw angle. To understand the lateral dynamics of the vehicle, we will need to find the equation for $\dot{v}$ (lateral acceler-

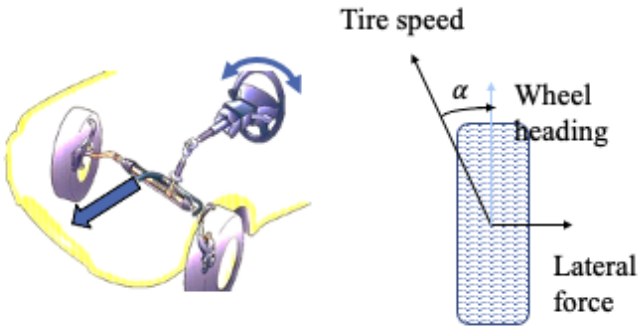


Figure 2.24: *Lateral dynamics is all about the steering system and the tire forces*

When the wheels are rotated (as in the above figure) the lateral force is acting to change the heading angle of the vehicle. We will share more information about these forces and calculations in the following parts.

2.3.2 Descriptive Figure

For the vehicle in a steady-state turn, the vehicle has the following force vectors at each axle.

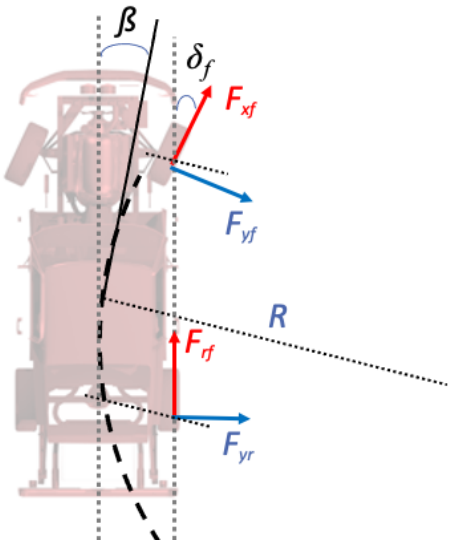


Figure 2.25: *Lateral force vectors while steering*

The turning action happens due to the slip angles (delta Front and delta Rear). In the next section, we will calculate the slip angles and the forces of the above figure. To calculate these variables we will need to know the variables in the table below that take part in our equations.

Abbrev.	**Constant / Force Vector Name**	**Description**
F_{xf} and F_{xr}	Tractive force for front and rear	This is the force generated with the engine or brakes.
F_{yf} and F_{yr}	Small slip forces for front and rear	The difference between the direction a vehicle is traveling (known as heading or course over ground) and the direction that the body of the vehicle is pointing (true heading).
R	Radius of the steady-state turn	The (instantaneous) radius of the circle which the vehicle is rotating around. Equals to u/r.
α_f and α_R	Front axle steering angle and Rear axle tire slip angles	
β	Vehicle's slip	
δ_f and δ_R	Front steering angle and Rear steering angle	
v	Lateral speed	
r	Yaw rate	
L = a + b	The wheelbase	
u	Forward speed	Assumed Constant
Cα	Cornering stiffness	is the ratio on how much lateral force is generated with respect to the wheel's slip angle (alpha).

Table 4: *Forces and descriptions of the force balance equation*

2.3.3 Equations

To calculate the $\dot{v}$ (lateral acceleration) and the $\dot{r}$ (change of the yaw rate), we will have some assumptions to start with.

1. The vehicle is in a steady turn, meaning that the forward speed (u) and the yaw rate (r) are constant throughout the turning action,
2. The tires are linear, applying equal amounts of friction and contact area.

To calculate the final values, we will first need to calculate the slip angles of each tire. For simplicity, we will behave the vehicle as a 2-wheel vehicle (like a bicycle), and we will base our calculations on a single wheel in the front and a single wheel at the back.

The equation for the slip angle is as follows;

$$\delta_f = \frac{L}{R} + \alpha_f - \alpha_r$$

Please note the relationship between the slip angle and the steering angles. There is a relationship between the front and rear steering angles and the slip angle. Assuming that the R is constant (we assume the curvature of the road is not changing like it is a perfectly circular loop), then the steering angle can take 3 different ranges of values around the L/R.

$$\alpha_f > \alpha_r \quad \text{understeer}$$

$$\alpha_f < \alpha_r \quad \text{oversteer}$$

$$\alpha_f = \alpha_r \quad \text{neutral steer}$$

As the slip angle calculation includes the front and rear steering angles, the α_f and α_r. They are also calculated with the following equations:

$$\alpha_f = \delta_f - tan^{-1}\left(\frac{v + ar}{u}\right) \approx \delta_f - \frac{v + ar}{u}$$

$$\alpha_r = \delta_r - tan^{-1}\left(\frac{v - br}{u}\right) \approx \delta_f - \frac{v - br}{u}$$

Next we will define the lateral forces related to the slip angles that we have calculated in the previous step.

Tire lateral forces depend on normal force, are a nonlinear function of tire slip, and depend on tire design (e.g., radial, bias-ply) and inflation pressure. As can be noted in the figure below, for small slip angles (e.g. the red portion in the figure), this force behaves linearly.

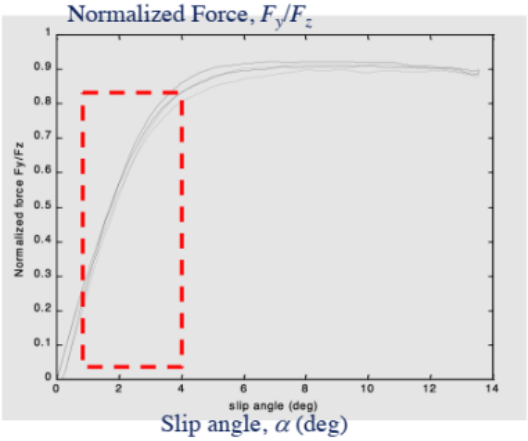


Figure 2.26: *Tire lateral force vs. Slip angle relationship*

So that we can define a linear equation between the slip angle and the lateral forces **for small slip angles** as follows:

$$F_y = C_\alpha \alpha$$

where F_y is the lateral force vector, $C\alpha$ is the understeer coefficient (which will be introduced later in this chapter), and the α is the front or rear slip angles that will be used to calculate the front and the rear lateral forces.

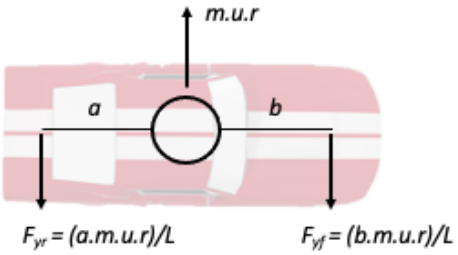


Figure 2.27: *Lateral force vectors during a steady-state turn*

$$mur = F_{yf} + F_{yr}$$

A model describing the lateral handling of the vehicle can be obtained using the force balance on the vehicle models given in Figures 2.25 and 2.27

$$\dot{\mathbf{x}} = \mathbf{Ax}+\mathbf{Bu} \tag{9}$$

$$\mathbf{x}^T = \begin{bmatrix} v & r \end{bmatrix} \tag{10}$$

$$\mathbf{u}^T = \begin{bmatrix} \delta_1 & \delta_2 \end{bmatrix} \tag{11}$$

$$\mathbf{A} = \begin{bmatrix} -\frac{C_{af}+C_{ar}}{mV_0} & \frac{bC_{ar}-aC_{af}}{mV_0} - V_0 \\ \frac{bC_{ar}-aC_{af}}{I_z V_0} & -\frac{a^2C_{af}+b^2C_{ar}}{I_z V_0} \end{bmatrix} \tag{12}$$

$$\mathbf{B} = \begin{bmatrix} \frac{C_{af}}{m} & \frac{C_{ar}}{m} \\ \frac{aC_{af}}{I_z} & -\frac{bC_{ar}}{I_z} \end{bmatrix} \tag{13}$$

Understeering Coefficient

$$\begin{aligned} \delta_f &= \frac{L}{R} + \alpha_f - \alpha_r \\ &= \frac{L}{R} + mur\frac{b}{LC_{\alpha f}} - mur\frac{a}{LC_{\alpha r}} \\ &= \frac{L}{R} + \underbrace{\left(\frac{mb}{LC_{\alpha f}} - \frac{ma}{LC_{\alpha r}}\right)}_{K_{us}\ \text{rad}/(\text{m}/s^2)} \frac{u^2}{R} \end{aligned}$$

Understeering coefficient is a property of the vehicle that can give information related to vehicle lateral stability.

2.3.4 Model Figures

In this sub-chapter, readers are provided with two Activate model examples for simulating the lateral movement of the car, as in this order:

- Lateral Simulation Version #1 - Constant Road - Bicycle Steering Model
- Lateral Simulation Version #2 - Road Following Model with Road Curvature

Although we have 2 different models in this chapter, they are very similar to each other. The main difference between the models is the change in the road curvature and how the vehicle measurements and animation are represented.

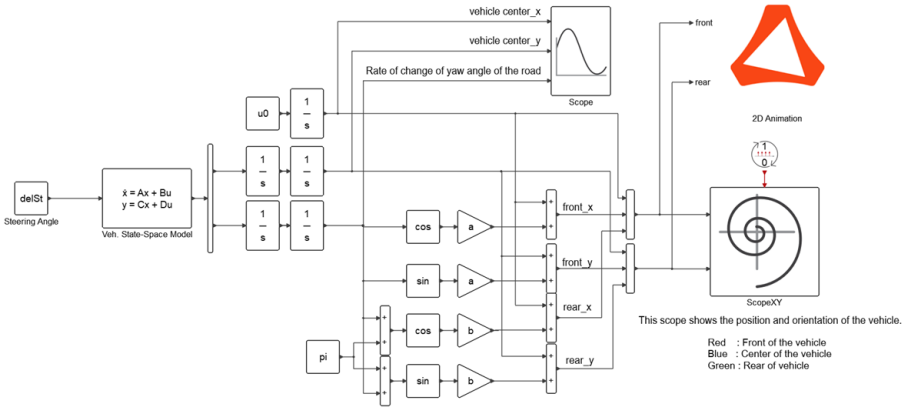


Figure 2.28: *Interior of the Lateral Simulation v1 model.*

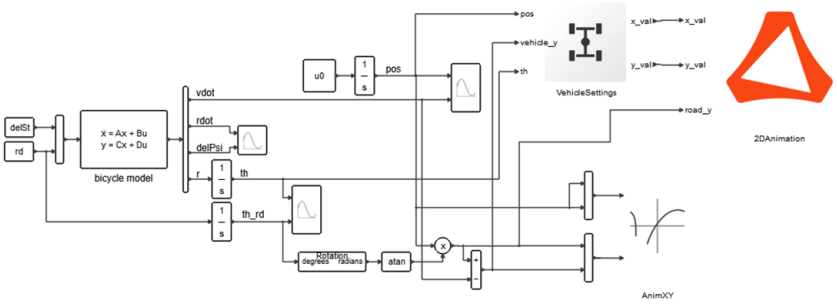


Figure 2.29: *Lateral Simulation v2 model. This model is similar to the v1 version, has an additional rd input, and is the change of the road curvature.*

Details of each simulation model are given in the next subsection.

2.3.5 Model Overview

The v1 model is created for analyzing the lateral position of the vehicle in a constant curved (like a circular) road. Here we assume that the driver is applying a constant steering angle to the wheels. In this

version, we will see what will happen if the steering angle is kept constant throughout the curvature of the road.

Our v2 model is the final version where we will also change the curvature of the road, as it is how it happens in general. We will also measure the distance between the center line of the curvature and the vehicle and show how well the vehicle tracks the center line.

Longitudinal Simulation Version #1

Action Item: Please locate and open the simulation file named Lateral Simulation v1.scm in the Altair Activate software.

The Lateral Simulation Version #1, is the first simulation model that is explained. For documentation purposes, a screenshot of the model is given below, and the main portions of the image are segmented with color codes and numbers.

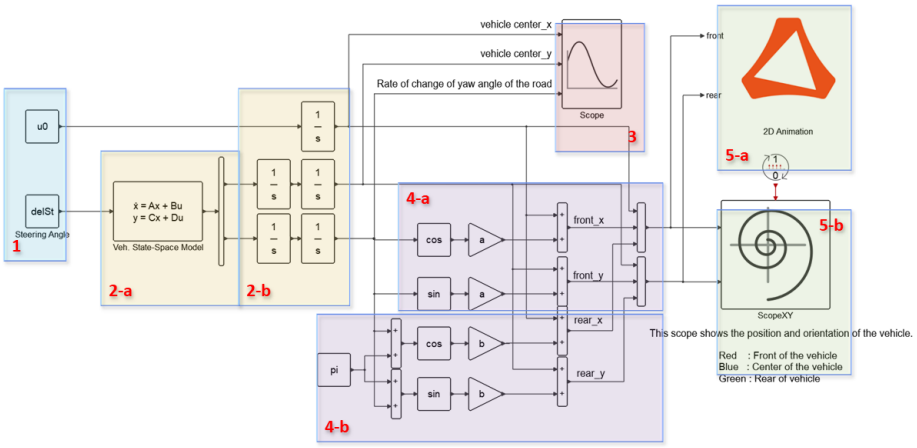


Figure 2.30: *Lateral Simulation Model (Version #1). This model is based on 8 different segments. Each part is separated with a different color code.*

The Segment **#1** (blue box) represents the input values ($u0$ as the initial speed and $delSt$ as the steering angle) applied to the equation. They can be changed with the model initialization page as well. As an exercise, students can try changing the turning radius, steering angle, and also the forward speed and watch how well the vehicle is following the constant turn.

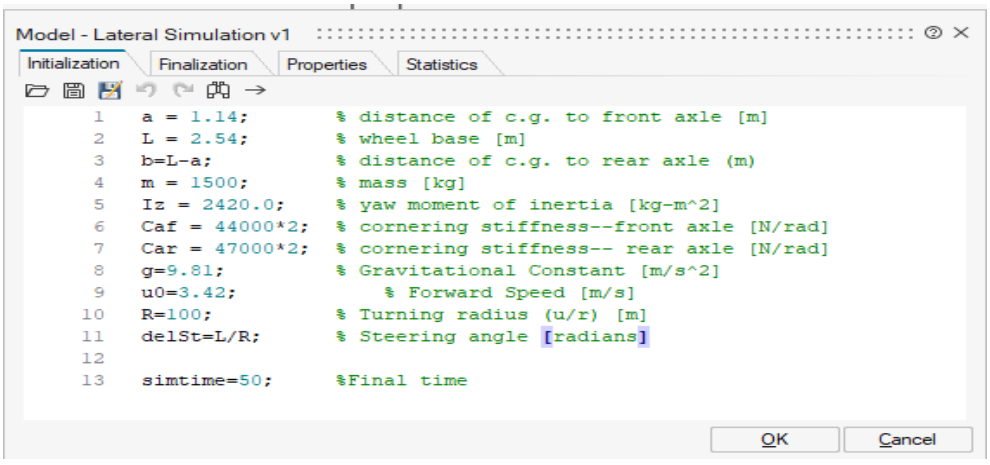


Figure 2.31: *Initial parameter values for the model. Parameters can be quickly changed using Activate's model initialization window.*

The Segment **#2-a, b** (orange boxes) is the place where the vehicle equation is calculated. In 2-b, we take derivatives to calculate the lateral position of the vehicle and its turning angle (i.e. yaw angle). In fact, this is the exact output of our Simulation model, and the rest of the blocks are used for visualization only.

The Segment **#3** (red box) is used to plot the outputs of the Segment #2. We have provided the necessary inputs to the Scope object so that all 3 variables can be seen on the same Scope screen.

The Segment **#4-a,-b** (purple boxes) is used to calculate the interim values that are used in the 2-D animation of Section #5. We need these values to draw the movement of our vehicle on the 2-D picture box.

The Segment **#5-a,-b** (green boxes) are used to visualize the output of the calculations. We have a Scope for X-Y plots and also a 2-D picture box to visualize the motion of the vehicle from an orthogonal field-of-view.

Longitudinal Simulation Version #2

Next version of our model calculates the overall Axle load, which covers the effect of the axial loads in the longitudinal calculations. This would help us to see the rolling movement of the vehicle.

Action Item: Please locate and open the simulation file named Lateral Simulation v2.scm in the Altair Activate software.

The Lateral Simulation Version #2, is the second and final simulation model that is explained. For documentation purposes, a screenshot of the model is given below, and the main portions of the image are segmented with color codes and numbers.

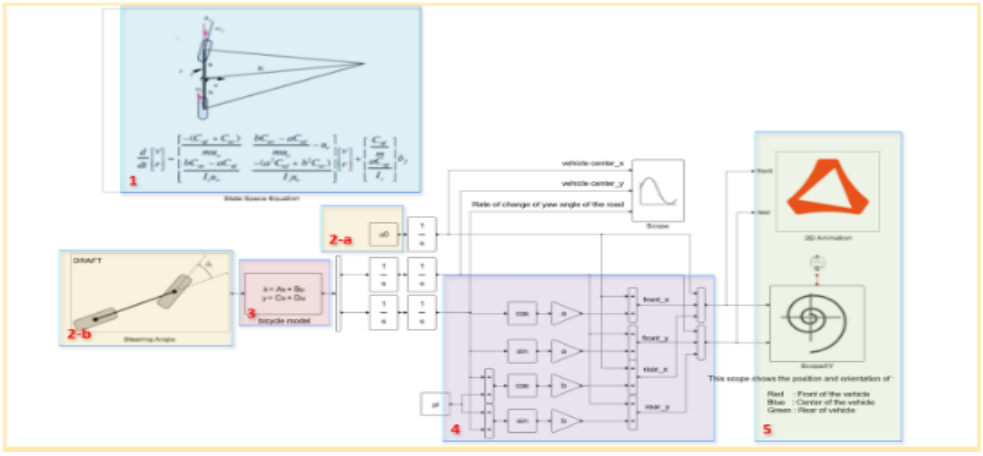


Figure 2.32: *Lateral Simulation Model (Version #2). This model is based on 5 different segments. Each part is separated with a different color code.*

2.3.6 Results

2.3.7 Try On Your Own

Please run the simulation file provided in this section. Change the speed, turning radius or stiffness of the vehicle and observe how well the vehicle can track the line.

Try to find optimum values for the best trajectory.

Try to observe an understeering condition.

2.3.8 3-D Visualization and Analysis in RTV-Simulator with FMI

2.4 Vertical Vehicle Motion

2.4.1 Brief Description

The vertical motion is the movement of the vehicle in its vertical plane, ie. the change of its yaw angle. To understand the vertical dynamics of the vehicle, we will need to find the equation for $\boldsymbol{\theta}$ (angular displacement) or the $\boldsymbol{Z}$ (elevation of the center of mass). The main step for this is to link input (ie. road excitation) to the vehicle dynamics and output it as the human ride perception.

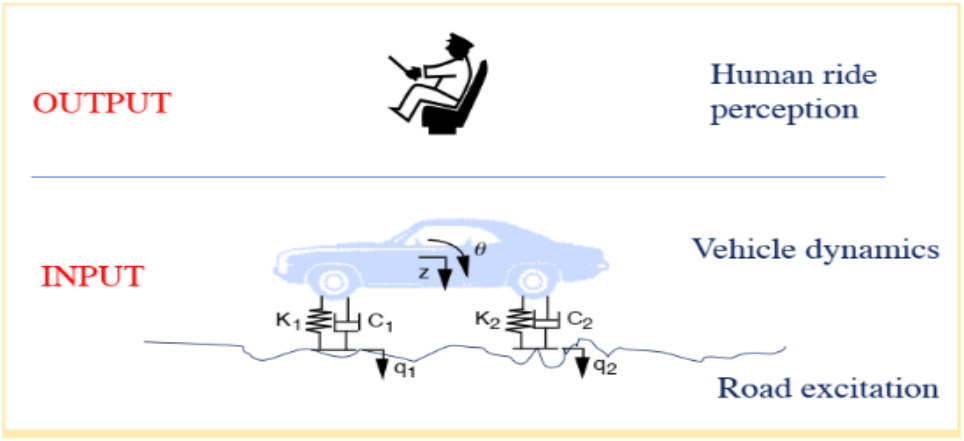


Figure 2.33: *Vertical dynamics affects the relationship between the road surface and the perception of the driver/passengers*

Ideally, the vehicle-ride dynamics consider the pitch and roll motions of the vehicle as well as the vertical (i.e., heave) motion. Various models have been developed to study passenger car ride quality. For a full-car model, we usually have 7-DOF. It is shown that the coupling between the pitch-and-roll and the heave motions is not significant for typical passenger vehicles. A two-DOF quarter-car model considers both the vehicle sprung mass and the unsprung mass associated with the wheel/tire/axle assembly and represents a good compromise between model simplicity and accuracy. It is adequate to consider a so-called quarter-car model to start with, and it is as illustrated in this sub-section.

2.4.2 Descriptive Figure

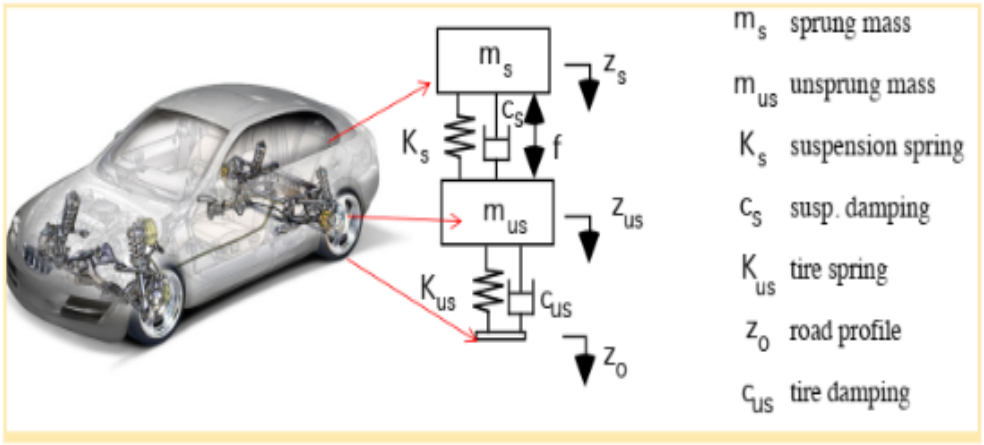


Figure 2.34: *Vertical dynamics equivalent diagram for one wheel*

Quarter-car models only represent the motions of a corner of the car and could be 1-DOF (sprung mass only) or 2-DOF (sprung and unsprung mass vertical motions). As can be seen in the descriptive figure,

the wheel assembly is modeled with three heights, split into two mass and two spring-damper sections. The constant and mass names and their detailed descriptions are given in the below table:

Abbrev.	Constant / Mass Names	Description
m_s	Sprung mass	
m_{us}	unsprung mass	
K_s	suspension spring constant	Passive part of the suspension
c_s	Suspension damping constant	Active part of the suspension
K_{us}	tire spring constant	
c_{us}	Tire damping constant	
Z_0	Vertical Road profile	Input to the model
Z_s	Z of the Sprung Mass	Elevation of the Sprung mass
Z_{us}	Z of the Unsprung Mass	Elevation of the Unsprung mass

Table 5: *Constants and Masses for the 2-DOF Quarter Car Vertical Model*

2.4.3 Equations

In the model, an active suspension is assumed to be in parallel with a passive suspension, k_s and c_s.The tire stiffness, k_{us}, and damping, c_{us}, also are modeled. Note that the tire damping, c_{us}, can sometimes be neglected. The dynamic equations that govern the motions of these two masses are as follows:

$$m_s\ddot{z}_s + c_s(\dot{z}_s - \dot{z}_{us}) + k_s(z_s - z_{us}) = -f$$

$$m_{us}\ddot{z}_{us} + c_s(\dot{z}_{us} - \dot{z}_s) + k_s(z_{us} - z_s) + c_{us}(\dot{z}_{us} - \dot{z}_0) + k_{us}(z_{us} - z_0) = f$$

As in the lateral equations and simulation, it is easier to represent the above dynamic equations with state-space matrix representation. The equivalent state-space form of the above equations is as follows:

$$\frac{d}{dt}\begin{bmatrix} z_{us}-z_0 \\ \dot{z}_{us} \\ z_s - z_{us} \\ \dot{z}_s \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_{us}}{m_{us}} & -\frac{(c_s+c_{us})}{m_{us}} & \frac{k_s}{m_{us}} & \frac{c_s}{m_{us}} \\ 0 & -1 & 0 & 1 \\ 0 & \frac{c_s}{m_s} & -\frac{k_s}{m_s} & -\frac{c_s}{m_s} \end{bmatrix} \begin{bmatrix} z_{us}-z_0 \\ \dot{z}_{us} \\ z_s - z_{us} \\ \dot{z}_s \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{m_s}{m_{us}} \\ 0 \\ -1 \end{bmatrix} \frac{f}{m_s} + \begin{bmatrix} -1 \\ \frac{c_{us}}{m_{us}} \\ 0 \\ 0 \end{bmatrix} \dot{z}_0$$

This state-space matrix could be simplified as a three-element normalized matrix form of ;

$$\dot{x} = Ax + Bu + Gw$$

where $x =$

$$\begin{bmatrix} z_{us}-z_0 \\ \dot{z}_{us} \\ z_s - z_{us} \\ \dot{z}_s \end{bmatrix}$$

and each A, B and G elements are combined as follows;

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -w_1^2 & -2(\rho\zeta_2 w_2 + \zeta_1 w_1) & \rho w_2^2 & 2\rho\zeta_2 w_2 \\ 0 & -1 & 0 & 1 \\ 0 & 2\zeta_2 w_2 & -w_2^2 & -2\zeta_2 w_2 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ \rho \\ 0 \\ -1 \end{bmatrix},\ G = \begin{bmatrix} -1 \\ 2\zeta_1 w_1 \\ 0 \\ 0 \end{bmatrix}$$

where;

$$\rho = \frac{m_s}{m_{us}},\ \ w_1 = \sqrt{\frac{k_{us}}{m_{us}}},\ \ \zeta_1 = \frac{c_{us}}{2m_{us}w_1},\ \ w_2 = \sqrt{\frac{k_s}{m_s}},\ \ \zeta_2 = \frac{c_s}{2m_s w_2}$$

2.4.4 Model Figures

In this subchapter, readers are provided with two Activate model examples for simulating the quarter car motions and then for calculating the vertical movement of the full car, as in this order:

- Vertical Simulation Version #1 - Quarter Car with Simulation in Activate
- Lateral Simulation Version #2 - Full Car Calculations with Simulation in MotionVIEW

Although we have 2 different simulations, their calculations are common. The main difference between the models is the number of wheels used for the calculations and how the simulations run.

