

Texture Projection

Note: The Katana scene file for this example - **textureProjection.katana** - is included in `$KATANA_RESOURCES/Examples/katana_files`.

In this example, we use texture projections as layer masks.

Texture projection using a camera coordinate system can be done via the following shaders:

- [PxrProjector](#) is a **projection manifold** that points to the camera coordinate system. This manifold is required.
- [PxrProjectionLayer](#) reads a **projection texture** using a projection manifold. Using [PxrProjectionLayer](#) is not required if we are projecting a procedural pattern.
- [PxrProjectionStack](#) combines [PxrProjectionLayer](#) nodes. Using [PxrProjectionStack](#) is not required if there is only one projection layer.

Projection Manifold

To create a projection manifold:

- Create a **CameraCreate** node as you would normally. Position the camera for the texture projection. The camera can be perspective or orthographic. It is important to name the camera appropriately, especially if there will be more than one projection camera.
- Create a **CoordinateSystemDefine** node. This is required for defining the coordinate system for RenderMan.

[blocked URL](#)

- **coordinateSystemName** sets the coordinate system name. Note that this will be the exact name set in the [PxrProjector](#) node (see below).
- **referenceLocation** sets the location of the projection camera created above, in the [CameraCreate](#) step.

It's a good idea to double-check the RIB to make sure the [CoordinateSystem](#) and its transformation are emitted. For example:

```
TransformBegin
Translate 0.99622 1.84374 12.2671
Rotate -1.98663e-06 0 0 1
Rotate 0.572957 0 1 0
Rotate 9.74029 1 0 0
Scale 1 1 1
CoordinateSystem "frontProjCamCoordSys"
TransformEnd
```

- Connect the [CameraCreate](#) node into the [CoordinateSystemDefine](#) node. The order of connection is important.

[blocked URL](#)

- Create a [PxrProjector](#) node.
 - Select the **Projection** type.
 - Specifying a **Coordinate System** is required for all types of projection. This is the coordinate system that is defined via [CoordinateSystemDefine](#) above. The names must match exactly.
 - Adjust the occlusion, camera, and texture manifold parameters.
 - Note that *invertT* is on by default. Depending on the camera orientation, *invertT* may need to be unchecked if the texture looks flipped.

[blocked URL](#)

Projection Layer

The [PxrProjectionLayer](#) node is similar to [PxrTexture](#) except it is optimized for using with [PxrProjector](#) and it handles the projection mask appropriately.

[blocked URL](#)

- **Filename** sets the texture for projection. If the texture is procedural, any pattern node can be used instead of [PxrProjectionLayer](#).
- Wire the result output from the [PxrProjector](#) node to the **Manifold** parameter.
- Wire the resultMask output from the [PxrProjector](#) node to the **Mask** parameter.

If necessary, adjust the **Missing Color** and **Missing Alpha** for the area that are missed by the projection.

Projection Stack

The [PxrProjectionStack](#) node combines the projection layer nodes.

[blocked URL](#)

It is tricky in Katana to connect a dynamic array of shader nodes, so we need to create a `ShadingNodeArrayConnector` that holds the array. Here are the steps:

- Create a `PxrProjectionStack` node. This node combines the projection layers.
- **Layers Mode** - Click on the *Add* button to create the number of layers needed.
- **Layers RGB** - Since this will be connected to an array of shading nodes, unlike Layers Mode above, there is no need to click on the *Add* button.
- Create a `ShadingNodeArrayConnector` node for the projected colors. Rename the node to make it descriptive, e.g. `projectionLayers_RGB`.
- Wire the `resultRGB` of the `PxrProjectionLayer` nodes to the appropriate index of the `ShadingNodeArrayConnector`. Note that `i0` is for the top-most layer.
- **Layers A** - This will also be connected to an array of shading nodes, so, once again, do not click on *Add*.
- Create another `ShadingNodeArrayConnector` node, for the projected alphas. Rename the node to make it descriptive, e.g. `projectionLayers_A`.
- Wire the `resultA` from the `PxrProjectionLayer` nodes (if your projected texture has an alpha) or `resultMask` from the `PxrProjector` nodes to the appropriate index of `ShadingNodeArrayConnector`. Note again that `i0` is the top most layer.

[blocked URL](#)

Wire `resultRGB` of the Projection Stack to any color parameter for a pattern or material (`Bxdf`) and you'll see the projected texture!