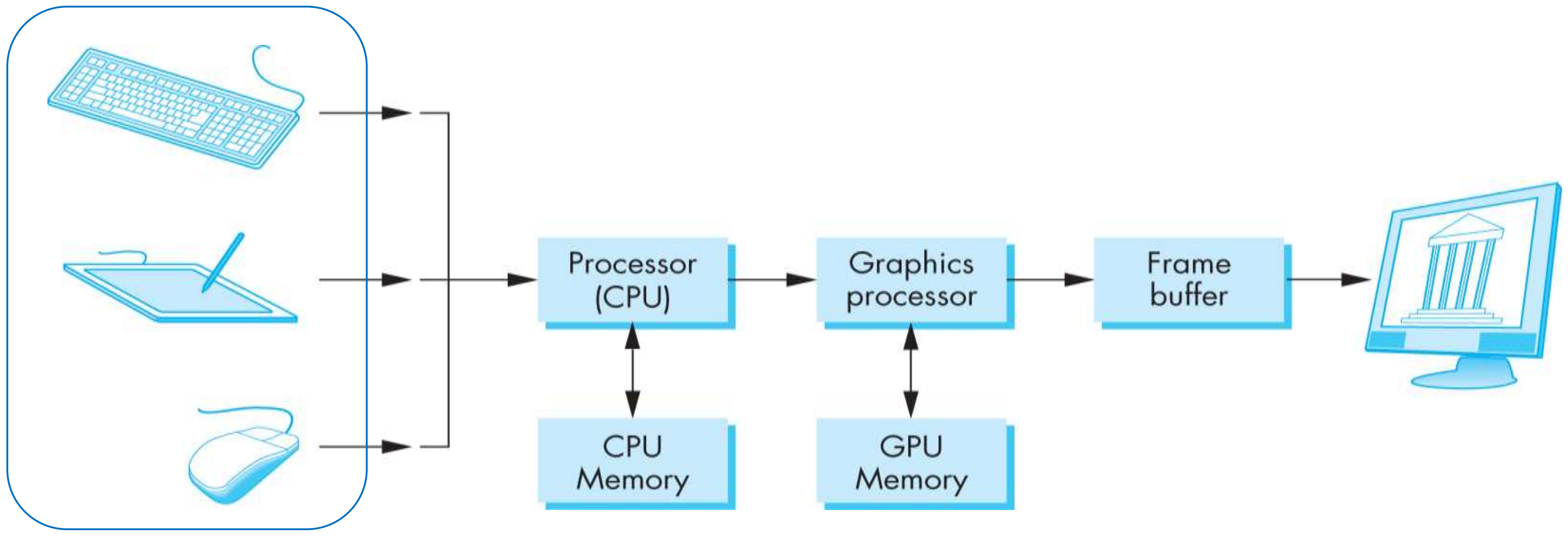
Computer Graphics System Outlines

- Input devices
- Output device
- Raster Imaging System-光栅成像系统*

- Image Formation Principle –成像原理
- Rendering Pipeline –渲染管线实现*
- The Programmer’s Interface-程序员编程接口

Basic Graphics System



Input devices

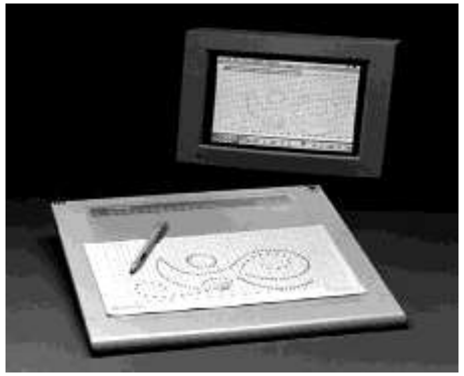
Raster Imaging System

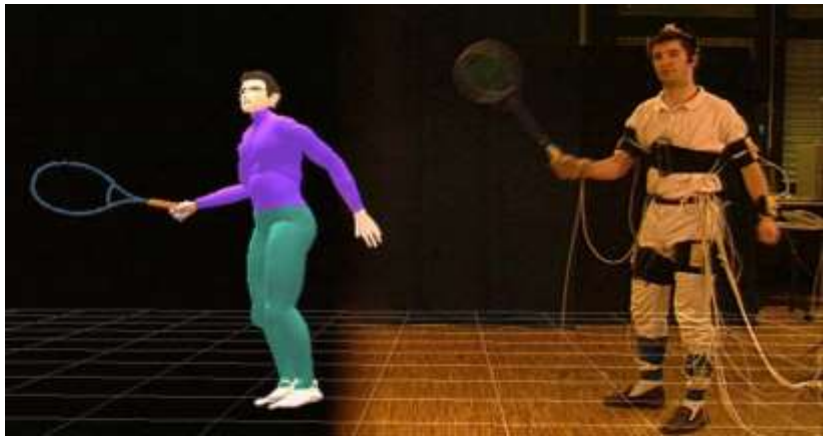
Output device

Input Devices

- Keyboard
- Mouse
- **Tablet/stylus**
  - consist of a flat drawing board and a pen
  - electromagnetic
  - high resolution
- **Joystick**

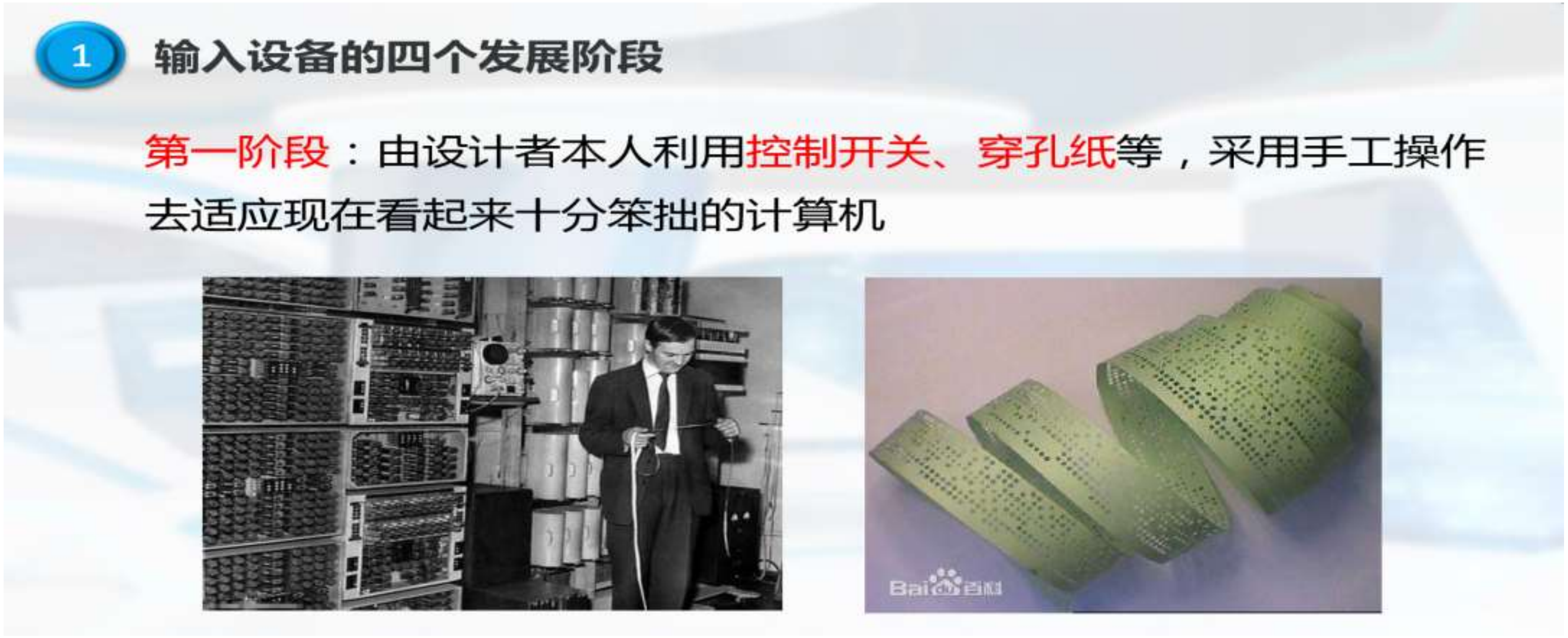
  the rate of cursor movement depends on the displacement of the stick
- **3D handheld laser scanning digitiz**
- **Touch-screen monitors**
- **Motion Tracking sensors**





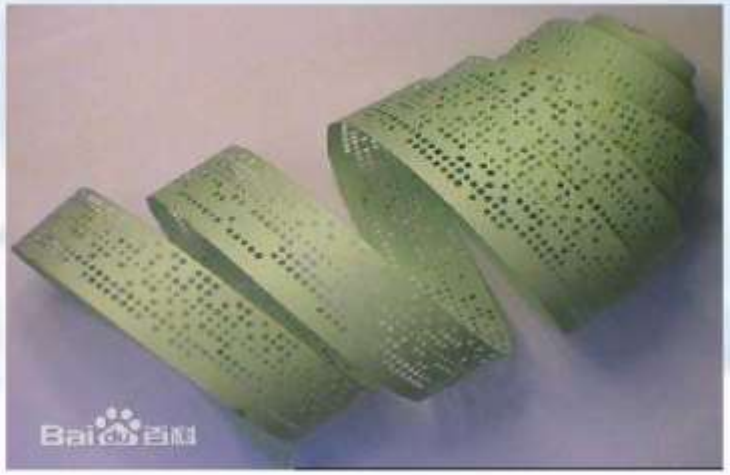
Input Devices（cont.)

- *Ref: 大学MOOC 万琳：计算机图形学*



1 输入设备的四个发展阶段

第一阶段：由设计者本人利用控制开关、穿孔纸等，采用手工操作去适应现在看起来十分笨拙的计算机

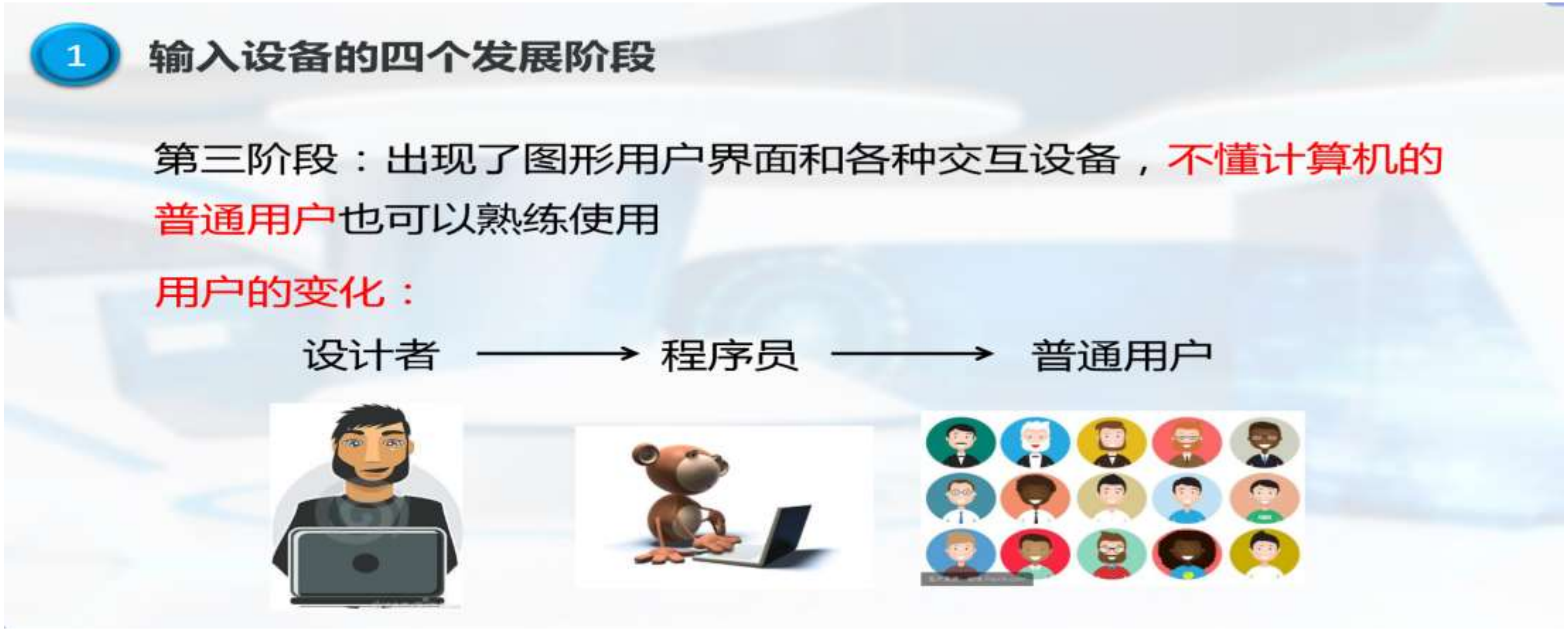


Input Devices(cont.)

1 **输入设备的四个发展阶段**

第二阶段：计算机的主要使用者——程序员利用键盘、光笔等输入设备，采用批处理作业语言或交互命令语言的方式和计算机打交道

Input Devices(cont.)



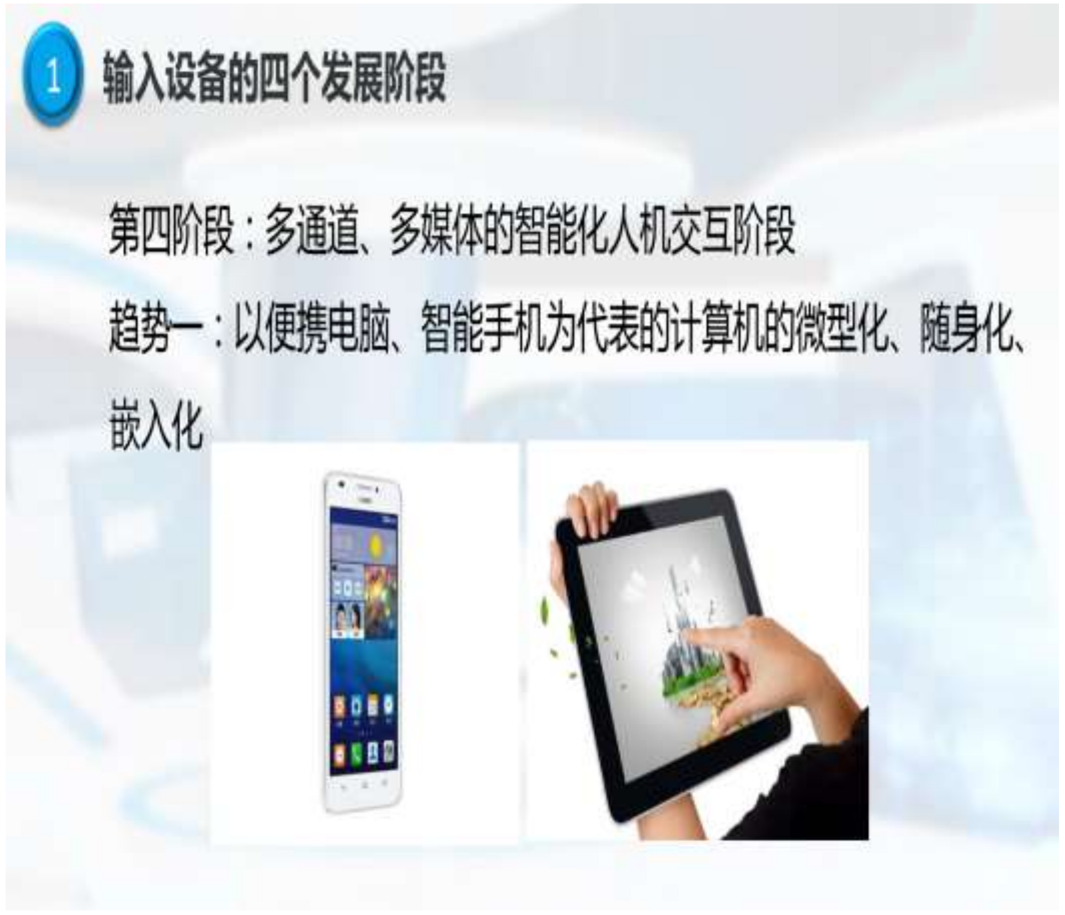
1 输入设备的四个发展阶段

第三阶段：出现了图形用户界面和各种交互设备，不懂计算机的普通用户也可以熟练使用

用户的变化：

设计者 ——→ 程序员 ——→ 普通用户

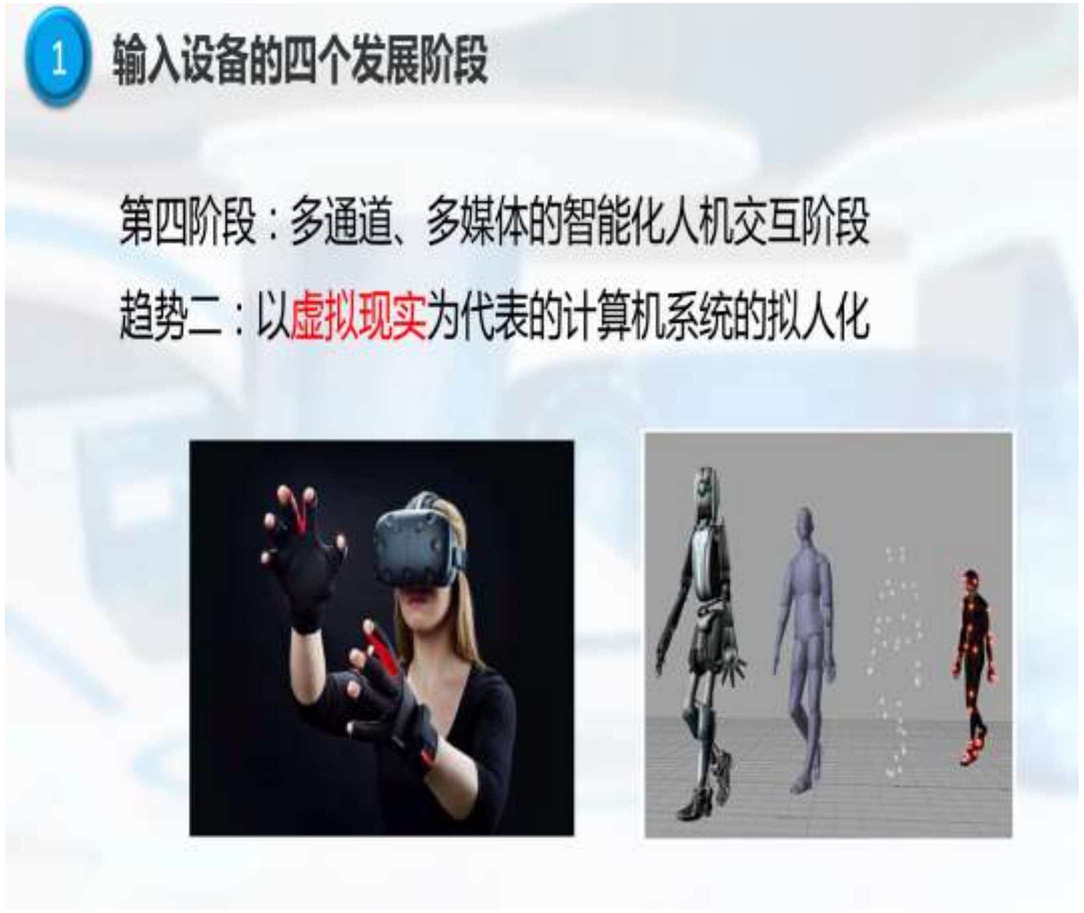
Input Devices(cont.)



1 输入设备的四个发展阶段

第四阶段：多通道、多媒体的智能化人机交互阶段

趋势一：以便携电脑、智能手机为代表的计算机的微型化、随身化、嵌入化



1 输入设备的四个发展阶段

第四阶段：多通道、多媒体的智能化人机交互阶段

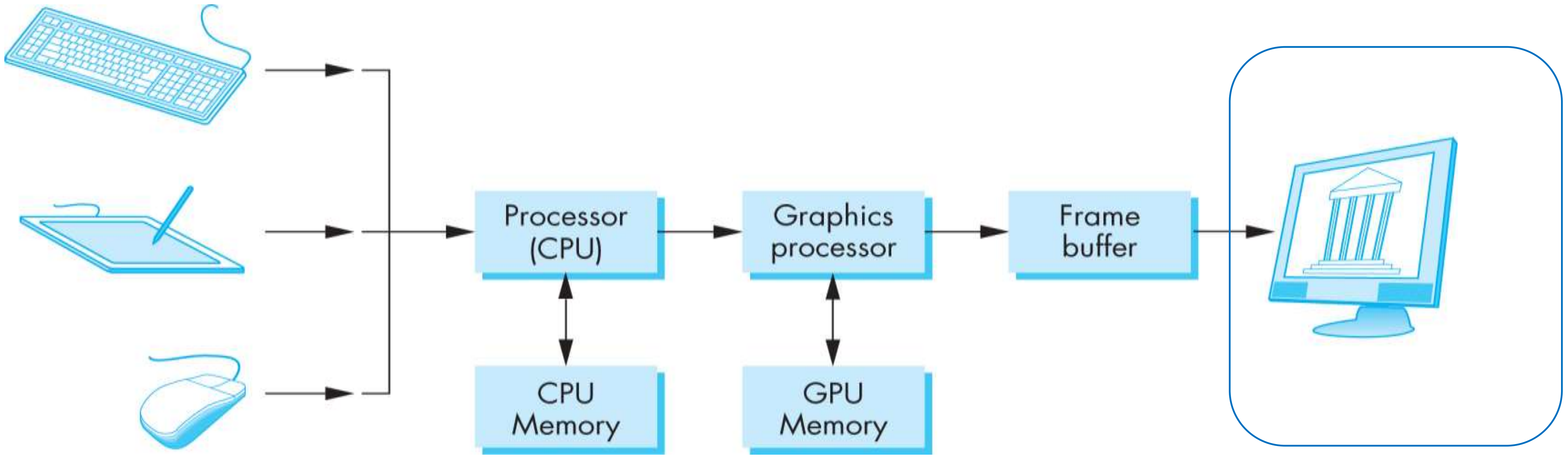
趋势二：以虚拟现实为代表的计算机系统的拟人化

Computer Graphics System Outlines

- Input devices
- Output device
- Raster Imaging System-光栅成像系统*

- Image Formation Principle –成像原理
- Rendering Pipeline –渲染管线实现*
- The Programmer's Interface-程序员编程接口

Basic Graphics System



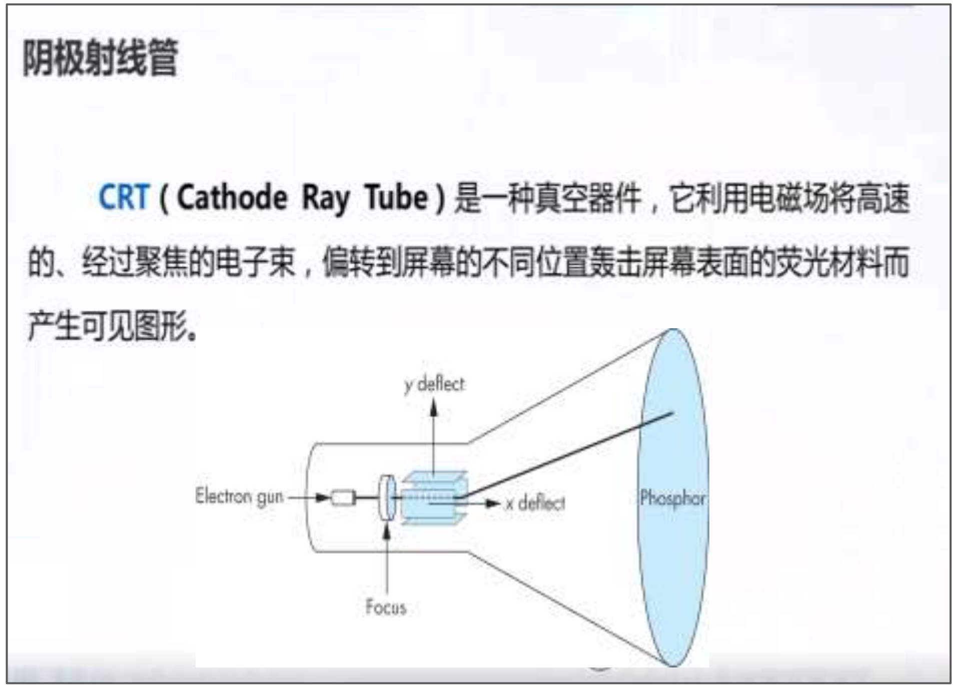
Input devices

Raster Imaging System

Output device

Display device

- 阴极射线管显示器



阴极射线管

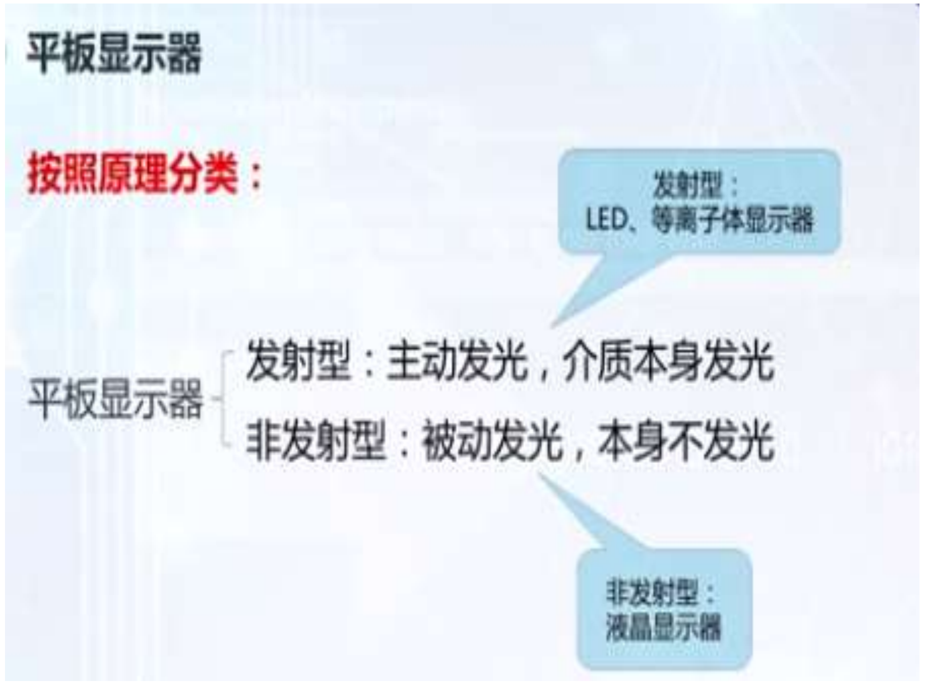
CRT（Cathode Ray Tube）是一种真空器件，它利用电磁场将高速的、经过聚焦的电子束，偏转到屏幕的不同位置轰击屏幕表面的荧光材料而产生可见图形。

阴极射线管

从外形上看，CRT分为：**管颈部分、锥体部分、屏幕部分**

Display device(cont.)