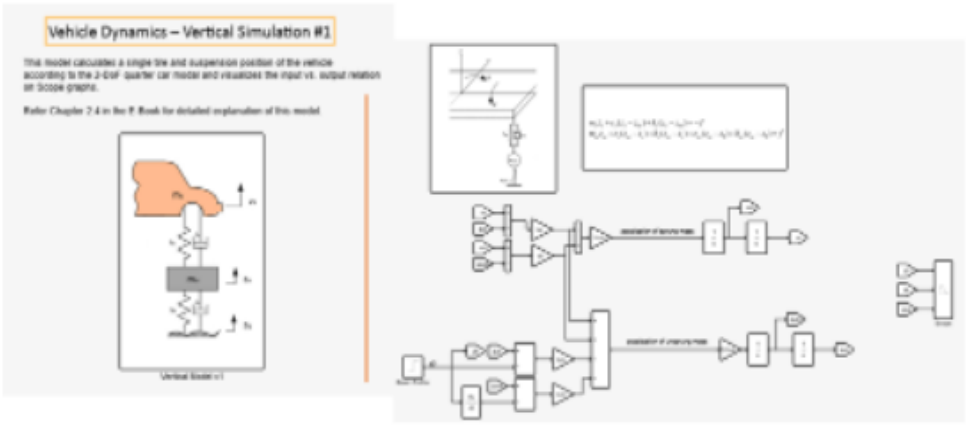


Figure 2.35: *Vertical Model v1. On the left top-level Superblock, on the right, the interior of the top-level can be seen. Notice the output block is a Scope Graph used to analyze the output of the Simulation.*

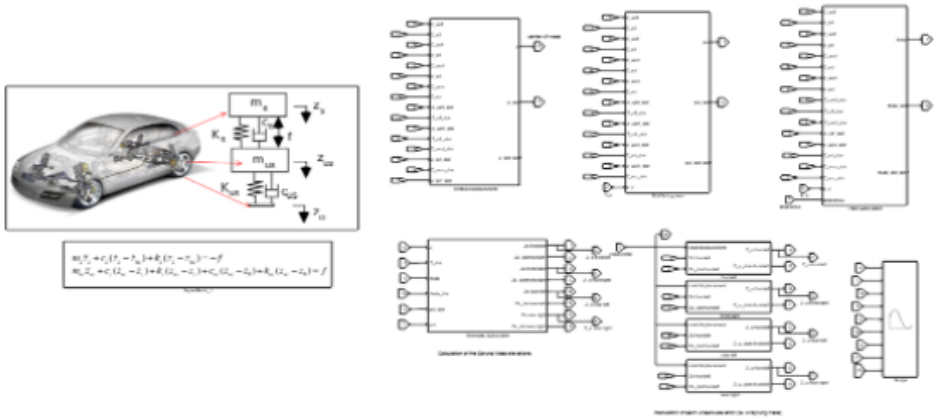


Figure 2.36: *Vertical Simulation v2 model. Note that this model has only input and output ports so there isn't any simulation. This model is used in the Full Simulation Model.*

Details of each simulation model are given in the next subsection.

2.4.5 Model Overview

In this section we will be explaining the details of each model. We will first provide the details of the v1 Model, which provides the basis of the vertical vehicle motion. Then we will provide details about the v3 model, which will be re-used in the Full Vehicle Dynamics section as well. Please note that MotionVIEW software is used for simulations, as it is another powerful 3D simulator with a built-in Multi-body dynamics solver. Readers are encouraged to pre-reading the "Appendix D - MotionVIEW Quick Start" section for

a quick introduction to this tool.

2.4.5.1. Vertical Simulation Version #1

Use Cases of this Model: The vertical vehicle models help to calculate the suspension system forces vs. vehicle body movement relationship of the vehicle. The v1 version is an isolated system model (i.e., Quarter of the Car), and it can also be used to solve 2-DoF mass-spring-damper system problems such as the one in this video.

Action Item: Please locate and open the simulation file named Vertical Simulation v1.scm in the Altair Activate software.

The Vertical Simulation Version #1 is the first simulation model that is explained. For documentation purposes, a screenshot of the model is given below, and the main portions of the image are segmented with color codes and numbers.

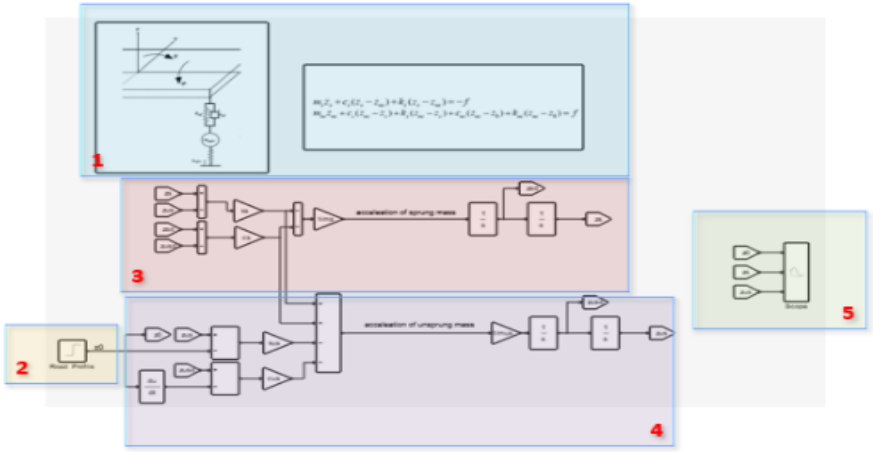


Figure 2.37: *Vertical Model v1. This model is based on 5 different segments. Each part is separated with a different color code.*

The Segment **#1** (blue box) of the model is visual only and describes the variables and calculations of the model. The mathematical calculations are implemented with the Activate blocks and please note how the order is followed.

The Segment **#2** (orange box) represents the road height profile. This is the input of our system. Please note that in the model we use a simple Step Function as the input and it is time-bounded. Users are encouraged to change the shape and timing characteristics of this input signal. Model output (i.e., the Z_{us} and Z_s elevations) are directly affected by the input signal/road profile.

The Segment **#3** (red box) calculates the height profile output of the Sprung mass (Zs). Notice that Z_s is calculated iteratively using its previous value and the previous elevation data of the unsprung mass (Z_{us})and also their derivatives.

The Segment **#4** (purple box) calculates the height profile output of the Unsprung mass (Z_{us}). Similar to the Z_s calculations, the Z_{us} is calculated using its previous value, and the previous elevation data of the sprung mass (Zs)and their derivatives plus uses the road profile as an input. Please remember that the unsprung mass has direct contact with the road surface; hence the input is only applied to this calculation.

The Segment **#5** (green box) is a simple scope output used for visualizing the input (road profile) and output signals (elevation info i.e. Z_s, Z_{us}).

2.4.5.2. Vertical Simulation Version #2

The Vertical Simulation Version #2 is the final version of the model, which is used in the Full vehicle model of the next subsection. As for documentation purposes, a screenshot of the model is given below, and the main portions of the image are segmented with color codes and numbers.

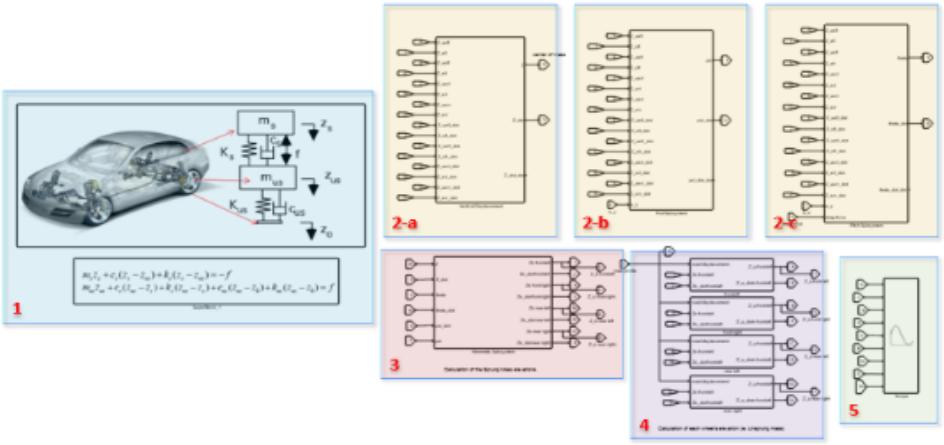


Figure 2.38: *Vertical Simulation Model (Version #2). This model is based on 5 different segments. Each part is separated with a different color code.*

The Segment **#1** (blue box) of the model is again for visual only. We need to repeat our calculations for all the wheels plus we also need to apply the kinematics of the system as wheels are bounded to each other and effects their calculations.

The Segment **#2-a, 2-b, and 2-c** (orange boxes) are the interim elevation calculations involving the instantaneous data of each wheel. We use the output of these blocks to calculate our final elevation matrix.

The Segment **#3** (red box) calculates the kinematics of each wheel and uses the data of other wheels and the output of the interim elevations.

The Segment **#4** (purple box) calculates the height profile of each wheel (i.e., Unsprung mass) while neglecting the effects of the other wheels. These values are then used in the interim elevation calculations of the Segment **#2**.

The Segment **#5** (green box) is a simple scope output used for visualizing the input (road profile) and output signals (elevation info i.e. Z_s, Z_{us}).

2.4.6 Results

In this section, we will be providing information about the Vertical Simulation v1 model. After we run the model, Activate will generate a simple scope output as below.

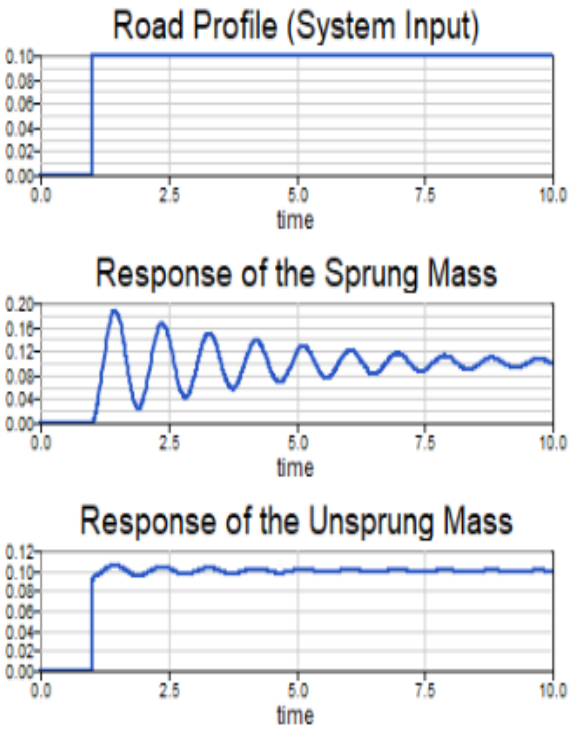


Figure 2.39: *Vertical Simulation Model (Version #1) results. Please note that Y-axis values are height values in meters (s).*

In our example we are applying a constant elevation change to the wheel. It can be considered like moving towards a higher road patch. As soon as the tire hits the patch, the vehicle coordinates also change concerning the change of elevation. It is important to note that the sprung mass (i.e., the elevation of the driver's seat in the vehicle) has a smoother response compared to the elevation of the wheels, which is (for obvious ride comfort reasons) the expected behavior.

2.4.7 3-D Visualization and Analysis in Simulator

Chapter#4 includes a full 3-D simulation with a focus on vertical vehicle dynamics. Readers are encouraged to check this section for visualization and analysis as well.

2.5 Combined Vehicle Dynamics

2.5.1 Brief Description

In this part, we will combine all 3 previous sub-models and combine the full vehicle model. We will also provide an outline of the equations and correlate the full model equation with the simulation model.

2.5.2 Descriptive Figure

Full car model represents all the forces and related motions of the vehicle. The components of the longitudinal, lateral and vertical vehicle dynamics are shown in the following 2 images. The first image represents vertical acting forces and coordinates, and the second shows lateral forces and coordinates.

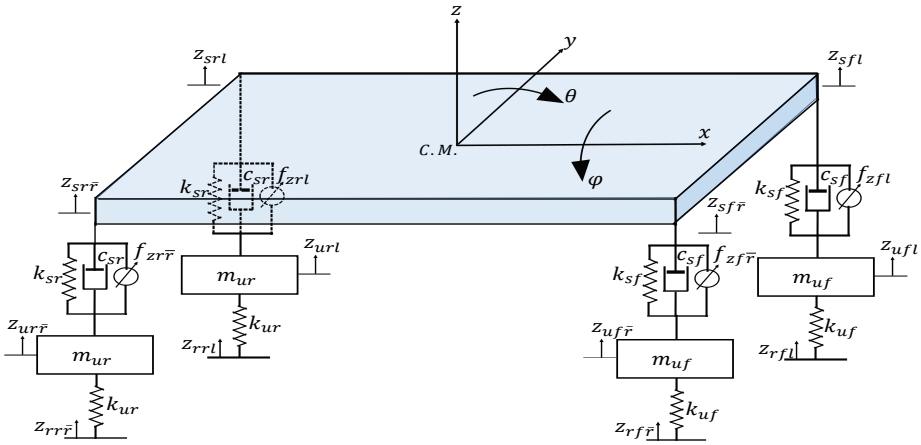


Figure 2.40: *Full Simulation Model Figure #1 contains all the angular and vertical movement around the center of the mass and is affected by the spring-suspension system of each wheel.*

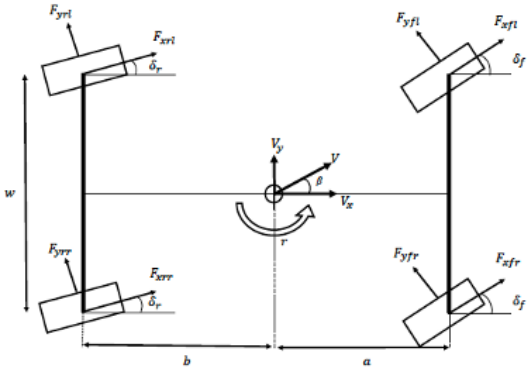


Figure 2.41: *Full Simulation Model Figure #2 contains all the forces and coordinates for lateral movement.*

The vehicle has 2 sets of actuators, which are two-wheel steering and two-wheel traction. The vehicle is also assumed to have a passive suspension (Fzf* = 0) system. The constants, masses, and coordinate names and their detailed descriptions are given in the below table:

Abbrev.	Constant / Mass Names	Description
m_s	Sprung mass	
m_{us}	unsprung mass	
K_s	suspension spring constant	Passive part of the suspension
c_s	Suspension damping constant	Active part of the suspension
K_{us}	tire spring constant	
c_{us}	Tire damping constant	
Z_0	Vertical Road profile	Input to the model
Z_s	Z of the Sprung Mass	Elevation of the Sprung mass
Z_{us}	Z of the Unsprung Mass	Elevation of the Unsprung mass

Table 6: *Coordinates, Constants, and Masses of the Full Car Model*

2.5.3 Equations

In this section, we will be providing the equations of the full vehicle model. These equations are based on the previously given multi-coordinate equations; however, due to coupling effects the full vehicle equations naturally have some input-output correlation as well. The coordinate systems correlations

will be expressed in the Model Overview section. Further analytical derivations of the full model equations can be read from the reference. [5]

$$m\ddot{z} = (2k_{sf} + 2k_{sr})z - (2c_{sf} + 2c_{sr})\dot{z}$$
$$+(2ak_{sf} - 2bk_{sr})\theta + (2ac_{sf} - 2bc_{sr})\dot{\theta}$$
$$+k_{sf}z_{ufl} + c_{sf}\dot{z}_{ufl} + k_{sf}z_{ufr} + c_{sf}\dot{z}_{ufr}$$
$$+k_{sr}z_{url} + c_{sr}\dot{z}_{url} + k_{sr}z_{urr} + c_{sr}\dot{z}_{urr}$$

where k_{sf}, c_{sf} are front sprung mass spring and damping coefficients, k_{sr}, c_{sr} are rear sprung mass spring and damping coefficients, z_u is unsprung mass displacement and f_i is active suspension forces at i^{th} wheel of the vehicle where i^{th} [1;4].
The pitch ($\ddot{\theta}$) rotational acceleration can be calculated as:

$$I_y\ddot{\theta} = (2ak_{sf} - 2bk_{sr})z + (2ac_{sf} - 2bc_{sr})\dot{z}$$
$$-(2a^2k_{sf} + 2b^2k_{sr})\theta - (2a^2c_{sf} + 2b^2c_{sr})\dot{\theta}$$
$$-ak_{sf}z_{ufl} - ac_{sf}\dot{z}_{ufl} - ak_{sf}z_{ufr} - ac_{sf}\dot{z}_{ufr}$$
$$+bk_{sr}z_{url} + bc_{sr}\dot{z}_{url} + bk_{sr}z_{urr} + bc_{sr}\dot{z}_{urr} - ma_xh$$

The roll ($\ddot{\Phi}$) rotational acceleration can be calculated as:

$$I_x\ddot{\phi} = -0.25w^2(2k_{sf} + 2k_{sr})\phi + 0.5wk_{sf}z_{ufl}$$
$$-0.25w^2(2c_{sf} + 2c_{sr})\dot{\phi} + 0.5wc_{sf}z_{ufl}$$
$$-0.5wk_{sf}z_{ufr} - 0.5wc_{sf}\dot{z}_{ufr} + 0.5wk_{sr}z_{url}$$
$$+0.5wc_{sr}\dot{z}_{url} - 0.5wk_{sr}z_{urr} - 0.5wc_{sr}\dot{z}_{urr} - ma_yh$$

Finally, elevation of unsprung mass z_{uij} where $i \in \{f, r\}$ and $j \in \{l, r\}$ can be calculated as:

$$m_u\ddot{z}_{ufl} = k_{sf}z + c_{sf}\dot{z} - ak_{sf}\theta - ac_{sf}\dot{\theta} + 0.5wk_{sf}\phi + 0.5wc_{sf}\dot{\phi} - (k_{sf} + k_{uf})z_{ufl} - c_{sf}\dot{z}_{ufl} + k_{uf}z_{rfl}$$

$$m_u\ddot{z}_{ufr} = k_{sf}z + c_{sf}\dot{z} - ak_{sf}\theta - ac_{sf}\dot{\theta} - 0.5wk_{sf}\phi - 0.5wc_{sf}\dot{\phi} - (k_{sf} + k_{uf})z_{ufr} - c_{sf}\dot{z}_{ufr} + k_{uf}z_{rfr}$$

$$m_u\ddot{z}_{url} = k_{sr}z + c_{sr}\dot{z} + bk_{sr}\theta + bc_{sr}\dot{\theta} + 0.5wk_{sr}\phi + 0.5wc_{sr}\dot{\phi} - (k_{sr} + k_{ur})z_{url} - c_{sr}\dot{z}_{url} + k_{ur}z_{rrl}$$

$$m_u\ddot{z}_{url} = k_{sr}z + c_{sr}\dot{z} + bk_{sr}\theta + bc_{sr}\dot{\theta} - 0.5wk_{sr}\phi - 0.5wc_{sr}\dot{\phi} - (k_{sr} + k_{ur})z_{urr} - c_{sr}\dot{z}_{urr} + k_{ur}z_{rrr}$$

2.5.4 Model Figures

In this final sub-chapter, we will be assembling our full vehicle dynamics model by combining the models from the previous sections. We will re-name these models and call them sub-models hereafter:

- Longitudinal Simulation Version #3 model → Longitudinal submodel
- Lateral Simulation Version #2 model → Lateral submodel
- Vertical Simulation Version #2 model → Vertical submodel

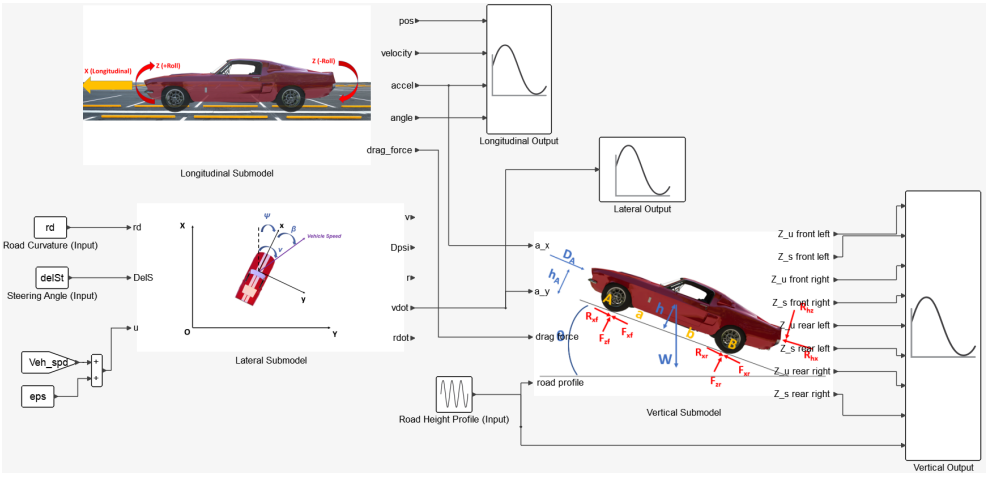


Figure 2.42: *Full Vehicle Simulation Model. This model is composed of previous chapter models.*

To make a combined simulation, we will need to share some input and output values between the submodels. The necessary inter-model signals will be explained in the Model Overview section.

2.5.5 Model Overview

Action Item: Please locate and open the simulation file named Full Vehicle Simulation.scm in the Altair Activate software.
As described, our model is based on 3 sub-models that have already been explained in the previous sub-sections. Combining these models would require combining some of the vehicle's coordinates and states as well. This time we have depicted the signal connections of each sub-model.

