# ME597: Independent Study (Summer 2024)

# AUTONOMOUS HI-SPEED RACECAR SIMULATION

**Nikhil Tiwari**

*Under the guidance of Professor Daniel Williams*

*Department of Mechanical Engineering, Purdue University*

## ABSTRACT

*This independent study paper focuses on developing and refining control systems simulation for an autonomous Hi-Speed Racecar for Indy Autonomous Challenge(IAC) associated with Purdue AI Racing (PAIR) team. It investigates the similarities and differences between two autonomous race car simulators: the Code Generated Simulator (CGS) and PAIRSIM. The study begins with a detailed examination of the mathematical model underlying the Code Generated Simulator (CGS), focusing on its implementation of vehicle dynamics, aerodynamics, and tire models, while providing a comparative overview of PAIRSIM's approach. Key parameters affecting the simulations are identified and their impact on vehicle behavior is analyzed. Through a series of controlled tests and parameter adjustments, the study demonstrates how these simulators can be brought into closer alignment, improving their consistency and reliability for autonomous racing applications. Understeer coefficients, yaw rate gain, slip angles, and cornering stiffness are compared before and after parameter modifications.*

*The results show that with careful tuning, both CGS and PAIRSIM can produce similar and more realistic results, increasing confidence in their use for autonomous race car development. The study concludes with recommendations for further improvements and outlines future research directions in system identification and real-world validation.*

Keywords: Autonomous racing, Indy Autonomous Challenge (IAC), Purdue AI Racing (PAIR), Code Generated Simulator (CGS), PAIRSIM, Understeer coefficient, Yaw rate gain, Slip angles, Cornering stiffness

## INTRODUCTION

Autonomous racing represents the cutting edge of automotive technology, pushing the boundaries of vehicle dynamics, control systems, and artificial intelligence. This project focuses on the critical role of simulation in developing and testing autonomous race cars, specifically comparing two simulation environments: the Code Generated Simulator (CGS) and PAIRSIM.

Accurate simulation is crucial for autonomous racing, as it allows for rapid iteration, safety testing, and performance optimization without the risks and costs associated with real-world testing. However, the fidelity of these simulations depends heavily on the underlying vehicle dynamics models and their implementation.

This study aims to analyze, compare, and improve the CGS and PAIRSIM simulators, with a particular focus on their vehicle dynamics models, key parameters, and their effects on closed-loop control. By understanding and refining these simulation environments, we can better predict and optimize the performance of autonomous race cars in real-world scenarios.

## CODEGEN SIMULATOR (CGS)

Code Generated Simulator (CGS), developed in Simulink, code-generated with ROS wrapped around it. The PassengerVehicle function contains a sub-function "Velocity_dynamics" which contains Vehicle Dynamics model for the Codegen Simulator. It uses a combination of a Non-linear Single track model and Kinematic model. It computes the differential equations governing the vehicle's motion, taking into account the steering input, powertrain forces, external forces, aerodynamics, tire forces, and inertia.

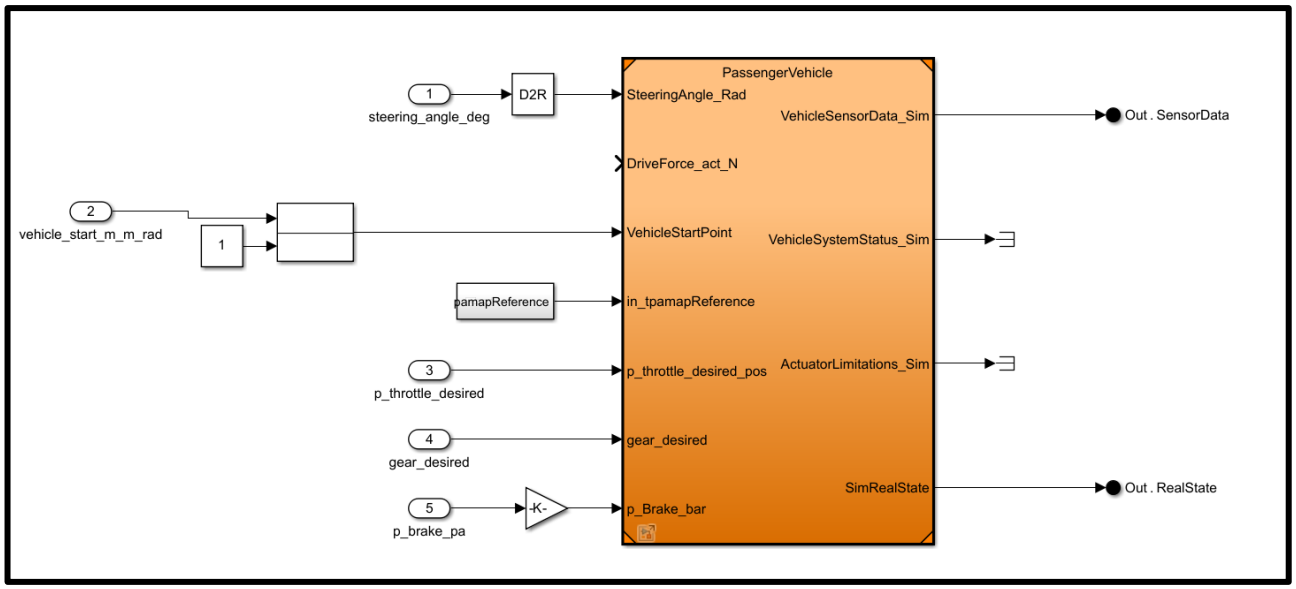
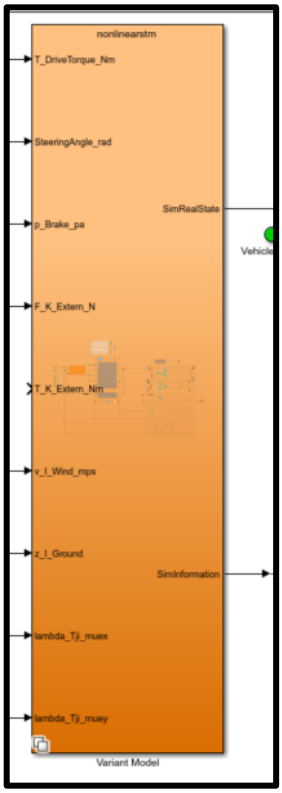



Figure 1 a) – Codegen Simulink Interface



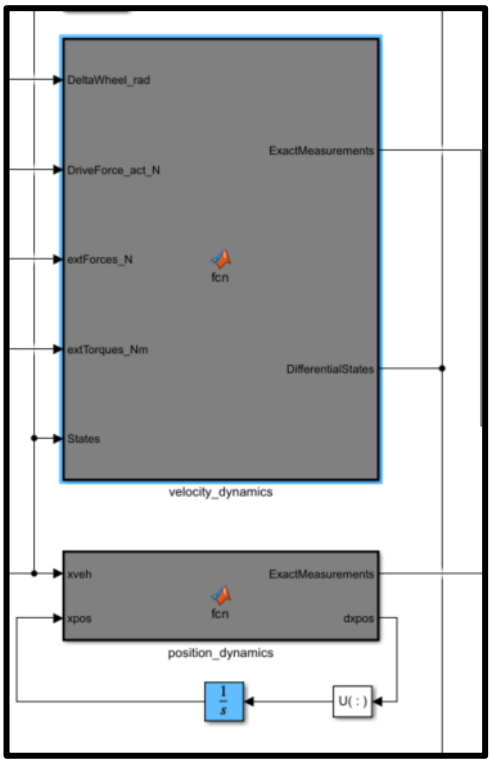




Figure 1 – b) Codegen Non-linear Single track model function c) Codegen Vehicle dynamics function

