GPU vegetation instancer

Overview

Thanks for downloading this asset.

The vegetation instancer is a unity project whose goal is to bring vegetation details into unity terrains. All objects are procedurally placed on the terrain without requiring any data saving. This makes the project very lightweight. Although the provided shaders work with HDRP, this project can work with URP and the default renderer as long as you provide your own shaders. To generate your shaders, see Shader Converter.

This asset consists in 3 main scripts :

- VegetationManager : this script sets up the terrain data required for full GPU positioning. There can only be one instance of it, and it is used by all the VegetationInstancers.

- VegetationInstancer : in charge of displaying large amounts of vegetation (up to millions) using any shader implementing GPU indirect instancing rendering. Everything from positioning, texture mapping, slope etc, is done on the GPU, leading to almost 0 CPU charge.

- ShaderConverter : allows to convert any shader into a GPU instancing capable shader. Attention : this only works with .shader files, not with shader graphs. However you can generate such files with a shader graph by selecting it, clicking "Copy Shader", and pasting this code into an empty .shader file.

Note that this project only spawns vegetation objects without colliders, and is reserved for small objects to populate your world.

Github page : https://github.com/Myrmecoman/Unity-GPU-Vegetation-Instancer-HDRP_URP

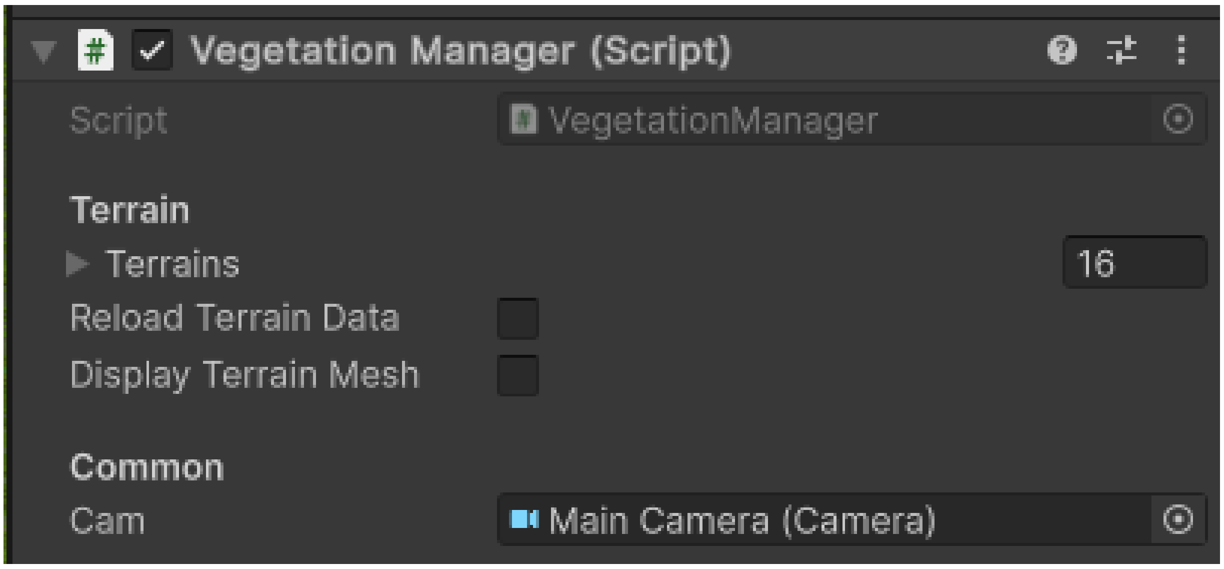
Get started

A demo scene is provided in the demo folder. It was tested on HDRP in Unity 2023.2.8. To play with the demo scene, simply import this package in an HDRP unity project and open the demo scene.

By default an example shader for HDRP is given in the VegetationInstancer/Shaders folder. GPU indirect instancing shaders can be obtained in 2 ways : either use the provided ShaderConverter script, or use Megaworld. If you use the provided script, your shader will work out of the box. If you want to use Megaworld shaders, you will need to import the GPUInstancedIndirectInclude file included with Megaworld, or entirely import Megaworld in your project. A detailed explanation to use the provided shader converter is written in the Shader Converter chapter.

This asset needs to generate terrain data to perform the procedural generation. In case you do not have a Resources folder for the asset to save the data, it will be generated automatically after a short time (generally takes 1 minute).

Vegetation manager


Vegetation Manager (Script)

Script: VegetationManager

Terrain

Terrains: 16

Reload Terrain Data: ☐

Display Terrain Mesh: ☐

Common

Cam: Main Camera (Camera)

The vegetation manager is in charge of generating and providing terrain data to the instancers. Only one should be in your scene.

