Gentle intro to Real-time Rendering

Kyung Jae Lee
SIAT-SFU

Today's' discussion

- Overview of Real-time rendering
- Discuss how real-time rendering could be used as a problem-solving tool to advance our understanding of Facial Animation project
- Demo

Real-time ?

- In computer science, real-time means “with bounded execution time” (i.e., real-time operating system)
- In computer graphics, real-time rendering means the rendering should be fast enough that we do not perceive it as taking any time at all; it seems like it is a real object moving under our control (i.e., interactive) from 15 fps to 60 fps.

Source: Real-time Shading by Olano et al.

- Two basic meanings in real-time
  1. Description of software and hardware systems
     - Hard real-time system: constraints on the response time should be satisfied. (e.g., air-bag deploy within millisecond)
     - Soft real-time system: performance degradation (e.g., performance of digital camera taking low-light scene, real-time HCI system with unnoticeable 45 ms visual display)
  2. Characterization of algorithms; fast algorithm to allow entire system to achieve real-time operation than some other algorithm. (e.g., Fast Fourier Transform)

Source: Real-time vision for human-computer interaction by Branislav Kisacanin et al. (Eds.)

Real-time in General

- Real-time system application around us
  - Mobile phone, alarm systems, microwave machine, ATM
- Definition of a real-time system
  - Specified limit on system response latency
  - Event-driven scheduling
  - Low-level programming
  - Software tightly coupled to special hardware
  - Dedicated specialized function
  - The computer may be inside a control loop
  - Volatile variables
  - Multi-tasking implementation
  - Run-time scheduling
  - Unpredicted environment
  - System intended to run continuously
  - Life-critical applications

Source: Real-time Systems Development by Rob Williams

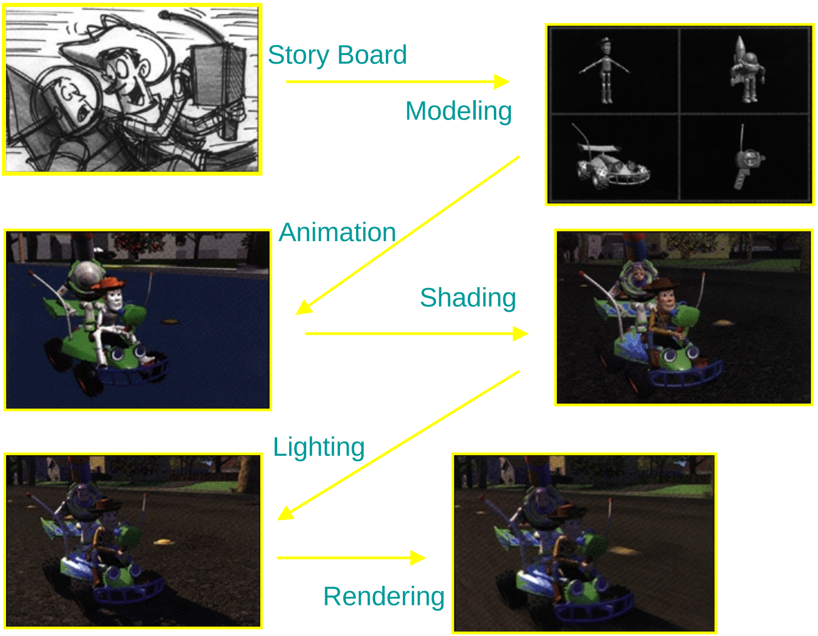
Real-time in Graphics

- By real-time graphics, we generally mean that the graphics-related computations are being carried out fast enough that the results can be viewed immediately. Being able to conduct operations at 60 HZ is considered real time. Once the time to refresh the display (frame rate) drops below 15 Hz, the speed is considered more interactive than it is real-time, but this distinction is not critical. Because the computations need to be fast, the equations used to render the graphics are often approximations to what could be done if more time were available. (Peter Shirley)

- Non-real time sometimes means pre-rendered.

Elements in Computer Graphics

1. Viewing and Projection
   - Orthographic projection, perspective projection
   - Clipping on view volume, hidden surface
   - Stereo viewing
2. Modeling
   - Point, line segment, triangle, quad, polygon, surface, spline
   - Antialiasing
   - Transformation, stacks, display list
3. Color
   - CIE XYZ, RGB, HSV, Black body
4. Lighting & Shading
   - Ambient, diffuse, specular
   - Spotlight, Positional light, Directional light
   - Local illumination: phong etc,
   - Global illumination: Radiosity
5. Texture mapping
6. Animation, dynamics
7. Events/interactive programming
8. Rendering pipeline
   - Rasterization



