EIGHT DEGREE OF FREEDOM VEHICLE MODEL WITH PITCH, YAW, TIRE CONTROL AND SENSOR INPUTS

A Thesis

Presented to the Faculty of

California Polytechnic State University

San Luis Obispo

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science in Mechanical Engineering

By

Sean T. Hirtle

May 4, 2015

COMMITTEE MEMBERSHIP

TITLE:	Eight Degree of Freedom Vehicle Model with Pitch, Yaw, Tire Control and Sensor Inputs
AUTHOR:	Sean T. Hirtle
DATE SUBMITTED:	May 4, 2015
COMMITTEE CHAIR:	Charles Birdsong, PhD Professor of Mechanical Engineering
COMMITTEE MEMBER:	Joseph Mello, PhD Professor of Mechanical Engineering
COMMITTEE MEMBER:	John Fabijanic, PhD Lecturer of Mechanical Engineering

ABSTRACT

Eight Degree of Freedom Vehicle Model with Pitch, Yaw, Tire Control and Sensor Inputs

By:

Sean T. Hirtle

This research focuses on the development of an eight degree of freedom vehicle model in the MATLAB computing language. Its purpose is to provide flexibility in the modeling and implementation of signal inputs and crash avoidance logic while maintaining accuracy in the physics of the vehicle's motion. Firstly, the equations of motion for the bodies involved under a reasonable set of assumptions were developed. Next the model was translated to computer code. By writing the model in SimuLink with **.m* files, the modularity of the code is enhanced. To validate the model, several well defined tests were simulated.

To establish some form of credibility, the solutions from this model were compared against three independent solution sets. Three different visual correlates were noted: dynamic response, steady state accuracy, and tendency to oscillate in the high frequency domain. The dynamic response of the model was shown to agree with the empirically measured results. Some steady state accuracy arguments were presented, with focus on further development of the tire model. Future research into other finite difference methods were also given.

Regarding three dimensional kinematics, it should be mentioned that this model uses the simplest approximation to a set of partial differential equations allowable, which gives it some form of presentability in the classroom. The method is comprehensible to even the most amateur computational physicist. For the tests presented, this approximation is convergent, and highlights the efficacy of residual methodologies.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1: Introduction

The development of the microchip has allowed technology to blossom in the digital age. High speed computing has given us the ability to combine electro-mechanical controllers with mechanical systems such that complex logic and control can be carried out in real time. One area of research that is being developed heavily is the use of digital controller logic to implement automated crash avoidance in the automotive industry. Over 30,000 humans die yearly due to automobile accidents. Many of these accidents are due to driver error alone (Matsubayashi et al., 2006). Enhanced computation power has given technology the ability to control the vehicle and prevent dangerous situations. This includes, but is not limited to: Adaptive Cruise Control, Electronic Stability Control, Automatic Braking Systems, and Lane Departure Warning Systems.

Crash Avoidance

The first major crash avoidance technology to be introduced on wheeled vehicles was the Anti-Lock Braking System (ABS). This system prevented skidding, and its associated dangers, by improving both the braking performance and controllability of the stopping vehicle. While ABS systems were original entirely mechanical in their design, almost all modern ABS systems make use of an electronic control unit These systems have been very effective since their inception and serve as a platform for even more complex accident avoidance technologies.

One important electronically assisted crash avoidance technology is Electronic Stability Control, originally introduced in luxury brand vehicles in the 1990's. By measuring the yaw rate of the vehicle as well as the rotation rate of all four tires, the stability of a particular maneuver could be determined electronically. If it became apparent to the system that a particular maneuver was becoming unstable, these systems could automatically apply Anti-Lock Braking and steering maneuvers to safely

correct the trajectory of the vehicle. An in depth literature review has shown these systems to be extremely effective in reducing the number and severity of accidents worldwide (Ferguson, 2007).

Adaptive Cruise Control is another technology introduced in the same time frame. The original systems used lasers to measure the distance between a traveling vehicle and those driving in front. If the distance began to shrink at a rate considered dangerous, the throttle/gearing of the vehicle would automatically be adjusted to compensate. Improvements to the system included the introduction of radar, which is more accurate than laser in terms of reflective properties, as well as the application of the brakes to reduce speed at a greater rate (Audi AG, 2009).

Since the turn of the century, several more advanced crash avoidance systems have been introduced in vehicles available to the public. One such system is Toyota's Pre-Collision System. This system uses forward aimed radar to determine if a collision is imminent. If a collision appears unavoidable, the driver is alerted, the brakes are pre-charged for immediate maximum braking, and all slack is removed from the seat belts (Toyota, 2008). This system was introduced in the Lexus LS 430 in 2003. In the same year Honda introduced their Collision Mitigation Brake System. This three stage system also uses radar to determine if a collision is going to occur. The first stage provides an automatic warning to the driver. Stage two uses seatbelt tugs and tactile stimulation to further warn the driver. If the driver fails to heed these warnings, stage three will tighten the seatbelts and automatically begin applying the brakes (Honda, 2003).

Perhaps the most advanced system to date is Toyota's Advanced Pre-Collision System. While similar to PCS, it includes a twin lens stereo camera and an infrared sensor to detect the presence of pedestrians and animals in the trajectory of the vehicle. It can also make use of Adaptive Variable Suspension and Variable Gear Ratio Steering

to enhance the driver's avoidance maneuvers. Known as Collision Avoidance Steering Support, it was introduced with the Lexus LS in 2006 (Toyota, 2006).

Blind Spot Monitoring is another more recently developed system. Using camera or radar, the blind spot of a vehicle can be monitored continuously. If a vehicle is detected within the blind spot, the driver will usually be notified by some form of warning indicator. The 2010 Infinity M will also implement a counter steering measure to prevent blind spot collision accidents (Motor Trend, 2009).

It is clear that several different systems have been developed or already exist that help reduce the number of fatal car accidents. These use optical cameras, infrared sensors, lasers, and radar to measure the environment of the vehicle. Some use visual and auditory signals or tactile stimulation to warn the driver, while others actually modify the mechanical inputs to parts of the vehicle. However, there are other options available. One sensing option that has been investigated at Cal Poly are magnetic sensors. These have the ability to detect the axle of a vehicle and actually verify the vehicle type by measuring the disturbance of the local magnetic field.

Existing Models

Almost every introductory vehicle dynamics textbook presents what is commonly referred to as the *bicycle model* as is shown in Figure 1 (Jazar, 2008). Developing the bicycle model requires several simplifying assumptions. First, the steering angles for the front wheels are assumed to be equal. It is also required that both the lateral and longitudinal forces induced by the tires are modeled linearly. This is only valid for very small steering angles at low acceleration rates. To simplify the model even further, the tire forces occurring at each axle are taken as an average neglecting any effects of dynamically induced weight transfer. By neglecting the body roll the roll-yaw coupling inherent in the general moment equations is lost. As a result the model is only valid for the mildest vehicle maneuvers. In terms of accident avoidance this is a serious problem

because most emergency maneuvers require the use of extreme turning and braking and the nonlinearities will become extremely important in developing an accurate solution. It is obvious that a more complex model is required.

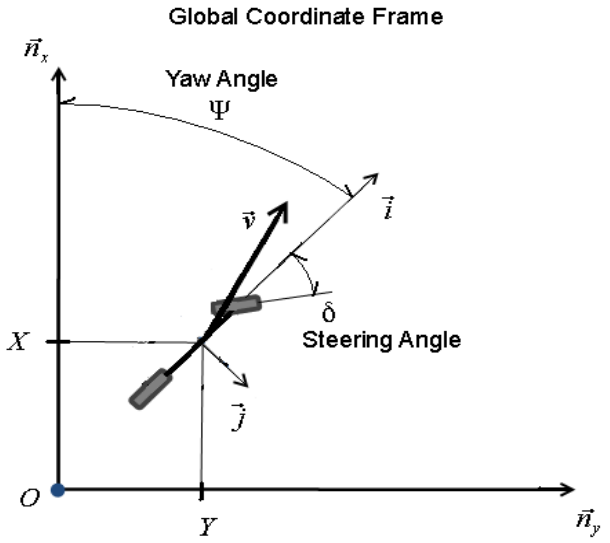

Figure 1: Bicycle Model.

A far more accurate model would incorporate all of the available degrees of freedom in the vehicle, as well as any and all non-linear tire effects. Because there are thousands of moving parts on a typical vehicle, trying to solve for them all will quickly become a physically impossible. By lumping the masses as much as possible (sprung mass/unsprung masses) the model is still accurate but with a much smaller number of degrees of freedom. This would include the three degrees of translational and rotational freedom given to the body, as well as the minimum number needed for an accurate suspension model. In Figure 2 a screen shot of CarSim, a professionally developed vehicle simulating software package, is displayed. The CarSim model would be an excellent example meeting the mentioned requirements. This 14 degree of freedom model includes all three translational and rotational degrees of freedom given to the

vehicle body, as well as one rotational and one translational degree of freedom for each tire.

