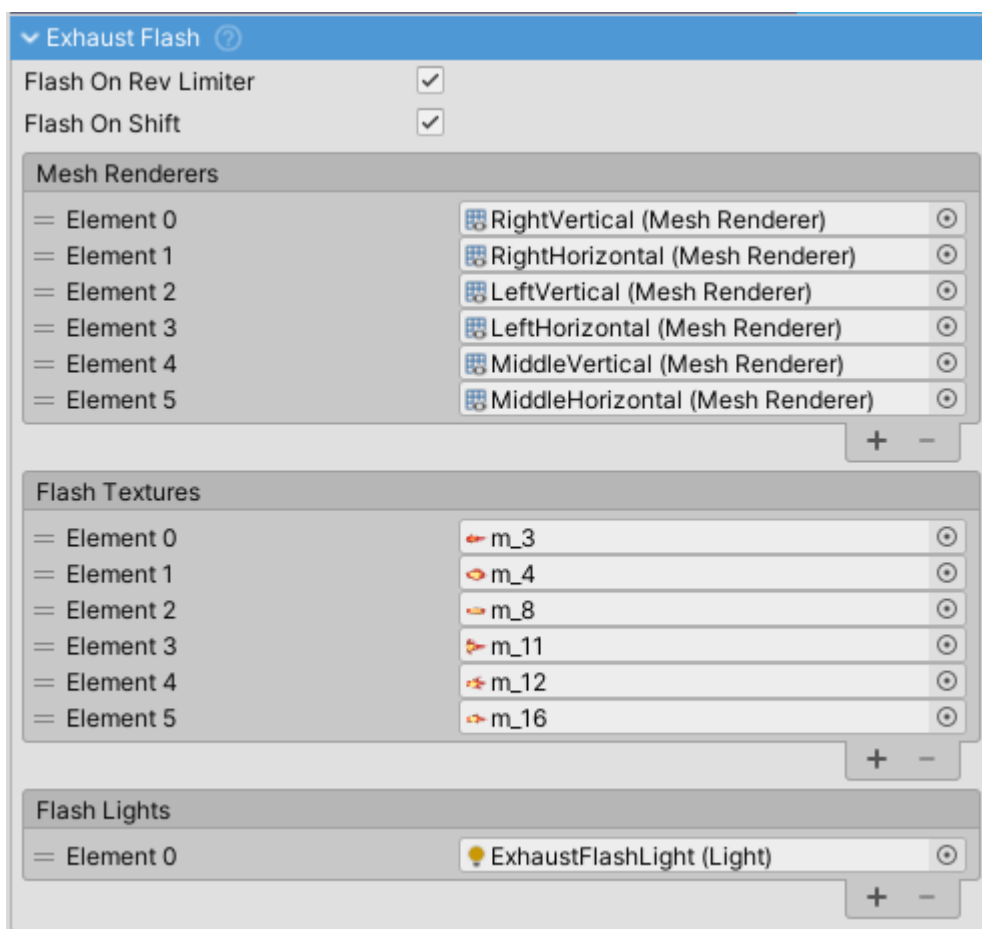


EffectsManager menu.

Effect system makes use of [VehicleComponents](#) and therefore each Effect can be turned on or off, be enabled or disabled or have LOD set.

- All Effects with *manager* in the name manage multiple instances of the effect, usually one for each wheel - e.g. skidmarks or surface particles.

Exhaust Flash



ExhaustFlash inspector.