Modeling

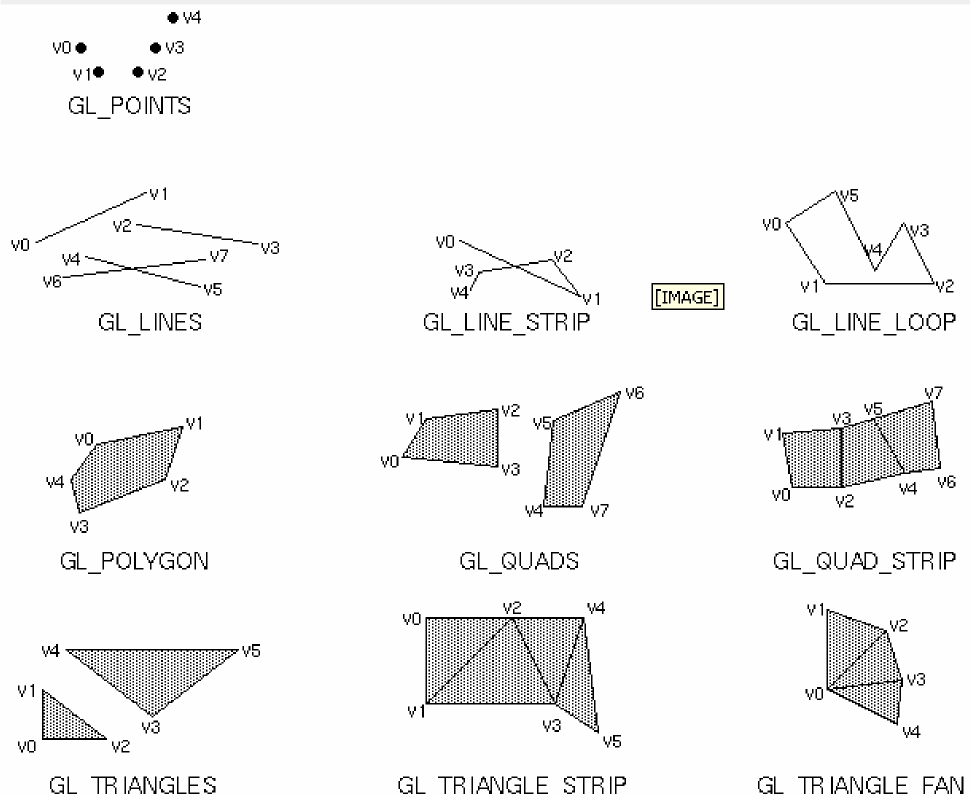


Figure 2-6 : Geometric Primitive Types

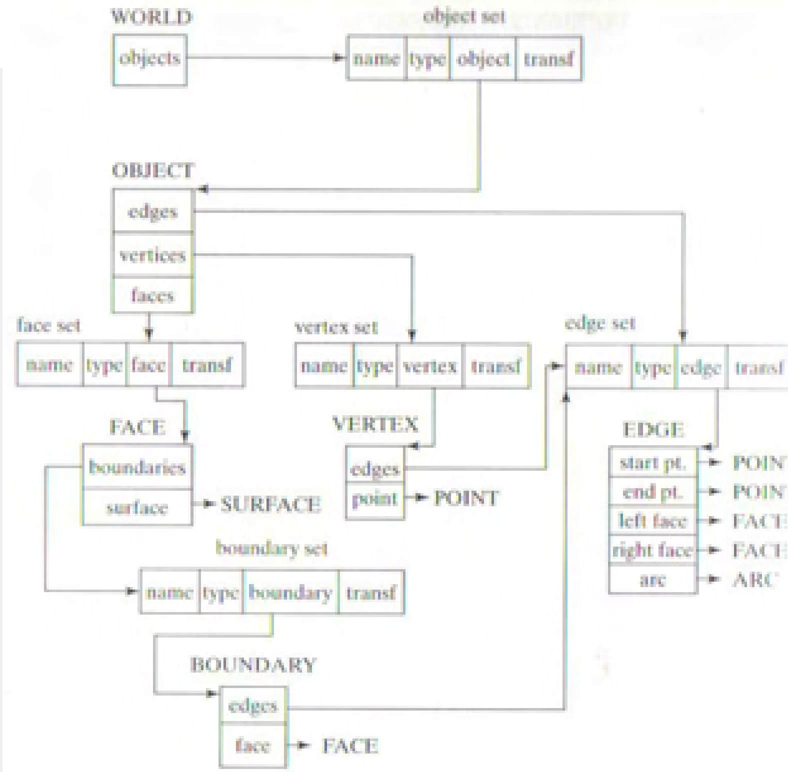


Figure 14.3 Surface-edge-vertex data structure.

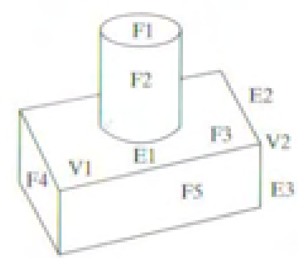


Figure 14.4 Sample 3D object with p[lanar] and cylindrical surfaces.

- Data Size matter (computational load)

- What could be the file size for this leaf image?

