Antilock Braking System

Name: Mukesh Raju Mahajan

Index

Introduction

An **anti-lock braking system** (**ABS**) is a safety anti-skid braking system used on aircraft and on land vehicles, such as cars, motorcycles, trucks, and buses. ABS operates by preventing the wheels from locking up during braking, thereby maintaining tractive contact with the road surface and allowing the driver to maintain more control over the vehicle.

ABS is an automated system that uses the principles of threshold braking and cadence braking, techniques which were once practiced by skillful drivers before ABS was widespread. ABS operates at a much faster rate and more effectively than most drivers could manage. Although ABS generally offers improved vehicle control and decreases stopping distances on dry and some slippery surfaces, on loose gravel or snow-covered surfaces ABS may significantly increase braking distance, while still improving steering control. Since ABS was introduced in production vehicles, such systems have become increasingly sophisticated and effective. Modern versions may not only prevent wheel lock under braking, but may also alter the front-to-rear brake bias. This latter function, depending on its specific capabilities and implementation, is known variously as electronic brake force distribution, traction control system, emergency brake assist, or electronic stability control (ESC).

So in order to design this ABS system I have used the tool which called MATLAB. In this I have designed a whole system by using basic building blocks. Also various techniques I have used in order to make the performance of system better.

Analysis and Physics

The wheel rotates with an initial angular speed that corresponds to the vehicle speed before the brakes are applied. I used separate integrators to compute wheel angular speed and vehicle speed. I use two speeds to calculate slip, which is determined by Equation 1. Note that I have introduce vehicle speed expressed as an angular velocity .

$$\omega_v = \frac{V}{R} \text{ (equals the wheel angular speed if there is no slip)}$$

Equation 1

$$\omega_v = \frac{V_v}{R_r}$$

$$slip = 1 - \frac{\omega_w}{\omega_v}$$

$\omega_v =$ vehicle speed divided by wheel radius

$V_v =$ vehicle linear velocity

$R_r =$ wheel radius

$\omega_w =$ wheel angular velocity

From these expressions, we see that slip is zero when wheel speed and vehicle speed are equal, and slip equals one when the wheel is locked. A desirable slip value is 0.2, which means that the number of wheel revolutions equals 0.8 times the number of revolutions under non-braking conditions with the same vehicle velocity. This maximizes the adhesion between the tire and road and minimizes the stopping distance with the available friction.

