GPU Framework for Point Membership Classification of Geometric Models

Background and Motivation

We implement a 3D parallel solver for Point Membership Classification in CUDA. This solver is capable to check whether the randomly selected grid points lie inside or outside the geometry. It has the applications in the field of simulation such as immersed boundary analysis methods; in the field of computer graphics such as rendering photorealistic pictures or adding "fancy" effects in animation. Since this evaluation involves large computation cost, we implement this algorithm using GPU framework. All results were computed on a PC equipped with a 3.20 GHz Intel i7 CPU and NVIDIA GeForce GTX 1080 Ti. We also ran the code on Pittsburgh Supercomputing Center with the machine specificaiton as follows
CPU:Intel Xeon E5-2683 v4 and GPU: NVIDIA Tesla P100

About Point Membership Classification (PMC)

PMC is used to define the representations for sets in computational geometry that determines whether a point is inside a geometry. The standard method for deciding if a point is in the interior is to create a ray starting at the point and aiming in some selected direction, and then to count the number of crossing intersections with the boundary representations of the geometry.

About Graphics Processing Unit (GPU) Framework

GPU is a chip processor primarily used to manage and boost the performance of video and graphics. Nowadays, GPU has become an important part of High-performance computing platforms. They are being used to accelerate computational workloads. Compared to CPU, GPU is optimized for taking a huge amount of data and performing the same operation over and over very quickly. Our parallel implementation demonstrates a hundred times faster speed. We use it to create a real-time point membership classification

About Ray Tracing

Ray tracing is a rendering technique in computer graphics which simulating the effect of light. The technique is capable of producing a very high degree of visual pictures or animation. Here we use ray tracing to determine whether a point is inside a geometry. Each ray will be used to create an intersection with some subset of the geometry. That intersection information will be combined to check the final result.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Background and Motivation

We implement a 3D parallel solver for Point Membership Classification in CUDA. This solver is capable to check whether the randomly selected grid points lie inside or outside the geometry. It has the applications in the field of simulation such as immersed boundary analysis methods; in the field of computer graphics such as rendering photorealistic pictures or adding "fancy" effects in animation. Since this evaluation involves large computation cost, we implement this algorithm using GPU framework. All results were computed on a PC equipped with a 3.20 GHz Intel i7 CPU and NVIDIA GeForce GTX 1080 Ti. We also ran the code on Pittsburgh Supercomputing Center with the machine specificaiton as follows
CPU:Intel Xeon E5-2683 v4 and GPU: NVIDIA Tesla P100

About Point Membership Classification (PMC)

PMC is used to define the representations for sets in computational geometry that determines whether a point is inside a geometry. The standard method for deciding if a point is in the interior is to create a ray starting at the point and aiming in some selected direction, and then to count the number of crossing intersections with the boundary representations of the geometry.

About Graphics Processing Unit (GPU) Framework

GPU is a chip processor primarily used to manage and boost the performance of video and graphics. Nowadays, GPU has become an important part of High-performance computing platforms. They are being used to accelerate computational workloads. Compared to CPU, GPU is optimized for taking a huge amount of data and performing the same operation over and over very quickly. Our parallel implementation demonstrates a hundred times faster speed. We use it to create a real-time point membership classification

About Ray Tracing

Ray tracing is a rendering technique in computer graphics which simulating the effect of light. The technique is capable of producing a very high degree of visual pictures or animation. Here we use ray tracing to determine whether a point is inside a geometry. Each ray will be used to create an intersection with some subset of the geometry. That intersection information will be combined to check the final result.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Coding Team

- Aishwarya Pawar (arpawar@andrew.cmu.edu)
- Angran Li (angranl@andrew.cmu.edu)
- Yuxuan Yu (yuxuany1@andrew.cmu.edu)

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Mathematical Abstraction

Point Classification for Boundary Representation Solids

To check whether the randomly selected grid points lie inside or outside the geometry, we can change this problem into mathematical abstraction: Generate a ray from this point and counting how many times it intersects the solid's boundary. If the number of intersection points is odd, the point is inside the geometry; if it is even, the point is outside the geometry. Figure 1 shows a 2-D example. Also note that the ray can be randomly chosen and is arbitrary. For different rays, they may have different numbers of intersections with the boundary, but they will have the same properties, all odd or all even. In our computation, we generate ray parallel to axis. This will not affect the results.

