Model based design of

Suspension System

Submitted by

Gourav Pani

Unique Id-2005178

Date-18th December 2020

PREFACE

|  |  |  |
| --- | --- | --- |
| Sl No | Title | Page No |
| 1 | Suspension system | 4 |
| 2 | Design requirements | 5 |
| 3 | Modelling | 5 |
| 4 | Working of the model | 7 |
| 5 | Data inspector | 8 |
| 6 | Callback | 9 |
| 7 | Matlab function block | 10 |
| 8 | Lookup table | 10 |
| 9 | Signal builder | 12 |
| 10 | Solver | 13 |

Figure Table

|  |  |  |
| --- | --- | --- |
| Sl No | Figure Name | Page No |
| 1 | Model block diagram of Suspension system | 6 |
| 2 | Model Output | 7 |
| 3 | Data inspector | 8 |
| 4 | Lookup table | 11 |
| 5 | Signal builder | 12 |

1. Suspension System

Suspension is the system of tires, tire air, springs, shock absorbers and linkages that connects a vehicle to its wheels and allows relative motion between the two. Suspension systems must support both road holding/handling and ride quality, which are at odds with each other. The tuning of suspensions involves finding the right compromise. It is important for the suspension to keep the road wheel in contact with the road surface as much as possible, because all the road or ground forces acting on the vehicle do so through the contact patches of the tires. The suspension also protects the vehicle itself and any cargo or luggage from damage and wear.

2. Design Requirements

A good automotive suspension system should have satisfactory road holding ability, while still providing comfort when riding over bumps and holes in the road. When the vehicle is experiencing any road disturbance (i.e. pot holes, cracks, and uneven pavement), the vehicle body should not have large oscillations, and the oscillations should dissipate quickly. Since the distance X1-W is very difficult to measure, and the deformation of the tire (X2-W) is negligible, we will use the distance X1-X2 instead of X1-W as the output in our problem. Keep in mind that this is an estimation.

The road disturbance (W) in this problem will be simulated by a step input. This step could represent the vehicle coming out of a pothole. We want to design a feedback controller so that the output (X1-X2) has an overshoot less than 5% and a settling time shorter than 5 seconds. For example, when the vehicle runs onto a 10 cm high step, the vehicle body will oscillate within a range of +/- 5 mm and return to a smooth ride within 5 seconds.

