Lab 2
Voxelizer

CS581 Computational Fabrication (Spring 2025)

Shape Lab

BOSTON UNIVERSITY

Updates on 3D printers

2 of the Makerbots in CDS 755 work fine

Use the one closest to the door, or the one in the middle

Worst case, you can always use the Bambu printers at EPIC

Reminders

Assignment 1: 3DBenchy Test Print (**Due: 2/13 in class**)

- ***Bring 3DBenchy to class***

Assignment 2: Voxelizer (**Due: 2/13 on Gradescope**)

- Will go over today!

Folder Structure

`data/`

- Input/output meshes (.obj)

`sample/`

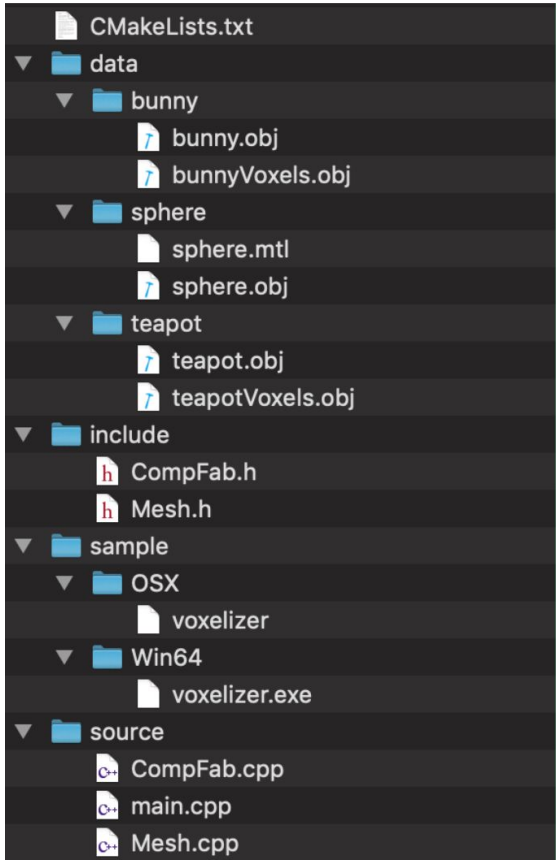
- Precompiled sample executables

`include/`

- Header files (.h)

`source/`

- Source code (.cpp), ***you will be editing this***



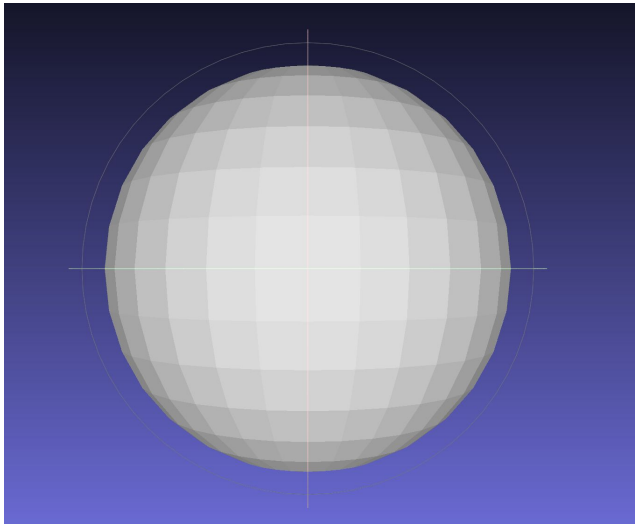
Shape Lab

BOSTON UNIVERSITY

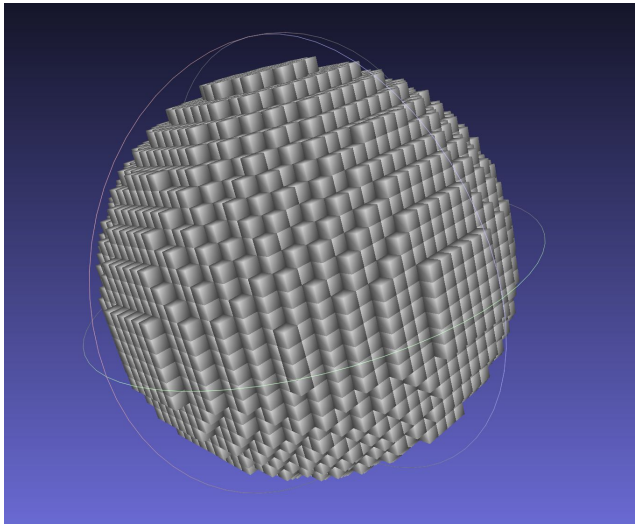
Running the Code

```
>>> cd /path/to/executable

>>> ./voxelizer ../../data/sphere/sphere.obj ../../data/sphere/sphere_output.obj
```



