

虚拟现实技术
如何绑架了你的大脑？





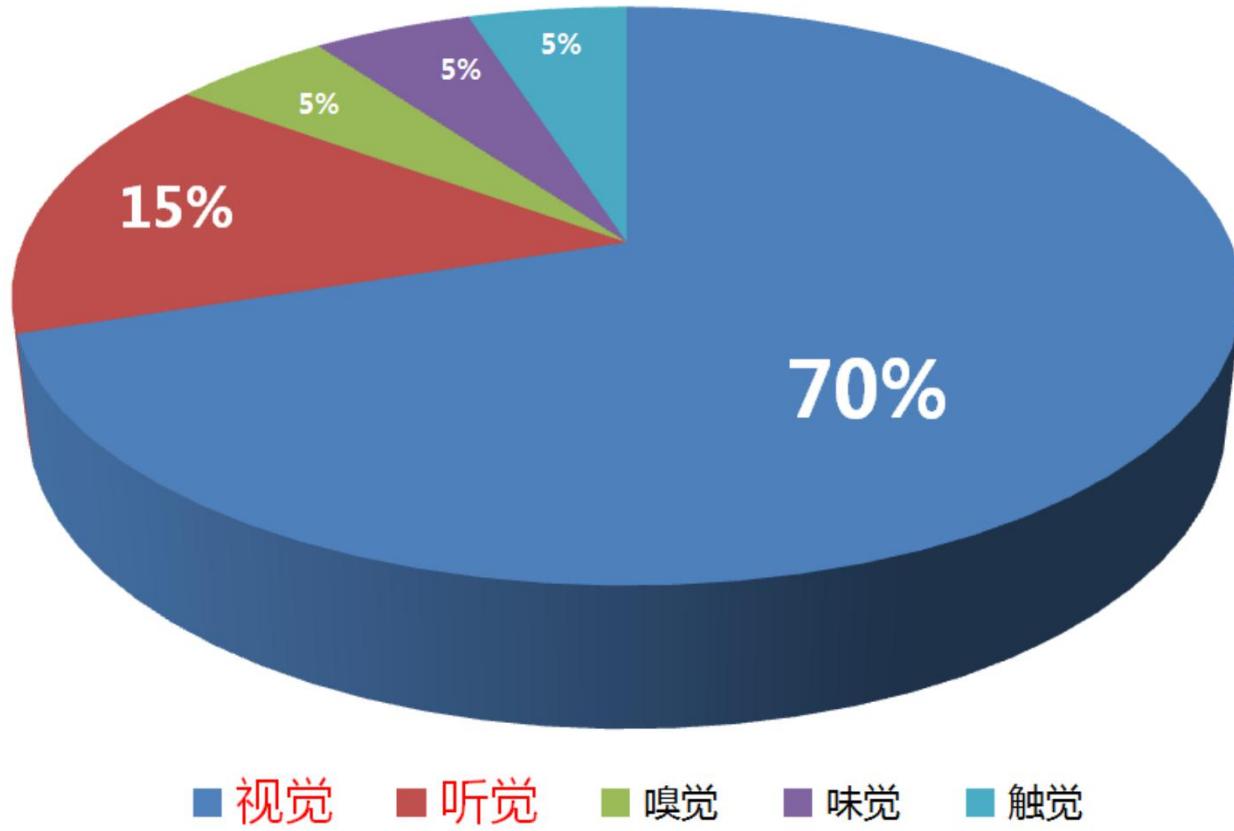
VR技术应用广泛





感官对大脑感知的影响

- ❖ 大脑对所处环境的感知主要来自视觉（70%）和听觉（15%）





实现过程

视觉

- 建立三维世界
- 提高虚拟世界真实度
- 存储三维世界信息

听觉

- 音频信息处理
- 音频信息数字化
- 音频信息类型