3. Modelling

m1 body mass

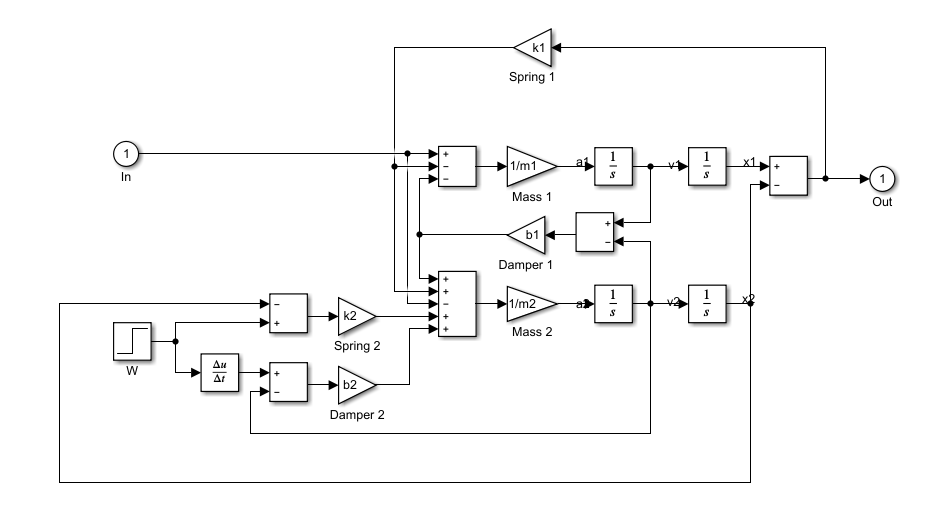
m2 suspension mass

k1 spring constant of suspension system

k2 spring constant of wheel and tire

b1 damping constant of suspension system

b2 damping constant of wheel and tire



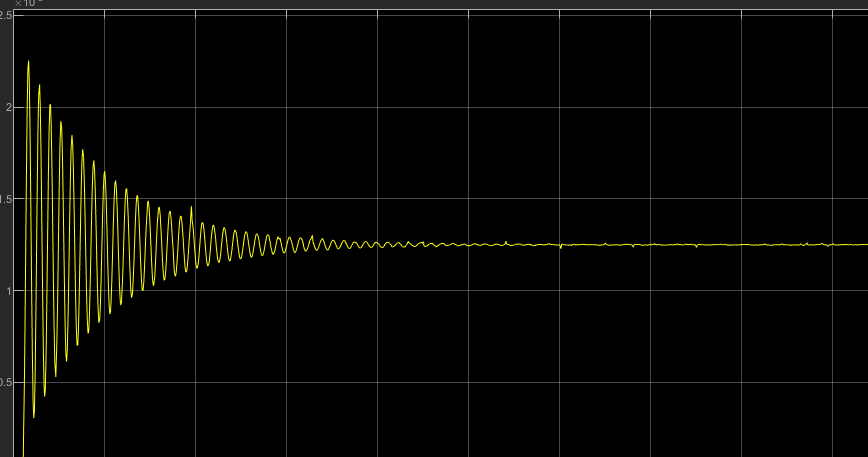
This system will be modelled by summing the forces acting on both masses (body and suspension) and integrating the accelerations of each mass twice to give velocities and positions

