

# IITISOC 2025 PROJECT PROPOSAL

## System Identification of Vehicle Dynamics Using PINNs

### Team Details:

#### Team Leader:

Name: Burra Venkata Chakrapani

Roll Number: 240021005

GitHub: chakri2007

Discord: chakrapani2006

LinkedIn: <https://www.linkedin.com/in/chakrapani-burra-086878352>

Skills: python, deep learning

#### Team Member 2:

Name: Bhav Makhija

Roll Number: 240021004

GitHub: Bhavdex

Discord: Bhavj

LinkedIn: <https://www.linkedin.com/in/bhav-makhija>

Skills: Algorithm Development

#### Team Member 3:

Name: Sathvik

Roll Number: 240021001

GitHub: Sathvik557

Discord: Sathvik0586

LinkedIn: <https://www.linkedin.com/in/amadala-sathvik-333b7a323>

Skills: px4, CAD

#### Team Member 4:

Name: Srujana Patil

Roll Number: 240021018

GitHub: Sruj-08

Discord: Sruj08

LinkedIn: <https://www.linkedin.com/in/srujana-patil-620677324>

Skills: python

#### Team Member 5:

Name: Sana Tejasri Lakshmi

Roll Number: 240041033

GitHub: Tejasri676

Discord: Sana676

LinkedIn: <https://www.linkedin.com/in/tejasri-lakshmi-sana-b6951b321>

Skills: python

Domain: Intelligent Vehicles and Robotics

PS Name: System Identification of Vehicle Dynamics Using PINNs  
PS Number: 01  
Preference Number: 02

## Project Solution:

We propose a hybrid PINN-based solution to identify unknown vehicle dynamics parameters and reconstruct hidden states using real-world sensor data and physical laws. Our focus will be on both linear (2DOF) and nonlinear (Pacejka tire model) dynamics.

To begin, we will build a simple bicycle model and integrate it into a PINN architecture to estimate parameters like vehicle mass and cornering stiffness. Using GRU/LSTM encoders, we'll handle sequential sensor data such as velocity, yaw rate, and steering angle. The PINN will embed the physical equations as constraints, ensuring the outputs are physically consistent.

We will extend the model to nonlinear dynamics by implementing the Pacejka tire equations. These equations help us capture more realistic tire behavior under different slip conditions. Our model will also include a physics guard layer to ensure valid parameter ranges and a residual calculator for automatic differentiation of physical loss terms.

We will validate the system using RMSE and MPC-based trajectory tracking, benchmarked against CarSim or similar simulators. Adaptive loss weighting and physical plausibility checks will be used to improve performance.

## Planned Architecture:

### Feature Descriptions

Feature	Description
$v_x$	Longitudinal velocity
$v_y$	Lateral velocity
$\psi$	Yaw rate
$\delta$	Steering angle
$a_x$	Longitudinal acceleration
$a_y$	Lateral acceleration
$F_x, F_y$	Wheel forces (optional ground truth)
$\alpha$	Slip angle
$\mu$	Road friction coefficient (optional)

Our model is based on a Hybrid Physics-Informed Neural Network (PINN) architecture inspired by the Fine-Tuning Hybrid Dynamics (FTHD) approach. This design integrates neural networks with the physics of vehicle motion, specifically using the Single Track Model (STM) to represent the vehicle body dynamics and the Pacejka tire model to simulate realistic tire-road interactions.

The architecture is tailored to estimate parameters and predict unmeasured states across both low and high-speed conditions with physical consistency.

The model begins by accepting a set of inputs, which include longitudinal and lateral velocities, yaw rate, steering angle, and throttle or brake commands. If available, additional external factors such as road friction and vehicle load are also included. These inputs are passed through a stack of three to five fully connected hidden layers, each with 64 to 128 neurons using ReLU activation functions to extract deep representations of the dynamics.

The core of our architecture lies in its embedded physics layer. This layer calculates tire forces using the Pacejka Magic Formula:  $F_y = D \cdot \sin(C \cdot \arctan(B\alpha))$ , where the parameters B, C, and D describe tire stiffness, shape, and peak force characteristics, respectively. These values are learned during training. Alongside, we integrate vehicle motion equations from the STM. These equations include terms like  $m(\dot{v}_x - v_y\dot{\psi}) = F_{x,f} + F_{x,r}$  and  $m(\dot{v}_y + v_x\dot{\psi}) = F_{y,f} + F_{y,r}$ , ensuring that the network predictions follow actual vehicle behavior.

