MMV Modern Military Vehicle System

Ruan Lucas

Oct 02, 2022

CONTENTS

1 Introduction **3**

2 Overview **5**
- 2.1 The Main System . . . 5
- 2.2 How the tracked vehicle behaves . . . 5

3 Tracked vehicle Component Configuration **9**
- 3.1 Vehicle Component . . . 9

4 Wheeled Vehicle **15**
- 4.1 Vehicle Component . . . 15

5 Configuration Files **21**
- 5.1 Creating Any Configuration File . . . 21
- 5.2 Engine Settings . . . 22
- 5.3 Wheels Settings . . . 23

6 Another Systems **25**
- 6.1 Shooter Manager Component . . . 25
- 6.2 Bullet Projectile . . . 26
- 6.3 Standart Camera Contoller Component . . . 28

Welcome to the **MMV (Modern Military Vehicle System**) documentation. Here you will have all the information you need to create your own vehicles based on the system, if you find any writing errors or even system inconsistency please report the problem to us via github. Implementation and correction requests are also welcome!

To get started, you can download the system through the store.

Video tutorials are also available on Youtube

Below you will see all the **MMV** documentation separated by themes, enjoy:

CHAPTER

ONE

INTRODUCTION

MMV is a military vehicle control system for games made in Unity, it should provide good vehicle physics in addition to a range of possibilities for different games through its architecture that allows the implementation of custom systems. The system also provides by default some systems ready to help and all of them will be explained in the following modules of the documentation.

CHAPTER

TWO

OVERVIEW

Here's a summary of how the **MMV** architecture works and what you can do.

2.1 The Main System

The main system consists of vehicle physics and the entire control system, such as acceleration, steering, braking and gun turret movement. All other systems are just **add-ons** to this main system and are **not mandatory**, allowing them to be replaced by one customized to the developer's taste. Here's a summary of how the **MMV** architecture works and what you can do.

The vehicle system is made up of **modules** that communicate, each module is responsible for an essential part of the vehicle. You can access some vehicle API functions and give commands, for example, tell the vehicle where to go or tell the gun turret module where to aim, (this will be explained in the next modules of this documentation).

2.2 How the tracked vehicle behaves

The **MMV** has a custom wheel physics system, we managed to make the wheel configuration very simple and it is still very stable for the purpose of Asset. The vehicle's system will already do most of the hard work for you and so you just need to take care of passing the control commands.

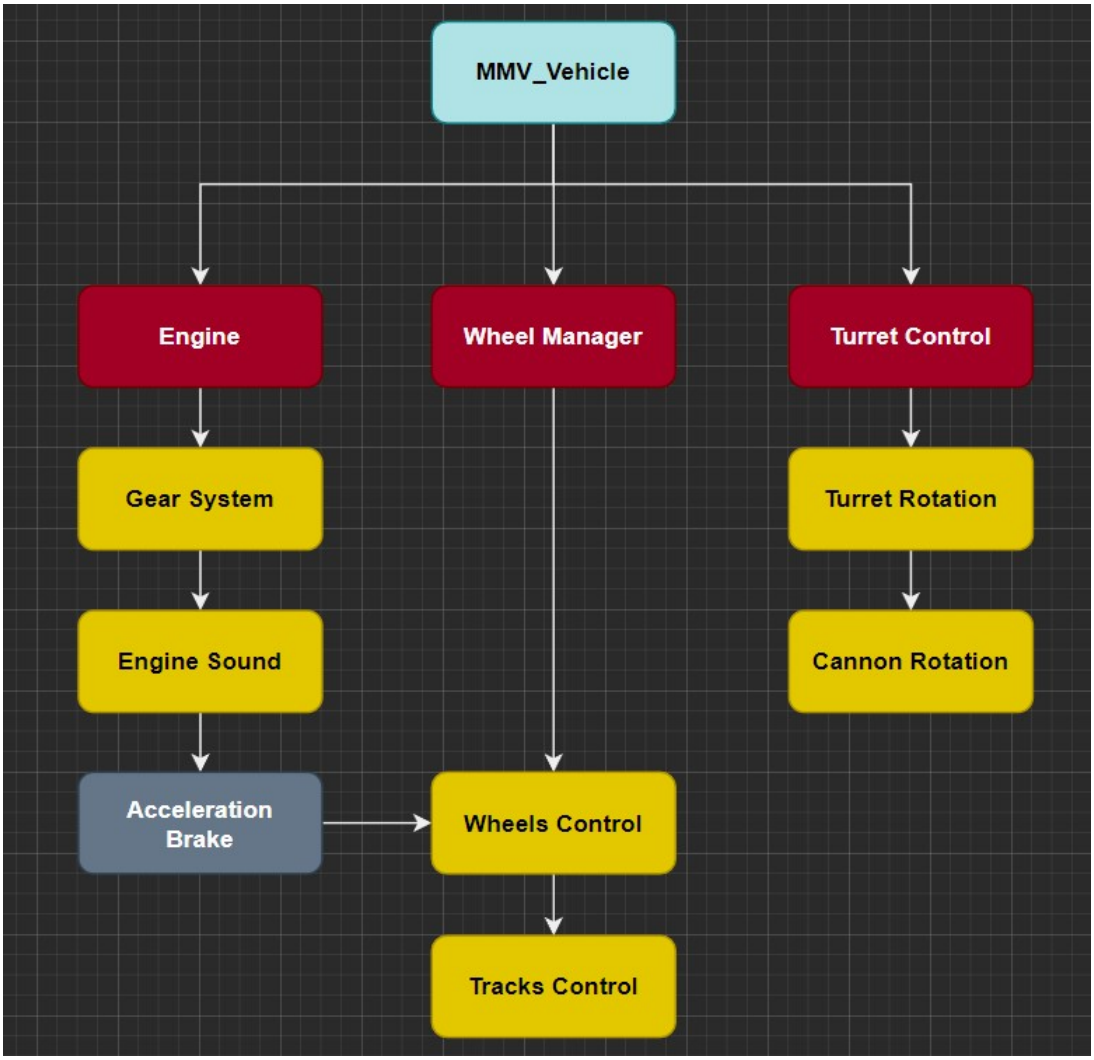
2.2.1 Movimentation

In the case of tracked vehicle, all your vehicle movement control is based on the acceleration of the wheels, whether they are going backwards or forwards and the speed at which they are moving. An MBT vehicle from the **MMV** is no different, we use this same principle, to better understand, below is an example of how the acceleration influences the vehicle's direction.

2.2.2 How do Tracks Work

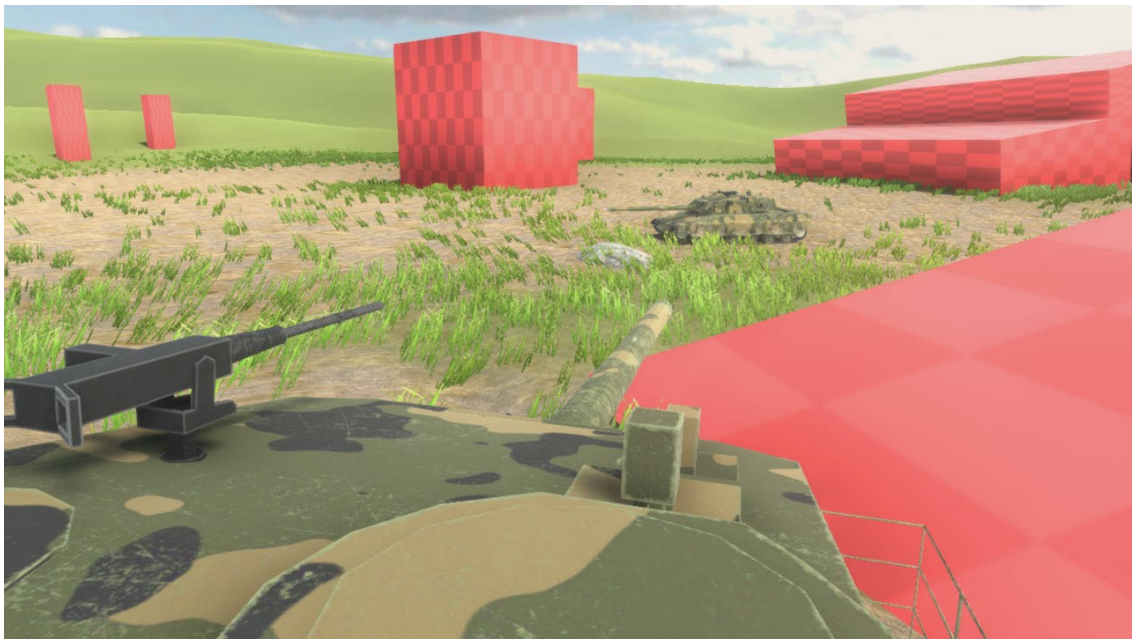
The tracks use fake simulation to look like they actually move. The treadmill model is linked to a bone armature, one bone per wheel and via script this bone will follow the wheel's position generating the effect that movement.

