



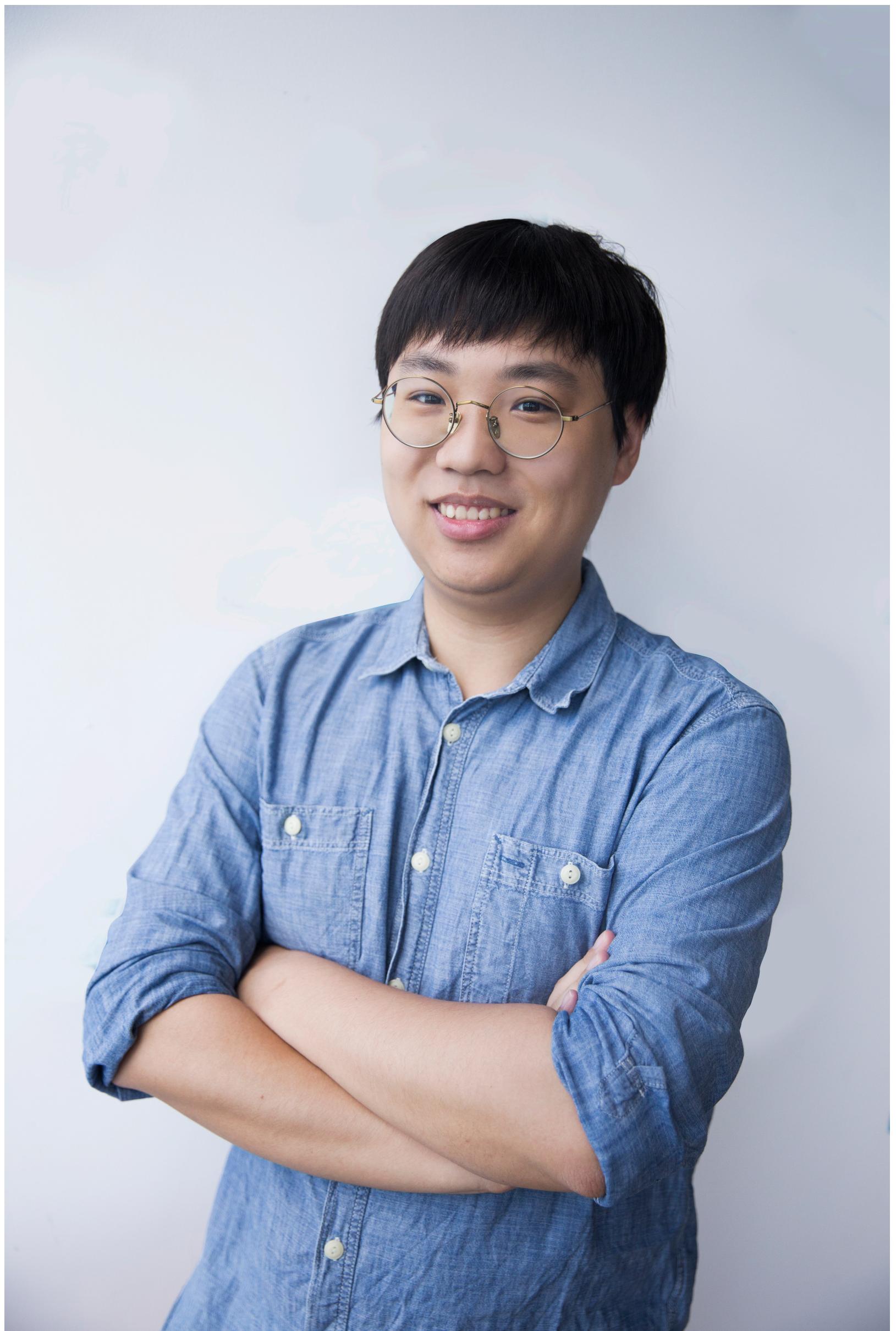
▪ 手机QQ勋章墙技术内幕

# 移动端流畅 3D 动画指南

cheeqi

# 我是谁? 1111

- Cheeqi (齐霖)
- 毕业于广东工业大学
- 2016 年正式加入腾讯
- 先后参与手机QQ、腾



Domain-Driven Design  
Tackling Complexity in the Heart of Software

# 领域驱动设计

——软件核心复杂性应对之道



(美)Eric Evans 著  
陈大峰 张泽鑫 等译

UMLChina  
特别推荐



清华大学出版社



▪ 手机QQ勋章墙技术内幕

# 移动端流畅 3D 动画指南

cheeqi



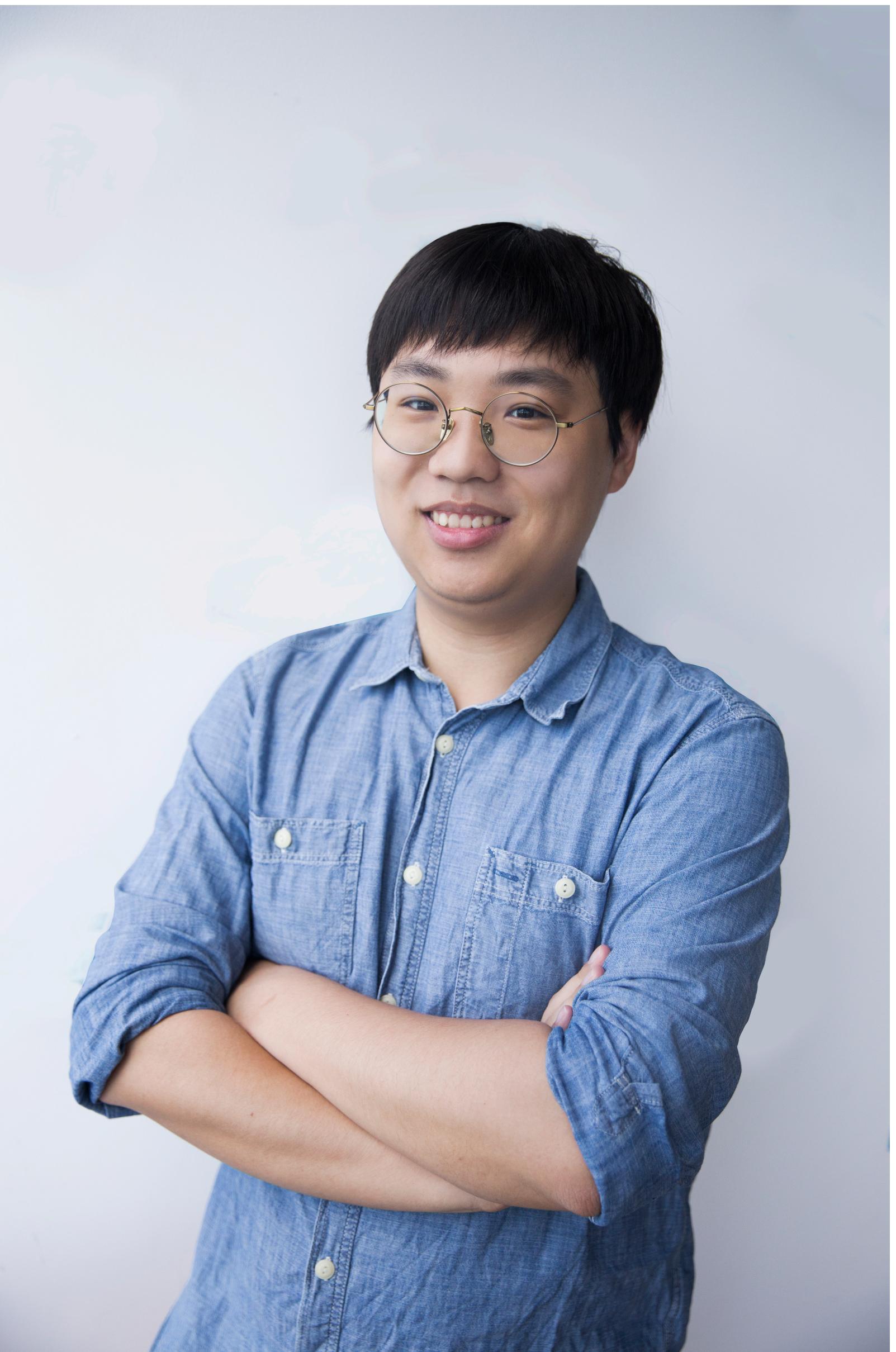
▪ 手机QQ勋章墙技术内幕

# 移动端流畅 3D 动画指南

cheeqi

# 我是谁？

- Cheeqi (齐霖)
- 毕业于广东工业大学
- 2016 年正式加入腾讯，现任职于 QQ-Web 团队
- 先后参与手机QQ、腾讯文档、QQ小程序等业务开发



# 手机QQ勋章墙

- 入口：QQ个人资料卡
- 用户量：PV千万+
- 第一批尝试webgl的产品



点击头像



点击XX枚



进入勋章墙

# 手机QQ勋章墙

- 入口：QQ个人资料卡
- 用户量：PV千万+
- 第一批尝试webgl的产品



点击头像



点击XX枚



进入勋章墙

# QQ勋章墙技术难点



实时光照



背面刻字



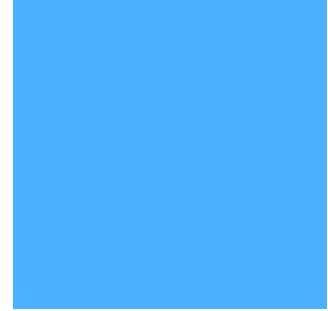
过渡动画



可控交互动画

# 目录

- 难点一：渲染带用户昵称的 3D 模型
- 难点二：实现流畅的可交互动画
- 难点三：打破次元壁
- 难点四：渲染半透明过渡动画
- 其他优化



# **难点一：渲染带昵称的 3D 勋章**

# 难点一：渲染带昵称的 3D 勋章

## 目标

- 渲染3D模型
- 模型背面刻字



# 难点一：渲染带昵称的 3D 勋章

渲染模型：技术选型

WebGL

Three.js

- 那么，应该选择什么文件格式呢？

# 难点一：渲染带昵称的 3D 勋章

## 渲染模型：格式选型

```
{  
  "metadata": {...}, // 3 items  
  "geometries": [  
    {  
      "uuid": "2E76BD82-D455-4BFA-AE2A-2670B75",  
      "type": "BufferGeometry",  
      "data": {  
        "attributes": {  
          "position": {  
            "itemSize": 3,  
            "type": "Float32Array",  
            "array": [...], // 8928 items  
            "normalized": false  
          },  
          "normal": {  
            "itemSize": 3,  
            "type": "Float32Array",  
            "array": [...], // 8928 items  
            "normalized": false  
          },  
          "uv": {  
            "itemSize": 2,  
            "type": "Float32Array",  
            "array": [...], // 5952 items  
            "normalized": false  
          }  
        },  
        "groups": [...] // 1 item  
      }  
    },  
    "materials": [  
      {  
        "uuid": "7AAB18E5-FF88-",  
        "type": "MeshPhongMaterial",  
        "color": 16714940,  
        "shading": "flat"  
      }  
    ]  
  ]  
}
```

JSON

```
Kaydara FBX Binary ... è.....  
FBXVersionIè.....Encryption  
... TypeS...UserData2... Vers  
... mixamo.com"....H...PS... Origin  
... $URL$ 88094db}....M...PS... Or  
... UpAxisSignS... intS... IntegerS...  
... FrontAxisSignS... PS ...  
... ,...PS  
... 0...PS  
... CoordAxisSignS...  
... 1...PS... OriginalUpAxisS... ir
```

FBX (解析前)

```
▼ FBXTree {FBXHeaderExtension: {...}, FileId: {...},  
  ► Connections: {singleProperty: false, connect:  
  ► CreationTime: {singleProperty: true, property:  
  ► Creator: {singleProperty: true, propertyList:  
  ► Definitions: {singleProperty: false, Version:  
  ► Documents: {singleProperty: false, Count: 1,  
  ► FBXHeaderExtension: {singleProperty: false,  
  ► FileId: {singleProperty: true, propertyList:  
  ► GlobalSettings: {singleProperty: false,  
  ► Objects:  
    ► AnimationCurve: {6...  
    ► AnimationCurveNode:  
    ► AnimationLayer: {72746416: {...}, 74020056: {}}
```

FBX (解析后)

```
# 3ds Max Wavefront OBJ Exporter v0.97  
# File Created: 12.12.2016 22:46:17  
  
#  
# object Object001  
#  
  
v -1.4416 0.6424 0.3099  
v -0.9121 0.7114 0.4134  
v -0.8927 -0.0190 0.4319  
v -1.4061 -0.0061 0.3369  
v -0.4880 0.7945 0.4324  
v -0.4629 -0.0157 0.4655  
v 1.4576 0.6317 0.3130  
v 1.9991 0.5980 0.2087  
v 1.9874 -0.0009 0.2275  
v 1.4131 -0.0074 0.3430  
v 0.4363 0.8039 0.4449  
v 0.4200 0.0162 0.4703
```

OBJ

# 难点一：渲染带昵称的 3D 勋章

## 渲染模型：格式选型

格式	JSON	FBX	OBJ + 材质贴图
文件体积	纯文本， 体积大 <span style="color:red">X</span>	二进制， 中等体积 <span style="color:green">✓</span>	中等体积 <span style="color:green">✓</span>
通用性	专用格式， 难导出 <span style="color:red">X</span>	行业通用标准 <span style="color:green">✓</span>	行业标准 <span style="color:green">✓</span>
复杂度	内容包含较复杂嵌套 <span style="color:red">X</span>	内容包含较复杂嵌套 <span style="color:red">X</span>	内容简单 <span style="color:green">✓</span>
操作贴图	需要hook解析过程 <span style="color:red">X</span>	需要hook解析过程 <span style="color:red">X</span>	容易操作 <span style="color:green">✓</span>
结论	不通用 <span style="color:red">X</span>	可以， 但不必要	<span style="color:green">✓</span>

