レイトレ合宿６Ω

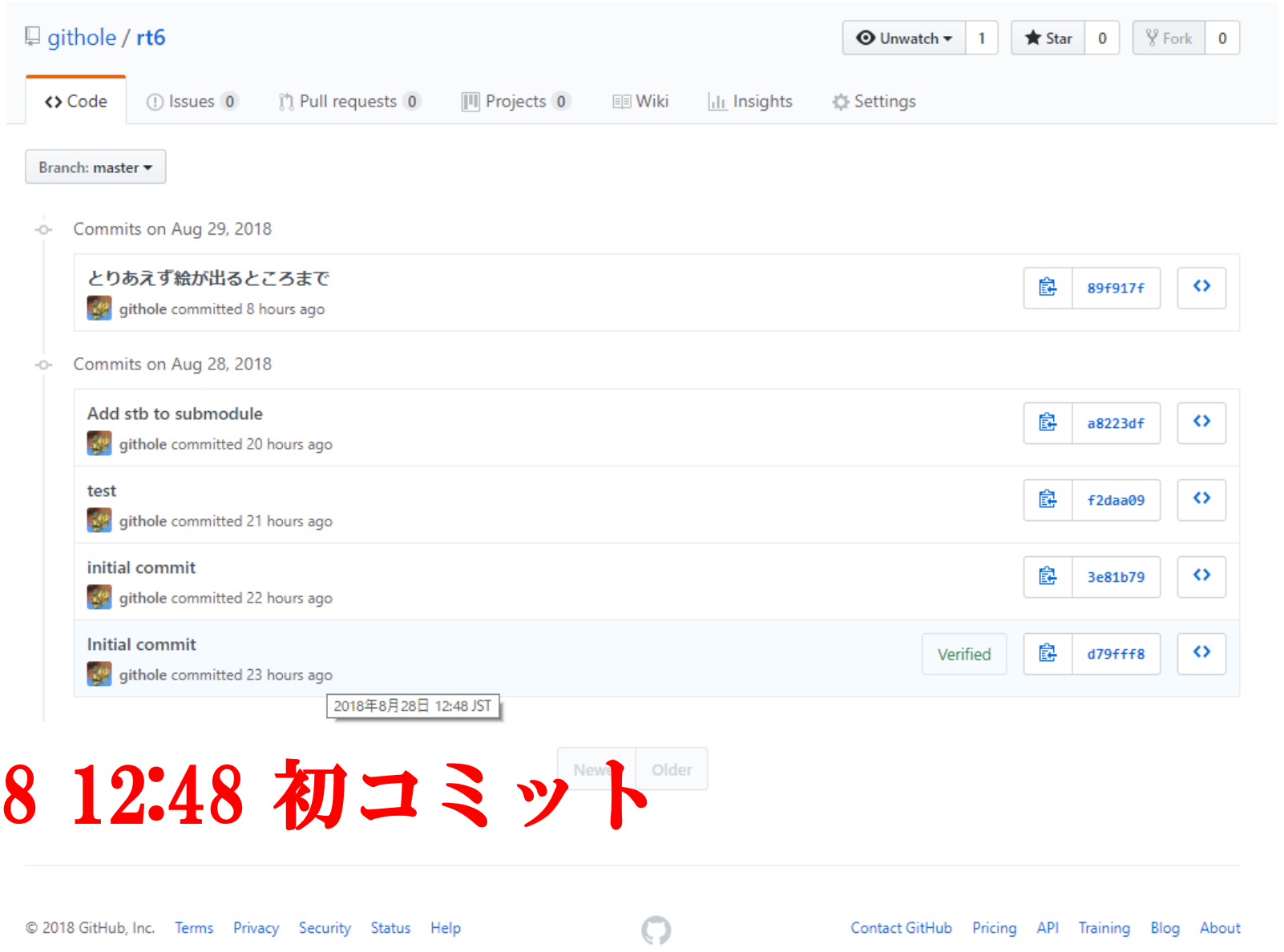
hole

エクストリームレイトレ合宿

テーマ「ゼロから始める」

エクストリームレイトレ合宿

「ゼロ」とは？



2018/8/28 12:48 初コミット

3:53 PM　hole 🏠 ベクトルクラス

出来た

2018/8/28 15:53 ベクトルクラス完成

開発は計画的にしましょう

エクストリームレイトレ合宿

- Basic Raytracing Framework
- Basic Parallel Rendering Framework
- Procedural Scene Asset
- Fully Global Illumination Unbiased Volume Renderer
- Heterogeneous Volume Support
- Adaptive Free Path Sampling
- Blue Noise Dithered Sampling
- Visual Debugger
- Simple Lens Distortion

