IPG AUTOMOTIVE

Taking you to the next level

IPG Documentation

CarMaker®

Quick Start Guide Version 5.1.4

SimulationSolutions • TestSystems • EngineeringServices

The information in this document is furnished for informational use only, may be revised from time to time, and should not be construed as a commitment by IPG Automotive GmbH. IPG Automotive GmbH assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

This document contains proprietary and copyrighted information and may not be copied, reproduced, translated, or reduced to any electronic medium without prior consent, in writing, from IPG Automotive GmbH.

© 1999 - 2017 by IPG Automotive GmbH – www.ipg-automotive.com
All rights reserved.

FailSafeTester, IPGCar, IPGControl, IPGDriver, IPGEngine, IPGGraph, IPGKinematics, IPGLock, IPGMotorcycle, IPGMovie, IPGRoad, IPGRoaddata, IPGTire, IPGTrailer, IPGTruck, RealtimeMaker, Xpack4 are trademarks of IPG Automotive GmbH.

CarMaker, TruckMaker, MotorcycleMaker, MESA VERDE are
registered trademarks of IPG Automotive GmbH.

All other product names are trademarks of their respective companies.

Contents

3

Contents 3

1 About this Quick Start Guide 7

2 Installation of CarMaker 8

3 Starting CarMaker 9

4 Performing your First Simulation within 5 Minutes 11

4.1 Loading a predefined environment parameterization 11

4.2 Starting and Stopping the Simulation 13

4.3 Interactive Viewing of the Current Simulation 14

4.4 IPGMovie 18

4.5 Result Management 20

4.5.1 IPGControl 20

4.5.2 Other Capabilities 24

5 Extending the Test Scenario Definition 26

5.1 Exploring the IPGRoad Features 26

5.1.1 Opening the Road GUI 27

5.1.2 Checking the Road Geometry 29

5.1.3 Interaction of IPGRoad with the Maneuver Definition 29

5.1.4 Creating a Closed 3D Road 32
5.1.5 Modifying the Width and Coefficient of Friction 34
5.1.6 Inserting Bumps, Markers and Movie Objects 38
5.2 Defining the Maneuver 40
5.2.1 Opening the Maneuver GUI 40
5.2.2 Limiting a Maneuver Step by Distance 41
5.2.3 Adding an Open Loop Maneuver Step 43
5.2.4 Adding a Closed Loop Maneuver Step with IPGDriver 44
5.2.5 Other Options for the Definition of the Maneuver Step 46
5.3 Advanced Features 48
5.3.1 Using a Digitalized Track 48
5.3.2 Driving Lane 50
5.3.3 Parametrizing IPGDriver 51
5.3.4 Comparing two Simulations with IPGMovie 53

6 Changing the Vehicle and Tire Data 54
6.1 Using another Vehicle Parameterization 54
6.2 Using another Trailer Parameterization 55
6.3 Using another Tire Parameterization 56
6.4 Saving the new Data Sets within the TestRun 57
6.5 What is a Data Set? 58

7 Parametrizing the Vehicle and the Tires 59
7.1 Creating a new Vehicle Data Set 59
7.1.1 Save as a New Data Set 60
7.1.2 Using the Vehicle Generator 61
7.2 Parameterizing the Vehicle Model 62
7.2.1 Vehicle Body 62
7.2.2 Bodies 63
7.2.3 Engine 64
7.2.4 Suspensions 64
7.2.5 Steering System 70
7.2.6 Brake System 71
7.2.7 Powertrain 71
7.2.8 Aerodynamics 75
7.2.9 Sensors 75
7.2.10 Vehicle Control 76
7.2.11 Misc. 76

7.2.12 Using the Import Feature 77
7.3 Creating a new Tire Data Set 79

8 Building your TestRun from Scratch 81
8.1 Step 1: Open CarMaker and Create a TestRun 81
8.2 Step 2: Load the Vehicle 82
8.3 Step 3: Putting the Vehicle on a Road 83
8.4 Step 4: Controlling the Vehicle (Maneuvers) 83
8.5 Step 5: Saving the TestRun and Simulating 84
8.6 Final Results 85

9 Simulating with CarMaker and Simulink 86
9.1 Starting CarMaker with Simulink 86
9.2 First simulation with CarMaker for Simulink 88
9.3 Description of the CarMaker Environment in Simulink 90
9.4 Advantages and Disadvantages using CarMaker with Simulink 91

10 Features of the CarMaker for Simulink blockset 92
10.1 General Information 92
10.1.1 Block Properties 92
10.1.2 Modeling Principles 93
10.1.3 Purpose of the Sync_In and Sync_Out Ports 93
10.2 Utility Blocks 93
10.2.1 CarMaker GUI 93
10.3 CarMaker Dictionary Blocks 94
10.3.1 Read CM Dict 94
10.3.2 Write CM Dict 94
10.3.3 Define CM Dict 95
10.3.4 Dictionary Initialization 96

11 How to extend the Simulink Model 97
11.1 Overview of CarMaker's Simulink Structure 97
11.2 Example 99
11.2.1 Contents 99
11.2.2 Create a new Simulink Model 99
11.2.3 Prepare the extension 99
11.2.4 Connect Inputs 100
11.2.5 Demonstration Examples 103

12 Additional Information about CarMaker 104
12.1 What are the Differences between each CarMaker Version? 104
12.2 Typical Tests 104
12.3 Feature List 106

Chapter 1

About this Quick Start Guide

This document intends to give a first introduction to CarMaker by showing how to perform simulations with the CarMaker, either as a stand-alone application (e.g. you run it “as is“, without any other software), or as an embedded environment in Simulink.

It can be seen as a tutorial, providing you insight into the powerful world of virtual test driving with CarMaker. The CarMaker documentation is more comprehensive than this documentation as it contains every aspect of using, adapting and programming the CarMaker software package.

So always keep in mind that the world of CarMaker is much larger than this tutorial can show.

Please note that all paragraphs having this format contain instructions about what you have to do.

Enjoy it!

Chapter 2

Installation of CarMaker

To install CarMaker, please log in as user with administrative privileges and extract the downloaded installation pacckage (.zip archive). Open the executable file “ipg-install.exe” via double-click.

In case you received an installation CD, please insert it to the CD-ROM drive on your PC. If the AutoRun is activated, IPGInstall should start automatically. Otherwise open the executable file “ipg-install.exe” manually.

Once you have installed CarMaker, you need to contact the IPG License Team to get your licence:

- within your Windows Explorer, go to the folder ./tools-win32 of the installation package
- from here, start the tool wlicinfo.exe.

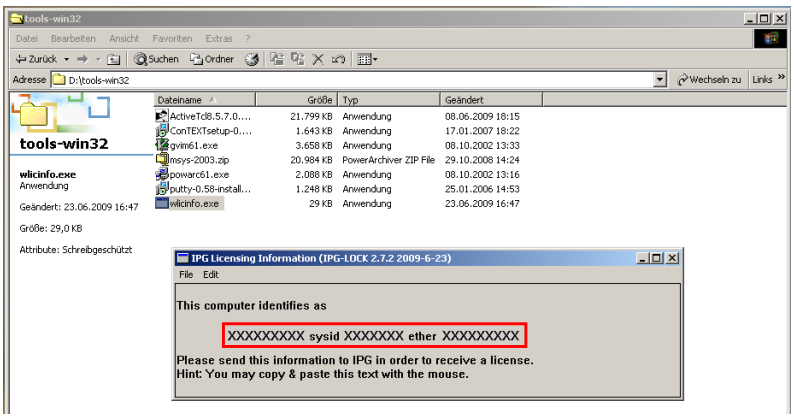


Figure 2.1: Executing the wlicinfo.exe and required output

- Please use the license key request form in the support area on our homepage (http://www.ipg-automotive.com/). Fill in the form and copy the output of wlicinfo.exe to the field “PC identification“.
- You will receive a valid license file from the IPG License Team.
- Save this license file under “<InstallationDirectory>\etc”.
- You can now work with the software.

Chapter 3

Starting CarMaker

To start CarMaker:
Click on the Windows “Start“ button > Programs > IPG > CarMaker Version No. > CarMaker.

For Linux users: open your console and enter “CM”.

Figure 3.1: Opening CarMaker using Windows

Once you have clicked on the final “CarMaker“ menu, the CarMaker software pops up:

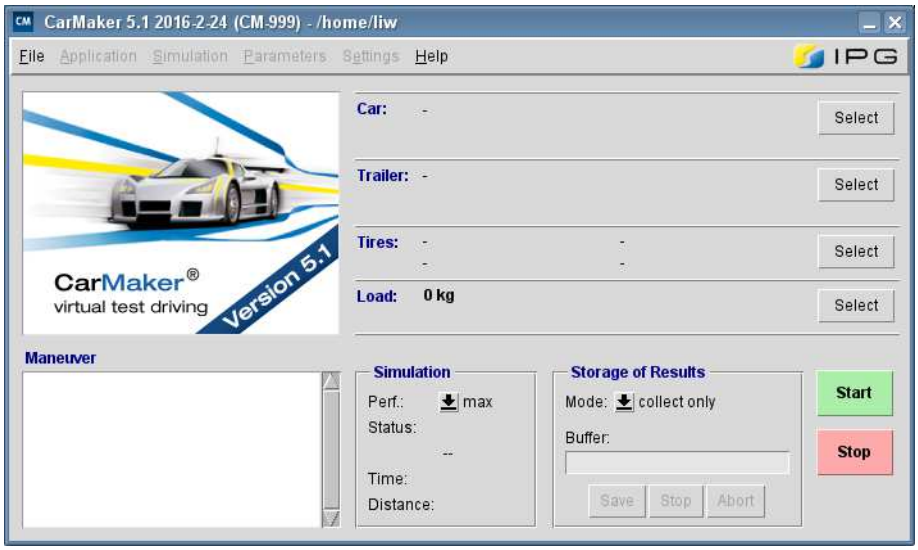


Figure 3.2: CarMaker main GUI

You will see the so called *CarMaker main GUI* (main Graphical User Interface), control center for CarMaker, through which all functionalities, windows, etc. can be accessed.

Next!

CarMaker is ready to work! Now, please read the entire section ’Performing your First Simulation within 5 Minutes’ on page 11 to know how to continue.

Chapter 4

Performing your First Simulation within 5 Minutes

4.1 Loading a predefined environment parameterization

To perform a simulation in CarMaker, it is just like on a real test drive: you need to define a vehicle, the tires, a driver, a test track and a maneuver the driver should complete.
In CarMaker there is a model for each of these requirements. The combination of these models forms a “TestRun“.

The standard installation of CarMaker includes many typical TestRuns containing all data required to describe each of these models. Thus, you can already work with it and see the main features of the program.

When you start CarMaker for the first time, you need to create a project folder which contains the TestRun data. This can easily be done selecting File > Project Folder > Create Project in the CarMaker main GUI.

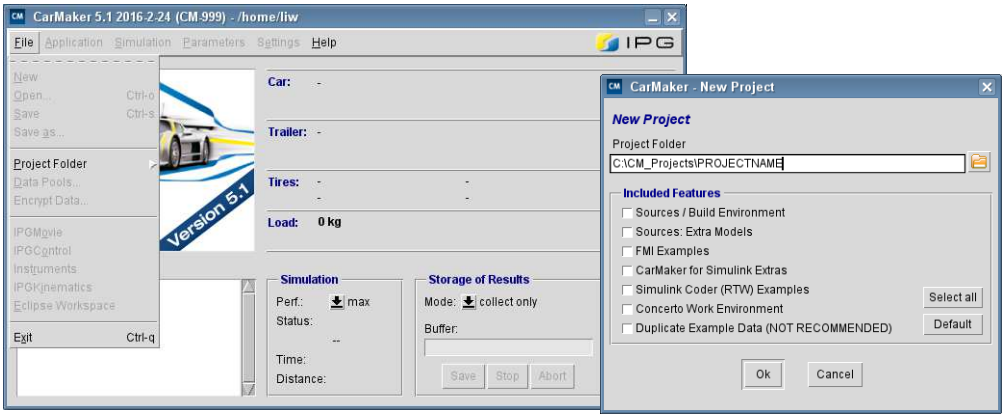


Figure 4.1: Creating a new project folder

Select the path where you want your project directory to be placed. The additional features available can be toggled on optionally. You will not need any of them in this session.

Now, you are able to load a first TestRun.

Load the TestRun “Hockenheim”:
Click in the CarMaker GUI on *File > Open, select the product data pool, then click on Examples > CarMakerFunctions> IPGRoad > Hockenheim.*

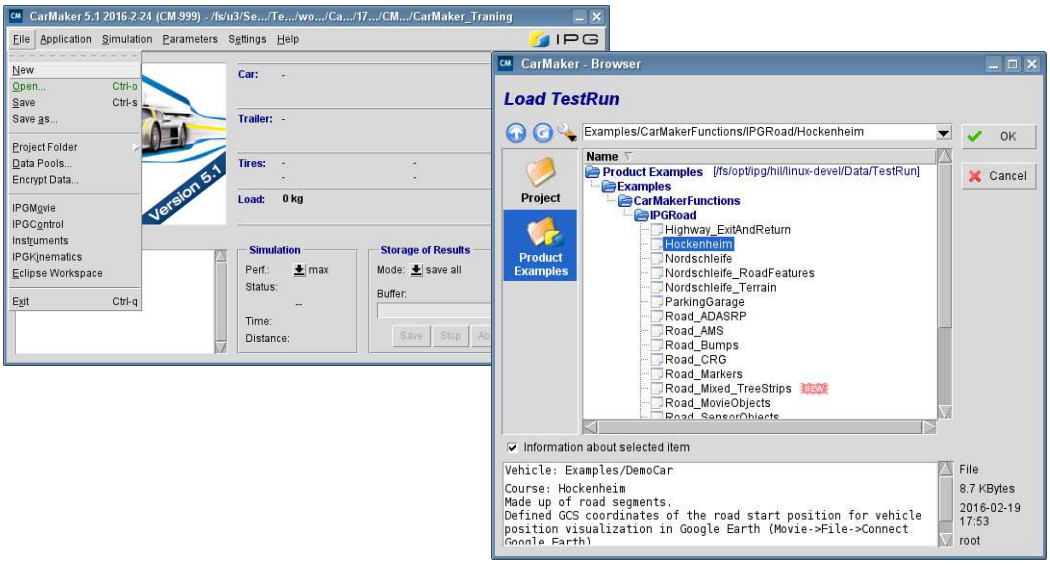


Figure 4.2: Loading the TestRun Hockenheim

As you can see, the GUI is now filled with the required data sets that will be used for the next simulation. The so called “TestRun“ is ready to be simulated:

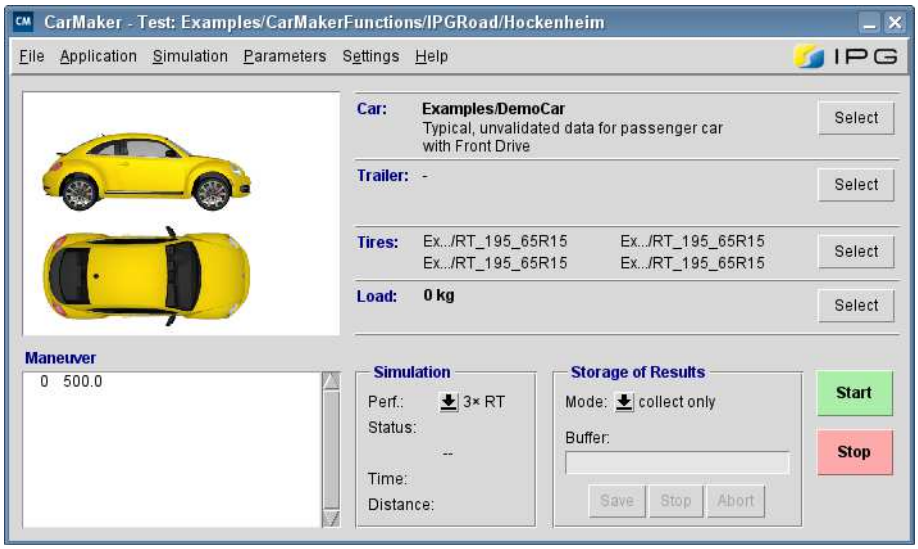


Figure 4.3: The CarMaker GUI contains the data sets for the TestRun

What is a TestRun?

CarMaker is a program based on fixed models (vehicle, suspension, tire,...) whose properties (e.g. values for the mass of each body, spring stiffness) can be varied.

This means the number of bodies and the DOF between them are already defined, you do not need to do it on your own! If you wish to extend the model, CM4SL is recommended: see the section 'Simulating with CarMaker and Simulink' on page 86.

Subsequently, each model has to be parameterized according to the environment, which means that a so called "data set" must be defined (or loaded) for each model. This includes selecting a vehicle, selecting or defining a road, selecting a driver type, defining the maneuver, etc. When all these choices and definitions are made, you have all the information required to parameterize and control the virtual vehicle environment (VVE).

These settings are stored in a file used by the VVE during a simulation. This file, which can be saved, loaded, edited, etc. is called a TestRun definition. Loading and executing such a TestRun definition results in the simulation of that specific test.

To sum up: a TestRun is a test scenario in which all parameters of your virtual environment (vehicle, driver, tires, ...) are defined.

4.2 Starting and Stopping the Simulation

In the section 4.1 on page 11, you have already loaded a TestRun: with this you provided CarMaker all data required to perform a simulation successfully.

Click on the green "Start" button of the CarMaker GUI.

As you can see, the simulation starts:

- in IPGMovie: you can see the animation movie of your current simulation
- in the CarMaker GUI: in the box "Simulation", you can see the time running, the distance covered by the car and the status of the simulation
- in the Instruments: you can see the displays operating, especially the tachometer

Note: If IPGMovie and Instruments have not popped up automatically with the start of CarMaker select "File > IPGMovie" and "File > Instruments"in the main GUI.

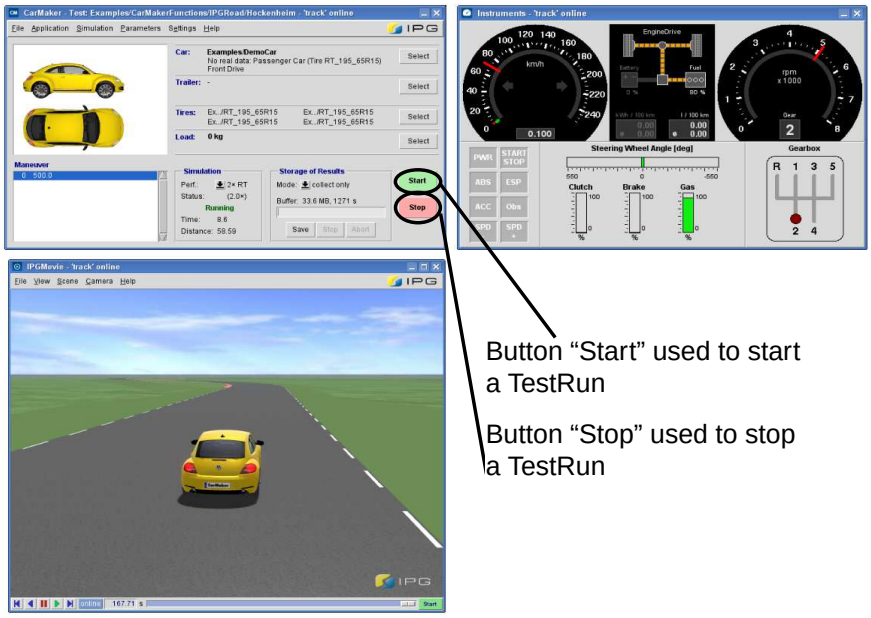


Figure 4.4: The Start and Stop buttons on the CarMaker GUI

Now click on the red “Stop“ button in the CarMaker GUI.

The IPGMovie animation stops, the Instruments come slowly to a stand still, and all other displays in the CarMaker GUI are stopped.

The “Stop“ button is used if you wish to abort a test run ahead of time. Otherwise, if the TestRun comes to its end, the simulation stops and comes back to an idle stand: you do not need to click on “Stop“.

By loading a TestRun and clicking “Start“ and “Stop“, you know the basics of the use of CarMaker! Now let us see what happens while a simulation is running.

Click again on the green “Start“ button in order to start the TestRun once again.

We will now have a look at the CarMaker GUI, IPGMovie, the Instruments and also IPG-Control while the simulation is running.

4.3 Interactive Viewing of the Current Simulation

IPGMovie

IPGMovie enables you to watch the animation of the currently running simulation.

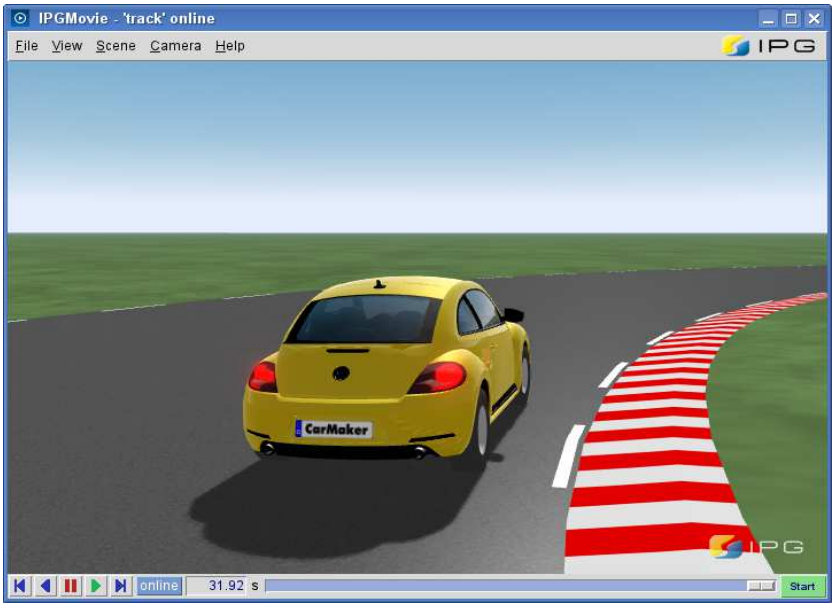


Figure 4.5: Online View of a simulation using IPGMovie

Of course, you can change the points of view and the background. But IPGMovie is much more than just an animation tool.

For instance, you can export the animation you are watching to a file in order to use it later for a presentation!

More detailed explanations about the use of IPGMovie are given in section 4.4 ’IPGMovie’ on page 18.

CarMaker GUI

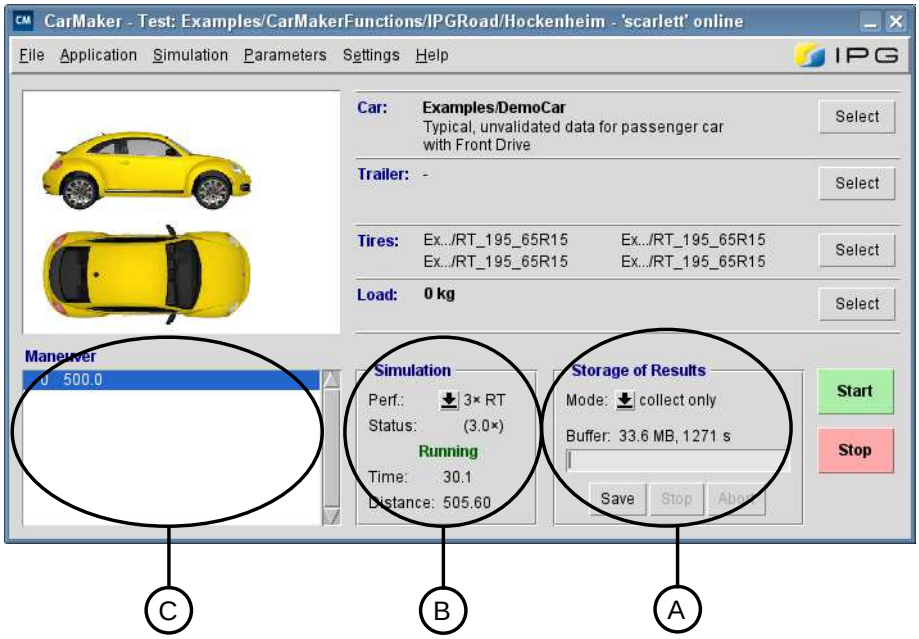


Figure 4.6: The CarMaker GUI

- Box (A) “Storage of Results“

The results of CarMaker are saved to a buffer of the computer memory.

You can save your results to a file, too, in order to analyze them later. This box enables you to control the saving strategy.

- Box (B) “Simulation” feedback

Note a powerful feature of CarMaker! You can define the speed of the simulation: real time, slower or faster than real time. The “max“ option enables to simulate as fast as your CPU allows: up to 40 times faster than real time.

That saves time, especially when you need to automate your TestRuns! And the simulation speed can be changed even while the simulation is running.

In this box the time and distance of the current simulation are displayed. You also receive information about the status of the simulation (“idle“ when no simulation is running, “preparing“ during the start phase, or “Running“).

Change the Simulation Speed to 2x, then Max and then back to “realtime”.

- Box (C) “Maneuver“

As you will see in section 5.2 ’Defining the Maneuver’ on page 40, you can define several maneuver steps. They are shown in this box. The current maneuver step is highlighted in blue. Thus you always know which maneuver step currently is being performed.

Instruments

The Instruments are an additional display used to check graphically the most important data. The content of the Instruments is very similar to what you see in the cockpit of a real car!

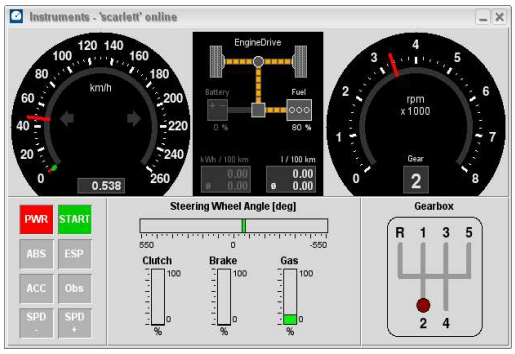


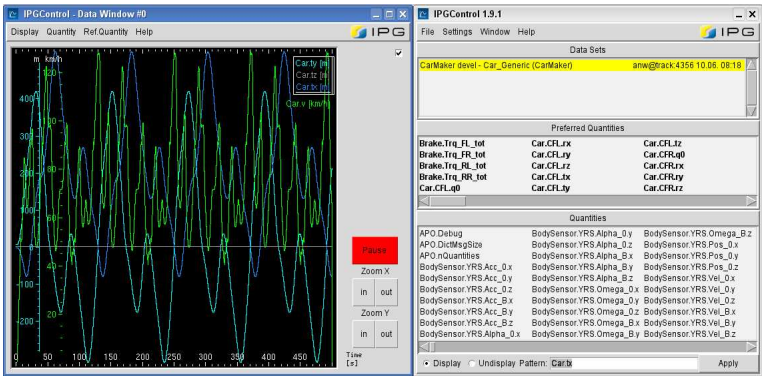
Figure 4.7: The Instruments display

Note that you can extend the content of the Instruments: CarMaker features an interface with the tcl/tk script language, which enables you to build your own GUIs.

IPGControl

IPGControl is an embedded tool to plot various diagrams of simulation results online.

Click in the CarMaker GUI on *File > IPGControl*



IPGControl Data Window IPGControl Selection Window

Figure 4.8: IPGControl Data and Selection Window

In the IPGControl Selection Window: In the field “Quantities“ click left on the variables Car.v and Car.ay in order to plot the speed and the lateral acceleration of the vehicle.

In the CarMaker GUI, start the simulation.

After a while, click on “Pause“ in the IPGControl data window and then click right in the middle of the window and select “Total fit“ in order to fit the diagram within the available space.

The use of IPGControl is explained in detail in section 4.5 ’Result Management’ on page 20.

4.4 IPGMovie

Click again on the green “Start” button to simulate the TestRun Hockenheim.

IPGMovie offers the functionality of an online animation. This means that the current simulation data is provided without delay - the virtual world is directly pictured during the simulation. By loading external result files you also have the possibility to watch the animation of a TestRun which has been carried out before. This is called the offline animation.

The first feature of IPGMovie is that you can change the point of view using the mouse *during the simulation*:

Keep the *left* mouse button pushed down and move the mouse to the left, to the right, up and down. Now keep the *middle* mouse button pushed down and move the mouse up and down in order to zoom in and out of the car (optionally you can use the mouse wheel).