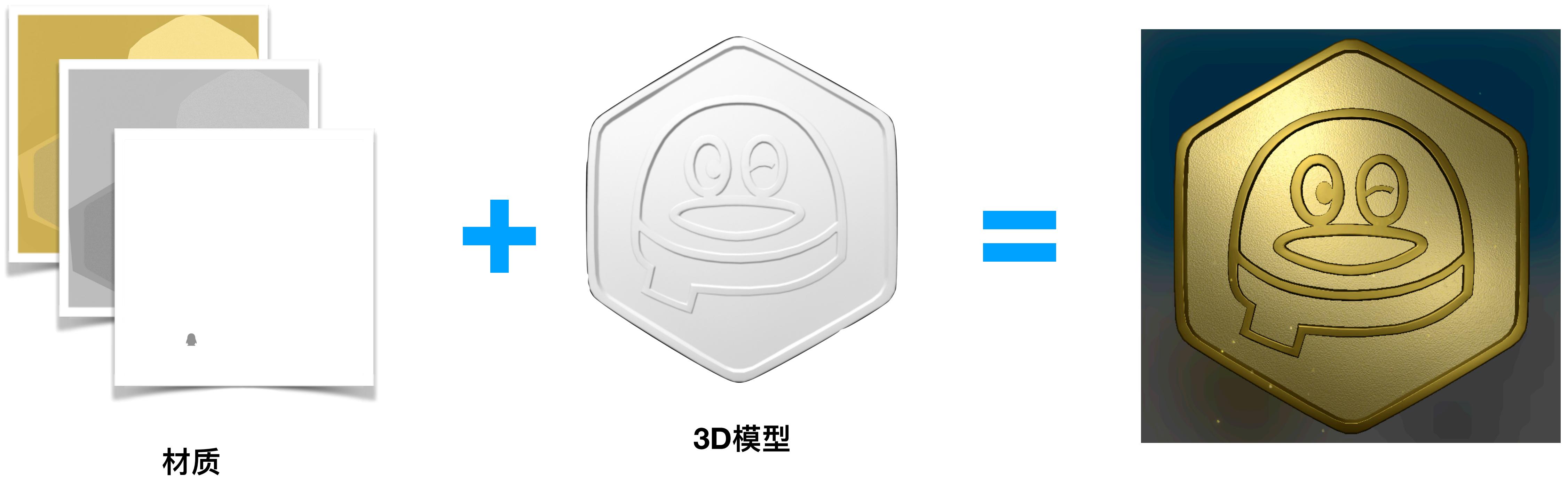
# 难点一：渲染带昵称的 3D 勋章

目标：勋章刻字



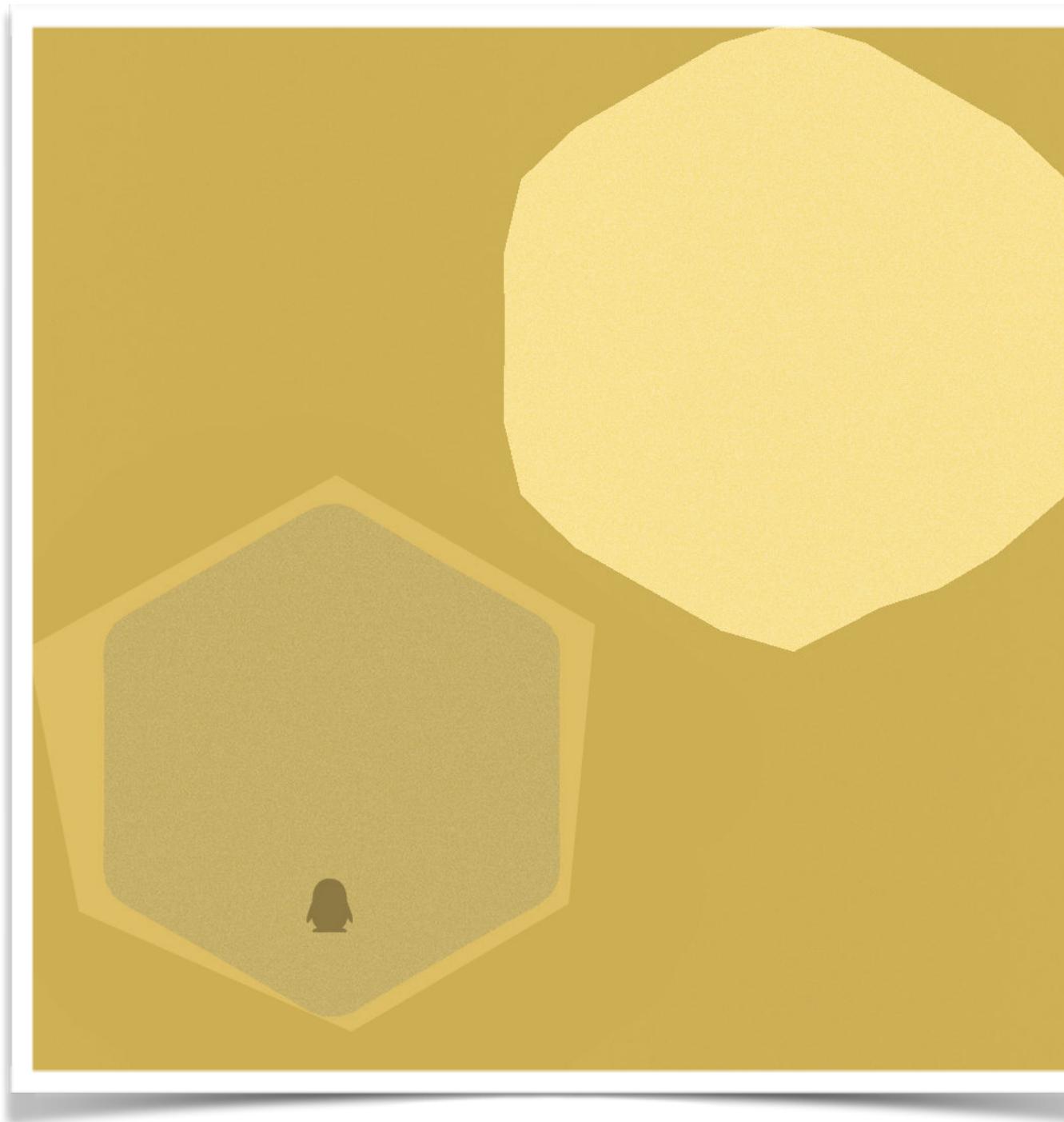
# 难点一：渲染带昵称的 3D 勋章

## 勋章刻字：模型的组成

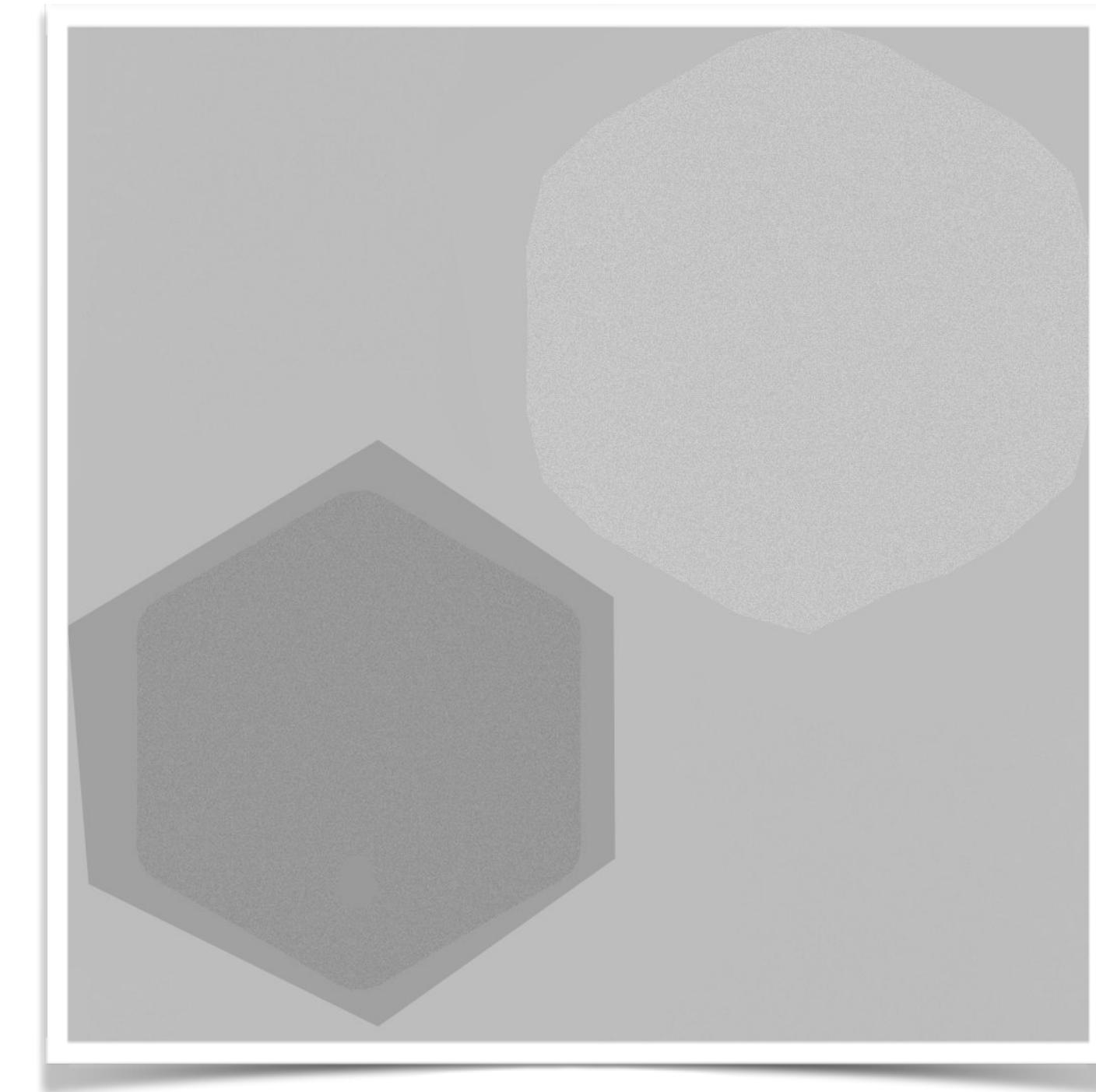


# 难点一：渲染带昵称的 3D 勋章

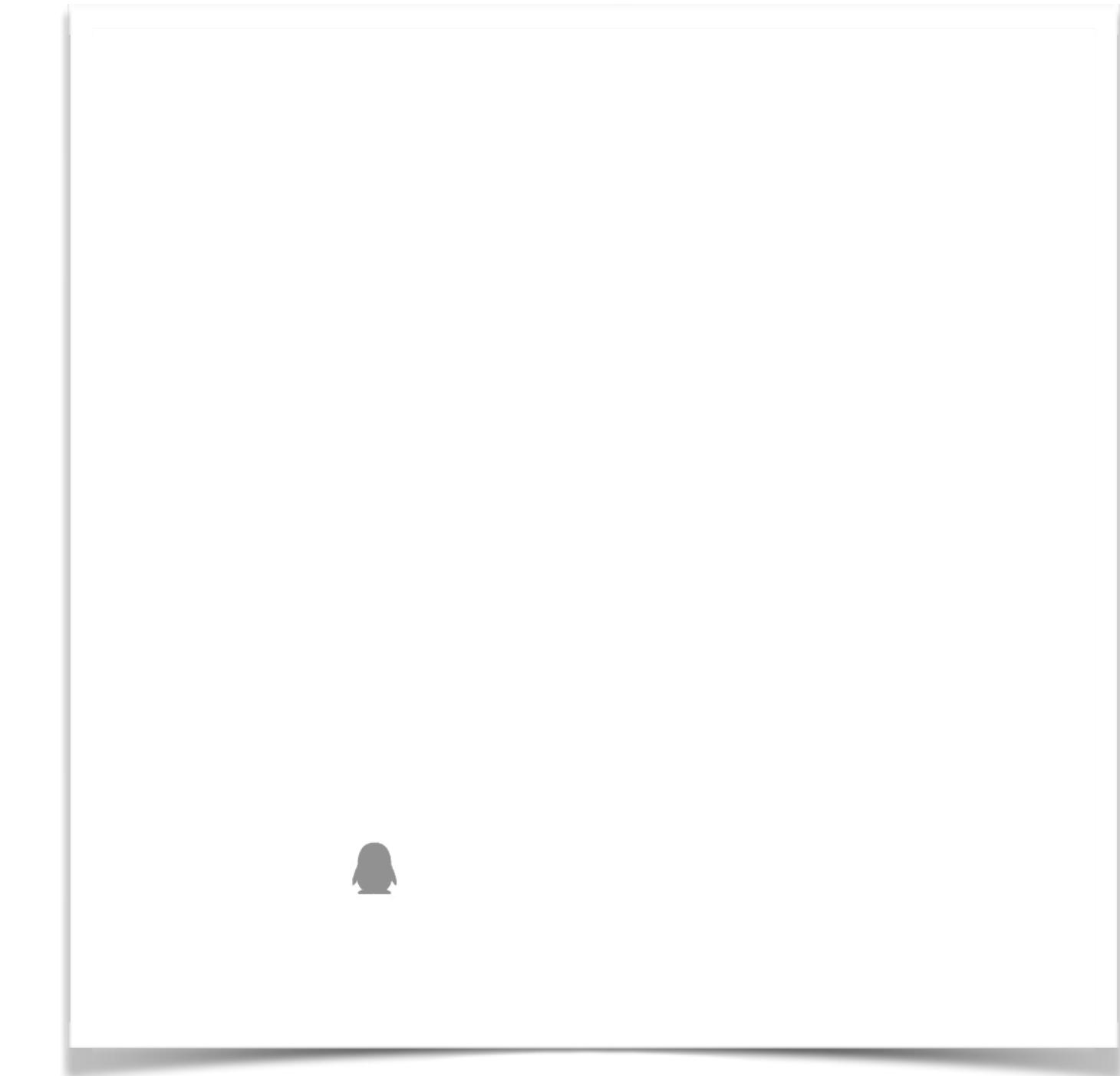
## 勋章刻字：模型贴图



固有色贴图



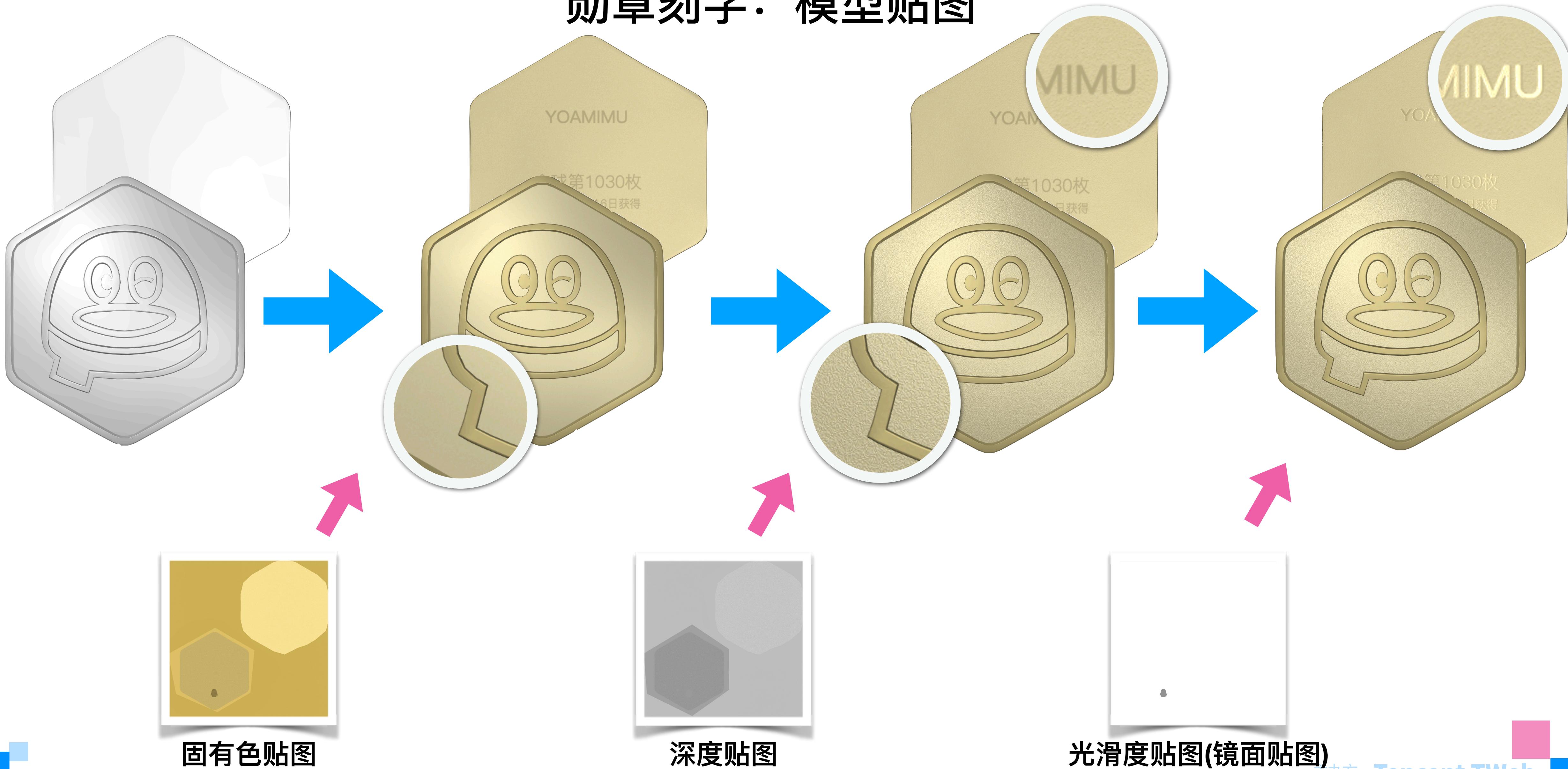
深度贴图



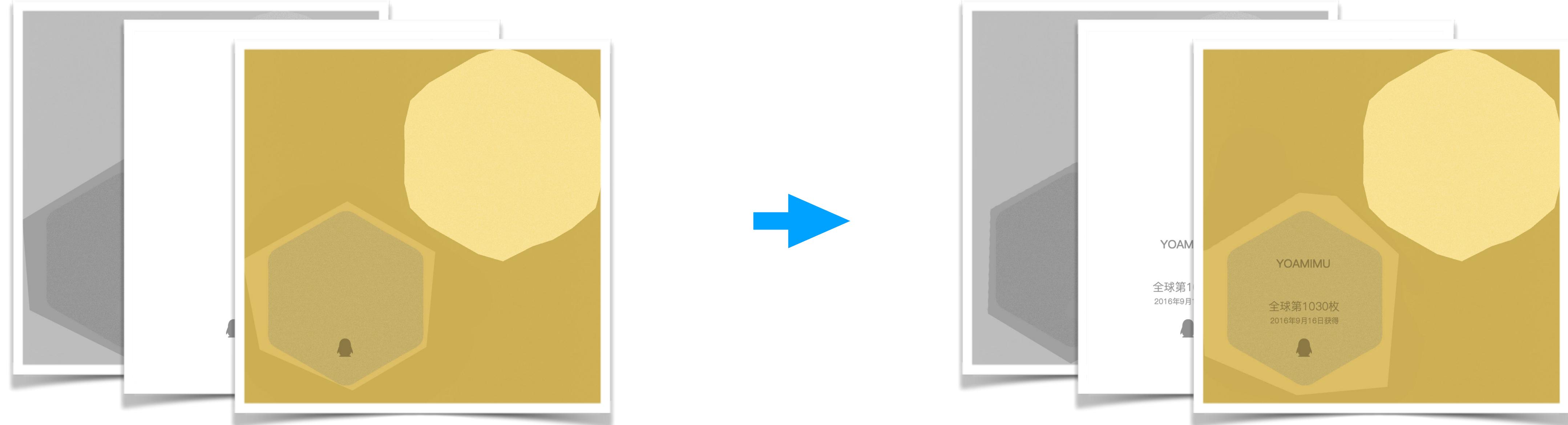
光滑度贴图(镜面贴图)

# 难点一：渲染带昵称的 3D 勋章

## 勋章刻字：模型贴图



# 难点一：渲染带昵称的 3D 勋章 勋章刻字

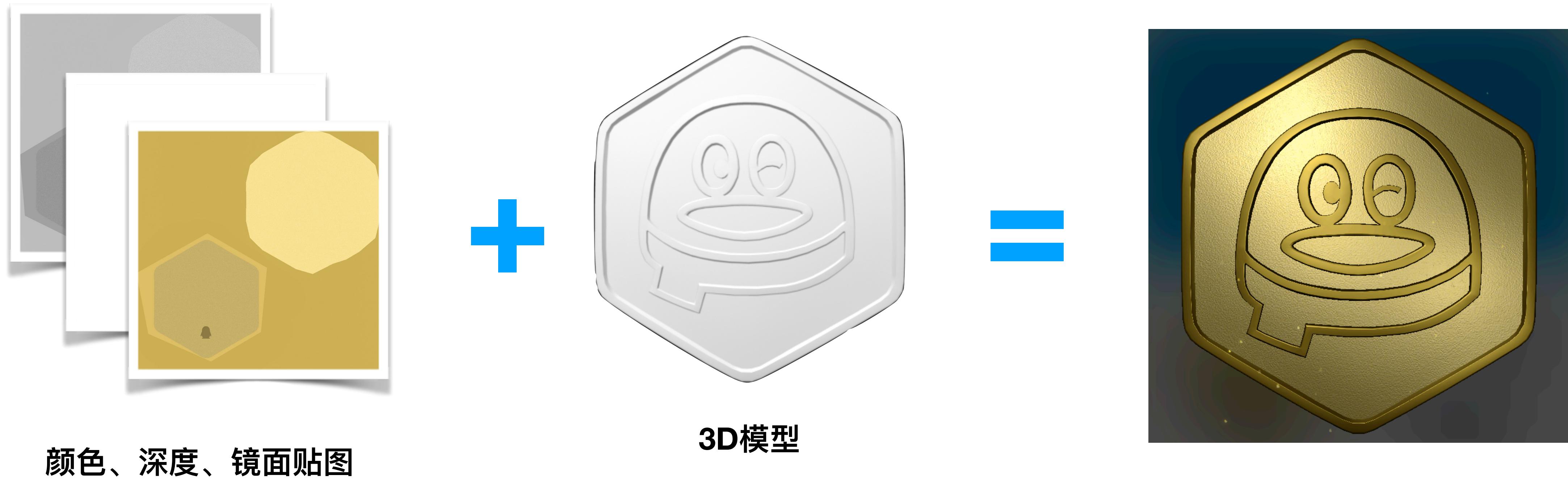


生成材质前用 canvas 在贴图上写字

文字应该写在什么位置呢？

# 难点一：渲染带昵称的 3D 勋章

勋章刻字：模型的组成

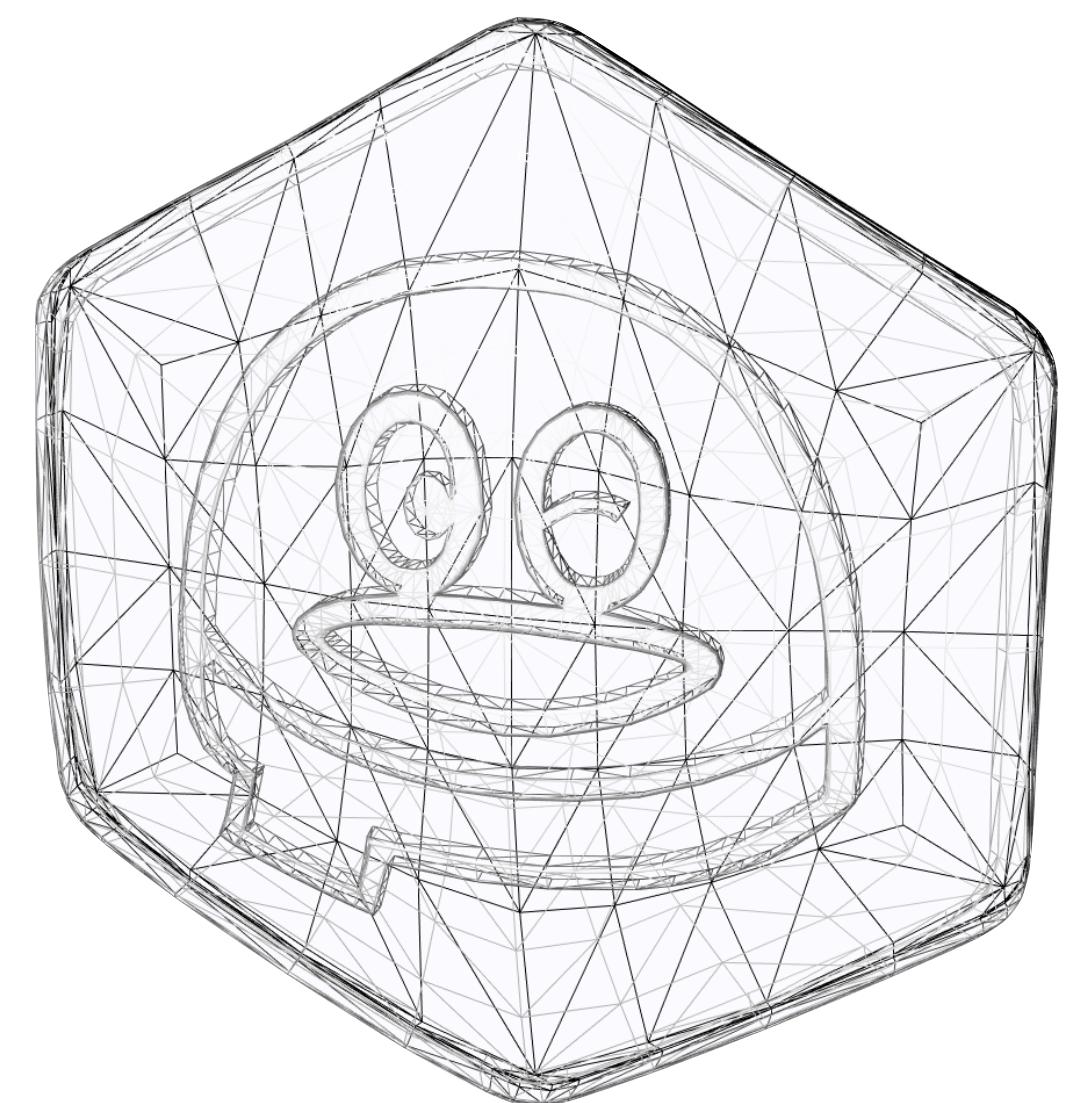


# 难点一：渲染带昵称的 3D 勋章

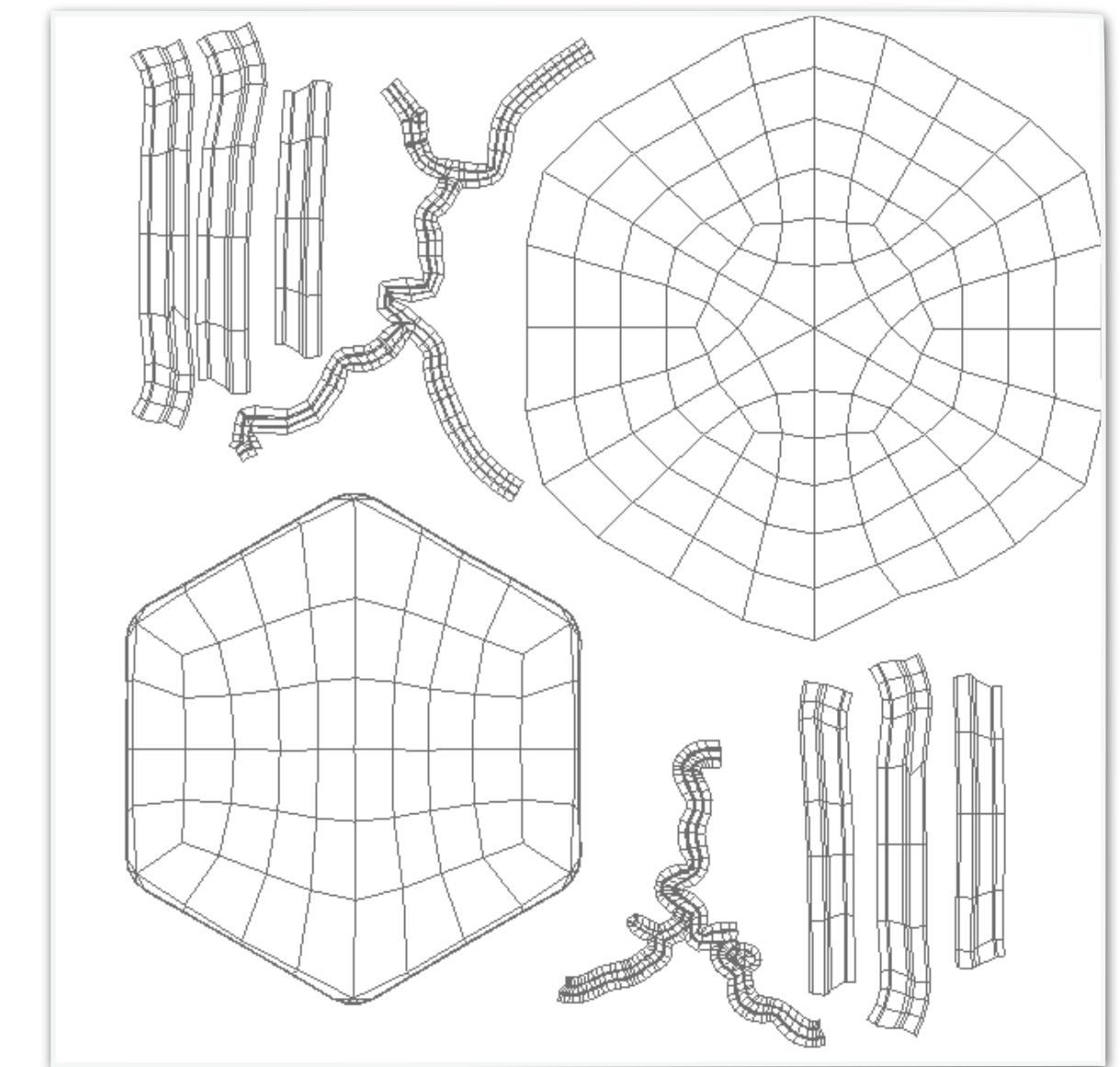
勋章刻字：模型的组成



=



+



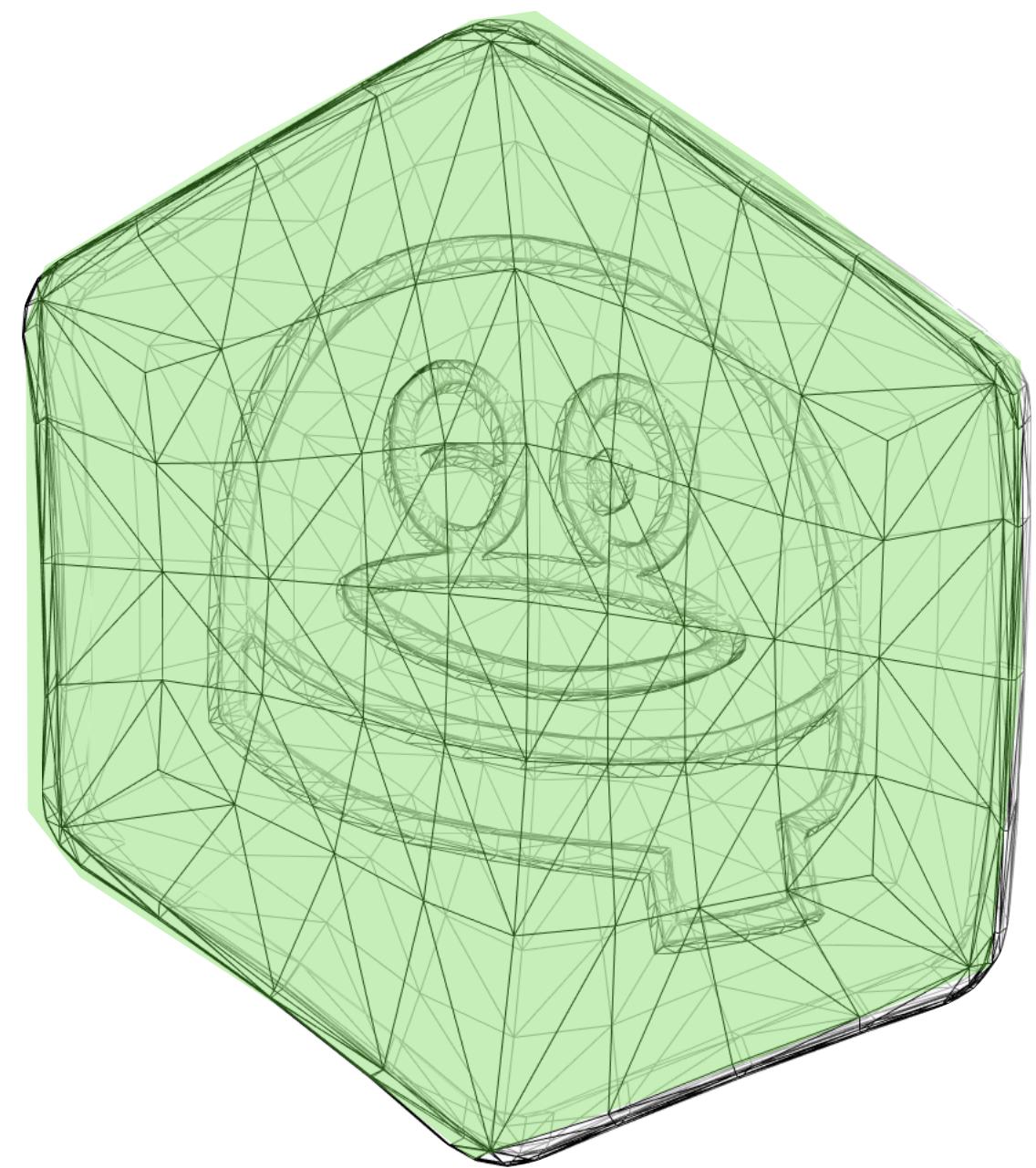
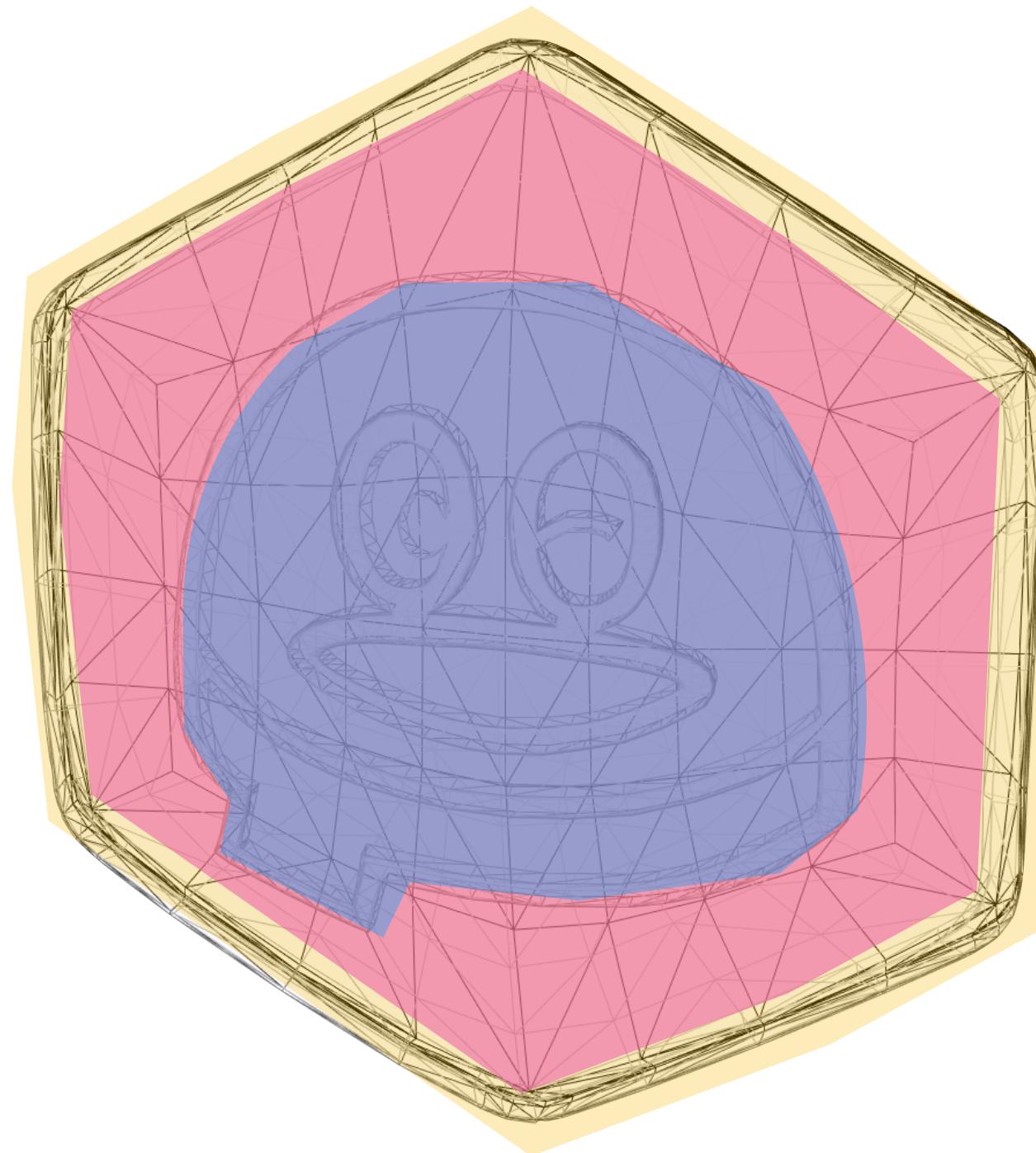
3D模型

