VFX Fractal Toolkit: Integrating Fractals into VFX Pipeline

Juraj Tomori
Animationsinstitut, Filmakademie Baden-Württemberg
Ludwigsburg, Germany
tomori.juraj@gmail.com

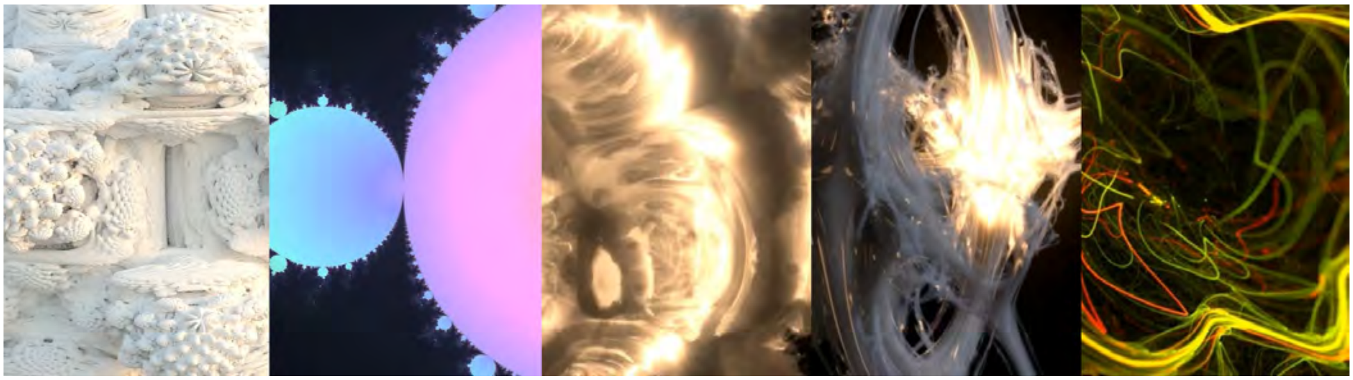

Figure 1: Imagery produced with VFX Fractal Toolkit

ABSTRACT

This paper proposes an innovative industry practice regarding fractal geometry generating and rendering processes. VFX Fractal Toolkit (VFT) aims to provide powerful, yet intuitive and artist-friendly workflows for exploration and generating of vast amounts of fractals. VFT allows for node-based description of fractals implemented in *SideFX Houdini*. VFT is built specifically for Visual Effects (VFX) pipelines and employs standard practices. It aims to provide artists with a toolset which would help them explore fractal forms of generative art directly in VFX applications.

CCS CONCEPTS

• **Computing methodologies** → **Animation**; **Rendering**.

KEYWORDS

fractals, distance fields, ray marching, vfx, pipeline

ACM Reference Format:

Juraj Tomori. 2019. VFX Fractal Toolkit: Integrating Fractals into VFX Pipeline. In *Proceedings of SIGGRAPH '19 Posters.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3306214.3338543

1 PREVIOUS WORK

Available tools for rendering fractals, among others, include *Mandelbulb 3D*, *Mandelbulber2* and *Fragmentarium*. Prior to developing the toolset an evaluation of the mentioned tools was made. Evaluated tools and VFT both rely on distance field representation of deterministic fractals. Generating or rendering of fractals in this representation involves raymarching those distance fields, which can be optimized by an "unbounding volumes" method which allows for skipping of empty space along the traced ray and thus reduces the number of computed ray steps [Hart et al. 1989].

The aforementioned tools have accurate and efficient rendering engines, but are not compatible with VFX Digital Content Creation (DCC) packages. Generated imagery or geometry is not easily transferable to, or replicable in, VFX tools. This suggests that those tools have not been built with VFX pipelines in mind [Kim et al. 2014]. Fractal scenes in the aforementioned tools are represented by a stack of layers. The VFT integration in *SideFX Houdini* takes advantage of node-based workflow which results in a more flexible fractal scenes description and in a better visual understanding of the logic behind fractal scenes.