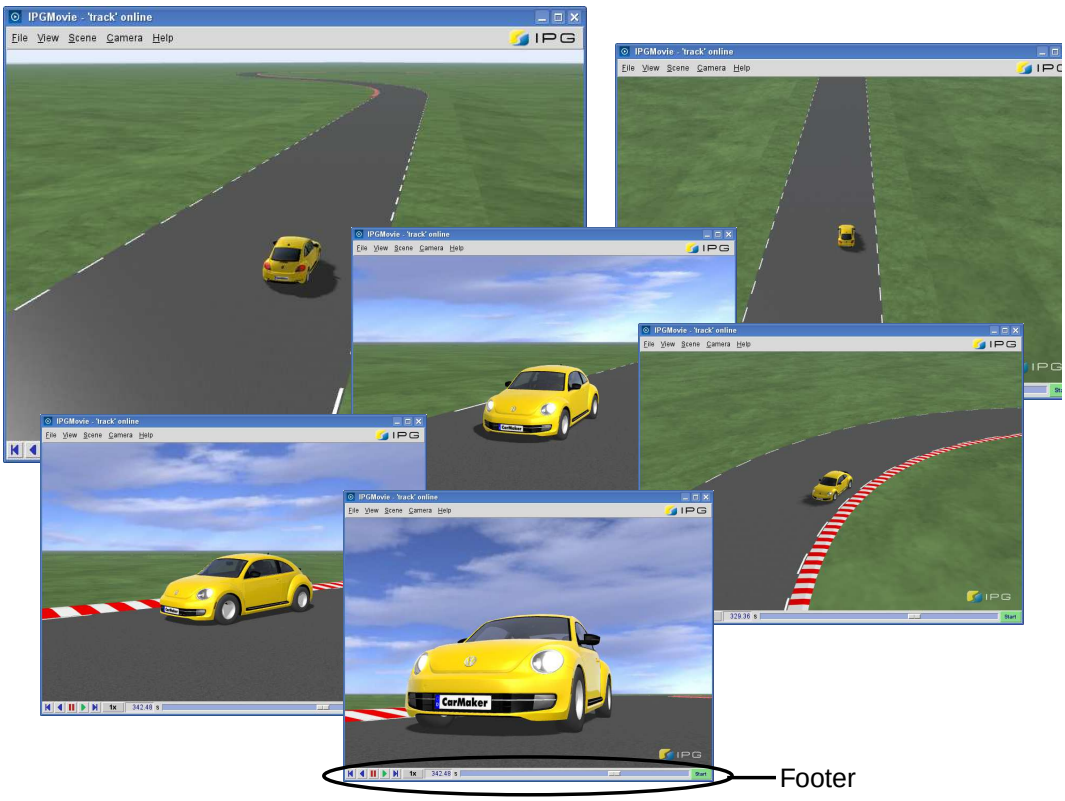


Figure 4.9: Changing the point of view during the simulation

Click on the red “Stop” button in the CarMaker main GUI to abort the TestRun.

After a simulation is finished (at the end of a TestRun or if you select the “Stop“ button on the CarMaker main GUI) you can replay the animation in IPGMovie thanks to the cursor which is in the footer of IPGMovie. You still can modify the camera’s point of view at this time.

Watch again the movie of the last simulation in IPGMovie by using the cursor on the footer. Change the point of view again.

IPGMovie offers many other functionalities you might also like to test:

- Compare the animation of two different simulations: in order to do so you need to perform two simulations with different settings. This feature is explained in the section 5.3.4 'Comparing two Simulations with IPGMovie' on page 53.
- Right-click within the IPGMovie window and test the options available.
- In the IPGMovie window, click on the top menu "Camera", and test the options available. You can also define your own camera view.
- In the IPGMovie window, click on the top menu "Scene", and test the options available (except the last three ones used to compare the animation to different simulations).
- Export videos or pictures of your animation: click on *File* > *Export* (DivX: only if you have the codec installed on your system).
- Viewing the tire forces with *View* > *Show* > *Forces*: The colored bars at the contact point of the tire represent the forces in each of the three directions.
- In the CarMaker GUI, click on *Parameters* > *Road* > *IPGMovie Interface* and test the options available.

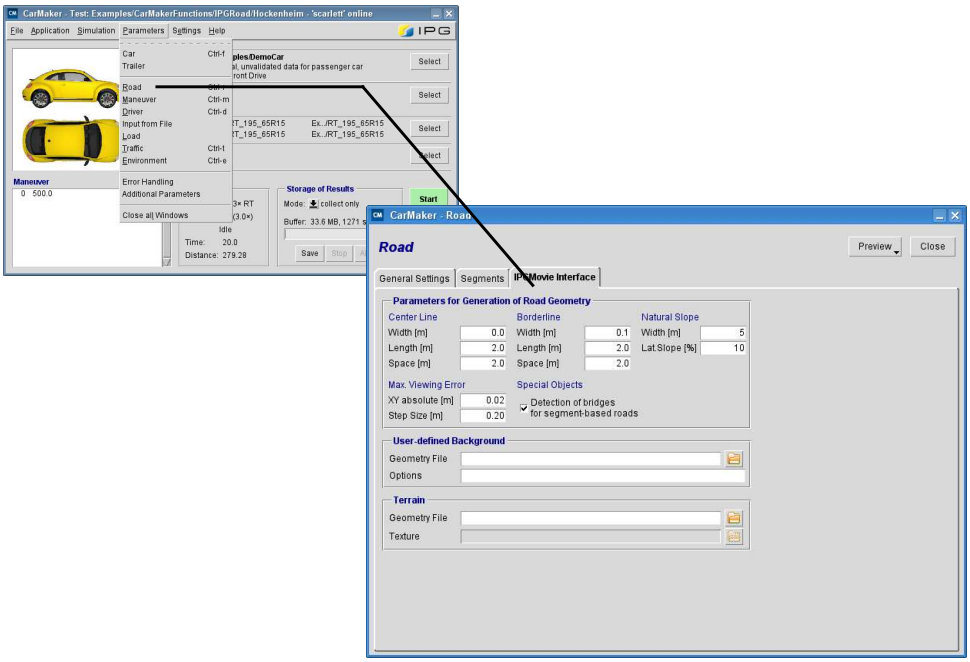


Figure 4.10: IPGMovie Interface

- Please also note that all of the modifications you will make in the road definition will be displayed in IPGMovie. However, you need to update the animation first: start and stop the simulation once. You will learn how to define your own road in the section 5.1 'Exploring the IPGRoad Features' on page 26.

4.5 Result Management

4.5.1 IPGControl

In this section you will learn how to analyze and plot the simulation results with IPGControl.

IPGControl offers the functionality of an online result management. This means that the current simulation data is provided without delay, which enables you to display diagrams directly during the simulation. By loading external result files, you also have the possibility to display the results of a TestRun which has been carried out before. This is called the offline result management. This new source of results is displayed in the “Data Sets“ list of the selection window. The data saved in this file can be plotted in another diagram.

From the figure hereunder, you can see that there is a number of quantities (simulation variables) that can be plotted. IPGControl allows the user to manipulate the plotted quantities, change the scaling, axis, etc. so that the results can be analyzed more efficiently and effectively.

Opening IPGControl

Select IPGControl in the *File* menu of the CarMaker GUI. The IPGControl selection window and a IPGControl Data Window appear.

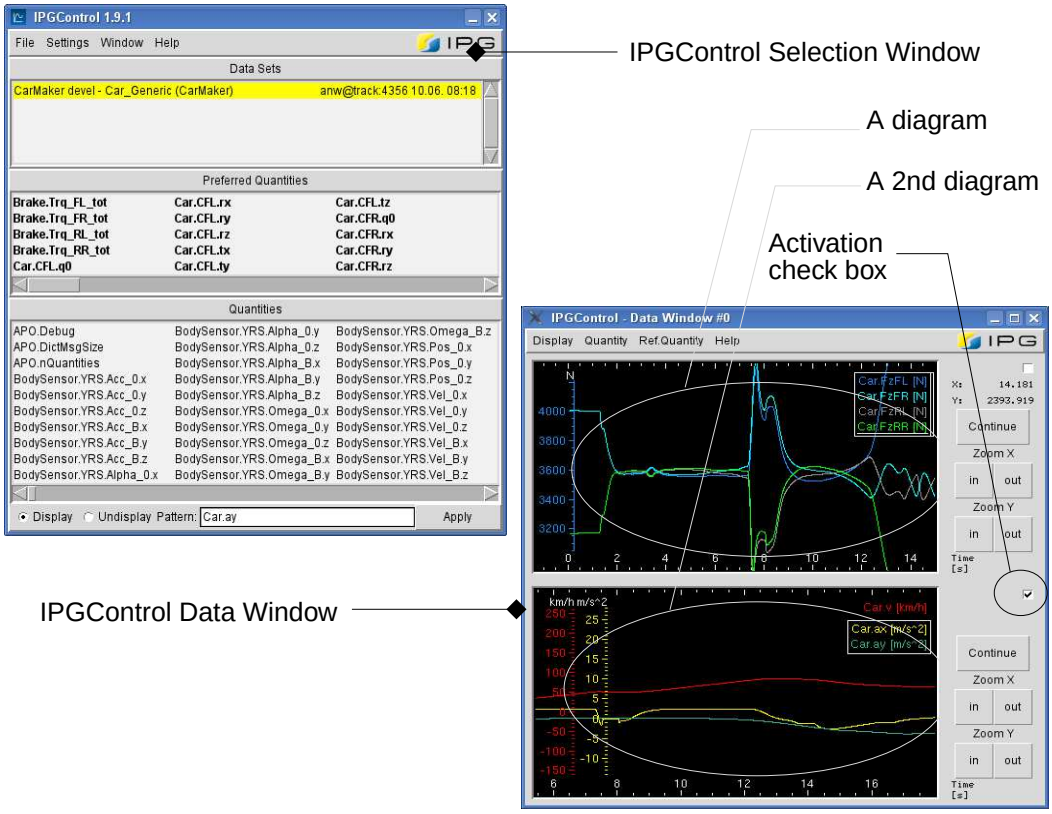


Figure 4.11: IPGControl and IPGControl Data Window

Using the selection window, you can select the result files you would like to display.

The data window(s) is (are) used to display the diagram(s).

Selecting the Quantities to be plotted

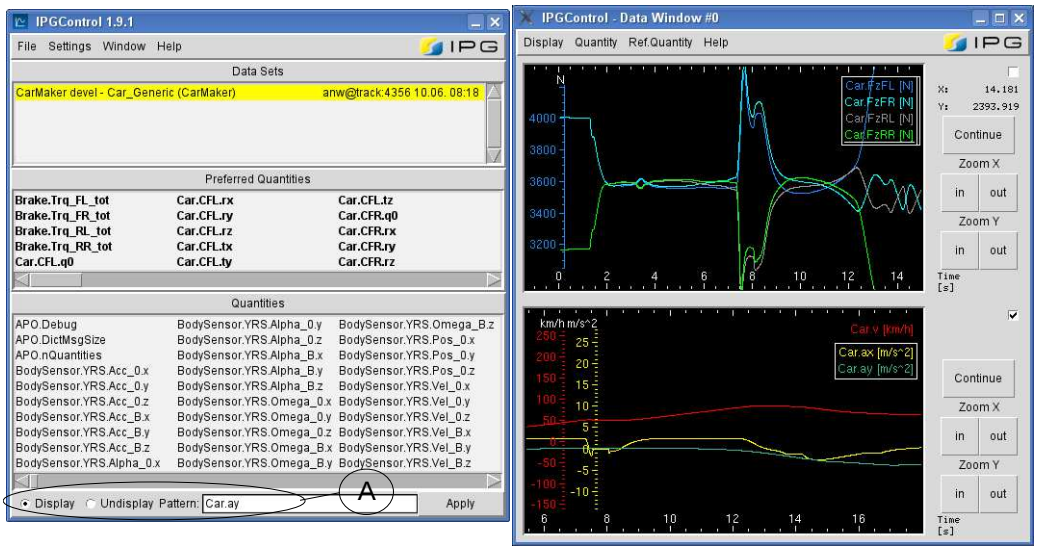


Figure 4.12: Quantities to be plotted

First and foremost, if you have several diagrams in your data window, you have to activate one of it in order to define the quantities that will be plotted: left-click once on it. The activation check box informs you about which diagram is active.

The quantity on the x-axis is selected in the selection window (field “Quantities” or “Preferred Quantities“) with the middle mouse button, the quantities on the y-axis with the left one. Alternatively, you can use the menus “Ref.Quantitiy“ (quantity on the x-axis) and “Quantity“ (quantities on the y-axis) of the data window.

Quantities can be removed from the diagram by clicking again in the list or with the menu of the right mouse button in the graphical window under “Undisplay”.

A “Quantity“ in IPGControl is also referred to as “User Accessible Quantity“ (in short: UAQ). Please refer to the Reference Manual, section “User Accessible Quantities” for an explanation of the quantities you can plot in IPGControl.

Functionalities of the Data Windows

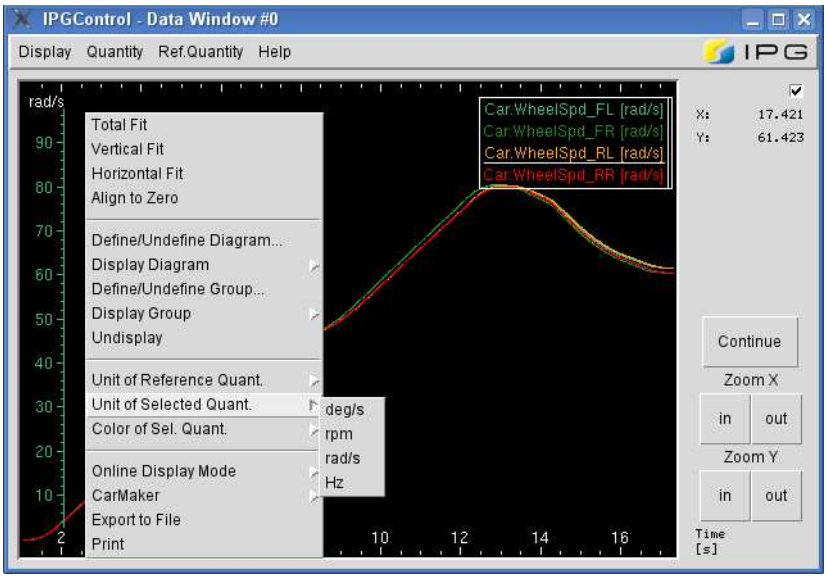


Figure 4.13: Functionalities of the Data Windows

- Diagram
  - Right-click in the middle of the diagram > *Fit* options: fit the curves to the available space of the diagram.
  - Press x and Shift-x: Change the scale of the x-axis.
  - Press y and Shift-y: Change the scale of all selected y-axes. To scale only one axis, click on it before pressing y or Shift-y.
  - Up/Down keys: Change the limits of all selected y-axes. To scale only one axis, click on it before pressing up or down.
  - Shift-Up/Down keys: Change the limits of all selected y-axes (small steps). To scale only one axis, click on it before pressing up or down.
  - Ctrl-Up/Down keys: Change the limits of all y-axes.
  - Shift-Ctrl-Cursor Up/Down keys: Change the limits of all y-axes (small steps).
  - PageUp and PageDown: Change the limits of the x-axis.
  - Home and End: Change the limits of the x-axis to display the whole range of data.
  - Delete and Backspace: Remove the selected quantities from the diagram.
  - Click left on a diagram: Activate the diagram.
  - Click left on a quantity name: Select the quantity.
  - Click left on a y-axis: Select the y-axis and all quantities attributed to it.
  - Ctrl-Click left on a y-axis or quantity name: Add to/remove from the list of selected quantities.
  - Click right: Pop up a diagram/quantity related context menu.
  - Click left on a quantity name, hold the mouse button, drag the quantity across the diagram and drop it on another quantity of the same unit kind (mouse cursor: |<-): Display the quantity on the same y-axis as the quantity targeted on a y-axis of the same unit kind.
  - Click left on a quantity name that uses the same axis as another one, hold the mouse button, drag the quantity across the diagram and drop it on a free location of the diagram (mouse cursor: _|_): Display the quantity on its own (new) y-axis.
  - Double click left on a y-axis: Open an axis parameters dialog window.

- Double click left on a quantity name: Undisplay the selected quantity.
- Right click on a selected quantity > Color / Unit of Selected Quantity: change the color or the unit of the selected quantity.
- Right click on a selected quantity > Unit of Ref.Quantity: change the unit of the reference quantity.
- Right click in the middle of the diagram > Define / Undefine Diagram: remember the quantities currently displayed in order to use the same set of quantities in future work (option “Display Diagram“). Do not forget to save the settings of IPGControl after that (selection window > Settings > Save Settings).
- Right click in the middle of the diagram > Define / Undefine Group: remember the quantities currently displayed in order to use the same set of quantities in future work (option “Display Diagram“). Do not forget to save the settings of IPGControl after that (selection window > Settings > Save Settings). The difference with the option “Save Diagram“ is that you can display several groups of quantities in the same diagram. If you choose the option “Display Diagram“, the quantities you have selected will be removed.
- Top menu Display > Add/Remove Diagram: add or remove a diagram in the data window.

Using the Selection Window

In the selection window you can choose the source from which the data is displayed (field “Data Sets“). You can either choose from the current simulation or load result files from previous simulations.

You can also select the sample rate of the data to be displayed under Settings > Online Sample Rate.

With the option Window > New Data Window, you can add some additional data windows.

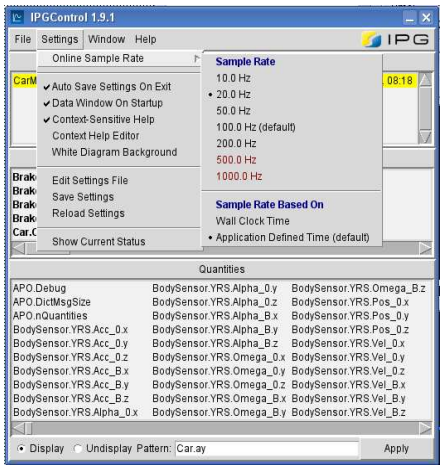


Figure 4.14: Adding Additional Data Windows

The field “Preferred Quantities“ contains only the most frequently used UAQs, while the field “Quantities“ contains more than 600 available UAQs.

- Key shortcuts available in the field “Quantities“ and “Preferred Quantities“:
  - Click left: Display/Undisplay the quantity in the active diagram.
  - Shift-Click left: Display/Undisplay the quantity in the active diagram, giving it its own y-axis.
  - Click middle: Make the quantity the reference quantity of the active diagram.
  - Ctrl-Click left: Add/remove the quantity to/from the list of preferred quantities.

- Bold font: The quantity data is monotone (increasing values).
- Red font: The Quantity is the reference quantity of the active diagram, e.g. the quantity of the x-axis.
- Yellow background: The quantity is displayed in the active diagram.

Example: Please do the following exercise:

Open the CarMaker GUI and load the TestRun SteadyCircle_042m:

- CarMaker GUI > File > Open > Examples > VehicleDynamics > SteadyCircle_042m

Perform a first simulation:

- CarMaker GUI > Start

Open IPGControl. In the data window, add a diagram:

- Data Window > Display > Add diagram

Select the upper diagram:

- click once in the middle of it

In this upper diagram, select the time for the X-axis, and the vehicle speed and the wheel speeds for the Y-axis:

- Data Window > Ref. Quantity > Time
- Data Window > Quantity > Car[S..Y] > Car.v
- Selection Window > Pattern (field (A) at the bottom of the window) > write *WheelSpd* > Select “Display“ > click “Apply”

Select the lower diagram:

- click once in the middle of it

Select the lateral acceleration for the X-axis, and the steering angle as Y-data:

- Data Window > Ref. Quantity > Car.ay
- Data Window > Quantity > Steer > Steer.WhlAng

Start the simulation again:

- CarMaker GUI > Start

Then you can see the curves being plotted online. After the end of the TestRun, try to scale the diagrams as you like, move the curves, try to use two different axes for the wheel speeds.

4.5.2 Other Capabilities

Additionally to IPGControl, CarMaker features an interface for both of the most important result management tools, Excel and Matlab.

Microsoft Excel

When saving the simulation results to a file, CarMaker uses a binary format. After saving your results, you can use a tool (resutil) that converts your result file to an ASCII file (text file), that can be directly loaded in Excel.

A second possibility is to the write the data plotted in a diagram directly to an Excel file using the right mouse button in the diagram window *Export to File*. Excel file formats as well as ASCIIs are available.

The Mathworks MATLAB

After running a MATLAB script for CarMaker, you can use a special command in MATLAB (cmread) that enables to write the content of a result file in a matrix.

Thus you can read the content of a result file directly in MATLAB and plot your results with it.

Next!

You have performed your first simulation using a predefined TestRun and tested the online capabilities of the main tools.

The section 'Extending the Test Scenario Definition' on page 26 will show you how to extend a predefined TestRun by yourself. In this way you will learn how to modify the main part of the CarMaker virtual environment.

Chapter 5

Extending the Test Scenario Definition

5.1 Exploring the IPGRoad Features

First and foremost, load a simple acceleration TestRun. In the CarMaker GUI, click on *File > Open, select the product data pool, then click on Examples > QuickStartGuide > Step1_Acceleration*.

Initially save the TestRun under another name so that you can retrieve your work. In the CarMaker GUI, click on *File > Save as* and choose *Project* on the left bar in the dialog. With right-click, create a new folder "My_TestRuns". Double-click on this new folder and in the field under *Save TestRun* enter any name for the TestRun, e.g.*"exc2_ChangeRoad" > OK.*

Close IPGControl and Instruments for the moment, but keep IPGMovie open.

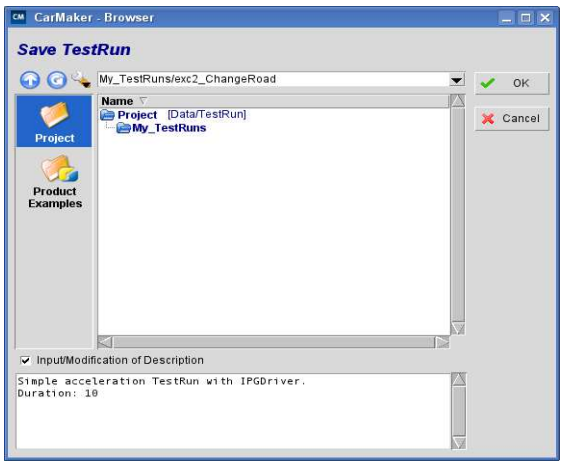


Figure 5.1: Saving the TestRun under another Name

In this chapter you will learn how to extend the TestRun Step1_Acceleration in order to build a closed circuit.

If you quit CarMaker ahead of the end of this chapter, do not forget to save your work by clicking in the CarMaker GUI on *File* > *Save* or optionally with the key combination *Ctrl* + *S*.

5.1.1 Opening the Road GUI

The road GUI is the Graphical User Interface used to parameterize the road.

The road model of IPG, called IPGRoad is a 3D model. You define the trajectory (middle line) of your track and then you define its width and the friction coefficient.

Please note that there are several ways to define the trajectory of your track:

- through the segment based interface of the road GUI
- by loading a road description written in an ASCII file (text file) containing the cartesian coordinates (x, y, z) or the geographic coordinates (long, lat)
- by importing a KML-files (Keyhole Markup Language) created by Google Earth or Google Maps.

The following examples explain you how to use the segment based approach.

Let us open the road GUI. In the CarMaker GUI, click on *Parameters* > *Road*.

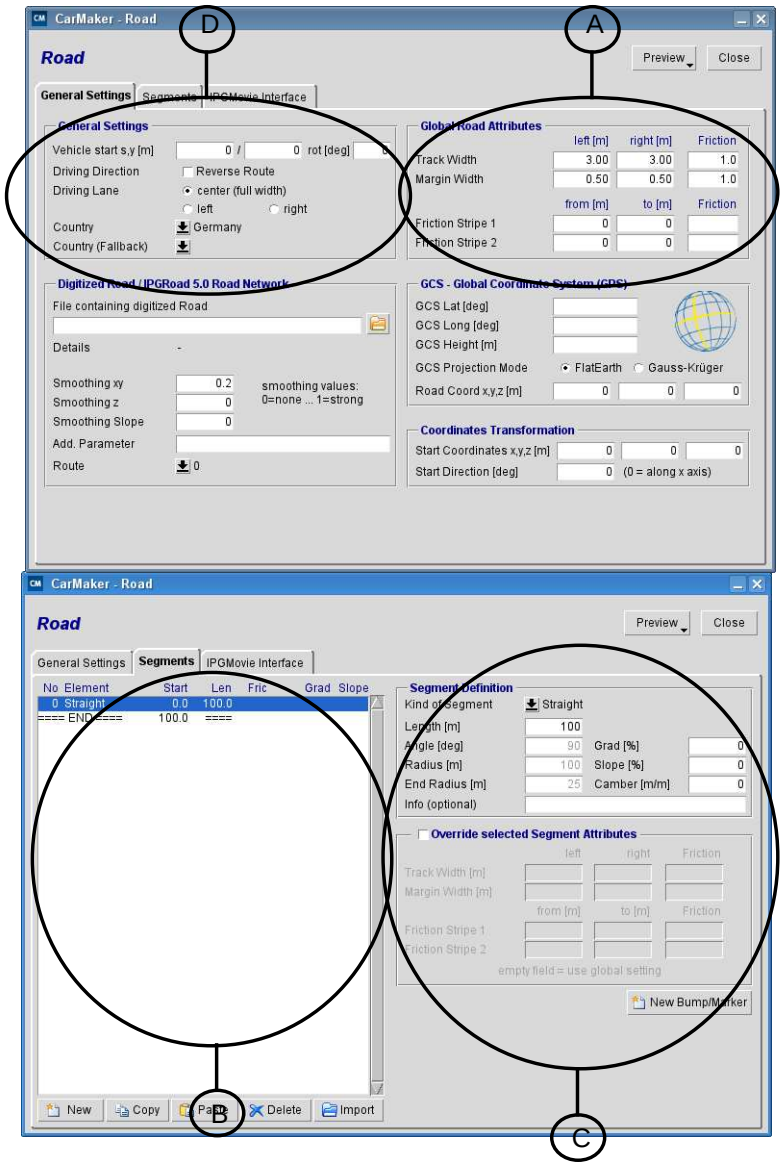


Figure 5.2: Creating a segment based road

In this GUI, we define the array of segments which make up the road.

- Field (A): here you define the width and the coefficient of friction of the entire road. You can also define up to two longitudinal friction stripes.
- Field (B): here we define the list of segments. In this example, we just have a single segment.
- Field (C): here, you can define the geometric properties of the segment that is currently selected (straight line, curve, ...). A segment is selected when it is highlighted in blue in field B. Optionally, you can define a special width and/or friction coefficient for this segment: you find exactly the same parameters as in the field A, but when using those of the field C, they will be applied only to the selected segment, and not to the entire road.
- Field (D): you may select an ASCII file of your choice in which you have digitized your favorite test track. You can also define at which point or distance the vehicle should start on the track.

5.1.2 Checking the Road Geometry

To check the road definition, we will consider two options: the bird's eye view of the road, and IPGMovie.

Click on the Preview button on the top right side of the Road dialog and keep holding it. A drop down menu appears, select the option "Bird's Eye View ".

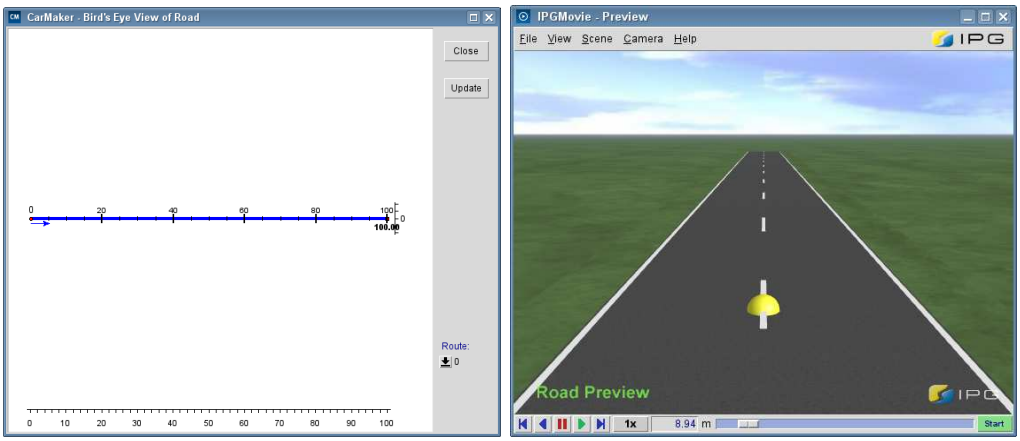


Figure 5.3: Bird's eye view

The bird's eye view of road shows the geometry of the road in two dimensions, which is very convenient to check quickly if the segments you have built looks the way you want them to. Keep this window open for the next examples.

Click on the Preview button on the top right side of the Road dialog and keep holding it. A drop down menu appears, select the option "3D Preview"

In IPGMovie you can see the road definition. You can move the yellow view point by moving the slider or hitting the play button in the footer of IPGMovie.

At this moment, it is just a straight line with a constant coefficient of friction.
You will see in the next section that the changes of the road definition will be displayed immediately in IPGMovie.

5.1.3 Interaction of IPGRoad with the Maneuver Definition

Let us build two additional segments.

In the field (B) of the road GUI, click once on the segment “END“ in the list, and then click twice on the button “New“ at the bottom of the field (B).

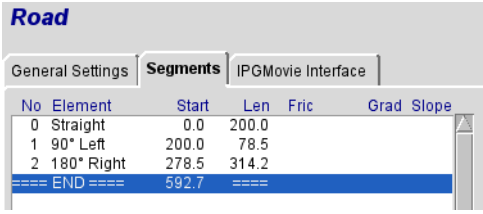


Figure 5.4: Inserting new road segments

You just have inserted two new segments.

Let us now define the geometry of those segments:

- Segment number 1 (the number of the segment is displayed in the segment list): left curve of 90 degrees and radius 50 m
- Segment number 2: right curve of 180 degree, radius 100 m

In the field (B), click on the segment number 1 to select it. In the field (C), click on the arrow “Kind of Segment“ and select “Turn left“. Then define the angle and the radius of the left curve by entering the parameters corresponding to the fields below.

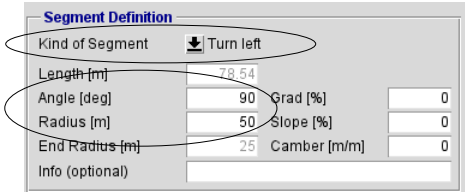


Figure 5.5: Defining the angle and radius of the left curve

In the field (B), click on the segment number 2 to select it. In the field (C), click on the arrow “Kind of Segment“ and select “Turn right“. Then define the angle and the radius of the left curve by entering the parameters corresponding to the fields below.