sphere.obj



sphere_output.obj

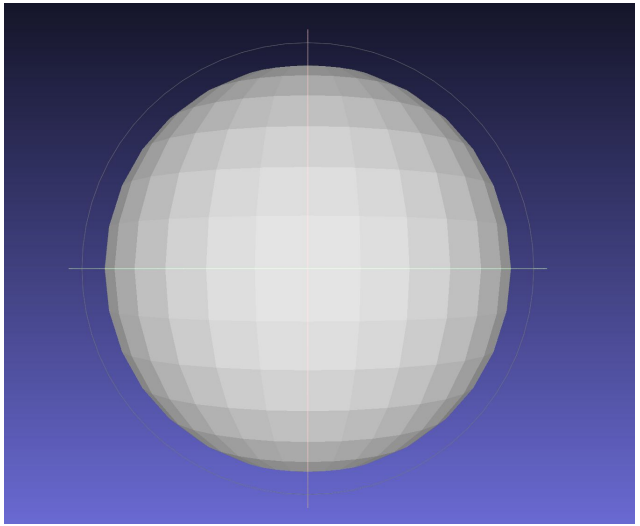
Shape Lab

BOSTON UNIVERSITY

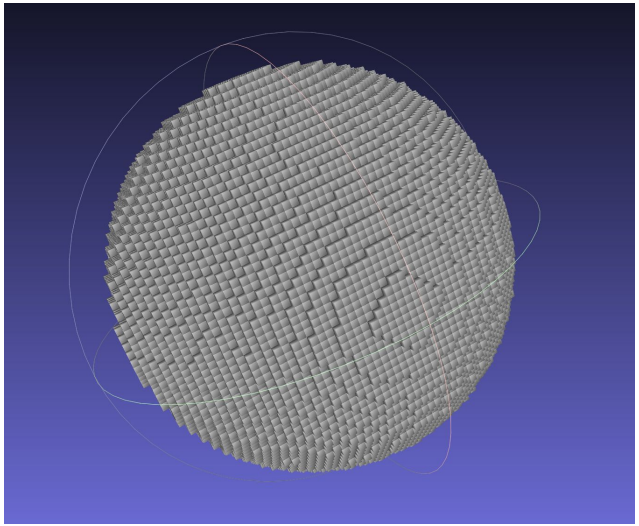
Running the Code

```
>>> cd /path/to/executable

>>> ./voxelizer ../../data/sphere/sphere.obj ../../data/sphere/sphere_output64.obj
```