This solves the suspension problem, but there is another point that needs attention, which is the issue of vehicle acceleration, the treadmill needs to respond to the speed at which the vehicle moves and for that there is an integrated system in the wheel control script of the MBT vehicle that changes the texture offset of the mat material giving the feeling of movement.


MMV_Vehicle
Engine
Wheel Manager
Turret Control
Gear System
Turret Rotation
Engine Sound
Cannon Rotation
Acceleration Brake
Wheels Control
Tracks Control

2.2.3 Weapon System

Only the turret system is integrated into the vehicle, fire control is done separately by other scripts. The main job of the turret is just aiming at some position in the world respecting the vehicle's angle limits.



CHAPTER

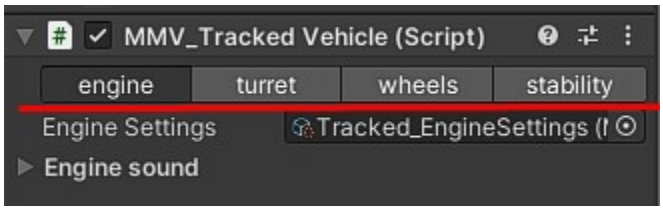
THREE

TRACKED VEHICLE COMPONENT CONFIGURATION

Responsible for simulating the entire system of a vehicle with tracks.

3.1 Vehicle Component

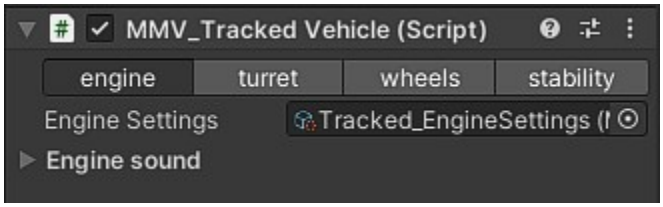
A vehicle component is composed of some modules, each module controls a specific part as you can see in the image.



Let's explain the vehicle component separated by modules.

3.1.1 Engine

Module responsible for all parts of the vehicle's power, movement, engine sound and braking system.



Engine Settings: *configuration file* containing all engine parameters

Engine Sound

Control the sound of the vehicle's engine.

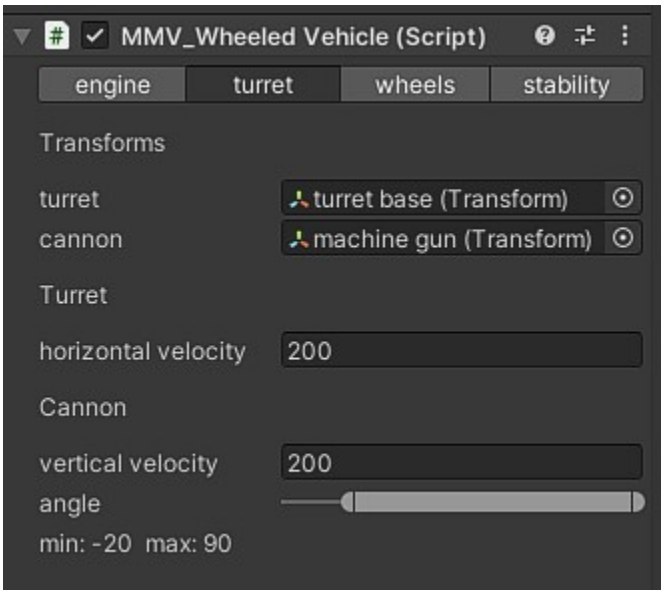
audio source: Any audio source that is in the vehicle that can be used to reproduce the sound of the engine.

audio clip: Your engine's audio clip.

min pitch: The lowest pitch that the engine sound can reach if it's not accelerating.

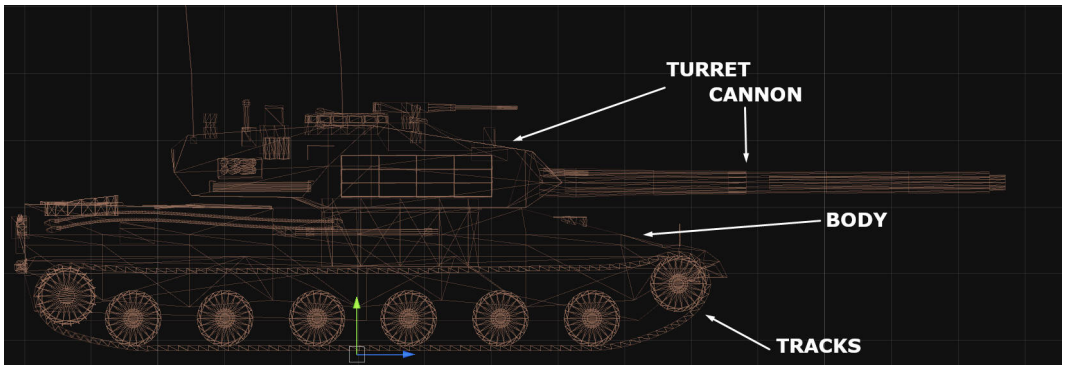
max pitch: The maximum pitch that the engine sound can reach when accelerating.

3.1.2 Turret



This module is responsible for controlling the turret and aiming the vehicle.

Transforms

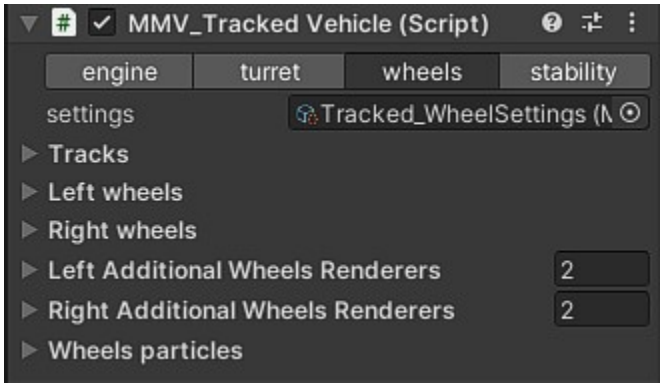


turret: Vehicle weapon system turret.

cannon: Cannon that is connected to the vehicle's turret.

horizontal velocity: The speed at which the turret flips horizontally towards the target. **vertical velocity:** The speed at which the cannon turns vertically towards the target.

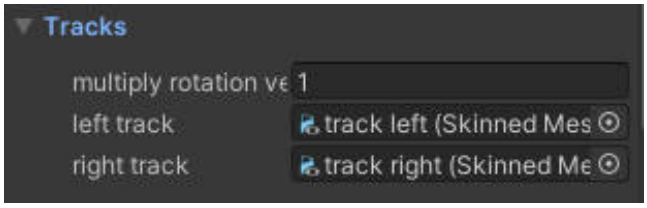
3.1.3 Wheels



The wheel module manages all of the vehicle's wheels, applies suspension physics and tells them when to accelerate or brake. It makes the vehicle turn and even the tracks move.

settings: *Configuration file* that contains all the parameters for suspension and behavior of the vehicle's wheels

3.1.4 Tracks



Add here the meshes of your vehicle's tracks, so that they follow the movement of the wheels.

multiply rotation velocity: If your belt is not moving at the correct speed, change this value to correct the speed.