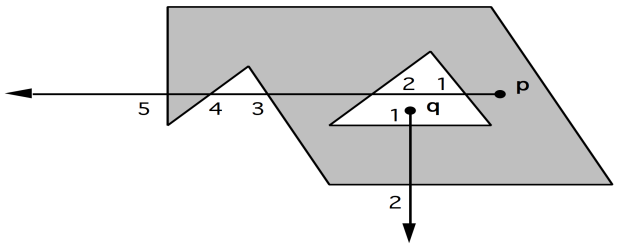


Fig.1 The ray from point p has 5 intersections with the polygon‚Äôs boundary, whereas the ray from point q has 2.

Intersection Abstraction

In the above section, we know that in order to check whether the randomly selected grid points lie inside or outside the geometry, we only need to know whether the number of intersection points is odd, or even. This section shows how to turn intersection into mathematical abstraction. We use triangle mesh as an example. It consists of two parts as follows:

- Find projection point on the plane a triangle lies on
- Find whether the projection point is inside or outside the triangle.

If the projection point is inside the triangle, we will know that the ray intersects this triangle. We will loop this operation for all the triangle and counting how many times it intersects the triangles of the solid's boundary. If the number of intersection points is odd, the point is inside the geometry; if it is even, the point is outside the geometry.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

File Format

Raw Geometry (raw)

Raw geometry files are simple ASCII files used to represent triangle meshes. See figure 2 for a simple raw file example. The basic file structure is as follows.

```
1  <numvertices><numtriangles>
2  <vertex 0>
3  .
4  .
5  .
6  <vertex n>
7  <triangle 0>
8  .
9  .
10  .
11  <triangle n>
```

The vertices section can be as follows:

```
*  Raw (type: float)
*  <vertX><vertY><vertZ>
```

```
3    1
0.0  0.0  0.0
1.0  0.0  0.0
0.0  1.0  0.0
0    1    2
```

Fig.2 An example of .raw file.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Point Classification Result

The left side figure is the input geometry and the right side figure is the output result.

Bunny

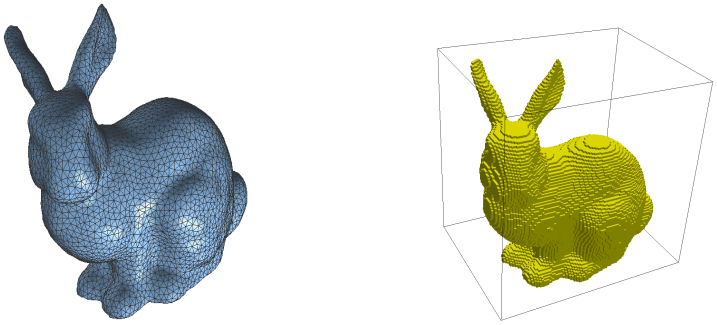


Fig.5 Bunny example.

Eight

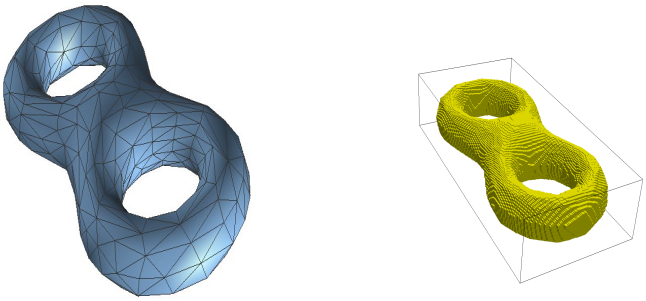


Fig.6 Eight example.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Reference