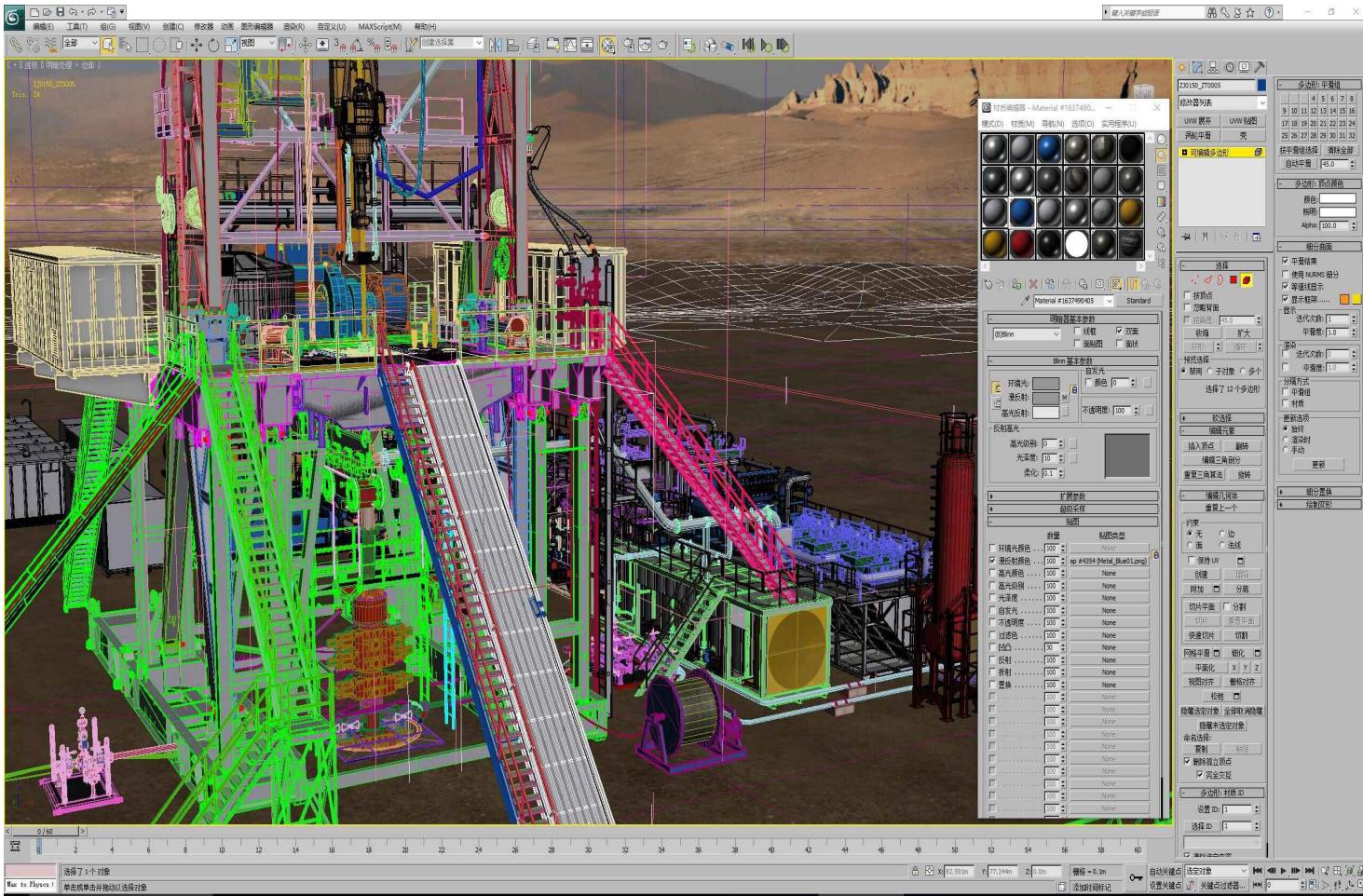
控制

- 读取三维世界信息
- 读取声音信息
- 通过控制程序完成人与三维空间的交互
- 并输出交互效果





视觉——三维环境构成



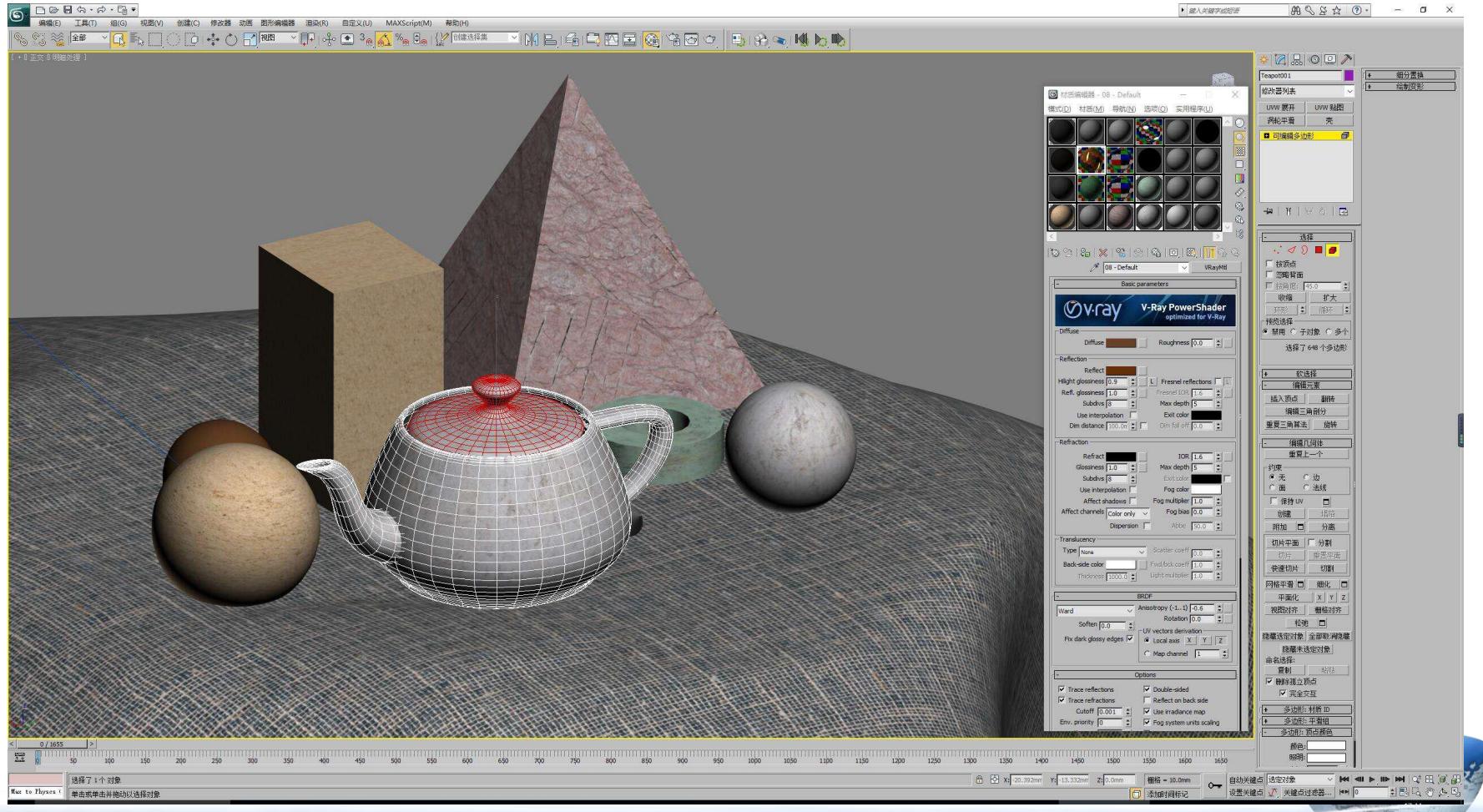
❖ 点
❖ 线
❖ 面
❖ 材质
❖ 贴图
❖ 动画





视觉——三维环境搭建

