Sample Code

Enabling video reflections in an immersive environment

Create a more immersive experience by adding video reflections in a custom environment.

Download (1.2 GB)

visionOS 2.0+ | Xcode 16.0+

Overview

RealityKit and Reality Composer Pro provide the tools to build immersive media viewing environments in visionOS. The Destination Video sample uses these features to build a realistic custom environment called Studio. The environment adds to its realism and makes the video player feel grounded in the space by applying reflections of the player's content onto the surfaces of the scene.

Play ⊙

RealityKit and Reality Composer Pro support two types of video reflections:

- Specular reflections provide a direct reflection of the video content, and are typically useful to apply to glossy surfaces like metals and water.
- Diffuse reflections provide a softer falloff of video content, and are useful to apply to rougher, more organic surfaces.

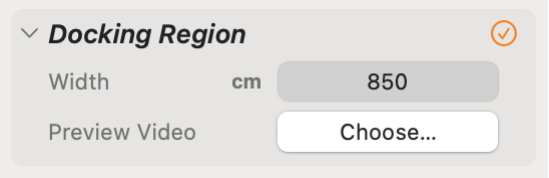
This article describes how to adopt reflections in your own environment, and shows how Destination Video's Studio environment supports these effects to create a compelling media viewing experience.

Define a video docking location

Apps that use `AVPlayerViewController` to present video participate in system docking behavior. When you play a full-window video inside an immersive space, the system docks the video screen into a fixed location and presents streamlined playback controls. By default, the system determines the docking location for the scene, but starting in visionOS 2, you can customize this location by specifying a custom docking region.

The Studio environment defines a custom docking region that anchors the player to the walkway at the top of the staircase like shown below.

To create the docking region, the project defines a `Player` entity that contains a `DockingRegionComponent`. This component defines the bounding region for the player, which has a depth of 0 and uses a fixed 2.4:1 aspect ratio. You configure the docking region's size through its `width` property, and you can optionally specify a preview video to display in the docking region's space within Reality Composer Pro.