1 Klein, Fritz. "A new approach to point membership classification in B-rep solids." IMA International Conference on Mathematics of Surfaces. Springer, Berlin, Heidelberg, 2009style="font-weight:bold" .
2 Wikipedia contributors. "Graphics processing unit." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 19 Apr. 2019. Web. 7 May. 2019.
3 Wikipedia contributors. "Ray tracing (graphics)." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 7 May. 2019. Web. 7 May. 2019.
4 Requicha, Ari. "Geometric modeling: A first course." (1996).
5 Hsu, Ming-Chen, et al. "Direct immersogeometric fluid flow analysis using B-rep CAD models." Computer Aided Geometric Design 43 (2016): 143-158.
6 Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., ... & Roskies, R. (2014). XSEDE: accelerating scientific discovery. Computing in Science & Engineering, 16(5), 62-74.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Background and Motivation

We implement a 3D parallel solver for Point Membership Classification in CUDA. This solver is capable to check whether the randomly selected grid points lie inside or outside the geometry. It has the applications in the field of simulation such as immersed boundary analysis methods; in the field of computer graphics such as rendering photorealistic pictures or adding "fancy" effects in animation. Since this evaluation involves large computation cost, we implement this algorithm using GPU framework. All results were computed on a PC equipped with a 2.66 GHz Intel X5500 CPU and NVIDIA GeForce GTX 1080 Ti.

About Point Membership Classification (PMC)

PMC is used to define the representations for sets in computational geometry that determines whether a point is inside a geometry. The standard method for deciding if a point is in the interior is to create a ray starting at the point and aiming in some selected direction, and then to count the number of crossing intersections with the boundary representations of the geometry.

About Graphics Processing Unit (GPU) Framework

GPU is a chip processor primarily used to manage and boost the performance of video and graphics. Nowadays, GPU has become an important part of High-performance computing platforms. They are being used to accelerate computational workloads. Compared to CPU, GPU is optimized for taking a huge amount of data and performing the same operation over and over very quickly. Our parallel implementation demonstrates a hundred times faster speed. We use it to create a real-time point membership classification

About Ray Tracing

Ray tracing is a rendering technique in computer graphics which simulating the effect of light. The technique is capable of producing a very high degree of visual pictures or animation. Here we use ray tracing to determine whether a point is inside a geometry. Each ray will be used to create an intersection with some subset of the geometry. That intersection information will be combined to check the final result.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Mathematical Abstraction

Point Classification for Boundary Representation Solids

To check whether the randomly selected grid points lie inside or outside the geometry, we can change this problem into mathematical abstraction: Generate a ray from this point and counting how many times it intersects the solid's boundary. If the number of intersection points is odd, the point is inside the geometry; if it is even, the point is outside the geometry. Figure 1 shows a 2-D example. Also note that the ray can be randomly chosen and is arbitrary. For different rays, they may have different numbers of intersections with the boundary, but they will have the same properties, all odd or all even. In our computation, we generate ray parallel to axis. This will not affect the results.

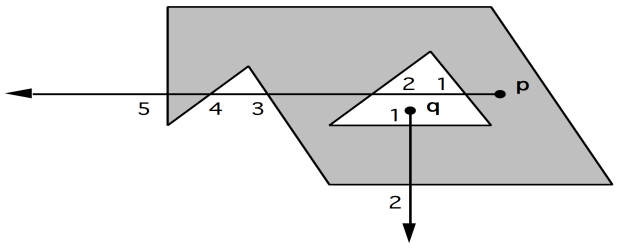


Fig.1 The ray from point p has 5 intersections with the polygon‚Äôs boundary, whereas the ray from point q has 2.

Intersection Abstraction

In the above section, we know that in order to check whether the randomly selected grid points lie inside or outside the geometry, we only need to know whether the number of intersection points is odd, or even. This section shows how to turn intersection into mathematical abstraction. We use triangle mesh as an example. It consists of two parts as follows:

- Find projection point on the plane a triangle lies on
- Find whether the projection point is inside or outside the triangle.