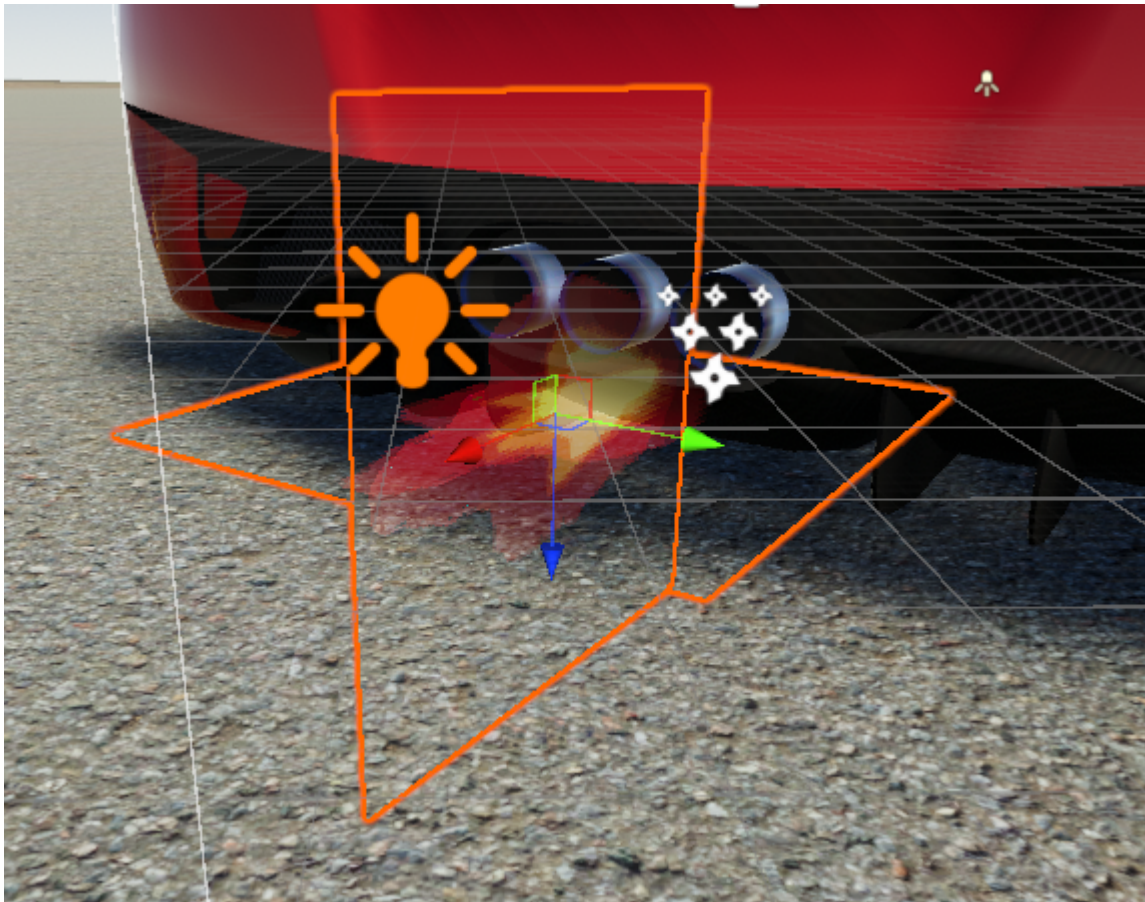
ExhaustFlash Effect is used to imitate flames shooting out of exhaust. The method to achieve this is identical to the one used for most muzzle flash in FPS games; that is a set of images gets enabled and disabled at rapid rate with different sprite and scale each time. This is a performant way to achieve the effect while avoiding particle effects.

Setup

Adding Quads

- Create and place two Quads perpendicular to each other. Move them to the location of the exhaust.

- Remove any colliders from Quads.
- Create a new material that uses Particles/Standard Unlit (Unity 2019) or equivalent shader with Transparent rendering and Multiply color mode. The included example on Sports Car prefab can be copied. Assigning one of the included flame textures will show if the rotation of the Quad is correct. Rotate Quad if needed.



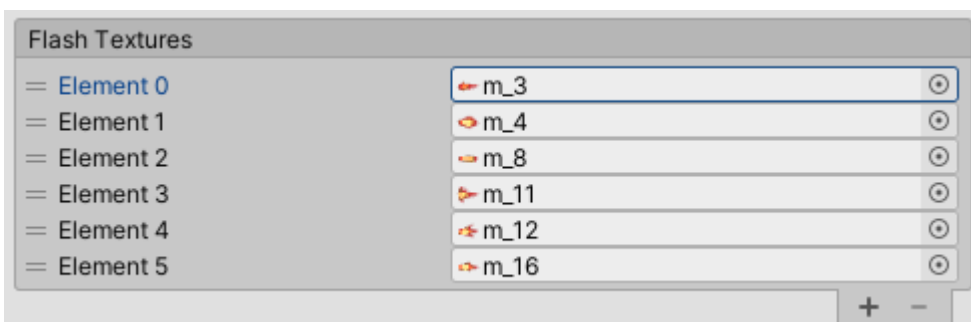
Two perpendicular quads with flame sprites.

