DirectX 11 Grass Shader

The easy way to high end graphics in Unity

STIXGAMES.COM

If you have wishes for the shader or its documentation, feel free to contact me at:
SUPPORT@STIXGAMES.COM

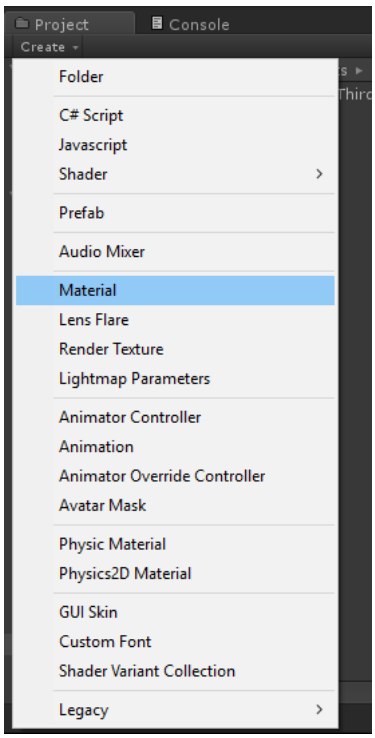
Contents

1. Tutorials

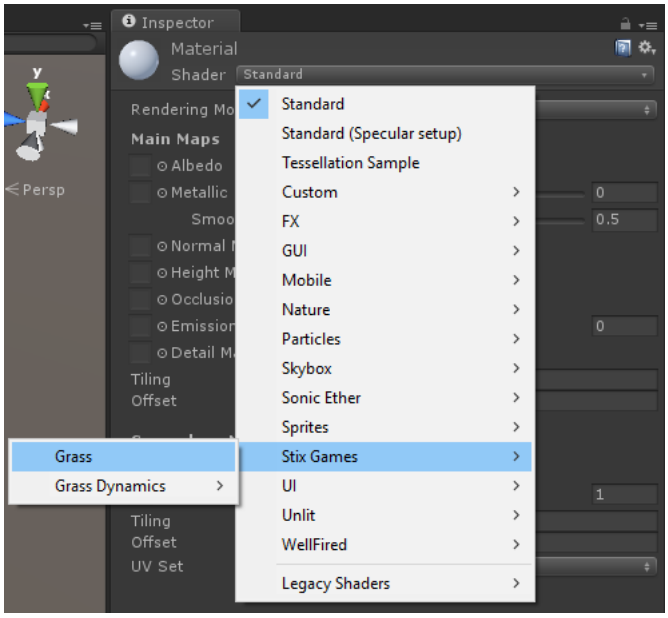
While the other sections of this manual might be more complete, some prefer a step-by-step approach, so here is the Tutorial section that was requested repeatedly. The tutorials are as non technical as possible, so if you want more technical details you can find them in the other sections.

1.1 Creating a new material with the grass shader

1. Create a new material.



2. Select the new material and change the selected shader in the inspector.

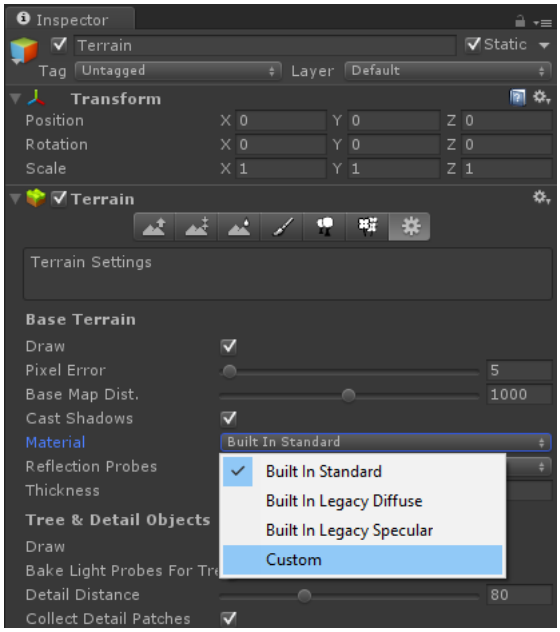


3. You can now apply the material to any object by dragging and dropping it at the target. For more basic informations concerning the interface, you could watch Unity's beginner tutorials.

1.2 Adding grass to Unity Terrain

While using the Unity Terrain can have some disadvantages, like grass popping artifacts, it is probably still the easiest way of getting a nice looking scene.