| Variable names | Description |
|---|---|
| **Track Widths and Lengths** | |
| tw_front_m, tw_rear_m | Track width of the front and rear wheels. |
| l_front_m, l_rear_m | Distance from the center of gravity (CG) to the front and rear axles. |
| l_total_m | Total length of the vehicle. |
| **Mass and Inertia** | |
| m | Mass of the vehicle. |
| J | Yaw moment of inertia. |
| **Aerodynamic and Friction Parameters** | |
| cD | Drag coefficient. |
| roh | Air density (1.22 kg/m³). |
| A_ref | Reference area for aerodynamic calculations. |
| cF | Rolling friction coefficient. |
| cF_max | Maximum rolling friction force. |
| cW_F, cW_R | Front and rear lift coefficients. |
| **Time Constants** | |
| Tvy | Lateral velocity time constant. |
| TdPsi | Yaw rate time constant. |
| T_Tire | Tire relaxation length. |
| **Vehicle Dynamics Parameters** | |
| vx_b, vx_d | Velocity thresholds for model blending. |
| vx_min | Minimum velocity for handling low speed scenarios. |
| PowertrainLimitLowVelocities_N | Powertrain force limits for low velocities. |
| **Tire Parameters** | |
| PacFrontLat, PacRearLat | Pacejka tire model parameters for lateral forces. |
| PacFrontLong, PacRearLong | Pacejka tire model parameters for longitudinal forces. |
| tyreradius_front_m, tyreradius_rear_m | Radii of the front and rear tires. |
| WheelInertia_Front_kgm2, WheelInertia_Rear_kgm2 | Wheel inertias for front and rear tires. |
| PowertrainLimitLowVelocities_N | Powertrain force limits for low velocities. |
| **Tire Parameters** | |
| PacFrontLat, PacRearLat | Pacejka tire model parameters for lateral forces. |
| PacFrontLong, PacRearLong | Pacejka tire model parameters for longitudinal forces. |
| tyreradius_front_m, tyreradius_rear_m | Radii of the front and rear tires. |
| WheelInertia_Front_kgm2, WheelInertia_Rear_kgm2 | Wheel inertias for front and rear tires. |
| **State Variables** | |
| vx_mps, vy_mps | Longitudinal and lateral velocities. |
| dPsi_rad | Yaw rate. |
| omega_rad | Angular velocities of the wheels. |
| lambda_perc | Longitudinal slip ratios. |
| alpha_rad | Slip angles. |
| **Outputs** | |
| DifferentialStates | State derivatives. |
| ExactMeasurements | Measured outputs. |

Table 1 – Codegen Parameters

Codegen begins by computing the powertrain forces. The total force exerted by the powertrain is computed by summing up the individual forces applied to each wheel. It also ensures that forces do not become excessively negative, which may happen during braking.

**Calculation of friction, aero and tire forces in Codegen**

1) Frictional force in x direction:

- Rolling friction force: cF -not consistent with passenger car experience.

$$F_{\text{rolling}} = cF \cdot v_x$$

This is the force due to rolling friction. It is capped at cFmax(=200N) and ensured to be non-negative using max(min(...), 0). cF is not consistent with passenger car model where rolling coefficient is usually a dimensionless number, but here, they have taken cF to be Rolling friction force per unit speed. (units of cF = N/m/s)

- Aerodynamic drag force

$$F_{\text{drag}} = 0.5 \cdot cD \cdot \rho \cdot A_{\text{ref}} \cdot v_x^2$$

This is the force due to air resistance acting on the vehicle.

Combining these:

$$F_{x,\text{friction}} = \max(\min(cF \cdot v_x, cF_{\max}), 0) + 0.5 \cdot cD \cdot \rho \cdot A_{\text{ref}} \cdot v_x^2$$

2) Aero forces in z direction:

- Static weight distribution on rear tires(each):

$$F_{z,\text{rear}} = \left(\frac{m \cdot 9.81 + F_{z,\text{ext}}}{2}\right) \cdot \frac{l_{\text{front}}}{l_{\text{rear}} + l_{\text{front}}}$$

- Aerodynamic lift force on rear tires:

$$F_{\text{lift, rear}} = 0.5 \cdot 0.5 \cdot \rho \cdot cW_R \cdot A_{\text{ref}} \cdot v_x^2$$

The factor 0.5 comes from splitting the lift force between two tires on the rear axle and assuming symmetrical distribution.

- Total normal force on each rear tire: $F_{z,\text{total, rear}} = F_{z,\text{rear}} + F_{\text{lift, rear}}$

And similarly for the front axle:

- Static weight distribution on front tires(each):

$$F_{z,\text{front}} = \left(\frac{m \cdot 9.81 + F_{z,\text{ext}}}{2}\right) \cdot \frac{l_{\text{rear}}}{l_{\text{rear}} + l_{\text{front}}}$$

Note: Fz,ext –does not include roll-moment. In fact, Codegen does not involve Roll Moment. Fz,ext includes banking influence but does not have the banking data to be fed into the simulation runs. So all in all, it is known that Codegen does not have roll moment nor banking considered, and this is a key difference from the other simulator – PAIRSIM.

- Aerodynamic lift force on front tires:

$$F_{\text{lift, front}} = 0.5 \cdot 0.5 \cdot \rho \cdot cW_F \cdot A_{\text{ref}} \cdot v_x^2$$

- Total normal force on front tires(each):

$$F_{z,\text{total, front}} = F_{z,\text{front}} + F_{\text{lift, front}}$$

3) Tire forces in longitudinal and lateral directions(x,y) :

- Fx_N and Fy_N derived from Pacejka Model.
- The lateral force (in N) $F = D \cdot \sin(C \cdot \arctan(Bx1 - E \cdot (Bx1 - \arctan(Bx1)))) + V$

| Coefficient | Name |
|---|---|
| C | Shape factor |
| D | Peak Factor |
| BCD | Stiffness |
| B | Stiffness factor |
| E | Curvature factor |
| H | Horizontal shift |
| V | Vertical shift |
| Bx1 | (composite) |

Table 2 - List of the Pacejka tire model coefficients

Coefficient D contains the Fz (normal vertical force) component. It will be later shown that in order to commonize the model with PAIRSIM simulator, some changes were made in Codegen (or CGS) to have apples to apples to comparison.

- The above formula, which has been used in the code is documented in the link below:
  - Pacejka '94 parameters explained – a comprehensive guide – Edy's Projects[3]
- Note that it is assumed that the Pacejka Model is correctly coded, and the function is giving the correct values of Fx_N and Fy_N.

Furthermore, the CODEGEN simulator's vehicle dynamics are primarily driven by Single non-linear track model and they switch to a Kinematic model at speeds less than 3 mps. The following section will delve into the equations used and the blending process used to calculate the exact measurement values and the differential state values.

**Single Track Model(Non linear):**

The non-linear single track model is well described in Vehicle Dynamics: Modeling and Simulation by Schramm, D. et al. (2017) and most equations in the code can be derived from referring their work.

