mapbox

ARKit

Guanshan Liu + @guanshanliu

Agenda

1. Introduction
2. Measuring Distance
3. Occlusion
4. Face Tracking

Introduction

🧠

ARSession

📣

ARSessionDelegate

ARConfiguration

ARConfiguration

1. AROrientationTrackingConfiguration
2. ARWorldTrackingConfiguration (plane detection, hit testing)
3. ARFaceTrackingConfiguration

📍

ARAnchor

🖼️

ARFrame

Rendering

→ SpriteKit

→ SceneKit

→ Metal

→ Unreal Engine

→ Unity

Placing Objects

```swift
let sceneView = ARSCNView()
sceneView.delegate = self
let configuration = ARWorldTrackingConfiguration()
configuration.planeDetection = .horizontal
configuration.isLightEstimationEnabled = true
sceneView.session.run(configuration)
```

```
@objc func tapped(recognizer :UIGestureRecognizer) {
  let sceneView = recognizer.view as! ARSCNView
  let touchLocation = recognizer.location(in: sceneView)
  if let hit = sceneView.hitTest(
    touchLocation,
    types: [.existingPlaneUsingExtent]
    ).first {
   sceneView.session.add(anchor: ARAnchor(transform: hit.worldTransform))
   return
  } else if let hit = sceneView.hitTest(
    touchLocation,
    types: [.featurePoint]
    ).last {
   sceneView.session.add(anchor: ARAnchor(transform: hit.worldTransform))
   return
  }
}
```

ARSCNViewDelegate

```
func renderer(
        _ renderer: SCNSceneRenderer,
        didAdd node: SCNNode,
        for anchor: ARAnchor)
```

Create and add SCNNode

ARSCNViewDelegate

```
func renderer(
        _ renderer: SCNSceneRenderer,
        didUpdate node: SCNNode,
        for anchor: ARAnchor)
```

Update SCNNode objects, update plane nodes

ARSCNViewDelegate

```
func renderer(
        _ renderer: SCNSceneRenderer,
        didRemove node: SCNNode,
        for anchor: ARAnchor)
```