网格（模型顶点）

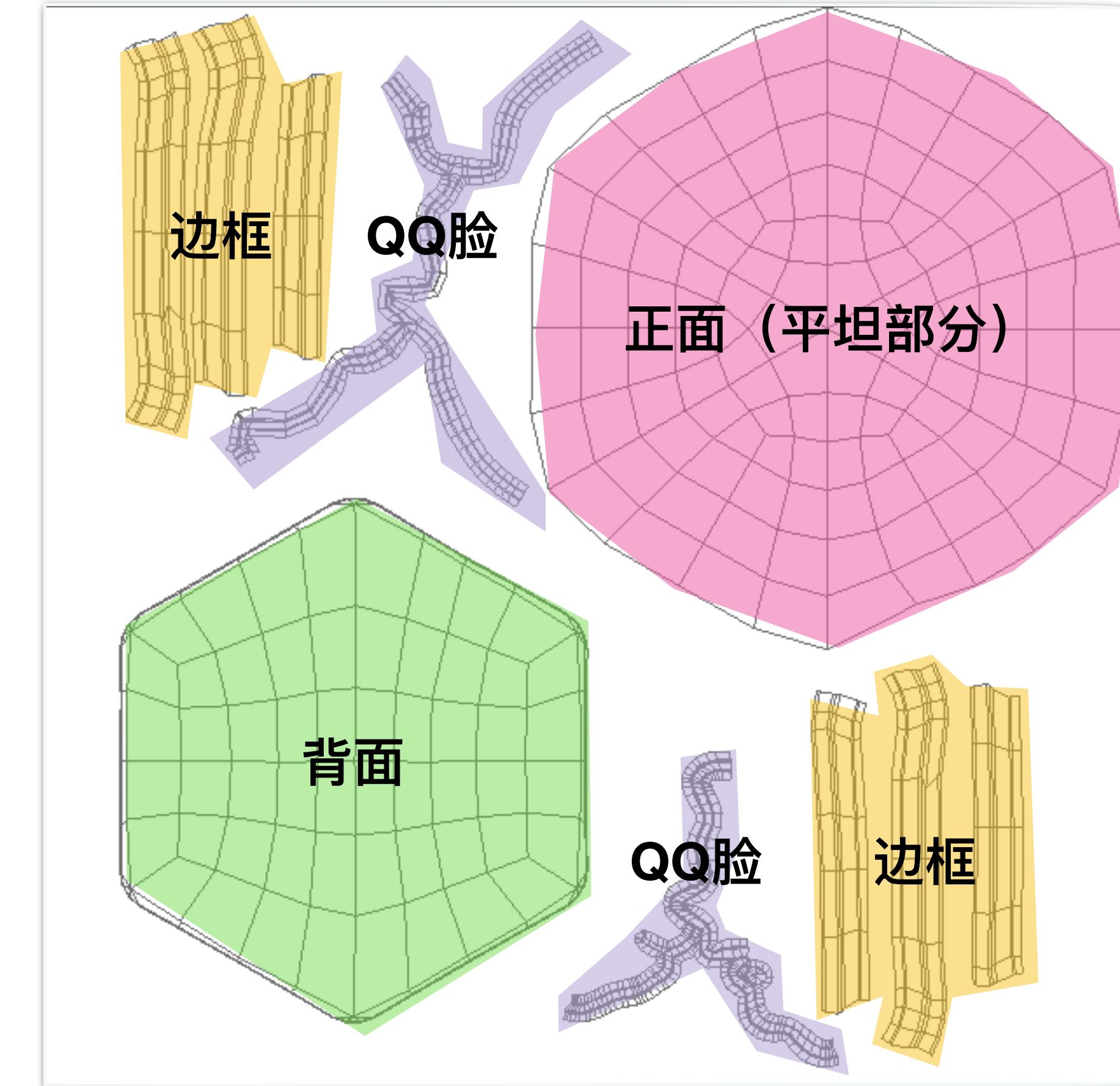
UV信息（贴图坐标映射）

# 难点一：渲染带昵称的 3D 勋章

勋章刻字：模型的组成



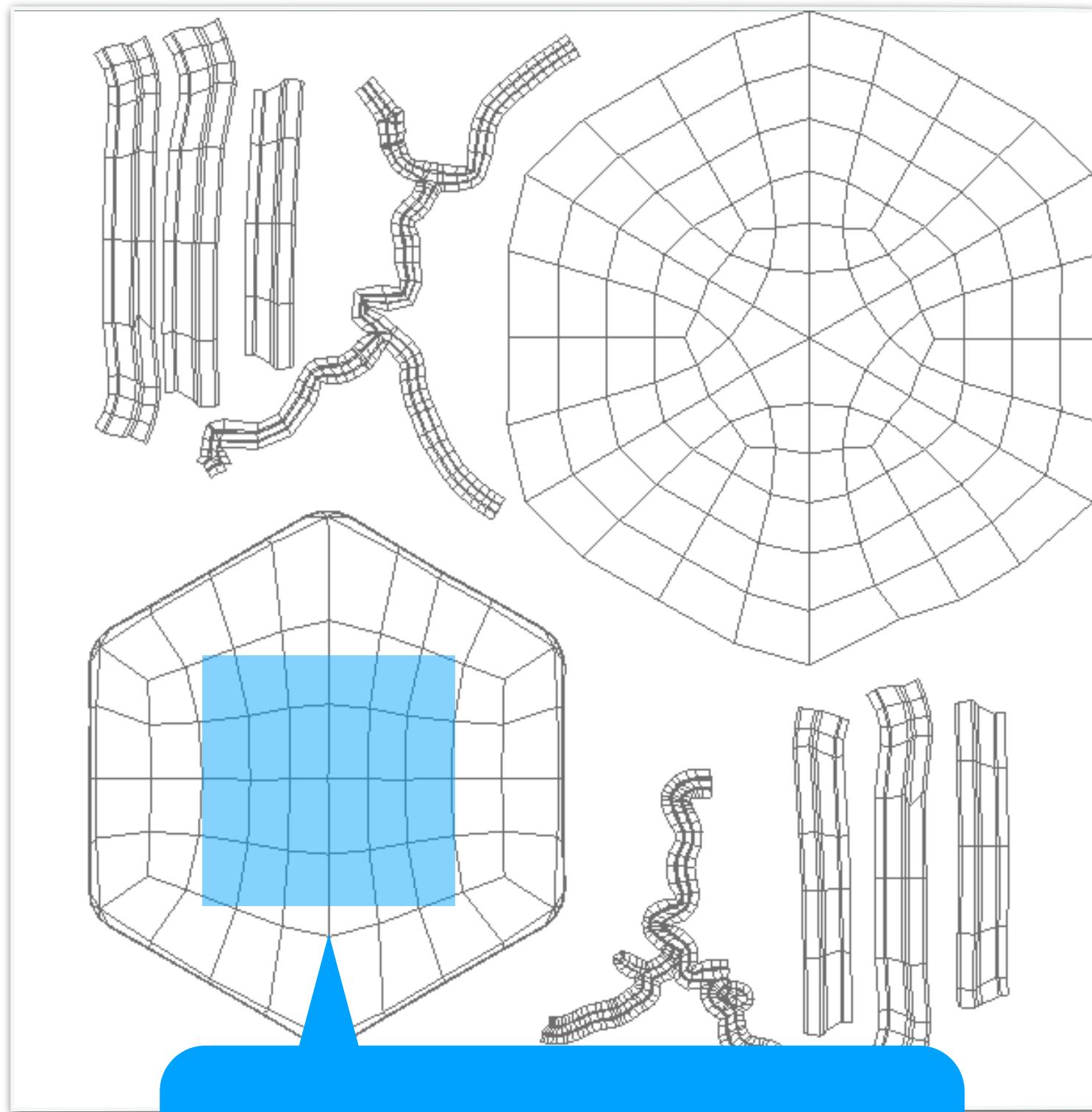
网格（模型顶点）



UV信息（贴图坐标映射）

# 难点一：渲染带昵称的 3D 勋章

勋章刻字：模型的组成



设计师建模时调整UV，  
保证背面在这个位置

```
textCanvasPos: {  
    w: 541, h: 541, x: 21, y: 421  
}
```

代码中配置text绘制区域的坐标

# 难点一：渲染带昵称的 3D 勋章

## 勋章刻字：方案总结

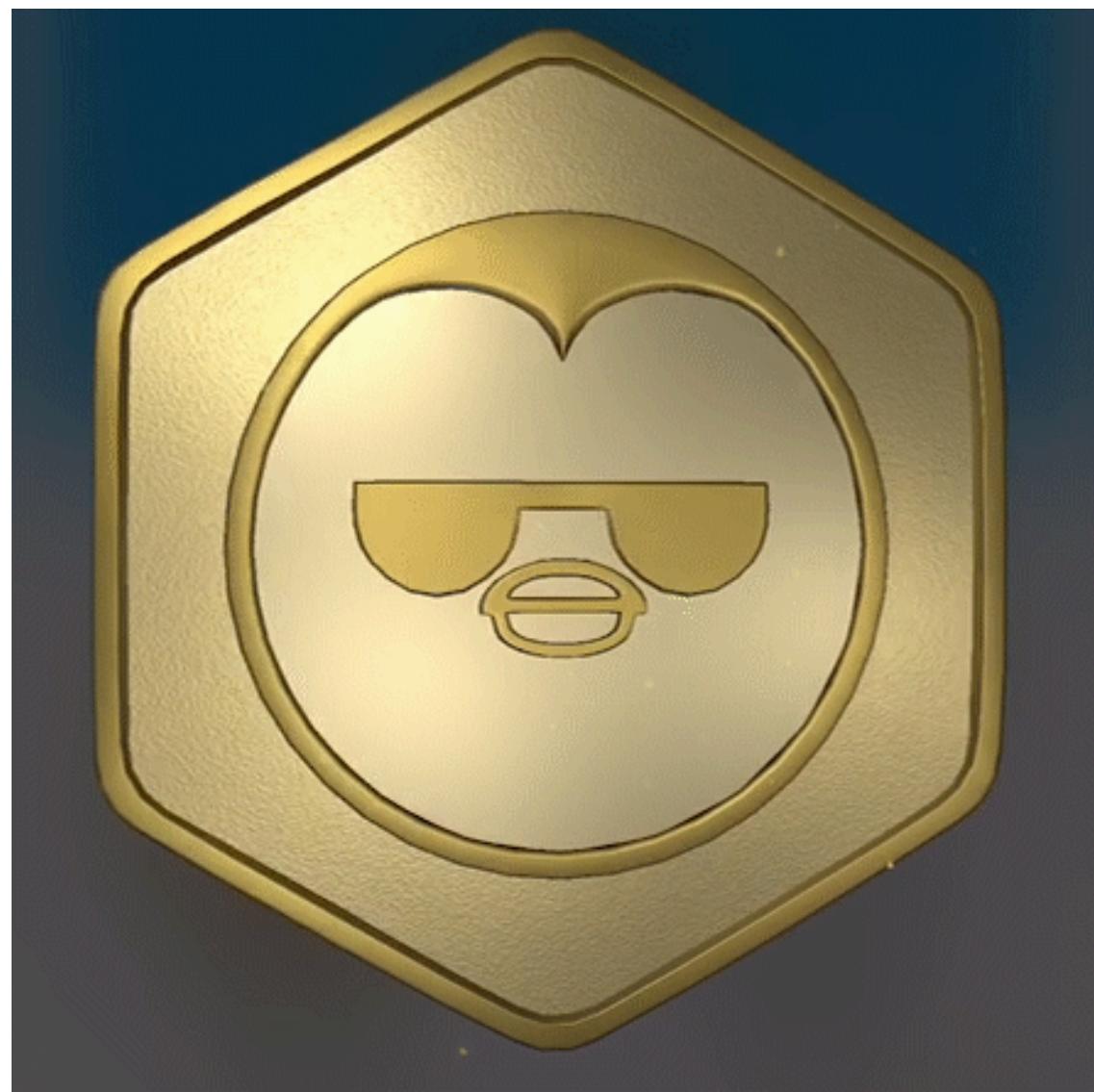
- 定制模型UV规范
- 基于 canvas 在贴图上绘制文字
- 将绘制后的canvas使用在模型上





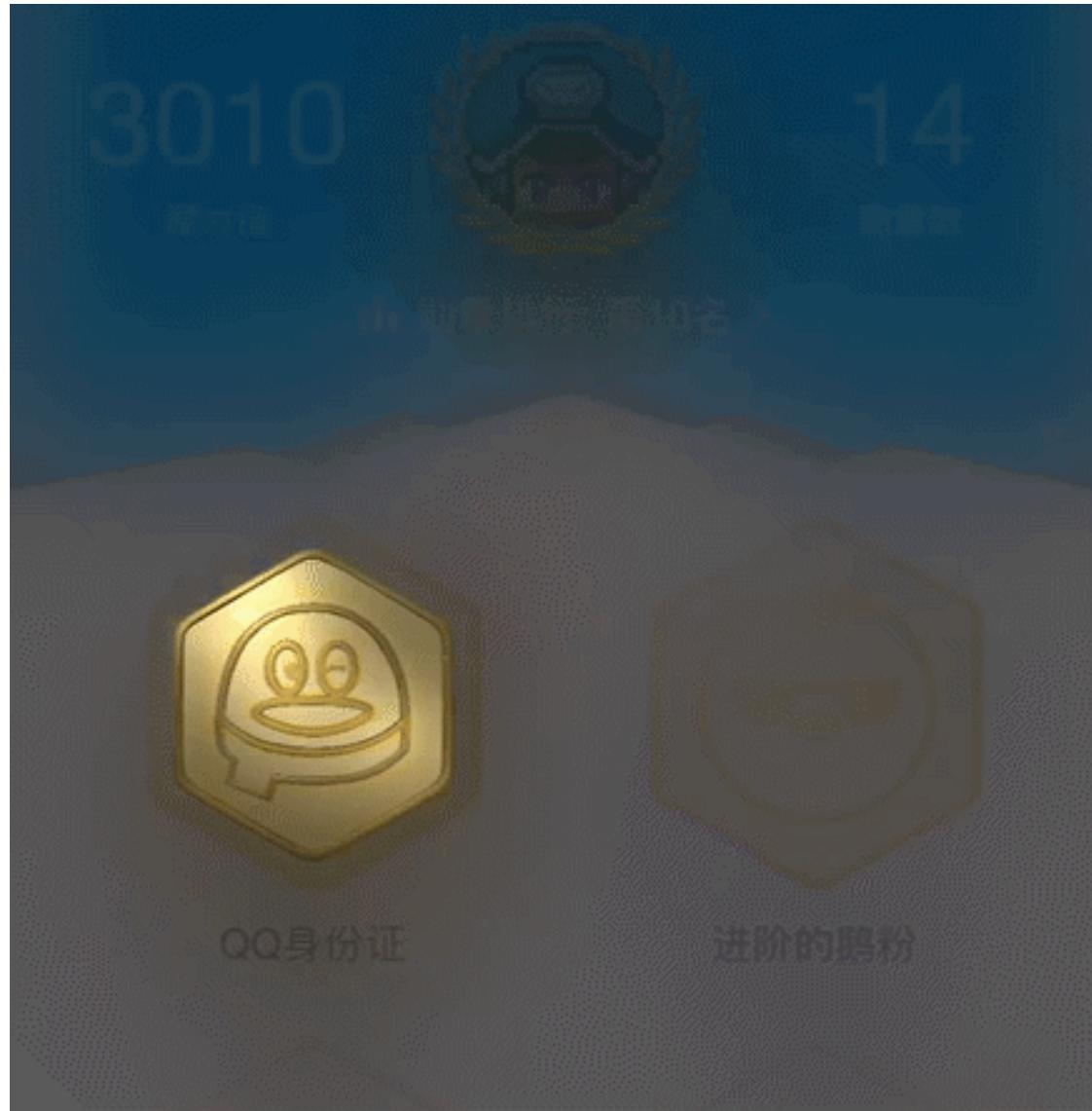
**难点二：制作流畅的可交互动画**

# 勋章墙要达到的效果



滑动旋转、自动停止

可交互



匀速出场 → 抖动结尾

动画衔接流畅

# 难点二：制作流畅的可交互动画

“可交互”的目标



慢慢转



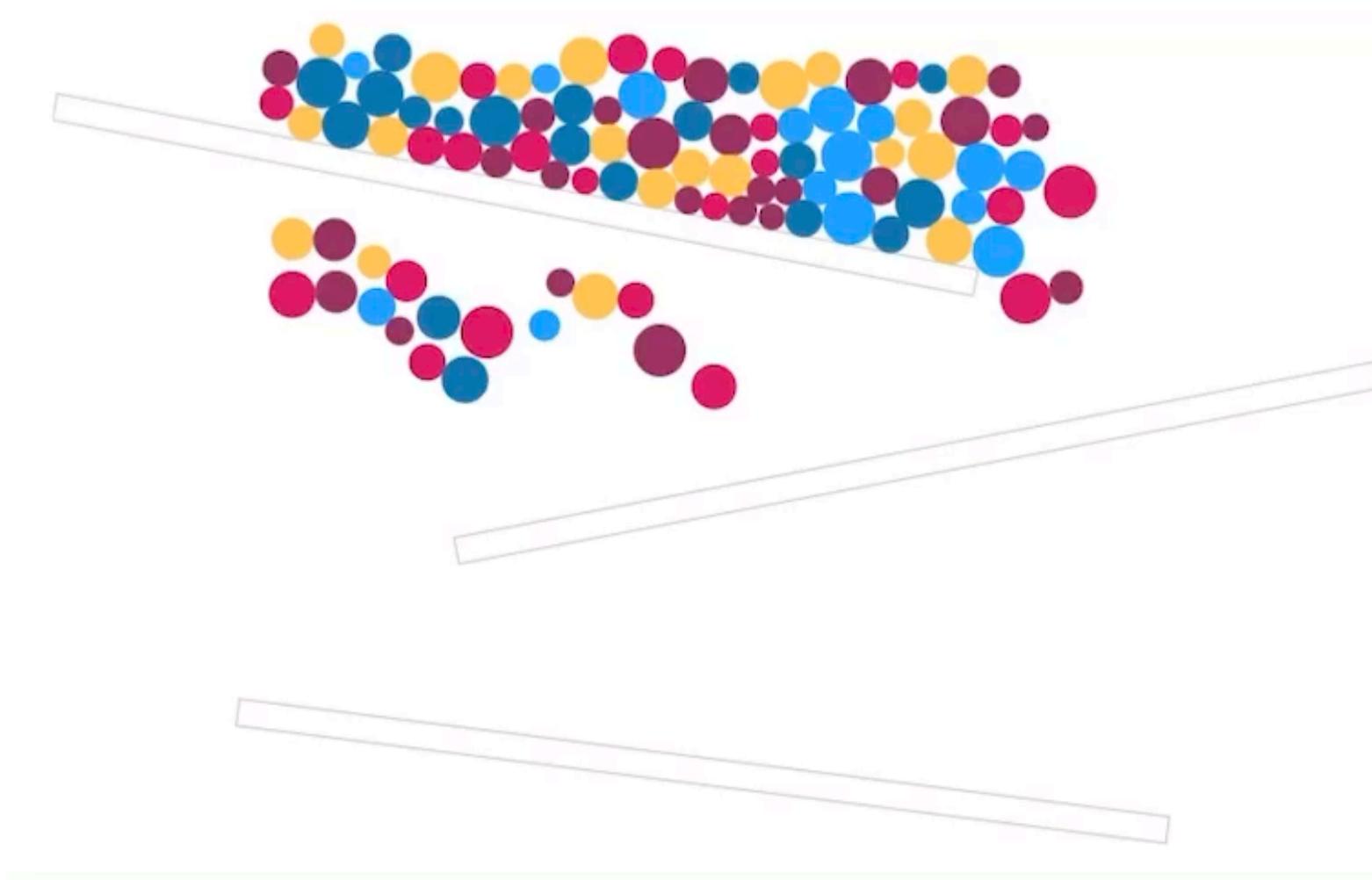
“用力”转

- 用户可以旋转勋章
- 当用户松手后，勋章缓慢停止在正面或者反面
- 交互动画需要衔接自勋章旋转的出场动画

代码怎么表达这种**物理规律**呢？

# 难点二：制作流畅的可交互动画

## 使用物理引擎？



 matter.js

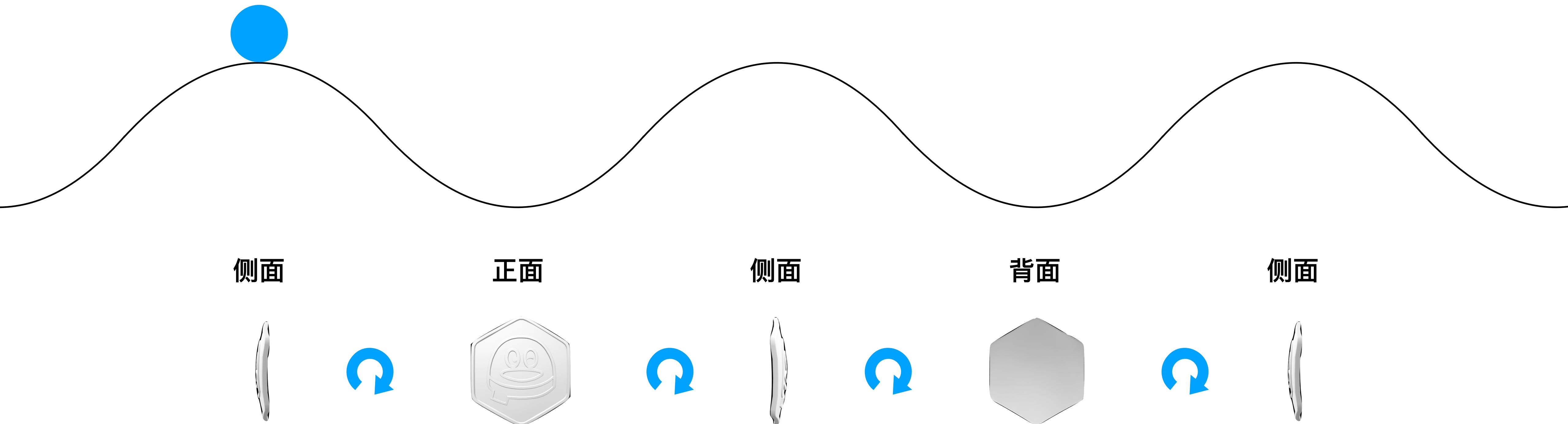
- 模拟物体选择、移动
- 模拟物体间相互作用
- 模拟重力、支持力、摩擦力
- .....

勋章墙并不需要如此复杂的物理效果

能不能自己实现？

# 难点二：制作流畅的可交互动画

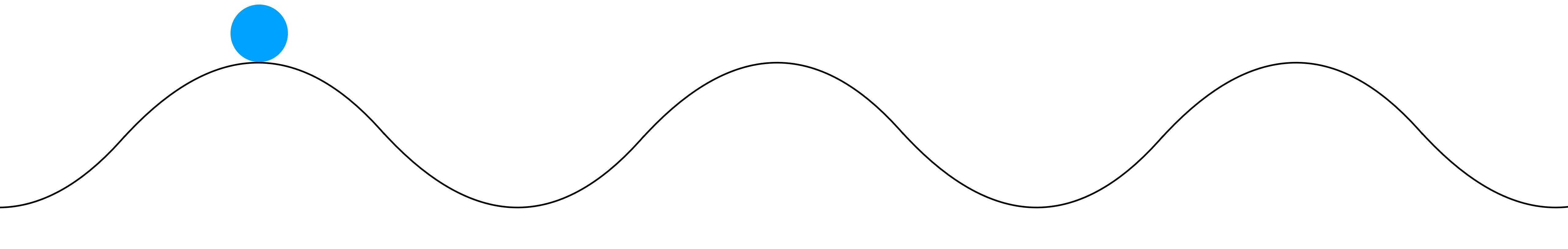
## 勋章旋转效果



勋章的旋转动画，有点像小球在凹凸不平的地面运动

# 难点二：制作流畅的可交互动画

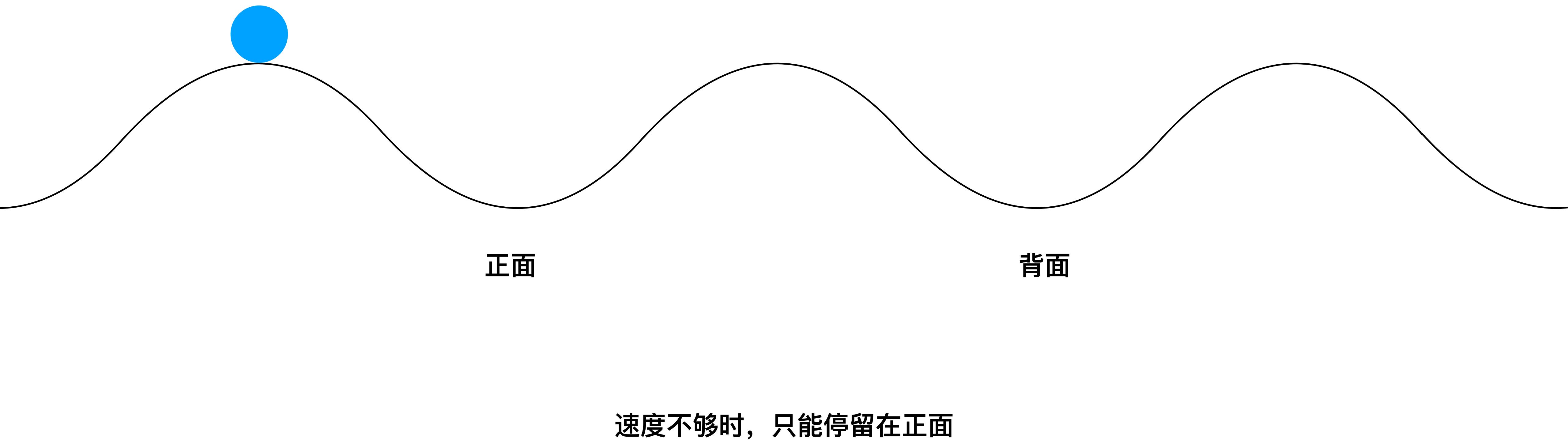
## 勋章旋转效果



小球速度够快时，可以从正面转到背面

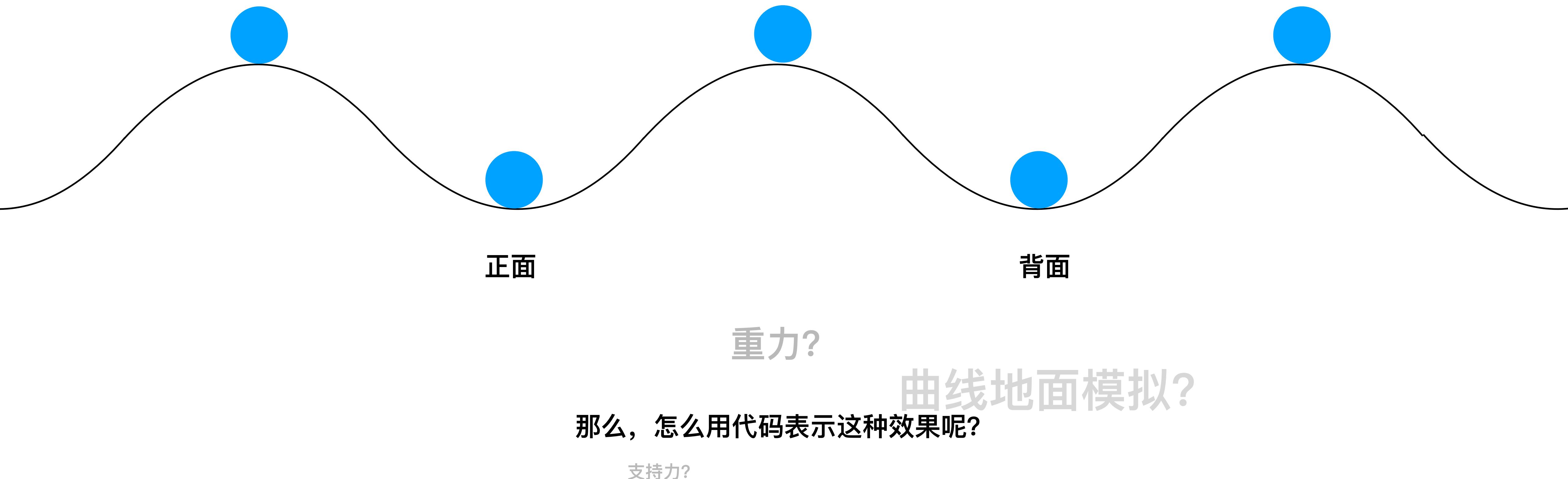
# 难点二：制作流畅的可交互动画

## 勋章旋转效果



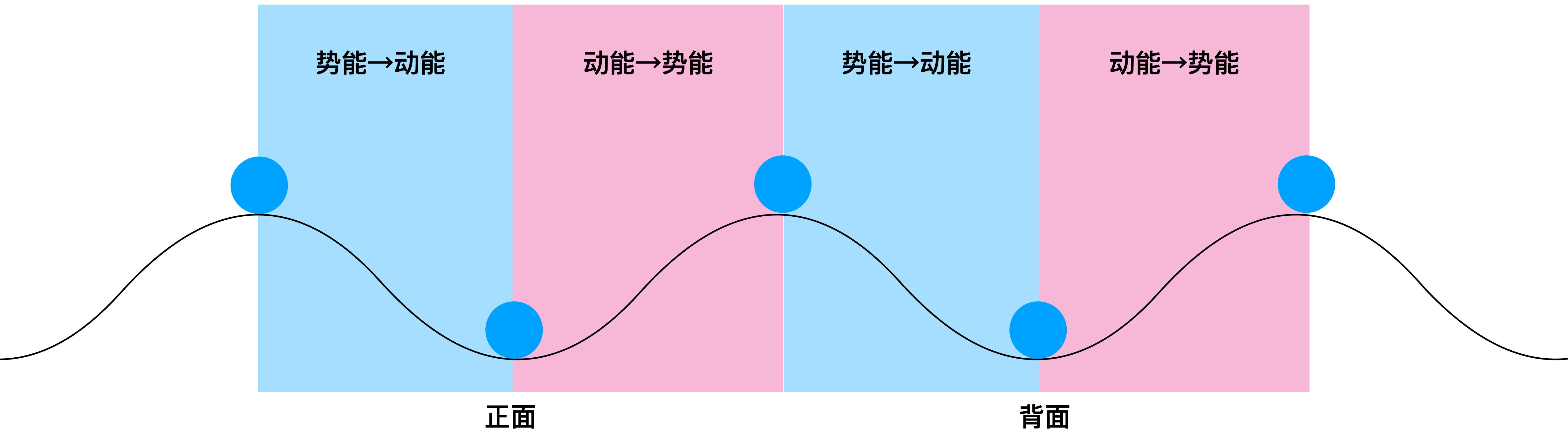
# 难点二：制作流畅的可交互动画

## 勋章旋转效果



# 难点二：制作流畅的可交互动画

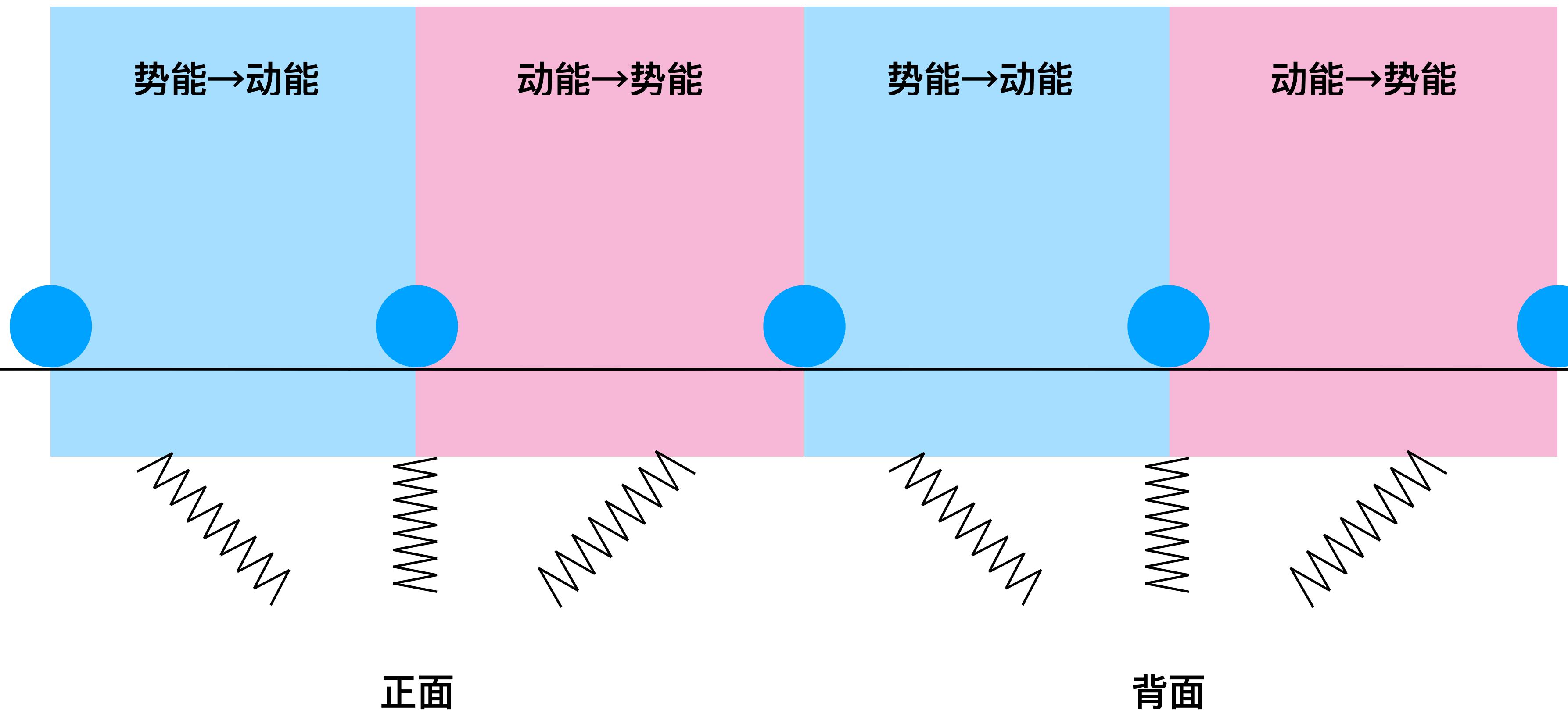
## 勋章旋转效果



可以归纳为能量转换的模型！

# 难点二：制作流畅的可交互动画

## 勋章旋转效果

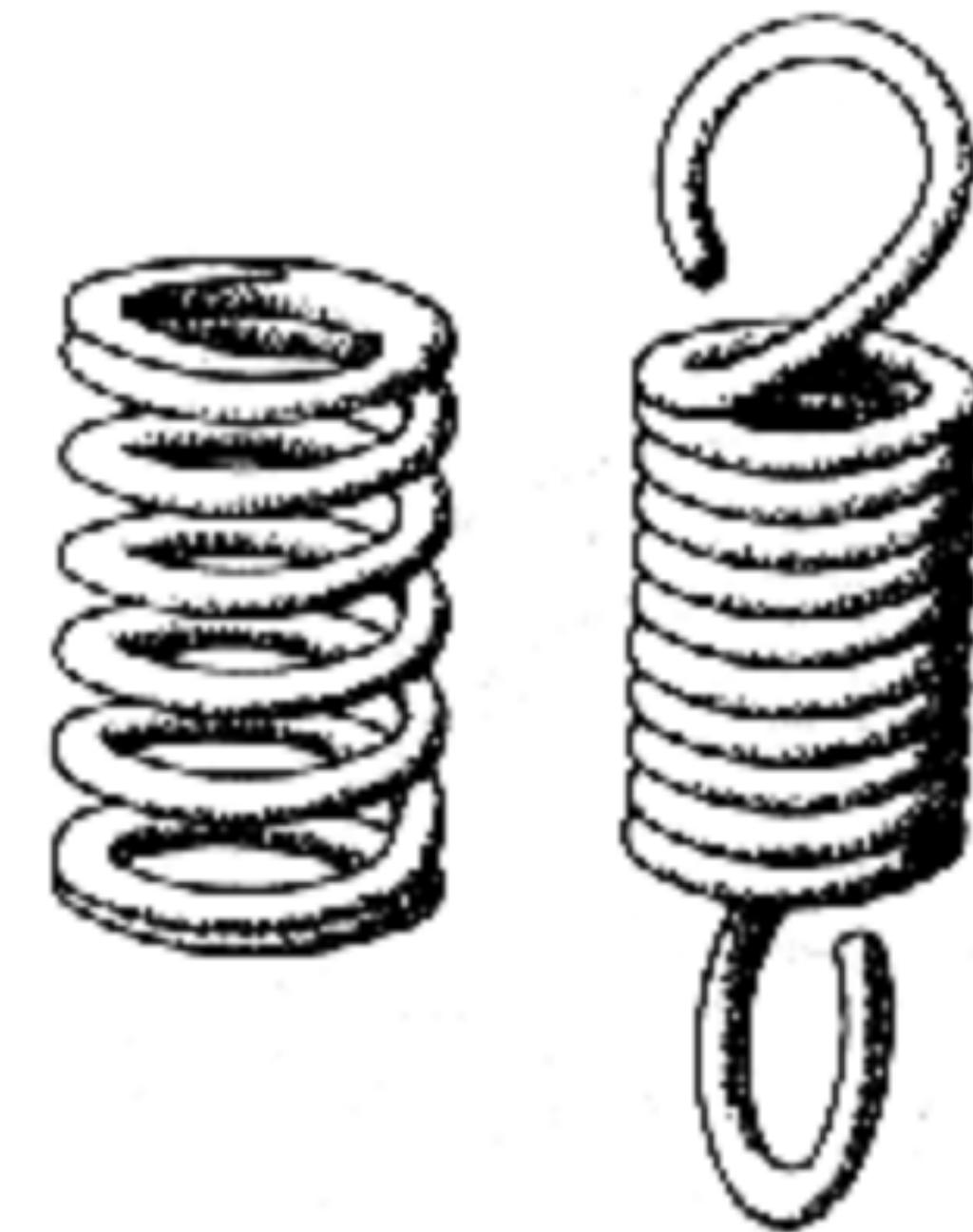


简化成弹簧就行了！

# 难点二：制作流畅的可交互动画

## 勋章旋转效果

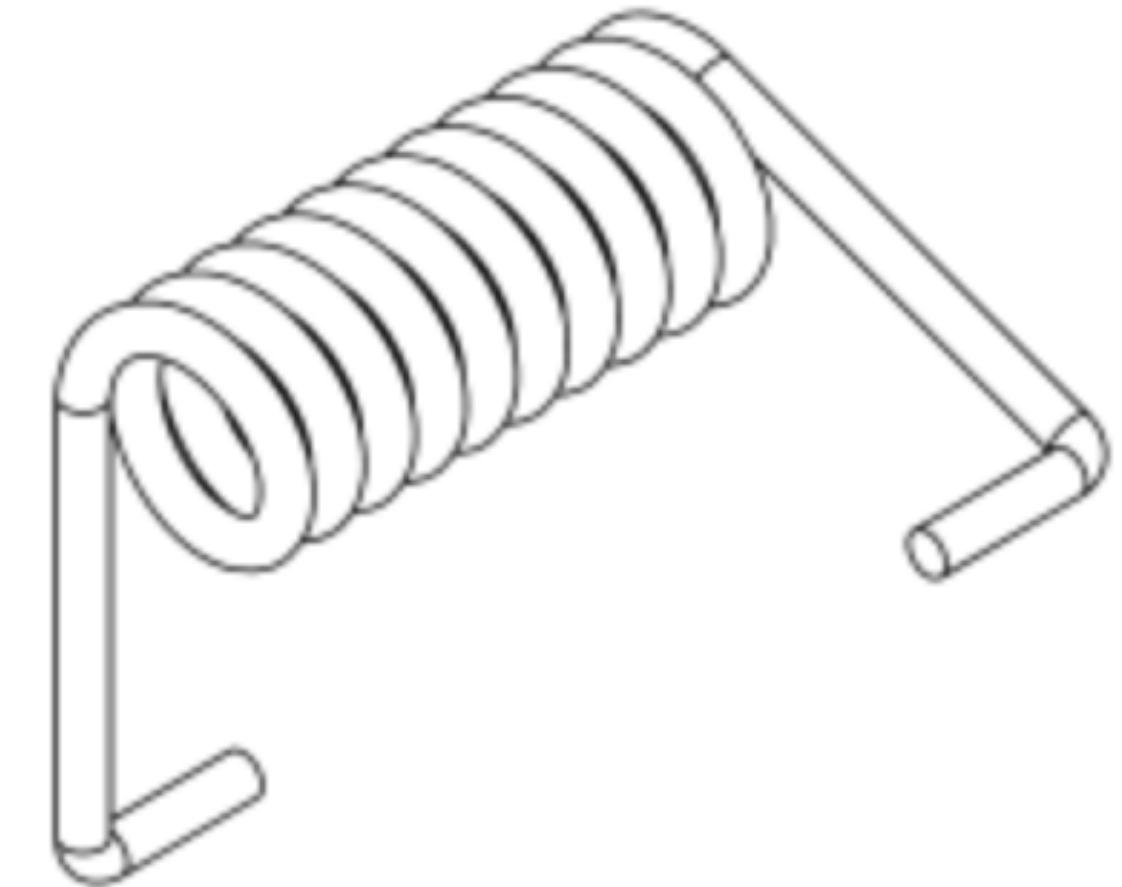
$$F = -k \cdot x$$



胡克定律（伸缩弹簧）

# 难点二：制作流畅的可交互动画

## 勋章旋转效果



$$F = -k \cdot \theta$$

弹簧弹力

勋章方向与正面/背面的夹角

胡克定律（扭转弹簧）

# 难点二：制作流畅的可交互动画

## 勋章旋转效果

弹簧弹力

$$F = -k \cdot \theta$$

勋章方向与正面/背面的夹角

加速度

$$a = F/m$$

速度

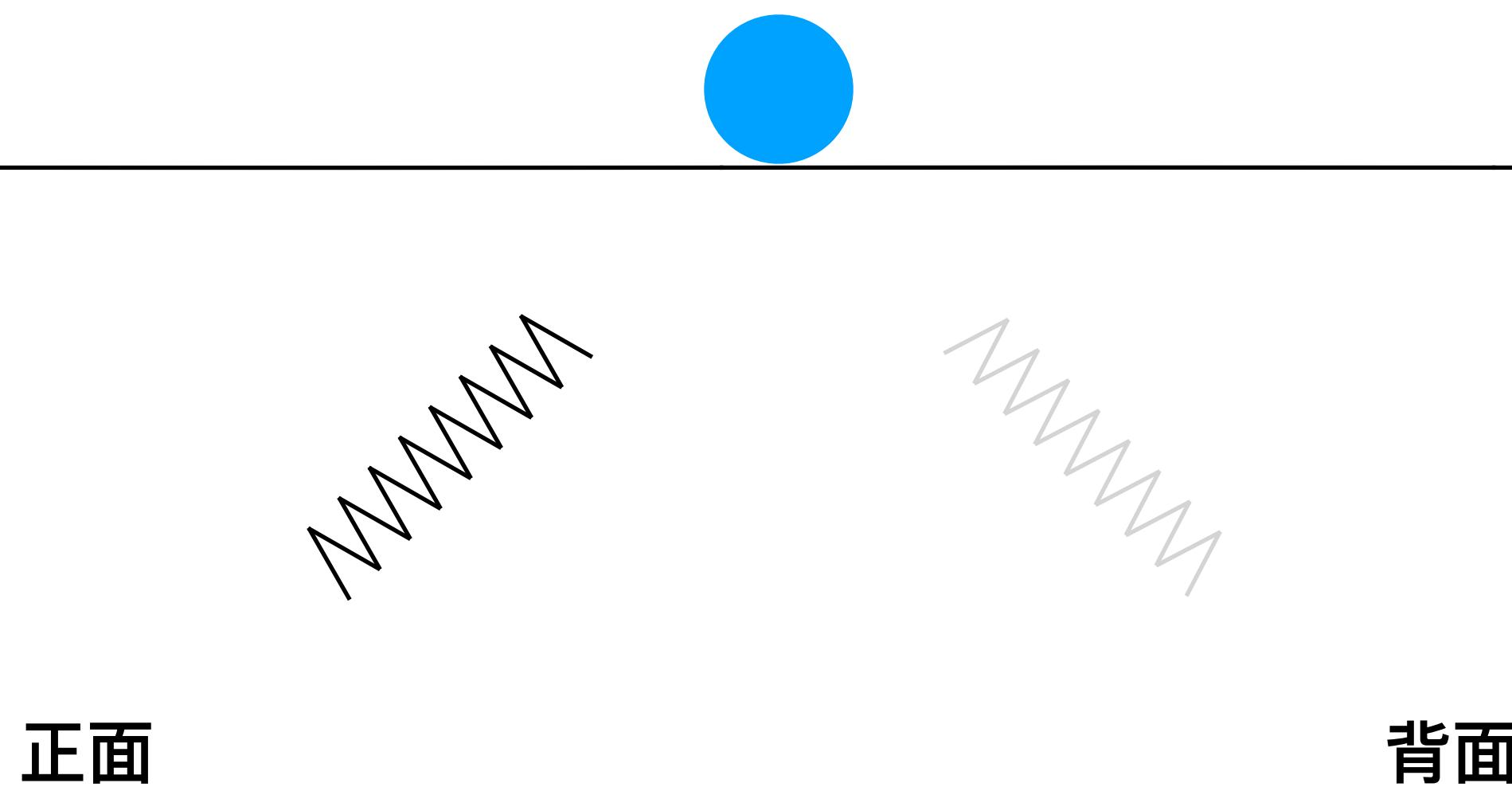
$$v' = v + a$$

旋转角度

$$\theta' = \theta + v'$$

# 难点二：制作流畅的可交互动画

## 勋章旋转效果



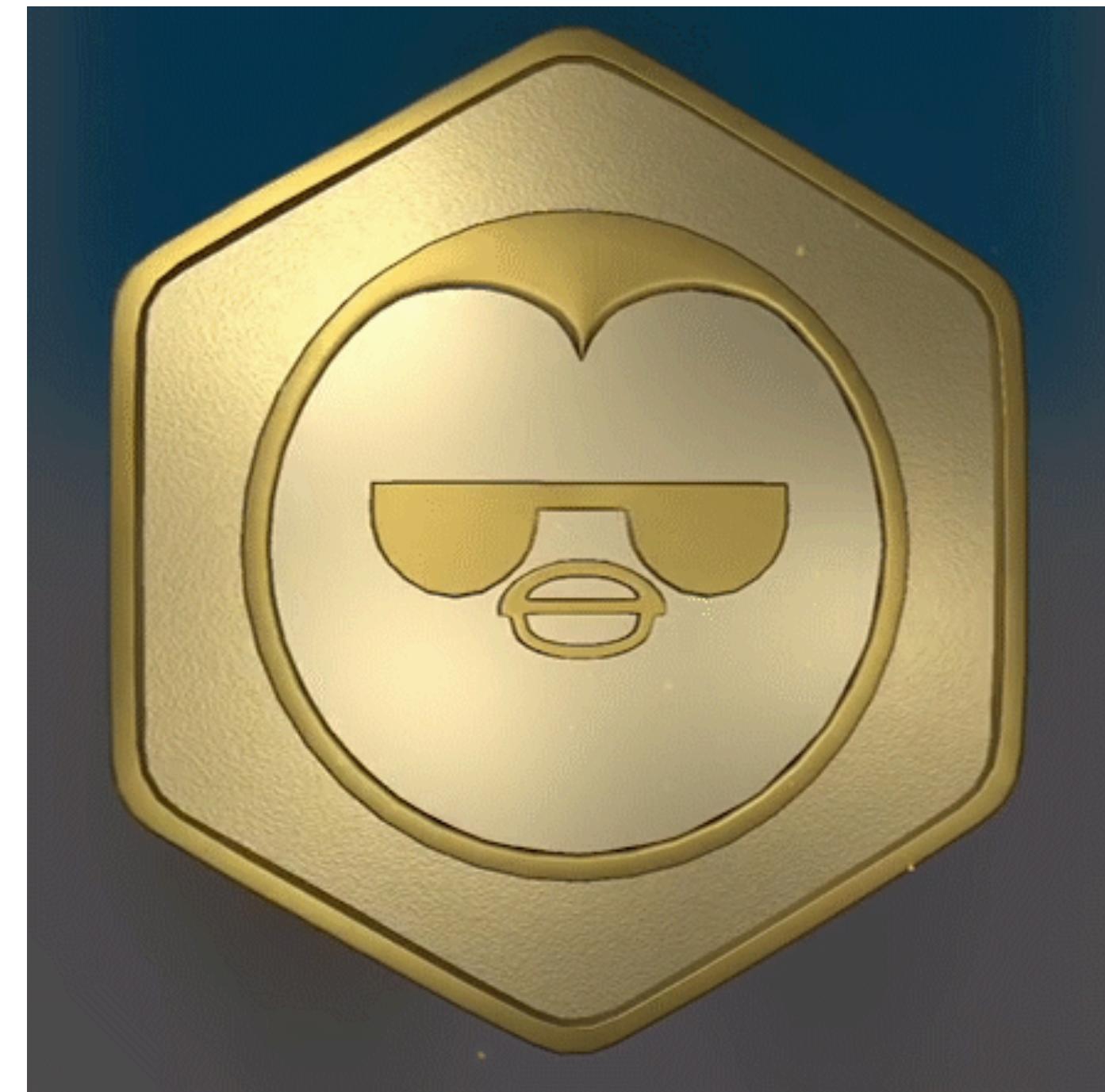
还有些要注意的地方：

- 同一时间只能有一个弹簧起作用
- 如果刚好处在临界点，就往前转，防止卡住

# 难点二：制作流畅的可交互动画

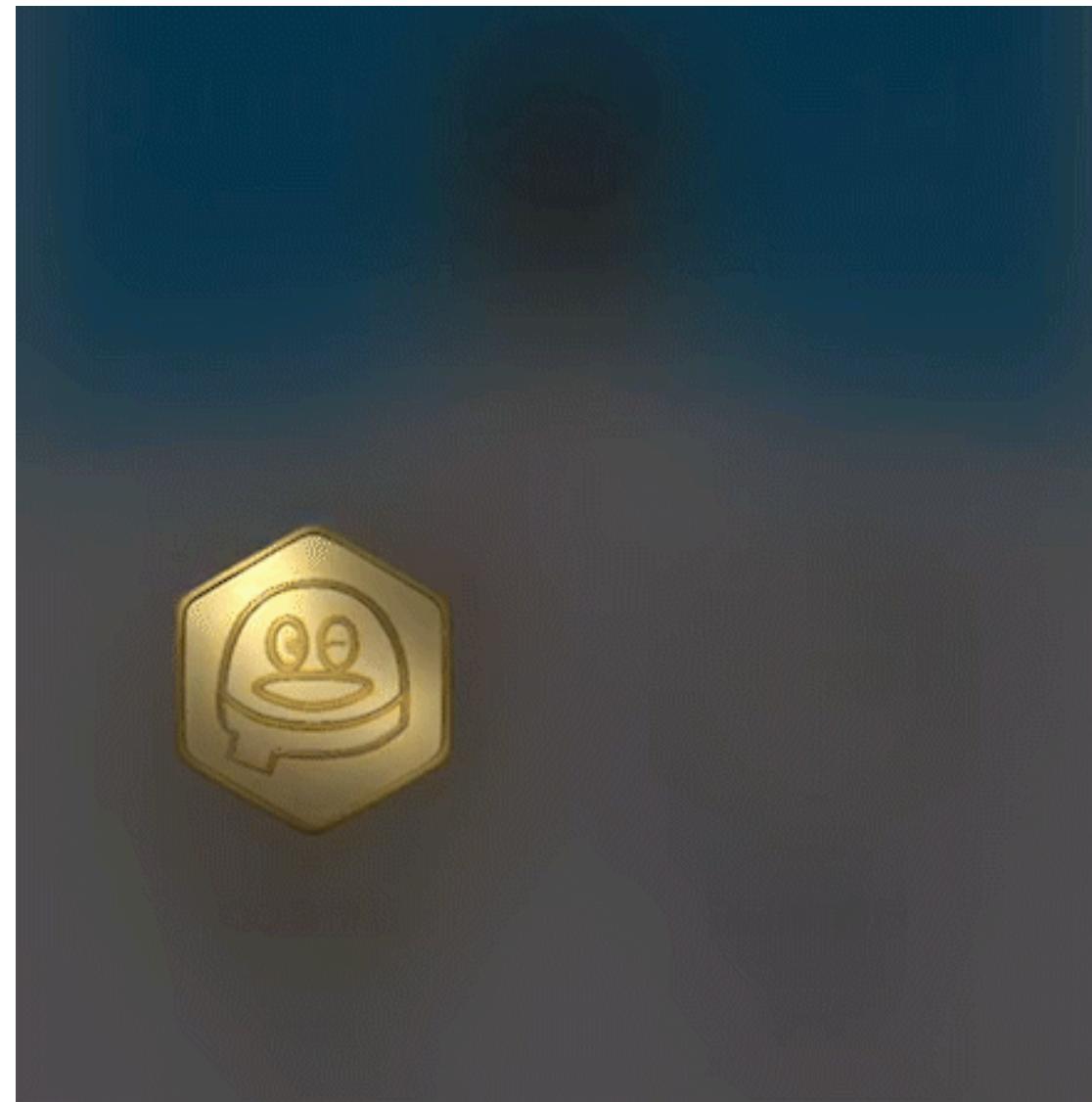
实现总结

- 效果简单，不使用物理引擎
- 结合生活中的现象，总结规律
- 简化规律



# 难点二：制作流畅的可交互动画

## 勋章墙要达到的效果



出场：匀速旋转

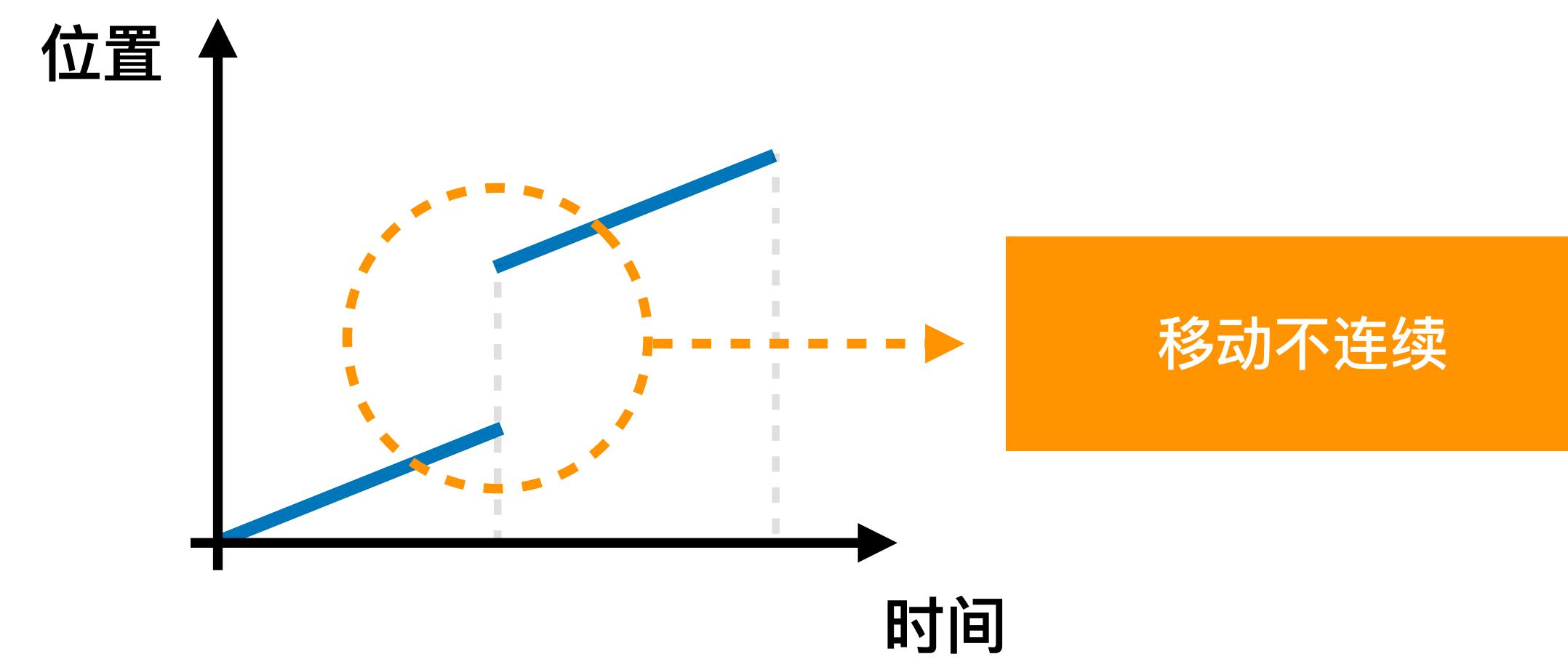


结束：弹性动画

如何让两个动画无缝衔接？

# 难点二：制作流畅的可交互动画

## 什么是流畅？



# 难点二：制作流畅的可交互动画

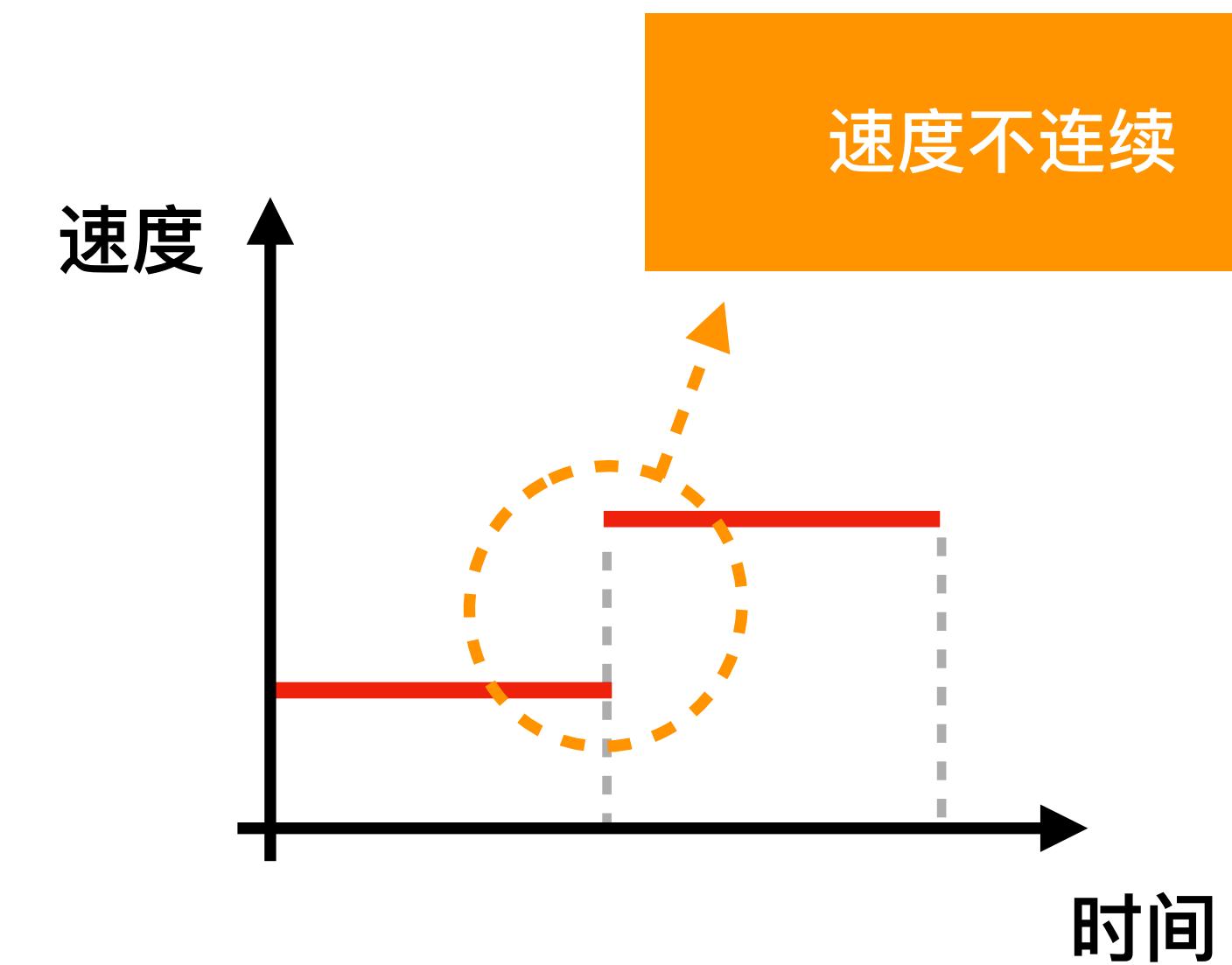
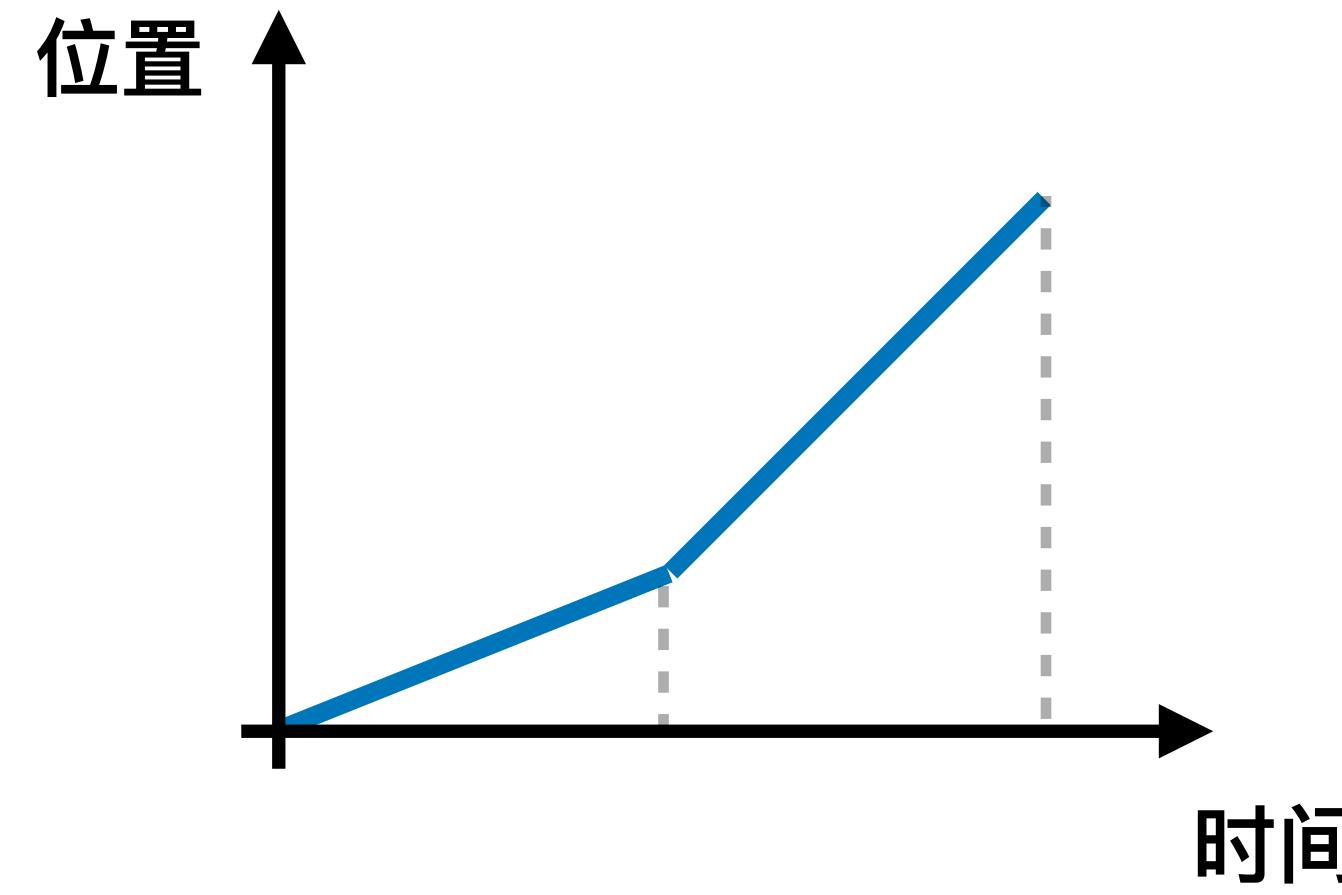
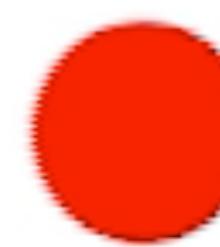
## 什么是流畅？

- 位移、角度变化连续

这样就流畅了吗？

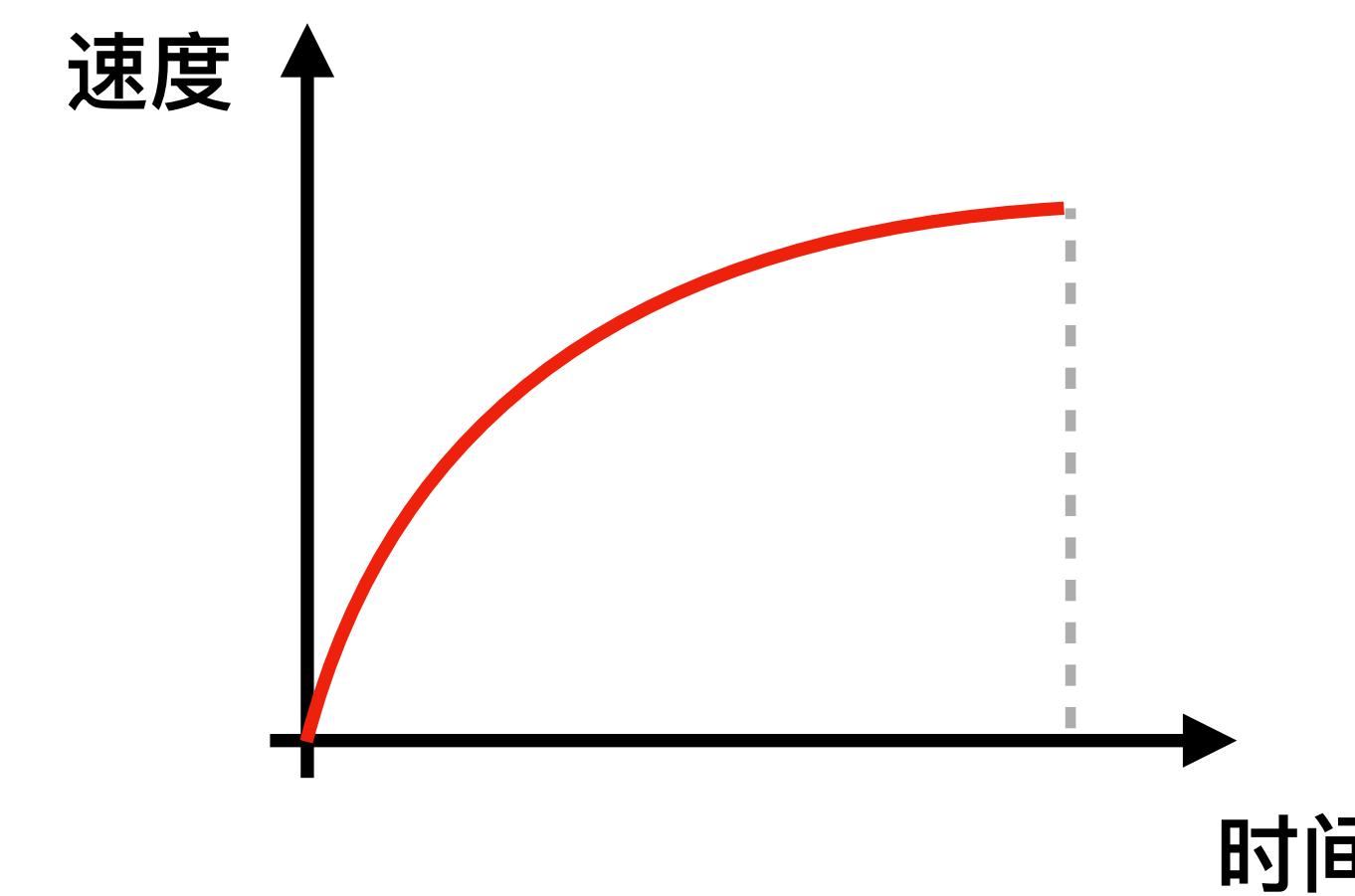
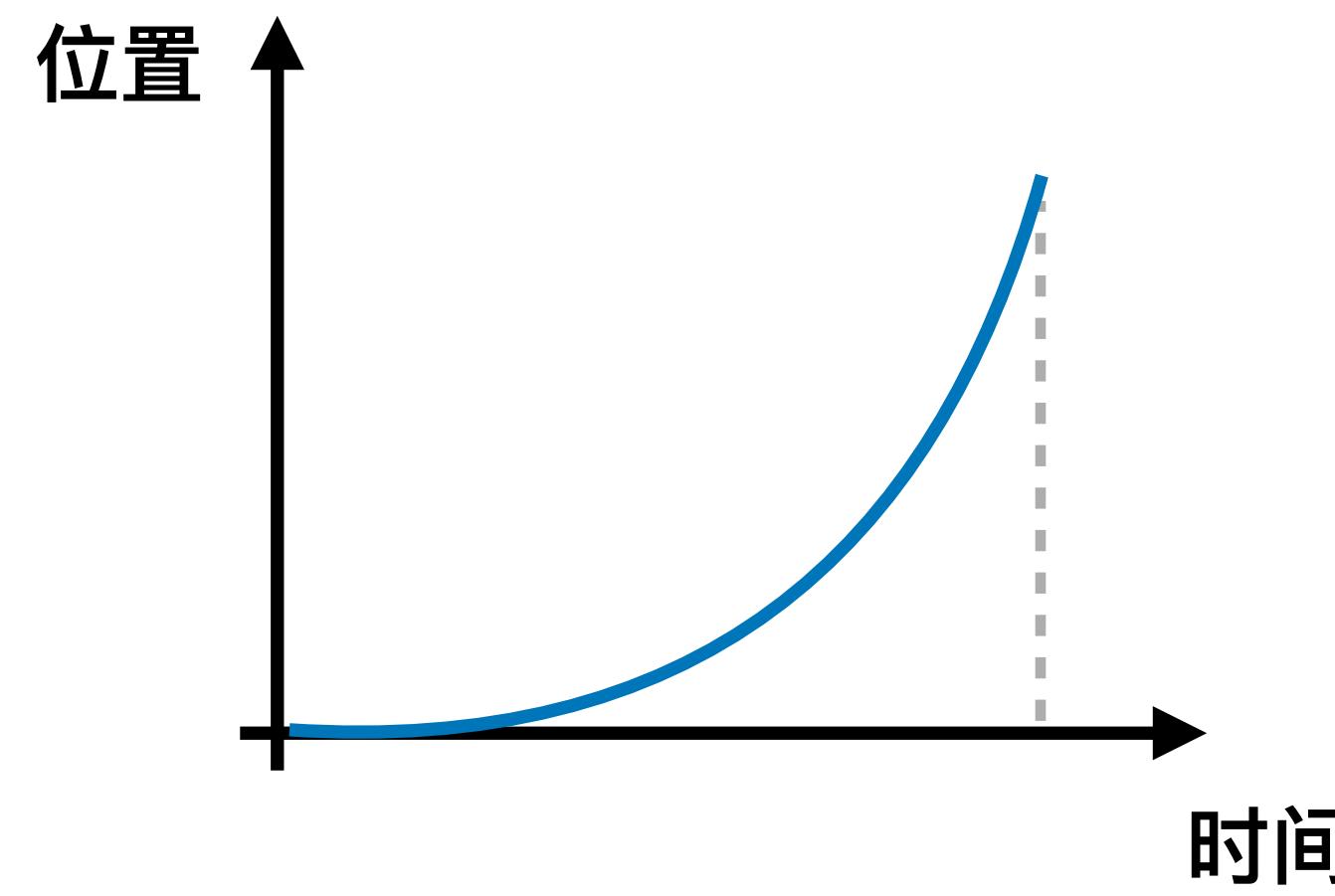
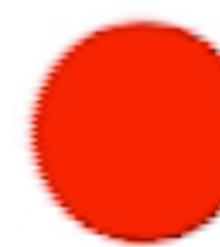
# 难点二：制作流畅的可交互动画

## 什么是流畅？



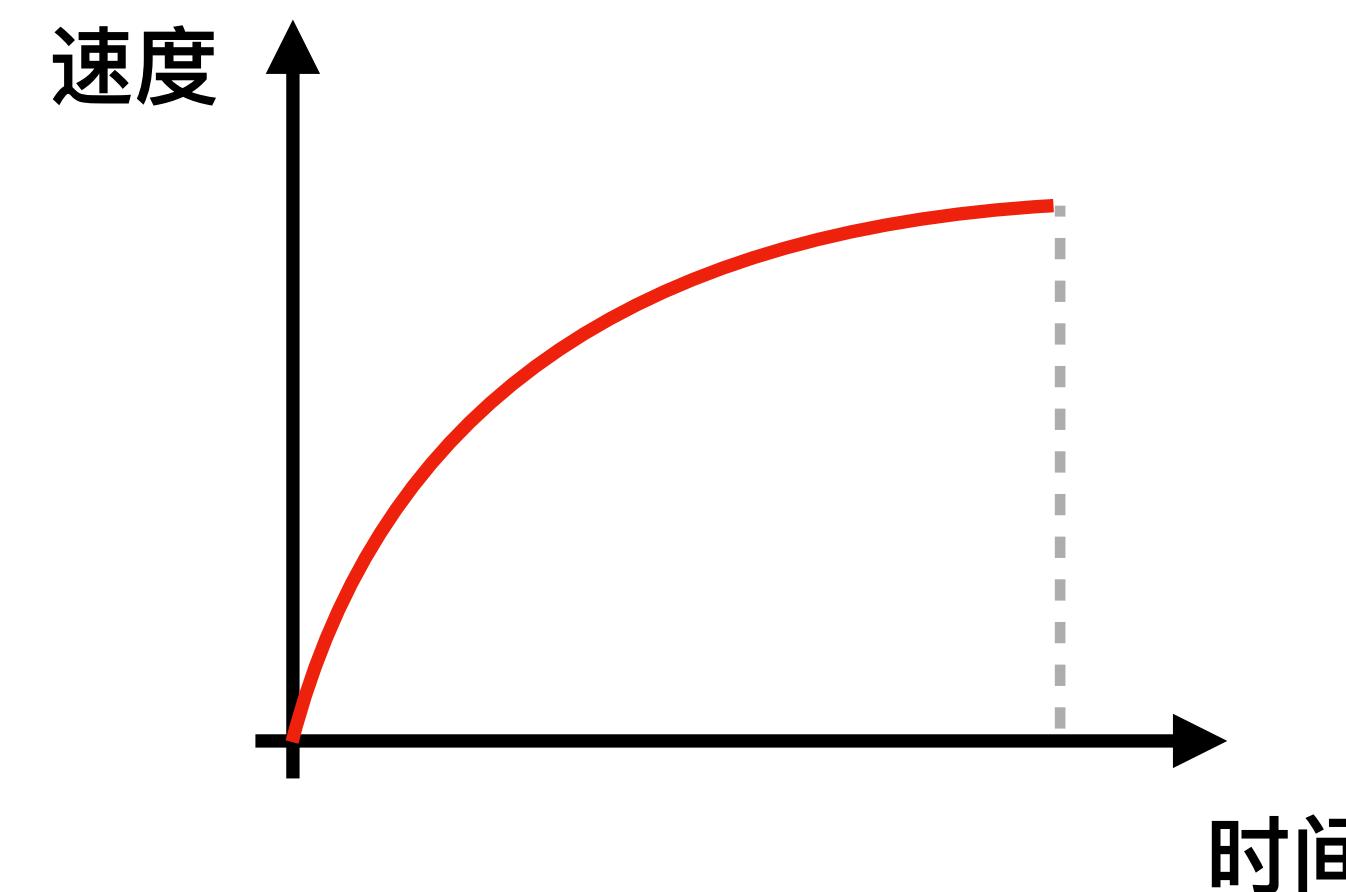
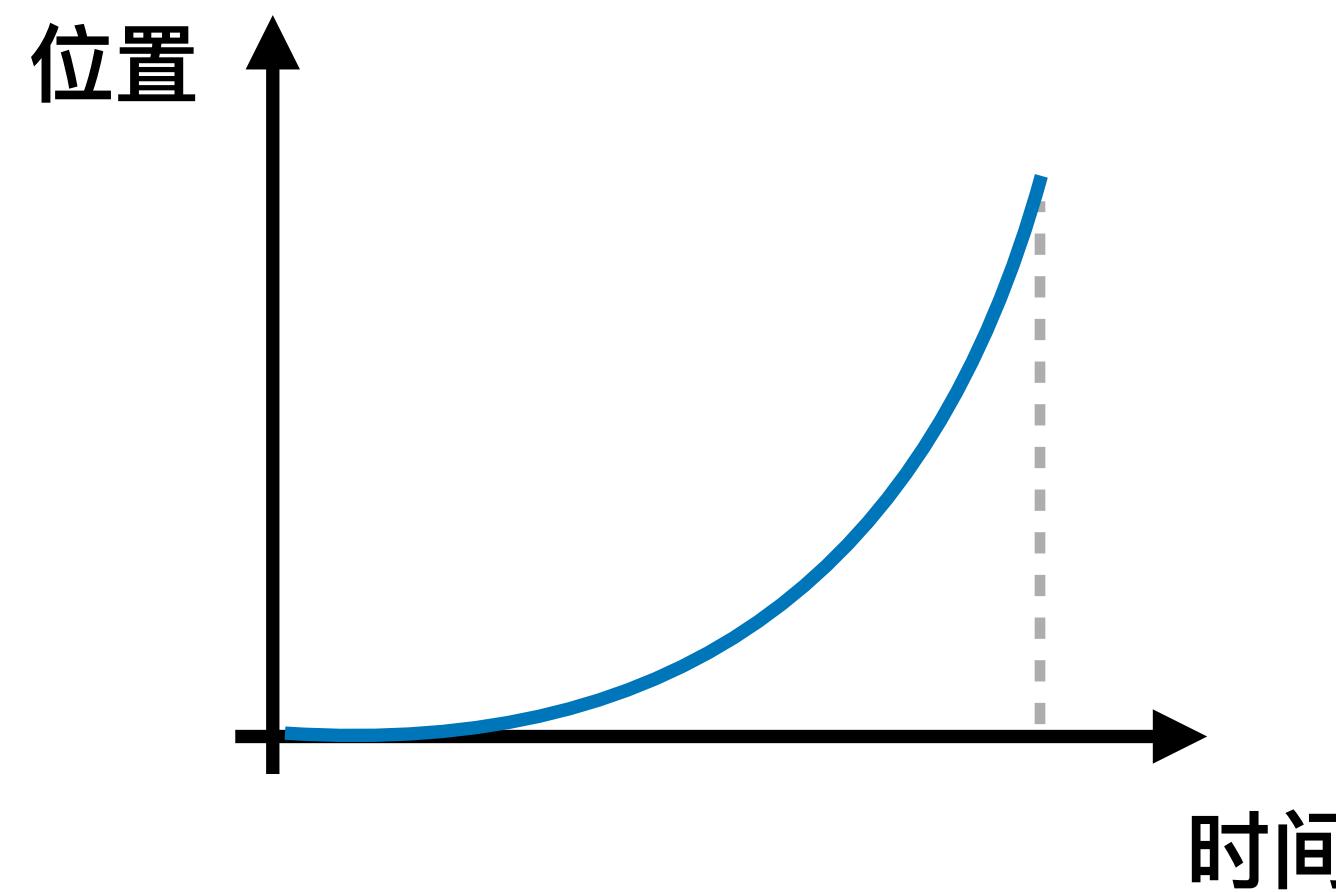
# 难点二：制作流畅的可交互动画

## 什么是流畅？



# 难点二：制作流畅的可交互动画

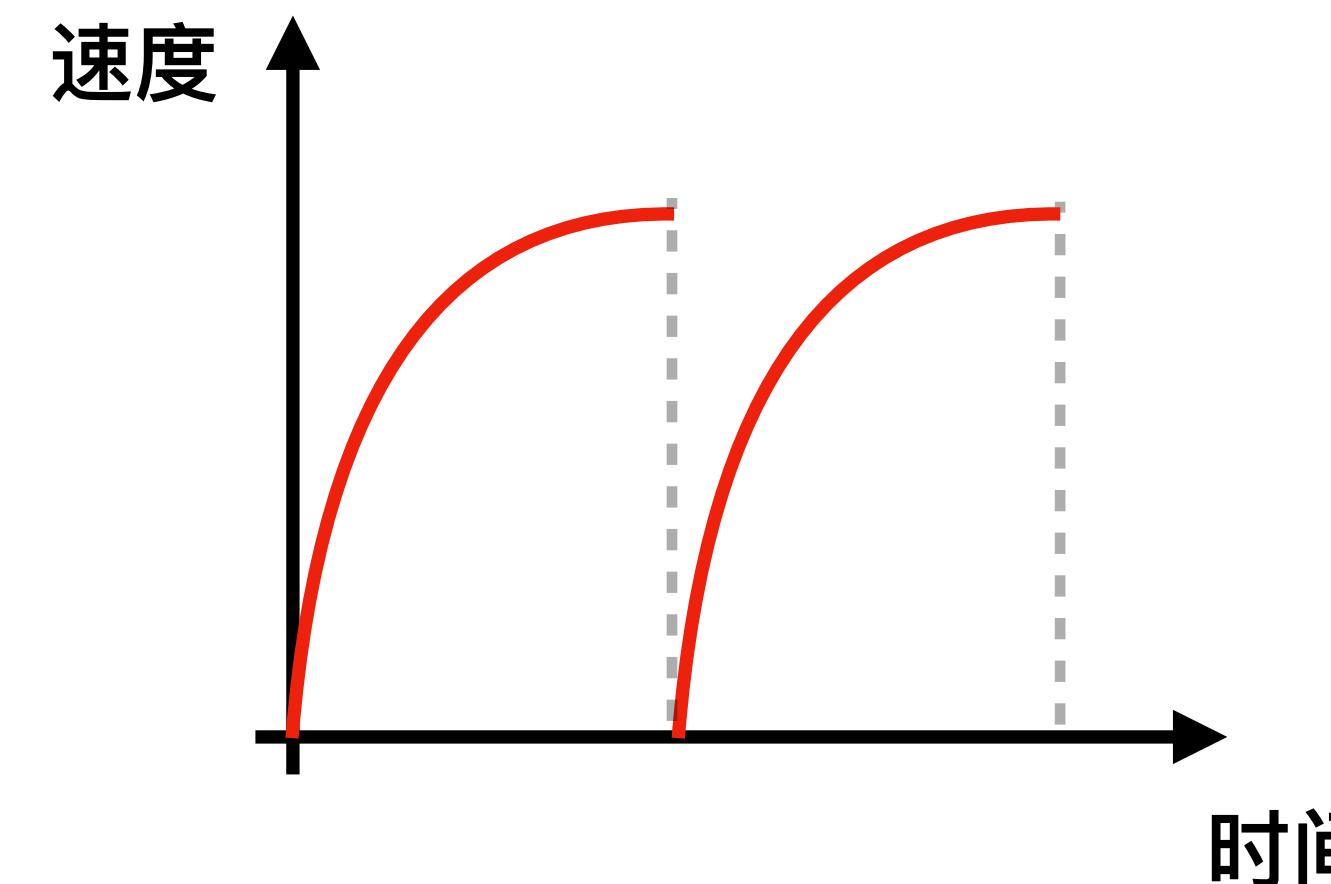
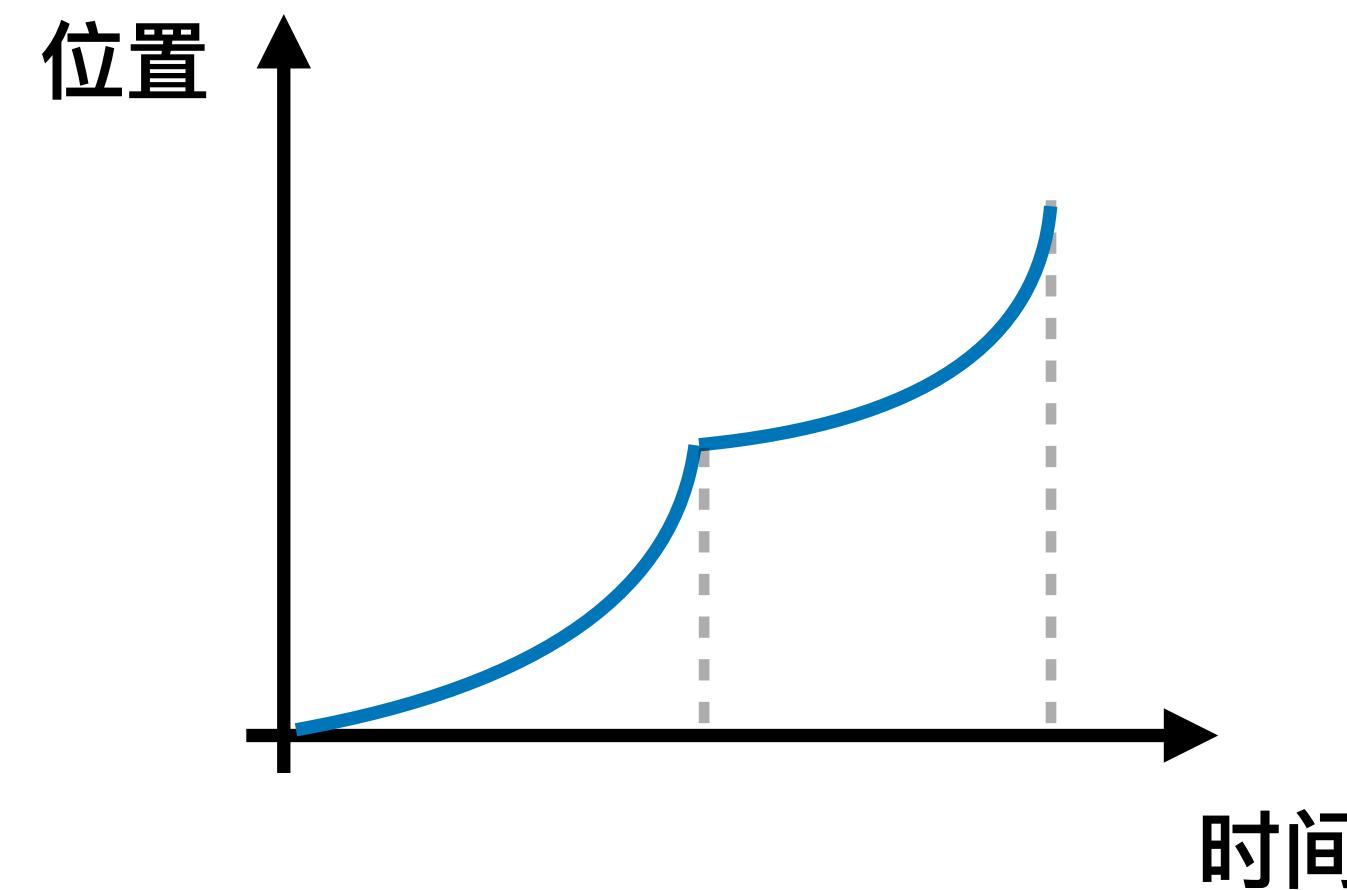
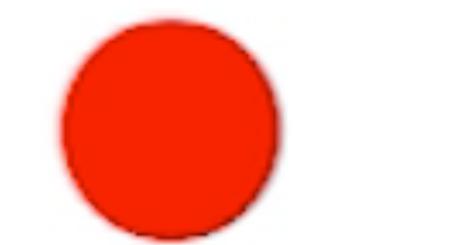
## 什么是流畅？



把速度变成曲线，就能看起来更舒服？

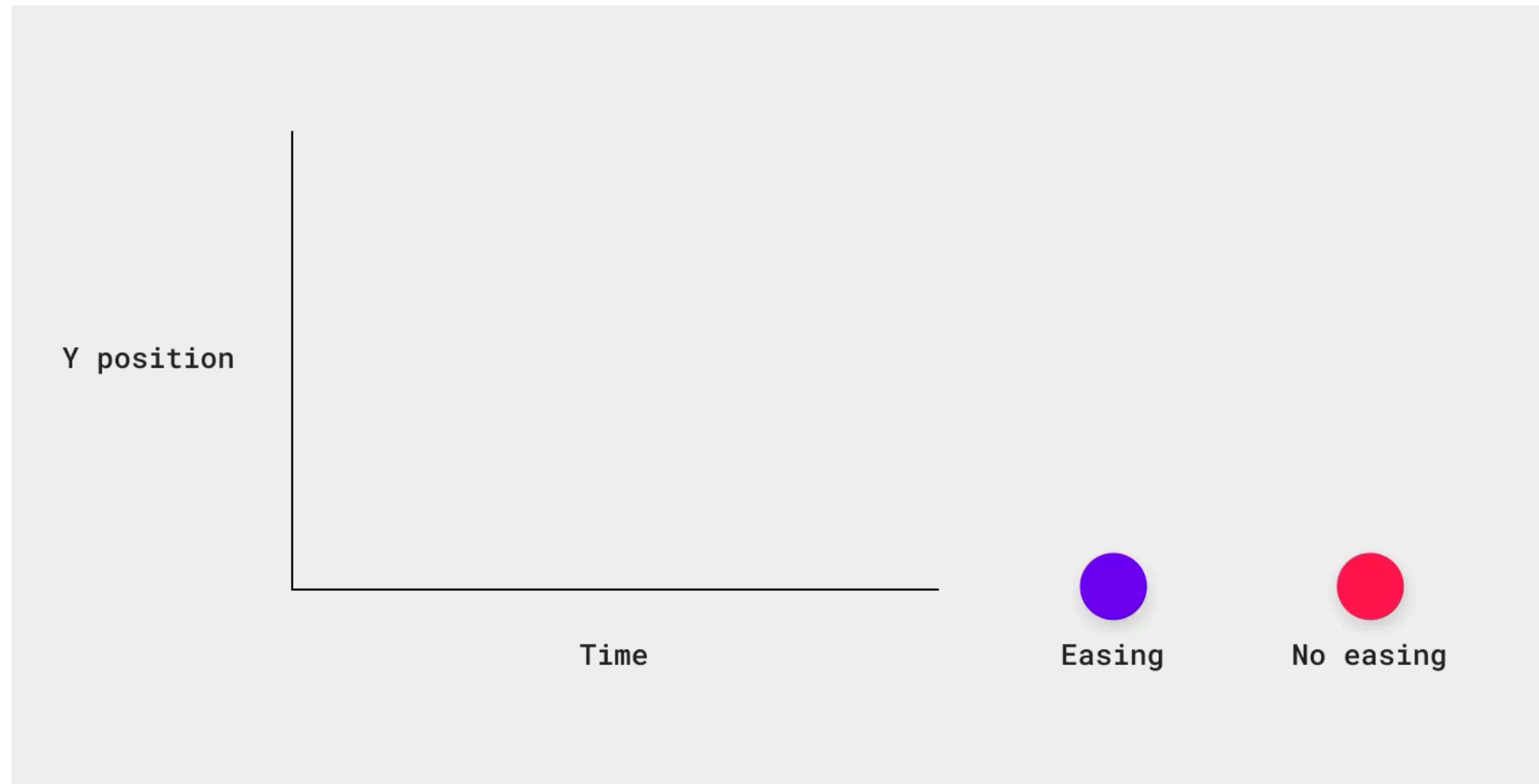
# 难点二：制作流畅的可交互动画

## 什么是流畅？

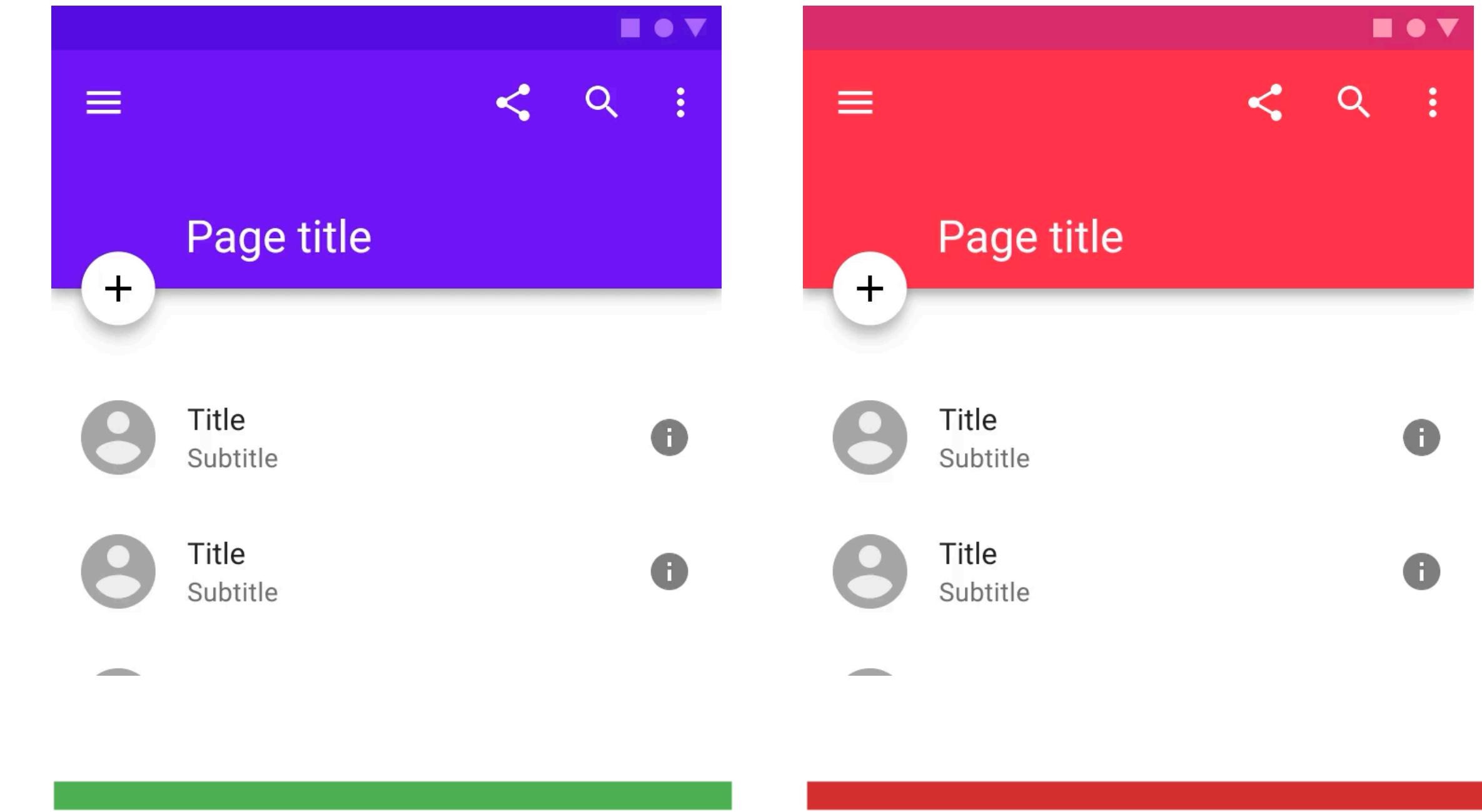


是不是曲线，重要的是速度变化要连续

# Material Design 给开发者的建议



建议1：使用缓动效果

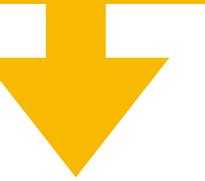


建议2：缓动时要用正确的曲线

# 难点二：制作流畅的可交互动画

## 什么是流畅？

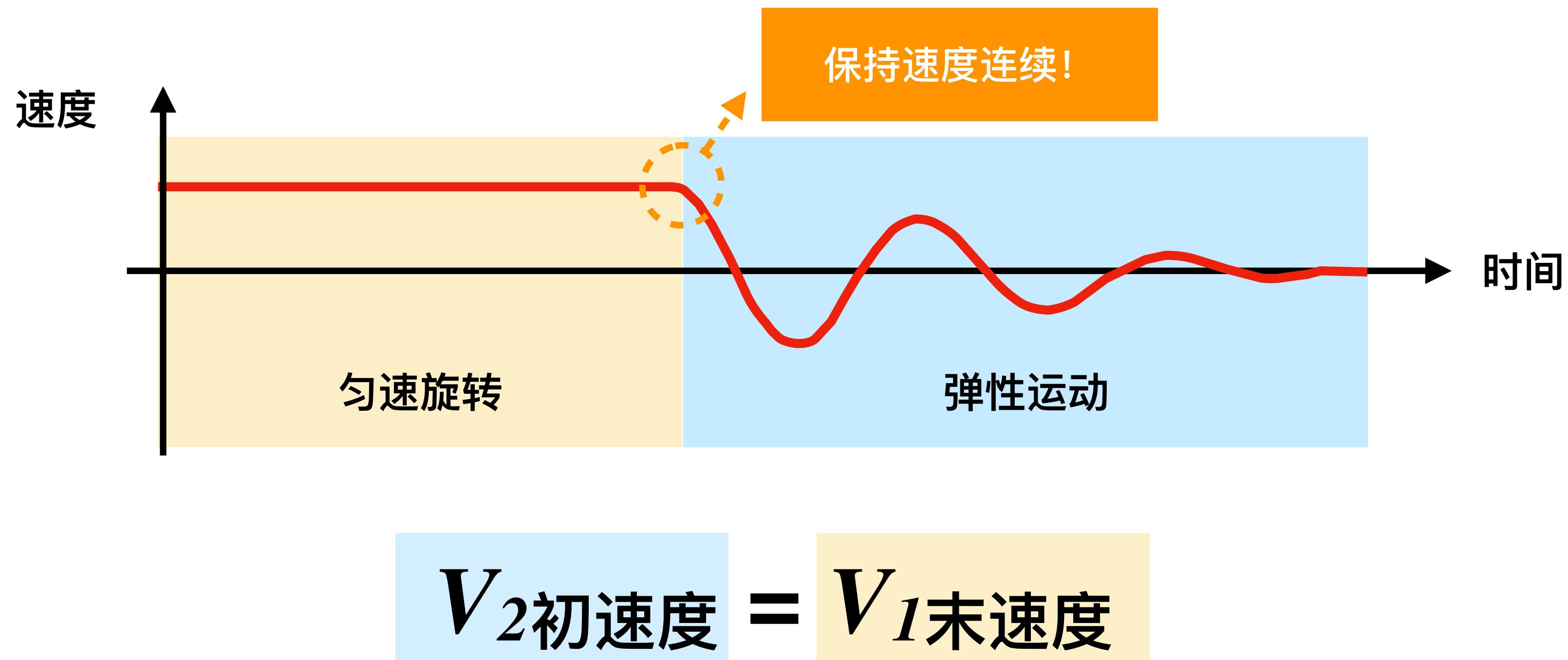
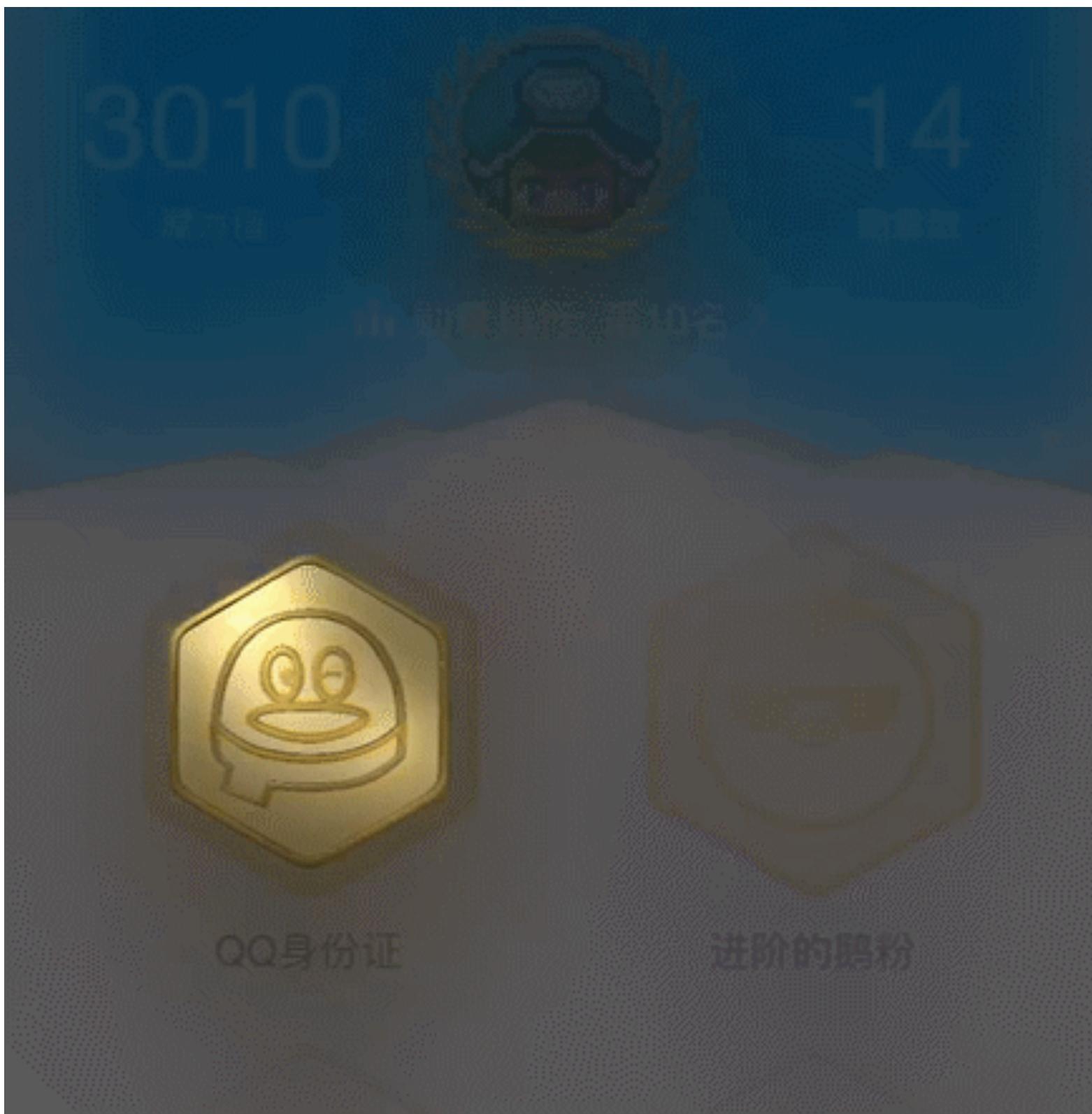
- 位移、角度变化连续
- 速度、角速度变化连续



连续且可导

# 难点二：制作流畅的可交互动画

## 如何衔接出场的旋转动画？



# 难点二：制作流畅的可交互动画

## 流畅衔接+可交互的秘密

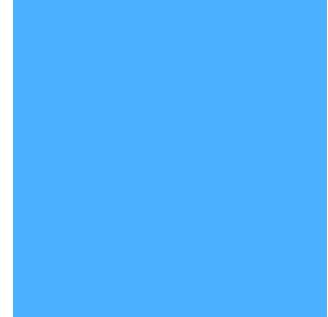
$$F = -k \cdot x$$

可交互

$$V_2\text{初速度} = V_1\text{末速度}$$

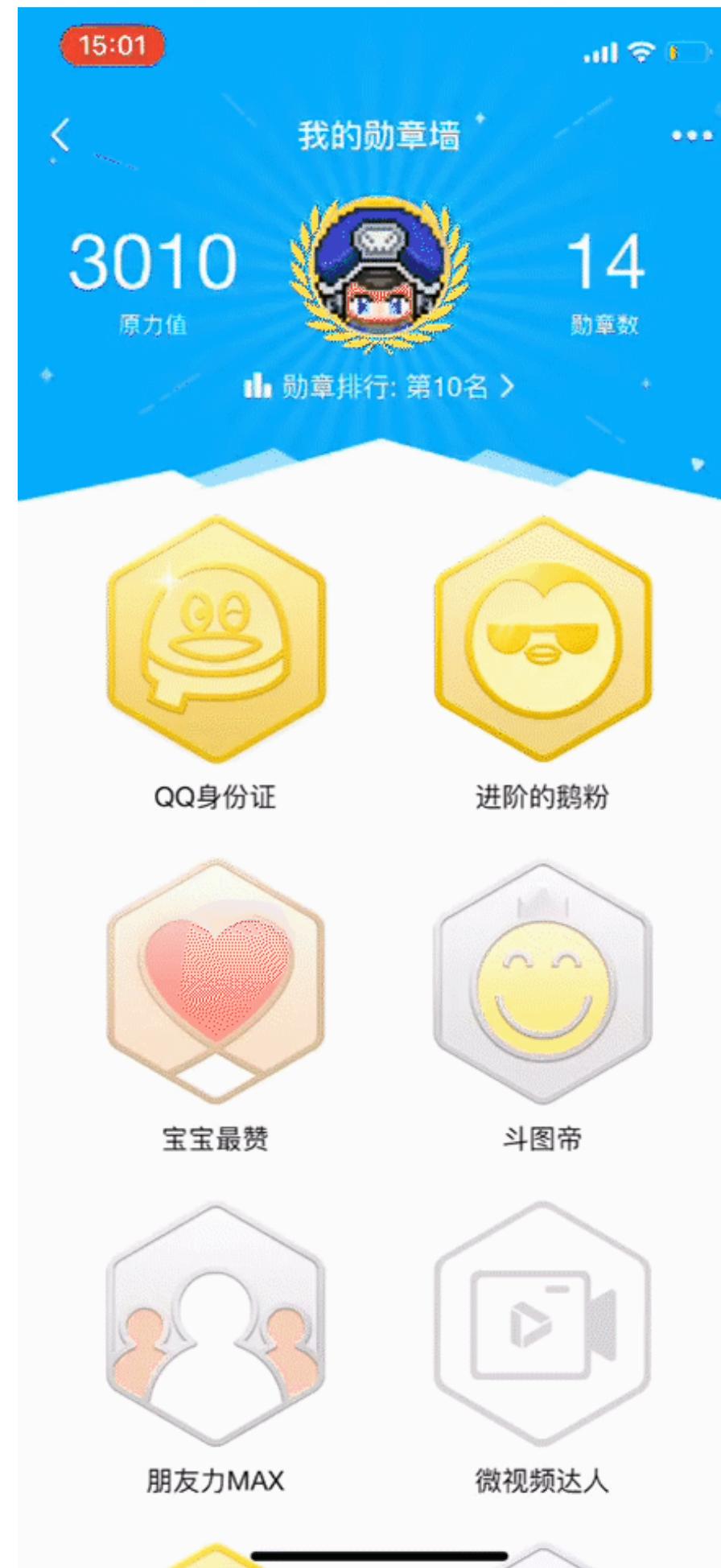
流畅衔接

符合物理规律，就能做出好动画！



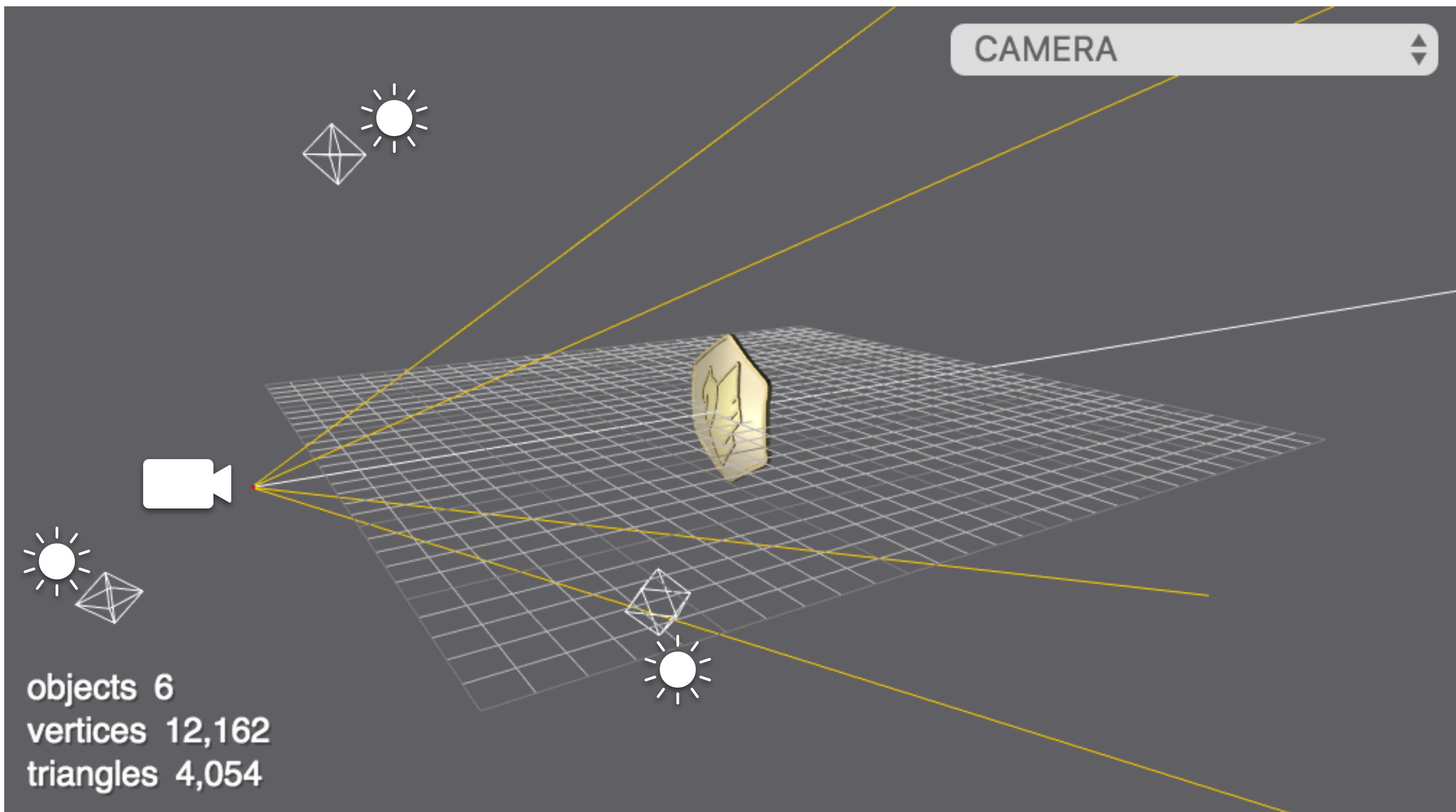
## **难点三：打破次元壁**

# 入场动画要达到的效果



- 从 2D 过渡到 3D
  - 3D 勋章要在盖在 2D 勋章上
  - 过渡过程中，模型需要缩放和位移

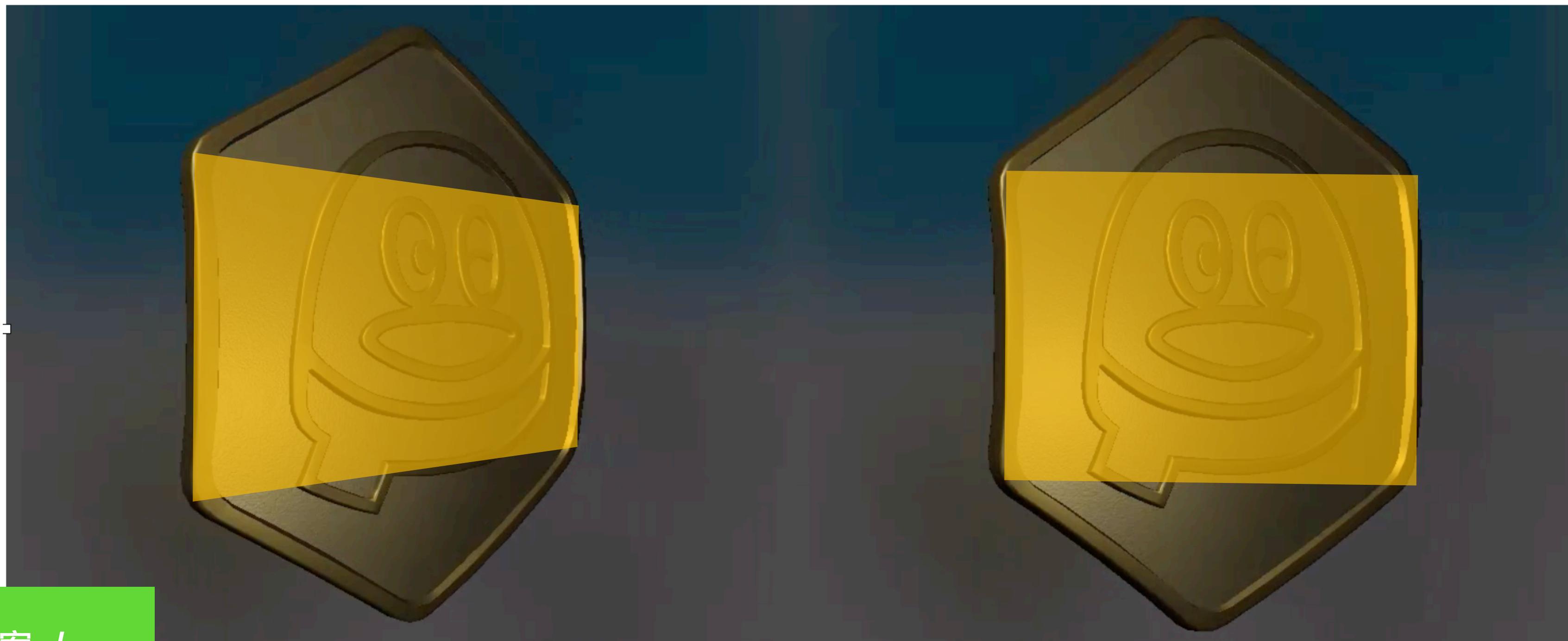
技术要点：打破次元壁



# 难点三：打破次元壁

2种透视方案分析

透视相机



最终方案 ✓

- ✗ 2D → 3D 坐标换算复杂
- ✗ 消失点? ? ?
- ✓ 旋转时透视变化正常

正交相机



- ✓ 坐标计算简单
- ✓ 不用考虑消失点
- ✗ 旋转时透视变化奇怪

# 难点三：打破次元壁

## 透视投影怎么让 3D 模型出现在 2D 图片的位置？

### Vector3

```
.unproject ( camera : Camera ) : this  
camera — camera to use in the projection.
```

- 使用 Three.js 内置函数可以将 2D 坐标映射到 3D 坐标系
- 坐标：  
调用 1 次，计算 3D 坐标
- 缩放比例：  
调用 2 次，传入 2D 图片最高点和最低点位置，可以量取 3D 模型的缩放比例

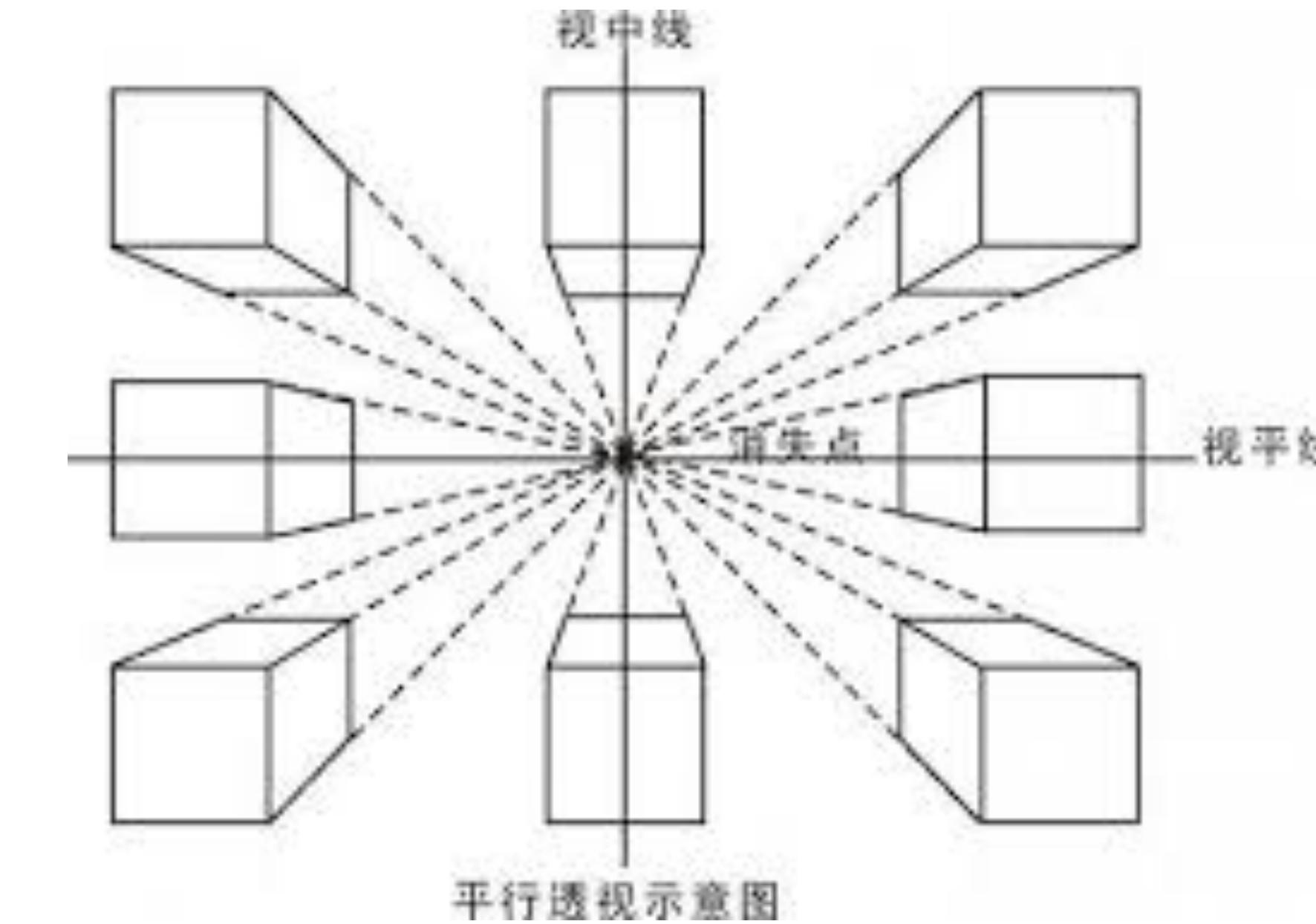
接下来，只要根据坐标和缩放比例调整模型就好了？

# 难点三：打破次元壁

## 缩放模型带来的问题1：消失点不正确



总觉得哪里怪怪的



透视相机默认的消失点是画面中间！

# 难点三：打破次元壁

## 缩放模型带来的问题2：渲染效果差距很大



实际渲染效果



设计师效果

原因：缩放后深度贴图深度、灯光位置、灯光强度等若干参数没有相应成倍

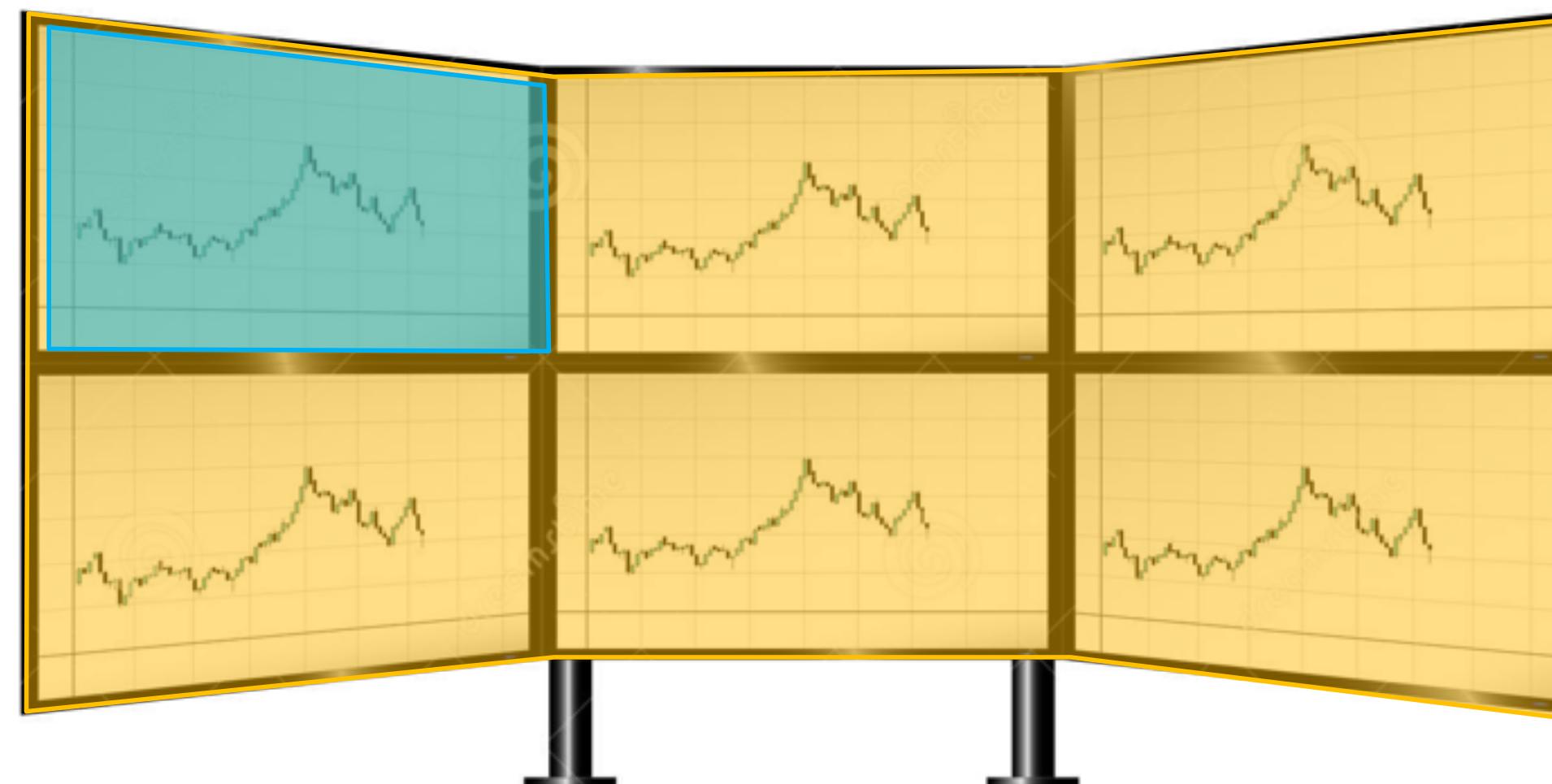
通过调整模型不行的话，也许我们应该把思路转变一下，调整摄像机

### 难点三：打破次元壁

那么，有没有一种既能移动消失点，又能自动调整各种参数的方法呢？

## PerspectiveCamera

`.setViewOffset ( fullWidth : Float, fullHeight : Float, x : Float, y : Float, width :  
Float, height : Float ) : null`



**subCamView**

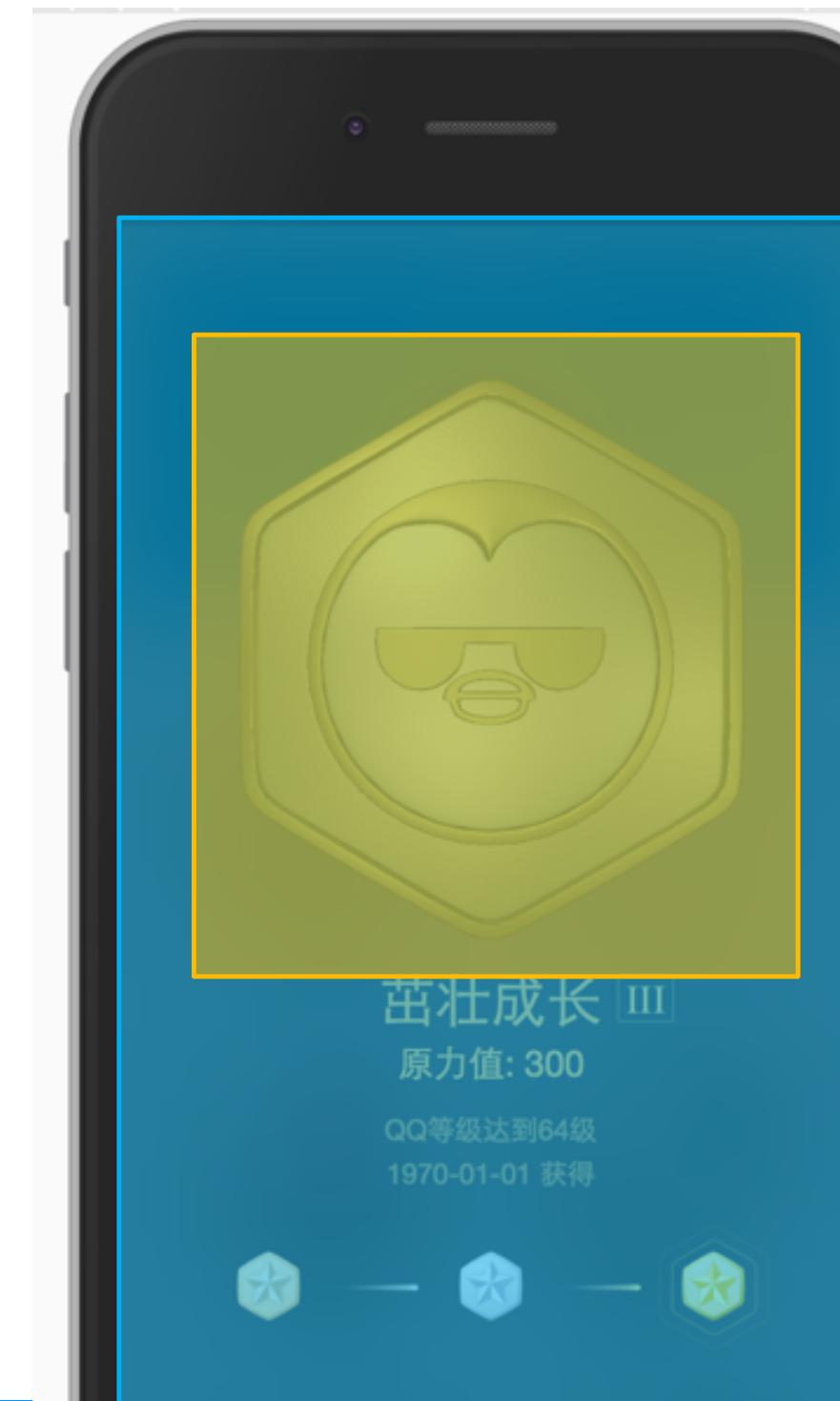
**fullView**

### 难点三：打破次元壁

那么，有没有一种既能移动消失点，又能自动调整各种参数的方法呢？

## PerspectiveCamera

`.setViewOffset ( fullWidth : Float, fullHeight : Float, x : Float, y : Float, width :  
Float, height : Float ) : null`