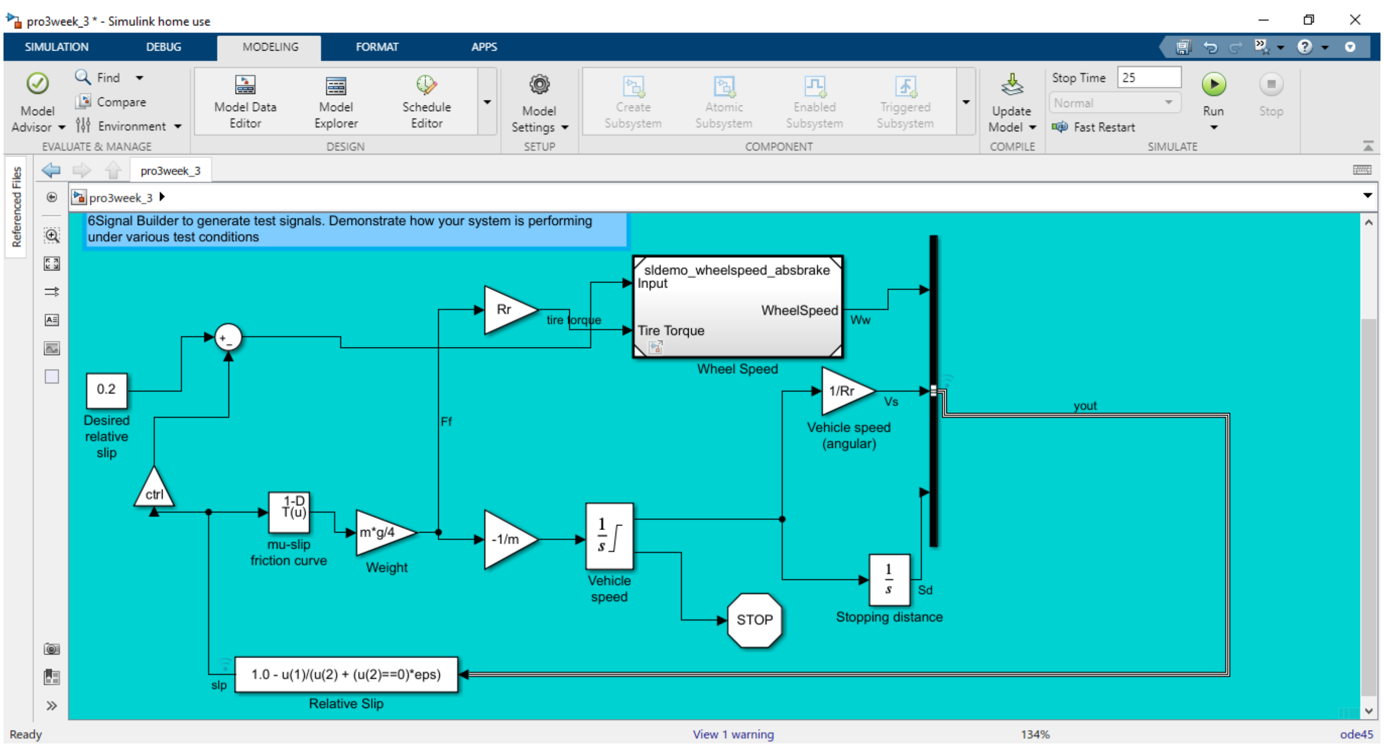
Modeling

The friction coefficient between the tire and the road surface, mu, is an empirical function of slip, known as the mu-slip curve. I have created mu-slip curves by passing MATLAB variables into the block diagram using a Simulink lookup table. The model multiplies the friction coefficient, mu, by the weight on the wheel, W, to yield the frictional force, Ff, acting on the circumference of the tire. Ff is divided by the vehicle mass to produce the vehicle deceleration, which the model integrates to obtain vehicle velocity.

In this model, I have used an ideal anti-lock braking controller, that uses 'bang-bang' control based upon the error between actual slip and desired slip. I set the desired slip to the value of slip at which the mu-slip curve reaches a peak value, this being the optimum value for minimum braking distance .

To control the rate of change of brake pressure, the model subtracts actual slip from the desired slip and feeds this signal into a bang-bang control (+1 or -1, depending on the sign of the error). This on/off rate passes through a first-order lag that represents the delay associated with the hydraulic lines of the brake system. The model then integrates the filtered rate to yield the actual brake pressure. The resulting signal, multiplied by the piston area and radius with respect to the wheel (Kf), is the brake torque applied to the wheel.

The model multiplies the frictional force on the wheel by the wheel radius (Rr) to give the accelerating torque of the road surface on the wheel. The brake torque is subtracted to give the net torque on the wheel. Dividing the net torque by the wheel rotational inertia, I, yields the wheel acceleration, which is then integrated to provide wheel velocity. In order to keep the wheel speed and vehicle speed positive, limited integrators are used in this model.