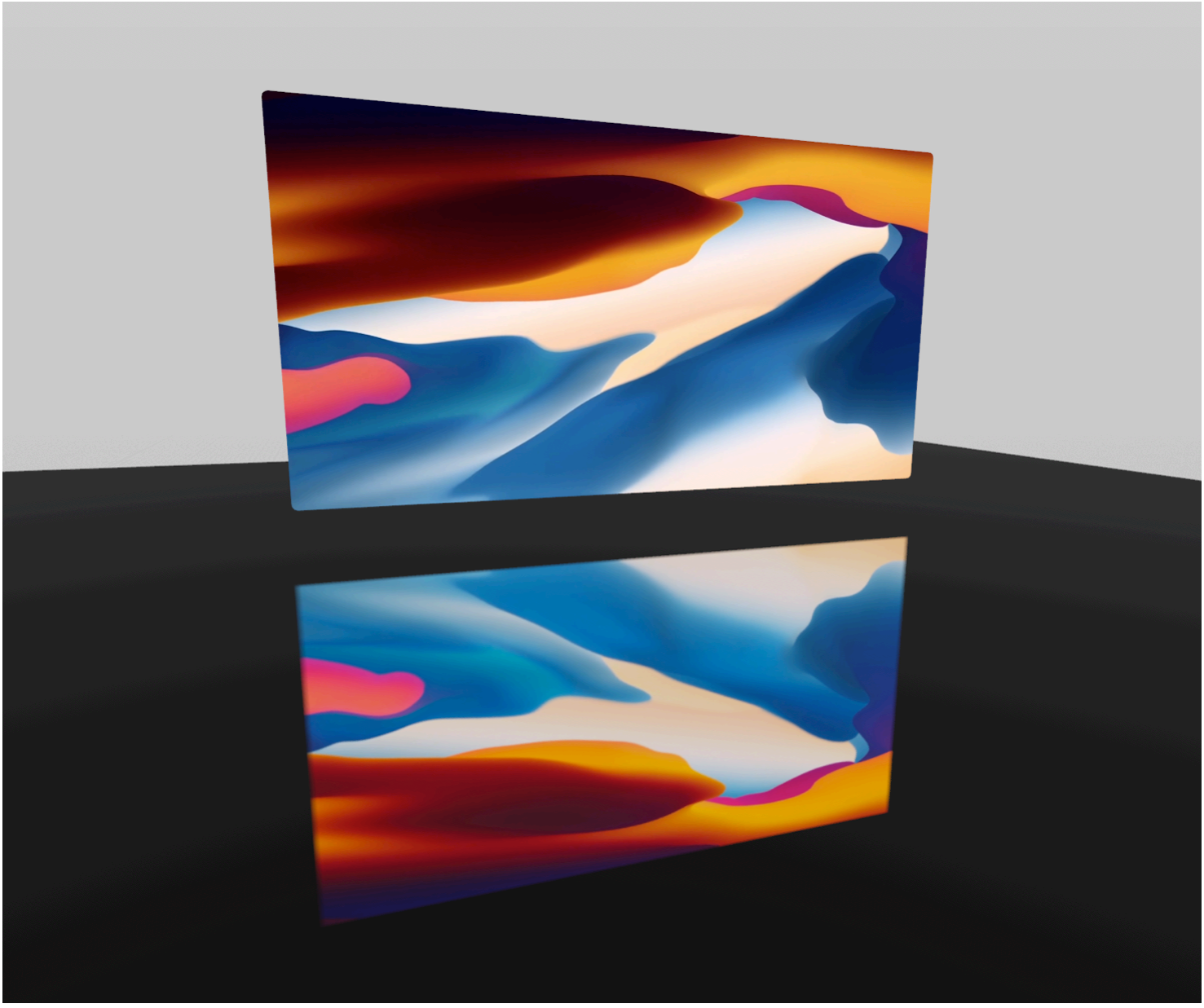


To provide an optimal viewing experience, the Studio environment minimizes objects between the viewer and the video. Additionally, it provides a comfortable viewing angle to avoid causing strain or discomfort during longer viewing sessions. Using Reality Composer Pro to define the docking region is a great way to visualize how it looks in context, but always review your environment on Apple Vision Pro to get a true sense of layout and scale.

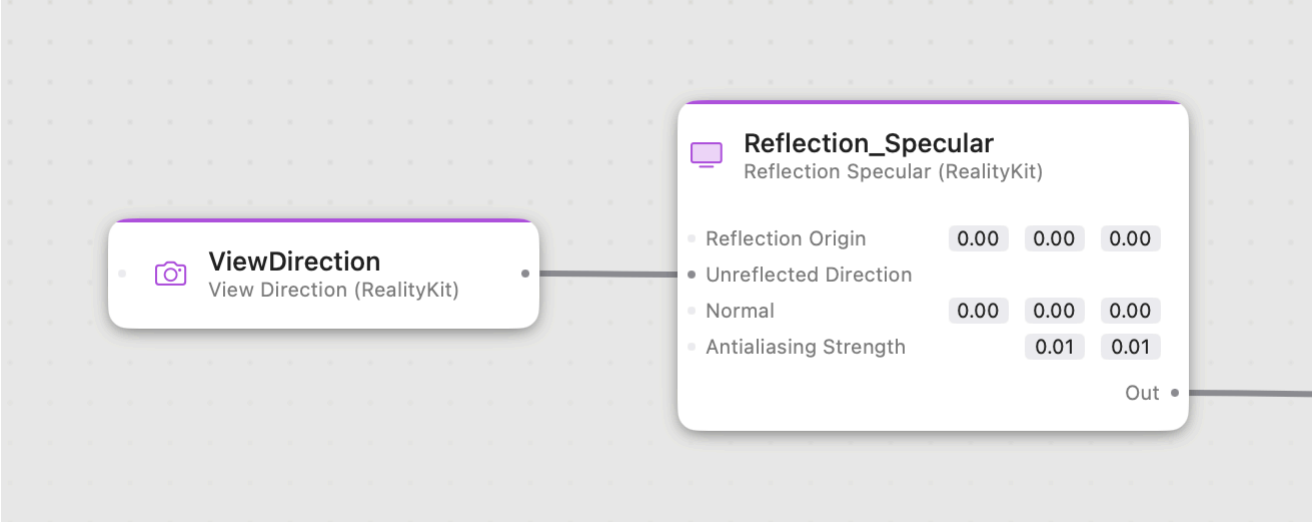
> **Note**
>
> Reality Composer Pro provides a template to set up a docking region and default video reflection configuration. You can access this template from the Insert menu by selecting Insert > Environment > Video Dock.