For each parameter, you can overlay your mouse on it to know what it corresponds to. The parameters are such :

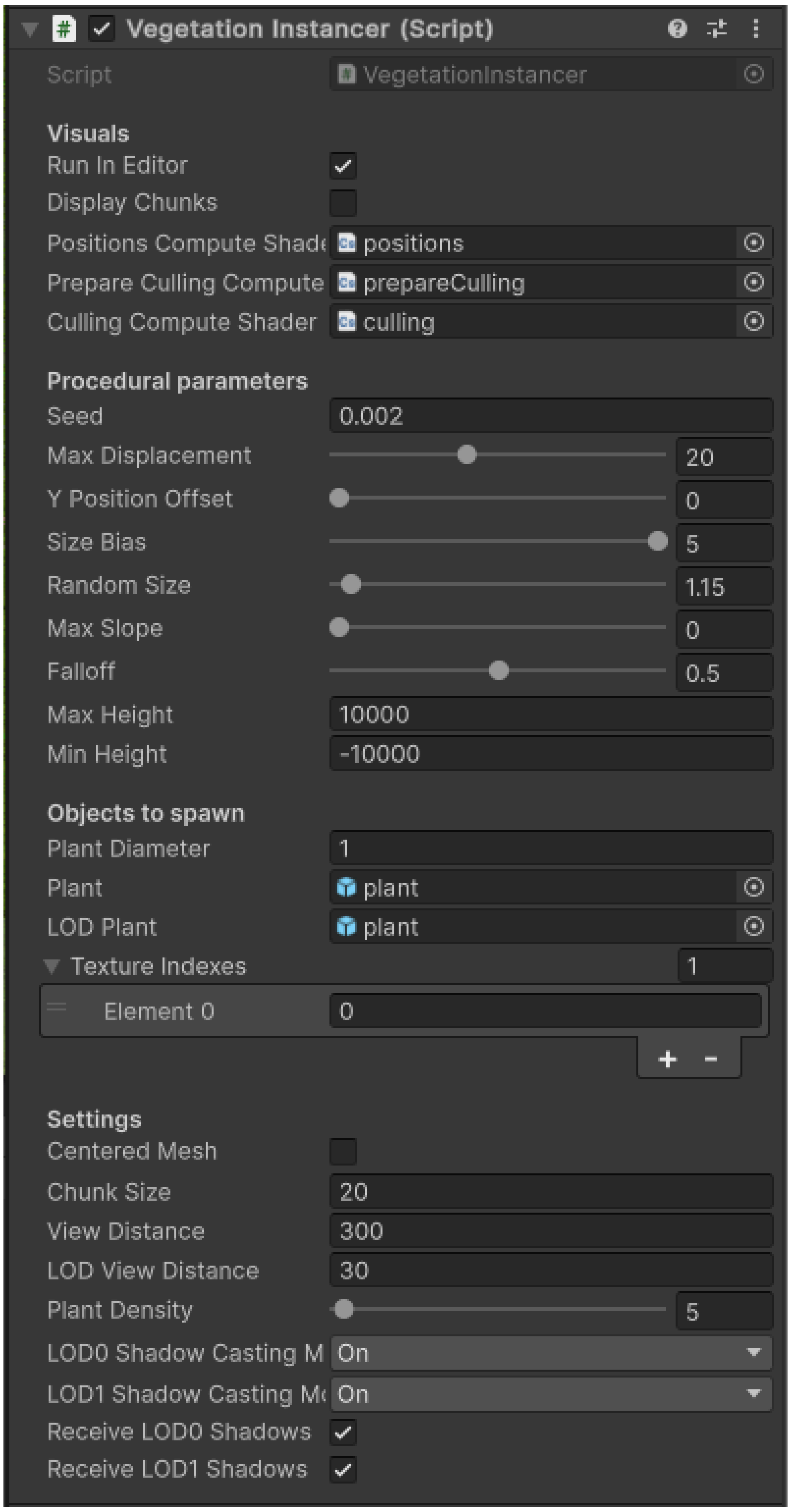
- Terrains : the terrain chunks on which you want to instantiate. They must have the same parameters (texture resolution, heightmap resolution, etc...) and be placed in a square. See the demo scene for an example of using multiple terrains.

- Reload Terrain Data : when checking this box, the manager regenerates the data necessary to instantiate on the terrains.

- Display Terrain Mesh : tick this box to visualise the generated terrain data. This is useful to ensure that it was generated correctly.

- Cam : the player camera. This camera is used to perform instantiation and frustum culling.

