Information Visualization

W12: Exercise - Implementation of Isosurface Extraction

Graduation School of System Informatics

Department of Computational Science

Naohisa Sakamoto, Akira Kageyama

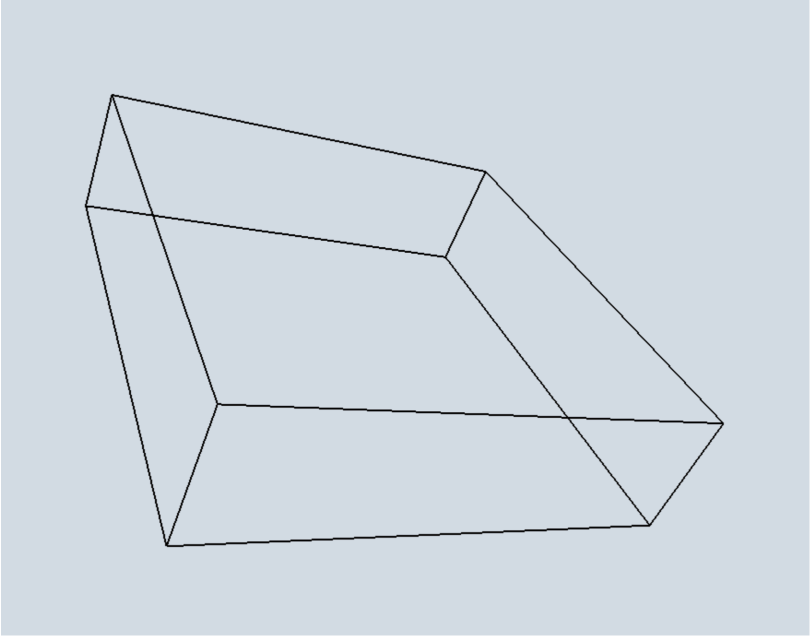
May.24, 2017

Schedule

W01	4/11	Guidance
W02	4/12	Setup
W03	4/18	Introduction to Data Visualization
W04	4/19	CG Programming
W05	4/25	Rendering Pipeline
W06	4/26	Coordinate Systems and Transformations
W07	5/09	Shading
W08	5/10	Shader Programming
W09	5/16	Visualization Pipeline
W10	5/17	Data Model and Transfer Function
W11	5/23	Scalar Data Visualization 1 (Isosurface Extraction)
W12	5/24	Implementation of Isosurface Extraction
W13	5/30	Scalar Data Visualization 2 (Volume Rendering)
W14	5/31	Implementation of Volume Rendering
W15	6/06	Student Presentations

Ex01: Bounding box

- Draw a bounding box for a structured volume data named as KVS.LobsterData.
  - Download
    - w12_main_ex01.js
    - w12_index_ex01.html
    - Bounds.js
  - Open
    - w12_index_ex01.html



Ex01: Bounding box

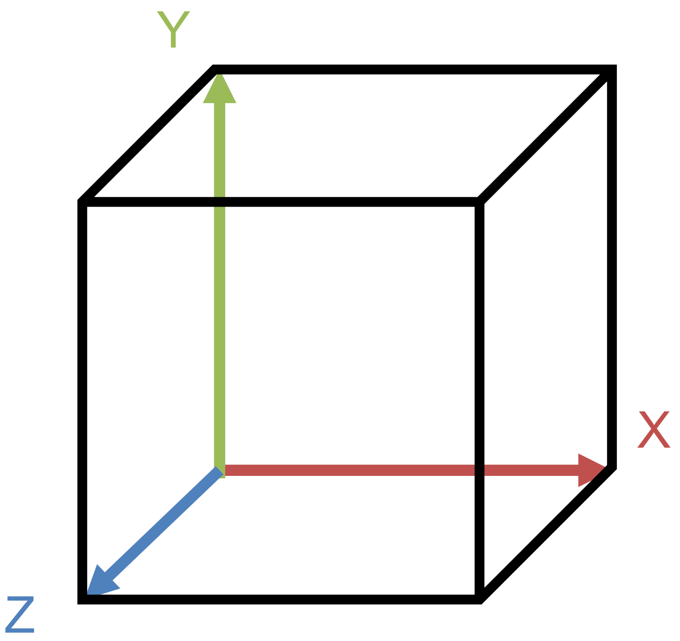
- Lobster data
  - KVS.StructuredVolumeObject

