

虚拟与现实的融合

- AR 客户端技术实践

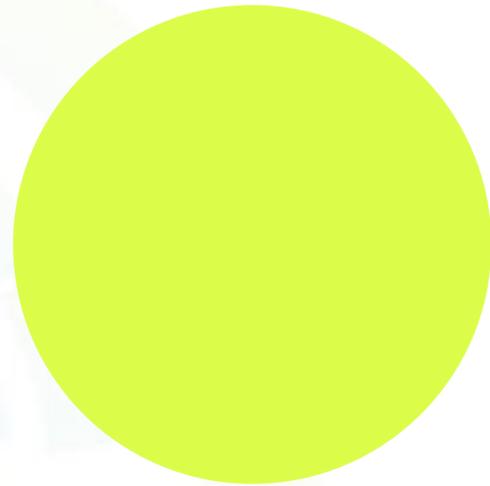
自我介绍



- 姓名：吴天求
- 部门：大住宿-android
- 简要介绍：2014加入去哪儿，主要负责酒店android客户端开发，android技术研发，性能提升和流程优化，项目管理等

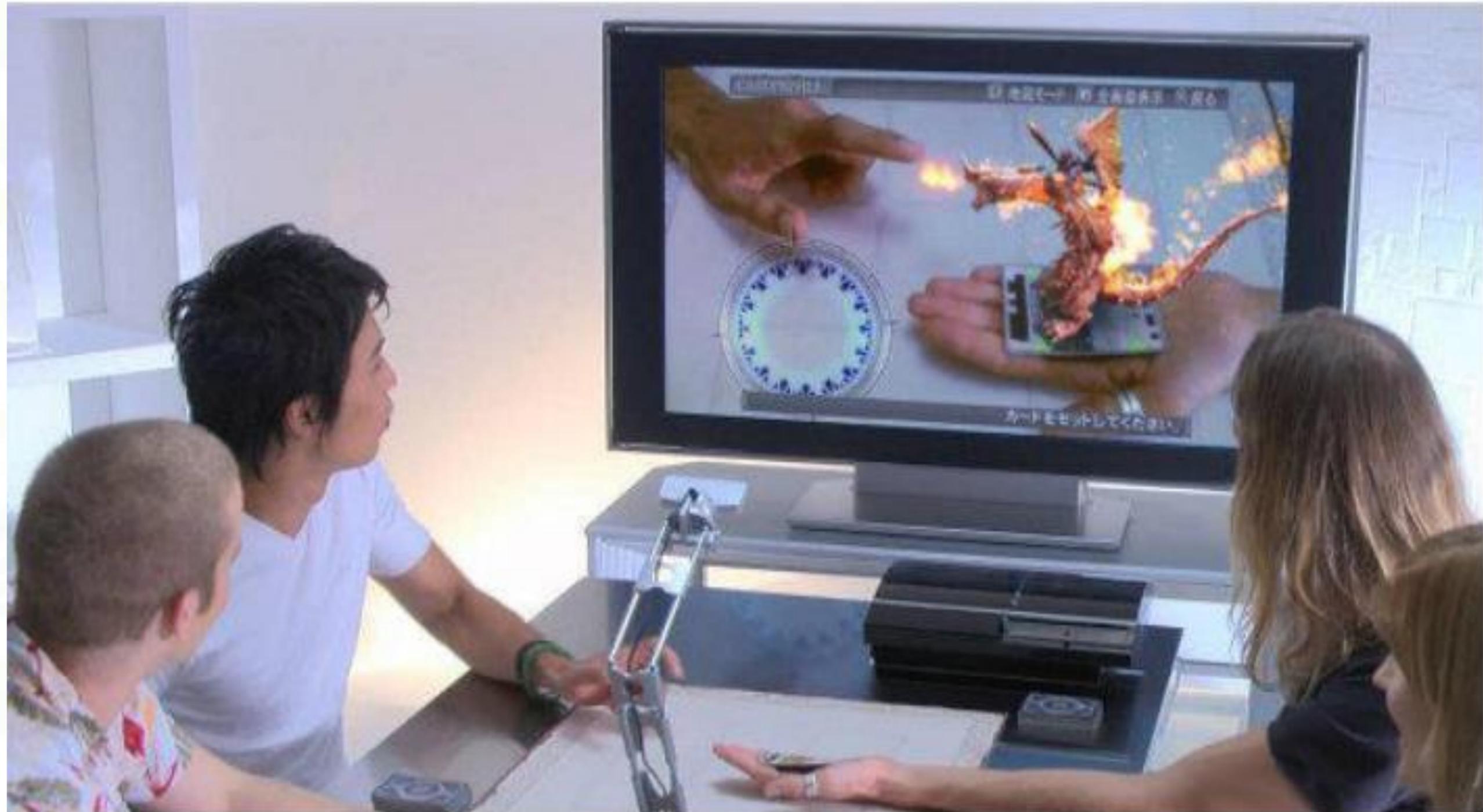
目录

- 1 ▶ AR概念简介
- 2 ▶ 市面上AR技术应用简介
- 3 ▶ 去哪儿AR初体验
- 4 ▶ 技术原理解析
- 5 ▶ 核心算法



一、AR概念简介

AR概念简介



AR概念简介



AR概念简介

它是一种将**真实世界信息**和**虚拟世界信息**“无缝”集成的新技术

AR概念简介



微软HoloLens

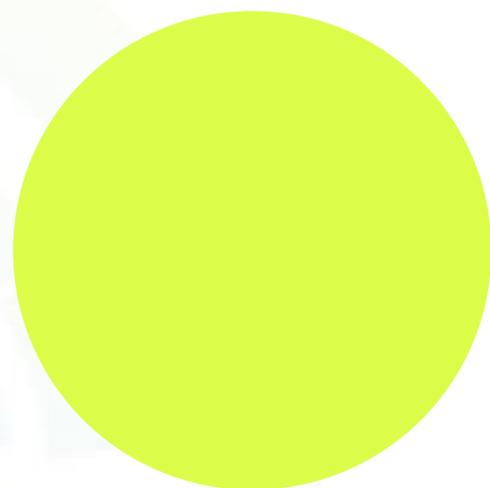


Meta



以色列Lumus

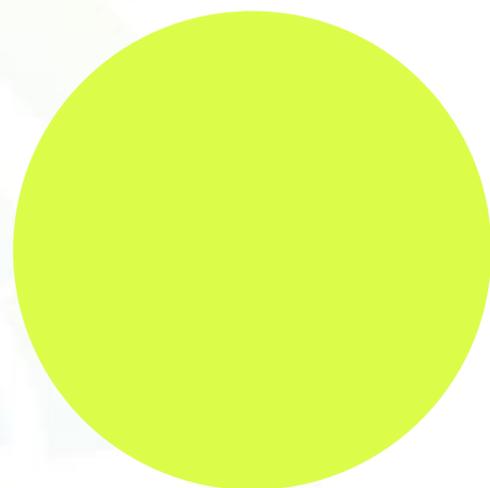
等等...



二、市面上AR技术应用简介

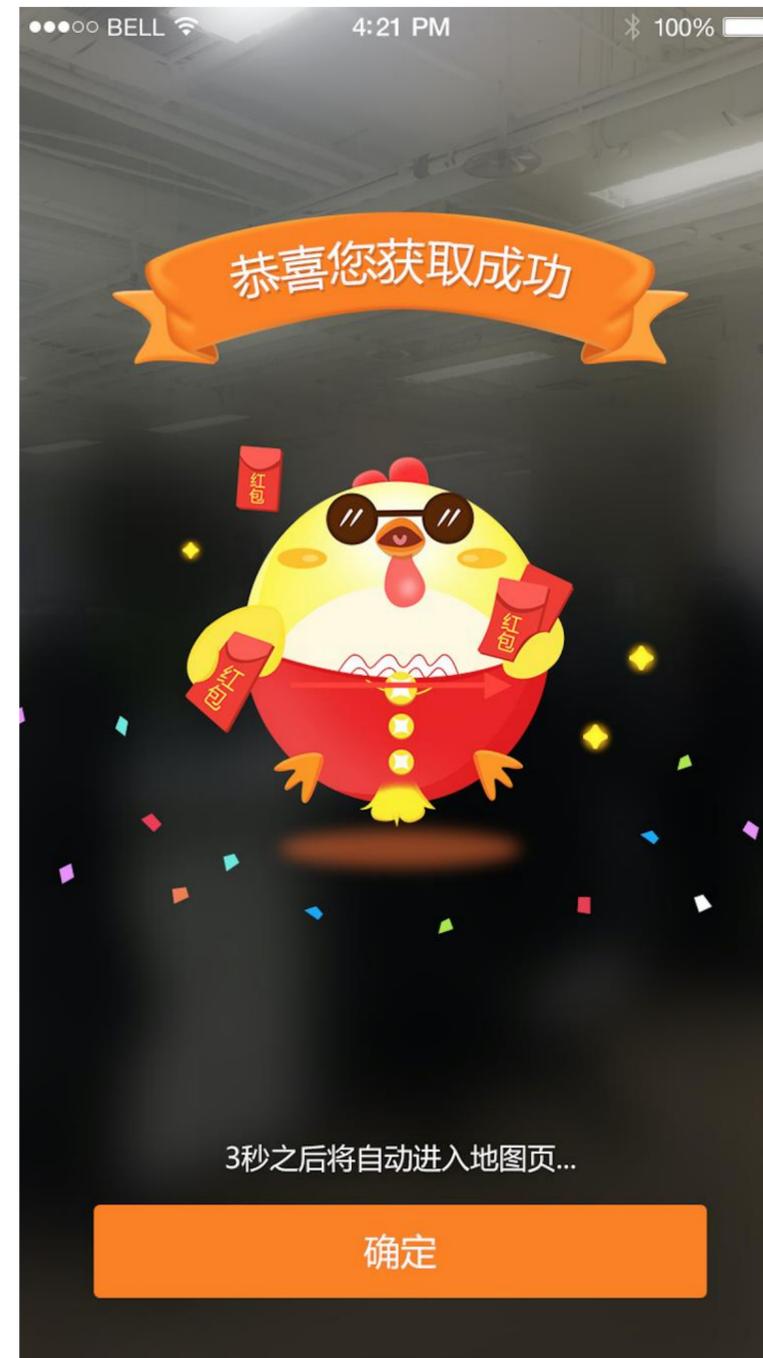
市面上AR技术应用简介



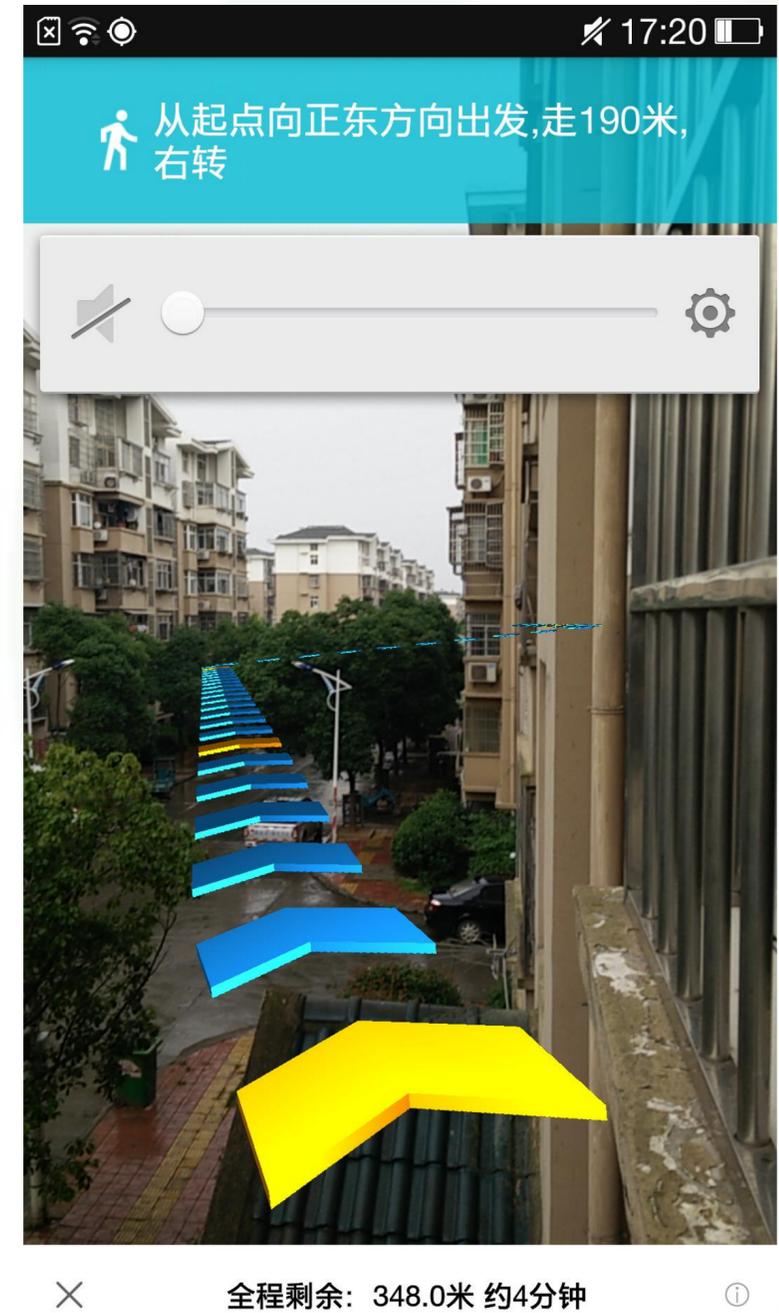
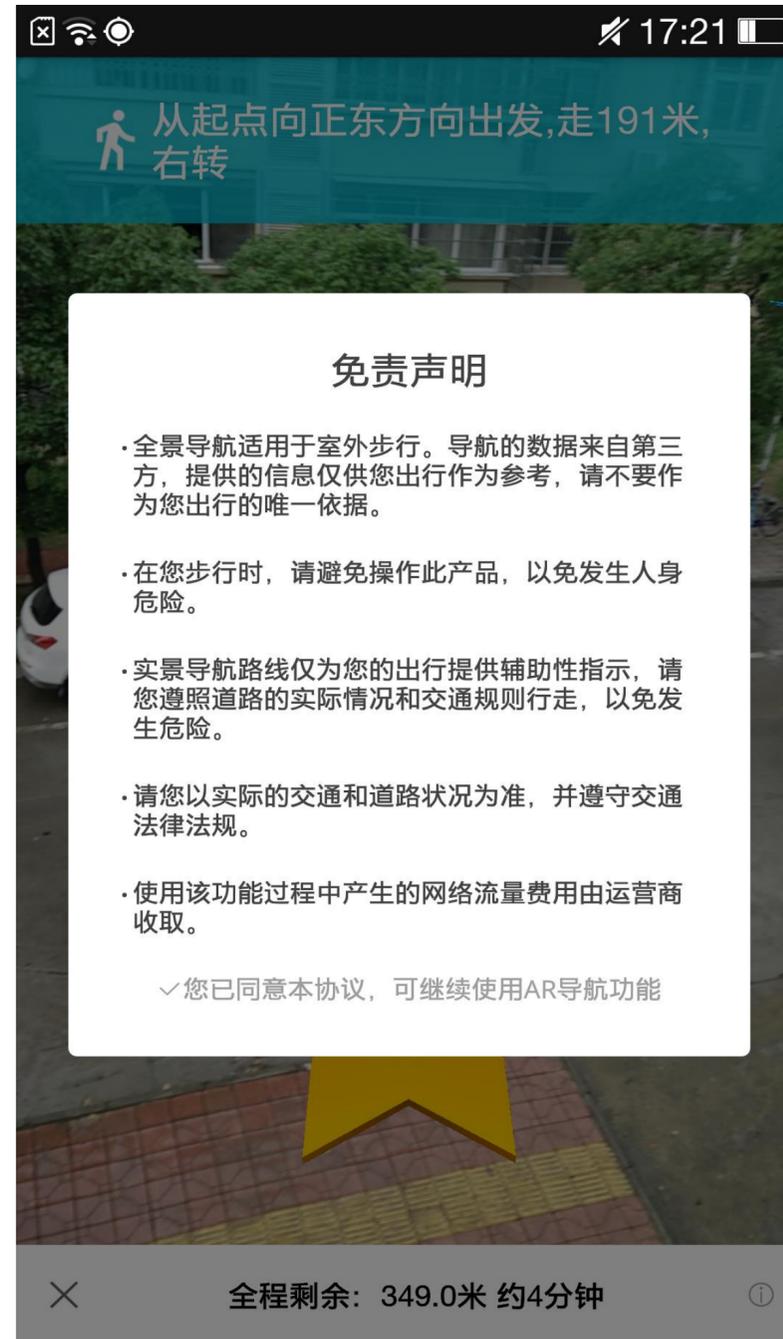


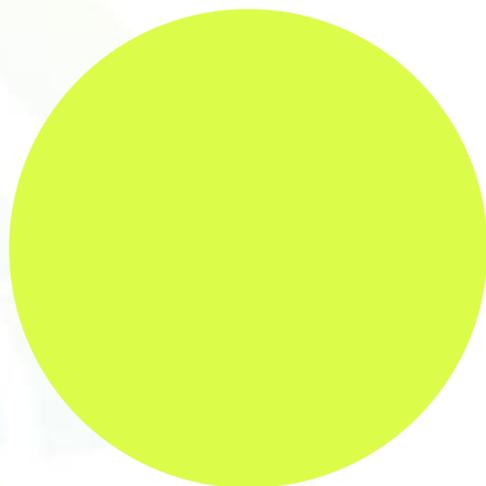
三、去哪儿AR初体验

去哪儿AR初体验-AR抓鸡红包



去哪儿AR初体验-AR实景路径导航

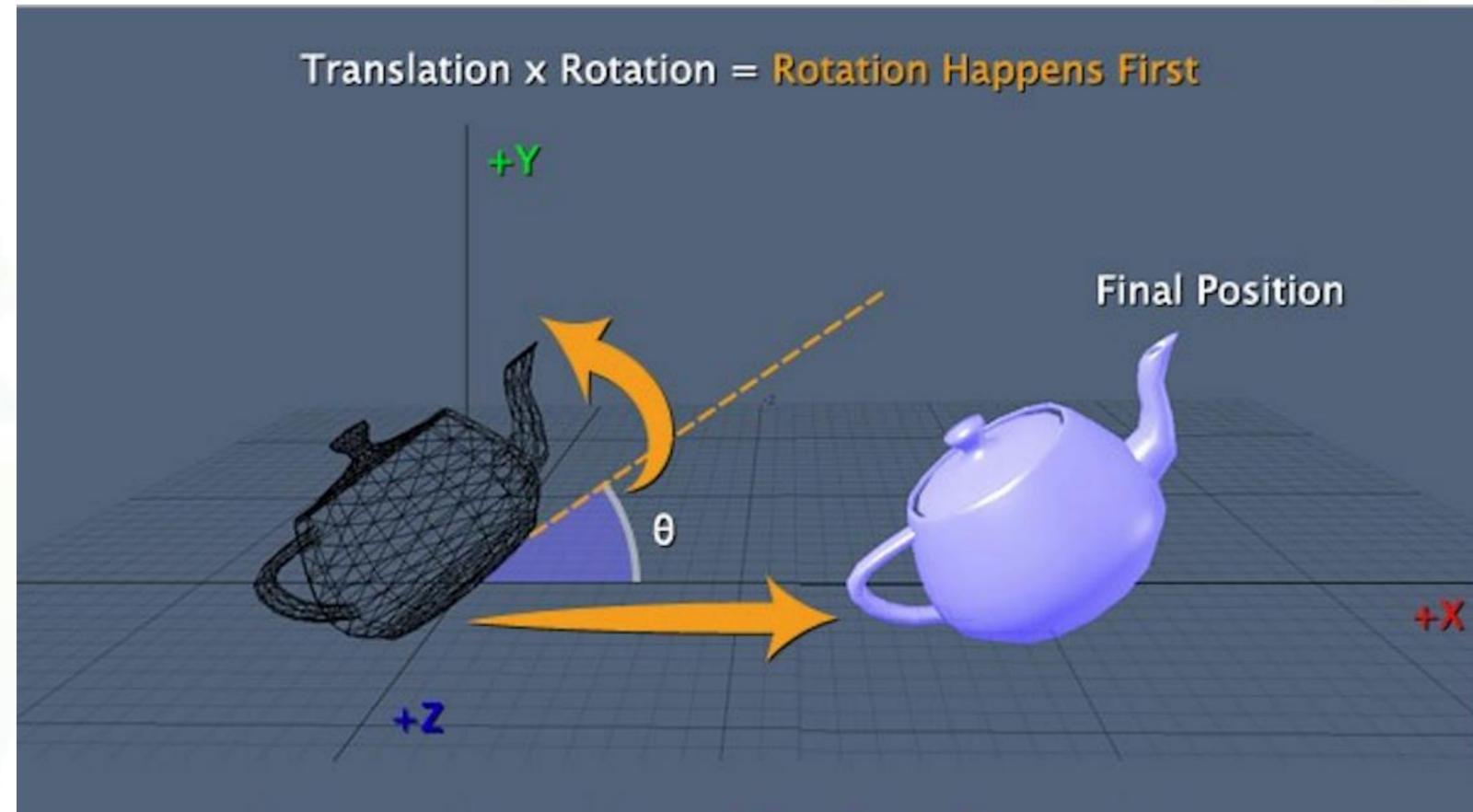




四、技术原理解析

技术原理解析

- 3D数学基础
- OpenGL ES 2.0
- GLSL



矩阵

1. Matrix 可用大小为 16 的一维数组或 4×4 的二维数组来表示

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

2. OpenGL 使用列主序矩阵，即列矩阵，因此我们总是倒过来算的（左乘矩阵，变换效果是按从右向左的顺序进行）

矩阵

$$\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} ax+by+cz \\ dx+ey+fz \end{bmatrix}$$

A B C

$$\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \times \begin{bmatrix} x & w \\ y & u \\ z & v \end{bmatrix} = \begin{bmatrix} ax+by+cz & aw+bu+cv \\ dx+ey+fz & dw+eu+fv \end{bmatrix}$$

A B C

矩阵的乘法

矩阵

$$\begin{matrix} - \\ + \end{matrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 2 & 3 & 4 & 1 \end{bmatrix} \times \begin{bmatrix} 2 & 5 \\ 6 & 7 \\ 1 & 8 \\ 1 & 1 \end{bmatrix} \begin{matrix} - \\ + \end{matrix}$$

- + - +

矩阵

平移矩阵

$$\begin{bmatrix} 1 & 0 & 0 & X \\ 0 & 1 & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

平移矩阵 \times 列矩阵(a, b, c, 1) = 列矩阵(a + x, b + y, c + z, 1)

矩阵

缩放矩阵

SX	0	0	0
0	SY	0	0
0	0	SZ	0
0	0	0	1

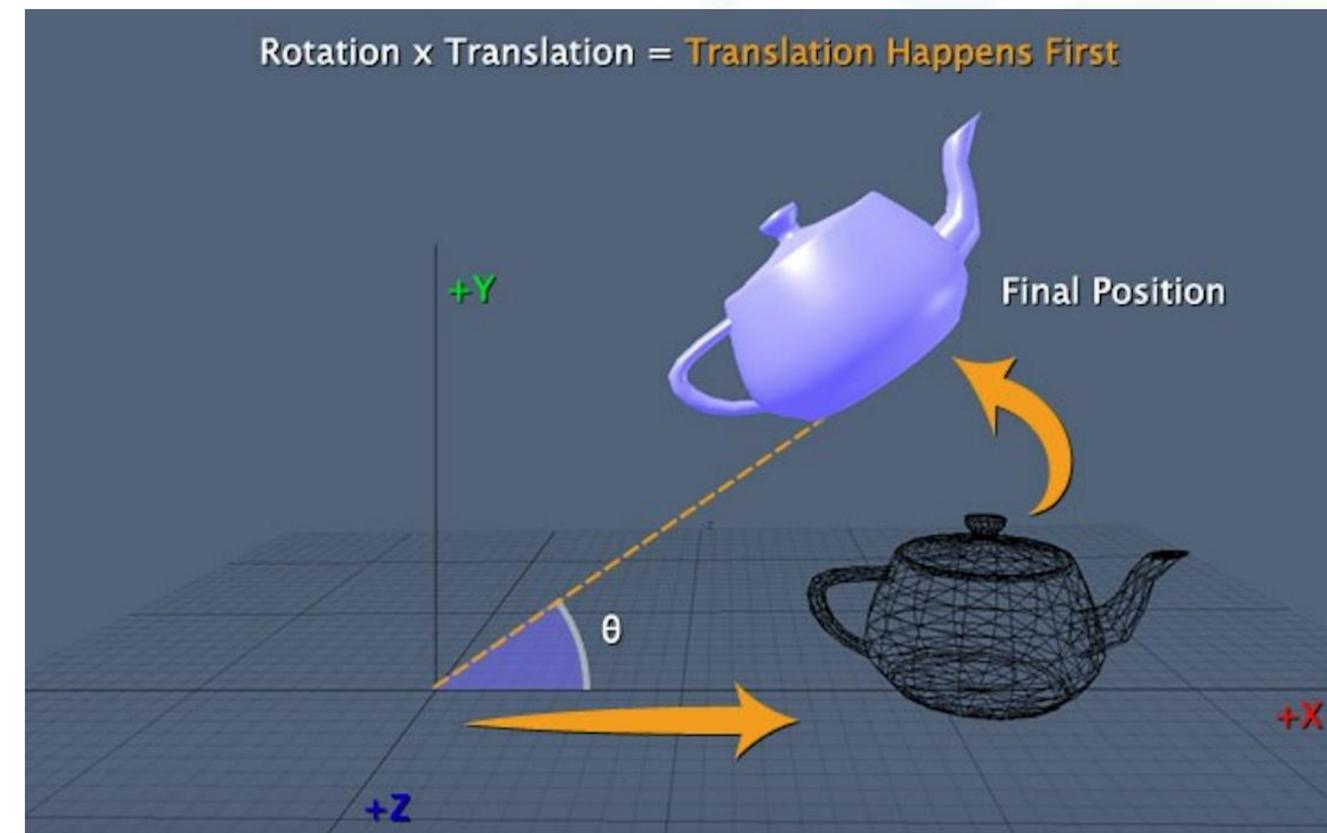
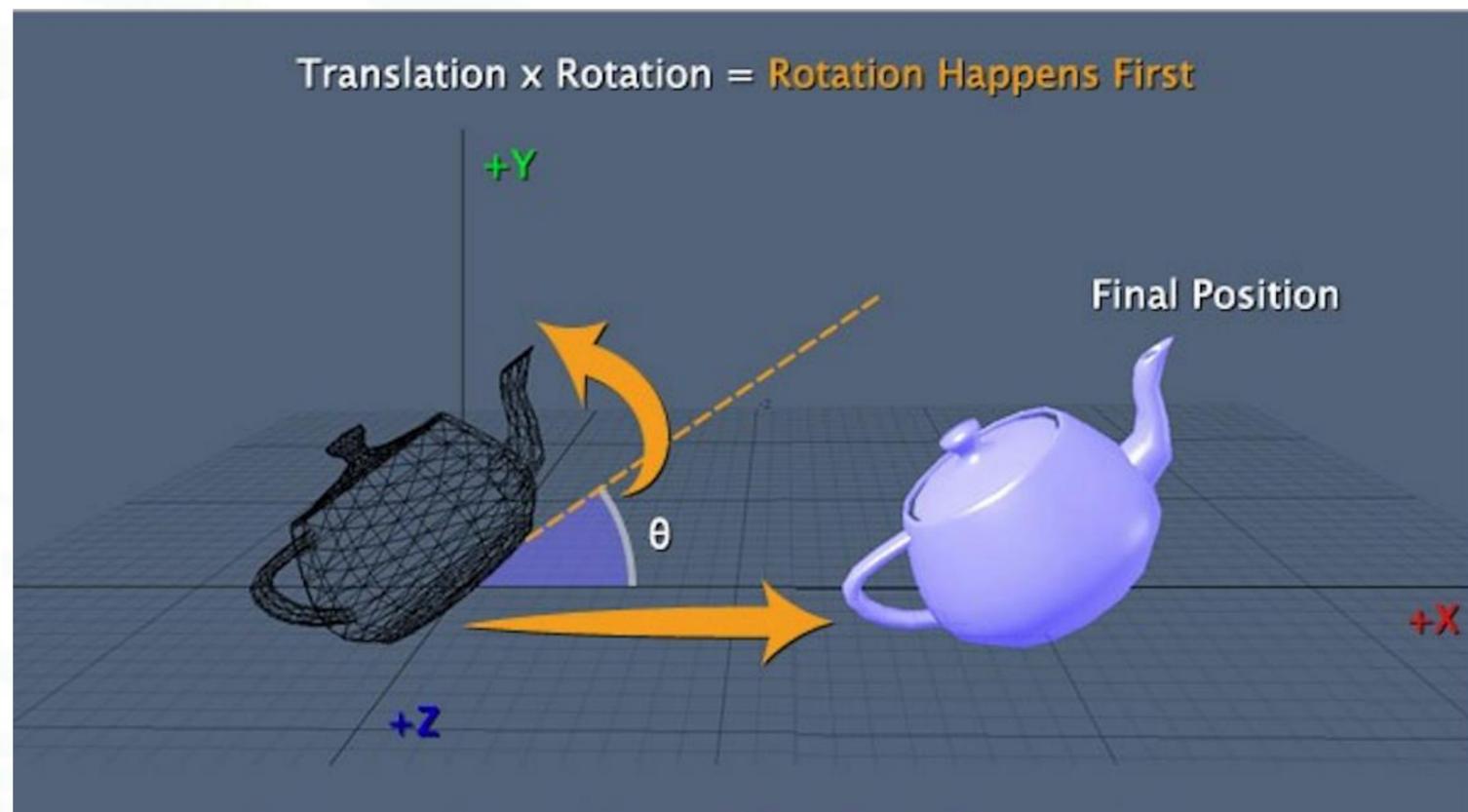
缩放矩阵 \times 列矩阵(a, b, c, 1) = 列矩阵(a \times sx, b \times sy, c \times sz, 1)

矩阵

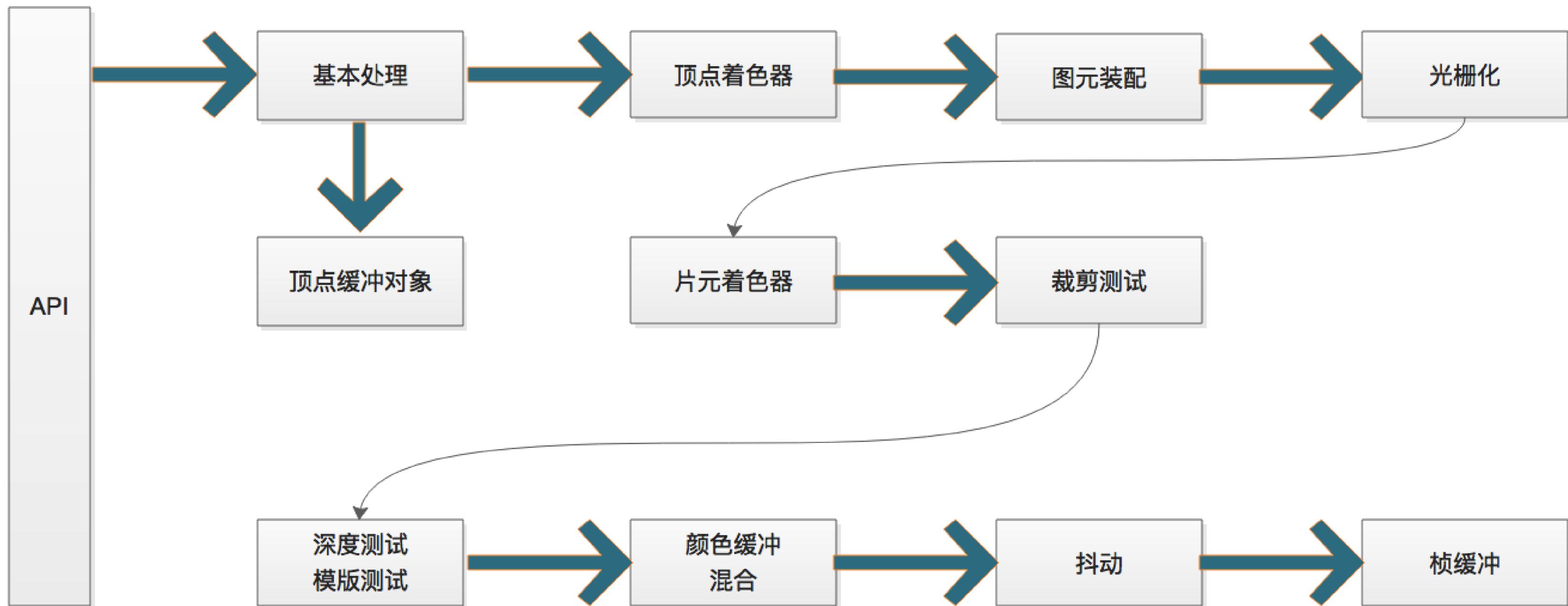
绕 X 轴旋转的旋转矩阵

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) & 0 \\ 0 & -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

旋转矩阵 \times 列矩阵 $(a, b, c, 1) =$ 列矩阵 $(a, b \times \cos(\theta) - c \times \sin(\theta), b \times -\sin(\theta) + c \times \cos(\theta), 1)$



技术原理解析



```
private class SceneRenderer implements GLSurfaceView.Renderer {
```

```
    public void onDrawFrame(GL10 gl) {
```

```
        // 清除深度缓冲与颜色缓冲
```

```
        // 绘制
```

```
    }
```

```
    public void onSurfaceChanged(GL10 gl, int width, int height) {
```

```
        // 设置视窗大小及位置
```

```
        // 设置投影矩阵
```

```
        // 设置摄像机的位置
```

```
    }
```

```
    public void onSurfaceCreated(GL10 gl, EGLConfig config) {
```

```
        // 创建三角形对象
```

```
        // 打开深度检测
```

```
        // 初始化模型 / 纹理
```

```
        // 关闭背面剪裁
```

```
    }
```

```
}
```

技术原理解析

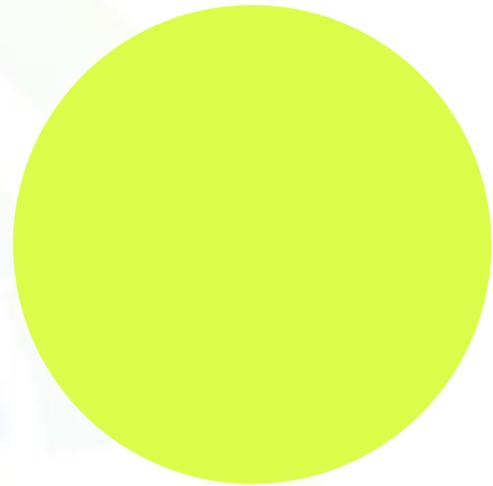
顶点着色器

```
1 uniform mat4 uMVPMatrix; //总变换矩阵
2 attribute vec3 aPosition; //顶点位置
3 attribute vec2 aTexCoord; //顶点纹理坐标
4 varying vec2 vTextureCoord; //用于传递给片元着色器的变量
5 void main()
6 {
7     gl_Position = uMVPMatrix * vec4(aPosition,1); //根据总变换矩阵计算此次绘制此顶点位置
8     vTextureCoord = aTexCoord; //将接收的纹理坐标传递给片元着色器
9 }
```

技术原理解析

片元着色器

```
1 precision mediump float;  
2 varying vec2 vTextureCoord; //接收从顶点着色器过来的参数  
3 uniform sampler2D sTexture; //纹理内容数据  
4 void main()  
5 {  
6     //给此片元从纹理中采样出颜色值  
7     gl_FragColor = texture2D(sTexture, vTextureCoord);  
8 }
```

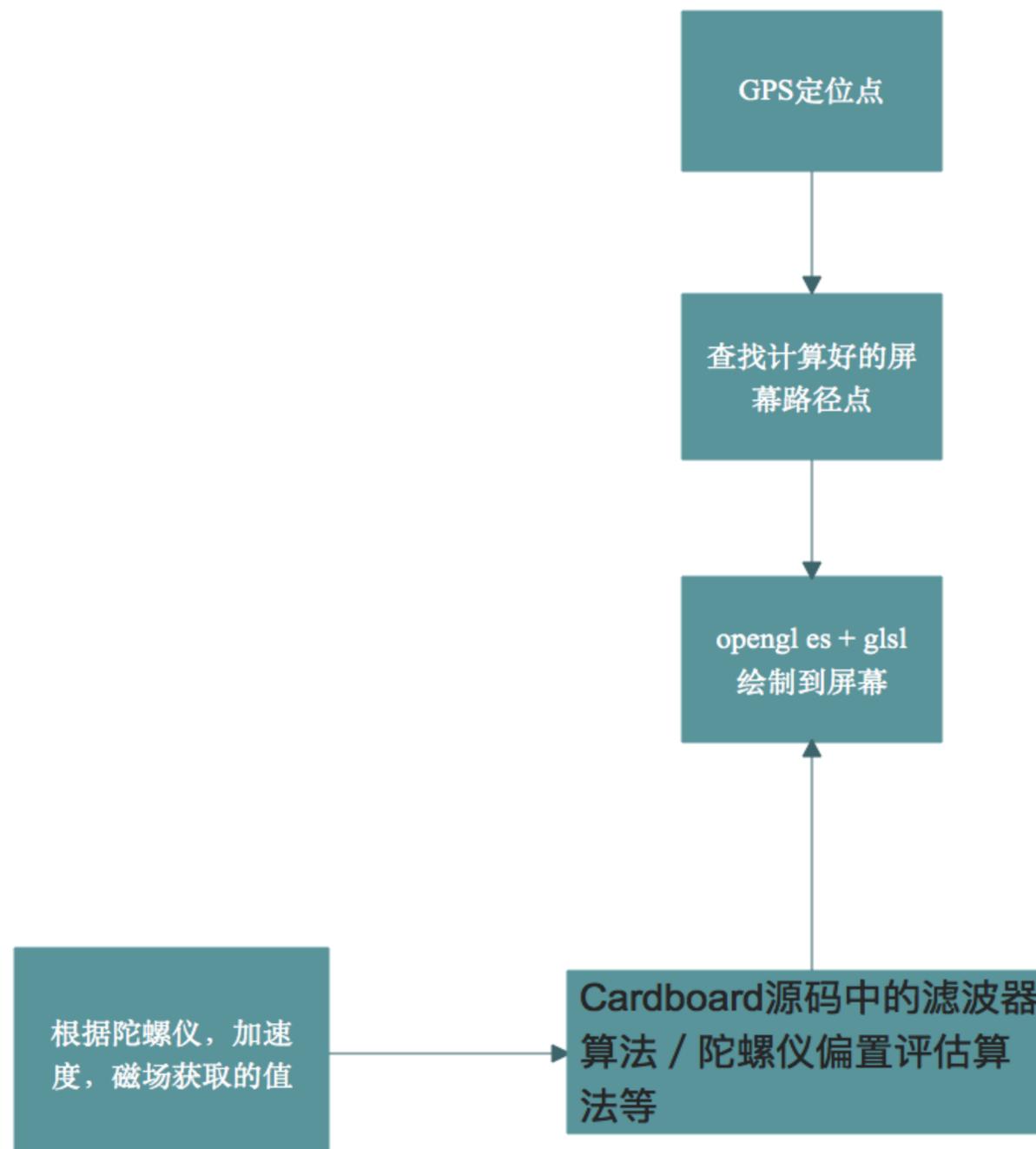
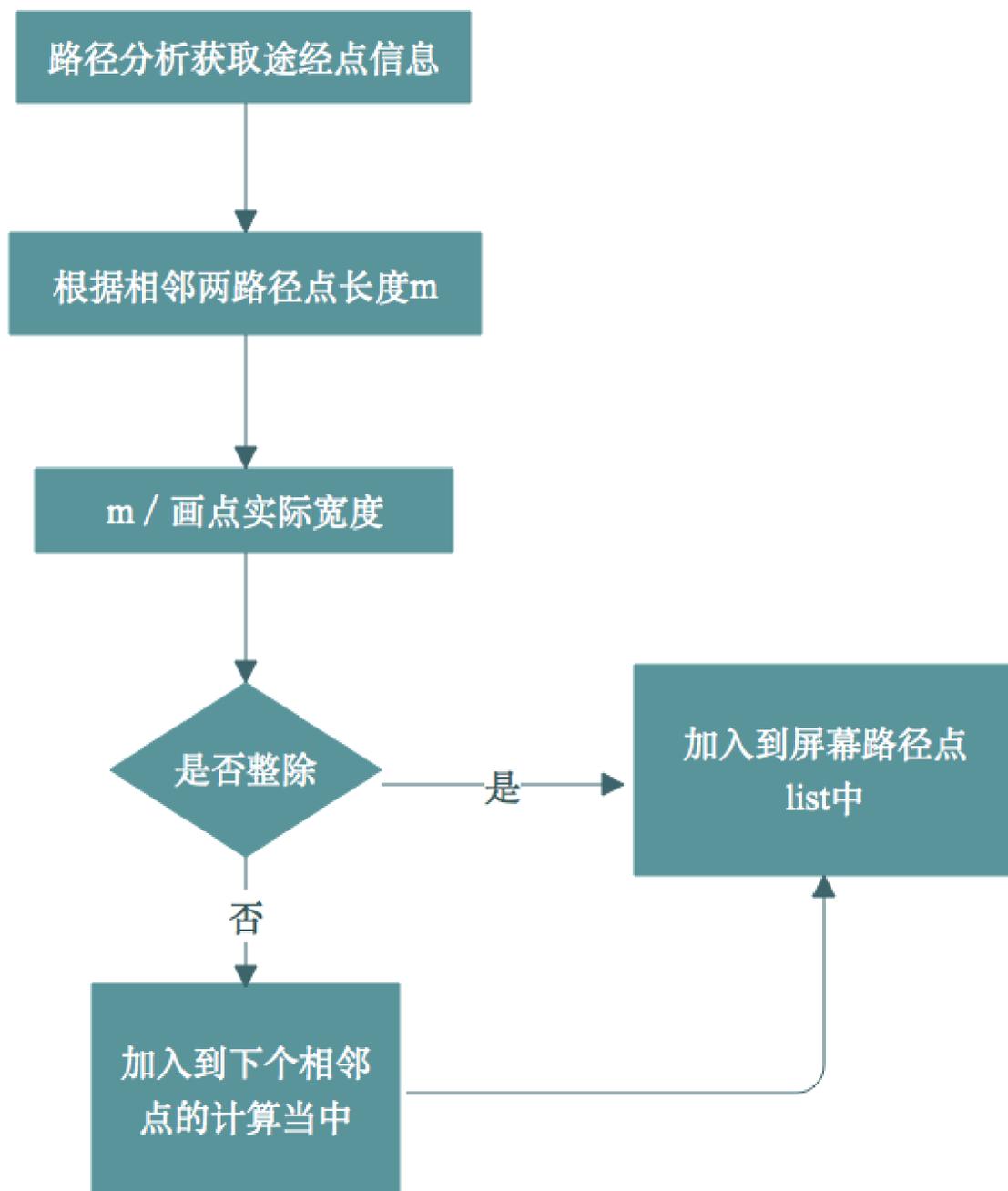


五、核心算法

核心算法

1. 陀螺仪偏置+加速度滤波+传感器融合算法
2. 路径点补点算法

核心算法



总结回顾

- 酒店现阶段AR使用的是opengl es2.0+glsl
- 使用酒店的AR实景路径导航：scheme即可
- 欢迎更多的同学参与

ARKit 早期观察与实践

什么是ARKit



ARKit

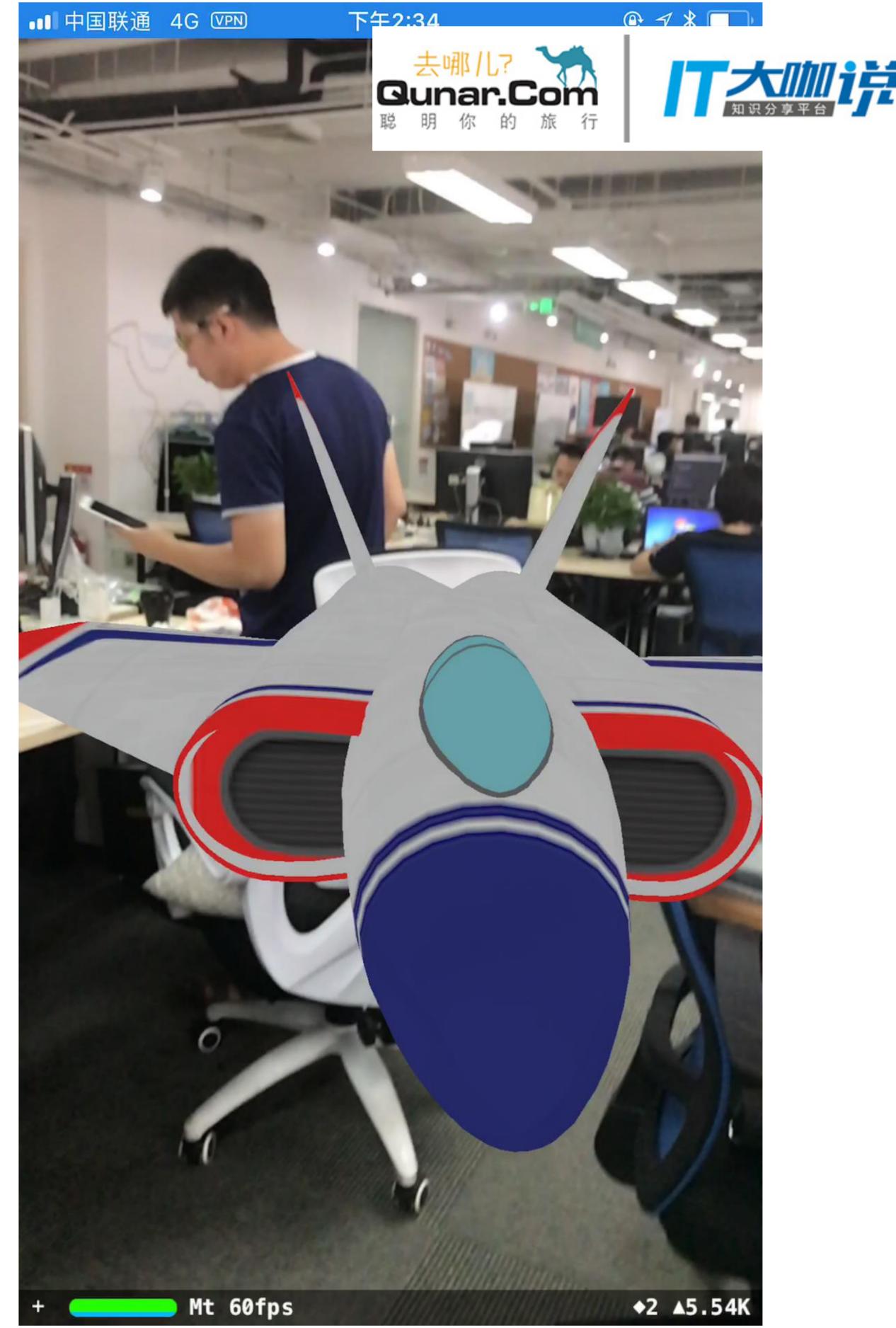
+

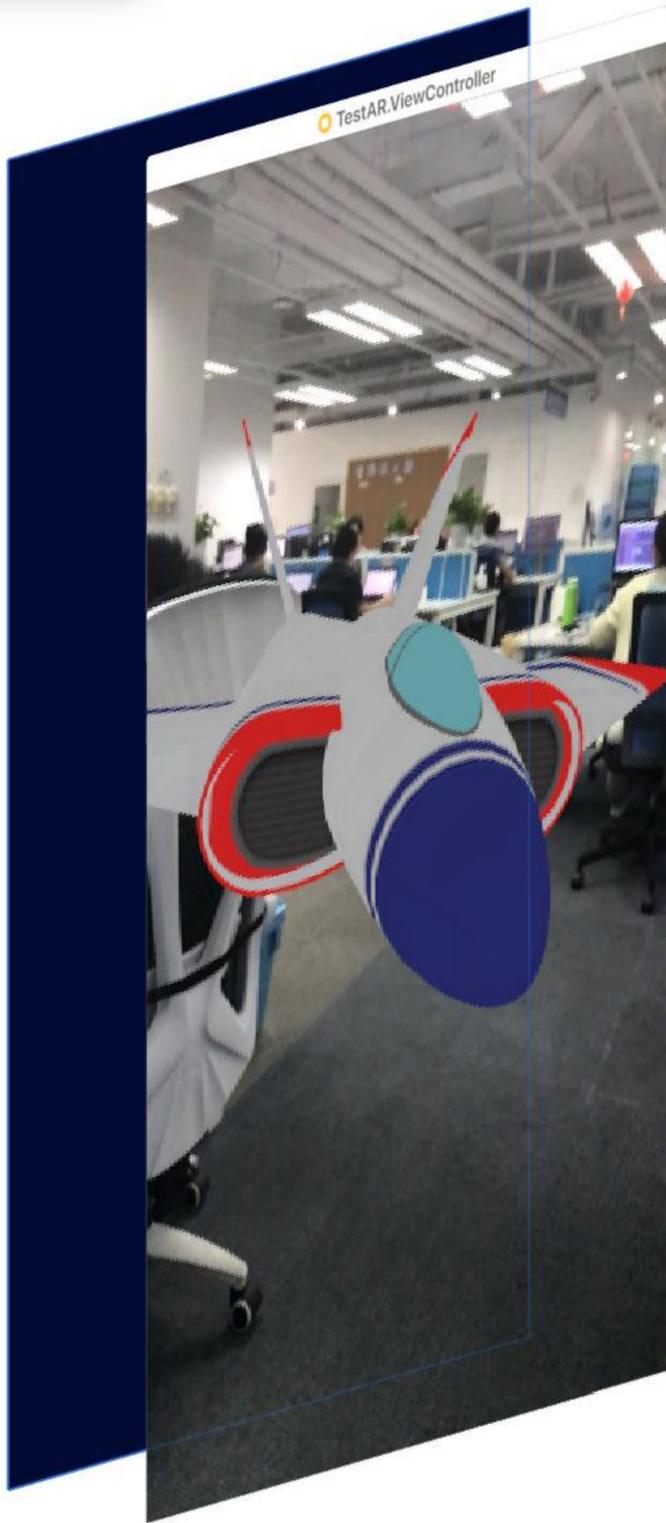


SceneKit/SpriteKit

=

通过ARKit的优秀SLAM能力，通过SceneKit/SprintKit，将2D屏幕效果还原到3D摄像头拍摄内容中





“为何ARKit效果这么好?”

S L A M

S Simultaneous

L Localization

A And

M Mapping

S Simultaneous

L Localization 定位

A And

M Mapping 制图

S L A M

希望机器人从未知环境的未知地点出发，在运动过程中通过重复**观测**到的地图特征（比如，墙角，柱子等）**定位自身位置和姿态**，再根据自身位置**增量式的构建地图**，从而达到**同时定位和地图构建**的目的

S L A M

观测，构建地图

单/双目摄像头，深度摄像头，雷达

自身位置和姿态

IMU，雷达

Apple的优势

1. 通过IMU来确定设备位置的误差大大低于其他方案
 2. 拥有快速的平面检测能力
 3. 结合游戏引擎能够满帧率运行
 4. 上线就可以有2.5亿设备可以使用
- 体现出Apple软硬件结合的好处

“能做什么样的东西？”

优势类型



游戏类

家居类

“我们能做什么样的东西？”

狸猫导游



功能

通过扫描二维码确定用户的位置

根据已经布置的ViewPoint进行导览

动画形象会根据虚拟地图带领你到达ViewPoint，并进行讲解

初始化AR



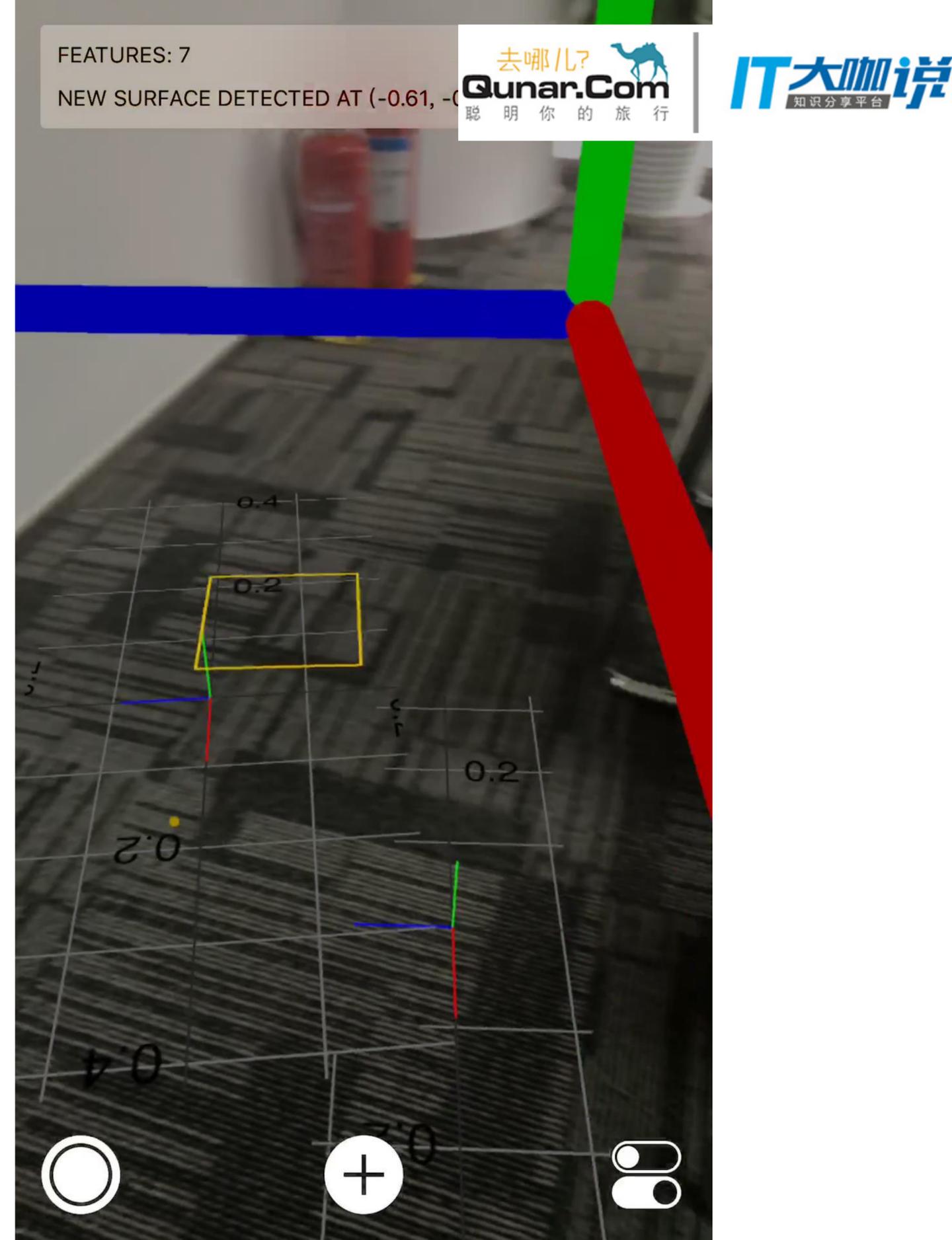
导览过程



怎么做的？

1. 开启 ARSession 得到初始坐标系

2. 查看地面获取平面坐标系



3. 扫描二维码，拿到二维码在这一帧的Frame

4. 通过二维码的frame，通过hitTest方法拿到映射到平面上的点

```
func hitTest(_ point: CGPoint,  
            types: ARHitTestResult.ResultType)
```

-> [[ARHitTestResult](#)]

5. 通过将ViewPoint之前录制在二维码坐标系中的坐标，转化为在初始坐标系的坐标



老外怎么玩的？

虚拟空间传送门



与VR设备结合



“想要更好玩的？”

“自己做啊！”

Q&A