エクストリームレイトレ合宿

- **Basic Parallel Rendering Framework**
  - 72スレッド全部立てる。NUMAも乗り越え全コア使うようにする。
  - 今回は、Procedural Assetなのでメモリアクセスが問題になることはなかろうという判断。実際、手元ではよい結果になった。

- **Procedural Scene Asset**
  - ひたすら手でハードコードボリュームを定義した。
  - 原始人。

CPU

Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz

論理プロセッサ

100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
100%	100%	100%	100%	100%	100%					

エクストリームレイトレ合宿

- **Fully Global Illumination Unbiased Volume Renderer**
  - ボリュームレンダリング方程式を素直に解く。Mie散乱のみ考慮。Distance samplingはDelta Trackingを採用。Ratio Trackingも実装してみたけど、かなり重くなるので力でゴリ押したほうが今回は良かった。

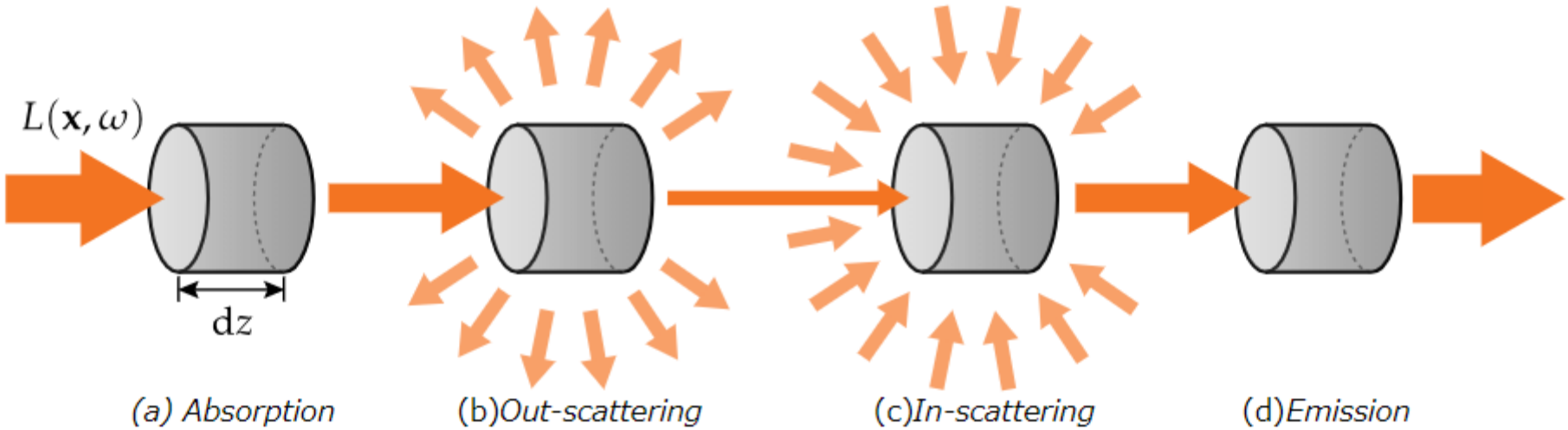
- **Heterogeneous Volume Support**
  - UnbiasedにHeterogeneousなVolumeからDistance sampling(Free path sampling)するとなると、やはりDelta Tracking系を使うことになるだろう。

Monte Carlo Methods for Volumetric Light Transport Simulation

Jan Novák[1] Iliyan Georgiev[2] Johannes Hanika[3] Wojciech Jarosz[4]

[1]Disney Research [2]Solid Angle [3]Karlsruhe Institute of Technology [4]Dartmouth College

In *Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports)*, 2018



Losses and gains of radiance due to absorption (a), scattering (b, c), and emission (d) within a differential volume element.