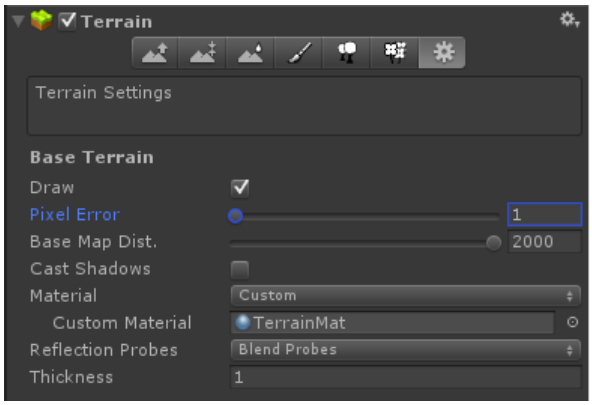
1. Create a terrain object and add it to the scene. Modify the terrain so it fits your scene, there are many tutorials for this on the Internet.
2. When you have done all changes to the terrain you wanted, duplicate it by pressing `Ctrl-D`.
3. Select the duplicate of the terrain and open the Terrain Settings tab in the Terrain component. (Can be seen in the next screenshot)
4. Set the material option from `Built In Standard` to `Custom`.



5. Drag and drop your grass material into the new Custom Material field.

1.3 Reduce grass popping when using unity terrain

1. Select your terrain and open the Terrain Settings tab.
2. Set the Pixel Error variable to 1.



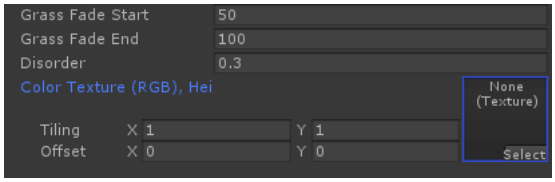
3. You should have a lot less popping now, but may have increased performance cost. There can still be some popping, which unfortunately cannot be fixed when using Unity Terrain.

1.4 Add textures to the grass material

Whether you use Unity terrain or a custom mesh, you will probably want something different than a uniform grass plane. These short steps will show you how to make your grass area more interesting by adding special textures.

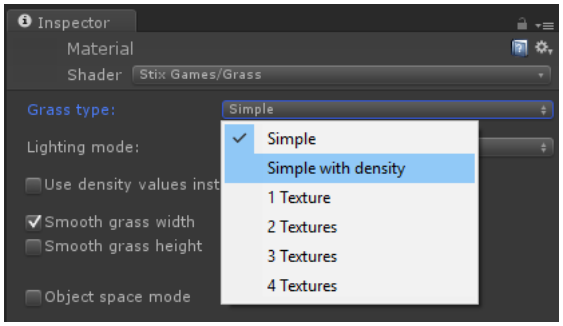
1.4.1 Color and Height texture

1. Create a Color/Height texture in your painting software of choice. The Red, Green and Blue channels define the color of the grass. The alpha channel sets the height of the grass, so that 1 is maximally high and 0 is maximally low.
2. Set the texture to the Color/Height texture field in the material inspector.

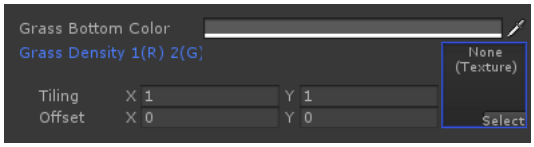


1.4.2 Density texture

1. Create a Density texture in your painting software of choice. If you have only one type of grass, the texture can be black and white, where white is full density and black represents no grass at all. If you have multiple grass types, each color channel represents one type of grass. Red, Green, Blue and Alpha are Texture 1, Texture 2, Texture 3 and Texture 4 respectively.
2. The simple grass type doesn't support densities, so switch to a different grass type.



3. Add your density texture to the Grass Density field.



1.5 Setting up the RenderTexture Displacement

1.5.1 Using prefebs

1. Place the *DisplacementCamera* prefab in your scene.
2. Create a layer for the displacement. Nothing should be on this layer, except displacement renderers. Set the culling mask of the displacement camera exclusively to this layer.
3. Set the follow variable of the *Camera Follow* script to the main camera of the scene.
4. As soon as you add displacement renderers to the scene, it should start working immediately now. Make sure that they are on the right layer.

1.5.2 Manual camera setup

1. Add a second camera to the scene.
2. Set the rotation of the camera to $(90,0,0)$.
3. Create a layer for the displacement. Nothing should be on this layer, except displacement renderers. Set the culling mask of the camera exclusively to this layer.
4. Set the camera to Orthographic Projection mode and set a size of 100. You can change this size depending on your scene.
5. Add the `StixGames/Render Texture Displacement` Component to the camera.
6. Add the `StixGames/Camera Follow` Component to the camera and set the follow variable to the scenes main camera (must be different from the displacement camera).
7. Create a new *RenderTexture*. Leave *Anti-Aliasing* at *None*. You can change the *Size* for your needs.
8. Set the *Color Format* variable of the *RenderTexture* to `ARGB Float`.
9. Add the *RenderTexture* to the displacement cameras *Target Texture* field.
10. As soon as you add displacement renderers to the scene, it should start working immediately now. Make sure that they are on the right layer.

