Candera Reference Shaders

Description

This document describes content and usage of Candera Reference Shaders part of CGI Studio releases. Within this document the set of Candera Reference Shaders is listed with additional information how to use and combine them in 3D applications based on Candera.

Location of Reference Shaders

Candera delivers a set of reference shader implementations, which can be found at following location:

- **\cgi_studio_candera\src\Candera\Engine3D\RefShaders**

The same set of reference shaders is available in SceneComposer.

Preferred way of usage is to export shaders to a binary asset file and to load it in an application based on Candera. Shaders part of exported scenes are automatically added to the asset file and loaded by Candera.

Used Include Files

Some parts of shader code used in Candera Reference Shaders have been collected into several files in order to be reused as include files.

Following files are available and can be used as include files:

Include File	Supports	Include In	Note	Displaces
RefEnvironment.inc	Platform specific defines	All Shaders	Shall be included in each Shader as first statement	Environment.incv, Environment.incf
RefLighting.inc	Lighting specific functions, also functions for anisotropic lighting	Shaders that support lighting	Useful helper functions for vertex lighting and pixel lighting	DefaultLighting.incv

RefTexturing.inc	Texturing specific functions, e.g. Environment Mapping	Shaders that support certain texture effects	Useful helper functions especially for common texture effects	Texturing.incv
RefSkinning.inc	Skinning specific features	Shaders that support skinning	Useful helper functions for skinning	Skinning.incv

Reference Vertex Shaders

Vertex Shaders	Supports	Combine with Fragment Shaders	GenericShaderParamSetter Configuration	Note
Core/RefTrans	Transformation, Texture	Core/RefTex, Core/RefUniColor, Core/RefUniColorTex, Core/RefTex2LightMap, Core/RefUniDiffuseMatTex, Effects/RefTexDofBlur, Effects/RefTexDofCombine	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Lighting has no influence.
Core/RefTransColor	Transformation, VertexColor, Texture	Core/RefColor, Core/RefColorTex, Core/RefColorTexAlpha, Effects/RefColorTexAlphaOutline	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Lighting has no influence.
Core/RefTransLight1	Transformation, Vertex Lighting with one light source, Texture	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	If Texture is used, depends on corresponding fragment shader only.
Core/RefTransLight2	Transformation, Vertex Lighting with two light sources. Texture	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	If Texture is used, depends on corresponding fragment shader only.

Core/RefTransLight3	Transformation, Vertex Lighting with three light sources. Texture	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	If Texture is used, depends on corresponding fragment shader only.
Core/RefTransWorldLight1	Transformation, Lighting in WorldSpace with 1 Light Texture	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(World)	Lighting calculations are processed in world space instead of model space (For details see function Light::SetCoordinateSpace). If Texture is used, depends on corresponding fragment shader only.
Effects/RefTransSphereMap	Transformation, Sphere Texture Mapping	Core/RefTex, Core/RefUniColorTex, Core/RefUniDiffuseMatTex	SetModelViewMatrix4Enabled(true) SetNormalModelViewMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Used to process a sphere texture for environment mapping.
Effects/RefTransLight1SphereMap	Transformation, Vertex Lighting with one light source, Sphere Texture Mapping	Core/RefColorTex	SetModelViewMatrix4Enabled(true) SetNormalModelViewMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Used to process a sphere texture for environment mapping.

Effects/RefTransLight2SphereMap	Transformation, Vertex Lighting with two light sources, Sphere Texture Mapping	Core/RefColorTex	SetModelViewMatrix4Enabled(true) SetNormalModelViewMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Used to process a sphere texture for environment mapping.
Effects/RefTransLight3SphereMap	Transformation, Vertex Lighting with three light sources, Sphere Texture Mapping	Core/RefColorTex	SetModelViewMatrix4Enabled(true) SetNormalModelViewMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Used to process a sphere texture for environment mapping.
Core/RefTransPointSprite	Transformation of PointSprite	Core/RefPointSprite	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Use with PointSprite. As usual for point sprites, lighting has no influence.
Core/RefTransLight1Morph	Transformation, VertexLighting with one lighe source, Morphing Weights	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Use with MorphingMesh. If Texture is used, depends on corresponding fragment shader only.