Display specular video reflections

Specular reflections, like shown below, provide a direct reflection of the video's content onto surrounding surfaces. You typically apply this type of reflection to glossy surfaces such as metals, mirrors, and water.



To enable this type of reflection, you define a material with the `Reflection Specular (RealityKit)` node connected and apply it to a surface in your scene. The system automatically calculates the appropriate reflection based on your viewing angle relative to the docking region.



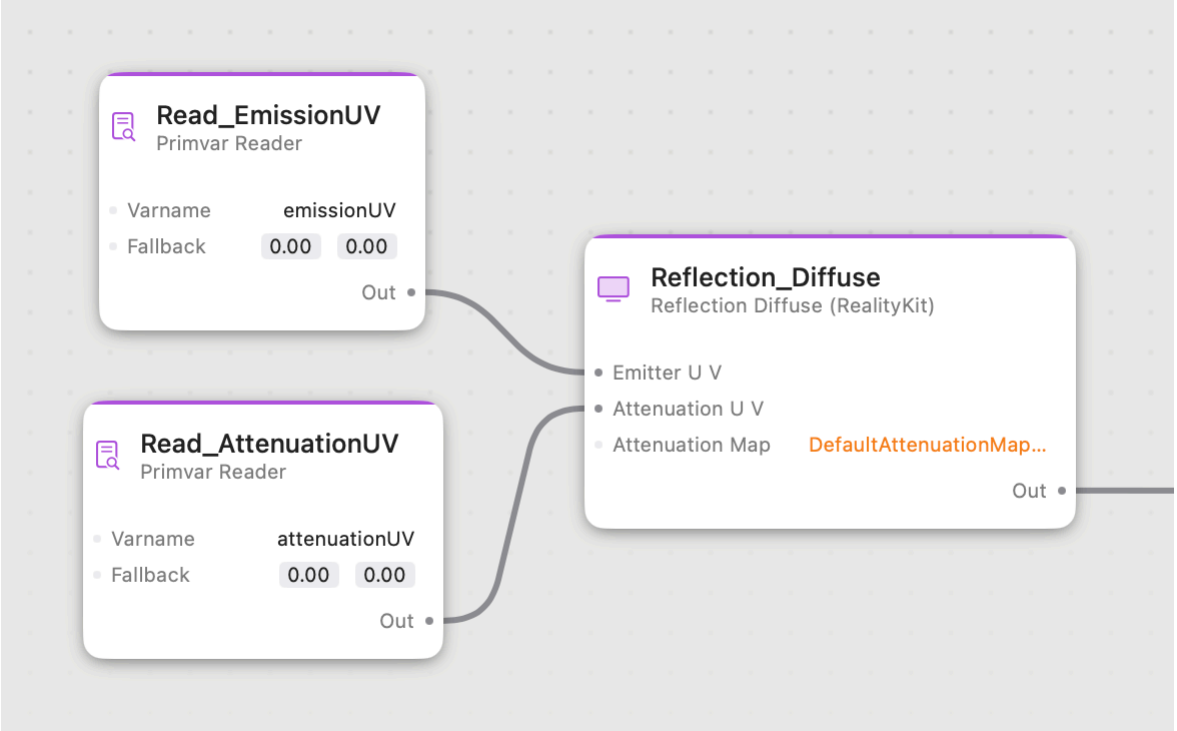
The output of this node contains the reflected color in the RGB channels, and a blend factor in the alpha channel, which you can use to composite the reflection with your existing material.

