# Planar Shadows

## Description

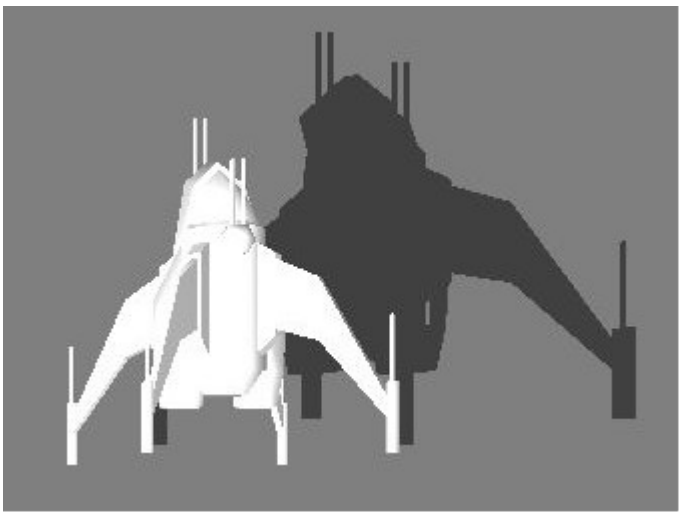
This chapter describes the planar shadow technique supported by Candera.

# Introduction

## Planar Shadows

Planar shadows are used to algorithmically simulate flat shadows on planar surfaces.

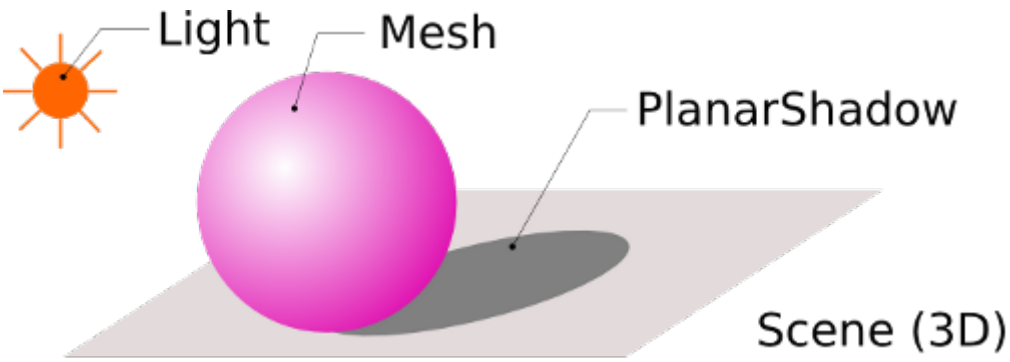
The shadow effect is obtained using a Candera::PlanarShadow object which is drawn using its world transformation matrix multiplied with a projection matrix, that projects its vertices onto a predefined plane with respect to the position of an associated light source.



# Workflow

## Inventory

The figure below shows the basic setup of a scene needed to generate the planar shadow effect.




## Light

Light should be of type point, spot or directional.

```
m_light = Light::Create();
m_light->SetName("Light1");
m_light->SetType(Light::Point);
m_light->SetDiffuse(Color(1.0f, 1.0f, 1.0f));
m_light->SetSpecular(Color(0.0f, 0.0f, 0.0f));
m_light->SetRenderingEnabled(true);
```

## Shadow Caster

The following objects can be used as shadow caster: Candera::LineList, Candera::Billboard and Candera::Mesh.

## Planar Shadow

The Candera::PlanarShadow is the object which actually draws the shadow onto a specified plane. By default, the parent node's vertex buffer is used. Consequently the PlanarShadow node should be attached as child of the shadow caster.

```
m_shadow = PlanarShadow::Create();
m_shadow->GetAppearance()->SetShader(m_shaderShadow); // RefTrans_RefUniDiffuseMat
m_shadow->SetName("Shadow");
m_shadow->SetPlane(Plane(Vector3(0.0f, 0.0f, 1.0f), 75.0f));
m_shadow->SetLight(m_light);
m_current->AddChild(m_shadow); // attach planar shadow to the shadow caster
```

