using System;

using UnityEngine;

namespace UnityStandardAssets.ImageEffects

{

[ExecuteInEditMode]

[RequireComponent (typeof (Camera))]

[AddComponentMenu ("Image Effects/Edge Detection/Edge Detection")]

public class EdgeDetection : PostEffectsBase

{

public enum EdgeDetectMode

{

TriangleDepthNormals = 0,

RobertsCrossDepthNormals = 1,

SobelDepth = 2,

SobelDepthThin = 3,

TriangleLuminance = 4,

}

public EdgeDetectMode mode = EdgeDetectMode.SobelDepthThin;

public float sensitivityDepth = 1.0f;

public float sensitivityNormals = 1.0f;

public float lumThreshold = 0.2f;

public float edgeExp = 1.0f;

public float sampleDist = 1.0f;

public float edgesOnly = 0.0f;

public Color edgesOnlyBgColor = Color.white;

public Shader edgeDetectShader;

private Material edgeDetectMaterial = null;

private EdgeDetectMode oldMode = EdgeDetectMode.SobelDepthThin;

public override bool CheckResources ()

{

CheckSupport (true);

edgeDetectMaterial = CheckShaderAndCreateMaterial (edgeDetectShader,edgeDetectMaterial);

if (mode != oldMode)

SetCameraFlag ();

oldMode = mode;

if (!isSupported)

ReportAutoDisable ();

return isSupported;

}

new void Start ()

{

oldMode = mode;

}

void SetCameraFlag ()

{

if (mode == EdgeDetectMode.SobelDepth || mode == EdgeDetectMode.SobelDepthThin)

GetComponent<Camera>().depthTextureMode |= DepthTextureMode.Depth;

else if (mode == EdgeDetectMode.TriangleDepthNormals || mode == EdgeDetectMode.RobertsCrossDepthNormals)

GetComponent<Camera>().depthTextureMode |= DepthTextureMode.DepthNormals;

}

void OnEnable ()

{

SetCameraFlag();

}

[ImageEffectOpaque]

void OnRenderImage (RenderTexture source, RenderTexture destination)

{

if (CheckResources () == false)

{

Graphics.Blit (source, destination);

return;

}

Vector2 sensitivity = new Vector2 (sensitivityDepth, sensitivityNormals);

edgeDetectMaterial.SetVector ("\_Sensitivity", new Vector4 (sensitivity.x, sensitivity.y, 1.0f, sensitivity.y));

edgeDetectMaterial.SetFloat ("\_BgFade", edgesOnly);

edgeDetectMaterial.SetFloat ("\_SampleDistance", sampleDist);

edgeDetectMaterial.SetVector ("\_BgColor", edgesOnlyBgColor);

edgeDetectMaterial.SetFloat ("\_Exponent", edgeExp);

edgeDetectMaterial.SetFloat ("\_Threshold", lumThreshold);

Graphics.Blit (source, destination, edgeDetectMaterial, (int) mode);

}

}

}