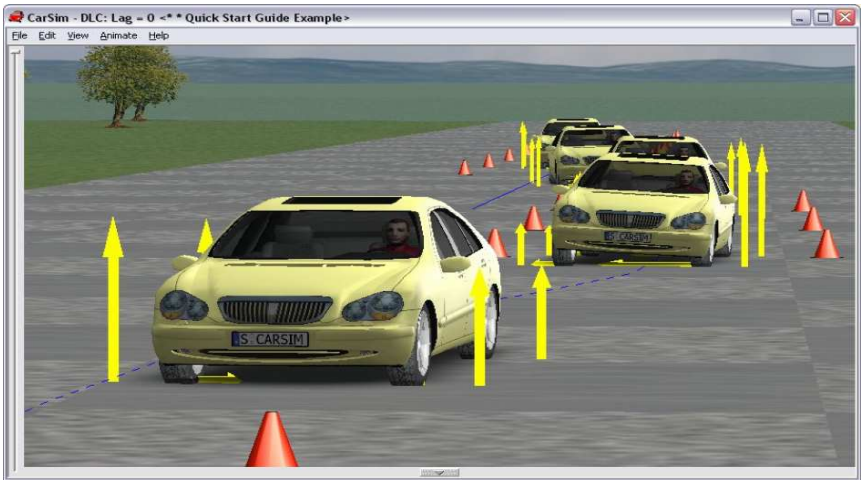

Figure 2: CarSim running a simulation.

The challenge with these models is that they require an extensive amount of suspension design knowledge before the motions of the tires can be accurately solved. Because the tires are essentially modeled as non-linear dampers, any inaccuracies in the suspension model will result in incorrect tire velocities, and the tire forces will not be accurate. This requires the knowledge of suspension link mounting positions and the various degrees of translation/rotational freedom for each link. A model that is more accurate than the bicycle model, but simpler than CarSim would be desirable.

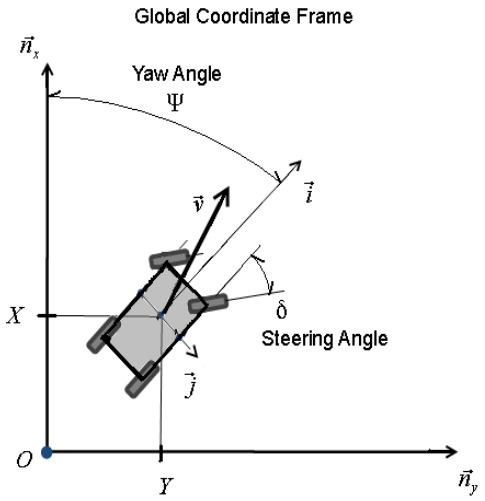

Figure 3: Track Model.

One way to improve the bicycle model is to no longer average the tire normal forces and resulting lateral and longitudinal forces. Pitch, roll, and body bounce are still neglected, but now there are four contact points supporting the vehicle mass and the dynamic weight transfer forces can be taken into consideration (Casanova, 2000). This is commonly referred to as the *track model,* and a representation is presented in Figure 3. The track model with non-linear tires serves as a very solid improvement, but the pitch-roll-yaw coupling is still neglected. Realizing that almost every vehicle suspension carries left-right symmetry there is the possibility to include roll. The roll axis of any vehicle is the line about which the body rolls such that the tires do not have any induced lateral velocities. For some passenger sedans, such as the Ford Taurus, the roll axis can be assumed horizontal (Demerly, 2000). It is then possible to keep track of the roll degree of freedom. It should be noted that most vehicles lack front-rear symmetries in their suspensions due to anti-pitch and anti-lift mechanisms, and thus determining the pitch becomes very difficult. To simplify the model pitch will be neglected, and our equations of motion will only include the roll-yaw coupling. The vertical translational degree of freedom is also neglected, which means that our model is only valid for flat,

smooth roads. The wheel translational degree of freedom is also neglected, but the rotational degrees of freedom will be kept. This can be done by treating the tire spring and the suspension spring as a pair of springs in series. Therefore we have an eight degree of freedom model that covers longitudinal translation, lateral translation, yaw, roll, and the rotation of the four wheels. The list of assumptions is provided below as a summary.

- Longitudinal and lateral velocities
- Roll about *x*-axis
- Yaw about *z*-axis
- Uses smooth, flat road
- Inputs: current states, steering, drive/brake, sensor data
- Outputs: Linear/Angular positions, velocities, rotations, slip angles

Modeling Methods

Aside from determining what degrees of freedom to model, the numerical solution scheme must also be considered. By choosing to use the MATLAB programming language, there are three possibilities. The first possibility is to write an entire solver from scratch using *.m files. This would be a time consuming, though detail oriented task. It can be vastly sped by making use of the numerical solvers already provided by MATLAB. The first sets of solvers available are the Runge-Kutta solvers. While these solvers have already been designed to maximize calculation efficiency, they lack the ability to return the highest order derivative obtained in the solution. For general vehicle motion through space this is typically not a problem, but for modeling collision avoidance it may be necessary to know the given accelerations at any time step. This information could be an input for a particular accident avoidance solver. It is also

necessary for computing the lateral acceleration gain. When validating the model we will also need the acceleration values as the data sets available all make acceleration comparisons. One option available is MATLAB's SimuLink environment. Simulink is commonly used when modeling controllers as it is an environment based on block diagrams. It also allows the accelerations calculated at every time step to be plotted in real time. This was the approach used in the NAVDyn Model (Demerly, 2000); all portions of the model existed as connected block diagrams. However, SimuLink also supports the use of *.m files, and allows the combination of block diagrams and modular programming. This is the method adopted here because it allows for maximum user flexibility without having to write a complex solver from scratch.

Chapter 2: Development of Model

Coordinate System Definition

In order to accurately model the vehicle physics three sets of coordinate axes are necessary. The method recommended in SAE J670e will be adopted as has been done by various other authors (Casanova, 2000), (Demerly, 2000). First, the Earth-fixed coordinate axes *XYZ* are defined and the uppercase letters will be used to denote this system. These axes can be considered an inertial frame of reference. A point of origin (*O)* must be defined as well as the direction of the *X* axis, and the coordinates are orthogonal and right handed.

The vehicle-fixed, chassis coordinate system *xyz* and body coordinate system *x'y'z'* are located at the same point at specified on the vehicle at rest. Assuming the vehicle has lateral symmetry, which is reasonable for most passenger sedans, it is possible to determine the location of the roll axis. This is the axis that allows the body to rotate without any induced velocities in the tires. By projecting a line vertically downward through the vehicle center of gravity,the origin of each system (*o,o'*) is located at the intersection of this line and the roll axis. The advantage of selecting this location is that it provides the simplest means of developing the tire forces. The *x* and *x'* axes are in the longitudinal (forward) direction, the *y* and *y'* axes are in the right hand lateral direction, and the *z* and *z'* axes point vertically downward. The unit vectors for each frame are given as

1. *XYZ*: $\vec{n}_x, \vec{n}_y, \vec{n}_z$
2. *xyz*: $\vec{i}, \vec{j}, \vec{k}$
3. *x'y'z'*: $\vec{i}', \vec{j}', \vec{k}'$

where the unit vectors are listed in x-y-z order. From here, any bold text will be used to indicate a vector. The standard SAE definition of right-hand rotations, starting with *xyz* aligned with *XYZ*, are given by

1. Yaw rotation ψ about the *z*-axis
2. Pitch rotation θ about the *y*-axis
3. Roll rotation φ about the *x*-axis

and these are taken about the vehicle-fixed axes xyz. The x'y'z' axis system will roll about the x-axis and remain fixed to the body. The transform between x'y'z' and xyz will be used when developing the equations involving the body.

Linear Equations of Motion

To begin, the origin of the inertial reference frame is defined. The distance from this origin to the origin of the chassis reference frame is

$$\vec{R}_o = X\ \vec{n}_x + Y\ \vec{n}_y \tag{2.1}$$

where *X* is the x-wise location coordinate, *Y* is the y-wise location coordinate, and $\vec{R}_o$ is the radius with respect to the inertial frame as shown in Figure 4.

