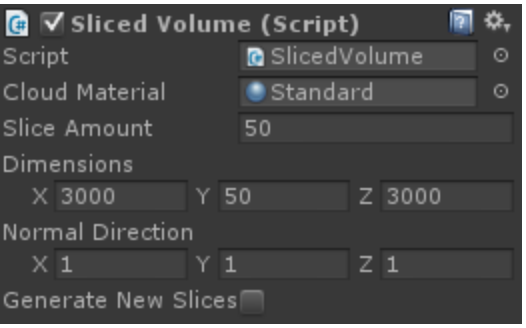
Volumetric Clouds

// Instructions

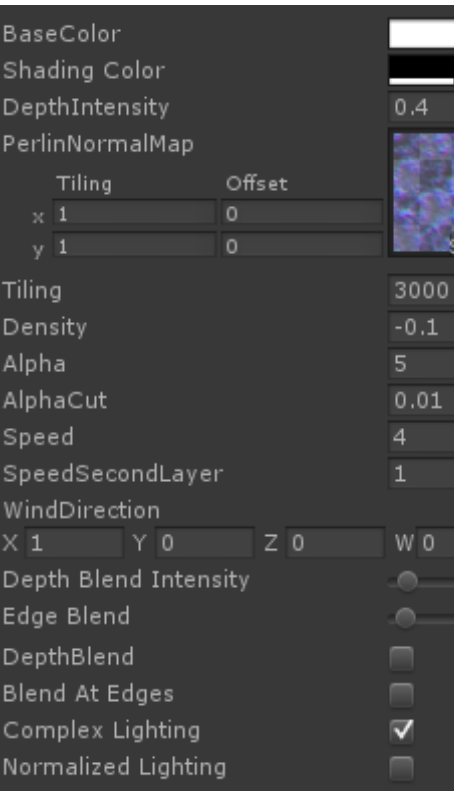
1. Place the slicedVolume script inside the script folder on an empty gameobject.
2. Drag and drop the cloud material onto the script's "Cloud Material" slot.
3. Change the next variables as you see fit.
4. Toggle the "Generate New Slices" boolean.

// Volume Script



- The amount of slices to generate.
- The area around the transform in which the slices will be generated.
- Invert the normals channel, do not modify this setting if lighting behaves correctly.
- Toggle to generate slices.

// Cloud Shader



- Main cloud color.
- Color(RGB) and intensity(A) of the shaded part
- Intensity of the fake depth shading.
- The map used to generate clouds(RGB = normals, A = Heightmap). Perlin noise is recommended.
  The texture has to use the alpha channel, dont mark it has a normal texture in the importer settings.

- The number of unity unit per texture tiling.
- Cloud density, higher means more and larger clouds.
- The alpha multiplier, higher means less transparent clouds.
- Threshold below which the alpha will be cut.
- The higher those values are, the faster the clouds will move.
- Those clouds will move towards this direction, Y and W channel aren't used.
- The hole size and density.
- Gradient size and density.
- Creates a hole near objects going trough clouds(mountains, jets, etc).
- Transparency increases towards the border of the clouds to blend them nicely in the horizon.

- Execute normalmaps, light direction dots and other light calculations.
- Switch between normalized dot product and standard.

// Tips

If you would like clouds at different levels, you could always create a second gameobject with a second modified material at another level.

The two “Speed” variables shouldn’t have the same value, clouds won't be able to morph through time if they are.

Do not mark the gameobject cloud as static, the script won't be able to update the scale which will kill the draw order of the transparency.

// Optimisations

The more pixels of those clouds you have on screen, the higher the cost, so
amount of slices,
dimensions (when looking towards the horizon in this case),
texture resolution + compression,
tiling,
density,
alpha + alpha cut
and screen resolution alter the performances by a lot, play with those values as you see fit.

Shadows casting, depth blend and edge blend should be avoided if you don't really have to use them, consider replacing the edge blend by fog in unity’s render settings.

Direct3D 11(Dx11) should render them a LOT faster, consider using it in your project if you target the PC platform.

// Clouds not behaving correctly ?

The texture used with the shader needs four texture channel(RGB = Normals, A = Heightmap), do not mark your texture as a normal map and check if the compression used doesn’t discard one of those four channels.

Check the variable “Cloud Material” on the sliced Volume script, it has to be the same as the one used on the same game object.

If you toggled DepthBlend on and the clouds simply disappear, you might have to create a depthmap.

If the above didn’t fix it, try changing “Normal Direction” on the script and if that still doesn’t help then drop by the official thread on Unity’s forum.