- **平板显示器**



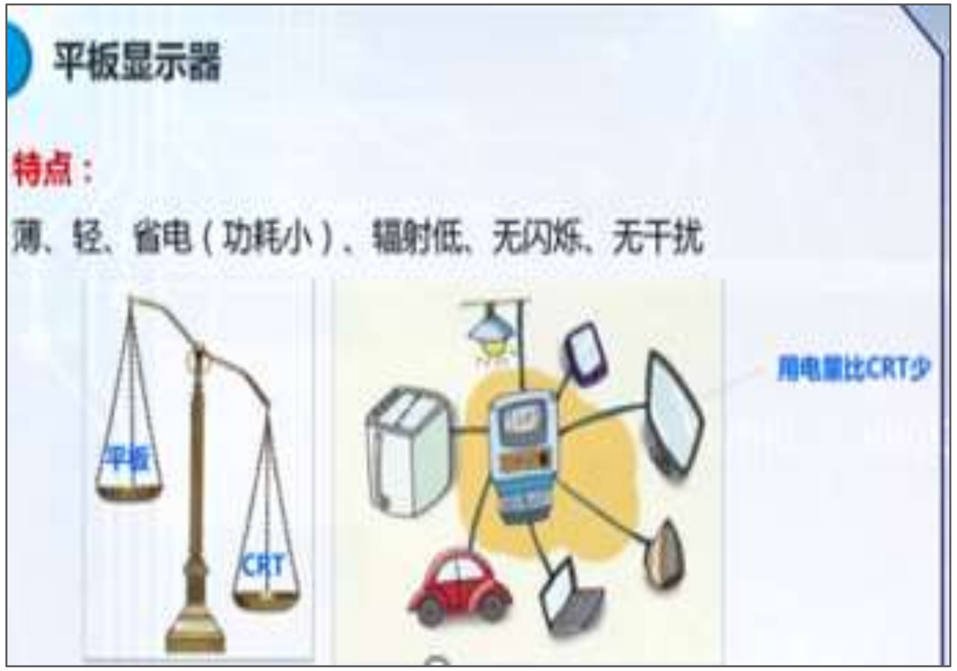
平板显示器

按照原理分类：

平板显示器
- 发射型：主动发光，介质本身发光
- 非发射型：被动发光，本身不发光

发射型：LED、等离子体显示器

非发射型：液晶显示器



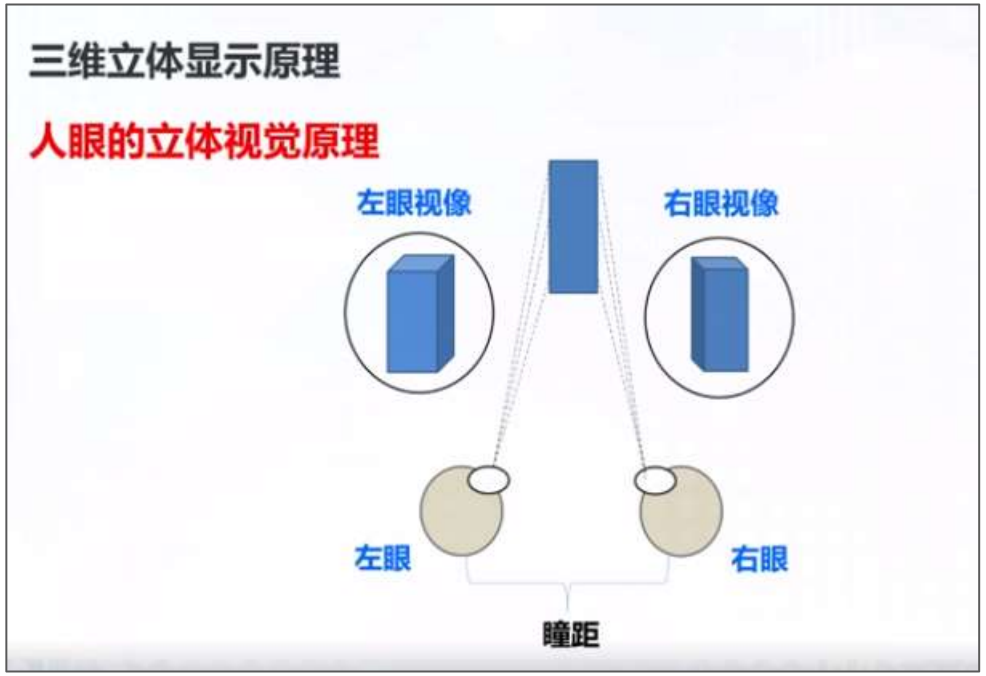
平板显示器

特点：

薄、轻、省电（功耗小）、辐射低、无闪烁、无干扰

Display device(cont.)

- 立体显示器


三维立体显示原理

人眼的立体视觉原理

裸眼立体显示器

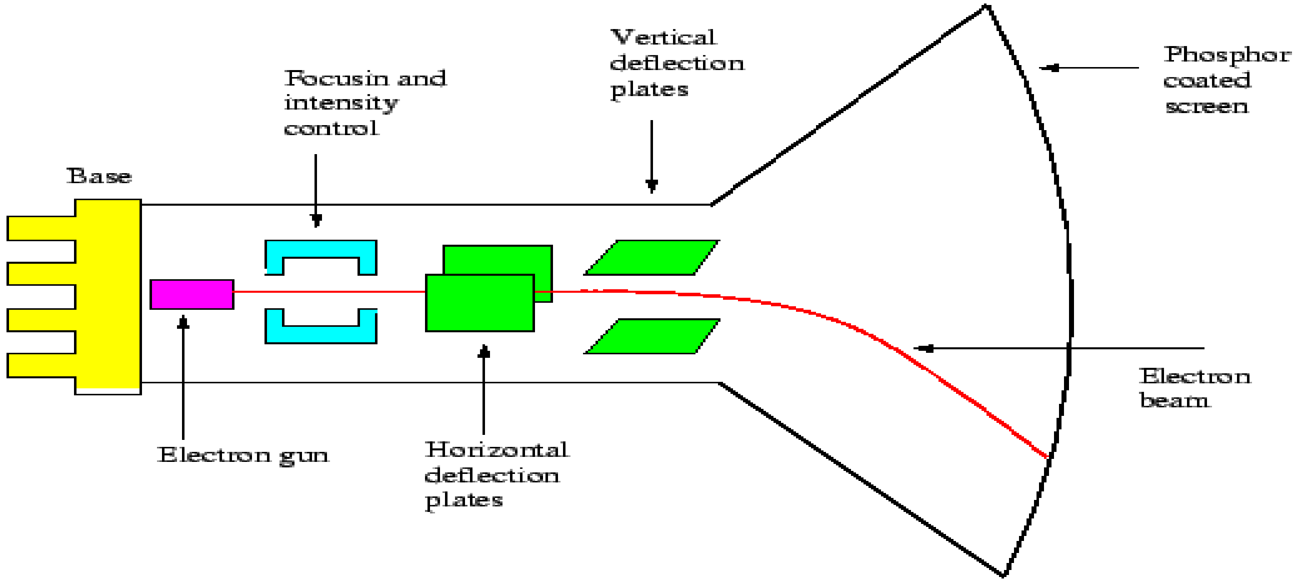
发展方向：裸眼、全息、交互

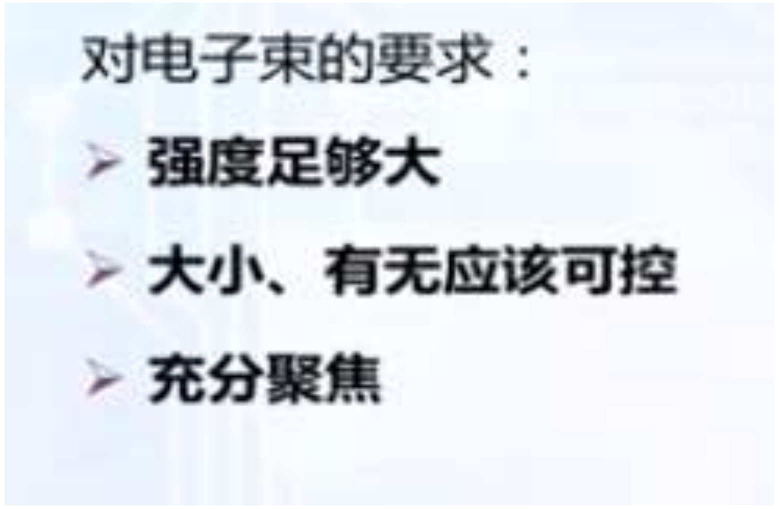
Display device(cont.)

- CRT(Cathode-ray tube display 阴极射线管):演示

➤ **显示器组成Components**

electron gun电子枪；focus聚焦；x&y deflect偏转；phosphor荧光体


Vertical deflection plates
Focusin and intensity control
Phosphor coated screen
Base
Electron gun
Horizontal deflection plates
Electron beam



对电子束的要求：

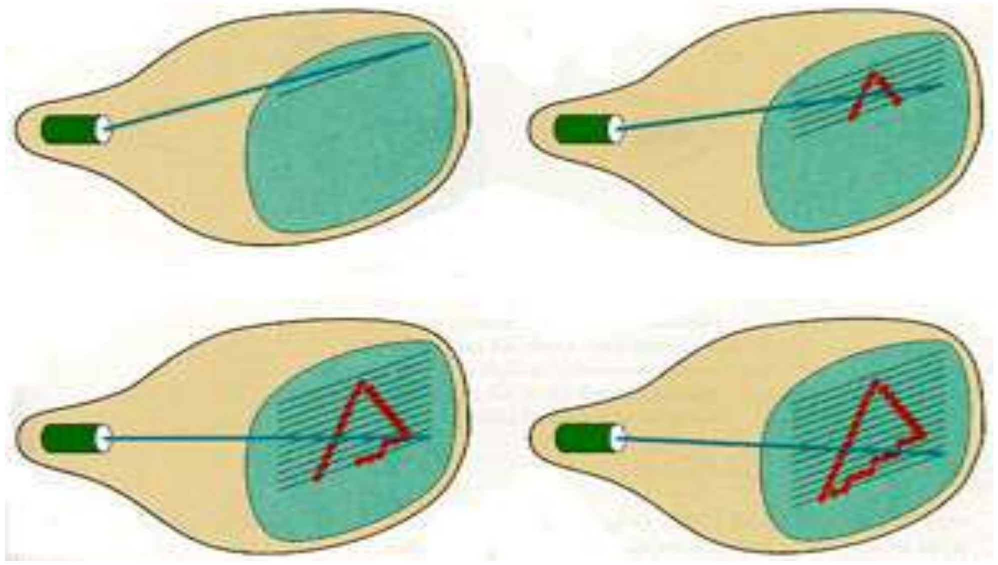
- ➤ **强度足够大**
- ➤ **大小、有无应该可控**
- ➤ **充分聚焦**

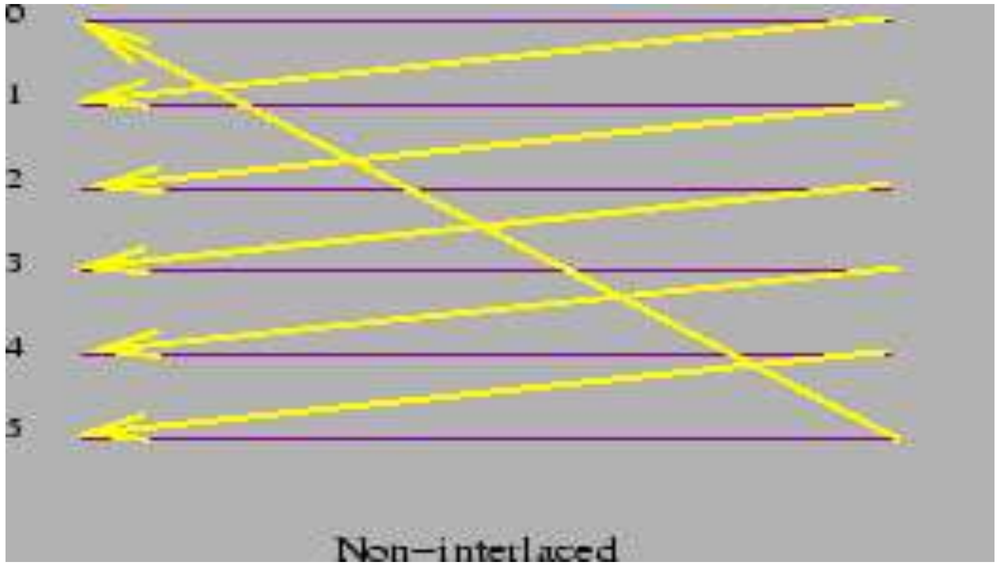
Display device(cont.)

CRT(Cathode-ray tube display 阴极射线管)(cont.)

- **光栅扫描：The beam光线 moves regularly in the trails（路径）**演示

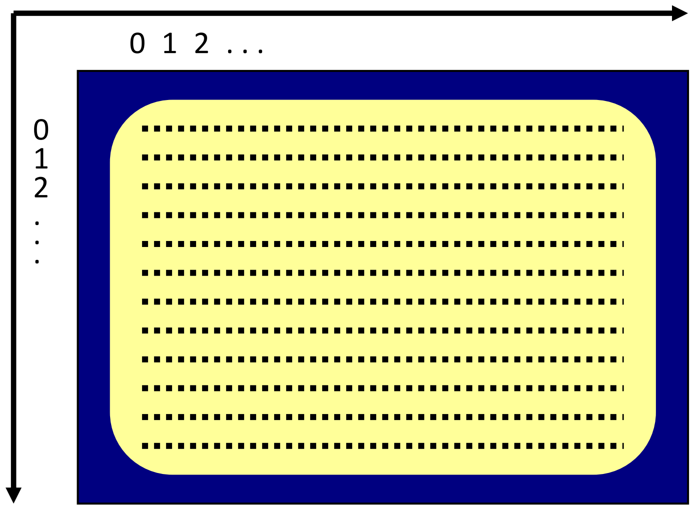
The screen of a computer monitor consists of lines numbered 0, 1, 2, … starting from the top， Each lines consists of luminous spot numbered from left to right





Display device(cont.)

- CRT(Cathode-ray tube display 阴极射线管)(cont.)
  - ➢**光栅扫描显示器基本概念**
    - Light spot（光点）
      - 最小像素
    - Raster(光栅)
      - A bitmap image, consisting of a grid of pixels 一种位图图像，由像素网格组成
    - physical resolution（物理分辨率）
      - 屏幕上的二维阵列所包含的光点数
      - 显示器的最大显示分辨率
      - Example:Dimensions of the screen
        - SVGA(800 *600)
        - XVGA(1024*768)
        - XVGA(1280*1024)



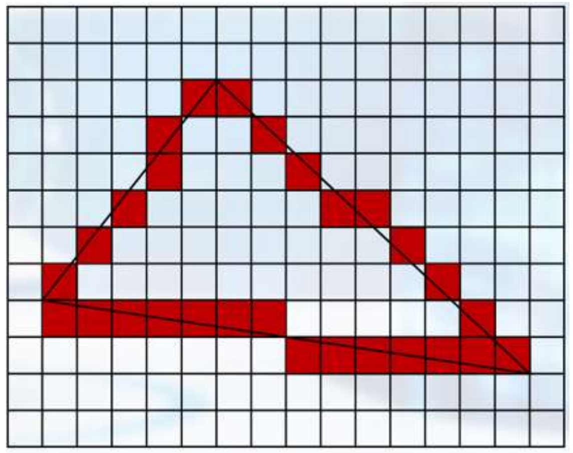
Display device(cont.)

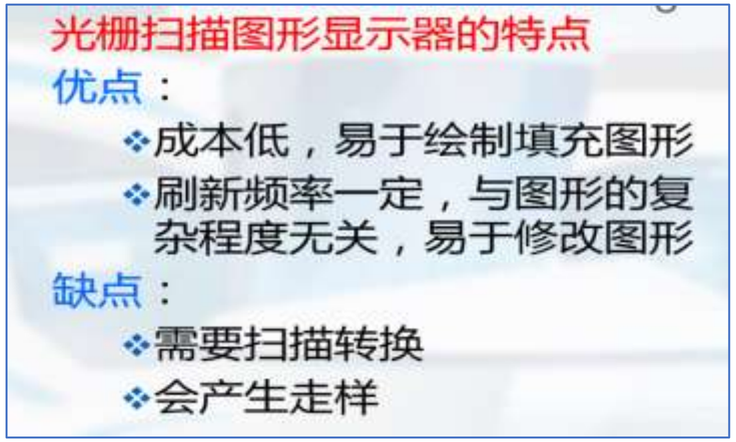
- CRT(Cathode-ray tube display 阴极射线管)(cont.)
  - When the beam hits the phosphor(荧光体) on the screen, the phosphor（荧光粉）lights up（亮起来）， but its intensity(亮度) decays(衰败) very fast.
  - Nonetheless, the image retains in the retinas(视网膜) of our eyes for about 1/20 sec.

➢**Question: 如何保证CRT光栅显示器的屏幕不闪烁?**

➢The entire screen is refreshed 30 to 60 times per second so that we can see a steady picture（整个屏幕每秒刷新30到60次，这样我们就可以看到一个稳定的画面）

➢**Refresh frequency（刷新频率）一般60Hz.**





光栅扫描图形显示器的特点

优点：

❖成本低，易于绘制填充图形

❖刷新频率一定，与图形的复杂程度无关，易于修改图形

缺点：

❖需要扫描转换

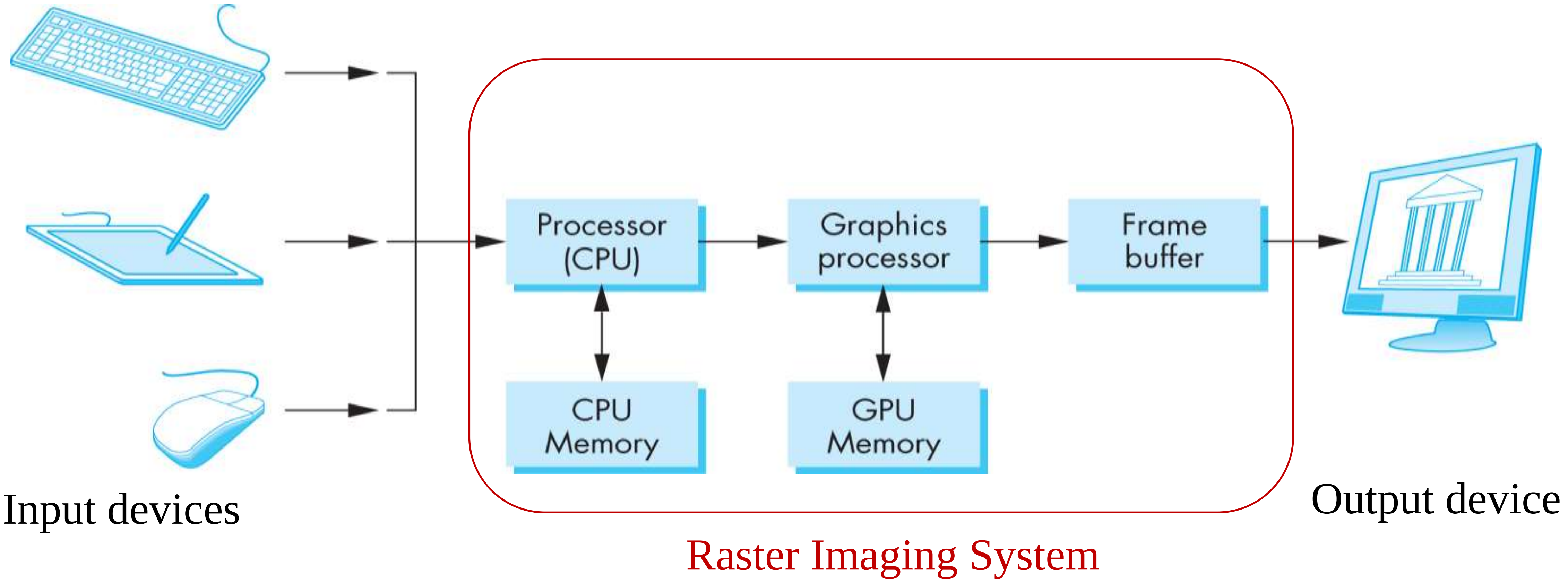
❖会产生走样

Computer Graphics System Outlines

- ➢Input devices
- ➢Output device
- ➢**Raster Imaging System-光栅成像系统***

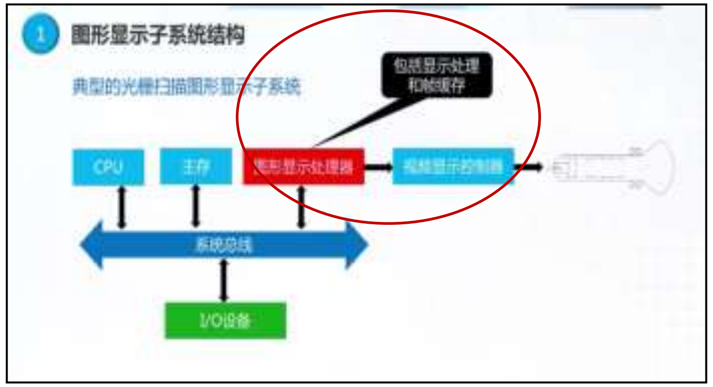
- ➢Image Formation Principle –成像原理
- ➢Rendering Pipeline –渲染管线实现*
- ➢The Programmer’s Interface-程序员编程接口

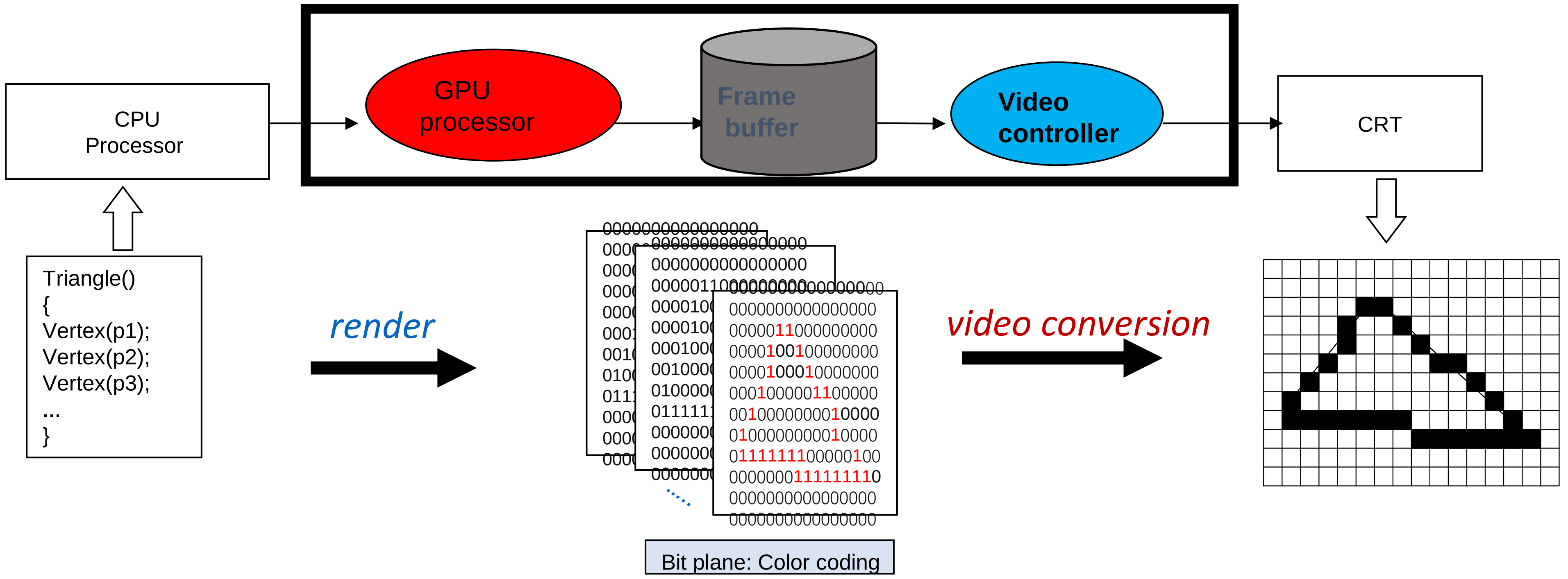
Basic Graphics System



Raster Imaging System

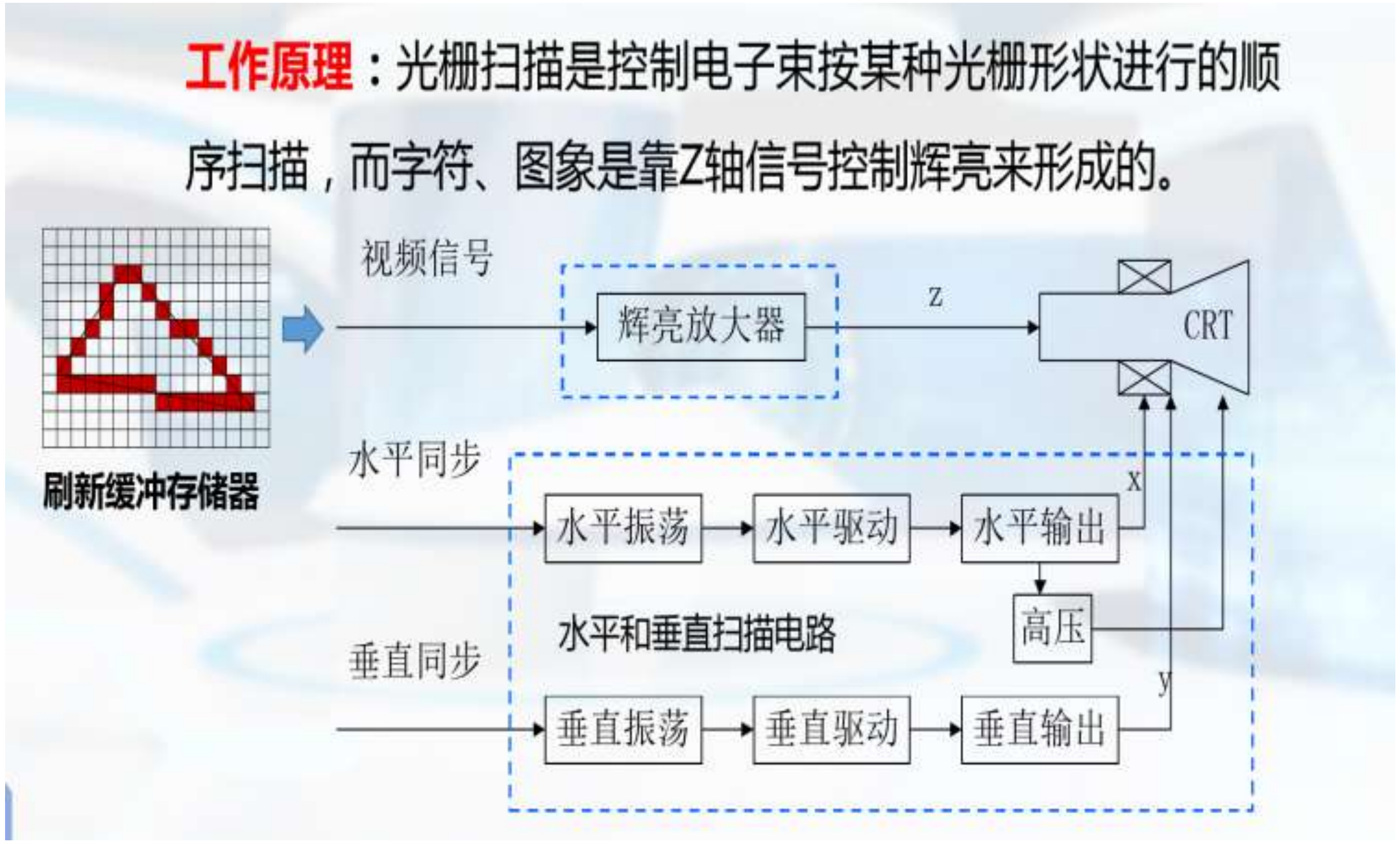
- ➢Main constituents
  - ➢**Controller视频控制器**
  - ➢**Framebuffer帧缓存**
  - ➢**Processor显示处理器**





Raster Imaging System(cont.)