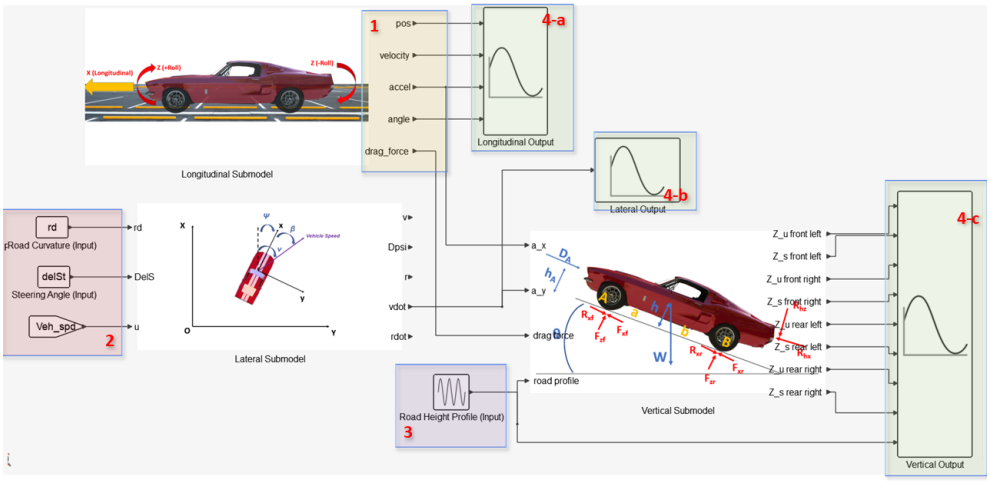


Figure 2.43: *Signal connections of submodels in the Full Vehicle Model.*

The Segment #1 output of the longitudinal sub-model is actually the main source for the signals used by other sub-models. We use the position, velocity, and drag force outputs of this model in the subsequent models. Please note that the same force input pattern as in Figure 2-13 is applied to the longitudinal submodel.

The Segment #2-a represents the 2 user selectable input values (rd as the curvature of the road and $delSt$ as the steering angle). This way a lateral road profile can be applied and the vehicle's motion outcome can be observed. The Segment #2-b is used to apply the forward speed for calculations of lateral movement.

2.5.6 Simulation Results

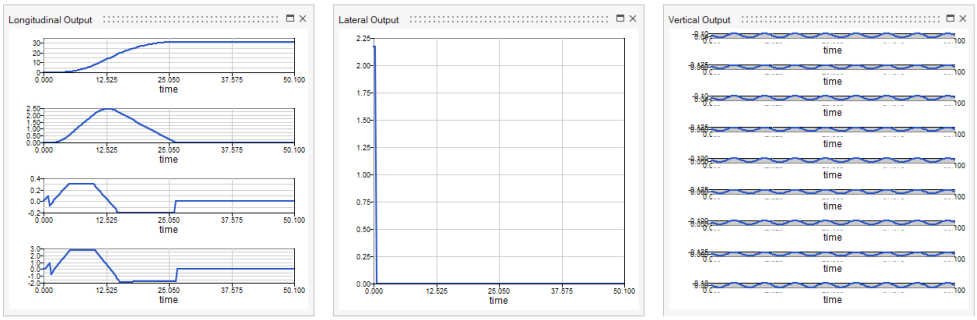


Figure 2.44: *Simulation Results for the Full Vehicle Dynamics with unit inputs. Inputs can be modified.*

2.6 References

[1] SAE Recommended Practice J670e, Vehicle Dynamics Terminology (first issued 1952, last updated 1976).

[2] ISO 8855, Road vehicles — Vehicle dynamics and road-holding ability - Vocabulary (1991).

[3] "Tires and Passenger Vehicle Fuel Economy: Informing Consumers, Improving Performance – Special Report 286. National Academy of Sciences, Transportation Research Board, 2006" (PDF). Retrieved 2007-08-11.

[4] Altair University, available on https://altairuniversity.com/ - Last visited in December 2021.

[5] Ozan Temiz, Melih Cakmakci, Yildiray Yildiz, Adaptive Control Allocation with Communication Delay for In-Wheel Propulsion Electric Vehicles, IFAC-PapersOnLine, Volume 52, Issue 20, 2019, Pages 157-162, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2019.12.151..

[6] A. Galip Ulsoy, Huei Peng, and Melih Cakmakci. *Automotive Control Systems*. Vol. 9781107010. Cambridge: Cambridge University Press, 2012, p. 406. ISBN: 9780511844577. DOI: 10.1017/CBO9780511844577. arXiv: arXiv:1011.1669v3. URL: https://www.cambridge.org/core/books/automotive-control-systems/644292EC087F83D87025E8CF7488F9E7.

SECTION 3

Electric Powertrain Overview

3.1 Introduction

This section will give brief background information for the three major powertrain parts of the electrical vehicle. These are:

1. Electric Motor (i.e., Motor/Generator Unit)
2. Battery System including the Battery and Inverter
3. Powertrain elements, which are simplified as the reduction gear and the drive wheel.

Related parts are depicted in the following figure. Please note that the image is a sample representation and it doesn't reflect any real-world vehicle. The exact location and position of the components may change from the manufacturers.

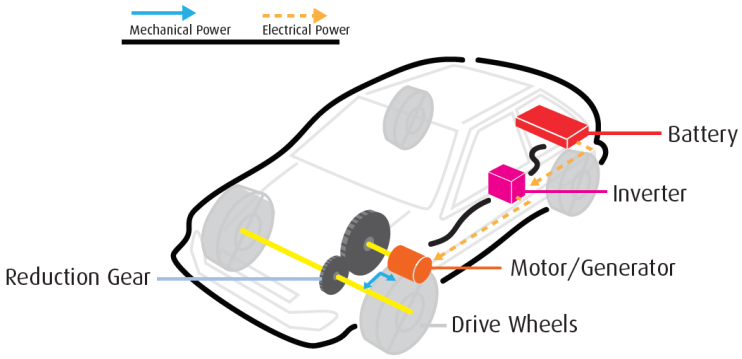


Figure 3.1: *Simulated Components of the EV*

Detailed information and individual simulation models for each major part is given in the following subsections. By the end of this chapter, there will be a final vehicle model involving all the parts and integrating the vehicle dynamics as well. The last exercise will fully simulate the EV in Activate and calculate

its mileage in the simulation software.

We will start with the electric motor first. Then we will add the battery system and finally the main powertrain elements of the EV.

3.2 Electric Motors

Based on the introduction section and the scope, here we will give a simple simulation model of a common EV Electric Motor and analyze its inputs and outputs.

The Electric Motor (EM for short) is the power unit of the EVs and is also used in hybrid vehicles as well. The EM generates electro-motor force (EMF) for accelerating the vehicle and also generates back-electro-motor force (ie. back-EMF) to decelerate. The back-EMF generated energy can also be stored back into the battery but it is not mandatory. So, its usage is two-fold, and it's one of the most important items of EVs.

In this sub-chapter, we will provide general information about the EMs that are used in the EVs and will also provide 3 different simulation models (in Activate) with different fidelities. We will provide the simulation results of these different fidelity models as well. Please note that this subchapter is not intended to provide thorough information about the EMs in general.

3.2.1 Brief Description

Depending on the choice of the manufacturers, there can be 1 or more EMs in an EV. The location of the EMs can also vary, however as a common practice and ease of forced cooling; the front-located EMs are common in the industry hence it is depicted this way in our simulations.

In the EV industry there are 2 dominant EM types that are used nowadays. These are:

- Permanent Magnet Synchronous Motor (PMSM)
- Three Phase Induction Motor

The main difference between the two motor types is how they generate the electromagnetic field and how their rotor is built. The PMSM is a type of brushless DC motor, and it has special magnets inside its rotor. The induction motor is an AC motor, and it doesn't have magnets inside its rotor. Manufacturers prefer different types of motors according to their design specs and also the cost vs. performance ratio.

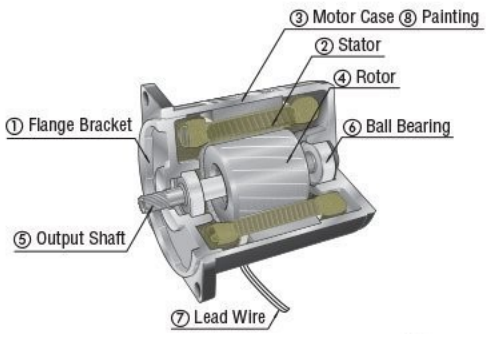


Figure 3.2: *A typical Electric Motor's Interior Structure*

A typical EM is based on 7-8 major components. From inside to out they are Rotor and output shaft, bearings, stator-stator windings, and their lead wire, motor case and flanges, etc. for mounting. External electronic circuitry is also required to drive and control the EM.

The EM works with the principle of electromagnetism. A rotating electromagnetic field (controlled by the external circuit) generates the EMF, and it rotates the rotor (i.e., the shaft) accordingly. EM design is a

multi-disciplinary topic including electromagnetic design and analysis. Tools like Altair FluxMotor^{TM} [1] are used to design all the aspects including the mechanics, electromagnetic, thermal designs, and various analyses.

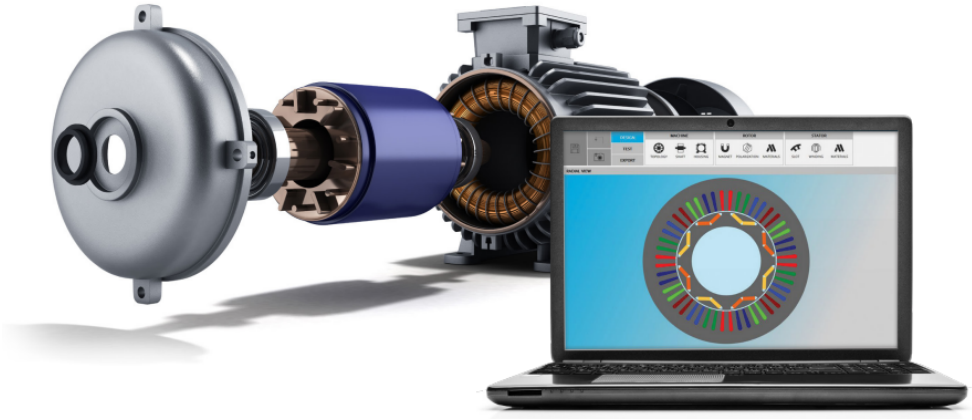


Figure 3.3: *FluxMotor is an example EM simulation Software - Image in courtesy of Altair Inc.*

3.2.2 Model Figure

Similar to Chapter 2 models, we have individual models for each component of the EV. Here we will share the outline images first, and then the details will be given. Note that the models are prepared to be used separately so that each model has some default (required) input signals and the model outputs are also connected to the scope graphs for visualization. Information about the input and output signals is also given.

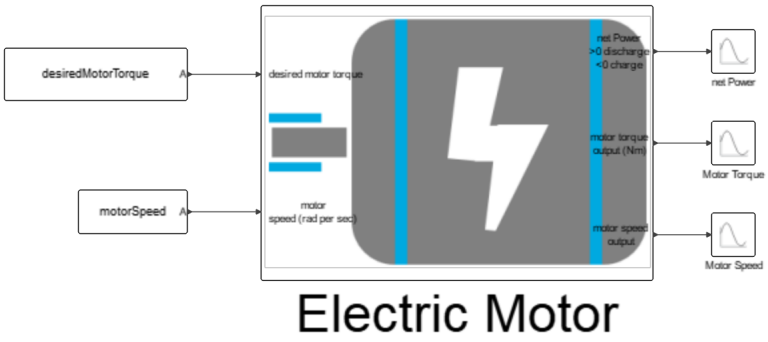


Figure 3.4: *EM Model Superblock Figure and Its Inputs & Outputs*

And here is the interior of the superblock:

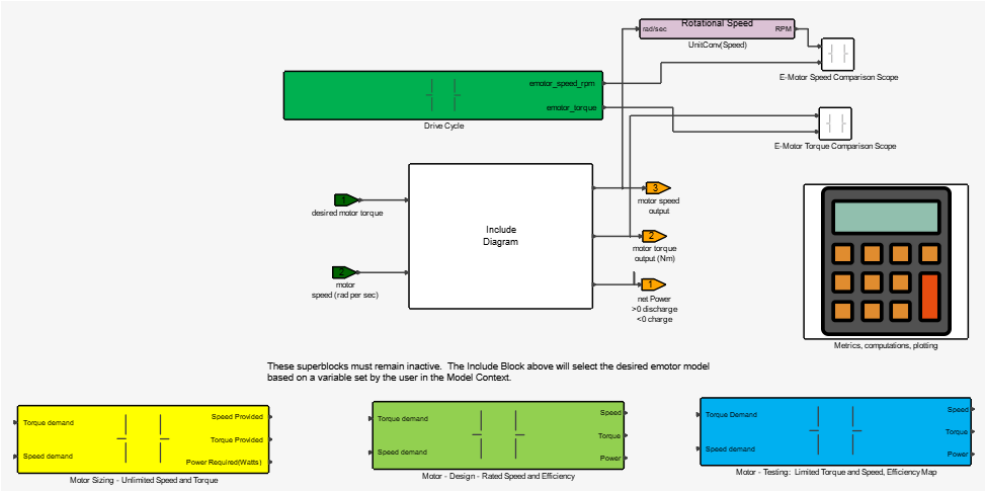


Figure 3.5: *Interior of the Electric Motor Model*

3.2.3 Model Overview

Before getting into the model details, we should first define the scope of this model and the expected goals of the simulation. Our EM model is a high-level simulation, and it aims for two goals for simulation. These goals are:

Goal #1 - Explore E-motor modeling fidelity with different options (component sizing vs. design with equations vs. testing look-up tables from FluxMotor).

Goal #2 - Prepare a system-level model for analyzing vehicle design/battery changes vs. performance (e.g. vehicle range/mileage).

Our model has 5 major parts. Starting from the top, we have the drive-cycle data import first.

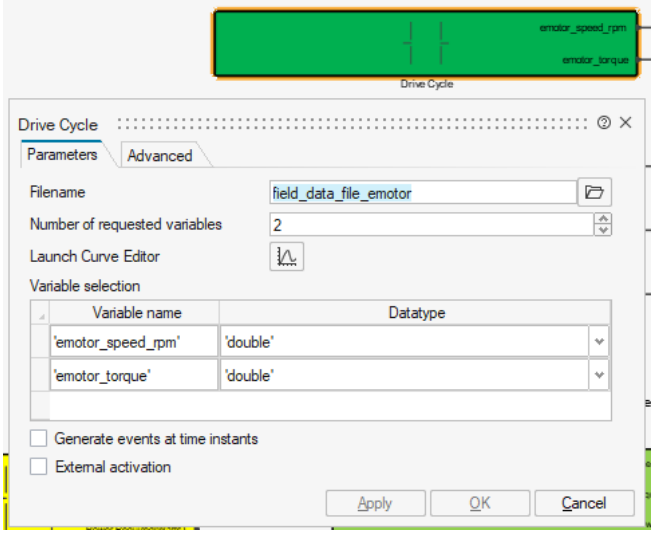


Figure 3.6: *Input Signals for EM Simulation. It is a CSV spreadsheet, and the data is imported during the simulation.*

This block is used to import pre-recorded data measured from a real EM test. The data is provided as a Comma-Separated-Values (.CSV for short) file named “field_data_file_emotor.csv”) and it is included in the example folder. We use this data to compare the results of our simulations with the real data, which is depicted in the next section, i.e., the comparison plots section.

The data can be viewed (and even changed, by using the built-in Curve Editor feature. The screenshot of the recorded data is given below for the reader's reference:

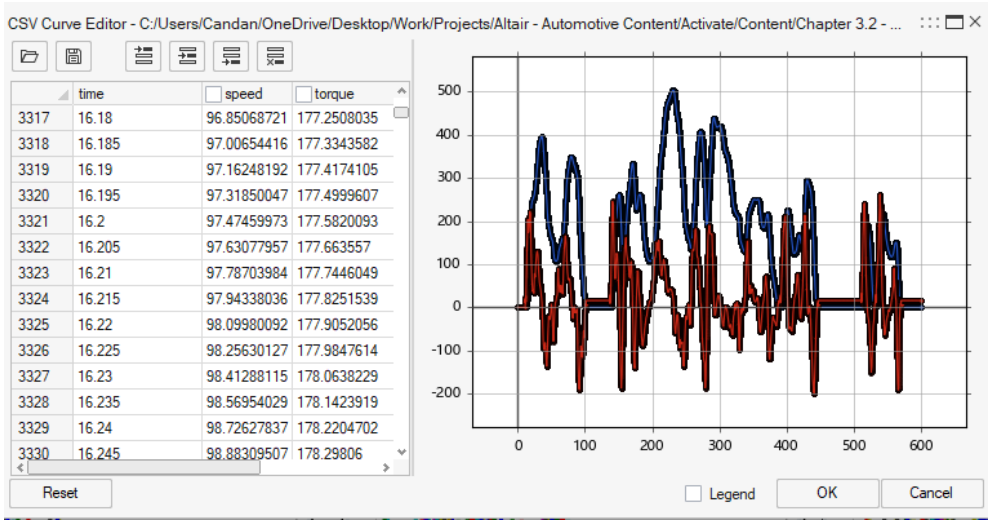


Figure 3.7: *EM Simulation Drive-Cycle (i.e. input) data plot. Red is Torque (Nm), and blue is speed (RPM) demand values.*

The output of the Drive Cycle recorded data is plotted together with the output of the simulation data. We use a radian/sec. to RPM conversion to match the units of both data.

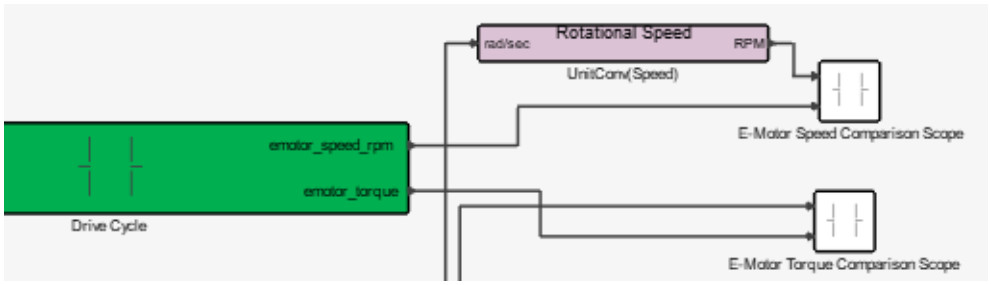


Figure 3.8: *EM Simulation Input and Output Data Plot Section*

As we have 3 different results, we will share the comparison plots in the results section of this subchapter individually.

The simulation data is sourced from the middle "Include Diagram" block. Actually, this is a non-content block, and its purpose is to select one of the superblocks given below. The below image depicts this situation.

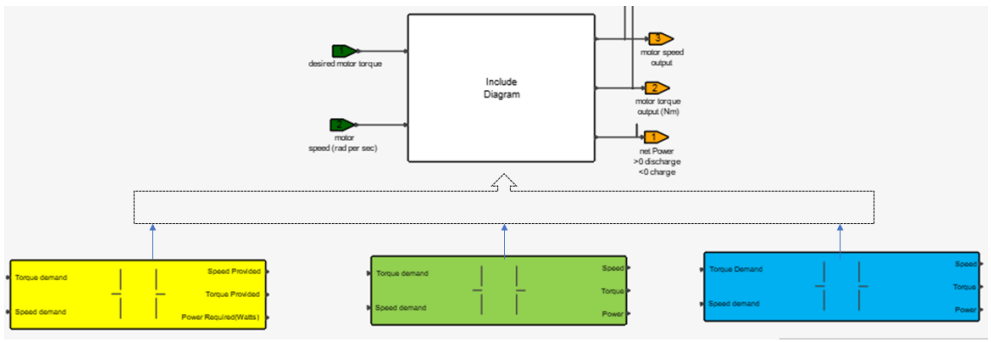


Figure 3.9: *EM Model different fidelity options and how they are selected.*

The superblock with the calculator image (named Metrics, computations, plotting) is used to post-process the output of the 3 superblocks and also plots the additional variables such as the motor powers in kW. It contains an OML Custom block to carry the calculations, plot Torque vs. Speed curves, and create a histogram of power usage. The code is straightforward and includes the necessary comments to grasp.

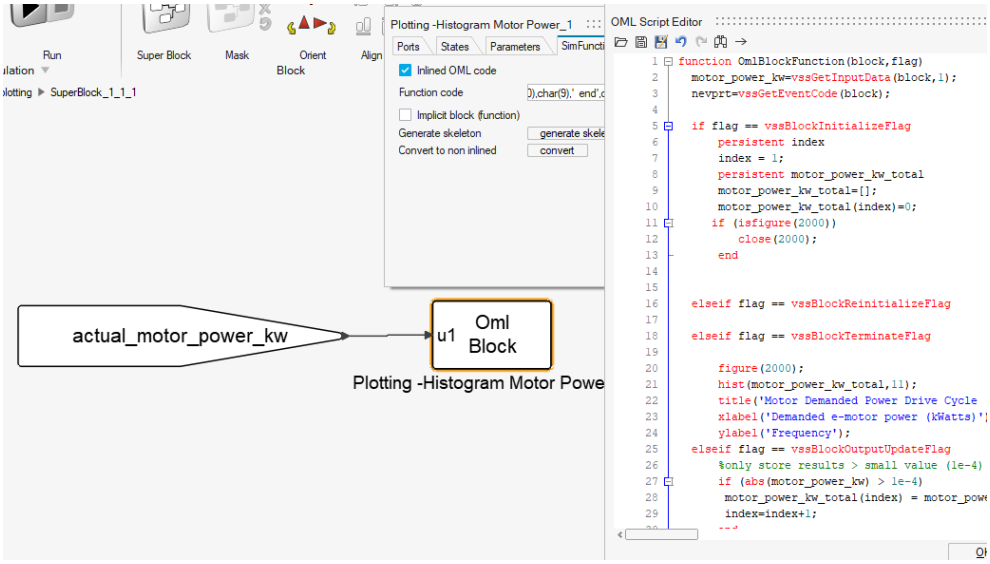


Figure 3.10: *EM simulation results calculation block*

Lastly but most importantly, we have 3 alternative EM models that are used to simulate our EM with different fidelity (i.e. simulation accuracy). We will share the interior of each block hereunder.

EM Model Option #1 is named and also used for sizing purposes. The EM sizing step of any EM application is the earliest phase where the required minimum motor size is determined based on the demand power and speed. Hence, in this model unlimited power/speed/torque is available/valid from our EM.

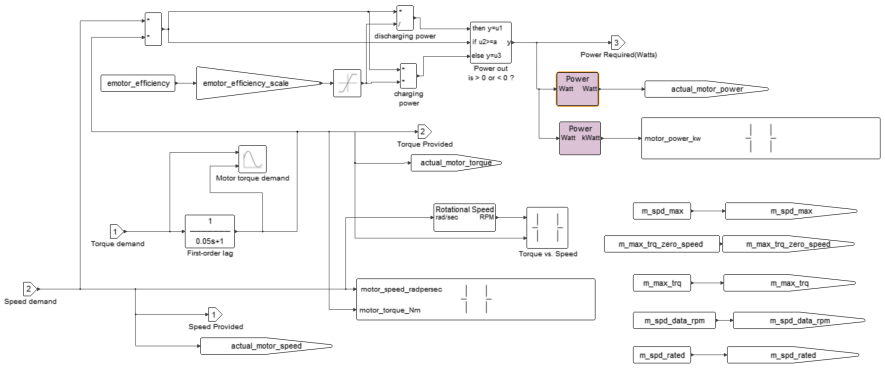


Figure 3.11: *EM Model Fidelity Option #1 - Motor Sizing: Unlimited Speed and Torque*

EM Model Option #2 has the EM simulation with limits on speed and torque. We have mathematical equations for torque vs. speed based on rated speed/rated torque/max speed characteristics of a typical EM.

Our parameters are efficiency, rated speed, rated torque, and maximum speed. Between the rated speed and max speed, constant power and torque computed from this constant power are applied.

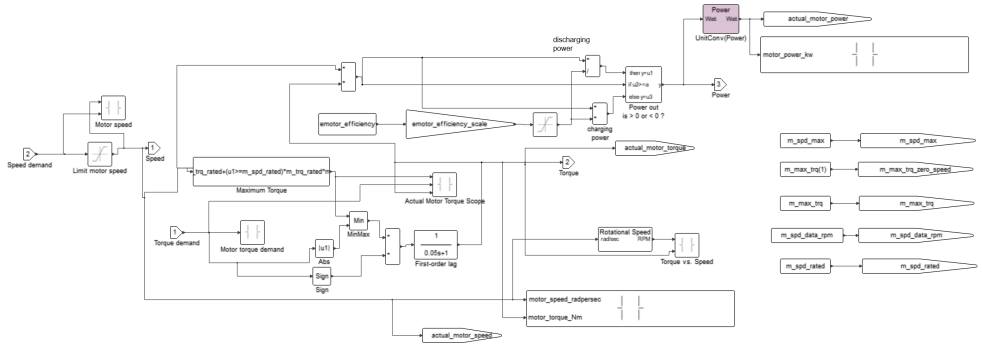


Figure 3.12: *EM Model Fidelity Option #2 - Motor - Design: Rated Speed and Efficiency*

EM Model Option #3 is used for testing purposes. It has the EM simulation with limits on speed and torque but this time from the look-up tables (Flux/FluxMotor) - limits on speed and torque based on look-up tables (e.g., From Flux/FluxMotor).

