```
// 被写界深度
//
string XFile = "misc/nanase_pose.x";
int BCLR = 0x000000ff;

// 変換マトリックス
float4x4 matWVP : WORLDVIEWPROJECTION;  // ワールド×ビュー×射影
float4x4 matW   : WORLD;                // ワールド

float3 camera   : CAMERAPOSITION;       //視点座標

// 光源ベクトル（光の進む方向）
float3 light <string UIDirectional = "Light Direction";> = {-0.35f, -0.25f, 0.9f};

// 光源の色
float4  I_d = {1.0f, 1.0f, 1.0f, 1.0f};//拡散光
float4  I_a = {0.2f, 0.2f, 0.2f, 1.0f};//環境光
float4  I_s = {1.0f, 1.0f, 1.0f, 1.0f};//鏡面反射光

// Xファイル　マテリアル
float4  k_d : MATERIALDIFFUSE;   //拡散色
float4  k_s : MATERIALSPECULAR;  //鏡面反射色
float   k_p : MATERIALPOWER;     //鏡面反射指数
float4  k_a : MATERIALAMBIENT;   //環境色
texture tex : MATERIALTEXTURE;
bool    tex_enable : MATERIALTEXTUREVALID;

texture texBlur;        // ぼかしテクスチャ
texture texDepth;       // 深度テクスチャ
float   focus;          // ピント位置(0～1)
float   range = 10.0f;  // フォーカス範囲

// サンプラステート
sampler Sampler = sampler_state
{
    Texture = (tex);
    MipFilter = LINEAR;
    MinFilter = LINEAR;
    MagFilter = LINEAR;
};
sampler SamplerBlur = sampler_state
{
    Texture = (texBlur);
    MipFilter = LINEAR;
    MinFilter = LINEAR;
    MagFilter = LINEAR;
};
sampler SamplerDepth = sampler_state
{
    Texture = (texDepth);
    MipFilter = POINT;
    MinFilter = POINT;
    MagFilter = POINT;
};

// 頂点シェーダ
void PhongVS(float4 InPos : POSITION,
             float3 InNor : NORMAL,
             float2 InTex : TEXCOORD0,
             out float4 OutPos  : POSITION,
             out float2 OutTex  : TEXCOORD0,
             out float3 OutWPos : TEXCOORD1,
             out float3 OutWNor : TEXCOORD2,
             out float4 OutDpth : TEXCOORD3)
{
    OutPos = mul(InPos, matWVP);
    OutTex = InTex;
    OutWPos = mul(InPos, matW).xyz;
    OutWNor = mul(InNor, (float3x3)matW);
    OutDpth = OutPos;   // Z値出力用
}

// ピクセルシェーダ
```

```
void PhongPS(float2 InTex  : TEXCOORD0,
             float3 InPos  : TEXCOORD1,
             float3 InNor  : TEXCOORD2,
             float4 InDpth : TEXCOORD3,
             out float4 OutCol  : COLOR0,
             out float4 OutDpth : COLOR1)
{
    float z = InDpth.z / InDpth.w;
    OutDpth = float4(z, z, z, 1);

    float3 L = normalize(-light);
    float3 N = normalize(InNor);
    float3 V = normalize(camera - InPos);
    float3 R = reflect(-V, N);
    OutCol = I_a * k_a + I_d * k_d * saturate(dot(L, N));
    if (tex_enable) {
        OutCol *= tex2D(Sampler, InTex);
    }
    OutCol += I_s * k_s * pow(saturate(dot(L, R)), k_p * 0.25f);
}

// 頂点シェーダ(単純コピー)
void SimpleVS(float4 InPos : POSITION,
              float2 InTex : TEXCOORD0,
              out float4 OutPos : POSITION,
              out float2 OutTex : TEXCOORD0)
{
    OutPos = InPos;
    OutTex = InTex;
}

// ピクセルシェーダ(強烈ぼかし&1/2コピー)
void DiffusePS(float2 InTex : TEXCOORD0,
               out float4 OutCol : COLOR0)
{
    // テクセル中央にずらす(出力サイズ400x300時)
    InTex.x += 0.5f / 400;
    InTex.y += 0.5f / 300;
    // テクセルサイズ計算
    float dx = 1.0f / 400;
    float dy = 1.0f / 300;
    // 該当テクセル周辺の平均を求める(9x9)
    OutCol = float4(0, 0, 0, 0);
    for (int y = -4; y < 5; ++y) {
        for (int x = -4; x < 5; ++x) {
            OutCol += tex2D(Sampler,
                InTex + float2(dx * x, dy * y)) / 81.0f;
        }
    }
    // アルファ値補正
    OutCol.a = 1.0f;
}

// 被写界深度
void FocusPS(float2 InTex : TEXCOORD0,
             out float4 OutCol : COLOR0)
{
    float depth = tex2D(SamplerDepth, InTex).r;
    float blend = abs(depth - focus) * range;
    blend = saturate(blend);

    OutCol = tex2D(Sampler, InTex) * (1.0f - blend)
        + tex2D(SamplerBlur, InTex) * blend;
}

// 通常描画
technique TPhong
{
    // パス
    pass P0
    {
        VertexShader = compile vs_3_0 PhongVS();
        PixelShader = compile ps_3_0 PhongPS();
    }
}

// ぼかしシェーダ
technique TDiffuse
{
    pass P0
    {
        VertexShader = compile vs_3_0 SimpleVS();
        PixelShader = compile ps_3_0 DiffusePS();
    }
}

// 被写界深度
technique TFocusFilter
{
    pass P0
    {
        VertexShader = compile vs_3_0 SimpleVS();
        PixelShader = compile ps_3_0 FocusPS();
    }
}
```