1.5.3 Using the displacement trail renderer

If you want to create trails behind characters, vehicles or similar, follow these steps:

1. Add a `StixGames/Displacement Trail Renderer` to your object.
2. Create a material with the *Normalmap Renderer* shader.
3. Add a special normal map to the material. A example for this is included in the package. Make sure that your texture is set to normal map in the importer.
4. Set the Layer variable to your displacement layer.

5. Now you can customize the settings of the renderer to achieve the results you wish for.

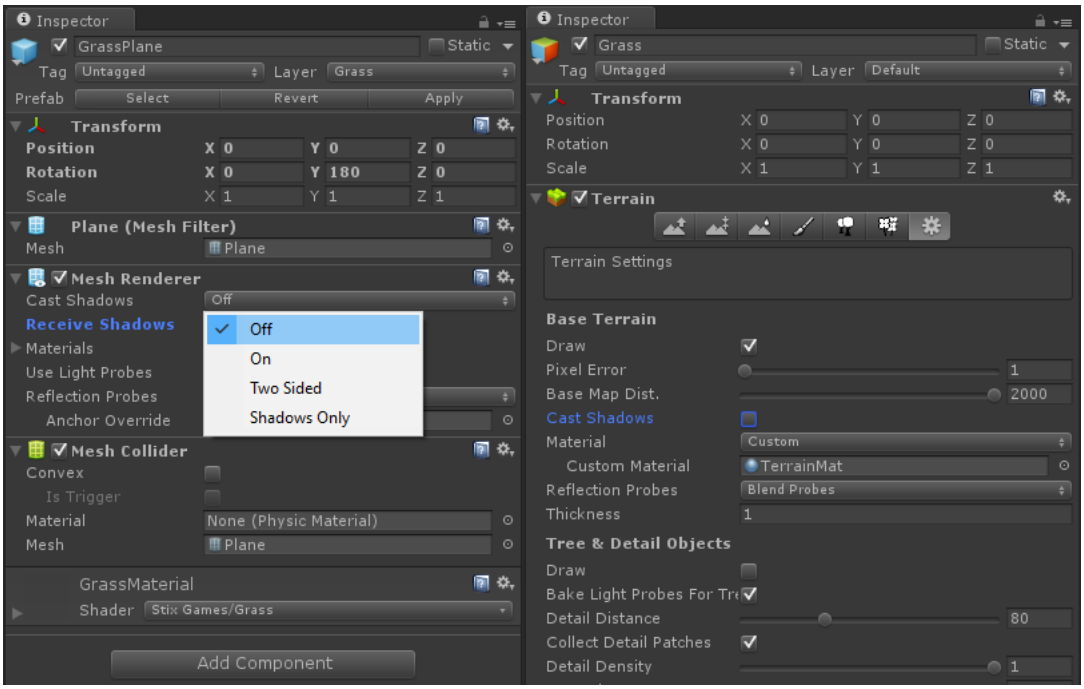
1.5.4 Using default renderers

You can either use a mesh directly, with the *Mesh Normal Renderer* shader, or use a normal map. To use a normal map follow these steps:

1. Create a quad object.
2. Create a material with the *Normalmap Renderer* shader.
3. Add a normalmap to the material and apply the material to the quad.
4. Set the layer of the object to the displacement layer. Now the grass should be displaced by the normal map.

1.6 Reducing performance cost by disabling shadow casting

Casting shadows is rather expensive with the grass shader, but you can easily disable it. The left image is when using a custom mesh, the right is a terrain.



1.7 Using the Curved World asset by VacuumShaders

To enable support for Curved World[1] just follow these steps:

1. Open the *GrassConfig.cginc* file in the main shader folder.
2. Uncomment the line `//#define GRASS_CURVED_WORLD` and the line `//#include "..."` immediately after it.
3. Now the shader will recompile, which can take some time depending on your system. After that you are good to go.

[1] `https://www.assetstore.unity3d.com/en/#!/content/26165`

2. Shader Options

2.1 Shader Variants

With shader variants it is possible to have one large shader that is very customizable, without sacrificing any performance. The down-side is, that changing one of these options will result in having to recompile the shader. More info about shader variant can be found in the Unity documentation: `http://docs.unity3d.com/Manual/SL-MultipleProgramVariants.html`

If you are using more than one very complex shader, Unity might encounter errors because there are too many shader keywords. Read about how to solve this problem in 7.4.

2.1.1 Grass type

The fundamental options of the shader. Changing them can change the performance impact significantly. In short: "Simple" is the cheapest and "4 Textures" is the most expensive shader version.

Simple grass

In this mode the shader generates simple, textureless blades of grass. This mode is ideal for large areas with high view distance.