$$
\int\int\frac{d^2x_1}{dt^2}\ dt = \int\frac{dx_1}{dt}\ dt = x_1
$$

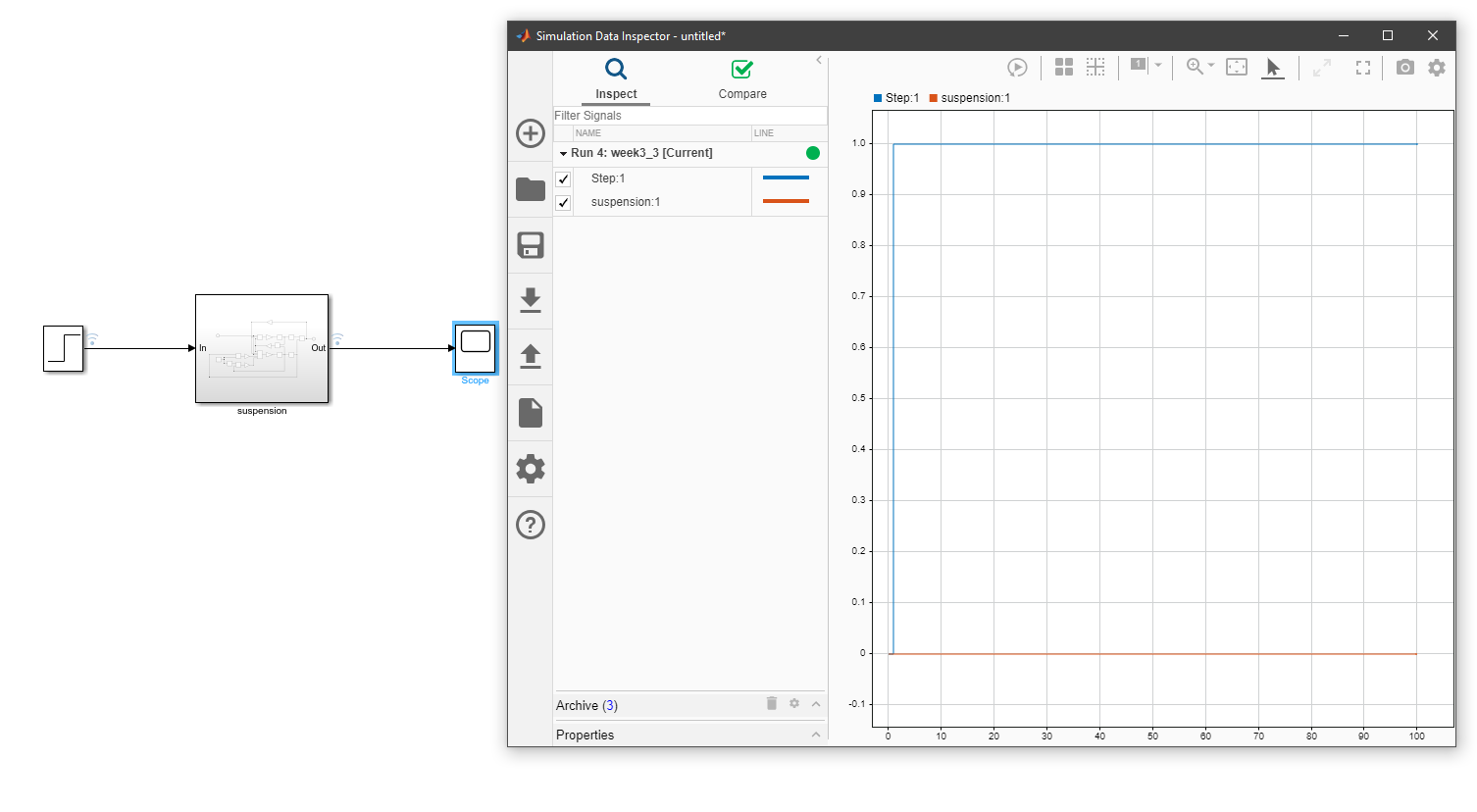
$$
\int\int\frac{d^2x_2}{dt^2}\ dt = \int\frac{dx_2}{dt}\ dt = x_2
$$

4. Working of the model

The figure shows system output of an active suspension system.



5. Data inspector



The Simulation Data Inspector visualizes and compares multiple kinds of data. Using the Simulation Data Inspector, you can inspect and compare time series data at multiple stages of your workflow. This example workflow shows how the Simulation Data Inspector supports all stages of the design cycle:

* View Data in the Simulation Data Inspector

Run a simulation in a model configured to log data to the Simulation Data Inspector, or import data from the workspace or a MAT-file. You can view and verify model input data or inspect logged simulation data while iteratively modifying your model diagram, parameter values, or model configuration.

* Inspect Simulation Data

Plot signals on multiple subplots, zoom in and out on specified plot axes, and use data cursors to understand and evaluate the data. Create Plots Using the Simulation Data Inspector to tell your story.

* Compare Simulation Data

Compare individual signals or simulation runs and analyse your comparison results with relative, absolute, and time tolerances. The compare tools in the Simulation Data Inspector facilitate iterative design and allow you to highlight signals that do not meet your tolerance requirements. For more information about the comparison operation, see How the Simulation Data Inspector Compares Data.

* Save and Share Simulation Data Inspector Data and Views.

Share your findings with others by saving Simulation Data Inspector data and views.

