Vehicle Dynamics Simulation Module: Documentation

Date: December 20, 2025 **Version:** 1.0

1. Overview

This module is a browser-based, 3D simulation designed to analyze the vertical dynamic response of a 4-wheeled, suspended vehicle. It allows users to visualize and quantify how a vehicle's suspension system reacts to various road irregularities (speed bumps, potholes, and rough terrain) in real time.

The simulation combines a high-fidelity physics engine (RK4 integrator) with 3D visualization (Three.js) and real-time data plotting (Chart.js) to provide immediate engineering feedback.

2. User Guide

2.1 Vehicle Geometry & Configuration

The Control Panel (Left Sidebar) allows users to define the physical properties of the vehicle. Updates occur in real time.

- **Chassis (Box 1):** Defines the main body dimensions and weight.
- **Cab/Cargo (Box 2):** Defines a secondary mass (e.g., truck cab or payload). The module automatically recalculates the Center of Gravity (CG) and Mass Moment of Inertia (I_{xx}, I_{yy})based on the combined masses.

Suspension:

- **Stiffness** (k)**:** Spring rate (lb/in). Higher values → stiffer ride.
- **Damping** (c)**:** Shock absorber resistance (lb·s/in). Controls oscillation decay.

Tires: Modeled as vertical stiff springs. Users can define tire stiffness $(k_t)$and diameter.

2.2 Simulation Controls

- **Solve:** Starts the physics integration loop. The vehicle begins moving forward at the specified velocity.
- **Stop:** Pauses the physics engine but keeps the 3D scene active.
- **Reset / Center Vehicle:** Returns the vehicle to the starting position and resets time to zero.
- **Speed:** Defines the constant forward velocity (V_x)in inches/second.

2.3 Road Features

Users can construct a custom test track by adding features at specific distances:

- **Full Speed Bump:** Sine-wave bump spanning full road width.
- **1/2 Bump (Left/Right):** Asymmetrical bump affecting one side; useful for roll dynamics (I_{xx}).
- **Rough Dirt Road:** Generated terrain using multi-frequency deterministic noise.

2.4 Data Analysis

A real-time graph at the bottom of the screen plots vertical acceleration (G-Force) for:

- **CG:** Center of Gravity
- **FL / FR / RL / RR:** Four chassis corners (suspension hardpoints)
- **User Point:** Custom location defined by coordinates (X, Y)

3. Solution Techniques & Physics Model

3.1 Coordinate System

The simulation uses a standard vehicle dynamics coordinate system (Z-Up):

- **X:** Lateral (Left/Right)
- **Y:** Longitudinal (Forward/Backward)
- **Z:** Vertical (Up/Down)
- **Origin:** Ground plane ($Z = 0$)

3.2 Dynamics Solver: Runge-Kutta 4 (RK4)

To ensure stability with stiff suspension springs, the module uses a 4th-Order Runge-Kutta (RK4) integration scheme. Unlike simple Euler integration, which can become unstable with high stiffness or damping, RK4 samples derivatives at four points within each timestep (dt)to accurately approximate the next state.

State Vector (14 variables):

$$S = [Z, \phi, \theta, \dot{Z}, \dot{\phi}, \dot{\theta}, Z_{w1..4}, \dot{Z}_{w1..4}]$$

Where:

- Z, ϕ, θ: Chassis heave, roll, pitch
- Z_w: Vertical position of each unsprung mass (wheel)

3.3 Suspension Model

The vehicle is modeled as a 7-DOF system (Chassis Heave, Pitch, Roll + 4 Wheel Vertical DOFs).

Suspension force at each corner:

$$F_{susp} = k_s \cdot (L_{free} - (Z_{body} - Z_{wheel})) + c_s \cdot (\dot{Z}_{wheel} - \dot{Z}_{body})$$

3.4 Tire Model

Tires use a point-follower model with linear vertical stiffness. Tire force is generated only when the tire penetrates the ground:

$$F_{tire} = \max(0, k_t \cdot (H_{road} - (Z_{wheel} - R)))$$

This allows wheels to leave the ground without generating non-physical forces.

3.5 Terrain Generation

The "Rough Dirt Road" uses deterministic spatial noise rather than random noise. Road height is computed as:

$$Z(x, y) = A \cdot [0.5\sin(f_1 y)\sin(f_1 x) + 0.3\sin(f_2 y + \delta) + 0.2\sin(f_3(x + y))]$$

This ensures consistent, repeatable roughness.

4. Technical Implementation

- **Language:** JavaScript (ES6 Modules)
- **Rendering:** Three.js (WebGL)
- **Charting:** Chart.js
- **Architecture:**
  - `buildVehicle`
  - `stepRK4` (physics)
  - `animate` (render) These loops decouple simulation frequency from frame rate.

5. References

- Milliken, W. F., & Milliken, D. L. (1995). *Race Car Vehicle Dynamics*. SAE International.
- Gillespie, T. D. (1992). *Fundamentals of Vehicle Dynamics*. SAE International.
- ISO 8855:2011. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary.*
- Press, W. H., et al. (2007). *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press.