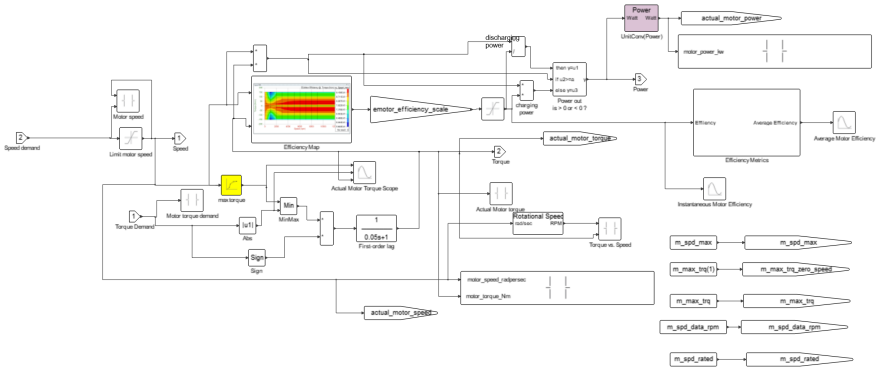


Figure 3.13: *EM Model Fidelity Option#3 - Motor - Testing: Limited Torque and Speed, Efficiency Map*

3.2.4 Results

As discussed in the previous section, our simulation model has 3 options for different fidelity. We will share the results of each fidelity and share our comments on their difference.

Before running our simulation, we also need to inform you about how the motor is controlled. As described in the previous part, the drive cycle is a virtual set of motor speed and torque values, where the vehicle's driver modulates the speed of the motor and (due to road conditions) the torque.

Please repeat the following sequence to run the EM model with different fidelity options:

1. Click the Play button - This will pop up the following selection message. Write 1, 2, or 3 and hit enter to select the e-motor model.

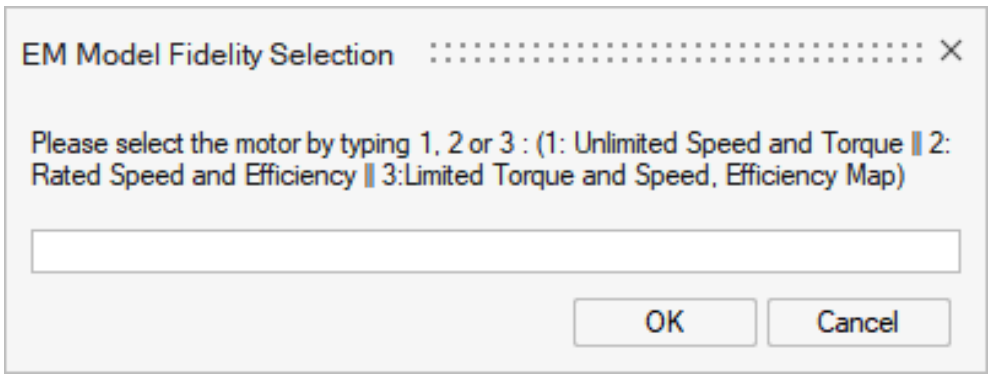


Figure 3.14: *EM Model Fidelity Selection pops up as the model runs*

2. After a while (the simulation is set to run a 600 sec. time period) you should see the Torque vs. speed plot to start adding the simulation data. This plot shows e-motor demands/behaviors. It will take a while to complete.

1st we will start with the ideal case, ie. Unlimited Speed and Torque. As expected from its name, our motor is capable of delivering unlimited speed and power. Hence we don't need to calculate the efficiency in this mode.

As the simulation completes, we will have 2 sets of figures. The Speed vs. time, Torque vs. time, and Power vs. time scope images will provide a single view for the most crucial data over time.

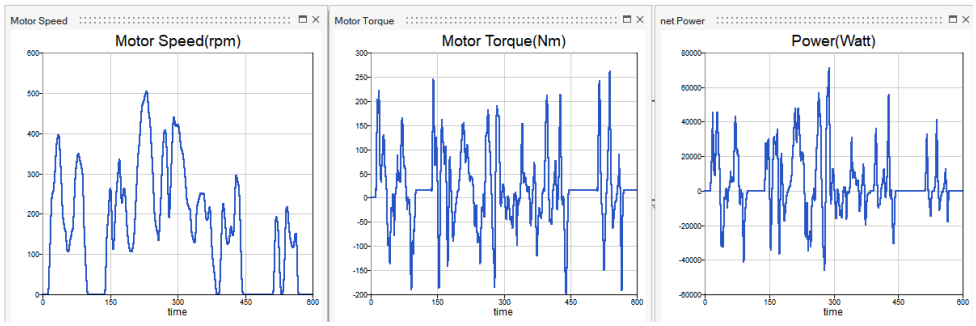


Figure 3.15: *EM simulation outputs vs. time scope graphs*

The main image we will focus on is the Torque vs. Speed in Drive Cycle. It should include all the data from the Drive Cycle (i.e. the requested speed and torque during the 600 sec. simulation period). Normally the torque should be diminished as the speed increases, which will be depicted in the next fidelity models.

2nd we will use the 2nd fidelity option, ie. Rated Speed and Efficiency. Here we will plot to check the Speed vs. Torque plot. The speed and torque ratings are applied, and the overshooting values are trimmed at the edges of the curves.

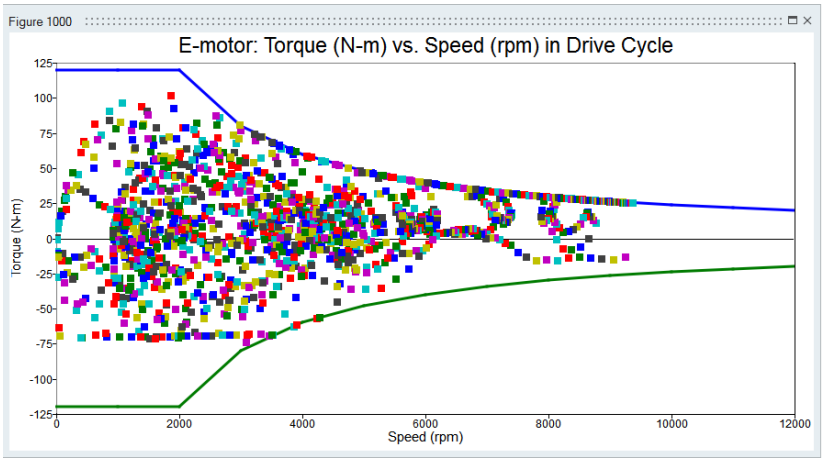


Figure 3.16: *When the simulation runs, it generates the simulation Output as Torque vs. Speed variation plot*

Finally, we will have the highest fidelity model, where the rated power is also applied. In this final fidelity model, we will also plot the motor's average and instantaneous efficiencies, which are closest to reality.

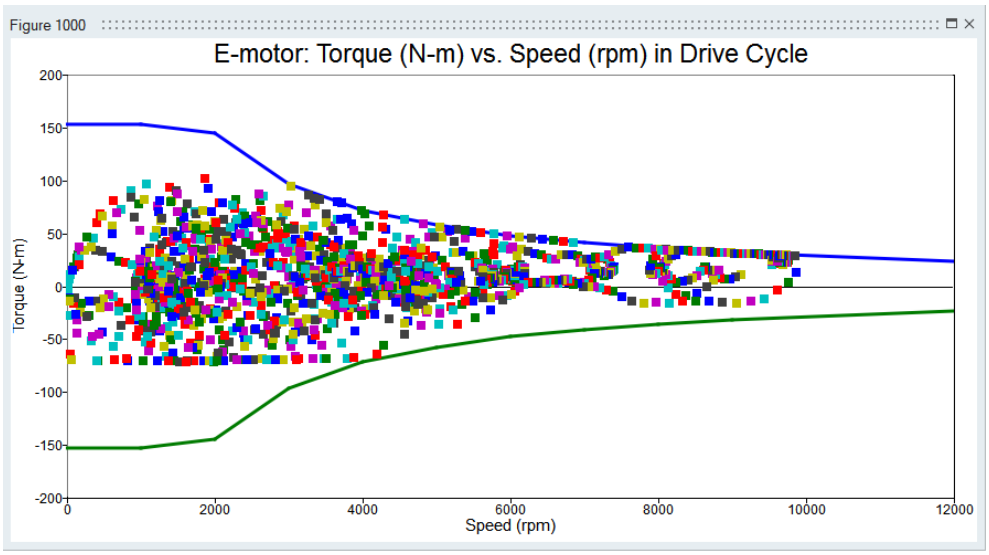


Figure 3.17: *Torque vs. Speed data for the highest fidelity model option.*

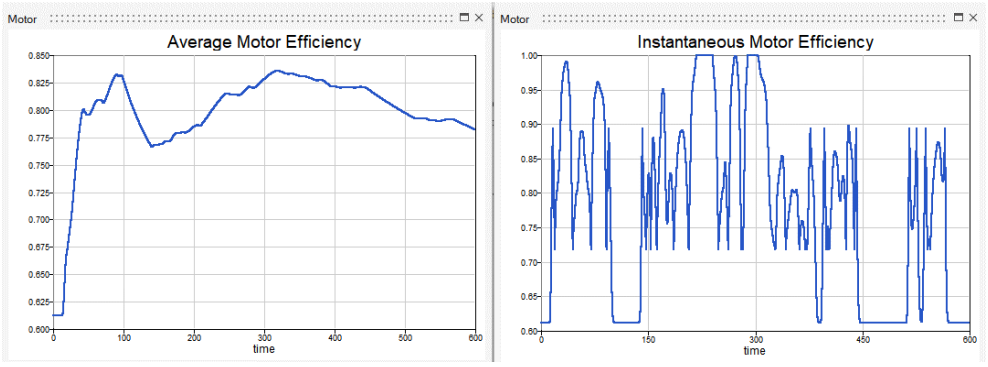


Figure 3.18: *Efficiency plots for the highest fidelity model option. On average, the EM has an efficiency of 0.8.*

3.3 E-Powertrain Elements

In this sub-chapter, we will be giving a simulation model to simulate the powertrain elements between the EM and the wheels. This will be a simple model and readers are encouraged to seek detailed powertrain models.

3.3.1 Brief Description

All of the EVs has some mechanical elements between their Electrical Motors and the wheel structure. We will be focusing on the Gear Reduction Unit (GRU) but there are other mechanical elements such as differential unit, rack & pinion, etc. which are common to EVs and conventional ICE vehicles as well.

The GRU is used to convert the mechanical power available at the shaft of the EM to the crankshaft (or directly to the wheels) of the vehicle. Its main purpose is to reduce the rotational speed of the EM while increasing the torque output in the same ratio.

The parameter of our interest is the Ratio of Gear (RG as short), and it is used to convert the EM speed into the wheels speed. Our model uses RG as a design spec along with a constant driveline efficiency parameter. The driveline efficiency is used to represent the friction, backlash, etc. inefficiencies of the mechanical system.

3.3.2 Model Figure

The superblock and inner diagram of the E-Powertrain are given below.

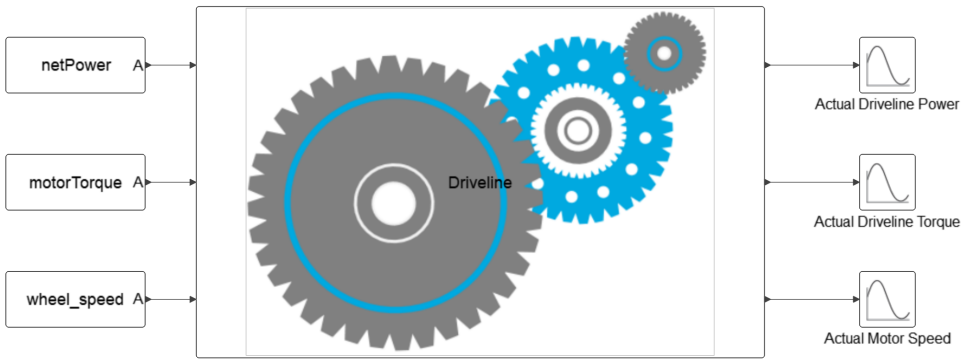


Figure 3.19: *E-Powertrain Superblock Figure and Its Inputs & Outputs*

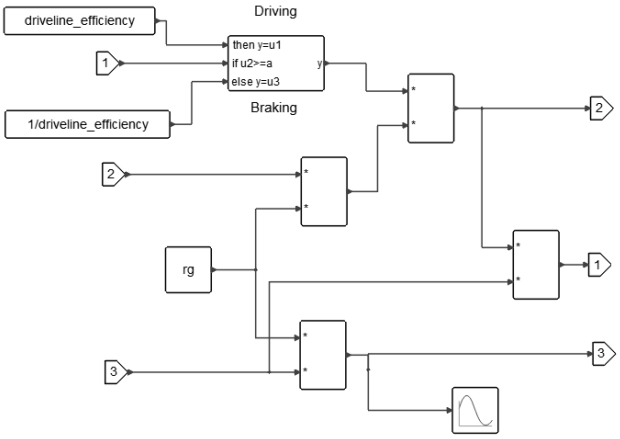


Figure 3.20: *E-Powertrain block diagram*

3.3.3 Model Overview

Our model has 3 inputs and also has 3 outputs as depicted below:

Input #1 - Net Power Input. Based on the sign of this signal (- or positive) we select the driveline efficiency between the acceleration or deceleration modes. Please note that, for simplicity, we applied the same efficiency values to both modes.

Input #2 - Instantaneous Motor Torque. This is the input from our EM which will be added (or subtracted) to the actual wheel speed with the RG factor and the driveline efficiency.

Input #3 - Wheel Speed Input. This is the current value of the wheel speed, which is used to iterate and calculate the next iteration's speed value.

Output#1 - Actual Power Applied to the Driveline. This is the power (negative or positive) that drives the wheels. It is optional and can be used for future calculations.

Output#2 - Actual Torque Applied to the Driveline. This is the amount of torque that is applied. It will be applied to our vehicle dynamics model to calculate the vehicle's speeds, accelerations, etc.

Output#3 - Actual Motor Speed. As the motor drives the driveline, its speed will change according to the load (mostly due to the road profile). This value will be fed back into the EM calculations for iterative evaluation.

3.3.4 Results

Similar to the previous EM model, we will be driving our E-Powertrain with pre-defined motor power, torque, and wheel speed values. These values are again imported with the FromCSV file blocks. The signal for Input #1 - Net Power is given below.

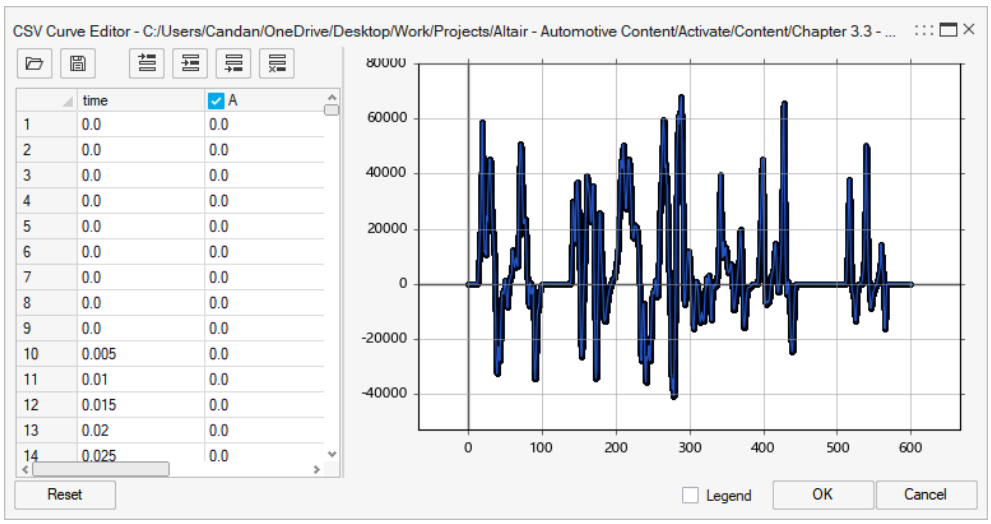


Figure 3.21: *Inputs for the E-Powertrain Model Simulation*

After we run our model with these predefined values, the outputs are calculated based on the RG and efficiency coefficients. Here are the results of the E-powertrain simulation model.

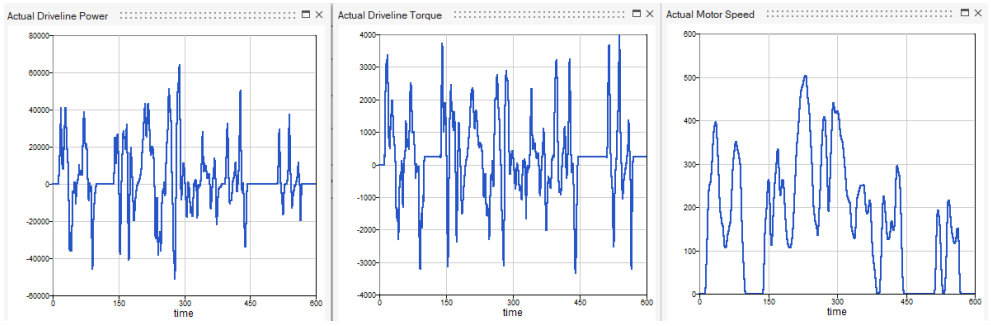


Figure 3.22: *Results of the E-Powertrain Simulation with the predefined energy, torque, and motor speed test values.*

3.4 Battery

Battery can be considered as the fuel source of the EV. Similar to re-fueling conventional Internal Combustion Engine (ICE) vehicles, users should also recharge the battery of the EV to travel any distance.

However, unlike ICE vehicles -where the engine can not generate any fuel-, the battery of the EVs are re-chargeable during the drive, and it is one of the key changes of the EV compared to ICE Vehicles.

We will be focusing on this topic a little bit more. To do so, we will give a simple simulation model of the Battery and analyze its energy inputs and energy outputs. The Inverter & Converter sub-system is responsible for charging the battery. Hence we will also give some information about this sub-system as well.

Extension of driving range and battery run time optimization are necessary key points in the modeling of Electric Vehicles (EV). In this view, Battery Management System (BMS) also plays a major role in ensuring a safe and trustworthy battery operation, especially when using Lithium-ion (Li-ion) batteries in an electric vehicle. The key function of the BMS is State of Charge (SoC) estimation. There are different SoC estimation methodologies such as direct estimation methods and model-based estimation methods. In this e-Book we will be using the direct estimation method. However, readers may refer to [2] [3] to learn more about the alternative SoC estimation methods as well.

3.4.1 Brief Description

As described above the battery system of the EV can be split into 2 major subsystems. These are;

- Inverter / Converter Electronic Subsystem
- Battery Pack Subsystem

The figure below depicts the relationship between these 2 parts. As can be seen, the Inverter/Converter subsystem collects all the electrical demands of the vehicle and then controls the battery pack by either pulling power from it (i.e., discharging) or pushing power to it (i.e., charging). This is depicted as the (total) Power Demand. The Inverter / Converter block is also responsible for AC / DC and DC / DC conversions to match the suitable DC voltages of the battery pack.

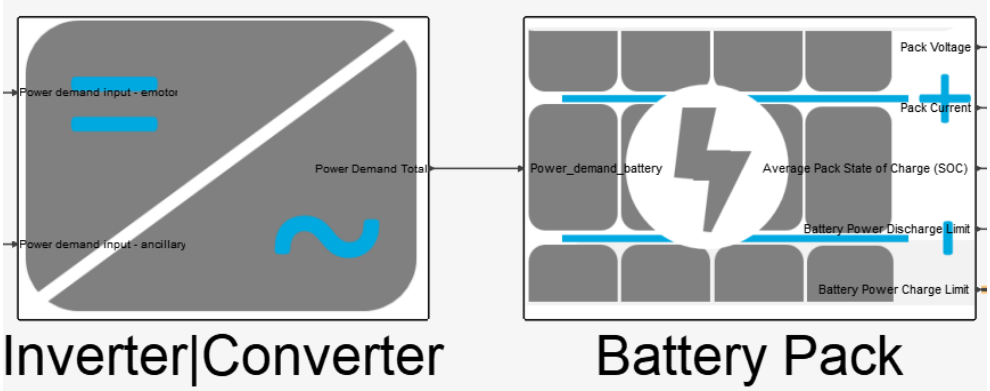


Figure 3.23: *Battery system is based on 2 consecutive subsystems.*

The battery pack is a stack of chemical battery objects. It is modeled as a subsystem, and our model makes necessary calculations inside to give outputs such as the voltage levels, current levels, and also the State of Charge (SoC) of the battery. We will investigate each model separately and again combine all the models in our chapter ending full vehicle simulation model.

3.4.2 Model Figure

We will start with the model of the Inverter / Converter subsystem first. Although it is a very simple simulation model, readers should know that this is a crucial part of the battery system and it plays an important role in the efficiency and mileage of the EVs.

The simulation file InverterConverter.scm (which is an isolated model) is our Inverter / Converter subsystem, and it has the following cover image. The DC and AC signal images represent the electrical conversion of this unit.

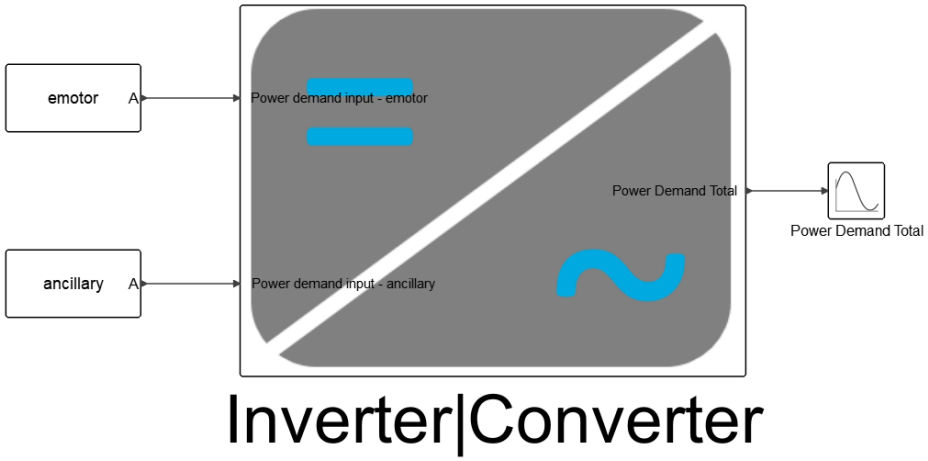


Figure 3.24: *Inverter / Converter Superblock Inputs & Outputs*

Our Battery Pack model is named BatteryPack.scm and it has the following model image and input & output names.

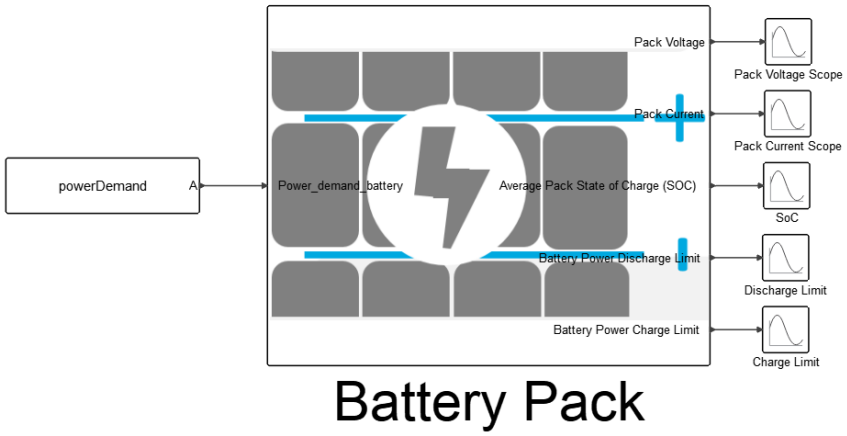


Figure 3.25: *Battery Pack Model Figure, its Inputs & Outputs*

As the image implies, the battery pack is a combination of multiple smaller battery units, and they are interconnected to each other to make the full battery pack.