6. Callbacks

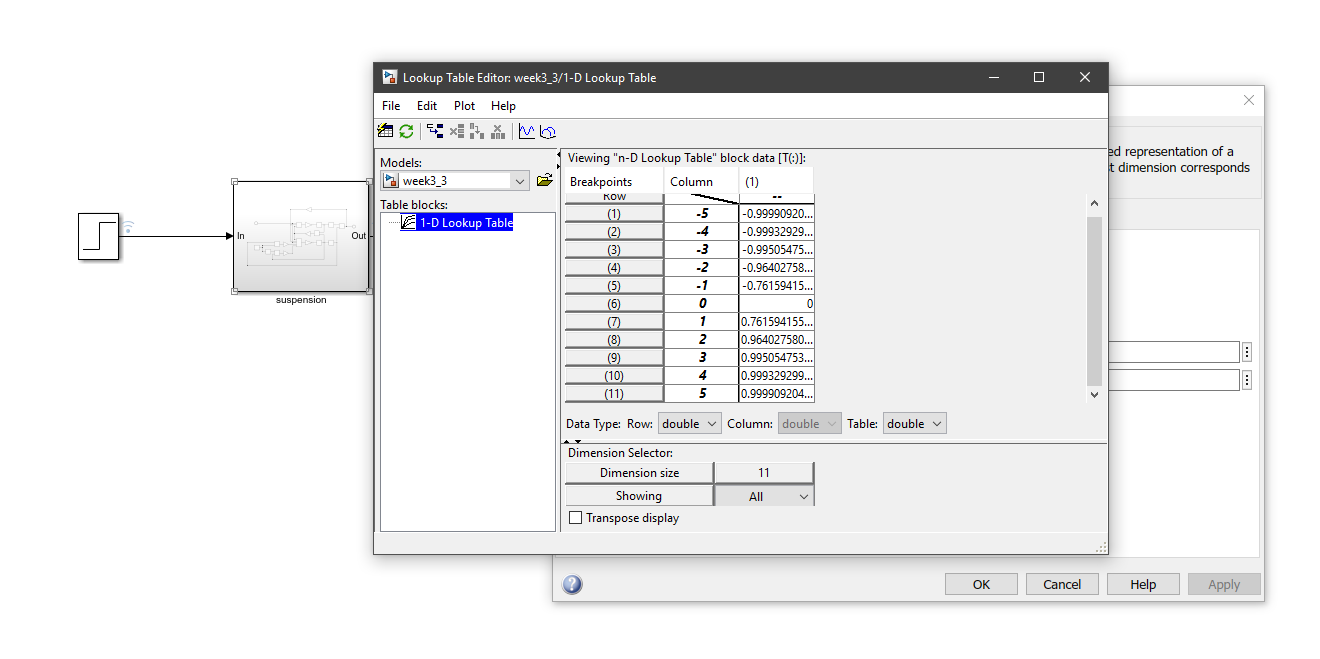
Callbacks are used to initialize values to model with preload, postponed options

Preload has chosen and values are initialised in it as model opens it preloads the values and keep ready for execution.

7. Matlab function Block

MATLAB Function blocks enable you to define custom functionality in Simulink models by using the MATLAB language. They are the easiest way to bring MATLAB code into Simulink. MATLAB Function blocks support C/C++ code generation from Simulink Coder and Embedded Coder.

