Realtime Graphics Lab Book

# Week 1 – Lab A

Date: 3rd Oct 2023

## Exercise 0

### Question: Select Tutorial04 as the start-up project. Open “Tutorial04.cpp” and familiar yourself with the program by walking through the code. Compile and run, you should see a coloured cube in rotation. If you do NOT want to see the cube in rotation, simply go to the line g\_World = XMMatrixRotationY( t ); and change the variable t in XMMatrixRotationY( t ) with a fixed number, say, 0.5f. With current eye position, you are unable to see the top face of the cube. You can move the eye position slightly higher than the one specified in the program to make the top face of the cube visible. To change the eye position, go to the line XMVECTOR Eye = XMVectorSet( 0.0f, 1.0f, -5.0f, 0.0f ); and change the current eye position (0.0f, 1.0f, -5.0f) to, say, (0.0f, 2.5f, -5.0f)

### Solution:

XMVECTOR Eye = XMVectorSet( 0.0f, 2.5f, -5.0f, 0.0f );

### Test data:

n/a

### Sample output:

### A colorful cube on a blue background Description automatically generated

### Reflection:

*Changed the camera position to be a little higher.*

### Metadata:

### Further information:

## Exercise 1

Modify the vertex list in indices[] or modify the parameters in the DrawIndexed( ) to draw: (1) two  
triangles; (2) one face of the cube; (3) the four walls of the cube  
You may notice that only one face of the triangle and square face being drawn. This is because by  
default, the back face of the cube is culled out. You can specify the cull mode to be none by filling.

### Solution:

1. WORD indices[] =

{

0,5,4,

2,7,6,

};

1. WORD indices[] =

{

0,5,4,

1,5,0,

};

1. WORD indices[] =

{

0,5,4,

1,5,0,

3,4,7,

0,4,3,

1,6,5,

2,6,1,

2,7,6,

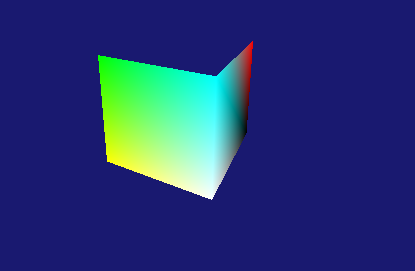
3,7,2,

};

### Sample output:

1. A group of triangles on a blue background

   Description automatically generated
2. A colorful triangle on a blue background

   Description automatically generated
3. 

### Reflection:

*First I added the rasterizer description and set the state on the device.*

*Now to draw two triangles, we set the indices array to have only 6 indices for two front facing triangles.*

*To draw the wall of the cube, we simply remove the top and bottom faces of the cube.*

*To draw one side of the cube, we specify the indices of two triangles of the same face.*

## Exercise 2

### Draw the cube as a wireframe. This can be done by modify the fill mode in the rasterizer description table.

### Solution:

rasterDesc.FillMode = D3D11\_FILL\_WIREFRAME;

### Sample output:

A colorful cube with lines

Description automatically generated

### Reflection:

*To simply draw wireframes we can just set the FillMode in the raster description to be wireframe instead of solid.*

## Exercise 3

### Modify the parameter in IASetPrimitiveTopology( ) and indices[] to draw: 1. A list of points corresponding to the cube’s eight vertices. 2. The 12 edges of the cube (not as a wireframe triangle mesh).

### Solution:

WORD indices[] =

{

0,1,2,3,4,5,6,7

};

g\_pImmediateContext->IASetPrimitiveTopology(D3D11\_PRIMITIVE\_TOPOLOGY\_POINTLIST);

1. WORD indices[] =

{

// Top face

0, 1, 1, 2, 2, 3, 3, 0,

// Bottom face

4, 5, 5, 6, 6, 7, 7, 4,

// Vertical edges

0, 4, 1, 5, 2, 6, 3, 7

};

g\_pImmediateContext->IASetPrimitiveTopology(D3D11\_PRIMITIVE\_TOPOLOGY\_LINELIST);

### Sample Output:

1. A blue background with many small colored dots

   Description automatically generated
2. A colorful cube on a blue background

   Description automatically generated

### Reflection:

*We set the topology to use POINTLIST and set only the indices of the points to draw each point of the cube.*

*And to draw the edges, we switch to point list, and then set the list of indices to match the change (every edge needs two indices).*

## Exercise 4

### Draw two wireframe cubes. There are different ways of achieving this. One simple way is to pass a different world matrix to the vertex shader.

### Solution:

g\_World \*= XMMatrixTranslation(-2, 2, -1);

g\_World \*= XMMatrixScaling(0.5, 0.5, 0.5);

cb.mWorld = XMMatrixTranspose(g\_World);

g\_pImmediateContext->UpdateSubresource(g\_pConstantBuffer, 0, nullptr, &cb, 0, 0);

g\_pImmediateContext->DrawIndexed(36, 0, 0); // 36 vertices needed for 12 triangles in a triangle list

### Sample output:

A colorful cube on a blue background

Description automatically generated

### Reflection:

*We can simply use the same vertices and indices because the cube model didn’t change, so we change the world matrix to translate the cube again after drawing it the first time and scale it down a little bit. And then we draw it again so it looks like we have two cubes now on the screen by recycling the same vertices and indices.*

## Exercise 5

### Draw the cube as triangle strips by setting primitive topology as D3D11\_PRIMITIVE\_TOPOLOGY\_TRIANGLESTRIP.

### Solution:

WORD wallIndices[] =

{

// Front face

0, 1, 4, 5,

// Right face

1, 2, 5, 6,

// Back face

2, 3, 6, 7,

// Left face

3, 0, 7, 4

};

WORD topIndices[] =

{

3, 2, 0, 1

};

WORD bottomIndices[] =

{

4, 5, 7, 6

};

### Sample output:

A colorful cube on a blue background

Description automatically generated

### Reflection:

*We cannot render the whole cube using triangle strip in one go, we can separate the cube into three triangle strips instead, the walls of the cube are a triangle strip, the bottom and the top faces are two individual triangle strips.  
So then we must separate our indices into 3 different arrays, and then we will draw each individual array as a triangle strip*