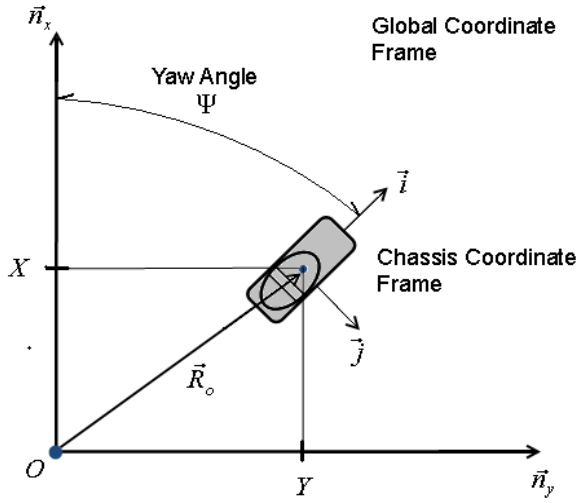

Figure 4: Chassis coordinate frame

This can be transformed into the chassis frame by setting

$$\begin{aligned} \vec{n}_x &= \cos\,\psi\,\vec{i} - \sin\,\psi\,\vec{j} \\ \vec{n}_v &= \sin\,\psi\,\vec{i} + \cos\,\psi\,\vec{j} \end{aligned} \tag{2.2}$$

and by inserting equation (2.2) into equation (2.1) it is shown that the radius is equal to

$$\vec{R}_o = \left(X\cos\,\psi + Y\sin\,\psi\right)\vec{i} + \left(-X\sin\,\psi + Y\cos\,\psi\right)\vec{j} \tag{2.3}$$

and this is given with respect to the chassis unit vectors. The velocity of the chassis origin is given by taking the time derivative of equation (2.3) giving

$$\begin{aligned} \vec{V}_o &= \frac{d\,\vec{R}_o}{d\,t} \\ &= \dot{X}\,\vec{n}_x + \dot{Y}\,\vec{n}_y \\ &= \left(\dot{X}\cos\,\psi + \dot{Y}\sin\,\psi\right)\vec{i} + \left(-\dot{X}\sin\,\psi + \dot{Y}\cos\,\psi\right)\vec{j} \end{aligned} \tag{2.4}$$

and by defining

$$\begin{aligned} V_{ox} &\equiv \dot{X}\cos\,\psi + \dot{Y}\sin\,\psi \\ V_{oy} &\equiv -\dot{X}\sin\,\psi + \dot{Y}\cos\,\psi \end{aligned} \tag{2.5}$$

the velocity equation (2.4) is simplified into

$$\vec{V}_o = V_{ox}\vec{i} + V_{oy}\vec{j} \tag{2.6}$$

which is given in the chassis reference frame. Because the body and chassis systems share the same origin, equation (2.6) defines the linear velocity of the body reference frame as well. To calculate the acceleration of the chassis/body origin the derivative with respect to time is applied again to equation (2.6) which results in

$$\vec{a}_o = \frac{d^2\vec{R}_o}{dt^2} = \frac{\partial \vec{V}_o}{\partial t} + \vec{\Omega}_c \times \vec{V}_o \tag{2.7}$$

To define the angular velocities the fact that the body is free to roll with respect to the chassis must be taken into consideration. Therefore the angular velocity of each is given by

$$\vec{\Omega}_c = \dot{\psi}\vec{k} \tag{2.8}$$

$$\vec{\Omega}_b = \dot{\varphi}\vec{i} + \dot{\psi}\vec{k} \tag{2.9}$$

and by inserting equations (2.4), (2.5), and (2.8) the acceleration of the chassis origin is shown to be

$$\vec{a}_o = \left(\dot{V}_{ox} - \dot{\psi}V_{oy}\right)\vec{i} + \left(\dot{V}_{oy} + \dot{\psi}V_{ox}\right)\vec{j} \tag{2.10}$$

where the symbol Ω represents the rotation vector and the subscripts *c* and *b* represent the chassis and body respectively.

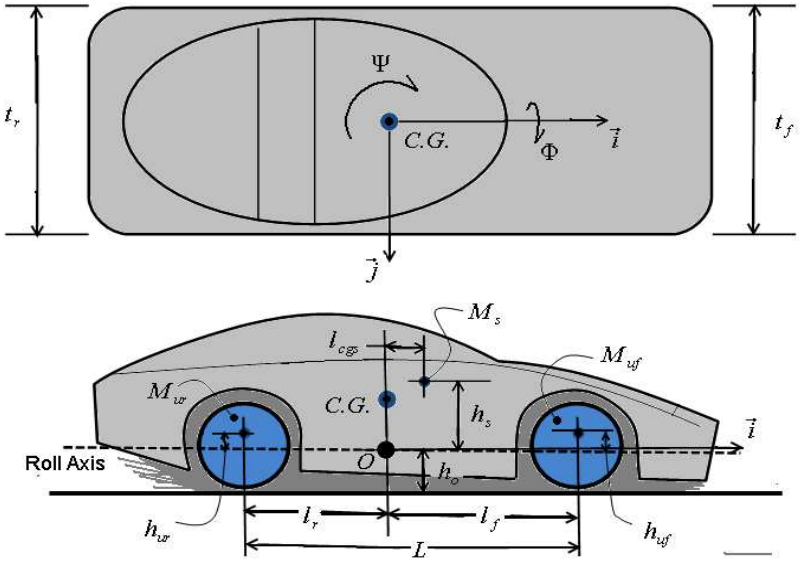

Figure 5: Vehicle geometry

Now that the equation of motion of the chassis coordinate system has been defined, the equations of motion of the sprung and unsprung masses can be considered. The location of the sprung and unsprung masses are displayed in Figure 5. The position of the front unsprung mass can be determined from

$$\vec{R}_{uf} = \vec{R}_o + \vec{r}_{uf} \tag{2.11}$$

where r_{uf} is the radius in the chassis frame. From the position of the front unsprung mass in the chassis coordinate system is shown to be

$$\vec{r}_{uf} = l_f \vec{i} - h_{uf} \vec{k} \tag{2.12}$$

where l_f is the longitudinal distance between the center of the front tire and the vehicle center of mass, and h_{uf} is the height, typically equal to the rolling radius. Now the time derivative of equation (2.11) can be taken to get the front unsprung mass velocity

$$\vec{V}_{uf} = \frac{d\vec{R}_{uf}}{dt} = \vec{V}_o + \frac{\partial \vec{r}_{uf}}{\partial t} + \vec{\Omega}_c \times \vec{r}_{uf} \tag{2.13}$$

and by inserting equations (2.6), (2.8), and (2.12) it is shown that

$$\vec{V}_{uf} = V_{ox}\vec{i} + \left(V_{oy} + l_f \dot{\psi}\right)\vec{j} \tag{2.14}$$

in the chassis frame of reference. The acceleration of the front unsprung mass is derived by taking the time derivative of the equation above to show

$$\vec{a}_{uf}=\frac{d\vec{V}_{uf}}{dt}=\vec{a}_o+\dot{\vec{\Omega}}_c\times\vec{r}_{uf}+\vec{\Omega}_c\times\left(\vec{\Omega}_c\times\vec{r}_{uf}\right) \tag{2.15}$$

and by inserting equations (2.8), (2.10), and (2.12) it is shown that

$$\vec{a}_{uf}=\left(\dot{V}_{ox}-\dot{\psi}V_{oy}-l_f\dot{\psi}^2\right)\vec{i}+\left(\dot{V}_{oy}+\dot{\psi}V_{ox}+l_f\ddot{\psi}\right)\vec{j} \tag{2.16}$$

again in the chassis frame of reference. The exact same process is carried out for the rear unsprung mass where the only change will be in the position vector of the rear unsprung mass in the chassis coordinate system. The sequence is presented below.

$$\vec{R}_{ur}=\vec{R}_o+\vec{r}_{ur} \tag{2.17}$$

$$\vec{r}_{ur}=-l_r\vec{i}-h_{ur}\vec{k} \tag{2.18}$$

$$\vec{V}_{ur}=\frac{d\vec{R}_{ur}}{dt}=\vec{V}_o+\frac{\partial\vec{r}_{ur}}{\partial t}+\vec{\Omega}_c\times\vec{r}_{ur} \tag{2.19}$$

$$\vec{V}_{ur}=V_{ox}\vec{i}+\left(V_{oy}-l_r\dot{\psi}\right)\vec{j} \tag{2.20}$$

$$\vec{a}_{ur}=\frac{d\vec{V}_{ur}}{dt}=\vec{a}_o+\dot{\vec{\Omega}}_c\times\vec{r}_{ur}+\vec{\Omega}_c\times\left(\vec{\Omega}_c\times\vec{r}_{ur}\right) \tag{2.21}$$

$$\vec{a}_{ur}=\left(\dot{V}_{ox}-\dot{\psi}V_{oy}+l_r\dot{\psi}^2\right)\vec{i}+\left(\dot{V}_{oy}+\dot{\psi}V_{ox}-l_r\ddot{\psi}\right)\vec{j} \tag{2.22}$$

It is now necessary to define the linear motion of the sprung mass. The process is similar to that carried out for the unsprung masses with the exception that the sprung mass is free to roll about the roll axis. Therefore it is necessary to project the sprung mass position vector from the body coordinate frame into the chassis coordinate frame. The vector locating the body is shown to be

$$\vec{R}_s=\vec{R}_o+\vec{r}_s \tag{2.23}$$

where r_s is the vector from chassis origin to sprung mass, and R_s is the net position vector. The conversion from body reference frame to chassis reference frame is given by

$$\begin{aligned}\vec{i}'&=\vec{i}\\ \vec{j}'&=\cos\varphi\vec{j}+\sin\varphi\vec{k}\\ \vec{k}'&=-\sin\varphi\vec{j}+\cos\varphi\vec{k}\end{aligned} \tag{2.24}$$

which can be used to redefine the position vector of the sprung mass in the chassis coordinate frame. This is shown to give

$$\begin{aligned}\vec{r}_s &= l_{cgs}\vec{i}' - h_s\vec{k}' \\ &= l_{cgs}\vec{i} + h_s \sin\varphi\vec{j} - h_s\cos\varphi\vec{k}\end{aligned} \tag{2.25}$$

where l_{cgs} is the longitudinal location of the body center of gravity and h_s is the height with the vehicle at rest. The velocity is determined the same way as before and is shown to be

$$\vec{V}_s = \frac{d\vec{R}_s}{dt} = \vec{V}_o + \frac{\partial\vec{r}_s}{\partial t} + \vec{\Omega}_c \times \vec{r}_s \tag{2.26}$$

and by again inserting equations (2.6), (2.8) and (2.25) it is shown that

$$\begin{aligned}\vec{V}_s &= \left(V_{ox} - h_s\dot{\psi}\sin\varphi\right)\vec{i} \\ &+ \left(V_{oy} + h_s\dot{\varphi}\cos\varphi + l_{cgs}\dot{\psi}\right)\vec{j} \\ &+ h_s\dot{\varphi}\sin\varphi\vec{k}\end{aligned} \tag{2.27}$$