Left/Right Wheels

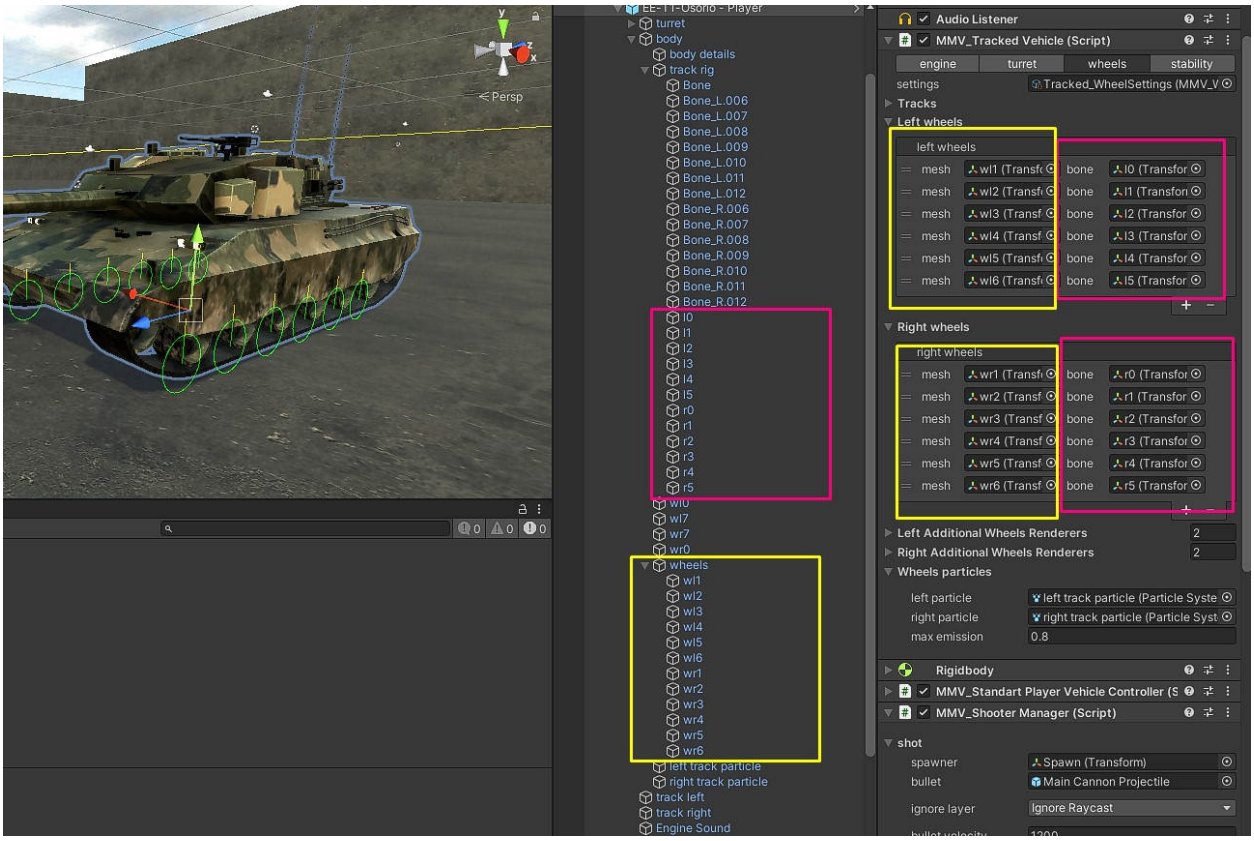
Add your vehicle's wheels here.

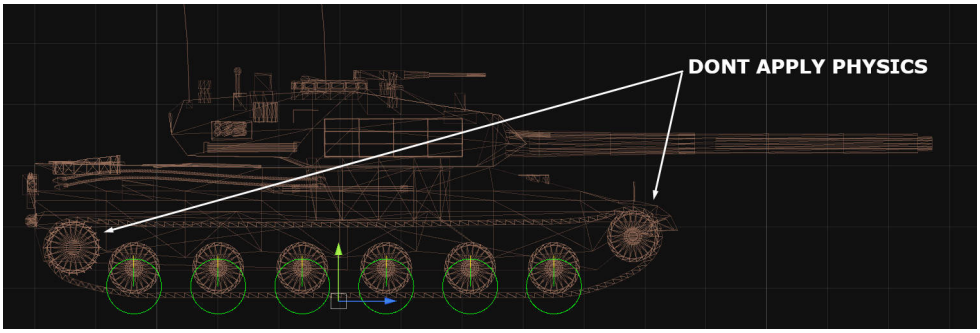
Mesh: Object that will be used to apply wheel physics on the vehicle.

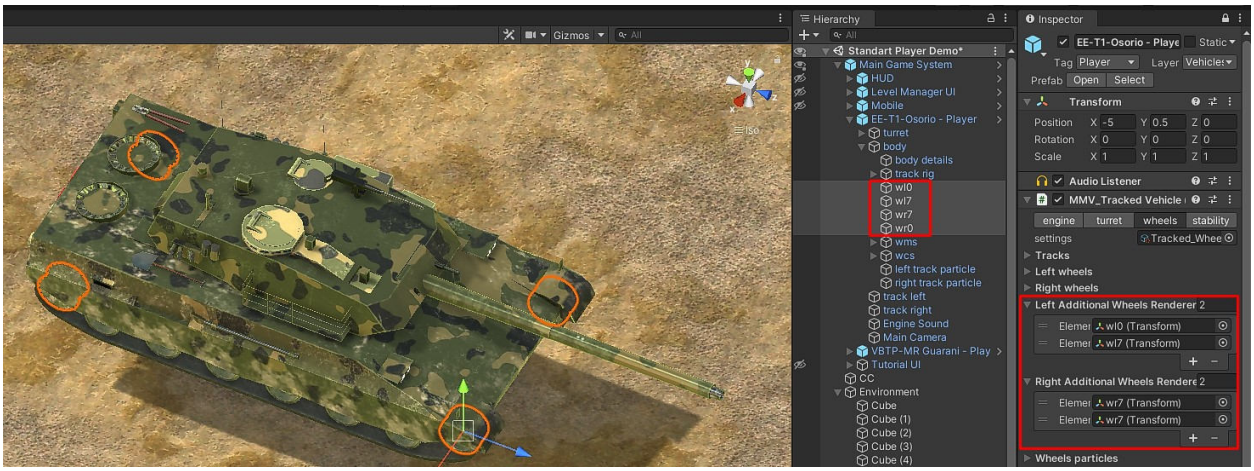
Bone: A track bone, which is next to the wheel.

Left/Right Additional Wheels Renderers

Add here the wheel meshes that don't apply physics but must rotate along with the others like the front and back wheels of the tank.