sphere.obj



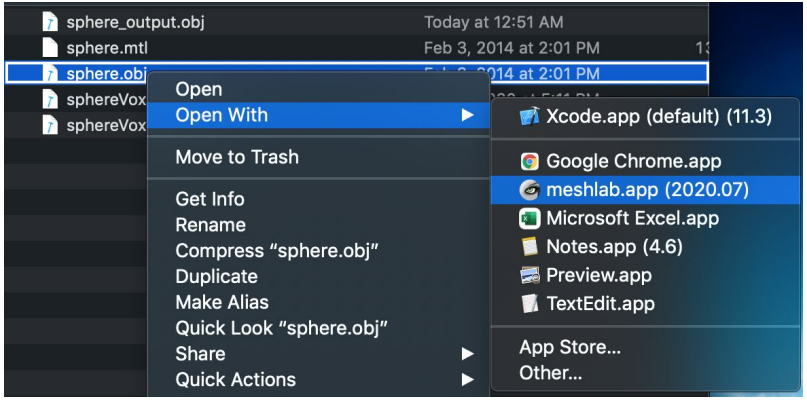
sphere_output64.obj

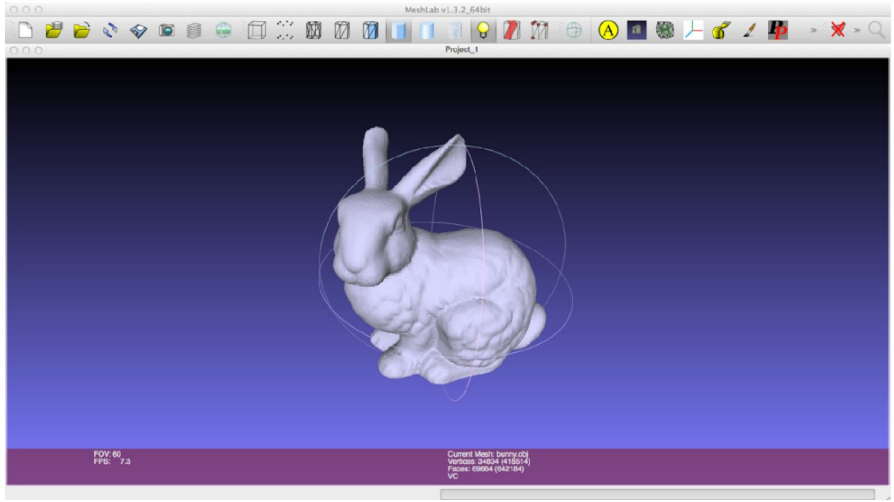
Shape Lab

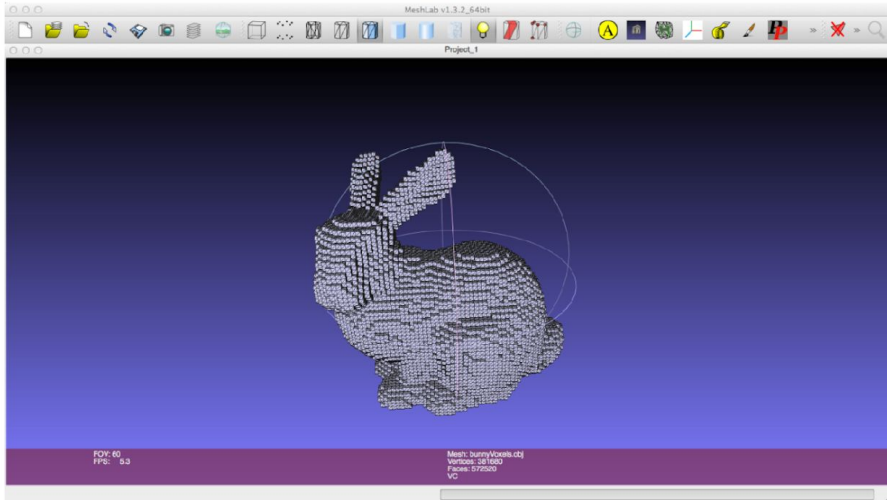
BOSTON UNIVERSITY

Viewing Meshes

MeshLab: https://www.meshlab.net/#download







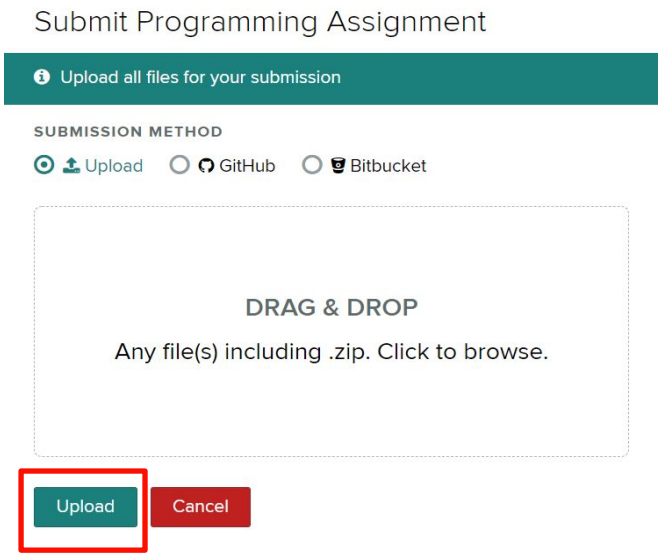
Shape Lab

BOSTON UNIVERSITY

Submission

Submit to Gradescope

- Please upload: **code**, **executable**, **report**, **additional files**



Shape Lab

BOSTON UNIVERSITY

Submission - Report (Also Listed on Blackboard)

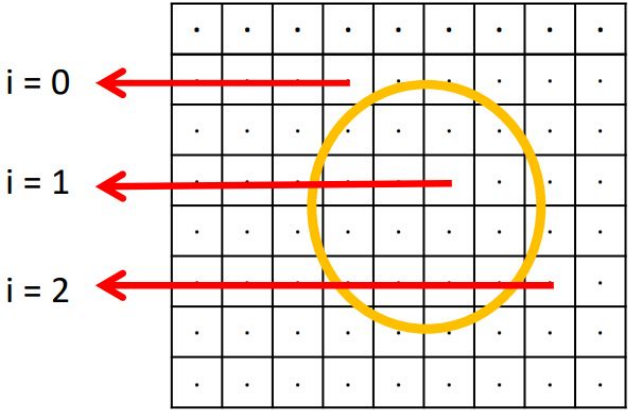
1. Results for 32x32x32 resolution and 64x64x64 resolution.
2. References (books, slides, etc.)
3. Any known bugs in your code. Describe your thought process (possible partial credit)
4. Comments (e.g. overview and understanding of this assignment, implementation details, etc.)