If the projection point is inside the triangle, we will know that the ray intersects this triangle. We will loop this operation for all the triangle and counting how many times it intersects the triangles of the solid's boundary. If the number of intersection points is odd, the point is inside the geometry; if it is even, the point is outside the geometry.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Flowchart

The Figure shows the flowchart of the raytracing algorithm.

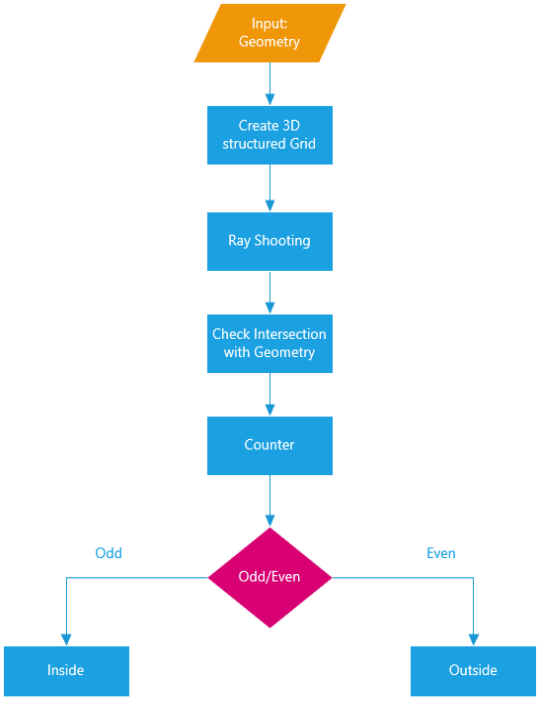


Fig. The flowchart of the raytracing algorithm.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Numerical Method

This section will show the numerical method used for the mathematical abstraction.

Find projection point on the plane which the triangle lies on

- Get the normal of the chosen triangle

, where Nx, Ny, Nz are the normal of the trangle, Ux is the x coordinate of one edge of the triangle.

- Get the plane of the chosen triangle lies on:

- Get the ray parametric equation

- Obtain the parameter value

, where N=[Nx, Ny, Nz] are the normal of the trangle.

- Obtain the projection points

Find whether the projection point is inside or outside the triangle

- Dot and Cross Product

Value
where is the ith edge of triangle and is the vector from the projection point to the vertices of the triangle. If value larger than 0 for all cases, the projection point is inside the triangle.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Challenge

There are several special cases we need to consider

The Ray And The Triangle Are Parallel

If the ray and the plane which the triangle lies on are parallel, it will be hard to determine whether they will intersect. To handle that case, we need to check whether the triangle's normal and the ray's direction is perpendicular.

Geometry limitation

The current algorithm only works for closed surfaces. The main problem of not closed surface is the counting step of the algorithm. The times it intersects the triangles of the solid's boundary will not fulfill the requirements of the algorithm that if the number of intersection points is odd, the point is inside the geometry; if it is even, the point is outside the geometry. This can be prevented if we checked whether it is closed surface in prepossessing step.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

File Format

Raw Geometry (raw)

Raw geometry files are simple ASCII files used to represent triangle meshes. See figure 2 for a simple raw file example. The basic file structure is as follows.

```
1  <numvertices><numtriangles>
2  <vertex 0>
3  .
4  .
5  .
6  <vertex n>
7  <triangle 0>
8  .
9  .
10  .
11  <triangle n>
```

The vertices section can be as follows:

```
*  Raw (type: float)
*  <vertX><vertY><vertZ>
```

```
3    1
0.0  0.0  0.0
1.0  0.0  0.0
0.0  1.0  0.0
0    1    2
```

Fig.2 An example of .raw file.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

File Format

VTK output (vtk)