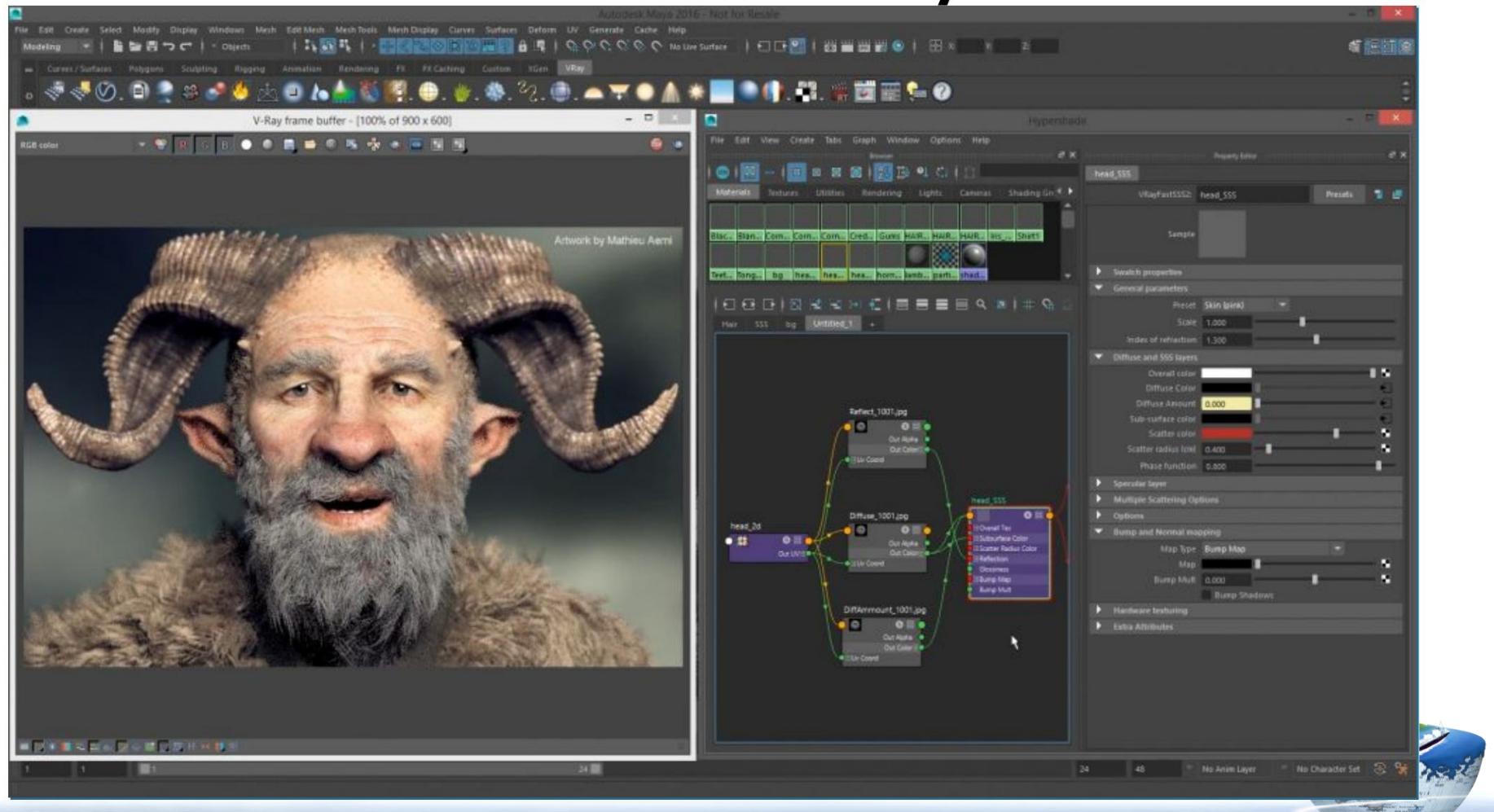
3D Studio Max





视觉——三维环境搭建

Autodesk Maya





如何存储模型信息？



- ❖ 点
- ❖ 线
- ❖ 面
- ❖ 材质

常用导出数据文件类型.obj，.FBX等



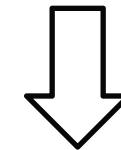
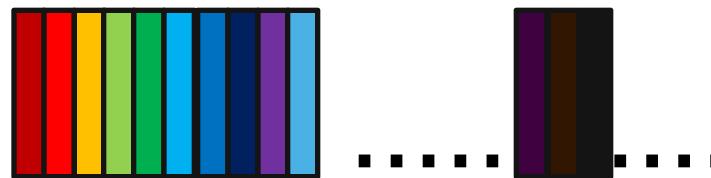