Next, we will share details of each model.

3.4.3 Model Overview

As explained, the Inverter/Converter subsystem is a very simple model used to represent the physical presence of the system. Below is the model interior.

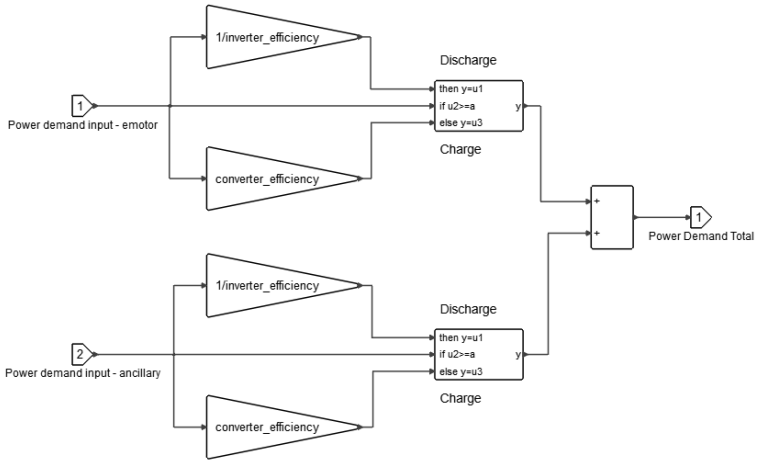


Figure 3.26: *Inverter / Converter Model Block Diagram.*

The model simply adds the demand powers by multiplying them with the inverter efficiency coefficient. This coefficient is available within the Model Context. The sign of the power demand input determines how the efficiency is applied. If there is a positive power demand (i.e., discharge), then it is divided by efficiency. If there is a negative demand (i.e., recharge), then the amount is multiplied by the efficiency.

Please note that due to simplicity we will not share simulation results for this sub-system.

The Battery Pack subsystem's interior is given below. We will analyze this model more in-depth.

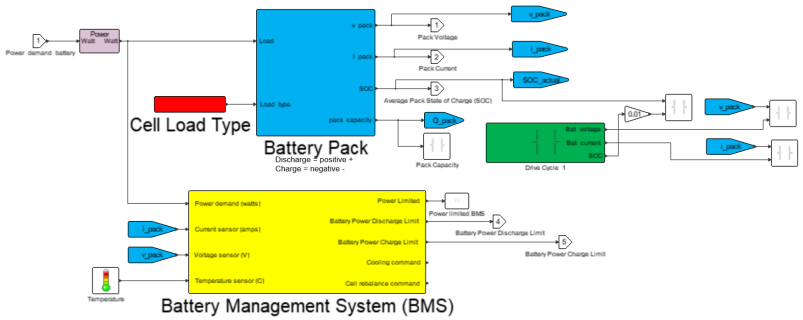


Figure 3.27: *Battery Pack Subsystem's block diagram*

Our Battery Pack subsystem is based on 2 main blocks and 3 minor (supportive) blocks. The main blocks are as follows:

1. Battery Pack (physical/chemical system) model.
2. Battery Management System (BMS) model.

The minor blocks are the Cell Load Type selection block (as an input to the Battery Pack), the power conversion block (if needed), and the temperature sensor simulation block.

The battery pack model has 2 different sub-model options (Coulomb counting method and Voltage LUT with Coulomb counting method) to simulate the pack and calculate its state of the charge. Block diagrams of these sub-models are given for reference:

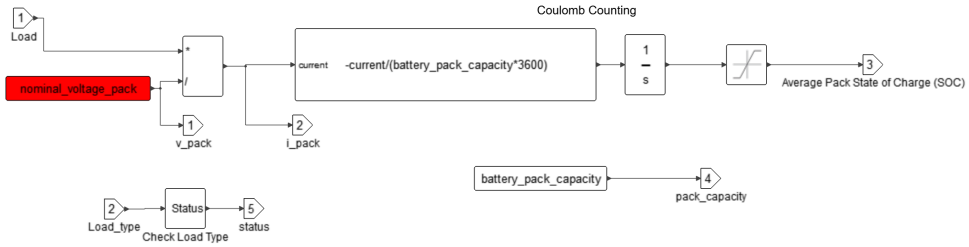


Figure 3.28: *Coulomb Counting Method for Battery Pack. This is a simple linear method to calculate the Battery Pack's Average SoC.*

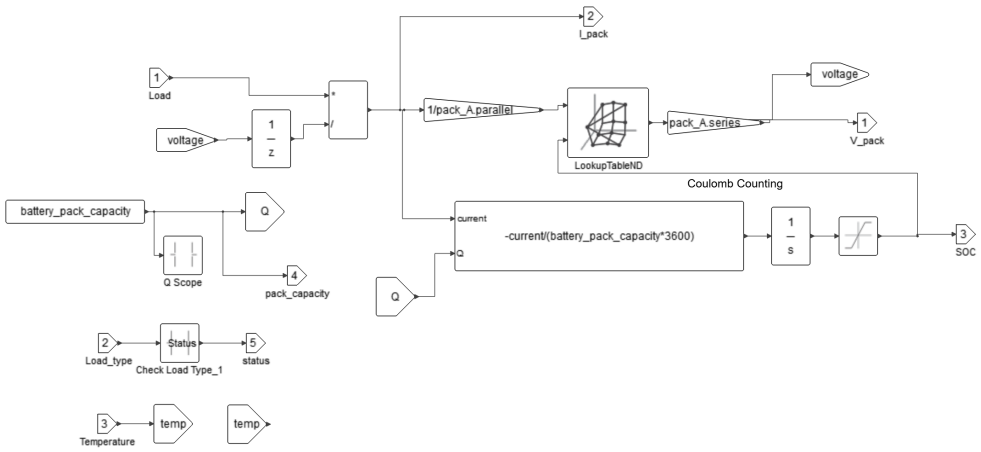
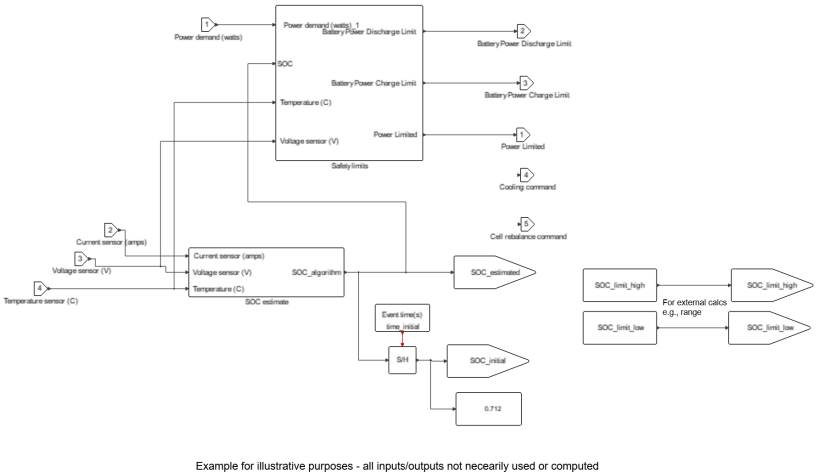


Figure 3.29: *Voltage LUT with Coulomb Counting Method for Battery Pack. This is a more sophisticated method to calculate the Battery Pack's Average SoC.*

The Battery Management System is a sophisticated electronic control system, using multiple sensor inputs and running different algorithms to control the charging/discharging currents of the battery pack. We have a simple model again to simulate the functionality of this subsystem. The Interior of this block is given hereunder:



Example for illustrative purposes - all inputs/outputs not necearily used or computed

Figure 3.30: *Battery Management System block diagram. This subsystem works together with the Battery Pack subsystem.*

3.4.4 Results

Similar to all the models in this chapter, we will first introduce the pre-recorded test values for the power demand input of our Battery Pack. The below image is acquired from the FromCSV block of the Battery Pack model. As expected the vehicle is supposed to demand power (positive power demands in the figure) in general, but there are some times (braking events) when power is fed back into the battery pack as well (negative power demands of the figure).

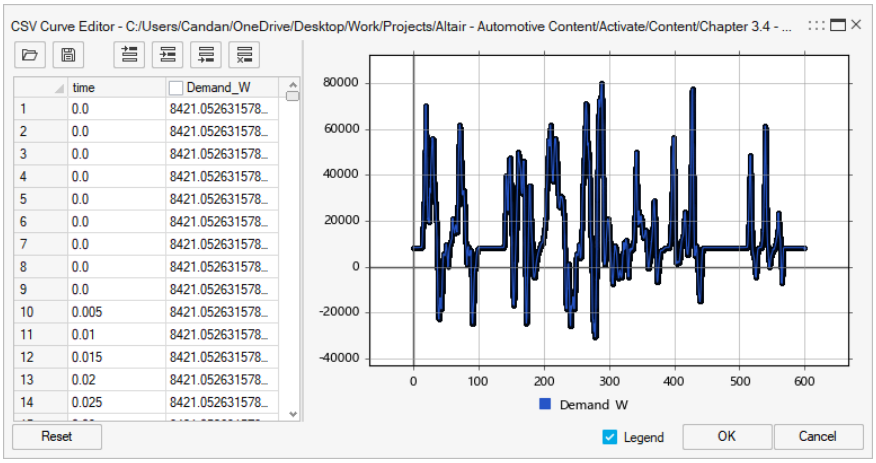


Figure 3.31: *Power Demand input data for Battery Pack simulation. The positive values are discharging events; negative values are charging events.*

As we run our simulation with this data, our model will generate 3 important scope graphs as follows.

Figure 3.32: *Battery Pack simulation outputs. Battery Voltage drops when the current is drawn (positive) and increases when the current is fed (negative).*

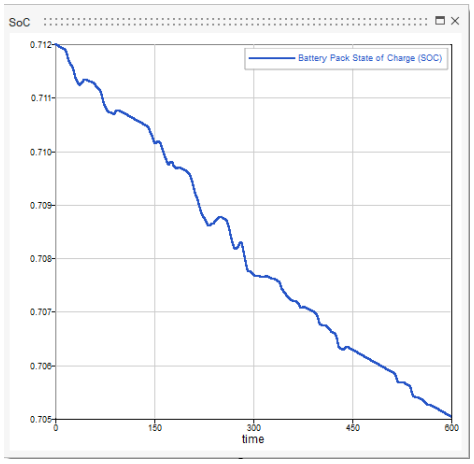


Figure 3.33: *State of Charge output of the Battery Pack simulation.*

As expected, the Voltage and SoC of the battery pack closely follow the current direction and amplitude of the current as well. Please also note that our drive cycle has some zero-velocity areas (where the vehicle is at a stop). In these areas, the ancillary power demand (such as Lighting, Vehicle Entertainment Systems, A/C, etc.) is at a constant level.

This concludes our explanation of the major EV subsystem models. Next, we will combine all these models along with our Vehicle Dynamics models and some ancillary simulation functions.

3.5 EV Full Simulation Model 1-D

3.5.1 Brief Description

In this final part, we will be assembling all the simulation models that we have presented so far. Please note that the pre-recorded signals of the subsystems will be deleted and this time the data from related subsystems will be fed into each. Our Full Simulation EV Model is composed of the following 9 subsystems:

1. **Drive Cycle:** This is the road profile (i.e., angle) and the desired speed (i.e., min. speed requirement)
2. **Driver:** We have a driver model that modulates the engine speed or applies brakes according to the desired speed input. More info will be given later.
3. **Electrical Motor model:** This is the same model that has been introduced in the Electric Motor sub-chapter already. Used to translate the Driver's speed (or brake) commands into the EM.

4. **Driveline model:** This is the same model that has been introduced in the E-Powertrain elements sub-chapter already. Used to translate the EM's power output to the wheels.

5. **Vehicle Dynamics:** This is the Full Vehicle dynamics model that has been introduced in Chapter #2. It is used for calculating the vehicle's speed, acceleration, and pitch angles based on the power transmitted from the Driveline block and the existing speed, angles, etc.

6. **Ancillary Systems:** This is a simple model to represent the electrical loads that are consuming electrical power from the battery system (such as A/C, internal cooling system components, etc.).

7. **Inverter / Converter System:** This is the same model that has been introduced in the Battery sub-chapter already. Used to combine the EM's power output and the Ancillary Systems' power requirements and commands the Battery Pack to maintain the overall power flow.

8. **Battery Pack:** This is the same model that has been introduced in the Battery sub-chapter already. Based on the net power requirement, it is charged or discharged and continuously monitored for its voltage, current, and State of Charge levels.

9. **Range Calculation Model:** It is a simple arithmetic model which continuously calculates the EV's range based on the Battery Pack's SoC, current speed, and low-limit conditions. More info will be given later.

Let's start with presenting the full model and take a closer look at each newly introduced /required subsystem.

3.5.2 Model Figure

Our full EV model is prepared as a stand-alone simulation model and has a simple read-me section. Here is the Top-Level of our model:

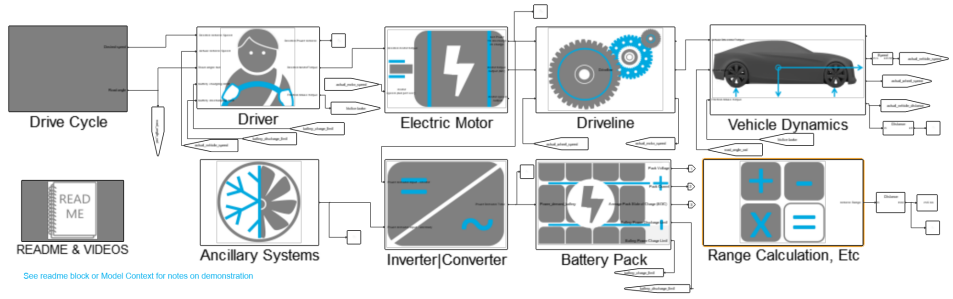


Figure 3.34: *EV Full Simulation Model (1-D)*

As can be seen, the subsystems are interconnected with each other either directly or through the Set or Get Signal variable blocks. We also have some unit conversions along with some scope graphs for showing the major variables (such as the range) in the top level.

Readers are encouraged to open each model and understand how they work and what variables are needed as well.

Next, we will share the results of a sample run.

3.5.3 Results

Before running the example, let's introduce the inputs to the model. We have 2 inputs coming from the Drivecycle block. These are the Desired Speed and the Road angle. This information is fed via the Read-CSV block in the Drivecycle subsystem. The block and its data are given in the below figure:

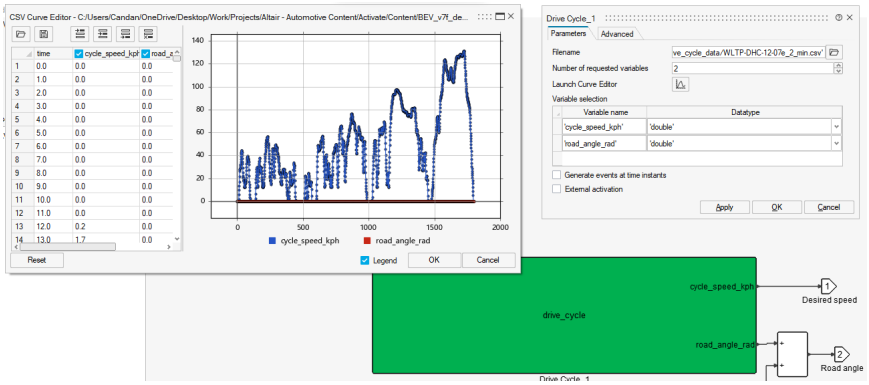


Figure 3.35: *Drivecycle data that is used in the Full EV Simulation.*

As we hit run in our model, this data is parsed and fed into consecutive blocks. Here are the major plots of the simulation run with this data:

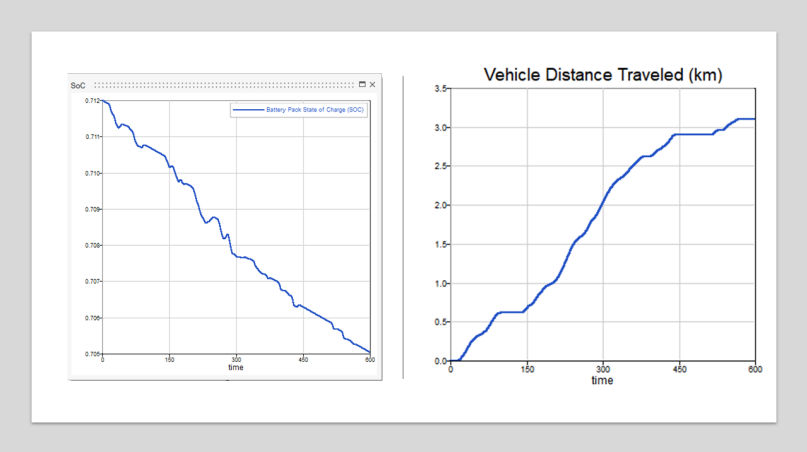


Figure 3.36: *Battery Pack's State of the Charge vs. the distance traveled (0-600 sec. plotted).*

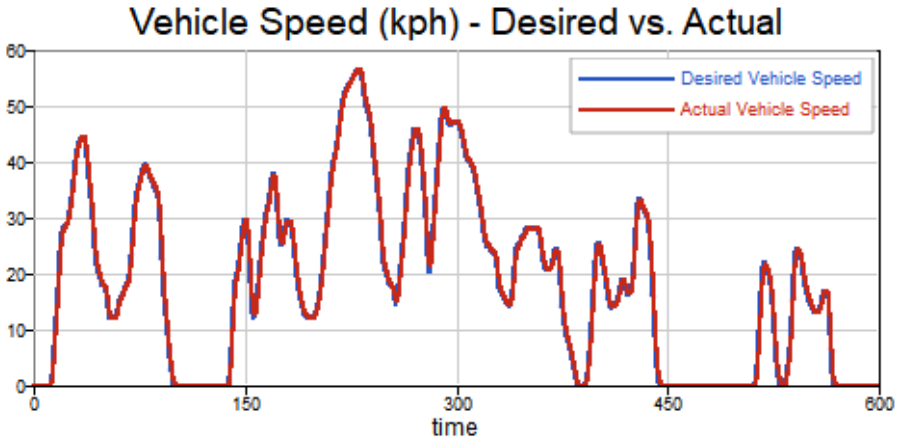


Figure 3.37: *Vehicle's desired speed vs. actual speed (0-600 sec. plotted).*

Our results are in alignment with the expectations. 1st, our vehicle is capable of generating the requested speed during the whole course of the simulation. We also note that the battery state of charge is reducing, which is also within expectations. Our vehicle stops at intervals, where the battery keeps discharging (due to ancillary power consumption).

Lastly, let's check how our EM delivers the power to the E-Powertrain. As seen in the figure below, our EM generally produces high speed while we still have room for torque. That means we can increase our speed a little more, but the EV has a speed limit as well.

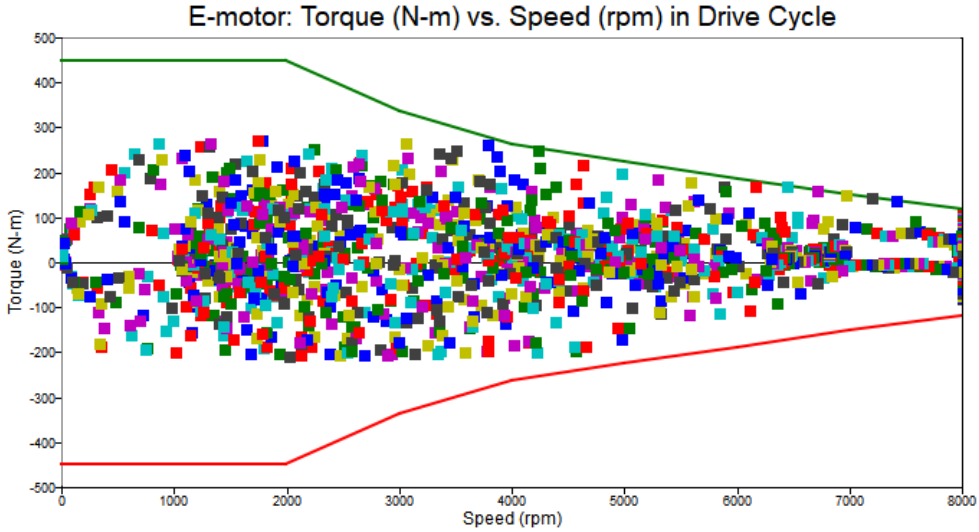


Figure 3.38: *Power output of the EM during the drive cycle.*

You may try changing the road's slope or vehicle speeds as an exercise. You should observe the battery state of the charge will change differently and can fully deplete the battery as well.

3.6 References

[1] FluxMotor - Online Resource Library, available on https://www.altair.com/resourcelibrary?keywords=/fluxmotor.

[2] Maheshwari Adaikkappan and Nageswari Sathiyamoorthy. "Modeling, state of charge estimation, and charging of lithium-ion battery in electric vehicle: a review". In: *International Journal of Energy Research* 46.3 (2022), pp. 2141–2165.

[3] Martin Murnane and Adel Ghazel. "A closer look at the state of charge (SOC) and state of health (SOH) estimation techniques for batteries". In: *Analog devices* 2 (2017), pp. 426–436.

4

SECTION

Electric Vehicle Simulation Development

4.1 Introduction

The models described in previous chapters are acceptable when you concentrate on lateral or longitudinal or vertical motions individually; however, for complex maneuvers that involve a combination of motions, a 3D modeling technology is needed.

The 3D-Multibody simulation uses the power of a computer to design, evaluate, and optimize complex systems using sophisticated mathematical modeling and solution tools [1]. Characterized by large displacements, it is the study of the motion of mechanical systems caused by external forces and motions. Along with 1D simulations, it acts as a companion to the traditional process of refining products using hardware prototypes. Altair's MotionSolve is a comprehensive multibody simulation software used to build and execute complex system models to evaluate the dynamic response of products and optimize their performance.

In today's automotive industry, multi-body simulation is routinely used to design and evaluate suspension systems, to improve ride and handling characteristics of passenger cars, to extract loads for durability analysis of various components and to help validate mechatronic components. Most models include controllers and modules [like an Anti-lock Braking System (ABS), Traction Control System (TCS), Electronic Stability Program (ESP), FMU based Electric Powertrain with Regenerative Braking] from system design packages like Activate that use techniques of classical and modern control theory. The multibody models can be used to run co-simulation with discrete element models (like Altair EDEM) to evaluate the interaction with discrete elements (like soil/, minerals). Often, these models are also coupled with optimization software (like Altair HyperStudy) to run Design-Of-Experiments (DOE) and Optimization. Together, these form a powerful set of tools to evaluate the complex physics of an automobile subjected to real-life events. A non-exhaustive list of applications of multibody simulation in various automotive systems is given in Table 4.1.

Chassis	Suspension design, vehicle dynamics, handling and ride analysis, durability, steering system design
Powertrain	Engine performance, transmission ratios, powertrain concepts, engine noise
Body Applications	Door slam, roll-over
Others(Components)	Disk brake system, manual clutch, retractable seats, wiper, door lock mechanism, vibration on the gear shift lever

Table 7: *Applications of multibody simulation in automotive industry*

Specifically, in vehicle dynamics, multibody simulation is used for evaluating:

1. Handling characteristics – tire road grip, stability during turns, steering response, lateral forces and accelerations, wheel load fluctuations, etc.
2. Ride/Comfort characteristics – suspension properties, NVH behavior, the effort for driving and parking

4.2 Input for Multibody Simulation

The virtual prototyping using multibody simulations involves the following phases: Build – Test – Review – Improve [1].In the 'Build' phase, physical systems are converted to meaningful computer models using mathematical formulation and idealization techniques. Properties of modeling elements like masses, inertias, stiffnesses, attachment locations, orientations, stiffness, damping, forces, and motions are all applied to the idealized model. In the 'Test' phase, simulations are conducted to ensure results from simulations and experiments match. Model changes are incorporated (correcting modeling assumptions, property modifications based on isolated tests, using higher fidelity modeling elements, etc.) In the 'Review' phase, the performance of the system is assessed, and the effect of changes to parameters on the performance is studied. In the 'Improve' phase, the performance measure is identified, along with design variables and their limits. A set of functional virtual tests are run to improve the performance.

In contrast to 1-D simulations, the process of building and analyzing models involves several additional input variables. Figure 4.1 lists the technologies used for acquiring these additional inputs.