Due to the relative motion of the body with respect to the chassis coordinate frame, the acceleration of the sprung mass is slightly different than the acceleration of the unsprung masses, and is given by

$$\vec{a}_s = \frac{d\vec{V}_s}{dt} = \vec{a}_o + \dot{\vec{\Omega}}_c \times \vec{r}_s + \vec{\Omega}_c \times \left(\vec{\Omega}_c \times \vec{r}_s\right) + \frac{2\vec{\Omega}_c \times \partial\vec{r}_s}{\partial t} + \frac{\partial^2\vec{r}_s}{\partial t^2} \tag{2.28}$$

and by substituting in (2.8), (2.10), and (2.25) it is shown that

$$\begin{aligned}\vec{a}_s &= \left(\dot{V}_{ox} - \dot{\psi}V_{oy} - 2h_s\dot{\psi}\dot{\varphi}\cos\varphi - h_s\ddot{\psi}\sin\varphi - l_{cgs}\dot{\psi}^2\right)\vec{i} \\ &+ \left(\dot{V}_{ox} + \dot{\psi}V_{oy} + h_s\ddot{\varphi}\cos\varphi - h_s\dot{\varphi}^2\sin\varphi + l_{cgs}\ddot{\psi} - h_s\dot{\psi}^2\sin\varphi\right)\vec{j} \\ &+ \left(h_s\ddot{\varphi}\sin\varphi + h_s\dot{\varphi}^2\cos\varphi\right)\vec{k}\end{aligned} \tag{2.29}$$

$$\sum F = \frac{d\vec{P}}{dt} = M\,\vec{a}_o \tag{2.30}$$

We can sum the accelerations of the three masses and rearrange the equations to obtain

$$\dot{V}_{ox} = \frac{\sum F_x + M_s\left(-2h_s\dot{\varphi}\dot{\psi}\cos\varphi - h_s\ddot{\psi}\sin\varphi\right)}{M} + \dot{\psi}V_{oy} \tag{2.31}$$

as well as

$$\dot{V}_{oy} = \frac{\sum F_y - M_s\left(-h_s\ddot{\varphi}\cos\varphi + h_s\dot{\varphi}^2\sin\varphi\, h_s\dot{\psi}^2\sin\varphi\right)}{M} - \dot{\psi}V_{ox} \tag{2.32}$$

which are the equations of motion in the longitudinal and lateral coordinates. Noting that the first term in each equation represent the accelerations derived in the *chassis coordinate frame*, while the second term represents the normal acceleration of the chassis frame as it rotates, it is possible to adjust the frame of reference used. Moving the second term from the right side to the left is the equivalent of adopting the rotating frame as the frame of reference. Since this frame is the chassis frame, and the driver is most familiar with the chassis frame, this frame will be adopted for the model. The above are therefore modified to

$$\dot{V}_{ox} = \frac{\sum F_x + M_s\left(-2h_s\dot{\varphi}\dot{\psi}\cos\varphi - h_s\ddot{\psi}\sin\varphi\right)}{M} \tag{2.33}$$

$$\dot{V}_{oy} = \frac{\sum F_y - M_s\left(-h_s\ddot{\varphi}\cos\varphi + h_s\dot{\varphi}^2\sin\varphi + h_s\dot{\psi}^2\sin\varphi\right)}{M} \tag{2.34}$$

by switching from the global frame to the chassis frame. The sum of the forces in the *x* and y directions are given by

$$\sum F_x = F_{xlf} + F_{xrf} + F_{xlr} + F_{xrr} \tag{2.35}$$

$$\sum F_y = F_{ylf} + F_{yrf} + F_{ylr} + F_{yrr} \tag{2.36}$$

and represent the resulting longitudinal and lateral forces provided by the tires. Additional forces, such as aerodynamic drag, can be added as desired.

Angular Equations of Motion

To define the angular motion of the vehicle the sprung mass angular momentum will be defined first. Because the unsprung masses are only permitted to rotate about the

z-axis, their equations of motion are much simpler and will be added later. For the sprung mass the standard definition of the angular momentum is given by

$$\vec{H}_c = I_c \vec{\Omega}_c \tag{2.37}$$

where $\vec{H}_c$ is the angular momentum and *I* is the inertia tensor. It should be recalled that the rotation vector is given in the chassis coordinate frame, but the inertia tensor is defined for the sprung body in the body coordinate frame. Therefore some details must be given for a proper definition. For the sprung mass, the inertia tensor is given as

$$\begin{aligned} I_{sb} &= \begin{bmatrix} I_{xxs} & 0 & I_{xzs} \\ 0 & I_{yys} & 0 \\ I_{zxs} & 0 & I_{zzs} \end{bmatrix} \\ &+ \begin{bmatrix} M_s h_s^2 & 0 & M_s h_s l_{cgs} \\ 0 & M_s\left(h_s^2 + l_{cgs}^2\right) & 0 \\ M_s h_s l_{cgs} & 0 & M_s l_{cgs}^2 \end{bmatrix} \end{aligned} \tag{2.38}$$

where the subscript *s* indicates that it is the sprung body and the subscript *b* indicates that this is calculated in the body frame of reference. The first term represents the moment of inertia tensor as calculated about the body center of mass. The zero elements on the off diagonal terms arise from the assumed vehicular symmetry when viewed in the *x-y* and *y-z* planes. Because the vehicle lacks symmetry when viewed on the *x-z* plane, this off diagonal term must be included as shown in Figure 6. The second term represents the corrections from the parallel axis theorem and the fact that the actual body origin is not the body center of mass, but slightly behind and below as has been shown.

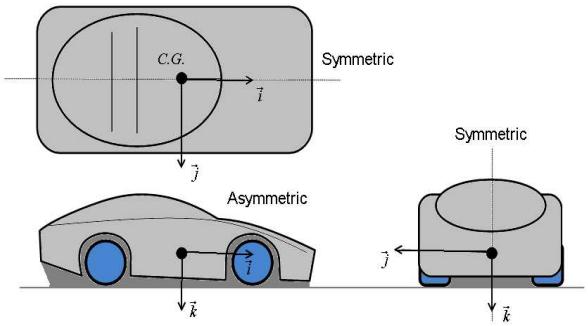

Figure 6: Symmetric and asymmetric planes of vehicle

Now that the moment of inertia tensor has been defined in the body frame of reference, it is necessary that it be projected into the chassis frame of reference. The rotation matrix for the transformation is given by

$$\mathbb{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \tag{2.39}$$

and the transformation itself is shown to be

$$I_{sc} = \mathbb{R}\, I_{sb} \mathbb{R}^T \tag{2.40}$$

Finally, recalling that the vehicle model assumes constant tire contact, rotational motion for an unsprung body about any axis besides the z-axis can be neglected. Therefore, the moment of inertia terms for the unsprung masses can be set to zero if they are not related to pure z-axis rotation, and represent a limitation of the current model. The terms added under these operating conditions are shown to be

$$I_{usc} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_{zzuf} + I_{zzur} + M_{uf} l_f^2 + M_{ur} l_r^2 \end{bmatrix} \tag{2.41}$$

where the *I* terms are the moments of inertia about the unsprung mass centers and the terms that follow represent the parallel axis corrections. Therefore the final moment of inertia tensor is shown to be

$$I_c = I_{sc} + I_{usc} \tag{2.42}$$

Taking the time derivative of equation and recalling that the pitch degree of freedom is neglected, we can show that

$$\begin{aligned}
\frac{\partial \vec{H}_c}{\partial t} &= \frac{d\vec{H}_c}{dt} + \vec{\Omega}_c \times \vec{H}_c \\
&= \left(\dot{H}_x - H_y \psi\right)\vec{i} + \left(\dot{H}_z + H_y \varphi\right)\vec{k} \\
&= \begin{pmatrix} \left(I_{xxs} + M_s h_s^2\right)\varphi + \left(I_{zxs} + M_s h_s l_{cgs}\right)\cos\varphi\,\ddot{\psi} \\ -\left(I_{zxs} + M_s h_s l_{cgs}\right)\sin\varphi\,\dot{\psi}\,\varphi \\ -\left(I_{zzs} - I_{yys} - M_s h_s^2\right)\sin\varphi\cos\varphi\,\dot{\psi}^2 \end{pmatrix}\vec{i} \\
&+ \begin{pmatrix} \left(I_{zxs} + M_s h_s l_{cgs}\right)\cos\varphi\,\varphi \\ +\left[\left(I_{yys} + M_s\left(h_s^2 + l_{cgs}^2\right)\right)\sin^2\varphi + \left(I_{zzs} + M_s l_{cgs}^2\right)\cos^2\varphi\right]\ddot{\psi} \\ +\left(I_{zxs} + M_s h_s l_{cgs}\right)\sin\varphi\,\dot{\varphi}^2 \\ +\left(I_{zzs} - I_{yys} - M_s h_s^2\right)\sin\varphi\cos\varphi\,\dot{\psi}\,\dot{\varphi} \end{pmatrix}\vec{k}
\end{aligned} \tag{2.43}$$

where the two terms represents the moments about the *x* and *z*-axes respectively.