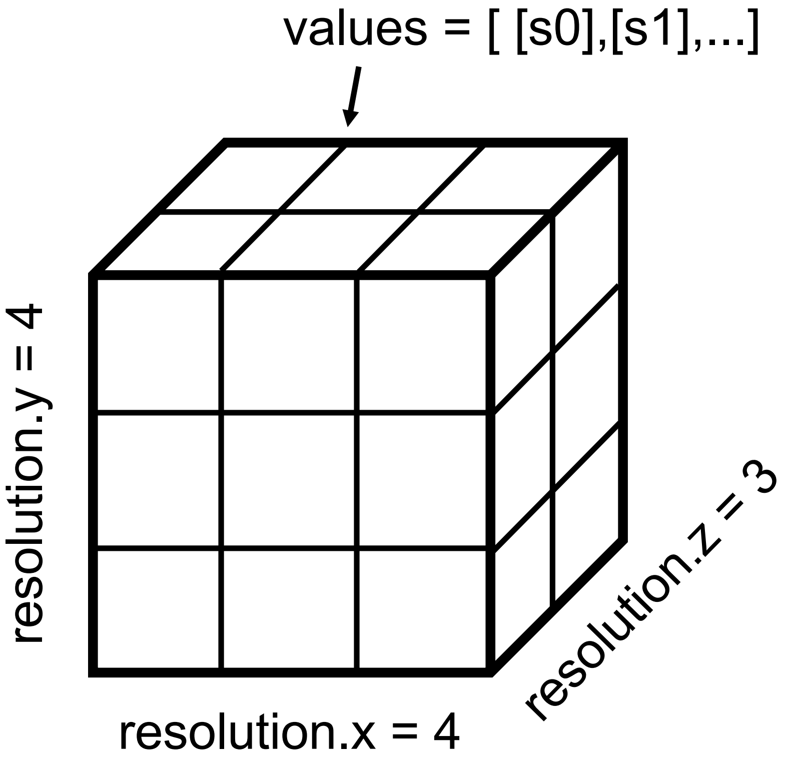
```
// Constructor
KVS.StructuredVolumeObject = function()
{
    this.resolution = new KVS.Vec3();
    this.values = [];
    this.min_coord = new KVS.Vec3();
    this.max_coord = new KVS.Vec3();
    this.min_value = 0;
    this.max_value = 0;
};
```

https://github.com/naohisas/KVS.js/blob/master/Source/Core/Object/StructuredVolumeObject.js

Ex01: Bounding box

- Lobster data
  - KVS.StructuredVolumeObject





Ex01: Bounding box

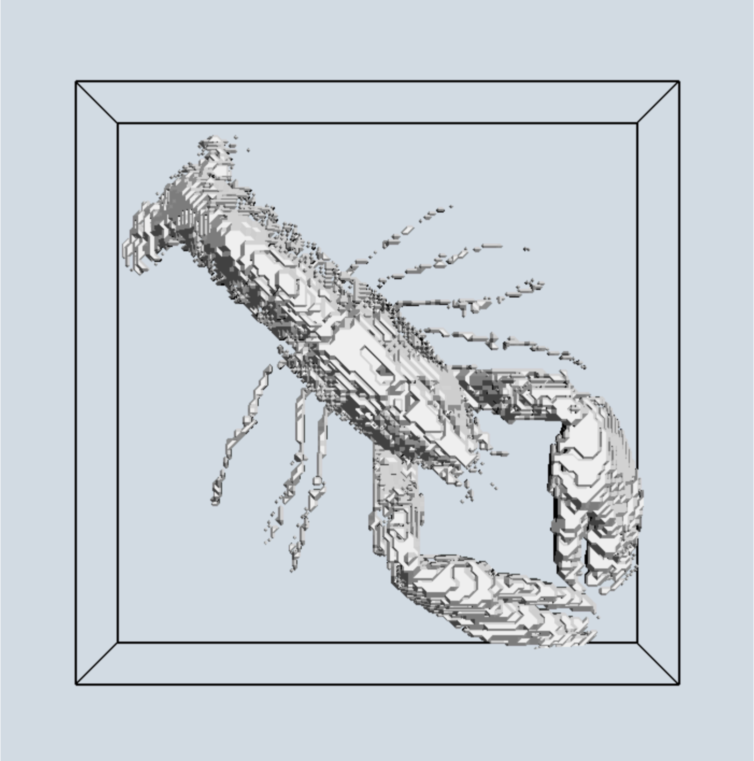
- KVS.THREEScreen

```
// Constructor
KVS.THREEScreen = function()
{
    this.width = 0;
    this.height = 0;
    this.scene = undefined;      // THREE.Scene
    this.camera = undefined;     // THREE.PerspectiveCamera
    this.light = undefined;      // THREE.DirectionalLight
    this.renderer = undefined;   // THREE.WebGLRenderer
    this.trackball = undefined;  // THREE.TrackballControls
};
```

