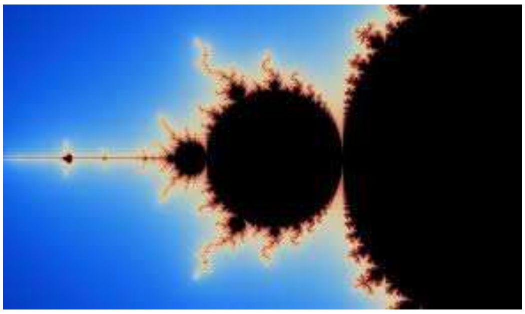
FRAKTAL GENERATOR

INTRODUCTION:

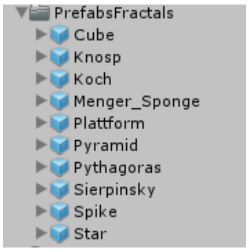


Fraktals are structures which show a repeating pattern in every displayed scale. They occur in nature especially organic substances are fraktals to meet their requirements as efficient as possible.

Fraktals can either be 2D like the famous Mandelbrot fraktal or complex 3D structures like the romanesco which is a fancy vegetable.

Fraktals often tend to look bizarre and randomly generated and only reveal their secrets when analyzing them. This means that the end result usually is unpredictable especially at higher iterations.

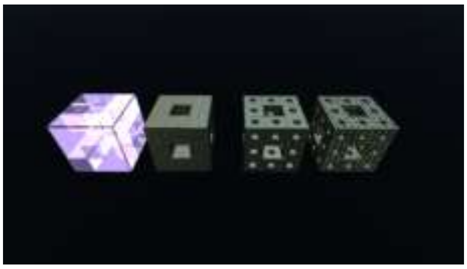
HOW TO USE:



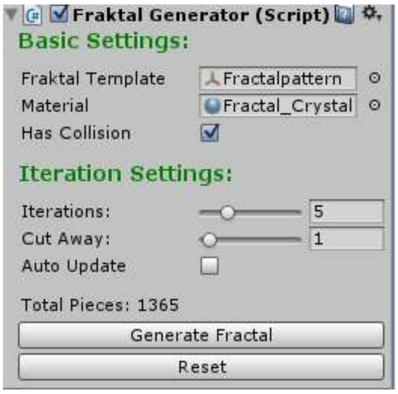
- PrefabsFractals
  - Cube
  - Knosp
  - Koch
  - Menger_Sponge
  - Plattform
  - Pyramid
  - Pythagoras
  - Sierpinsky
  - Spike
  - Star

For fast testing, just drag in one of those 10 example prefabs in your scene and hit the “Generate Fraktal” button which generates the fraktal based on a Fraktal Pattern. The material used for this fraktal can be set in the inspector. By default, the generated fraktal allows collision detection and is turned on by default. This can be disabled by unchecking the “Has Collision” flag.

The main settings are the **Iterations and Cut Away.** These values describe how complex the fraktal should get. Fraktals are generated with iterations and have an exponential complexity increment which means the higher the iteration, the more complex it will become. The image below shows a typical “menger sponge” which is one of the most famous fraktals. The bluish colored cube shows the pattern and the other 3 the results.



When messing around with the iteration slider, it calculates the total amount of mesh pieces generated and the amount of game objects required for this. The amount of pieces increases drastically on a high iteration count and is an indicator about how complex your result will be.



☑ Fraktal Generator (Script)

Basic Settings:

Fraktal Template: Fractalpattern
Material: Fractal_Crystal
Has Collision: ☑

Iteration Settings:

Iterations: 5
Cut Away: 1
Auto Update: □

Total Pieces: 1365

Notice: When dealing with recursion and iterations, high iteration counts can increase the complexity to a point where current technology is unable to handle the result.

To prevent accidentally high complexity, safety measures are implemented which are active if the amount of pieces is greater than 100000. However you have the option to bypass this safety feature but it is on your own risk.

If your pc cannot handle the complexity, unity usually freezes as your RAM memory is completely used up. My computer can handle a menger sponge with 5 iterations which is 5 year old now.

The **Cut Away** is another important value which is required for some fraktals where the last iteration is the final results. This value simple cuts away iterations up to that point. The menger sponge is one of those examples where the cut away is important because else the first iterations would overlap with last iteration. Also sierpinski triangles need this as without culling only the initial object would be visible.

