True Clouds

How to start

1. Create cloud meshes
2. Assign Cloud Material to them *(located in TrueClouds/Materials)*
3. Arrange clouds on scene and put them in a separate layer *(we'll call it Cloud Layer)*
4. Add *Cloud Camera 3D* component to your camera
5. Set *Cloud Mask* to *Cloud Layer*.
6. Set *Blocking Mask* to everything except for *Cloud Layer*
7. Play with settings and see the clouds change!

Options

General Settings

Clouds Mask	Layers that contains clouds
Blocking Mask	Layers that contain objects that can occlude clouds
Fallback Distance	Distance in which objects are completely occluded inside a cloud
Light Mask	Layers with additional lights for clouds
Late Cut	Apply depth test after the blurring step
Approximate Distance	Approximate distance to clouds. Just scales other values to make your life easier. Set up once and don't change:)
Depth Precision	How precise do you want the depth to be? Keep low

Light Settings

Sun	Transform for the sun. Clouds act as if it was a directional light
Use Ramp For Coloring	Enables ramp coloring
Ramp **if Ramp is on*	Texture for lightning. From left to right color goes from the light to shadow
Light Color	Color of light
Light End **if Ramp* is off	Angle threshold, below which there is only *ShadowColor*
Shadow Color **if Ramp* is off	Color of shadow
Silverline Power	Power of light to shine through the cloud

Silverline Distance **if *Halo Power* > 0	Radius of shining through

Blur Settings

Radius	Radius of blur for normals and alpha
Threshold	Threshold is subtracted from alpha. Then alpha is normalised. I.E. *a = (a - threshold) / (1-threshold)*
Power	*a = pow(a, power)*
Quality	Quality of blur (affects count of samples for gaussian)
Depth Filtering	How much does blur radius depend on distance If clouds in the distance are getting to blurry, consider increasing this parameter
Downsample Clouds	How strong should the render textures be downsampled Keep as low as possible if performance is bad
Downsample World	How strong should world's depth be downscaled Keep as low as possible if performance is bad Not as important as previous setting

Noise Settings

Use Noise		Should noise be enabled?
Can Be Locked Together	*Normal*	Power off normal noise
	Displacement	Power of displacement noise
	Depth	Power of cloud's depth noise
Wind		Direction and speed of wind applied to noise
Sinus Time Scale		Time scale for wavy depth noise
Texture		Noise source
Can Be Locked Together	*Noise Scale*	Scale applied to Noise Texture for normals and displacement
	Depth Noise Scale	Scale applied to Noise Texture for depth noise

Lighting

There are some tools for additional artistic control: point lights and tinting.

Point Light

⚠ Tints and Lights should be placed in layers of Light Mask

To add a Point Light go to *AddComponent -> Cloud Point Light*

Start	Where does the light start to fade off
Range	Range of the light
Color	Color of the light
Shadow Intensity	How much should the clouds be lit if the normals face away from the light source

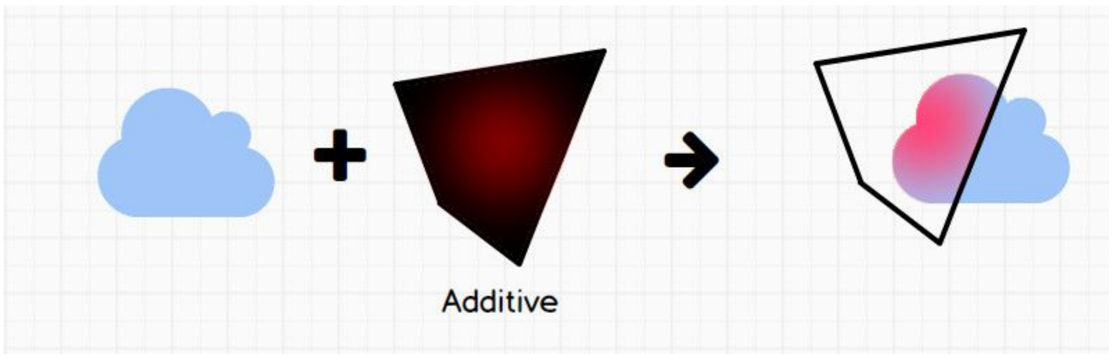
Tint

⚠ Tints and Lights should be placed in layers of Light Mask

Tint is a bit trickier to use, but it is more flexible in terms of what you can achieve.
Tint *Clouds/Tint** is a shader that can be used to locally modify clouds color: brighten, darken, or in a simple overlay mode.