如何存储图像信息？

❖ 提高三维世界的真实感和细节度——贴图

- 人眼可以感知颜色，计算机可以存储和显示颜色信息
- 颜色信息是如何保存在计算机中的？

颜色感知：



计算机数据：

颜色编码（二进制数据）





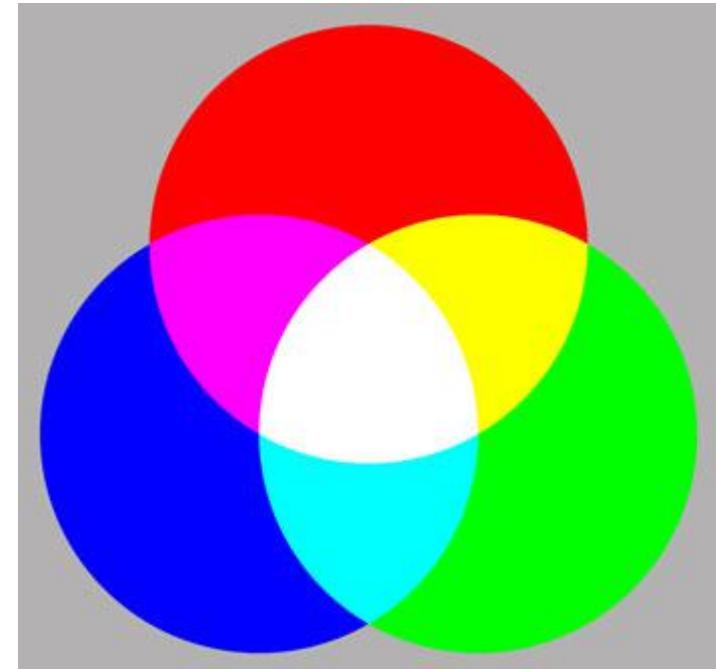
RGB模型

❖ 色光三原色：

- 红 (Red)
- 绿 (Green)
- 蓝 (Blue)

❖ 任何色彩的光，都可由这三原色按不同比例混合得到。

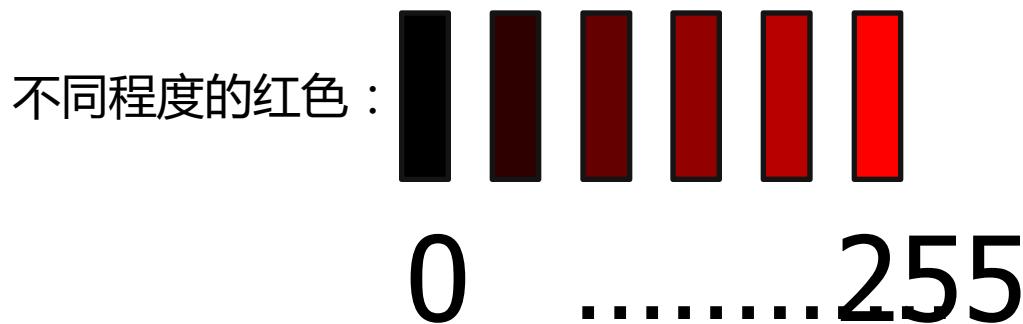
❖ 由于是色光混合，光线越多，颜色越浅





RGB模型

- ❖ 将不同程度的红(R)、绿(G)、蓝(B)光分为256等份，以0~255表示。



- ❖ 0表示程度最低，即没有，没有光即黑色
- ❖ 255表示程度最高，即纯红色
- ❖ 绿色光和蓝色光以此类推





图像信息编码

- ❖ 图像又称像素图、点阵图和位图。图像由一个一个的像素点排列构成。
- ❖ 记录每个像素点的颜色信息，完成对图像信息的记录



位图

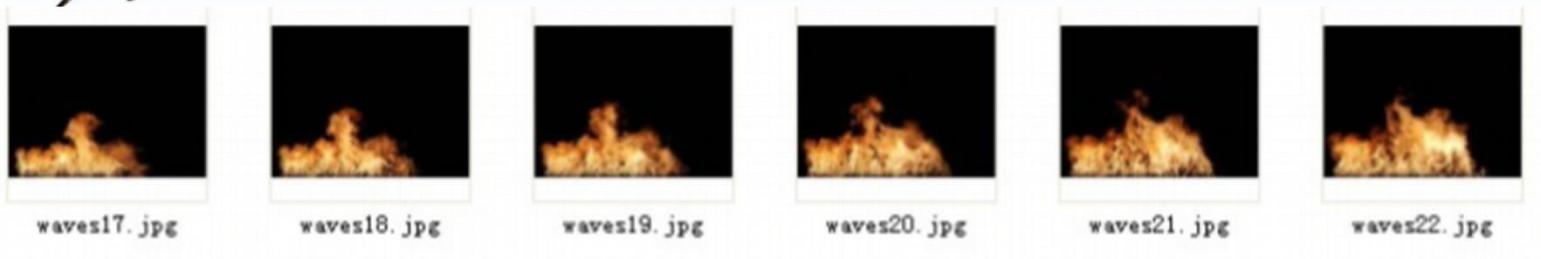
R (255) G (255) B (255) , 白色
R (0) G (0) B (0) 黑色
R (255) G (0) B (0) 红色
R (0) G (0) B (255) 蓝色
R (255) G (255) B (0) 黄色
其他的颜色：

- R (200) G (133) B (78)
- R (60) G (232) B (45)

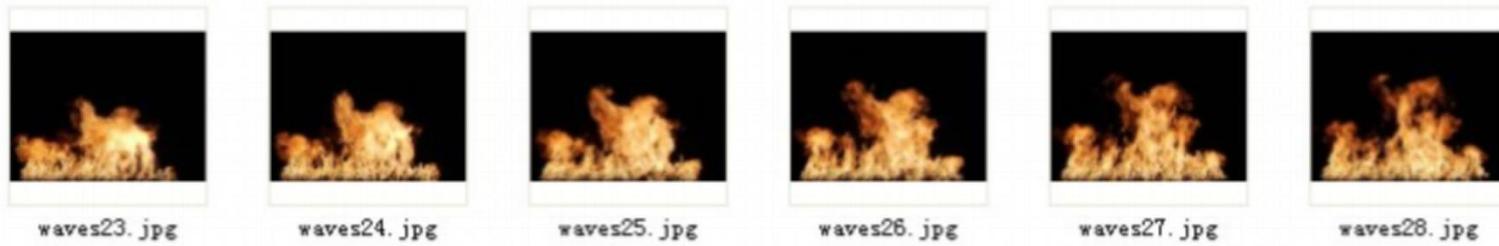




动画是如何存储的



- ❖ 医学证明人类具有“视觉暂留”的特性，人的眼睛看到一幅画或一个物体后，在0.34秒内不会消失。
- ❖ 动画就是采用逐帧拍摄对象并连续播放而形成运动的影像技术
- ❖ 在一幅画还没有消失前播放下一幅画，就会给人造成一种流畅的视觉变化效果





实现过程

视觉

- 建立三维世界
- 提高虚拟世界真实度
- 存储三维世界信息

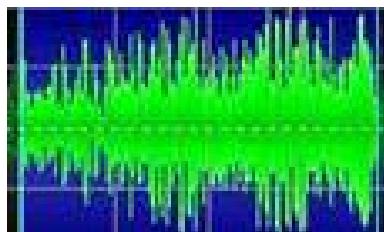
听觉

- 音频信息处理
- 音频信息数字化
- 音频信息类型

控制

- 读取三维世界信息
- 读取声音信息
- 通过控制程序完成人与三维空间的交互
- 并输出交互效果

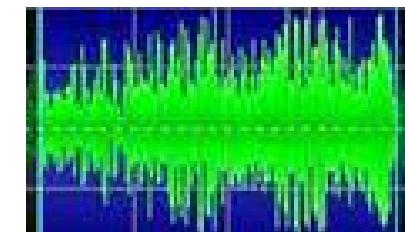




听觉——音频信息



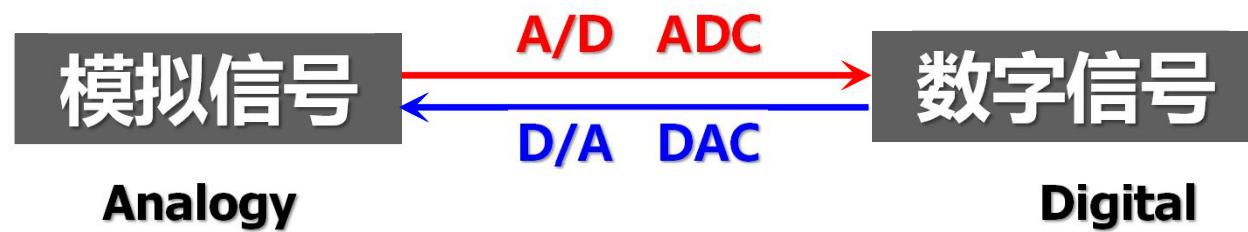
01001011100
01101101011
01100111101
1110...