- Assign the MeshRenderers from the newly created Quads to the ExhaustFlash ⇒ Mesh Renderers list.

Assigning Textures

To make flames look more convincing a random texture is assigned to each Quad on each flash. A number of default textures is included.

- Assign textures to Flash Textures list.

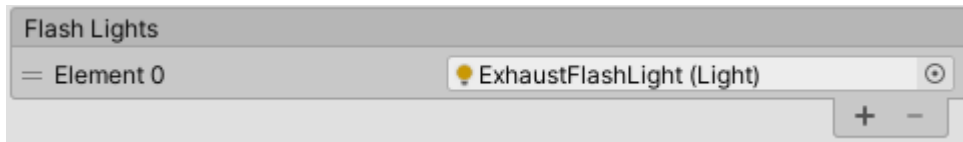


Flash Textures list.

Adding Point Lights

If the material used for exhaust flash is not emissive additional `PointLight`(s) can be added to light the surrounding area on flash.

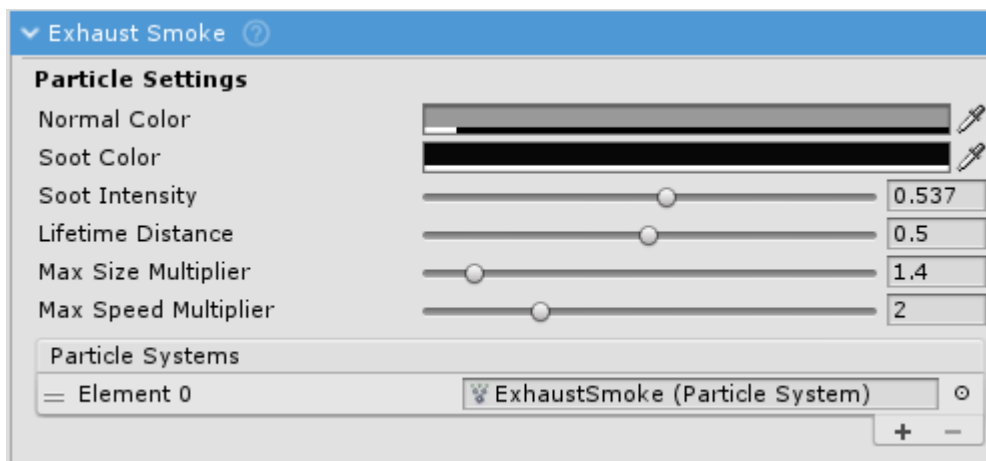
- Set up a `PointLight` and place it at the exhaust location.
- After configuring, disable the light.
- Add the light to the `Flash Lights` list.



Flash Lights list.

2020/04/08 12:36

Exhaust Smoke



ExhaustSmoke inspector.

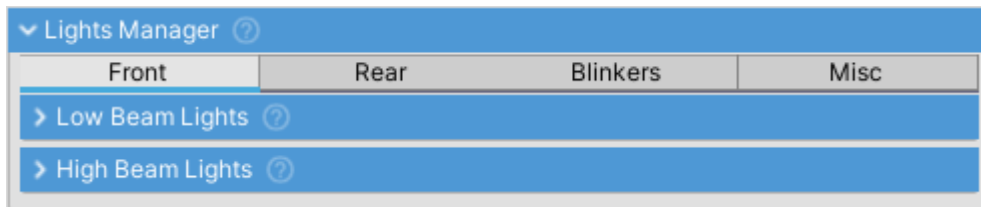
`ExhaustSmoke` controls exhaust `ParticleSystem` color, size and emission speed. It interpolates between `NormalColor` and `SootColor` based on engine state.

Setup

- Position `ParticleSystems` at vehicle exhaust position. Prefab of pre-configured `ParticleSystem` comes with the asset (*Effects > Particles > Prefabs*).
- Assign the `ParticleSystems` to the `Particle Systems` list inside `ExhaustSmoke` inspector.