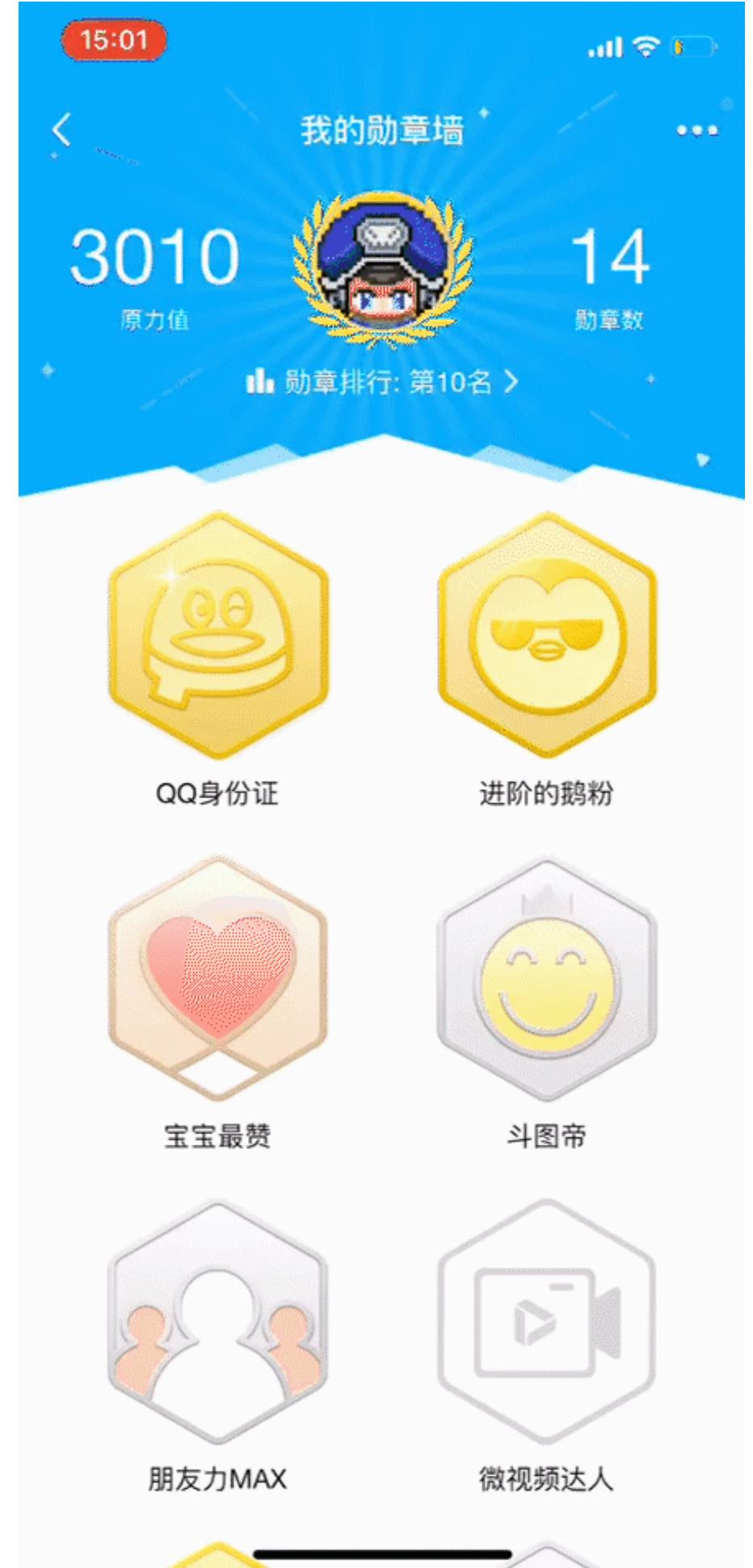
**subCamView**

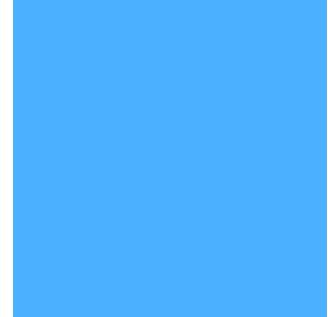
**fullView**

# 难点三：打破次元壁

## 方案总结

- 使用透视相机，达到真实的透视效果
- 通过 .unproject 计算模型初末状态坐标、缩放比例
- 通过 .setViewOffset 逐帧改变摄像机偏移



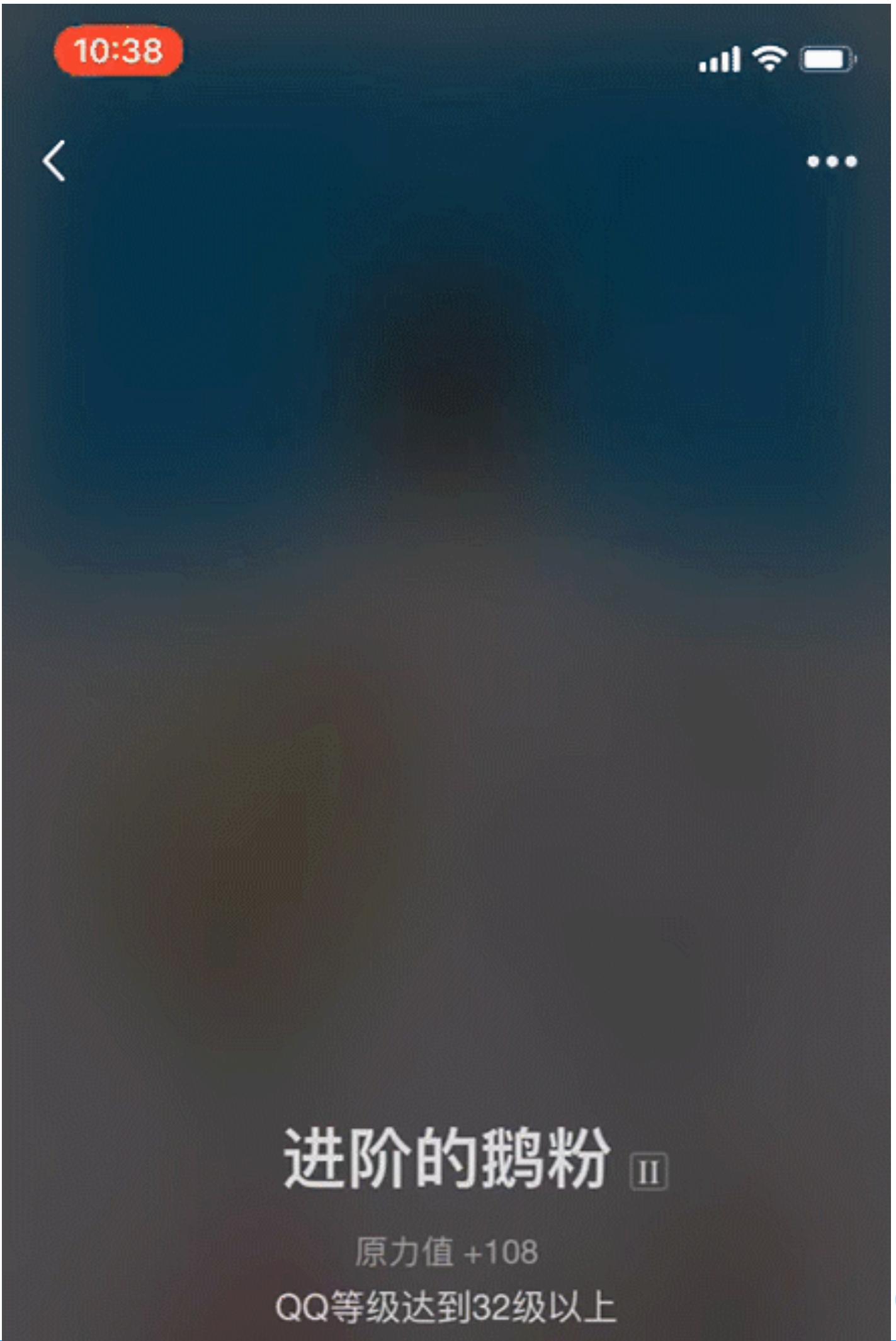


## **难点四：渲染半透明模型**

# 勋章墙预期的效果

- 切换不同等级的勋章时，勋章需要渐渐出现

只是调整透明度，有什么难度呢？



# 难点四：渲染半透明模型 有什么难点呢？

- 直接设置 canvas 透明度?  
✖ 低端安卓不兼容
- 直接设置材质透明度?  
✖ 相当于设置模型每个面的透明度，会出现重影



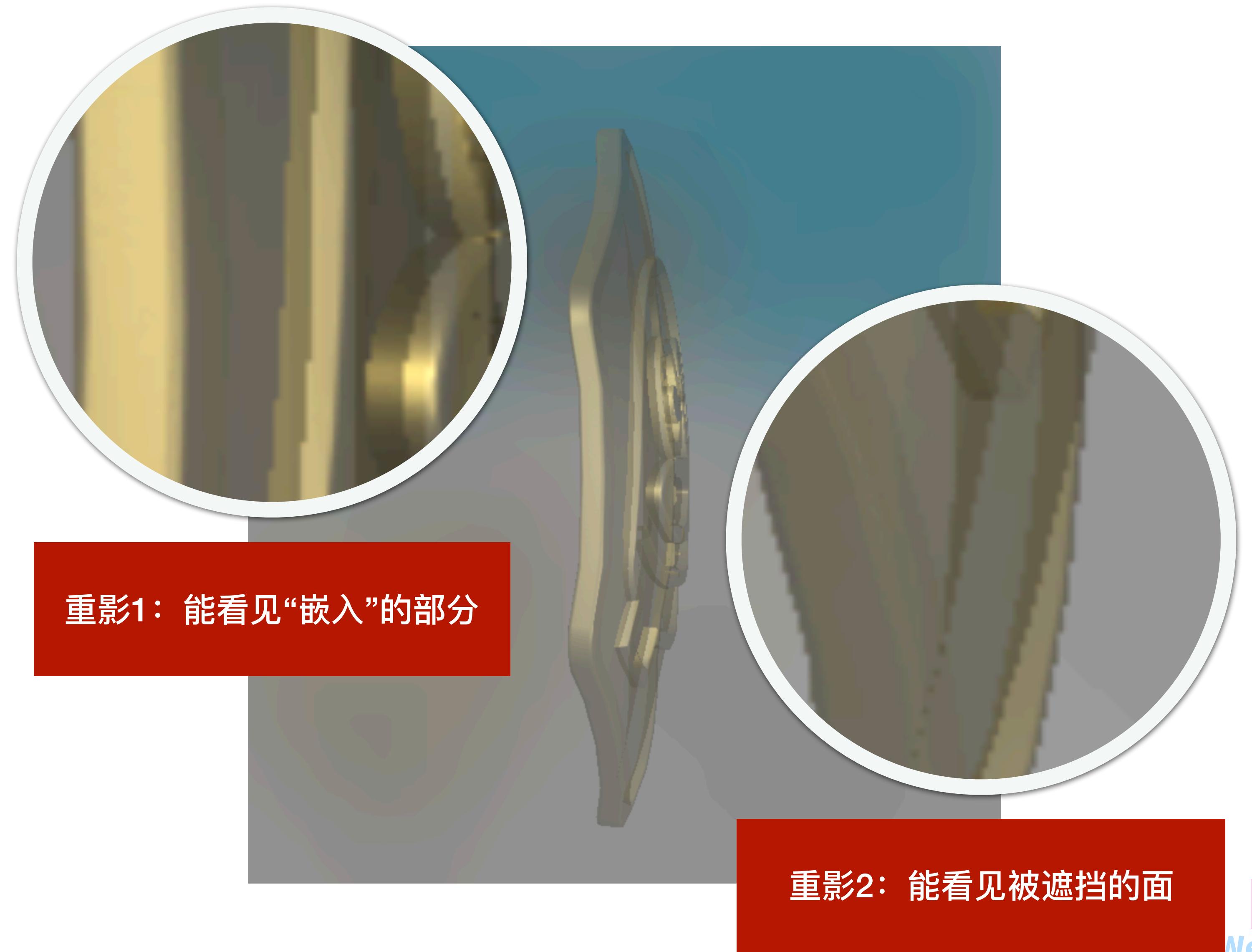
脑补效果



真实效果：飞机翅膀、引擎重影了！

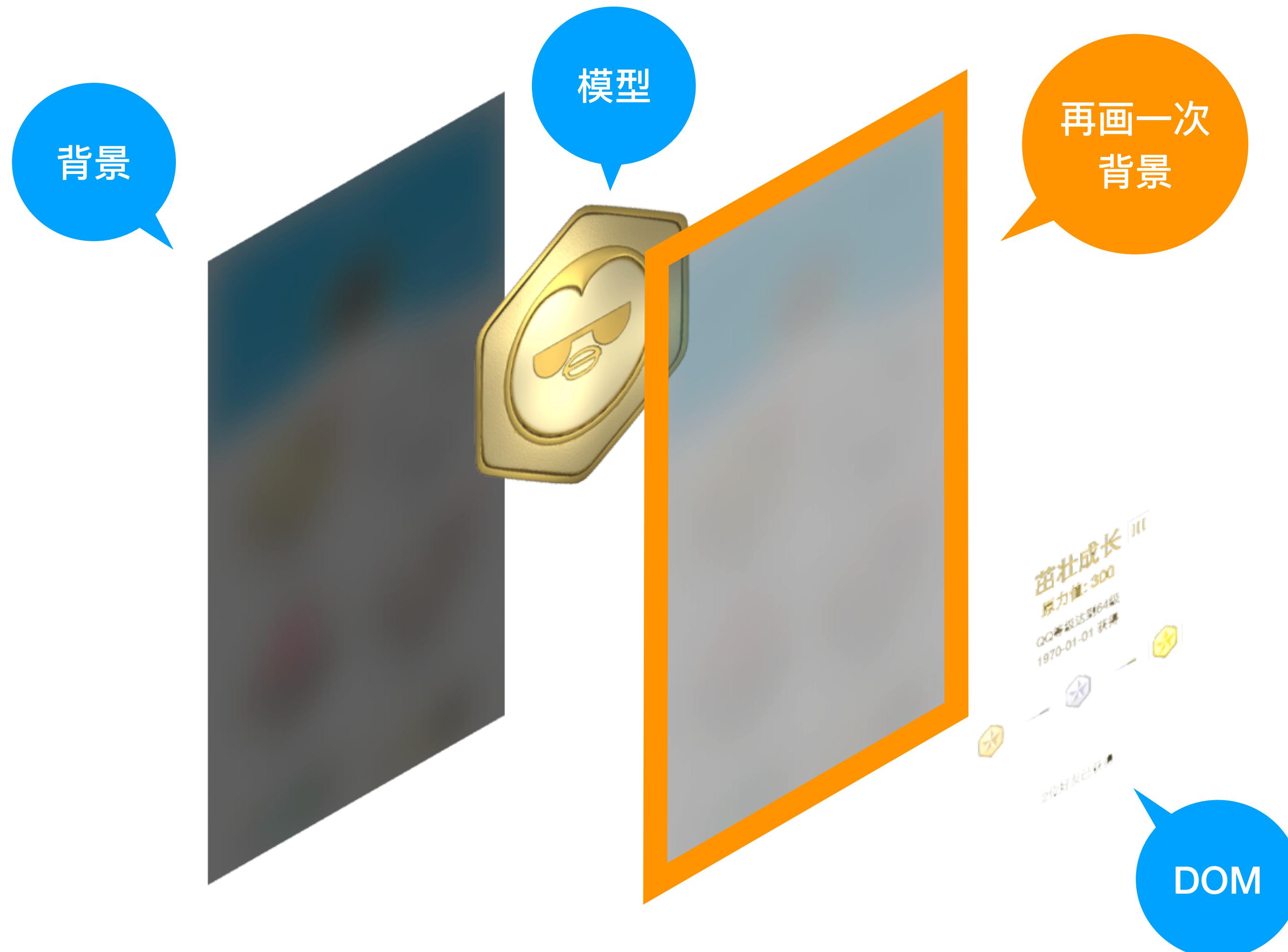
# 难点四：渲染半透明模型 有什么难点呢？

- 直接设置 canvas 透明度?  
✗ 低端安卓不兼容
- 直接设置材质透明度?  
✗ 相当于设置模型每个面的透明度，会出现重影



# 难点四：渲染半透明模型

## 方案二：在画好的模型前再画一层“前景”

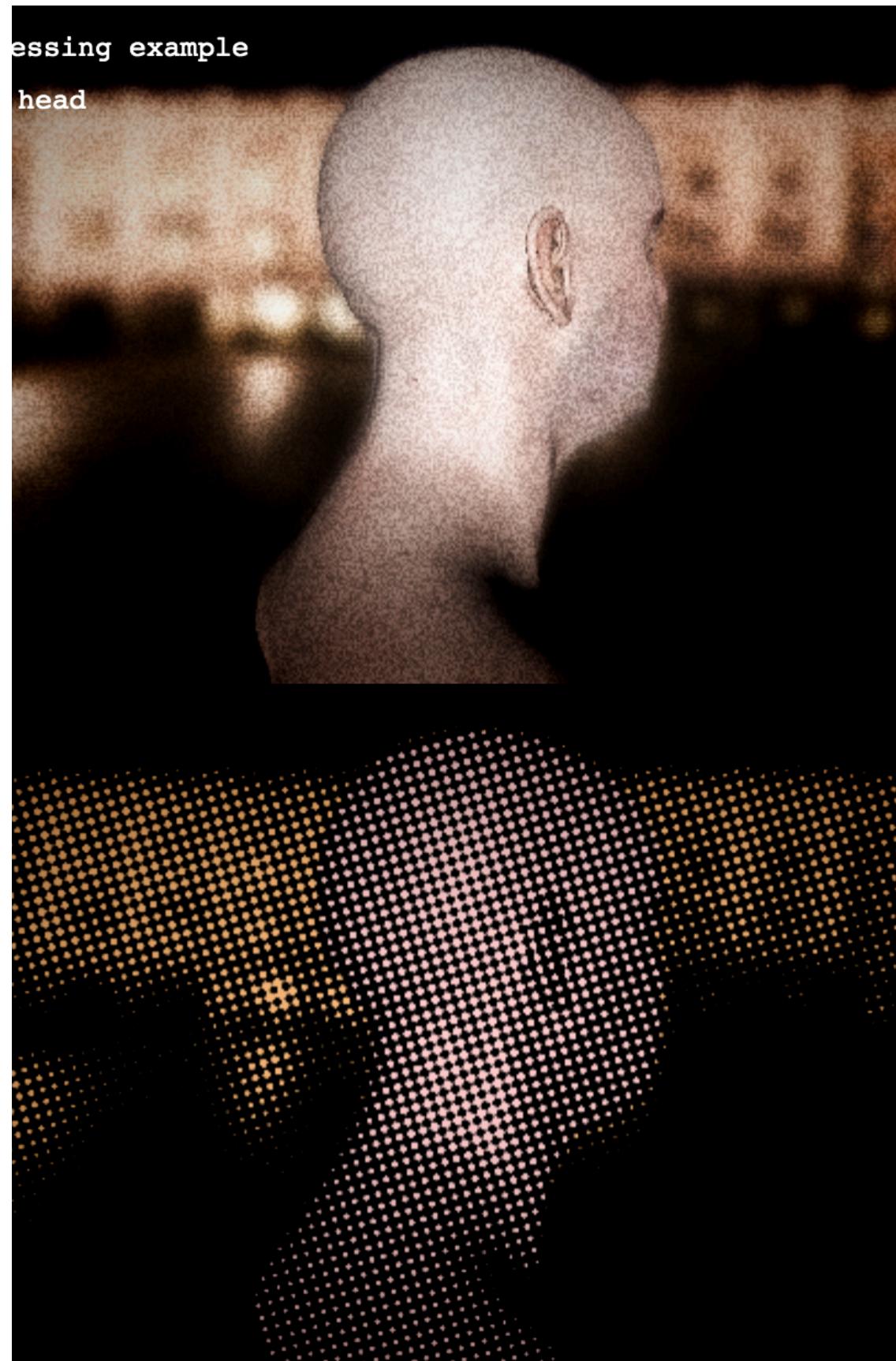


只要改变前景图层的透明度，  
模型看起来就像透明了一样

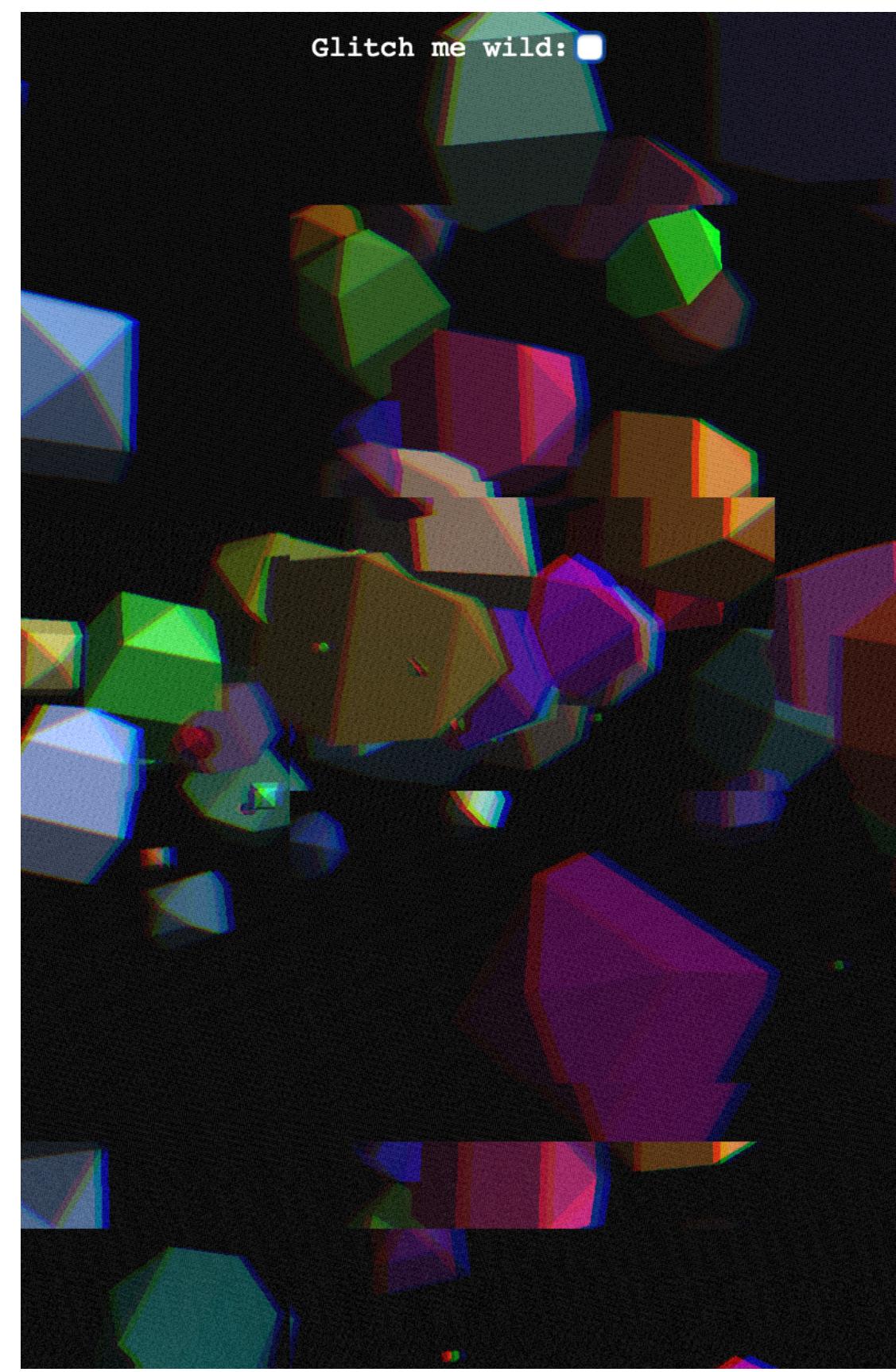
基于 post processing  
就很简单了

# Post Processing 是什么?

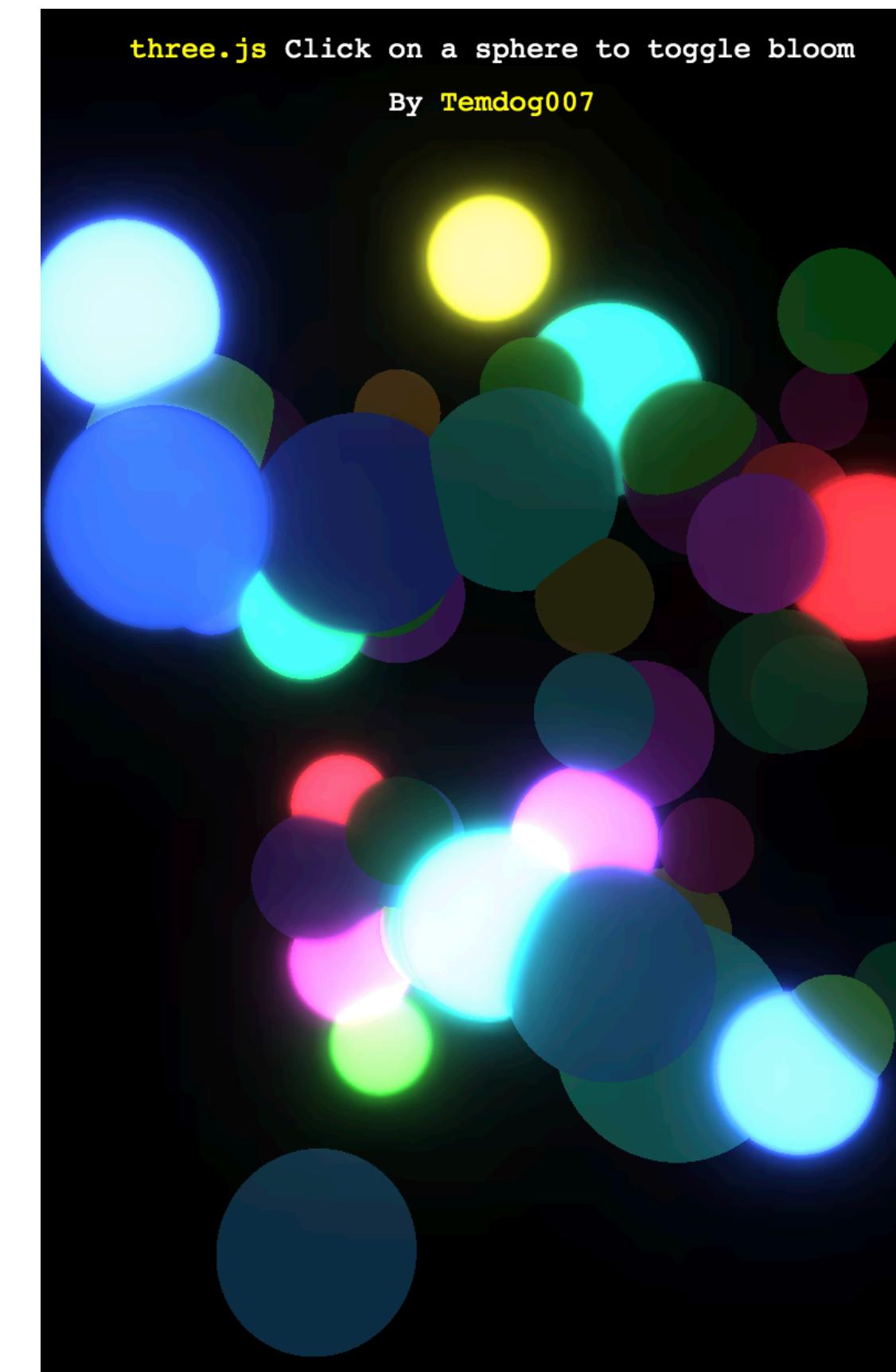
对渲染后的画面做进一步调整



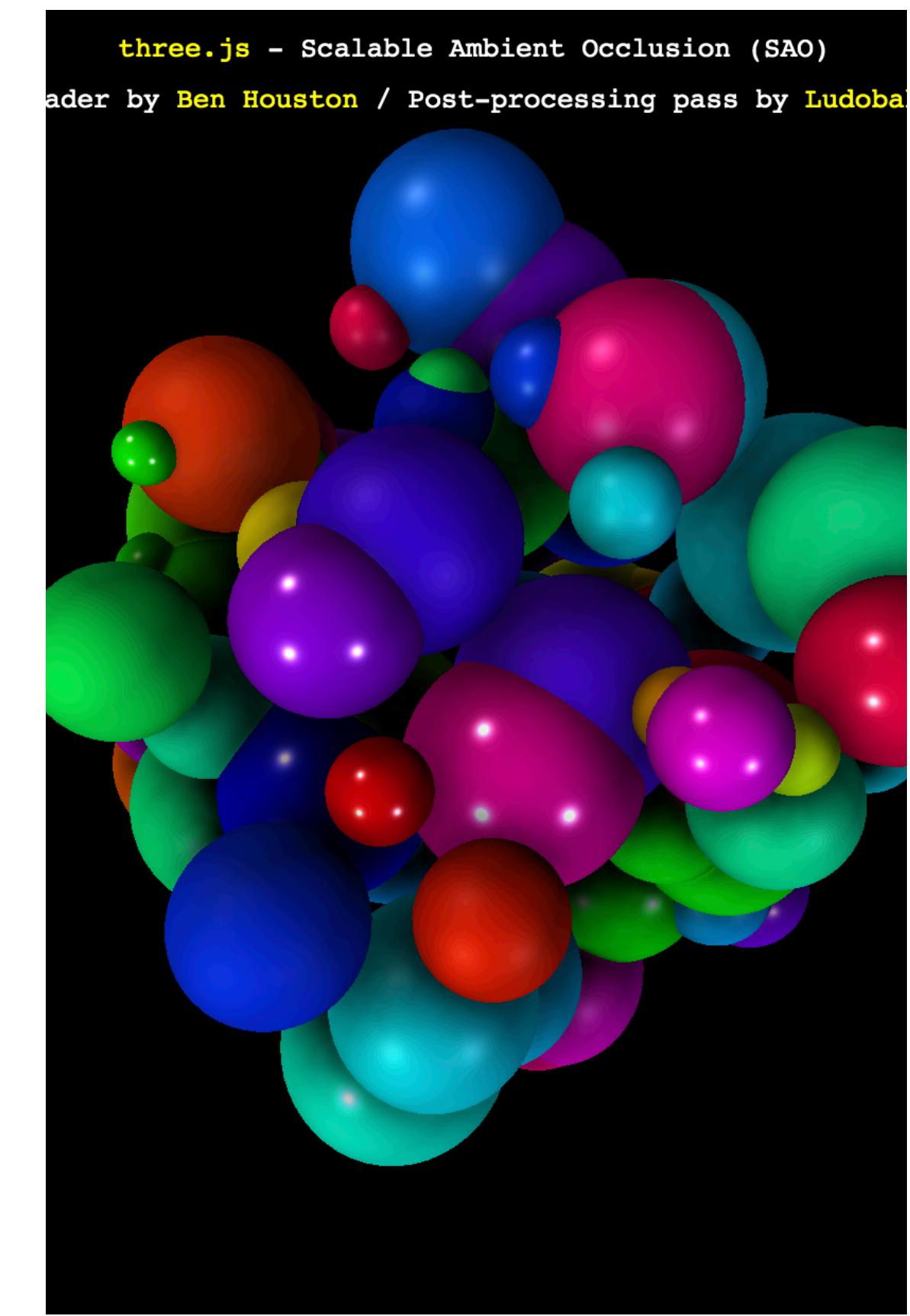
印刷效果滤镜



故障艺术(glitch art)滤镜



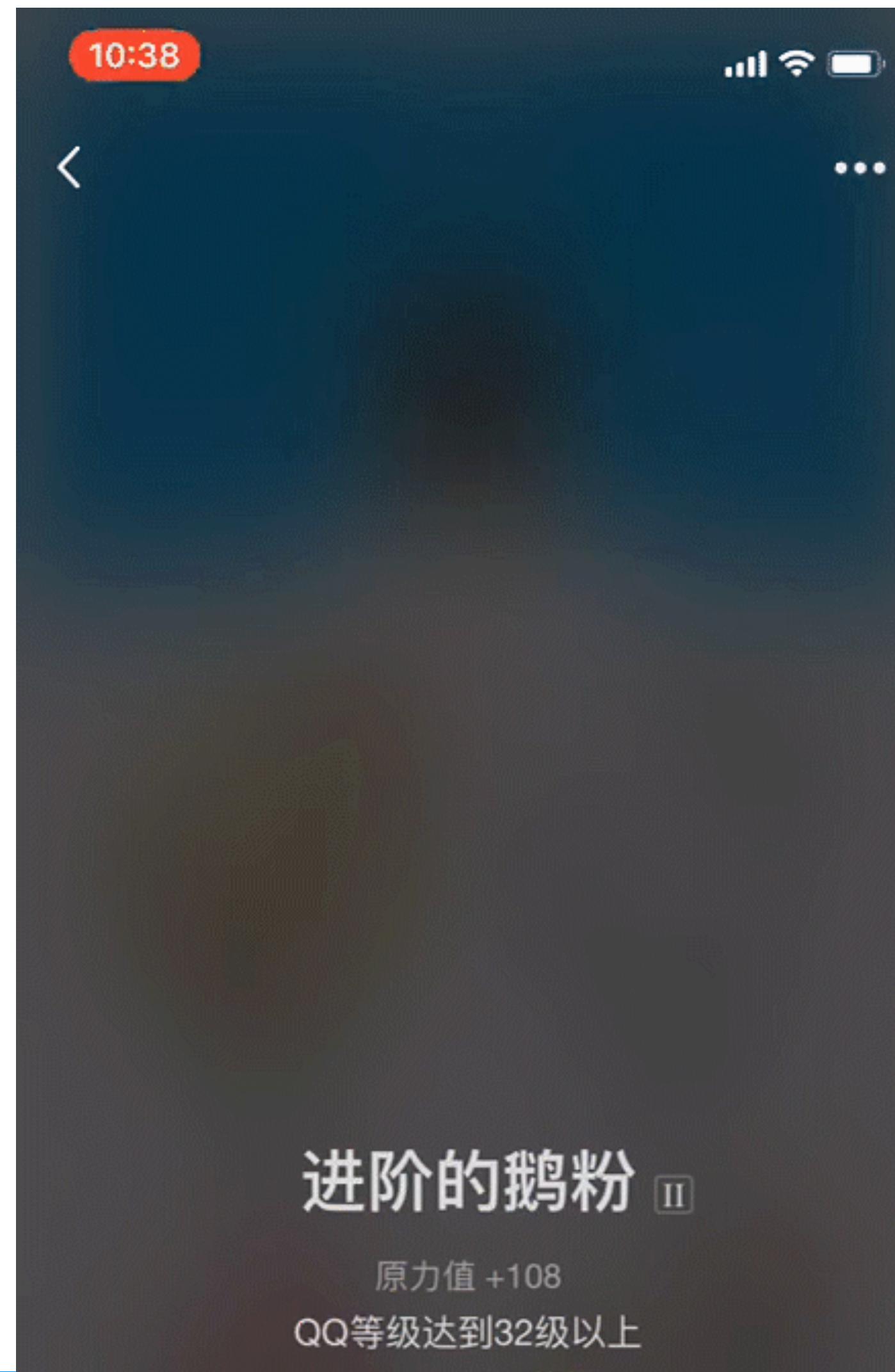
眩光效果



生成全局光照(AO)

<https://threejs.org/examples/?q=postprocessing>

# 方案总结



- 调整模型透明度会“穿帮”
- 用后处理器(post processing)渲染好的图像上再渲染一层背景图

# 其他优化

# 低端机优雅降级



正常效果

去除“磨砂质感”

降低画布尺寸

- 卡顿机器：定期采样 fps，去除深度贴图、降低画布尺寸
- 不支持WebGL的机器：直接展示2D图片

线上平均fps：45+

# 模型文件cache

- 基于 localStorage 缓存模型文本
- 资源 MD5 命名，不需要重复加载模型文件
- 静态资源使用QQ离线包

# 未来？

- 异形屏幕手机适配
- 使用更高效模型文件 gltf 压缩文件大小，加快渲染速度



# <Thanks/>

主办方  
Organizer

Tencent TWeb