https://github.com/naohisas/KVS.js/blob/master/Source/THREE/THREEScreen.js

Ex02: Isosurface Extraction

- Extract isosurfaces from the lobster data and draw it with the bounding box.
  - – Download
    - w12_main_ex02.js
    - w12_index_ex02.html
    - Isosurfaces.js
  - – Open
    - w12_index_ex02.html



Ex02: Isosurface Extraction

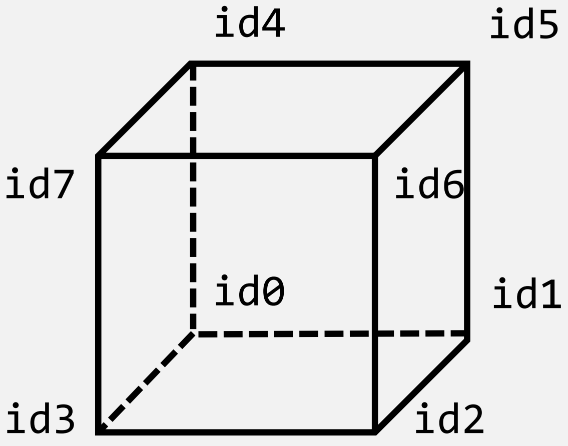
- Marching process
  - For each cell

```
var cell_index = 0;
var counter = 0;
for ( var z = 0; z < volume.resolution.z - 1; z++ )
{
    for ( var y = 0; y < volume.resolution.y - 1; y++ )
    {
        for ( var x = 0; x < volume.resolution.x - 1; x++ )
        {
            // Extract surfaces for a cube
        }
    }
}
```

Ex02: Isosurface Extraction

- Cell node indices

```
var lines = volume.resolution.x;
var slices = volume.resolution.x * volume.resolution.y;

var id0 = cell_index;
var id1 = id0 + 1;
var id2 = id1 + lines;
var id3 = id0 + lines;
var id4 = id0 + slices;
var id5 = id1 + slices;
var id6 = id2 + slices;
var id7 = id3 + slices;

return [ id0, id1, id2, id3, id4, id5, id6, id7 ];
```



Ex02: Isosurface Extraction

- Table index

```
var s0 = volume.values[ indices[0] ][0];
var s1 = volume.values[ indices[1] ][0];
var s2 = volume.values[ indices[2] ][0];
...
var s7 = volume.values[ indices[7] ][0];

var index = 0;                          //   0 = 0000,0000
if ( s0 > isovalue ) { index |=   1; }  //   1 = 0000,0001
if ( s1 > isovalue ) { index |=   2; }  //   2 = 0000,0010
if ( s2 > isovalue ) { index |=   4; }  //   4 = 0000,0100
...
if ( s7 > isovalue ) { index |= 128; }  // 128 = 1000,0000

return index;
```