Simple grass with density

Like the simple grass, but with the option to use densities. Do NOT use this with *density values* (2.1.3), you can easily use the *Density Falloff* (2.2.1) and *Max Density* (2.2.2) values, that improve the performance. Only use this when you want to use a texture for specifying the density per area. If you do not need that, use simple grass instead.

1 - 4 textures

This mode allows you to have up to 4 different textures. You can set their density either by using sliders, textures, or vertex colors, see 2.1.3. Each grass texture will have their own set of controls, more info about that in 2.3.

Less textures will result in better performance.

2.1.2 Lighting mode

Unlit

If you are going for a very comic-like look, or just don't want to waste unnecessary performance you can completely ignore the lighting and just output the unaffected color of the grass.

Inverted Specular PBR

When using Unity's PBR rendering the specular highlights are visible in the opposite direction of the sun. While this may be realistic and can be observed in nature, it doesn't look as good as the more intuitive version, where the specular light is at the same side as the sun.

The inverted specular option switches the direction where the specular highlights can be seen. For this, a modified version of Unity's PBR lighting is used, which could result in slightly higher performance costs.

Default PBR

This lighting mode uses the default Unity PBR.

2.1.3 Density mode

Texture density

In this mode the shader uses a grass density lookup texture. This is perfect if you want to have control over small details in the grass. This mode requires UV coordinates on your mesh.

Vertex density

Use the vertex color to set grass densities. This mode is perfect for procedurally generated content, or if you don't need small details and want to save performance.

Value Based density

If this option is checked, you can set the grass density directly. Use this option if you want to have a large field with uniform densities. As the shader uses one texture lookup less in this mode, it slightly increases performance.

2.1.4 Smoothing options

With these option you can enable or disable smoothing between tessellation (density) levels.

Smooth grass width

Smooths between tessellation levels by changing the width of blades of grass.

Smooth grass height

Smooths between tessellation levels by changing the height of blades of grass. This requires some additional calculations to make the height change less obvious, so it can increase the performance cost.

2.1.5 Improve viewing grass from above

This option can slightly improve the visuals when your camera is looking down at the grass, for example when flying over it. It may cause artifacts in other situations.

There is a known bug with this setting that causes grass to get clipped at certain camera angles.

2.1.6 Object space mode

If you want moving platforms, this is the right setting for you. It may have a slight performance cost and could lead to artifacts in some situations.

Follow surface normals

If you want to have a spherical world, you can use this setting. Wind calculations and grass displacement will not work as expected here, you will have to experiment with custom solutions if you want these advanced features in this mode.

2.2 General grass options

These values control properties that affect the entire material. Properties of individual grass types are in 2.3.

2.2.1 Density Falloff

Density Falloff is the most important setting for performance. The higher the slider value is, the more the grass density will reduce with distance. Try to set the value as high as possible, without losing visual quality.

If the density is very high, even with high *Density Falloff* values, your geometry is probably too dense. Read more about how geometry affects the grass density in 6.2.

2.2.2 Max Density

Besides *Density Falloff* this is the most important setting for performance. The lower *Max Density* is, the lower is the maximum amount of blades of grass per triangle. By using a low *Max Density* and low *Density Falloff*, you can have very large fields of grass, even on lower end systems.

If the density is too high, even at low *Max Density* values, your geometry is probably too dense. Read more about how geometry affects the grass density in 6.2.

2.2.3 Level of Detail (LOD)

The level of detail settings change the amount of triangles the blades of grass consist of. The lower the amount of triangles is, the lower the performance cost, but with too little LOD, the grass will look very spiky.

LOD Start

The distance from the camera at which the LOD will start to decrease.

LOD End

The distance from the camera at which the LOD will be at the minimum.

Max LOD

Changes the maximum LOD.

> If you wish to optimize the *Max LOD* setting, you can change the maximum vertex count in the shader itself, by opening the *GrassConfig.cginc* and changing the *MAX_VERTEX_COUNT* define. This will improve the performance slightly, compared to using the Max LOD setting.

2.2.4 Grass Fading

The grass fading options fade out grass that is far away from the camera. It can be used to hide grass with too little density in the distance. This does not improve performance significantly.

Grass Fade Start

The distance from the camera at which grass will start to fade.

Grass Fade End

The distance from the camera at which the grass will be completely faded out.

2.2.5 Disorder

Changes the base disorder of the grass which is independent from wind. Lower values give you a very smooth field of grass, higher values a chaotic, natural look.

2.2.6 Color Texture

This texture map modifies the color and height of the grass. Both will be multiplied with the base values set in the grass options, 2.3.

2.2.7 Displacement Texture

This texture displaces the grass. Each pixel represents a vector. Setting this manually can be used to create crop circles or similar effects. Its real advantage comes from modifying this texture in real time: You can let the grass react dynamically to external forces, like vehicles driving trough the grass. For more information about the dynamic reactions, read chapter 4.

