Tiny Car Controller Advance

User manual v1.1.0

Table of Contents

Description

This controller allows you to create and control a basic vehicle with fully configurable, arcade-like physics.

Instead of using Unity's default wheel collider, which is often needlessly complicated and prone to glitches, the wheels on this controller are made of sphere colliders with several specifically configured joints to mimic a vehicle's motor, steering, and suspension.

Features

- Easy hassle-free setup
- Control acceleration, speed, friction, transmission, collisions, and more
- Uses Unity's physics engine for perfect compatibility with other assets
- Lightweight, perfect for mobile games
- Includes example scripts to take care of input and camera
- Compatible with any OS, render pipeline, or Unity version

Requirements

Unity version 2019.4 or later is recommended, but it should also work with any newer versions.

Intermediate C# programming knowledge is strongly advised.

Contact

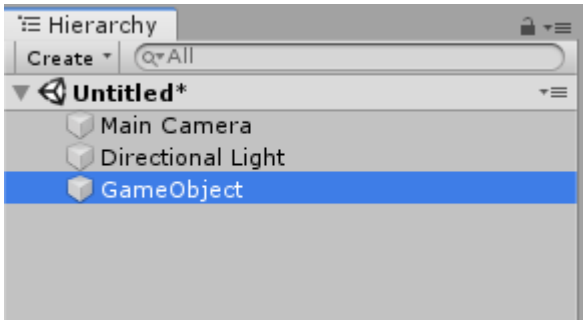
David Jalbert (programmer)
Email: jalbert.d@hotmail.com
Twitter: https://twitter.com/DavidJayIndie

Unity asset store package
https://assetstore.unity.com/packages/slug/198873

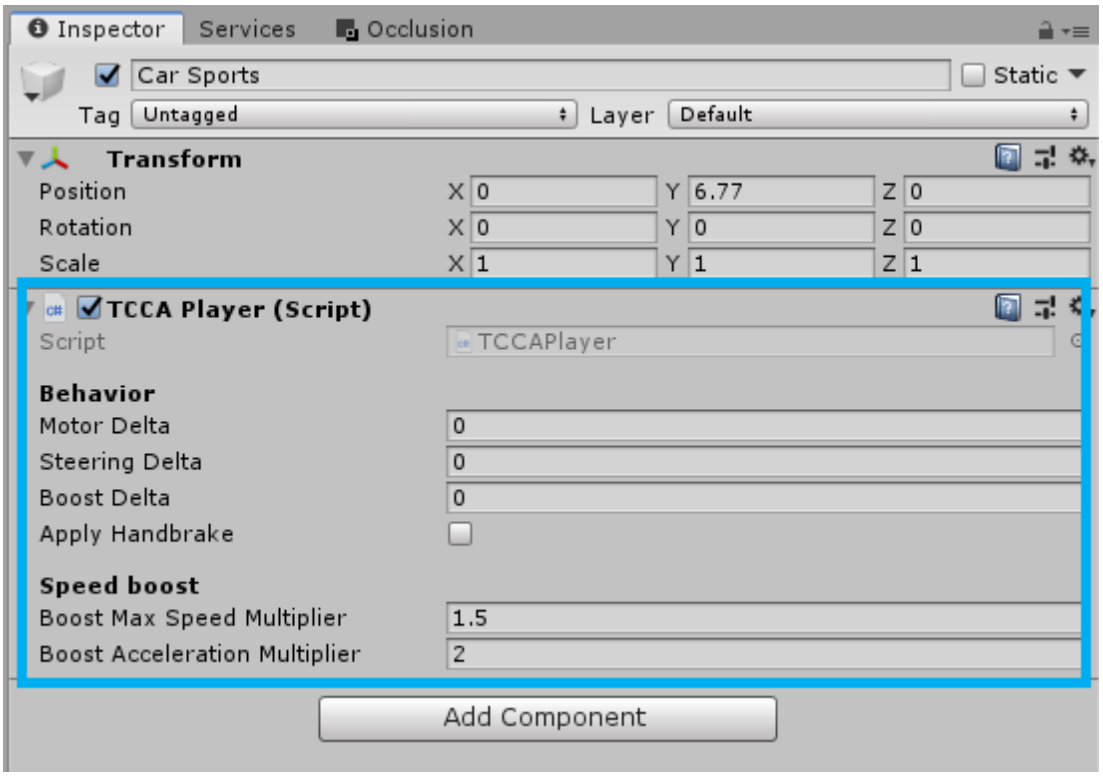
WebGL demo
https://davidjalbert.itch.io/tiny-car-controller-advance-webgl-demo