Abstract

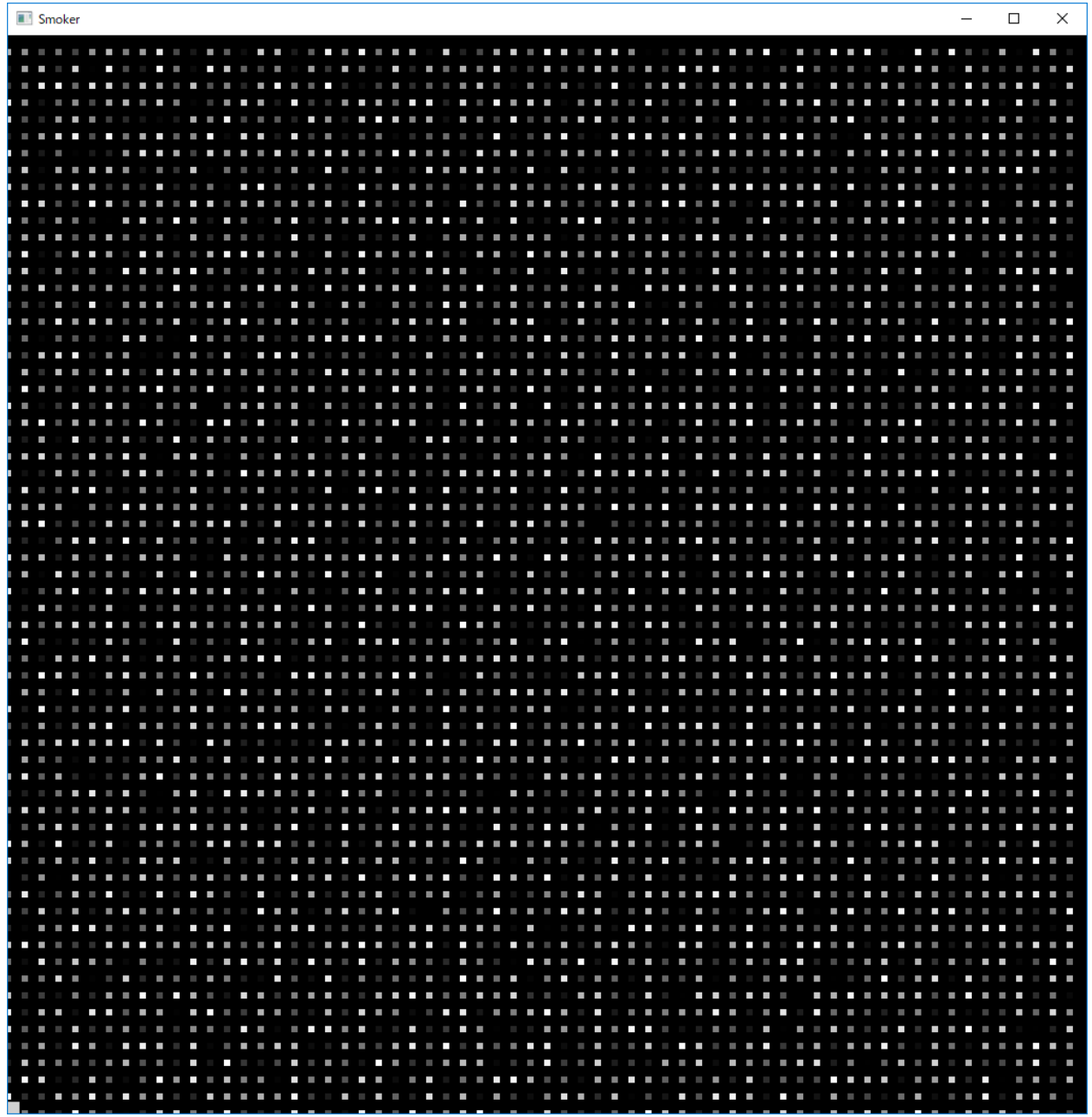
エクストリームレイトレ合宿

- **Adaptive Free Path Sampling**
  - ボリュームの空間を密度に応じて分割することでDelta Trackingを高速化するテクニック。
  - Yue (2010), “Unbiased, Adaptive Free Path Sampling”.
  - 今回はボリュームをボクセルに変換、適当なスレッショルドで子ノードをまとめていくようなOctreeを採用した。

エクストリームレイトレ合宿

- **Blue Noise Dithered Sampling**
  - Blue Noise使ってピクセル毎の乱数オフセットすると良い感じになるよ、というやつ。
  - http://raytracing.hatenablog.com/entry/2016/08/22/190948 を参照しつつ適当に実装＆投入。まあ使わないよりはマシかな感。