Having defined the time derivative of the angular momentum, and recalling that

$$\sum M = \frac{\partial \vec{H}_c}{\partial t} = \frac{d\vec{H}_c}{dt} + \vec{\Omega}_c \times \vec{H}_c \tag{2.44}$$

the following equations of motion can be extracted. The first equation is

$$\ddot{\varphi} = \frac{\begin{pmatrix} \sum T_{xs} - \left(I_{xzs} - M_s h_s l_{cgs}\right)\cos\varphi\,\ddot{\psi} \\ +\left(I_{xzs} - M_s h_s l_{cgs}\right)\sin\varphi\,\dot{\varphi}\,\dot{\psi} \\ +\left(I_{zzs} - I_{yys} - M_s h_s^2\right)\sin\varphi\cos\varphi\,\dot{\psi}^2 \end{pmatrix}}{I_{xxs} + M_s h_s^2} \tag{2.45}$$

which is the roll angular acceleration and the second equation is

$$\ddot{\psi} = \frac{\begin{pmatrix} \sum T_z - \left(I_{xzs} - M_s h_s l_{cgs}\right)\cos\varphi\,\ddot{\varphi} \\ -\left(I_{xzs} - M_s h_s l_{cgs}\right)\sin\varphi\,\dot{\varphi}^2 \\ +\left(I_{zzs} - I_{yys} - M_s h_s^2\right)\sin\varphi\cos\varphi\,\dot{\psi}^2 \\ -M_s h_s a_x \sin\varphi \\ -\left(I_{zzs} - I_{yys} - M_s h_s^2\right)\sin\varphi\cos\varphi\,\dot{\varphi}\,\dot{\psi} \end{pmatrix}}{I_{zzo}} \tag{2.46}$$

which is the yaw angular acceleration. The sum of the torques acting on the sprung mass about the *x* axis is shown to be

$$\sum T_{xs} = T_{\varphi f} + T_{\varphi r} + M_s g h_s \sin\varphi \tag{2.47}$$

where

$$T_{\varphi f} + T_{\varphi r} = -\left(K_{\varphi f} + K_{\varphi r}\right)\varphi - \left(B_{\varphi f} + B_{\varphi r}\right)\dot{\varphi} \tag{2.48}$$

which represent the roll springing and roll damping combined. The sum of torques about the *z* axis are given by

$$\begin{aligned} \sum T_z &= \left(F_{ylf} + F_{yrf}\right) l_f - \left(F_{ylr} + F_{yrr}\right) l_r \\ &+ \left(F_{xlf} - F_{xrf}\right)\frac{t_f}{2} + \left(F_{xlr} - F_{xrr}\right)\frac{t_r}{2} \\ &+ M_{zlf} + M_{zrf} + M_{zlr} + M_{zrr} \end{aligned} \tag{2.49}$$

where the first four terms are the moments developed by the tire forces about the body and the last four terms are the tire self aligning moments.

Dynamic Weight Transfer Force

Having derived the equations of motion for both translation and rotation, it is obvious that the summation of forces and moments will be necessary in order for each time step to be evaluated. These forces result from interactions of the tires with the ground, and are dependent upon the tire normal force. Because the chassis coordinate system is not an inertial coordinate system, a handful of correction accelerations must be supplied before the coordinate system is valid. The normal acceleration terms used to simplify the longitudinal and lateral equations of motion are one set of accelerations that will develop dynamic weight transfer in the vehicle. The other accelerations are the tangential accelerations provided by driving or braking the tires. These accelerations serve to cancel whatever acceleration is being experienced by the frame of reference and therefore make it an inertial system and valid for the Newtonian laws of physics.

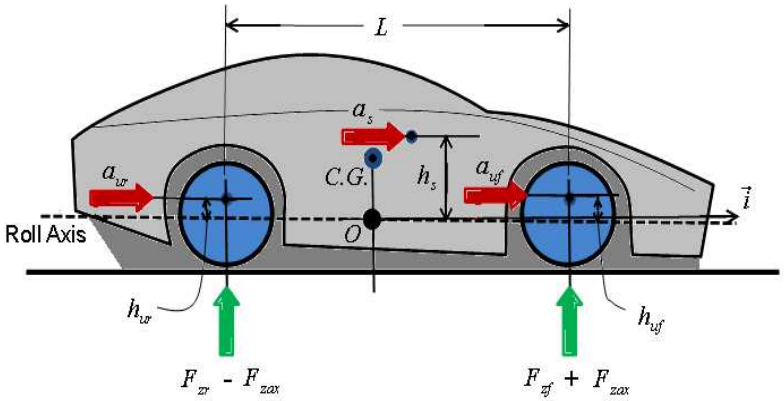

Figure 7: Longitudinal weight transfer force.

The longitudinal correcting accelerations are presented in Figure 7, and the resulting normal forces can be computed by the solution of a moment equation. The resulting normal force correction is given by

$$F_{zax} = \frac{M_s h_s a_{sx} + M_{uf} h_{uf} a_{ufx} + M_{ur} h_{ur} a_{urx}}{2\mathrm{L}} \tag{2.50}$$

and these terms can be understood as the moment balance for the system shown. For lateral acceleration across the front of the vehicle the normal force compensation is shown to be

$$F_{zayf} = \frac{1}{t_f}\left(\frac{M_s l_r h_f}{2\mathrm{L}} a_{sy} + M_{uf} h_{uf} a_{ufy}\right) \tag{2.51}$$

and the rear normal force compensation is shown to be

$$F_{zayr} = \frac{1}{t_r}\left(\frac{M_s l_f h_r}{2\mathrm{L}} a_{sy} + M_{ur} h_{ur} a_{ury}\right) \tag{2.52}$$

To compensate for the normal force due to roll it is shown that the necessary force is equal to

$$F_{z\varphi f} = \frac{-1}{t_f}\left(K_{\varphi f}\varphi + B_{\varphi f}\dot{\varphi}\right) \tag{2.53}$$

for the front half of the vehicle and

$$F_{z\varphi r} = \frac{-1}{t_r}\left(K_{\varphi r}\varphi + B_{\varphi r}\dot{\varphi}\right) \tag{2.54}$$

for the rear half of the vehicle. The normal forces on each tire are then found by applying the above equations as follows

$$F_{zlf} = \frac{M\,g\,l_r}{2\mathrm{L}} - F_{zax} + F_{zayf} + F_{z\varphi f} \tag{2.55}$$

$$F_{zrf} = \frac{M\,g\,l_r}{2\mathrm{L}} - F_{zax} - F_{zayf} - F_{z\varphi f} \tag{2.56}$$

$$F_{zlr} = \frac{M\,g\,l_f}{2\mathrm{L}} + F_{zax} + F_{zayf} + F_{z\varphi f} \tag{2.57}$$

$$F_{zrr} = \frac{M\,g\,l_f}{2\mathrm{L}} + F_{zax} - F_{zayf} - F_{z\varphi f} \tag{2.58}$$

Tire Models

With the normal force on each wheel determined, the longitudinal and lateral tire forces can be derived. The processes involved in the generation of actual tire forces is remarkably complex and computationally expensive to pursue. Fortunately, several different methods exist that can approximate the behavior of these forces under varying conditions. One of the simplest tire models is known as the linear model. Under this model, the lateral and longitudinal tire forces are considered to vary linearly with slip angle and slip respectively. The slip angle of a tire is taken to be

$$\alpha = \tan^{-1}\left(\frac{v_y}{v_x}\right) \tag{2.59}$$

which measures the angle of the tires net velocity to the direction the tire is facing as shown in Figure 8.

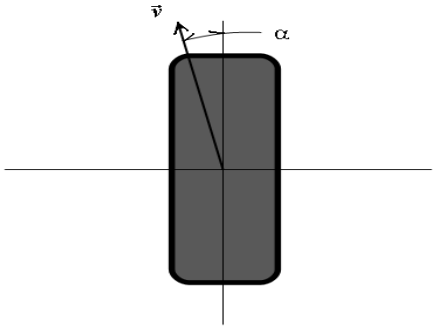

Figure 8: Tire velocity and slip angle

The net slip of a tire is given by the equation

$$slip = \frac{|v_T - v|}{max(v_T, v)} \tag{2.60}$$

which is a normalization of the velocity difference between the tire and the ground. The linear model is valid only for small slip and slip angles as the tire behavior quickly becomes nonlinear.