Vegetation instancer



Vegetation Instancer (Script)

Script	VegetationInstancer
Visuals	
Run In Editor	✓
Display Chunks	
Positions Compute Shad	positions
Prepare Culling Compute	prepareCulling
Culling Compute Shader	culling
Procedural parameters	
Seed	0.002
Max Displacement	20
Y Position Offset	0
Size Bias	5
Random Size	1.15
Max Slope	0
Falloff	0.5
Max Height	10000
Min Height	-10000
Objects to spawn	
Plant Diameter	1
Plant	plant
LOD Plant	plant
Texture Indexes	1
Element 0	0
Settings	
Centered Mesh	
Chunk Size	20
View Distance	300
LOD View Distance	30
Plant Density	5
LOD0 Shadow Casting M	On
LOD1 Shadow Casting M	On
Receive LOD0 Shadows	✓
Receive LOD1 Shadows	✓

The vegetation instancer script is in charge of spawning one type of object.
For each parameter, you can overlay your mouse on it to know what it corresponds to. The parameters are such :

- Run In Editor : runs the instancer in editor view. This way you can tweak the other parameters and see the changes in realtime.

- Display Chunks : displays the chunks of vegetation used to instantiate objects. Red is for LOD0, yellow for LOD1. Only 2 levels of LODs are supported for now. Big chunks are generally better since they are faster to compute.
- Positions compute shader : the compute shader to compute the object positions. This should always be the file at Assets/VegetationInstancer/Scripts/ComputeShaders/positions.compute
- Prepare culling compute shader : the compute shader to prepare frustum culling. This should always be the file at Assets/VegetationInstancer/Scripts/ComputeShaders/prepareCulling.compute
- Culling compute shader : the compute shader for frustum culling. This should always be the file at Assets/VegetationInstancer/Scripts/ComputeShaders/culling.compute

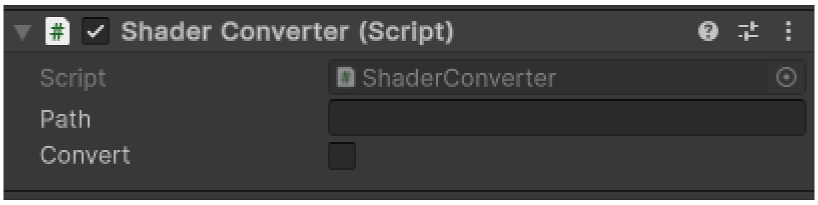
- Seed : the seed to randomly place the objects. Values between 0.1 and 0.001 are generally fine.
- Max Displacement : the maximum offset distance from the spawn point of your object. If this is 0, all objects will be instantiated in a perfect grid without any random offset.
- Y Position Offset : specify an additional height value to your objects. For example Unity's quad object is centred at the middle, so for it to be at the terrain level you have to set the offset to 0.5.
- Size Bias : multiply the size of your objects by the specified value.
- Random Size : applies random size changes from 1/size to size. A value of 2 will give you sizes from ½ to 2.
- Max Slope : specifies the maximum slope until which objects are allowed to spawn.
- Falloff : allows objects mapped on a texture to overextend on another texture.
- Max Height : the maximum allowed height to spawn the objects.
- Min Height : the minimum allowed height to spawn the objects.

- Plant diameter : The maximum plant diameter. This is used for frustum culling.
- Plant : the object to spawn.
- LOD Plant : the LOD of the object.
- Textures Indexes : the texture index on which the object is allowed to live. -1 spawns the object everywhere. Up to 4 values can be used.

- Centered Mesh : specify whether the object mesh is centered at its base or at its center.
- Chunk Size : the chunks size. The bigger the faster they are to compute, but they should not be too big else they can start to be visible.
- View Distance : the maximum distance to instantiate objects.
- LOD View Distance : the distance after which LOD objects are spawned.
- Plant Distance Int : the number of objects along the dimension of a chunk. 50 means there are 50x50 = 2500 objects in each chunk.

This project is provided with an example scene, feel free to play around with it to understand fully how to use this asset.

Shader converter



The shader converter is a script which allows you to convert a shader to make it usable by this asset. Simply provide the path of your shader in the editor and tick the "convert" checkbox. The generated shader will be placed in the VegetationInstancer/Shaders folder, and can be used directly for GPU indirect instancing by this package.

- Path : the path of the shader to convert. This path can be retrieved by right clicking on the .shader file and selecting “Copy path”
- Convert : perform the conversion of the shader.

Attention : This script can only convert .shader files. If you have a .shadergraph, you can convert it to a .shader file and use it.

Support

If this asset was of any use to you and you would like to support me, I have set up a patreon at this link :
https://patreon.com/UnityGPUProceduralVegetationInstancer?utm_medium=clipboard_copy&utm_source=copyLink&utm_campaign=creatorshare_creator&utm_content=join_link

Thanks !