2020/04/08 12:36

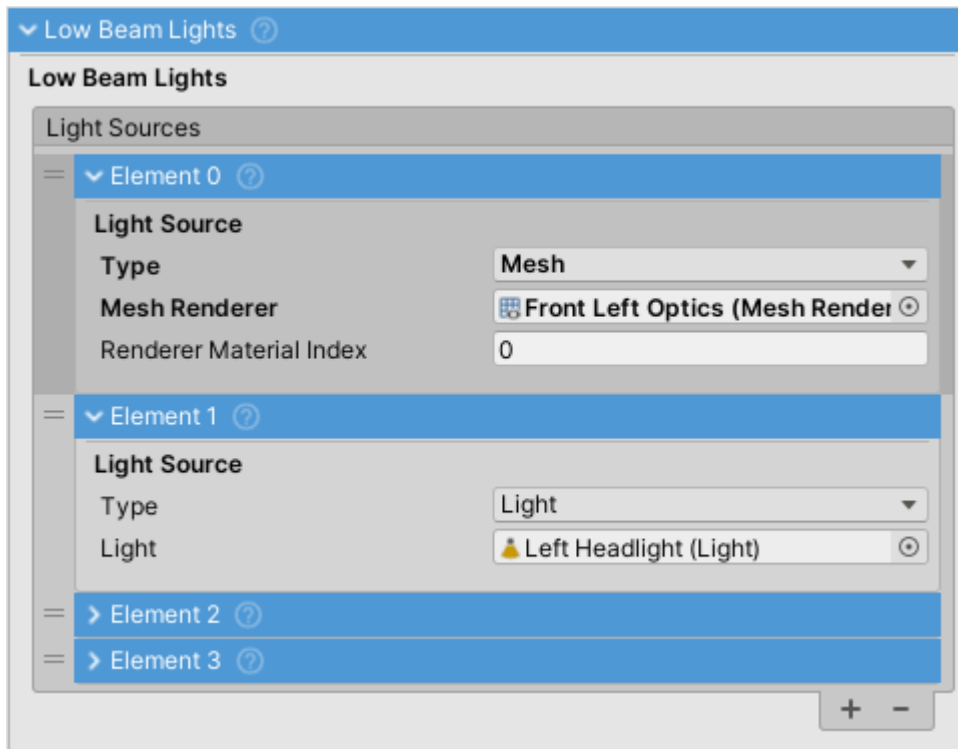
Lights Manager



LightsManager inspector.

LightsManager is tasked with turning VehicleLights on or off depending on user input.

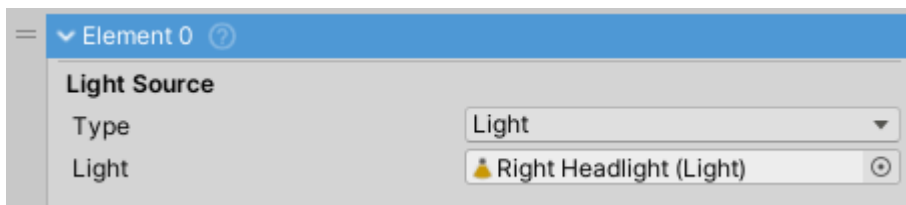
Vehicle Light



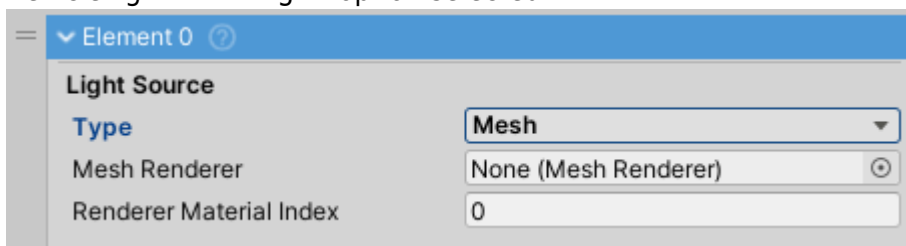
VehicleLight inspector.

VehicleLight is a collection of LightSources. When the light is turned on all LightSources are turned on and vice versa.

Light Source



VehicleLight with Light option selected.



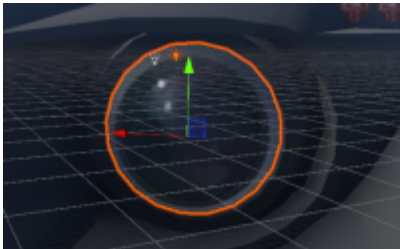
VehicleLight with Mesh option selected.