Minimal setup

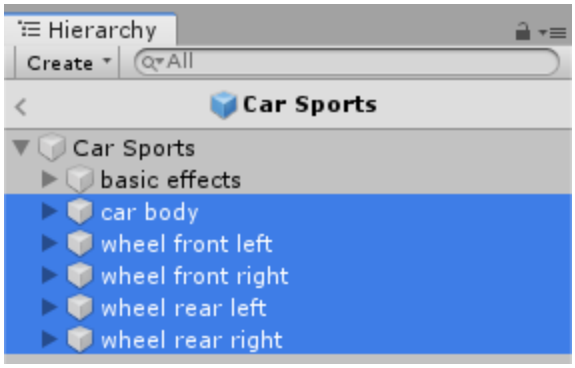
1) Create an empty GameObject in your scene.



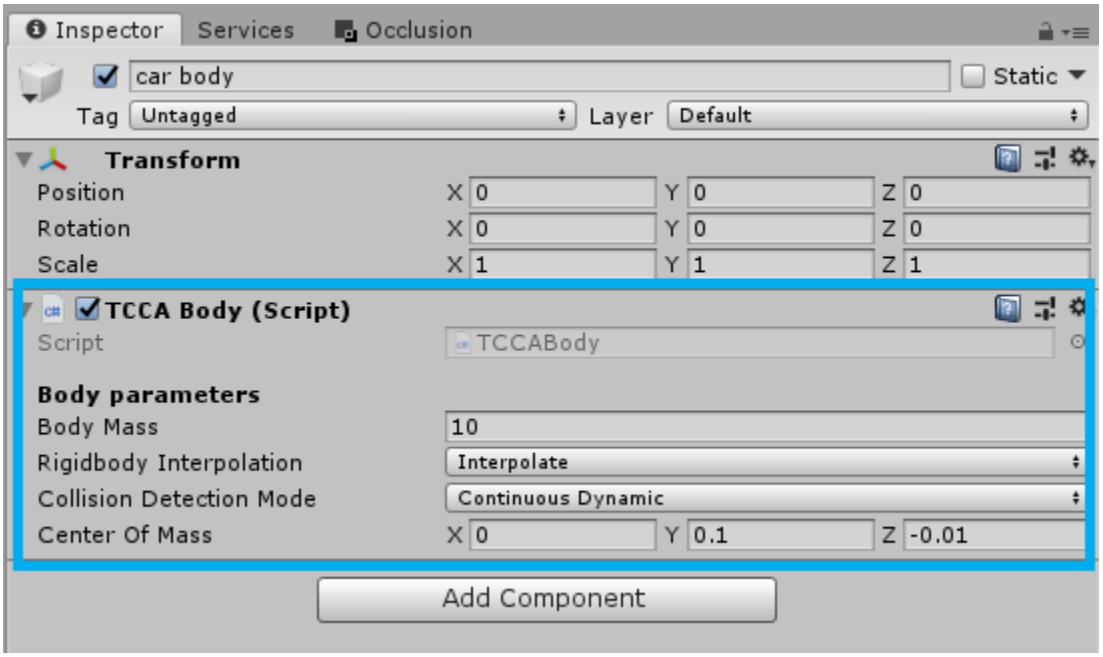
2) Add the script at “Assets/DavidJalbert/TinyCarControllerAdvance/Scripts/Core/TCCAPlayer.cs” to the empty GameObject