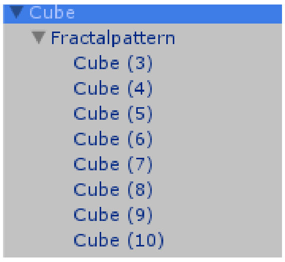
The last parameter is the amount of iterations attached to an object. The maximum amount of vertices for a mesh in unity is somewhere above 64000. This limit is exceeded very fast therefore the fraktal will be split into several mesh pieces.

This asset also includes a large mesh combiner. If you attach this script to the game object, the fraktal generator will use this to generate the mesh instead. This is recommended for very complex objects as you want the least amount of game objects needed as possible.

CREATING THE PATTERN

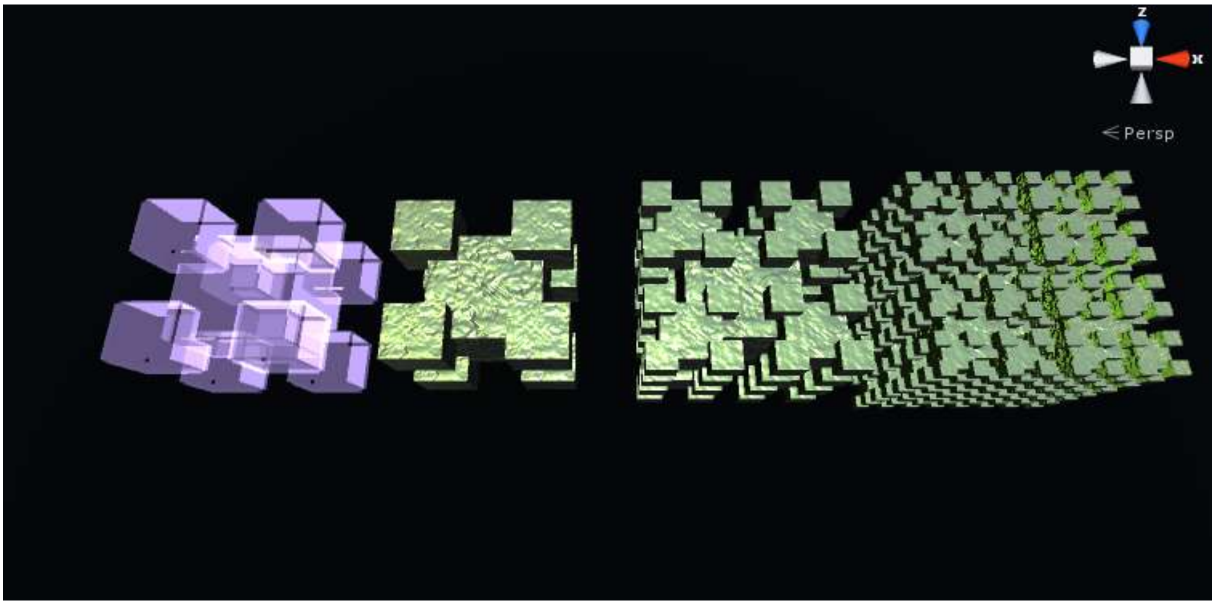
This topic describes how to create your custom fraktal. Every the fraktal pattern consists of a mesh object with other mesh objects attached to it as children.

In this image right, the “Fraktalpattern” object is the parent and describes the mesh used for the fraktal generation. It consists of 8 children which describes how the fraktal will develop. The more children the pattern has, the more complex the fraktal will become.



▼ Cube
  ▼ Fractalpattern
    Cube (3)
    Cube (4)
    Cube (5)
    Cube (6)
    Cube (7)
    Cube (8)
    Cube (9)
    Cube (10)