2.2.8 Grass Bottom Color

Changes the grass color closer to the ground. This value is multiplied with the base color of the grass, so it can only darken the grass. Making the bottom of the grass darker than the top will result in a more three-dimensional look than uniform colors.

2.2.9 Burn Color

The color grass will be multiplied with when being burned. For more info about burning grass, see 3.2.3.

2.2.10 Grass density values / Grass density texture

There are three ways to define grass density. As values, that are applied uniformly to the whole geometry, as a vertex color they can be set directly in your modeling software and are easy to procedurally generate, or as texture, which enables different densities depending on the UV coordinates. While the texture gives you more control, the density values and vertex color have a slightly improved performance.

Density values

The X, Y, Z, and W values represent grass types 1, 2, 3, and 4 respectively.

Density texture / Vertex color

The R, G, B, and A channels represent grass types 1, 2, 3, and 4 respectively.

2.2.11 Wind

These options control the effect of wind on the grass. By adjusting these you can achieve very different effects, from slight breezes to heavy storms. By decreasing the speed and increasing the strength you can even create something that looks like underwater plants.

Because the wind is calculated independently each frame, these values can not be changed in real time. It should be possible to calculate the wind outside of the shader and apply them in the displacement texture, but would have a big performance impact. Read more about the displacement texture in chapter 4.

Wave strength
Changes the strength of large scale wind waves.

Wave speed
Changes the speed of large scale wind waves.

Ripple strength
Changes the strength of small scale wind ripples.

Ripple speed
Changes the speed of small scale wind ripples.

Wind direction
Changes the general direction in which the wind will blow.

2.3 Grass visuals

These properties set the look of each type of blades of grass. There can be up to 4 sets of these options, depending on the grass type and texture count, 2.1.1. The different sets are identical in functionality.

2.3.1 Color
The base color of each blade of grass. It will be randomly blended with the secondary color.

2.3.2 Secondary Color
The secondary color of each blade of grass. It will be randomly blended with the base color.

2.3.3 Specular Color
Sets the specular color, or color of the reflecting light, identical to the Unity Standard shader.

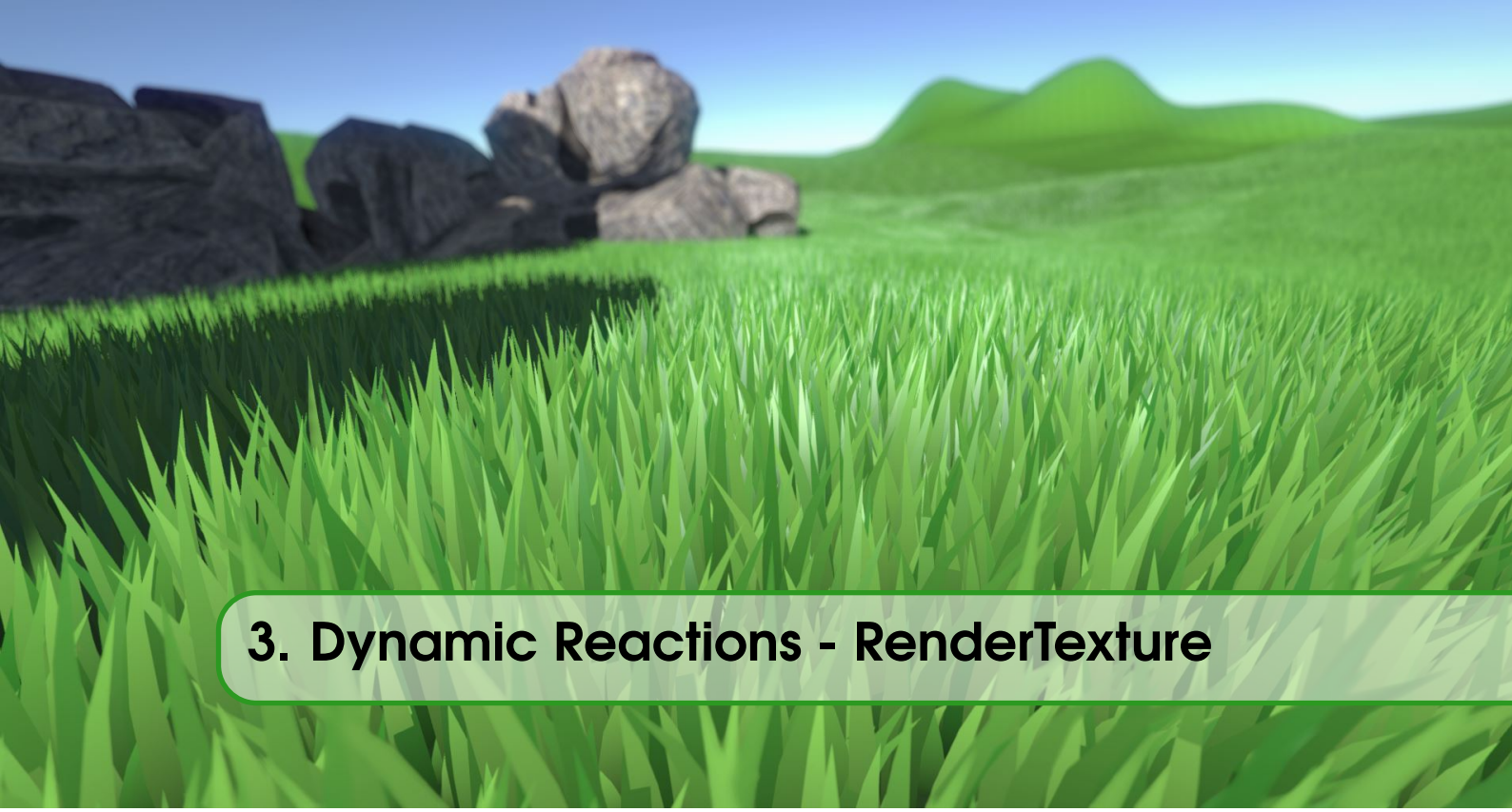
2.3.4 Smoothness
Sets the smoothness of the grass type, identical to the Unity Standard shader.

