AMD **Accelerated** Parallel Processing TECHNOLOGY

SAMPLE

DX11 AMP Interop

1 Overview

1.1 Location

`$<APPSDKSamplesInstallPath>\samples\C++Amp\`

1.2 How to Run

See the *Getting Started* guide for how to build samples. You first must compile the sample.

Use the command line to change to the directory where the executable is located. The default executables are placed in `$<APPSDKSamplesInstallPath>\samples\C++Amp\bin\x86` for 32-bit builds and `$<APPSDKSamplesInstallPath>\samples\C++Amp\bin\x86_64\` for 64-bit builds.

Before building this sample:

- Ensure you have the latest Windows 8 SDK.
- Ensure you have installed Microsoft® Visual Studio® 2012.
- Ensure you have installed the latest AMD display driver.

2 Implementation Details

1. Type the following command: `DX11AMPinterop` .

   A point turns counter-clockwise from (x, y) to (u,v).

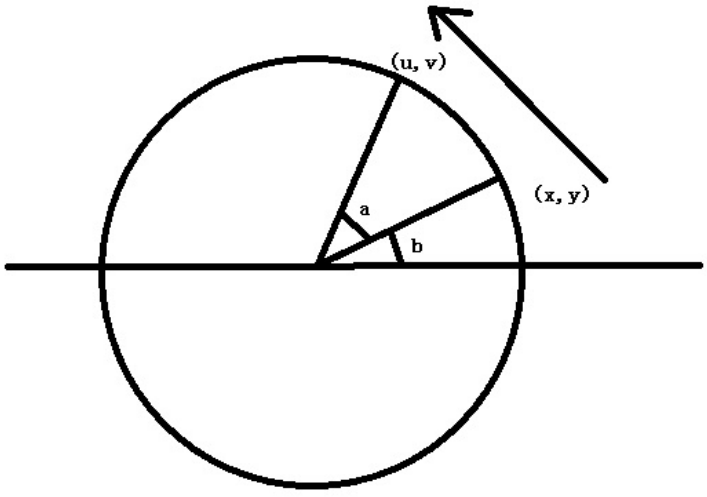

Figure 1 Algorithm of the Visual Effect

2. The new coordinate is (u,v).¨From the X axis direction, point(x, y) turns counter-clockwise by angle B. So,

```
x/(x^2+y^2) = cos(b)

y/(x^2+y^2) = sin(b)

u/(x^2+y^2) = cos(a + b) = cos(a)cos(b) - sin(a)sin(b)
       = cos(a) * x / (x^2 + y^2) - sin(a) * y / (x^2 + y^2)

v/(x^2+y^2) = sin(a+b) = sin(a)cos(b) + cos(a)sin(b)
       = x * sin(a) / (x^2 + y^2) + y * cos(a) / (x^2 + y^2)
```

u = x * cos(a) - y * sin(a) v = x * sin(a) + y * cos(a)

Thus, the new coordinate is `(u,v) = ( x * cos(a) - y * sin(a) , x * sin(a) + y * cos(a) )`

3 Sample Design

1. The input of this sample consists of four vertex-position data. We draw two lines based one these four vertices.

```
XMFLOAT2 vertices[nVerticesNum] =
{
    XMFLOAT2(-0.5f,  -0.5f),
    XMFLOAT2( 0.5f,   0.5f),
    XMFLOAT2(-0.5f,   0.5f),
    XMFLOAT2( 0.5f,  -0.5f),
}
```

2. Create an array buffer with these vertices and the default accelerator_view.

```
pData = new array<XMFLOAT2, 1>(nDataNum, vertices, acclView)
```

3. Get the underlying Direct3D buffer from the array object.

```
concurrency::direct3d::get_buffer(*pData)->QueryInterface(__uuidof(ID3D11Buffer),
(LPVOID*)d3dbuffer);
```

4. Rotate the two lines counter-clockwise. Use the amp engine to finish this work.

   The angle of this sample is `#define ANGLE (3.1416f/2000)`

```
parallel_for_each(pData->extent, [=, &arrayData] (index<1> idx) restrict(amp)
{
    // Compute new pos data
    XMFLOAT2 pos     = arrayData[idx];
    arrayData[idx].x = pos.x * cos(ANGLE) - pos.y * sin(ANGLE);
    arrayData[idx].y = pos.x * sin(ANGLE) + pos.y * cos(ANGLE);
})
```

5. Render the shader resource view base on the D3D buffer.

4 DirectX11 Environment and Render

Figure 2 shows the order of intializing DirectX11.

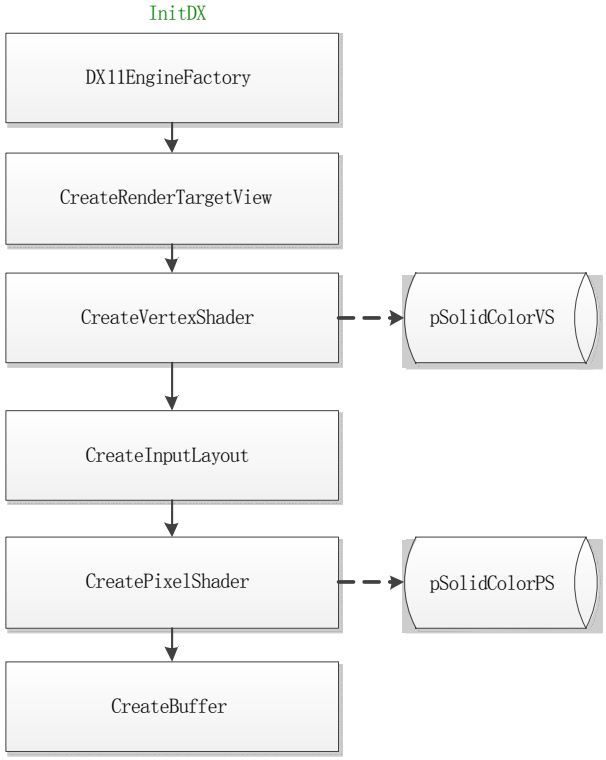

Figure 2 DX11 Initialization Sequence

5 Object Interop

Figure 3 shows the program flow.