Destination Video uses subtle specular reflections in its custom environment like shown below. Applying specular reflections helps to add depth and space to the experience. To learn more about how the environment uses specular reflections, open the Studio project in Reality Composer Pro to view its configuration.

Provide diffuse video reflections

Diffuse reflections provide a softer falloff of media content, which can be useful to apply to rougher, more organic surfaces like a concrete or wood floor. The image below shows a diffuse reflection from a video screen.

You enable diffuse reflections by adding a material on a surface with the `Reflection Diffuse (RealityKit)` node connected.



This node requires the following inputs:

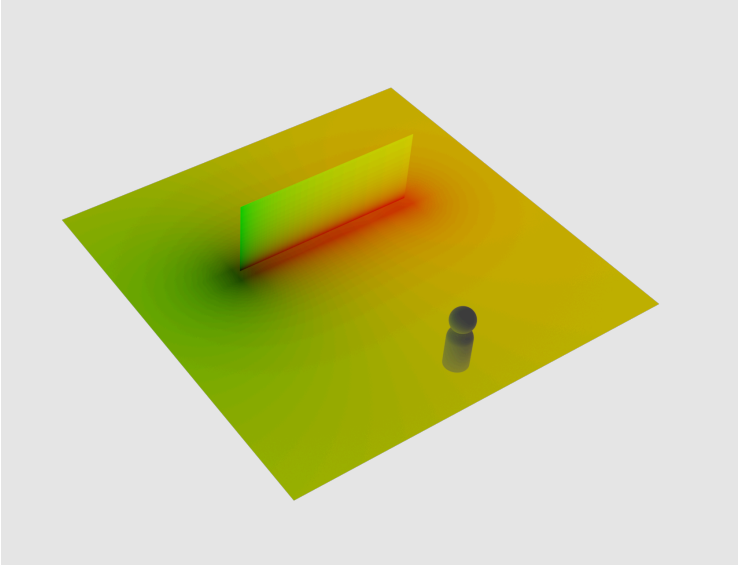
Emitter UV
: This UV samples the system-generated emitter texture that contains low-frequency light and color information from the docked video. The diffuse reflection node uses the UV to calculate where to show the soft light reflection.

Attenuation UV
: This UV samples the provided attenuation mask texture. An attenuation texture contains a soft falloff mask that's used to shape the light from the emitter. Use a higher bit-depth texture format, such as `.exr`, to reduce any possible banding artifacts.

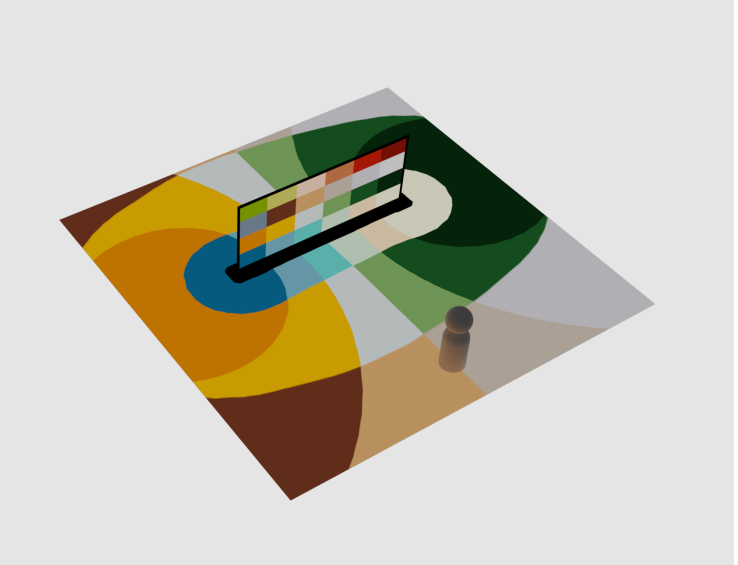
Destination Video's custom environment applies diffuse reflections to the surfaces immediately surrounding the docked video screen as shown below:

Enabling diffuse reflections enhances the level of immersion by making the video player feel grounded in the experience.