2.3.5 Softness
Sets the softness of grass. Wind will have more effect on smooth types of grass.

2.3.6 Width
The width of each blade of grass.

2.3.7 Min Height
The maximum height of grass.

2.3.8 Max Height
The minimum height of grass.



3. Dynamic Reactions - RenderTexture

With this simplified system for dynamic reactions it is easily possible to add displace grass around character, add trails behind them and similar.

The displacement works by rendering the scene with a secondary camera from above. Every object that is supposed to displace the grass has a special renderer attached, which renders a normal map, which in turn is passed to the grass shader. The grass will then be displaced by the normal map. You don't have to add the RenderTexture manually to each grass object, it will be activated for every object with the grass shader in the scene. Because the displacement area is handled by a camera that can be moved around, this approach can be used for very large areas or terrains.

For ease of use, a prefab with a complete displacement camera is provided. If you wish to know how to create one manually, you can read the tutorial section, in 1.5.

3.1 Render Texture Displacement

This script manages handing the RenderTexture to the grass shader.

3.1.1 Border Smoothing Area

The camera only renders a specific area. Instead of just clipping everything outside of this area, this property can be used to smooth out the border. The smoothing area is the distance to the border in Unity Unity / Meters at which the smoothing will start.

3.2 Special Displacement Shaders

3.2.1 Normalmap Renderer

This shader uses a normal map to displace the grass. The blades of grass will always look in the direction of the normal.

3.2.2 Mesh Normal Renderer

This shader just renders the world normals of the selected mesh. The blades of grass will always look in the direction of the normal.

3.2.3 Common Parameters

Influence Strength

The higher the influence strength is, the more this displacer will influence the grass displacement in comparison to other displacers. For example, if you have a object that creates wind effects, you want it to have a smaller influence than the character walking through the grass, so you should set the influences accordingly.

Burn Map / Burn Strength

Both displacement shaders support the use of burn maps. The burn map is a texture that defines the pattern with which the grass will be burned. The only channel currently used is the red channel. If the red channel is full the grass will be fully burned away, if there is no red in the color, no grass will be burned.

The Burn strength is a multiplier that can be used to slowly burn or regrow the grass.

Of course, the burn value can be used for other things as well. If you change the burn color in the grass settings to white, you could use it to remove grass under a building or let grass grow on a field.

3.3 Displacement Trail Renderer

As the default Unity Trail Renderer does not support normal maps, this package includes a simple replacement component. It was created only for the displacement, for everything else, there are probably better alternatives available.

3.3.1 Min Vertex Distance

The minimal distance between path points, in Unity Units / Meters. In short: The smaller this value is, the denser and more accurate the trail will be rendered.

3.3.2 Lifetime

The lifetime of the trail. For example if you set this to 20 seconds, a newly created part of the trail will completely vanish after 20 seconds.

3.3.3 Width

The width of the trail at a given part of the lifetime. The x-axis is the percentage of the lifetime, so it will always be between 0 and 1. The y-axis is the width in Unity Units / Meters.

3.3.4 Strength

The displacement strength of the trail at a given part of the lifetime. The x-axis is the percentage of the lifetime, so it will always be between 0 and 1. The y-axis is the strength in percent, so it should only be between 0 and 1.

3.3.5 Material

The material the trail will be rendered with. It should have the `Stix Games/GrassDynamics/Normalmap Renderer` shader, or you probably won't have sensible results.