音频信息数字化

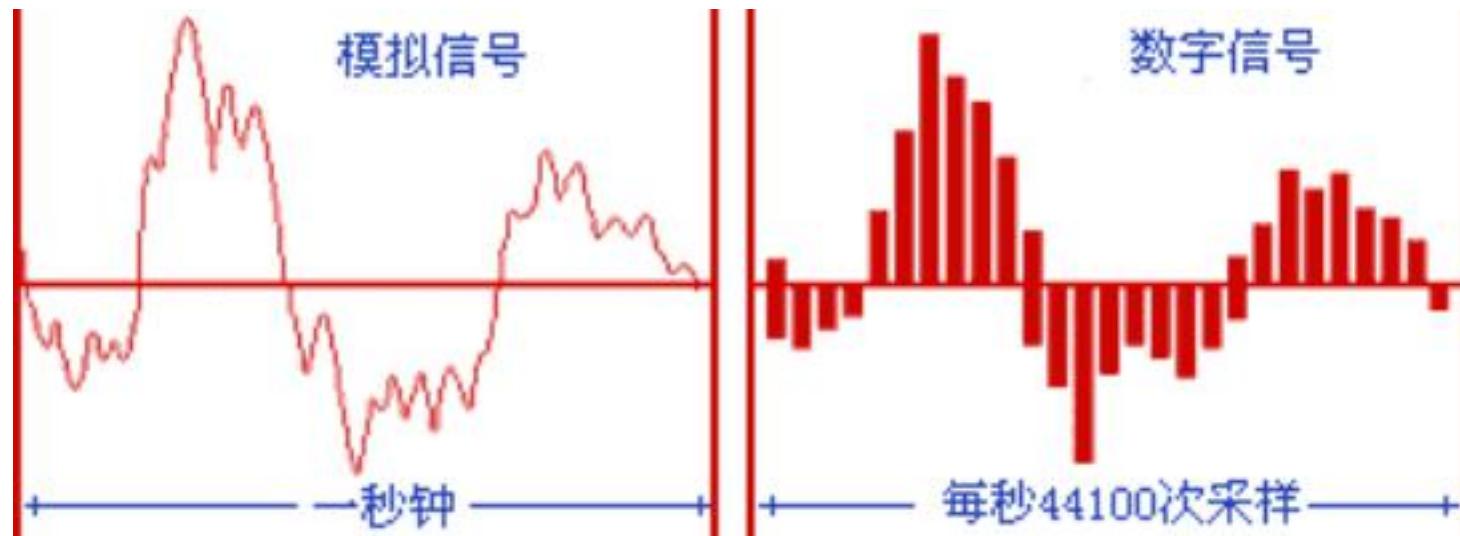
- ❖ 声音的数字化
- ❖ 2. 声音数字化过程





音频信息数字化

- ◆ 声音：用波形文件、MIDI音乐文件或压缩音频文件方式表示



WAV MID MP3 WMA ...





实现过程

视觉

- 建立三维世界
- 提高虚拟世界真实度
- 存储三维世界信息

听觉

- 音频信息处理
- 音频信息数字化
- 音频信息类型

控制

- 读取三维世界信息
- 读取声音信息
- 通过控制程序完成人与三维空间的交互
- 并输出交互效果





程序控制三维空间交互

The screenshot displays a Unity Personal (64bit) - YJYL234.unity - YJYL1017 - Web Player <DX11> window. The main view shows a detailed 3D model of an industrial oil rig or platform. Several characters in red protective suits are visible on the platform. The Unity interface includes:

- Hierarchy Panel:** Shows the project structure with nodes like MainCamera, Models, GameObject, SceneManager, Camera, HumanAnimation, AudioObject, ParticleObject, UIRoot, Minimapneedman, KONGFU, jxkf, SynchronizerRole, ZJ0039, ZJ_XZ, Drillsystem_part01, and numerous ZJ00xx objects.
- Project Panel:** Shows Favorites, Assets, and Resources. Favorites include BopSwitchItem, CameraControl, Executing, FreeCameraControl, NameDisplay, Parameter, Raycast, SwitchHandle, SwitchInput, SynchronizerRole, Vane, WorkerAnimation, and WorkerController. Assets include Animation, Demigiant, DrillSystem, Emergency, Examine, Highlighting, kolmich, Material-Engine, Materials, Model, oldMat, particle effects, Plugins, and Prefabs. Resources include Script, Sound, Standard Assets, Th123, UI, and UnityVS workflow.
- Inspector Panel:** Shows the SynchronizeRole component for My Animator. The code in the Inspector is as follows:

```
using UnityEngine;
using System.Collections;

public class SynchronizeRole : MonoBehaviour
{
    public Animator MyAnimator;
    float Forward;
    float Turn;
    bool Leg;

    Vector3 pos;
    Quaternion rot;

    void Start()
    {
        //MyAnimator = this.GetComponent<Animator>();
    }

    void OnSerializeNetworkView(BitStream stream, NetworkMessageInfo info)
    {
        //本机代码
        if (stream.isWriting)
        {
            pos = transform.localPosition;
            rot = transform.localRotation;

            if (MyAnimator != null)
            {
                Forward = MyAnimator.GetFloat("Forward");
                Turn = MyAnimator.GetFloat("Turn");
                Leg = MyAnimator.GetBool("Leg");
            }
        }
        stream.Serialize(ref pos);
        stream.Serialize(ref rot);
        stream.Serialize(ref Forward);
        stream.Serialize(ref Turn);
        stream.Serialize(ref Leg);
    }

    //其它计算机代码
    else
    {
        stream.Serialize(ref pos);
        stream.Serialize(ref rot);
        stream.Serialize(ref Forward);
        stream.Serialize(ref Turn);
        stream.Serialize(ref Leg);

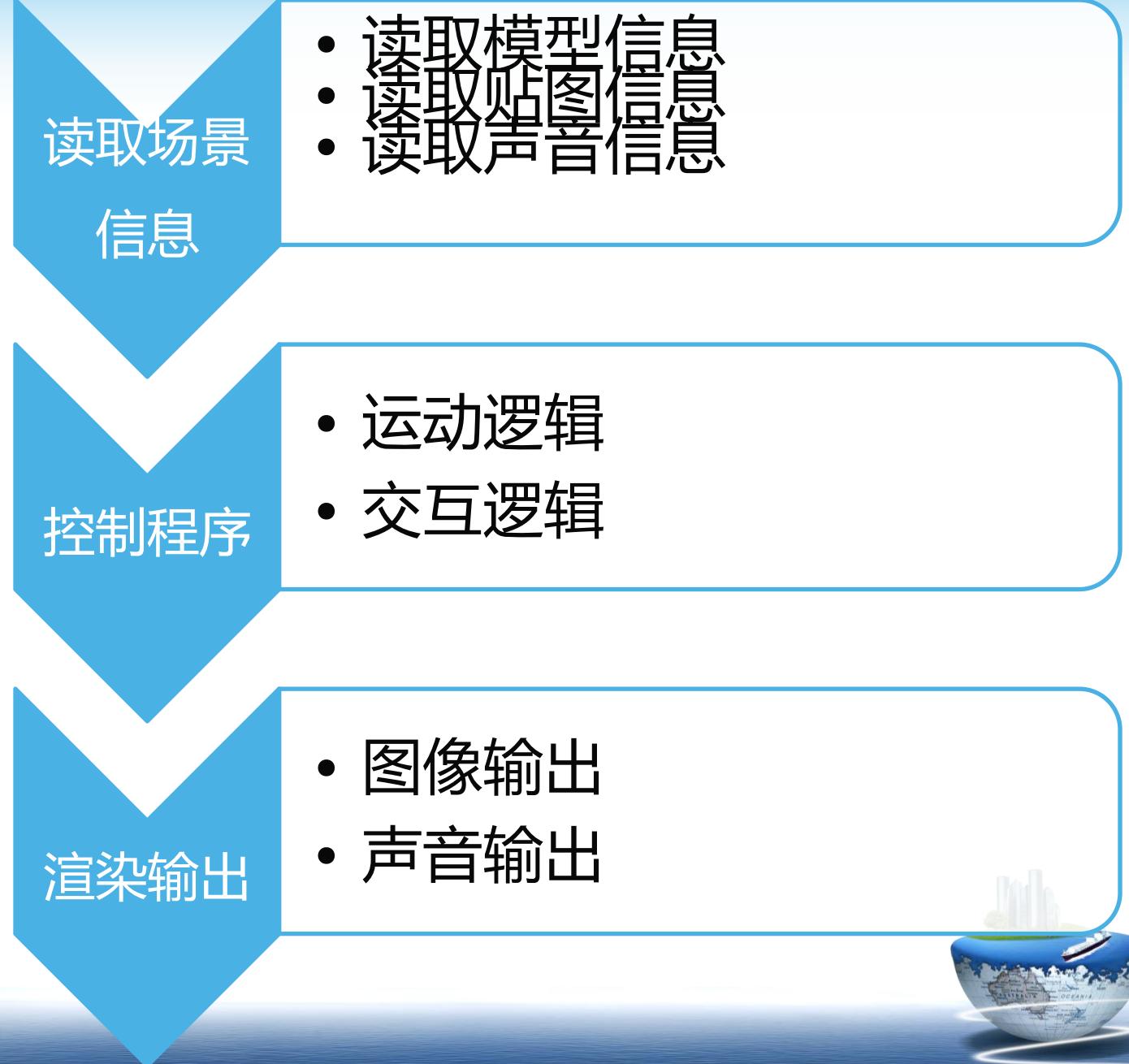
        transform.localPosition = pos;
        transform.localRotation = rot;

        if (MyAnimator != null)
        {
            MyAnimator.SetFloat("Forward", Forward);
        }
    }
}
```