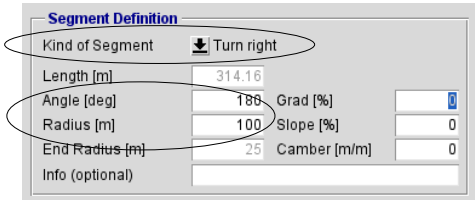


Figure 5.6: Defining the angle and radius of the right curve

In the GUI of the Bird's Eye View, click on Update.

Check the geometry of the road after the update:

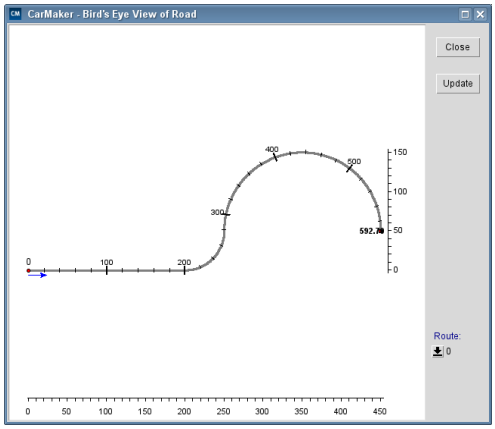


Figure 5.7: Checking the geometry using Bird's Eye View of Road

Click in the CarMaker GUI on the green "Start" button to start the TestRun.

As you can see, the vehicle stops ahead of the end of the road!

Figure 5.8: The vehicle stops ahead of the end of the road

This small example shows you the dependency between the road definition and the maneuver definition.

To make the vehicle drive up to the end of the road, we have to define a maneuver that does not end before the end of the road is reached.

We will make this small change at once in the maneuver definition, although the full maneuver features will be explained later on.

In the CarMaker GUI, click on Parameters > Maneuver.
The maneuver GUI pops up.

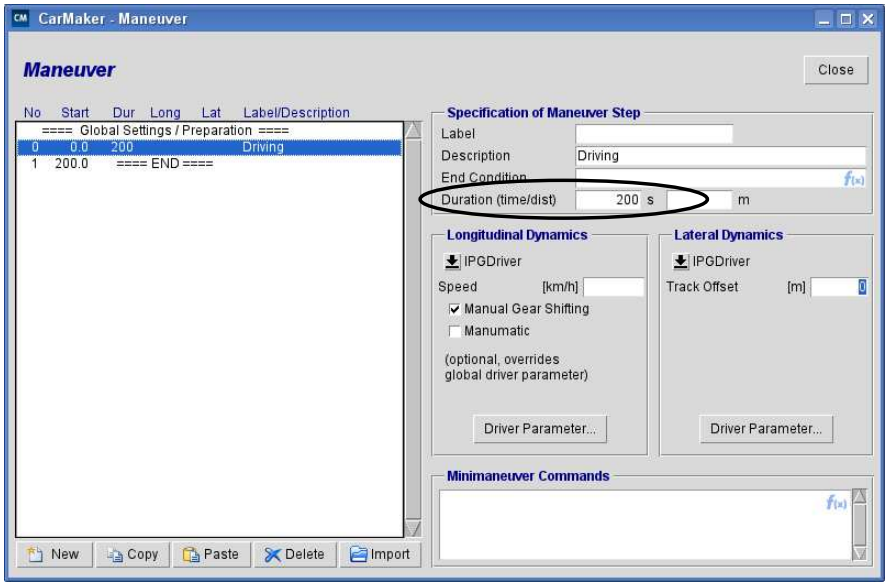


Figure 5.9: The maneuver GUI of CarMaker

In the field "Duration" write the following value: 9999.
Start the simulation again.
Save the TestRun: in the CarMaker GUI click on *File > Save.*

The vehicle now drives up to the end of the road. The maneuver definition will be explained in detail in section 5.2 'Defining the Maneuver' on page 40.

5.1.4 Creating a Closed 3D Road

We will now extend the road in order to build a closed 3D track. If you have closed CarMaker at the end of the previous chapter, please load the TestRun *exc2_ChangeRoad* you had created: In the CarMaker GUI click on *File* > *Open* > *exc2_ChangeRoad* > *OK*. Open IPG-Movie too.

For this example, let us insert 7 new segments which have the following geometry:

- Segment number 3 (the number of the segment is displayed in the segment list): straight line of 100 m, gradient of 10% (gradient: longitudinal inclination)
- Segment number 4: straight line of 300 m
- Segment number 5 : straight line of 100 m, gradient of -10%
- Segment number 6: right curve of 180 degrees, radius 100 m, slope of 25% (lateral slope)
- Segment number 7: left curve of 90 degrees and radius 50 m
- Segment number 8: straight line of 200 m
- Segment number 9: right curve of 180 degrees and radius 200 m

In the field (B) of the road GUI, click once on the segment “END“ in the list, and then click seven times on the button “New“ at the bottom of the field (B).
For each new segment start to define its kind: Straight, Turn left or Turn right. Then you can enter a comment that is displayed in the list of segments, and finally define its geometry (“Length” or “Radius” and “Angle”, “Grad” for the longitudinal inclination, and “Slope“ for the lateral slope).
Then check in the bird’s eye view the geometry, and compare your segment list with the following screenshots:

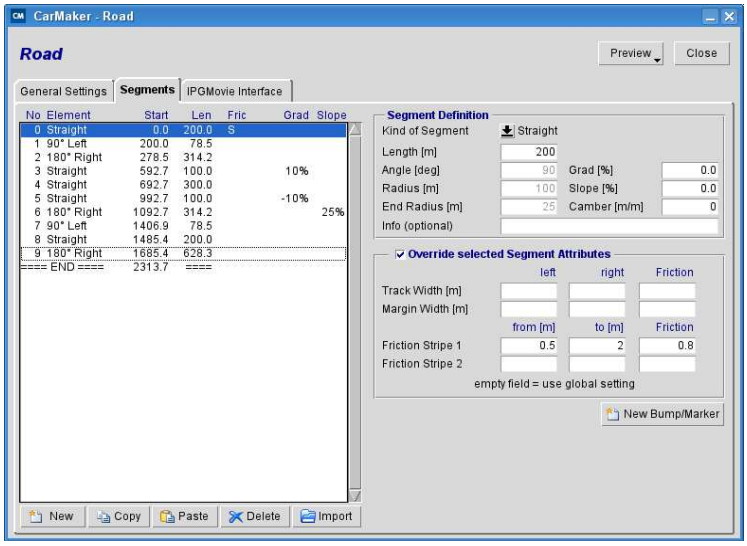


Figure 5.10: Constructing a segment based road

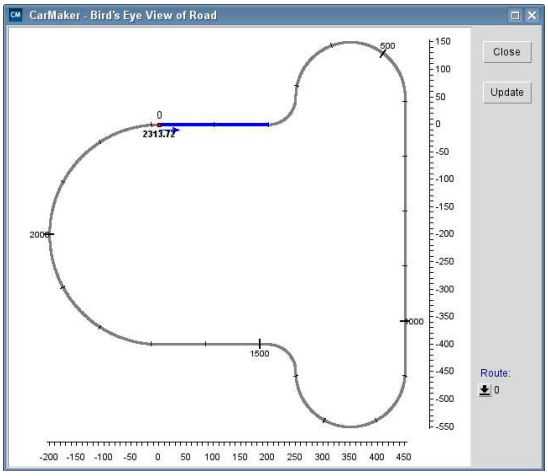


Figure 5.11: Bird’s Eye View of the constructed road

Save your TestRun and start the simulation again.

In IPGMovie you can watch the road characteristics in detail, especially the longitudinal and lateral slopes.

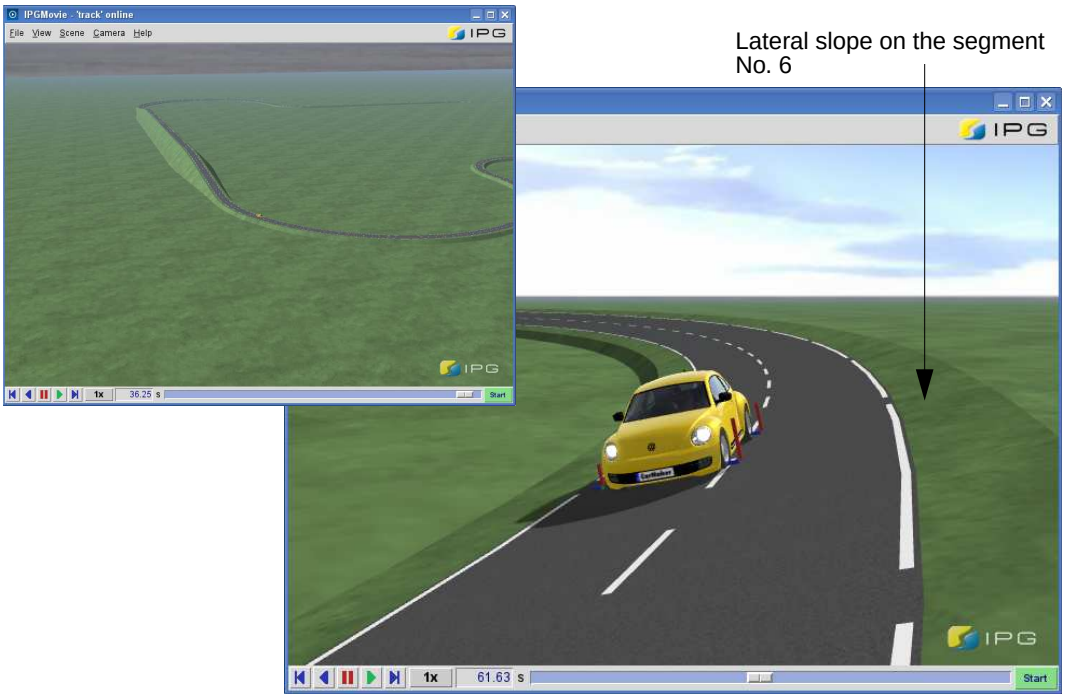


Figure 5.12: Longitudinal and lateral slopes

5.1.5 Modifying the Width and Coefficient of Friction

Up to now we have defined new segments based on the width and coefficient of friction of the original TestRun (Step1_Acceleration). Let us now change these values.

If you have closed CarMaker at the end of the previous chapter, please load the TestRun exc2_ChangeRoad you created: In the CarMaker GUI click on *File* > *Open* > *exc2_ChangeRoad* > *OK*. Open IPGMovie, too.

On the first tab of the road GUI "General Settings", in the area labelled as "Global Road Attributes" define the following values for the track width: left = 3 m, right = 4 m.

Start again the simulation and watch the differences in IPGMovie.

As you can see in IPGMovie, you have the possibility to define the left and right width independently and the widths are also displayed in the animation movie.

An important feature of those parameters is that they apply to the entire track.

What if you want to change those parameters only on a certain section of the track?

Of course, you have the possibility to do so. For this, you have to select the segment on which you want to change those parameters (click on it in the segment list), in the field (C) of the road GUI, second tab where you define the segments, activate the option “Override selected Segment Attributes“: some extra parameters that were hidden before are now accessible.

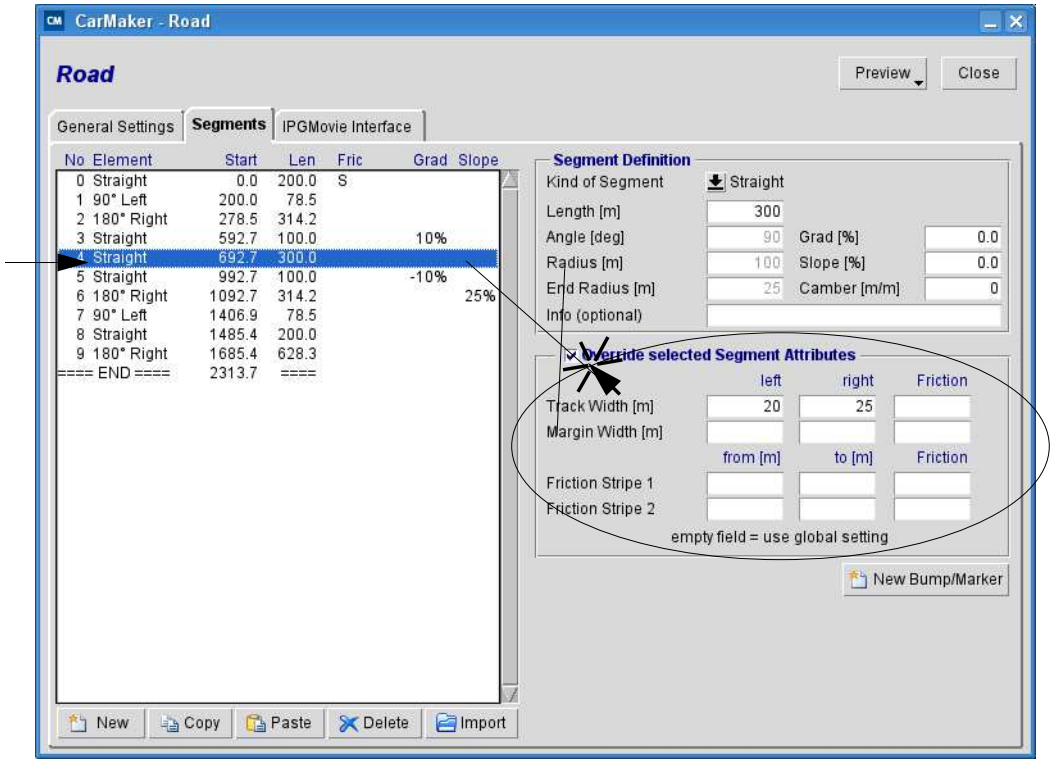


Figure 5.13: “Override selected Segment Attributes” dialog

Here you find the same parameters as in the “Global Settings”, but if you enter a value for one or more parameters, you modify this parameter only for this segment. If you want to modify only one parameter and leave the others as they are, you just have to write a value for the parameter you want to change and leave the other parameters empty.

For instance, in the picture above, the width of the track on both sides for the segment number 4 was changed. However, the friction coefficient, the margin width and the friction stripes were left as is.

Let us now try out this feature and change the width and the coefficient of friction of two segments:

- on the segment number 0 (the first one), we will add a friction stripe. E.g. the coefficient of friction of the road might change along the road width.

In the segment list: select the first segment (No. 0) which is a straight line and activate the option “Override selected Segment Attributes“.

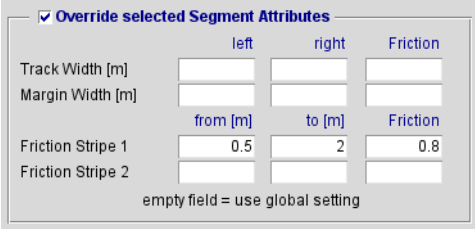


Figure 5.14: Override selected Segment Attributes, Segment No. 0

Then define the following values for “Friction Stripe 1”(leave the rest empty): from = 0.5 m (positive value is the left side of the road) to = 2 m: Friction = 0.8

Start the simulation and look at the changes in IPGMovie. Save the TestRun.

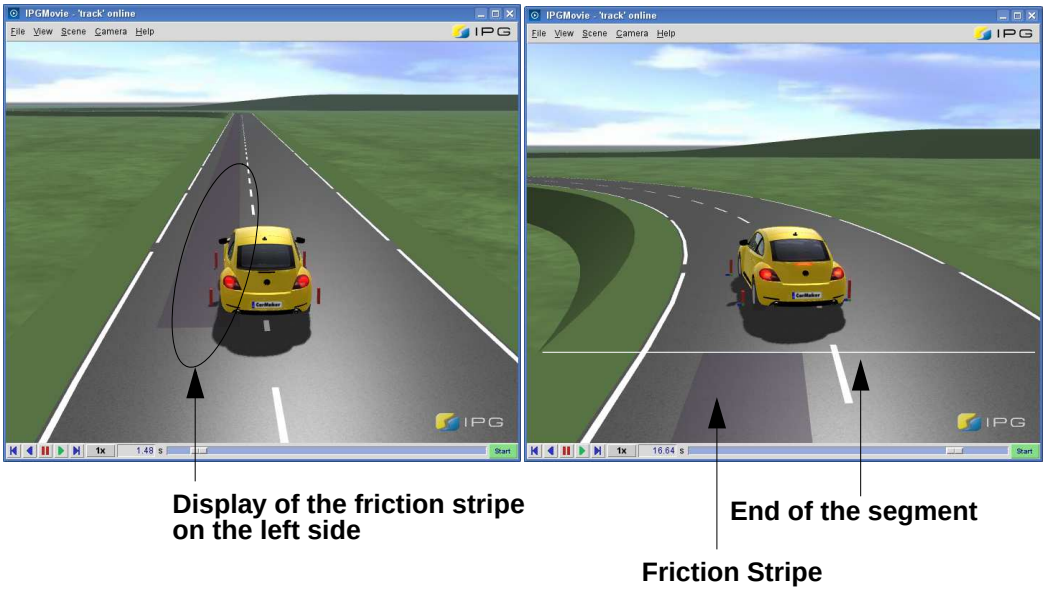


Figure 5.15: Defined Friction Stripe

- on the segment number 4, we will generate a small vehicle dynamic platform by enlarging the track width for this segment.

In the segment list: select the segment No. 4, which is the straight line between the two inclinations and activate the option “Override selected Segment Attributes“.

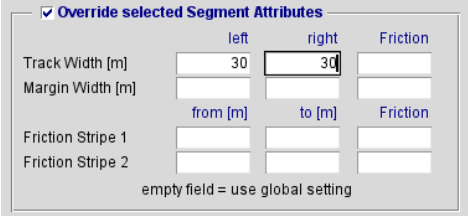


Figure 5.16: Override selected Segment Attributes, Segment No. 4

Then set the following values for the “Track Width” (leave the rest empty):
left = 30 m
right = 30 m

Start the simulation and look at the changes in IPGMovie. Save your TestRun.

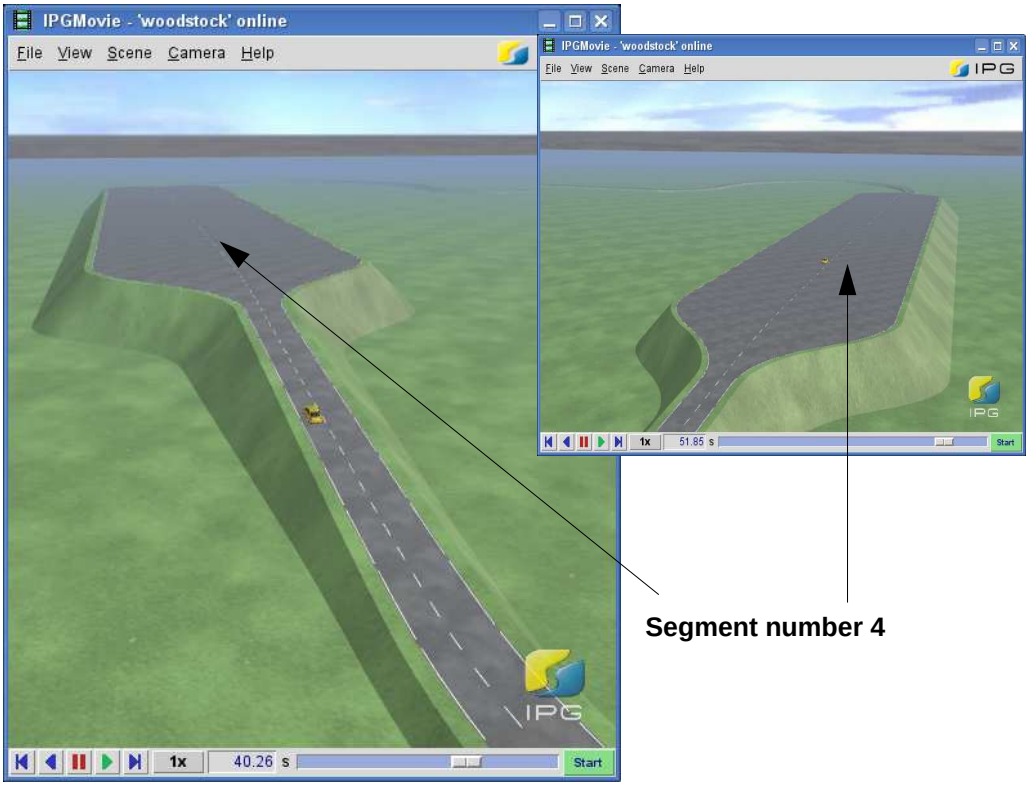


Figure 5.17: Changes in IPGMovie

5.1.6 Inserting Bumps, Markers and Movie Objects

Changing the road layout through track width and friction is only one of the numerous possibilities that IPGRoad offers. On the right side of the segments tab in the Road GUI there is the button “New Bump/Marker”. Hitting the button you can add some special items to the current road segment.

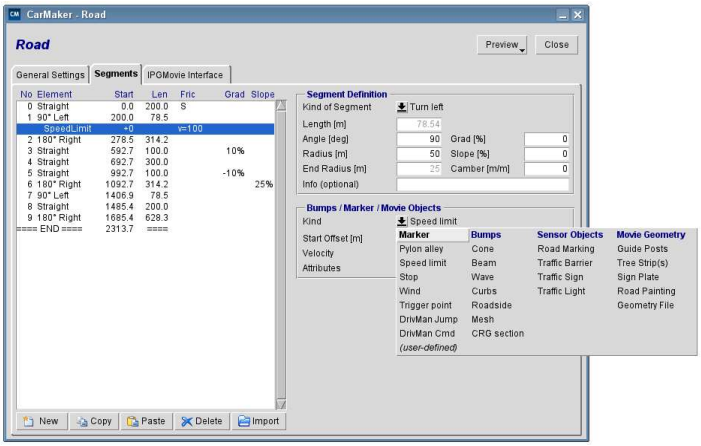


Figure 5.18: Bumps/Markers available in CarMaker

The selectable features are grouped in Markers, Bumps and Movie Objects:

- Markers:
  Markers are features which have a special functionality: e.g. a speed limit will be regarded by the driver and he will follow a certain velocity. The stop marker makes the driver stand still for a certain time span. Moreover, a side wind can be implemented as well as timing measurements.
- Bumps:
  Give your road a three-dimensional appearance, you can insert ground waves, potholes etc.
- Movie Objects:
  Do not influence the simulation but serve as design elements. You can insert objects such as guide posts or tree stripes. With the option “sign plate” you have the possibility to integrate your own image, e.g. to model a gateway sign or a zebra crossing.

For more information about the parameters of the single elements please have a look at the User’s Guide, chapter 5.6 “Road”.

Figure 5.19: A Road with activated Markers and Movie Objects

Next!

At this point, you can compare your work (saved under exc2_ChangeRoad) with the predefined TestRun “Step2_ChangeRoad” (located under Examples > QuickStartGuide). After having built a road, let us now look in detail at the maneuver definition.

5.2 Defining the Maneuver

If you have closed CarMaker at the end of the previous chapter, please load the TestRun exc2_ChangeRoad TestRun you created: In the CarMaker GUI click on *File* > *Open* > *exc2_ChangeRoad* > *OK*. Open IPGMovie, too.

Save at once this TestRun as “exc3_ChangeManeuver“.

In this chapter, we are going to use the road you have built in the previous chapter. Based on this TestRun, we will extend the maneuver definition.

5.2.1 Opening the Maneuver GUI

In the CarMaker GUI, click on *Parameters* > *Maneuvers*.

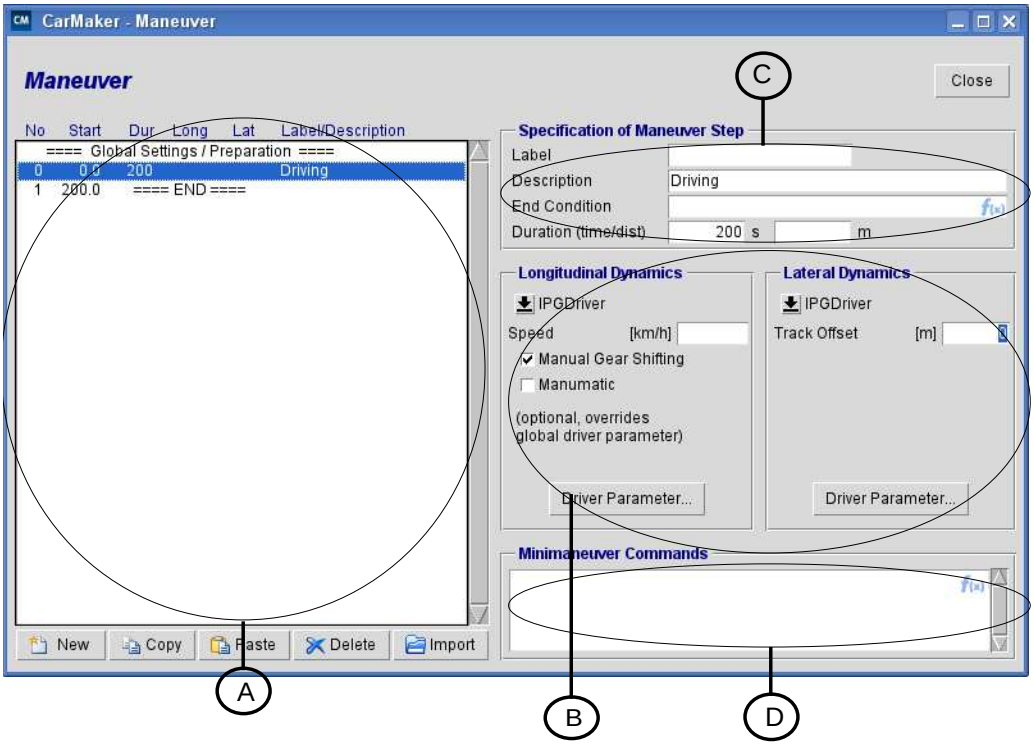


Figure 5.20: The maneuver GUI in CarMaker

Let us take a look at this GUI:

- field (A): this is the list of all maneuver steps.
- field (B): longitudinal and lateral dynamics. Here you define how the car should be controlled. Note that the longitudinal and the lateral control of the vehicle are defined completely independently from each other.
- field (C): here you define at which time or distance the selected maneuver should stop.
- field (D): the field “Additional Action” is a feature for experienced CarMaker users. Here you can program advanced maneuver definitions. As this field is optional, we will leave it empty at this point.

5.2.2 Limiting a Maneuver Step by Distance

Up to now, we have always simulated with a single maneuver step which is 9999 s long (see section 5.1.3 page 32).

Let us try to limit the maneuver step duration not by time, but rather by distance. It is possible to abort the maneuver step when the vehicle has travelled a certain distance.

In the following example, we will stop the maneuver when the vehicle reaches the beginning of the dynamic test area (the segment No. 4 which is at the top of the slope).

First and foremost, you have to know where, e.g. at which distance on the track, this segment starts.

Open the road GUI (*CarMaker GUI > Parameters > Road*)

In the segment list, read and remember the value in the column “Start“ corresponding to the segment No. 4.

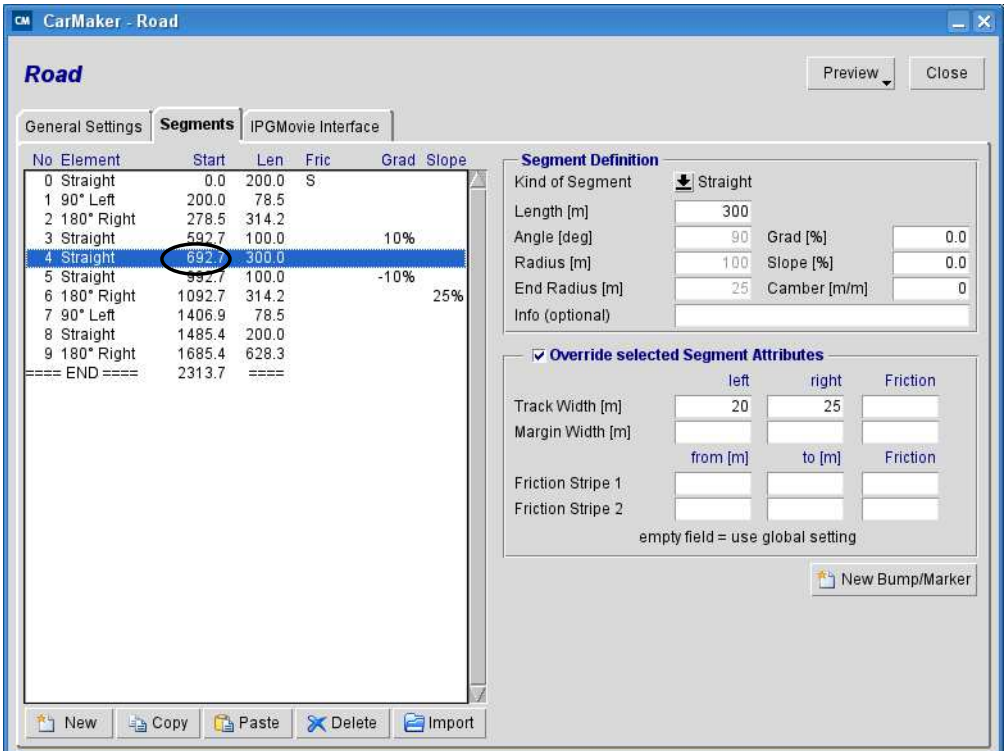


Figure 5.21: The distance where to stop the maneuver

This value should be 692.7(m), and corresponds to the distance at which this segment starts.

Close the road GUI, and now open the GUI used to define the maneuvers.
In the CarMaker GUI, click on *Parameters > Maneuvers*.

In the field "Distance", write the value you found in the road GUI: 692.7 m.