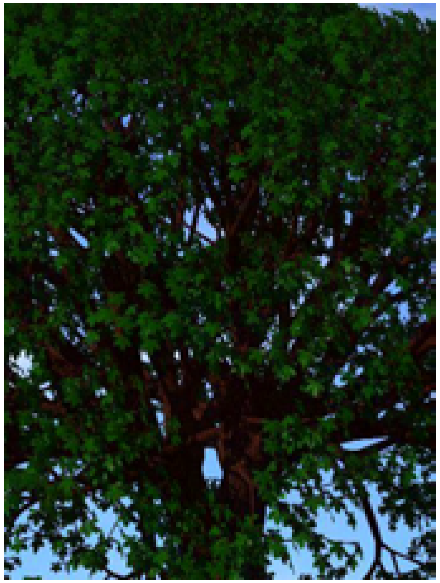
(Generated from Alias Maya 3D and rendered in Pixar's Renderman API)

- 7 KB using Pixar's Renderman API

- ##RenderMan RIB version 3.03 Format 320 240 1 Orientation "rh" Sides 2 PixelSamples 3 3 LightSource "distantlight" 1 ObjectBegin 1 NuPatch 8 4 [0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1] 0 1 8 4 [0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1] 0 1 "Pw" [-0.953489 -6.12303e-017 0.358152 1 -1.13953 -5.30663e-017 0.401559 1 -1.51162 -3.67382e-017 0.190705 1 -2.293 -1.22461e-017 -0.0604601 1 -1 1.22461e-017 -0.2 1 -1 3.67382e-017 -0.6 1 -1.73487 5.30663e-017 -1.22015 1 -1.72557 6.12303e-017 -1.4465 1 -0.336444 -6.12303e-017 0.330245 1 -0.336444 -5.30663e-017 0.196912 1 -1.04341 -3.67382e-017 0.134893 1 -1.51782 -1.22461e-017 -0.265107 1 -0.866667 1.22461e-017 -0.2 1 -0.866667 3.67382e-017 -0.6 1 -1.48991 5.30663e-017 -1.31317 1 -1.40619 6.12303e-017 -1.70696 1 -1.11162 -6.12303e-017 1.08372 1 -1.11162 -5.30663e-017 0.950386 1 -0.320936 -3.67382e-017 0.395353 1 -0.6 -1.22461e-017 0.2 1 -0.6 1.22461e-017 -0.2 1 -0.627906 3.67382e-017 -0.6 1 -0.348842 5.30663e-017 -0.159703 1 -0.348842 6.12303e-017 -0.293037 1 -0.2 -6.12303e-017 0.553497 1 -0.2 -5.30663e-017 0.420164 1 -0.2 -3.67382e-017 0.423259 1 -0.2 -1.22461e-017 0.2 1 -0.2 1.22461e-017 -0.2 1 -0.358137 3.67382e-017 -0.6 1 -0.395345 5.30663e-017 -1.62944 1 -0.0976764 6.12303e-017 -2.84183 1 0.2 -6.12303e-017 0.637216 1 0.2 -5.30663e-017 0.503883 1 0.2 -3.67382e-017 0.423259 1 0.2 -1.22461e-017 0.2 1 0.2 1.22461e-017 -0.2 1 0.348834 3.67382e-017 -0.73023 1 0.404647 5.30663e-017 -1.62944 1 0.302324 6.12303e-017 -2.84183 1 0.358144 -6.12303e-017 0.748842 1 0.358144 -5.30663e-017 0.615509 1 0.358144 -3.67382e-017 0.348842 1 0.609302 -1.22461e-017 0.2 1 0.6 1.22461e-017 -0.2 1 0.590698 3.67382e-017 -0.73023 1 0.265123 5.30663e-017 -0.103891 1 0.265123 6.12303e-017 -0.237224 1 0.773645 -6.12303e-017 0.897676 1 0.5783 -5.30663e-017 0.624811 1 0.18761 -3.67382e-017 0.348842 1 1.38759 -1.22461e-017 0.237209 1 1.84339 1.22461e-017 -0.358137 1 0.0852862 3.67382e-017 -0.00466253 1 1.50851 5.30663e-017 -1.35968 1 1.50851 6.12303e-017 -1.56743 1 0.813957 -6.12303e-017 0.776748 1 0.711633 -5.30663e-017 0.624811 1 0.320943 -3.67382e-017 0.348842 1 1.52092 -1.22461e-017 0.227906 1 1.97673 1.22461e-017 -0.358137 1 0.21862 3.67382e-017 -0.00466253 1 1.8837 5.30663e-017 -1.35968 1 1.58604 6.12303e-017 -1.49301 1] ObjectEnd ObjectBegin 2 NuPatch 8 4 [0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1] 0 1 8 4 [0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1] 0 1 "Pw" [-0.953489 -6.12303e-017 0.358152 1 -1.13953 -5.30663e-017 0.401559 1 -1.51162 -3.67382e-017 0.190705 1 -2.293 -1.22461e-017 -0.0604601 1 -1 1.22461e-017 -0.2 1 -1 3.67382e-017 -0.6 1 -1.73487 5.30663e-017 -1.22015 1 -1.72557 6.12303e-017 -1.4465 1 -0.336444 -6.12303e-017 0.330245 1 -0.336444 -5.30663e-017 0.196912 1 -1.04341 -3.67382e-017 0.134893 1 -1.51782 -1.22461e-017 -0.265107 1 -0.866667 1.22461e-017 -0.2 1 -0.866667 3.67382e-017 -0.6 1 -1.48991 5.30663e-017 -1.31317 1 -1.40619 6.12303e-017 -1.70696 1 -1.11162 -6.12303e-017 1.08372 1 -1.11162 -5.30663e-017 0.950386 1 -0.320936 -3.67382e-017 0.395353 1 -0.6 -1.22461e-017 0.2 1 -0.6 1.22461e-017 -0.2 1 -0.627906 3.67382e-017 -0.6 1 -0.348842 5.30663e-017 -0.159703 1 -0.348842 6.12303e-017 -0.293037 1 -0.2 -6.12303e-017 0.553497 1 -0.2 -5.30663e-017 0.420164 1 -0.2 -3.67382e-017 0.423259 1 -0.2 -1.22461e-017 0.2 1 -0.2 1.22461e-017 -0.2 1 -0.358137 3.67382e-017 -0.6 1 -0.395345 5.30663e-017 -1.62944 1 -0.0976764 6.12303e-017 -2.84183 1 0.2 -6.12303e-017 0.637216 1 0.2 -5.30663e-017 0.503883 1 0.2 -3.67382e-017 0.423259 1 0.2 -1.22461e-017 0.2 1 0.2 1.22461e-017 -0.2 1 0.348834 3.67382e-017 -0.73023 1 0.404647 5.30663e-017 -1.62944 1 0.302324 6.12303e-017 -2.84183 1 0.358144 -6.12303e-017 0.748842 1 0.358144 -5.30663e-017 0.615509 1 0.358144 -3.67382e-017 0.348842 1 0.609302 -1.22461e-017 0.2 1 0.6 1.22461e-017 -0.2 1 0.590698 3.67382e-017 -0.73023 1 0.265123 5.30663e-017 -0.103891 1 0.265123 6.12303e-017 -0.237224 1 0.773645 -6.12303e-017 0.897676 1 0.5783 -5.30663e-017 0.624811 1 0.18761 -3.67382e-017 0.348842 1 1.38759 -1.22461e-017 0.237209 1 1.84339 1.22461e-017 -0.358137 1 0.0852862 3.67382e-017 -0.00466253 1 1.50851 5.30663e-017 -1.35968 1 1.50851 6.12303e-017 -1.56743 1 0.813957 -6.12303e-017 0.776748 1 0.711633 -5.30663e-017 0.624811 1 0.320943 -3.67382e-017 0.348842 1 1.52092 -1.22461e-017 0.227906 1 1.97673 1.22461e-017 -0.358137 1 0.21862 3.67382e-017 -0.00466253 1 1.8837 5.30663e-017 -1.35968 1 1.58604 6.12303e-017 -1.49301 1] ObjectEnd FrameBegin 1 Display "Renderman" "framebuffer" "rgba" Format 320 240 1 Projection "perspective" "fov" [42.1847] Clipping 0.1 1000 Transform [0.999848 0.017365 -0.00174499 0 1.73472e-018 -0.0999858 -0.994989 0 0.0174524 -0.994837 0.0999706 0 0.173454 -0.871769 6.205 1] WorldBegin AttributeBegin Attribute "identifier" "name" ["nurbsPlane2"] ConcatTransform [1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1] AttributeBegin Attribute "identifier" "name" ["nurbsPlaneShape2"] Surface "matte" Color [0.5 0.5 0.5] ObjectInstance 1 AttributeEnd AttributeEnd WorldEnd FrameEnd