It's also possible to explicitly provide a dedicated vertex buffer which will be rendered as shadow, see the snippet below:

```
                m_shadow->SetAutoVertexBufferEnabled(false);
                m_shadow->GetVertexBuffer()->Unload();
                m_shadow->SetVertexBuffer(m_meshes[1]->GetVertexBuffer());
                m_shadow->GetVertexBuffer()->Upload();
```

Candera::PlanarShadow supports also an optional alignment of the shadow plane to a given node (alignmentNode). If this node it set, the shadow plane's distance and direction are transformed locally according to alignmentNode's transformation matrix. If node is not set, the shadow plane's parameters are not transformed at all. See snipped below to set alignmentNode:

```
m_shadow->SetAlignmentNode(m_meshes[3]);
```

## Overlapping Shadows

To avoid double blending of semi transparent shadows stencil buffers can be used. The PlanarShadow's appearance already contains a pre-configured RenderMode with stencil buffers configured to only draw on fragments with a stencil value other than this PlanarShadows unique Id. After drawing, the value gets replaced to the current stencil reference value, such that the stencil test doesn't pass any more on future fragments to write.

> To get this mechanism running ensure that the camera always clears the stencil buffer to 0 every frame.

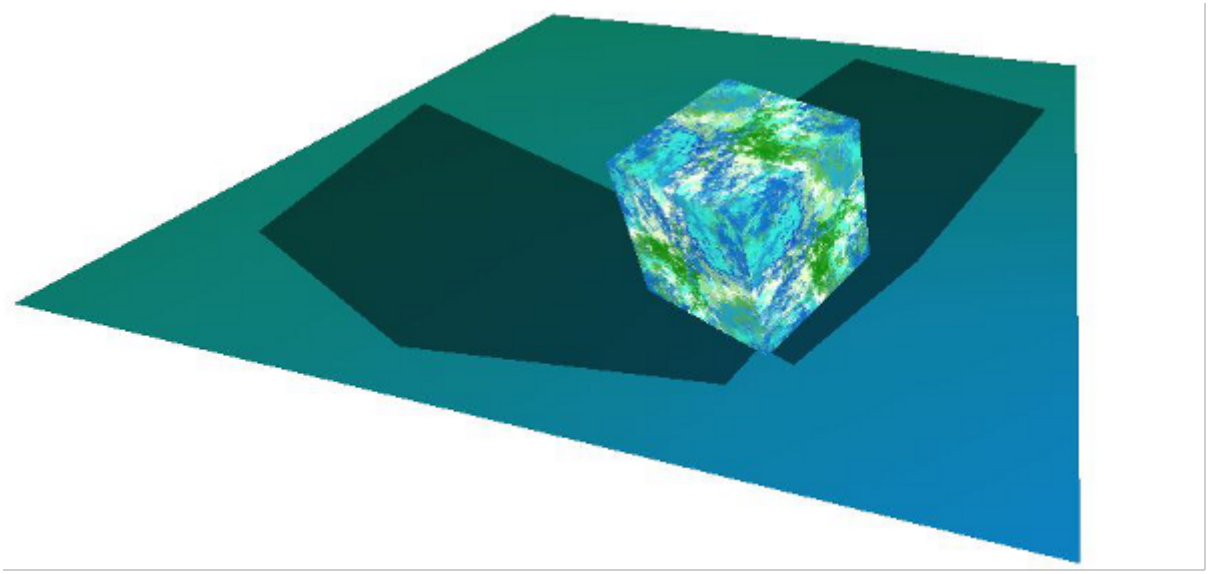
> The GraphicDeviceUnit has to be initialized accordingly to support stencil buffers.

## Constraints

- Only flat planar surfaces are considered as areas to draw the shadow on.
- Every object that intends to cast a shadow, must have a dedicated shadow child node for each light source, as well as for each plane acting as shadow receiver.
- If vertex buffer transformations shall be considered (e.g. morphing meshes or displacement mapping), a dedicated shader has to be implemented for the shadow object.

# Example

The example shows two planar shadows casts by a cube.



## Planar Shadow Example in SceneComposer

To experiment with the Planar Shadow in SceneComposer use solution **PlanarShadow** from folder *cgi_studio_player/content/Tutorials/08_SpecialRenderTechniques/Special3DRenderTechniques*.

---

Revision #6
Created 2023-02-22 08:38:16 UTC by Tsuyoshi.Kato
Updated 2025-01-17 08:42:40 UTC by Elisabeth Zauner