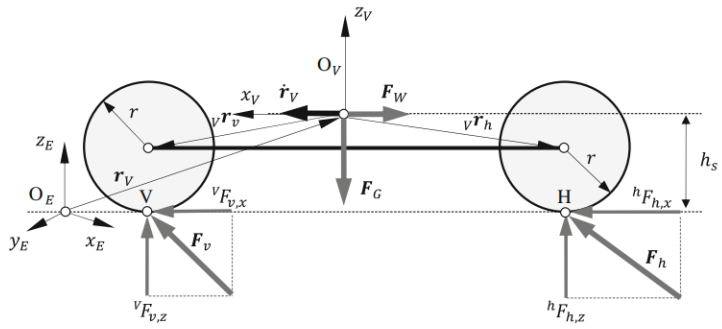



**Fig. 10.8** Nonlinear single track model—*side view*

Figures 2 a), and b) are only for reference. The nomenclature of the book is different from the code. (Refer chapter 10)

Source – [2] *Schramm, D. et al. (2017) Vehicle Dynamics: Modeling and Simulation* [Full Ebook Link here]

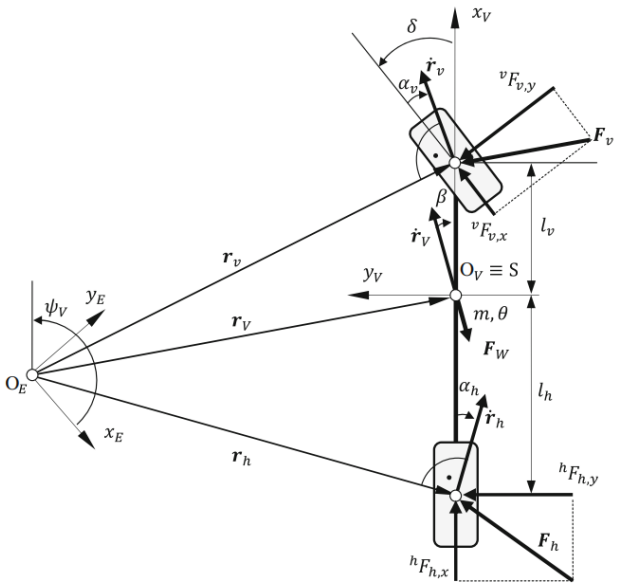



**Fig. 10.9** Nonlinear single track model–*top view*

The nonlinear single track model equations used in the code are given below:

$$F_{xF} = F_{x1} + F_{x2} \qquad F_{xR} = F_{x3} + F_{x4} \qquad F_{yF} = F_{y1} + F_{y2} \qquad F_{yR} = F_{y3} + F_{y4}$$

Calculation of accelerations:

- Longitudinal acceleration:

$$a_{x_{\text{stm}}} = \frac{F_{xF}\cos(\delta_{\text{wheel}}) - F_{yF}\sin(\delta_{\text{wheel}}) + F_{xR} - F_{x_{\text{friction}}} + F_{\text{ext},x}}{m}$$

- Lateral acceleration:

$$a_{y_{\text{stm}}} = \frac{F_{yF}\cos(\delta_{\text{wheel}}) + F_{xF}\sin(\delta_{\text{wheel}}) + F_{yR} + F_{\text{ext},y}}{m}$$

- Longitudinal and lateral velocity changes:

$$\dot{v}_{y_{\text{stm}}} = a_{y_{\text{stm}}} - \dot{\Psi} v_x \quad \dot{v}_{x_{\text{stm}}} = a_{x_{\text{stm}}} + \dot{\Psi} v_y$$

- Yaw acceleration:

$$\ddot{\Psi}_{\text{stm}} = \frac{(F_{yF}\cos(\delta_{\text{wheel}}) + F_{xF}\sin(\delta_{\text{wheel}}))l_{\text{front}} - F_{yR}l_{\text{rear}} + \tau_{\text{ext}}}{J}$$

- Wheel rotational acceleration:

$$\dot{\omega}_{\text{stm}} = (F_{x_{\text{PT}}} - F_x) \cdot \text{wheelInertiaFactors}$$

**Kinematic Model Equations:**

- Longitudinal dynamics:

$$a_{x_{\text{km}}} = \frac{\sum F_{x_{\text{PT}}} - F_{x_{\text{friction}}}}{m} \qquad \dot{v}_{x_{\text{km}}} = a_{x_{\text{km}}}$$

- Lateral dynamics:

  Assuming the center of gravity is at 50/50: $\beta = \frac{\delta_{\text{wheel}}}{2}$

  With a first-order low-pass dynamics:

$$\dot{v}_{y_{\text{km}}} = \frac{1}{T_{vy}}(\beta v_x - v_y)$$

- Yaw rate dynamics:

  Neutral steer yaw rate:

$$\dot{\Psi}_{\text{neutral}} = \frac{\delta_{\text{wheel}} v_x}{l_{\text{front}} + l_{\text{rear}}}$$

  With first-order low-pass dynamics:

$$\ddot{\Psi}_{\text{km}} = \frac{1}{T_{\dot{\Psi}}}(\dot{\Psi}_{\text{neutral}} - \dot{\Psi})$$

- Slip updates:

  Calculate target slip angles and ratios: ( Done by another function – calcWheelSlips)

```
[lambda_T_perc, alpha_T_rad] = calcWheelSlips(omega_rad, [vx_mps; vy_mps; dPsi_rad],DeltaWheel_rad, tw_front_m, tw_rear_m, l_front_m, l_rear_m, tyreradius_front_m, tyreradius_rear_m, vx_min);
```

  Update slip angles and ratios with first-order dynamics:

$$\dot{\alpha}_{\text{stm}} = \frac{1}{T_{\text{tire}}}(\alpha_T - \alpha) \quad \dot{\lambda}_{\text{stm}} = \frac{1}{T_{\text{tire}}}(\lambda_T - \lambda)$$

  For the kinematic model (assuming zero target slips):

$$\dot{\alpha}_{\text{km}} = \frac{1}{T_{\text{tire}}}(0 - \alpha) \quad \dot{\lambda}_{\text{km}} = \frac{1}{T_{\text{tire}}}(0 - \lambda)$$

The kinematic model's goal is to bring the slip ratio λ to zero, indicating a non-slipping condition. The term $\frac{1}{T_{\text{tire}}}(0 - \lambda)$ means that the rate of change of λ is proportional to the current value of λ, scaled by the relaxation time constant. This ensures that over time, λ will approach zero, achieving a steady-state condition where the tire rolls without slipping. Hence, this make the model ideal for low-speeds.

- Wheelspeed target speed to free rolling & Update wheel rotational speeds:

$$\omega_{\text{rolling}} = \frac{[v_x - \dot{\Psi} \cdot \text{tw}_{\text{front}} \cdot 0.5, v_x + \dot{\Psi} \cdot \text{tw}_{\text{front}} \cdot 0.5, v_x - \dot{\Psi} \cdot \text{tw}_{\text{rear}} \cdot 0.5, v_x + \dot{\Psi} \cdot \text{tw}_{\text{rear}} \cdot 0.5]}{[\text{tyreRadius}_{\text{front}}, \text{tyreRadius}_{\text{front}}, \text{tyreRadius}_{\text{rear}}, \text{tyreRadius}_{\text{rear}}]}$$

$$\dot{\omega}_{\text{km}} = \frac{1}{T_{\text{tire}}}(\omega_{\text{rolling}} - \omega)$$

- Alternative lateral acceleration for small velocities:

$$a_{y_{\text{km}}} = \dot{\Psi} v_x$$

**Blending Kinematic and single track models:**

- Calculate weighting factors:

$$w_{\text{stm}} = 0.5 \left( \tanh \left( \frac{v_x - v_{x_b}}{v_{x_d}} \right) + 1 \right) \qquad w_{\text{km}} = 1 - w_{\text{stm}}$$

(Note- $w_{stm}$ when greater than 0.5, implies STM dominance, at speeds over 3 mps)