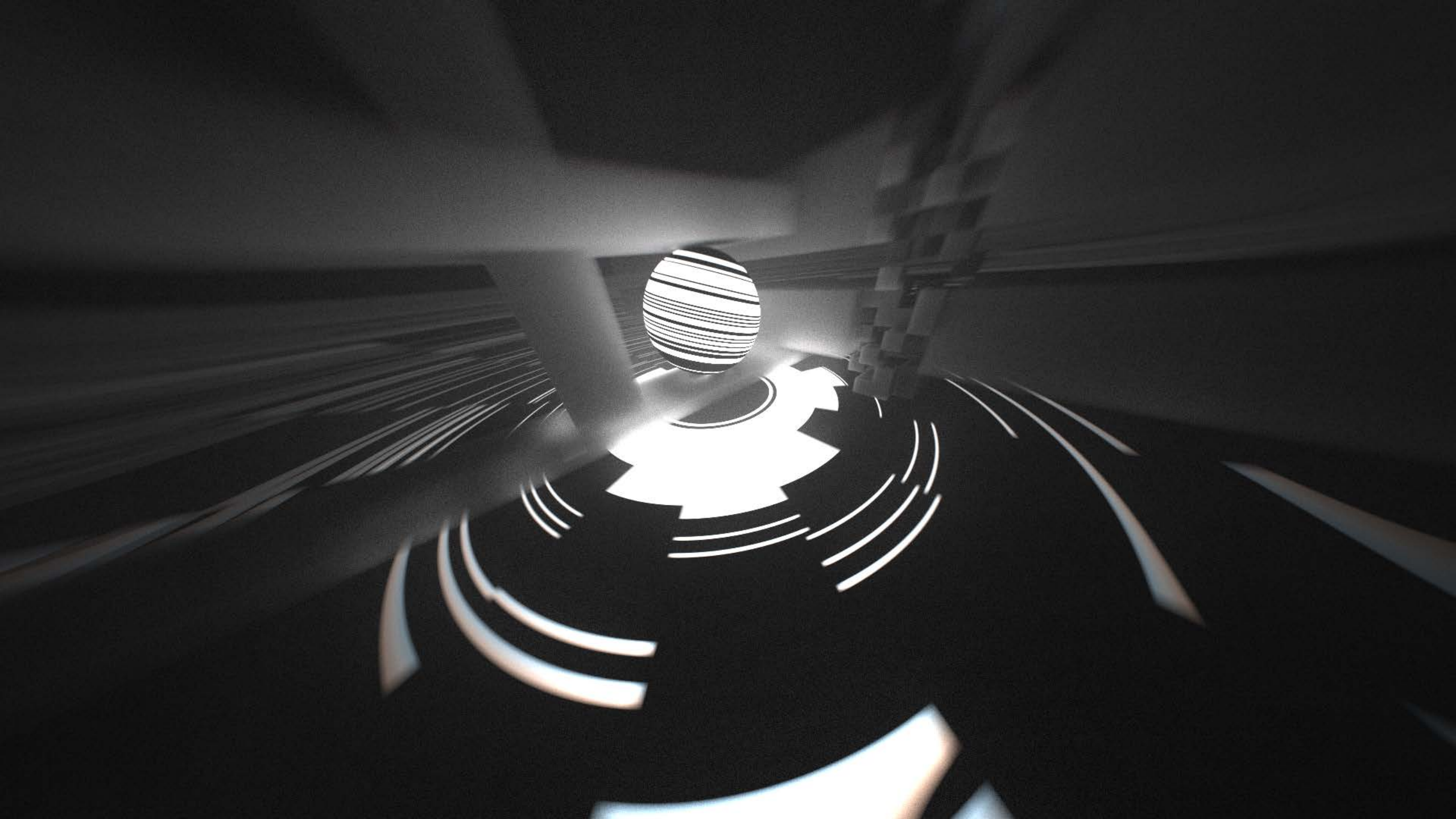
- **Visual Debugger**
  - Blue Noise生成のデバッグのために可視化ツールも用意。
  - 構想段階ではほかの用途にも使っていた。