To tint a cloud, create a new object in scene and assign a material that uses *Clouds/Tint** shader.
It works great with particle systems as well.

Tint Color	Color of the tint
Main	Main Texture
Max Distance	Maximum distance from cloud's surface at which the tint is applied.



For examples go to *TrueClouds/ExampleScenes/Scenes/CloudMobile*

Optimizations:

1. Start with maxing-out the *Downsample Clouds*. On mobile platforms I suggest you a minimum of 2
2. Next goes the *Downsample World.* It is especially important when *Late Cut* is enabled, but unfortunately artifacts are more visible as well
3. Decrease blur quality
4. Optimize your clouds geometry. You don't need high-frequency details as they will be lost in the blur

Tips & Tricks

This cloud system is just a big illusion, that makes player think that objects are fluffy and have volume. There is no ray tracing/path tracing going on. There even is no 3d noise, only one pre-computed noise texture. This is what makes the system so fast, but it makes you work a bit harder.

Here are some tips to keep the illusion working:

1. Keep camera outside the cloud, since the clouds don't actually have any volume
2. Create big clouds (bigger than the fallback distance) if you want them to be near an real-world object
3. Turn *LateCut* on if there are a lot of opaque objects in front of the clouds. Especially if they are small.
4. Pick *Light Color* and *Shadow Color* close to the color of the sky to achieve believable look

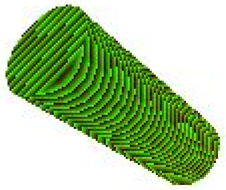
Performance

Demo with clouds:
Nexus 5x − 50 fps on whole scene. 7 ms on clouds
One plus 5 − solid 60 fps
Modern IOS devices − solid 60 sps
PC with gtx 940m − 180 fps on whole scene, 0.5ms on clouds

How does it work?

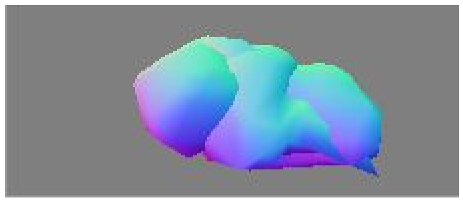
I will describe the *Late Cut* workflow

1. Render textures are created, according to *Resolution Divider* option you've set
2. Render loop
    a. Objects in *Blocking Mask* are rendered to the depth texture



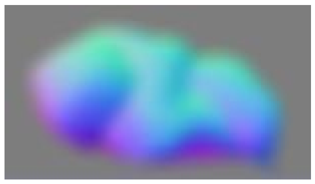
    b. Objects in *Clouds Mask* are rendered to the depth texture

    c. Objects in *Clouds Mask* are rendered to the normal texture



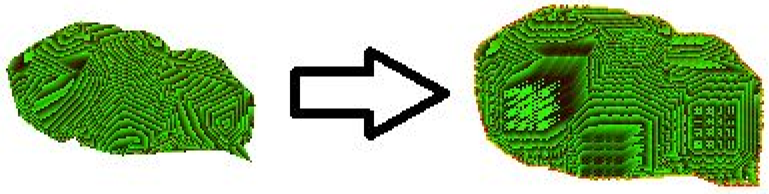
    d. Normal texture is blurred using gaussian blur. RGB which represents normal and A which represents the blending factor are blurred with different kernels
    e. Normal texture's A channel is clamped to a *Threshold*, normalised and raised in the power of *Power*

Normals (RGB)