Effects/RefTransLight1BumpMap	Transformation, Calculation of lighting vectors in tangent space.	Effects/RefLight1BumpMap	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(World)	Used to prepare lighting vectors for bump map shaders.
Effects/RefTransLight1ProceduralWood	Transformation, Vertex Lighting with one light source, Passing position to fragment shader for procedural calculations.	Effects/RefProceduralWood	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	
Effects/RefTransAnisotropicLight1Strand	Transformation, passes vertex attributes in world space to fragment shader. Calculates direction of anisotropy, eye vector and light direction.	Effects/RefAnisotropicLight1StrandTex	SetModelViewProjectionMatrix4Enabled(true) SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetCameraPositionEnabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(World)	

Effects/RefTransAnisotropicLight1	Transformation, Passes vertex attributes in world space to fragment shader.	Effects/RefAnisotropicLight1SpecularTex	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(World)	
Effects/RefTransFur	Transformation, Indexed Multi pass upscaling in direction of normals, Texture	Effects/RefUniColorTexFur	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Lighting has no influence.
Effects/RefTransCubeMapReflection	Transformation, Texturing reflection using one CubeMap texture	Core/RefCubeMapTex	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Environment mapping reflections using a CubeMap texture. Lighting is not considered.
Effects/RefTransCubeMapRefraction	Transformaion, Texturing refraction using one cubemap texture.	Core/RefCubeMapTex	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Environment mapping refractions using a CubeMap texture. Lighting is not considered. Refraction ratio has to be passed as additional uniform float refractionRatio.

Effects/RefTransCubeMapReflectionRefraction	Transformation, Texturing reflection and refraction using one cubemap texture.	Core/RefCubeMapTex2	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Environment mapping refractions and reflections using a CubeMap texture. Lighting is not considered. Refraction ratio has to be passed as additional uniform float refractionRatio.
Core/RefTransCubeMap	Transformation, Texturing using one CubeMap texture, addressed directly by object space vertex coordinates.	Core/RefCubeMapTex	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Direct addressing of CubeMap texture using object space vertex positions. Can be used for e.g. a SkyBox using a CubeMap texture.
Effects/RefTransDof	Transformation, Texturing with one texture, First render pass of depth-of-field rendering: Rendering the scene, Scaled depth value passed as varying.	Effects/RefTexDof	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. Light, bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements. Additional uniform u_depthScale has to be passed to scale the depth values.

Effects/RefTransLight1Dof	Transformation, Texturing with one Texture Vertex Lighting with one light source, First render pass of depth-of-field rendering: Rendering the scene, Scaled depth value passed as varying.	Effects/RefColorTexDof	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(Light::Object);	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements. Additional uniform u_depthScale has to be passed to scale the depth values. Supports per vertex lighting.
Effects/RefTransFragmentLightDof	Transformation, Texturing with one Texture, Normal and position for per fragment lighting, First render pass of depth-of-field rendering: Rendering the scene, Scaled depth value passed as varying.	Effects/RefTexFragmentLightDof	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(Light::World); SetModelMatrix4Enabled(true); SetNormalModelMatrix3Enabled(true);	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements. Additional uniform u_depthScale has to be passed to scale the depth values. Supports per Fragment lighting.

Core/RefTransPos	Transformation, Texture, Vertex position to fragment shader	Core/RefFlatLight1	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightActivationEnabled(true)	No light calculation, passing vertex position to fragment shader for FlatShading calculations.
Effects/RefTransLight1Gooch	Transformation, Assignment of ambient light component, Calculation of varyings needed for computation of specular component in world space. First pass of gooch-shading: Shading.	Effects/RefGoochShading	SetModelMatrix4Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetCameraPositionEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightActivationEnabled(true) SetLightsCoordinateSpace(Light::World)	This vertex shader in conjunction with RefGoochShading.frag p realizes the first pass of an implementation of Gooch's rendering technique to simulate the appearance of technical illustrations, often occuring in e.g. manuals or schematic presentations etc. (SIGGRAPH'98 paper "A Non-Photorealistic Lighting Model For Automatic Technical Illustration"). The first pass renders the object using a transition from "cool" to "warm" colors according to the lighting model. While phong shading is normally used to simulate real lighting, this lighting model aims more on preserving details away from the light source. The second pass then draws the silhoutte of the object. For this the object is upscaled and rendered with a fixed color (e.g. black). On Candera side the first and the second pass have to be rendered with different Culling states, such that first pass renders the front of the object, and the second only the back side.