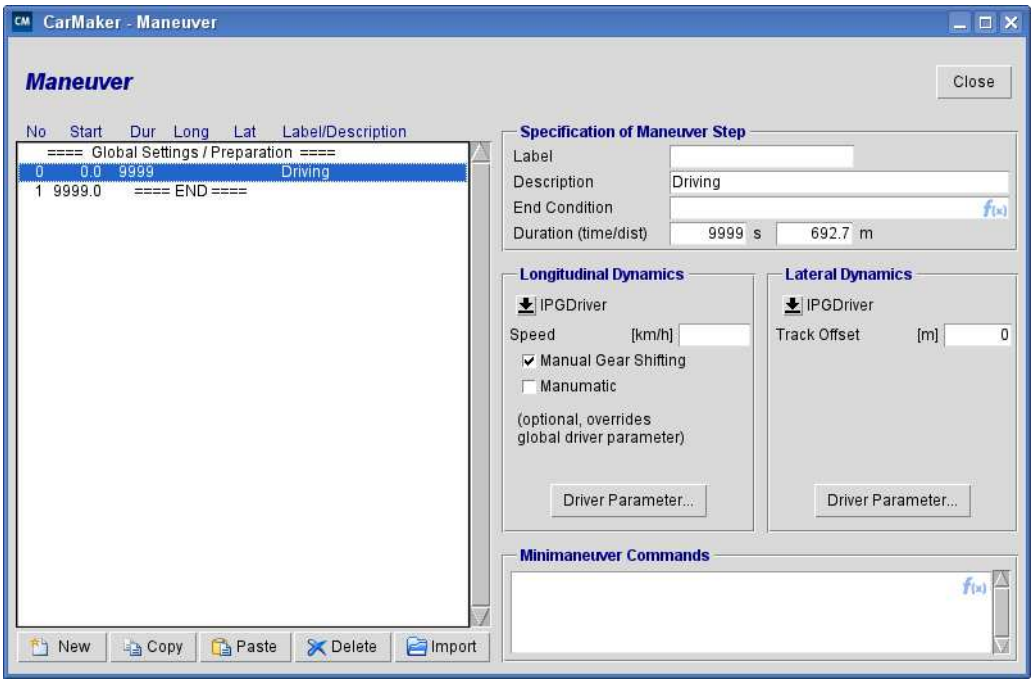


Figure 5.22: Defining the distance as an end condition of the maneuver

Save your TestRun and start the simulation.

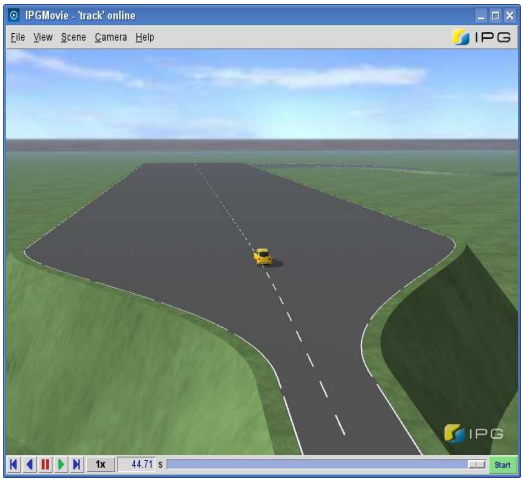


Figure 5.23: The new maneuver step displayed in IPGMovie

Now, the vehicle stops at the beginning of the dynamic test platform, as we wanted it to.

However, you may have a question: why does the vehicle stop at this distance, and not after 9999 s, since we did not remove this value from the maneuver GUI?

The answer is simple: The vehicle reaches the distance of 692.7 m before 9999 s have passed. At the opposite, what would happen if we defined, let us say, a time of 5 s and a distance of 692.7 m? The maneuver would stop at 5 s since the vehicle would not reach 692.7 m within such a short time!

5.2.3 Adding an Open Loop Maneuver Step

In this section we are going to use several maneuver steps during the same simulation by inserting an open loop maneuver step.

We will use the dynamic test platform to perform a sinus steering maneuver.

In the maneuver GUI, click once on the last maneuver step ("END") to select it, and then click once on the button "New" at the bottom of the list.

Define this new maneuver step as follows (select it - it must be highlighted in blue):
Duration: 3 s,
longitudinal dynamics: IPGDriver,
lateral dynamics: sinus steering, amplitude 50 deg, period 1.0 s, number of periods=3

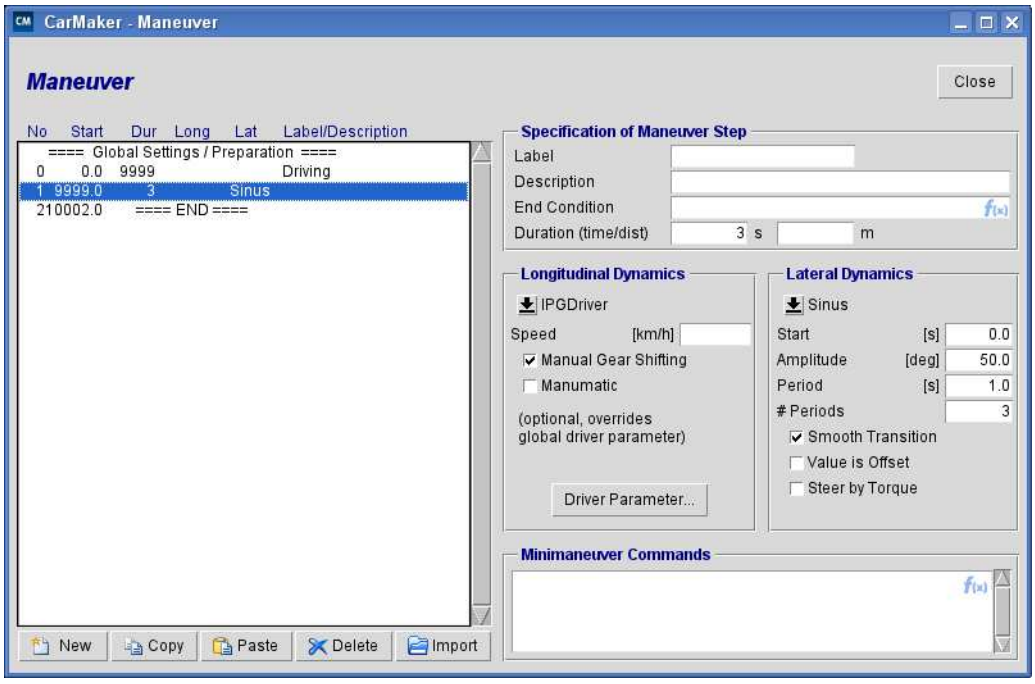


Figure 5.24: Defining an open loop maneuver step

Save your TestRun.
In the CarMaker GUI, turn the simulation speed on "realtime" (box "Simulation").
Start the simulation again, watch what happens in IPGMovie and pay attention to the box "Maneuver" in the CarMaker GUI.

In the "Maneuver" box of the CarMaker GUI, both maneuver steps are displayed.

During the first part of the maneuver which lasts up to the end of the inclination, the first maneuver step is highlighted. Then, as the sinus steering phase starts, the second maneuver step is highlighted.

This very simple example shows you how to insert an open loop maneuver, and then check which maneuver step of the TestRun is running right now!

Now, let us insert a closed loop maneuver step.

5.2.4 Adding a Closed Loop Maneuver Step with IPGDriver

After the sinus steering maneuver step, we will extend the maneuver with a closed loop maneuver performed by IPGDriver.

IPGDriver is IPG's driver model which acts exactly like a real driver. The virtual driver is composed of a controller to follow a course and of a speed controller.

In the maneuver GUI, click once on the last maneuver step ("END") to select it, and then click once on the button "New" at the bottom of the list.

Define the new maneuver (No. 2) step as follows:
Duration: 9999 s
longitudinal dynamics: IPGDriver, Speed (100 km/h)
lateral dynamics: IPGDriver.

Save the TestRun, open the Instruments, and start the simulation.

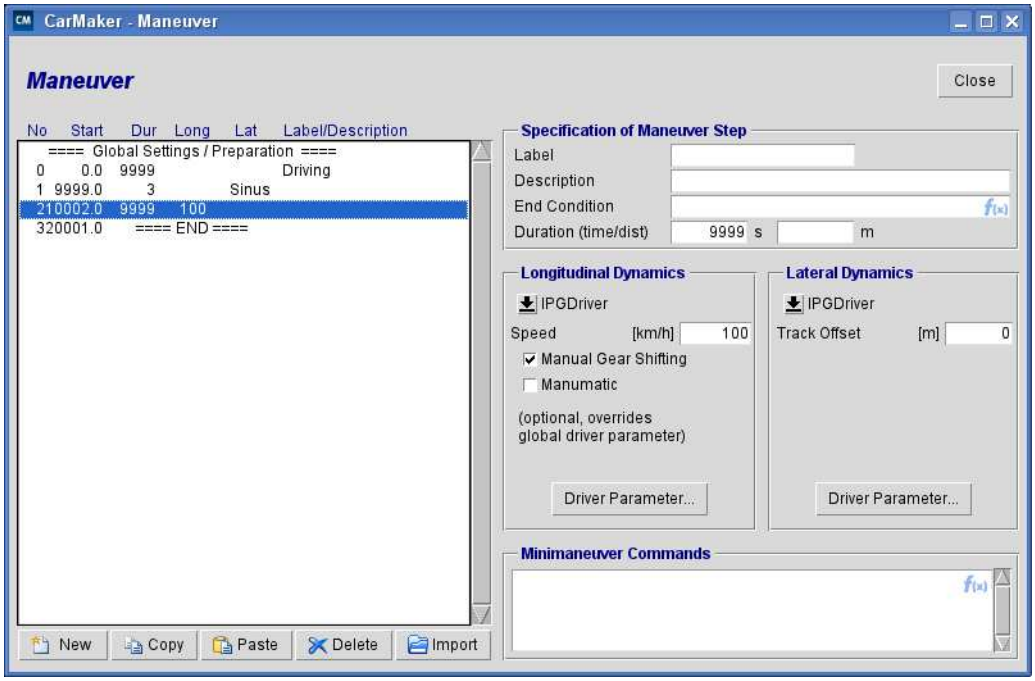


Figure 5.25: Defining a closed loop maneuver step

The vehicle now drives further after the sinus steering, since you have inserted another maneuver step which lasts 9999 s.

Observe the speed using Instruments: during the third maneuver step, the driver accelerates up to 100 km/h. These values correspond to what we have defined in the corresponding maneuver (No. 2).

This example shows you two things:

- You can define the maximum speed reached by the driver independently for each maneuver step.
- The driver stays on the track and finds the best trajectory automatically. The driver's parameters will be explained in section 5.3.3 'Parametrizing IPGDriver' on page 51.

5.2.5 Other Options for the Definition of the Maneuver Step

In this TestRun, we have only worked with IPGDriver and a very simple sinus steering, since they are among the most important options used in CarMaker.

However, you saw that there also are other options available. Let us have a look at them.

Duration of the maneuver

Figure 5.26: Defining the duration of the Maneuver Step

We discussed in section 5.2.2 'Limiting a Maneuver Step by Distance' how to restrict the maneuver step by a certain distance passed by the vehicle.

But it is far-off being the single solution: in the parameter "End Condition", you can define other parameters which will abort the maneuver step. Right-click in the entry field of End Condition and you will find a list of all User Accessible Quantities in CarMaker. You can choose one of it and combine it with any logical operator such as smaller as (<=), larger than (>=) or same as (==) to define an end condition. There is also the possibility to set more than one condition at the same time: you can combine more than one quantity using the logical AND (&&) or OR (||).

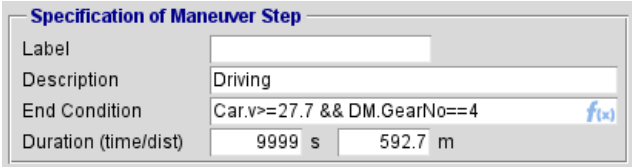


Figure 5.27: Defining the "End Condition" Parameter

Longitudinal Dynamics

Figure 5.28: Defining the longitudinal dynamics

To control the longitudinal dynamics, e.g. the speed and/or longitudinal acceleration of the vehicle, several options are available:

- IPGDriver: fully automated control of the vehicle speed and acceleration. See section 5.3.3 'Parametrizing IPGDriver' on page 51.
- Speed Profile: this option enables to use speed measurements.
- Speed Control: this option is a simple speed controller.
- Manual: enables to directly control the position of the pedals.

- Stop Vehicle: enables to stop the vehicle. This option is especially designed for engineers working on active park assistance systems.
- Drive Backwards: same option as IPGDriver, but backwards.
- IPGDriver + User Driver: activate your own driver model.

Lateral Dynamics

Figure 5.29: Defining the lateral dynamics

The vehicle's direction can be defined through several options:

- IPGDriver: fully automated control of the steering wheel. See section 5.3.3 'Parametrizing IPGDriver' on page 51.
- Sinus: sinus input at the steering wheel.
- Sinus Sweep: alternating sinus input at steering wheel.
- Steer Step: steer step input at the steering wheel.
- Follow course: this option is a steering wheel controller like IPGDriver, but in a much more simple way..

Next!

At this point, you can compare your work (saved under "exc3_ChangeManeuver") with the pre-defined TestRun "Step3_ChangeManeuver" (located under Examples > QuickStart-Guide).

Now, you can parameterize a road and define maneuvers.

In the next chapter you will find further information about IPGRoad and IPGDriver.

5.3 Advanced Features

5.3.1 Using a Digitalized Track

In the CarMaker GUI, load the TestRun “Nordschleife“: *Examples > CarMakerFunctions> IPGRoad > Nordschleife*.

Set the simulation speed to “Realtime“, open IPGMovie, and start the simulation.

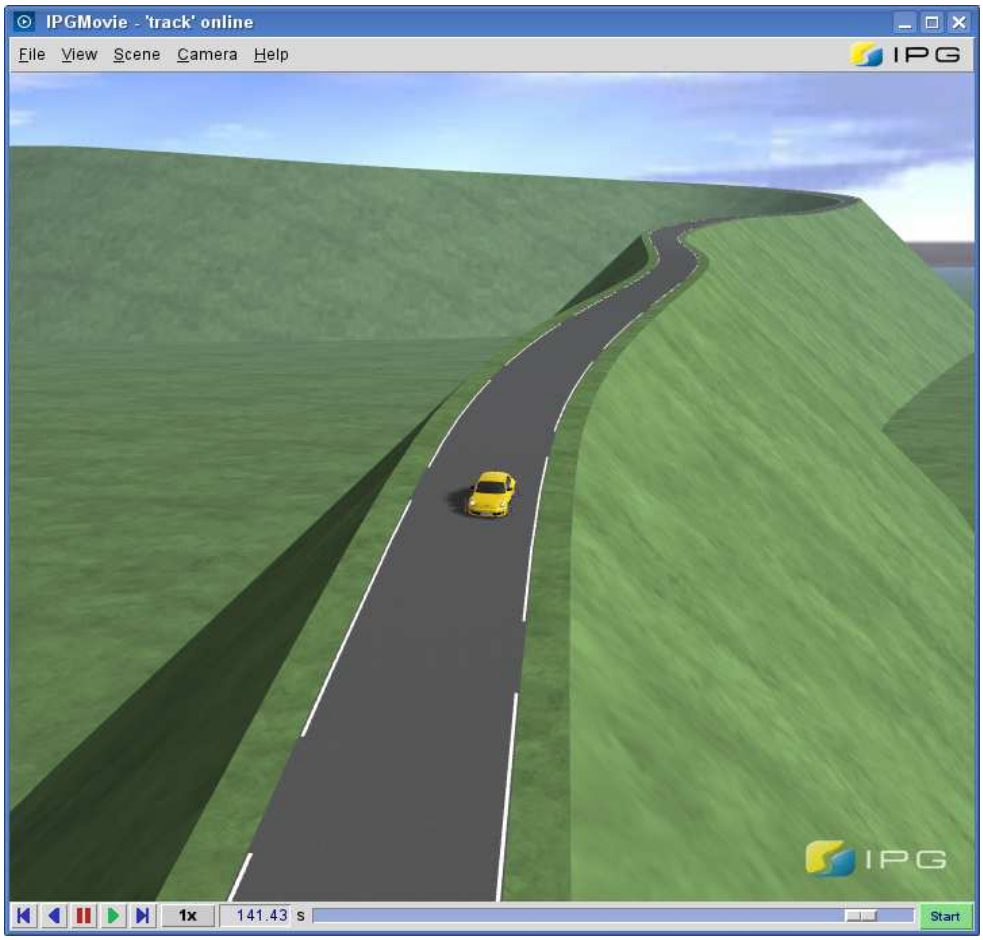


Figure 5.30: The TestRun “Nordschleife” in IPGMovie

The striking difference with the road you have built in the previous chapters is that the road used here has a much higher resolution.

This is very difficult to obtain with a segment based track. To get this result, we use in this TestRun a digitalized road.

This means that the x, y and z coordinates of the middle line of this track have been written to a file. In this case, there is no need to define the segments since the trajectory is already given by the data in the file.

Save this TestRun as “exc4_ChangeDriver”.

In the CarMaker GUI, click on *Parameters* > *Road* to open the Road GUI.

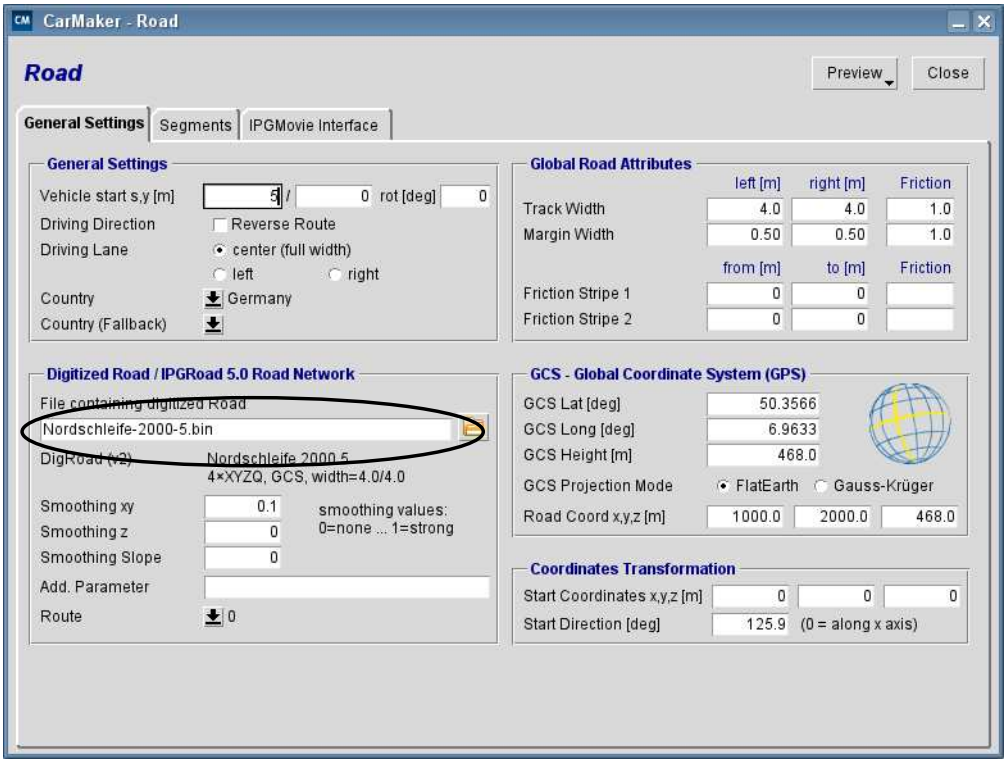


Figure 5.31: Loading a digitized road in Ithe Road Dialog

In the field “General Settings“, you will find the option to load the file with your digitalized track.

This file is a simple ASCII file (text file). Optionally, this ASCII file can be converted into a binary file in order to enable the simulation to run faster, as it is the case for this TestRun.

With a digitalized road, you still need to parameterize the “Global Settings“ of the road in order to define its width and coefficient of friction.

You also have the possibility to override those “Global Settings“. As an example, let us insert the same friction stripe as defined in the TestRun “exc2_ChangeRoad”.

At the bottom of the segment list, which is currently empty, select “New“.

Then, in the field where the segment geometry is defined, do the following:
1) leave the segment geometry as it is (Straight, Length, Grad, Slope, Camber)
2) activate the option “Override selected Attributes for this Segment“
3) in “Friction Stripe 1“, define the width from 0.5 m to 2 m, friction coefficient 0.8

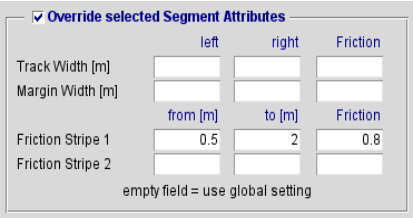


Figure 5.32: Defining a Friction Stripe

Start the simulation.

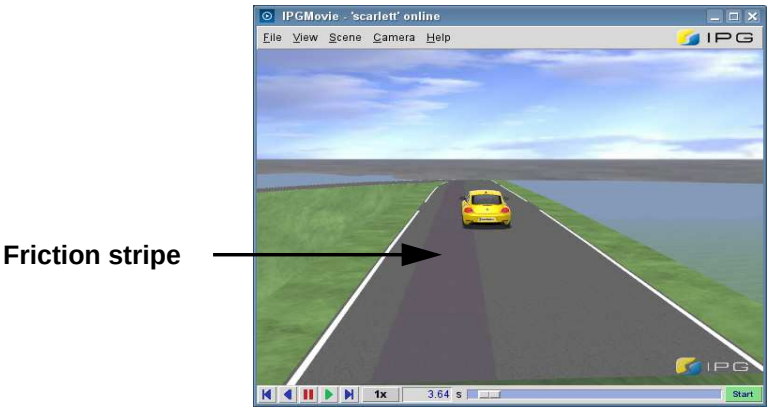


Figure 5.33: The visualization of a friction stripe in IPGMovie

Thus, even with a digitalized road you can define friction stripes, other widths on a given distance, ...

5.3.2 Driving Lane

Let us discover how you can define the driving lane.

First we will parameterize IPGMovie to make the changes obvious.

In the CarMaker GUI, click on *Parameters> Road* > last tab *IPGMovie Interface*.
For the parameter *Center Line*, define a width of 0.1 m.

Now let us change the driving lane.

In the CarMaker GUI, click on *Parameters > Road*.
In the first tab “General Settings” of the Road GUI, select in the menu “Driving Lane“ “right”.
Start the simulation again.

Figure 5.34: Selected the right driving lane in IPGMovie

Now the driver stays on the right lane.

You see how you can modify the animation according to your needs. As an example, the missing white line in the middle of the road corresponds to a race track.

5.3.3 Parametrizing IPGDriver

Now a few words about the parameterization of IPGDriver.

In the CarMaker GUI, click on *Parameters > Driver* to open the IPGDriver GUI.

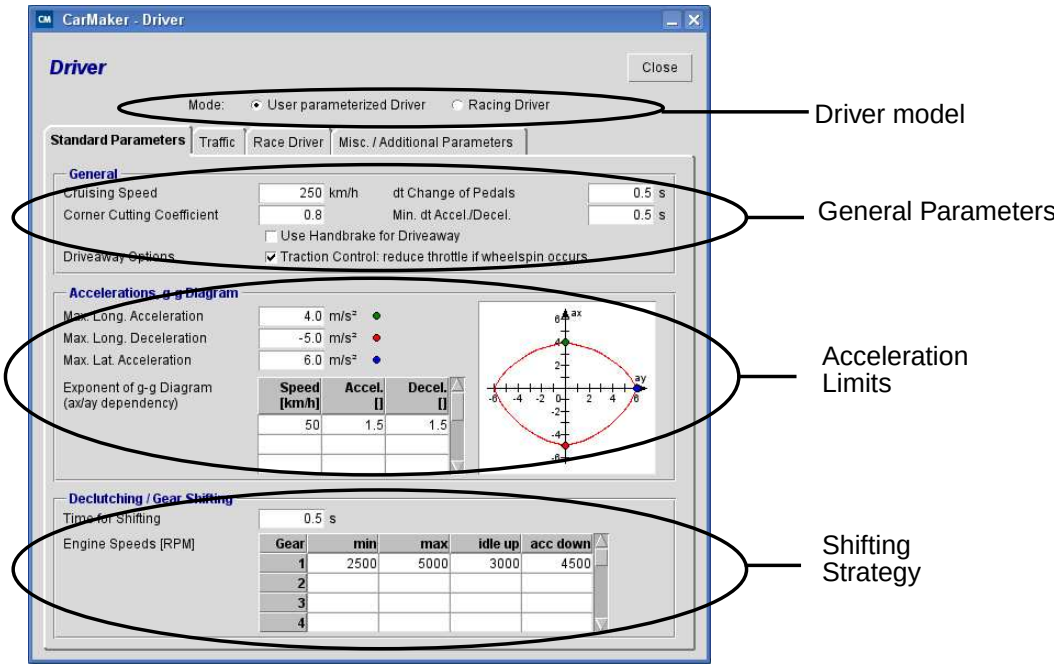


Figure 5.35: Parameterization of IPGDriver

IPGDriver provides two kinds of models:

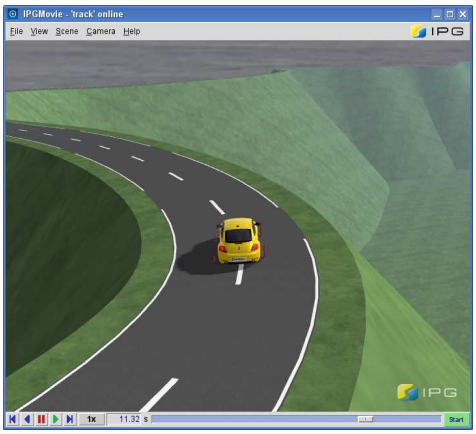
- The “User parameterized Driver”: here you can define by yourself every parameter of IPGDriver. We will use this driver model in the following examples.
- The “Racing Driver“: after a learning phase, this model calculates the physical limits of the driver; you do not need to define them by yourself.

Through the GUI of IPGDriver, you have access to several parameters:

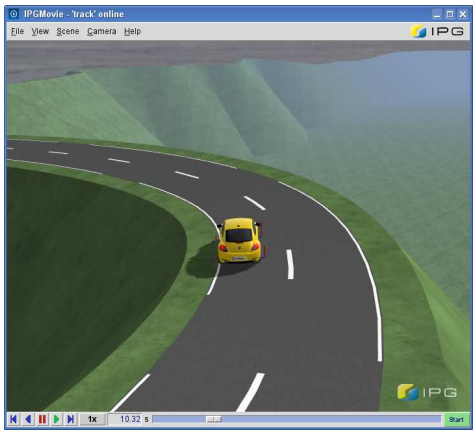
- General parameters, the most important ones are:
  - Cruising Speed: speed that the driver will try to reach, keeping the vehicle on the track. This speed can be re-defined for each maneuver, as we tested it in the section 5.2.4 ’Adding a Closed Loop Maneuver Step with IPGDriver’ on page 44.
  - Corner Cutting Coefficient: controls how the driver should cut the curves. Values range from 0 to 1.0 (0: the driver does not cut the curves at all).
- Acceleration limits: via the G-G diagram you define
  - the driver’s maximal longitudinal and lateral acceleration (green, blue and red points on the diagram).
  - the exponents: they enable to define the shape of the diagram (red line). It gives the driver the information about his maximal acceleration when longitudinal and lateral acceleration are coupled. The exponent can be defined for various speeds.
- Shifting strategy: the engine speed at which the driver shifts up or down is defined here.

Let us test the corner cutting coefficient.

Set the corner cutting coefficient to 0.0, set the simulation speed to “realtime” and start the simulation again.



Corner Cutting Coefficient 0.0



Corner Cutting Coefficient 0.9

Figure 5.36: Different Corner Cutting Coefficients

This small example shows how you can modify the driver’s trajectory.

On the picture above, we can see a clear difference. However, it is not very convenient to make two different screenshots... In the next section we will explain how to compare the animation of two different simulations in IPGMovie.

Do not close IPGMovie!

5.3.4 Comparing two Simulations with IPGMovie

In the IPGMovie window:
1) Click on *Scene > Reference Vehicle > Copy current Motion Data*
2) Activate the option *Scene > Primary and Reference Vehicle*

With the first action, you saved the animation of the last simulation in a buffer memory (e.g. the TestRun on the Nordschleife with a corner cutting coefficient of 0.8).

With the second action, you let IPGMovie display the animation of the simulation that will be started (“Primary Vehicle“), but also the one that is in the buffer memory (“Reference Vehicle“) that will be displayed as a shadow vehicle.

Open the IPGDriver GUI, and set back the corner cutting coefficient to 0.8.
Keep the simulation speed to “realtime” and start the simulation again.

Figure 5.37: Primary and Reference Vehicle displayed in IPGMovie

Two vehicles appear now: the shadow vehicle is the one from the previous simulation, and the other is the one from the current simulation.

Do not save the last change (corner cutting coefficient).

Next!

Now, you can compare your work (saved under “exc4_ChangeDriver”) with the pre-defined TestRun “Step4_ChangeDriver” (located under Examples > QuickStartGuide).

At this point, you know how to load a pre-defined TestRun, start a simulation, define the road, the maneuvers, and how to use IPGMovie and IPGControl.

Let us now jump to the next level: in the following chapter we will explain you how to use other vehicles, trailers and tires in CarMaker.

Chapter 6

Changing the Vehicle and Tire Data

6.1 Using another Vehicle Parameterization

In the previous chapter, you have always used the same vehicle, the so called DemoCar. We will now teach you how to use another vehicle parameterization. If necessary, load again the TestRun “exc3_ChangeManeuver”.

Save your TestRun as “exc5_ChangeVehicle“.
In the CarMaker GUI, click on the button “Select“ in the box “Car“, choose the “Demo_Medium“ car and click on OK.