➢1. **Video controller视频控制器**

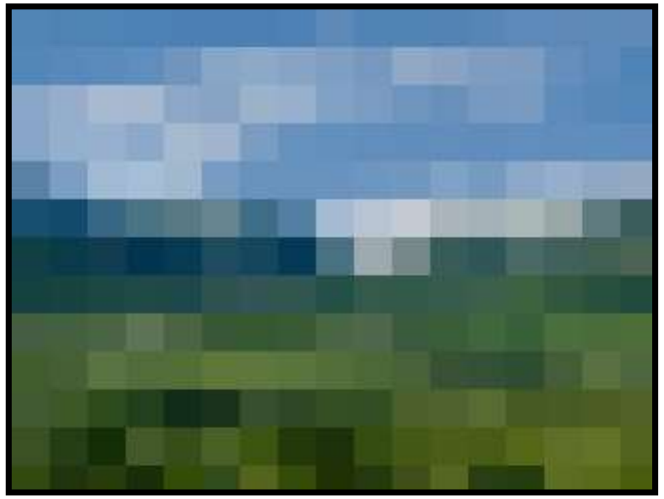


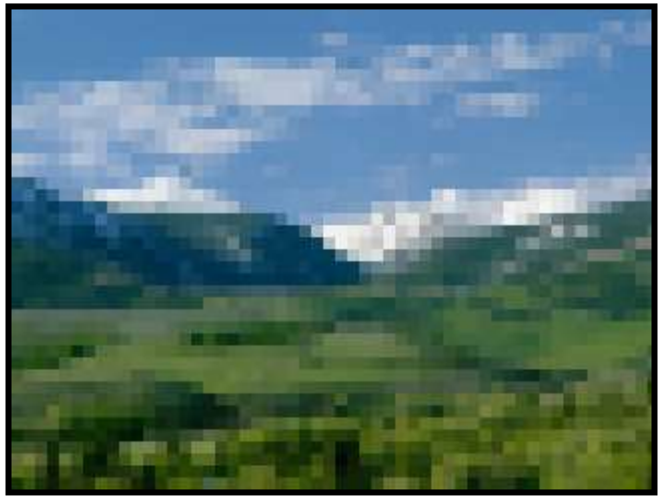
工作原理：光栅扫描是控制电子束按某种光栅形状进行的顺序扫描，而字符、图象是靠Z轴信号控制辉亮来形成的。

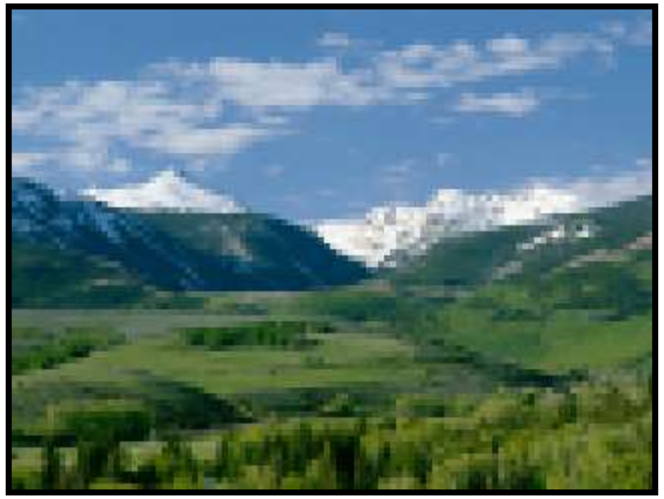
Raster Imaging System(cont.)

➢2. Frame Buffer帧缓存(cont.)

- Pixel像素: Picture element.
- Resolution分辨率: The number of pixels in the framebuffer , determines the detail that you can see in the image, expressed as the number of horizontal pixels multiplied by the number of vertical pixels
- Frame帧: a frame is one of the many still images which compose the complete moving picture. (一帧是构成完整的运动图像的许多静止图像之一) ref: Wikipedia "video frame"
- Framebuffer帧缓存: a portion of random-access memory (RAM)containing a bitmap that drives a video display. It is a memory buffer containing data representing all the pixels in a complete video frame.


Resolution: 17 * 13


51* 26


102 * 52

➢ 对同样大小屏幕(帧)，若像素越小，则分辨率越大。

Raster Imaging System(cont.)

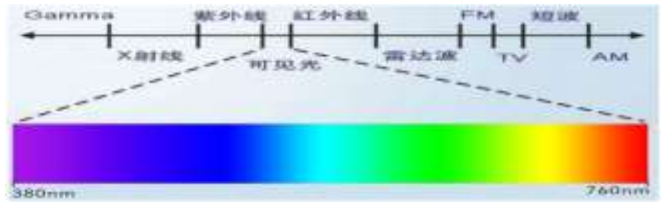
- **2.Frame Buffer帧缓存(cont.)**
  - **How colors are represented/coded ? (颜色如何表示/编码？)**
    - Color Model/Color Space（颜色模型/颜色空间）
  - **How much capacity does a frame image need（帧缓存需要多大容量?）**
    - Pixel Depth(像素深度)
    - Framebuffer capacity(帧缓存容量)
  - **How color codes are stored in framebuffer? （颜色编码如何存储?)**
    - Packed Pixel Method(组合像素法)
    - Bit Plane Method(位面法)
    - Color Lookup Table(CLT查色表)

Raster Imaging System(cont.)

- ➢**Frame Buffer帧缓存(cont.)**
  - ➢**How colors are represente/coded？(颜色如何编码表示？）**
    - ➢Color Model/Color Space（颜色模型/颜色空间）

✓Color& Light

- Light is the part of the electromagnetic spectrum that causes a reaction in our visual systems，Generally these are wavelengths in the range of about 350-750 nm (nanometers)
- Long wavelengths appear as reds and short wavelengths as blues

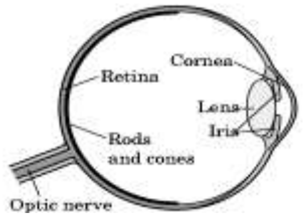


✓Three-Color Theory(三基色理论）

- Human visual system has two types of sensors
- Rods视杆细胞: monochromatic, night vision
- Cones视锥细胞：Color sensitive，Three types of cones，Only three values (the tristimulus values) are sent to the brain

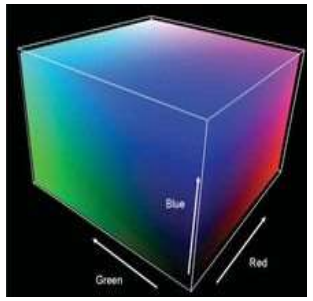
  Need only <u>match these three values</u>

  Need only <u>three primary colors</u>

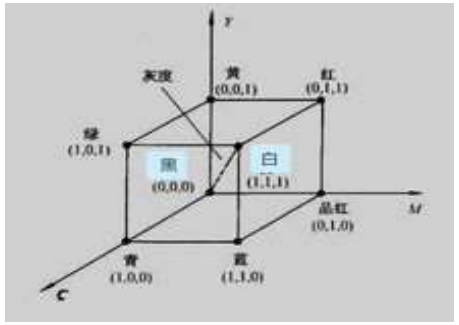


Raster Imaging System(cont.)

- 2.Frame Buffer帧缓存(cont.)
  - How colors are represente/coded ? (颜色如何编码表示?)
    - Color Model/Color Space（颜色模型/颜色空间）



- RGB Model/ RGB space:　　　　C=r[R]+g[G]+b[B]
  - 任何一种色光C都可由R、G、B三基色按不同的比例(r,g,b) 混合相加而表示（r+g+b=1）
    - 当三基色分量都为0.0时,混合为黑色（没有光）
    - 当三基色分量都为1.0时,混合为白色光。

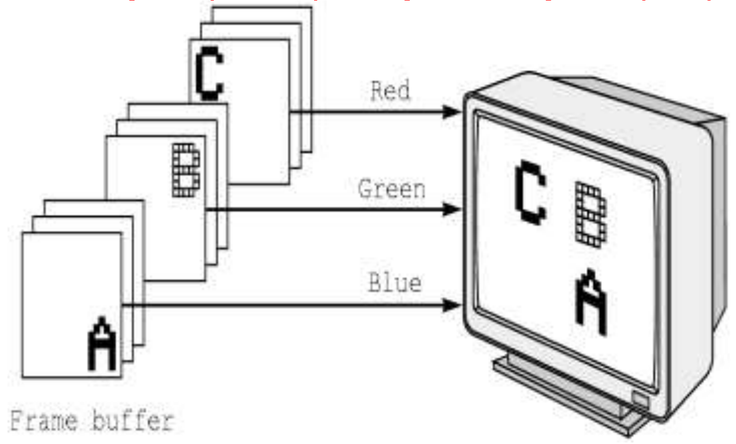


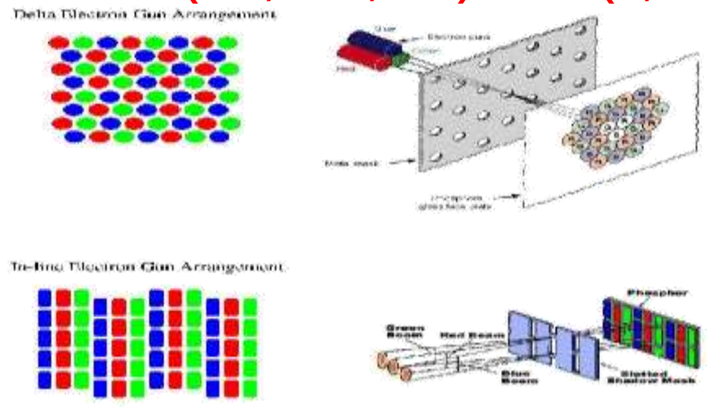
Raster Imaging System(cont.)

- ➢**2.Frame Buffer帧缓存(cont.)**
  - ➢**How colors are represente/coded ? (颜色如何编码表示？)**
    - ➢Color Model/Color Space（颜色模型/颜色空间）

□Color CRT using RGB Color Model 演示

- *Have 3 guns: Red, Green and Blue ,*
- *3 component and each color component is stored separately in the frame buffer, Usually 8 bits per component in buffer, Color values can range from 0.0 (none) to 1.0 (all) using floats or over the range from 0 to 255 using unsigned bytes*
  - **Red: (1.0, 0.0, 0.0) or (255, 0 . 0) ; Green: : (0.0, 1.0, 0.0) or (0, 255, 0)**




Delta Electron Gun Arrangement
Tri-line Electron Gun Arrangement
Phosphor
Green Beam
Red Beam
Blue Beam
Slotted Shadow Mask

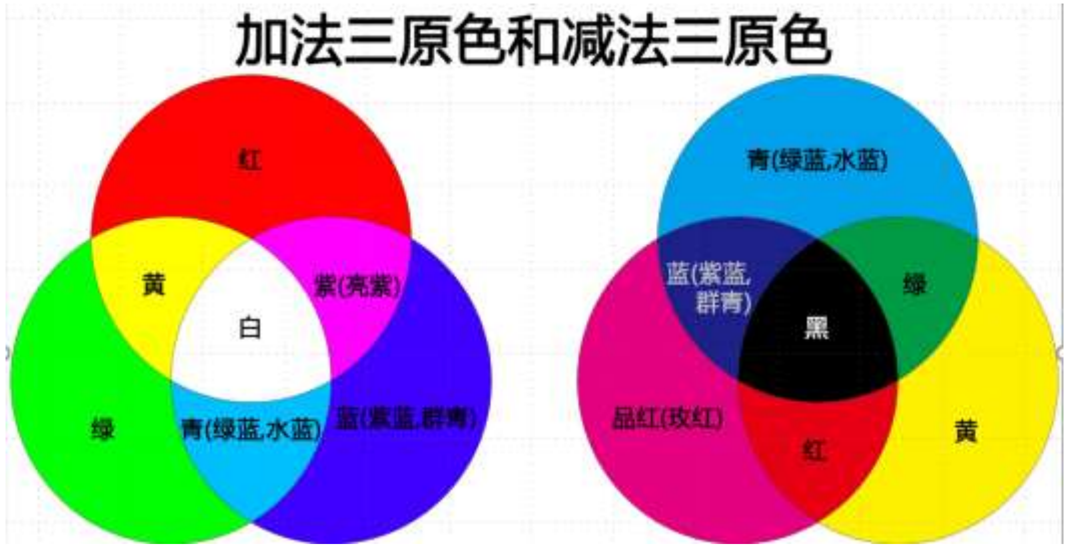
Raster Imaging System(cont.)

➢2.Frame Buffer帧缓存(cont.)

   ➢How colors are represente/coded？(颜色如何编码表示?）

      ➢Color Model/Color Space（颜色模型/颜色空间）

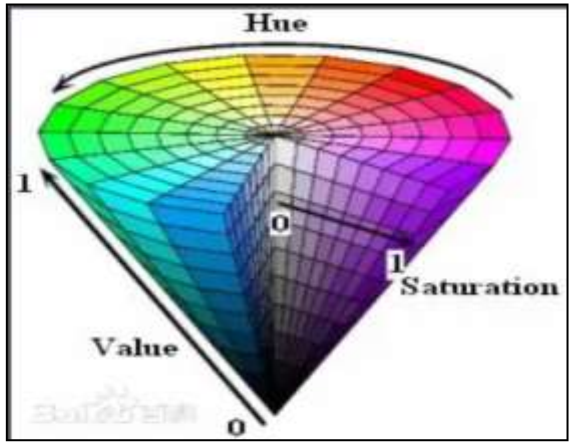
- Other Color Model
  - CMY model： Subtractive color
    - Form a color by filtering white light with cyan (C), Magenta (M), and Yellow (Y) filters
    - Light-material interactions, Printing（印刷）,Negative film(底片）
    - 白色(0,0,0)，黑色（1，1，1）

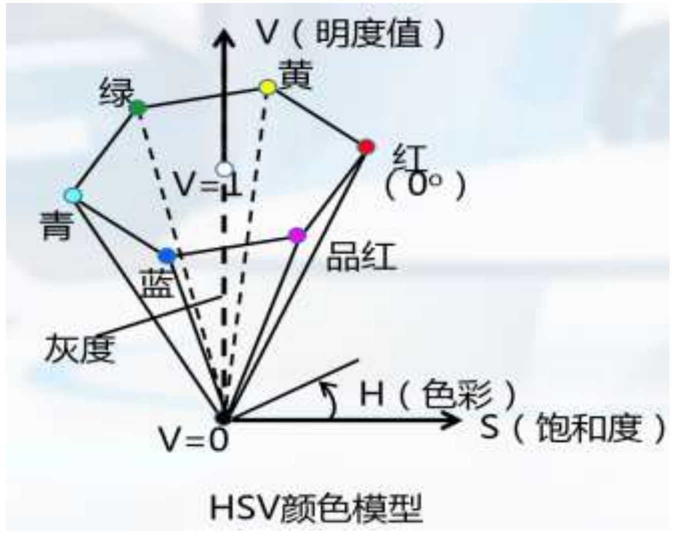


Raster Imaging System(cont.)

- **2.Frame Buffer帧缓存(cont.)**
  - **How colors are represente/coded？(颜色如何编码表示？）**
    - Color Model/Color Space（颜色模型/颜色空间）

- **Other Color Model**
  - HSV model: 色调Hue， 饱和度Saturation， 亮度Value
  - HSL/HSB/ HSI: Hue, Saturation, Lightness/ Luminance /Bright/ Intensity



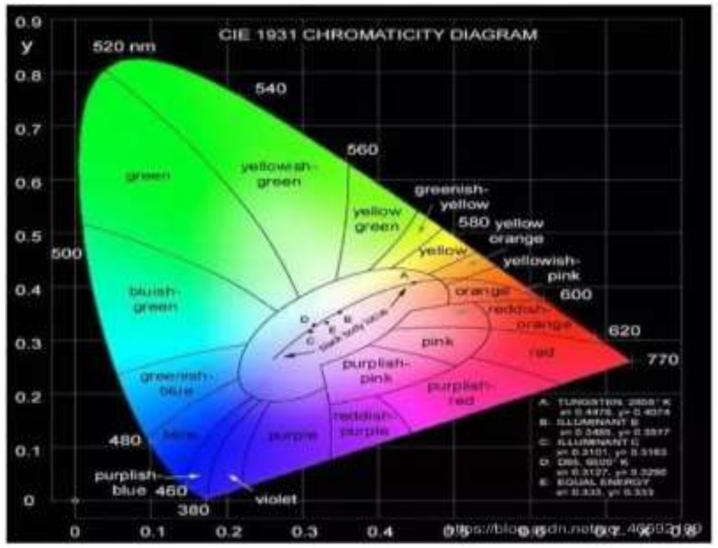


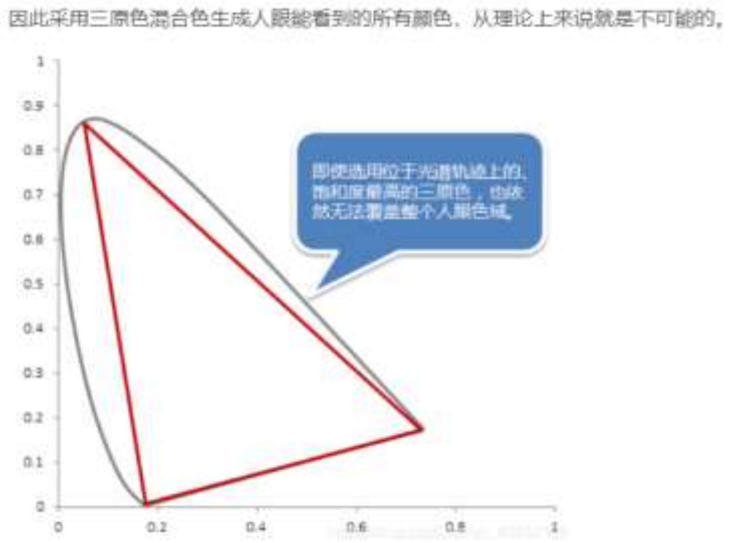
Raster Imaging System(cont.)

➢2.Frame Buffer帧缓存(cont.)

- ➢How colors are represente/coded？(颜色如何编码表示？）
  - ➢Color Model/Color Space（颜色模型/颜色空间）

- Other Color Model
  - XYZ Model：CIE色度图：
    - 坐标标注的是RGB三原色之间的比例，是相对大小，不是绝对大小。
    - 不包含明度信息，低明度的颜色在色度图中都没有。

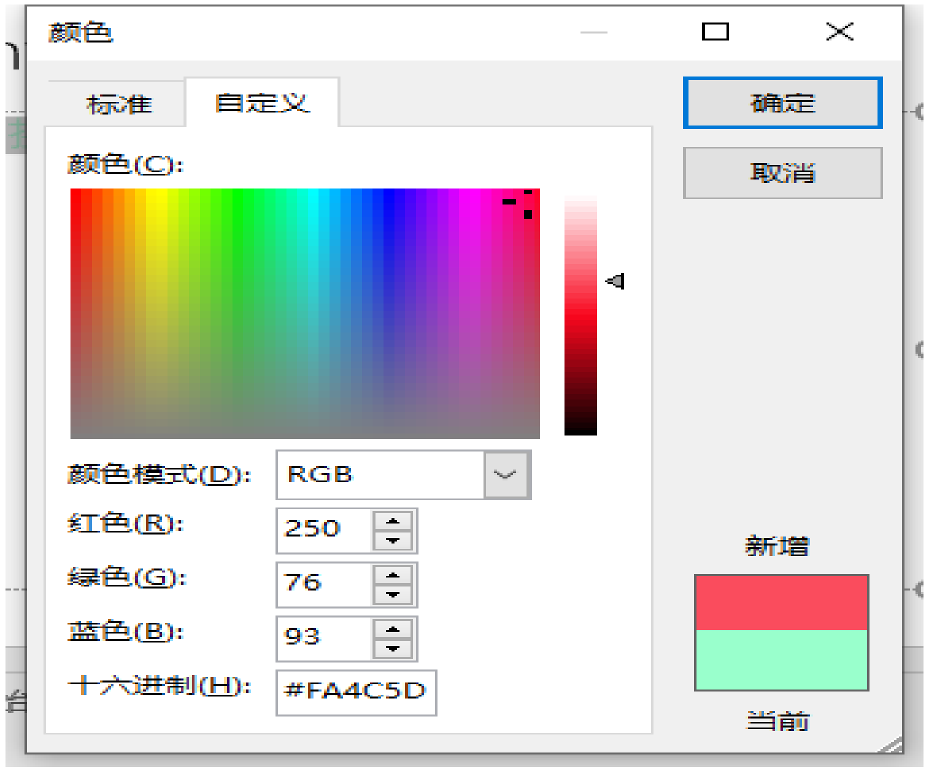



因此采用三原色混合色生成人眼能看到的所有颜色，从理论上来说就是不可能的。

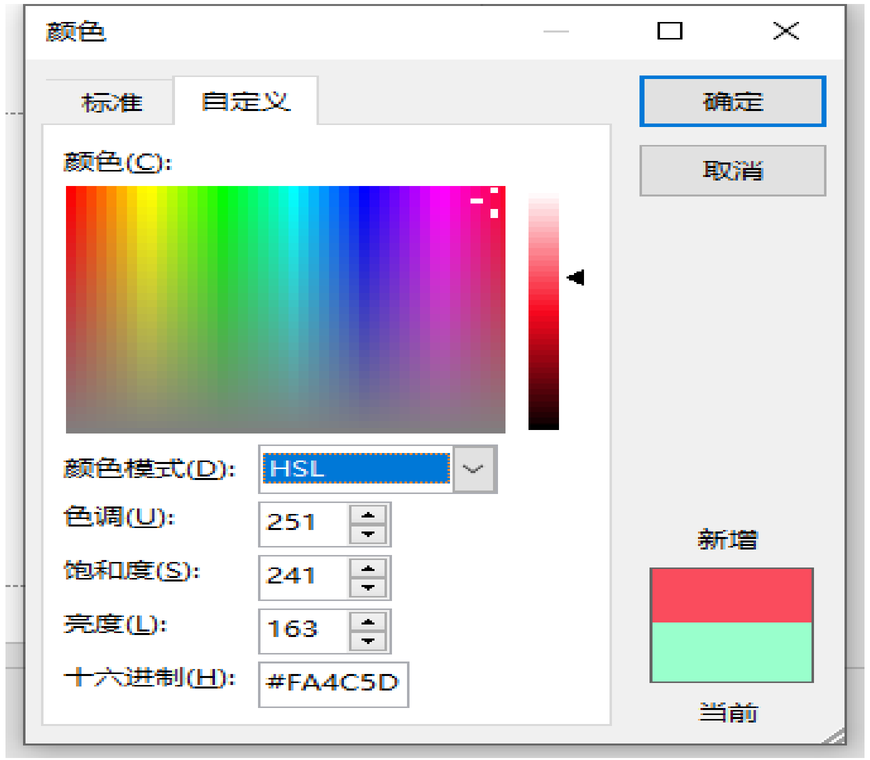
Raster Imaging System(cont.)

- 2.Frame Buffer帧缓存(cont.)
  - How colors are represente/coded？(颜色如何编码表示？）
    - Color Model/Color Space（颜色模型/颜色空间）

✓颜色模型表示之间可以相互转换： 数值范围：0..255



颜色模式(D): RGB
红色(R): 250
绿色(G): 76
蓝色(B): 93
十六进制(H): #FA4C5D



颜色模式(D): HSL
色调(U): 251
饱和度(S): 241
亮度(L): 163
十六进制(H): #FA4C5D

Raster Imaging System(cont.)