- For the realistic image representation,

- 7 KB

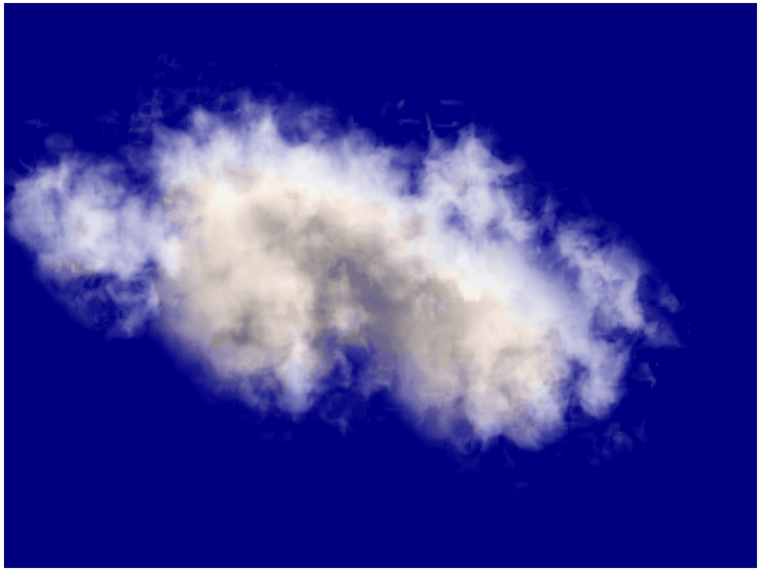


- More than 10 MB

- More than 20 MB

Demo

- Real-time cloud
  - Dr.David Ebert (Purdue University)