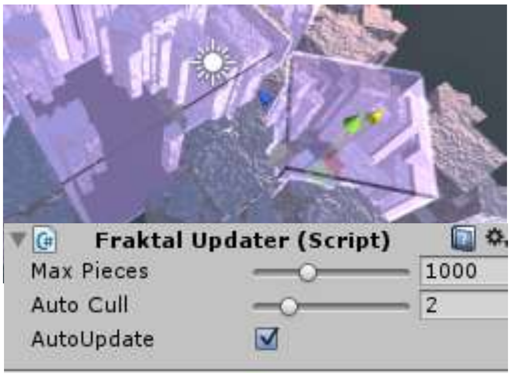
The parent of the pattern is the actual generator and has the pattern assigned to it. When creating the fraktal by hitting the “Generate Fraktal” button, the pattern itself gets disabled and is enabled again when you reset the fraktal. You can see the resulting pattern and their results in this image below: The bluish transparent object is the pattern and the others are the results



AUTO UPDATE:

Is also possible to let the generator auto update the fraktal while you are editing the fraktal pattern. This creates an intuitive workflow where you immediately see the result when you design the fraktal pattern.

In order to use it, select one component of the fraktal pattern and activate the AutoUpdate flag in the Fraktal Updater component. It calculates the maximum iteration possible with a piece count smaller than max pieces.



Fraktal Updater (Script)
Max Pieces 1000
Auto Cull 2
AutoUpdate ☑

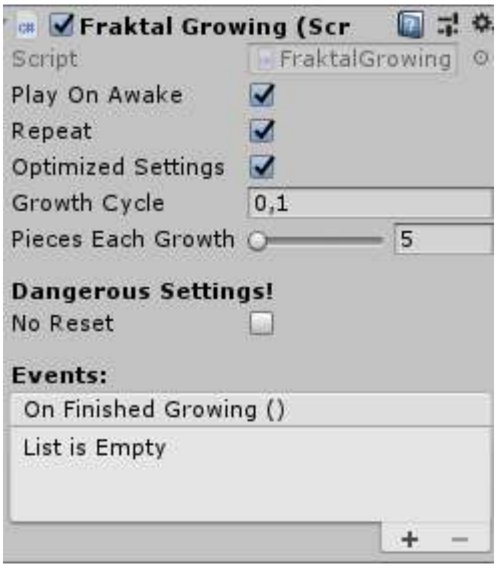
The "Max Pieces" value is set to 1000 by default and can be increased to 3000 and Auto cull culls away the first 2 iterations as it would obstruct the pattern itself.

GROWING A FRAKTAL:

It is possible to grow a fraktal in real time where you see the generation steps. An example is shown in the sample scene where fraktals are growing during playmode. The growth cycle is in seconds and describes the time between growth cycles and the pieces each growth value describes how many pieces are added to the fraktal each cycle.

The NoReset parameter will prevent the fraktal from being destroyed when Play is called. If repeat is also true, the fraktal will grow an infinite amount of pieces over time so be careful.

To do this, several functions are exposed and can be called using the standard unity event system:



☑ Fraktal Growing (Scr
Script FraktalGrowing
Play On Awake ☑
Repeat ☑
Optimized Settings ☑
Growth Cycle 0,1
Pieces Each Growth 5

Dangerous Settings!
No Reset □

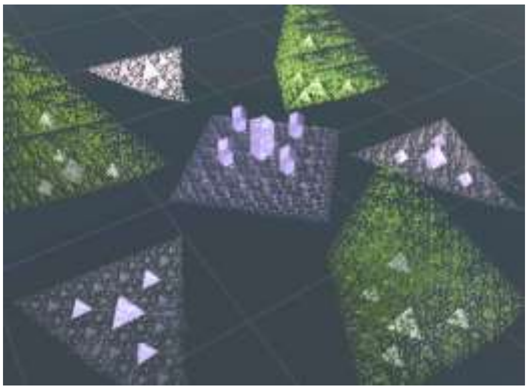
Events:
On Finished Growing ()
List is Empty

- **Play:** Starts the growing process. Destroys the previous fraktal unless NoReset is true.
- **Stop:** Stops the growing process, cannot be resumed.
- **Pause:** Pauses the growing and can be continued later on.
- **Resume:** If paused, continues the growing.
- **Reset:** Destroys the fraktal. If called while fraktal is half finished, the unfinished result is destroyed and the remaining parts will still grow.