- Blend Process: The below equations describe the dynamics of a vehicle model, incorporating both single track and kinematic aspects, blended based on vehicle velocity to provide more accurate simulation results across different driving conditions.

$$\dot{v}_x = \dot{v}_{x_{\text{km}}} \cdot w_{\text{km}} + \dot{v}_{x_{\text{stm}}} \cdot w_{\text{stm}} \qquad \dot{\alpha} = \dot{\alpha}_{\text{km}} \cdot w_{\text{km}} + \dot{\alpha}_{\text{stm}} \cdot w_{\text{stm}}$$

$$\dot{v}_y = \dot{v}_{y_{\text{km}}} \cdot w_{\text{km}} + \dot{v}_{y_{\text{stm}}} \cdot w_{\text{stm}} \qquad \dot{\lambda} = \dot{\lambda}_{\text{km}} \cdot w_{\text{km}} + \dot{\lambda}_{\text{stm}} \cdot w_{\text{stm}}$$

$$\ddot{\Psi} = \ddot{\Psi}_{\text{km}} \cdot w_{\text{km}} + \ddot{\Psi}_{\text{stm}} \cdot w_{\text{stm}} \qquad a_x = a_{x_{\text{km}}} \cdot w_{\text{km}} + a_{x_{\text{stm}}} \cdot w_{\text{stm}}$$

$$\dot{\omega} = \dot{\omega}_{\text{km}} \cdot w_{\text{km}} + \dot{\omega}_{\text{stm}} \cdot w_{\text{stm}} \qquad a_y = a_{y_{\text{km}}} \cdot w_{\text{km}} + a_{y_{\text{stm}}} \cdot w_{\text{stm}}$$

The blending is done to take advantage of the strengths of each model in different driving conditions. The blending factor ($w_{stm}$ and $w_{km}$) is determined based on the vehicle's longitudinal velocity (vx_mps).

- Nonlinear Single Track Model: This model is more accurate at higher speeds and during aggressive maneuvers where vehicle dynamics are nonlinear. It considers lateral and longitudinal tire forces, yaw moment, and other dynamics that are critical for high-speed stability and cornering.

- Kinematic Model: This model is simpler and more accurate at low speeds. It is often used for low-speed maneuvers or when the car is nearing standstill/running at low speeds.

- Advantages:
  - At low speeds, the kinematic model is computationally less intensive and sufficiently accurate, reducing the computational load.
  - At higher speeds, where more detailed dynamics are necessary, the single track model is used to capture the more complex behaviors accurately.
  - It ensures that the model remains reliable and accurate whether the vehicle is moving slowly or at high speeds.

Finally, the function Vehicle Model returns the exact measurements and the differential states:

- `DifferentialStates = [dvx;dvy;ddPsi;domega_rad;dlambda_perc;dalpha_rad;];`
- `ExactMeasurements = [vx_mps;vy_mps;dPsi_rad;ax;ay;omega_rad;];`

These values are then fed into subsequent functions which then become part of the overarching model.

## PAIR SIMULATOR (PAIRSIM)

PAIRSIM is developed in Python and is run on ROS. There are no extensive vehicle dynamics that can be extracted from the simulator, but it does have 2 scripts – CarController.cs and WheelController.cs that work in tandem with Unity's physics engine to carry out the Vehicle dynamics simulation on ROS.

The block diagram in Figure 3 gives an overview of the working of these scripts.

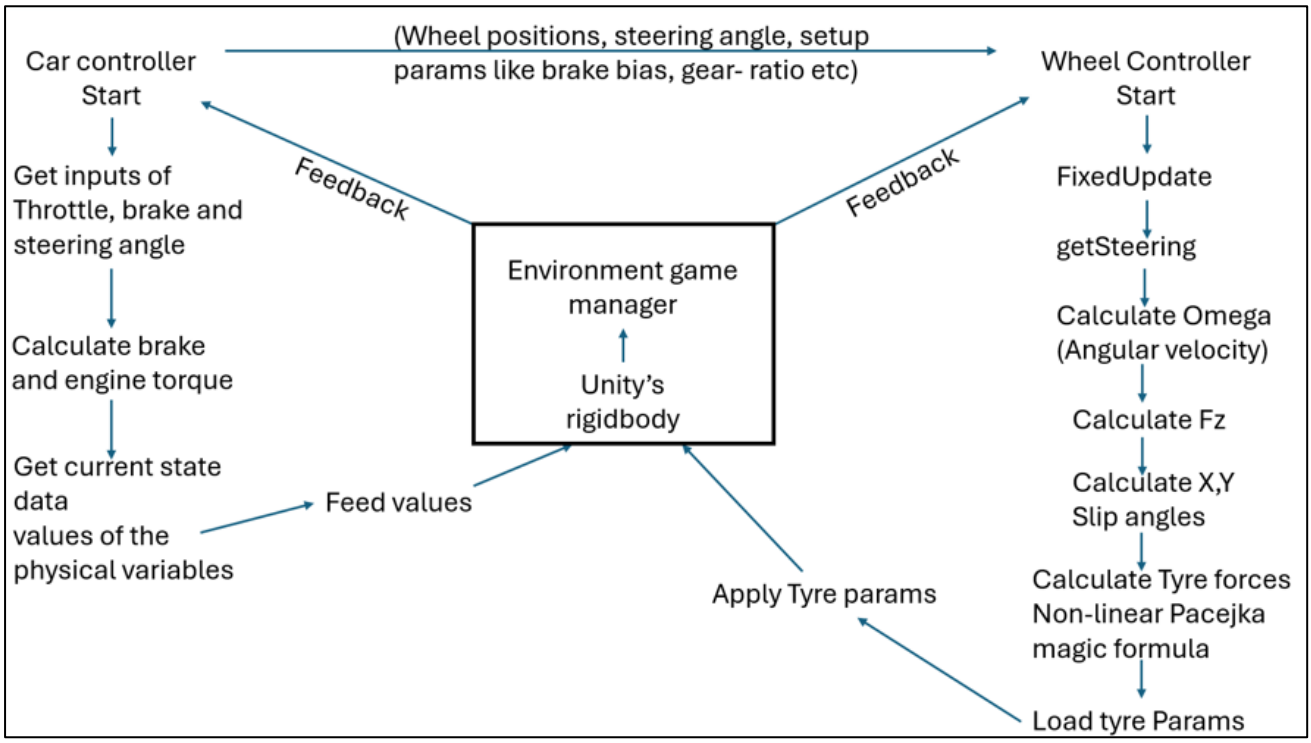



Figure 3 – PAIRSIM Overview

PAIRSIM provides a user interface, and also a 3D visualization of the car on any available/given track. The track has to be loaded into the program using ROS commands, along with the controller type, and after setting up the code environment properly, one may start running the simulation.

PAIRSIM shares a lot of commonalities with the actual car code environment, for example – having several topics being the same helps in quickly correlating on-track test data with the simulation results.

The main motive of the paper is to point out the differences that these simulations have and how can they be made to point towards a common expected result. The upcoming sections of the paper will report the differences and the changes made to both of these simulators in order to get some good results, using the same input parameters on a circular skid-pad with no banking and with constant steer input.