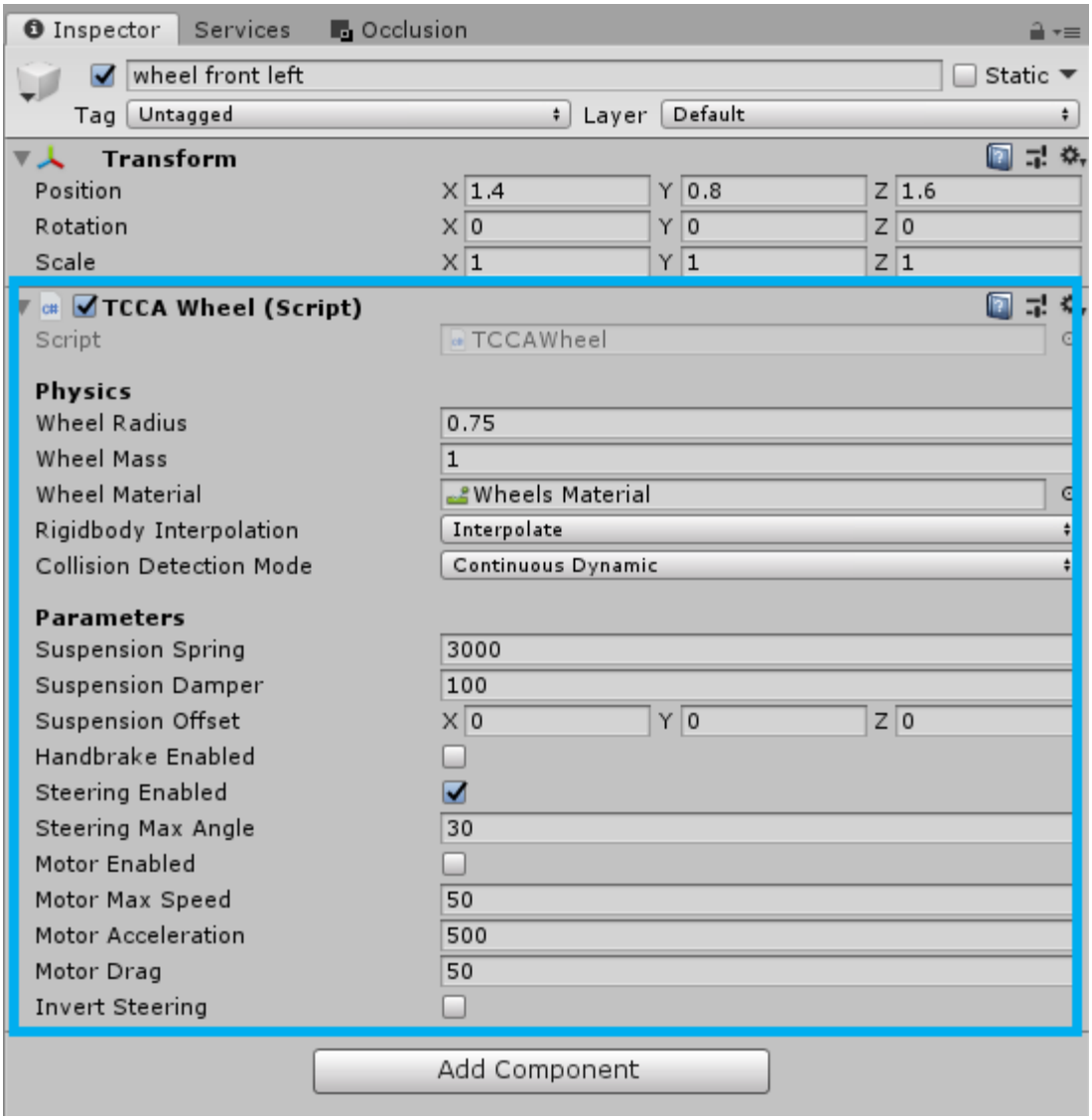
3) Create children for the car body and its wheels. Put your 3d models inside these objects.



4) Add the script at “Assets/DavidJalbert/TinyCarControllerAdvance/Scripts/Core/TCCABody.cs” to the car body object.



5) Add the script at “Assets/DavidJalbert/TinyCarControllerAdvance/Scripts/Core/TCCAWheel.cs” to each of the car's wheel objects. Note that if there are colliders to the wheel models, they will be disabled when the game starts.



6) Set the parameters of the three scripts to your liking (see below for an explanation).

At this point, your controller is ready to use, but it will not have any input or camera control. To do that, you need additional scripts, which are used in the example scene. Check out the scene at “DavidJalbert\TinyCarControllerAdvance” to see how to use them.

Parameters

TCCA Player

Motor Delta

How much torque to apply to the wheels. 1 is full speed forward, -1 is full speed backward, 0 is rest.

Steering Delta

How much steering to apply to the wheels. 1 is right, -1 is left, 0 is straight.

Boost Delta

How much boost to apply to the wheels. 1 is full boost, 0 is no boost.

Apply Handbrake

Whether to apply the handbrake to the wheels.

Boost Max Speed Multiplier

Speed multiplier to apply when using the boost.

