Puffy Smoke Particle Emitter

by Alexandre Labedade - alesk@alesk.fr

Puffy Smoke is a particle system with a special render setup which displays a faked volumetric smoke on classic billboard particles, using a directional light as reference.
Its primary purpose is to emit smoke trails behind moving objects, but it can be used to also make clouds or any other smoke effect.

==============

Disclaimer and terms of usage :

This product is working with the free version of the great Unity Threading Helper,
more informations here : http://forum.unity3d.com/threads/90128-Unity-Threading-Helper

==============

How to use :

Puffy smoke is made of 2 scripts, 2 textures, some materials, and a shader.

- **Puffy_Emitter.cs** : this script can be assigned on all gameObjects emitting smoke or can be used on a single (not moving) gameObject as a main smoke emitter for multiples gameObjects.
It only generates particles data and doesn't render anything alone.

- **Puffy_Renderer.cs** : this script will render the particles generated by the emitters. It can be added to any gameObject in the scene.

- The default main texture is used to change the lighting on the smoke and the second is used to add small animated details.

- The shader can be found in the shaders list under "**Puffy_Smoke / FakedVolumetric**"

To link an emitter to a renderer, drop the Puffy_Emitter gameObject in the "Emitters" list in the inspector view of the Puffy_Renderer object.

You can also do it by script :

first identify the **Puffy_Renderer** script with something like :

```
Puffy_Renderer theRendererScript =
Puffy_Renderer.GetRenderer( "name_of_the_object_holding_the_renderer_script" );
```

or :

```
Puffy_Renderer theRendererScript = Puffy_Renderer.GetRenderer( (int)index );
```

or if you have only one renderer in the scene :

```
Puffy_Renderer theRendererScript = Puffy_Renderer.GetFirstRenderer();
```

then :

```
theRendererScript.AddEmitter(theEmitterScript);
```

To unlink an emitter from a renderer :

```
theRendererScript.RemoveEmitter( "name_of_the_object_holding_the_emitter_script" );
theRendererScript.RemoveEmitter( (int)index );
theRendererScript.RemoveEmitter( emitterScript );
```

The **Puffy_Emitter** script can be assigned to any gameObject in the scene, and will emit smoke as long as its auto emit parameter is on.
If auto emit is off, you can spawn single particles from script using its SpawnParticle() function :

```
Puffy_Emitter theEmitterScript = GetComponent<Puffy_Emitter>();

theEmitterScript.SpawnParticle(Vector3 start_position, Vector3 start_direction, float
start_speed, float start_lifetime, float start_size, float end_size, Color start_color, Color
end_color);
```

To create a row of particles between two positions (useful for trails) use the SpawnRown() function :

```
theEmitterScript.SpawnRow(Vector3 row_start_position, Vector3 row_end_position, float
stepDistance, Vector3 start_direction, float start_speed, float start_lifetime, float start_size,
float end_size, Color start_color, Color end_color, float intermediateDensity);
```

or :

```
theEmitterScript.SpawnRow(Vector3 row_start_position, Vector3 row_end_position, Vector3
velocityOffset, float intermediateDensity);
```

This function allows to quickly spawn particles using the default values specified in the emitter.

In both functions, intermediateDensity is optionnal, if omitted it will be set to the emitter inspector value.

Here is a simplified version of particle's spawn call in the homing missile script

```
// spawn particles in the fixed update function to ensure the trail regularity
void FixedUpdate(){
    // change the emitter colors, if you want a specific color for the smoke of this missile
    emitter.startColor = color;
    emitter.endColor = color;
    // in the same way you can modify the other emitter parameters (lifetime,size, random
values...) before spawning particles. Have a look at the Puffy_Emitter_Inspector script found in
Editor/PuffySmoketo see which parameters are interesting.

    // spawn particles between the missile's last position and its current position
    emitter.SpawnRow(previousPosition , _transform.position, moveVector * -smokeSpeed);
    // moveVector * -smokeSpeed => to eject the particles behind the missile direction

    // update the start position for the next row of particles
    previousPosition = _transform.position;
}
```

In this homing missiles demo, a single emitter is used for all missiles.
So its auto emit parameter must be off, since the particles creation is controlled by each missile script.

==============

Inspector parameters :

Puffy_Emitter script

- ***Auto Emit*** : the emitter will automatically spawn particles, one every fixed frame update (or more if trail mode is enabled and the emitter is moving).
- ***Freezed*** : particles updates will be stopped as long as this option is enabled.

- ***Chunk Size*** : number of particles available in the first place.
- ***Auto resize*** : if enabled, the particles count of the emitter will be unlimited. As soon as the limit is reached, it will be upped by the value of ChunkSize.

- ***Trail mode*** : if enabled, multiple particles will be generated as a row to fill the gap between the current position of the emitter and its previous position.

- ***Trail auto step*** : automatically define the distance between the particles created to fill the gap (based on the start particle size).
- ***Trail step factor*** : the auto step value will be multiplied by this factor, to tweak it depending on the visual result (lower values means more particles).

- ***Trail step distance*** : if auto step is off, this value will define the distance between each particle in the row.