Effects/RefTransScale	Transformation, Upscaling in direction of normals, texture coordinates.	Core/RefTex, Core/RefUniColor, Core/RefUniColorTex, Core/RefUniDiffuseMat, Core/RefTex2LightMap	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Lighting has no influence. Used to upscale the vertices inside the vertex shader, using a float scale factor passed as uniform u_Scale. Is used for silhouette rendering in gooch-shading, but may be also used for other techniques.
Core/RefViewportSpace	Texture Adjusting the position of the screen aligned billboard by setting uniform u_offsetLeft and u_offsetTop.	Effects/RefTexGaussianBlurX, Effects/RefTexGaussianBlurY, Effects/RefTexDofCombine, Effects/RefBloomThreshold, Effects/RefCombineBloom, Core/RefInterlaceMask	SetModelViewProjectionMatrix4Enabled(false)SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true)	Used in subsequent post processing passes in conjuction with a screen-aligned billboard. The billboard should be of size 0.0 to 1.0 with 1.0 being a fullscreen billboard. The position of the billboard on the screen can be adjusted by setting the left and top offset in the range of 0.0 to 1.0. Passes the position and texture coordinate to the fragment shader without any matrix transformation. Therefore the incoming vertex buffer should already be in homogenous screen coordinates. Note: Overlapping Billboards (in screen space) would produce Z-fighting artifacts. Disabling depth test and using render order as well as using depth bias can be used to avoid them.

Effects/RefTransLight1Carbon	Transformation, Lighting with one light source, Texture, Line of Sight	Effects/RefCarbon	SetModelMatrix3Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(Light::Object)	Seamless texture utilized as carbon image
Effects/RefTransLight1FlopCarPaint	Transformation, Lighting with one light source, Line of Sight, Mixes two color components diffuse and emissive based on camera position	Effects/RefFlopCarPaint	SetModelMatrix3Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(false) SetLightsCoordinateSpace(Light::Object)	Mixes two color components (diffuse and emissive) based on the camera position
Effects/RefTransLight1MetalFlakesCarPaint	Transformation, Lighting with one light source, Line of Sight, Texture	Effects/RefMetalFlakesCarPaint	SetModelMatrix3Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(Light::Object)	Metal flake effect with one noise texture

Effects/RefTransLight1FlopMetalFlakesCarPaint	Transformation, Lighting with one light source, Line of Sight, Texture, Mixes two color components diffuse and emissive based on camera position	Effects/RefFlopMetalFlakesCarPaint	SetModelMatrix3Enabled(true) SetNormalModelMatrix3Enabled(true) SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetLightsCoordinateSpace(Light::Object)	Metal flake effect with one noise texture Mixes two color components (diffuse and emissive) based on the camera position
Core/RefCanvasTrans	Applying scale and offset to vertices before transformation, Transformation into world space, Texturing with one texture.	Core/RefTex, Core/RefUniColor, Core/RefUniColorTex, Core/RefTex2LightMap , Core/RefUniDiffuseMatTex	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetCanvasActivationEnabled(true)	Similar to RefTrans, but additionally applies canvas offset and size before multiplying the mvp matrix.
Core/RefCanvasTransLight1	Applying scale and offset to vertices before transformation, Transformation, Texturing with one tetxture, One light source with lighting in model space.	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetCanvasActivationEnabled(true) SetLightsCoordinateSpace(Light::Object)	Similar to RefTransLight1, but additionally applies canvas offset and size before multiplying the mvp matrix.
Core/RefCanvasTransUniDiffuseMat	Applying scale and offset to vertices before transformation, Transformation into world space, Texturing with one texture, Forwarding of material diffuse color to fragment shader.	Core/RefCanvasUniDiffuseMat	SetModelViewProjectionMatrix4Enabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetCanvasActivationEnabled(true)	Similar to RefTrans, but additionally applies canvas offset and size before multiplying the mvp matrix. Additionally it passes material diffuse color to fragment shader. This shader is used to achieve the same effect as a SolidColorBrush in 2D.

