MASTER ELECTRONIC DESIGN

Homework 3(T1_2)

Miguel Tlapa Juárez

17/05/2014

This document describes the system architecture and design about the body controller module, it's have block diagram and flowchart to describe software and hardware architecture.

Revision History

Date	Revision Number	Author/Editor	Modifications
June2014	0.1	Miguel Tlapa	Created file

Disclaimers

OBJECTIVES:

Declaration of Private Attributes:

```
class BoxApp : public DXApp
{

private: // ...
```

```
    //Counter for Animation
    float m_count;

    //Rotation Angle
    float mAngle;

    //Rotation Axis
    int mRotationAxis;
    float axis_x = 0;
    float axis_y  = 1;
    float axis_z  =1;

    //camera
    float m_count_camera;
    float eye_camx = 0;
    float eye_camy = 0;
    float eye_camz  =-10;

    float up_x = 0;
    float up_y = 1;
    float up_z = 0;

    float fpointx = 0;
    float fpointy = 0;
    float fpointz = 0;
```

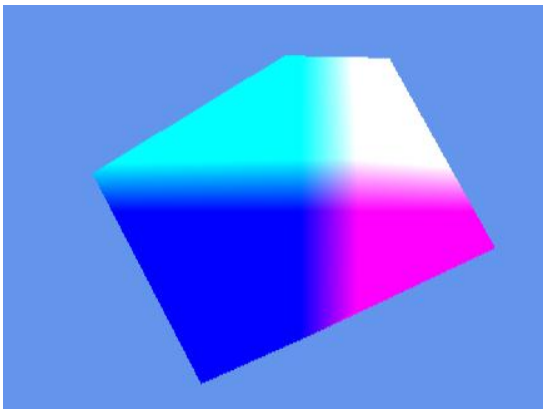
```
    // Size of Cube
    int m_count_size;

    // Animation
    int m_count_animation;
    int flag_increase;
private:
    void InitShaders();
    void InitPrecompiledShaders();
    void InitGraphics();
    void InitConstantBuffers();
```

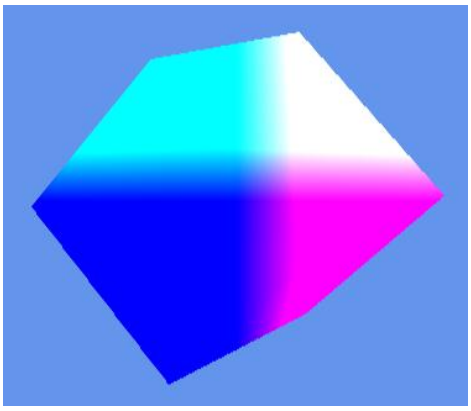
I modified the method update:

```
XMVECTOR rotationAxis = XMVectorSet(axis_x, axis_y,axis_z, 0);
XMMATRIX W = XMMatrixRotationAxis(rotationAxis, XMConvertToRadians(mAngle));

 XMStoreFloat4x4(&mWorld, W);
 m_pImmediateContext->UpdateSubresource(mpConstantBuffers[CB_Object], 0, nullptr, &mWorld, 0, 0);
```

1. **Increase the rotation angle when you press the key “A”.**

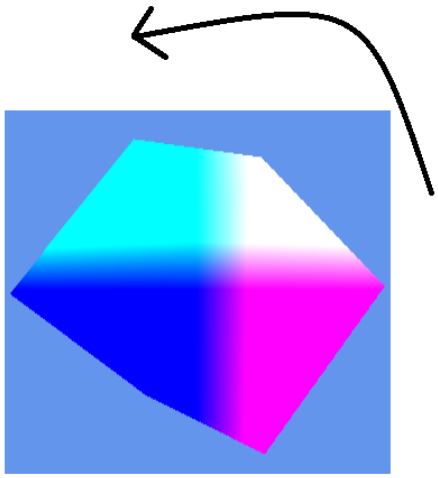
```
if (GetAsyncKeyState('A') & 0x01)
{
    mAngle++;

}
```