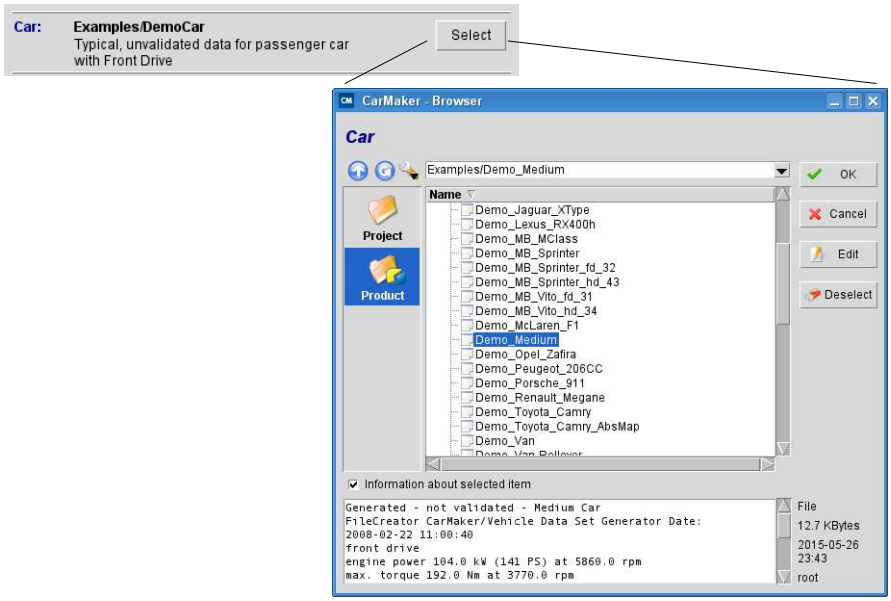


Figure 6.1: Changing the car

Now the “Medium“ car is displayed in the CarMaker GUI, and it will be used in the next simulation.

Save your TestRun and open IPGMovie.
Start the simulation.

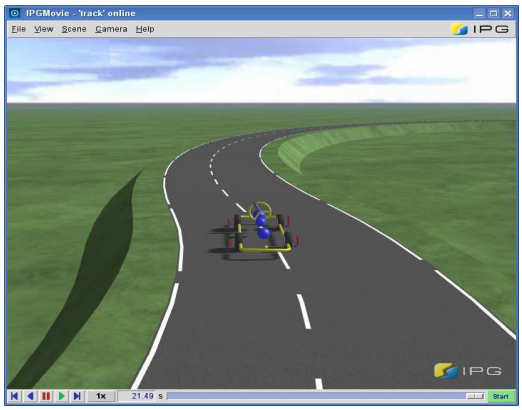


Figure 6.2: Starting the simulation

The animation in IPGMovie does not show any body any longer as no body has been defined for the vehicle data set “Demo_Medium“ to be displayed in IPGMovie.

By the way, this example shows you the difference between the data in the parameterization of the vehicle model, and what you see in IPGMovie: the body used in IPGMovie is just a file that can be changed.
E.g. we could display in IPGMovie the New Beetle used in the previous TestRun even if we simulate the Medium car. We will do that in the next chapter.

In the section ’Parametrizing the Vehicle and the Tires’ on page 59, we will learn how to parameterize the vehicle model according to your needs.

6.2 Using another Trailer Parameterization

CarMaker has a trailer model, too. You can parameterize it and choose the data set in the same manner as for the vehicle model.

For further information regarding the Trailer Parameterization please refer to the User’s Guide or the Reference Manual.

6.3 Using another Tire Parameterization

In the CarMaker GUI, click on the button “Select“ in the box “Tire“, choose the tire “MF_205_60R15_V91“, and then click on OK.

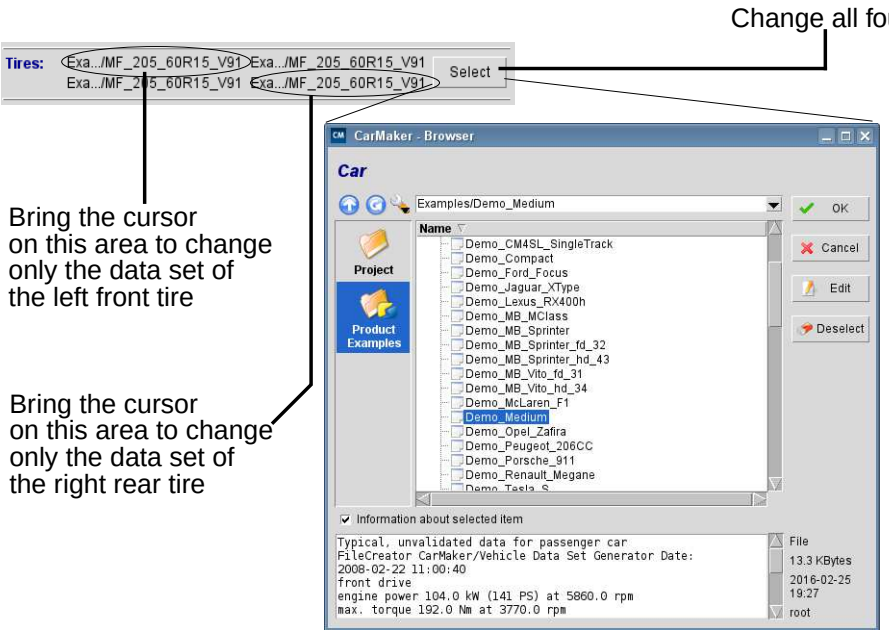


Figure 6.3: Changing the tires

Now, the new tire data set is displayed for the four wheels.

Optionally, you can change each tire data set separately in order to have different tires at the front and at the rear.

DO NOT go to the folder “Mappings”: you will find no usable file in it!

Bring the pointer on the “Select”- button shown on the picture above, and click there once. Select the data set “RT_195_60R15“.

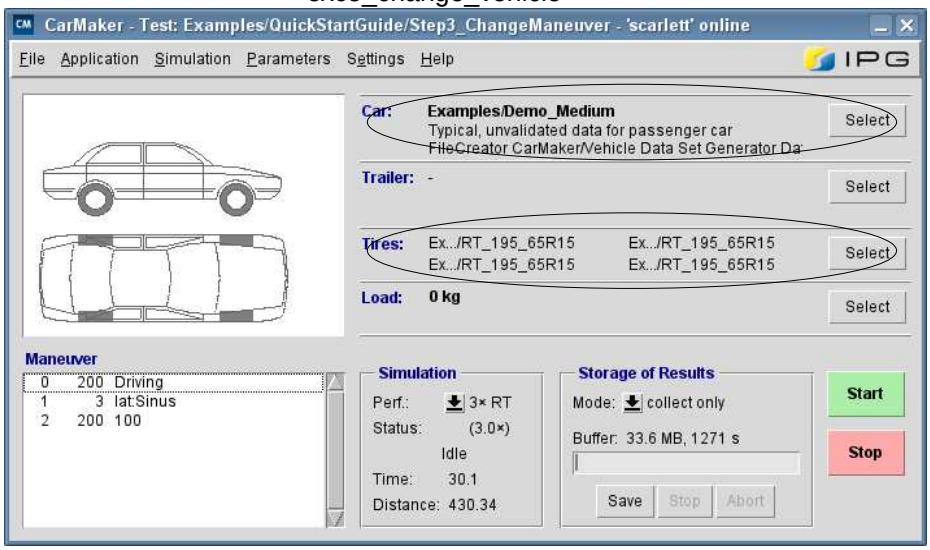
Bring the pointer on the area corresponding to the left rear tire, and click once here. Select the data set “RT_225_60R15“. Do the same for the right rear tire.

In the section 7.3 ’Creating a new Tire Data Set’ on page 79, we will show you how to create a new tire data set that corresponds to your needs. You will also discover which tire models are available in CarMaker.

6.4 Saving the new Data Sets within the TestRun

Save your TestRun ("exc5_change_vehicle").
Now open the TestRun "exc3_change_maneuver" and look at the differences.

exc5_change_vehicle



exc3_change_maneuver

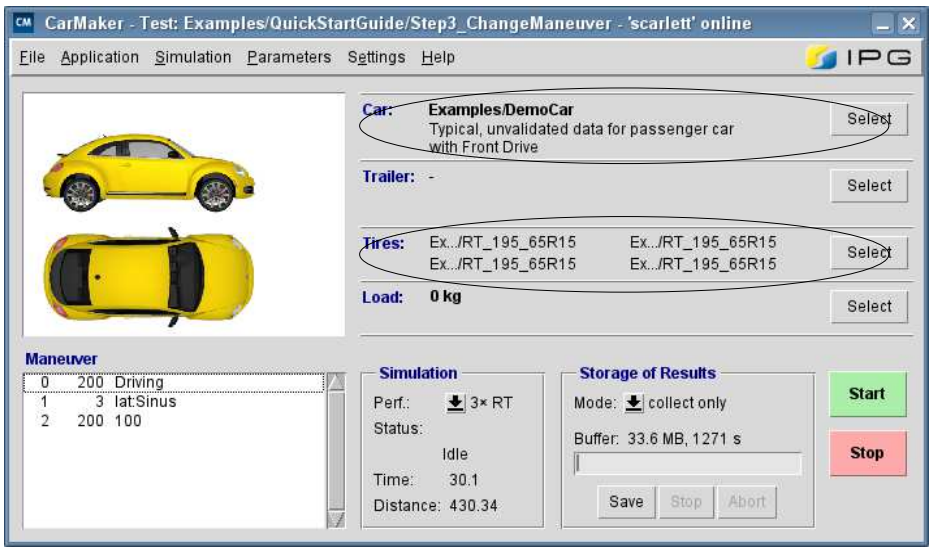


Figure 6.4: Comparison between two vehicles

This action shows you that you can build all possible combinations of tests. By just loading a TestRun you can retrieve a complicated test you have built before without having to build it again.

6.5 What is a Data Set?

You probably noticed that when you expect us to speak of “Vehicle Model“, we always speak of a “Vehicle data set“. Why is this?

For the simple reason that CarMaker has only ONE vehicle model, which has defined and fixed masses, DOF, spring/damper elements...

What we do in CarMaker is just to parameterize this *model* with various *data sets* according to the user’s needs. The data set contains the values required to parameterize the model, e.g the weight of the masses, the stiffness of the springs, ...

Next!

At this point, you can compare your work (saved under “exc5_change_vehicle”) with the pre-defined TestRun “Step5_ChangeVehicle” (located under Examples > QuickStart-Guide).

In the last chapter, you have learned how to change the data sets which parameterize the vehicle and the tire.

In the next chapter, you will learn how to create these data sets.

Chapter 7

Parametrizing the Vehicle and the Tires

7.1 Creating a new Vehicle Data Set

If necessary, open CarMaker and load the TestRun “exc5_ChangeVehicle“.

Save the TestRun as “exc6_GenerateVehicle”.

7.1.1 Save as a New Data Set

In the CarMaker GUI, click on *Parameters* > *Car* in order to open the vehicle editor.

In the vehicle editor, click on the menu *File* > *New*.

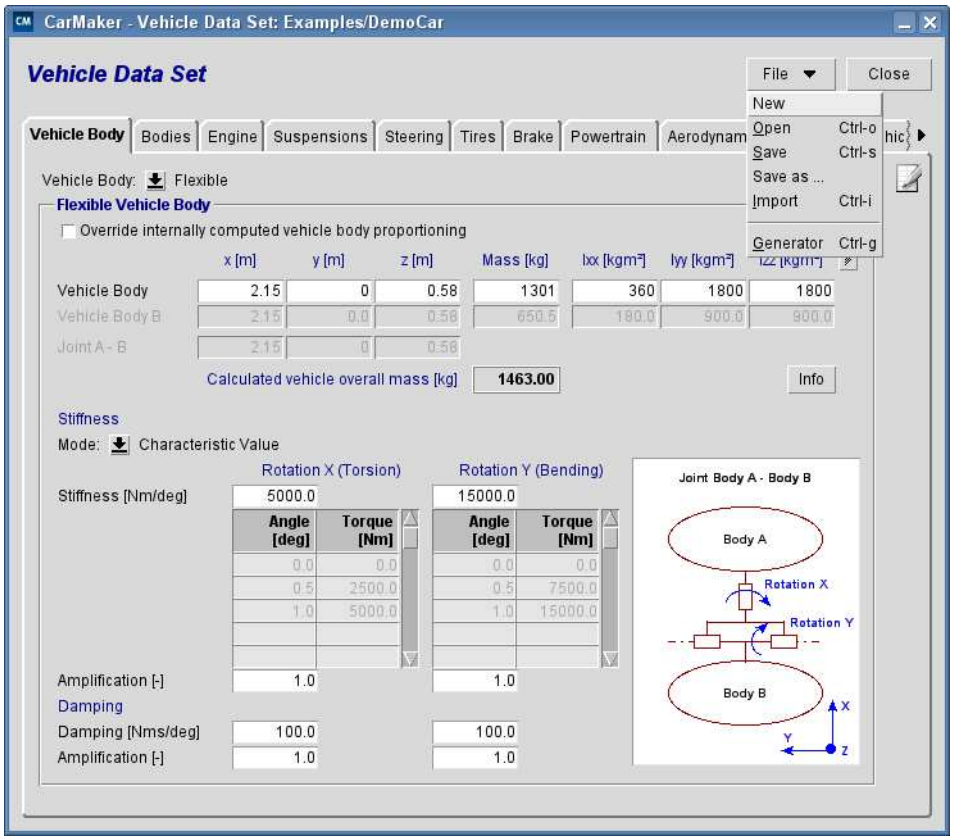


Figure 7.1: Vehicle Data Set dialog

With this action you have generated a default vehicle data set. Save it at once to keep it.

In the vehicle editor, click on the menu *File* > *Save As* > type the name of the new vehicle data set you are generating e.g. “my_Car” > *OK*.

The new data set is now saved.

Now we need to parameterize it correctly, since there are only default values filled in at the moment.

7.1.2 Using the Vehicle Generator

In the vehicle editor, click on the menu *File > Generator*.

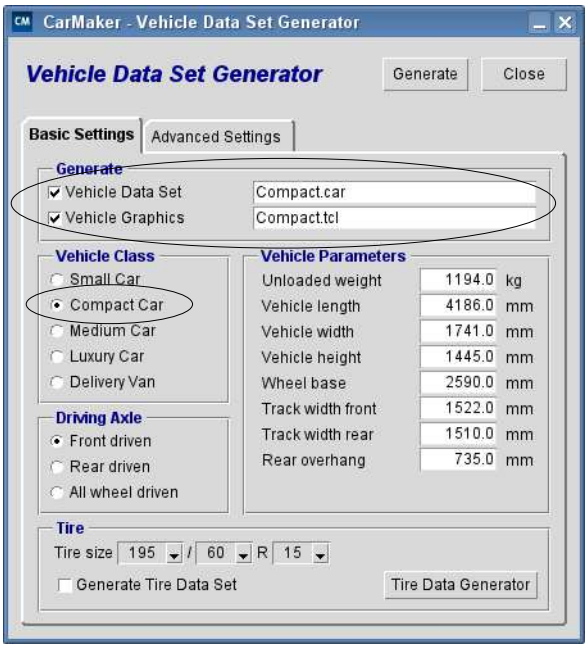


Figure 7.2: Vehicle Data Set Generator

The vehicle data set generator is a tool that automatically generates a plausible vehicle data set, based on the most important data that characterizes a vehicle.

Furthermore, you can directly select a vehicle class, e.g “Compact” in order to get the typical values corresponding to this kind.

Select the vehicle class “Compact Car“.
In the field “Vehicle Data Set“, write the name of the vehicle: “my_Car“ and click on “Generate”.
Close the Generator.

As you can see, the values written in the vehicle editor have changed.

We are now working with a new vehicle data set.

In the vehicle editor, click on the menu *File > Save*.

From now on, the new values are not only displayed in the editor, but also really saved to the corresponding file.

7.2 Parameterizing the Vehicle Model

The GUI used to parameterize the vehicle model pops up. Let us look at the parameters you have to specify. The following sub-sections (Main Body, Masses, ...) refer to the various tabs of the vehicle editor.

7.2.1 Vehicle Body

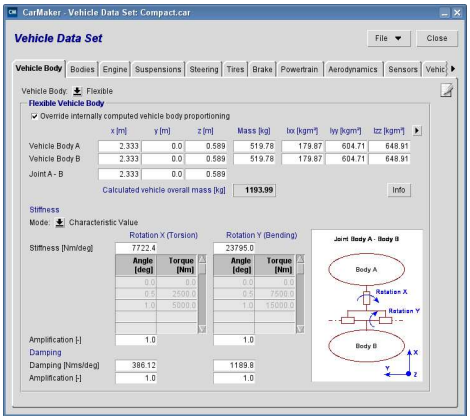


Figure 7.3: Choosing the type of vehicle body

Here, you can choose which vehicle model you want to use thanks to the option “Vehicle Body“:

- either a model with a flexible body,
- or a simple vehicle model without flexibility.

According to the model you have chosen, you must define for each body (A front and B rear):

- the position of the mass(es),
- the weight and the moments of inertia of the mass(es),
- optionally the stiffness of the vehicle body joint, it can be linear or non-linear,
- optionally the damping of the vehicle body joint.

The masses are defined in the design position, which means a fictitious state where no forces and weights are applied.

7.2.2 Bodies

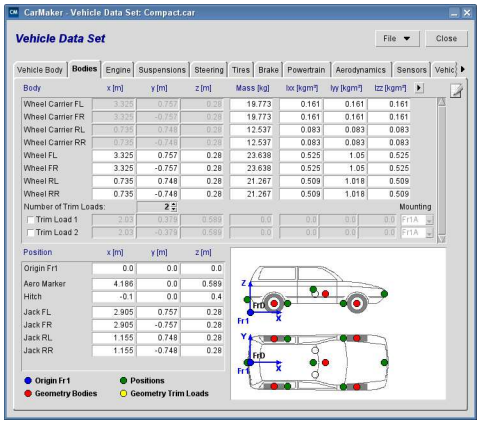


Figure 7.4: Body definition

In this tab, you define all other masses, still in the design position, according to the vehicle model chosen in the “Main Body“ tab.

- Engine: it can be placed either on the body A or the body B, or on both bodies (engine masses are optional, since the engine mass can be merged with the body mass).
- Wheels: unsprung spinning masses.
- Wheel carriers: unsprung masses that do not rotate.
- Trim Loads: additional loads that can be fixed on body A or B. They are available by clicking on the arrow as shown on the following picture:

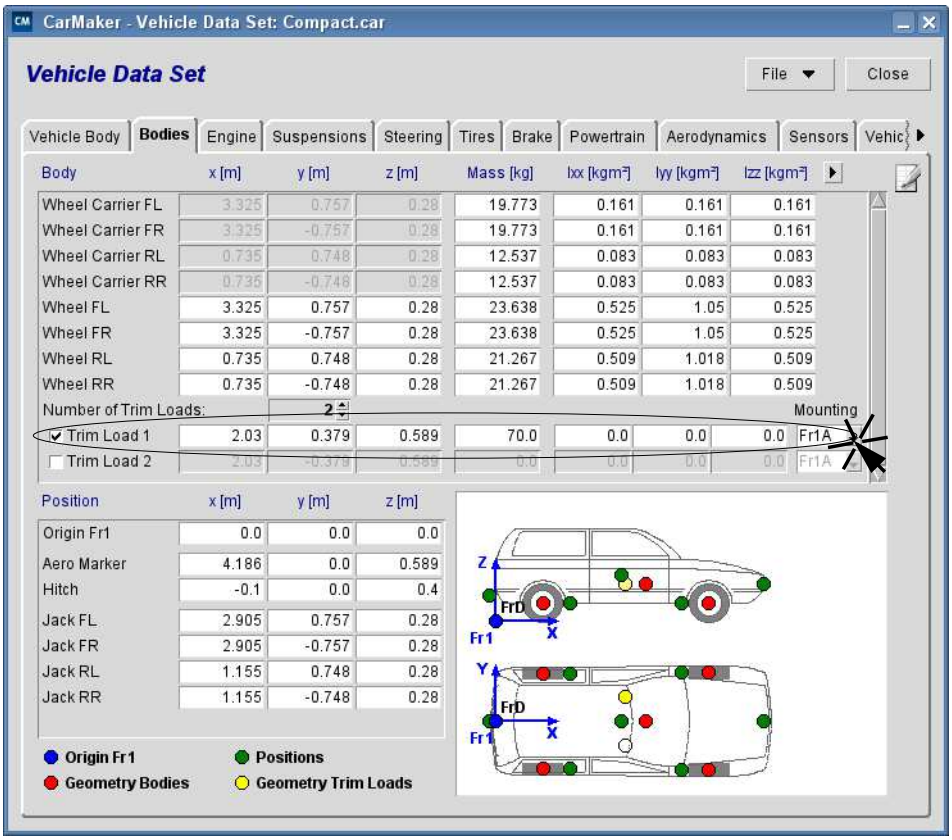


Figure 7.5: Trim Load definition

- Other fields:
  - Origin F1: enables to move the origin of the coordinate system of the vehicle.
  - Aero Marker: point where aerodynamic forces are applied.
  - Hitch: position of the hitch if you use a trailer.

7.2.3 Engine

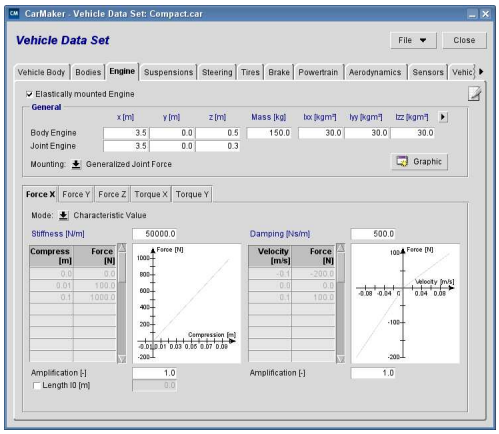


Figure 7.6: Engine definition

The CarMaker multi-body model also includes a separate engine body which is elastically mounted on the main body. Once activated, the position and engine mass needs to be defined as well as the location of the joint between engine and main body. The joint composes of a spring damper element which can be characterized in the lower part of the dialog.

7.2.4 Suspensions

Spring

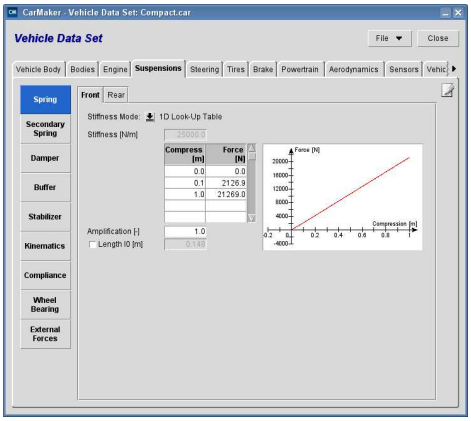


Figure 7.7: Spring definition

In the tab “Spring“ you define independently the spring stiffness and free length for the front and rear axle.

Secondary Spring

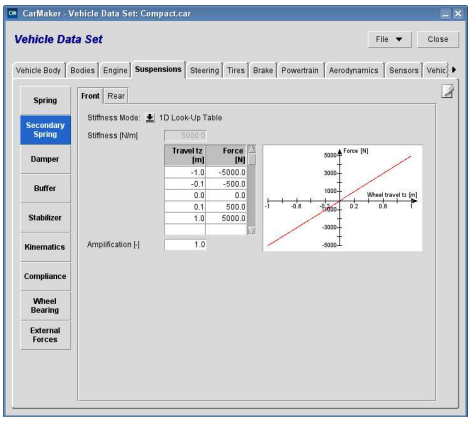


Figure 7.8: Secondary Spring definition

The tab “Secondary Spring” allows you to define a spring force, which describes the force generated by the suspension bushing torsion.

Damper

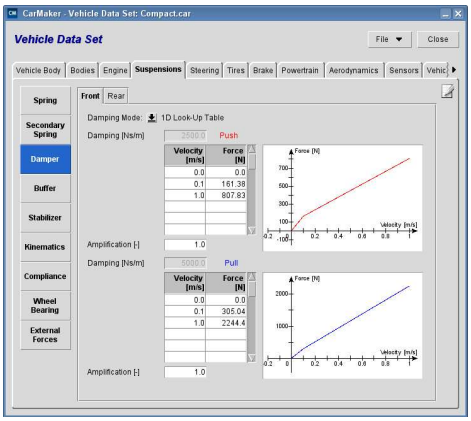


Figure 7.9: Damper definition

Here you define the damper characteristics, independently for the front and rear axle. Push and pull domain are separately parameterized.

Buffer

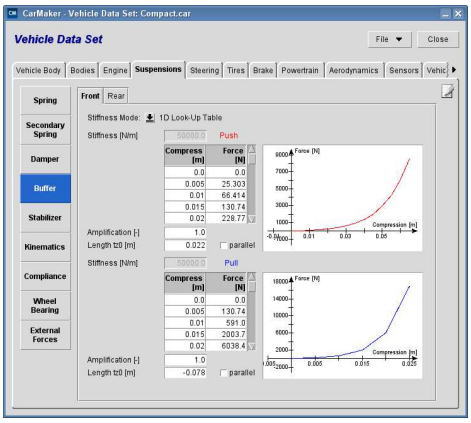


Figure 7.10: Buffer definition

The buffer enables to limit the wheel travel. This limit is given by the parameter “length tz0”.

If the wheel travel reaches the given distance (tz0), the buffer acts like an additional spring, which is parameterized in the tables.

Stabilizer

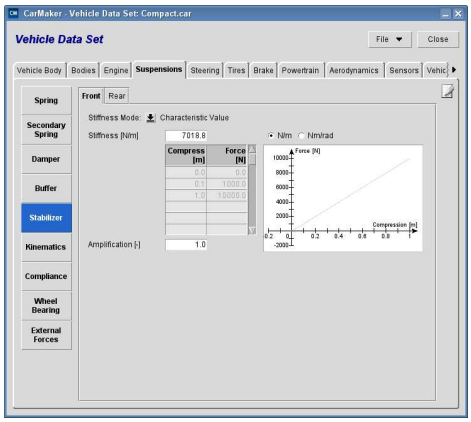


Figure 7.11: Stabilizer definition

The stiffness of the stabilizer is entered in N/m (approximation of virtual spring on small displacements).

Optionally, you can parameterize it in Nm/rad.

Kinematics

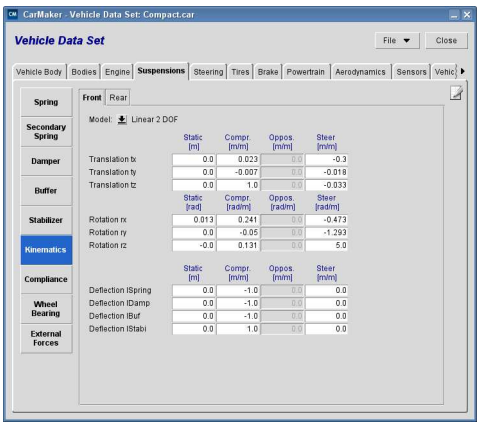


Figure 7.12: Kinematics definition

CarMaker offers a kinematics model, in which it is not necessary to parameterize the geometry of the vehicle axle.

For CarMaker, the only important information are the positions of the wheels according to the wheel travel, the steering rack displacement (for a front axle only) and the movement of the opposite wheel (in case of a rigid rear axle).

In the “Model“ option of the kinematics tab, you can choose between the following kinematics characteristics descriptions:

Mode	Description	Axle
Linear 1 DOF	Linear uni-dimensional kinematics. The kinematics only depend on the wheel travel. With this option, the kinematics are parameterized by the coefficients in the tables of the vehicle editor.	rear
Linear 2 DOF	Linear two-dimensional kinematics. The kinematics depend on the wheel travel and the steering rack displacement. With this option, the kinematics are parameterized by the coefficients in the tables of the vehicle editor.	front
	Linear two-dimensional kinematics. The kinematics depend on the wheel travel and the movement of the opposite wheel. With this option, the kinematics are parameterized by the coefficients in the tables of the vehicle editor.	rear
Linear 3 DOF	Linear three-dimensional kinematics. The kinematics depend on the wheel travel, the steering rack displacement and the movement of the opposite wheel (e.g. a steered rigid axle). With this option, the kinematics are parameterized by the coefficients in the tables of the vehicle editor.	front
external File	Non-linear kinematics. The kinematics are parametrized by the kinematics tables written to an external file.	front or rear

tx/ty/tz and rx/ry/rz define the translations and rotations of the wheel in the three directions:

	Static [m]	Compr. [m/m]	Oppos. [m/m]	Steer [m/rad]
Translation tx	0.0	0.023	0.0	-0.3
Translation ty	0.0	-0.007	0.0	-0.018
Translation tz	0.0	1.0	0.0	-0.033
	Static [rad]	Compr. [rad/m]	Oppos. [rad/m]	Steer [rad/rad]
Rotation rx	0.013	0.241	0.0	-0.473
Rotation ry	0.0	-0.05	0.0	-1.293
Rotation rz	-0.0	0.131	0.0	5.0
	Static [m]	Compr. [m/m]	Oppos. [m/m]	Steer [m/rad]
Deflection lSpring	0.0	-1.0	0.0	0.0
Deflection lDamp	0.0	-1.0	0.0	0.0
Deflection lBuf	0.0	-1.0	0.0	0.0
Deflection lStabi	0.0	1.0	0.0	0.0

Figure 7.13: Translations and Rotations

Parameter	Description	Unit (Compress)	Unit (Steering)
tx	wheel base variation	$m/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$
ty	track variation	$m/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$

Parameter	Description	Unit (Compress)	Unit (Steering)
tz	wheel travel variation	$m/m_{wheel\ travel}$ (it is of course always 1!)	$m/m_{steering\ rack\ dist}$
rx	camber variation	$rad/m_{wheel\ travel}$	$rad/m_{steering\ rack\ dist}$
ry	spin anglevariation	$rad/m_{wheel\ travel}$	$rad/m_{steering\ rack\ dist}$
rz	toe variation	$rad/m_{wheel\ travel}$	$rad/m_{steering\ rack\ dist}$
lSpring	Spring length variation	$m/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$
lDamp	Damper length variation	$m/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$
lBuff	Buffer length variation when activated	$m/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$
lStabi	Stabilizer length or angle variation (according to the unit you have written the stabilizer stiffness)	$m/m_{wheel\ travel}$ or $rad/m_{wheel\ travel}$	$m/m_{steering\ rack\ dist}$ or $rad/m_{steering\ rack\ dist}$

The columns stand for the following:

Column	Description
Static	The values in this column indicate the design static values, i.e when the wheel travel is null.
Compress	The values in this column give the variation of a parameter (e.g the camber angle rx, in rad), per meter of wheel travel.
Steering	The values in this column give the variation of a parameter (e.g the camber angle rx in rad), per meter of steering rack displacement. This column is used only for a front axle (Mode Linear2D).
Opposites	The values in this column give the variation of a parameter (e.g the camber angle rx in rad), per meter of wheel travel of the opposite wheel.