One vehicle light source. Can be a Light or emissive Mesh.

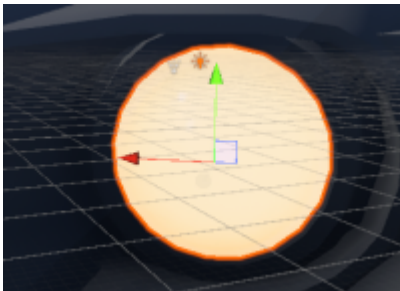
- If Light is selected as Type any Unity Light can be assigned. It will be turned on and off according to user input.
- If Mesh is selected as Type any Mesh with Standard shader can be used. It's Emission field will be toggled to imitate a working light. Since this does not emit enough light to be a headlight usually another light source is used in tandem, this one with SpotLight assigned.

For an emissive mesh light to work it needs to be a separate object. If the model does not come with lights and blinkers as separate objects these will need to be separated in 3D modelling software such as Blender (free, open source).

When using Mesh light source make sure to tick the Emission checkbox on the material. This lets Unity know that this variant of the material is in use and should be included in the build.



Mesh with Emission turned off.



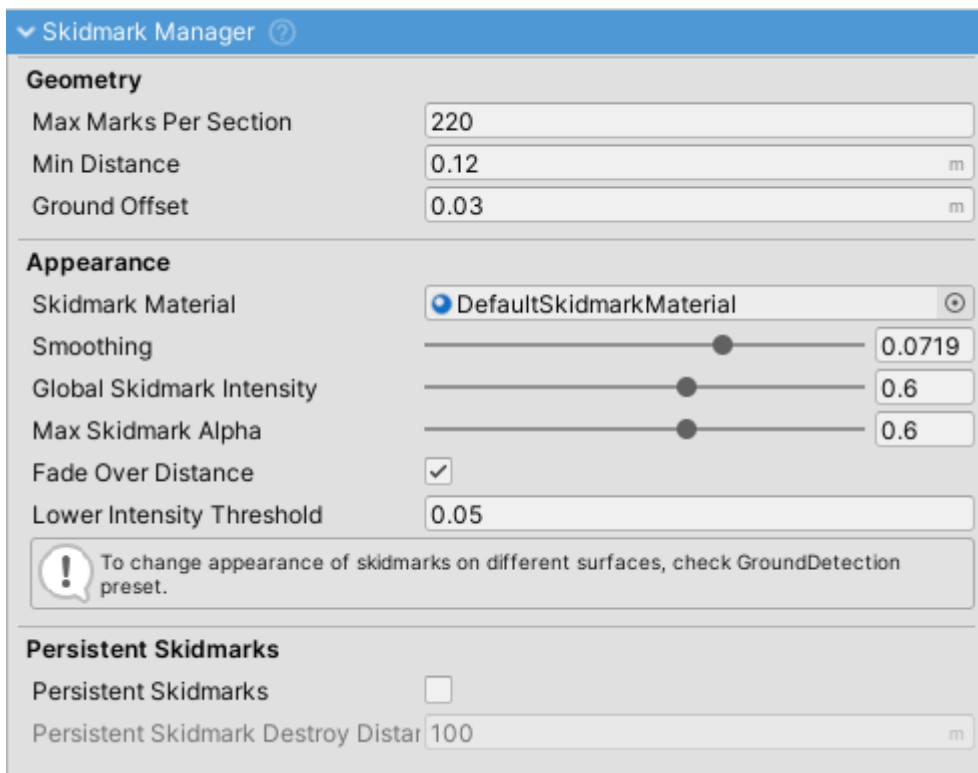
Mesh with Emission turned on.

2020/04/08 12:36

2020/04/08 12:36

2020/04/08 12:36

Skidmark Manager



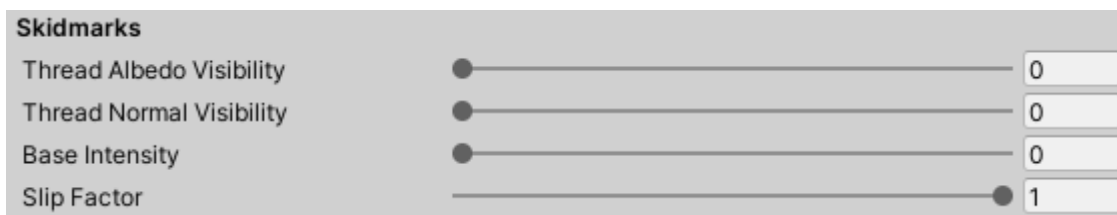
SkidmarkManager inspector.

