Morphing

Description

This chapter briefly describes the morphing technique supported by Candera.

See also:

- Morphing Mesh in SceneComposer User Manual

Example Solutions

Refer to following solutions in folder *cgi_studio_player/content/Tutorials/08_SpecialRenderTechniques/0801_Special3DRenderTechniques*

- MorphSolution
- MultiMorphing_MultiMorphShader
- MultiMorphing_AnimationGroup

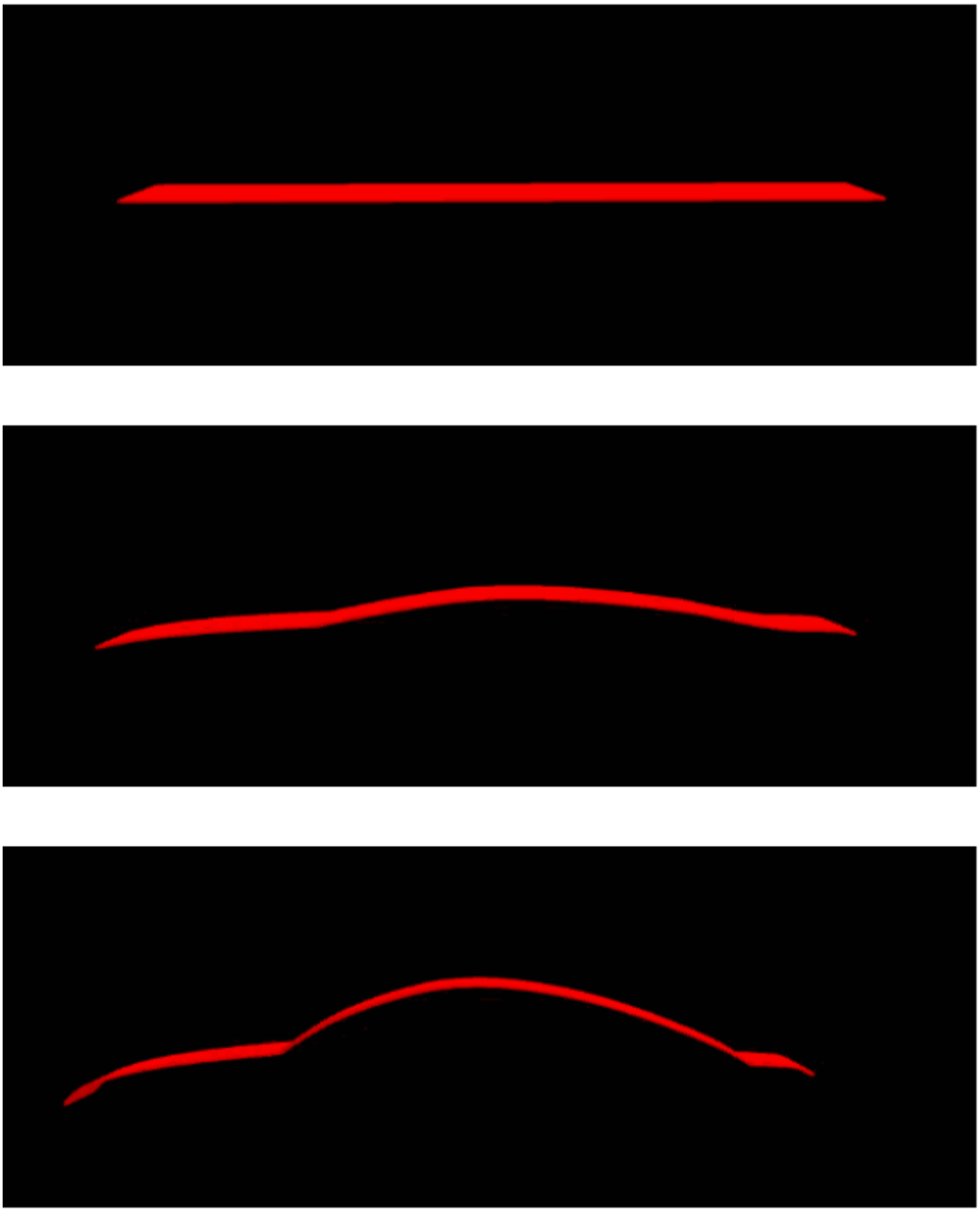
Introduction

Morphing

Morphing means transforming the 3D geometry of an object smoothly from one shape into another.

Morphing itself (smooth interpolation between vertices) is done in a shader (not on the CPU).

Morping example



What you need for Morphing

Suitable content: "Morph targets" are different geometric states of the same object. The input vertices must correspond one-to-one between morph targets. This means, to morph between two shapes, you need **two VertexBuffer** with exactly the **same number of vertices**.