2 FEATURES

The toolkit uses, and is fully integrated with, animation, geometry processing and rendering tools available in industry standard DCC applications. This makes it easy to be integrated into established VFX pipelines. VFT is designed as a framework, which can be easily extended or customized to suit specific needs. It features fractal functions from multiple sources: *Mandelbulber2*, [Christensen 2011], [Marczak 2010].

Thanks to the nature of fractals, each sample (point, voxel or pixel) computation is independent from its neighbors during the the generating process [Crane 2005]. This results in efficient scalability, both vertically (GPU, multiple CPU cores) and horizontally (multiple machines processing individual tiles of a partitioned domain).

SIGGRAPH '19 Posters, July 28 - August 01, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6314-3/19/07.
https://doi.org/10.1145/3306214.3338543

The fractal generating process outputs rich color information using an orbit traps technique [W. Carlson 1999]. This color information follows fractal structures, is customizable and allows for artistic shading and illumination. The color information can be stored either in point attributes, volume grids or in image planes depending on the context.

VFT aims to encourage artists to intuitively explore many fractal combinations. It is achieved by promoting animatable fractal parameters to artists and using node-based workflow within *SideFX Houdini* for creation of "hybrid" fractals. In VFT each fractal can be represented by its Julia set for an arbitrary coordinate. Multiple fractals can also be combined into a weighted "hybrid" fractal, which inherits features from its sources [Marczak et al. 2018]. A Signed Distance Fields (SDF) representation of fractals makes Boolean operations easy to perform. This allows for creation of more complex shapes through use of Constructive Solid Geometry (CSG) [Frisken and Perry 2006]. The number of promoted parameters, and the toolkit's flexibility, results in numerous possible combinations.

3 IMPLEMENTATION

VFT is integrated into multiple DCC applications. As a result, artists who are familiar with DCC packages can employ standard workflows for using the toolkit. The current state of the toolkit includes the integration into *SideFX Houdini*, which means that generated data can be further processed by standard Houdini node-based workflow, can be exported into various standard VFX formats and can be rendered with multiple production renderers. Fractal generating in *SideFX Houdini* is GPU accelerated and leverages modern parallel hardware thanks to the inherent parallelism of SDFs. GPU acceleration is achieved by performing calculations using OpenCL kernels, which can run on hardware from multiple manufacturers and, in case of memory constraints, can yield identical results when executed on CPU. The initial approach was implemented in VEX scripting language similar to [Kim et al. 2014], [Ebb et al. 2017]. However, the OpenCL implementation provided better performance and allowed for more interactive exploration.

The toolkit is also implemented with *The Foundry Nuke* compositing package where it provides a set of tools for generating deep dives into 2D fractal imagery. Promoted parameters can be creatively explored and animated. *The Foundry Nuke* implementation relies on shaders written in Blink scripting language which can be executed both on CPU and GPU devices gaining similar advantages as the OpenCL implementation.

VFT is also integrated with the *Solid Angle Arnold* production renderer in a form of Open Shading Language (OSL) shaders. Fractal volumes are evaluated at render time, which does not require their rasterization into volume grids and storage.

4 *SIDEFX HOUDINI* INTEGRATION

Fractal scenes are described by a node graph in a SOP (Surface Operator) context which is parsed into an OpenCL kernel. Individual nodes represent various fractal functions or space transformations. Fractals can be generated in multiple forms depending on VFX workflow needs. If a set of points on a surface is required, the surface can be raymarched from the projection plane of a perspective camera [Ebb et al. 2017; Kim et al. 2014], a point in space (e.g. for VR applications), or an enclosing shape (e.g. sphere or volume encompassing fractal surface) [Kim et al. 2014]. If occluded points are missing, fractal surface points can be generated from multiple sources. Dense point clouds can be rendered as sphere primitives with overwritten normals which provide the look of solid surfaces [Ebb et al. 2017]. If full internal structure is required, fractals can be sampled into voxel grids (fog or SDF volume) in specified bounds or camera frustum and afterwards can be converted into OpenVDB format for efficient post-processing and storage. Volumetric fractals contain rich inner structures and color patterns which are otherwise not visible [McGraw 2018].