- **2.Frame Buffer帧缓存(cont.)**
  - **How colors are represented/coded ? (颜色如何表示/编码？）**
    - Color Model/Color Space（颜色模型/颜色空间）
  - **How much capacity does a frame image need（帧缓存需要多大容量?）**
    - **Pixel Depth(像素深度）**
    - **Framebuffer capacity(帧缓存容量）**
  - **How color codes are stored in framebuffer? （颜色编码如何存储?）**
    - Packed Pixel Method(组合像素法）
    - Bit Plane Method(位面法)
    - Color Lookup Table(CLT查色表）

Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- ➢How much capacity does a frame image need（帧缓存需要多大容量?）
  - ➢Pixel Depth(像素深度）/ Pixel Precision像素精度
  - ➢Framebuffer capacity(帧缓存容量）
- **Pixel Depth 像素深度 or Pixel Precision精度**
  - Defined as the number of bits that are used for each pixel,
  - **Pixel depth determines properties such as how many colors can be represented on a given system**
- **Framebuffer capacity** （帧缓存最小容量）
  - Pixel Depth * Resolution (bit)
  - Pixel Depth * Resolution / 8 (byte)

➢ **Different Luminance Images have Different Precision and Capacity**

- ☐Monochromatic image 单色图/黑白图
- ☐Gray levels image灰度图
- ☐Bit Colour 位色图像
- ☐Color Image 颜色图

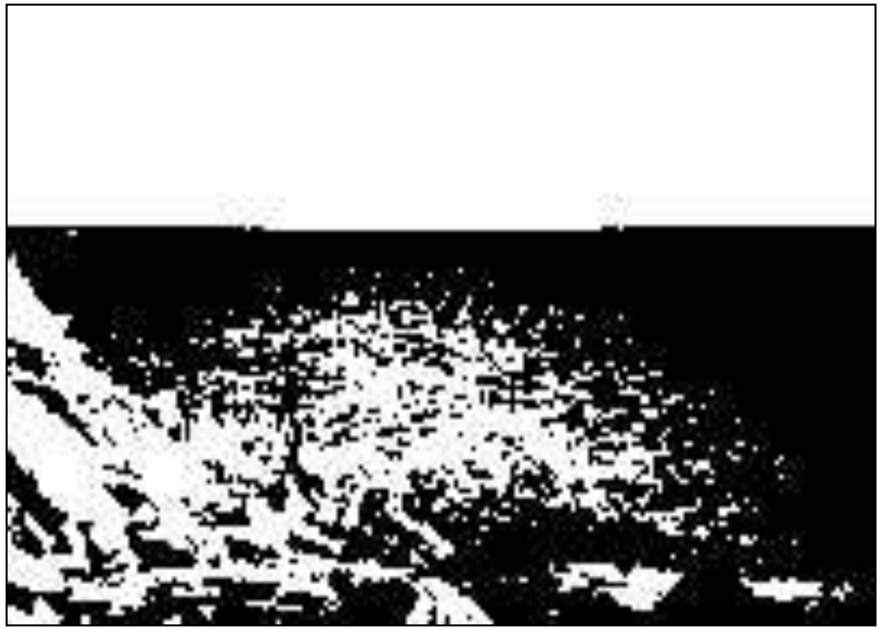
Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- ➢ How much capacity does a frame image need（帧缓存需要多大容量?）
  - ➢ Pixel Depth(像素深度）/ Pixel Precision像素精度
  - ➢ Framebuffer capacity(帧缓存容量）

☐Monochrome(黑白图像)： 1 bits Per Pixel

- ➢Each color component is represented by 1 bit.
- ➢pure **black** or pure **white**



Pixel Precision: 1
Supply Colors: 2^1=2

If Resolution is： 1024*768
Framebuffer Capacity: 1* 1024*768 bit

Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- **How much capacity does a frame image need（帧缓存需要多大容量?）**
  - **Pixel Depth(像素深度）/ Pixel Precision像素精度**
  - **Framebuffer capacity(帧缓存容量）**

Gray-scale Image灰度图像： 8 bits per pixel

No colours besides black, white and grey

Analogous to working with black and white film or television

Pixel Precision: 8

Supply Colors: : 2^8=256

If resolution is： 1024*768

Framebuffer Capacity: 8* 1024*768 bit

Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- How much capacity does a frame image need（帧缓存需要多大容量?）
  - Pixel Depth(像素深度）/ Pixel Precision像素精度
  - Framebuffer capacity(帧缓存容量）

- **Bit Colour位色图像: 8 bits per pixel**

  Each pixel may composes of red-3bit, green-3bit and blue-2bit.

  Pixel Precision: 8
  Supply Colors: 2^8=256

  If resolution is： 1024*768
  Framebuffer Capacity: 8* 1024*768 bit

Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- **How much capacity does a frame image need（帧缓存需要多大容量?)**
  - **Pixel Depth(像素深度）/ Pixel Precision像素精度**
  - **Framebuffer capacity(帧缓存容量)**

True color真彩色图像**: 24 bits per pixel**

- Each pixel composes of red, green and blue (3components).
- Each color component is represented by 1 byte (8 bits).

Pixel Precision: 8*3=24
Allows Colors : 2^24 =16M

If resolution is： 1024*768
Framebuffer Capacity: 24*1024*768 bit

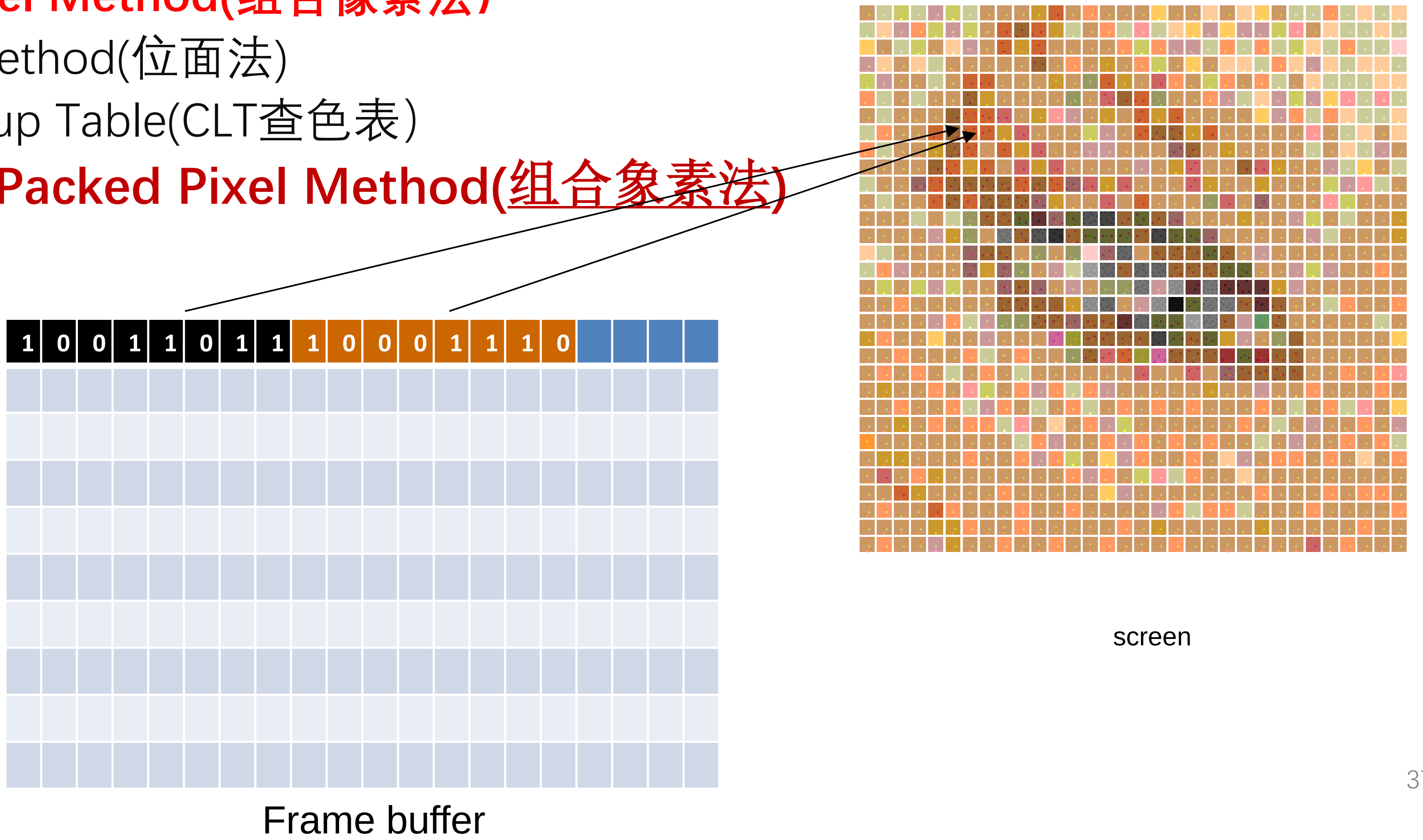
Raster Imaging System(cont.)

- 2.Frame Buffer帧缓存(cont.)
  - How colors are represented/coded ? (颜色如何表示/编码？)
    - Color Model/Color Space（颜色模型/颜色空间）
  - How much capacity does a frame image need（帧缓存需要多大容量?）
    - Pixel Depth(像素深度）
    - Framebuffer capacity(帧缓存容量）
  - How color codes are stored in framebuffer? （颜色编码如何存储?）
    - Packed Pixel Method(组合像素法）
    - Bit Plane Method(位面法)
    - Color Lookup Table(CLT查色表）

Raster Imaging System(cont.)

- 2.Frame Buffer帧缓存(cont.)
  - How color codes are stored in framebuffer? （颜色编码如何存储?）
    - **Packed Pixel Method(组合像素法)**
    - Bit Plane Method(位面法)
    - Color Lookup Table(CLT查色表)

□**Color Storage：Packed Pixel Method(组合象素法)**



Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

How color codes are stored in framebuffer? （颜色编码如何存储?

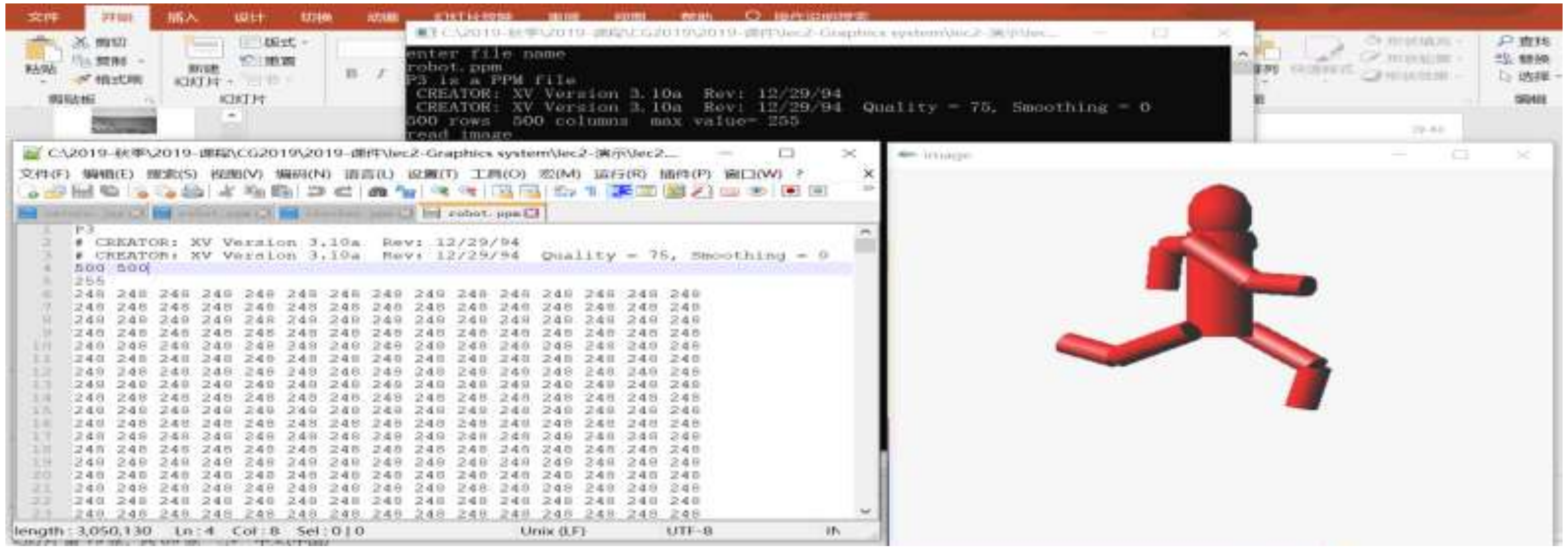
Packed Pixel Method(组合像素法）

Bit Plane Method(位面法)

Color Lookup Table(CLT查色表）

□Example: PPM file (Portable Pixelmap)

- **Resolution**:500*500
- **RGB Color Model**, 3 components and each occupy one byte, color value:0~255



```
enter file name
robot.ppm
P3 is a PPM file
 CREATOR: XV Version 3.10a  Rev: 12/29/94
 CREATOR: XV Version 3.10a  Rev: 12/29/94  Quality = 75, Smoothing = 0
500 rows  500 columns  max value= 255
read image
```

```
P3
# CREATOR: XV Version 3.10a  Rev: 12/29/94
# CREATOR: XV Version 3.10a  Rev: 12/29/94  Quality = 75, Smoothing = 0
500 500
255
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
248 248 248 248 248 248 248 248 248 248 248 248 248 248 248
```

length : 3,050,130   Ln : 4   Col : 8   Sel : 0 | 0   Unix (LF)   UTF-8   IN

文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
robot.ppm
image

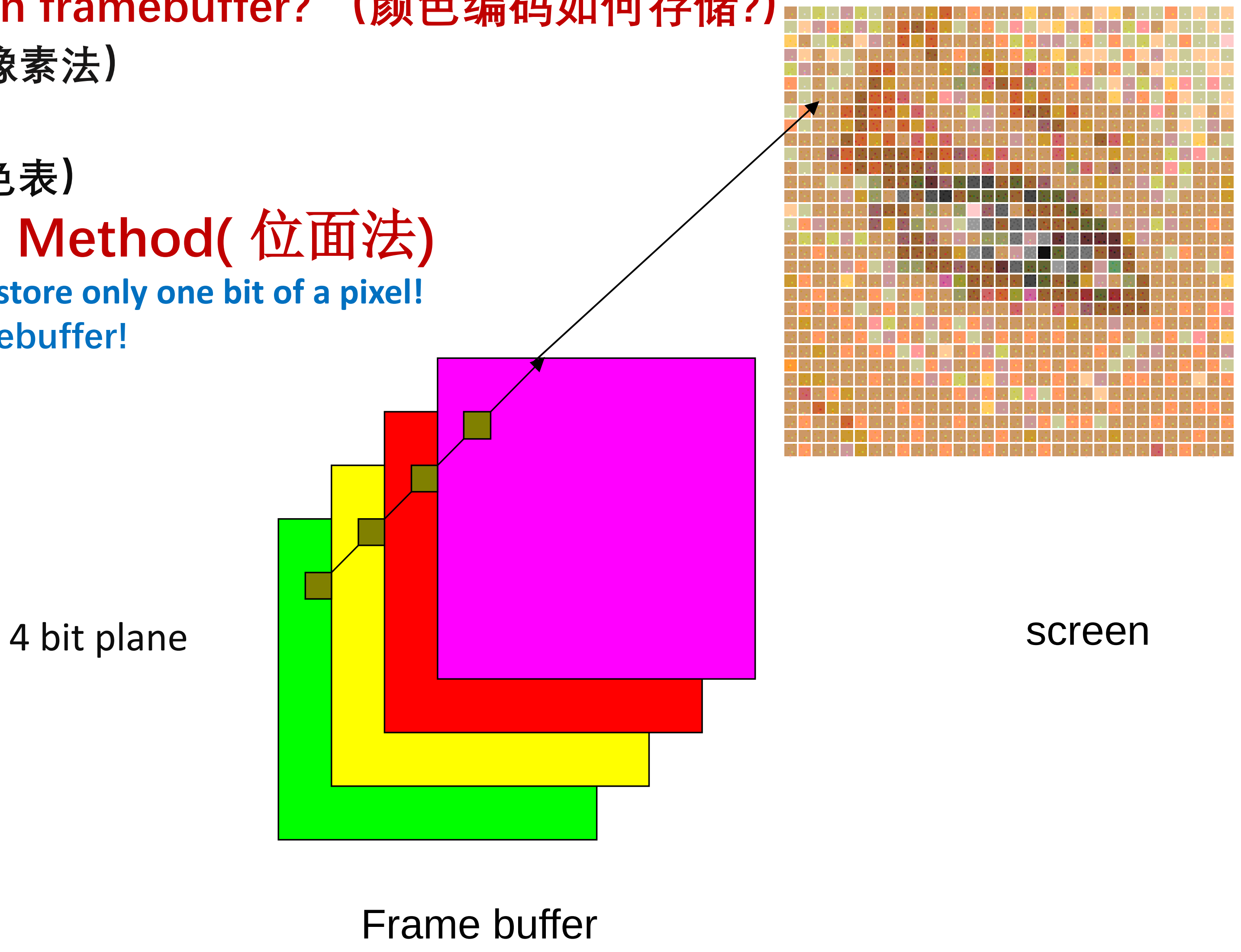
Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

- How color codes are stored in framebuffer? （颜色编码如何存储?）
  - Packed Pixel Method(组合像素法)
  - Bit Plane Method(位面法)
  - Color Lookup Table(CLT查色表)

□ Color Storage : Bit Plane Method(位面法)

- **bit plane(位面):One unit(bit plane) store only one bit of a pixel!**
- This Method is choosen for framebuffer!



Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

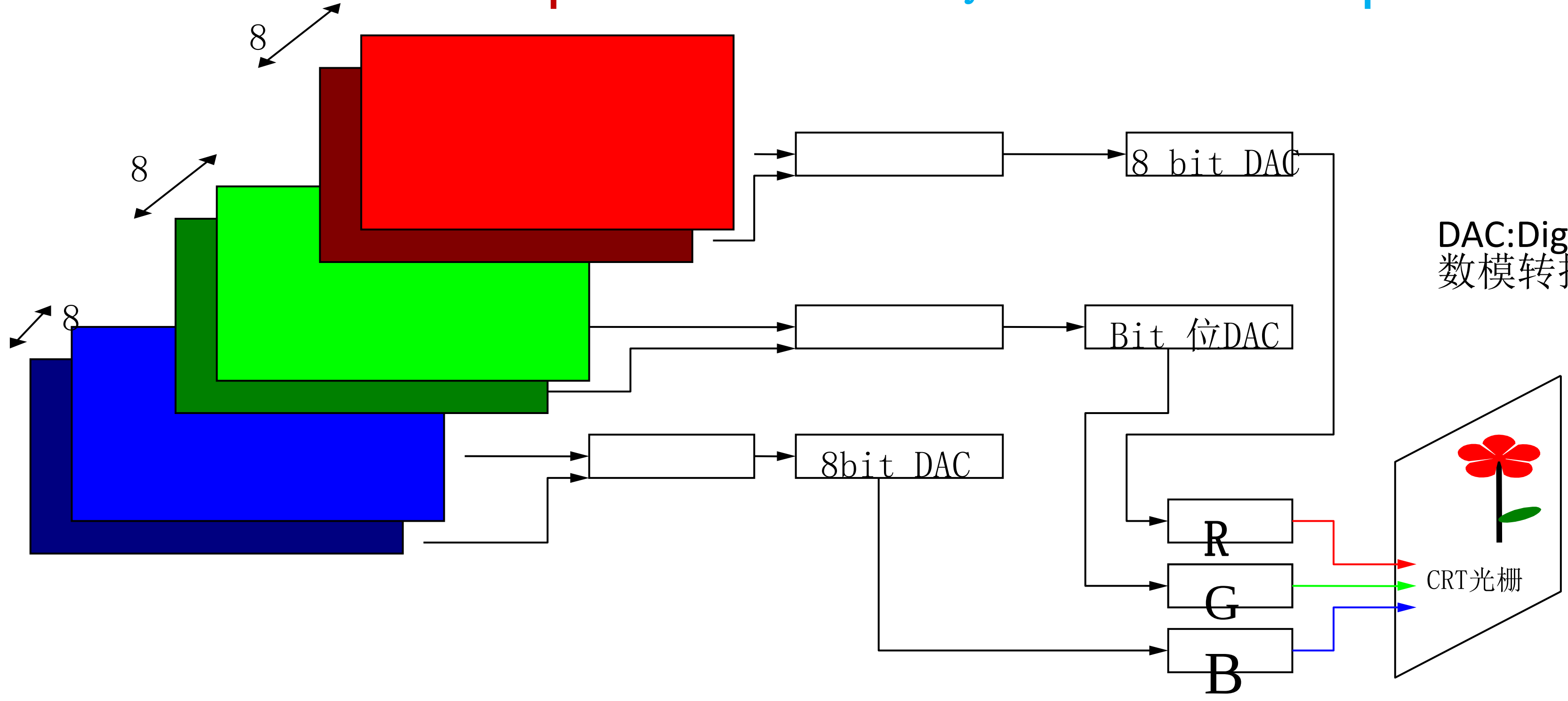
How color codes are stored in framebuffer? （颜色编码如何存储?

Packed Pixel Method(组合像素法）

Bit Plane Method(位面法)

Color Lookup Table(CLT查色表）

□Bit Plane Method Example : True color system has 24 bit planes to store color codes



DAC:Digital-to-Analog Converter
数模转换器：数字信号转换为模拟信号

Raster Imaging System(cont.)

2.Frame Buffer帧缓存(cont.)

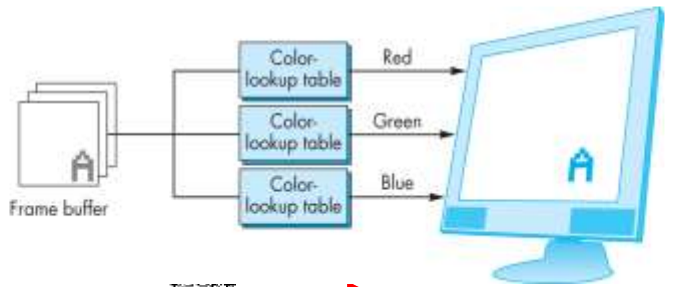
How color codes are stored in framebuffer? （颜色编码如何存储?)

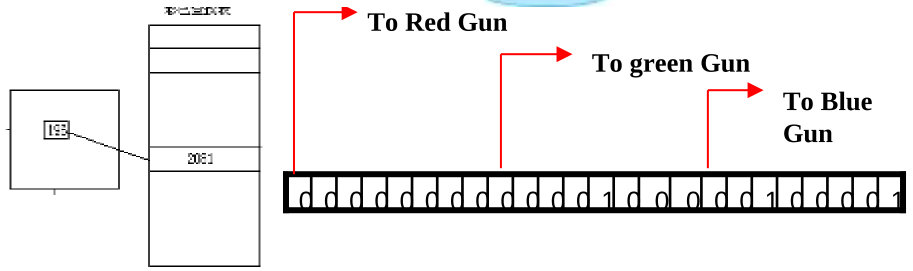
Packed Pixel Method(组合像素法)

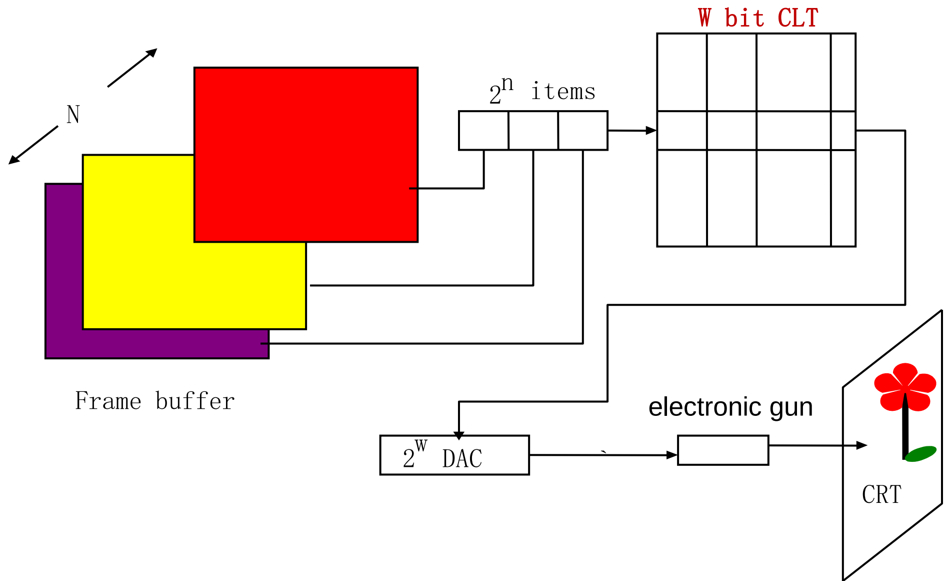
Bit Plane Method(位面法)

Color Lookup Table(CLT查色表)

- 帧缓存存查色表的索引地址，真正的颜色编码RGB存储在查色表CLT中
  - Framebuffer store Indices（索引）: Indices usually 8 bits,
  - CLT store true color：has 256 items and each item has 24bit color code.



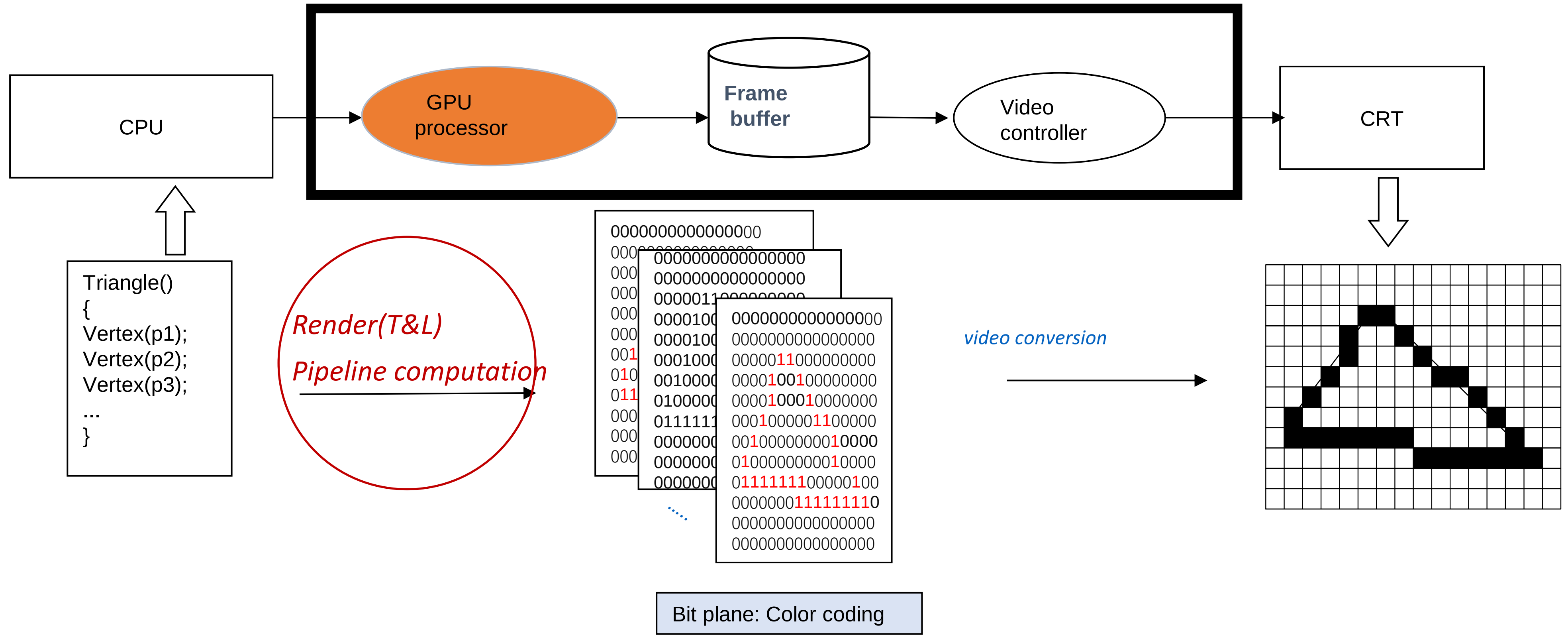




Raster Imaging System(cont.)