Effects/RefFxaa	Coordinates in normalized device space.	Effects/RefFxaa	SetModelViewProjectionMatrix4Enabled(false) SetLightActivationEnabled(false) SetMaterialActivationEnabled(false)	Applies the "Fast approximate anti-aliasing" algorithm to its input, which should be a quad using normalized device coordinates. The uniform (u_Resolution) needs to be set to the width and height of the input texture measured in texels.
Effects/Ref	Texture	Effects/RefTexGaussianBlurX, Effects/RefTexGaussianBlurY, Effects/RefTexDofCombine, Effects/RefBloomThreshold, Effects/RefCombineBloom	SetModelViewProjectionMatrix4Enabled(false) SetLightActivationEnabled(false) SetMaterialActivationEnabled(false) SetTextureActivationEnabled(true)	Passes texture coordinates and vertex position to fragment shader without applying a transformation. This is useful for post processing effects when the vertex buffer is just a quad in normalized device coordinates (-1,1).
Effects/RefTransGradientLinear	Transformation into world space, Vertex based linear gradient computations.	Effects/RefGradientLinear	SetModelViewProjectionMatrix4Enabled(true)	Vertex Shader for linear gradient computations.
Effects/RefTransGradientRadial	Transformation into world space, Passing vertex position to fragment shader.	Effects/RefGradientRadial	SetModelViewProjectionMatrix4Enabled(true)	Vertex Shader for radial gradient computations.
Core/RefTransLight1Skin	Skinning	Core/RefColor, Core/RefColorTex	SetModelViewProjectionMatrix4Enabled(true) SetLightActivationEnabled(true) SetMaterialActivationEnabled(true) SetTextureActivationEnabled(true) SetSkinningMeshActivationEnabled(true)	If Texture is used, depends on corresponding fragment shader only. And, use with SkinningMesh.

Vertex Shader for radial gradient computations.

Reference Fragment Shaders

Fragment Shaders	Supports	Combine with Vertex Shaders	GenericShaderParamSetter Configuration	Note
Core/RefColor	Color from varying	Core/RefTransLight*, Core/RefTransWorldLight*, Core/RefTransLight1Morph	See configuration of RefTransLight* or RefTransWorldLight1	Normaly used for models that support lighting but no texture. varying color (v_Color) is typically either set by lit material or by vertex color.
Core/RefTex	Texture	Core/RefTrans, Effects/RefTransSphereMap, Effects/RefTransScale	See configuration of RefTrans or RefTransSphereMap	Lighting has no influence.
Core/RefColorTex	Color from varying, Texture	Core/RefTransLight*, Core/RefTransWorldLight*, Core/RefTransColor, Core/RefTransLight1Morph, Core/RefTransLight*SphereMap	See configuration of RefTransLight* or RefTransWorldLight1	Normaly used for models that support lighting and one texture. varying color (v_Color) is normaly either set by lit material or by vertex color.
Core/RefUniColor	Color from uniform	Core/RefTrans, Effects/RefTransScale	See configuration of RefTrans	Useful if certain color is set via uniform (u_Color) directly. Lighting or vertex colors have no influence.
Core/RefUniColorTex	Color from uniform, Texture	Core/RefTrans, Effects/RefTransSphereMap, Effects/RefTransScale	See configuration of RefTrans or RefTransSphereMap	Useful if certain color is set via uniform (u_Color) directly and blended with one texture. Lighting or vertex colors have no influence.