3.3.6 Layer

The layer the trail will be rendered on. You should keep this layer exclusively for the displacement.

3.4 Camera Follow

While it's a rather general and simple script, it is necessary for the RenderTexture Displacment to work. It will let the displacement camera follow the main camera at a specific height, so that the main camera is always in the middle of the displacement area and will, if possible, never see the border.

3.4.1 Follow

The object the camera will follow.

3.4.2 Follow Height

The height at which the camera should follow.

4. Dynamic Reactions - Texture Manipulation

This approach was the first implemented for the Stix Games DirectX 11 Grass Shader, is works by directly manipulating a texture, so it is flexible and changes are preserved, but it is not really useful for very large areas of grass and is rather hard to program for. If you don't explicitly need some parts of this approach, you should use the new improved system instead, which you can find in Chapter 3.

4.1 Grass Texture Adder

The Grass Texture Adder class can be used to add empty and editable color and displacement textures to a material. You could add these manually by creating a texture and activating "Read/Write Enabled" in the "Advanced" texture type.

4.2 Circular Displacer

The Circular Displacer class can be used to displace a circular area around an object. In order for this script to work a few conditions have to be met:

- The object with the grass material has to be on a layer you have selected in the layer mask.
- The object with the grass material has to have a collider.
- The object with the grass material must have a UV map. The displacement is based on a texture, so without one it will not work.
- The displacing object does not need a collider. The displacement is done by raycasting. Of course you can add one if you need it otherwise, just don't put the character collider on the grass layer.
- The grass material has to have a editable displacement texture. It can be added with the Grass Texture Adder class.
- The texture must be large enough for the area. Keep in mind that very large textures will significantly decrease performance. If you want to have displacement in a very large area, you could split it into smaller parts and use several small textures instead of one large one.

4.2.1 Pressure Threshold

The pressure threshold prevents the displacement of grass that has been displaced too far to the ground.

4.2.2 Max Angle

The maximum angle of the displacement. With this you can only displace grass that is in the forward direction of your character.

4.2.3 Radius

The radius of the area that will be changed. This value is in meters / Unity units.

4.2.4 Directional Displacement

If set to true, the displacement will always be in the direction of travel. If set to false, the grass will be displaced from the center.

4.2.5 Displacement Strength

The strength of the displacement, meaning the amount the grass will be pushed to the side.

4.2.6 Displacement Falloff

With a low falloff, the displacement will be equal in the whole radius. With a high falloff the displacement will be high in the middle and low at the border of the radius.

4.2.7 Pressure Strength

The strength of the pressure, meaning the amount the grass will be pushed to the floor.

4.2.8 Pressure Falloff

With a low falloff, the pressure will be equal in the whole radius. With a high falloff the pressure will be high in the middle and low at the border of the radius.

4.2.9 Offset

The offset of the origin of the ray. When your character's origin-point is at its feet, there might be problems because the ray is starting below the collider of the grass, which can be prevented by offsetting the rays origin.

4.2.10 Ray Direction

The direction the ray will travel. You could, for example, put a displacer on the top of a vehicle and set the ray direction to up. That way, the grass will be displaced when the vehicle topples over.

4.2.11 Max Distance

The maximal distance the ray will travel. With this you can stop displacing the grass while a character is jumping.

4.2.12 Grass Layer

The layer where your grass will be located.

4.3 Grass Manipulation Utility

The Grass Manipulation Utility class contains some functions that are useful for creating your own grass displacement class. You can see how they are used in the Circular Displacer class.

5. Additional scripts

5.1 Readjust Bounding Box

This script can be used to prevent visual errors, where the grass vanishes at certain camera angles and small light sources become invisible. The script simply tells Unity to expand the bounding box of the grass object, in other words it will be rendered even when the source object is slightly off-screen.

5.1.1 Max Grass Height

The maximum grass height the grass material has. This does not have to be exactly like in the shader, it has to be larger or equal.

6. Improving Visual Quality

6.1 Density

By using a low *Max Density* and low *Density Falloff*, you can have very large fields of grass, even on lower end systems.

6.2 Base geometry

When using triangles of different sizes, the density of the grass will be irregular. As long as you aren't using this to optimize density (More about that in 7.2) it is probably an unwanted effect, so keep the following rule in mind:

Try to keep triangle shape and area as regular as possible. The easiest way of doing this is using a perfect grid, like most terrain systems do.

> Keep in mind that actually using a terrain system might have unwanted effects. Most of them have their own LOD system, which can lead to popping of grass. Ideally you would not want to change the base geometry of visible grass at all.