Here is the piece of code used to define the intermediate particles count along the distance between the previous and current position of the emitter :

```
if(autoStep){
    count = Mathf.FloorToInt(distance / (start_size * stepFactor));
}else{
    count = Mathf.FloorToInt(distance / stepDistance);
}
```

If auto step is on, the start size of the particles will be used to define the distance between each particles, so they will touch each others. When multiplied by the step factor, if its value is below 1, the distance will be smaller and the particles will start to overlap and give a better density effect.

If auto step is off, then the fixed step distance will be used to define the offset between the particles.

- ***Intermediate density*** : this value will affect the end size and lifetime of the particles created in the gap.
This helps to improve smoke density at the begining of the main particles lifetime, and remove them sooner (if value <1) to keep performances.

When using the SpawnRow() function all particles of the row will be affected by the intermediate density factor, **except the last particle of the row** which will keep the original values.

- ***Direction*** : vector defining the starting direction of the particle
- ***Direction*** variation : values randomly added or substracted to the previous parameter.

- ***Life time*** : how long will live each particle, in seconds
- ***Life time variation*** : value randomly added or substracted to the previous parameter. (assume the same explanation for all other 'variation' parameters)

- ***Start speed*** : speed of the particle along its direction vector.

- ***Start size*** : size of the particle at its birth

- ***End size*** : size of the particle at its death

- ***Start color / End Color*** : color of the particle at its birth and death.
Alpha is not taken in account here.
NOTE : The alpha value of the particle color is used internally to store the age of the particle.

Puffy_Renderer script

- ***Debug*** : display debug informations as an overlay in the game view
- ***Render*** : enable or disable render updates
- ***Update threshold*** : value used to split the render processing over multiples frames, if it takes too long to process in one frame.
- ***Light*** : the directional light used to define the faked volumetric lighting
- ***Particles material*** : material used to render all the particles
- ***Use Ambient color*** : add the ambient light color defined in Edit / Render Settings to the particles colors
- ***Camera Background Color as Ambient Color*** : use the camera background color instead of the ambient color

- ***Texture col count & Texture row count*** : value linked to the texture used (how many rows and columns are present on the texture) You don't need to change these values unless you provide a new texture with a different setup.
- ***Details scaling*** : size of the smoke details (may move to the shader parameters in a future update)
- ***Max Render Distance*** : particles beyond this distance from the camera will not be rendered
- ***Emitters*** : list of the emitters linked to this renderer
- ***Use Thread*** : use all your cpu cores to process particles, should stay always on.

FakedVolumetric shader

- ***Shadow color*** : color of the shadowed side of the particles
- ***Particle Texture*** : use the provided atlas
- ***Particle Details Texture*** : use any tiling grayscale noise texture to add details on the smoke
- ***Soft Particles Factor*** : (pro only) fade particles when clipping occurs (**seems broken for now**)
- ***Opacity*** : global opacity of the smoke
- ***Scattering*** : light transmission through smoke while it's spreading (older particle = wider spread = less shadow color)
- ***Density*** : define the amount of small details
- ***Sharpness*** : sharpness of the small details
- ***Details speed*** : how fast is moving the details texture in particle space

==============

Homing Missiles demo

- Use left click to launch a new missile

- Use arrows and mouse to move and look around (maintain shift to lock mouse look)

- Use right click to move the target

- Use P to toggle pause (so you can move around the frozen smoke)

- Use U / I keys to rotate the light (the big sphere is here to have a light reference)

==============

History :

29/09/2013 - Version 1.07

- Fix on the auto emit behavior
- Updated readme with more scripting details

15/09/2013 - Version 1.06

- Speed improvements : almost x2 on the benchmark scene with 50k particles
- MaxRenderDistance added to the Puffy_Renderer script
- Tweak on the shader to cast raw shadows on other objects (no self shadowing).
- Added the intermediate density parameter in the Puffy_Emitter script

07/09/2013 - Version 1.05

- Speed improvements on the meshes rebuilding and updates (+50% speed for this part)
- Tweaks in the shader on the details animation parameter
- New material + new texture for a first attempt at making a cartoon smoke look (not convincing yet)
- New parameters on the homing missile script (craziness and some random offsets tweaks)

03/09/2013 - Version 1.04

- Speed improvements : overall refresh rate multiplied by around 3
- Some changes in the smoke color of the missiles demo (end color = start color, instead of white)
- Launch count added to the missile launcher script
- Some corrections in this readme file

27/08/2013 - Version 1.03

- Tint color as been removed from the shader
- Changes on the shader and lighting options : the smoke now reacts to the light color and intensity, and to the scene ambient light color
- Two new parameters in Puffy_Renderer : "Use Ambient color" and "Camera Background Color as Ambient Color"

26/08/2013 - Version 1.02

- Cleanup in the package (old test files were added)

26/08/2013 - Version 1.01

- Homing missile demo updated with cleaner code

21/08/2013 - Version 1.0

- First release.

==============

TODO List :

- Improve performances
- Try to make it work on iOS (the shader is the biggest issue for now)
- Parallel work on a DX11 version whith compute shaders
- Add more textures and demo scenes