Ex02: Isosurface Extraction

- For each triangle face

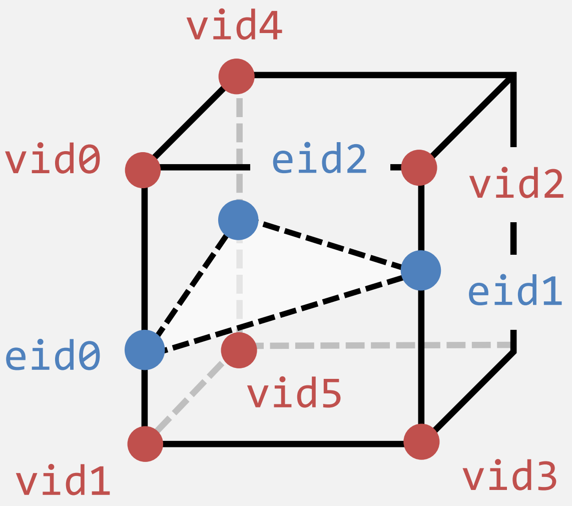
```
for ( var j = 0; lut.edgeID[index][j] != -1; j += 3 )
{
    // Extract a triangle face
}
```

```
KVS.MarchingCubesTable = function()
{
    this.edgeID = [
        [-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1],
        [ 0,  8,  3, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1],
        ...
        [ 5, 10,  2,  5,  2,  4,  1,  9,  2,  9,  4,  2, -1, -1, -1, -1],
        ...
    ];
    ...
};
```

Ex02: Isosurface Extraction

- Edges and its end-points

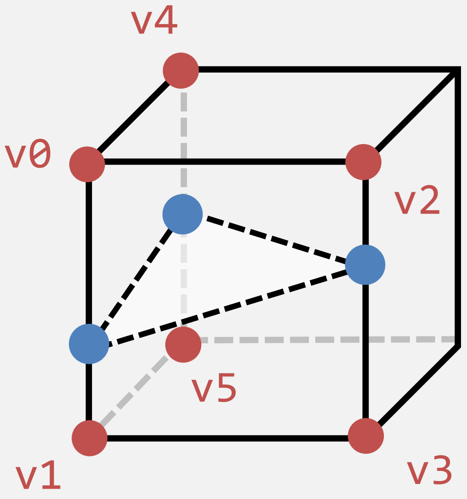
```
var eid0 = lut.edgeID[index][j];
var eid1 = lut.edgeID[index][j+2];
var eid2 = lut.edgeID[index][j+1];

var vid0 = lut.vertexID[eid0][0];
var vid1 = lut.vertexID[eid0][1];
var vid2 = lut.vertexID[eid1][0];
var vid3 = lut.vertexID[eid1][1];
var vid4 = lut.vertexID[eid2][0];
var vid5 = lut.vertexID[eid2][1];
```



Ex02: Isosurface Extraction

- Vertex coordinates of the end-points

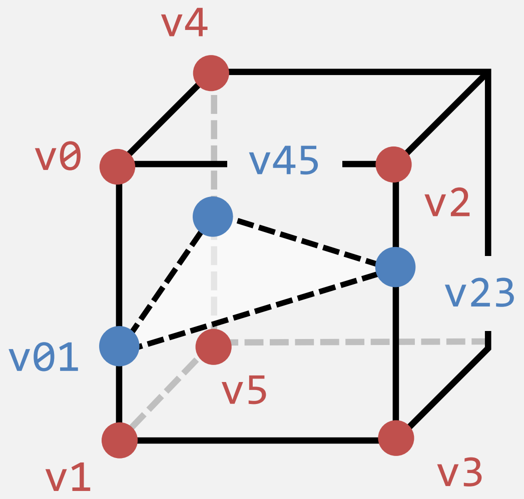
```
var v0 = new THREE.Vector3( x + vid0[0], y + vid0[1], z + vid0[2] );
var v1 = new THREE.Vector3( x + vid1[0], y + vid1[1], z + vid1[2] );
var v2 = new THREE.Vector3( x + vid2[0], y + vid2[1], z + vid2[2] );
var v3 = new THREE.Vector3( x + vid3[0], y + vid3[1], z + vid3[2] );
var v4 = new THREE.Vector3( x + vid4[0], y + vid4[1], z + vid4[2] );
var v5 = new THREE.Vector3( x + vid5[0], y + vid5[1], z + vid5[2] );
```



Ex02: Isosurface Extraction

- Vertex coordinates of the triangle face

```
var v01 = interpolated_vertex( v0, v1, isovalue );
var v23 = interpolated_vertex( v2, v3, isovalue );
var v45 = interpolated_vertex( v4, v5, isovalue );

geometry.vertices.push( v01 );
geometry.vertices.push( v23 );
geometry.vertices.push( v45 );




var id0 = counter++;
var id1 = counter++;
var id2 = counter++;
geometry.faces.push( new THREE.Face3( id0, id1, id2 ) );
```