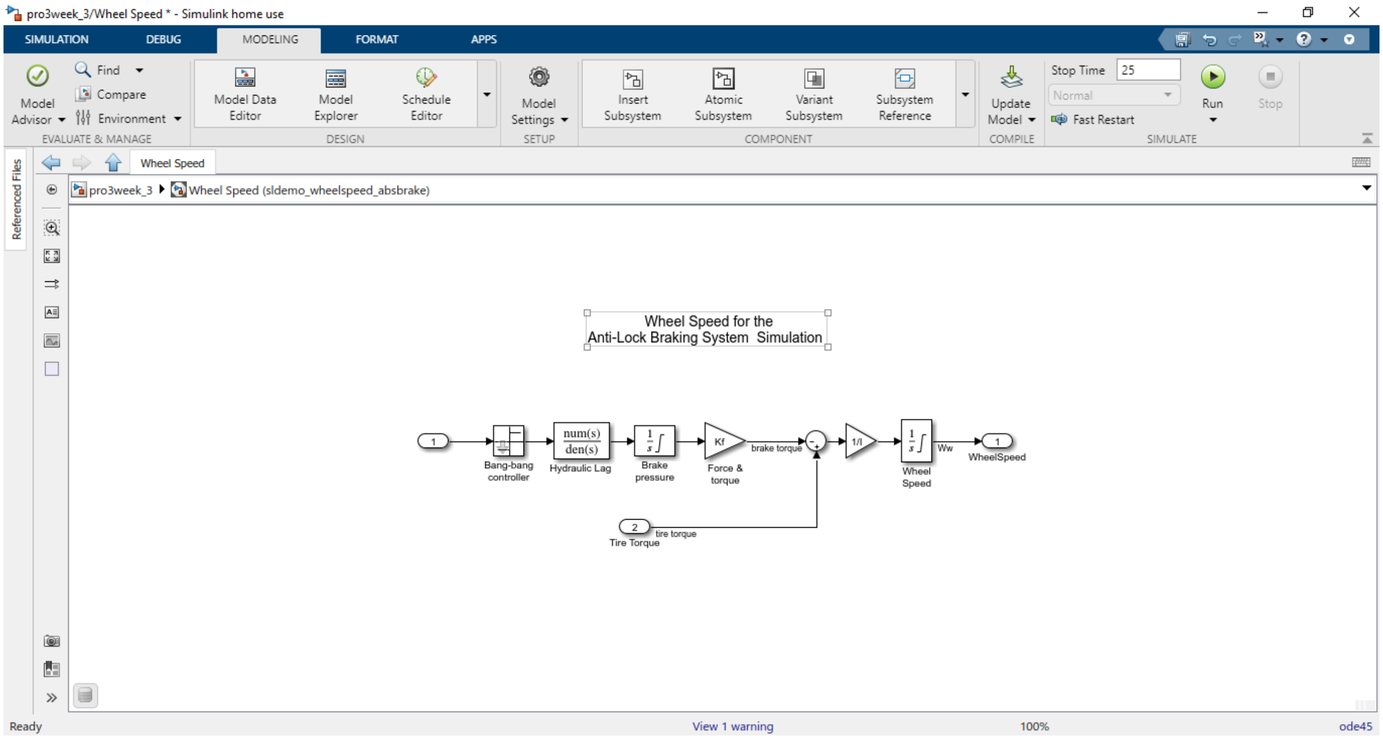