To calculate Emitter UVs, iterate over each vertex of the surface mesh, and sample a set number of random points on the docking region. The u-value and v-value of each of the random sample points on the docking region are weighted by measuring both the distance and the angle to the mesh vertex. The resulting emitter UV set is the average of the weighted docking region UV values.



A visualization of the emitter UVs generated from the docking region.

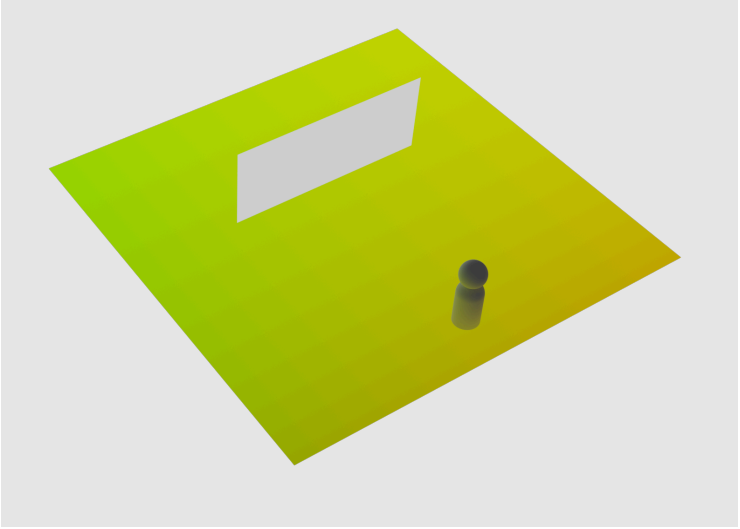


An example that uses a debug texture to show how different colors from the docking region map on to the surface mesh.

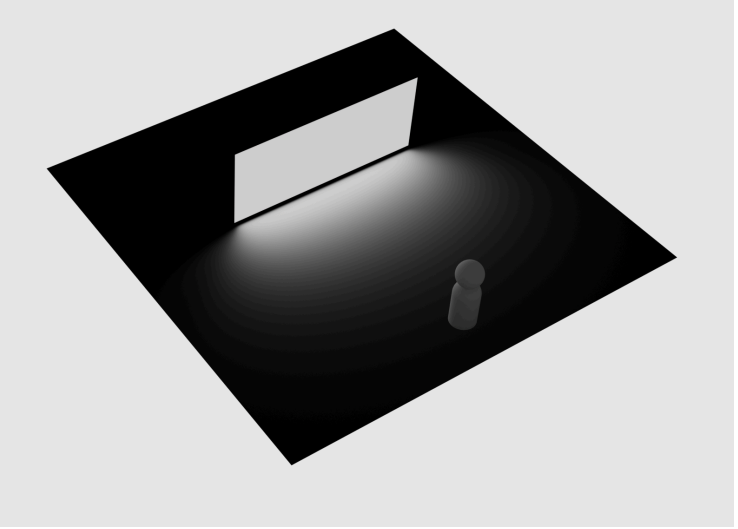
Important

The number of random points sampled from the docking region can have a large impact on the overall computation time when generating emitter UVs. You can configure how many samples to use when calculating emitter UVs with the ComputeDiffuseReflectionUVs python script.

Attenuation UVs are a top-down projection of the attenuation texture onto the input geometry (UV-coordinate system). An attenuation texture contains a soft falloff mask that shapes the light from the emitter.