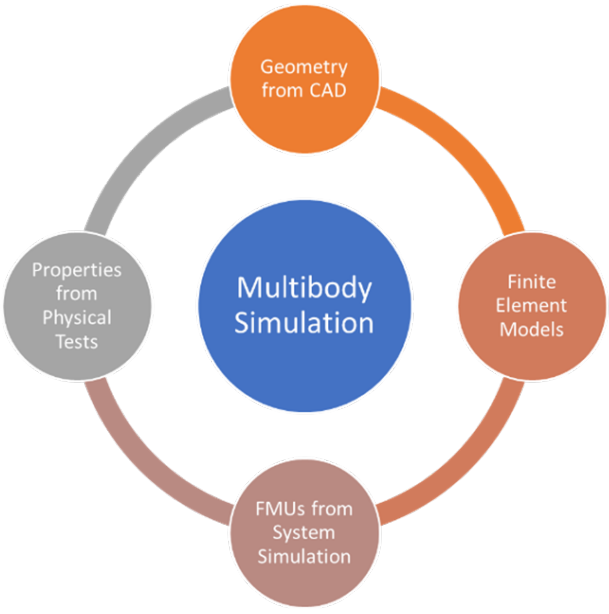


Figure 4.1: *Input for multibody simulation*

1. The geometry from CAD software is used to create rigid parts (mass and inertias), locate joints, and create joints between various components of the subsystems.

2. Finite element models are used to include flexibility of components in multibody simulation, thus increasing the fidelity of the model. The loads and boundary conditions from multibody simulation can be used for durability analysis.

3. Electrical, hydraulic, pneumatic, and mechanical subsystems can be modeled in 1-D system simulation software and exported as Functional Mockup Units (FMUs) to multibody simulations. Some of the built-in systems from Activate that are available for use in MotionSolve are given in Figure 4.2. Since Activate files are provided in the installation folder, users can modify them according to their needs and re-export the systems for use in a multibody simulation.

4. The output from a physical test of a prototype is frequently used to define linear and nonlinear component properties in a multibody model. Frequency and amplitude-dependent bushings, tire properties, and road profiles are some examples that use tests to obtain input to the multi-body simulation. Physical testing is also used to validate the behavior of a multibody model.

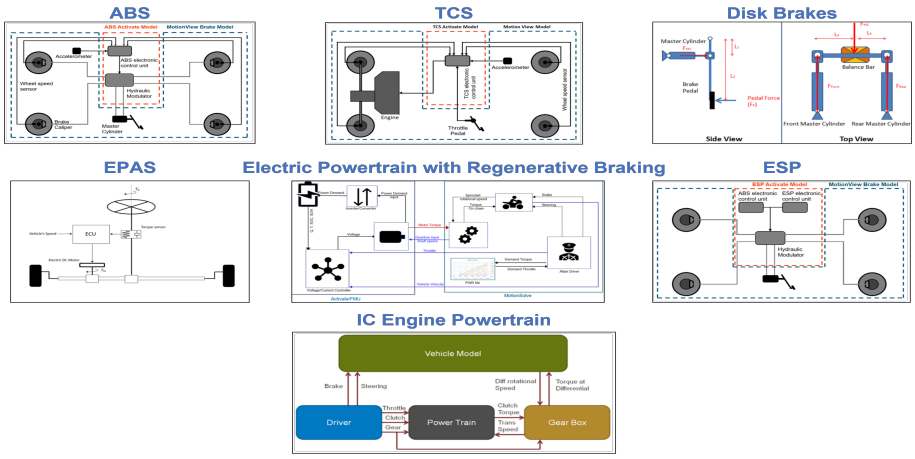


Figure 4.2: *1-D systems available in MotionSolve*

4.3 A typical case study of a multibody simulation

In this section, we examine a typical multibody simulation workflow used in the automotive industry. MotionView is used for modeling [2], MotionSolve for solving [3] and HyperStudy is used to understanding and improving the performance [4].

4.3.1 Definition of the problem

The dynamic behavior of a passenger car is an important aspect of active vehicle safety. One of the tests used is the severe lane-change maneuver, which is performed to determine how well a certain vehicle evades a suddenly appearing obstacle. This test has been standardized in ISO 3888-2. [5]

In this example, a severe lane-change event (double lane change event) of a passenger car is performed at a certain speed. The objective of the analysis is to keep the vehicle from veering off the given path at 65 kmph.

4.3.2 Build and Test the model

A baseline model is built from the example model given in MotionView. An ESP (with ABS) module along with a double lane change event are added to the vehicle. The event file is edited to ensure that the braking and acceleration of the vehicle inside the course are disabled and the driver is allowed only to steer the vehicle to maintain the course.

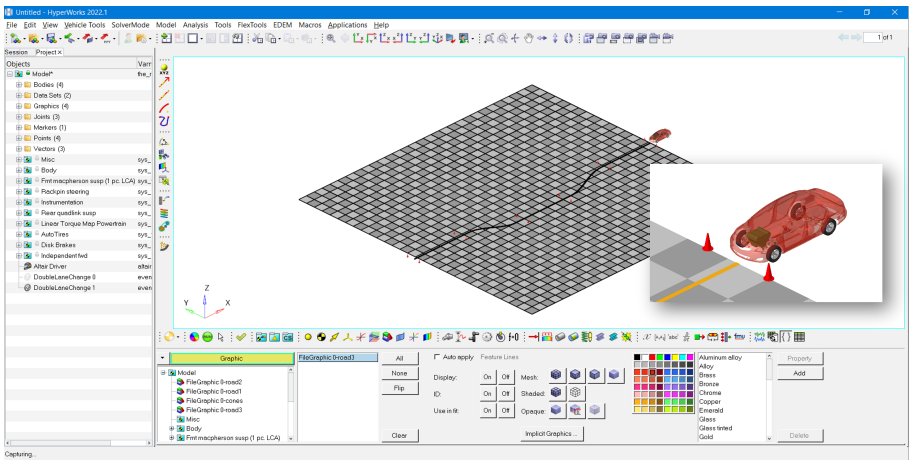


Figure 4.3: *MotionView simulation model of the Vehicle (The image is courtesy of Altair Inc.)*

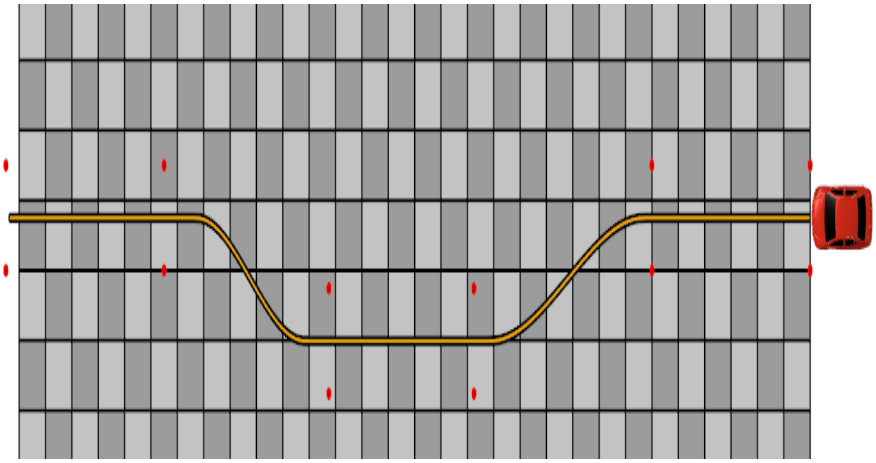


Figure 4.4: *A double lane change event*

4.3.3 Review the results

The baseline vehicle passes this test at 60 kmph. The vehicle can maintain the course. Please click on the link below to see the animation.

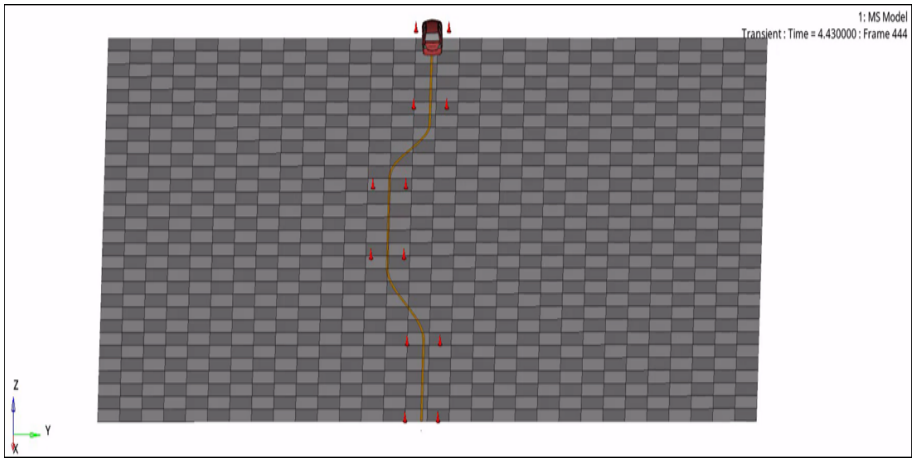


Figure 4.5: *Behavior of baseline model at 60 kmph*

The same vehicle model when run at 65 kmph does not show satisfactory behavior. It veers off the path

and ends up outside the course. An animation of the same can be seen below.

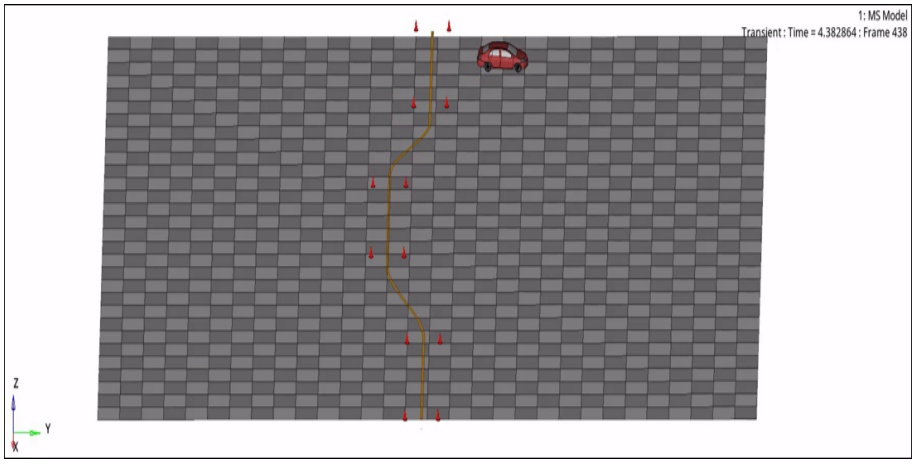


Figure 4.6: *Behavior of baseline model at 65 kmph*

From the baseline simulation, it is very clear that the current model will need design changes to achieve the required target of following the path.
There are two ways to improve performance:

1. Change suspension parameters (Hardpoint locations, Suspension Stiffness)
2. Change the Electronic Stability Control (ESC) program to handle this event

In this example, we attempt to modify the suspension parameters to achieve the desired performance.

4.3.4 Improve the model

Design of Experiments
Before we attempt to improve the model, it is best to screen the design variables and reduce the number of variables that need to be used in optimization. Design-of-Experiments (DOE) is used to evaluate the effect of each design variable on the response variable we are trying to improve.

In this example, design variables considered for DOE studies are

1. Hardpoint locations: Lower ball joint (front and rear), Outer tie rod ball joint, lower control arm, Rear lower bush location, Struct rod upper and Strut tube lower
2. Stiffness of coil springs (front and rear)

We need a response (a measure) that shows the extent to which the vehicle stays on course. The simplest response is the YAW angle of the center of gravity of the vehicle during the event (although more complex responses can be used: e.g., RMS difference between an ideal path and the current path).

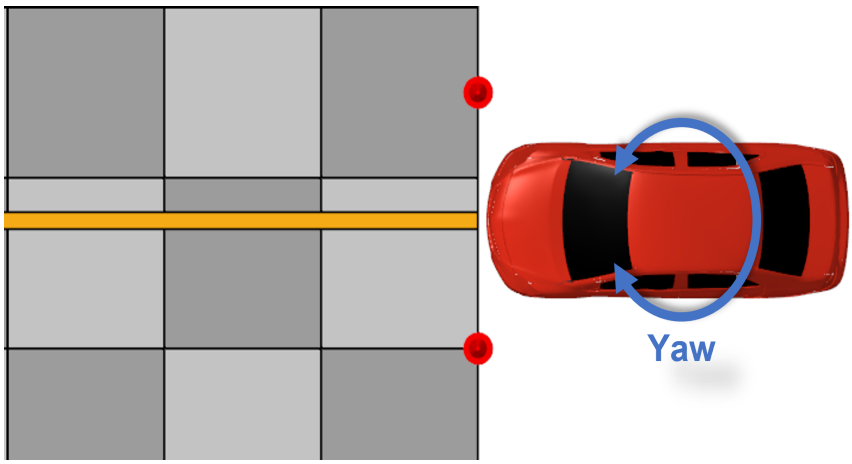


Figure 4.7: *Yaw angle of the vehicle*

As can be seen from the graph below, the YAW for the baseline model at 60 kmph stays between +20 degrees and -20 degrees. We can, therefore, use the Max(YAW) and Min(YAW) as the two responses.

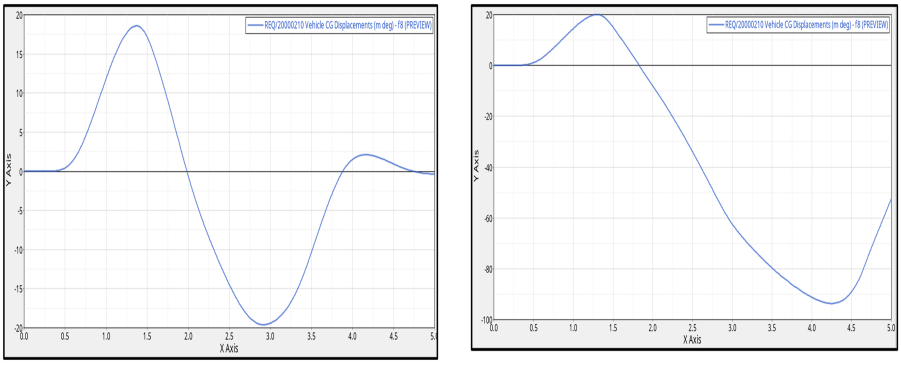


Figure 4.8: *YAW angle variation with respect to time for the baseline model (a) at 60 kmph (b) 65 kmph*

At 65 kmph, the YAW angle variation is well beyond +/-20 deg; clearly, this means that the vehicle is not following the required course.

We run a DOE study after connecting the MotionView model to HyperStudy. The DOE study provides the relationship between the design variables and responses.

As shown in figure 4.8, design variables like the Z coordinate of the outer tie rod ball joint and the Z coordinate of the front lower ball joint are more significant than the Y coordinate of the Strut rod upper location Z coordinate of the Front lower bush location. Using these charts, the top 13 design variables are chosen as design variables for an optimization problem.

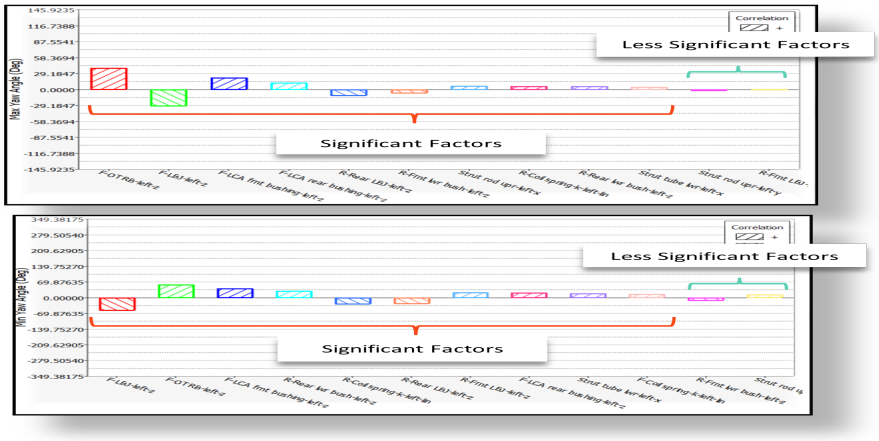


Figure 4.9: *Results from DOE studies showing significant and less significant factors/design variables*

Optimization

Once the top input variables are identified, limits for these are selected. In this example, a limit of +/-10% is used for the hardpoints. Coil spring lower and upper limits were 10 N/mm and 50 N/mm, respectively.

A minimization problem was set up to achieve the maximum yaw angle to be 20 deg and min yaw angle to be -20 degrees. The two new responses are:

1. Response 1: (1-max(ds_1)/20.0) $\hat{2}$
2. Response 2: (1-abs(min(ds_1))/20.0) $\hat{2}$

where ds_1 is the value of the YAW angle during the simulation.

A Global Response Surface Methodology (GRSM) is used to improve the vehicle's design. The result is shown below in Figure 4.9. It can be seen that the max and min yaw angles are very close to the required value of 20 degrees.

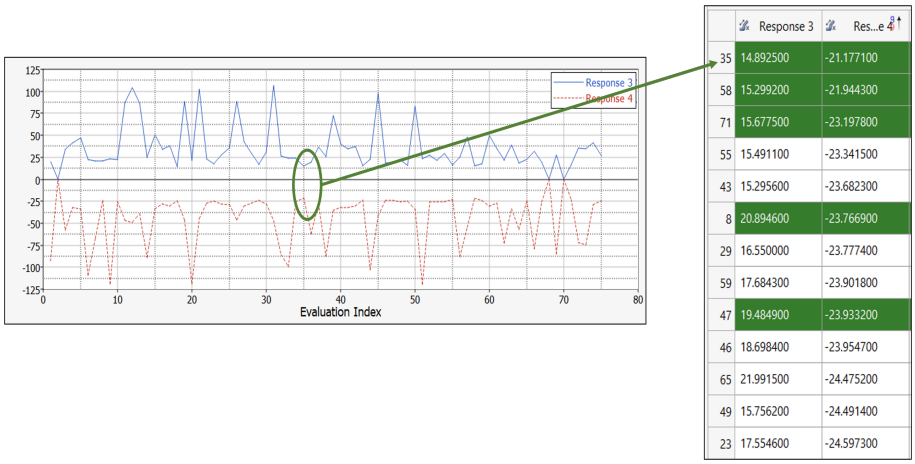


Figure 4.10: *Evaluation plot (75 iterations) and iteration showing the improved design yaw angles*

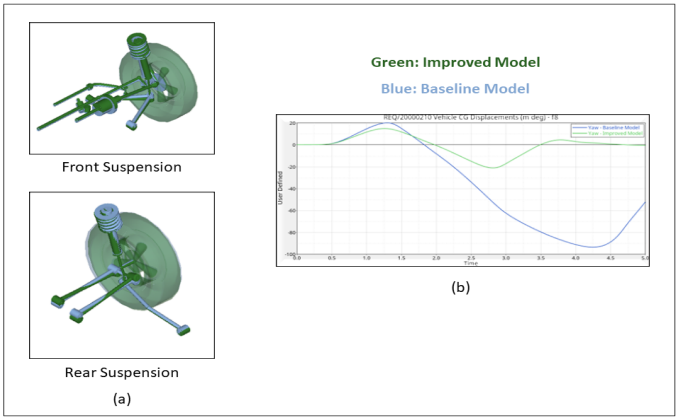


Figure 4.11: *(a) Suspension change from baseline to improved model (b) Yaw angle of baseline and improved model*

The animation for the baseline and improved models (rendered in Inspire Studio) is enclosed below.

4.4 References

[1] Practical Aspects of Multi-Body Simulation with HyperWorks, 2015, Altair University – Academic Program.

[2] MotionSolve - User Guide, Reference Guide and Tutorials, Online Help Manual, available on https://2021.help.altair.com/2021/hwsolvers/ms/topics/getting_started/motionsolve_main_overview.htm.

[3] MotionView - Tutorials and Vehicle Modeling, Online Help Manual, available on https://2021.help.altair.com/2021/hwdesktop/mv/index.htm.

[4] HyperStudy, Online Help Manual, (V2022).

[5] https://en.wikipedia.org/wiki/Moose_test.

5

SECTION

Appendix A - Altair® Products Installation

In this chapter you will find useful information to properly install some of the Altair Products needed to run all the models of this e-Book. Specifically you will need to install the following:

- Activate™ and MotionView/Solve™ for calculations and visualizations
- HyperView™ for 3D visualizations (post-processors).
- RTV Simulator™ for running 3D animations with FMI exported models of Activate (co-processing).

REMARK: If you are a student, you can request a student licence via this link: http://altairuniversity.com/free-altair-student-edition/. Then you may follow this video for the guidance and skip the rest of the installation section until the Visual Studio part.

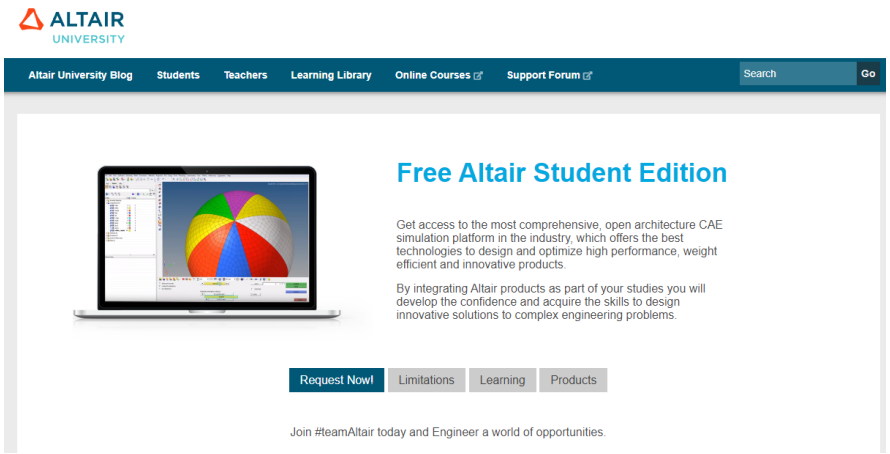


Figure 5.1: *Altair Student License*

For non-student users, please follow below installation steps:

1. Go on Altair Connect at https://connect.altair.com/ and Sign up to create an account (Figure A-1).

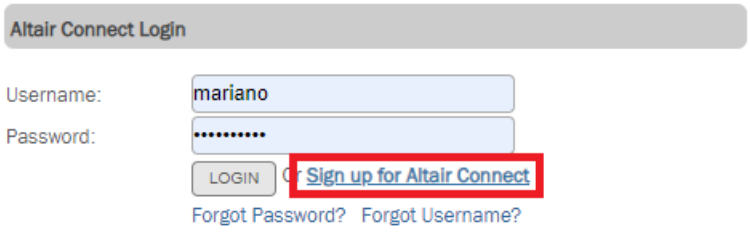


Figure 5.2: *Altair Connect*

2. Once approved, go under the tab Downloads and click on HyperWorks (Figure A-2).

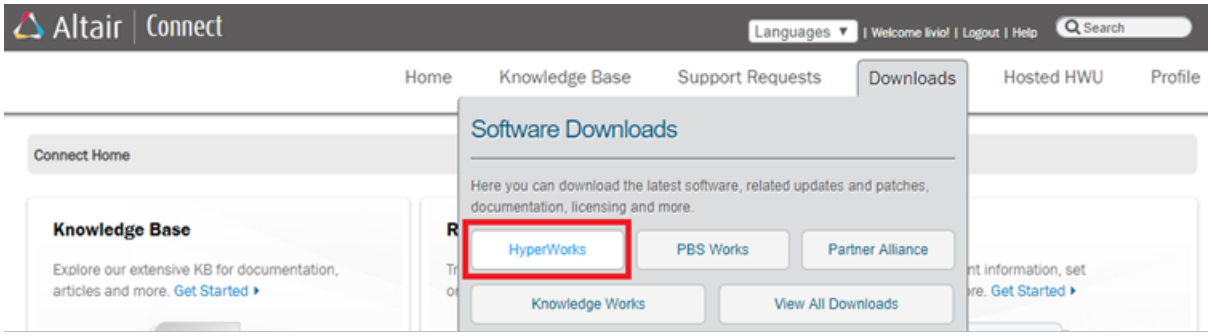


Figure 5.3: *Download Panel*

3. Choose the settings as in Figure A-3.

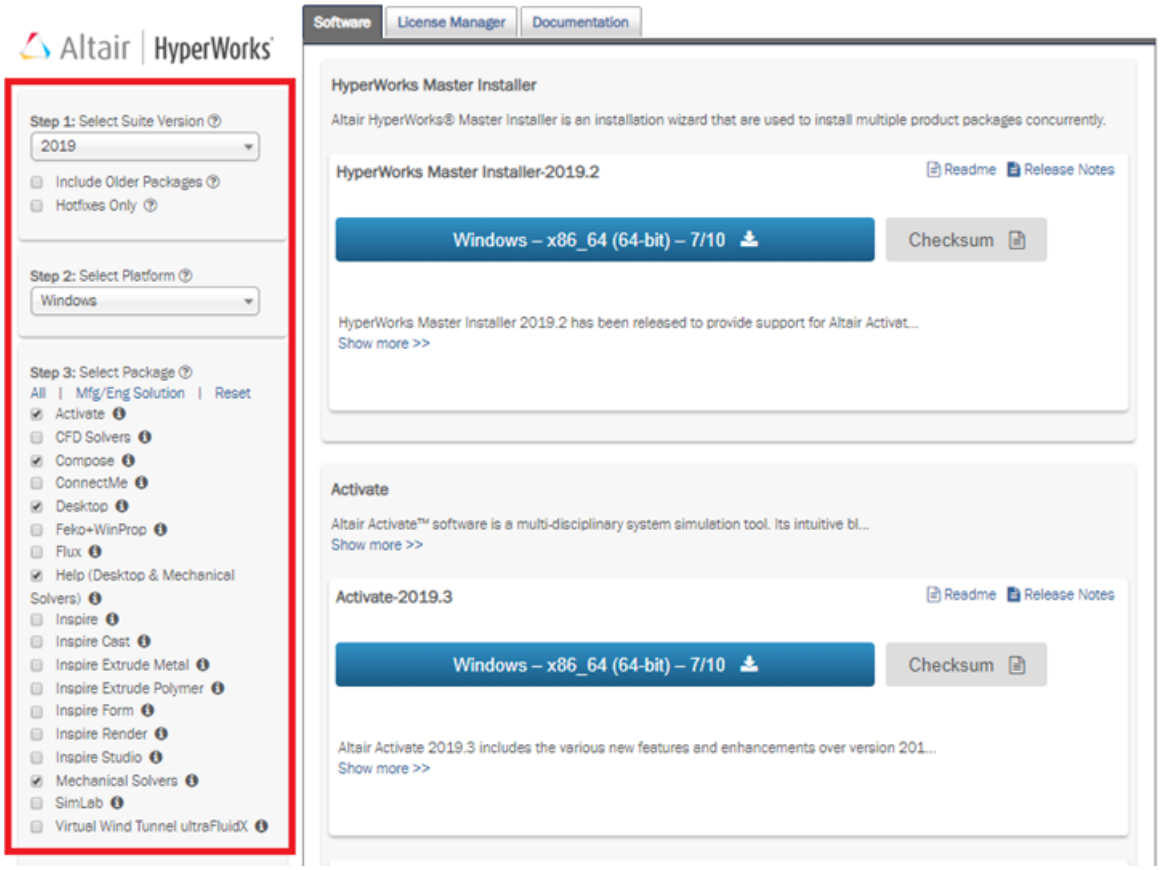


Figure 5.4: *HyperWorks Download Panel*

4. Download and place all the *.exe files in the same folder and run the master installer.

Once the installation ends, you need to rename the license as altair-lic.dat and place it in the security folder under the installation folders of HyperWorks as shown in Figure 1.4.