Core/RefUniDiffuseMatTex	Material diffuse color, Texture	Core/RefTrans, Effects/RefTransSphereMap	See configuration of RefTrans or RefTransSphereMap	Final color = materials diffuse color * texel color. The uniform value of the materials diffuse color is set by Candera automatically if a Material is applied to a Node. Lighting or vertex colors have no influence.
Core/RefPointSprite	Pointsprite incl. Texturing	Core/RefTransPointSprite	See configuration of RefTransPointSprite	Used for point sprites only.
Core/RefTex2LightMap	Texture, LightMap	Core/RefTrans, Effects/RefTransScale	See configuration of RefTrans	Lightmap Multitexturing: A Lightmap typically defines an illumination texture at unit 1 that is multiplied with texture at unit 0.
Effects/RefLight1BumpMap	Per pixel lighting using the normals supplied in u_Texture1 (normal map). Computed color is combined with the texture in u_Texture0.	Effects/RefTransLight1BumpMap	See configuration of RefTransLight1BumpMap	Used to display rough surfaces without adding polygon complexity to it. The shader accepts two textures. u_Texture0 defines the color map used. u_Texture1 defines a normal map, that stores normals in tangent space to be used for lighting calculations.
Effects/RefProceduralWood	Procedural calculation of interpolated rings, which appears like annual rings of wood if the right colors are chosen.	Effects/RefTransLight1ProceduralWood	See configuration of RefTransLight1ProceduralWood	For procedural calculations two color, the orgin of the annual rings and a multiplier that controls the densitiy of the rings can be passed.
Effects/RefAnisotropicLight1StrandTex	Per pixel anisotropic lighting, using a strand texture	Effects/RefTransAnisotropicLight1Strand	First texture for color, second for strand.	The strand texture must hold the diffuse and specular light intensity in the R and G channels, respectively.
Effects/RefAnisotropicLight1SpecularTex	Per pixel anisotropic lighting	Effects/RefTransAnisotropicLight1	See configuration of RefTransAnisotropicLight1	Using alternative anisotropic specular highlights, according to Ward's SIGGRAPH 92 paper "Measuring and Modeling Anisotropic Reflection". Additional calculated specular component is combined with u_Texture0. Used to display the reflection behavior of multiple surfaces. Uniforms u_AlphaX and u_AlphaY model the distribution of the specular highlight. Combined with a texture it can for example be used to simulate metallic surfaces like brushed steel. In the cited paper alphaX and alphaY for several materials can be found.

Effects/RefUniColorTexFur	Assignment of product of alpha mask, texture and fixed color.	Effects/RefTransFur	See configuration of RefTransFur	Simulates the appearance of fur if applied with MulitPassRendering. Is also intended to demonstrate multi pass rendering.
Core/RefCubeMapTex	Addressing one cube map texture using a single varying vec3.	Effects/RefTransCubeMapReflection, Effects/RefTransCubeMapRefraction, Core/RefTransCubeMap	See configuration of RefTransCubeMapReflection, RefTransCubeMapRefraction, RefTransCubeMap	Cube Map addressing, material colors or lighting are not considered.
Core/RefCubeMapTex2	Mixing two different addressed fragments of a CubeMap using a varying ratio. Addressing is done using two varying vec3.	Effects/RefTransCubeMapReflectionRefraction	See configuration of RefTransCubeMapReflectionRefraction	Cube Map addressing, material colors or lighting are not considered.
Core/RefUniDiffuseMat	Material diffuse color	Core/RefTrans	See configuration of RefTrans	Final color = materials diffuse color The uniform value of the materials diffuse color is set by Candera automatically if a Material is applied to a Node. Lighting or vertex colors have no influence.
Effects/RefTexDof	Assignment of Texture, First render pass of depth-of-field rendering: Rendering the scene, Depth encoded into alpha channel.	Effects/RefTransDof	See configuration of RefTransDof	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. Light, bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements.
Effects/RefColorTexDof	Color from varying, Assignment of Texture, First render pass of depth-of-field rendering: Rendering the scene, Depth encoded into alpha channel.	Effects/RefTransLight1Dof	See configuration of RefTransLight1Dof	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements. Support per vertex lighting.