The axis are oriented as follows:

- X forward,
- Y on the left,
- Z upwards.

To parameterize an axle, you need to consider only the left wheel since CarMaker mirrors automatically the values for the other side.

A non-linear kinematics description is more accurate of course, but according to your needs a simple linear2D model might be sufficient.

The detailed explanation of the non-linear kinematics can be found in the User's Guide as well as in the Reference Manual.

Compliance

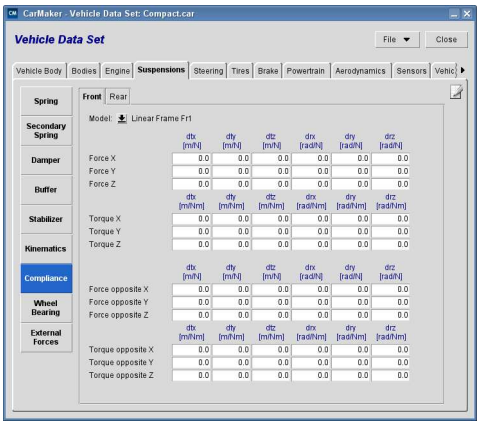


Figure 7.14: Compliance definition

The compliance description is optional:

- to activate the linear description of the compliance, choose the mode “CoeffConstrFr1“ (coefficient defined in the vehicle frame) or “CoeffConstrFr2“ (coefficient defined in the wheel carrier frame).
- to activate the non-linear compliance model, you have to write the compliance table in the same file as the one used for the linear kinematics. The explanation of the non-linear compliance is provided by User’s Guid and Reference Manual. The compliance triggers additional movements of the wheel depending on the forces applied at the wheel center.

The parameters in the tables of the editor are coefficients which define the variation of the wheel position in meter or rad per Newton (when a force is applied) or Newton-meter (when a torque is applied) at the wheel center, and possibly at the wheel center of the opposite wheel.

Wheel Bearing

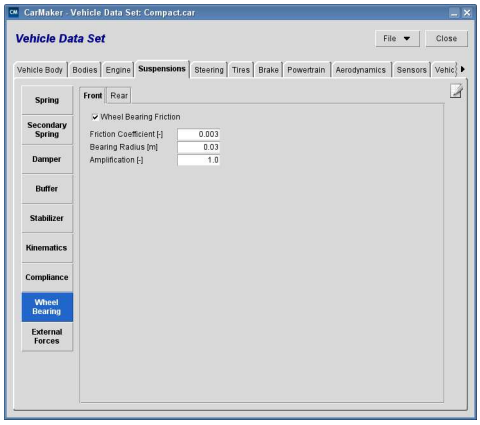


Figure 7.15: Wheel Bearing definition

This option activates/deactivates the consideration of the friction at the wheel bearings. If activated, other parameters can be visualized.

The wheel bearing friction is characterized with a single constant coefficient that defines the friction force on the bearing. The bearing radius enables to evaluate the friction torque on the bearing based on the friction force.

External Forces

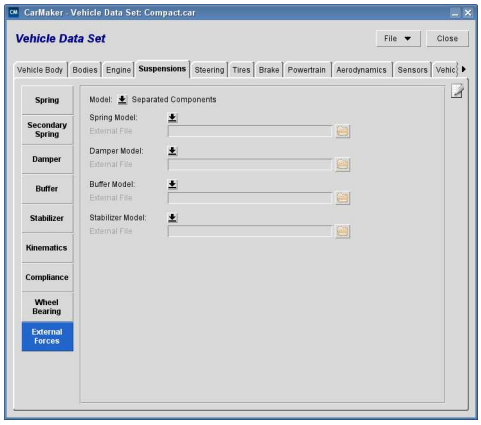


Figure 7.16: External Forces definition

This is useful only when the kinematics model of CarMaker is extended, e.g. when modelling an active stabilizer.

7.2.5 Steering System

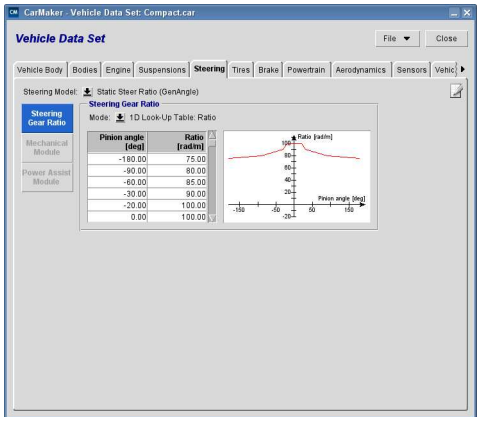


Figure 7.17: Steering System definition

The relation between the steering wheel angle or the steering wheel torque and the steering rack displacement is defined in this window.

This relation can be linear (Mode “Static“) or non-linear (Mode “Dynamic“).

The overall steering ratio, e.g. the ratio between the steering wheel angle and the wheel angle is calculated by CarMaker by multiplying the steering ratio “Rack to steering wheel“ with the kinematics parameter “rz“ (angle variation of the wheel at the z axis) in the column “Steering“ of the kinematics description.

An extended steering model called “Pfeffer with Power Steering“ is also available. This model provides a highly detailed mechanical model of the steering column, intermediate shaft, torsion bar and steering rack including damping and frictions effects. The power steering can be either hydraulic or electric.

7.2.6 Brake System

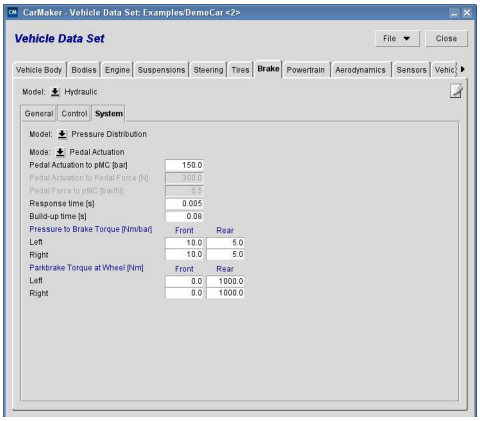


Figure 7.18: Brake System definition

The user has the choice between two hydraulic models:

- “PresDistrib“: it is a simple model. This model converts the force at the brake pedal to a pressure in the master cylinder and then to a braking torque at each of the four wheels. The parameter “Pedal Actuation to Pedal Force“ defines the force required by the driver to push the brake pedal to the maximum.
- External File > HydESP: You can use this model if you want to simulate an ESP controller model. With this hydraulic model, you can parameterize the hydraulic part of the brake system as well, which is specific to the ESP assembly.

7.2.7 Powertrain

Model

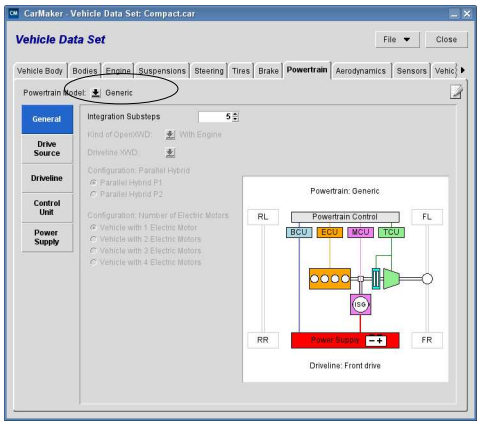


Figure 7.19: Powertrain Model definition

The option “Powertrain - Model“ selects the kind of torque generation. Generic would be a common combustion engine, the OpenXWD disconnects the gearbox output from the wheel to insert e.g. a generator or this model can even be used to replace the whole engine and drivetrain. The third option “AVL_CRUISE“ offers an interface to AVL CRUISE which is a powerful drivetrain modeler from AVL List GmbH.

The option “Driveline - Model“ enables you to choose the kind of transmission: front, rear, or any kind of 4WD.

Engine

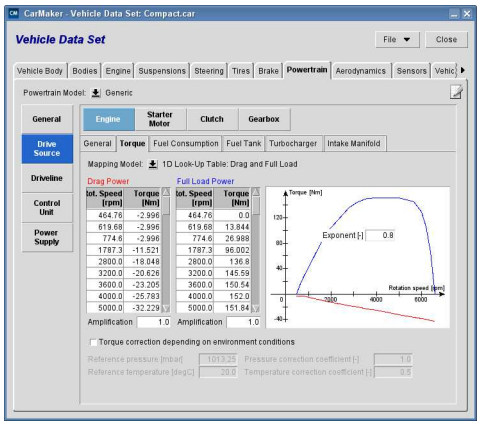


Figure 7.20: Engine definition

You define the torque characteristics of the engine: In the menu “Engine Model” choose Kind: “Look-Up Table“ and then in the menu “Engine Mapping” select Kind:“1D Look-Up Table: Drag and Full Load“.

This option lets you parameterize the full load power characteristic as well as the negative torque of the engine when slowing down.

Fuel Consumption

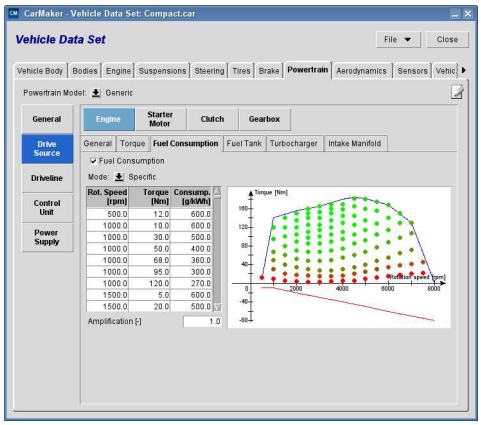


Figure 7.21: Fuel Consumption definition

Optionally, you can determine the consumption characteristic of your drivetrain.

Clutch

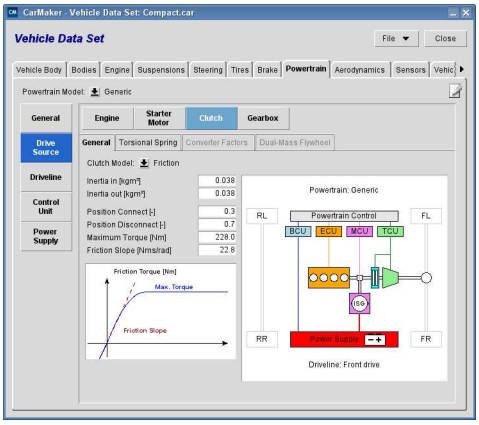


Figure 7.22: Clutch definition

CarMaker offers the possibility to parameterize a manual clutch, as well as a converter for automatic transmissions.

For the model “manual”, the parameters “Pos. Connect“ and “Pos. Disconnect“ enable to define at which position of the clutch pedal the clutch is fully connected or disconnected (1: pedal fully pressed).

Gear Box

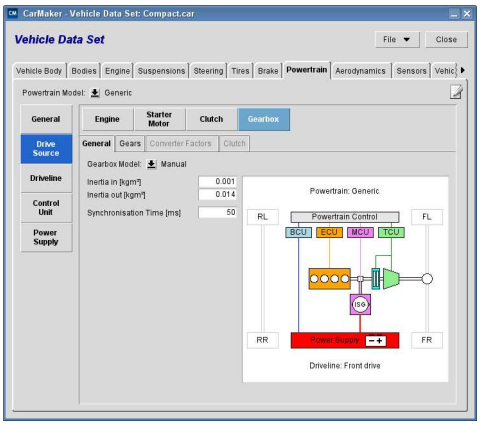


Figure 7.23: Gear Box definition

Here you can parameterize the gear ratios. Two models are available: a manual gearbox as well as an automatic one.

Driveline

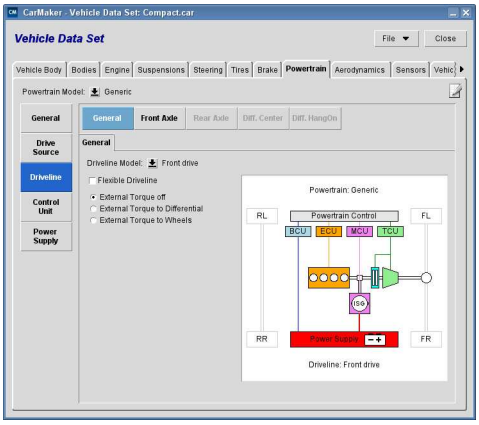


Figure 7.24: Driveline definition

In CarMaker you have the possibility to choose, if your car is front, rear or all wheel driven. You also have the option to select a shiftable all wheel drive and to determine if you want your driveline to be considered flexible or stiff.

Differentials

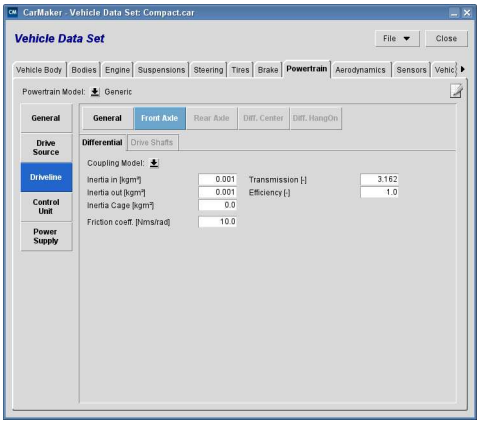


Figure 7.25: Differentials definition

According to the transmission mode you chose, you have to parameterize various differentials.

The differential models are the same. Default “Mounting“ is “left to right”, other options are for very specific use.

If you wish to have a very simple differential coupling model, as common on most European cars, choose “Coupling Model” = “not specified“.

Please note that modeling torque vectoring is possible by choosing the option “Coupling Model” = “Torque Vectoring“. Then, you can manipulate the torque distribution by DVA (see section ’Additional Information about CarMaker’ on page 104).

7.2.8 Aerodynamics

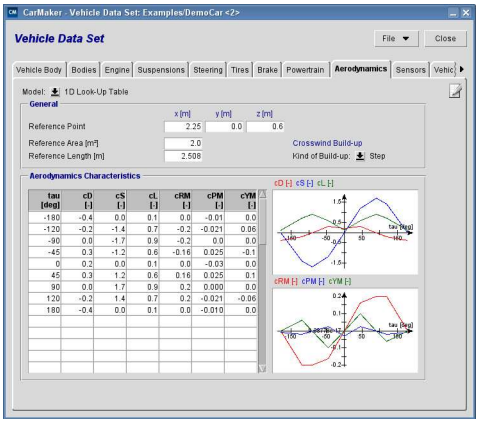


Figure 7.26: Aerodynamics definition

The aerodynamic model applies additional forces and torques depending on the vehicle speed and wind direction at the point “Aero Marker“. This point is defined in the “Bodies“ tab.

The explanation of the coefficients can be found in User’s Guide and Reference Manual.

7.2.9 Sensors

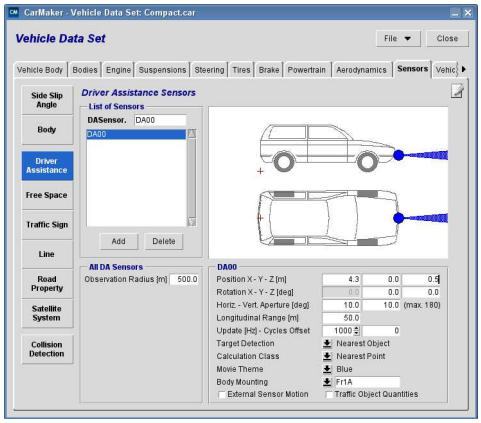


Figure 7.27: Sensor definition

The sensor model enables to place nine different sensor types on your car:

- SideSlipAngle Sensors (as reference where the side slip angle should be measured),
- Body Sensor (sensors to measure accelerations and translations at certain points on the car),
- DA Sensors (for driver assistance systems),
- Free Space Sensors (sensors to detect free Space in different directions),
- Traffic Sign Sensor (for traffic sign recognition),
- Line Sensor (sensor which detects road markings),
- Road Property Sensors (sensors to determine at a predefined preview distance important road specific attributes),
- Satellite System (simulates the positions of satellites and a receiver build in the vehicle),
- Collision Detection Sensor (recognizes collisions with traffic objects).

7.2.10 Vehicle Control

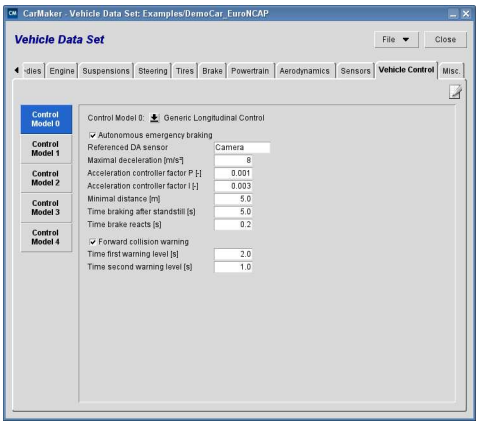


Figure 7.28: Vehicle Control selection

In the Vehicle Control tab you can select controller models like driver assistance models. A maximum of five Vehicle Control models can be used in one vehicle. Available example controllers are the following:

- "Acceleration Control + ACC"
  Example for an active cruise control including an acceleration controller to transfer the target acceleration to a gas / brake pedal position.
- "Generic Longitudinal Control"
  The example longitudinal controller comes with an autonomous emergency braking assistant and a forward collision warning algorithm.
- "Generic Lateral Control"
  The example lateral controller consists of a lane keeping assistance system and a lane departure warning.

7.2.11 Misc.

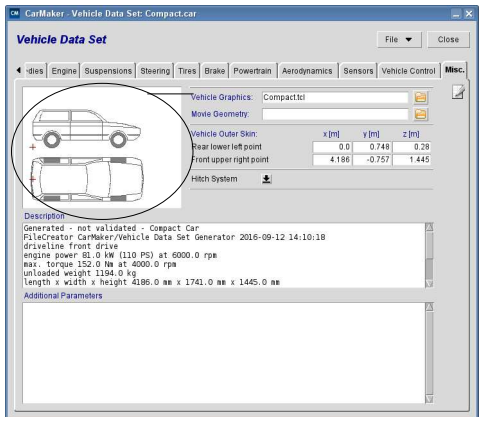


Figure 7.29: Misc. definitions

Here you can optionally choose a picture in the supported JPEG or PNG formats to be displayed in the CarMaker GUI and in the vehicle editor. You can also create your own picture by making snapshots of the three dimensional object file in IPGMovie.

The object file used by IPGMovie can be selected under “Movie Geometry“. You can create this file by yourself with the help of your preferred CAD software, or buy one on the web. You must choose a file in the .obj or .3ds format.

- File definition:

- ONLY polygons and triangles, NO splines or free forms,
- no more than 50 000 nods,
- if this option is available while generating the file: generate the normal for all points,
- colors: generate the corresponding material file.

The field “Additional Parameters“ enables to add some extra parameters optionally.

7.2.12 Using the Import Feature

Before closing the vehicle editor, let us have a look at an unique feature of CarMaker.

When you build your vehicle data sets, you may need to parameterize some similar vehicles, which have a part of their parameterization in common.

With CarMaker it is possible to import every part of a data set from another data set separately.

Let us practice it: Go to the vehicle data set of “my_Car” and import the front axle from the DemoCar.

In the vehicle editor, click on the menu *File > Import* and select the modules “Kinematics front“ and “Misc.” and click on “Import“. In the dialog “Import Vehicle Data Parameters from...” choose the “DemoCar”.
Then save your vehicle data set (“my_Car”): *Vehicle Editor > File > Save*

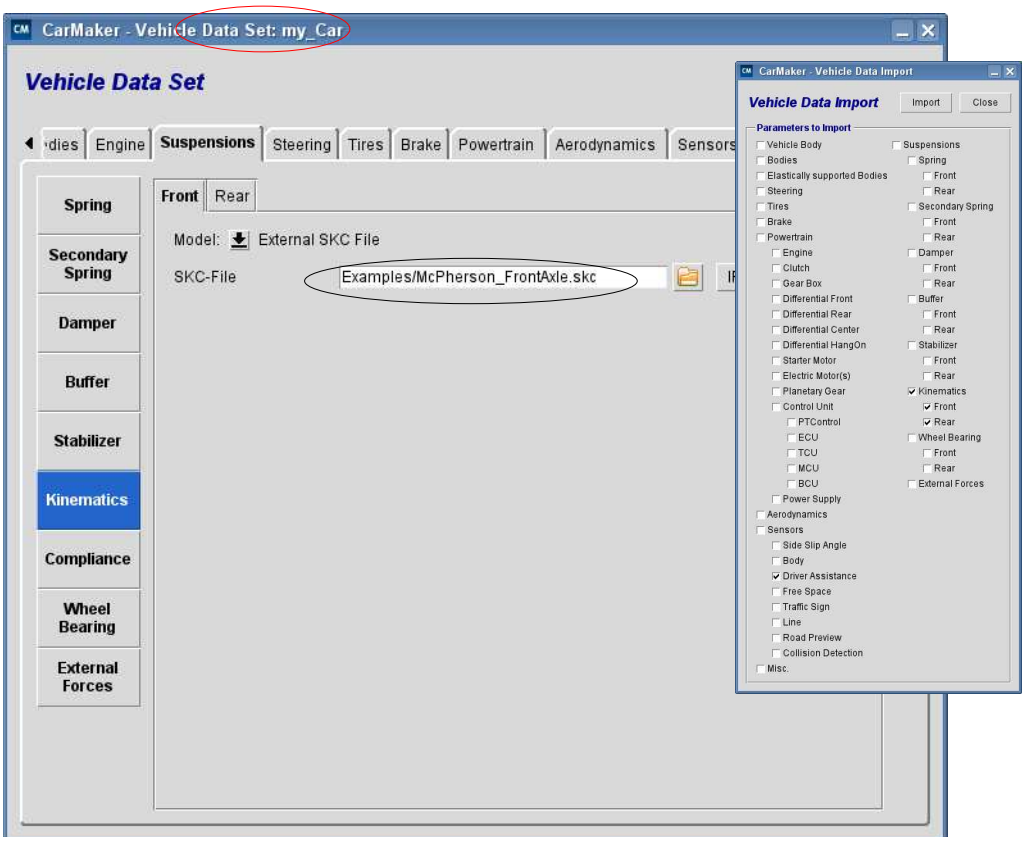


Figure 7.30: Vehicle Data Import for “my car”

Your vehicle “my_Car” now uses the same non-linear kinematics description as the “Demo Car”, the same picture, and the same object file: when you start the simulation again, you will now see the yellow “New Beetle”.

Save your vehicle and save your TestRun.

7.3 Creating a new Tire Data Set

In CarMaker you can use the following different tire models:

- Real Time Tire (RTTire), based on the TYDEX format
- Pacejka 5.2
- Michelin TameTire

RTTire

TYDEX is a common format used to save the characteristics measured on a tire. With the help of a tool, you convert the TYDEX file into a binary file and, at the same time, you generate an “Infofile“ that points to the binary file.

Finally, you load the Infofile in the CarMaker GUI.

As soon as you have the characteristic curves of your tire (e.g. from measurements or from the tire manufacturer), you can generate the corresponding RTTire file for CarMaker on your own.

Pacejka 5.2 and ADAMS property file

CarMaker supports the standardized Pacejka format, as well as the ADAMS Property file.

In the first case, if you have the Pacejka parameters, you just need to write them to a file, and load this file in the CarMaker GUI.

Michelin TameTire

CarMaker provides an interface to the thermo-mechanical tire model from Michelin. This model is no part of the CarMaker evaluation package. Please contact the CarMaker Service Team for further information (CarMaker-Service@ipg.de).

In the CarMaker GUI, click on the button “Select“ in the tire box, click on the tire MF_205_60R15_V91 and then click “Edit”.

This is an example of a Pacejka file, in which you can write directly the values of your parameters.

In the CarMaker GUI, click on the button “Select“ in the tire box, click on the tire MF_205_60R15_V91r and then click “Edit”.

This is an Infofile that points to an ADAMS property file (see tab Model Parameters > Import Adams Property File).

Open the ADAMS property file: Use a text editor to open the file from your CarMaker project folder under Data > Tire > Examples > Pacejka > MF_205_60R15_V91.tir.

You have just opened the same Pacejka data set as previously, but this time it is written with the ADAMS format.

Please note that an internal mechanism enables to erase the instability at standstill caused by the Pacejka model.

The changes done in this section must not be saved: load again "exc6_GenerateVehicle".

Please note: Each CarMaker example vehicle contains a suitable tire data set. You only need to select tires in the main GUI, if you would like to change the tire configuration for this test.

Next!

At this point, you can compare your work (saved under "exc6_GenerateVehicle") with the predefined TestRun "Step6_GenerateVehicle" (located under Examples > QuickStart-Guide).

After having done some changes to the original configuration ("Step1_Acceleration"), we extended a TestRun ("exc6_GenerateVehicle", which should be the same as "Step6_GenerateVehicle"). Now let us generate a completely new TestRun from scratch within 5 minutes!

Chapter 8

Building your TestRun from Scratch

Step by step, you have unveiled the world of CarMaker so far.

Now, let us sum up how to build the same TestRun as you created in the section 'Extending the Test Scenario Definition' from scratch.

8.1 Step 1: Open CarMaker and Create a TestRun

If necessary, open CarMaker.

Click on the Windows Start button > Programs > IPG > CarMaker.

Create a new TestRun.

In the CarMaker GUI, click on File > New.

Save at once under "my_second_TestRun" the TestRun you are going to build.

In the CarMaker GUI, click on File > Save As > my_second_TestRun > OK.

8.2 Step 2: Load the Vehicle

Load the vehicle data set “my_Car“ which you created in section 7.1.2 ’Using the Vehicle Generator’ on page 61.

In the CarMaker GUI, click in the box “Car” on the button “Select“ > my_Car > OK.

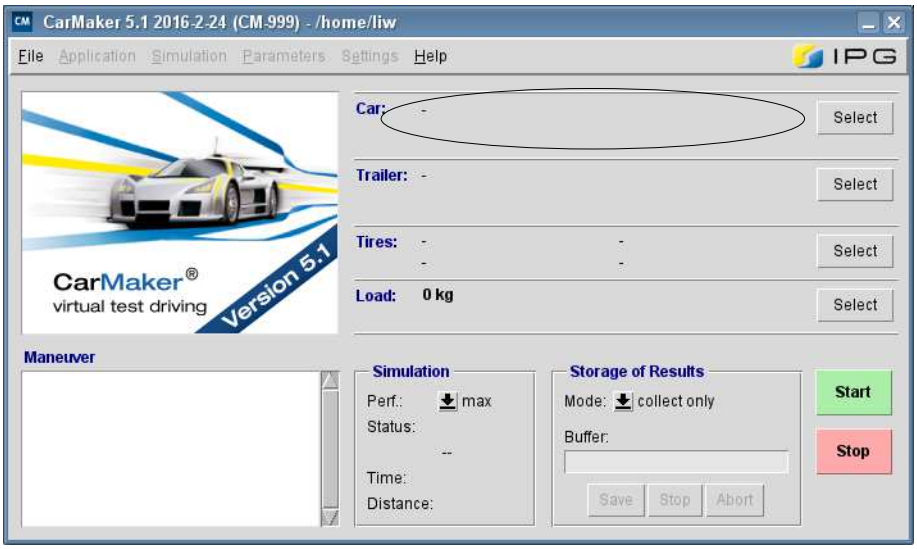


Figure 8.1: Selecting a car via the CarMaker GUI

8.3 Step 3: Putting the Vehicle on a Road

Define a segment based road as follows:
In the Road GUI click the right mouse button > Import > Examples > QuickStartGuide > Step2_ChangeRoad > OK.

To open the Road GUI: in the CarMaker GUI, click on Parameters > Road.