8. Lookup table

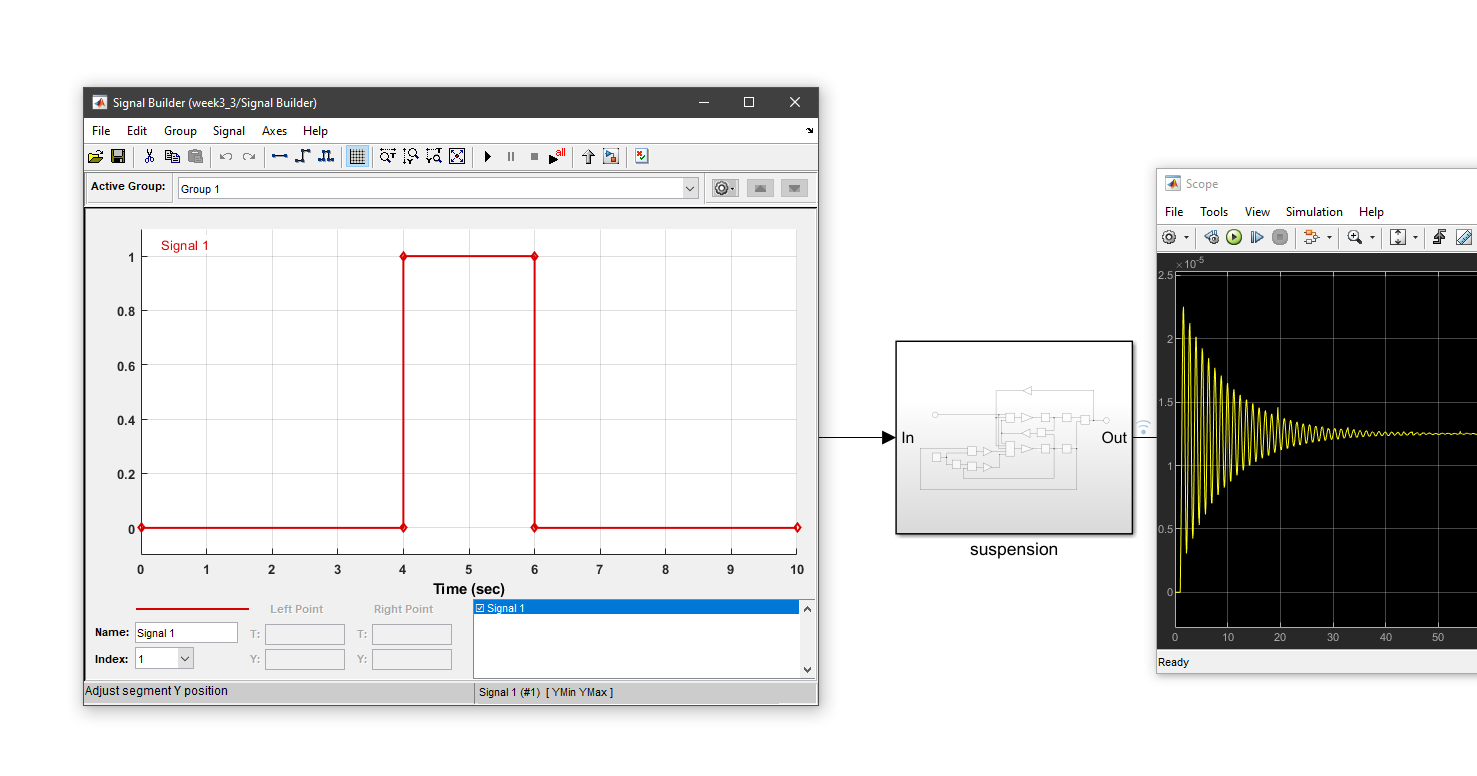


A lookup table is an array of data that maps input values to output values, thereby approximating a mathematical function. Given a set of input values, a lookup operation retrieves the corresponding output values from the table. If the lookup table does not explicitly define the input values, Simulink can estimate an output value using interpolation, extrapolation, or rounding, where:

* An interpolation is a process for estimating values that lie between known data points.
* An extrapolation is a process for estimating values that lie beyond the range of known data points.
* A rounding is a process for approximating a value by altering its digits according to a known rule.

A lookup table block uses an array of data to map input values to output values, approximating a mathematical function. Given input values, Simulink performs a “lookup” operation to retrieve the corresponding output values from the table. If the lookup table does not define the input values, the block estimates the output values based on nearby table values.

9. Signal builder



The Signal Builder block allows you to create interchangeable groups of piecewise linear signal sources and use them in a model. You can quickly switch the signal groups into and out of a model to facilitate testing.

10. Solver

Ordinary Differential Equations

An ordinary differential equation (ODE) contains one or more derivatives of a dependent variable, y, with respect to a single independent variable, t, usually referred to as time. The notation used here for representing derivatives of y with respect to t is y′ for a first derivative, y′′ for a second derivative, and so on. The order of the ODE is equal to the highest-order derivative of y that appears in the equation.