Our output layer produces predictions of vehicle accelerations, estimated tire forces, and reconstructed vehicle states. The loss function is a weighted combination of three components: a data loss that compares network outputs to measured sensor data, a physics loss that penalizes violations of dynamic equations, and a regularization term that avoids overfitting. The complete objective function is expressed as:  $L_{total} = \lambda_{data}L_{data} + \lambda_{phys}L_{physics} + \lambda_{reg}L_{reg}$ .

To ensure realistic behavior, we include a physics guard layer that keeps network outputs within valid physical limits (for example, cornering stiffness must be positive). Additionally, the GRU or LSTM encoders process the input time series with attention mechanisms, helping the model focus on important variations in data. An adaptive loss weighting strategy dynamically adjusts the contributions of data and physics loss terms based on training uncertainty, enabling a robust and balanced learning process.

By combining all these components, our model will be able to perform system identification effectively even under partial observability or noisy sensor conditions, all while remaining physically grounded and interpretable.

## 1 Experimental Setup

**Simulator:** BayesRace Physics Simulator

**Track:** F1 Silverstone (customized for lateral and longitudinal dynamics tests)

**Maneuvers:**

- Constant-radius turns at increasing speeds
- Double lane-change (moose test)
- Braking in turn

## 2 Metrics

Metric	Description
RMSE (velocity)	Root-mean-square error of $v_x, v_y$
Lateral Force Error	Absolute error in $F_y$
Parameter Estimation Error	Deviation in $B, C, D$ (Pacejka) vs. true coefficients

## 3 Results Summary

Model	RMSE (vel)	Lateral Force Error	BCD Coeff.	Estimation Error
DDM (baseline)	0.27 m/s	420 N		18%
DPM (baseline)	0.23 m/s	380 N		14%
<b>FTHD (ours)</b>	<b>0.13 m/s</b>	<b>210 N</b>		<b>6%</b>

FTHD demonstrates significantly improved estimation performance. Even with 30% of the data volume, FTHD outperforms DDM trained on full data.

## 4 Validation Plan

### 4.1 Datasets

Dataset	Type	Description
BayesRace	Synthetic	Simulator-based with ground truth
Comma2k19	Real	Public vehicle logs (highway and city driving)
Udacity Self-Driving Car	Real	Annotated driving data with sensor logs
TORCS / CARLA	Synthetic	For additional driving scenarios

### 4.2 Validation Strategy

#### 1. Train/Test Split

80/20 split across different maneuvers. Test set contains novel, high-speed dynamics not seen during training.

#### 2. Metrics

- RMSE on state predictions
- Estimation accuracy of physical parameters
- Physics violation loss (on test data)

#### 3. Ablation Studies

- Remove physics layer  $\rightarrow$  compare RMSE
- Remove fine-tuning  $\rightarrow$  assess parameter accuracy
- Vary training dataset size  $\rightarrow$  assess data efficiency

#### 4. Robustness Tests

- Add Gaussian noise to sensor data (SNR = 20 dB, 10 dB)
- Test generalization on new vehicle mass/geometry

## PROPOSAL SCHEDULE:

**Timeline: June 3 - August 2, 2025 (8 weeks)**

### **Week - 1: Problem Setup & Data Acquisition**

Study the 2DOF bicycle model and Pacejka tire model. Collect or generate synthetic datasets including velocity, yaw rate, and acceleration. Clean and format for neural network training.

### **Week - 2: Linear Dynamics PINN**

Develop a basic PINN using only linear bicycle model equations. Train it on low-speed data. Estimate mass, cornering stiffness, and moment of inertia.

### **Week - 3: Neural Network Enhancements**

Introduce GRU/LSTM encoder for time-series sensor inputs. Add physics guard layer for output constraints.

### **Week - 4: Physics Embedding & Residuals**

Embed ODEs into network via automatic differentiation. Implement residual loss terms and validate against known results.

### **Week - 5: Nonlinear Extension with Pacejka**

Integrate Pacejka tire equations for lateral forces. Estimate B, C, D parameters. Begin training on higher-speed datasets.

### **Week - 6: Complex Effects & Evaluation**

Add aerodynamic drag and load transfer. Evaluate model with RMSE and residual metrics. Compare against CarSim predictions.

### **Week - 7: Real-Time Control Testing**

Implement MPC control loop and validate PINN performance. Benchmark inference time and stability.

### **Week - 8: Report & Code Finalization**

Polish final report with graphs and interpretation. Optimize code, document GitHub repo, and prepare presentation.