Persp
EE-T1-Osorio - Player
turret
body
body details
track rig
Bone
Bone_L.006
Bone_L.007
Bone_L.008
Bone_L.009
Bone_L.010
Bone_L.011
Bone_L.012
Bone_R.006
Bone_R.007
Bone_R.008
Bone_R.009
Bone_R.010
Bone_R.011
Bone_R.012
l0
l1
l2
l3
l4
l5
r0
r1
r2
r3
r4
r5
wl0
wl7
wr7
wr0
wheels
wl1
wl2
wl3
wl4
wl5
wl6
wr1
wr2
wr3
wr4
wr5
wr6
left track particle
right track particle
track left
track right
Engine Sound
Audio Listener
MMV_Tracked Vehicle (Script)
engine
turret
wheels
stability
settings
Tracked_WheelSettings (MMV_V
Tracks
Left wheels
left wheels
mesh
bone
Right wheels
right wheels
Left Additional Wheels Renderers
Right Additional Wheels Renderers
Wheels particles
left particle
left track particle (Particle Syste
right particle
right track particle (Particle Syst
max emission
0.8
Rigidbody
MMV_Standart Player Vehicle Controller (S
MMV_Shooter Manager (Script)
shot
spawner
Spawn (Transform)
bullet
Main Cannon Projectile
ignore layer
Ignore Raycast


DONT APPLY PHYSICS


Gizmos
Hierarchy
Standart Player Demo*
Main Game System
HUD
Level Manager UI
Mobile
EE-T1-Osorio - Player
turret
body
body details
track rig
wl0
wl7
wr7
wr0
wms
wcs
left track particle
right track particle
track left
track right
Engine Sound
Main Camera
VBTP-MR Guarani - Play
Tutorial UI
CC
Environment
Cube
Cube (1)
Cube (2)
Cube (3)
Cube (4)
Inspector
EE-T1-Osorio - Playe
Static
Tag
Player
Layer
Vehicles
Prefab
Open
Select
Transform
Position
Rotation
Scale
Audio Listener
MMV_Tracked Vehicle
engine
turret
wheels
stability
settings
Tracked_Whee
Tracks
Left wheels
Right wheels
Left Additional Wheels Renderer 2
Element
wl0 (Transform)
wl7 (Transform)
Right Additional Wheels Renderer 2
wr7 (Transform)
wr7 (Transform)
Wheels particles

Wheels Particles

It is possible to add particles to the wheels so that when the vehicle moves, they are installed, such as dust.

left/right particle: The particle on either side of the vehicle.

max emission: The particle on either side of the vehicle.

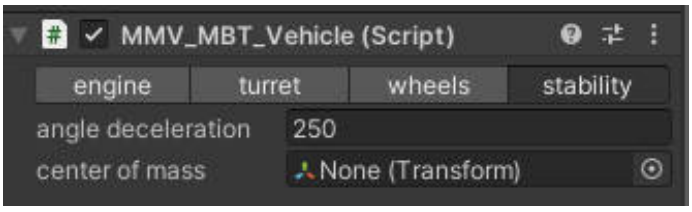
3.1.5 Stability

Control vehicle stability.

Angle deceleration: how much gravity influences the vehicle when going uphill or steep places.

center of mass: The vehicle's center of mass, recommended to leave in the center, the higher on the Y axis, the easier it will be for the vehicle to tip over in curves.


left particle
Particle System
Open Editor...
left particle
Emission
Shape
Shape
Box
Emit from:
Volume
Texture
None (Texture 2D)
Position
Rotation
Scale
Align To Direction
Randomize Direction
Spherize Direction
Randomize Position
Velocity over Lifetime
Limit Velocity over Lifetime
Inherit Velocity
Lifetime by Emitter Speed
Force over Lifetime
Color over Lifetime
Color by Speed
Size over Lifetime
Size by Speed
Rotation over Lifetime
Rotation by Speed
External Forces
Noise
Collision
Triggers
Sub Emitters
Texture Sheet Animation
Lights
Trails
Custom Data
Renderer
Particle System Curves
Particle Effect
Play
Restart
Stop
Playback Speed
Playback Time
Particles
Speed Range
Simulate Layers
Nothing
Resimulate
Show Bounds
Show Only Selected


MMV_MBT_Vehicle (Script)
engine
turret
wheels
stability
angle deceleration
250
center of mass
None (Transform)

CHAPTER

FOUR

WHEELED VEHICLE

Responsible for simulating all types of vehicle that uses wheels.

4.1 Vehicle Component

A vehicle component is composed of some modules, each module controls a specific part as you can see in the image.

Let's explain the vehicle component separated by modules.

4.1.1 Engine

Module responsible for all parts of the vehicle's power, movement, engine sound and braking system.

Engine Settings: *configuration file* containing all engine parameters

Engine Sound

Control the sound of the vehicle's engine.

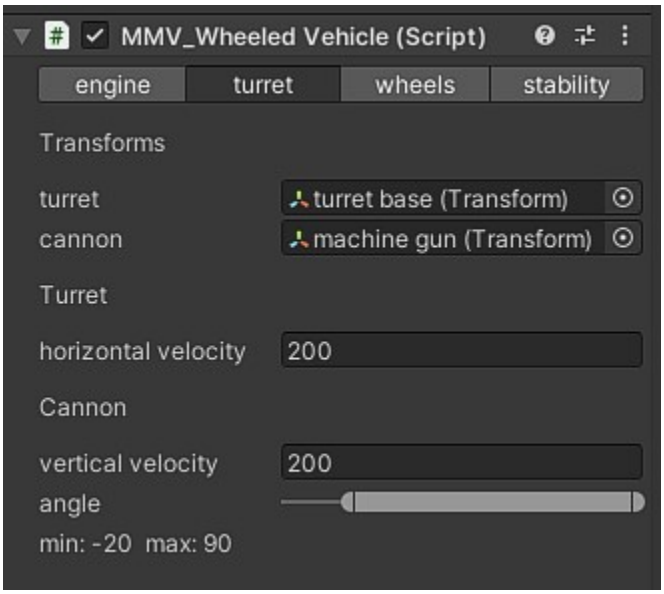
audio source: Any audio source that is in the vehicle that can be used to reproduce the sound of the engine.

audio clip: Your engine's audio clip.

min pitch: The lowest pitch that the engine sound can reach if it's not accelerating.

max pitch: The maximum pitch that the engine sound can reach when accelerating.

4.1.2 Turret



This module is responsible for controlling the turret and aiming the vehicle.

Transforms

turret: Vehicle weapon system turret.

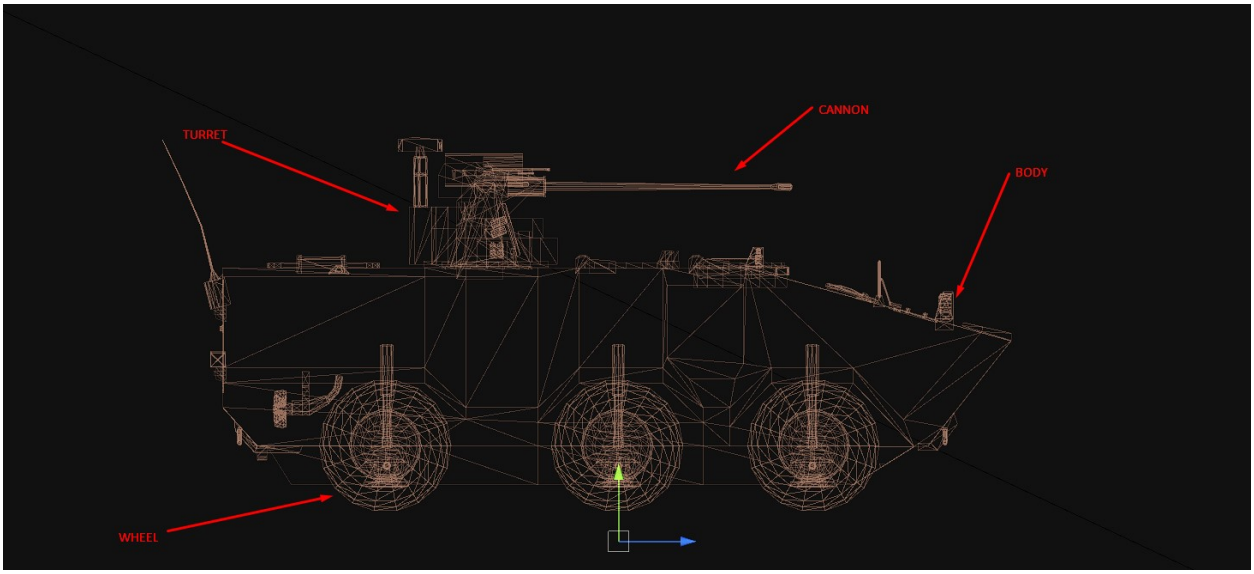
cannon: Cannon that is connected to the vehicle's turret.