Empirical tire functions have been developed to handle the observed nonlinearities in the measured data. One of these is known as the *Pacejka Model* (Jazar, 2008). The equation set used by the model is

$$F_y = A \sin\left(B \tan^{-1}\left(Cx - D\left(Cx - \tan^{-1}(Cx)\right)\right)\right) \tag{2.61}$$

$$A = \mu F_z \tag{2.62}$$

$$C = \frac{C_\alpha}{AB} \tag{2.63}$$

$$B, D = shape\ factors \tag{2.64}$$

where C_α is the cornering stiffness of the tire, x is considered the slip variable, and the different constants allow for the development of accurate force approximations over a wide range of operating conditions.

A second nonlinear tire function that makes use of sin functions is displayed in the following set of equations

$$F_y = A \sin\left(B \tan^{-1}\left(C\Phi\right)\right) \tag{2.65}$$

$$\Phi = \left(1-E\right)\left(\alpha+\delta\right)\mu F_z \tag{2.66}$$

$$C = \frac{C_\alpha}{AB} \tag{2.67}$$

$$C_\alpha = C_1 \sin\left(2\tan^{-1}\left(\frac{F_z}{C_2}\right)\right) \tag{2.68}$$

$$A, B = shape\ \ factors \tag{2.69}$$

$$C_1 = Maximum\ \ cornering\ \ stiffness \tag{2.70}$$

$$C_2 = Tire\ \ load\ \ at\ \ maximum\ \ cornering\ \ stiffness \tag{2.71}$$

where again the functions are entirely empirical and highly tunable to measured data. This model has been recommended as effective for computational use (Jazar, 2008). For the purposes of this investigation, we have used two linear functions to model a tire's lateral and longitudinal response.

Chapter 3: Program Structure

SimuLink Model

The equations of motion derived at this point are highly nonlinear and cannot be solved analytically. In order to obtain an approximate solution, some numerical scheme will have to be adopted. SimuLink, a professionally developed extension to MATLAB, allows for the approximation of large, highly nonlinear systems via the method of block diagrams. It also allows for the use of logic in a manner similar to the use of *.m files. The initial solving scheme for the vehicle model is shown in Figure 9.

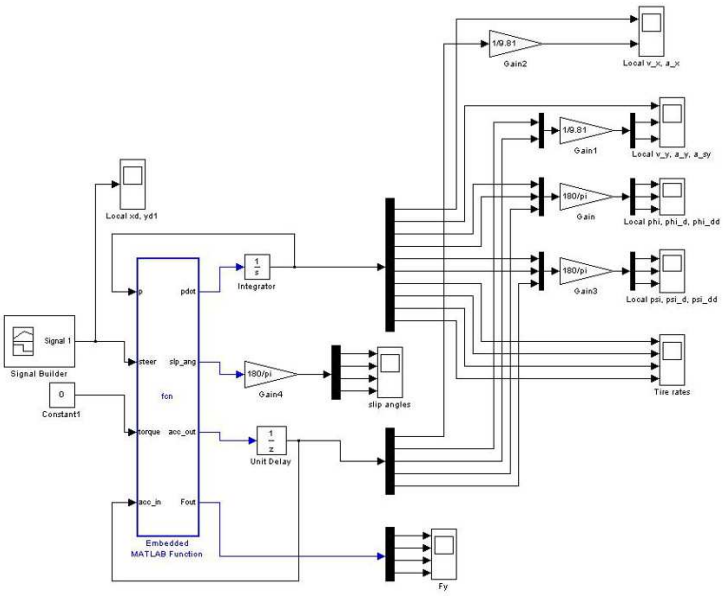

Figure 9: Simulink Diagram

The large blue rectangle represents the main *function* within the loop. It receives four inputs: the state vector *p,* the steering signal *steer*, the braking signal *brake*, and a vector of the previously calculated accelerations *acc_in*. The outputs of the function are the time derivatives of the state vector *pdot*, the slip angles *slp_ang*, the calculated accelerations *acc_out*, and the tire forces *Fout*. The derivatives are run through an integrating block and fed back into the function as the new state vector. The

accelerations are also fed directly back into the function. The rest of the outputs are simpy modified and plotted as necessary.

The steering input *steer* is fed by a signal building block. This builder allows any signal to be sketched on a time axis and fed into the system, giving the user some flexibility in steering choices. The initial conditions for the system are stored in the integrator. The function block itself has a large amount of code stored inside. The solving procedure for each time step is as follows:

- Define Car parameters
- Obtain state vectors
- Calculate accelerations experienced by sprung/unsprung masses
- Calculate the dynamic weight transfer forces
- Determine the longitudinal and lateral tire forces
- Determine the sum of forces and moments
- Update State

This list is to be repeated until the amount of simulation time required for the test has elapsed. By varying the steering input, brake inputs, and initial conditions it is possible to generate a wide range of driving situations.

Chapter 4: Program Validation

Step Steer Test

In order to validate the model developed an outside data set will be required. There are three different sets of data that this system can be compared to. The first set are the results of a CarSim analysis carried out on a 1990's model Ford Taurus. The second set are the solutions to the NAVDyn block diagram simulator. This model was written entirely in SimuLink using block diagrams. The third data set arises from actual accelerometer measurements taken off a real Ford Taurus.

The first validation test used was the step steer test. Under this test, the vehicle starts at a specific speed with the steering wheel at the zero angle position. At some point in time, the steering wheel 'steps' from zero to whatever value is desired. The first test performed was for a wheel turn of 42 degrees at a speed of 40 kph. The results are displayed in Figure 10. Solutions obtained from the model developed here appear on the left while the three sets used in this comparison are displayed on the right.

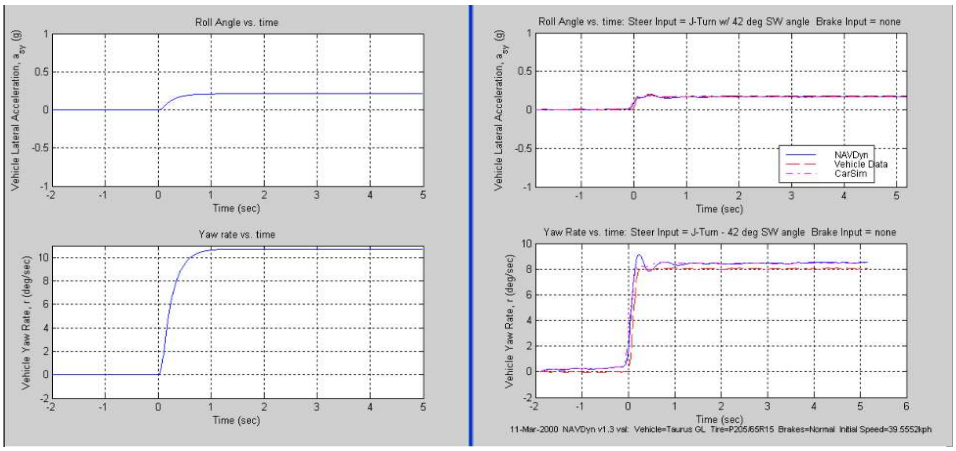

Figure 10: Sprung mass lateral acceleration and yaw rate vs. time for a 42 degree steering wheel step turn at 40 kph.

From the initial test it is shown that the dynamic solution obtained with this model generally matches the behavior observed in the other sets. The lateral acceleration of

the sprung mass steps up to a value similar to the actual vehicle and in the same time frame, and the same can be said for the yaw rate. Noting the differences between various solutions, the NAVDyn yaw rate solution is much more oscillatory than any of the others. A similar set of observations comes from the 142 degree steering wheel turn at 40 kph as shown in Figure 11. The roll angle and roll rates for this test are also displayed in Figure 12. It is interesting to note the differences between the final roll angles for the four different solutions. One thing that stands out is the difference between the final value between our model and the ones we are comparing it to. This difference may suggest an improved, non-linear tire model for computing the given longitudinal and lateral forces. This would induce a change in the steady state roll value obtained. Interestingly, our model happens to be much smoother, with curvature changes that more closely emulate the measured data. NavDyn's outputs are oscillatory in nature, a form of behavior not supported by the measured empirical data outside the measured fluctuations.