5 FURTHER WORK

VFT aims to be available to artists in multiple areas of a VFX pipeline. Fractals can be raymarched in screen space in a form of fragment shader [Crane 2005; Quilez 2008], which makes them suitable for being integrated into game engines or even web applications. Various renderers (*Cycles, V-Ray*) support OSL shaders that make their host DCC applications suitable for implementation of VFT. Fractal logic and parameters from one tool can be transferred into another, which will result in identical fractals generated, but in an environment and format specific to the package. Further work will aim for basic integration into those areas, which will enable artists and studios to enhance it and integrate it into their workflows and pipelines.

REFERENCES

Mikael Hvidtfeldt Christensen. 2011. Distance Estimated 3D Fractals (V): The Mandelbulb & Different DE Approximations. Retrieved October 10, 2018 from http://blog.hvidtfeldts.net/index.php/2011/09

Keenan Crane. 2005. Ray tracing quaternion julia sets on the gpu.

Matt Ebb, Richard Sutherland, Daniel Heckenberg, and Miles Green. 2017. Building Detailed Fractal Sets for "Guardians of the Galaxy Vol. 2". In *ACM SIGGRAPH 2017 Talks (SIGGRAPH '17)*. ACM, New York, NY, USA, Article 12, 2 pages. https://doi.org/10.1145/3084363.3085060

Sarah F. Frisken and Ronald N. Perry. 2006. Designing with Distance Fields. In *ACM SIGGRAPH 2006 Courses (SIGGRAPH '06)*. ACM, New York, NY, USA, 60–66. https://doi.org/10.1145/1185657.1185675

J. C. Hart, D. J. Sandin, and L. H. Kauffman. 1989. Ray Tracing Deterministic 3-D Fractals. *SIGGRAPH Comput. Graph.* 23, 3 (July 1989), 289–296. https://doi.org/10.1145/74334.74363

Alex Kim, Daniel P. Ferreira, and Stephen Bevins. 2014. Capturing the Infinite Universe in "Lucy": Fractal Rendering in Film Production. In *ACM SIGGRAPH 2014 Talks (SIGGRAPH '14)*. ACM, New York, NY, USA, Article 1, 1 pages. https://doi.org/10.1145/2614106.2614166

Krzysztof Marczak. 2010. A Mandelbox distance estimate formula. Retrieved October 10, 2018 from http://www.fractalforums.com/3d-fractal-generation/a-mandelbox-distance-estimate-formula/msg14893/#msg14893

Krzysztof Marczak, Graeme McLaren, Sebastian Jennen, and Robert Pancoast. 2018. Mandelbulber End User Manual. , 17-24 pages. https://github.com/buddhi1980/mandelbulber_doc/releases/download/untagged-f939f0542fc9b9e38305/handbook-compressed.pdf

Tim McGraw. 2018. Fractal Anatomy: Imaging Internal and Ambient Structures. In *ACM SIGGRAPH 2018 Posters (SIGGRAPH '18)*. ACM, New York, NY, USA, Article 9, 2 pages. https://doi.org/10.1145/3230744.3230748

Iñigo Quilez. 2008. Rendering Worlds with Two Triangles with raytracing on the GPU in 4096 bytes. Retrieved October 10, 2018 from http://iquilezles.org/www/material/nvscene2008/rwwtt.pdf

Paul W. Carlson. 1999. Two artistic orbit trap rendering methods for Newton M-set fractals. *Computers & Graphics* 23 (12 1999), 925–931. https://doi.org/10.1016/S0097-8493(99)00123-5