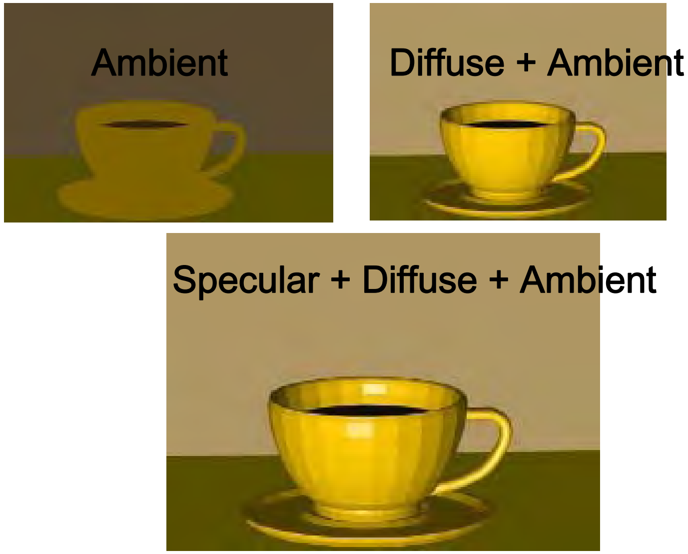


http://news.uns.purdue.edu/mov/boilingup.mov

Local & global Illumination

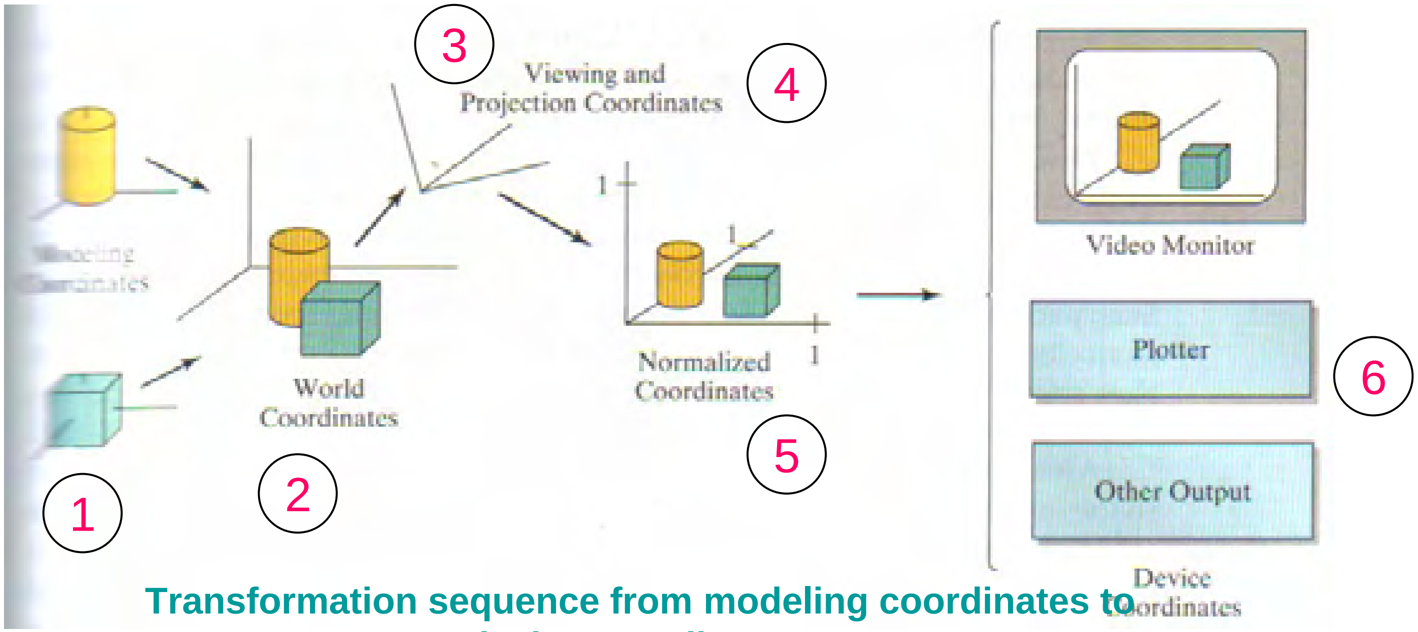
- Local Illumination
  - Light arriving at the surface directly from light source
  - Diffuse/Lambertian,
  - Specular reflection (Phong 1975; Blinn 1977; Cook-Torrance 1981; He et al. 1991)
  - Anisotropic (Kajiya 1985)

- Global Illumination
  - Indirect lighting due to reflection, refraction, scattering light from other surfaces or participating media in the scene
  - Ray tracing, radiosity