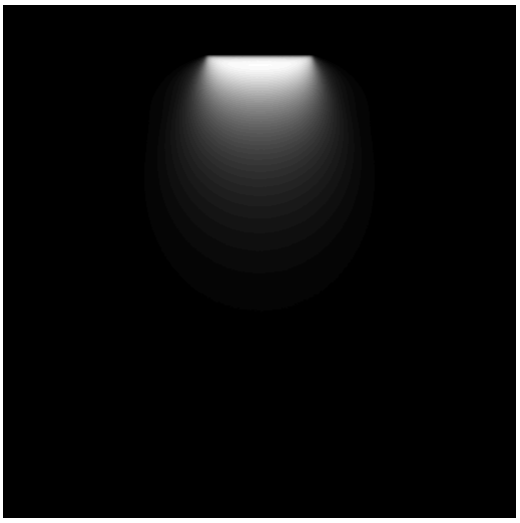


A visualization of the attenuation UVs generated from the docking region.

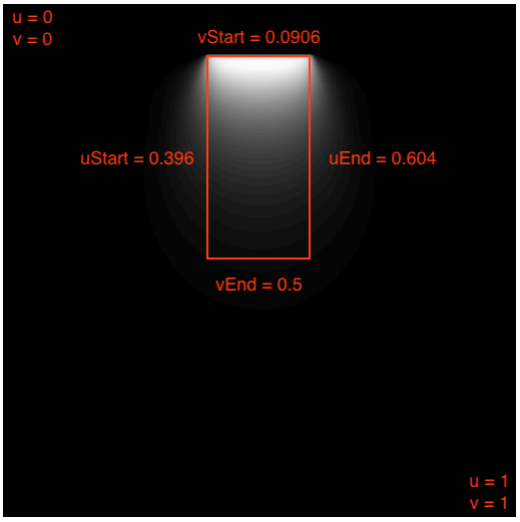


Reality Composer Pro's default attenuation texture on the visualization.

The attenuation texture contains a falloff pattern that shapes the the diffuse reflection on to the surface mesh. The image below shows the default Reality Composer Pro attenuation texture.



The default falloff pattern doesn't extend all the way to the edges of the texture. In order to generate the attenuation UV set, calculate the edges of the falloff pattern from the texture. The image below shows the default falloff pattern in a standard UV-coordinate system, with the top-left point equal to `(0,0)` and the bottom-right point equal to `(1,1)`.



The following four values define the attenuation UV set:

`uStart`
: The UV-space value where the sharp line of the falloff pattern starts horizontally.

`uEnd`
: The UV-space value where the sharp line of the falloff pattern ends horizontally.

`vStart`
: The UV-space value where the sharp line starts vertically.

`vEnd`
: The UV-space value where the falloff pattern ends in black.

After calculating the attenuation texture, map it to the geometry. To visualize the attenuation texture mapping, the image below shows a square red mesh as the custom surface mesh that extends towards the user with sides that are equal to the width of the docking region.

The attenuation UVs are calculated from mapping the surface mesh, in world space, to the area defined by the `uStart`, `uEnd`, `vStart`, and `vEnd` values, in the UV-coordinate space. The image below shows the surface mesh with the attenuation texture applied.

> **Note**
>
> When using the ComputeDiffuseReflectionUVs python script for mapping using a custom attenuation texture, you only need to measure the the `uStart`, `uEnd`, `vStart`, and `vEnd` values of your attenuation texture. If you're using the default attenuation texture in Reality Composer Pro, the script uses the default values.

To learn more about how the environment sets up and applies diffuse reflections, open the Studio project in Reality Composer Pro to view its configuration.

See Also

Related samples

{} Destination Video

Leverage SwiftUI to build an immersive media experience in a multiplatform app.

Related articles

{} Building an immersive media viewing experience

Add a deeper level of immersion to media playback in your app with RealityKit and Reality Composer Pro.

Related videos

Enhance the immersion of media viewing in custom

Create custom environments for your immersive apps in

environments visionOS