An instance of MorphingMesh: This is a Mesh that stores

- the spliced vertex buffer created from the vertex buffers to be morphed (see MorphingMesh::CreateSplicedVertexBuffer)

- weight values for each of the shapes, which are activated as shader uniforms at render time (see MorphingMesh::SetMorphWeight)
- The appearance of the MorphingMesh must define a suitable morphing shader, which matches the number of shapes to be morphed (see **RefTransLight1Morph** vertex shader for morphing two shapes)

Dynamics: Morphing is about change, so an application will have to set some morphing parameters (often depending on time). Use Animation::AllMorphWeightsPropertySetter to animation all the weight values of the MorphingMesh.

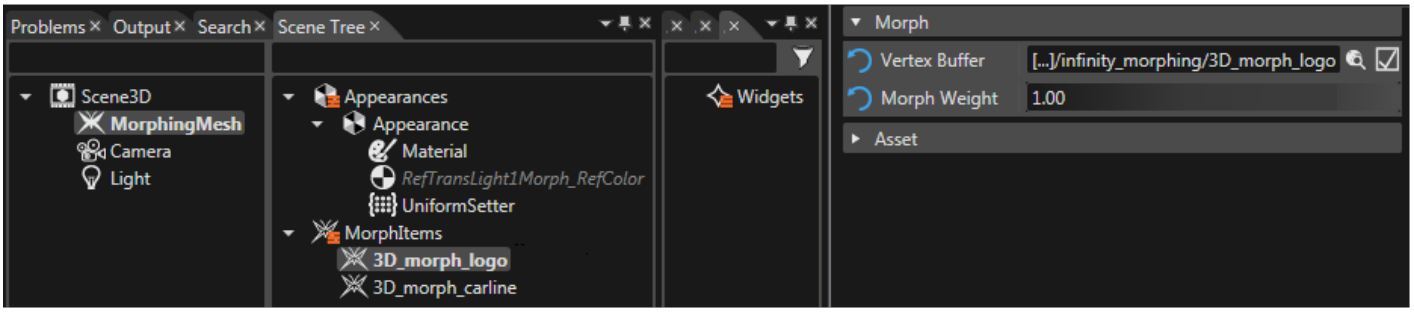
Morphing Examples

Morphing between 2 Morph Items

In below example solution, the reference shader **RefTransLight1Morph_RefColor** is used to morph between two morph items, by default having weight value 1,00 on the logo.

- *MorphSolution* from folder *cgi_studio_player/content/Tutorials/08_SpecialRenderTechniques/0801_Special3DRenderTechniques*

MorphingMesh setup for two morphing shapes



The animation part of this solution animates both weight values and also adds a rotation animation.

Morphing between Multiple Shapes using Shader

Since a MorphingMesh is capable of handling up to 8 different vertex buffers, a special morphing shader must be used for more than 2 shapes. A morphing shader handling 5 shapes is used in the example below:

- *MultiMorphing_MultiMorphShader* from folder *cgi_studio_player/content/Tutorials/08_SpecialRenderTechniques/0801_Special3DRenderTechniques*

Shader code for morphing 5 shapes

TransLight1Morph5.vert

```
449
450   /*---------------------------------- MAIN ----------
451   void main(void)
452   {
453       vec3 position =
454           a_Position4.xyz * u_MorphWeight[4]
455         + a_Position3.xyz * u_MorphWeight[3]
456         + a_Position2.xyz * u_MorphWeight[2]
457         + a_Position1.xyz * u_MorphWeight[1]
458         + a_Position.xyz  * u_MorphWeight[0];
459       vec3 norm =
460         normalize(
461           a_Normal4.xyz * u_MorphWeight[4]
462         + a_Normal3.xyz * u_MorphWeight[3]
463         + a_Normal2.xyz * u_MorphWeight[2]
464         + a_Normal1.xyz * u_MorphWeight[1]
465         + a_Normal.xyz  * u_MorphWeight[0]);
466       v_TexCoord =
467           a_TextureCoordinate4.st * u_MorphWeight[4]
468         + a_TextureCoordinate3.st * u_MorphWeight[3]
469         + a_TextureCoordinate2.st * u_MorphWeight[2]
470         + a_TextureCoordinate1.st * u_MorphWeight[1]
471         + a_TextureCoordinate.st  * u_MorphWeight[0];
```

Morphing between Multiple Shapes using AnimationGroup

As an alternative to using a multi morph shader, multi-step morphing can also be realized by using an AnimationGroup, chaining separate morphing animations for each morphing shape pair. This approach is useful in case a specific target shader compiler has some restrictions regarding the maximum number of shader parameters.

- *MultiMorphing_AnimationGroup* from folder *cgi_studio_player/content/Tutorials/08_SpecialRenderTechniques/0801_Special3DRenderTechniques*

Revision #6
Created 2023-02-22 08:38:07 UTC by Tsuyoshi.Kato
Updated 2025-01-10 14:07:58 UTC by Elisabeth Zauner