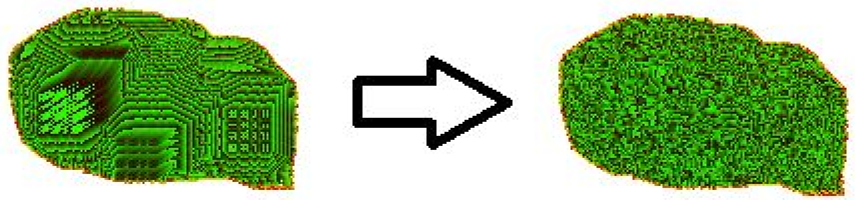


Alpha

    f. Cloud's depth is blurred



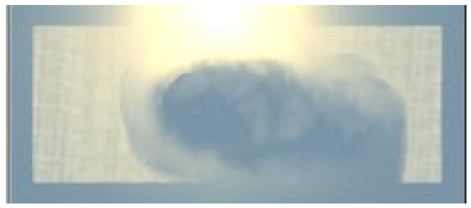
g. Noise is applied to depth



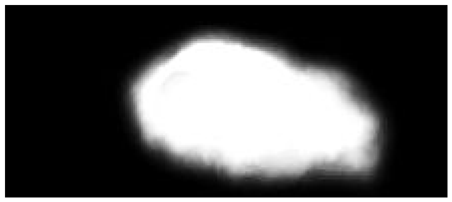
h. Color is calculated

    i. Noise is projected on clouds using [Triplanar Texturing]
    ii. Using the Noise, displacement is applied, i.e. sampling from different location
    iii. Noise is applied to normals
    iv. Lightning is calculated in a model similar to diffuse
    v. Silverlining is applied to semi transparent pixels.

Normals (RGB)

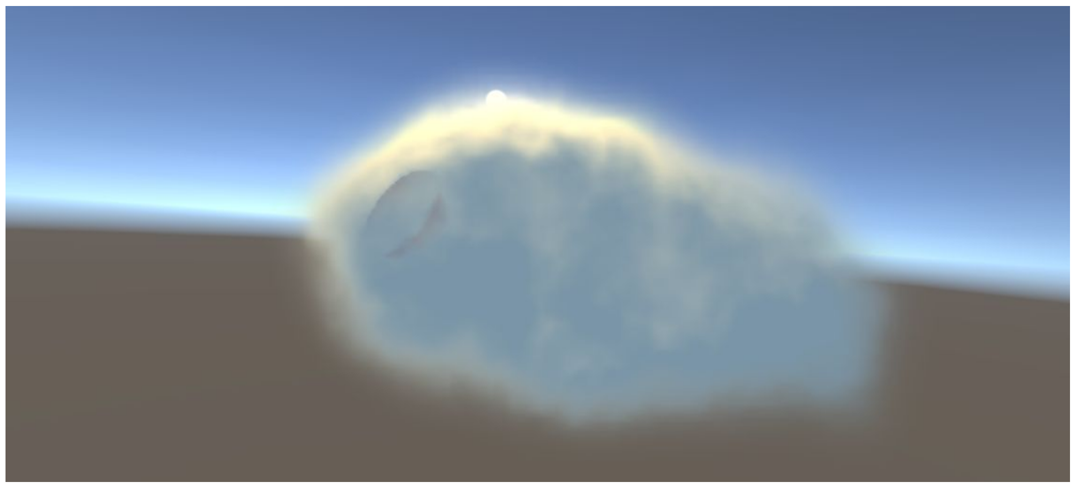


Alpha



i. Depth is applied

    i. If *World Resolution Divider* **= 1**, color buffer is blended with the screen, applying additional transparency where world depth is less or close to cloud's depth
    ii. If *World Resolution Divider* **< 1**, color buffer is blit to a small-res temporary buffer, applying additional transparency where world depth is less or close to cloud's depth. And that buffer, in turn, is blended with screen.



P.S.
Thank you very much for reading this!
If any problem occurs, please feel free to write me at mischapanin@gmail.com