Asset Labels: Sync

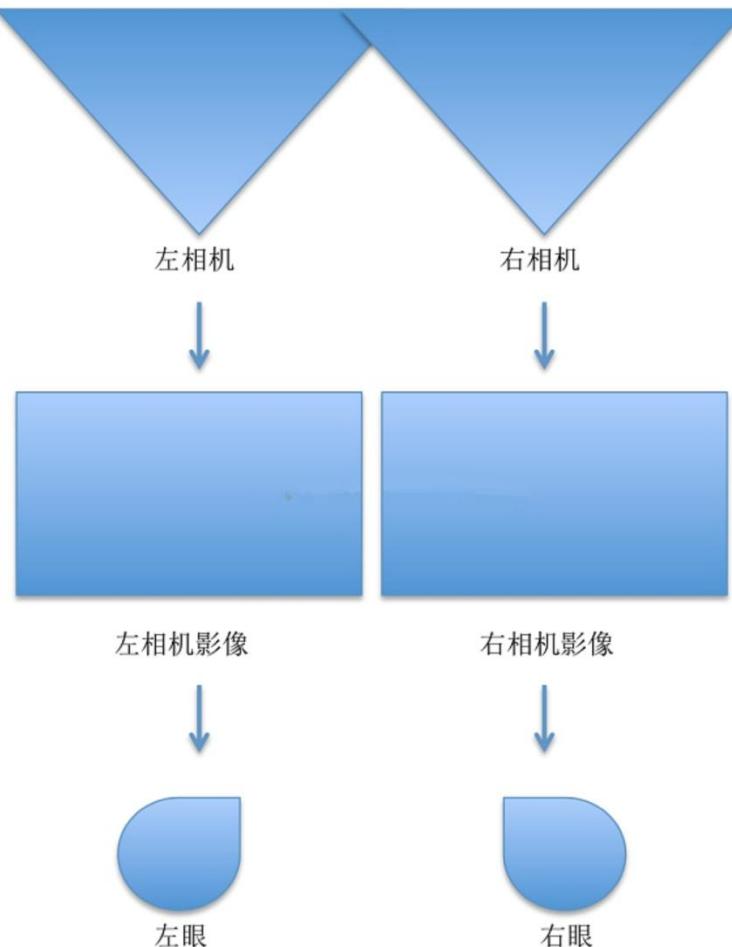


程序控制三维空间交互





虚拟现实头盔——沉浸式虚拟现实



◆ 三维成像原理

- 人的两眼之间有距离，看到的是有景深的图像
- 运用计算机生成合适的图像偏移，模拟左右眼看到的图像
- 分别播放给左右眼观看，即可形成立体图像



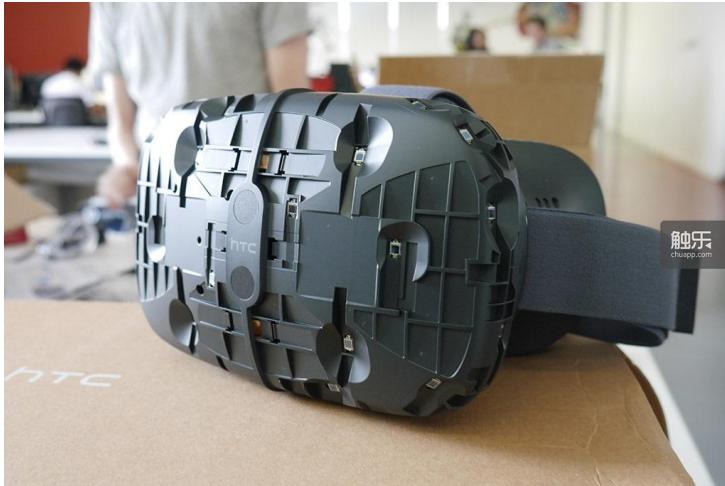


三维成像原理





沉浸式VR定位原理

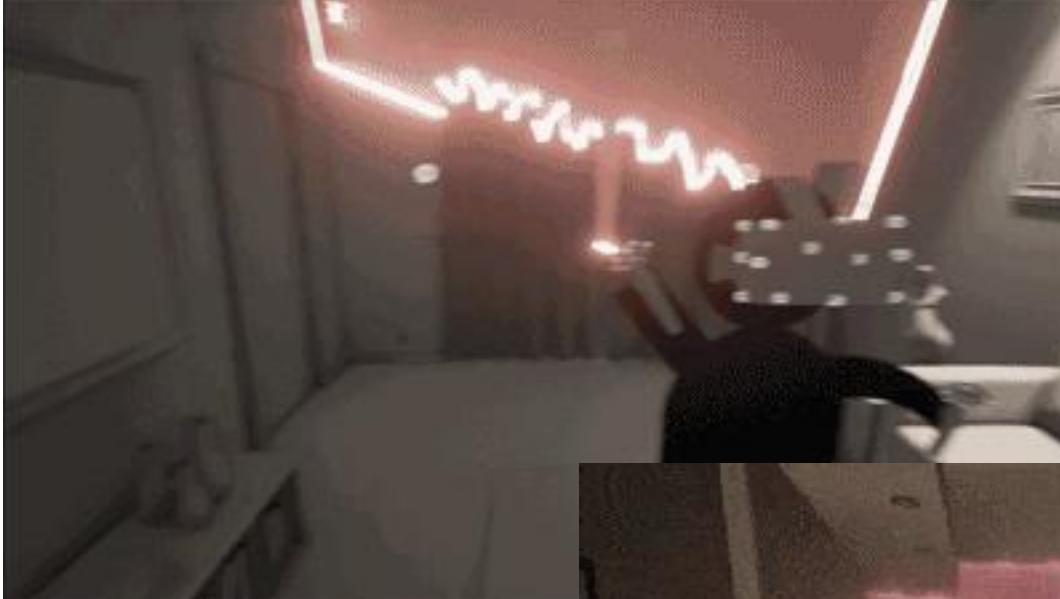


- ❖ 传统的光学系统是摄像头式的，摄像头跟踪头显上标记的马克点实现跟踪
- ❖ 最新的HTCVive使用目前最好的VR光学跟踪方案：Lighthouse技术





沉浸式VR定位原理





更好的沉浸式虚拟现实体验



触感：力反馈系统





混合式虚拟现实的应用



Thank You !