3.显示处理器GPU

A graphics processing unit (GPU) is a specialized electronic circuit designed to manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device.



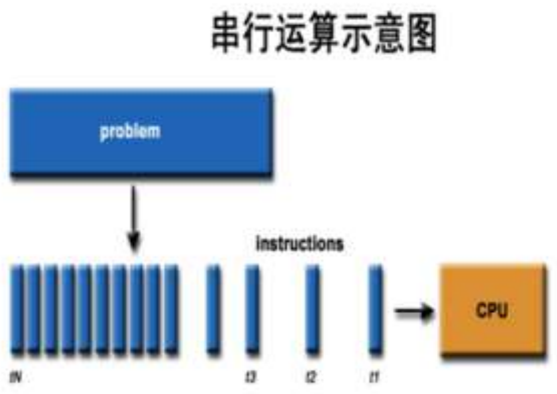
Raster Imaging System(cont.)

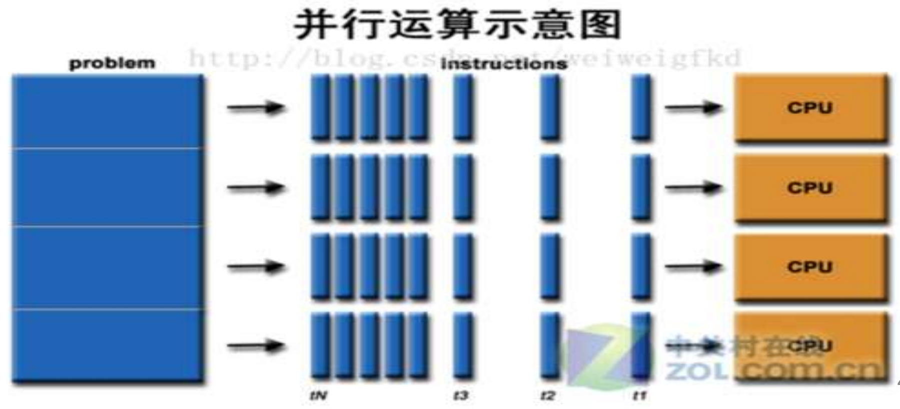
3.显示处理器GPU(cont.)

➢ 并行计算： 以“相同方式”处理“数量巨大”的顶点和像素:

- CPU的单个强核“串行”处理每个数据（顶点或像素）, 进行数据间控制和串行计算.
- GPU的多个弱核“并行”处理每个数据（顶点或像素）， “流”内任意元素的计算不依赖于其它同类型数据（如：计算一个顶点的世界坐标位置，不依赖于其他顶点的位置）

- 举例类比：餐馆的外卖采用“多个小摩托车”运 比 “单个大货车运” 运 要快，客户等候时间短
  - CPU型餐馆用一辆大货车送货，每次可以拉很多外卖，但是送完一家才能到下一家送货，每个人收到外卖的时间必然很长；
  - 而GPU型餐馆用多辆小摩托车同时送货，每辆车送出去的不多，但是并行处理的效率高点餐之后收货就会比大货车快很多

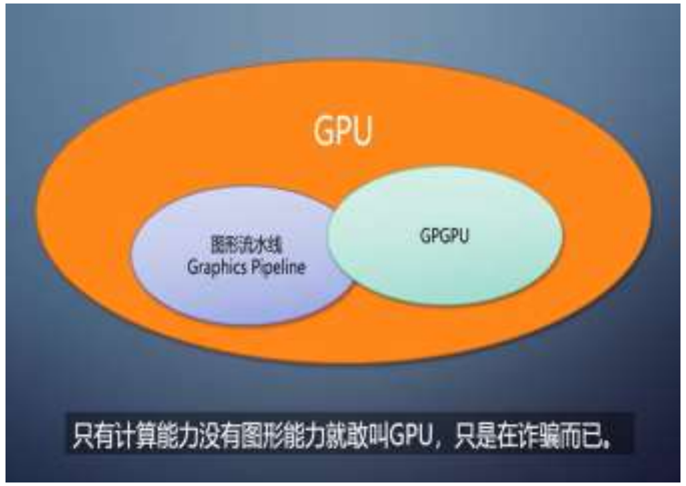




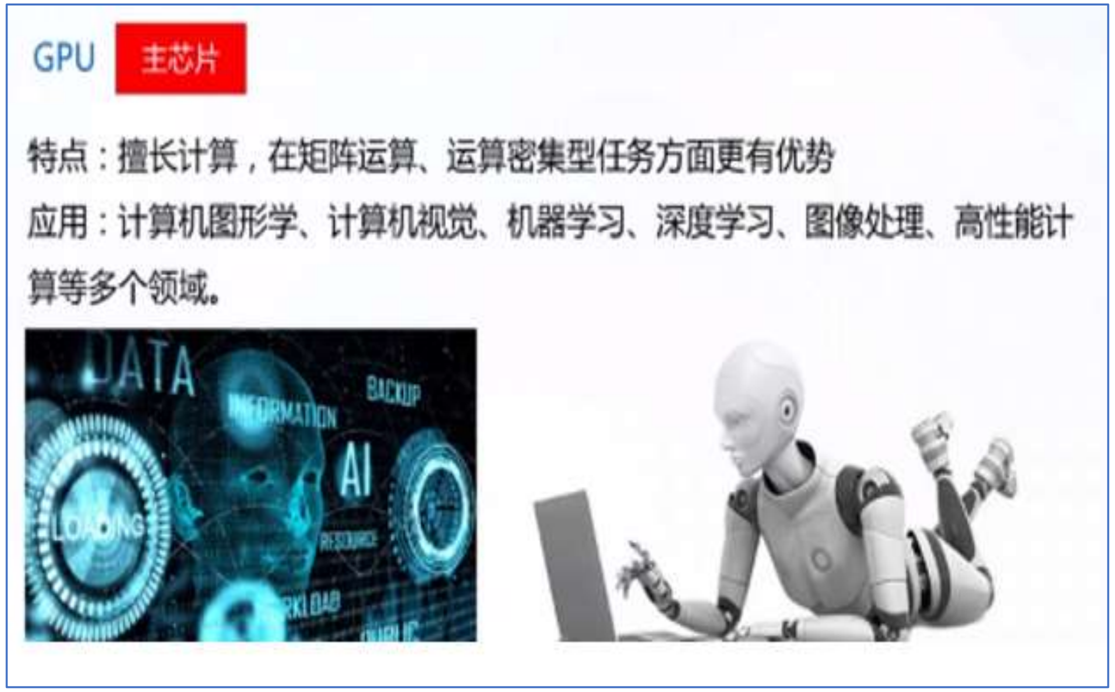
Raster Imaging System(cont.)

3.显示处理器GPU(cont.)

- 郭敏敏: “上帝视角看GPU”，参: https://www.bilibili.com/video/BV1P44y1V7bu/?spm_id_from=333.788.recommend_more_video.0&vd_source=c93f9d5c2c6ee8043fe0db22203390ee



只有计算能力没有图形能力就敢叫GPU，只是在诈骗而已。



特点：擅长计算，在矩阵运算、运算密集型任务方面更有优势

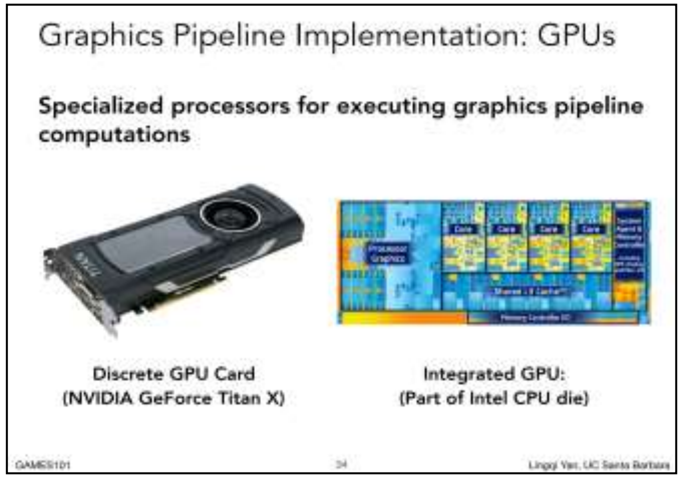
应用：计算机图形学、计算机视觉、机器学习、深度学习、图像处理、高性能计算等多个领域。

Raster Imaging System(cont.)

3.显示处理器GPU(cont.)

➢ **显卡：集成显卡，独立显卡**

- NVIDIA（英伟达）显卡发展史1999~2022” 参https://www.youtube.com/watch?v=as-aVVm9JZI
  - 1999,Geforce256， NVIDIA(英伟达公司）首次推出GPU
  - 2003, Geforce5800 Ultra, shader programming!
  - 2018, RTX 2080,Turing GPU, flagship: support real-time ray tracing in games! Using DLSS to scale to 4K, A hybrid rasterization and ray traced demo, …….
  - 2023, Racer RTX, starting “generative AI era”, (ref: SIGGRAPH2023)


Graphics Pipeline Implementation: GPUs

Specialized processors for executing graphics pipeline computations

Discrete GPU Card
(NVIDIA GeForce Titan X)

Integrated GPU:
(Part of Intel CPU die)

GAMES101 34 Lingqi Yan, UC Santa Barbara

Keynote Event

观看黄仁勋在 SIGGRAPH 的现场演讲

独家了解我们的一些最新技术，包括屡获殊荣的研究、通用场景描述 (OpenUSD) 开发以及最新的 AI 内容创建解决方案。

游戏

全新 DLSS 3.5 借助 AI 提升光线追踪

全新光线重建技术利用 AI 生成更多像素，从而为所有 GeForce RTX GPU 提升画质。

Computer Graphics System Outlines

- Input devices
- Output device
- Raster Imaging System-光栅成像系统*

- **Image Formation Principle –成像原理**
- Rendering Pipeline –渲染管线实现*
- The Programmer’s Interface-程序员编程接口

Image Formation Principle

- 需要作哪些操作才能生成一帧画面？（如何渲染rendering？）

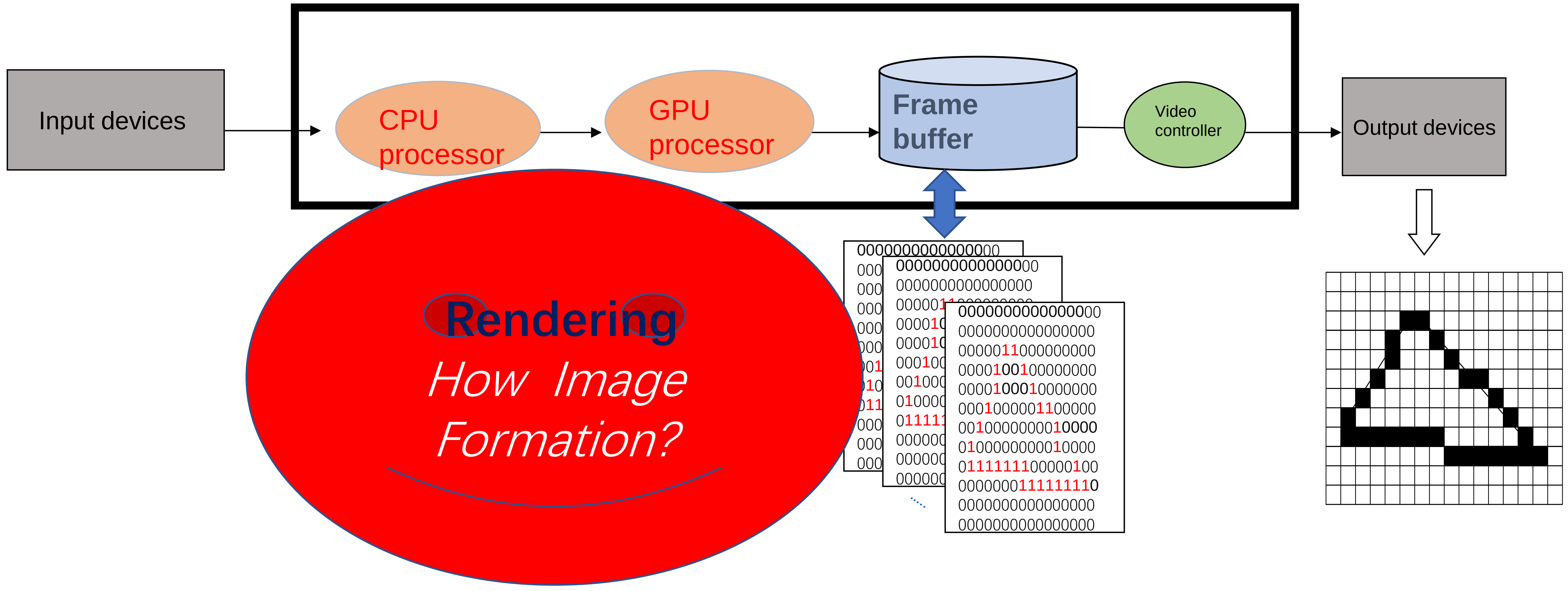
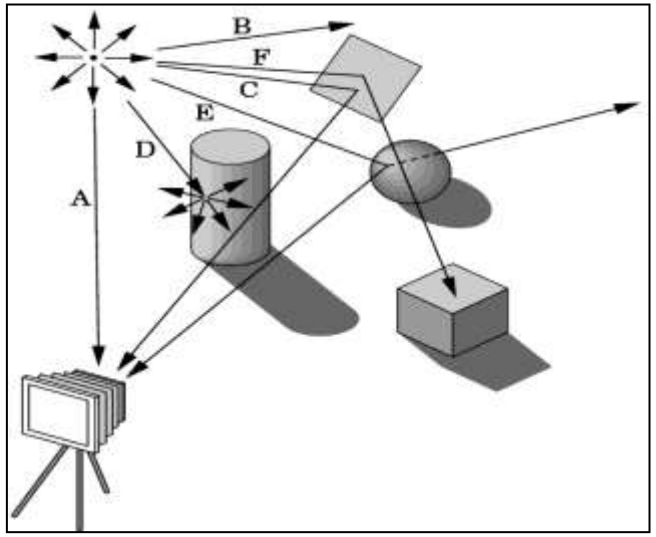


Image Formation Principle(cont.)

Three Elements of Image Formation:

- ➢**Objects（物体对象）**
- ➢**Light sources（光源）**
- ➢**Viewer/Camera（观察者/相机）**



■Scene场景由<u>对象</u>和<u>光源</u>构成，同一场景因<u>观察者</u>不同而成像不同。

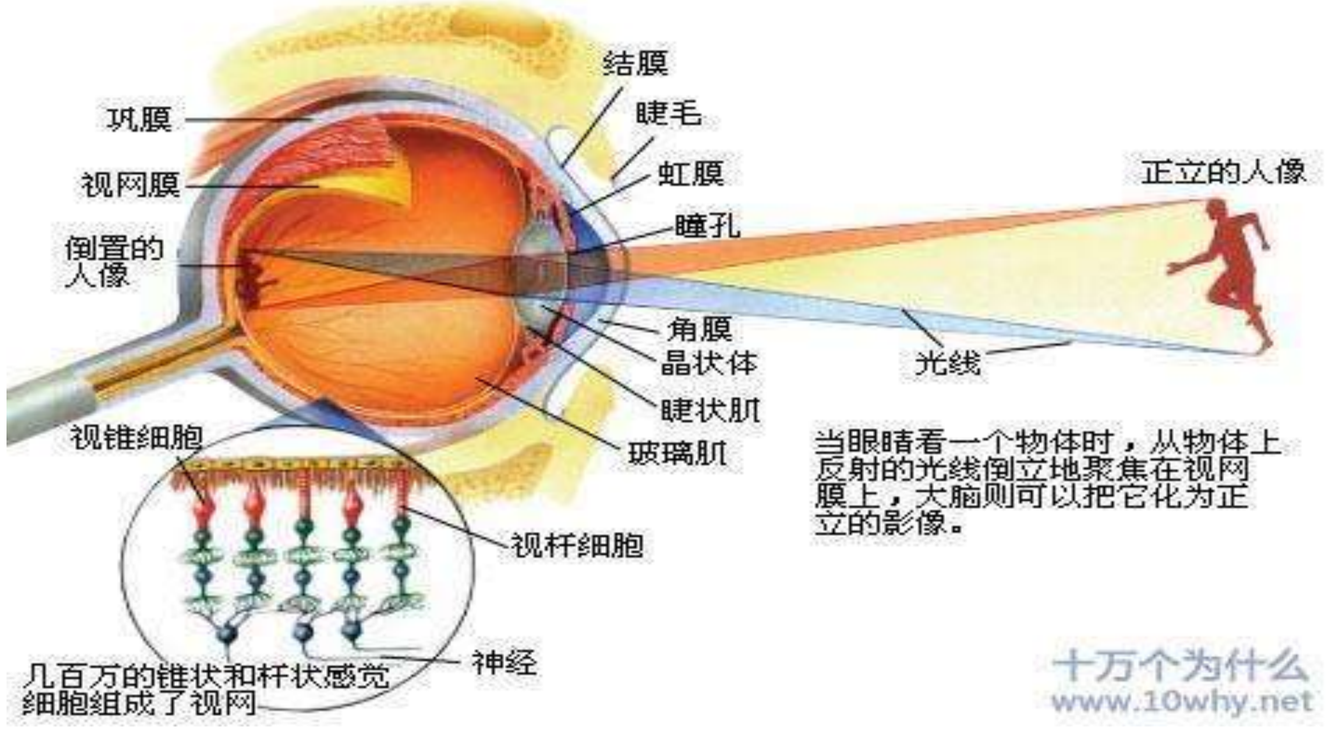
Image Formation Principle

1. **Nature Image System:** Human visual system

receiving light and transforming it into electrical energy light reflects from objects.

物体表面光线进入“人眼”，像是人眼对进入视神经的信号的主观感知！

有“晶状体”，形成有“景深”感（depth of field）的“倒”像，靠感知位“正”！

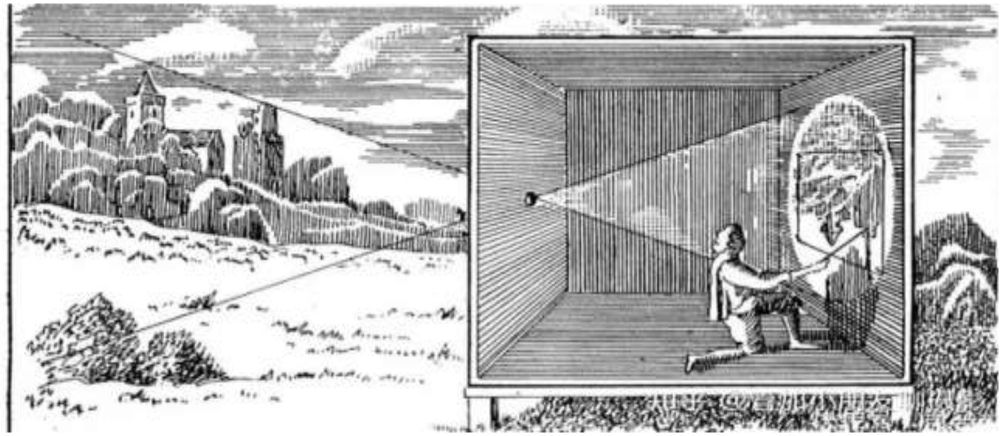


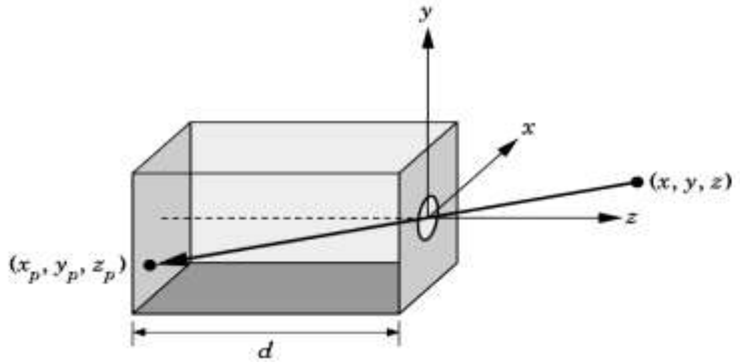
当眼睛看一个物体时，从物体上反射的光线倒立地聚焦在视网膜上，大脑则可以把它化为正立的影像。

Image Formation Principle(cont.)

2.Pinhole Image Formation(小孔成像）

- Use trigonometry（三角法） to find projection（xp,yp,zp） of point at (x,y,z)，These are equations of simple perspective （简单透视）
- 物体表面光线进入穿过小孔，在投影平面成“倒”像，“无景深”效果！



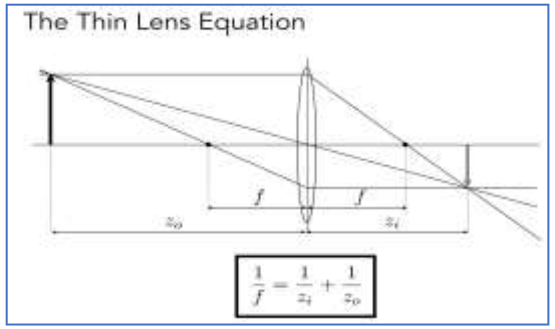


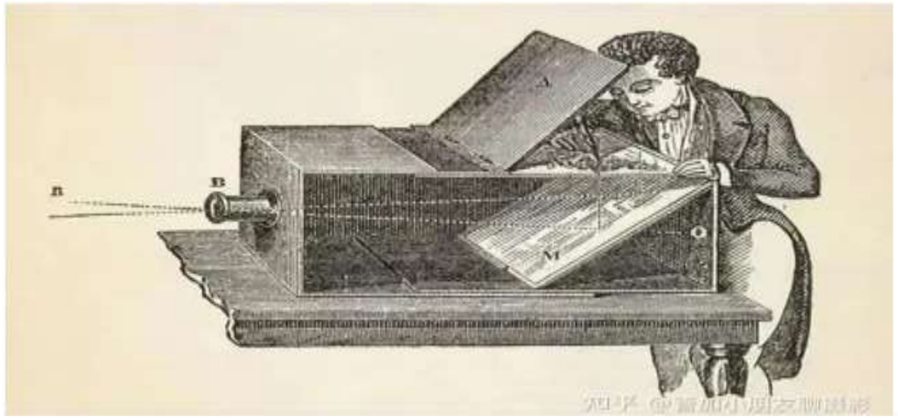
$x_p = -x/z/d$ $y_p = -y/z/d$ $z_p = d$

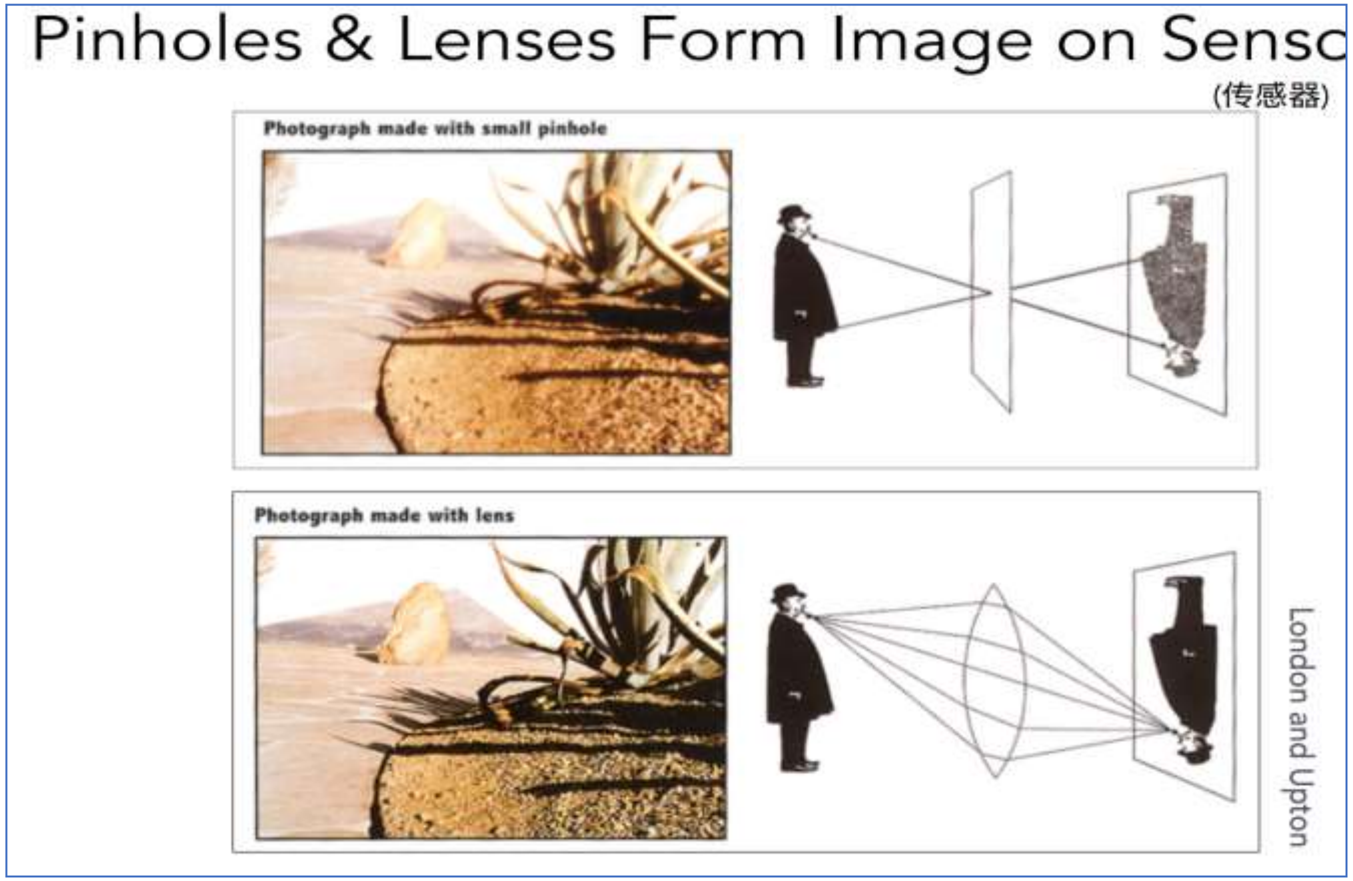
Image Formation Principle(cont.)