Ray Casting

Voxelization is based on ray casting

- 2D: For each pixel, cast a ray and count the number of intersections with the contour.

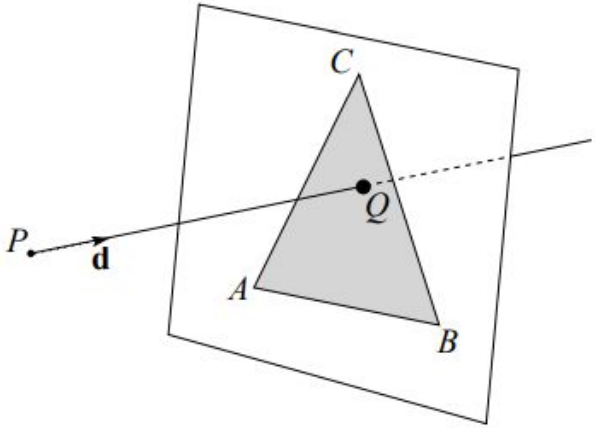
- 3D: For each voxel, cast a ray and count the number of intersections with surface triangles.



Ray Triangle Intersection

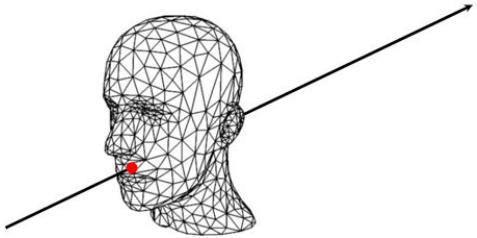
Ray Triangle Intersection

- Determine if Q lies inside of the triangle



Ray Mesh Intersection

- For each triangle in mesh, perform Ray Triangle Intersection

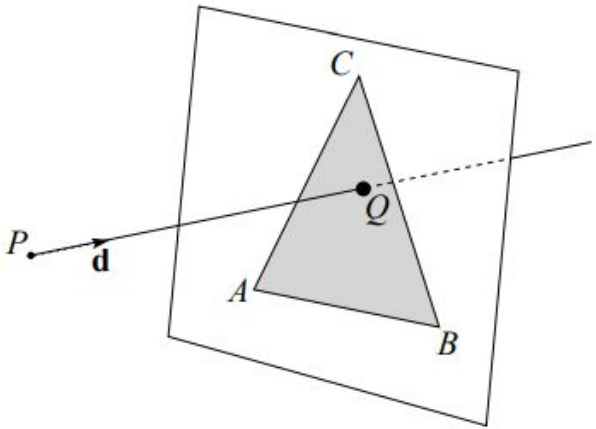


Shape Lab

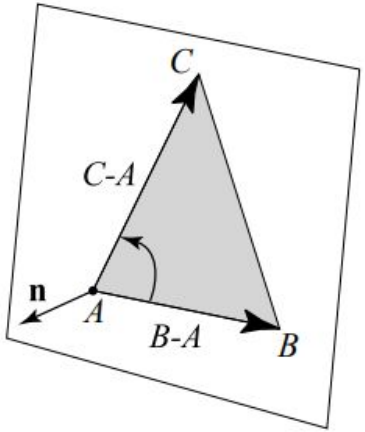
BOSTON UNIVERSITY

Ray Triangle Intersection

Ray: $\mathbf{r}(t) = \mathbf{P} + t\mathbf{d}$



Plane normal: $\mathbf{n} = \dfrac{(B-A)\times(C-A)}{\|(B-A)\times(C-A)\|}$



Find t^* that gives point on plane, $\mathbf{Q} = \mathbf{r}(t^*)$:

$$\big(\mathbf{r}(t^*) - \mathbf{C}\big)\cdot\mathbf{n} = 0$$

$$(\mathbf{P} + t^*\mathbf{d} - \mathbf{C})\cdot\mathbf{n} = 0$$

$$\frac{(\mathbf{C}-\mathbf{P})\cdot\mathbf{n}}{\mathbf{d}\cdot\mathbf{n}} = t^*$$