Clean up

WWDC 2017 Session 602

Introducing ARKit: Augmented Reality for iOS

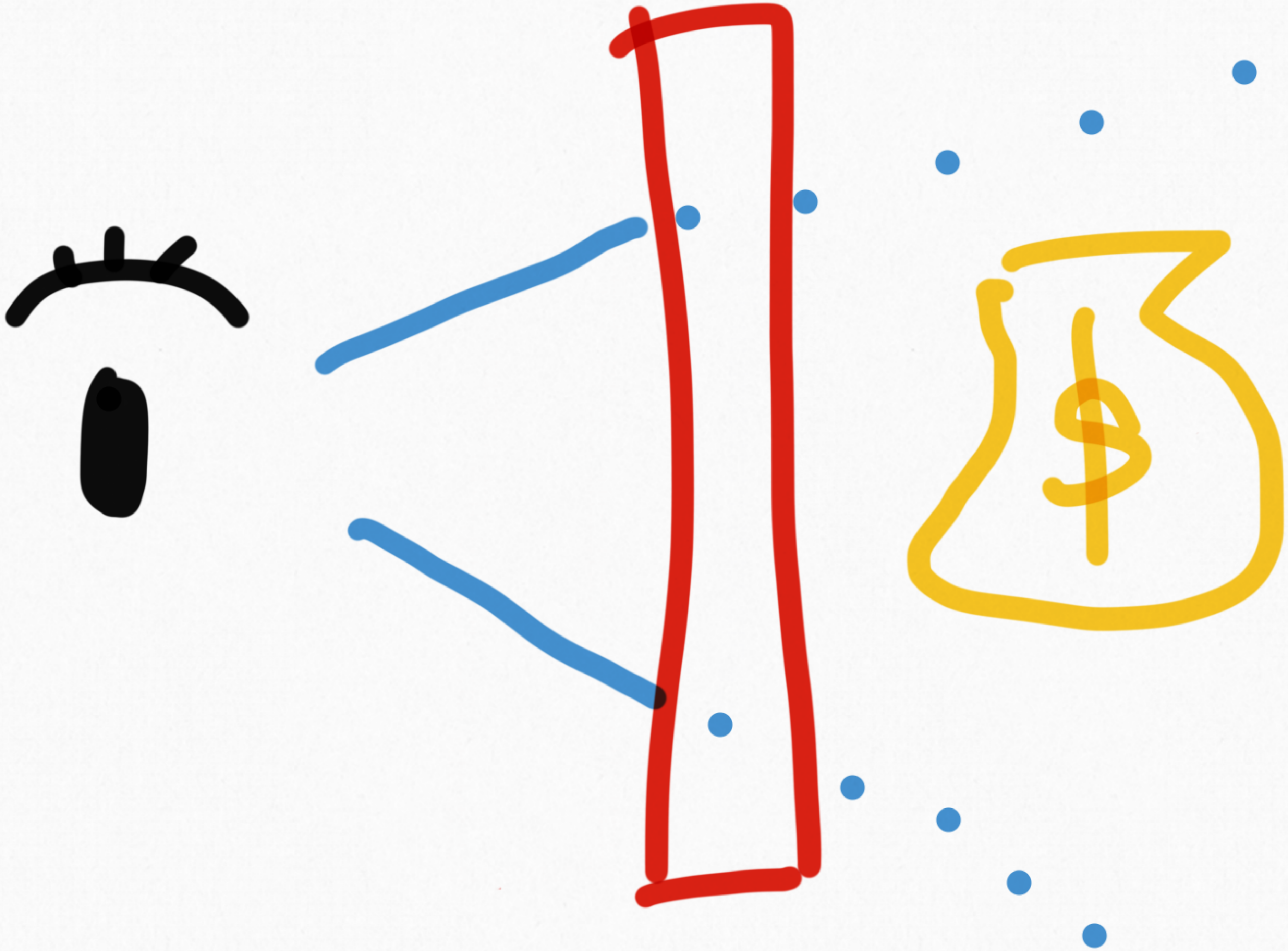
Measuring Distance

Steps

1. Placing two AR objects
2. Calculate the distance between them
3. 1 coordinate point in SceneKit == 1 meter

DEMO

Occlusion



Occlusion Geometry

- Does not render any visible content
- Obstructs the camera's view of other virtual contents

Occlusion Geometry

```
geometry.firstMaterial!.colorBufferWriteMask = []
occlusionNode = SCNNode(geometry: geometry)
occlusionNode.renderingOrder = -1
```

Face Tracking

Run

```
guard ARFaceTrackingConfiguration.isSupported else { return }
let configuration = ARFaceTrackingConfiguration()
configuration.isLightEstimationEnabled = true
session.run(configuration,
            options: [.resetTracking, .removeExistingAnchors])
```

Add node

```
func renderer(_ renderer: SCNSceneRenderer,
        didAdd node: SCNNode,
        for anchor: ARAnchor)
{
    faceNode = node
    serialQueue.async {
        guard let node = self.faceNode else { return }
        for child in node.childNodes {
            child.removeFromParentNode()
        }
        node.addChildNode(self.maskNode)
    }
}
```

Face mesh geometry

```
let device = sceneView.device!
let maskGeometry = ARSCNFaceGeometry(device: device)!
```

Update node

```
func renderer(_ renderer: SCNSceneRenderer,
        didUpdate node: SCNNode,
        for anchor: ARAnchor)
{
    guard let faceAnchor = anchor as? ARFaceAnchor else { return }

    let faceGeometry = maskNode.geometry as! ARSCNFaceGeometry
    faceGeometry.update(from: faceAnchor.geometry)
}
```

DEMO

Blend Shapes

Blend Shapes

> ***An abstract model of the user's facial expressions in the form of a blendShapes dictionary***
> — *Apple*

Blend Shapes

- → faceAnchor.blendShapes
- → 51 `ARFaceAnchor.BlendShapeLocationS`

```
guard let eyeBlinkLeft = blendShapes[.eyeBlinkLeft] as? Float,
let eyeBlinkRight = blendShapes[.eyeBlinkRight] as? Float
else { return }
eyeLeftNode.scale.z = 1 - eyeBlinkLeft
eyeRightNode.scale.z = 1 - eyeBlinkRight
```

DEMO

Recaps

1. ARKit Basics
2. Placing Objects
3. Measuring Distance
4. Occlusion
5. Face Tracking

Resources

- → ARKit
- → ARKit-Sampler by Shuichi Tsutsumi
- → Bjarne Lundgren's GitHub repos
- → Mastering ARKit for iOS on Udemy
- → ARKit for iOS Developers

❣️

from

Guanshan Liu + @guanshanliu