Visualizing the Cosmos: a Procedural Approach

Dominique Vidal*
VFX supervisor, BUF Compagnie

Figure 1: *Stills from Cosmos a Spacetime Odyssey, 2014 ©Cosmos Studios / Fuzzy Door / National Geographic*

Abstract

This talk presents an overview of the specific tools that BUF has developed for cosmic visualization purpose. First created for the movie *Thor*, these methods and tools have been enhanced for the TV series *Cosmos a Spacetime Odyssey*, and specially optimized for a broadcast economic model. Integrated in the BUF's unique in-house software pipeline, these tools have been meant to resolve specific challenges for visual effects, such as huge scale issue, designing quasi-static states for nebulae, or building procedural photo-based assets.

1 Infinite ride

For a sequence showing our cosmic address, we had to create, in a single shot, a pull out from the Milky Way to the exterior of the known universe. That means a variation of scale around 10^7 meters, impossible to handle in a single 3d scene. We created an exponential camera animation that provides a constant zoom out of the interest plane. Thus, we used this camera to split the scene into several passes corresponding to different orders of magnitude (Milky Way and neighborhood, galaxy cluster, super cluster, limit of the observable universe). Relying on the constant camera zoom out effect, all separate parts of the universe were easily composed like Russian dolls.

2 Quasi-static design

Many cosmic environments are in a quasi-static state, result of a long and complex dynamic evolution. A nebula is just the remaining dust of a supernova explosion. The challenge was to build this kind of asset looking like a frozen dynamic simulation without the cost of a real simulation.

*e-mail:dom@buf.com

SIGGRAPH 2015 Talks, August 09 – 13, 2015, Los Angeles, CA.
ACM 978-1-4503-3636-9/15/08.
http://dx.doi.org/10.1145/2775280.2792539

For galactic arms, many nebulae or stellar nursery, we have scripted a tool to design fractal trees made of curves which became the spine of our asset. Each branch of these trees is the trajectory of a particle in a simple dynamic simulation. The user can easily tune each tree shape, playing with many parameters (number of branches, recursion level, noise, central collapsing, etc.). Then, the tree is sampled to produce various elements, like particles or lights, which are noised in position to loose the tree shape a bit. Next, we can dress the particles with textures of stars, galaxies, or even clusters of galaxies to reach a higher fractal level. Finally, this set of stars/galaxies seems to be randomly distributed in space. However, a subtle dynamic structure can be perceived.

3 Procedural modeling

We had to make a landscape of the Jupiter planet, seen near and inside the Great Red Spot, aka Jupiter's Eye. To build this gigantic vortex (the size of 3 Earths), we decided to represent it as canyons made of almost solid clouds. To ease the process, we chose to work with procedural modeling. We have created special manipulation tools to design and animate the huge cyclonic shapes. A procedural displacement texture comes at the end to add thin details. Upon this model, we have designed many paths flowing along the canyon walls to lead particles of smoke.

4 Procedural Rendering

This scripting approach of the pipeline has been also applied to rendering. We have coded many procedural shaders for meteor surfaces, black hole accretion disc, surface of stars, fractal texture of the observable universe, landscape of Venus, Europa, Io or Triton. We have also developed volumetric shaders for a comet vapor tail, alien cosmic dust, or the clouds of sulfuric gas that cover Venus. Here, the challenge was to create a high efficient rendering and compositing pipeline to deliver spectacular and long shots in time (with a mean duration of 500 frames). In the end, we achieved an average rendering time per frame of 5 min on a standard 8 cores computer.