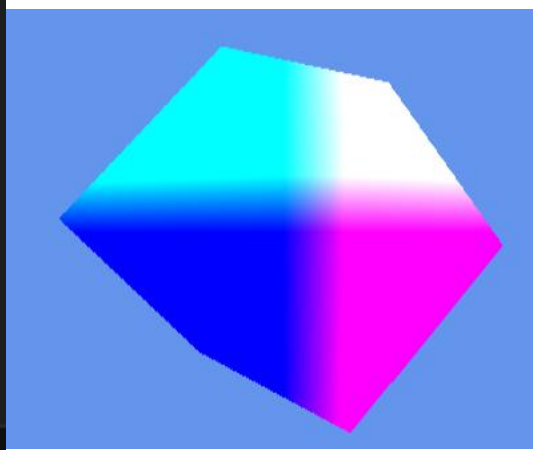
2. **Create a counter that increases the rotation angle every n iterations.**

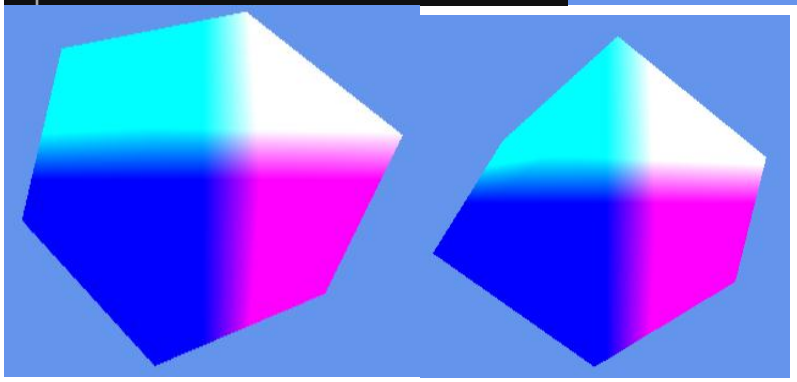
```
if (GetAsyncKeyState('F') & 0x01)
{
    m_count_animation++;
    if (m_count_animation == 1)
    {
        flag_increase = 1;
    }
    if (m_count_animation == 2)
    {
        flag_increase = 0;
        m_count_animation = 0;

    }

}

if (flag_increase == 0)
{
    mAngle++;

}
```



3. **Modify the rotation axis when you press another key.**

```
if (GetAsyncKeyState('X')& 0x01)
{
    mRotationAxis++;
    if (mRotationAxis == 1)
    {
        //Axis Z
        axis_x = 1;
        axis_y = 1;
        axis_z = 0;
    }
    if (mRotationAxis == 2)
    {
        //Axis Y
        axis_x = 1;
        axis_y = 0;
        axis_z = 1;
    }

    if (mRotationAxis == 3)
    {
        //Axis X
        axis_x = 0;
        axis_y = 1;
        axis_z = 1;
        mRotationAxis = 0;
    }


}
```