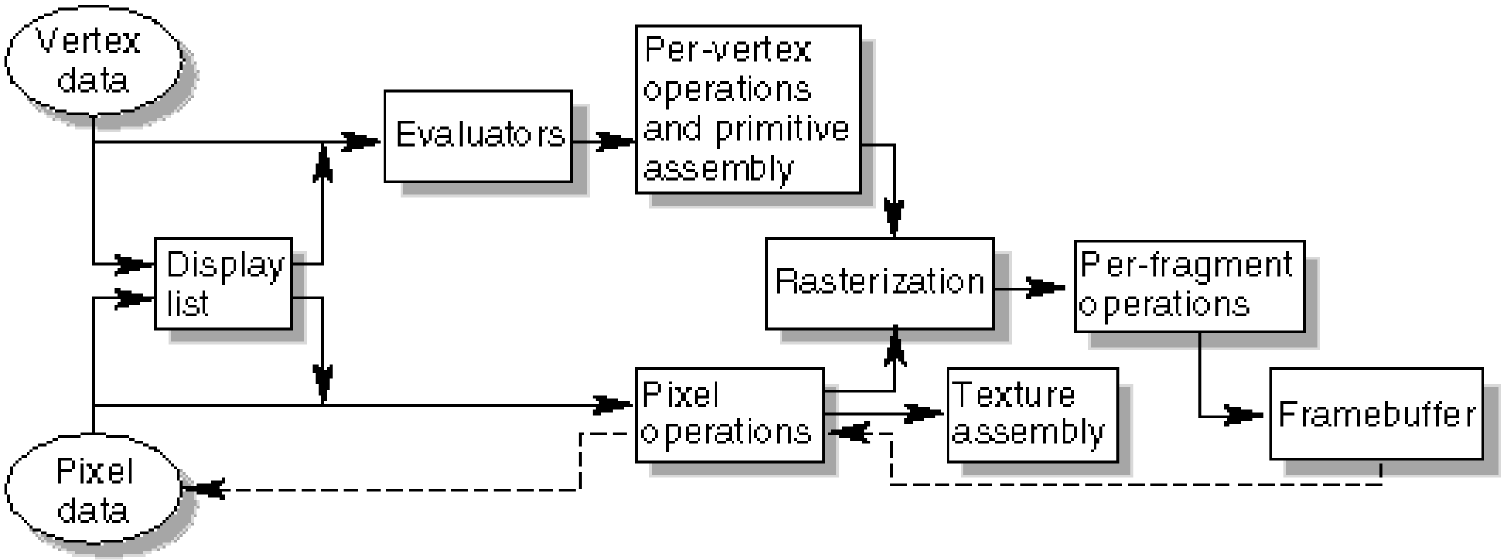
(David Ebert, 1995)

Rendering Pipeline



Transformation sequence from modeling coordinates to device coordinates.
Source: Computer Graphics with OpenGL
By Hearn & Baker

OpenGL



Source: OpenGL Programming Guide
http://glprogramming.com/red/

History of Graphics Cards

- IBM VGA (Video Graphics Array) – 1987
  - “dumb” frame buffer due to CPU’s responsibility to update all the pixels
- Silicon Graphics, Evans & Sutherland
  - Vertex transformation, texture mapping
  - Very expensive, fail to achieve mass-market success
- First Generation GPU - 1998
  - NVIDIA’s TNT2, ATI Rage, 3dfx’s Voodoo3
    - Able to rasterize pre-transformed triangles and two textures
    - DirectX 6
    - Limitation during vertex transformation in CPU, not GPU
- Second-generation GPU - 2000
  - NVIDIA GeForce 256, ATI Radeon 7500 etc.
  - 3D vertex transformation and lighting
- Third-generation - 2001
  - NVIDIA’s GeForce 3 & 4, MS Xbox, ATI Radeon 8500
  - Vertex-programmability
- Fourth-generation
  - NVIDIA GeForce FX-CineFX, ATI Radeon 9700
  - Both vertex-level and pixel-level programmability

Source: Mathematics for 3D Game Programming & Computer Graphics by Eric Lengyel (2004)

GPU (Graphics Processing Unit)

- Terminology
  - CPU are general purpose. GPU is much faster at graphics tasks and process tens of millions of vertices per second and rasterize hundreds of millions of fragments per second. GPU cannot execute general-purpose program that CPU can.
  - Shading language: Ability to programmatically control the shape, appearance, and motion of objects rendered using graphics hardware.
  - Cg: C for graphics (NVIDIA)
  - HLSL: High-Level Shading Language (Direct X 9.0 and above)
  - Different from conventional programming language because being based on flow computational model. Computation occurs in response to data flowing through a sequence of processing steps.