## DIFFERENCES BETWEEN CGS AND PAIRSIM

| | **Original Attributes before any changes in either models, i.e 1st run parameters** | |
|---|---|---|
| **Parameter** | **CGS** | **PAIRSIM** |
| Rolling effect | No | Yes |
| ARB(antiroll bar) | No | Front(Rear detachable) |
| Banking | No | Yes* |
| Spring stiffness | No | Yes |
| Weight | **787** | 790 |
| Tire Temperature effects | No | **Yes** |
| Tire model Param PacLat E (represents curvature factor) | **1** | 0 |
| Vehicle Dynamics Model | Single track model (Bicycle) | Unknown (Run by Unity Physics engine) |
| Tire model | Pacejka | Pacejka |
| Aero Effects | Yes | Yes |
| Cd= | **0.85** | 0.8581 |
| Lift coefficient front (cWf) | **1.8** | 0.65 |
| Lift coefficient rear (cWr) | **2.1** | 1.18 |
| Rear differential setting | N/A | **Spool** |

Table 3 – CGS v/s PAIRSIM before modifications ( * represents not utilized for following tests, and bolded parameters were updated for 2nd run)

| | **Attributes after changes in both models** | |
|---|---|---|
| **Parameter** | **CGS** | **PAIRSIM** |
| Rolling effect | No | Yes |
| ARB(antiroll bar) | No | Front(Rear detachable) |
| Banking | No | Yes* |
| Spring stiffness | No | Yes |
| Weight | **790** | 790 |
| Tire Temperature effects | No | **No** |
| Tire model Param PacLat E (represents curvature factor) | **0** | 0 |
| Vehicle Dynamics Model | Single track model (Bicycle) | Unknown (Run by Unity Physics engine) |
| Tire model | Pacejka | Pacejka |
| Aero Effects | Yes | Yes |
| Cd= | **0.8581** | 0.8581 |
| Lift coefficient front (cWf) | **0.65** | 0.65 |
| Lift coefficient rear (cWr) | **1.18** | 1.18 |
| Rear differential setting | N/A | **Limited Slip Differential** |

Table 4 – CGS v/s PAIRSIM after modifications ( * represents not utilized for following tests, and bolded parameters were updated for 2nd run)

Now, the following constant speed tests were conducted on both CGS and PAIRSIM, and the results before and after the parameter changes are published. The new parameters in both simulations were updated based on actual car data that came to light with further research.

**Understeer Coefficient variation with speed**

- Varied the vehicle speed from 15 mps to 70 mps
- A constant steering input of 1° was given on the front tires
- The understeer value was calculated using yaw rate gain and speed ( note $l_2$ = -1.6875m)

$$K_2 = \frac{\delta_1}{ru} - \frac{l_2}{u^2}$$

- If the $K_2$ values are positive, then it indicates understeer, else vice-versa. [2]

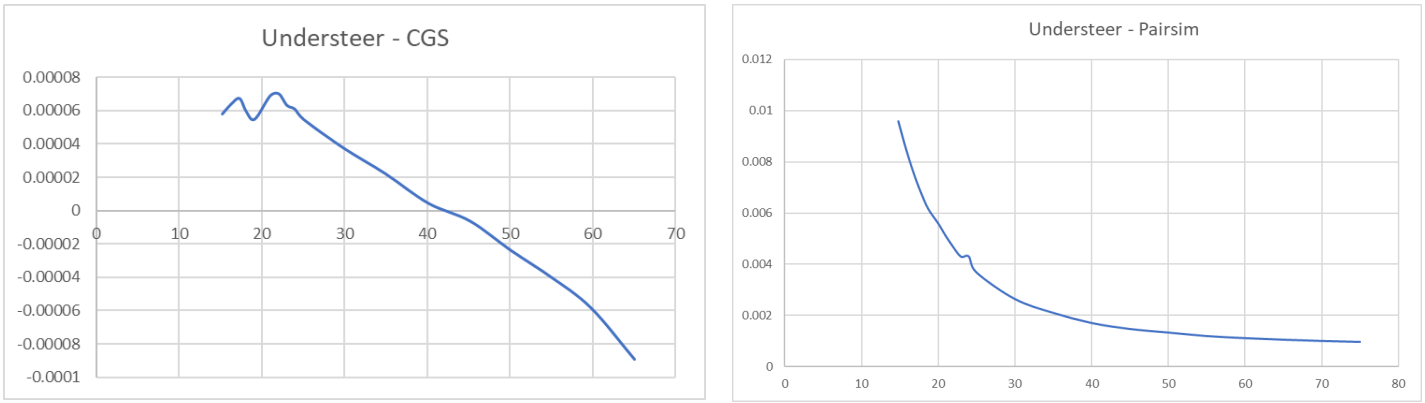



Figure 4 – CGS v/s PAIRSIM Understeer (before)

CGS shows really small positive values at low speeds, but goes negative at higher speeds, indicating understeery(but closer to neutral steering) behaviour at low speeds and Oversteery behaviour at higher speeds.

PAIRSIM shows understeery behaviour throughout the test speed regime.

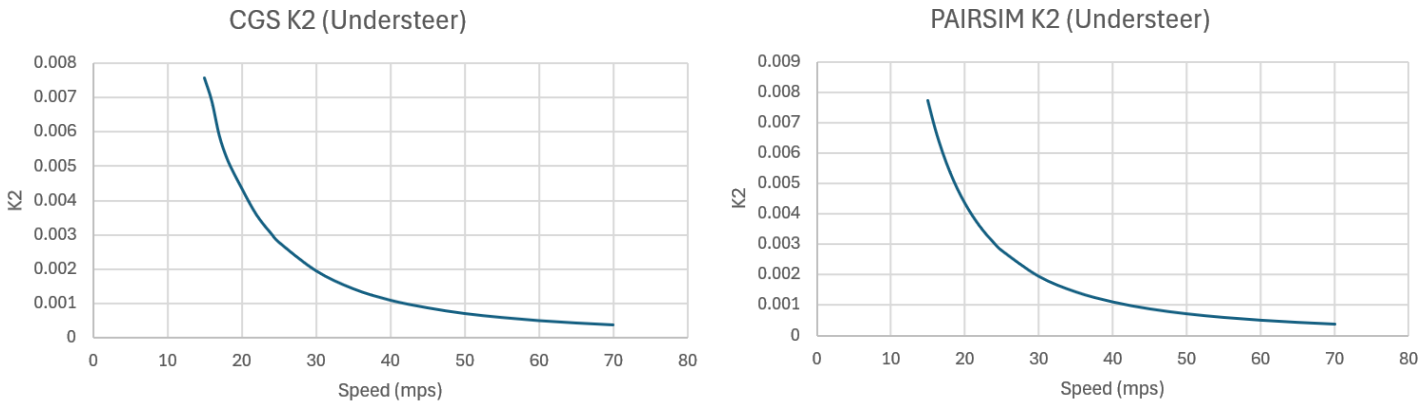



Figure 5 - CGS v/s PAIRSIM Understeer (After)

Both CGS and PAIRSIM are now on the same page – understeery behaviour at both low and high speeds. Although, the magnitude of understeering is slightly different, updating the parameters in both models certainly improved the coherency between the two simulator models.