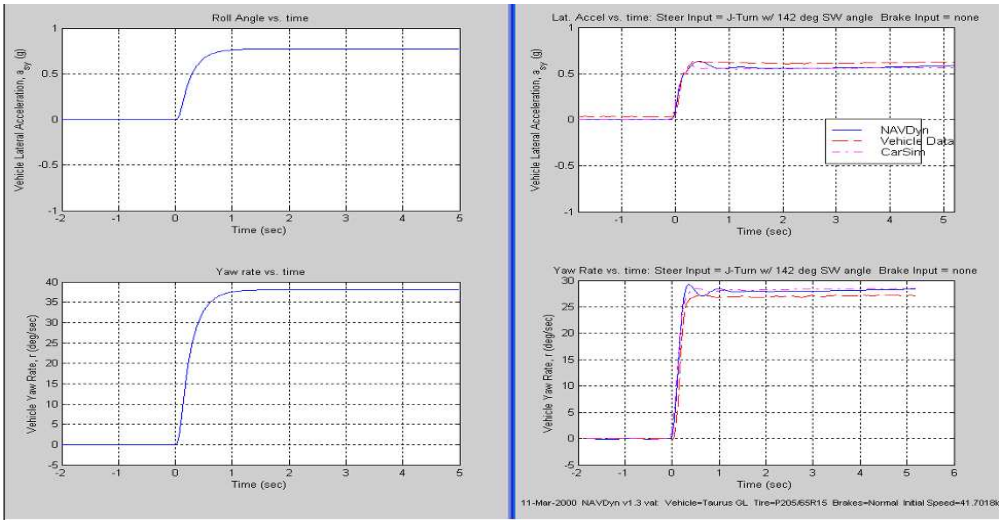

Figure 11: Sprung mass lateral acceleration and yaw rate vs. time for a 142 degree steering wheel step turn at 40 kph

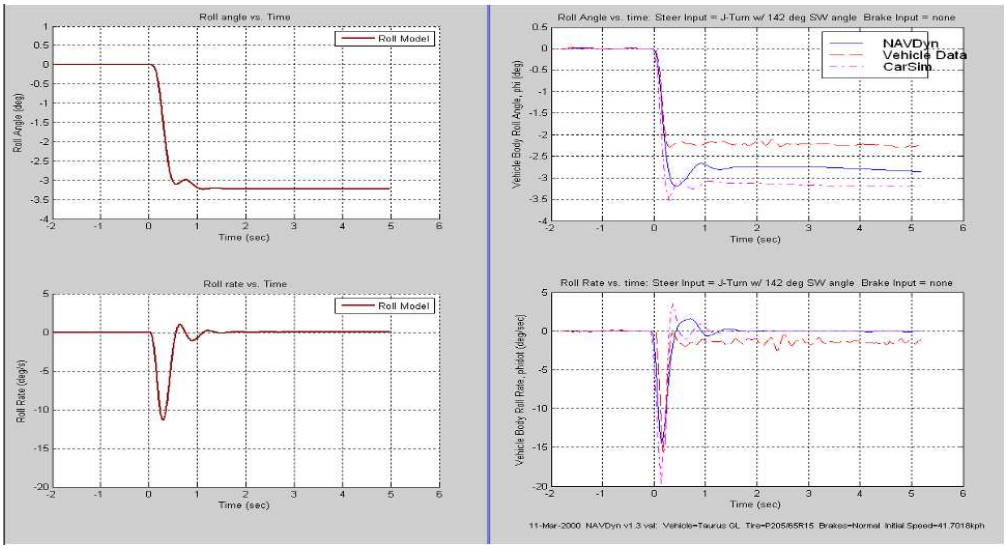

Figure 12: Roll angle and roll rate vs. time for a 142 degree steering wheel step at 40 kph.

Chapter 5: Conclusions and Further Research

The model developed in this paper was compared against three separate data sets and shown to give the same general dynamics for the step steer tests. Because the model is written in SimuLink, as well as *.m files, any reader can have access to the core code. It is relatively simple for the model to be expanded or modified as desired by any curious user highlighting the advantages of a modular solution.

It should be noted that most of the physics developed in this model are only the simplest cases. More complex formulas for tire force generation, tire self aligning moments, and unequal steering angles could be developed. Due to the flat road assumption, any road banking would require that the equations of motion be modified accordingly. Regarding integrations over time, there is some error introduced by Euler's method that decays with resolution in time. More accurate integrators could be used. A common scheme used in MATLAB is Runge-Kutta's fourth order method. Unfortunately, for a more complex state change such as the one observed in our vehicle model, this method is not so readily adapted. A second order method, commonly referred to as Heun's method, is appealing because of it's avoidance of a half-step in time. These methods are free to be pursued by the interested reader.

This model, along with all of it's undeveloped extensions, will give the researcher an artificial vehicle to manipulate digitally at will. This could include accident detection and avoidance, traction control methodologies, fastest lap times, and drifting controllers. These fields of knowledge are difficult, if not prohibitively expensive to conduct in reality. The ability to translate complex phenomena into a handful of mathematical relationships carries weight in the engineering community as it is as profitable as it is rare. Computation analysis allows one to approximate laboratory experiments in the digital domain. Anytime this can be achieved in a reasonable period of time, it will be cheaper

than the alternative. All of the MATLAB code required to run the presented experiments has been provided in the Appendix.

References

Audi AG. (2009). *The New Audi A8.* Product Communications, Ingolstadt.

Breuer, J. J. *Real World Safety Benefits of Brake Assistance Systems.* DaimlerChrysler AG.

Casanova, D. (2000). *On Minimum Time Vehicle Manoeuvring: The Theoretical Optimal Lap.*

Dixon, J. C. (1996). *Tire, Suspension and Handling.* Warrendale: Society of Automotive Engineers.

Ferguson, S. A. (2007). *The Effectiveness of Electronic Stability Control in Reducing Real-World Crashes: A Literature Review.* Traffic Injury Prevention, Volume 8, Issue 4, pp. 329-338.

Honda. (2003, May 20). *Honda Develops World's Firt 'Collision Mitigation Brake System' (CMS) for Predicting Rear-end Collisions and Controlling Brake Operations*. Retrieved May 6, 2010, from http://world.honda.com/news/2003/4030520.html

Jazar, R. N. (2008). *Vehicle Dynamics: Theory and Application.* New York: Springer.

Matsubayashi et al. *Development of Rear Pre-Crash Safety System For Rear-End Collisions.* Toyota Motor Corporation.

Meriam, J. L. (1987). *Engineerig Mechanics: Dynamics.* New York: John Wiley and Sons.

Reimpell, J. S. (2001). *The Automotive Chassis: Engineering Principles.* Warrendale: Elsevier.

Toyota. (2008, January 22). *Toyota Enhances Pre-crash Safety System with Eye Monitor*. Retrieved May 2, 2010, from http://www.toyota.co.jp/en/news/08/0122.html

.