Boost Acceleration Multiplier

Acceleration multiplier to apply when using the boost.

TCCA Body

Body Mass

The mass that will be applied to the body.

Rigidbody Interpolation

Whether to apply interpolation to the body.

Collision Detection Mode

Which collision detection mode to use on the body.

Center Of Mass

The center of mass of the body in local space. Ideally this should be the center of the car at ground level. Change the Z value to make the car lean backward or forward when in the air.

Roll Counter Mode

When to apply roll countering force.

Roll Counter Target Angle

The angle in degrees to which to rotate the vehicle. Set to 0 to roll perfectly upright.

Roll Counter Force

How much force to apply to rotate the vehicle upright if it rolls over.

Roll Counter Smoothing

How fast to rotate the vehicle upright if it rolls over. Set to zero to make this instantaneous.

Roll Counter Over Speed

How much force, between 0 (none) and 1 (max), to apply relative to the vehicle's speed, between 0 (stationary) and 1 (max speed).

Pitch Counter Mode

When to apply pitch countering force.

Pitch Counter Target Angle

The angle in degrees to which to rotate the vehicle. Set to 0 to level perfectly straight.

Pitch Counter Force

How much force to apply to level the vehicle.

Pitch Counter Smoothing

How fast to level the vehicle. Set to zero to make this instantaneous.

Pitch Counter Over Speed

How much force, between 0 (none) and 1 (max), to apply relative to the vehicle's speed, between 0 (stationary) and 1 (max speed).

TCCA Wheel

Wheel Radius

Radius of the wheel collider. This should be equal to the size of the wheel model.

Wheel Mass

Mass of the wheel rigidbody.

Wheel Material

Material of the wheel rigidbody.

Rigidbody Interpolation

Whether to use interpolation for the wheel rigidbody.

Collision Detection Mode

Which collision detection mode to use for the wheel collider.

Suspension Spring

Force applied to the suspension. Higher values make the suspension stiffer.

Suspension Damper

Damper applied to the suspension. Higher values make the suspension settle faster.

Suspension Offset

Shifts the position of the wheel relative to its initial position. Useful to mimic hydraulics.

Steering Enabled

Whether to allow the wheel to turn left or right.

Steering Max Angle

Maximum angle at which the wheel can turn.

Steering Over Speed

How much steering, between 0 (none) and 1 (max), to apply relative to the vehicle's speed, between 0 (stationary) and 1 (max speed).

Steering Spring

The amount of force to apply to the steering axle.

Steering Damper

The amount of friction to apply to the steering axle.

Invert Steering

Whether to invert steering. Useful for the rear wheels if you want to have a four wheel steering.

Motor Enabled

Whether to allow the wheel to accelerate.

Motor Max Speed

Maximum speed to which the wheel can keep accelerating when using the motor.

Motor Acceleration

Maximum acceleration to apply to the wheel when using the motor.

Motor Acceleration Over Speed

How much acceleration, between 0 (none) and 1 (max), to apply relative to the vehicle's speed, between 0 (stationary) and 1 (max speed).

Motor Drag

Speed at which the wheel will decelerate when not accelerating.

Handbrake Enabled

Whether to allow the wheel to use the handbrake.

Troubleshooting

Can I use different colliders for the wheels than the default sphere?

Unfortunately, this controller has been designed to use a specific combination of rigidbodies, colliders, and joints for the wheels, so you can't use custom colliders. However, you can use any type of collider for the car body, and the wheels won't collide with it, so if you wanted for instance to have the wheels only touch the ground and not the sides of another object, you could create a collider on the car body that intersect the wheels.

How can I detect when the wheels collide with a custom object?

You can create a script that extends the TCCAWheel script and use that instead. Then you can override the following collision functions, which are called at the same time as their Rigidbody counterparts;

- public virtual void onCollisionStay(Collision collision) { }
- public virtual void onCollisionEnter(Collision collision) { }
- public virtual void onCollisionExit(Collision collision) { }
- public virtual void onTriggerStay(Collider other) { }
- public virtual void onTriggerEnter(Collider other) { }
- public virtual void onTriggerExit(Collider other) { }

The car jumps a bit when running over seams in the road. How can I fix that?

Try changing the values of the "Default Contact Offset" parameter in the Physics tab of the Project Settings to minimize the effects of "ghost collisions". Ideally, this controller works best on connected meshes.