**Yaw rate gain for CGS(old/new) & PAIRSIM(old/new)**

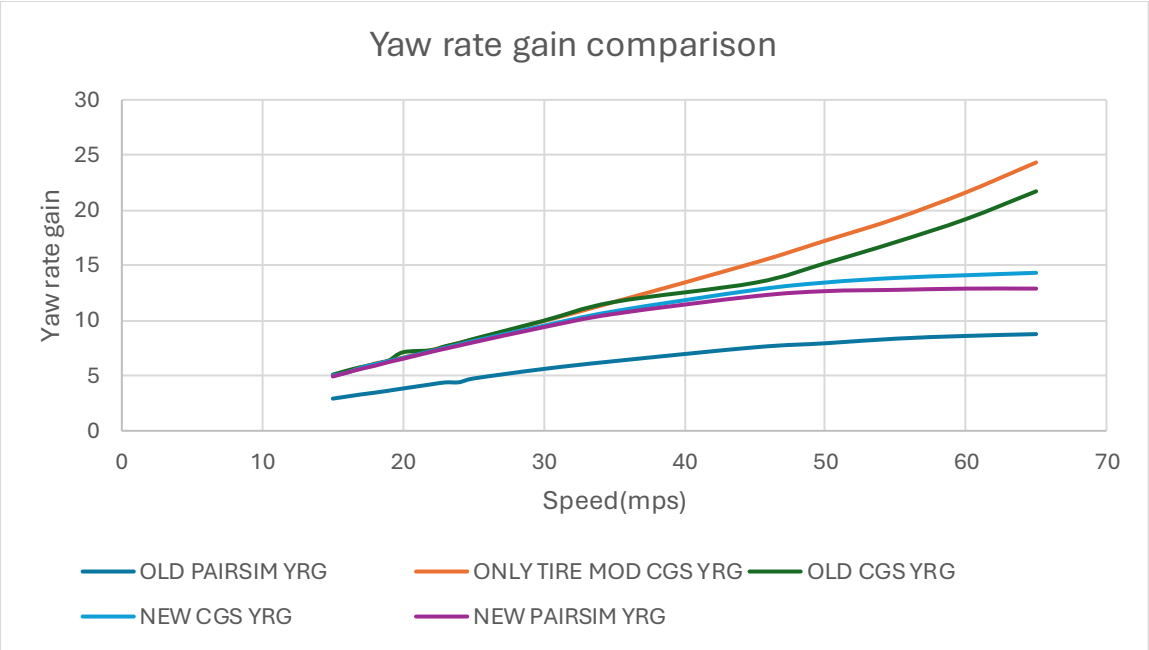



Figure 6 - CGS v/s PAIRSIM YRG (Before and After)

To track the effects of changing the parameters on the response of the simulators, a systematic series of changes was done - I then II then III ( Figure 7). To clarify further,

- OLD PAIRSIM YRG: Represents “Before” changing parameters or Old PAIRSIM
- OLD CGS YRG: Represents “Before” changing parameters or Old CGS
- ONLY TIRE MOD CGS YRG: Represents only Tire modifications in Old CGS model
- NEW CGS YRG: represents Tire + Aero modifications in the Old CGS model, making it NEW CGS thereafter

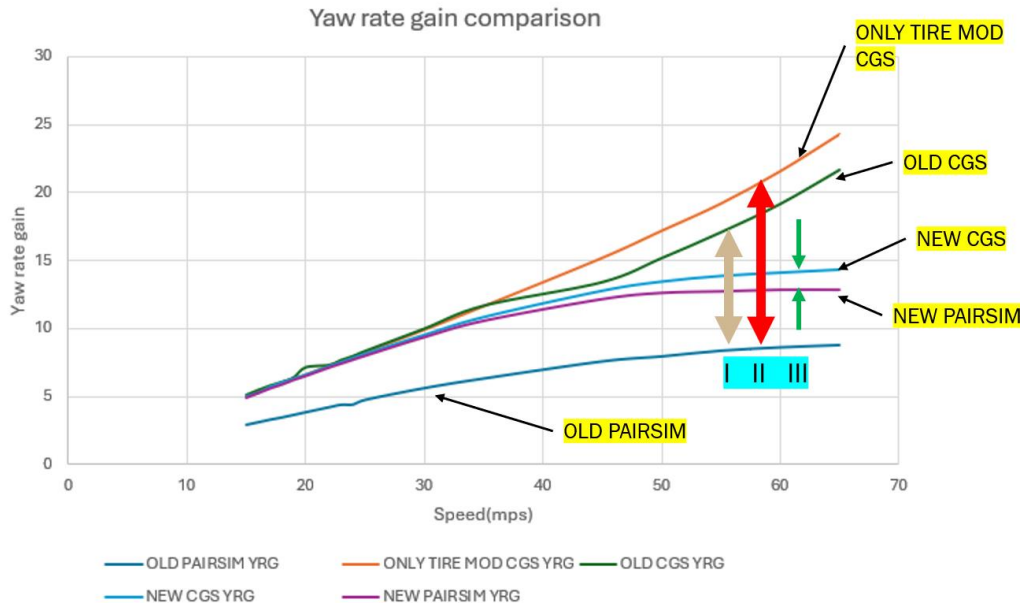



Figure 7- I II III Changes highlighted

**Comparison of Front and rear slip angles**

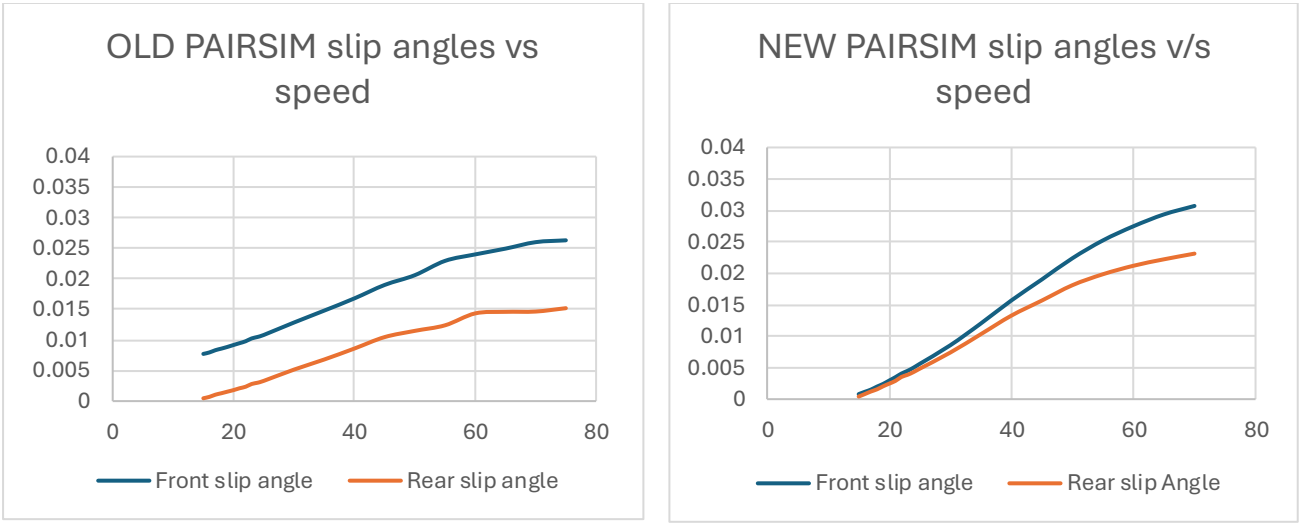



Figure 8 – PAIRSIM Front/Rear attack angles (Before v/s After)

It can be observed that the front slip angle has reduced significantly at lower speeds, indicating a more neutral steer though remaining understeery. This change is explained by the changing of the rear differential setting to Limited Slip Differential(LSD) from Spool in PAIRSIM.

When the setting was spool, the rear wheels were rotating together at the same rpm even during turns, simulating a stiffer rear axle. With the updated LSD setting, we can observe a significant change in the front slip angle while running on constant steer on a flat skid pad. The LSD softens the rear axle in PAIRSIM in a way that brings it closer to the behavior of NEW CGS with updated Aero and tire model parameters. (Figure 9) . Since, before the changes in CGS, Rear slip angle> Front slip angle at speeds above 45 mps, it was proof that CGS is oversteery at higher speeds, but after modifications, even CGS became understeery at higher speeds just like PAIRSIM.

The overall take-away would be that this is sufficient proof that both models have been able to generate similar results of crucial Vehicle dynamics parameters when subjected to exact same conditions, and which closely match with actual field parameters of the race car.

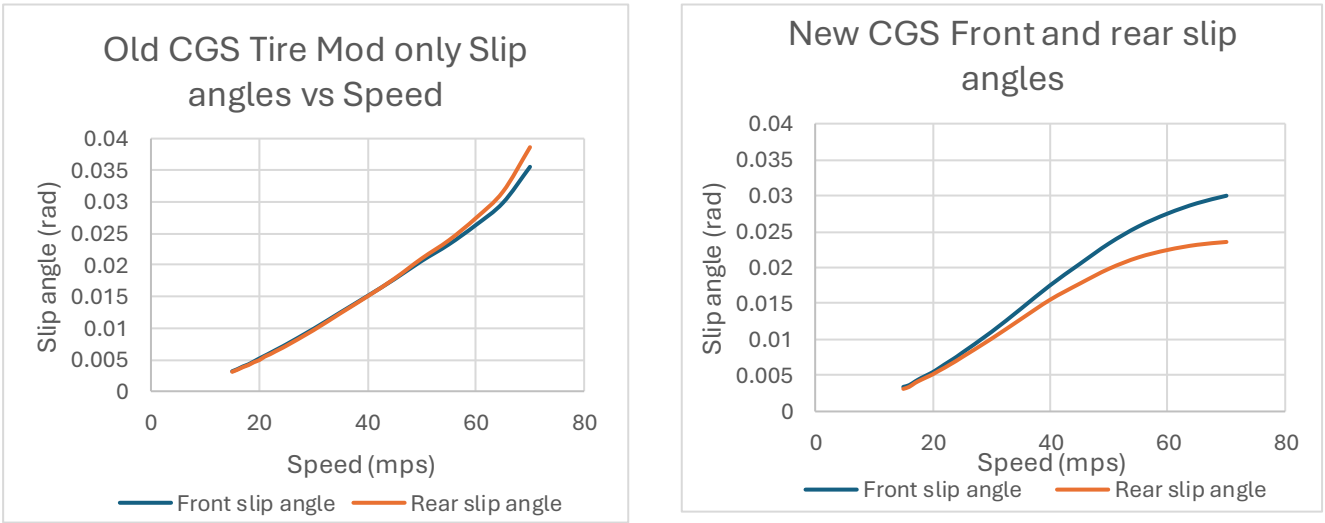



Figure 9- CGS Front/Rear slip angles(Before v/s After)

**Comparison of Front and rear slip angles across simulators**

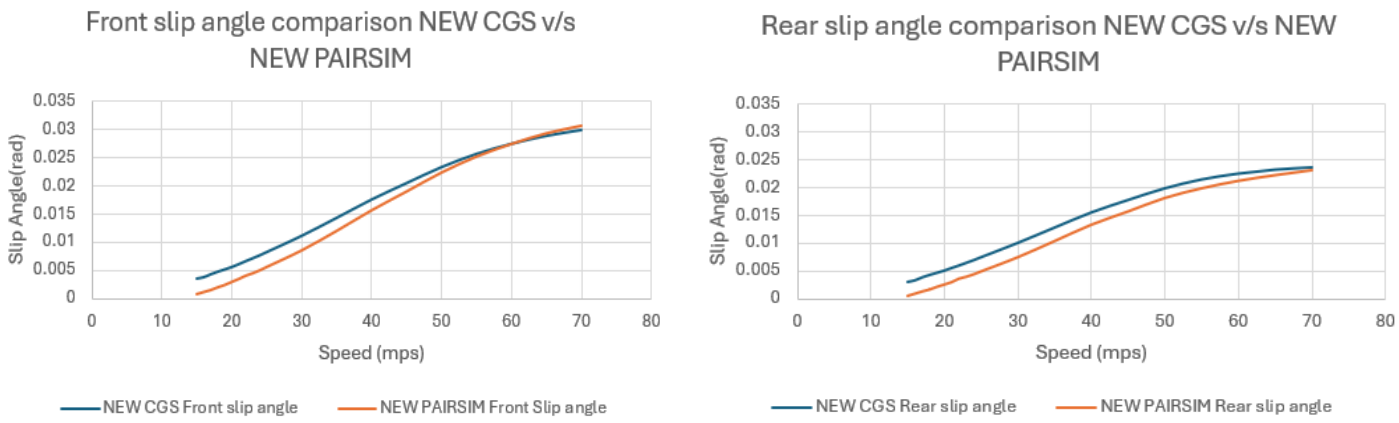



Figure 10 - Comparison of Front and rear slip angles across simulators (after)

Comparing NEW CGS and NEW PAIRSIM apples to apples, we can clearly see that the new values of the slip angles of both simulators are closely knit together, and approach approximately the same slip values even at higher speeds.

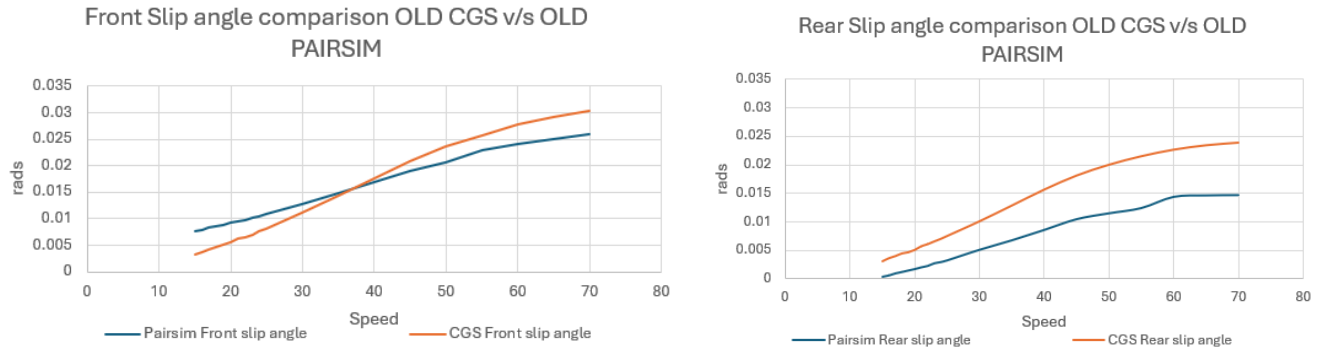



Figure 11 - Comparison of Front and rear slip angles across simulators (before)

The Old model parameters in both the simulations did not yield a good overlap between the front and front CGS/PAIRSIM, and rear and rear CGS/PAIRSIM, and the significant factor to cause this change in the new run is changing the rear differential setting to LSD.

