

Table of contents

- [1. What is VertexDirt?](#)
 - [What is ambient occlusion?](#)
 - [What are vertex colors?](#)
 - [Shader requirements](#)
 - [Supported Unity versions](#)
- [2. Usage quickstart](#)
 - [Sample scenes](#)
- [3. VertexDirt settings explained](#)
- [4. Modeling guidelines](#)
- [5. Save baked meshes](#)

1. What is VertexDirt?

VertexDirt is a Unity plugin for baking ambient occlusion to vertex colors. It not uses lightmaps, no need of unwrapped uv coordinates. Good for mobile game development. VertexDirt is an offline technique, which means it is not updating the results realtime, you have to 'bake' it instead in the Unity Editor. The way of baking is similar to the Unity Lihtmapping baking.

What is ambient occlusion?

Ambient occlusion is a general term referring to a method to approximate how much indirect lighting a part of a surface recieves. Ambient Occlusion is a non realistic technique, but adds realism to your scenes. This is a commonly used technique in games. The idea behind ambient occlusion is that in nature, bouncing light reaches occluded areas (e.g. the ground under a car) less often compared to open areas, which lead to dark shadows under the car even in overcast day.

What are vertex colors?

Vertex colors are part of the data structure of a mesh, just like the vertex positions or the vertex normals. This array of colors are contains only white colors by default. It can be used for different purposes. VertexDirt uses vertex colors to store the baked AO. Storing and using colors this way is fast and have a small memory and storage footprint.

Shader requirements

In Unity, vertex colors are only visible if you using shaders which supports them. Although the Default and Standard shaders doesn't supports vertex colors, it is really simple to add this feature to them. You can find vertex color shader in the Unity Forums ans Asset Store too. VertexDirt package also bundled with some sample shaders to start with.

Supported Unity versions

VertexDirt is compatible with both versions of Unity 4 and all versions of Unity 5.

2. Usage quickstart

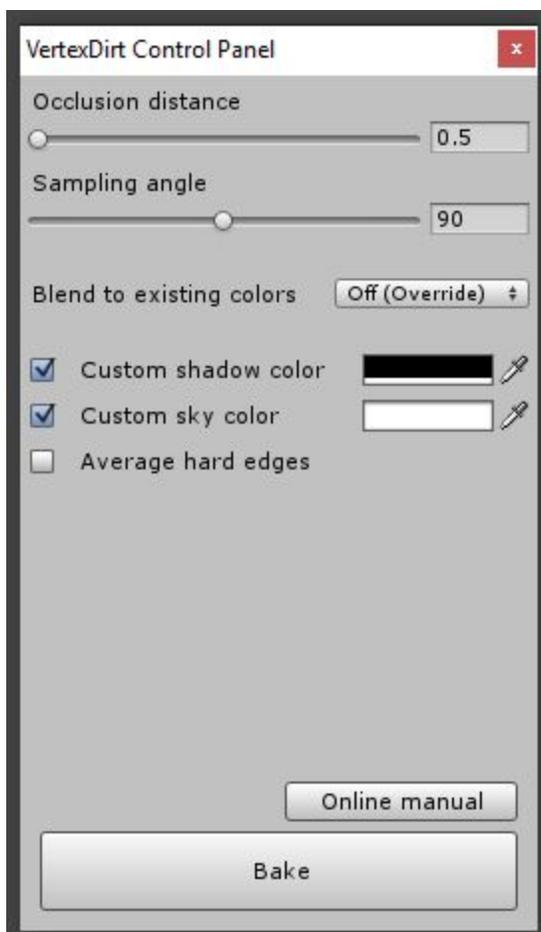
VertexDirt is a very simple, easy-to-use plugin. No need of colliders or special setup. To bake with VertexDirt, just follow these steps:

1. Select objects you want to bake.
2. Open Tools/Zoloigo/VertexDirt bake window
3. Set the parameters as you wish.
4. Hit the bake button and wait.
5. If you are not satisfied with the results, just tweak the settings and repeat baking.

Sample scenes

You can find several sample scenes bundled with VertexDirt in the \Plugins\VertexDirt\Samples' folder. These scenes helps you to learn using VertexDirt.

3. VertexDirt settings explained



Occlusion distance

The distance in units after the geometry clipped.

Sampling angle

Narrow angles gives strong contrast and sharp gradients. The wider angles gives smoother, faded results.

Edge smooth enabled

Edge smooth averages the normals of vertices in the same position during baking, giving smoother results.

Blend to existing colors

If the meshes already have vertex colors, you can set this property to multiply for blending the existing colors with the baked colors.

Custom shadow color

The color of the completely occluded areas. Uncheck this property if you want default black shadow color.

Sky color

The color of the unoccluded areas. Uncheck this property if you want default white sky color.

4. Modeling guidelines

This guide specifies some modeling tricks to achieve better results when using per-vertex ambient occlusion. Although this chapter has been written for the Unity VertexDirt plugin, these tips could be useful for any other application where you can bake indirect lighting to vertex colors (e.g. 3D Studio Max).

Ambient Occlusion is a popular way to bring indirect lighting and shadowing effects into your realtime 3D scenes. There are basically two different approaches, you can pre-generate the AO and store it somehow for use in your geometry, or you can calculate the AO in realtime through a post image effect (e.g. SSAO).

The per-vertex ambient occlusion stores the generated AO data in vertex colors. This is a mobile friendly approach as it does not require additional images, specific unwrapping or heavy calculations. Although it requires good topology and more detailed geometry as usual.

Some examples when per-vertex ambient occlusion is a GOOD idea:

- generally for mobile applications / games, where small file size and low system requirements are important.
- for organic models, because these models usually already have good topology and curvy surfaces.
- for detailed / subdivided models as they usually have enough vertices for good results.
- models textured with texture atlases or any other examples when UV overlapping happens.

And some cases when per-vertex ambient occlusion is a BAD idea:

- for extremely low poly models
- when the models are completely unwrapped and / or lack of UV overlapping (in that case, you can simply use regular lightmapping)
- when vertex-colors are not supported or already taken into account (for texture blending for example).

So it needs more vertex than usual, why this is a good idea on mobile?

- VertexDirt bakes to the vertex colors. Reading them is possible in the shaders's vertex program, so you can have lighter fragment/surface programs.

Differences between Beast Lightmapping and VertexDirt

- VertexDirt only bakes AO, can not calculate direct lighting, or bouncing like Beast do, just approximates the indirect lighting.
- VertexDirt writes to the vertex colors. This requires a new mesh instance in Unity right now, but does not require textures.
- VertexDirt can bake objects one by one, in any time without ruining the previous bakes.
- VertexDirt's quality depends on the mesh's topology. Sometimes it needs more attention to get good results.

5. Save baked meshes

You may want to save the baked meshes for using them elsewhere. One way to do that is to prefabing the baked gameObjects. Other way is to use the VertexDirt Save meshes tool. It is also useful when you want to make your baked models independent from VertexDirt (normally, VertexDirt adds a ColorHandler component to the baked GameObjects).

To use the Save meshes tool, follow these steps:

1. Select one parent GameObject which contains the baked meshes.
2. Open Tools/Zoloigo/VertexDirt save meshes
3. Set the path for the new mesh assets.
4. Hit the Save meshes button.