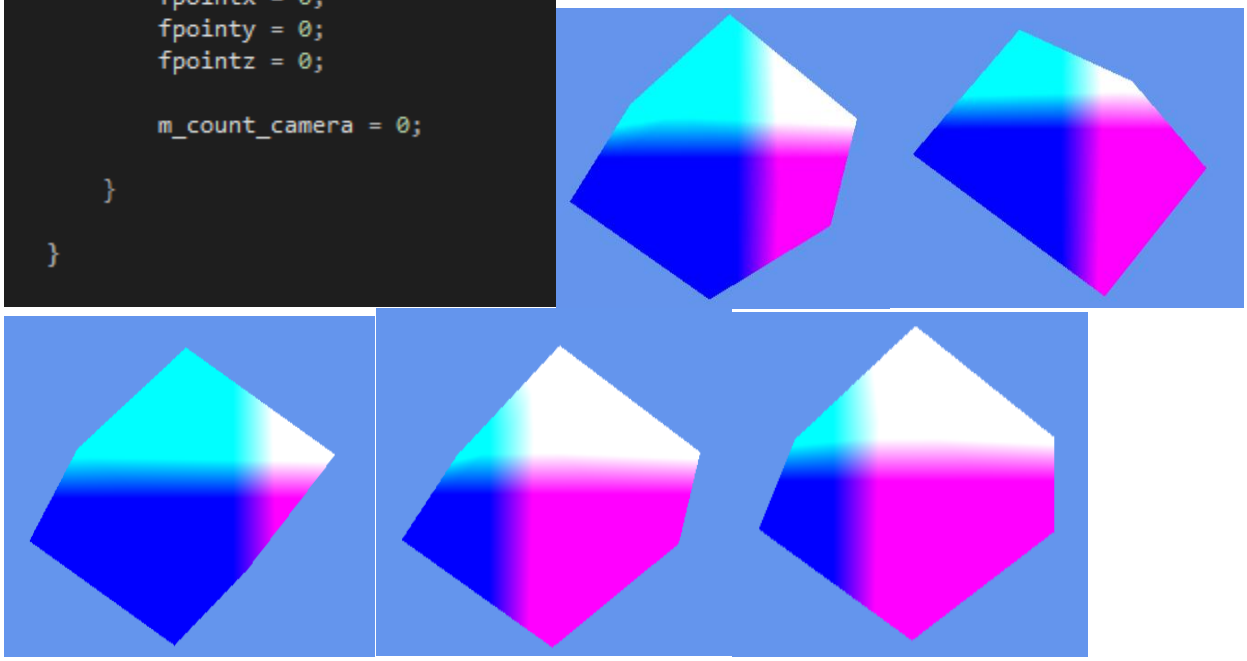
4. **Modify the position of the camera when you press another key.**