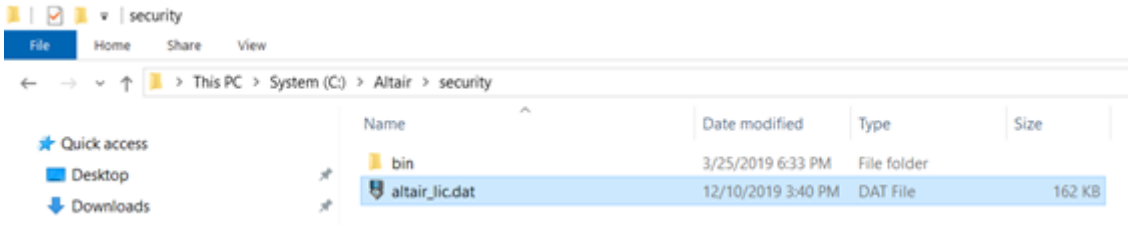


Figure 5.5: *License location*

5.1 Installation of Visual Studio™ C++ Compiler

To enable Activate to use Modelica™ components and export FMUs, you need to install the Visual Studio (VS) C++ compiler. Please follow the below instructions to install the C++ compiler tool:

1. Visit https://visualstudio.microsoft.com/downloads/
2. Download the free version available under the 'Community' version of Visual Studio.
3. Ensure that the Internet is connected because the C++ compiler automatically downloads the respective files.
4. Run the downloaded file.
5. In the pop-up, select the 'Desktop Development with C++' check box under 'Workloads' Menu.
6. Select (unless it is selected already) 'Windows 10 SDK' present at the top of the 'Optional' section list under 'Installation Details'. Select 'Install' button.
7. Once the Installation is done, now the Activate tool is ready for use.

SECTION 6

Appendix B - Activate Quick Start

Please note that this chapter is a gentle introduction to Activate. Readers are encouraged to check the Learn Basics of System Modeling and Control Systems with Altair Activate e-Book as a detailed resource.

Altair Activate^{TM} is a tool for system modeling. It supports both block-based and script-based languages.

1. Activate native files include the ".scm" and ".scb" extensions (for models and blocks respectively). 2. The general User Interface of the Activate can be seen in the image below.

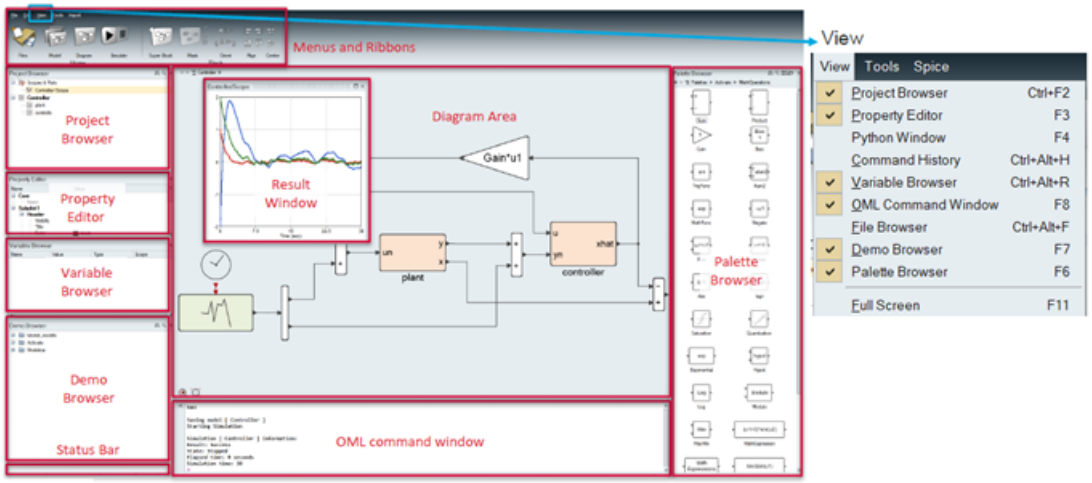


Figure 6.1: *Activate GUI*

3. When opening Activate, on the top left corner you will find some menus. Select the 'View' menu first to customize the user interface by activating/deactivating the different panels.

6.1 View Menu Options

1. The Project Browser panel gives you a hierarchical structure of the model being developed. Double-clicking on a level allows you to view that level in the diagram area.

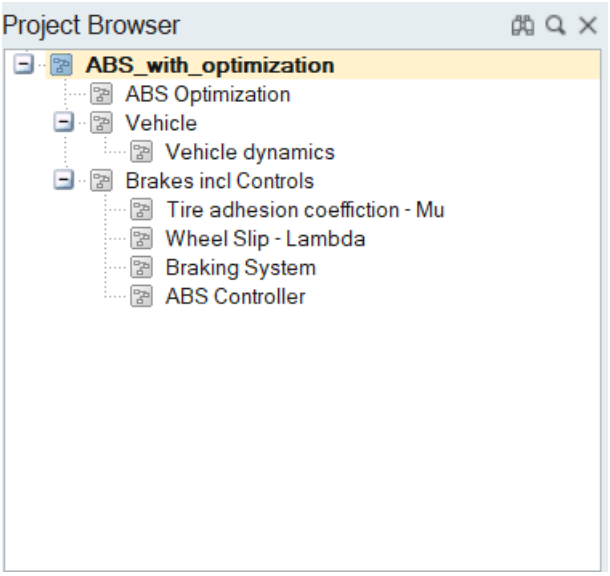


Figure 6.2: *Project Browser*

2. The Property Editor option gives you the ability to modify the appearance of a block (Figure 2.3).

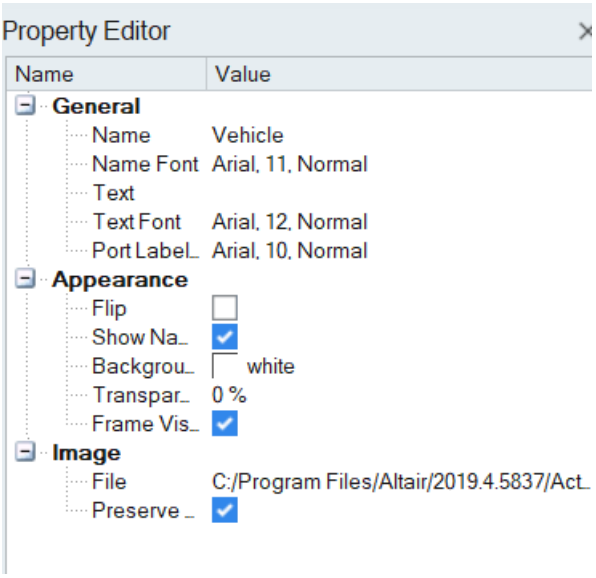


Figure 6.3: *Property Editor*

3. The Variable browser gives out the value of the variables exported in “Base” (that is the workspace), if any, at the end of the simulation.
4. The OML command window shows the execution status of the simulation. OML is the new powerful, open-source scripting language for Activate. OML offers access to extensive mathematical libraries such as the Control Toolbox.
5. The File browser allows for browsing folder contents, adding new files and deleting existing files. The displayed list can be filtered by file types, or by matching a pattern.
6. The Demo Browser contains some built-in example models that you can open and run to understand the working of the different blocks better.
7. The Palette Browser contains the built-in blocks from Activate, Modelica, Fluidon, and Hyperspice

which can directly be dragged and dropped into the diagram area for use. You can also generate and add here custom libraries (a combination of blocks).
8. The File menu lets you open, close, save files, seek help and manage libraries.
9. The Edit menu lets to undo and redo actions, or cut, copy and paste model components.
10. The Tool menu helps, among the options, to integrate OML codes (0D), export FMU, generate C-code and run optimizations (if needed).

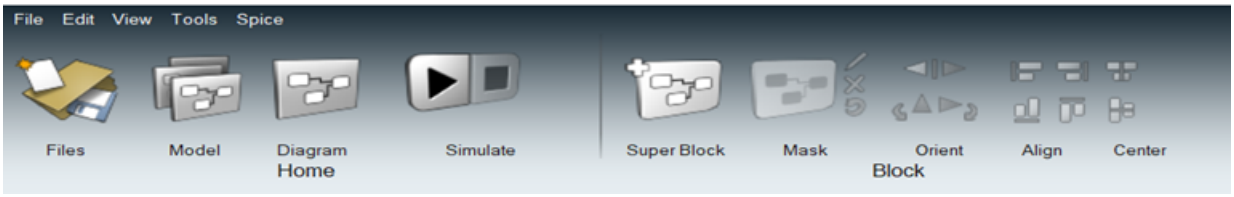


Figure 6.4: *Menus tabs*

6.2 Home and Block Icons

The icons under Home and Block (previous figure) provide some more features for user convenience like:

1. The 'Files' button allows you to open a new file or save the present file.
2. The 'Simulate' button provides the user the option to start, pause or stop a simulation for a model. If you take your mouse next to the Simulate area, a box-like icon appears next to the stop sign. Pressing this box icon lets you define the simulation parameters like the Final time.

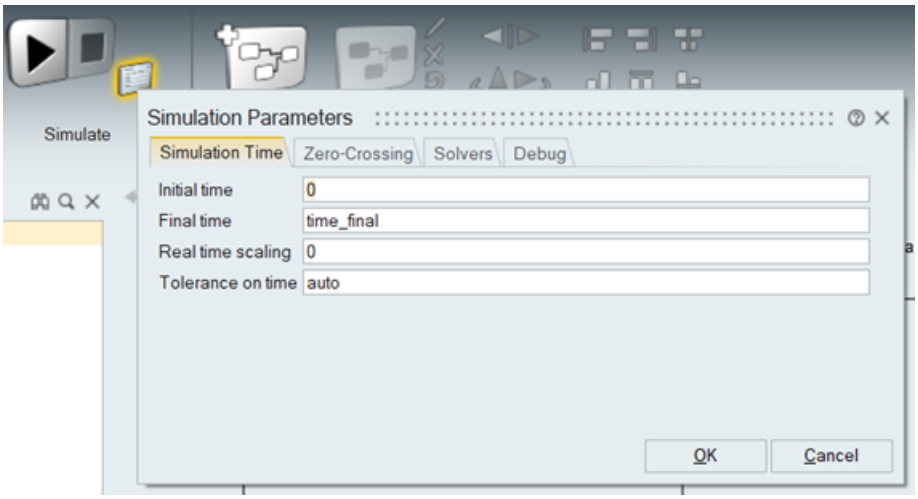


Figure 6.5: *Simulation Parameters*

3. Simulations generally require some input variables to execute and produce relevant output/results. Activate models may be parameterized with design variables for easy/powerful design changes and as a basis for optimization. Activate gives you two options to initialize a variable based on its scope:
* Model Initialization and Diagram are OML scripts used to define variables, and functions, to be used at a different level of the model according to the scoping rules (see below figure).
* Model Initialization: The variables defined in initialization are valid for the whole model.

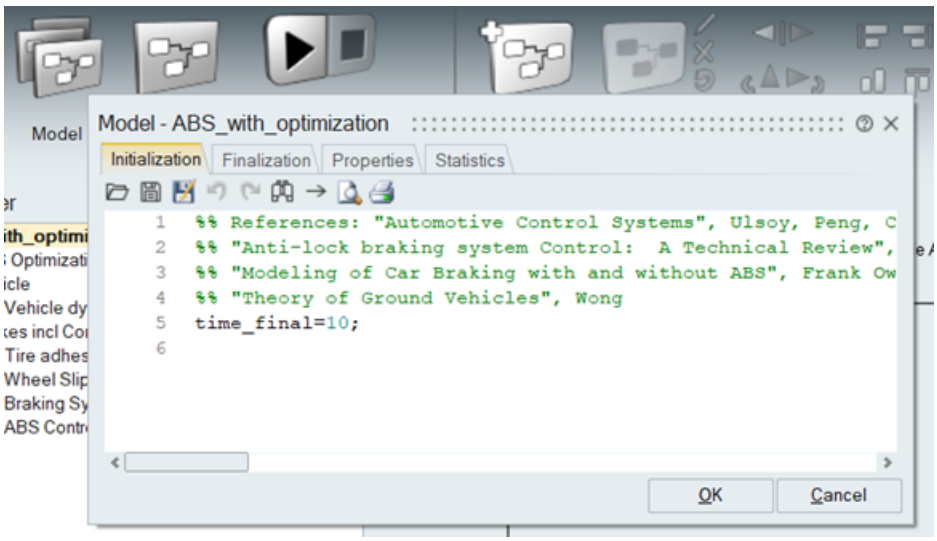


Figure 6.6: *Model Initialization*

* Diagram Context: The variables defined in Diagram (Context) are valid at the model view level and lower ones. Go To: Diagram -> Available Variables to check the variable available at that specific model view level and lower.

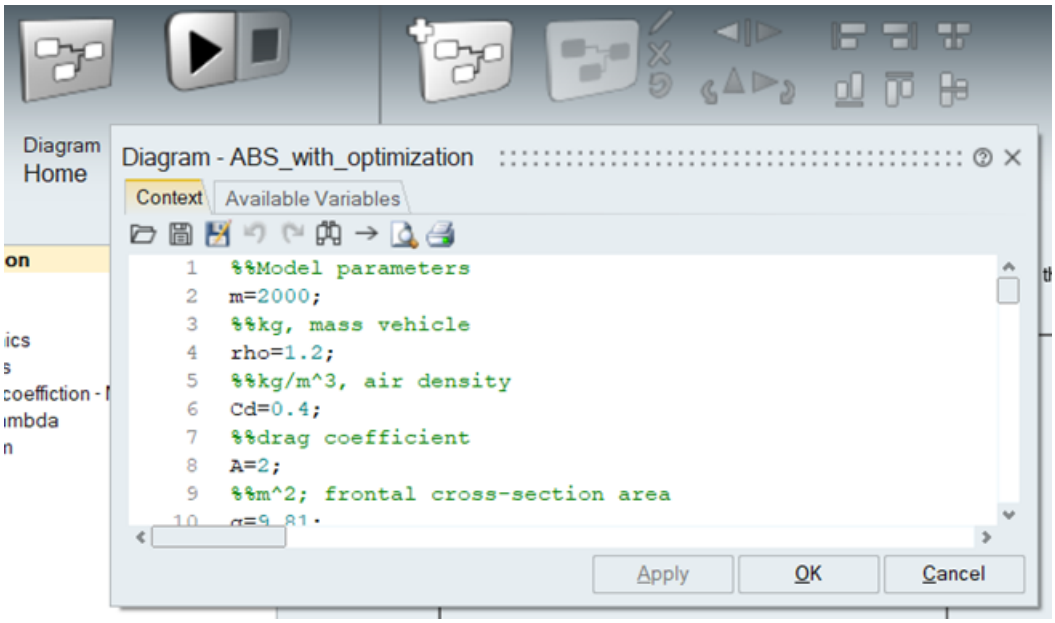


Figure 6.7: *Diagram/Context Initialization*

* Hierarchically, the definition in Model->Initialization is above the Diagram->Context definition in scope. So, the Context definition inherits the value of variables from the Initialization definition, but it can override these values locally.
* Some variables can be defined locally for the respective blocks by double-clicking on them and then entering the values you need.
4. A Super Block is a group of blocks combined into a child diagram of the parent model. It helps to organize the model better. To create a Super Block, you need first to select the blocks that you want to group. You can:
* Left mouse + drag window from left to right (solid line for the box). It selects anything within the box and touching.
* Left mouse + drag window from right to left (dashed line for the box). It selects blocks and links only fully within the box.
Then you have 2 options to create the Super Block, and you can:

* Click on the Super Block icon or do right-click with your mouse and click on Super Block.

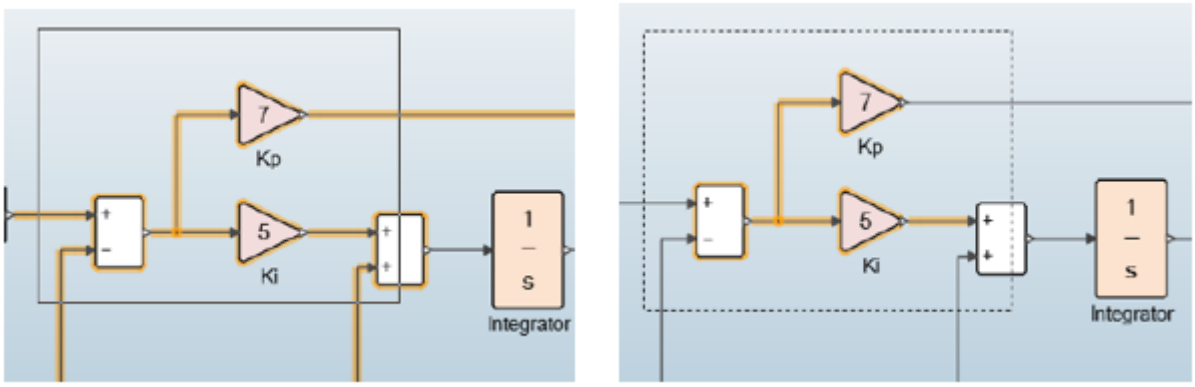


Figure 6.8: *The two ways to select blocks for Super Block Creation (L & R)*

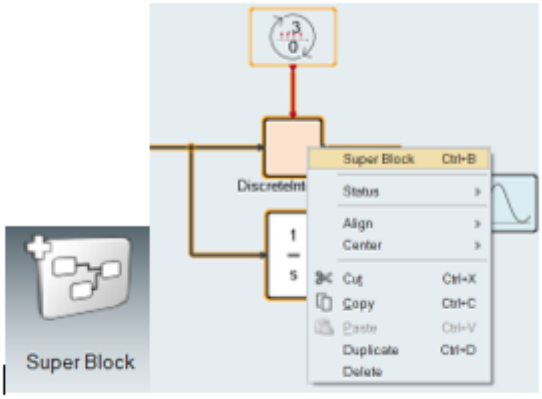


Figure 6.9: *Super Block generation: Super Block icon or right click*

5. The Orient, Align, and Centre buttons help you to adjust the blocks that are already present in the diagram area.
6. Adding ports to the contents of the Super Block will add ports to the Superblock itself. Ports can be added by dragging input or output ports from 'Ports' in the Activate Library of the Palette Browser window.
7. Click on a block/super block/port and press the F2 key on the keyboard to rename it. You can also select the block and edit its name from the property editor. Property Editor also lets you change the background color or orientation of a block. Another way to edit a block is by right-clicking on it to get a list of edit options. You can also choose to show/hide the name set for the block in the property editor.
8. Images can also be set for a block/ Super Block. This can be done by:
* Select Block -> Property Editor -> Image -> File -> Browse (folder) icon -> Select image saved in your system.
* Right click on the block -> Image -> Set -> Choose image saved in system.
9. Right-clicking in an empty area of the diagram gives a set of options to choose.

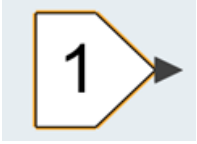


Figure 6.10: *A port block*

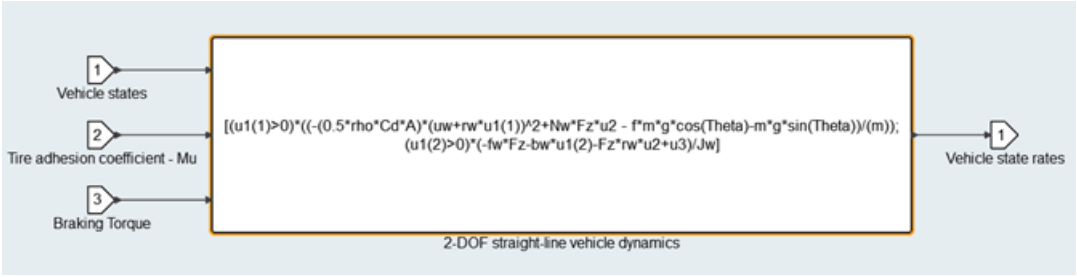


Figure 6.11: *Ports inside a Super Block*

6.3 Results & Scope

The results from a simulation can be viewed by using blocks scope, display, and Anim2D blocks in the Signal Viewers section of Activate Library in Palette Browser. Scope are blocks that you can include in a model for plotting and monitoring simulation results. The scope blocks can be connected to the output signal of the model.
A variety of predefined scope blocks are available in the Palette Browser from Palettes -> Activate -> Signal Viewers (see below Figure).

With scope blocks, you can plot signals versus simulation time or other signals, such as:

1. Scope: The Scope block displays its input with respect to simulation time. The Scope block allows for multiple inputs.
2. ScopeXY: The ScopeXY block displays the evolution of the two regular input signals by drawing the second input as a function of the first at instants of events on the event input port.
3. Display: This block displays the current value of the input signal in a given C string format and is updated at specified intervals in real-time seconds.
4. Anim2D: The animation block is used to animate different types of graphical objects in a figure (on a canvas).

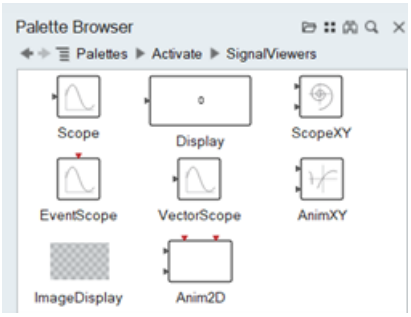


Figure 6.12: *Scope Palette includes different Scope options*

6.4 Co-simulation with Activate and MotionSolve

Activate is a system integration platform able to manage in the same simulation different solvers if needed. In this e-Book, you'll experience the co-simulation between Activate and MotionSolve which is the Altair Multi-Body Solver.