Appendix A: MATLAB Code

```
function [pdot,slp_ang,acc_out,Fout] = fcn(p,steer,torque,acc_in)

%==output==%
pdot=zeros(10,1);

%==constants==%
g = 9.81;
%-------------%

%==current states==%
v_x = p(1);
v_y = p(2);
phi = p(3);
phi_d = p(4);
psi = p(5);
psi_d = p(6);

a_x = acc_in(1);
a_y = acc_in(2);
phi_dd = acc_in(4);
psi_dd = acc_in(5);
%------------------%

%Car physical variables
l_fs = 1.01476;
l_rs = 1.67524;
t_f = 1.540;
t_r = 1.530;
h_f = 0.130;
h_r = 0.110;
h_cgs = 0.567851;
h_cguf = 0.320;
h_cgur = 0.320;
M = 1704.7;
M_uf = 98.1;
M_ur = 79.7;
I_xxs = 440.911;
I_xys = 0;
I_xzs = 7.54097;
I_yys = 2498.900;
I_yzs = 0;
I_zzs = 2619.28;
I_zzuf = M_uf*(t_f/2)^2;
I_zzur = M_ur*(t_r/2)^2;
I_tlf = 0.99;
I_trf = 0.99;
I_tlr = 0.99;
I_trr = 0.99;

%spring/dampers
K_spf = 27.85; %N/mm
K_spr = 18.16; %N/mm

B_f = 2.9915; %N-s/mm
B_r = 2.9915; %N-s/mm

%anti_roll bars
K_rf = 384.0*180/pi; %Nm/rad
```

```
K_rr = 344.4*180/pi;

%roll stiffness
K_phif = 0.766*K_spf*1000*t_f^2/2 + K_rf;%
K_phir = 0.827*K_spr*1000*t_r^2/2 + K_rr;%
K_phi = (K_phif + K_phir);

%roll damping
B_phif = 0.766*B_f*1000*t_f^2/2; %Nm-s/rad
B_phir = 0.827*B_r*1000*t_r^2/2;
B_phi = (B_phif + B_phir);

K_sr = 15.97; %steering-to-wheel angle ratio

%==calculated parameters==%
L = l_fs + l_rs;
M_s = M - M_uf - M_ur;
l_cgs = (M_ur*l_rs - M_uf*l_fs)/M;
l_f = l_fs + l_cgs;
l_r = l_rs - l_cgs;
M_f = M*l_r/L;
M_r = M*l_f/L;
h_o = h_f + l_f*(h_r - h_f)/L;
h_s = h_cgs - h_o;
h_uf = h_cguf - h_o;
h_ur = h_cgur - h_o;
h_cg = (M_s*h_cgs + M_uf*h_cguf + M_ur*h_cgur)/M;
%----------------------------%

%==tire properties==%
varx = [...
    0.0, 0.0;...
    0.1, 0.6;...
    0.15, 0.85;...
    1.0, 0.4]; %slip curve coordinates (slip-%, mu)
vary = [...
    0.0, 0.0;...
    0.08, 0.4;...
    0.15, 0.6;...
    0.3, 0.8;...
    1.0, 0.4]; %slip curve coordinates (slip-%, mu)
varM = [...
    0.0, 0.0;...
    0.1, 0.0005;...
    0.3, 0.00025;...
    1.0, 0.0]; %slip angle (slip-angle, Moment-arm)


%tire properties
t_wid = 0.175;
t_rad = 0.292;
t_pressure = 275000;

%---------------------%

%==steering inputs==%
d_steer = steer;
%-------------------%

%==calculate internal accelerations==%
a_sx = a_x ...
```

```
     - psi_d*v_y ...
     - 2*h_s*phi_d*psi_d*cos(phi) ...
     - l_cgs*psi_d^2;
a_sy = psi_d*v_x ...
     + l_cgs*psi_dd ...
     + h_s*cos(phi)*phi_dd ...
     - h_s*phi_d^2*sin(phi) ...
     - h_s*psi_d^2*sin(phi);
a_ufx = a_x - l_f*psi_d^2;
a_ufy = a_y + l_f*psi_dd;
a_urx = a_x + l_r*psi_d^2;
a_ury = a_y - l_r*psi_dd;
%--------------------------------------%

%==rotation Matrix==%
ROT = [1,0,0;
       0,cos(phi),sin(phi);
       0,-sin(phi),cos(phi)];

I_b = [I_xxs+M_s*h_s^2,I_xys,I_xzs+M_s*h_s*l_cgs;...
       I_xys,I_yys+M_s*(h_s^2+l_cgs^2),I_yzs;...
       I_xzs+M_s*h_s*l_cgs,I_yzs,I_zzs+M_s*l_cgs^2];

I_c = ROT*I_b*ROT'; %chassis M.o.I. tensor

I_zzus = M_uf*l_f^2 + ...
         M_ur*l_r^2 + ...
         I_zzuf + I_zzur;

I_xxo = I_c(1,1);
I_zzo = I_c(3,3) + I_zzus; %effective I_zz
%-----------------------------------------------%

%==Dynamic Weight Transfer Forces==%
Fzax = (M_s*h_s*a_sx + M_uf*h_uf*a_ufx + M_ur*h_ur*a_urx)/(2*L);
Fzayf = 1/t_f*(M_s*l_r*h_f/L*a_sy + M_uf*h_uf*a_ufy);
Fzayr = 1/t_r*(M_s*l_f*h_r/L*a_sy + M_ur*h_ur*a_ury);
Fzphif = -1/t_f*(K_phif*phi + B_phif*phi_d);
Fzphir = -1/t_r*(K_phir*phi + B_phir*phi_d);
%----------------------------------%

%==Tire Normal Forces==%
Fzlf = M*g*l_r/(2*L) - Fzax - Fzayf + Fzphif;
Fzrf = M*g*l_r/(2*L) - Fzax + Fzayf - Fzphif;
Fzlr = M*g*l_f/(2*L) + Fzax - Fzayr + Fzphir;
Fzrr = M*g*l_f/(2*L) + Fzax + Fzayr - Fzphir;
%----------------------%

%==Tire Longitudinal/Lateral Forces==%

%left front wheel velocity
v_xlf = v_x + psi_d*t_f/2;
v_ylf = v_y + psi_d*l_f;

%right front wheel velocity
v_xrf = v_x - psi_d*t_f/2;
v_yrf = v_y + psi_d*l_f;

%left rear wheel velocity
v_xlr = v_x + psi_d*t_r/2;
v_ylr = v_y - psi_d*l_r;
```

```
%right rear wheel velocity
v_xrr = v_x - psi_d*t_r/2;
v_yrr = v_y - psi_d*l_r;

%tire updates
omega_lf = (v_xlf^2 + v_ylf^2)^(1/2);
omega_rf = (v_xrf^2 + v_yrf^2)^(1/2);
omega_lr = (v_xlr^2 + v_ylr^2)^(1/2);
omega_rr = (v_xrr^2 + v_yrr^2)^(1/2);

%==Linear Cruise Control=%
if v_x < 11.1
    if omega_lf > omega_rf
        Mtlf = (11.1 - v_x)*200;
        Mtrf = 0;
    else
        Mtlf = 0;
        Mtrf = (11.1 - v_x)*200;
    end
else
    Mtlf = 0;
    Mtrf = 0;
end

Mtlf = 0;
Mtrf = 0;
Mtlr = 0;
Mtrr = 0;
%-------------------------%

%get steering angle
delta = d_steer/(180*K_sr)*pi; %radians
beta = atan(v_y/v_x);
beta_lf = atan(v_ylf/v_xlf);
beta_rf = atan(v_yrf/v_xrf);
beta_lr = atan(v_ylr/v_xlr);
beta_rr = atan(v_yrr/v_xrr);

%Interpolate forces
%Linearized Force Coefficients (C_alpha)
C_alf = vary(2,2)/vary(2,1)*Fzlf;
C_arf = vary(2,2)/vary(2,1)*Fzrf;
C_alr = vary(2,2)/vary(2,1)*Fzlr;
C_arr = vary(2,2)/vary(2,1)*Fzrr;

a_lf = beta_lf - delta;
a_rf = beta_rf - delta;
a_lr = beta_lr;
a_rr = beta_rr;

slp_ang = [a_lf,a_rf,a_lr,a_rr];

%Linear Cruise Control for constant v_x
if v_x < 11.1
    Fxtf = (11.1-v_x)*20000;
else
    Fxtf = 0; %no torque
end

Fxtlf = 0;
```

```
Fxtrf = 0;
Fxtlr = 0;
Fxtrr = 0;

Fytlf = -a_lf*C_alf;
Fytrf = -a_rf*C_arf;
Fytlr = -a_lr*C_alr;
Fytrr = -a_rr*C_arr;


Fxlf = Fxtlf*cos(delta) - Fytlf*sin(delta);
Fxrf = Fxtrf*cos(delta) - Fytrf*sin(delta);
Fxlr = Fxtlr;
Fxrr = Fxtrr;

Fylf = Fytlf*cos(delta) + Fxtlf*sin(delta);
Fyrf = Fytrf*cos(delta) + Fxtrf*sin(delta);
Fylr = Fytlr;
Fyrr = Fytrr;

Fout = [Fylf,Fyrf,Fylr,Fyrr];

Mzlf = 0;
Mzrf = 0;
Mzlr = 0;
Mzrr = 0;
%---------------------------------------%

%Omeaga/Alpha of Wheels
alpha_lf_new = (Fxtlf*t_rad + Mtlf)/I_tlf;
alpha_rf_new = (Fxtrf*t_rad + Mtrf)/I_trf;
alpha_lr_new = (Fxtlr*t_rad + Mtlr)/I_tlr;
alpha_rr_new = (Fxtrr*t_rad + Mtrr)/I_trr;

% alpha_lf_new = 0;
% alpha_rf_new = 0;
% alpha_lr_new = 0;
% alpha_rr_new = 0;


%==Torques==%
T_phif = -K_phif*phi - B_phif*phi_d;
T_phir = -K_phir*phi - B_phir*phi_d;
%-----------%

%==Sum forces and torques==%
sigFx = Fxlf + Fxrf + Fxlr + Fxrr;
sigFy = Fylf + Fyrf + Fylr + Fyrr;
sigTxs = T_phif + T_phir - ...
         M_s*g*h_s*sin(phi) + M_s*h_s*a_sy*cos(phi);
sigTz = (Fylf + Fyrf)*l_f - (Fylr + Fyrr)*l_r + ...
        t_f/2*(Fxlf - Fxrf) + t_r/2*(Fxlr - Fxrr) + ...
        Mzlf + Mzrf + Mzlr + Mzrr;
%--------------------------%

%==Update independent variables==%
a_x_new = (sigFx + ...
           M_s.*(-2*h_s.*phi_d.*psi_d.*cos(phi) - ...
           h_s.*psi_dd.*sin(phi))./M;
a_y_new = (sigFy - ...
           M_s*(-h_s*phi_dd*cos(phi) + ...
```

```
            h_s*phi_d^2*sin(phi) + ...
            h_s*psi_d^2*sin(phi))./M;
phi_dd_new = (sigTxs - ...
            (I_xzs+M_s*h_s*l_cgs)*cos(phi)*psi_dd + ...
            (I_xzs+M_s*h_s*l_cgs)*sin(phi)*phi_d*psi_d + ...
            (I_zzs - I_yys - M_s*h_s^2)*sin(phi)*cos(phi)*psi_d^2)./ ...
            (I_xxs+M_s*h_s^2);
psi_dd_new = (sigTz - ...
            (I_xzs+M_s*h_s*l_cgs)*cos(phi)*phi_dd - ...
            (I_xzs+M_s*h_s*l_cgs)*sin(phi)*phi_d^2 - ...
             M_s*h_s*a_x*sin(phi) - ...
            (I_zzs - I_yys - M_s*h_s^2)*sin(phi)*cos(phi)*psi_d*phi_d)./
...
             I_zzo;

%acceleration outputs
acc_out = [a_x_new,a_y_new,a_sy,phi_dd_new,psi_dd_new];

%X
pdot(1) = a_x_new;
%Y
pdot(2) = a_y_new;
%PHI
pdot(3) = phi_d;
pdot(4) = phi_dd_new;
%PSI
pdot(5) = psi_d;
pdot(6) = psi_dd_new;
%WHEELS
pdot(7) = alpha_lf_new;
pdot(8) = alpha_rf_new;
pdot(9) = alpha_lr_new;
pdot(10) = alpha_rr_new;
%SPRUNG BODY
%----------------------------------%
%end
%nd
%d
%
```