Volumetric Clouds 2.x

// Instructions

1. Drag the "RaymarchedCloud" script inside the folder to your camera.
2. Drag and drop the cloud material onto the script's material slot.

// Material variables

- **Perlin Normal Map**
  - Texture used once rendering to determine where and how clouds should be rendered.
- **Colors**
  - **Base Color**
    - rgb = Main color, a = shading intensity towards white
  - **Shading Color**
    - Rgb = Shaded areas color, a = shading intensity towards black
  - **Use Normalmap**
    - Shade based on surface direction.
    - **Indirect Lighting**
      - Amount of GI received
    - **Normalized**
      - How far light "scatters" before stopping
  - **Depth Intensity**
    - Top down shading
  - **Light Attenuation**
    - Light intensity and color inside clouds
  - **Distance Blend**
    - Opacity increases based on distance from the camera to blend nicely with the skybox.
  - **Screen space shadows**
    - **Shadow Color**
      - Shadow color.
    - **Draw Distance**
      - Maximum shadow rendering distance, shouldn't be greater or equal to raymarcher draw distance to avoid depth artifacts.
  - **Render Shadows Only**

- ■ Only render cloud shadows !
    - ○ High quality point light
        - ■ Render point lights in a higher quality than the base one, point light rendering is only supported on unity 5.4+.
- **Shape**
    - ○ **Density**
        - ■ Overall amount and size of clouds
    - ○ **Density Map**
        - ■ Same as above but in world space where the red channel controls the density(black : low, red : high), tiling and offset is in world space so 100 tilling means that the texture's size is equal to 100 unity unit.
    - ○ **Alpha**
        - ■ How thick are those clouds overall
    - ○ **AlphaCut**
        - ■ Discard pixels where opacity is less than that
- **Animation**
    - ○ **Speed**
        - ■ Wind speed over the first layer
    - ○ **Speed Second Layer**
        - ■ Wind speed over the second layer, should be different from 1st to create cloud evolution through time
- **Dimensions**
    - ○ **Cloud Transform**
        - ■ x is cloud height
        - ■ y is cloud size
        - ■ z is offset on the x axis
        - ■ w is offset on the y axis
    - ○ **Spherical**
        - ■ **Sphere position**
            - ● The sphere position in 3D space, w does nothing
        - ■ **Sphere stretch horizon**
            - ● Stretches the texture towards the horizon to create an illusion of great distance
    - ○ **Tiling**
        - ■ Amount of clouds per meter
    - ○ **Orthographic Perspective**
        - ■ A factor which fakes perspective if the camera used to render clouds is orthographic

- **Raymarcher**
  - **Draw distance**
    - Stop rendering once reached past that distance
  - **STEPS**
    - A raymarcher works by launching a ray from the camera's point of view and figuring out if we hit something. To do so, we check each x unit along the ray a function which tells us if we hit something,
      so a step is just that, a check along the ray.
    - **Steps Max**
      - Maximum number of steps to do, will stop before getting to that amount in most cases.
    - **Step Size**
      - How far should we go before asking if we hit something. Base distance.
    - **Step Skip**
      - How far should we go based on the current distance travelled, is useful to save performance on distant clouds.
    - **Step near surface**
      - All of the above multiplied by that once we are near a surface to accurately represent surface shapes.
  - **Lod Base**
    - Changes the mipmaps used, effectively reducing texture resolution.
  - **Lod Offset**
    - Same as above but based on distance to the camera.
  - **Opacity Gain**
    - The gain in opacity each time we hit clouds.
  - **Skip Pixel**
    - Skips pixels and takes the derivatives instead, helps performances
- **Debug**
  - **Render Queue**
    - When should the clouds be rendered, 2501 is the default and is right after opaque objects, 3000 is for most transparent objects, modify this value if clouds are rendered behind or on top of something it shouldn't.
  - **ZWrite**
    - Writes to depth, should be off

// Tips

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

Draw distance too short ? Try increasing the variable *Draw Distance* to the max that you would like to render (in unity unit, 1=1) then increase *Steps Max,* and if that didn't do anything, change *Lod Offset*.

// Optimisations

Everything under the raymarcher header changes performances a ton, play with those values and you should see significant changes.

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

// 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 that the variable "Material Used" on the "Raymarched Clouds" script is filled with your material.

If the clouds simply don't show up, you might have to [create a depthmap](#).

If none of this helped then drop by the [official thread](#) on the Unity forum.