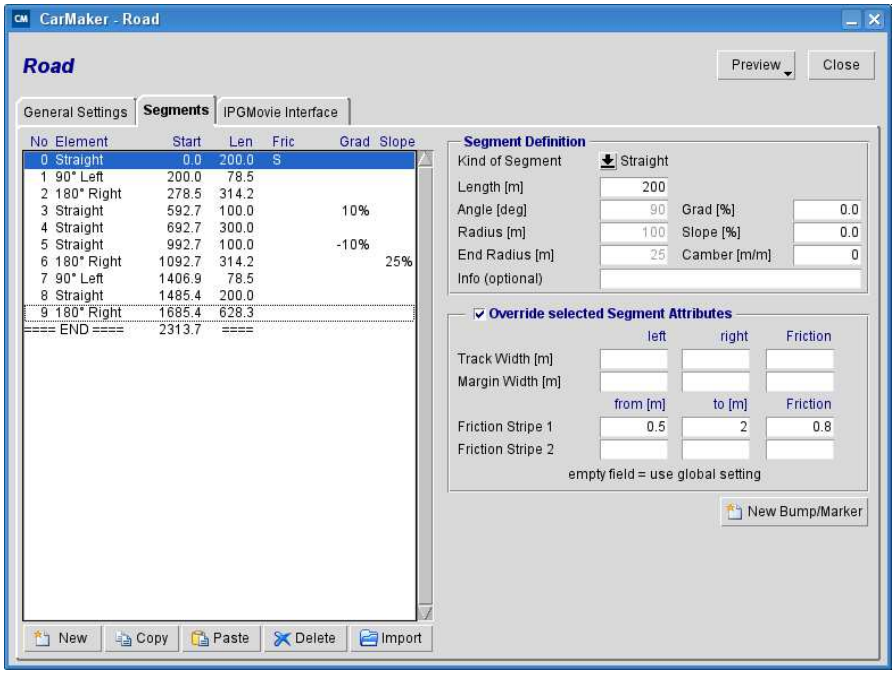


Figure 8.2: Defining a segment based road

8.4 Step 4: Controlling the Vehicle (Maneuvers)

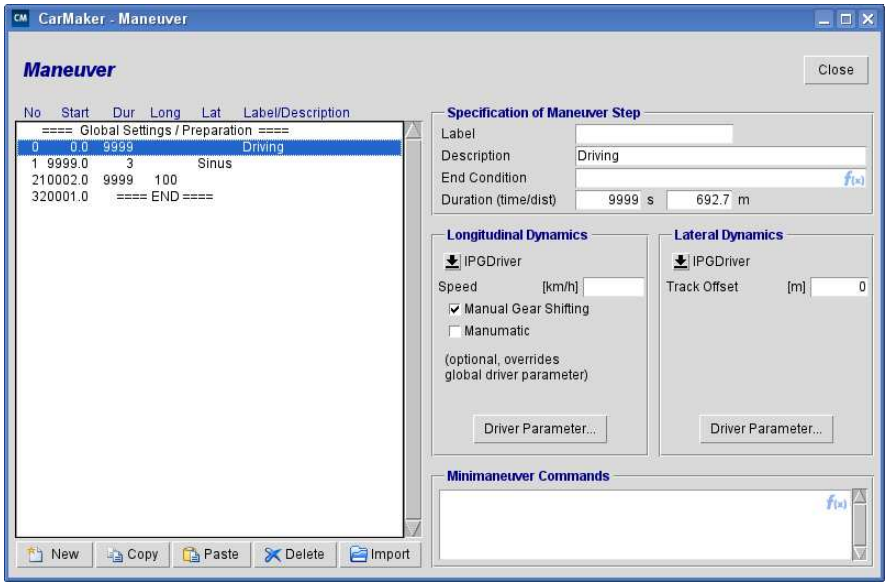


Figure 8.3: Maneuver window

Define the maneuvers as follows:
In the Maneuver GUI, click on “Import“ at the bottom of the window > exc3_ChangeManeuver > OK.

To open the Maneuver GUI: in the CarMaker GUI, click on Parameters > Maneuvers.

8.5 Step 5: Saving the TestRun and Simulating

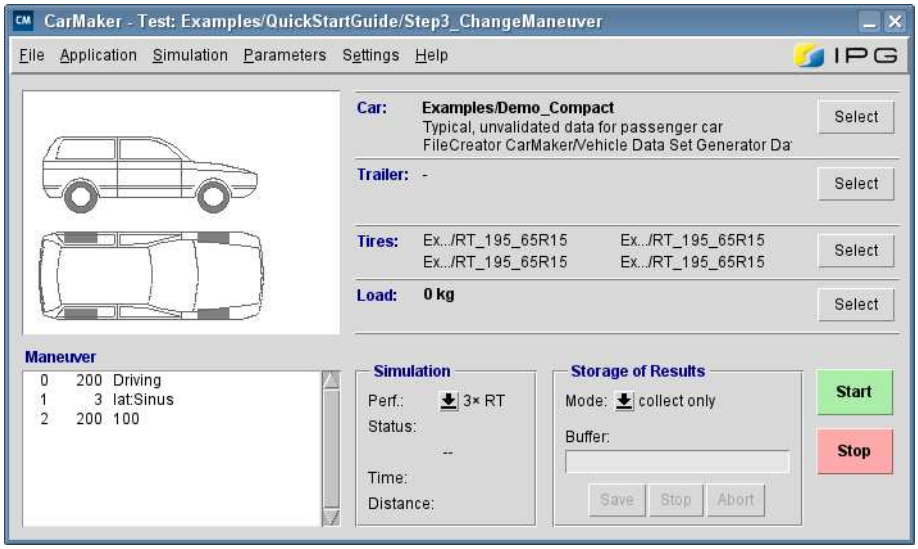


Figure 8.4: Saving the TestRun

Save your TestRun.

In the CarMaker GUI, click on File > Save.

In the CarMaker GUI, click on Application > Start & Connect.

This action enables to initialize the simulation environment (useful for IPGControl).

Open IPGMovie, Instruments, and IPGControl.

All these tools can be found by clicking in the CarMaker GUI on File > [name of the tool]

In IPGControl, plot the vehicle speed during the simulation (Quantities > Car[P..Y] > Car.v).

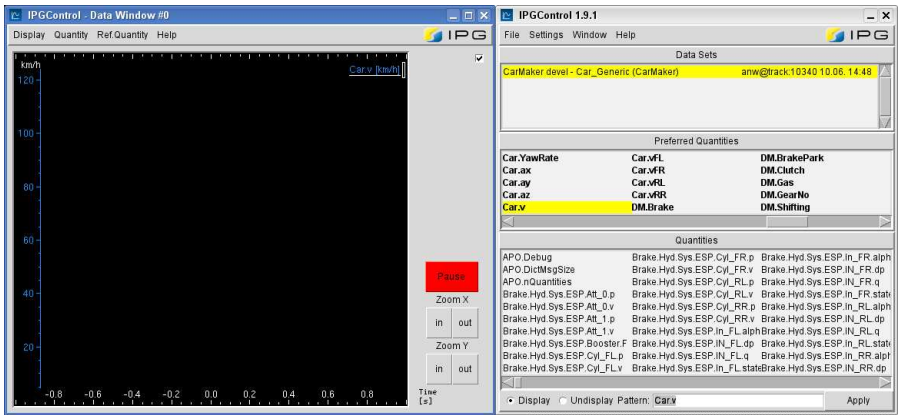


Figure 8.5: IPGControl Data and Selection window

Start the simulation.

8.6 Final Results

As you see, in 5 minutes you managed to repeat what you learned during the last 3 chapters!

Loading a TestRun, parameterizing your environment, changing the tire, defining the maneuvers, creating a new vehicle data set, analyzing the results, watching a movie of your simulation... All that takes you only a few clicks!

Next!

If you wish, and if you have the right software installed on your PC, you can also use CarMaker in a MATLAB/Simulink environment.

The next chapter explains you what you need to know about CarMaker with Simulink.

Chapter 9

Simulating with CarMaker and Simulink

9.1 Starting CarMaker with Simulink

Open CarMaker via the Windows Start Button and create a new project (CarMaker GUI > File > Project Folder > Select / update project) and toggle on the option "CarMaker for Simulink Extras".

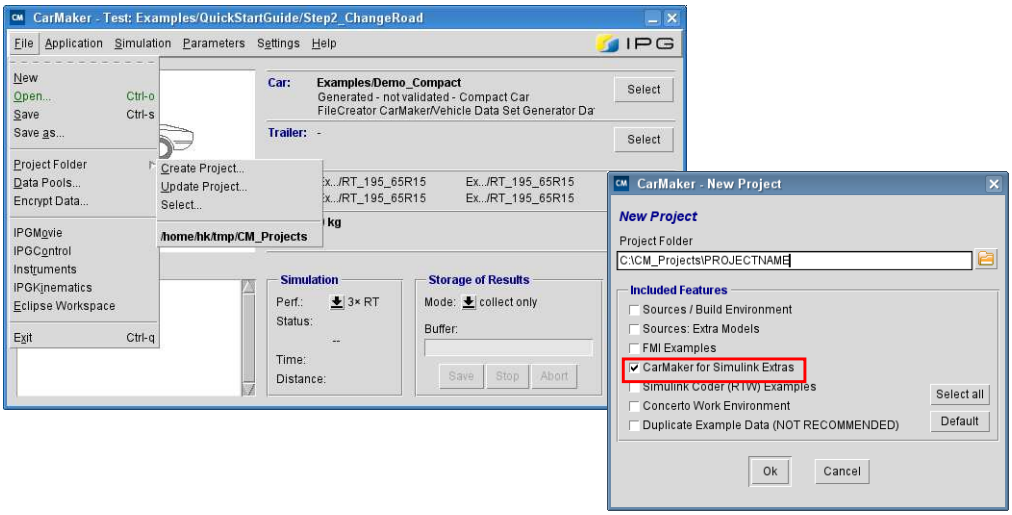


Figure 9.1: Creating a new project with "Carmaker for Simulink Extras"

Now a project folder is created that contains an additional folder called "src_cm4sl" which provides the interface to Simulink.

Close the CarMaker GUI and open Matlab. As working directory, select the folder “src_cm4sl” of your new project. Open the generic.mdl model.

Now, the Matlab window should have opened followed by the Simulink environment and the Simulink model “generic.mdl”. A successful start-up procedure leads to the following window:

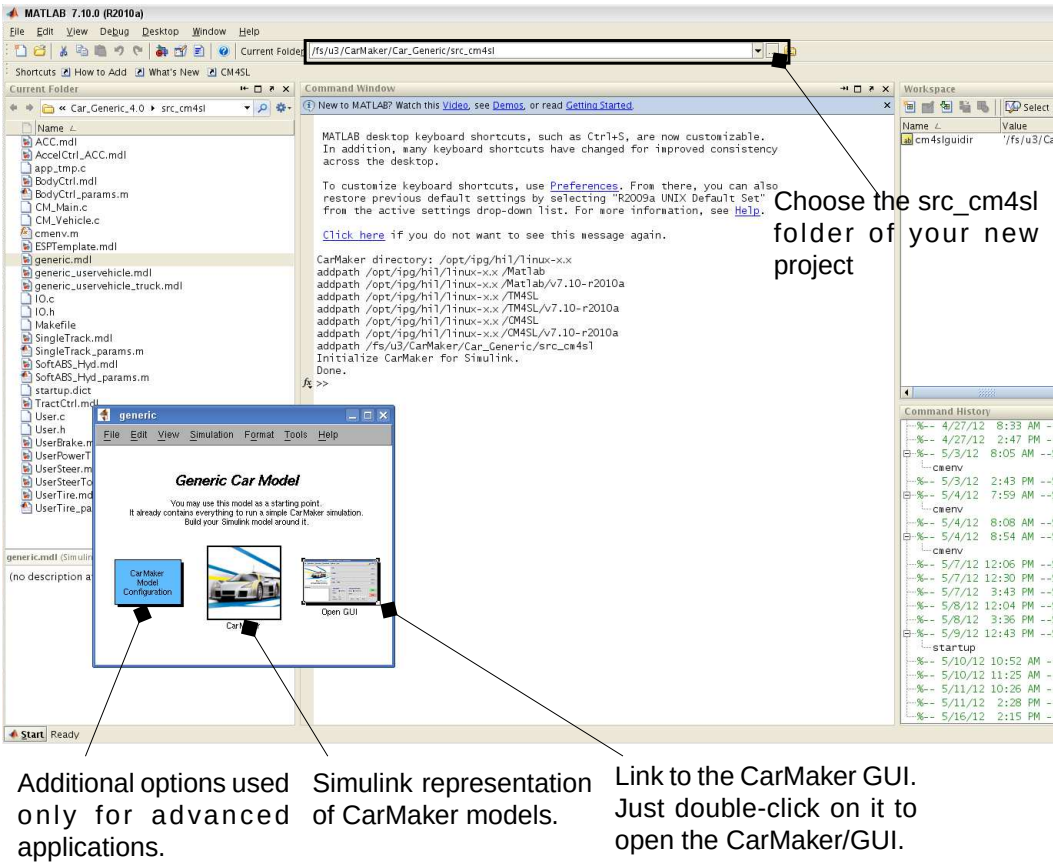


Figure 9.2: MATLAB window with Simulink model “generic.mdl”

Troubleshooting

Sometimes, the file type .mdl is not associated with Matlab. A new dialog appears which asks you to open the file with one of the following programs.

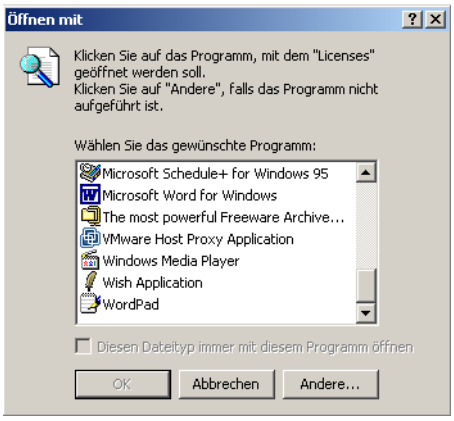


Figure 9.3: “Open with...” dialog window

Please search for your Matlab executable and confirm the execution.

Start-up confirmation

Matlab and CarMaker for Simulink work together by extending Matlab's search path, so that Matlab knows where to find the CarMaker for Simulink blockset and auxiliary commands. The Matlab search path is extended by the execution of a small script named cmenv.m, which is located in the mdl subdirectory "src_cm4sl" of the CarMaker project directory. The execution of this script may be done manually, but there is also a way to invoke it automatically each time Matlab is started or a model is loaded.

A golden rule is: Always keep the cmenv.m script in the same directory as your Simulink model. Whenever you load a Simulink model which contains the CarMaker subsystem block from the CarMaker for Simulink blockset, cmenv.m will be executed automatically. This default behavior should be sufficient for most uses of CarMaker for Simulink.

The cmenv.m script should generate a message like the following in your Matlab console window:

```
CarMaker directory: c:/ipg/hil/win32-x.x
addpath c:/ipg/hil/win32-x.x/Matlab
addpath c:/ipg/hil/win32-x.x/Matlab/vx.x-r20xxx
addpath c:/ipg/hil/win32-x.x/CM4SL
addpath c:/ipg/hil/win32-x.x/CM4SL/vx.x-r20xxx
addpath <x:/path/to/your/project_dir>/src_cm4sl
Initialize CarMaker for Simulink.
Done.
```

9.2 First simulation with CarMaker for Simulink

The generic.mdl example is intended as a start-up example model and provides a basic CarMaker subsystem (similar to the stand-alone version) without any additional control blocks. More details of this system will follow in the next section.

Double-click on the “Open GUI“ block, the CarMaker windows pop up:

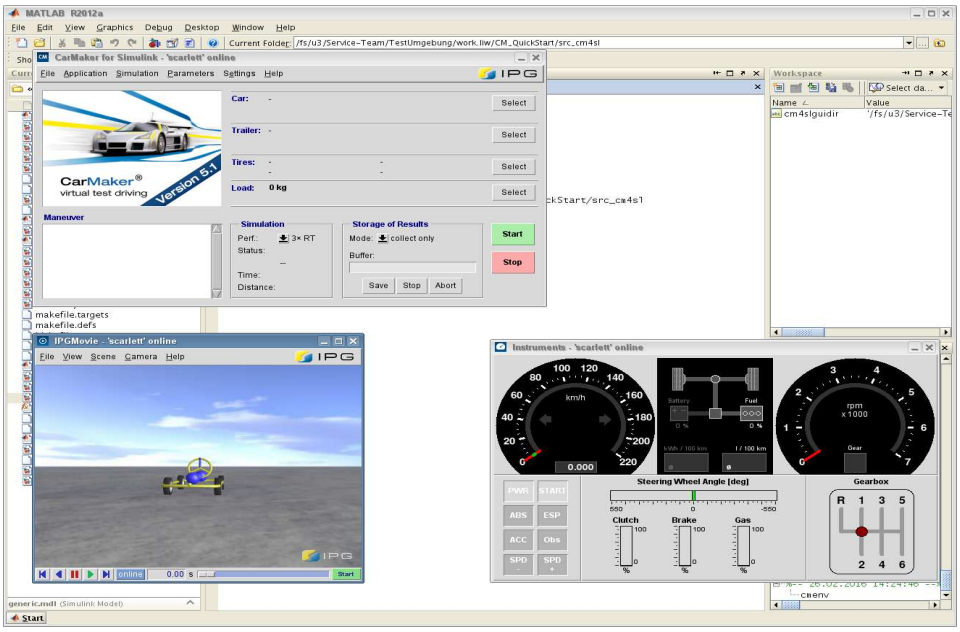


Figure 9.4: CarMaker GUI called up from MATLAB

You find the same GUIs as in the previous chapters, where you will find information about loading a TestRun, changing a vehicle, etc.

Load the TestRun Examples/CarMakerFunctions/IPGRoad/Hockenheim and start the simulation.

As you see, the environment is very similar as shown previously. Even under Simulink, you do not need to prepare a special Simulink model.

When you load the model “Generic“, you just load the simplest vehicle model (this time, we speak of a model, not of the data set!), which is in fact the counterpart of the model available in the stand alone version of CarMaker.

Next!

The CarMaker for Simulink is ready to work! Now, please read the entire section ’Performing your First Simulation within 5 Minutes’ on page 11 to know how to perform your first simulation in 5 minutes. If you have already done this part, you can switch to the next section.

In the following section, you will learn what are the basic differences between CarMaker stand-alone and CarMaker for Simulink.

9.3 Description of the CarMaker Environment in Simulink

CarMaker with Simulink is a complete integration of IPG's vehicle dynamics simulation software CarMaker, into the MathWorks modeling and simulation environment, Matlab/Simulink.

Thus, in CarMaker for Simulink, you will be able to extend the vehicle *model* itself.

The highly optimized and robust features of CarMaker were added to the Simulink environment using a S-function implementation and the API functions that are provided by Matlab/Simulink.

CarMaker for Simulink is not a loosely coupled co-simulation but a closely linked combination of the two best-in-class applications, resulting in a simulation environment that assures both good performance and stability.

Because of this integration, it is now possible to use the power and functionality of CarMaker in the intuitive and full-featured environment of Simulink. And using CarMaker in Simulink is no different than using standard S-function blocks or built in Simulink blocks. The CarMaker blocks are connected the same way other Simulink blocks are connected, and existing Simulink models can now easily be added to the CarMaker vehicle model with literally a few clicks of the mouse.

Integration does not, however, mean a loss of functionality, as all the features that make CarMaker the premier software in its domain have been included, and can now be used together with Simulink's rich tool set. The CarMaker GUI can still be used for simulation control and parameter adjustments, as well as defining maneuvers and road configurations. IPGControl can still be used for data analysis and diagram plotting. IPGMovie can still be used to bring the vehicle model to life, with realistic animation and rendering of the multibody vehicle model in three-dimensional space.

In short, CarMaker for Simulink is not a stripped down version of the original, but a complete system that can become a part of any Simulink simulation quickly and easily. The next steps will show you just how easy it can be.

9.4 Advantages and Disadvantages using CarMaker with Simulink

Feature	CarMaker - standalone	CarMaker running in Simulink	Comment
Simulation Speed	Very fast.	Slow, especially with additional models.	The simulink environment needs a lot of processing power. The best performance is reached when the simulink model "generic.mdl" is used. Even then, there is a big difference in simulation speed. Even little add-ons to the model can lead to performance loss.
Flexibility	Full flexibility.	Limited flexibility, every implemented model will be simulated by Simulink even when it is not needed.	The c-code offers the opportunity to use parts of the simulation only when they are needed (e.g. additional submodels, dynamic traffic objects).
Exchange of models	Easy, all global variables accessible in the c-library are available.	Very easy due to the graphical interface, but only the interface variables and User Accessible Quantities are available.	To use the interfaces of the stand alone version, c-code knowledge is required.
Manageability	Very good. C-Code is ideal to be structured. Creating libs and objects, you can also share models easily.	Very good up to a middle diversity level. From there on models become harder and harder to manage.	The c-code is mostly well arranged and is better to understand than the complex simulink models.
Graphical modeling	Less comfortable, only using RealTime-Workshop.	Very easy.	Especially for beginners and people without c-code knowledge, it is a lot easier to extend CarMaker in Simulink.
Value for Money	Very good.	Same as stand alone + additional costs coming with Matlab/Simulink.	Both variants are available in the CarMaker/Office package. For the Simulink version you need Matlab and Simulink which cost a few thousand Euros in addition.

Chapter 10

Features of the CarMaker for Simulink blockset

You may find the CarMaker for Simulink blockset in the Simulink library browser, under *Blocksets and Toolboxes*. Double click on *CarMaker for Simulink*, or simply type

```
>> CarMaker4SL
```

in Matlab's command window.

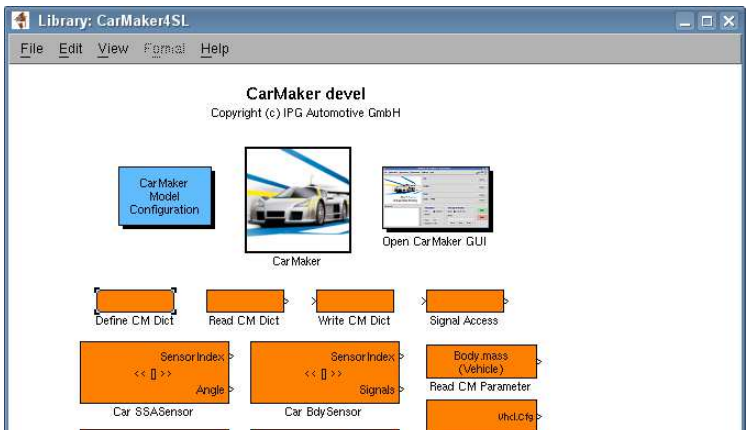


Figure 10.1: Matlab command window

10.1 General Information

10.1.1 Block Properties

The CarMaker for Simulink blocks of the CarMaker subsystem are *directly feed through* and run with a fixed step size of 1 ms, independently of the rest of the model.

CarMaker for Simulink blocks do not have continuous states in the Simulink sense of the word. They do have internal state variables but these are integrated using CarMaker internal solvers independently of Simulink.

10.1.2 Modeling Principles

The CarMaker for Simulink simulation model is made up of a subsystem containing a chain of individual blocks. When Simulink executes CarMaker, all blocks of this CarMaker block chain must be executed in order and exactly once. Algebraic loops involving individual blocks of the CarMaker block chain are not permitted.

Replacing existing CarMaker functionalities by Simulink blocks can be done by

- overwriting signals and
- disabling unneeded internal CarMaker functionalities using special parameters.

It can *NOT* be done by

- replacing, removing or reordering CarMaker blocks.

For further details please see the demonstration examples and their description in this manual.

10.1.3 Purpose of the Sync_In and Sync_Out Ports

The *Sync_In* and *Sync_Out* ports are an important concept in CarMaker for Simulink.

- They guarantee proper order of execution of the CarMaker blocks.
- They let you choose exactly when a CarMaker dictionary variable is accessed with a *Read CM Dict* or *Write CM Dict* block. You may want to read the most up to date value of a variable (and not the value calculated in the previous cycle) or override the value of a variable only after it is been calculated internally by CarMaker.

10.2 Utility Blocks

In the following sections we will introduce the most important elements of the CarMaker for Simulink library. You can find a complete description of all CarMaker for Simulink blocksets in the Programmer's Guide.

10.2.1 CarMaker GUI

The *CarMaker GUI* block is used to connect the Simulink model to a running CarMaker GUI process. If no running process is detected it will prompt the user to open a new CarMaker GUI, and to manually reconnect the undetected CarMaker GUI, or to cancel the attempt to connect the CarMaker GUI.

Double clicking on the *CarMaker GUI* block can also be used to switch between models, e.g. to “activate” a certain model in case of several models being loaded in Matlab. The “active” model is the one that will be simulated when you click on the *Start* button in the CarMaker GUI.

10.3 CarMaker Dictionary Blocks

10.3.1 Read CM Dict

The *Read CM Dict* block reads a variable in the CarMaker dictionary and provides its current value on the block's output port. The variable needs not to be defined with a *Define CM Dict* block in the model; any existing dictionary variable may be read.

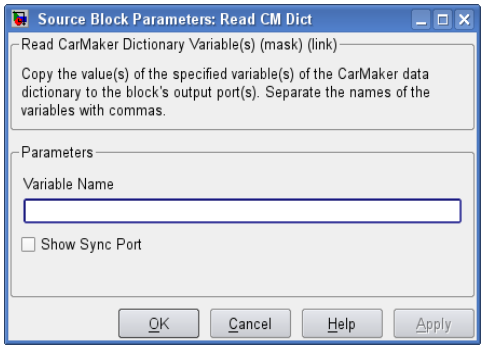


Figure 10.2: Read CM Dict block parameters dialog

Enter the name of the variable in the block's parameters dialog. In the model, the name will be displayed inside the block's symbol.

When a non-existing dictionary variable is given as a parameter of a *Read CM Dict* block, Simulink will report the following error at the start of a simulation:

```
Variable 'abcde' not in dictionary.
```

10.3.2 Write CM Dict

The *Write CM Dict* block writes the current value at the block's input port to a variable in the CarMaker dictionary. The variable needs not to be defined with a *Define CM Dict* block in the model; any existing dictionary variable also can be overwritten.

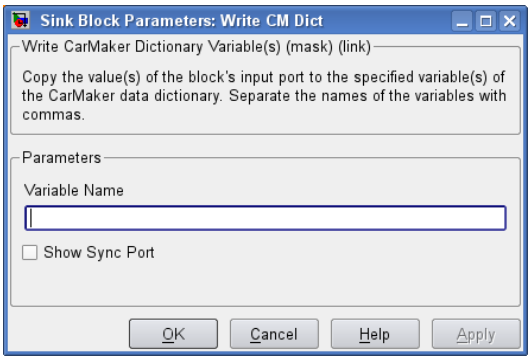


Figure 10.3: Write CM Dict block parameters dialog

Enter the name of the variable in the block's parameters dialog. In the model, the name will be displayed inside the block's symbol.

When a non-existing dictionary variable is given as a parameter of a *Write CM Dict* block, Simulink will report the following error at the start of a simulation:

```
Variable 'abcde' not in dictionary.
```

10.3.3 Define CM Dict

A Simulink model can define its own variables in the CarMaker dictionary, because there might be signals that are to be monitored with IPGControl or accessed using Direct Variable Access. Also, only signals defined in the CarMaker dictionary can be saved as part of a CarMaker simulation results file.

The *Def CM Dict* block defines a variable in the CarMaker dictionary.

When you define a dictionary variable in a Simulink model, it is recommended to prefix its name with the model's name or with a convenient abbreviation. This makes it easier for you to identify the model's variables in the dictionary with tools like the CarMaker GUI or IPGControl. Example: A dictionary variable `xyz` defined in a Simulink model called `MyModel` should be given the name `MyModel.xyz` or `MM.xyz`.

Be careful not to define a dictionary variable with the *Define CM Dict* block that is already defined somewhere else. Currently no error message will be issued in this case.

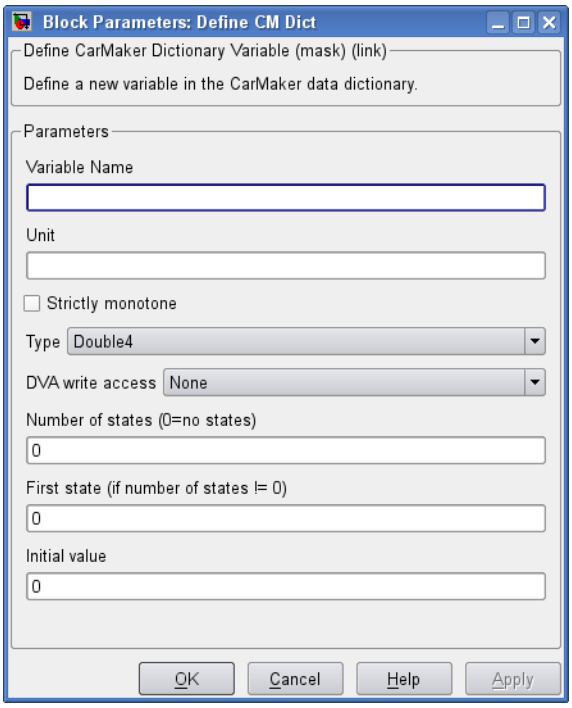


Figure 10.4: Define CM Dict block parameters dialog

Enter the name of the variable in the parameters dialog of the block. In the model, the name will be displayed inside the block's symbol.

The variable may also be given an unit, this is recommended but optional. See IPGControl for units used with CarMaker. Specifying an unit serves as some kind of documentation about the variable, but also allows IPGControl to display it on the same axis with other dictionary variables of the same unit.

If the variable's values are strictly monotonic increasing over time, you should check the *Stricly monotone* checkbox. Again, this is an information that tools like IPGControl need for proper display of the variable's values.

Choose the variable's type according to your needs and the range of values of the variable. You may choose between two floating point types and six integer types. When in doubt use *Double*.

For discrete variables (any of the integer types) with values starting at 0 and a reasonably small upper limit (e.g. an indicator light that is either on or off), it may make sense to specify the number of discrete states of the variable. For the indicator light there would be 2 discrete values (0=off, 1=on). Again this information serves mainly IPGControl which displays variables with a limited number of states in a special, space saving way. Specifying a value of 0 in this field means that no special state info is available. For the *Double* and *Float* type, the value of this field is ignored.

10.3.4 Dictionary Initialization

When CarMaker for Simulink has been started, but before the first simulation is run, the CarMaker dictionary does not yet know about any additional dictionary variables that will be defined in your model. To make these variables known to CarMaker for Simulink at start-up, you may create a file called *startup.dict* in your model directory, that describes the variables' properties.

