SDK White Paper

Custom Clip Plane
Using Near Clip Plane Shearing

WP-01386-001-v01
July 2004

nVSDK

Abstract

Near Clip Plane Searing

Custom user clip planes are a useful tool for implementing a variety of interesting visual effects. One example would be water reflections. You can use a custom user clip plane to clip world geometry and only reflect object above the water. However, many graphics cards do not support (or have problems with) custom user clip planes.

One solution to obtain a single custom user clip plane is to shear the near clip plane and use the normal clipping hardware to perform your custom clip. This sample implements **`NearClipPlane`** shearing.

This sample is written to run under DirectX9.0b. It should run on all 3D hardware,

Bryan Dudash
bdudash@nvidia.com
NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050

Custom Near Clip Plane Shearing

Shearing the near clip plane can be an easy way to custom-clip your scene. All you need to do is implement a change to the projection matrix. This change effectively changes the projection clip space **z** axis to be along the plane normal. The following section contains a code excerpt from the sample with important math to shear the near clip plane. It has been pseudo-codified a little bit to improve readability (primarily removing variable decl's). For executable code, refer to the sample source.

Near Clip Shearing Code Sample

```
    // normalize WorldSpace to ProjectionSpace transform
    D3DXMatrixInverse(&WorldToProjection, NULL,
&WorldToProjection);
    D3DXMatrixTranspose(&WorldToProjection, &WorldToProjection);

    D3DXVECTOR4 projClipPlane;

    // transform clip plane into ProjectionSpace
    D3DXVec4Transform(&projClipPlane, &clipPlane,
&WorldToProjection);

       // Ensure Near clip always faces away from Eye point.
    if (projClipPlane.w > 0)
    {
        D3DXVec4Transform(&projClipPlane, &(-clipPlane),
&WorldToProjection);
    }

    // Create a transform to convert the projection Matrix
    // to our custom near clip space.
    D3DXMATRIXA16 matClipProj;
    D3DXMatrixIdentity(&matClipProj);
    matClipProj(0, 2) = projClipPlane.x;
    matClipProj(1, 2) = projClipPlane.y;
    matClipProj(2, 2) = projClipPlane.z;
    matClipProj(3, 2) = projClipPlane.w;

    // Create a new custom clip projection matrix
    D3DXMATRIXA16 projClipMatrix = matProj * matClipProj;
    m_pd3dDevice->SetTransform( D3DTS_PROJECTION,
&projClipMatrix);
```

Details

The important pieces from the code sample are that the basic algorithm converts the plane into our projection space and then use that converted plane to define a space transform to convert our projection space into the custom projection space. This custom projection space has the plane normal as the **z** axis, and thus distances for near clip uses this axis and implements the custom plane with the near clipper.

A couple of things to note:

- You must *normalize* your World to Projection transform, since the plane is like a normal and will be incorrect if you do not.
- You must ensure that the plane faces away from the eye point, if it does not, you need to flip it.
- The space transform just assigned the **z** axis of an identity matrix to be the plane. This creates a space-to-space transform that changes the far clip distances, and the far clip will not be equivalent, unless you have an infinite clip distance

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com