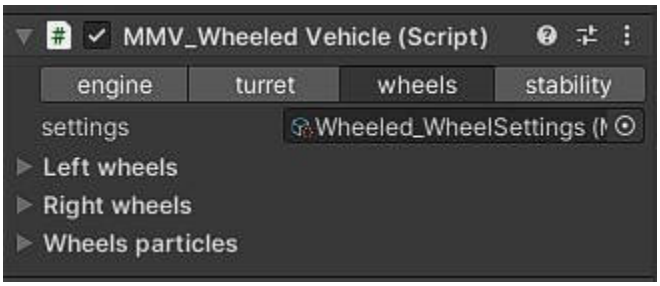
horizontal velocity: The speed at which the turret flips horizontally towards the target. **vertical velocity:** The speed at which the cannon turns vertically towards the target.

4.1.3 Wheels

The wheel module manages all of the vehicle's wheels, applies suspension physics and tells them when to accelerate, brake and steer.

settings: *Configuration file* that contains all the parameters for suspension and behavior of the vehicle's wheels





Left/Right Wheels

Add your vehicle's wheels here.

Mesh: Object that will be used to apply wheel physics on the vehicle

Accelerate: If the wheel is able to use the acceleration force

Brake: If the wheel is able to brake

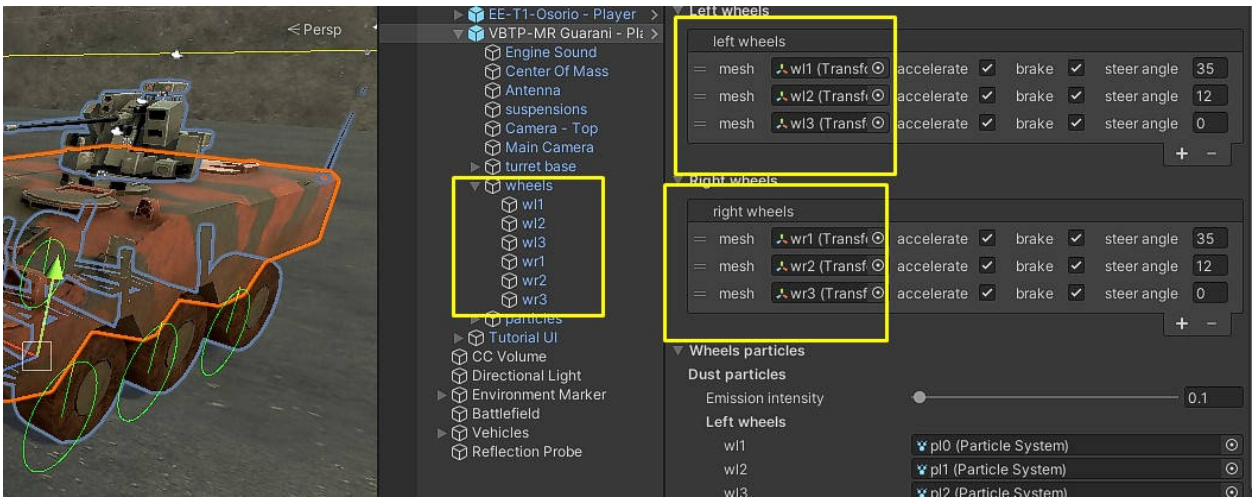
Steer Angle: How far can this wheel turn

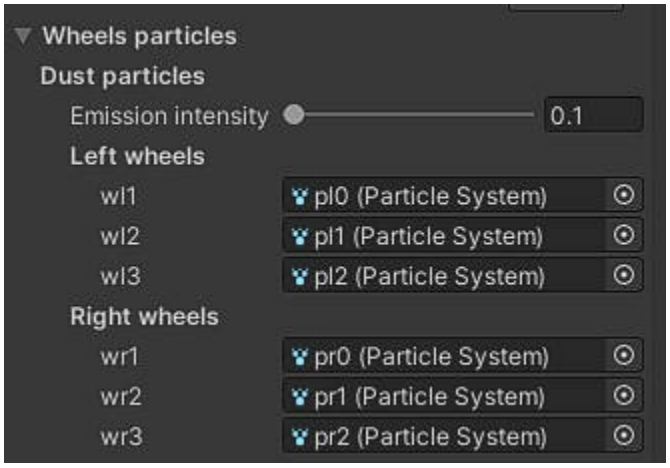
Wheels Particles

Add a particle to each of your wheels so that when the vehicle is driven, dust can come out of the wheels.

Emission intensity: The intensity of particles that will be instantiated at vehicle speed

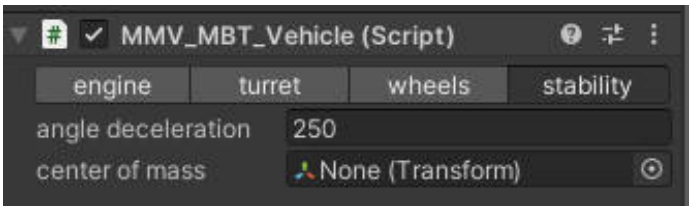
Left/Right wheels: particle from each of your wheels


Persp
EE-T1-Osorio - Player
VBTP-MR Guarani - Pla
Engine Sound
Center Of Mass
Antenna
suspensions
Camera - Top
Main Camera
turret base
wheels
wl1
wl2
wl3
wr1
wr2
wr3
particles
Tutorial UI
CC Volume
Directional Light
Environment Marker
Battlefield
Vehicles
Reflection Probe
Left wheels
left wheels
mesh
wl1 (Transfo
wl2 (Transfo
wl3 (Transf
accelerate
brake
steer angle
35
12
0
Right wheels
right wheels
wr1 (Transfo
wr2 (Transf
wr3 (Transf
Wheels particles
Dust particles
Emission intensity
0.1
Left wheels
wl1
pl0 (Particle System)
wl2
pl1 (Particle System)


Wheels particles
Dust particles
Emission intensity
0.1
Left wheels
wl1
pl0 (Particle System)
wl2
pl1 (Particle System)
wl3
pl2 (Particle System)
Right wheels
wr1
pr0 (Particle System)
wr2
pr1 (Particle System)
wr3
pr2 (Particle System)

4.1.4 Stability

Control vehicle stability.



Angle deceleration: how much gravity influences the vehicle when going uphill or steep places.

center of mass: The vehicle's center of mass, recommended to leave in the center, the higher on the Y axis, the easier it will be for the vehicle to tip over in curves.

