Displaying 3D Content in a SpriteKit Scene

Draw SceneKit content in a SpriteKit scene by using a 3D node.

Overview

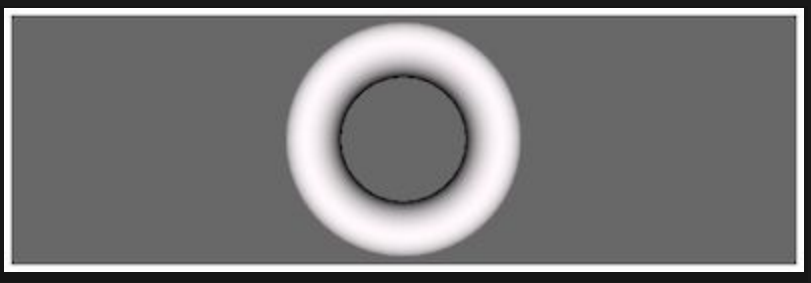
SceneKit content rendered in SpriteKit is automatically assigned a camera and, because `autoenablesDefaultLighting` defaults to true, lights. That means you require very little code to add simple 3D primitives to your scene. The following code shows how to create a simple scene containing a torus and display it in a SpriteKit scene.

```
let scnScene: SCNScene = {
    let scnScene = SCNScene()

    let torusGeometry = SCNTorus(ringRadius: 10, pipeRadius: 3)
    let torusNode = SCNNode(geometry: torusGeometry)
    torusNode.eulerAngles = SCNVector3(x: CGFloat.pi / 2, y: 0, z: 0)
    scnScene.rootNode.addChildNode(torusNode)
    return scnScene
}()

let node = SK3DNode(viewportSize: CGSize(width: 200, height: 200))
node.scnScene = scnScene
```

After node is added to a `SKScene`, the 3D torus is visible:



Control How Your Content Is Rendered

Although SK3DNode creates a default camera automatically, you can create your own camera for precise control over how the 3D content is rendered. The following code shows how you can create a 3D node with an explicitly created camera that looks at the first object in the SceneKit scene's node tree.

```
let node = SK3DNode(viewportSize: CGSize(width: 200, height: 200))
node.position = position
node.scnScene = scnScene
node.name = "3dnode"
let camera = SCNCamera()
let cameraNode = SCNNode()
cameraNode.camera = camera
if let lookAtTarget = scnScene.rootNode.childNodes.first {

    let constraint = SCNLookAtConstraint(target: lookAtTarget)
    cameraNode.constraints = [ constraint ]
}
node.pointOfView = cameraNode
node.pointOfView?.position = SCNVector3(x: 0, y: 0, z: 20)
```

Set the Position and Orientation of Your 3D Content

You can create many instances of SK3DNode, each sharing the same SceneKit scene but each with an independent point of view. By updating the position of each 3D node's point of view, you can create code that simulates a top-down, one-point perspective view. The following example shows how to do this by enumerating over all the nodes named 3dnode in the update method of a SKSceneDelegate.

```
scene.enumerateChildNodes(withName: "3dnode") {
    node, _ in
    if let node = node as? SK3DNode {
        let positionX = (width * 0.5 - node.position.x) / 10
        let positionY = (height * 0.5 - node.position.y) / 10
        node.pointOfView?.position = SCNVector3(x: positionX, y: positionY, z: 20)
    }
}
```

The following image shows how this code gives the impression of perspective inside SpriteKit: