

NAMRATA BANDEKAR

SO YOU WANT TO BUILD AN ARKIT APP



ARKIT BY TUTORIALS

By the raywenderlich.com Tutorial Team

Chris Language, Namrata Bandekar, Antonio Bello & Tammy Coron

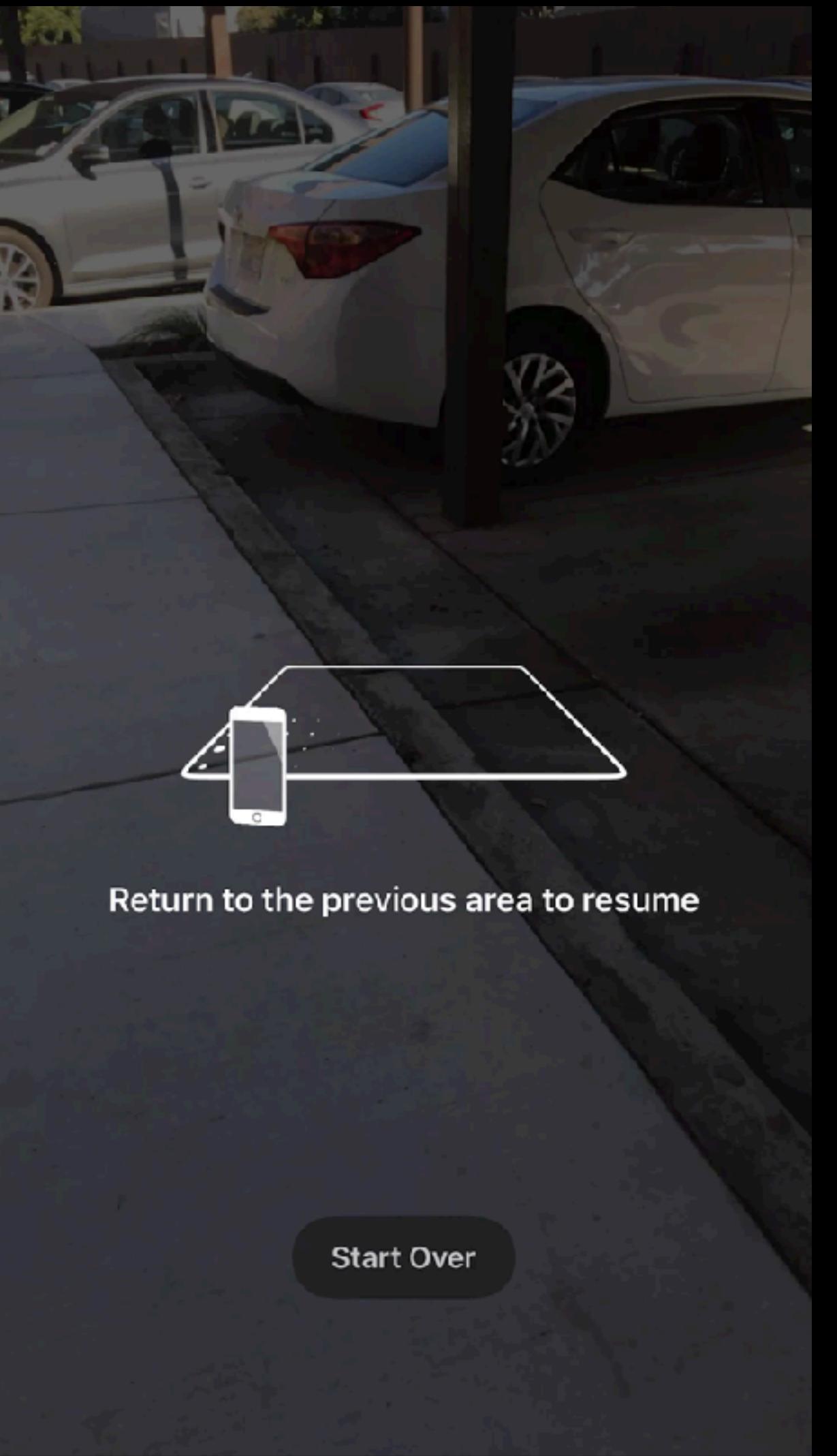
AGENDA



- ▶ Getting Started
- ▶ Diving Deeper
- ▶ Do's and Don'ts

PORTAL

PORTAL

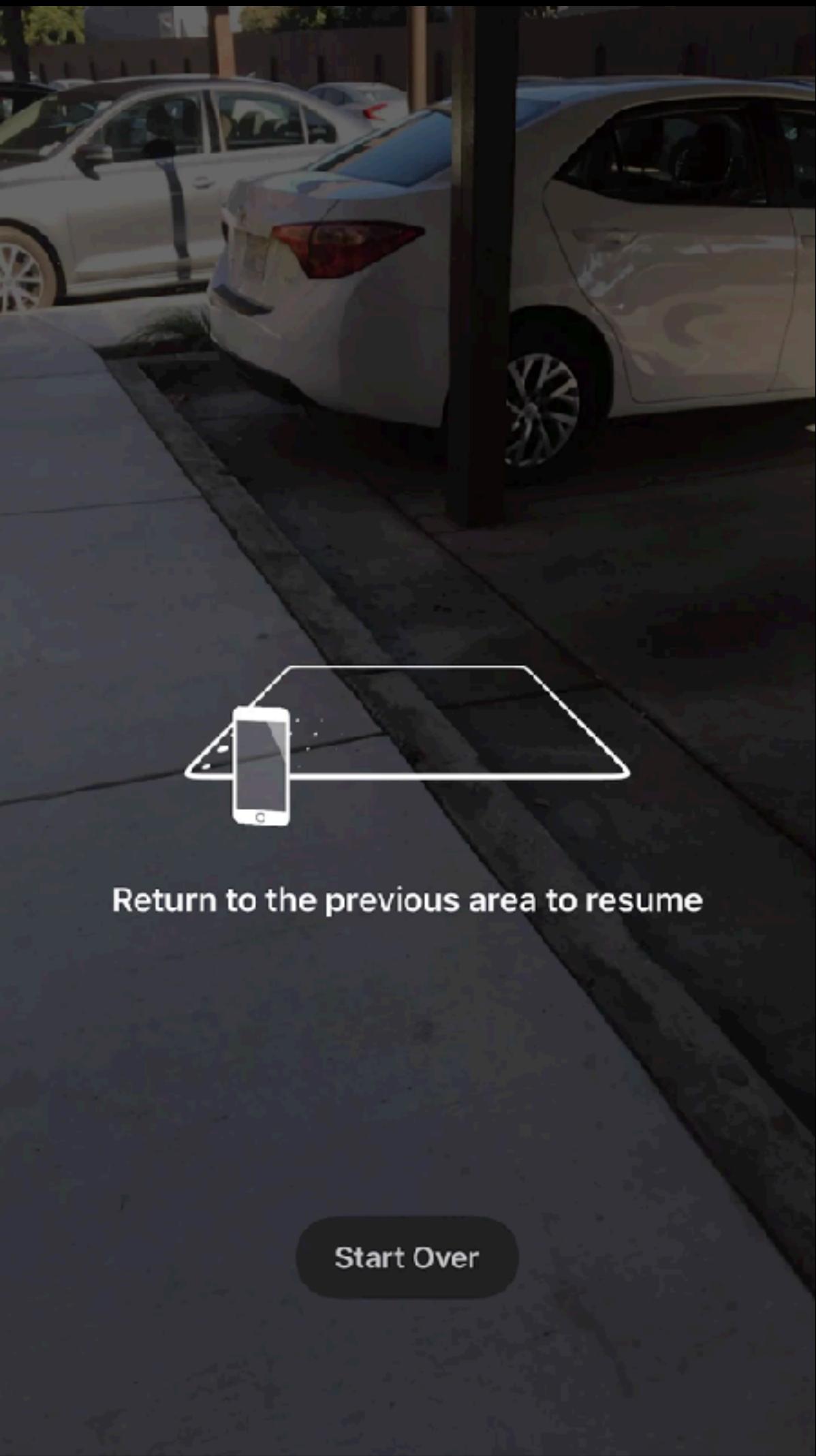


PORTAL

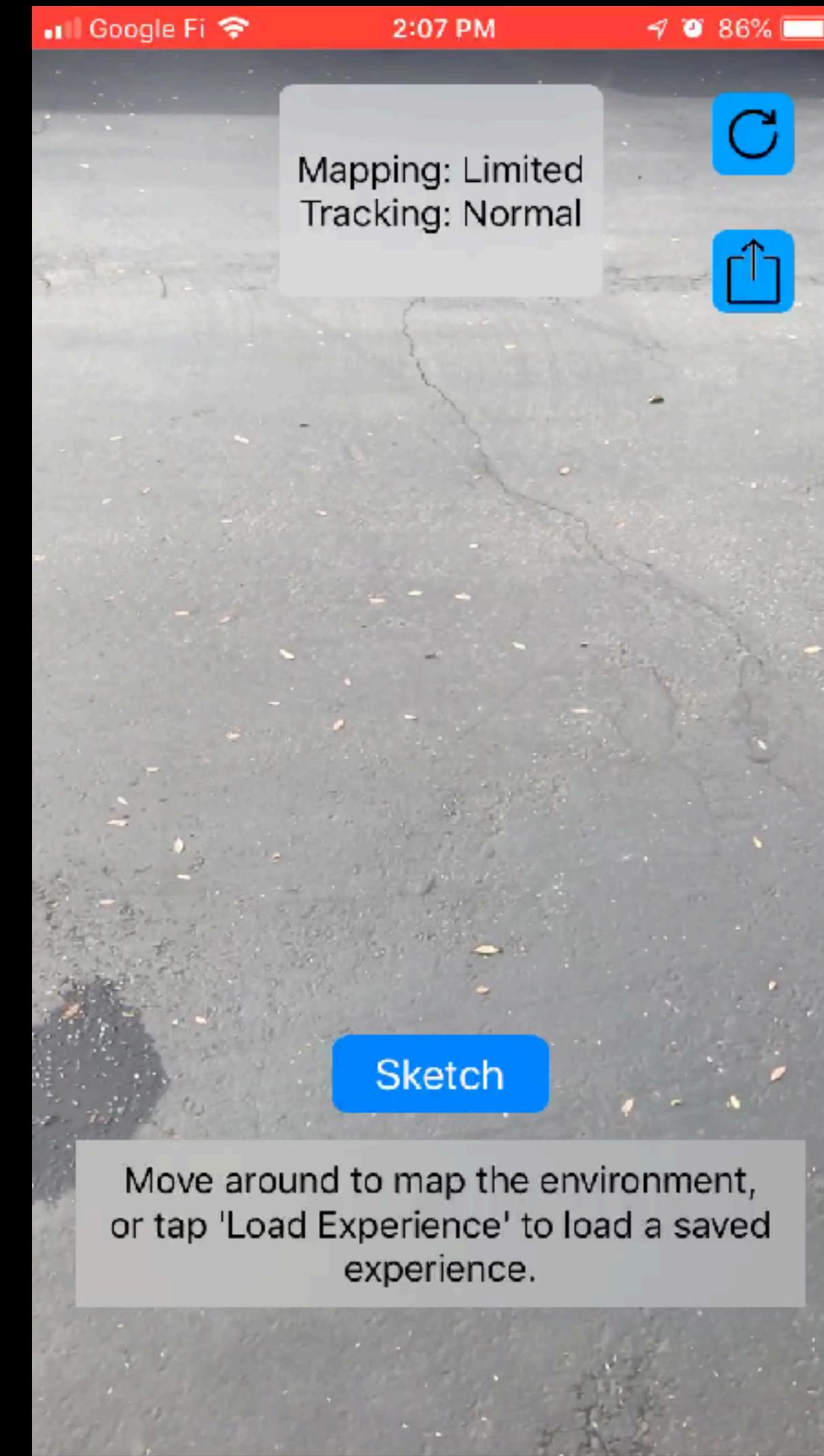
ARSKETCH



PORTAL



ARSKETCH



AUGMENTED REALITY EXPERIENCE

AUGMENTED REALITY EXPERIENCE

- ▶ Detect and understand environment geometry

AUGMENTED REALITY EXPERIENCE

- ▶ Detect and understand environment geometry
- ▶ Render virtual 3D content

AUGMENTED REALITY EXPERIENCE

- ▶ Detect and understand environment geometry
- ▶ Render virtual 3D content
- ▶ User guidance

GETTING STARTED

GETTING STARTED

- ▶ Device Compatibility

GETTING STARTED

- ▶ Device Compatibility
 - ▶ A9 processor (iPhone 6) or later

GETTING STARTED

- ▶ Device Compatibility
 - ▶ A9 processor (iPhone 6) or later
 - ▶ iOS 11 and up

GETTING STARTED

- ▶ Device Compatibility
 - ▶ A9 processor (iPhone 6) or later
 - ▶ iOS 11 and up
- ▶ Handle unsupported devices and iOS versions

GETTING STARTED

- ▶ Device Compatibility
- ▶ A9 processor (iPhone 6) or later
- ▶ iOS 11 and up
- ▶ Handle unsupported devices and iOS versions

```
<key>UILaunchStoryboardName</key>
<string>LaunchScreen</string>
<key>UIMainStoryboardFile</key>
<string>Main</string>
<key>UIRequiredDeviceCapabilities</key>
<array>
    <string>arkit</string>
</array>
```

AR Required

GETTING STARTED

- ▶ Device Compatibility
 - ▶ A9 processor (iPhone 6) or later
 - ▶ iOS 11 and up
- ▶ Handle unsupported devices and iOS versions

GETTING STARTED

- ▶ Device Compatibility
 - ▶ A9 processor (iPhone 6) or later
 - ▶ iOS 11 and up
- ▶ Handle unsupported devices and iOS versions

```
guard ARFaceTrackingConfiguration.isSupported else {  
    updateMessage(text: "Face Tracking Not Supported.")  
    return  
}
```

AR Optional

GETTING STARTED

- ▶ Device permissions

```
<key>LSRequiresiPhoneOS</key>
<true/>
<key>NSCameraUsageDescription</key>
<string>Portal needs to use your camera.</string>
<key>UILaunchStoryboardName</key>
<string>LaunchScreen</string>
```

GETTING STARTED

▶ Privacy and data handling

The screenshot shows a mobile device screen with a privacy policy document. At the top, the status bar displays signal strength, the time (6:34 PM), battery level (86%), and a battery icon. Below the status bar, the title "Privacy Policy" is centered above a blue header bar with a back arrow, a share icon, and a close (X) icon. The main content area starts with a section titled "Privacy Policy of the Medium Facebook app". It includes a note about receiving information about personal data, the purposes, and parties it's shared with. Below this, the "Data Controller and Owner" section lists the address (221B Baker St, Marylebone, London NW1 6XE, UK) and email (example@medium.com). A "Types of Data collected" section notes that the owner does not provide a list of personal data types. It explains that other personal data may be described in other sections or by dedicated explanation text. It also states that personal data may be freely provided by the user or collected automatically. The section goes on to describe the use of cookies for tracking and identifies users as third parties. A "Mode and place of processing the Data" section covers methods of processing, noting security measures like unauthorized access, disclosure, modification, or destruction. It details the use of computers and IT tools, organizational procedures, and involvement of external parties like technical service providers and hosting providers. A "Place" section states that data is processed at the controller's offices and other locations where parties involved are located. A "Retention time" section specifies that data is kept until requested by the user or until purposes are met. A "The use of the collected Data" section is partially visible at the bottom.

DIVING DEEPER

TRACKING



TRACKING

- ▶ ARAnchors fix position in real world 



TRACKING

- ▶ ARAnchors fix position in real world 
- ▶ ARAnchors adjust for drift



TRACKING

- ▶ ARAnchors fix position in real world 
- ▶ ARAnchors adjust for drift
- ▶ Don't use feature points to place content



TRACKING

- ▶ ARAnchors fix position in real world 
- ▶ ARAnchors adjust for drift
- ▶ Don't use feature points to place content



TRACKING QUALITY

TRACKING QUALITY

- ▶ Ambient light

TRACKING QUALITY

- ▶ Ambient light



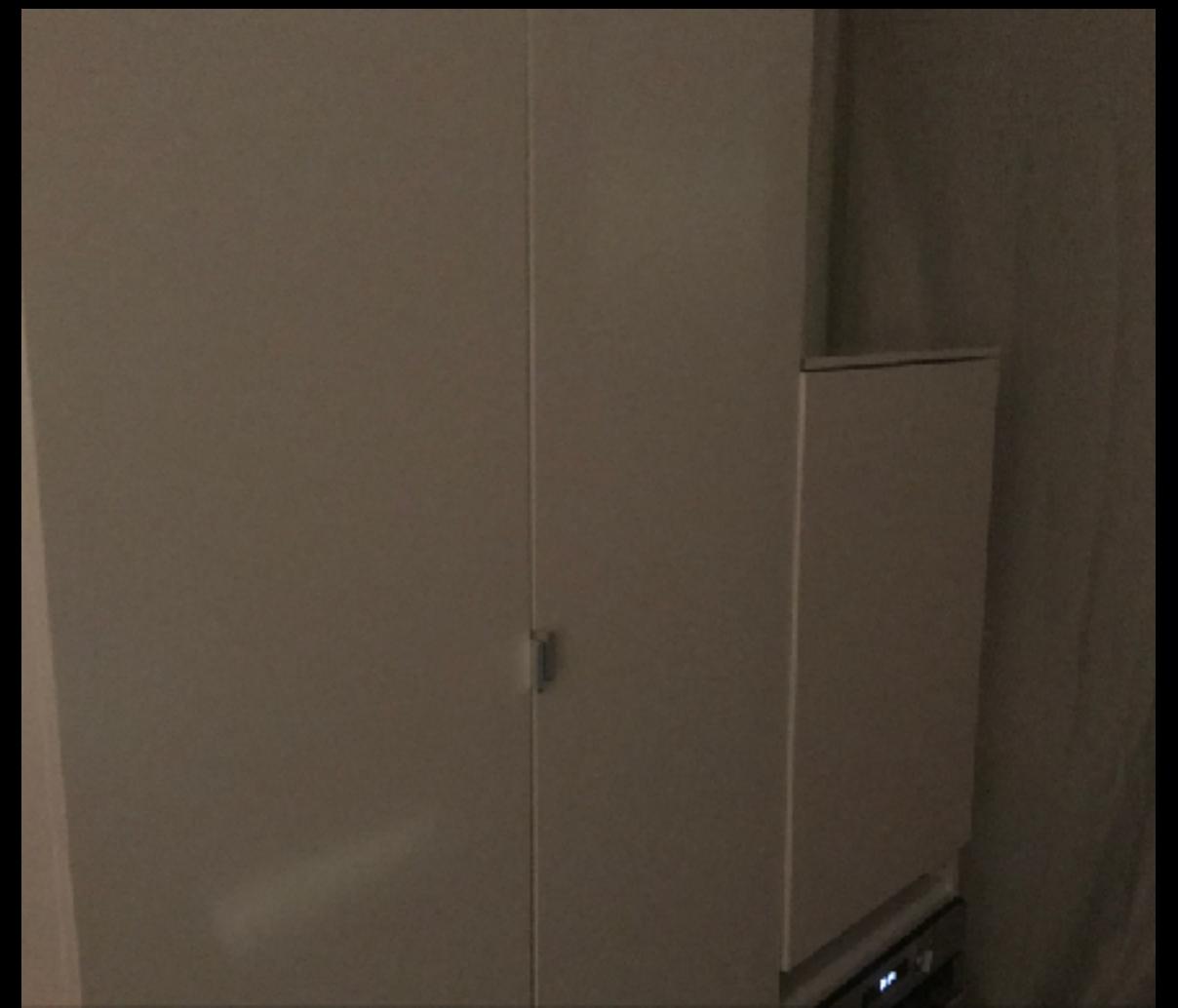
TRACKING QUALITY

- ▶ Ambient light



TRACKING QUALITY

- ▶ Ambient light



TRACKING QUALITY

- ▶ Ambient light



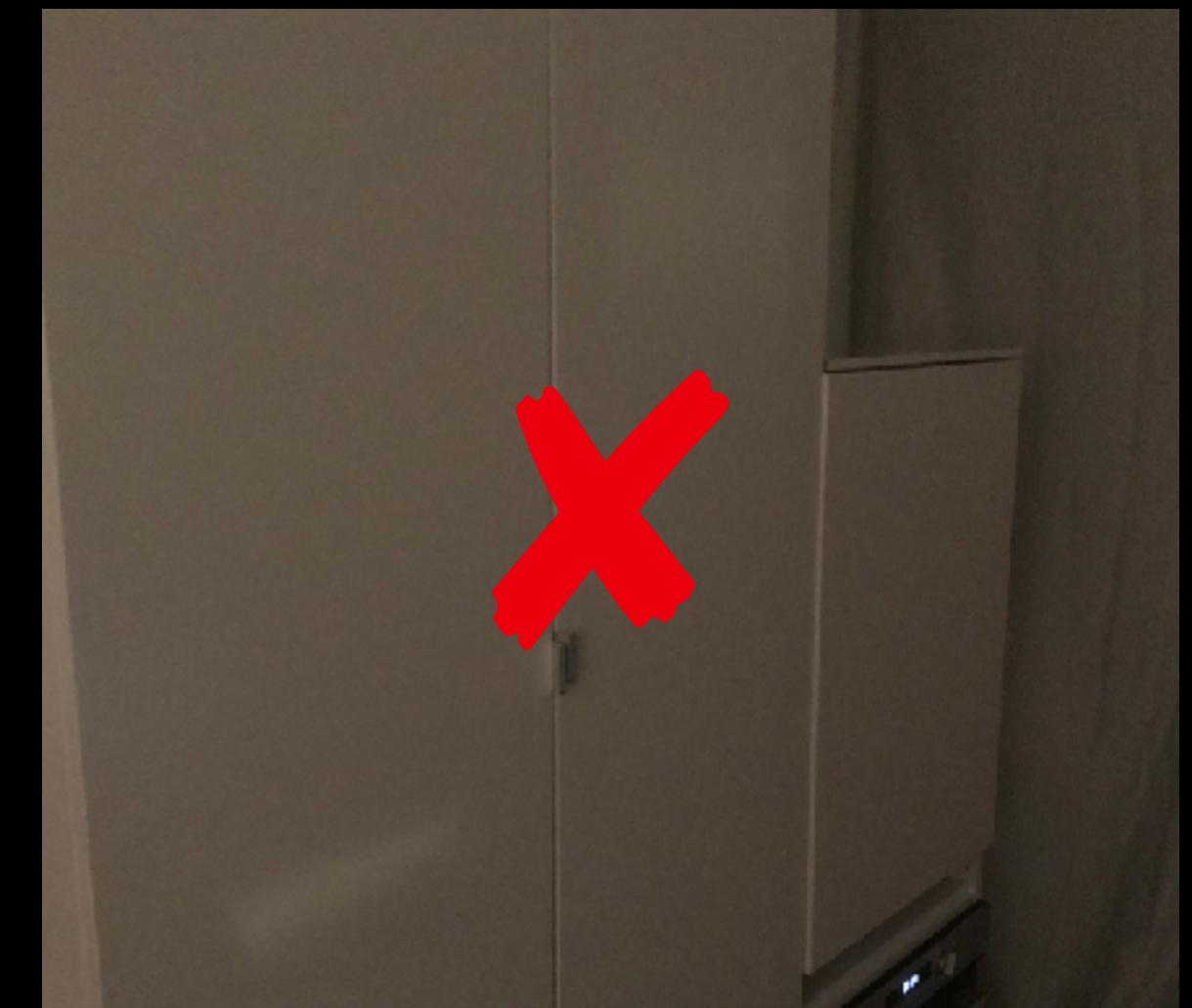
TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments



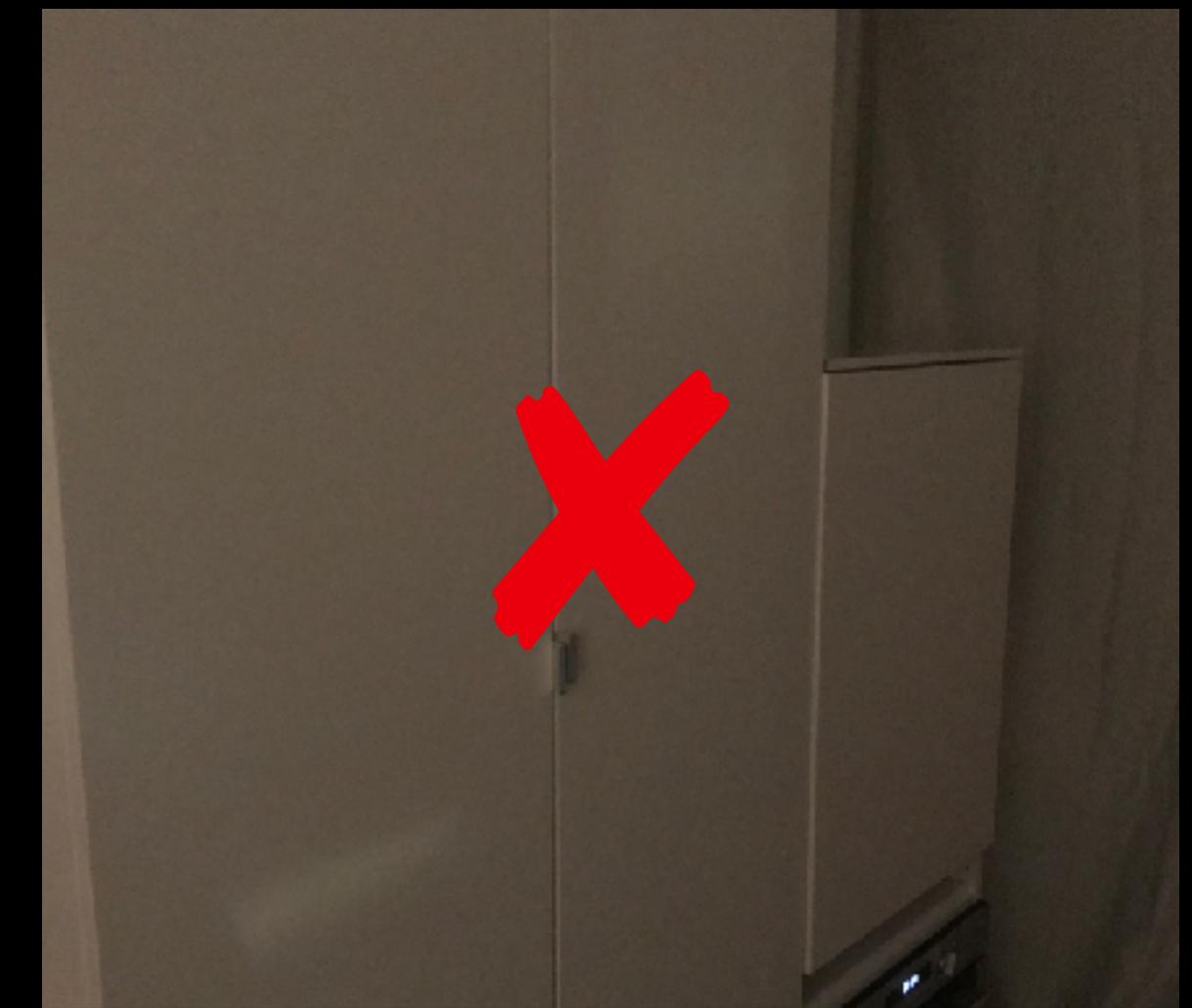
TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments



TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments



TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments
- ▶ Surface texture



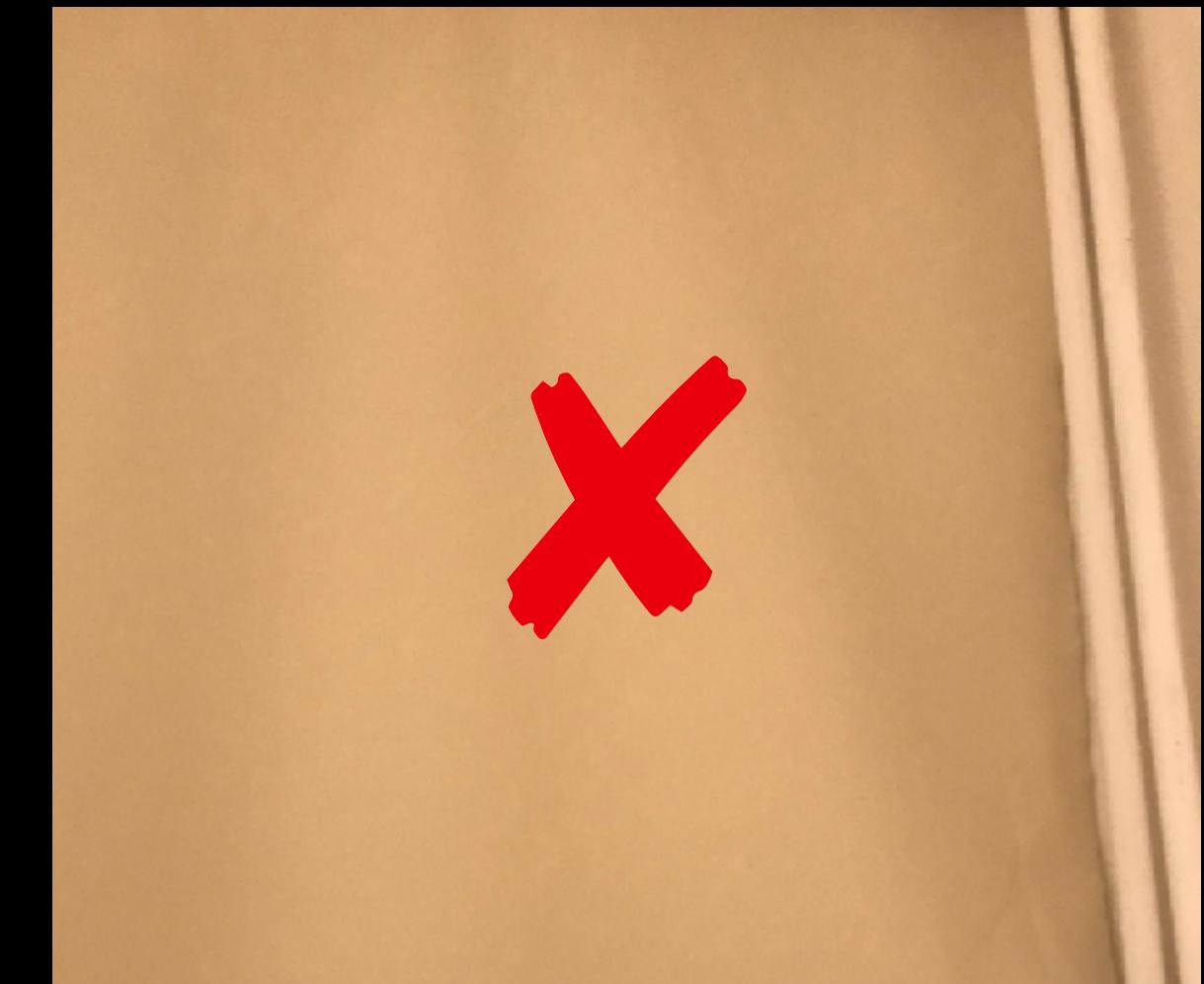
TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments
- ▶ Surface texture



TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments
- ▶ Surface texture



TRACKING QUALITY

- ▶ Ambient light
- ▶ Static environments
- ▶ Surface texture
- ▶ Sensor data quality



TRACKING QUALITY

TRACKING QUALITY

```
ARWorldTrackingConfiguration
```

```
open class ARCamera : NSObject, NSCopying {
```

```
    open var transform: SIMD4X4 { get }
```

```
    open var trackingState: ARTrackingState { get }
```

```
    open var trackingStateReason: ARTrackingStateReason { get }
```

```
    ...
```

```
}
```

TRACKING QUALITY

```
ARWorldTrackingConfiguration
```

```
open class ARCamera : NSObject, NSCopying {
```

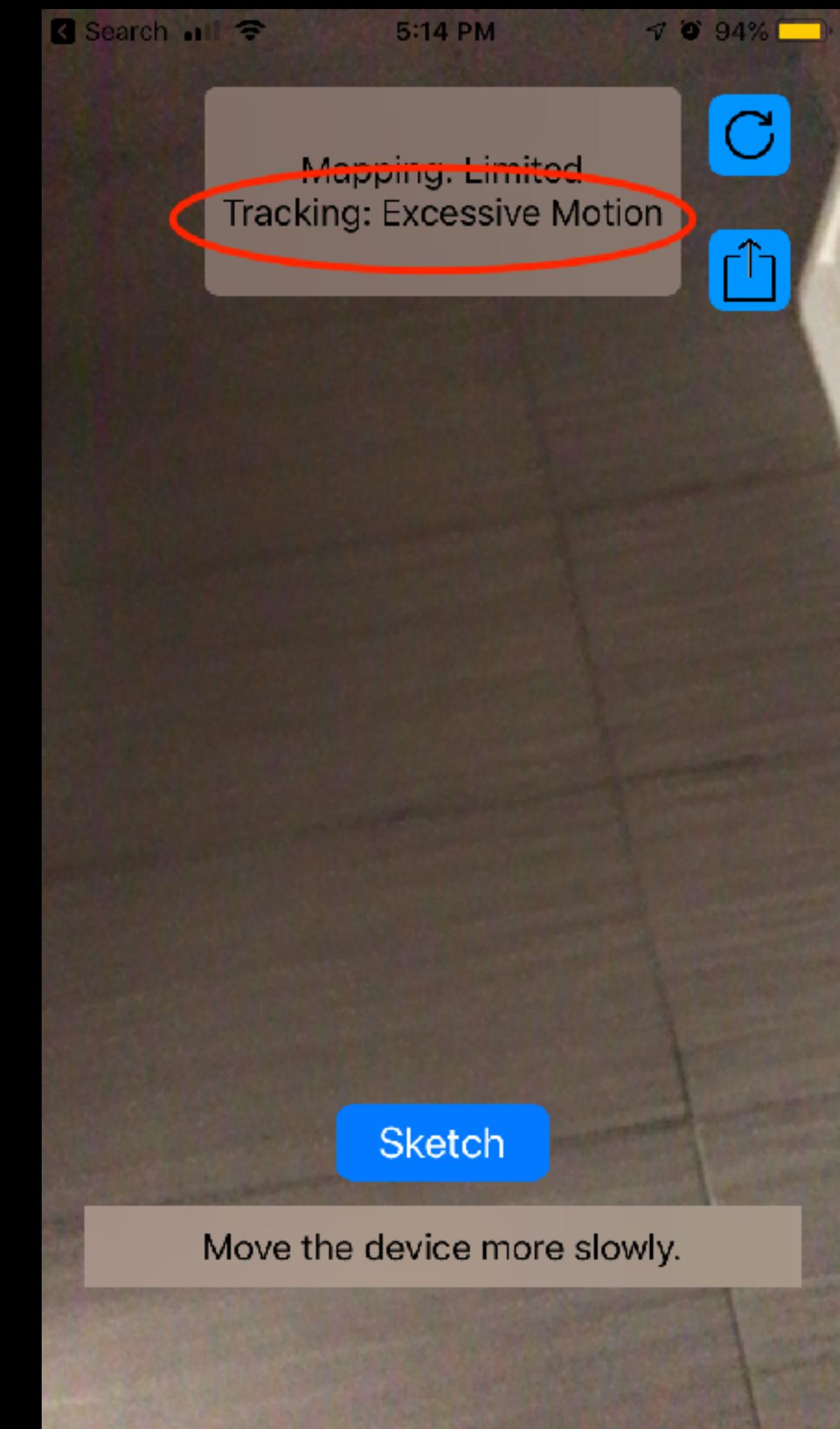
```
    open var transform: SIMD4X4 { get }
```

```
    open var trackingState: ARTrackingState { get }
```

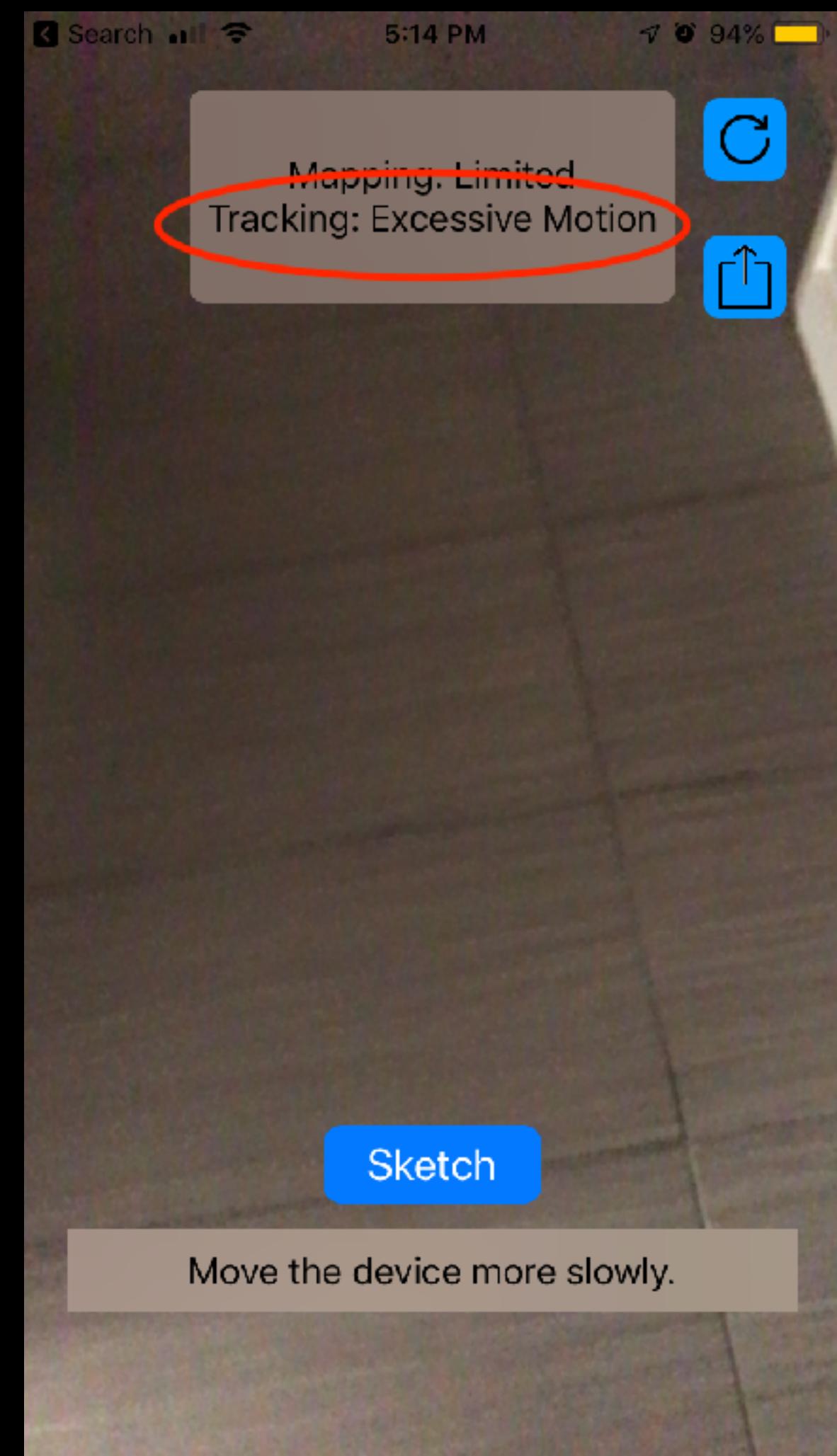
```
    open var trackingStateReason: ARTrackingStateReason { get }
```

```
...
```

```
}
```

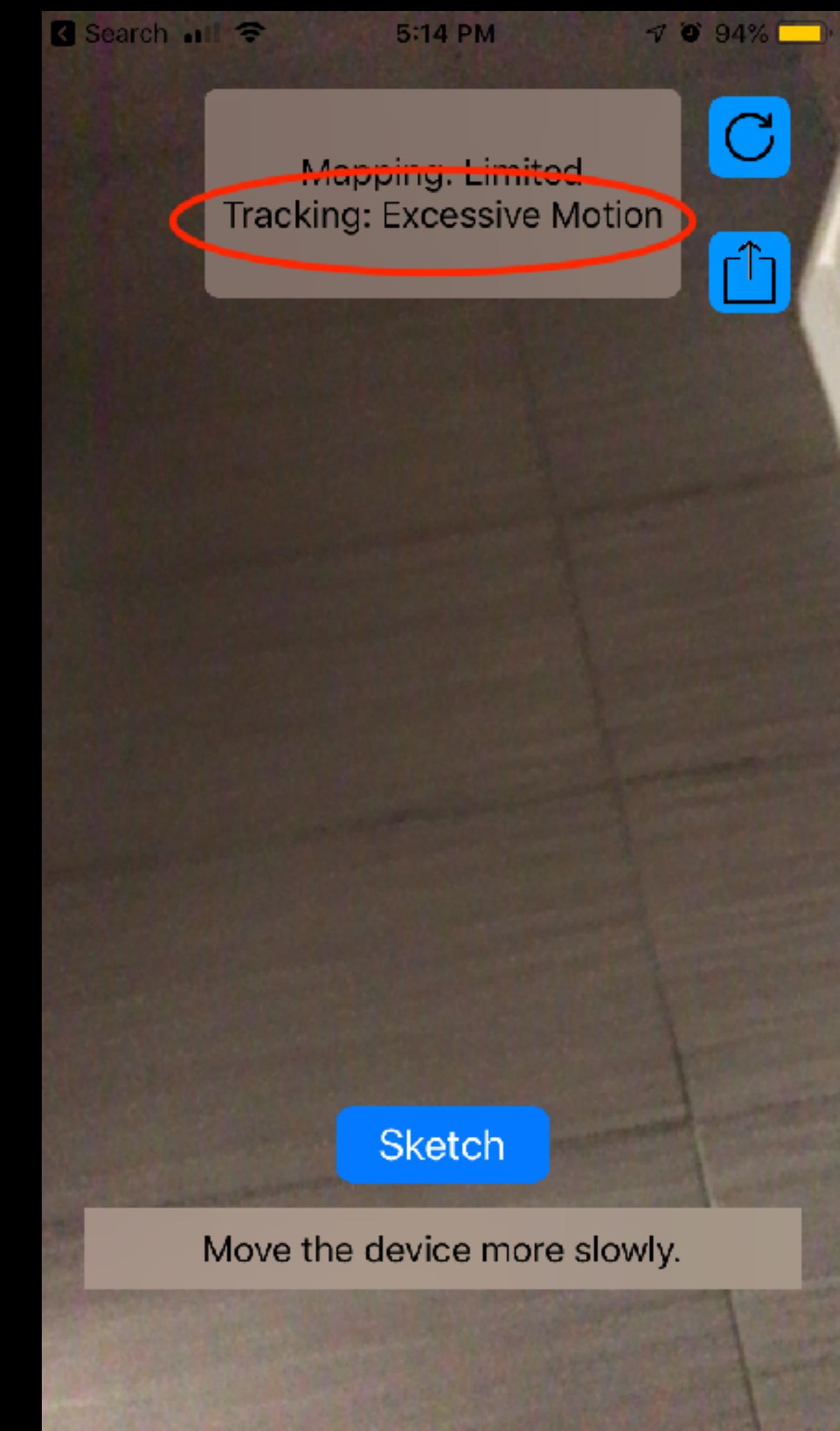


TRACKING QUALITY



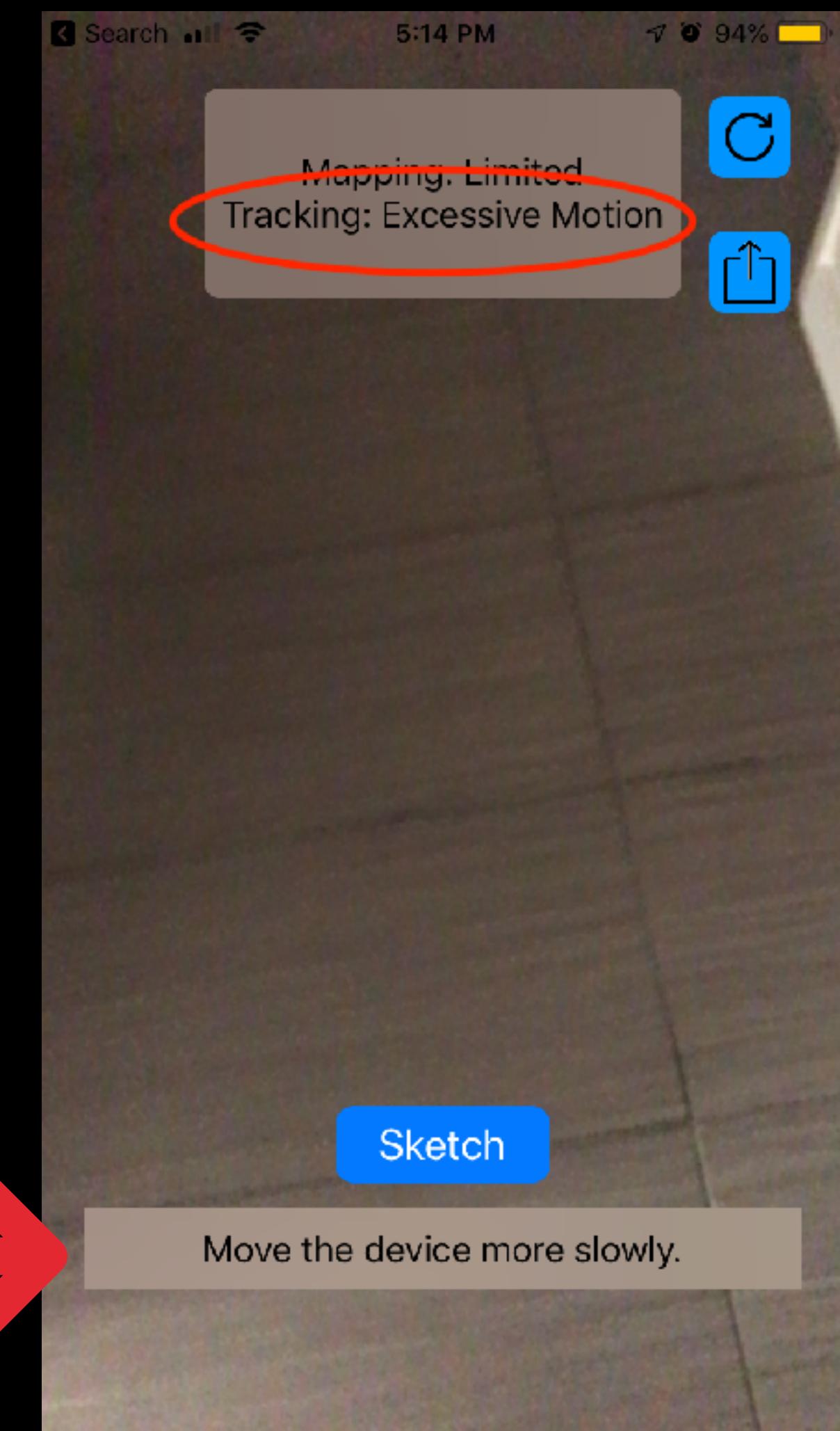
TRACKING QUALITY

```
extension ARCamera.TrackingState {  
    var localizedFeedback: String {  
        switch self {  
        case .normal:  
            return "All good!"  
        case .notAvailable:  
            return "Tracking unavailable."  
        case .limited(.excessiveMotion):  
            return "Move the device more slowly."  
        case .limited(.insufficientFeatures):  
            return "Point the device at an area with visible surface detail."  
        case .limited(.relocalizing):  
            return "Resuming session – move to where you were when the session  
        case .limited(.initializing):  
            return "Initializing AR session."  
        }  
    }  
}
```



TRACKING QUALITY

```
extension ARCamera.TrackingState {  
    var localizedFeedback: String {  
        switch self {  
        case .normal:  
            return "All good!"  
        case .notAvailable:  
            return "Tracking unavailable."  
        case .limited(.excessiveMotion):  
            return "Move the device more slowly."  
        case .limited(.insufficientFeatures):  
            return "Point the device at an area with visible surface detail."  
        case .limited(.relocalizing):  
            return "Resuming session – move to where you were when the session  
        case .limited(.initializing):  
            return "Initializing AR session."  
        }  
    }  
}
```



PERSISTING VIRTUAL CONTENT

PERSISTING VIRTUAL CONTENT

- ▶ ARWorldMap - serializable

PERSISTING VIRTUAL CONTENT

- ▶ ARWorldMap - serializable
- ▶ Use NSSecureCoding protocol

PERSISTING VIRTUAL CONTENT

- ▶ ARWorldMap - serializable
- ▶ Use NSSecureCoding protocol
- ▶ ARAncors store object state and position

CAPTURING THE ARWORLDMAP

CAPTURING THE ARWORLDMAP

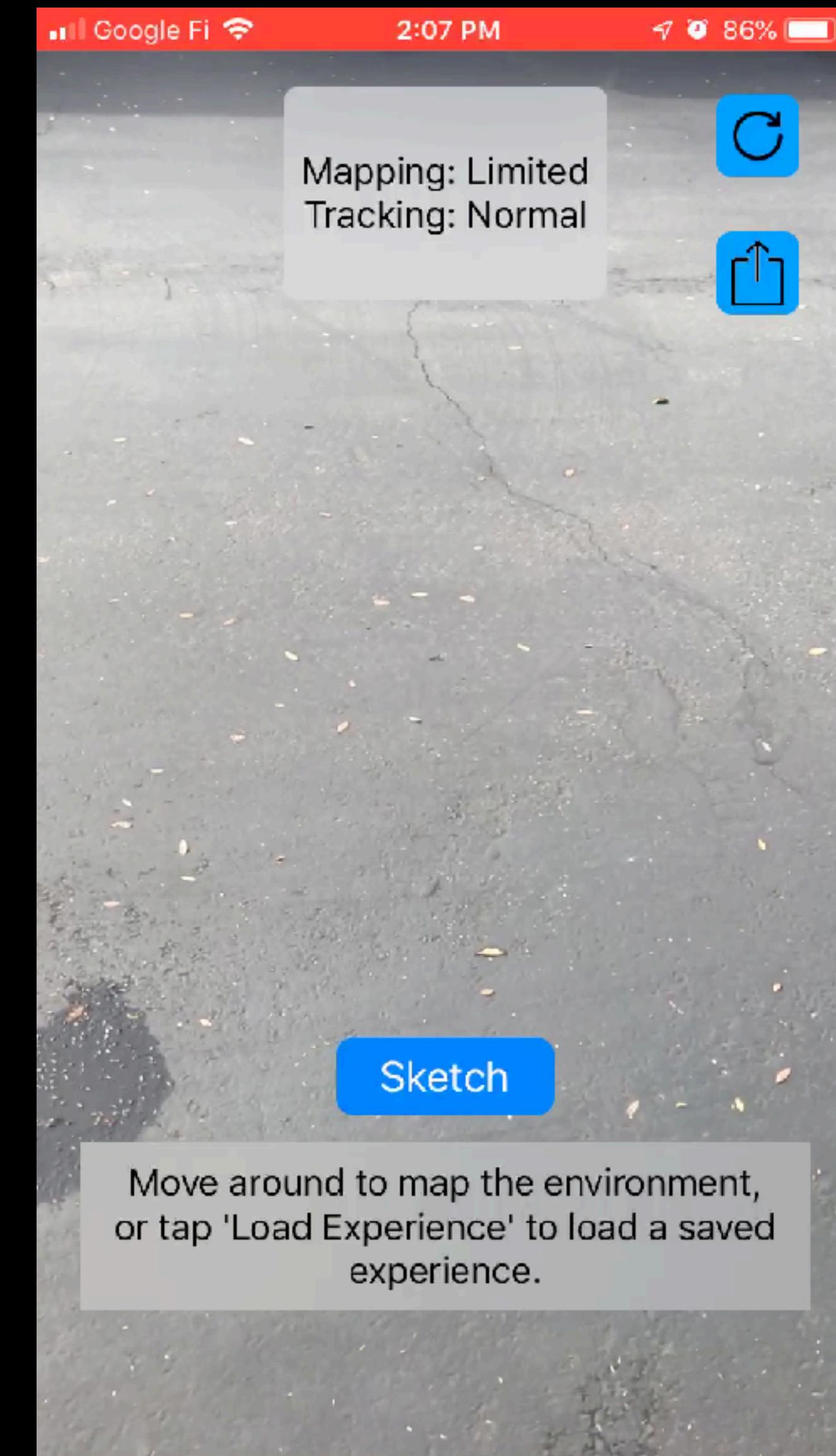
```
extension ARFrame.WorldMappingStatus: CustomStringConvertible {  
    public var description: String {  
        switch self {  
            case .notAvailable:  
                return "Not Available"  
            case .limited:  
                return "Limited"  
            case .extending:  
                return "Extending"  
            case .mapped:  
                return "Mapped"  
        }  
    }  
}
```

CAPTURING THE ARWORLDMAP

▶ Use WorldMappingStatus

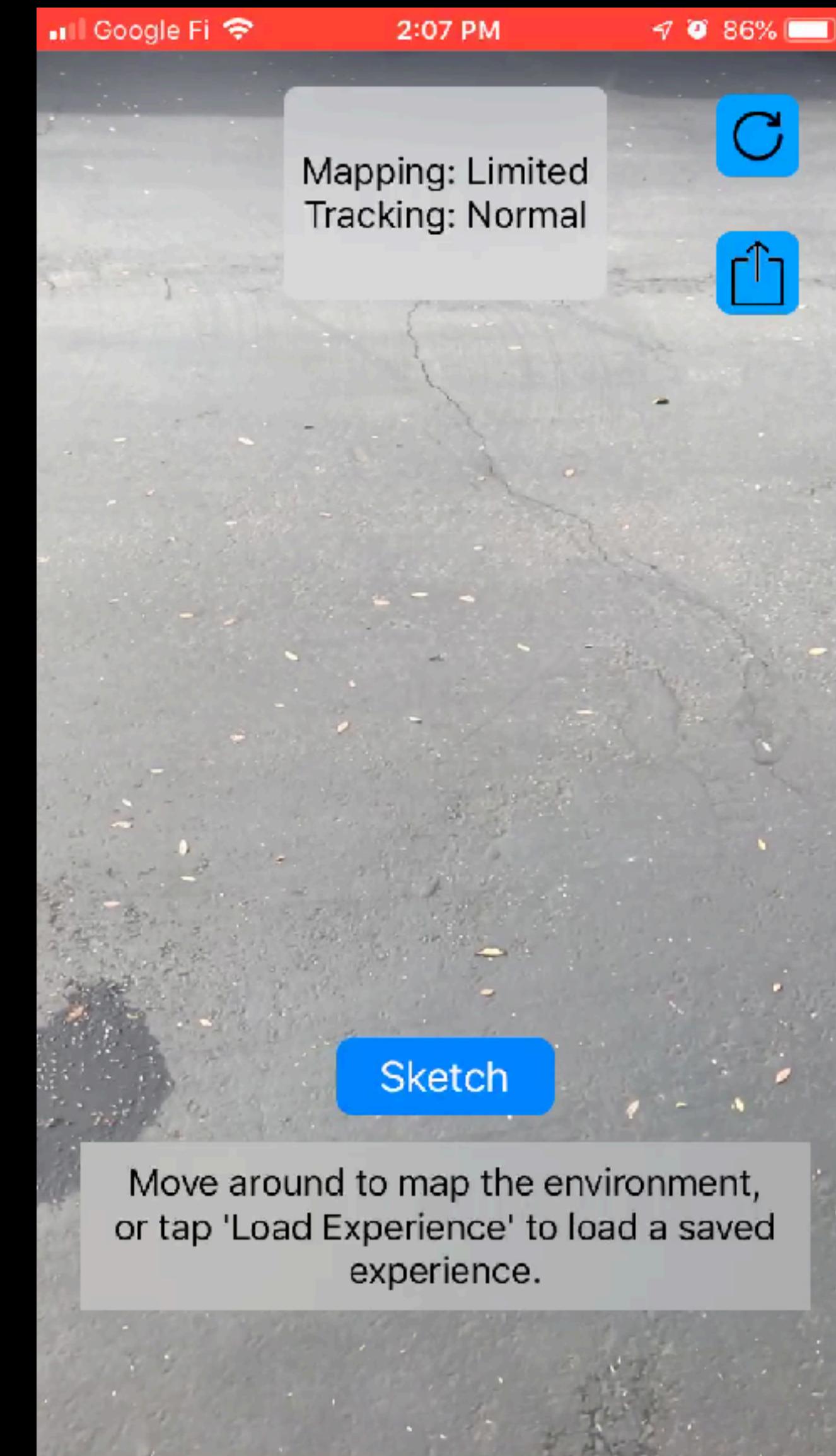
```
extension ARFrame.WorldMappingStatus: CustomStringConvertible {  
    public var description: String {  
        switch self {  
        case .notAvailable:  
            return "Not Available"  
        case .limited:  
            return "Limited"  
        case .extending:  
            return "Extending"  
        case .mapped:  
            return "Mapped"  
        }  
    }  
}
```

WORLD MAPPING



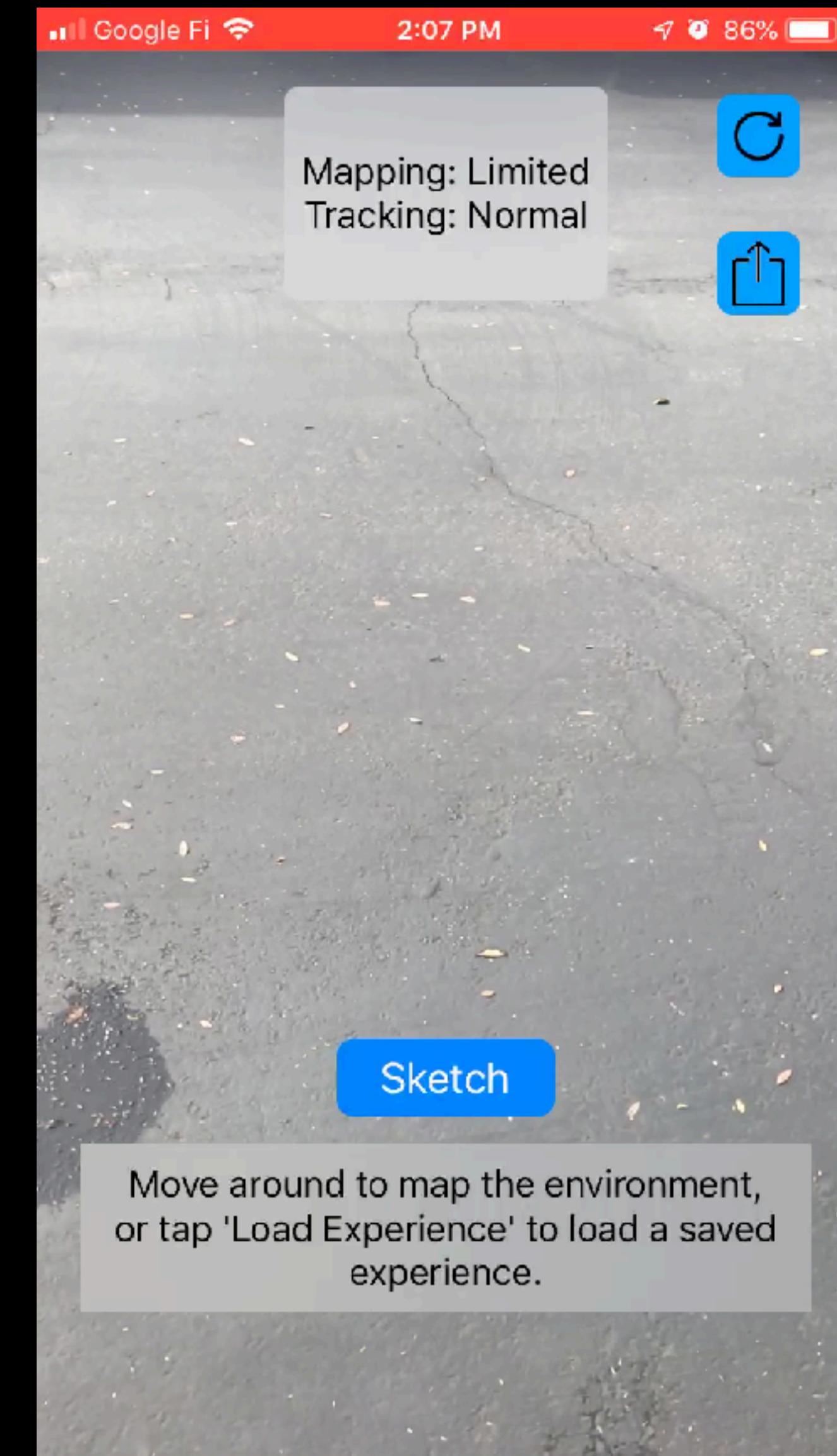
WORLD MAPPING

- ▶ Don't display UI until ready



WORLD MAPPING

- ▶ Don't display UI until ready



DO'S  **AND DON'TS** 

USE THE ENTIRE SCREEN



COMFORTABLE AND SAFE TECHNOLOGY

KISS - Keep it short and safe

ENABLE ONE-HANDED APP USE



Measure

Level

ENABLE ONE-HANDED APP USE



Footprint

Measure

Level

INITIALIZATION TAKES TIME



Do 

Try moving around, turning on more lights and making sure your environment has sufficiently textured surfaces

Move your phone slower

Tap a location to place the [name of object to be placed]

Don't 

Surface detection is taking too long

Excessive motion detected

Tap a plane to anchor an object

GHOST EFFECT



GHOST EFFECT







DON'T ASSUME IDEAL ENVIRONMENT

MAPPING FROM A DISTANCE





SCENEKIT DELEGATES

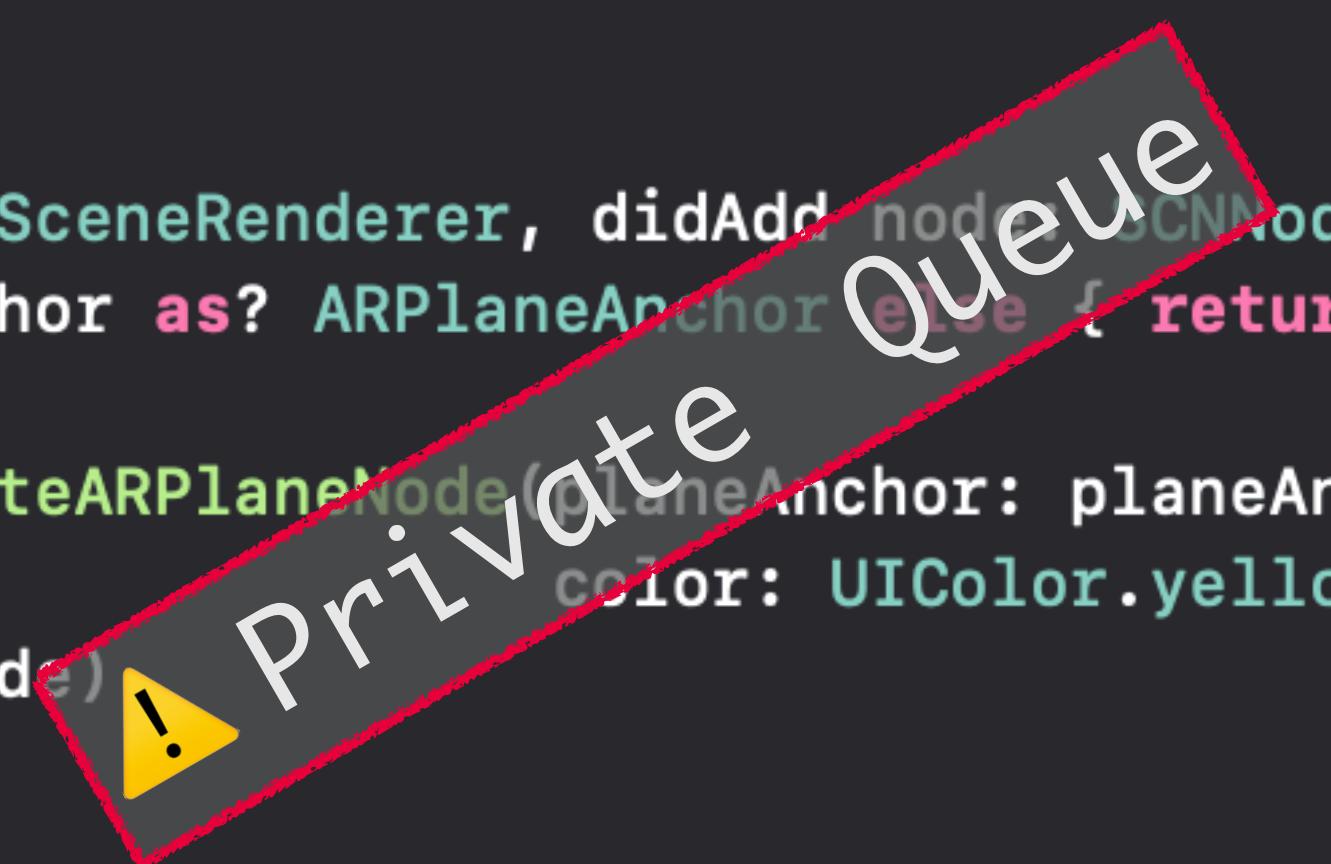
```
extension ViewController : ARSCNViewDelegate {
    func renderer(_ renderer: SCNSceneRenderer, updateAtTime time: TimeInterval) {
        DispatchQueue.main.async {
            selfStatusLabel.text = self.trackingStatus
        }
    }

    func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNNode, for anchor: ARAnchor) {
        guard let planeAnchor = anchor as? ARPlaneAnchor else { return }
        DispatchQueue.main.async {
            let planeNode = self.createARPlaneNode(planeAnchor: planeAnchor,
                                                    color: UIColor.yellow.withAlphaComponent(0.5))
            node.addChildNode(planeNode)
        }
    }

    func renderer(_ renderer: SCNSceneRenderer, didUpdate node: SCNNNode, for anchor: ARAnchor) {
        guard let planeAnchor = anchor as? ARPlaneAnchor else { return }
        DispatchQueue.main.async {
            self.updateARPlaneNode(planeNode: node.childNodes[0], planeAnchor: planeAnchor)
        }
    }
}
```

SCENEKIT DELEGATES

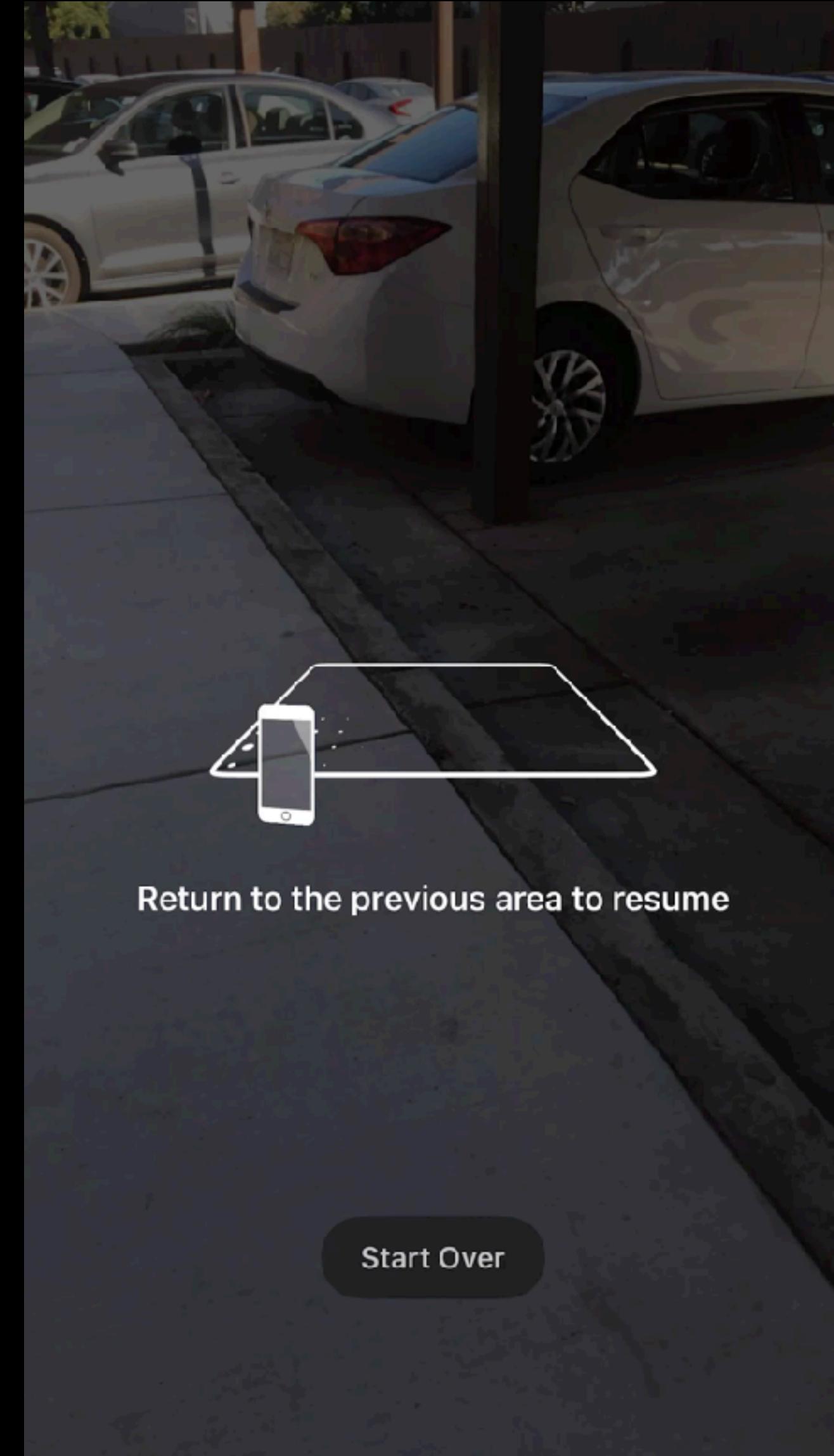
```
extension ViewController : ARSCNViewDelegate {  
    func renderer(_ renderer: SCNSceneRenderer, updateAtTime time: TimeInterval) {  
        DispatchQueue.main.async {  
            selfStatusLabel.text = self.trackingStatus  
        }  
    }  
  
    func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNode, for anchor: ARAnchor) {  
        guard let planeAnchor = anchor as? ARPlaneAnchor else { return }  
        DispatchQueue.main.async {  
            let planeNode = self.createARPlaneNode(planeAnchor: planeAnchor,  
                                                    color: UIColor.yellow.withAlphaComponent(0.5))  
            node.addChildNode(planeNode)  
        }  
    }  
  
    func renderer(_ renderer: SCNSceneRenderer, didUpdate node: SCNNode, for anchor: ARAnchor) {  
        guard let planeAnchor = anchor as? ARPlaneAnchor else { return }  
        DispatchQueue.main.async {  
            self.updateARPlaneNode(planeNode: node.childNodes[0], planeAnchor: planeAnchor)  
        }  
    }  
}
```



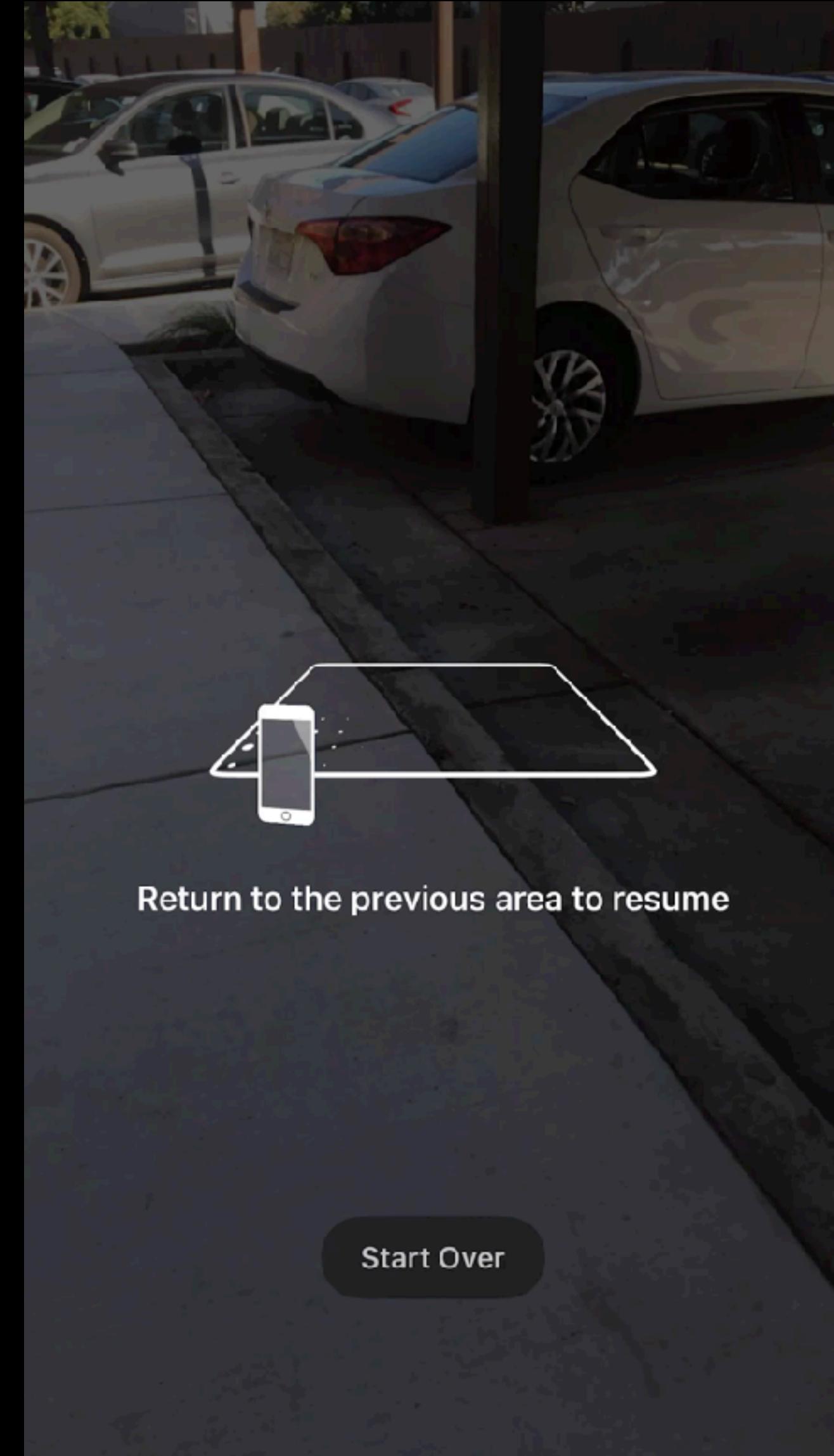
BATTERY DRAIN!



SET YOUR USERS UP FOR SUCCESS



SET YOUR USERS UP FOR SUCCESS



THANK YOU!



@NAMRATACODES



NAMRATABANDEKAR

<https://tinyurl.com/swiftfest-arkit>