```
using UnityEngine.UIElements;
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Threading.Tasks;



public class RobotController : MonoBehaviour

{



    [SerializeField] private WheelCollider FLC;
    [SerializeField] private WheelCollider FRC;
    [SerializeField] private WheelCollider RLC;
    [SerializeField] private WheelCollider RRC;
    [SerializeField] private Transform FLT;
    [SerializeField] private Transform FRT;
    [SerializeField] private Transform RLT;
    [SerializeField] private Transform RRT;
    [SerializeField] private Transform FRS;
    [SerializeField] private Transform L1S;
    [SerializeField] private Transform L2S;
    [SerializeField] private Transform L3S;
    [SerializeField] private Transform R1S;
    [SerializeField] private Transform R2S;
    [SerializeField] private Transform R3S;
    [SerializeField] private Transform ORS;
    [SerializeField] private float maxturning = 30;
    [SerializeField] private float urgeforce = 50;

    private WheelCollider[] wheels;



    private void Start()
    {

        rb = GetComponent<Rigidbody>();


        SenLoc(FRS, 20, 0, 0);
        SenLoc(L1S, 0, -10, 0);
        SenLoc(L2S, 8, -30, 0);
        SenLoc(L3S, 16, -60, 0);
        SenLoc(R1S, 0, 10, 0);
        SenLoc(R2S, 8, 30, 0);
        SenLoc(R3S, 16, 60, 0);
        SenLoc(ORS, 50, 180, 0);


        wheels = GetComponentsInChildren<WheelCollider>();
    }
    private Rigidbody rb;
    [SerializeField] private float angle_x;
    [SerializeField] private float angle_z;
    [SerializeField] private float carpower;
    private float steerAngle;
    private bool isHalting;
    private float senor1place = 5;
    private float senor2place = 6;
    private float senor3place = 6;

    private void SenLoc(Transform senORS, float x_angle, float y_angle, float
z_angle)
    {
        senORS.transform.Rotate(x_angle, y_angle, z_angle);
    }



    private void PowerMotor()
    {
        float CurrentAcceleration;
        CurrentAcceleration = isHalting ? 0 : urgeforce;
        FRC.motorTorque = CurrentAcceleration;
        FLC.motorTorque = CurrentAcceleration;
        RLC.motorTorque = CurrentAcceleration;
        RRC.motorTorque = CurrentAcceleration;

    }



    private void WheelOfFortune(WheelCollider Collider, Transform trans)
    {
        Vector3 position;
        Quaternion rotation;
        Collider.GetWorldPose(out position, out rotation);
        trans.position = position;
        trans.rotation = rotation;
    }



    private void WheelyAwesome()
    {
        WheelOfFortune(FLC, FLT);
        WheelOfFortune(FRC, FRT);
        WheelOfFortune(RLC, RLT);
        WheelOfFortune(RRC, RRT);
    }



    private void SteerLikeYouMeanIt(float direction)
    {
        steerAngle = direction * maxturning;
        FLC.steerAngle = steerAngle;
        FRC.steerAngle = steerAngle;
    }



    private bool SpideySense(Transform senORS, float dist, string layer)
    {
        LayerMask mask = LayerMask.GetMask(layer);


        if (Physics.Raycast(senORS.position,
senORS.TransformDirection(Vector3.forward), dist, mask))
        {

            Debug.DrawRay(senORS.position,
senORS.TransformDirection(Vector3.forward) * dist, Color.red);
            return true;
        }
        else
        {

            Debug.DrawRay(senORS.position,
senORS.TransformDirection(Vector3.forward) * dist, Color.blue);
            return false;
        }
    }




    private void KeepOnTrack()
    {
        if (!SpideySense(L3S, senor3place, "Road") || !SpideySense(R3S,
senor3place, "Road"))
        {

            if (!SpideySense(L3S, senor3place, "Road"))
            {
                SteerLikeYouMeanIt(1);
            }
            if (!SpideySense(R3S, senor3place, "Road"))
            {
                SteerLikeYouMeanIt(-1);
            }
        }
        else
        {
            SteerLikeYouMeanIt(0);
        }
    }


    private void SetFriction(float forwardFriction, float sidewaysFriction)
    {
        // Create a new WheelFrictionCurve for forward friction
        WheelFrictionCurve forwardFrictionCurve = new WheelFrictionCurve();
        forwardFrictionCurve.extremumSlip = 1f;
        forwardFrictionCurve.extremumValue = forwardFriction;// Set the maximum
friction
```

```
        forwardFrictionCurve.asymptoteSlip = 2f;
        forwardFrictionCurve.asymptoteValue = forwardFriction * 0.75f;// Gradual
decrease in friction
        forwardFrictionCurve.stiffness = 1f;
        // Create a new WheelFrictionCurve for sideways friction
        WheelFrictionCurve sidewaysFrictionCurve = new WheelFrictionCurve();
        sidewaysFrictionCurve.extremumSlip = 1f;
        sidewaysFrictionCurve.extremumValue = sidewaysFriction;// Set the maximum
friction
        sidewaysFrictionCurve.asymptoteSlip = 2f;
        sidewaysFrictionCurve.asymptoteValue = sidewaysFriction * 0.75f;// Gradual
decrease in friction
        sidewaysFrictionCurve.stiffness = 1f;
        foreach (WheelCollider wheel in wheels) // Apply these friction curves to
all wheels
        {
            wheel.forwardFriction = forwardFrictionCurve;
            wheel.sidewaysFriction = sidewaysFrictionCurve;
        }
    }

    private void SpeedyMcSpeedface()
    {
        if (carpower < 2 && urgeforce <= 50)
        {
            urgeforce = urgeforce + 340f;
        }
        else if (carpower > 6 && urgeforce > 0)
        {
            urgeforce = urgeforce - 30f;
        }
        else if (carpower > 12 && urgeforce < 0)
        {
            urgeforce = urgeforce - 80f;
        }
    }



    private void DodgeThis()
    {
        if (SpideySense(L1S, senor1place, "Obstacles"))
        {
            SteerLikeYouMeanIt(1);
        }
        if (SpideySense(R1S, senor1place, "Obstacles"))
        {
            SteerLikeYouMeanIt(-1);
        }
        if (SpideySense(L2S, senor2place, "Obstacles"))
        {
            SteerLikeYouMeanIt(1);
        }
        if (SpideySense(R2S, senor2place, "Obstacles"))
        {
            SteerLikeYouMeanIt(-1);
        }
    }
    private void FixedUpdate()
    {
        KeepOnTrack();
        DodgeThis();
        SpeedyMcSpeedface();
        PowerMotor();
        WheelyAwesome();

        angle_x = ORS.eulerAngles.x;
        angle_z = ORS.eulerAngles.z;
        carpower = rb.velocity.magnitude;
    }
}
```