Skidmarks are generated when wheel skids / slips over a surface.

- Skidmarks are achieved by procedurally generating a mesh. One mark consists of two triangles and the number of triangles per one section can be calculated as $\text{Max Marks Per Section} * 2$.
- Min Distance is the distance a wheel needs to travel before a new mark is created.
- Default behavior is to delete the oldest triangles as soon as number of marks reaches Max Marks Per Section - similar to the old snake game. To make this transition smooth Fade Over Distance can be enabled or Persistent Skidmarks can be used. Check the section below for more info.
- If skidmarks are not visible or clip into the terrain GroundOffset needs to be increased.
- To define when the wheel is slipping Longitudinal Slip Threshold and Lateral Slip Threshold from vehicle settings tab are used.

Surface-based Skidmarks

- Skidmarks use settings of currently active SurfacePreset to get the settings for the current surface type.



Skidmark section of AsphaltTireFrictionPreset.

- Also check [GroundDetection](#) page for more info.

Persistent Skidmarks

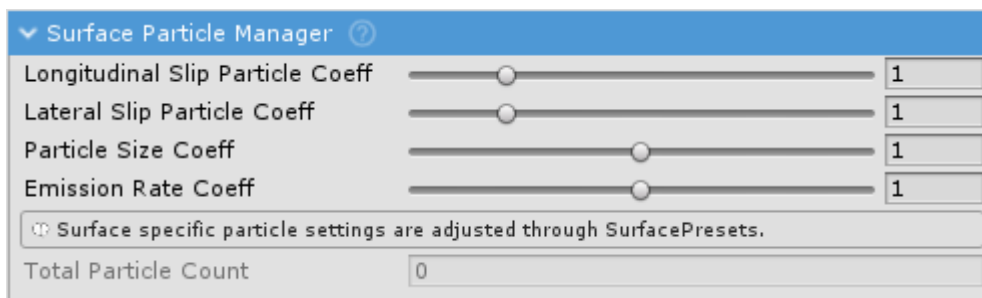


Persistent skidmarks enabled.

- Persistent skidmarks get stored into sections of Max Marks Per Section size.
- Sections do not get destroyed as long as the the player is under Persistent Skidmark Destroy Distance.
- Downside to this approach is that over time this will cause the number of triangles in the view-port to increase.

2020/04/08 12:36

Surface Particle Manager



SurfaceParticleManager inspector.

SurfaceParticleManager creates and manages particles based on current SurfacePreset settings.

- Particles are emitted on per-wheel basis. Total surface particle count is a sum of particle counts from all the wheels' ParticleSystems.
- Lateral Slip Threshold and Longitudinal Slip Threshold under *Settings* tab determine the lowest wheel slip threshold needed for wheel to be considered to be slipping. This affects particle effects.

2020/03/18 12:26

2020/04/08 12:36

Mirrors

Mirrors are set up through a combination of Cameras, RenderTextures and Materials with specific UV mapping.

Camera first renders its view to a RenderTexture which is assigned to a Material (any shader that supports albedo map will work). This Material has an UV map that corresponds to the surface of the vehicle mirror.

Possible setups are:

- Render each mirror with a separate Camera to a separate RenderTexture which is assigned to a separate Material. Best visual quality but slow.
- Using UV map that is split between the mirrors which means that each mirror gets only a part of the UV map, but this comes at a disadvantage that the interior of the vehicle can not be seen in rearview mirror since the camera needs to start rendering at the rear of the vehicle to prevent intersection with the vehicle mesh.
- Hybrid approach: one camera for rearview mirror and one camera for left and right mirrors.

This guide will demonstrate how to set up mirrors on a vehicle using Blender.

From:

<http://nwhvehiclephysics.com/> - **NWH Vehicle Physics 2 Documentation**

Permanent link:

<http://nwhvehiclephysics.com/doku.php/NWH/VehiclePhysics2/Effects>

Last update: **2020/08/24 11:10**