Effects/RefTexFragmentLightDof	Per fragment lighting with one light source, Assignment of Texture, First render pass of depth-of-field rendering: Rendering the scene, Depth encoded into alpha channel.	Effects/RefTransFragmentLightDof	See configuration of RefTransFragmentLightDof	First pass of depth-of-field rendering has to be applied to every node that is rendered. The resulting image then contains the scene's depth values as alpha value, which is used for further processing. If other effects (e.g. bump mapping etc.) have to be applied on the nodes, then also the corresponding shaders have to be adapted to meet the depth-of-field requirements. Supports per fragment lighting.
Effects/RefTexDofBlur	Assignment of texture, second render pass of depth-of-field: blurring the texture, simulating an optical circle of confusion, caused by lens refraction out of focus.	Core/RefTrans	See configuration of RefTrans	Second pass of depth-of-field rendering: blurring the incoming image. Result is a blurred image of the passed texture. u_CocScale controls the size of the circle of confusion, and therefore the blur strength.
Effects/RefTexDofCombine	Third pass of depth of field: Assignment and blending of two textures, representing the sharp and blurred image. Blend weights result from focus and focus range of the simulated camera lens.	Core/RefViewportSpace, Effects/Ref	See configuration of RefTrans	Third pass of depth rendering: Combines the sharp and the blurred image according to the set focus of the simulated camera lens. u_Focus controls the distance of the focus plane (where a sharp image would be formed). u_Range controls the size of the area around the focus plane, where a sharp image would be drawn.
Effects/RefTexGaussianBlurX	Assignment of texture, Subsequent pass of post-processing effect: blurring the texture with a Gaussian blur, Gaussian blur step width from unifrom	Core/RefViewportSpace, Effects/Ref	See configuration of RefViewportSpace plus SetUniform("u_blurFactor",Shader::Float, &blurFactor)	Subsequent pass of post-processing effect: blurring the incoming image horizontally. Result is a blurred image of the passed texture. u_blurFactor controls the step width of the Gaussian function. u_blurFactor should always be 1.0/TextureWidth so that every step will correlate with one pixel.

Effects/RefTexGaussianBlurY	Assignment of texture, Subsequent pass of post-processing effect: blurring the texture with a Gaussian blur, Gaussian blur step width from unifrom	Core/RefViewportSpace, Effects/Ref	See configuration of RefViewportSpace plus SetUniform("u_blurFactor",Shader::Float, &blurFactor)	Subsequent pass of post-processing effect: blurring the incoming image vertically. Result is a blurred image of the passed texture. u_blurFactor controls the step width of the Gaussian function. u_blurFactor should always be 1.0/TextureHeight so that every step will correlate with one pixel.
Core/RefFlatLight1	Per fragment lighting with one light source for Flat Shading.	Core/RefTransPos	See configuration of RefTransPos	In order to use derivative functions dFdx and dFdy the extension GL_OES_standard_derivatives has to be enabled and supported by driver.
Effects/RefGoochShading	Per Fragment calculation of specular component, Mixing cool and warm color using the dot product of light direction and vertex normal as weight, Output: Mixed color + specular and ambient lightcomponent.	Effects/RefTransLight1Gooch	See configuration of RefTransLight1Gooch	This fragment shader in conjunction with RefTransLight1Gooch.vertp realizes the first pass of an implementation of Gooch's rendering technique to simulate the appearance of technical illustrations, often occuring in e.g. manuals or schematic presentations etc. (SIGGRAPH'98 paper "A Non-Photorealistic Lighting Model For Automatic Technical Illustration"). For details please refer to the description of the vertex shader, and to the cited paper. The following unifiorms are needed: u_CoolColor and u_CoolColor are 4-component vectors that describe the colors for the cool to warm transition. u_CoolDiffuseWeight and u_WarmDiffuseWeight are float values that describe, how the original diffuse color is combined with the warm and cool colors.
Effects/RefBloomThreshold	Assignment of texture, Subsequent pass of bloom effect: extracting only the bright parts of the image that should be blurred, brightness threshold from unifrom	Core/RefViewportSpace, Effects/Ref	See configuration of Core/RefViewportSpace	Subsequent pass of post-processing bloom effect: extracting only the bright parts of the incoming texture. Those bright parts are blurred in the next step to generate a bloom effect.