VTK files are also ASCII files used to represent ray-tracing results. See figure 3 for a simple vtk output example. The basic file structure is as follows.

```
1 # vtk DataFile Version 3.0
2 VTK from C
3 ASCII
4 DATASET STRUCTURED_GRID
5 DIMENSIONS <dimX><dim><dim>
6 POINTS <numPoint> float
7 <vertX><vertY><vertZ>
8  .
9  .
10  .
11  POINT_DATA <numPoint> .
12  SCALARS title float .
13 LOOKUP_TABLE default .
14  0 or 1
15  .
16  .
17  .
```

In point data, if the value is 0, it means it is outside the geometry. If the value is 1, it means it is inside the geometry.

```
# vtk DataFile Version 3.0
VTK from C
ASCII
DATASET STRUCTURED_GRID
DIMENSIONS 3 1 1
POINTS 3 float
0.0   0.0   0.0
1.0   0.0   0.0
0.0   1.0   0.0

POINT_DATA 3
SCALARS title float
LOOKUP_TABLE default

0
0
1
```

Fig.3 An example of vtk file.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Complie

Entire source depends on Standard Template Library (STL). Openmp and Cuda-9.1 are required for parallel parts. It is mainly testing with GCC-5.4

To compile the code

In ./src folder, run >> make will compile the code to excutable file kernel

The code has been compiled and tested on Ubuntu 16.04 with GCC 5.4.0 and CUDA 9.1.

The code was run on CMU qwe cluster with the machine specification:

- CPU: Intel(R) Core(TM) i7 CPU 960 @ 3.20GHz
- GPU: Nvidia GeForce GTX 1080 Ti

We also ran the code on Pittsburgh Supercomputing Center with the machine specificaiton as follows:

- CPU: Intel Xeon E5-2683 v4
- GPU: NVIDIA Tesla P100

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Point Classification Result

The left side figure is the input geometry and the right side figure is the output result.

Bunny

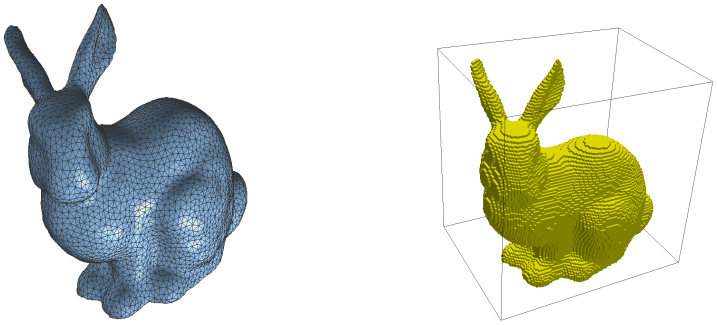


Fig.5 Bunny example.

Eight

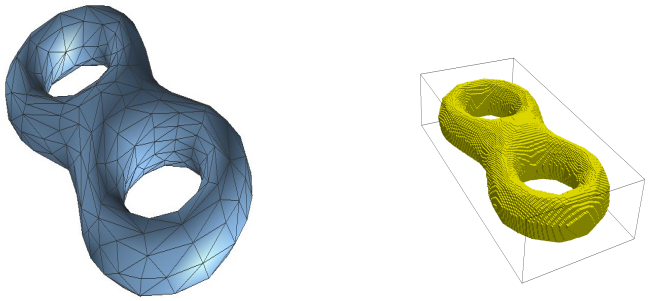


Fig.6 Eight example.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Point Classification Result

The left side figure is the input geometry and the right side figure is the output result.

Kitten

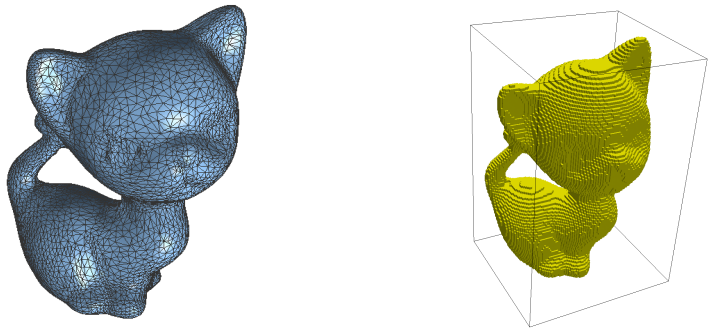


Fig.7 Kitten example.