To enable this capability, we need "to inform" Activate about the installation location of Motion.

1. Go to the tab File and click Preferences. 2. The Preferences panel will show up as in the below figure. Check that under Cosimulation with MotionSolve you have a setting similar to the one reported in the red rectangle.

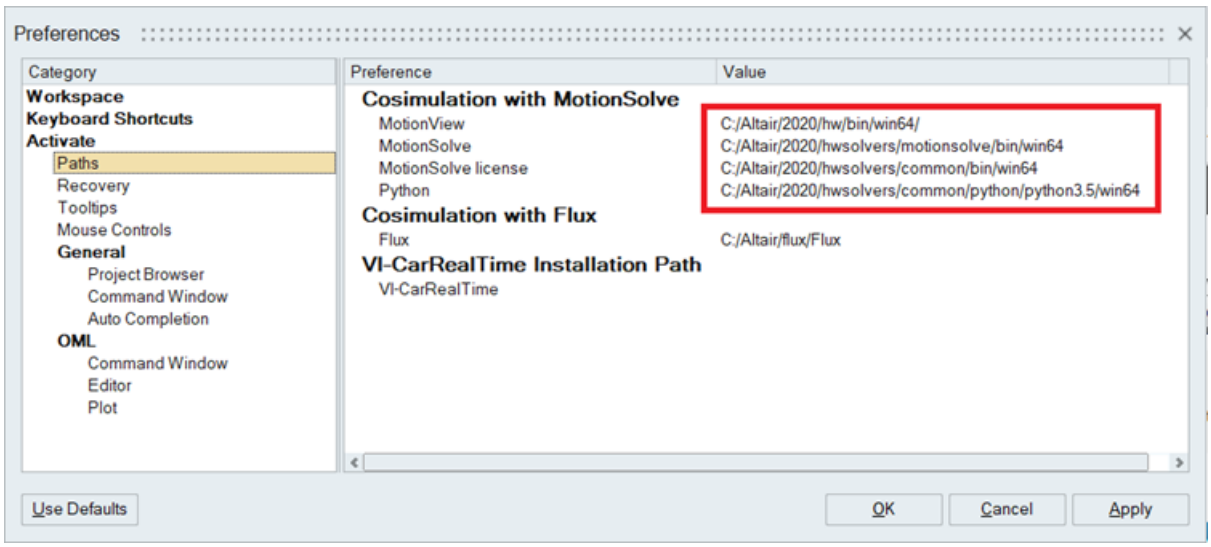


Figure 6.13: *Preferences panel*

SECTION 7

Appendix C - RTV Simulator™ Quick Start

Altair's Real-Time Visualizer™ (RTV) is a tool used for visualizing the system model behaviors and outputs in a 3-D simulation environment. It uses FMU (Functional Mock-Up) interface to integrate the Activate generated models into the simulation engine.

To use the RTV Simulator with the FMU files, please follow these steps:

1. Download the Activate generated FMU Models from this link.
2. Copy the 3D Simulation Files folder under the $C:\backslash$ folder.
3. Copy the $RTV\backslash$0.6.1 folder to the same file path.
4. The file hierarchy should be as in the image below:

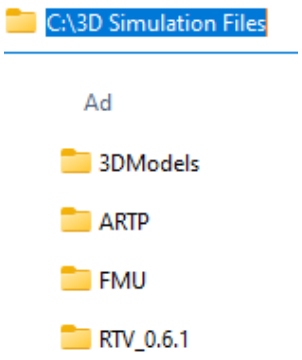


Figure 7.1: *RTV Simulator Folder and Files*

5. Run AltairRTV.exe in $C:\backslash 3DSimulationFiles\backslash RTV\backslash$0.6.1$\backslash WindowsNoEditor$. This should open the RTV Simulator.

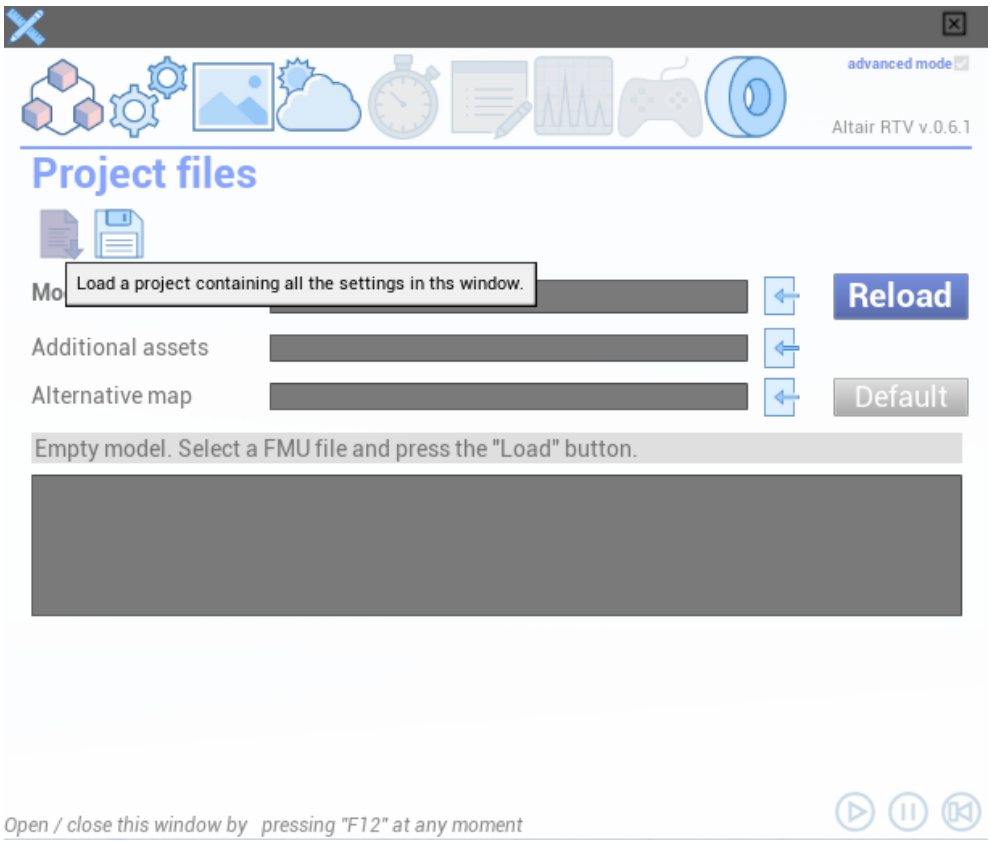


Figure 7.2: *RTV Simulator - Default View*

6. Click the Load project button and load the artp file.

7. Click the View tab from the menu that opens in the upper left corner and select the Obj. orbit camera.

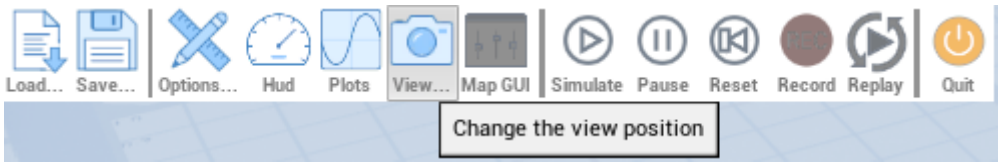


Figure 7.3: *RTV - Top Ribbon and Camera Selection*

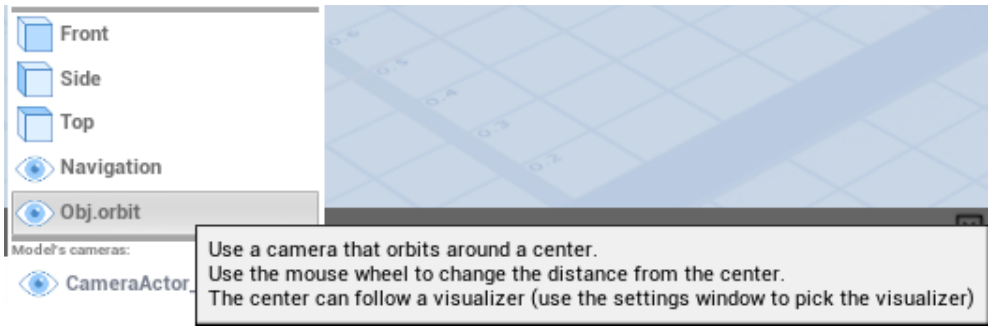


Figure 7.4: *Activate GUI*

8. Run the simulation via the toolbar's start button.

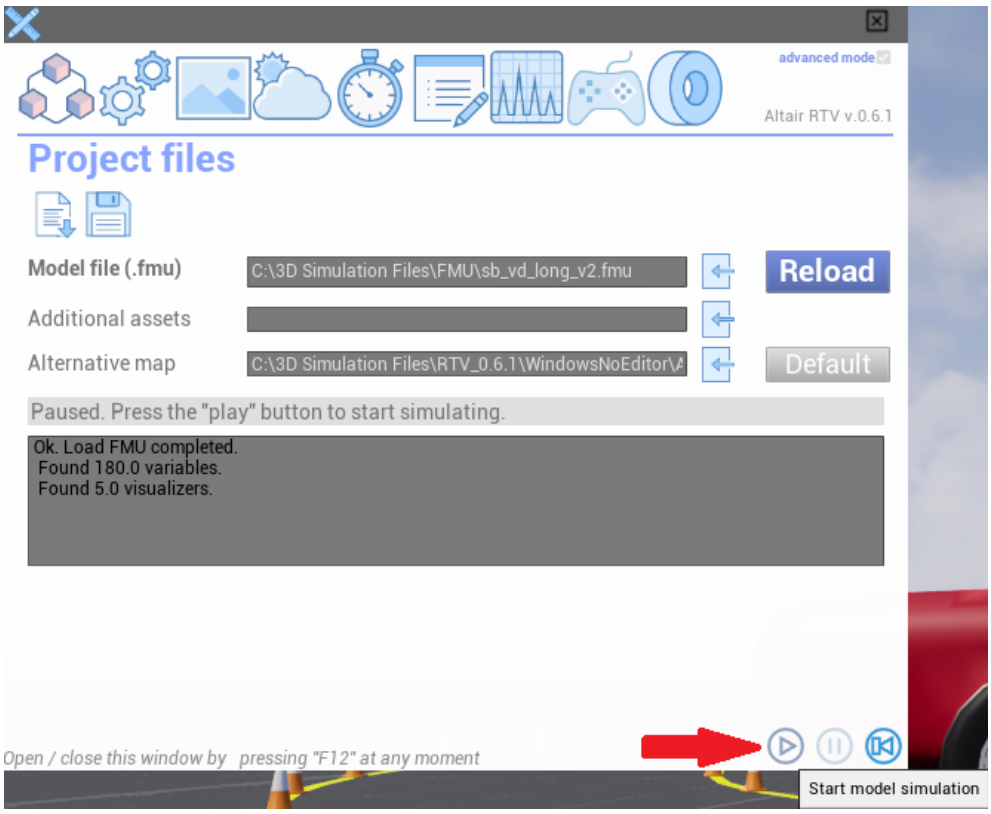


Figure 7.5: *Activate GUI*

7.1 Navigation and Controls in RTV

1. To be added later.

8

SECTION

Appendix D - MotionVIEW™ Quick Start

MotionView™ is the Altair tool dedicated to the preparation of Multi-Body models. When we run the model, the Altair Multi-Body solver, MotionSolve™, is called to solve the mechanical dynamics. MotionView files use ".mdl" format. When we run the simulation starting from the ".mdl" file a ".xml" file is generated and used as input to MotionSolve. For details on creating and working with .mdl files, please check the reference [1].

The main 3-D simulation file used in Chapter-4 of this E-Book is the file named "Vehicle_Study_3.hstx". It is created by HyperStudy, a zipped archive that contains all the files required. It can be opened in HyperStudy (v2021 or higher) As the file is opened in HyperStudy, a folder by the name of the file is created automatically. That folder will include the .mdl file – which can be used as in MotionView.

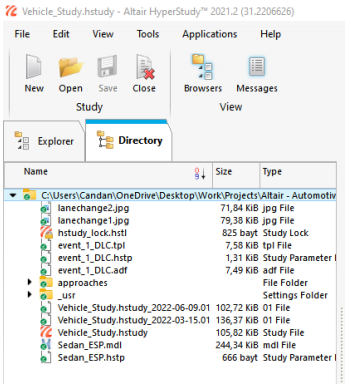


Figure 8.1: *.mdl file for MotionVIEW simulation*

While the results of baseline, DOE, and Optimization are already populated in HyperStudy (as tables), the output files themselves are not available in the archive. But one can run the studies again to regenerate all output files. Please check reference [2] for more information on HyperStudy files.

To open the .mdl files, users need to:

1. Open the .hstx file in HyperStudy. All results will be visible in tables.

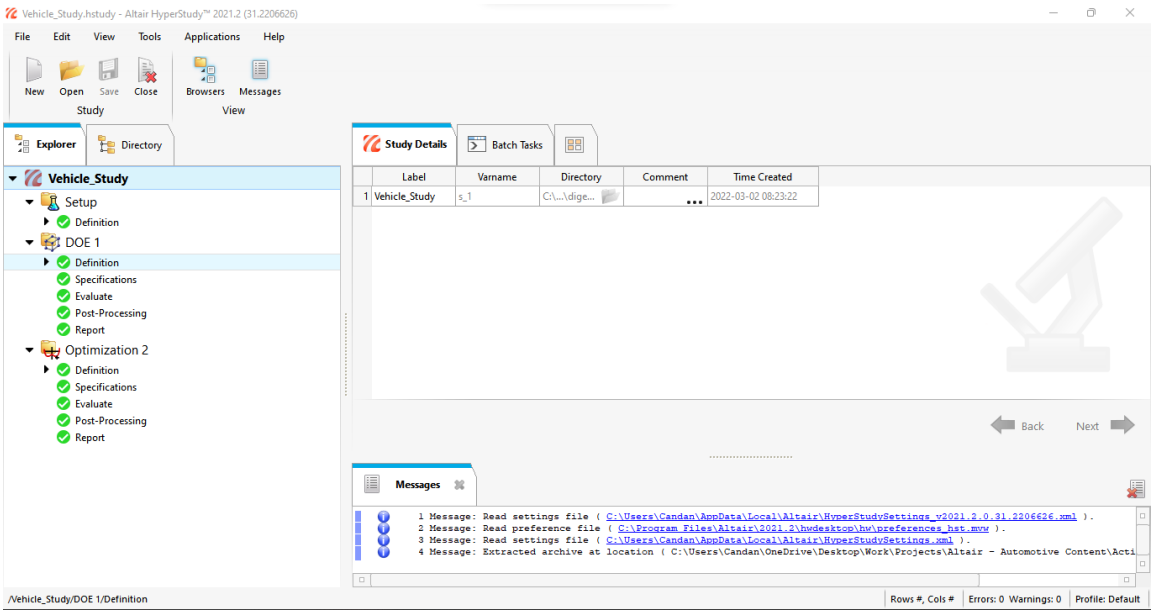


Figure 8.2: *Result Tables in HyperStudy*

2. Users can regenerate all output files (DOE and Optimization) by rerunning the studies (Evaluate->Evaluate Tasks – available under DOE 1 & Optimization 2).

3. MotionView files "Sedan_ESP.mdl and "event_1_DLC.adf", which are used in the Chapter-4, are available in the folder "Vehicle_Study_2", which gets created once the hstx file is opened in HyperStudy. Users can see how the model is set up to run a 'Double Lane Change' event.

To run the model from MotionView, users need to follow the instructions given below:

1. Open the MDL in MotionView

2. Activate 'DoubleLaneChange 0' ('event_0') by right-clicking on the event in the browser – this is the built-in double-lane-change event. It has road graphics and everything else.

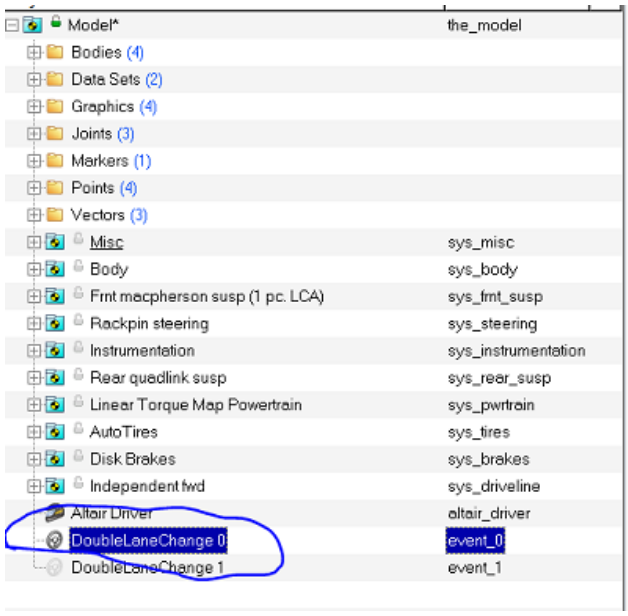


Figure 8.3: *Double Lane Change Event Activation for 3-D Simulation*

3. Open the Event Editor in the Panel and click on Apply.
   The road graphics should show up along with the cones as in the below image.

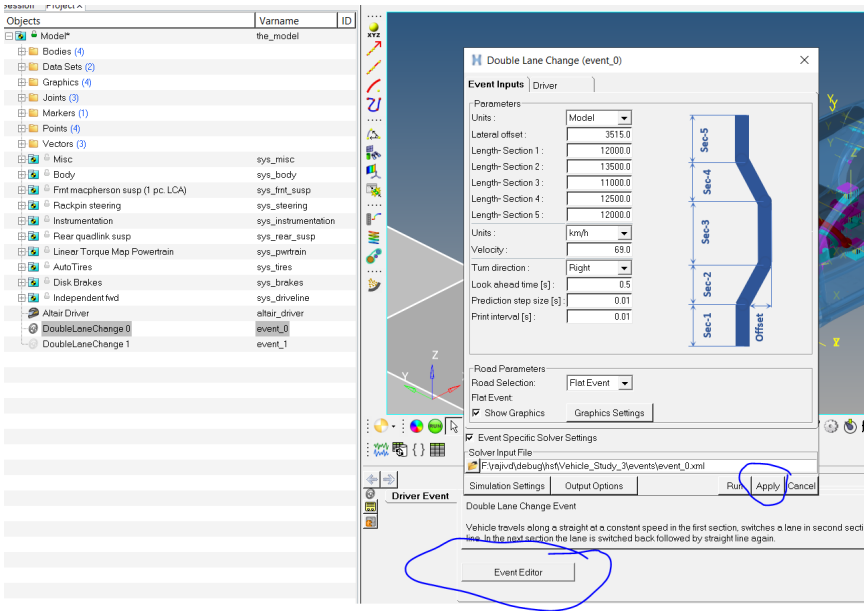


Figure 8.4: *Changing Simulation Parameters in Event Editor*

4. You can change the 'Velocity' as required and click 'Run' in the 'Event Editor' – the model should run and the MotionSOLVE Console should open as below.

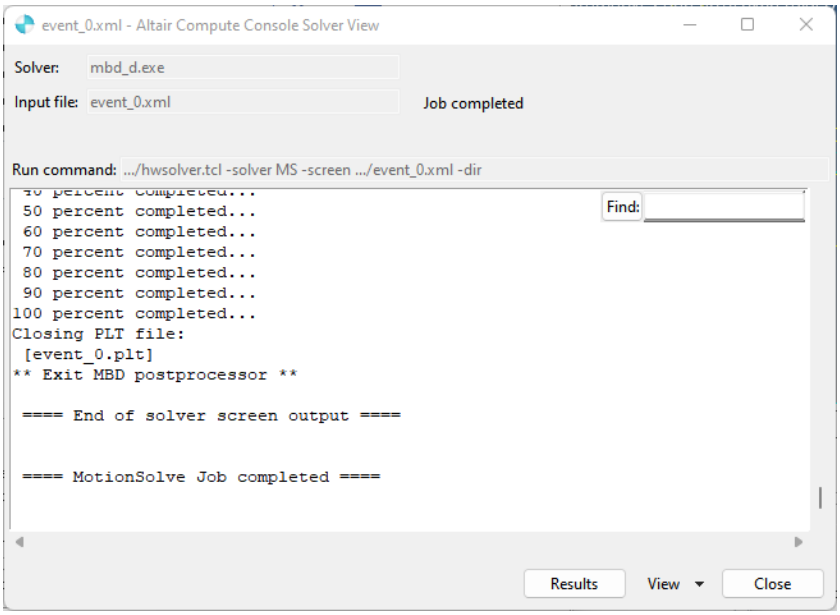


Figure 8.5: *Console Solver View for MotionVIEW Simulation*

5. Once the simulation completes the Results button will get active as well. Hit this button to open the results viewer in Hyperworks. You may use the Run, Rewind, etc. buttons to animate the simulation.

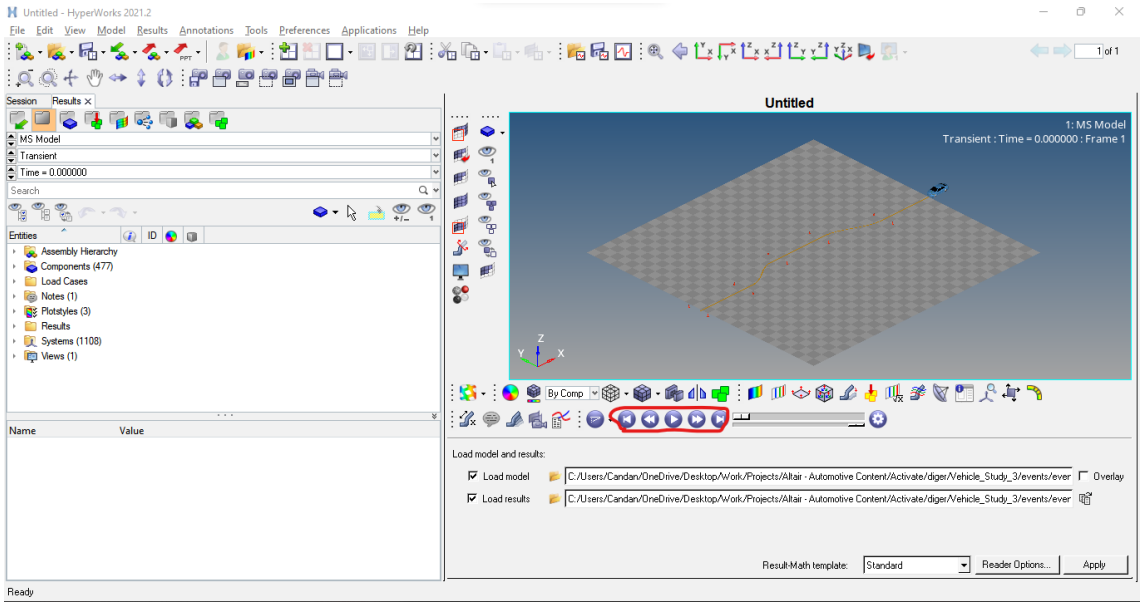


Figure 8.6: *Hyperworks Simulation Player*

All these steps are not required when using HyperStudy; this is mainly because HyperStudy is set up to interact with an ADF file (and not with the above built-in event) to modify the velocity, and we ask HyperStudy to link the ADF file to MDL for all runs.

Readers are encouraged to read the references [3], and [4] to learn more about using HyperStudy with MotionView.

Please note that if you activate event_0 and save the model, HyperStudy runs might fail afterward. So, if you need this file with event_0, please save it by a different name.

8.1 References

[1] MotionView - Tutorials and Vehicle Modeling, Online Help Manual, available on https://2021.help.altair.com/2021/hwdesktop/mv/index.htm.

[2] HyperStudy, Online Help Manual, (V2022) available on https://www.altair.com/resource/altair-hyperstudy-new-online-help.

[3] MV-3000: DOE using MotionView – HyperStudy available on https://2021.help.altair.com/2021.1/hwdesktop/mv/topics/solvers/ms/doe_using_motionview_hyperstudy_intro_t.htm.

[4] MV-3010: Optimization using MotionView - HyperStudy available on https://2021.help.altair.com/2021.2/hwdesktop/mv/topics/solvers/ms/optimization_using_mv_hs_intro_r.htm.