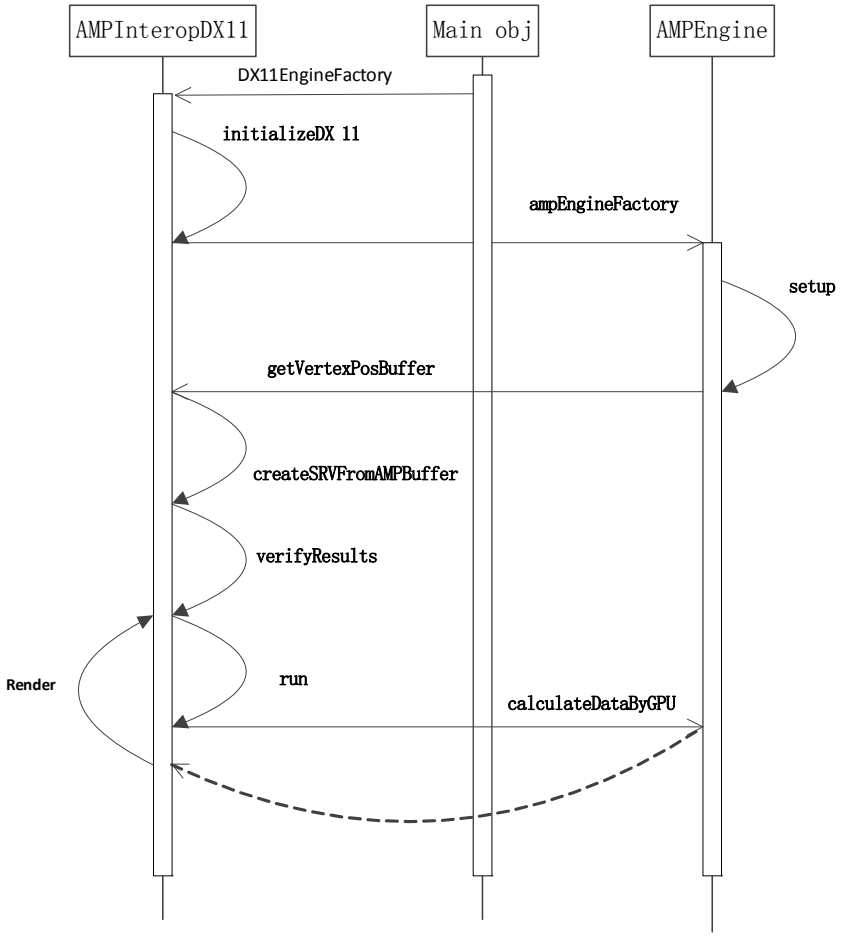

Figure 3 Program Flow

6 Sample Effect

Figure 4 shows the before and after states for this sample interop.

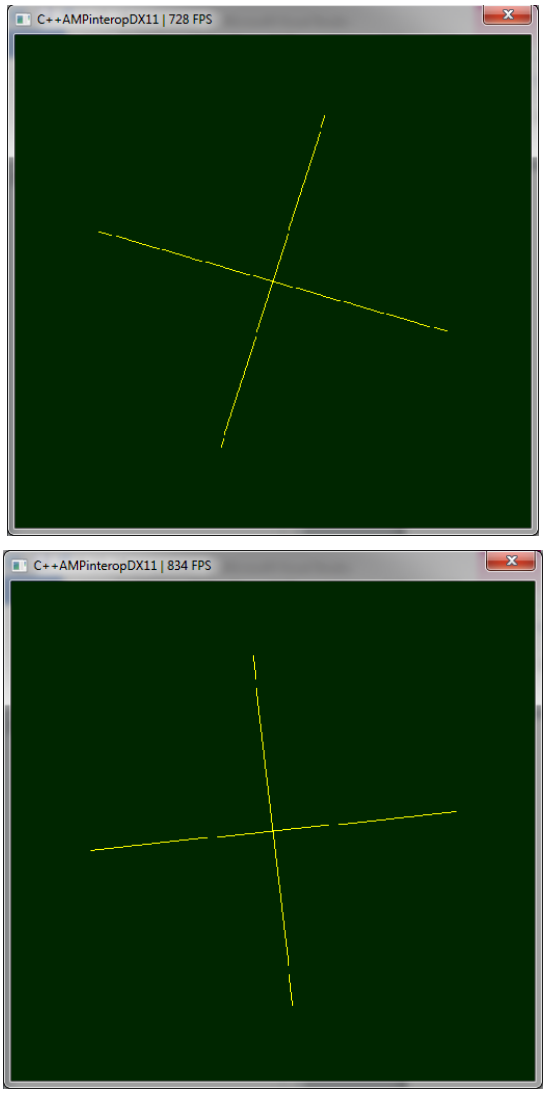

Figure 4 Screenshots of Before and After Rotation

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:

URL: developer.amd.com/appsdk
Developing: developer.amd.com/
Forum: developer.amd.com/openclforum