The file is in plain ASCII and may contain empty lines, comment lines starting with "#", and lines defining dictionary variables. Each line containing a variable definition consists of 5 columns, separated by tabs and spaces:

- Column 1: The name of the variable.
- Column 2: The unit of the variable. A "-" (minus, hyphen) character means "no unit". The unit is used only for display purposes, in tools like IPGControl. CarMaker for Simulink uses SI units internally, no automatic unit conversion is done.
- Column 3: The type of the variable. Here is a list of allowed types and their corresponding C type:

```
double              double
float               float
ulongunsigned       long
longsigned          long
ushortunsigned      short
shortsigned         short
ucharunsigned       char
charsigned          char
```

- Column 4: The number of states.
- Column 5: Special properties. Again, this is used only for display purposes. If the variable is monotonic and increasing, specify "M". Otherwise specify "-".

Examples:

```
PT.GearBox.GearNo   -      long      0      -
PT.Engine.rotv      rad/s  float     0      -
Car.Distance        m      float     0      M
```

See also the documentation of the *Define CM Dict* block.

Chapter 11

How to extend the Simulink Model

Knowing about the advantages and disadvantages coming with Simulink you surely have some tasks, which can only be handled with Simulink and you want to continue. Therefore we will try to give you a short but intensive insight to make you familiar with the Simulink implementation of CarMaker.

Of course, we expect at this point at least a basic knowledge of Matlab and Simulink.

11.1 Overview of CarMaker's Simulink Structure

Please double-click on the CarMaker block.

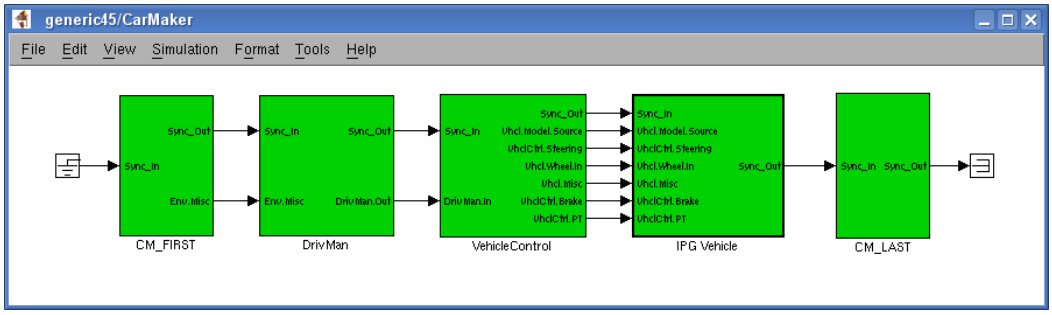


Figure 11.1: General structure of CarMaker in Simulink

Now, you can see the general structure of CarMaker. The only information you need at this point is that CarMaker's Simulink representation consists mainly of a Driver/Driving Maneuver model, Vehicle Control model and the Vehicle model.

The rest (IPGRoad, ...) is contained in the CarMaker library for Simulink.

Double-click on the DrivMan block.

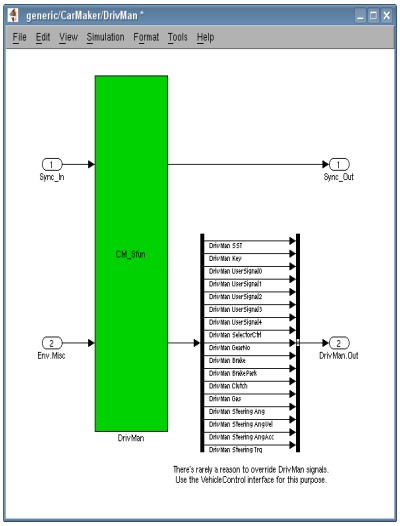


Figure 11.2: DrivMan block in Simulink

You recognize two details:

- First, the green block on the left is a S-function. That means, the full functionality is masked behind this level.
- Second, the output of the S-function is a multiplexed signal which contains all information about the interface of the S-function.

Double-click on the Vehicle block.

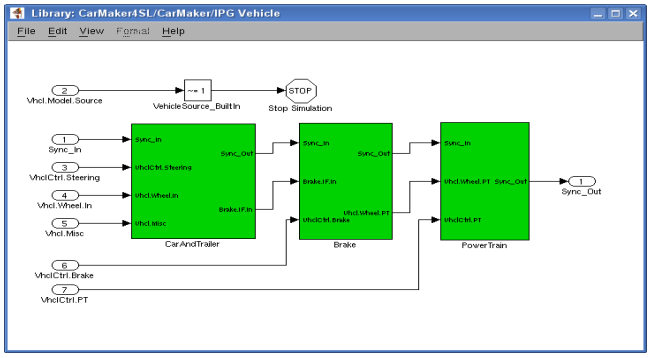


Figure 11.3: The Vehicle block in Simulink

You find a more detailed segmentation. You can now explore the Simulink block representation which ends each time with a S-function.

11.2 Example

11.2.1 Contents

- Calculate a new variable.
- Link this variable to CarMaker's User Accessible Quantities.
- Observe your changes with IPGControl.

11.2.2 Create a new Simulink Model

Open the Simulink Model "Generic.mdl".
Save it as "my_Model.mdl".

Thus you will be able to find your work later again. Open the "Generic.mdl" model and then from this model open your model "my_Model.mdl".

11.2.3 Prepare the extension

Sometimes the calculation of additional variables is done independently from the vehicle behavior. But mostly you need some information about one or more state variables of the vehicle to generate the information you want.

For example, if you want to calculate the current engine power you need information about the engine speed and the engine torque.

Multiplying both and using the factor Watt to Horsepower, you will receive the current engine power in the unit Horsepower.

Now, let us realize this: move to the mdl folder of your project directory, open your model "my_Model.mdl" and double-click on the CarMaker block.

It does not matter in which subsystem you carry out your change. Simulink offers sub-systems only for graphical purposes. At simulation time, sub-systems do not exist.

Switch now to the CarMaker/Vehicle/PowerTrain subsystem.

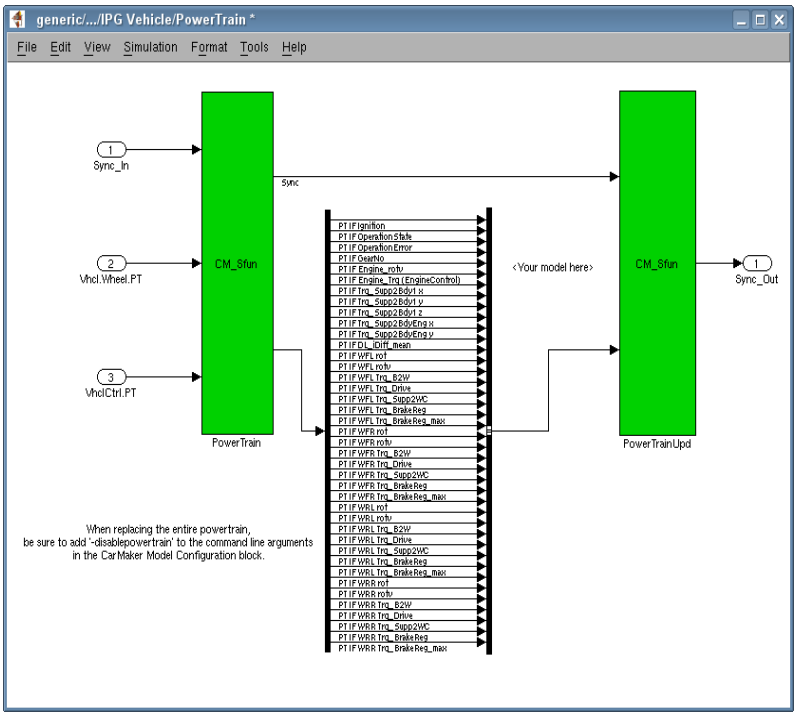


Figure 11.4: Powertrain Subsystem

Now we want to access some information from the powertrain component “Engine”.

11.2.4 Connect Inputs

The most important values of the simulation are the signals and busses between the sub-models. If possible, always use those signals to get information about the simulation.

Sometimes you need additional information which you can also see in IPGControl as a User Accessible Quantity, but may not be available as Simulink signal.

In this case you can access the information using a block of CarMaker4SL blockset.: “Read CM Dict”. More information about the CarMaker’s Simulink blockset can be found in section ’Features of the CarMaker for Simulink blockset’ on page 92.

With this block you can access all User Accessible Quantities offered by CarMaker.

Take a multiplicator block, and link the engine speed signal “PT.Engine.rotv“ to one entry of the multiplicator block.

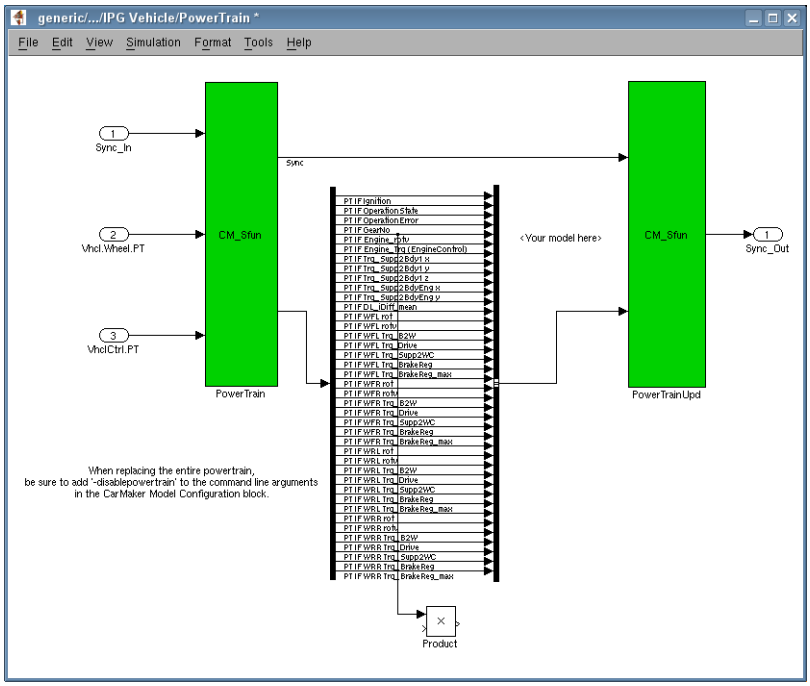


Figure 11.5: Multiplicator Block

Let us read a signal from CarMaker in order to receive the engine torque.
Indeed, the engine torque is not directly available as Simulink signal, thus we have to gain the same values from CarMaker, which are also available in IPGControl.

Open the CarMaker for Simulink library (see section 'Features of the CarMaker for Simulink blockset' on page 92).

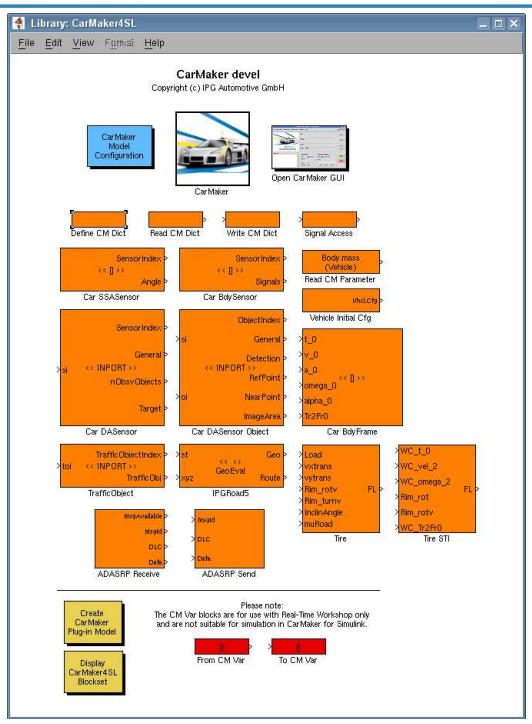


Figure 11.6: CarMaker for Simulink Library

Import a block "Read CM Dict" to your model, and define its property as "PT.Engine.Trq" in order to read the torque from CarMaker.

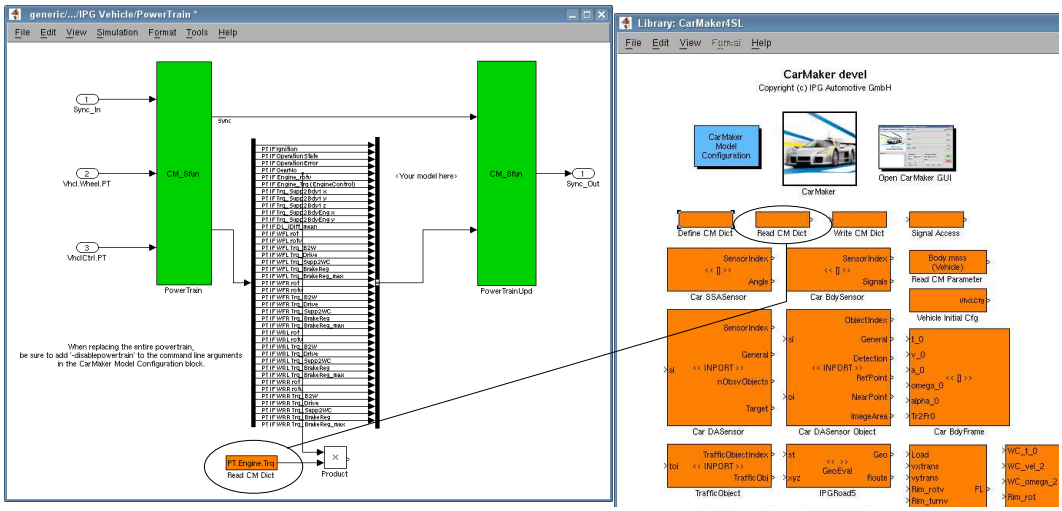


Figure 11.7: Read CM Dict

The power is in Watt, and we want to monitor it in Hp; therefore insert a gain block to make the conversion. Set a value of 1/735,5 for the gain block. Define also a new quantity to calculate the engine power, and plot its value in IPGControl.

Add a “Define CM Dict” block to your model, and define its property as “Engine.Power”.
Add a block “Write CM Dict“ to your model, and define its property as “Engine.Power“:

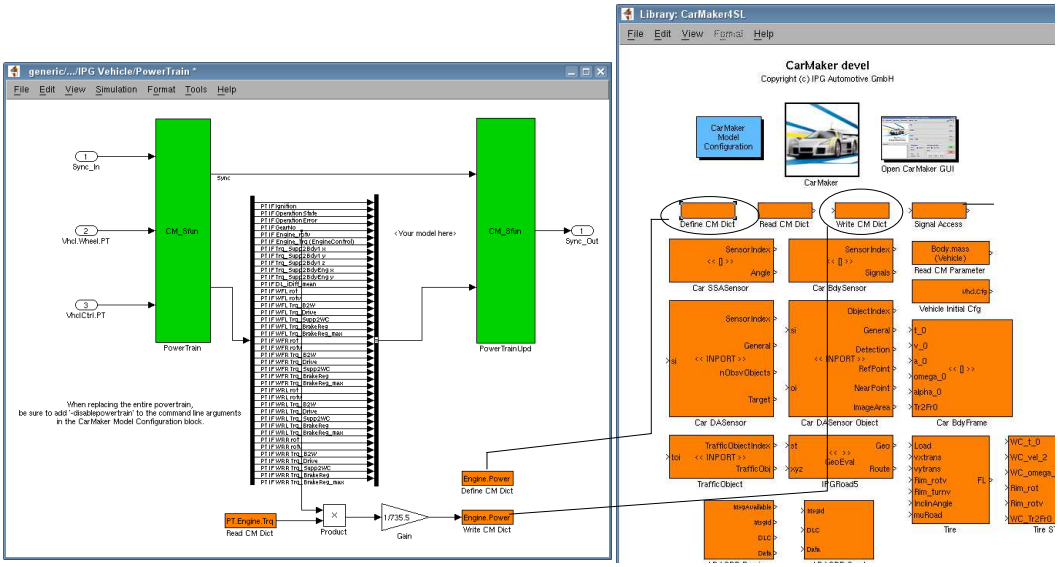


Figure 11.8: Engine Power

Please switch now to the CarMaker/GUI and load the TestRun "Examples>QuickStartGuide>Step1_Acceleration”. Open IPGControl and start the simulation.

You can observe the variable Engine. Power you have just created.

11.2.5 Demonstration Examples

CarMaker comes with a number of example models demonstrating various applications, features and modeling techniques. All examples can be found in the src_cm4sl folder of the CarMaker project directory.

The examples include:

- the Simulink model,
- Matlab parameter files and scripts, if needed,
- configuration files, e.g. TestRuns, vehicle data, tire data, etc.

Please find a detailled description of all models and the special blocksets uses in the Programmer’s Guide, section "CarMaker for Simulink".

Please note: The examples intend to demonstrate how to make use of a particular interface of CarMaker for Simulink, but not to provide an elaborate application or a full fledged replacement for a particular CarMaker subsystem.

All examples can be used with the Matlab versions supported by CarMaker. An overview of these versions can be found in the CarMaker Release Notes (CarMaker GUI > Help > Release Notes).

Chapter 12

Additional Information about CarMaker

12.1 What are the Differences between each CarMaker Version?

First let us differ between:

- CarMaker/Office: pure software simulations.
- CarMaker/HIL: CarMaker software adapted for Hardware In the Loop simulations.

CarMaker is a virtual vehicle environment. You can use it “as is“, e.g. without any other software. This version of CarMaker applies to CarMaker Office as well as to CarMaker/HIL.

CarMaker can also be used together with Simulink: In this case, the CarMaker models are integrated in the Simulink environment. When you start the simulation, Simulink is the simulation software and it uses the library of CarMaker to perform the calculations. This version applies only to CarMaker Office.

Alternatively, you can build some extra models with Simulink (thus, out of the CarMaker environment for Simulink), by generating the C-code corresponding to this model with the Real Time Workshop, and integrating the resulting library in CarMaker in the C level. This version applies to CarMaker Office as well as to CarMaker/HIL.

12.2 Typical Tests

The following table is a list of the most important pre-defined TestRuns which show you other capabilities of CarMaker.

We suggest you to load the TestRun that seems to be of interest for you and to look at how it was built. The column path indicates you in which sub-directory you should search for the TestRun:

Topic	Path	Comments
Real Tests	./Examples/VehicleDynamics	These TestRuns show you how to a build common vehicle dynamic tests including slalom, lane change and steady circle tests. Pay attention to the definition of IPGDriver, especially to the maximal values of longitudinal and lateral acceleration and to the road definition.
Real Tests	./Examples/IntegratedControl-Systems	These tests show the influence of control systems such as ABS or ESP. Pay attention to the road definition (“Override Attributes“ for each road segment), and the maneuver definition (manual control of the pedals).
Using real measurements	./Examples/CarMakerFunctions/ExternalInputs	Usage of real measurements. Pay attention to the following menu: CarMaker GUI > Parameters > Input from File. You can check the content of the input file by clicking on “Content“ (Box Input File).
Drive Cycles	./Examples/FuelEfficiency	These tests include the most popular drive cycles such as FTP, NEDC short and NEDC long. Pay attention to the maneuver definition (Input From File) and the vehicle’s powertrain configuration.
Traffic Environment	./Examples/CarMakerFunctions/IPGTraffic	How to simulate traffic objects and define an overtaking maneuver. Pay attention to the traffic definition (Parameters > Traffic), and to the maneuver definition.
Traffic Environment	./Examples/DriverAssistance-Systems/AutonomousFunctions/PedestrianDetection_CrosstheRoad	How to create a crossroad section and a city environment. Pay attention to the road and traffic definition.
Driver Assistance Systems	./Examples/DriverAssistance-Systems	These tests show special assistance systems: ACC, lane keeping and parking assistance systems. Pay attention to the sensor definition (Parameters > Vehicle > Sensors).
Vehicle features	./Examples/CarMakerFunctions/FlexBody ./Examples/CarMakerFunctions/PfefferSteering ./Examples/CarMakerFunctions/Sensors	These tests present some of the special features of CarMaker’s vehicle model, like flexible body, Pfeffer steering model, sensor definitions. Pay attention to the vehicle parameterization (Parameters > Vehicle).
Driving as usual	./Examples/CarMakerFunctions/IPGDriver	How to simulate reversing, turning on intersections, race conditions etc.. Pay attention to the settings of IPGDriver.
Road features	./Examples/CarMakerFunctions/IPGRoad	How to realize comprehensive road configurations including real measurements, 3D tracks and special bumps and markers.
Simulation specific functions	./Examples/CarMakerFunctions/ManeuverControl ./Examples/CarMakerFunctions/DVA ./Examples/CarMakerFunctions/RTExpressions	Special maneuver definitions including the minimaneuver command language, DVA commands and RTExpressions. Pay attention to the Minimaneuver Commands in the maneuver definition.
Testautomation	./Examples/CarMakerFunctions/TestManager	Using the TestManager functionality to create a test series including parameter variations, criteria evaluation etc.. See CarMaker GUI > Simulation > TestManager.

12.3 Feature List

DVA: How to Manipulate the Variables Online

CarMaker enables the user to check online the values of some variables, e.g. to plot the diagram of the speed during the simulation. All these variables are called User Accessible Quantities.

We make use of the UAQs, not only to *read* them during the simulation, but also to write them during the simulation: we call that the Direct Variable Access.

It is an unique and very powerful feature of CarMaker, since you can influence the simulation *online*, e.g. to trigger some exceptional actions such as an accident, or a failure in the vehicle which blocks the steering system.

The DVA functionality also is programmable.

You have access to the DVA GUI through the CarMaker GUI > Application > Direct Variable Access.

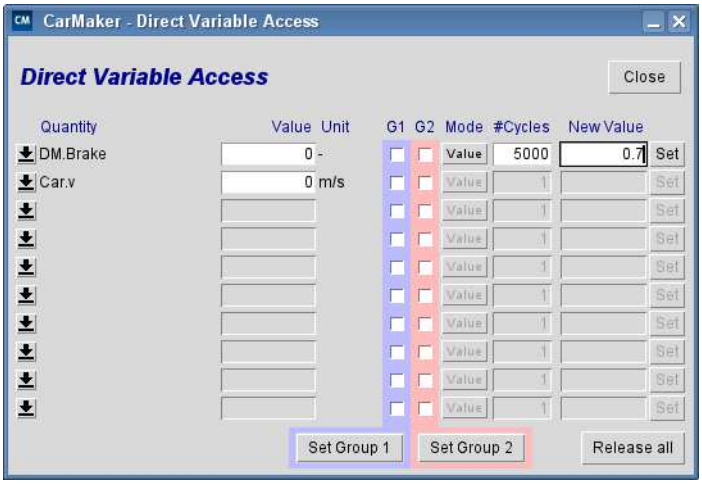


Figure 12.1: DVA window

Model Check: How to check your Vehicle Parameterization

The ModelCheck is a tool used to plot diagrams from the CarMaker TestRun definition that are standard in the automotive domain.

You can access the ModelCheck GUI through the CarMaker GUI > Simulation > ModelCheck.

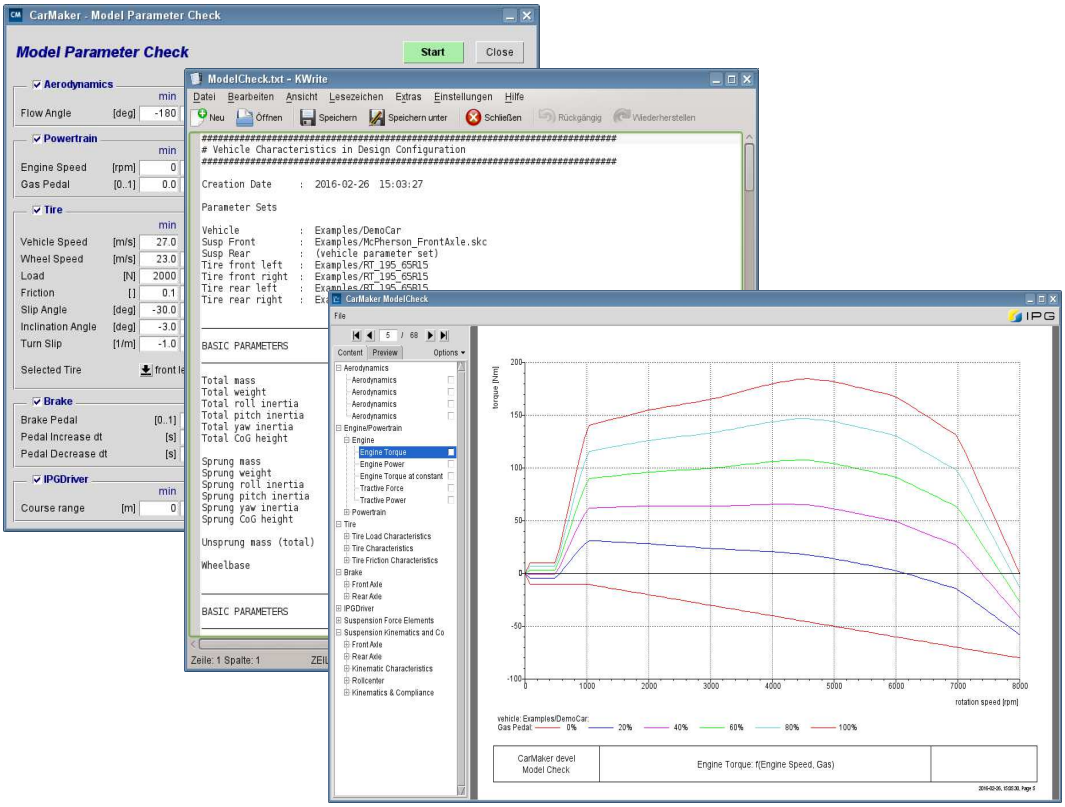


Figure 12.2: ModelCheck GUI

Test Automation

CarMaker offers many solutions to automate your simulation tests. The description of these tools is documented in User's Guide and Reference Manual.

- Test Manager

The Test Manager functionality enables to define a row of TestRuns, so that CarMaker automatically starts the next TestRun, when one is finished. Thus, if you set your "Storage of Results" options on "Save all", you can start a series of tests before leaving your office, and come back the next morning to analyze your results!

Variations of quantities and parameters in a TestRun are easily conductible and there is also the possibility to integrate your own scripts for controlling the execution of TestRuns.

- Script Control

The ScriptControl enables to automate all what you can do through the GUI, but it can do even more! As it is based on a script language (tcl/tk), you can define some closed loops to load your tests or save your results, define some counters...

- DDE

The ScriptControl can be also controlled by DDE channels.

- MATLAB

Last but not least, you can control the SciptControl via MATLAB.

IPGDriver: Driver Model

- Standard Driver

The standard driver model is the one used during the previous chapters. It is much more than a simple speed and steering wheel controller.

The driver can shift the gears, decide when it is better to shift sooner to get more power at the end of a curve, looks forward to predict its speed and course...

- Race Driver

The race driver learns himself the limits of the vehicle in order to optimize its trajectory.

Input From File: Integration of Measurements to the Simulation?

This feature enables to define the maneuver based on measurements you have performed.

For instance, this can be very useful to compare the simulation results of a braking phase with your measurements: you give the measured position of all pedals during the maneuver, and then compare the braking distance, longitudinal acceleration, etc. of the simulation with the data from your measurements. For this, see the TestRun “Examples/CarMakerFunctions/ExternalInputs/ActionFromFile“ .

Obstacles on the Road Surface

The road model of CarMaker features a set of obstacles to alter its surface:

- cylinders
- beams
- waves
- cones

See the examples under “Examples/CarMakerFunctions/IPGRoad”.

Simulation of Wind Effects

It is also possible to simulate wind effects. See the example “Examples/CarMakerFunctions/IPGRoad/Markers_CrossWindSection”.

Traffic Simulation

You have the possibility to simulate the presence of additional vehicles on the road, in both directions.

You have also the possibility to slow down the driver thanks to velocity signs, and to display those signs in IPGMovie.

See the examples from the topic traffic environment on “Examples/CarMakerFunctions/IPGTraffic”.

Secured Files: How to Exchange Safely your Data Sets?

With CarMaker you can convert your data sets into binary files.

Your partner can not read the values defined in the file, but he can use it for its simulation with CarMaker.

IPGControl

IPGControl is a tool to plot online the diagrams of any of the UAQ (see definition in “Examples/CarMakerFunctions/DVA”).

IPGControl also works offline.

IPGMovie

IPGMovie is the online animation tool of CarMaker.

IPGMovie also works offline.

Online Capabilities

You always can check online how the simulation is running.

E.g. if you see in IPGMovie that you made a mistake by choosing your vehicle, you can stop at once the simulation, load the correct vehicle or TestRun, and start the simulation again. You do not need to wait for the end of the simulation to check if something goes wrong.

3D road

The road model of CarMaker is a 3D model.

You can create your road even without any data, thanks to a segment based interface.

If you have road measurements, you can use them in order to perform your simulation tests on the same road as the one you use for your real tests.

Integration of External Models

CarMaker delivers several interfaces to integrate external models:

- Simulink: you can modify the models used in CarMaker, and also read some values in files from Simulink.
- C-code: you can program your own code, or simply your interfaces with your own tools.
- AVL Cruise
- Sherpa Engineering thermodynamical engine model

Post processing

- MATLAB

CarMaker is also interfaced with MATLAB in order to analyze your results. You can load the CarMaker results in Matlab.

- EXCEL

You can export your results in ASCII files in order to read them with EXCEL.

Documentation of CarMaker

To get further information about CarMaker and its functions please refer to the comprehensive documentation of CarMaker (User's Guide, Reference Manual and Programmer's Guide).