Effects/RefCombineBloom	Subsequent pass of bloom effect: Assignment and djustable combination of the original and the bloomed texture	Core/RefViewportSpace, Effects/Ref	See configuration of RefViewportSpace	Subsequent pass of post-processing bloom effect: The incoming texture of the original scene and the incoming texture of the bright, bloomed parts are combined. The blending of the two textures is adjustable by uniforms representing the intensity and saturation of the images.
Core/RefInterlaceMask	Uses a texture's color to create a mask through discarding pixels with low transparency. No value is written to the discarded fragments, so also no depth or stencil values.	Core/RefViewportSpace	See configuration of RefViewportSpace	Is especially useful for e.g. Creating stencil masks for stereoscopic 3D.
Effects/RefCarbon	Wraps a seamless texture multible times around the object to achieve carbon look.	Effects/RefTransLight1Carbon	See configuration RefTransLight1Carbon	Creates a car paint with typical carbon look.
Effects/RefFlopCarPaint	Mixes the two color components together based on the camera position and a user defined bias	Effects/RefTransLight1FlopCarPaint	See configuration RefTransLight1FlopCarPaint	Creates a car paint with shiny, anisotropic two color look.
Effects/RefMetalFlakesCarPaint	Adding a noise to the spectral part of the light. Noise can be wrapped around the object multible times. Create the illusion of metal flakes in the car paint.	Effects/RefTransLight1MetalFlakesCarPaint	See configuration RefTransLight1MetalFlakesCarPaint	Creates a car paint with shiny, sparkling look produced by metal flakes within specular highlight

Effects/RefFlopMetalFlakesCarPaint	Mixes the two color components together based on the camera position and a user defined bias. Adding a noise to the spectral part of the light. Noise can be wrapped around the object multible times. Create the illusion of metal flakes in the car paint.	Effects/RefTransLight1FlopMetalFlakesCarPaint	See configuration RefTransLight1FlopMetalFlakesCarPaint	Creates a car paint with shiny, sparkling look produced by metal flakes within specular highlight combined with anisotropic two color look.
Core/RefCanvasUniDiffuseMat	Assignment of material's diffuse color to fragment color	Core/RefCanvasTransUniDiffuseMat	See configuration RefCanvasTransUniDiffuseMat	This shader is used to achieve the same effect as a SolidColorBrush in 2D.
Effects/RefFxaa	Coordinates in normalized device space.	Effects/RefFxaa	See configuration of Effects/RefFxaa	Applies the "Fast approximate anti-aliasing" algorithm to its input, which should be a quad using normalized device coordinates. The uniform (u_Resolution) needs to be set to the width and height of the input texture measured in texels.
Effects/RefGradientLinear	Mixing fragment color according to gradient computations.	Effects/RefTransGradientLinear	See configuration RefTransGradientLinear	Fragment Shader for linear gradient computations.
Effects/RefGradientRadial	Assignment of radial gradient color to fragment color.	Effects/RefTransGradientRadial	See configuration RefTransGradientRadial	Fragment Shader for radial gradient computations.
Core/RefColorTexAlpha,	Assignment of product of varying color and texture color to fragment color.	Core/RefTransColor	See configuration RefTransColor	This fragment shader shall be used for text or alpha mask rendering.

Effects/RefColorTexAlphaOutline	This fragment shader renders an outline of color "u_outlineColor" and width specified by "u_outlineWidth" around the contents in the texture "u_Texture". The input texture is also multiplied by the vertex color. If the "u_outlineWidth" value is bigger than 10, the contents of the texture need be bigger than 10 too, otherwise artefacts will start to appear with increasing outline width. There is no instancing version of this shader, because it will usually be applied to text rendering which is batched by Canvas rather than the Renderer.	Core/RefTransColor	See configuration RefTransColor Additionally: SetTexelSizeActivationEnabled(true) Custom uniforms: uniform vec4 u_outlineColor; // Color of outline. uniform float u_outlineWidth; //Width of outline.	This fragment shader shall be used if a text or an alpha mask shall receive an outline.

Revision #15
Created 2023-02-22 01:45:31 UTC by Tsuyoshi.Kato
Updated 2026-08-18 06:17:16 UTC by Tsuyoshi.Kato