Task 1

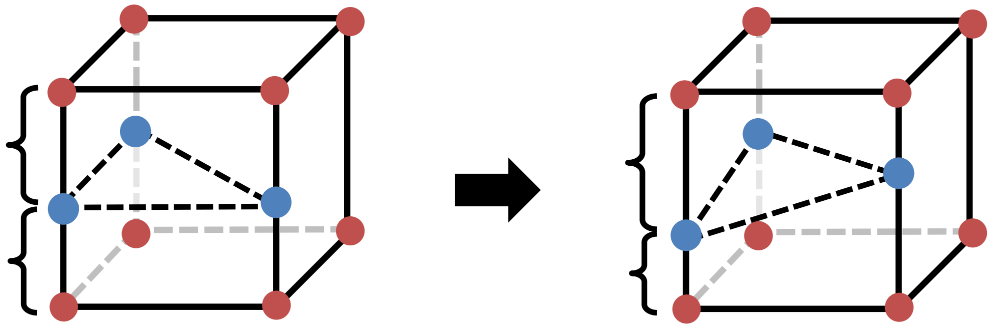
- Assign a color by using a transfer function (color map)
  - Modify the following code

```
material.color = new THREE.Color( "white" );
```

Task 2

- Interpolate vertices between the end-points of the edges in the extraction process of triangle faces.
  - Modify the following code

```
function interpolated_vertex( v0, v1, s )
{
    return new THREE.Vector3().addVectors( v0, v1 ).divideScalar( 2 );
}
```

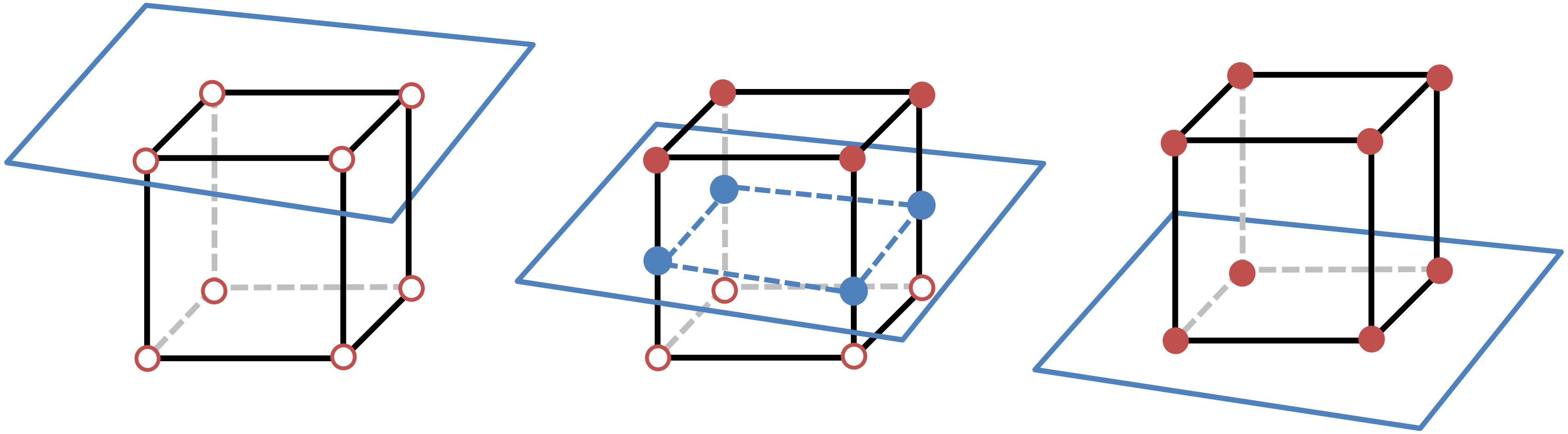


Task 3

- Apply shaders to the isosurface rendering

Advanced Task

- Implement slice plane extraction based on isosurface extraction algorithm.



Plane: $a x + b y + c z + d = 0$

● : $a x_i + b y_i + c z_i + d \geqq 0$

○ : $a x_i + b y_i + c z_i + d < 0$

Polling

- Take the poll
  - Student ID Number
  - Name
  - URL to Task 1
  - URL to Task 2
  - URL to Task 3
  - URL to Task 4 (advanced)