Could use any triangle vertex

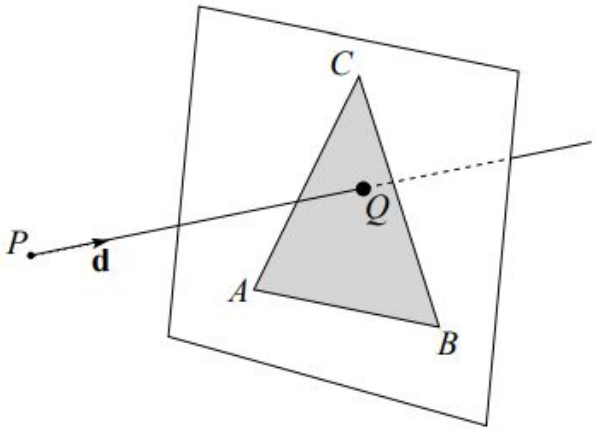
Shape Lab

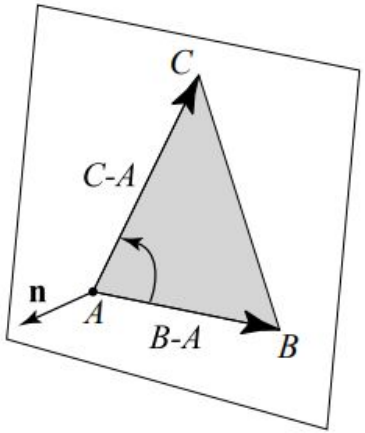
BOSTON UNIVERSITY

Ray Triangle Intersection

$\mathbf{Q}$ lies inside triangle if:

$$(\mathbf{B} - \mathbf{A}) \times (\mathbf{Q} - \mathbf{A}) \cdot \mathbf{n} > 0,$$
$$(\mathbf{C} - \mathbf{B}) \times (\mathbf{Q} - \mathbf{B}) \cdot \mathbf{n} > 0,$$
$$(\mathbf{A} - \mathbf{C}) \times (\mathbf{Q} - \mathbf{C}) \cdot \mathbf{n} > 0.$$





Code

```
//Ray-Triangle Intersection
//Returns 1 if triangle and ray intersect, 0 otherwise
int rayTriangleIntersection(CompFab::Ray &ray, CompFab::Triangle &triangle)
{
    /********* ASSIGNMENT *********/
    /* Ray-Triangle intersection test: Return 1 if ray intersects triangle,
     * 0 otherwise */

    return 0;

}
```

Given a ray and a triangle, determine if they intersect.

Code

```
//Number of intersections with surface made by a ray originating at voxel and cast in direction.
int numSurfaceIntersections(CompFab::Vec3 &voxelPos, CompFab::Vec3 &dir)
{

    unsigned int numHits = 0;

    /********* ASSIGNMENT *********/
    /* Check and return the number of times a ray cast in direction dir,
     * from voxel center voxelPos intersects the surface */

    return numHits;
}
```

Given a ray direction and a voxel, determine how many times the ray intersects with the surface

- Loop over all the surface triangles

Code

```
int main(int argc, char **argv)
{

    unsigned int dim = 32; //dimension of voxel grid (e.g. 32x32x32)

    //Load OBJ
    if(argc < 3)
    {
        std::cout<<"Usage: Voxelizer InputMeshFilename OutputMeshFilename \n";
        exit(0);
    }

    std::cout<<"Load Mesh : "<<argv[1]<<"\n";
    loadMesh(argv[1], dim);



    //Cast ray, check if voxel is inside or outside
    //even number of surface intersections = outside (OUT then IN then OUT)
    // odd number = inside (IN then OUT)
    CompFab::Vec3 voxelPos;
    CompFab::Vec3 direction(1.0,0.0,0.0);

    /********* ASSIGNMENT *********/
    /* Iterate over all voxels in g_voxelGrid and test whether they are inside our outside of the
     * surface defined by the triangles in g_triangleList */

    //Write out voxel data as obj
    saveVoxelsToObj(argv[2]);

    delete g_voxelGrid;
}
```

Main:

- Not your task: Load/save mesh
- Your task: Determine resolution (32 or 64), Loop over all voxels and determine if inside/outside

Shape Lab

BOSTON UNIVERSITY

Code Structure

```
using namespace CompFab
```

- Vectors: `CompFab::Vec3 vector`
  - Contains three doubles (`vector.m_x`, `vector.m_y`, `vector.m_z`).
- Rays: `CompFab::Ray ray`
  - Contains two vectors (`ray.m_origin`, `ray.m_direction`).
- Triangles: `CompFab::Triangle triangle`
  - Contains three vectors specifying vertices (`triangle.m_v1`, `triangle.m_v2`, `triangle.m_v3`).
- Dot product: `v1 * v2`
- Cross product: `v1 % v2`