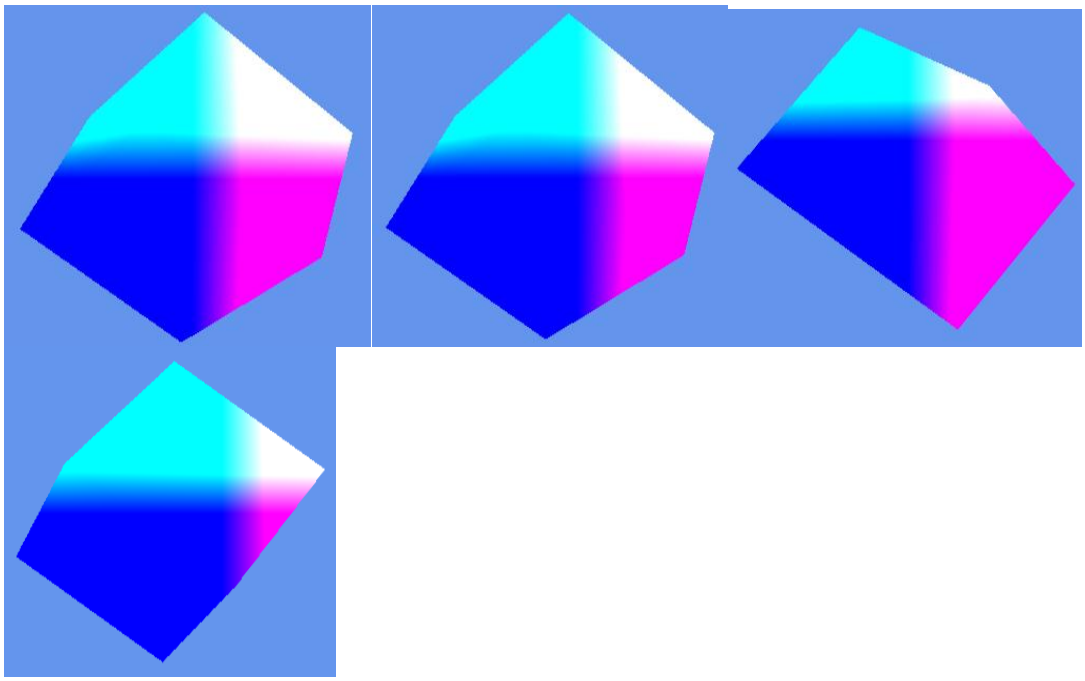
```
if (GetAsyncKeyState('C') & 0x01)
{
    m_count_camera++;
    if (m_count_camera == 1)
    {
        eye_camx = 0;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 0;
        up_y = 1;
        up_z = 0;


        fpointx = 0.2;
        fpointy = 0;
        fpointz = 0;

    }
```

```
    if (m_count_camera == 2)
    {
        eye_camx = 2;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 1;
        up_y = 0;
        up_z = 0;

        fpointx = 0.4;
        fpointy = 0;
        fpointz = 0;
    }

    if (m_count_camera == 3)
    {
        eye_camx = 4;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 0;
        up_y = 4;
        up_z = 0;

        fpointx = 0.6;
        fpointy = 0;
        fpointz = 0;

    }
```

```
    if (m_count_camera == 4)
    {
        eye_camx = 6;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 0;
        up_y = -2;
        up_z = 0;

        fpointx = 0.8;
        fpointy = 0;
        fpointz = 0;


    }
    if (m_count_camera == 5)
    {
        eye_camx = 8;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 0;
        up_y = -4;
        up_z = 0;

        fpointx = 1.0;
        fpointy = 0;
        fpointz = 0;
```

```
    }

    if (m_count_camera == 6)
    {
        eye_camx = 0;
        eye_camy = 0;
        eye_camz = -10;

        up_x = 0;
        up_y = 1;
        up_z = 0;

        fpointx = 0;
        fpointy = 0;
        fpointz = 0;

        m_count_camera = 0;

    }

}
```





5. **Modify the vertex position to show a prism.**

```
if (GetAsyncKeyState('Y') & 0x01)
{
    m_count_size++;

    if (m_count_size == 1)
    {
        bVertices[2].Pos.x = 4.0f;
        bVertices[2].Pos.y = -0.5f;

        bVertices[3].Pos.x = 4.0f;
        bVertices[3].Pos.y = -0.5f;

        bVertices[6].Pos.x = 4.0f;
        bVertices[6].Pos.y = 0.5f;

        bVertices[7].Pos.x = 4.0f;
        bVertices[7].Pos.y = -0.5f;

    }

    if (m_count_size == 2)
    {
        bVertices[2].Pos.x = 1.0f;
        bVertices[2].Pos.y = 1.0f;

        bVertices[3].Pos.x = 1.0f;
        bVertices[3].Pos.y = -1.0f;

        bVertices[6].Pos.x = 1.0f;
        bVertices[6].Pos.y = 1.0f;

        bVertices[7].Pos.x = 1.0f;
        bVertices[7].Pos.y = -1.0f;

        m_count_size = 0;

    }


}
//Create the vertex buffer

D3D11_BUFFER_DESC bdesc;
ZeroMemory(&bdesc, sizeof(bdesc));
bdesc.Usage = D3D11_USAGE_DYNAMIC;
bdesc.ByteWidth = sizeof(VERTEX)* ARRAYSIZE(bVertices);
bdesc.CPUAccessFlags = D3D11_CPU_ACCESS_WRITE;
bdesc.BindFlags = D3D11_BIND_VERTEX_BUFFER;

D3D11_SUBRESOURCE_DATA vData;

vData.pSysMem = bVertices;// Es el MemCopy
HR(m_pDevice->CreateBuffer(&bdesc, &vData, &mpBoxVBuffer));
```