alpine

elk

今年のテーマ

Wide BVHを実装する

BVHのアルゴリズムが無数にあるので、できるだけ実装がシンプルなものを選択

Binary BVH

PBRT Bounding Volume Hierarchiesを参考にBinary BVHを実装

- Top-down construction
  - Binning
  - SAHが最小になるSplitting planeで分割
  - Subtree構築を並列化
- Flattening
  - Depth-first orderでノードをメモリにレイアウト
- Traversal
  - Rayの方向と分割軸をみて子ノードの Traversal orderを決定 (Front-to-back traversal)

Wide BVH

Binary BVHをWide BVHに変換する

- Top-down construction
  - Binary BVHと同じコードを使用
- Flattening
  - 2分木をn分木に展開
  - Surface areaが大きい子ノードを分割
    - 子ノードがn個になるまで行う
  - Depth-first orderでノードをメモリにレイアウト
- Traversal
  - Traversal orderを固定
  - 交差判定処理をSIMD化

Traversal order

実装時に試した方法

- Rayがヒットした距離順で子ノードのTraversal orderをソート
- Traversal前に分割軸で子ノードをソートしておいて、Rayの方向でTraversal orderを昇順か降順にする

- Traversal order固定にくらべて、いずれの方法もノードアクセス数自体は減らせたが、レンダリング時間は増えた

SIMD

Intel ISPCを使って実装

- 1 ray x n bounding boxesの交差判定
- 1 ray x n trianglesの交差判定

テスト

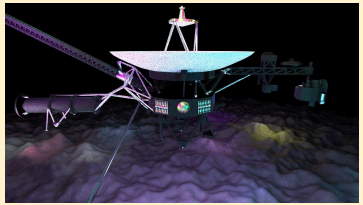
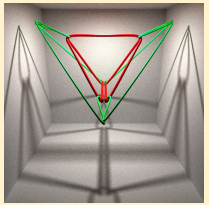
- Intel(R) Core(TM) Ultra 9 285K (3.70 GHz, 24コア)

- BVH2, BVH8, Embreeを比較

- テストシーンはRTCamp9, RTCamp10, RTCamp11のアセット

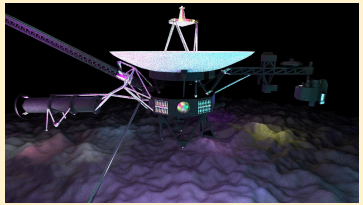
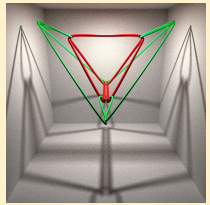
BVH構築

- BVH2, BVH8は1回Warm-up実行後に計測
- BVH8のリーフノードの最大プリミティブ数は、BVH2より大きい値を設定

	RTCamp9	RTCamp10	RTCamp11
	195,602 triangles 	569,496 triangles 	115,210 triangles 
BVH2	30.0 [ms]	83.5 [ms]	18.1 [ms]
BVH8	28.0 [ms]	74.2 [ms]	15.2 [ms]
Embree	7.3 [ms]	16.9 [ms]	4.7 [ms]
性能比(Embree/BVH8)	0.26x	0.23x	0.31x

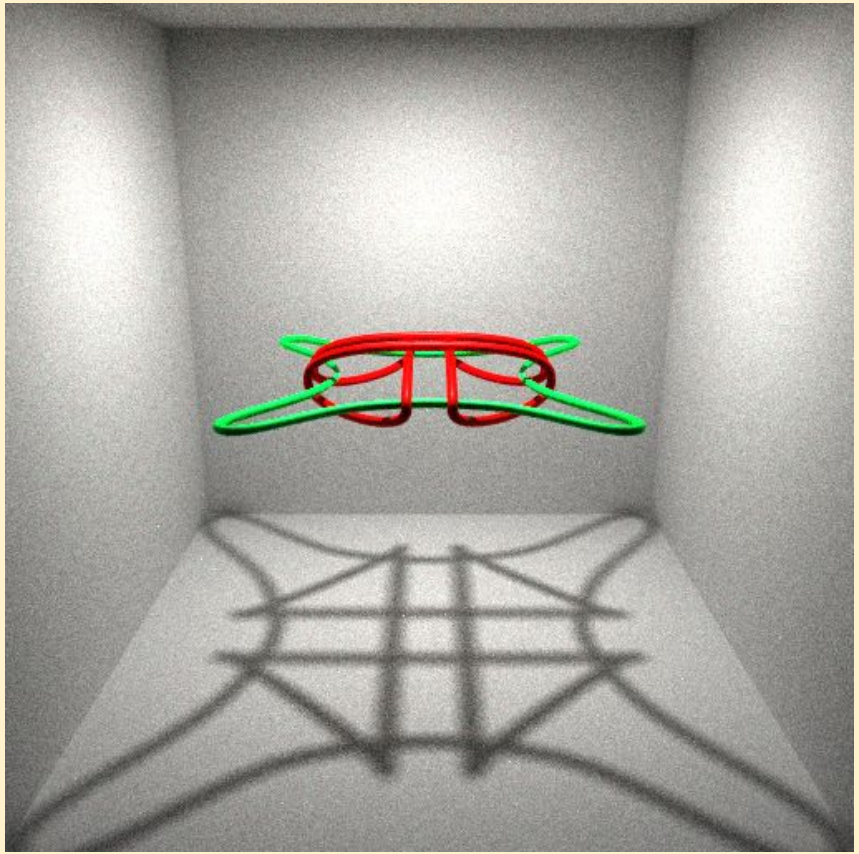
レンダリング

- 交差判定以外の処理も含む

	RTCamp9	RTCamp10	RTCamp11
64 spp	1280 x 720 	1280 x 720 	768 x 768 
BVH2	15.22 [s]	4.28 [s]	4.28 [s]
BVH8	9.88 [s]	2.40 [s]	2.49 [s]
Embree	7.65 [s]	2.02 [s]	2.21 [s]
性能比(Embree/BVH8)	0.77x	0.84x	0.89x

レンダラ設定

- CPU
- 解像度: 768 x 768
- サンプル数: 64
- フレーム数: 240
  - FPS: 24
  - アニメーション時間 : 10s
- ライト: 1 Disk Light
- BVH8



ライブラリ・アセット

ライブラリ

- Intel ISPC
- tinygltf
- fpng
- stb

アセット

- アーティスト提供のアセット
- 115,210ポリゴン
- モーフィングアニメーション