3.Camera imaging system(相机成像)

- 有“棱镜”， 产生有“景深”效果的“倒”像
  - *Ref: GAMES101_Lecture_19 Cameras, Lenses and Light Fields*






Pinholes & Lenses Form Image on Senso
(传感器)
Photograph made with small pinhole
Photograph made with lens
London and Upton

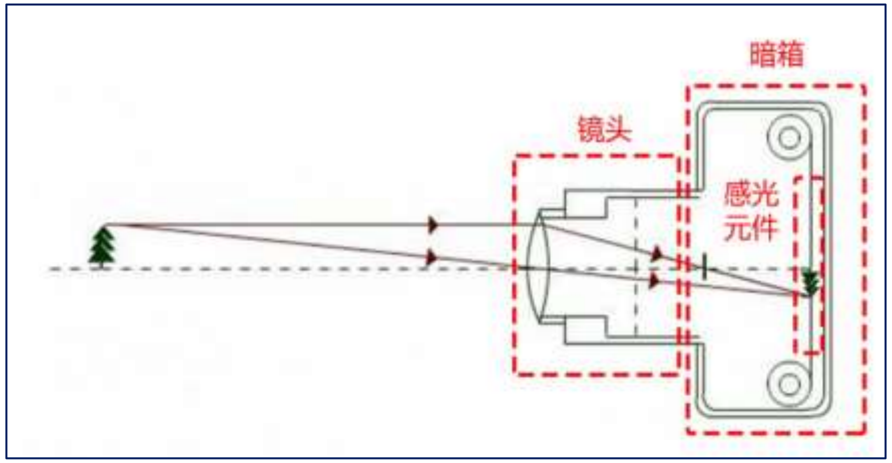
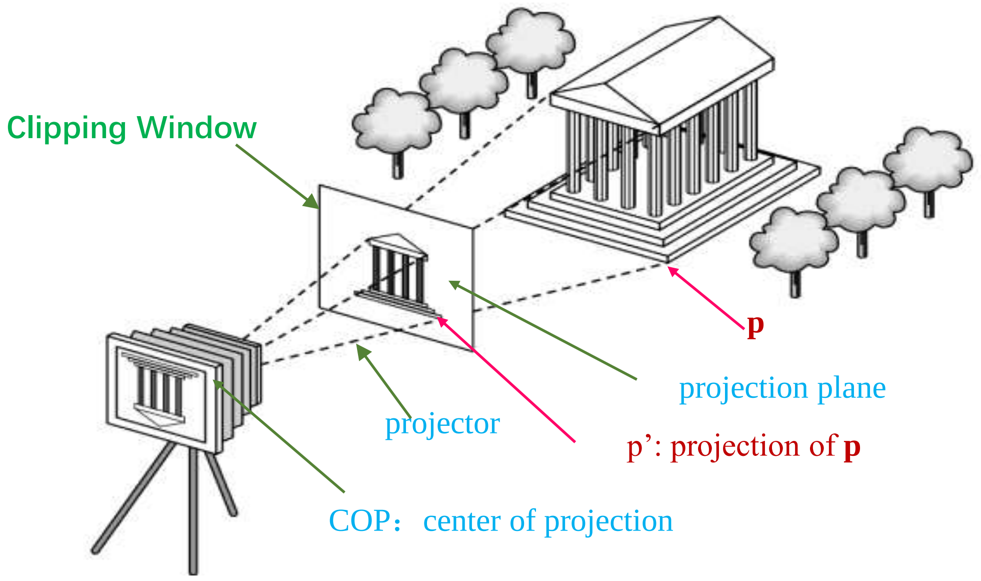


Image Formation Principle(cont.)

4.Synthetic Camera Model 虚拟相机模型/合成相机模型

- 1.对象，光源不依赖于观察者(相机），三要素相互独立。
- 2.可以如针孔相机一样，利用简单的几何方法计算图像（**无景深**）
- 3.为了达到**正图像**，我们在透镜（投影中心点COP)前面设另一个平面作为投影面(Project Plane)
- 4.**图像大小是受限的**，通过在投影平面内设置一个裁剪矩形(裁剪窗口)加以限制



- 实际计算机实现时采用这种模型，并且通过技术实现“景深”效果。

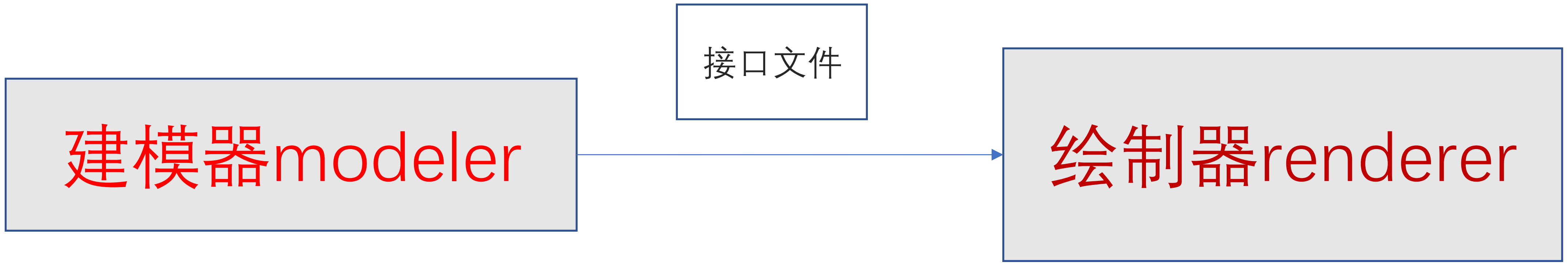
Computer Graphics System Outlines

- Input devices
- Output device
- Raster Imaging System-光栅成像系统*

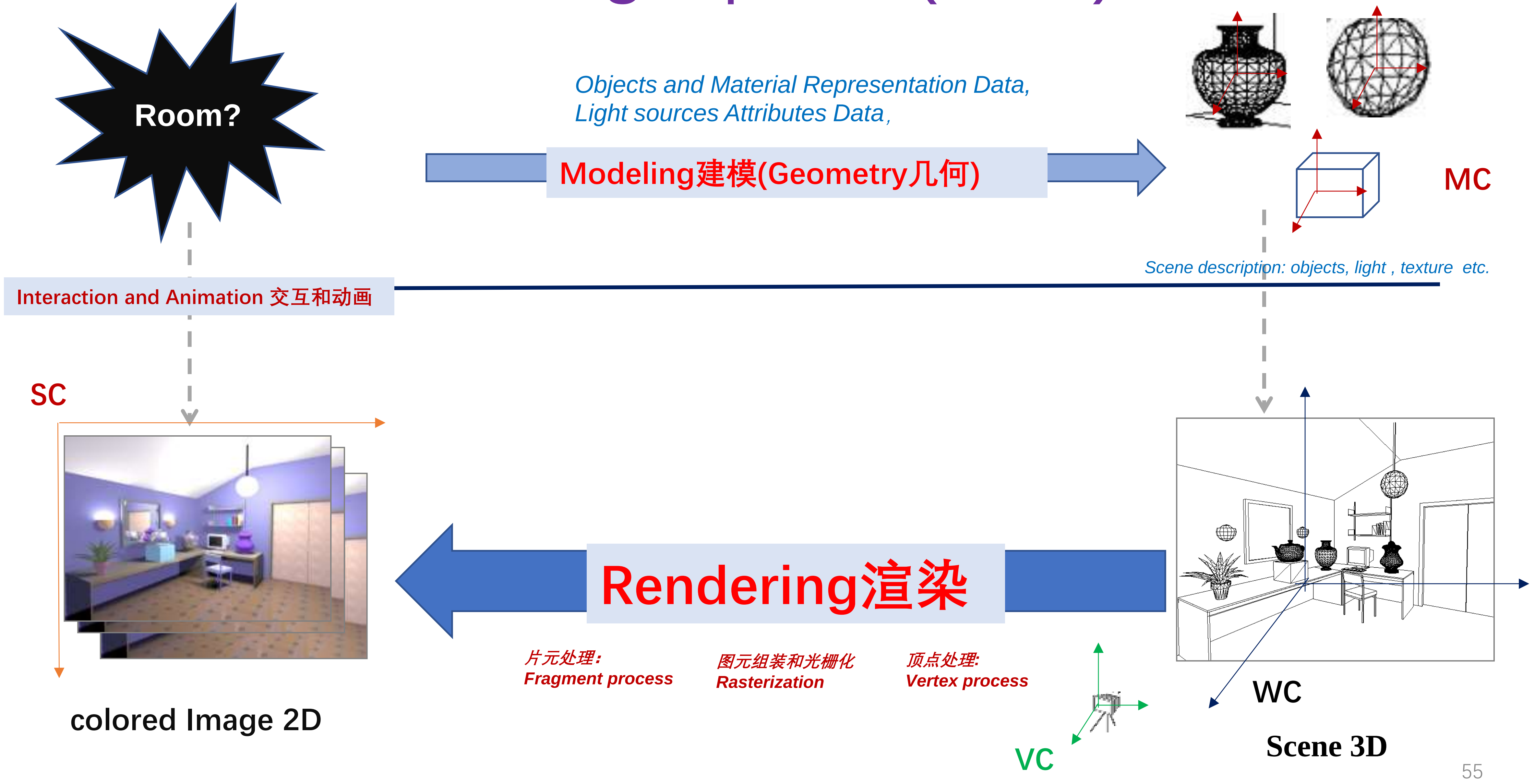
- **Image Formation Principle –成像原理**
- Rendering Pipeline –渲染管线*
- The Programmer’s Interface-程序员编程接口

Rendering Pipeline

- 用计算机绘图，主要两部分：
  - ➢建模(modeling)：
    - 专用软件(Maya,3Dmax, Blender…)生成各种复杂几何模型，光源模型，纹理对象等，生成场景scene中对象的参数文件（如obj文件）
    - 自己编写代码生成几何模型参数
  - ➢渲染rendering (T&L: Transformation & Lighting)
    - 基于虚拟相机模型，根据三要素（场景物体，光源，相机）参数，采用某种渲染方法，一步一步管线生成一帧画面数据，即完成绘制（渲染/烘培）。

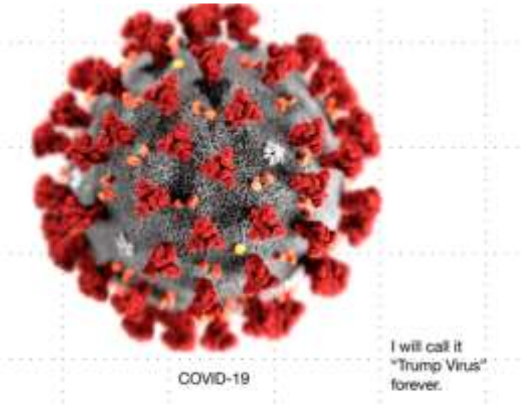


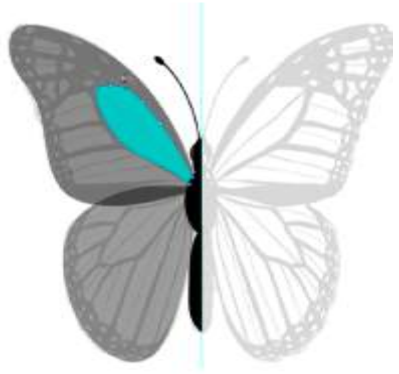
Rendering Pipeline(cont.)

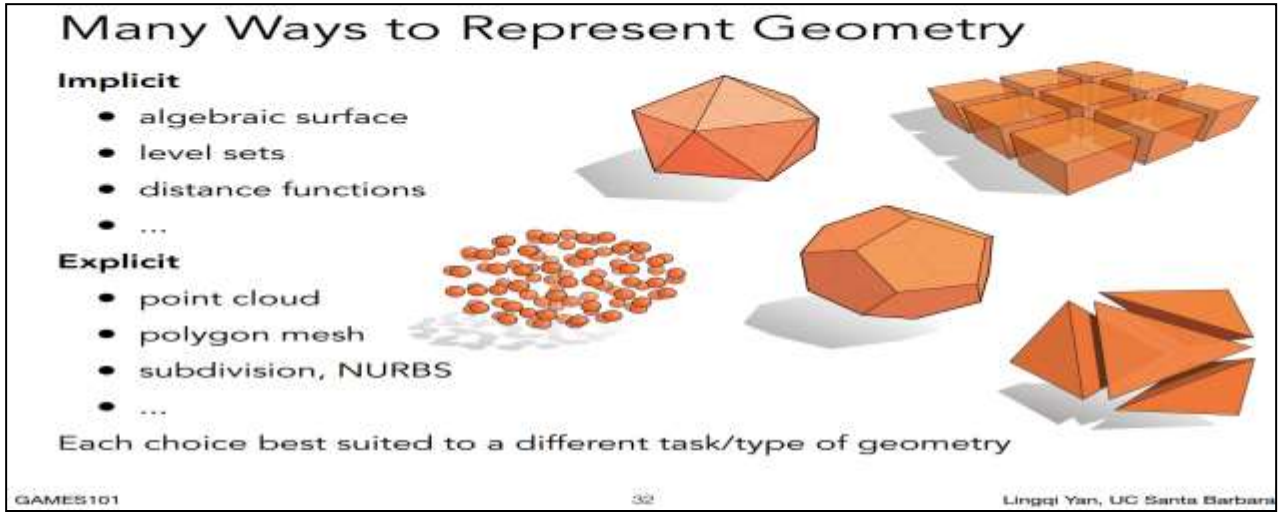

Room?
Objects and Material Representation Data,
Light sources Attributes Data,
Modeling建模(Geometry几何)
MC
Scene description: objects, light , texture etc.
Interaction and Animation 交互和动画
SC
Rendering渲染
片元处理:
Fragment process
图元组装和光栅化
Rasterization
顶点处理:
Vertex process
VC
WC
colored Image 2D
Scene 3D

Rendering Pipeline(cont.)

Geometry/Modeling： *ref:GAME101 geometry*







Many Ways to Represent Geometry

Implicit

- algebraic surface
- level sets
- distance functions
- ...

Explicit

- point cloud
- polygon mesh
- subdivision, NURBS
- ...

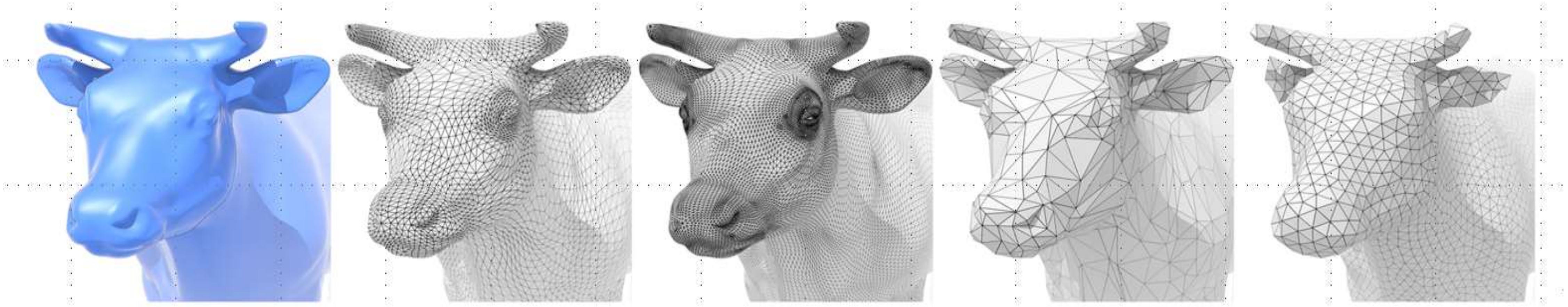
Each choice best suited to a different task/type of geometry

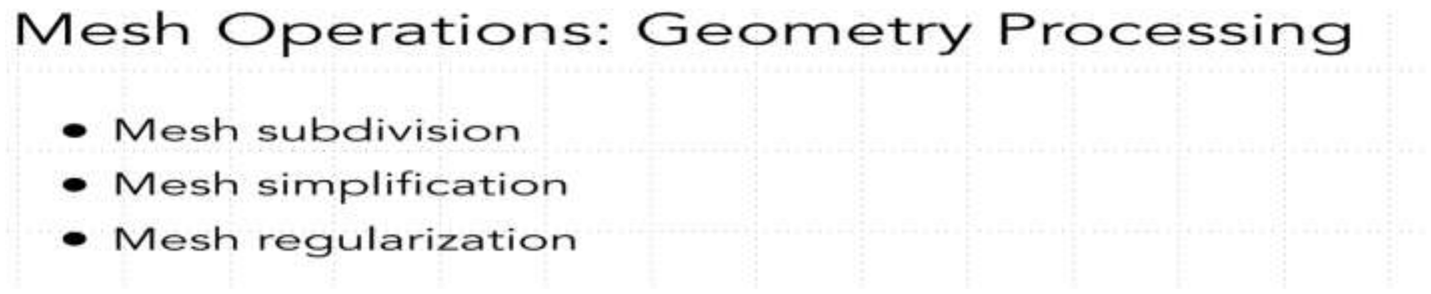
GAMES101 32 Lingqi Yan, UC Santa Barbara

Rendering Pipeline(cont.)

➢**Geometry/Modeling(cont.)**

Triangular Mesh 三角形网格（普遍采用的物体表面表示法）



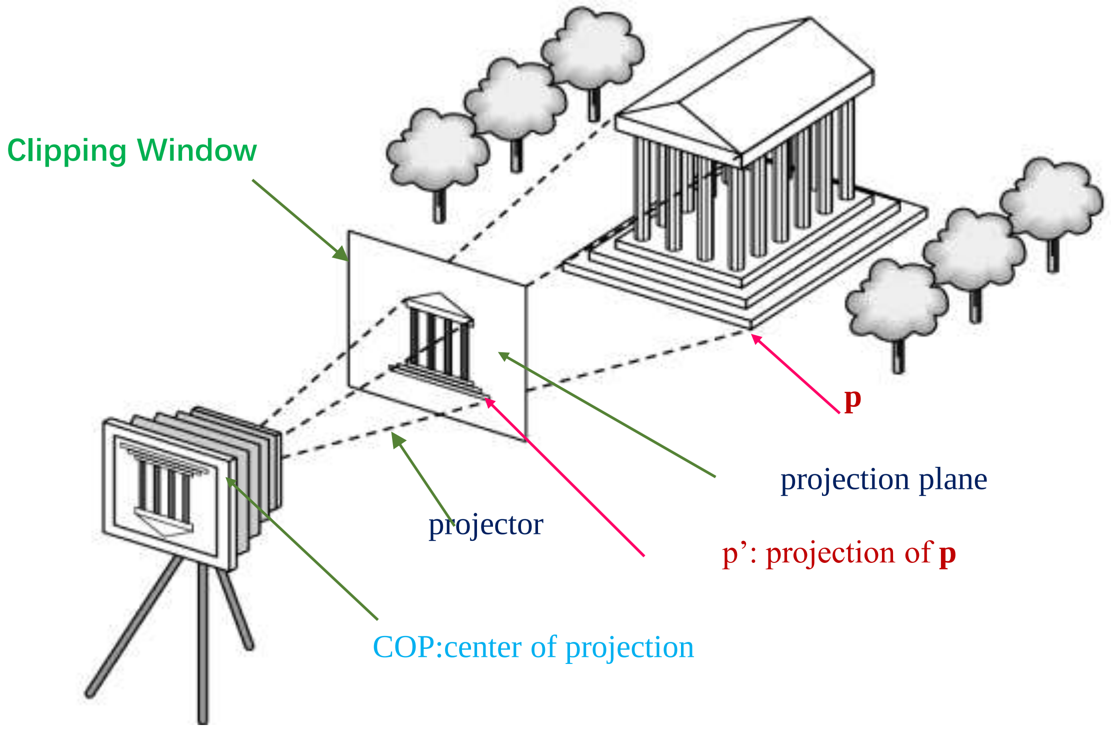


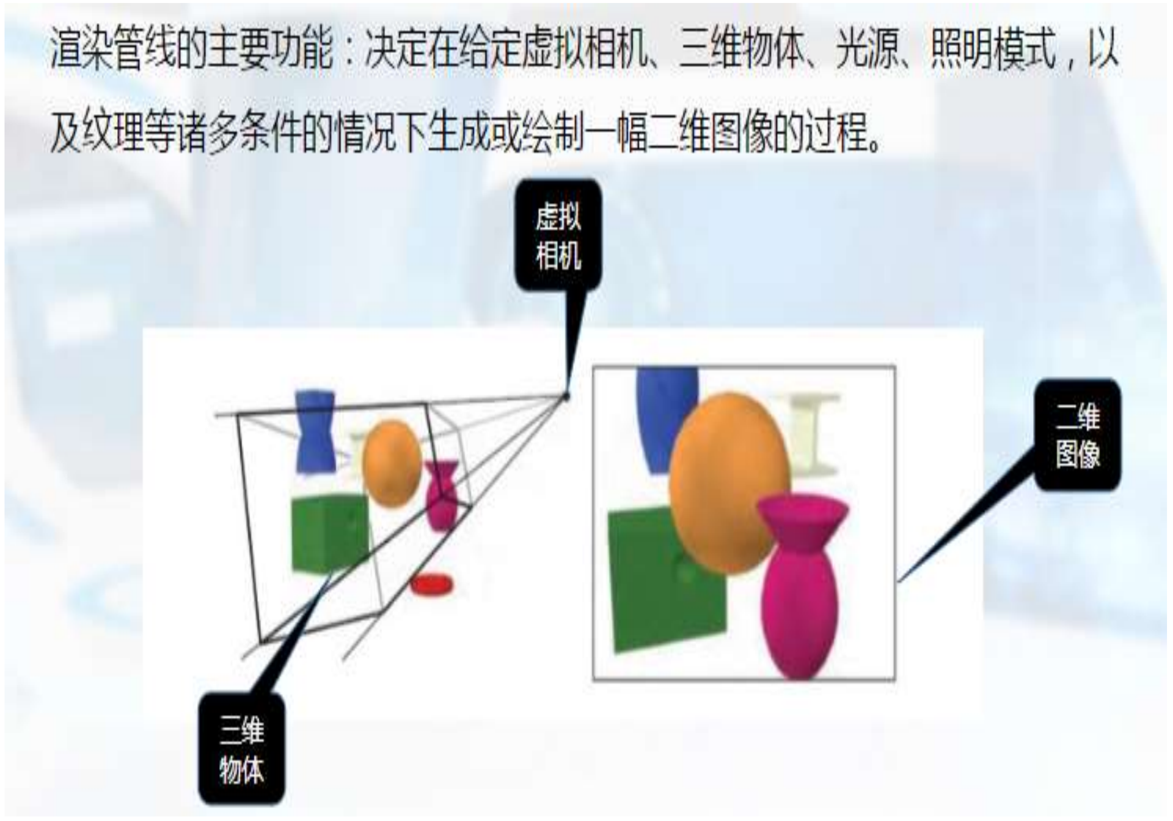
Mesh Operations: Geometry Processing

- Mesh subdivision
- Mesh simplification
- Mesh regularization

Rendering Pipeline(cont.)

➢**Rendering（渲染）**





渲染管线的主要功能：决定在给定虚拟相机、三维物体、光源、照明模式，以及纹理等诸多条件的情况下生成或绘制一幅二维图像的过程。

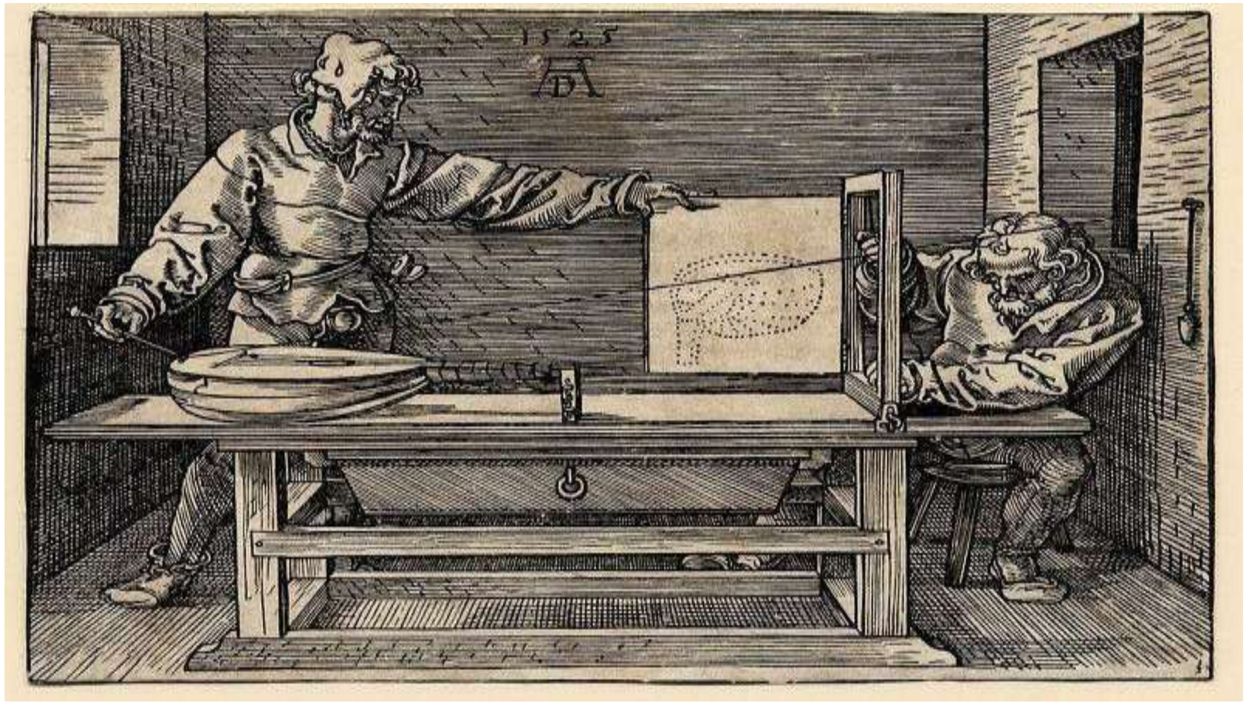
Rendering Pipeline(cont.)