CHAPTER

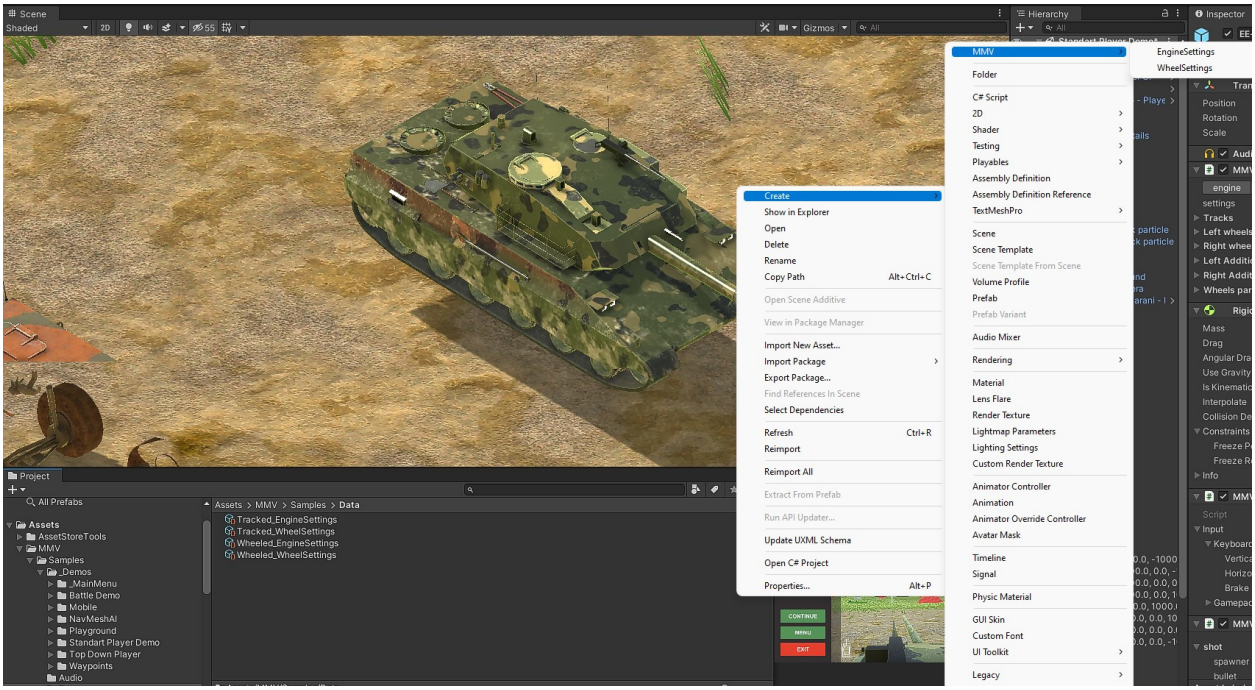
FIVE

CONFIGURATION FILES

Configuration files are scriptableObjects that contain data that can be used by different MMV systems. For example, create a configuration file that contains all information about a type of engine, and that engine can be used by many different vehicles.

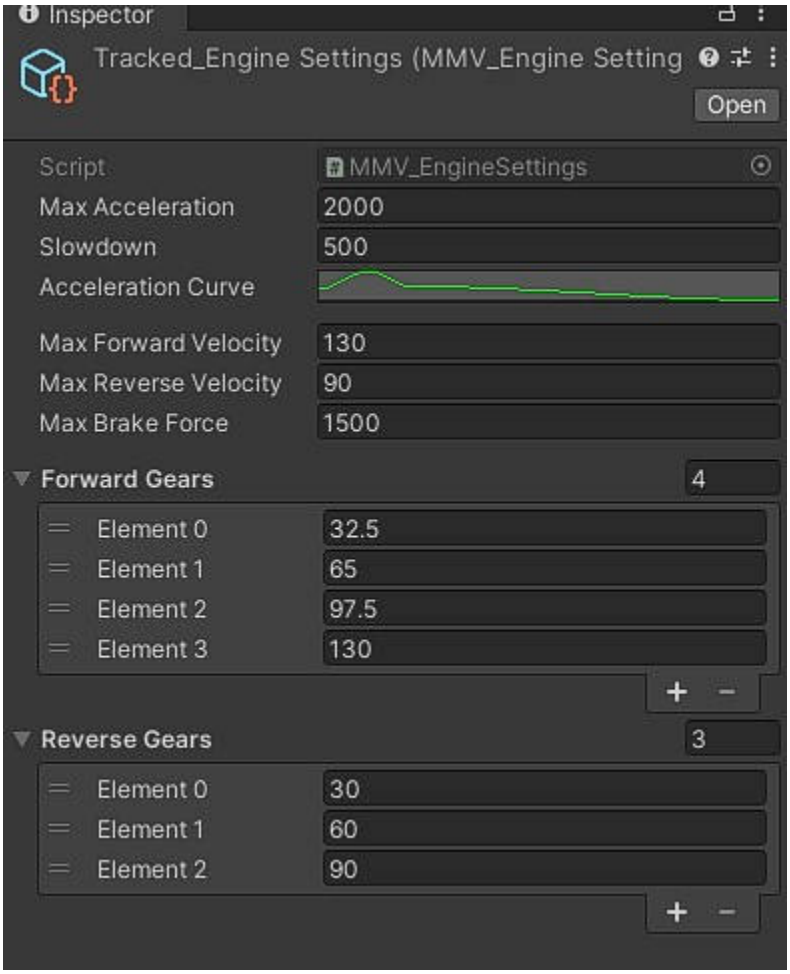
5.1 Creating Any Configuration File

To create a configuration file, go to your “Project” window and right click somewhere:
Create -> MMV -> ....



5.2 Engine Settings

An engine configuration file is responsible for storing all the behavior data of an engine in an MMV vehicle.



Max acceleration: The maximum acceleration force that the vehicle can achieve

Slowdown: Speed at which the vehicle decelerates

Acceleration curve: Acceleration curve based on current vehicle speed, must never go beyond (0 - 1) horizontally or vertically

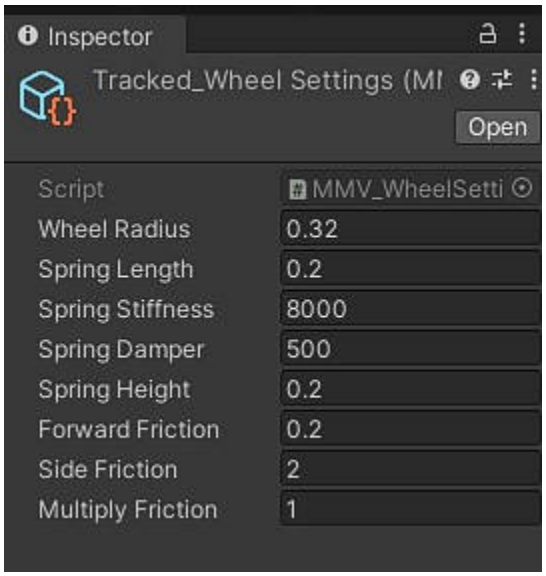
Max forward velocity: Maximum speed the vehicle can reach by accelerating forward

Max reverse velocity: Maximum speed the vehicle can reach by accelerating backwards

Max brake force: Vehicle braking force

Gears Forward/Backward: Forward or reverse gear shifting speeds

5.3 Wheels Settings



Wheel radius: vehicle wheel size

Spring length: Suspension size of each wheel

Spring stiffness: Suspension force

Spring damper: Suspension smoothing

Spring height: Height of start of suspension

Forward friction: front wheel friction

Side friction: side wheel friction

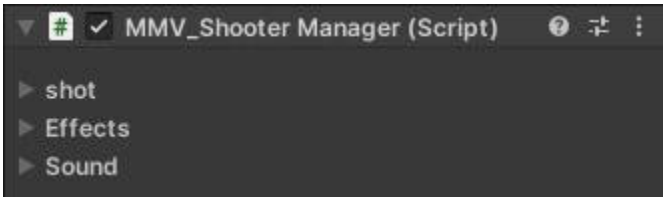
CHAPTER

SIX

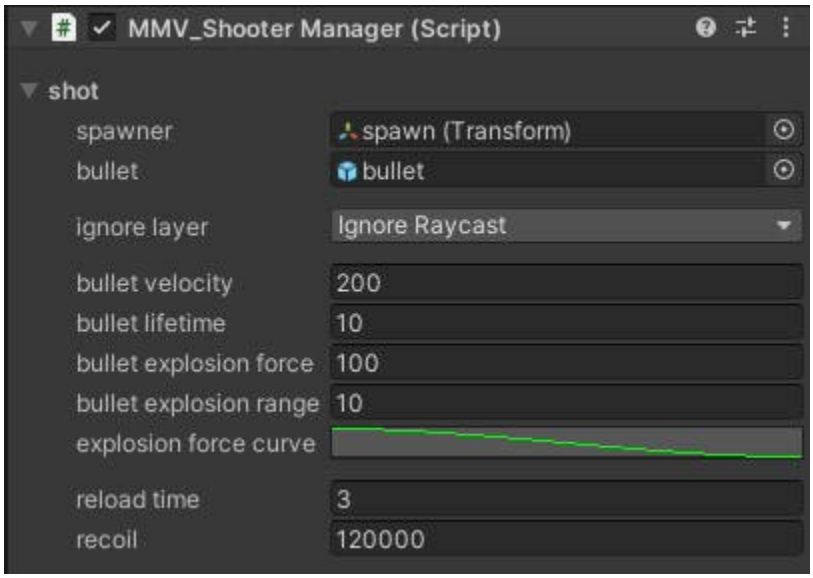
ANOTHER SYSTEMS

6.1 Shooter Manager Component

Responsible for managing gun fire.