**Cornering stiffness comparison (after all changes, i.e NEW)**


Figure 12 – Cornering Stiffness comparison (NEW)

**Cornering stiffness comparison (before PAIRSIM changes, i.e OLD )**

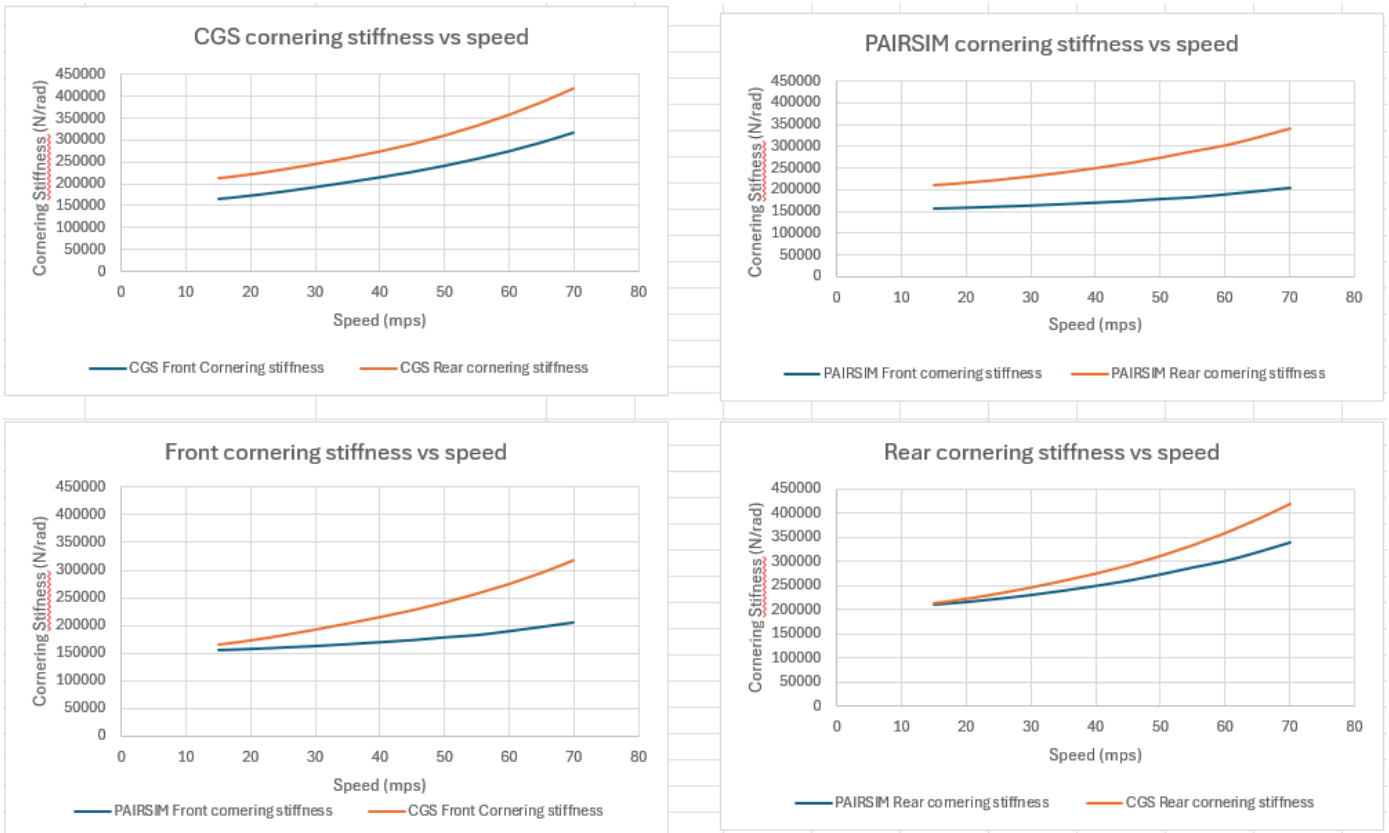



Figure 13 - Cornering stiffness comparison(OLD)

Referring to the results of cornering stiffness calculations on the last page, it was observed that the Newly derived values of cornering stiffnesses in both simulators are almost the same at every speed.

To calculate all the **new** values of cornering stiffnesses, the lateral tire forces of each of the 4 tires were derived directly from the simulation environment and then were divided by slip angle to give the cornering stiffness values. Since these values were directly extracted from the simulation environment and PAIRSIM had undergone a major change in LSD setting, a good correlation between the parameters can be observed across the 2 simulators.

This is further proof that the NEW simulators are yielding similar results, unlike the OLD setup runs which had uncommon parameters and inconsistent results causing ambiguity in interpretation.

Comparing the rear-rear of CGS/PAIRSIM and front-front of CGS/PAIRSIM in the OLD setup, it was observed that the stiffnesses start to diverge at higher speeds. But with the updated settings and commonised parameters, both PAIRSIM and CGS are yielding closely knit results bolstering the confidence in PAIRSIM, because PAIRSIM is the simulator whose Vehicle Dynamics model is not clearly understood since it runs on Unity's Physics engine.

## Conclusion

This comprehensive study of the CGS and PAIRSIM simulators has revealed critical insights into the modeling and simulation of autonomous race cars. Initially, significant discrepancies were observed between the two simulators in their handling of key vehicle dynamics aspects, including understeer characteristics, yaw rate gain, and slip angles.

Through systematic analysis and targeted modifications, the two simulators were brought into much closer alignment. Key changes included:

1. Updating the tire model parameters, particularly the curvature factor (PacLat E) in CGS.
2. Adjusting aerodynamic lift coefficients in CGS simulator to match PAIRSIM's
3. Changing the rear differential setting in PAIRSIM from Spool to a Limited Slip Differential (LSD), which significantly improved its low-speed behavior.

These modifications resulted in more consistent and realistic behavior across both simulators, as evidenced by:

- Convergence of understeer coefficients, with both simulators now showing understeery behavior across the entire speed range.
- Closer alignment of yaw rate gain curves.
- More consistent front and rear slip angles between the simulators.
- Nearly identical cornering stiffness values across the speed range.

The improved correlation between CGS and PAIRSIM not only enhances our confidence in these simulation tools but also provides a more reliable platform for developing and testing autonomous racing algorithms. The study underscores the importance of accurate modeling of vehicle dynamics, particularly in areas such as tire behavior, aerodynamics, and drivetrain characteristics.

Moreover, this research highlights the value of comparative analysis between different simulation environments. By identifying and reconciling discrepancies, we can improve the overall fidelity of our simulation tools, leading to more robust and reliable autonomous racing systems.

## Future Scope

While this study has made significant strides in aligning and improving the CGS and PAIRSIM simulators, several areas warrant further investigation:

1. System Identification Planning:
    - Based on the identified key parameters, develop a comprehensive plan for safe vehicle dynamics tests that can be conducted on an oval track.
    - Design and implement software modifications to perform these tests in both simulators.
    - Analyze the effectiveness of these tests in the simulation environment and compare them with real-world data.
2. Real-world Validation:
    - Conduct physical tests with an instrumented race car to collect data for validating and further refining the simulation models.
    - Develop methodologies for continuous improvement of the simulators based on real-world feedback.
3. Advanced Tire Modeling:
    - Implement more sophisticated tire models that account for temperature effects, wear, and dynamic load variations.
    - Investigate the impact of these advanced models on the accuracy of long-run simulations.

By pursuing these areas of future research, we can continue to improve the fidelity and usefulness of our simulation tools, ultimately leading to safer and more capable autonomous racing systems.

## Acknowledgments

I'm deeply grateful to Professor Daniel Williams for his invaluable guidance throughout this project. Special thanks to Harshvardhan Patil for his crucial assistance with ROS and PAIRSIM. Professor Shaver's inputs were essential for accurate data analysis. Finally, I sincerely appreciate the Purdue AI Racing team for their continuous support and patience with my queries. This project was made possible by their collective expertise and encouragement.

## References:

1) Williams, Daniel. (2022). Generalized Vehicle Dynamics. 10.4271/9781468601411.

2) Schramm, D. et al. (2017) Vehicle Dynamics: Modeling and Simulation Full Ebook Link here

3) Pacejka ’94 parameters explained – a comprehensive guide – Edy's Projects