LARGE MESH COMBINER:

This asset uses the large mesh combiner which breaks the vertices limit of a mesh and allows exporting. It will usually be included in my procedural mesh generation products as it is a core component. For further details, check out the second included documentation about the Large Mesh Combiner

NATIVE FRAKTAL GENERATOR

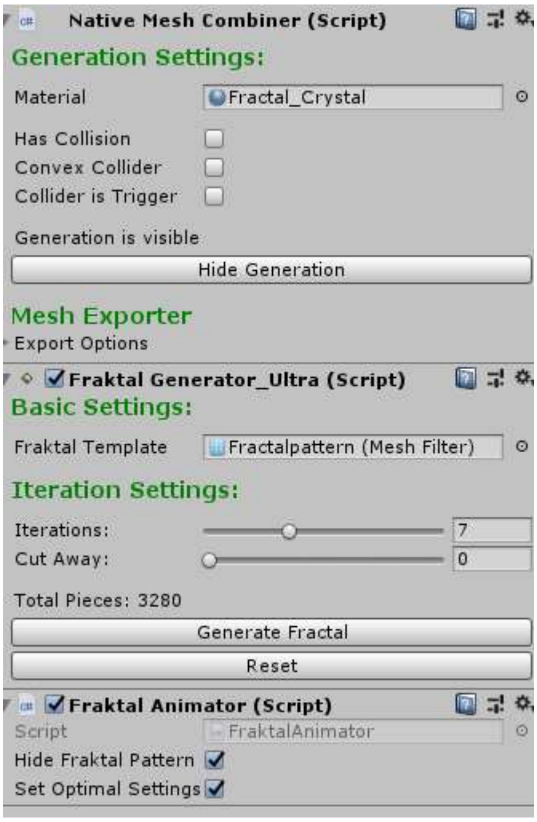


Unity 2018.1.0 implemented the burst compiler which allows the implementation stuff in real time never seen before. The FraktalGenerator_Native uses this functionality and allows animated fraktals. Also it allows the generation of fraktals with high iterations that would freeze Unity immediately. Now the RAM and GPU memory are the limit. In order to gain the full performance boost, the burst package must be installed. Therefore an additional document is provided inside the documentation folder.

The Native fraktal generator is similar to the normal fraktal generator but uses the NativeMeshCombiner which is instead of the LargeMeshCombiner. For animations, the fraktal animator must be attached to the generator object else it will not update the fraktal when the fraktal pattern changes.

The fraktal animator has the set optimal settings and it is highly recommended to flag it. It changes the settings for improved performance like disabling collision because updating a mesh collider is very expensive especially on huge meshes. Now when moving, rotating, or scaling fraktal patter components, the fraktal is updated immediately.

In the sample, a simple rotating script is attached to the fraktal pattern objects, animating the fraktals. Components of the pattern can also be animated using the animation feature from Unity itself.



Native Mesh Combiner (Script)

Generation Settings:

Material Fractal_Crystal

Has Collision □

Convex Collider □

Collider is Trigger □

Generation is visible

Mesh Exporter

Export Options

☑ Fraktal Generator_Ultra (Script)

Basic Settings:

Fraktal Template Fractalpattern (Mesh Filter)

Iteration Settings:

Iterations: 7

Cut Away: 0

Total Pieces: 3280

☑ Fraktal Animator (Script)

Script FraktalAnimator

Hide Fraktal Pattern ☑

Set Optimal Settings ☑

POSSIBLE ERRORS/DANGERS:

UNITY FREEZES AHHHHHHHHHHHHHH:

- **YOU COULD HAVE IGNORED THE WARNINGS ☺.**
- Amount of mesh pieces is way to high.
- Amount of game objects is very large

BAD LOOKING MESH:

- Mesh looks horrible if the vertice count for a mesh is higher than the maximum allowed. A error message indicates this. To solve this, either lower the iterations per object value or add the large mesh combiner as component.

Z-FIGHTING:

- This happens occasional when dealing with symmetric fraktals. The resulting fraktal will consist of overlapping meshes. One good solution for this is simply alter the pattern so it is not exactly symmetric anymore. The spike fraktal sample is one of those cases as the objects of the pattern are slightly rotated to avoid this.

LAST NOTES:

If you have any questions, suggestions, bug reporting don’t hesitate to contact me. If you are going to sell a game which uses this asset, inform me because I may buy your game and play it ☺

Contact Information:

E-Mail: mhartl.mmt-b2013@fh-salzburg.ac.at

Homepage: http://fraktalia.org/