6.1.1 Shot



Describe all main shooting behavior.

spawner: Transform from the position where the shot will come from.

bullet: *The projectile* that will be instantiated. It is important that this object has the Projectile component.

ignore layer: The projectile will only identify a collision with another object if it does not have that layer defined.

bullet velocity: Speed in meters per second that the projectile moves.

bullet life time: After "X" seconds the projectile will be destroyed automatically even without having collided with another object.

bullet explosion force: The explosion force that will be assigned to nearby objects when they collide with something.

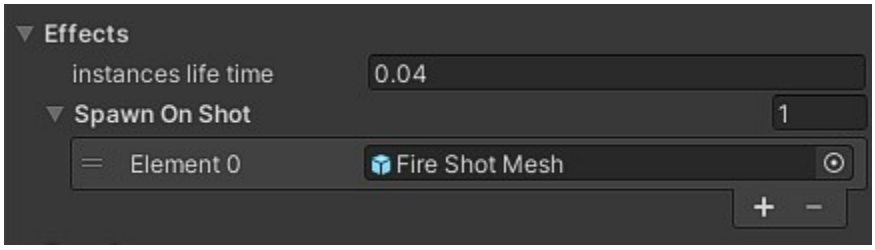
bullet explosion range: The distance to identify nearby objects to apply explosion force after colliding with another object.

explosion force curve: The strength of the explosion force over the distance when the projectile collides.

reload time: The weapon's reload time.

recoil: The force of the shot applied to the vehicle.

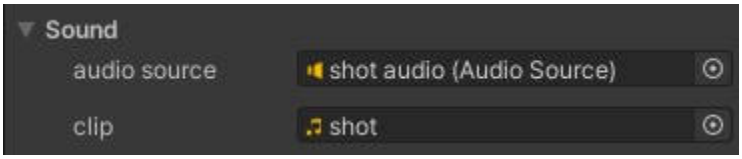
6.1.2 Effects



Instantiates objects when a projectile is instantiated.

Spawn on shot: Objects that must be created, such as particles

6.1.3 Sound



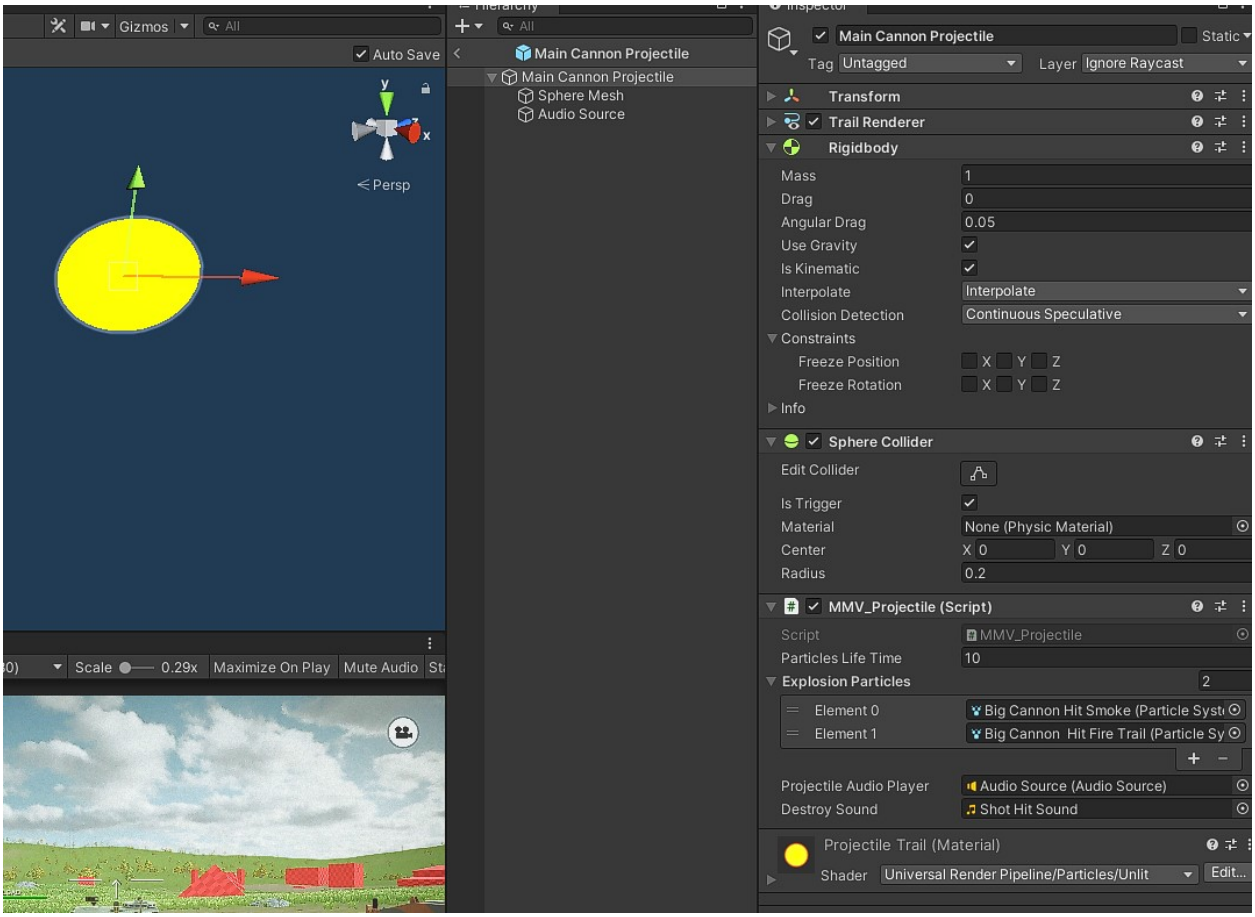
audio source: Audio source that will be used to play the trigger sound.

clip: The audio clip of the shot.

6.2 Bullet Projectile

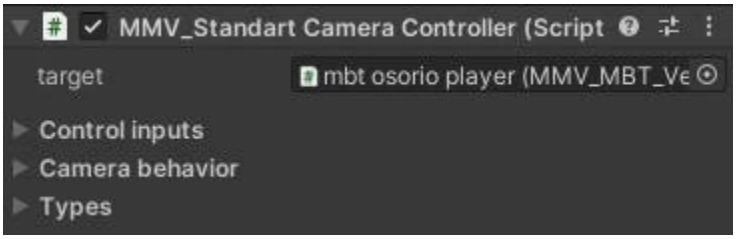
A projectile is an object that can be used by a Shooter manager.

it is formed by a GameObject with a **kinematic Rigidbody** and a **Collider marked as trigger**, add a **projectile component** and your projectile will be fired.