Rod

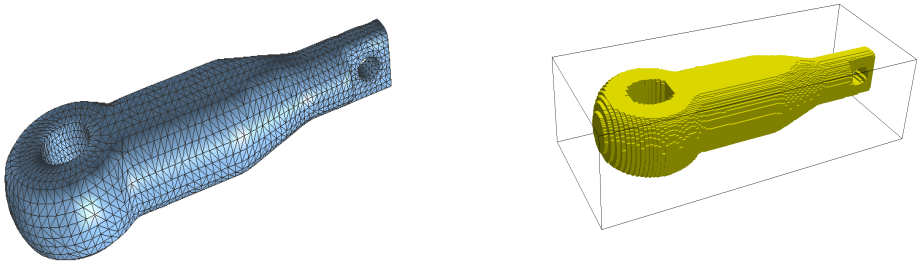


Fig.8 Rod example.

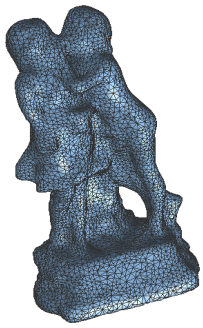
This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Point Classification Result

The left side figure is the input geometry and the right side figure is the output result.

Sculpture



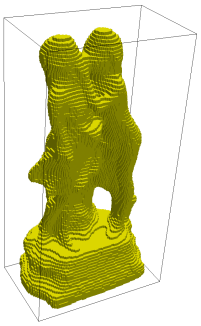


Fig.9 Sculpture example.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Execution Time Comparison

The tables shown are the comparison of execution Time of CPU and GPU implementation. Table 1 is for different model. Table 2 is for the kitten model with different resolution.

Table 1. Execution Time of CPU and GPU implementation.

	CPU time (s)	GPU time (s)	Speedup
kitten	294.21	0.0839	3506.674613
rod	123.53	0.0364	3393.681319
bunny	274.98	0.0712	3862.078652
sculpture	485.76	0.1274	3812.872841
eight	17.76	0.00836	2124.401914
resolution: 41*41*41			

Table 2. Execution Time of CPU and GPU of the kitten model with different resolution.

Resolution	kitten CPU time (s)	kitten GPU time (s)	Speedup
11	5.79	0.00905	639.7790055
21	39.96	0.0207	1930.434783
41	294.21	0.0839	3506.674613
81	2295.04	0.512	4482.5
161	17964.22	3.687	4872.313534

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).

GPU Framework for Point Membership Classification of Geometric Models

Reference

1 Klein, Fritz. "A new approach to point membership classification in B-rep solids." IMA International Conference on Mathematics of Surfaces. Springer, Berlin, Heidelberg, 2009style="font-weight:bold" .
2 Wikipedia contributors. "Graphics processing unit." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 19 Apr. 2019. Web. 7 May. 2019.
3 Wikipedia contributors. "Ray tracing (graphics)." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 7 May. 2019. Web. 7 May. 2019.
4 Requicha, Ari. "Geometric modeling: A first course." (1996).
5 Hsu, Ming-Chen, et al. "Direct immersogeometric fluid flow analysis using B-rep CAD models." Computer Aided Geometric Design 43 (2016): 143-158.
6 Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., ... & Roskies, R. (2014). XSEDE: accelerating scientific discovery. Computing in Science & Engineering, 16(5), 62-74.

This is Raytracer program created by Aishwarya Pawar (arpawar@andrew.cmu.edu), Angran Li (angranl@andrew.cmu.edu), Yuxuan Yu (yuxuany1@andrew.cmu.edu).