6.3 Fine tuning wind

There are no real rules for setting the wind and you will have to find out the details through experimentation, but there might still be some guidelines to get a nicer look:

- **Making the wind too slow will make the scene look like it's under water.** You will never get completely rid of this, but that is just how it looks like in real life, because both wind and water are working under very similar fluid dynamics.
- Try balancing the wind ripples to the weather you are trying to create. Wind waves alone won't be able to create a convincing result.
- If you have balanced both wind waves and ripples, but the grass is still looking too regular, try changing the disorder parameter, more in 2.2.5.

6.4 Grass vanishing at certain camera angles / Small Point lights are invisible

This problem can be easily fixed by adding the `ReadjustBoundingBox` script to your grass object. Read more about the script in 5.1.

7. Optimization

The main bottlenecks of the shader are the generation of the blades of grass as well as the fill rate, in most cases only the first one has a real impact. Some GPUs are better at generating geometry, so keep in mind that performance may vary strongly.

7.1 Reducing render passes

If Unity has to render the scene less often, the performance will increase significantly. The easiest way to do this is disabling shadow casting on the grass object.

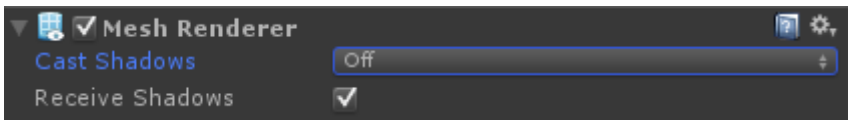


Figure 7.1: Disabling shadow casting increases performance significantly.

If you have more than one camera you might consider not rendering the grass there. Remember that the size on the screen is actually less important than the amount of grass that's generated.

7.2 Density

In most scenarios this will be the most important factor for performance. Besides changing the base geometry, more about that in 6.2, the best way to reduce grass density, is to have a high *Density Falloff* and low *Max Density*. Every other optimization is minor in comparison to these two values.

> **Important:**
> Be aware that neither the density sliders, nor density textures improve the performance substantially. They only make blades of grass invisible, but they still have to be generated.
>
> Use *Max Density* and *Density Falloff* instead.

7.2.1 Base Geometry

The base geometry, the model the grass material is applied to, is directly related to the density of the grass. This is because of the technique that is used for generating the grass, tessellation. It cuts the given triangles into smaller parts, which will then be used to generate the grass itself. If the base geometry is very small, this will result in a very high density of grass. **You can use the polygon count of the mesh to fine-tune the density of the grass.**

7.3 Fill Rate

Fill rate describes the amount of pixels that have to be filled by the shader. It can be very high if large parts of the screen are drawn to or more importantly when a pixel is drawn to many times, overwriting the same information many repeatedly in the process.

While the fill rate is not the main bottle neck on most hardware, it can still be optimized.

- **Reduce the amount of blades of grass.** This one might be obvious, but it's the best way of reducing performance needs. It not only reduces the fillrate, but also the geometry that has to be generated.
- **Reduce the size of blades of grass.** If you have very large blades of grass, the fill rate will have a high performance impact. Just reduce their height and width to gain performance.
- **Don't waste texture space.** If your textures are mostly transparent, you are wasting system resources. The performance is dependent on the actual triangle size, not on the parts that are visible on the screen.

Figure 7.2: Always try to use the full area, like shown on the right.

7.4 Reducing shader keywords

With shader keywords it's possible to switch between multiple variants of a single, large shader. More info in the Unity documentation:
`http://docs.unity3d.com/Manual/SL-MultipleProgramVariants.html`.

The Stix Games DirectX 11 Grass Shader uses many keywords to make it as customizable as possible. While this does not have a performance impact, it can lead to a situation where there are not enough free keywords. Unity only supports 128 different shader keywords (as of version 5.0). This could happen when using multiple very complex shaders.

To remove shader keywords, open the `Grass.shader` file. Search for the shader-feature blocks.

```
#pragma shader_feature SIMPLE_GRASS SIMPLE_GRASS_DENSITY THREE_GRASS_TYPES FOUR_GRASS_TYPES
#pragma shader_feature __ PBR_GRASS_LIGHTING
#pragma shader_feature __ UNIFORM_DENSITY
#pragma shader_feature __ NO_TESSELLATION_SMOOTHING
```

There are multiple blocks like this in the shader file, they have to stay identical, or the shader will no longer work as intended.
If you want to remove a keyword, replace the `shader_feature` line like this:

```
#define SIMPLE_GRASS
#define PBR_GRASS_LIGHTING
#define UNIFORM_DENSITY
//NO_TESSELLATION_SMOOTHING is not defined
```

You can replace one, or multiple lines, but they have to be identical in every `shader_feature` block in the shader. "`__`" is a empty keyword, which can be used to switch a single feature on or off. If you want to disable a keyword using it, just don't create a `#define` line, like shown with the `NO_TESSELLATION_SMOOTHING` keyword above.