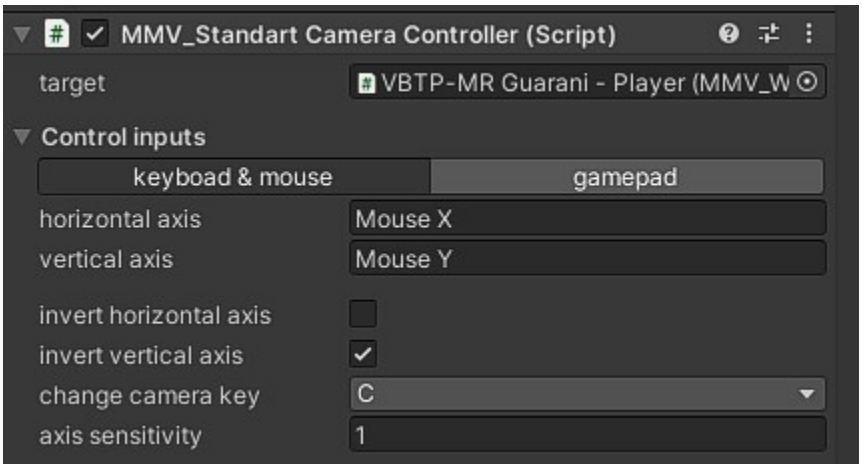

Gizmos
All
Auto Save
Persp
Scale 0.29x
Maximize On Play
Mute Audio
Main Cannon Projectile
Main Cannon Projectile
Sphere Mesh
Audio Source
Main Cannon Projectile
Static
Tag Untagged
Layer Ignore Raycast
Transform
Trail Renderer
Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Use Gravity
Is Kinematic
Interpolate Interpolate
Collision Detection Continuous Speculative
Constraints
Freeze Position X Y Z
Freeze Rotation X Y Z
Info
Sphere Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Radius 0.2
MMV_Projectile (Script)
Script MMV_Projectile
Particles Life Time 10
Explosion Particles 2
Element 0 Big Cannon Hit Smoke (Particle Syst
Element 1 Big Cannon Hit Fire Trail (Particle Sy
Projectile Audio Player Audio Source (Audio Source)
Destroy Sound Shot Hit Sound
Projectile Trail (Material)
Shader Universal Render Pipeline/Particles/Unlit
Edit...

6.3 Standart Camera Contoller Component

Standard MMV camera controller, capable of delivering different types of camera positioning such as 3rd person, commander's view and sniper's view.



target: add the target vehicle to be followed by the camera.



Add player controls to be able to control the camera, it is possible to configure both keyboard and mouse and gamepad.

See for configure your Axes: Unity Input Manager

horizontal Axes: Horizontal Axes of the Input Axes to rotate the camera horizontally.

vertical Axes: Vertical Axes of the Input Axes to rotate the camera vertically.

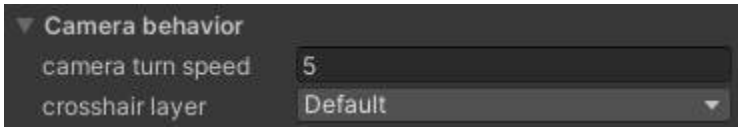
invert horizontal Axes: Invert the direction of the player control's horizontal Axes.

invert vertical Axes: Invert the direction of the player control's vertical Axes.

change camera key: The key or button to switch between cameras if you have more than one.

axis sensitivity: Camera movement sensitivity

6.3.1 Camera Behaviour



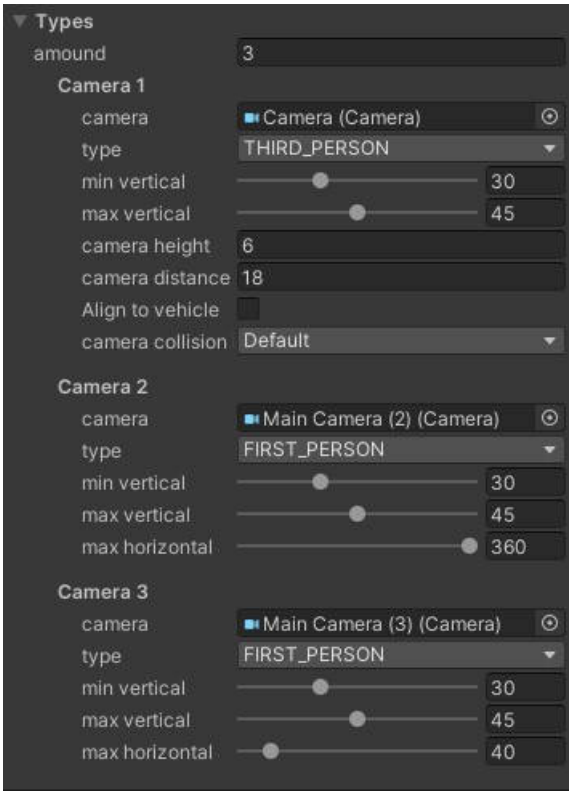
Describe how the camera should behave.

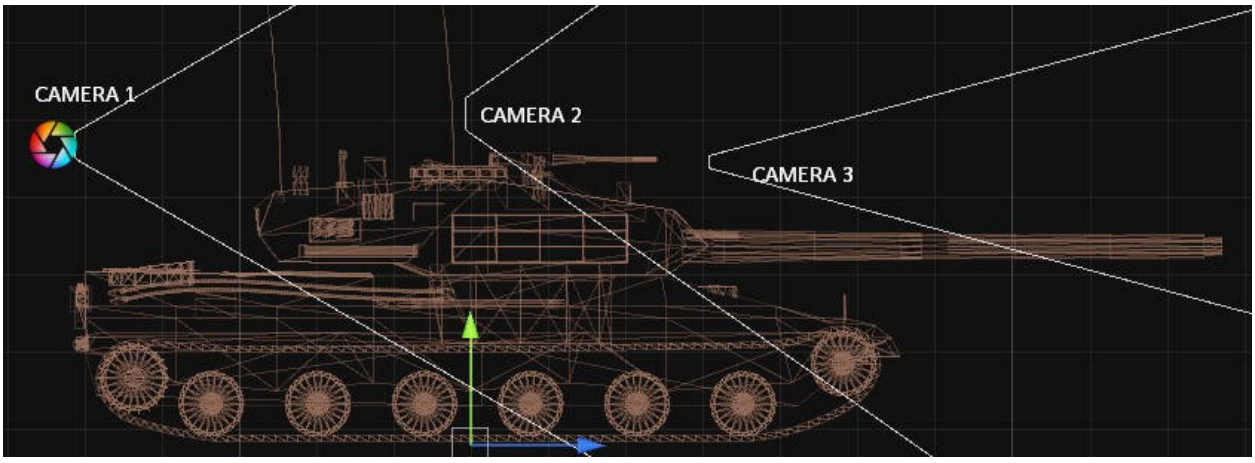
camera turn speed: The speed at which the camera rotates

crosshair layer: Layer of objects that have a collider.

6.3.2 types

Game cameras





amount: Number of vehicle cameras.

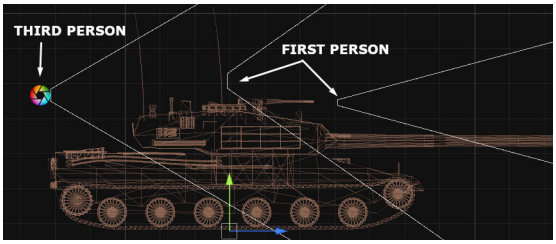
Camera "X"

camera: The chosen camera.

type: The type of camera.

Note: **THIRD_PERSON:** The camera moves around the vehicle and uses the "Camera Collider" to avoid obstacles.

FIRST_PERSON: Stays in the same place, but can be rotated vertically and horizontally.



---

min vertical: The minimum angle to the vertical.

max vertical: The maximum angle vertically.

align to vehicle: Aligns the Y axis of the camera with that of the vehicle, by default it is already activated in FIRST_PERSON mode.

Options for FIRST_PERSON

max horizontal: The maximum angle the camera can turn horizontally.

Options for THIRD_PERSON

camera height: The height of the camera relative to the vehicle.

camera distance: The distance of the camera from the vehicle

camera collision: Camera collision sensor, prevents it from entering walls, add here the collision layers of your scene, by default the layer is *"Default"*.