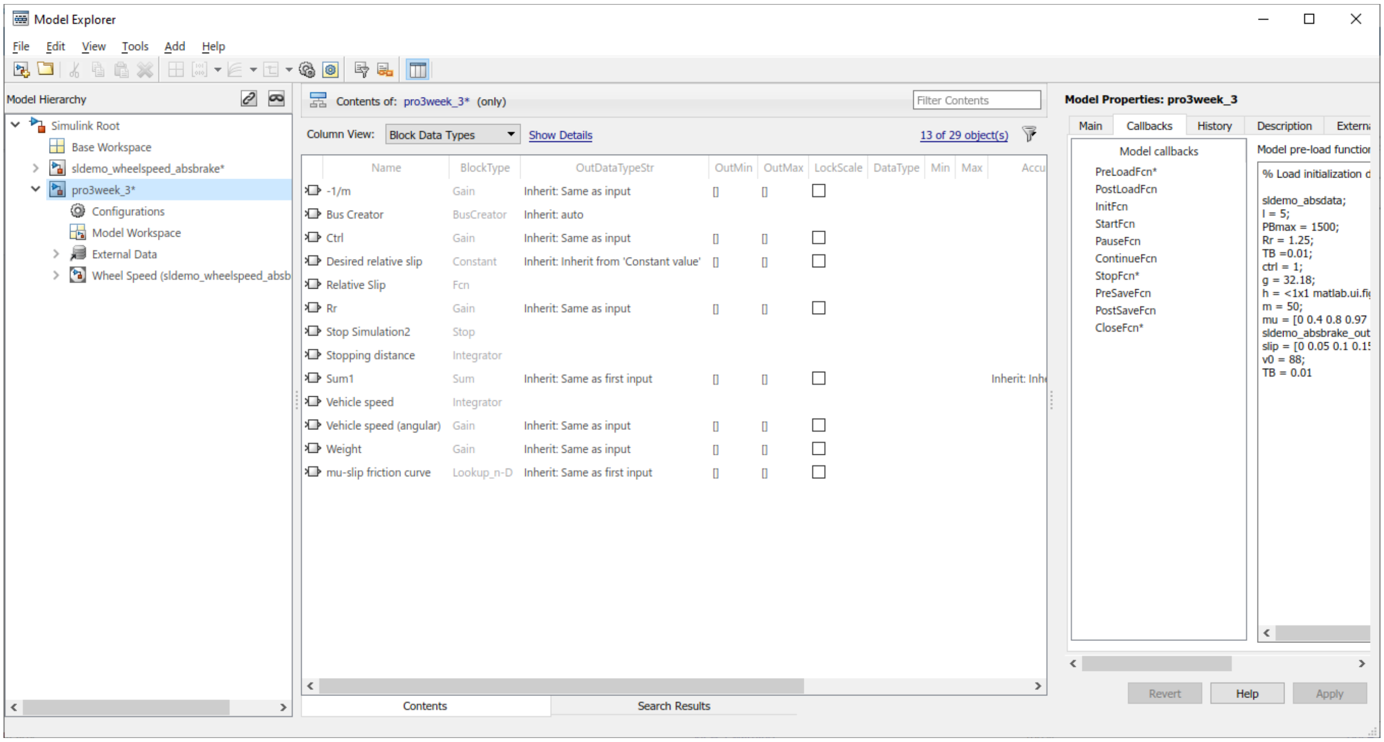


