**Unity Self driving car code!**

using System.Collections;  
using System.Collections.Generic;  
using UnityEngine;

public class CarEngine : MonoBehaviour

{  
public Transform path;

//to store the value of all the node which are predefined  
private List<Transform> nodes;

//check the current node of predefined path  
private int currentnode = 0;

//Max steer angle for the wheel  
public float maxSteerAngle=30f;

//Fronr Left and Right wheel  
public WheelCollider WheelFL;  
public WheelCollider WheelFR;

//Maximum torque for wheel  
public float maxMotorTorque = 30f;

//current speed of wheel  
public float currentSpeed;

//maximum speed of wheel  
public float maxSpeed=100f;

//defining center of mass for the wheel  
public Vector3 centerOfMass;  
private int stop;

// Use this for initialization

private void Start () {

//defining rigidbody for the vehichle and assigning the mass

GetComponent<Rigidbody>().centerOfMass = centerOfMass;

//getting all the nodes which are predefined

Transform[] pathTransforms = path.GetComponentsInChildren<Transform>();

nodes = new List<Transform>();

//to add all the predefined nodes into a list apart from the car’s point

for (int i = 0; i < pathTransforms.Length; i++)

{

if (pathTransforms[i] != path.transform)

{

nodes.Add(pathTransforms[i]);

}

}

stop = 0;

}

// Update is called once per frame

private void FixedUpdate () {

if (stop != 1)

{

ApplySteer();

Drive();

CheckWaypointDistance();

}

}

//Calculate SteerAngle for FL and FR wheel usimg Angle concept and Inverse Transformation

public void ApplySteer()

{

Vector3 relativeVector = transform.InverseTransformPoint(nodes[currentnode].position);

float newSteer = (relativeVector.x / relativeVector.magnitude)\*maxSteerAngle ;

WheelFL.steerAngle = newSteer;

WheelFR.steerAngle = newSteer;

}

//Moving the vehicle ahead by calculating and assigning the speed

public void Drive()

{

currentSpeed = 2 \* Mathf.PI \* WheelFL.radius \* WheelFL.rpm \* 60 / 1000;

if (currentSpeed < maxSpeed)

{

WheelFL.motorTorque = maxMotorTorque;

WheelFR.motorTorque = maxMotorTorque;

}

else

{

WheelFL.motorTorque = 0;

WheelFR.motorTorque = 0;

}

}

//moving the car based on the pre-defined node and if the node has reached last then freeze the position and give zero speed

public void CheckWaypointDistance()

{

if (Vector3.Distance(transform.position, nodes[currentnode].position)< 3f)

{

if (currentnode==nodes.Count-1)

{

WheelFL.motorTorque = 0;

WheelFR.motorTorque = 0;

WheelFL.brakeTorque = 50f;

WheelFR.brakeTorque = 50f;

GetComponent<Rigidbody>().constraints = RigidbodyConstraints.FreezeRotationX | RigidbodyConstraints.FreezeRotationY | RigidbodyConstraints.FreezeRotationZ | RigidbodyConstraints.FreezePositionX | RigidbodyConstraints.FreezePositionY | RigidbodyConstraints.FreezePositionZ;

stop =1;

}

else

{

currentnode++;

}

}

}

}