Source: The Cg tutorial by Fernando & Kilgard (2003)

Demo

- Stanford Real-time shader

Vertex & Rasterization - Fragment Operation

- Vertex Operation
  - In object space: Vertices stored local to particular model
  - In world space: Position and orientation of each model stored
  - Camera space, Clip space, window space
  - Per-vertex lighting
    - Calculate color and intensity of light reaching each vertex and how much reflected toward camera.
    - Interpolating areas where colors assigned to each vertex
  - Texture coordinates
    - Interpolate over areas of primitive and look colors in a texture map

- Rasterization & Fragment operation
  - Rasterization
    - Process of determining set of pixels covered by geometric primitives.
  - Fragment operation
    - Combined with pixel information after GPU calculates depth, interpolated vertex colors and textures.

Source: Mathematics for 3D Game Programming & Computer Graphics by Eric Lengyel (2004)

CPU & GPU

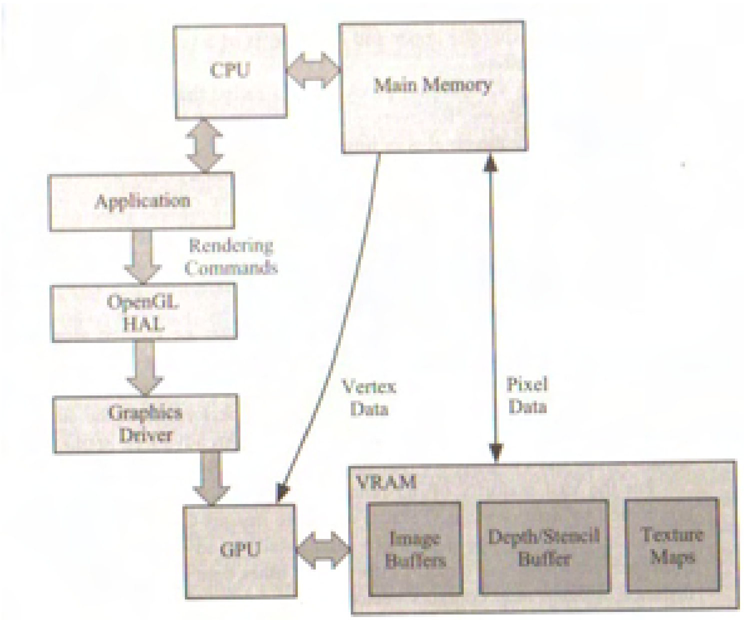


Figure 0.2 The communications that take place between the CPU and GPU.

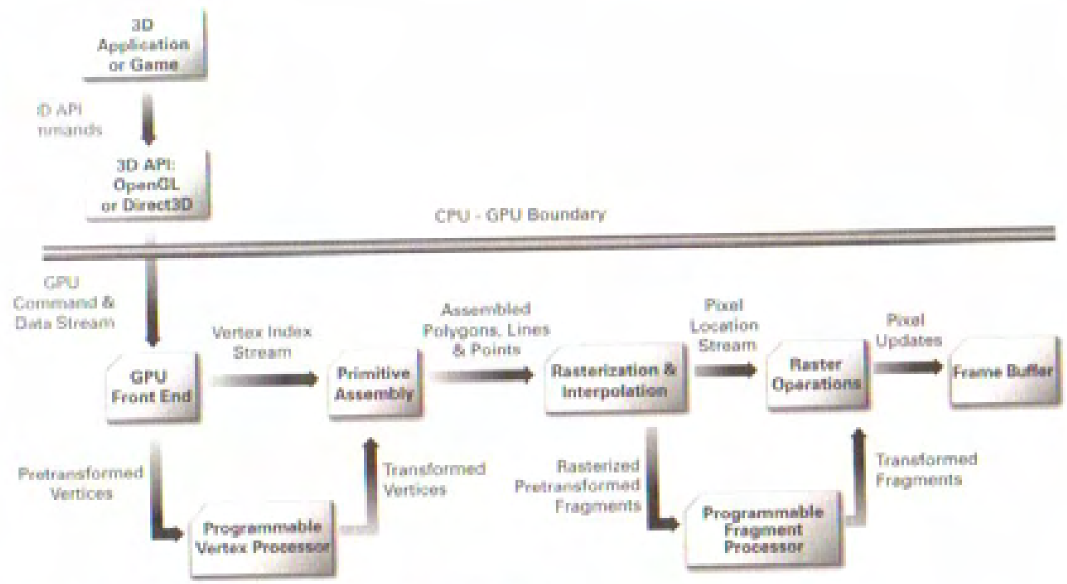
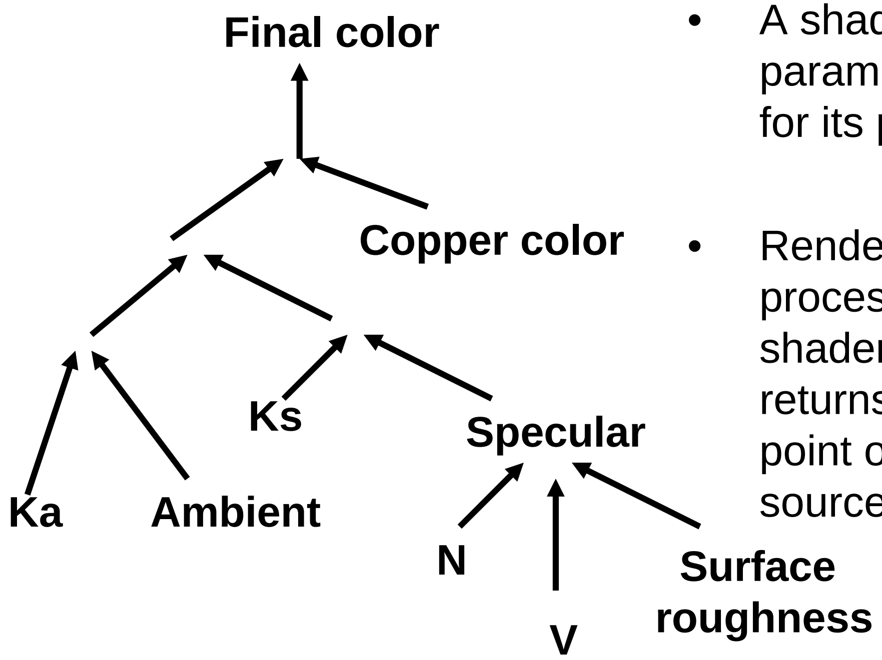