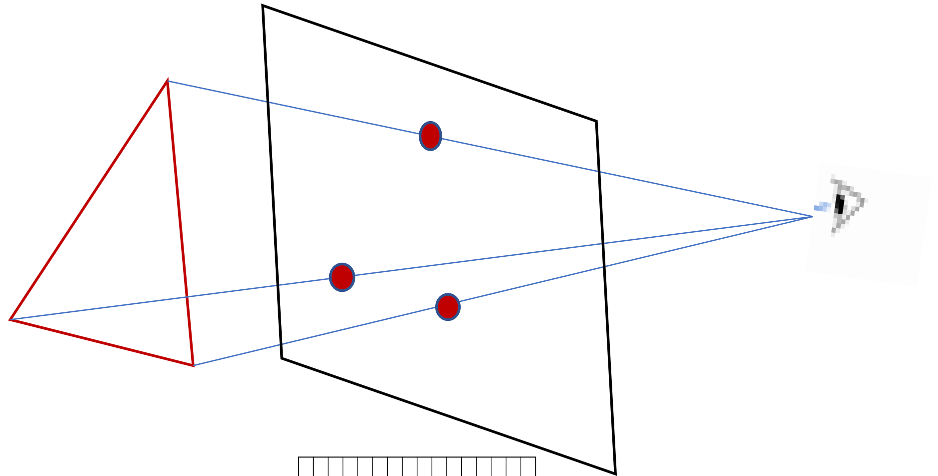
➢**Rendering（cont.）**

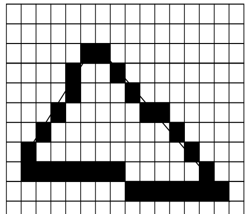
渲染方法1： 光栅化渲染➔正向

➢人工操作：采用指针细线木框合页生成的图：从物体表面采样点出发，细线(投影线）连接该采样点和墙上点（投影中心点）的直线交于合页（投影面）处画投影点。适合于简单场景

- *木刻画"鲁特琴图" Ref:《计算机图形学原理及实践》约翰.F.休斯等著，彭群生等译 机械工业出版社*







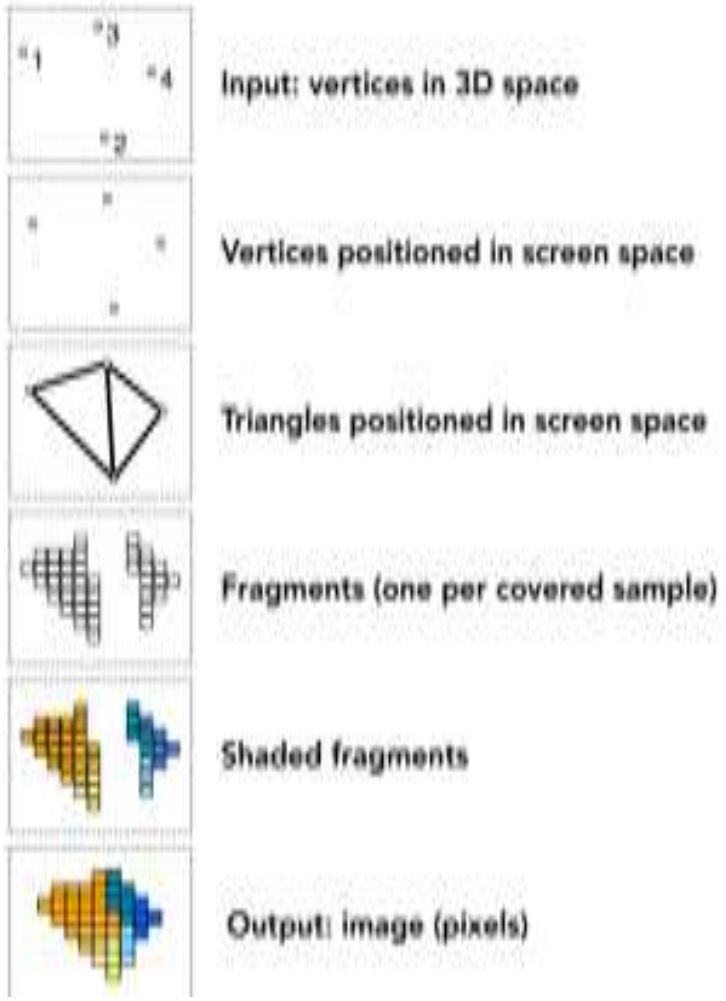
Rendering Pipeline(cont.)

➢**Rendering（cont.)**

渲染方法1： 光栅化渲染➔正向(cont.)

➢**Rendering Steps:**

- **vertex process**顶点处理
- **primitive assembling**图元裁剪组装
- **Rasterization** 光栅化(从顶点到片元)
- **fragment processing**片元处理



Input: vertices in 3D space
Vertices positioned in screen space
Triangles positioned in screen space
Fragments (one per covered sample)
Shaded fragments
Output: image (pixels)

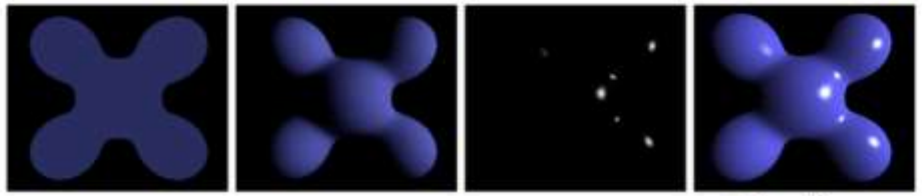
Rendering Pipeline(cont.)

➢**Rendering（cont.）**

方法1： 光栅化渲染➔正向（cont.)

• 特点： 局部光照计算

Blinn-Phong Reflection Model

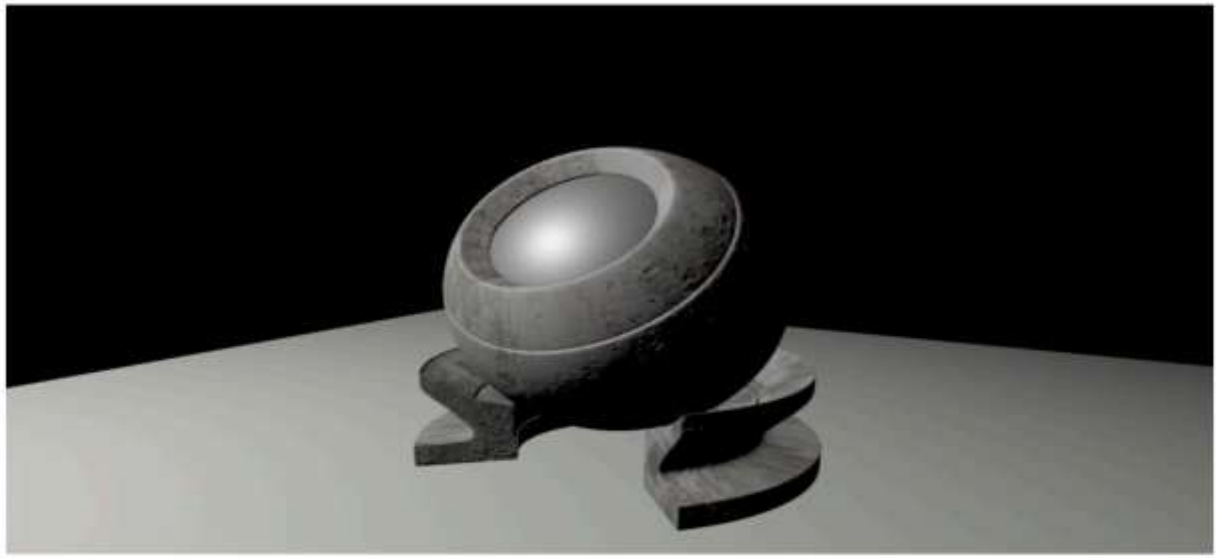


$$
\begin{aligned}
L &= L_a + L_d + L_s \\
&= k_a\, I_a + k_d\,(I/r^2)\max(0, \mathbf{n}\cdot\mathbf{l}) + k_s\,(I/r^2)\max(0, \mathbf{n}\cdot\mathbf{h})^p
\end{aligned}
$$

GAMES101 12 Lingqi Yan, UC Santa Barbara

Shading is Local

No shadows will be generated! (**shading ≠ shadow**)



GAMES101 23 Lingqi Yan, UC Santa Barbara

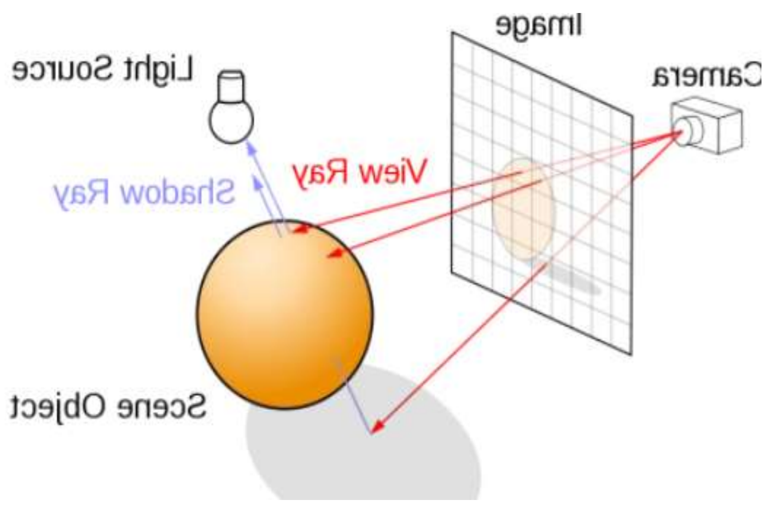
Rendering Pipeline(cont.)

➢**Rendering（cont.)**

渲染方法2：光线跟踪 ➔逆向

人工过程：从投影中心点（眼睛）出发，视线穿过投影面方格（像素），交于物体表面，再将“看到”的内容绘制到对应图纸方格（像素）中。适合于复杂场景。

Ref:《计算机图形学原理及实践》约翰.F.休斯等著，彭群生等译 机械工业出版社

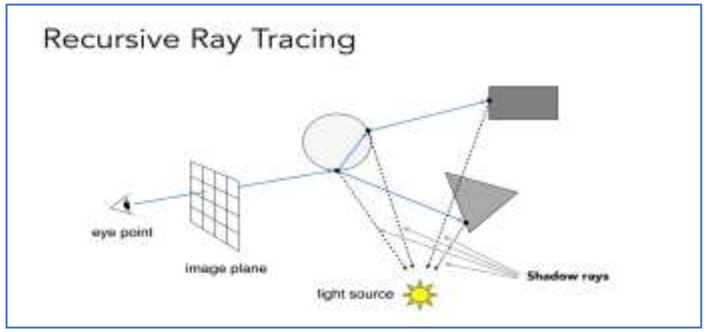


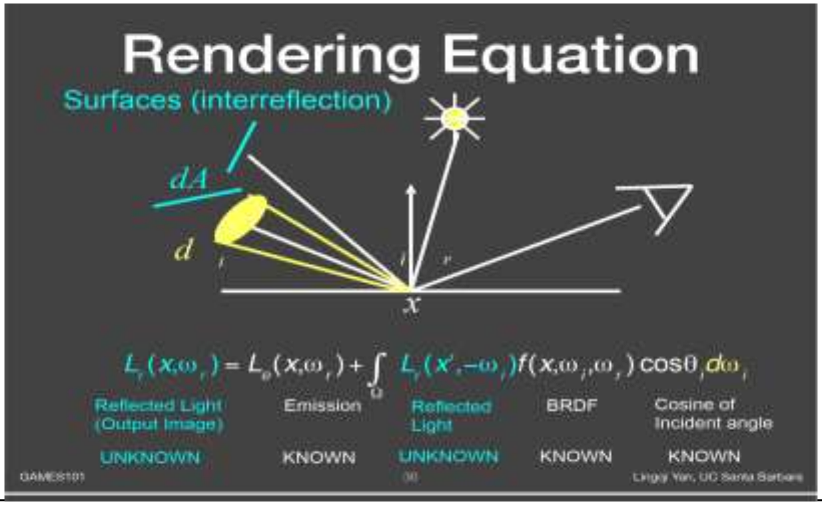
Rendering Pipeline(cont.)

➢**Rendering（cont.）**

渲染方法2：光线跟踪 ➔逆向（cont.)

- **递归；可全局光照计算（PBR)**



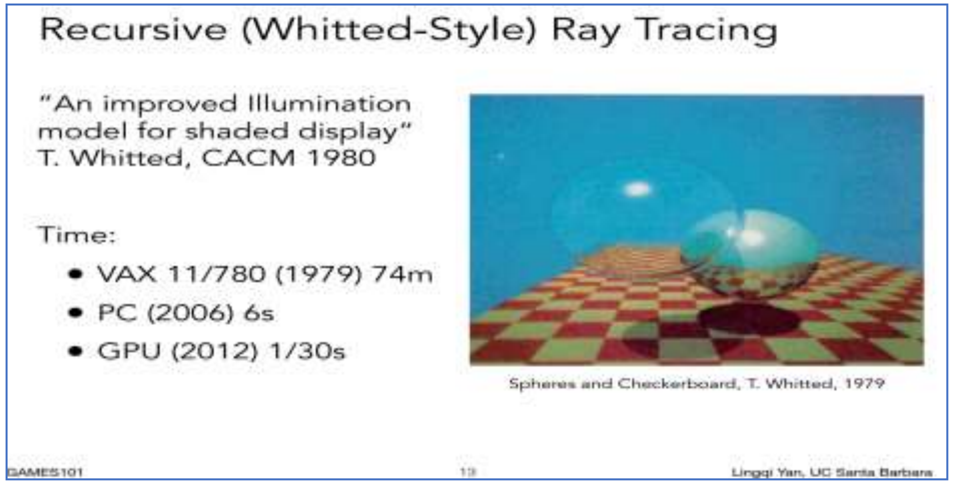


Rendering Equation

$L_r(x,\omega_r) = L_e(x,\omega_r) + \int_\Omega L_r(x',-\omega_i) f(x,\omega_i,\omega_r) \cos\theta_i d\omega_i$

Reflected Light (Output Image)	Emission	Reflected Light	BRDF	Cosine of Incident angle
UNKNOWN	KNOWN	UNKNOWN	KNOWN	KNOWN

GAMES101 Lingqi Yan, UC Santa Barbara



Recursive (Whitted-Style) Ray Tracing

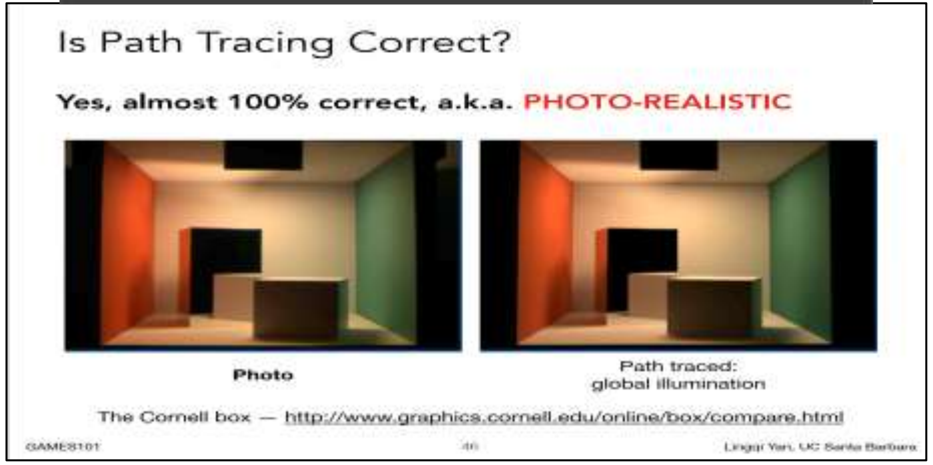
"An improved Illumination model for shaded display"
T. Whitted, CACM 1980

Time:

- VAX 11/780 (1979) 74m
- PC (2006) 6s
- GPU (2012) 1/30s

Spheres and Checkerboard, T. Whitted, 1979

GAMES101 13 Lingqi Yan, UC Santa Barbara



Is Path Tracing Correct?

Yes, almost 100% correct, a.k.a. PHOTO-REALISTIC

Photo

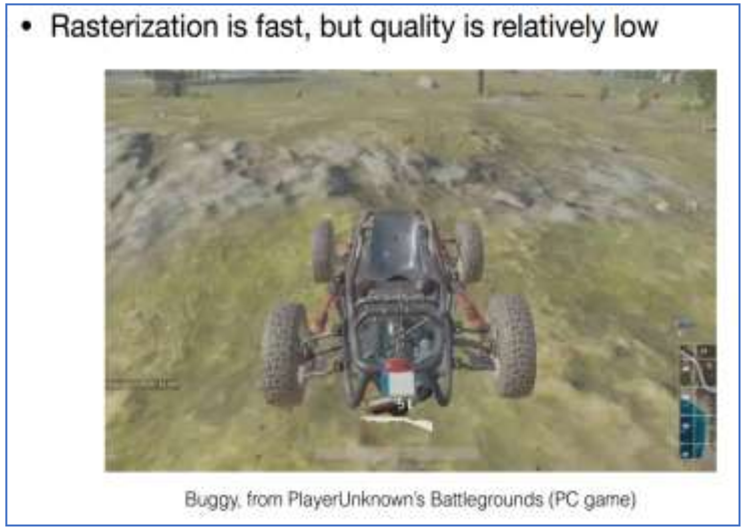
Path traced: global illumination

The Cornell box — http://www.graphics.cornell.edu/online/box/compare.html

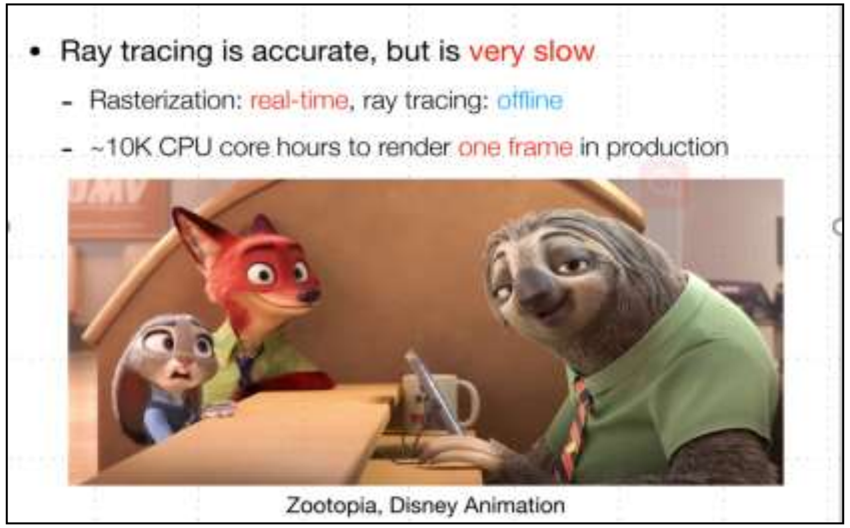
GAMES101 46 Lingqi Yan, UC Santa Barbara

Rendering Pipeline(cont.)

- **Rendering（cont.）**
  - Rasterization光栅化方法
    - 需要仔细选择物体采样点， 生成场景中的物体的轮廓， 然后光栅化为像素。
    - **光线正向跟踪：从物体出发，跟踪光线到眼睛**
  - Ray Tracing光线跟踪方法
    - 在每一个方格中填充某一颜色。不判断场景物体之间遮挡关系却总是能绘制可见的那部分物体表面颜色。
    - **光线逆向跟踪：从眼睛出发，跟踪光线到物体**
- **因为光线跟踪计算复杂，早期受硬件条件限制，实时渲染通常采用“光栅化渲染法”设计管线并沿用至今，可采用各种技术来提高画面的真实感。**


- Rasterization is fast, but quality is relatively low

Buggy, from PlayerUnknown's Battlegrounds (PC game)


- Ray tracing is accurate, but is very slow
  - Rasterization: real-time, ray tracing: offline
  - ~10K CPU core hours to render one frame in production

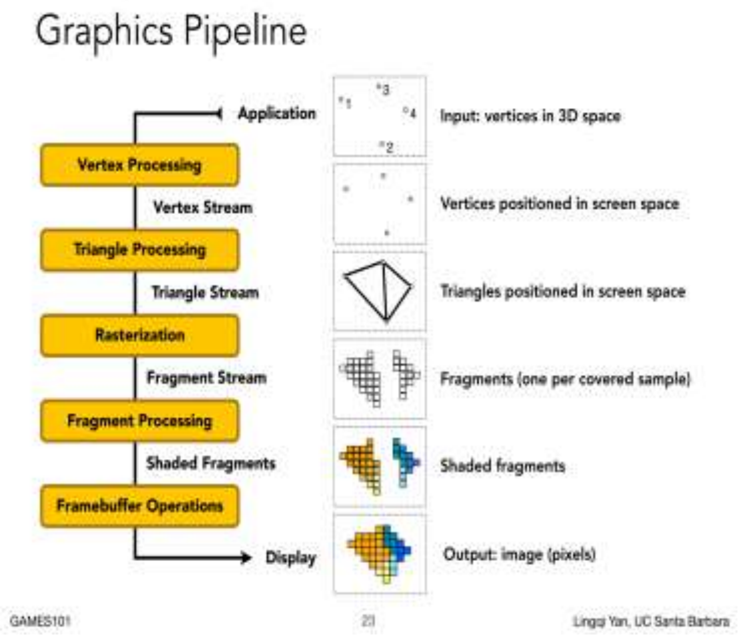
Zootopia, Disney Animation

Rendering Pipeline(cont.)

➢Rendering（cont.)

■Rasterization-based Pipeline Main Steps :

- ➢Vertex processing（顶点/几何处理）
- ➢Triangle Processing（图元组装)
- ➢Rasterization(光栅化：从几何到片元）
- ➢Fragment Processing（片元处理）
- ➢Fragmentbuffer Operations



Bilibili视频介绍图形渲染管线（后续学习主要内容）

- ➢ *1: 深入浅出计算机图形学*
  https://www.bilibili.com/video/BV1xKWdeSE7v/?spm_id_from=333.788.recommend_more_video.5&vd_source=c93f9d5c2c6ee8043fe0db22203390ee
- ➢ *2: GPU架构与渲染优化-光栅化篇*

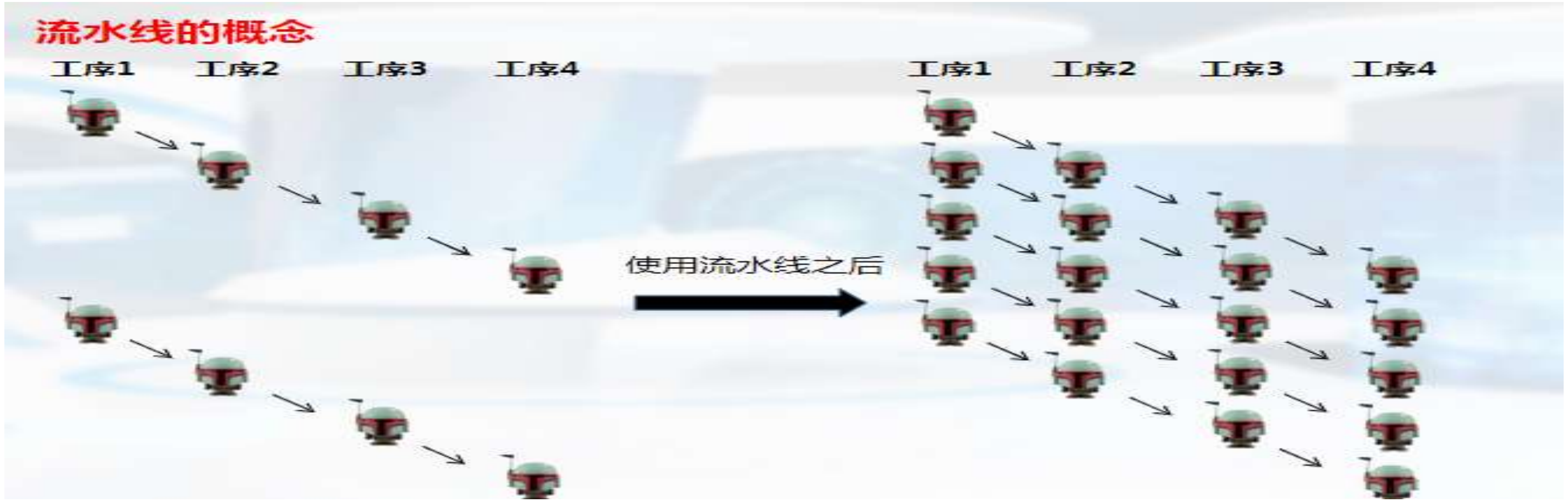
https://www.bilibili.com/video/BV1gUWDe2ECU/?spm_id_from=333.788.recommend_more_video.3&vd_source=c93f9d5c2c6ee8043fe0db22203390ee

Rendering Pipeline(cont.)

➢Rendering（cont.）

■ why Pipeline?

- **Vertex processing, fragment Processing 都是大规模数据的计算处理**
- 如果采用**GPU**的并行处理，可极大提高系统的吞吐量，提高计算效率！



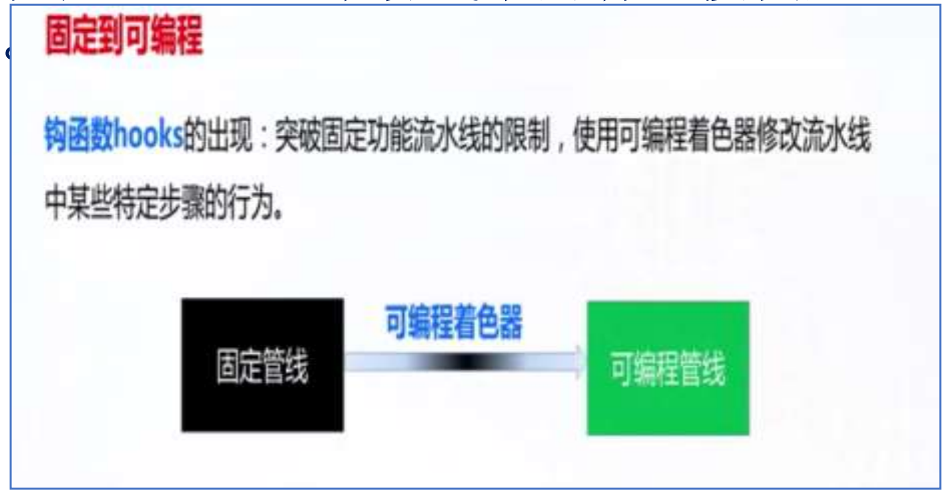
Rendering Pipeline(cont.)

➢Rendering（cont.）

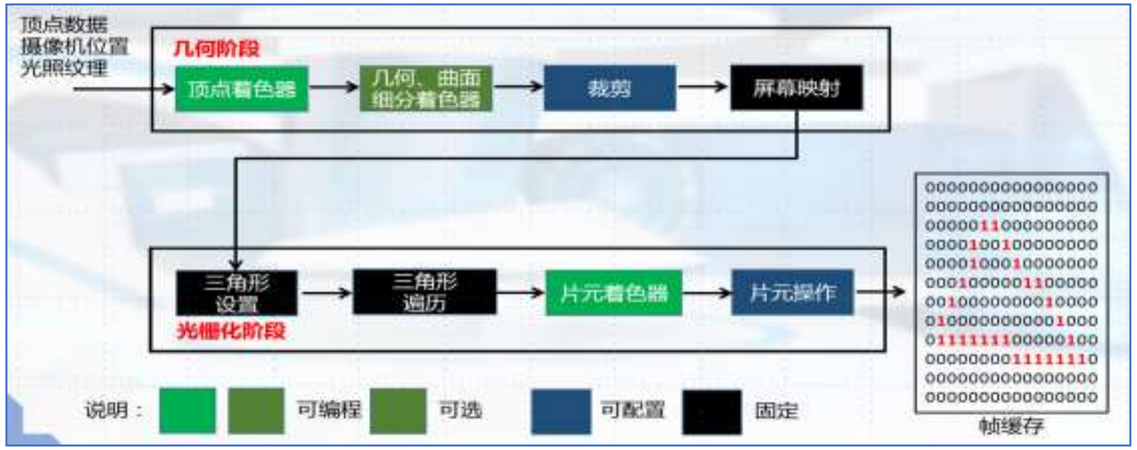
□ why Programmable Rendering Pipeline?

- The Fix Functions Pipeline(固定管线）： limited Functions
  - GPU上提供的功能模块是“固定”，程序员只能通过接口API调用使用，但不能修改模块。
- The Programmable Pipeline(可编程): Extend Functions
  - GPU上的功能模块可以被程序员“编程”替换掉，可灵活扩展图形渲染流水线的功能。
  - 如“顶点着色器”，“片元着色器”，就是可由程序员编写shader程序替换的模块

➢现在GPU上渲染管线中的有些模块是“可编程的”，有些是“固定的”；今后向全可编程发展


固定到可编程

钩函数hooks的出现：突破固定功能流水线的限制，使用可编程着色器修改流水线中某些特定步骤的行为。



Computer Graphics System Outlines

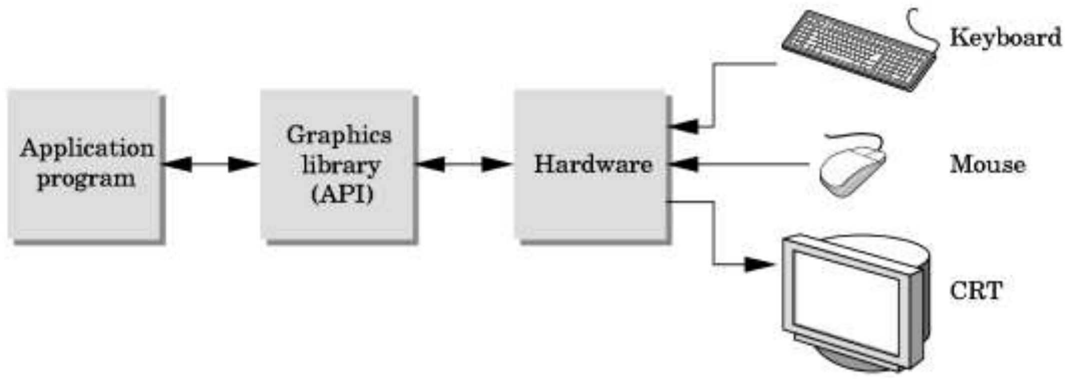
- ➢Input devices
- ➢Output device
- ➢Raster Imaging System-光栅成像系统*

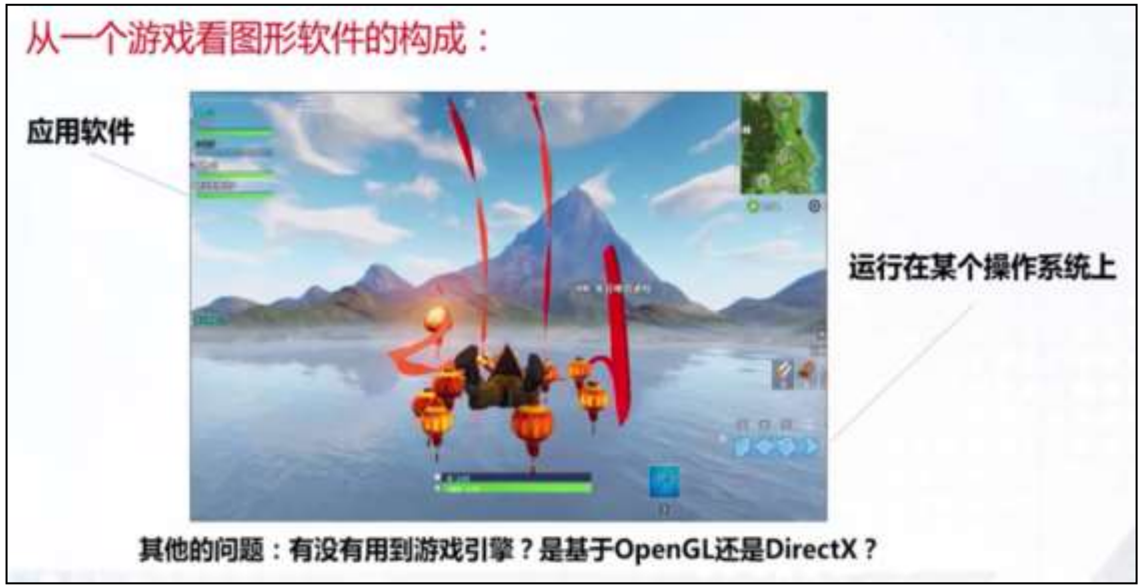
- ➢Image Formation Principle –成像原理
- ➢Rendering Pipeline –渲染管线*
- ➢**The Programmer’s Interface-程序员编程接口**

Application Programming Interface(API)

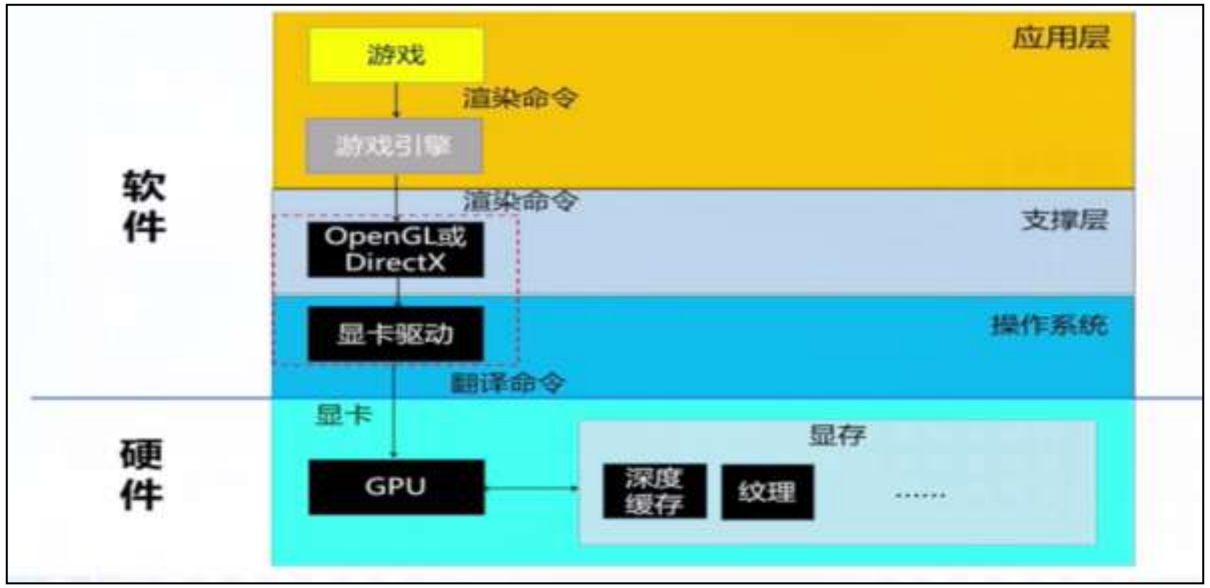
➢ 图形系统的软件层次

- 应用层(游戏引擎)
- **支撑层(图形库GL)**
- 操作系统（显卡驱动）





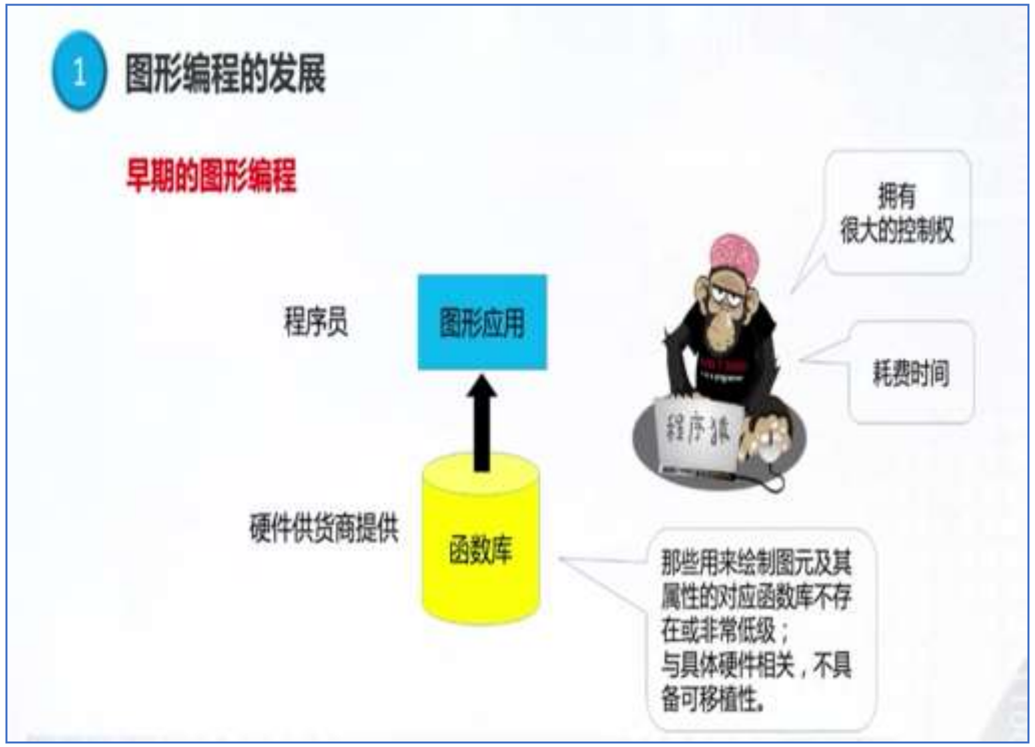
其他的问题：有没有用到游戏引擎？是基于OpenGL还是DirectX？



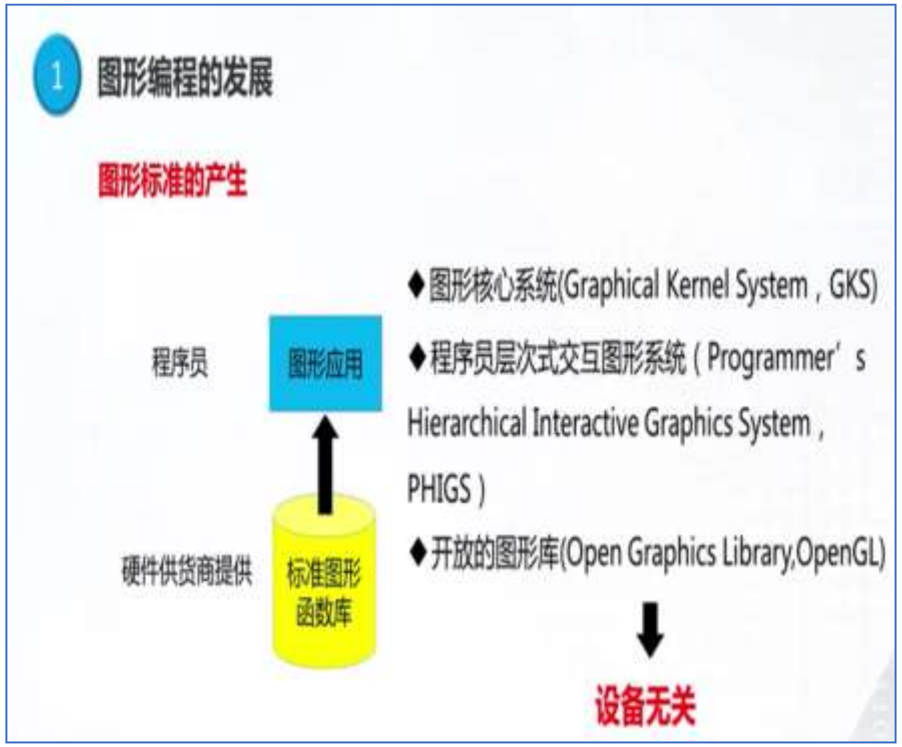
Application Programming Interface（cont.)

➢图形系统的支撑层(图形库Graphics Library)的标准化：

• 标准化可以使图形库的编写不依赖于显卡硬件，所编写的程序具有可移植性



那些用来绘制图元及其属性的对应函数库不存在或非常低级；与具体硬件相关，不具备可移植性。



♦图形核心系统(Graphical Kernel System，GKS)

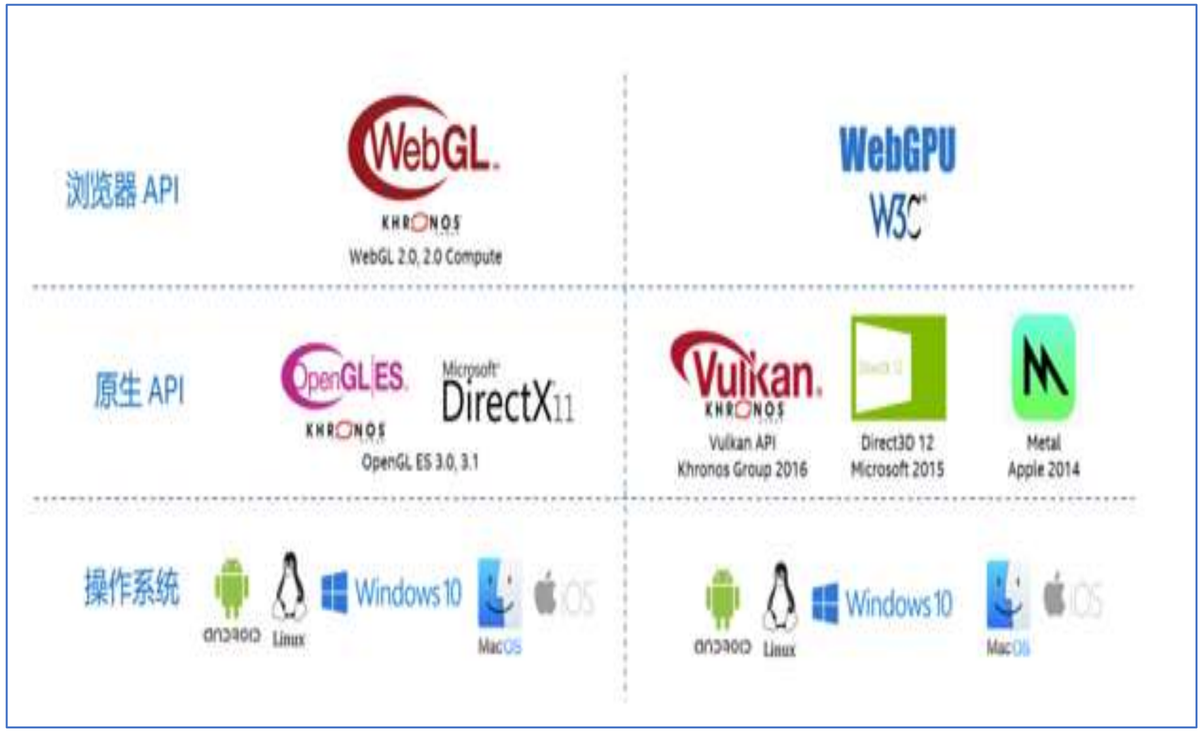
♦程序员层次式交互图形系统（Programmer’s Hierarchical Interactive Graphics System，PHIGS）

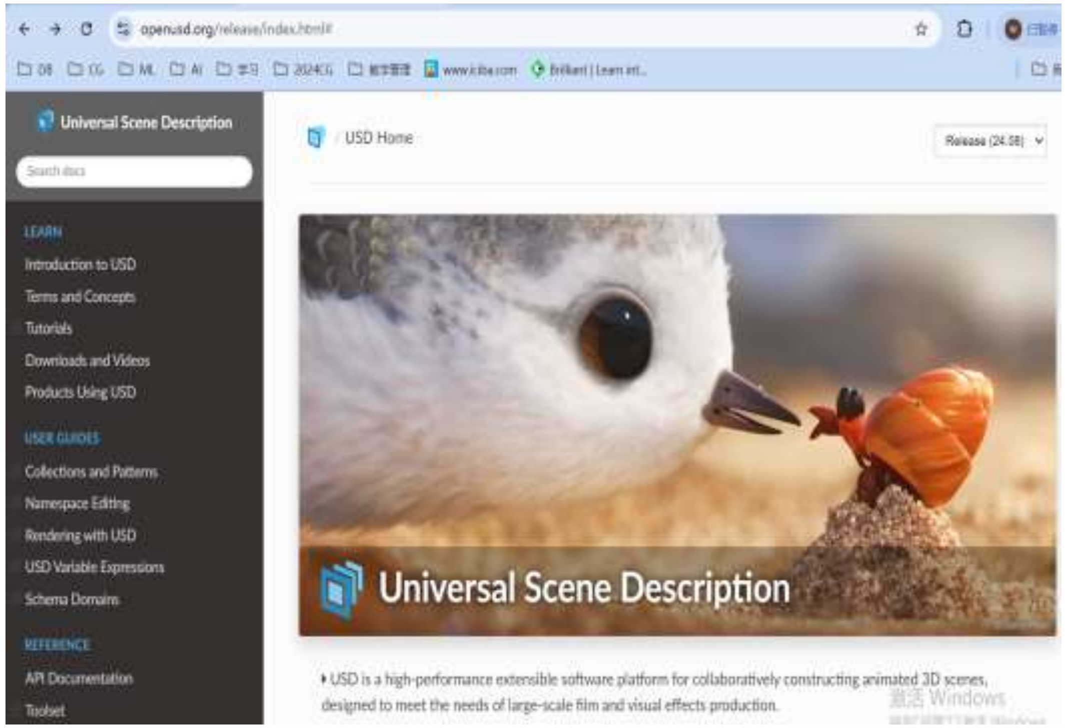
♦开放的图形库(Open Graphics Library,OpenGL)

Application Programming Interface（cont.)

➢ **图形标准化库：**

- Khronos标准库： **如OpenGL/webGL; Vulkan/webGPU;** **ref: https://www.khronos.org/**
- Omniverse标库： **如OpenUSD， ref:** https://zhuanlan.zhihu.com/p/648456231




openusd.org/release/index.html#

Universal Scene Description

Search docs

LEARN
Introduction to USD
Terms and Concepts
Tutorials
Downloads and Videos
Products Using USD

USER GUIDES
Collections and Patterns
Namespace Editing
Rendering with USD
USD Variable Expressions
Schema Domains

REFERENCE
API Documentation
Toolset

/ USD Home

Release (24.08)

Universal Scene Description

USD is a high-performance extensible software platform for collaboratively constructing animated 3D scenes, designed to meet the needs of large-scale film and visual effects production.

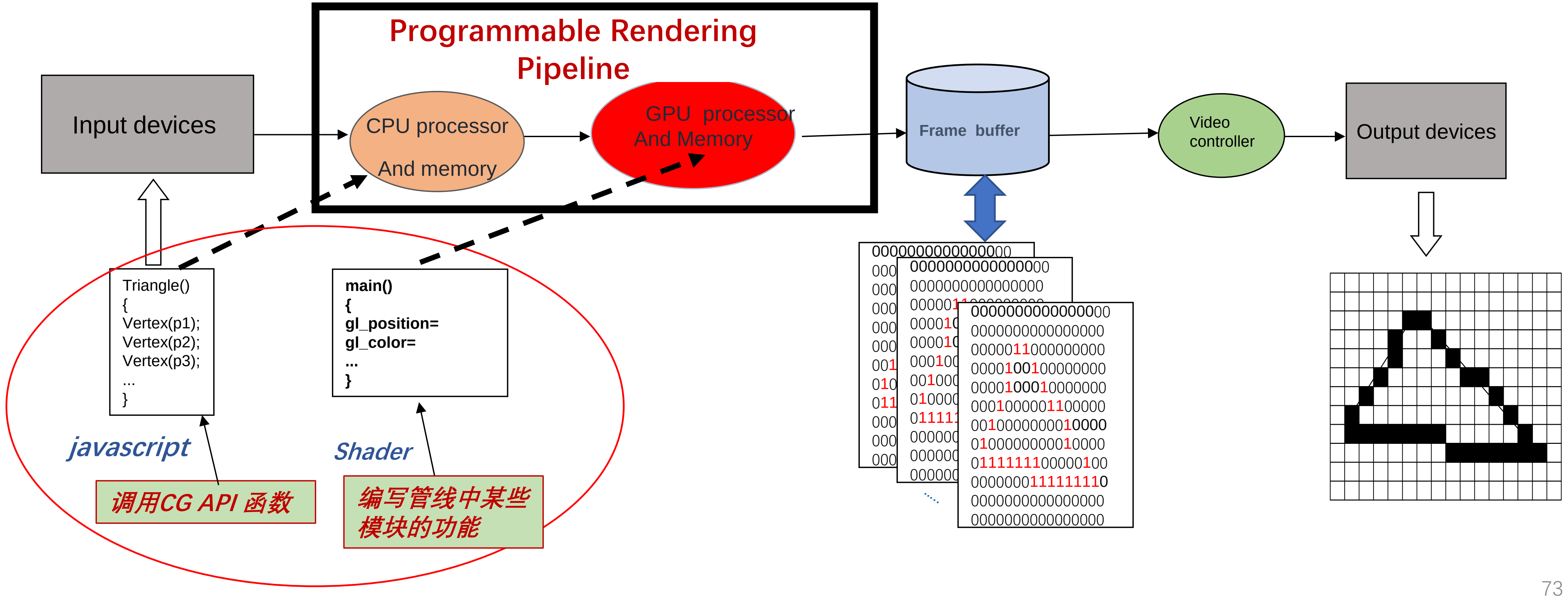
Application Programming Interface（cont.)

CG API Functions

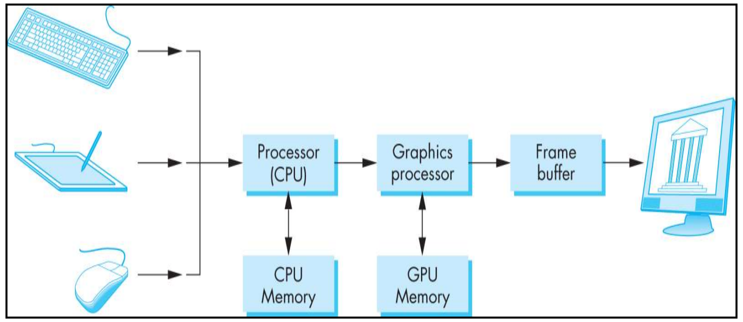
- ➢**Functions that specify what we need to form an image**
  - Objects & Materials
  - Viewer
  - Light Sources
- ➢**Other Function**
  - Input from devices such as mouse and keyboard
  - Capabilities of system
- **Shader**
  - **由程序员采用Shader Language 语言编写的并行执行的小程序。**
  - **Shader 在图形卡的GPU （Graphic Processor Unit图形处理单元）上执行，代替了固定的渲染管线的一部分，使渲染管线具有可编程性,**

Application Programming Interface（cont.)

- CG application Programming



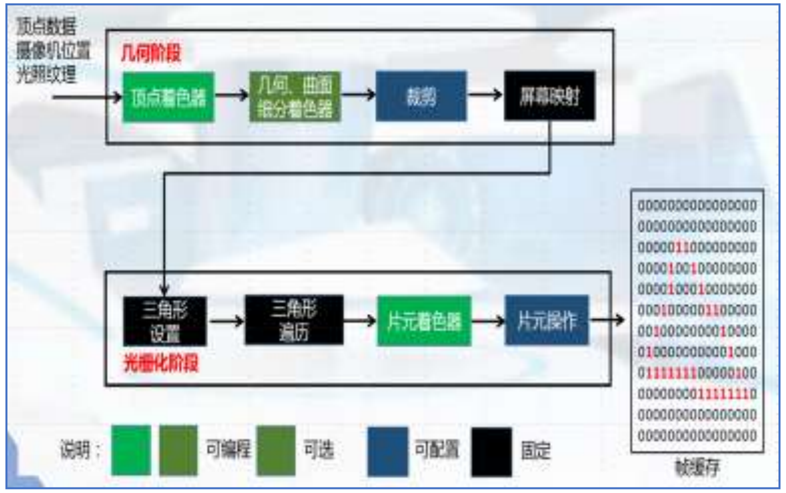
Summary



- 输入设备（键盘，鼠标，绘图仪等）
- 输出设备（光栅显示设备：CRT发光原理，刷新频率，光栅扫描显示器）
- **处理及存储设备（显卡：显示子系统）**
  1. 视频处理器controller：将颜色缓存中的颜色编码用来控制显示器进行显示
  2. 帧缓存framebuffer：存放一帧画面的像素的颜色编码
     - 基本概念：像素，分辨率，帧，帧缓存
     - 颜色表示：颜色模型（RGB, CMY，HBV，XYZ)；像素深度/像素精度
     - 颜色的存储：组合像素，颜色位面Bit Plane，查色表CLT
  3. 显示处理器GPU：将程序中输入的顶点表示的图形或者图像，经过图形处理流水线操作，转换为帧缓存中的一帧像素的颜色编码。
     - GPU是什么 ,GPU和CPU各自的优点是什么？

Summary(cont.)

- **Image Formation Principle –成像原理**
  - 人眼成像
  - 小孔成像
  - 相机成像
  - 虚拟相机模型
- **Rendering Pipeline –渲染管线实现***
  - 方法: 光栅化方法，光线跟踪方法
  - 图形处理管线：基于光栅化渲染方法设计
  - 固定管线和可编程渲染管线
- **The Programmer’s Interface-程序员编程接口**
  - 图形库：OpenGL/Vulkan，WebGL/WebGPU


顶点数据
摄像机位置
光照纹理
几何阶段
顶点着色器
几何、曲面细分着色器
裁剪
屏幕映射
三角形设置
三角形遍历
片元着色器
片元操作
光栅化阶段
说明：
可编程
可选
可配置
固定
帧缓存