Fig: Subsystem of Wheel Speed

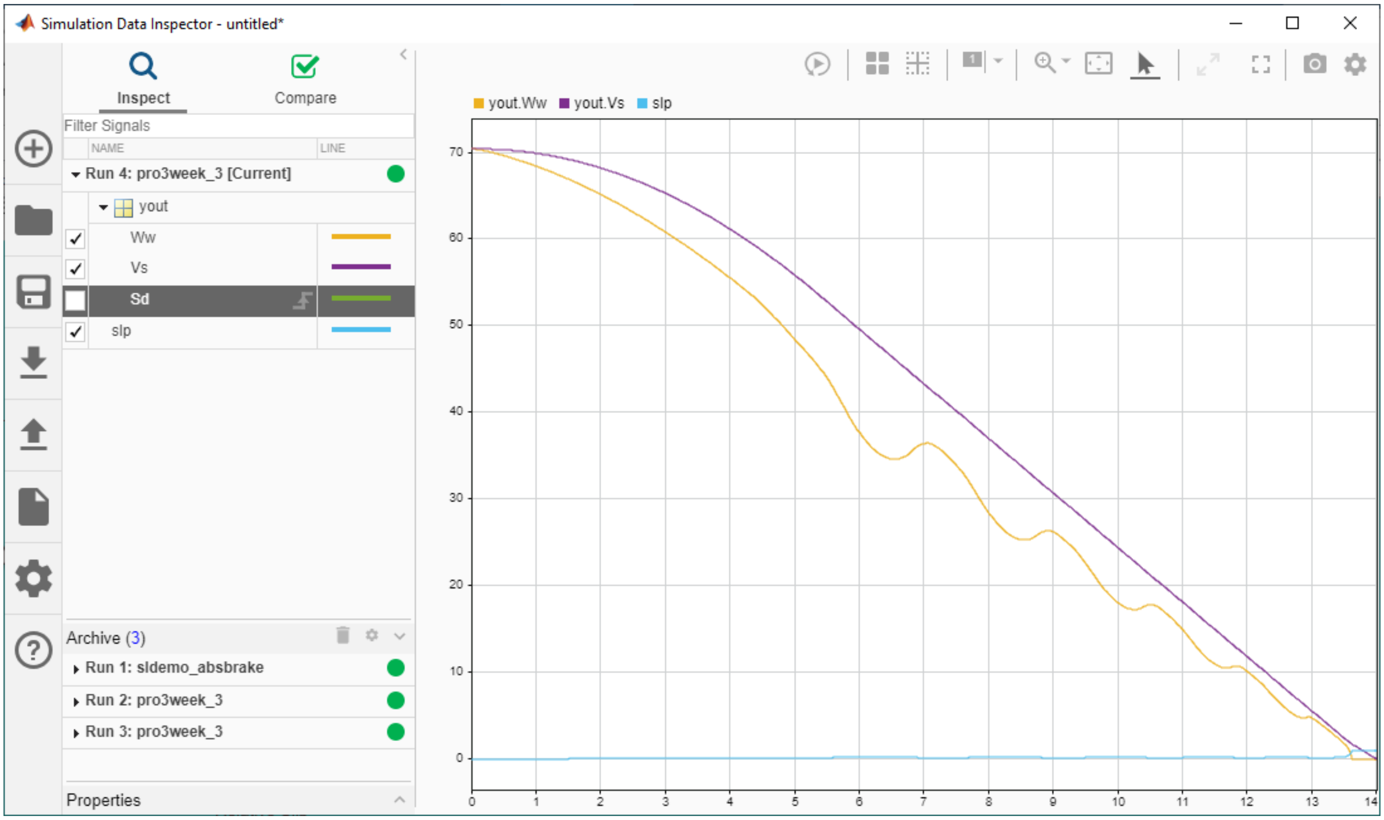
Various Skills used in Model

1. Callbacks:

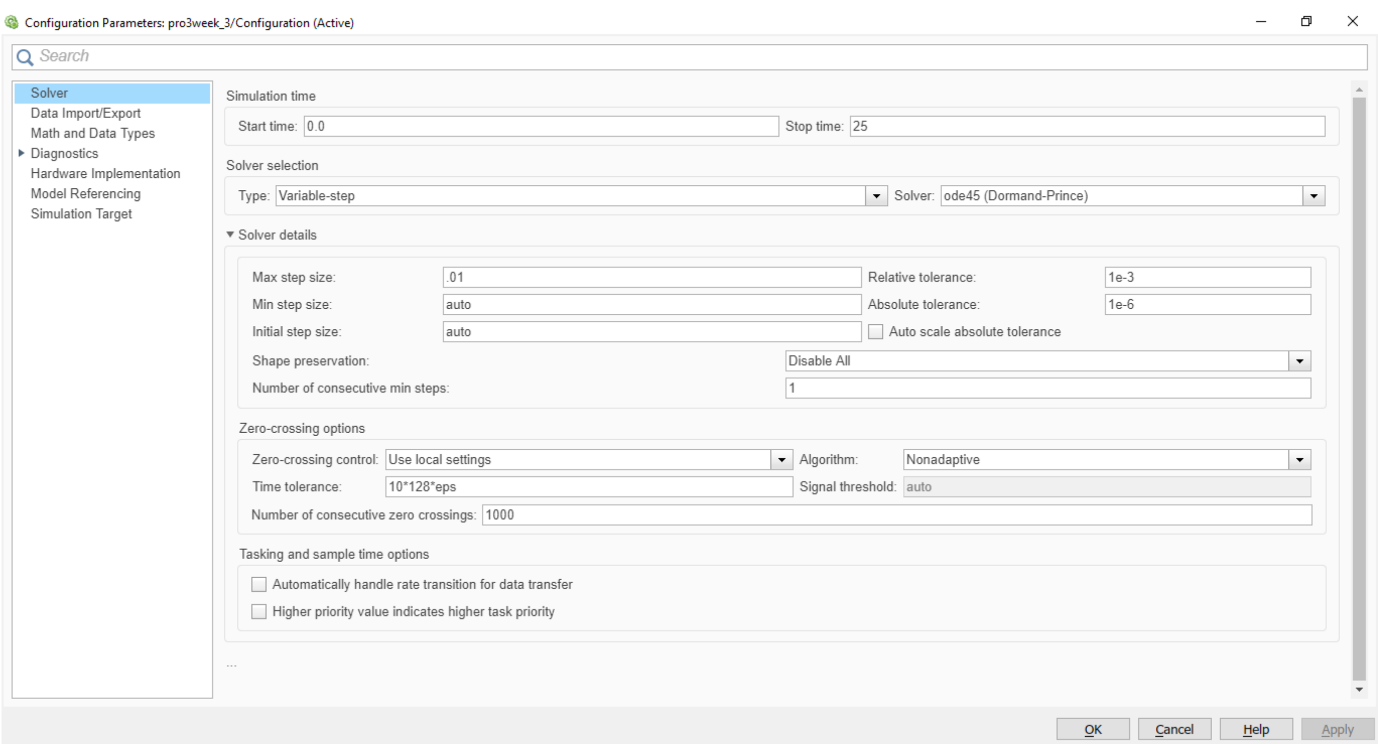
I have used Preload Function in order to load variables which I have used in my model. By using this all the variables are get loaded when we open the model.


Model Explorer
File Edit View Tools Add Help
Model Hierarchy
Simulink Root
Base Workspace
sldemo_wheelspeed_absbrake*
pro3week_3*
Configurations
Model Workspace
External Data
Wheel Speed (sldemo_wheelspeed_absb
Contents of: pro3week_3* (only)
Filter Contents
Column View: Block Data Types
Show Details
13 of 29 object(s)
Name BlockType OutDataTypeStr OutMin OutMax LockScale DataType Min Max Accu
-1/m Gain Inherit: Same as input [] []
Bus Creator BusCreator Inherit: auto
Ctrl Gain Inherit: Same as input [] []
Desired relative slip Constant Inherit: Inherit from 'Constant value' [] []
Relative Slip Fcn
Rr Gain Inherit: Same as input [] []
Stop Simulation2 Stop
Stopping distance Integrator
Sum1 Sum Inherit: Same as first input [] [] Inherit: Inh
Vehicle speed Integrator
Vehicle speed (angular) Gain Inherit: Same as input [] []
Weight Gain Inherit: Same as input [] []
mu-slip friction curve Lookup_n-D Inherit: Same as first input [] []
Contents
Search Results
Model Properties: pro3week_3
Main Callbacks History Description Externa
Model callbacks
PreLoadFcn*
PostLoadFcn
InitFcn
StartFcn
PauseFcn
ContinueFcn
StopFcn*
PreSaveFcn
PostSaveFcn
CloseFcn*
Model pre-load functio
% Load initialization d
sldemo_absdata;
I = 5;
PBmax = 1500;
Rr = 1.25;
TB =0.01;
ctrl = 1;
g = 32.18;
h = <1x1 matlab.ui.fi
m = 50;
mu = [0 0.4 0.8 0.97
sldemo_absbrake_out
slip = [0 0.05 0.1 0.1
v0 = 88;
TB = 0.01
Revert Help Apply

2. Data Inspector:



3. Solver selection



4. Lookup table