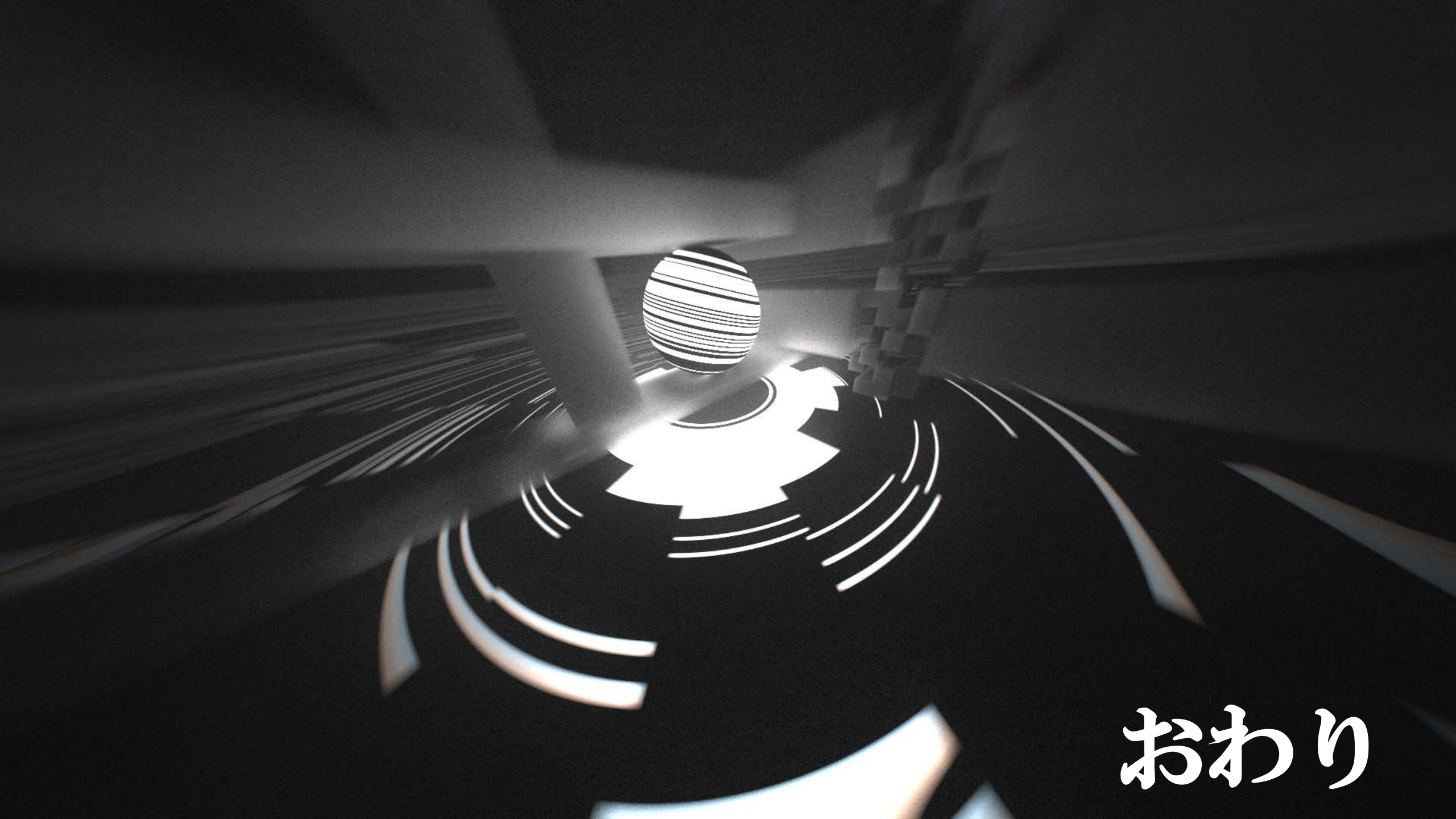

Smoker

高速化も頑張った

- 単純な実装だと遅くてノイズがやばかったのできばって高速化した。最終的に４，５倍くらい高速にはなった。
- **特に効いた手法。**
  - Volume Renderingで頻出のpow,exp,logを https://github.com/herumi/fmath で置き換え。
  - Adaptive Free Path Sampling
  - 様々なデータをLook Up Tableに格納して実行時に計算しないようにする。

- プロファイラはお友達。
  - 単位時間あたりの処理サンプル数を出せるようにしたので高速化しやすかった。




おわり