Figure 1-7. The Programmable Graphics Pipeline

1.2 Vertices, Fragments, and the Graphics Pipeline

Shade Tree (Cook, 1984)



- A shade tree is a tree of nodes, each of which takes parameters from its children and produces parameters for its parent.
- RenderMan defines a set of key places in the rendering process at which user-defined or system-defined shaders can be called. For example, a surface shader returns the light reflected in a specified direction given a point on the surface, its orientation, and a set of light sources.

Source: Computer Graphics: Principles and Practice
By Foley et al. (1995)

Renderman REYES (Renders Everything You Ever Saw')

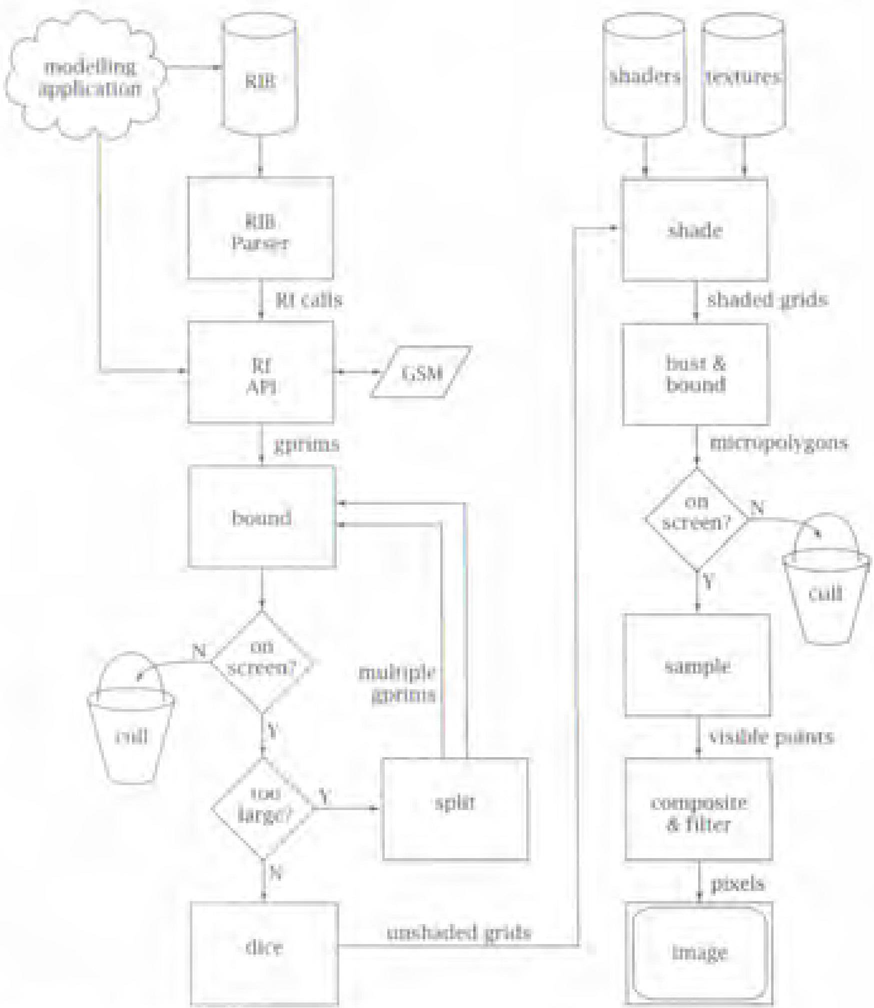


Figure 6.1 The REYES rendering pipeline.

Source: Advanced RenderMan by Apodaca et al. (1999)

An architecture is presented for fast high-quality rendering of complex images. All objects are reduced to common world-space geometric entities called micropolygons, and all of the shading and visibility calculations operate on these micropolygons. Each type of calculation is performed in a coordinate system that is natural for that type of calculation. Micropolygons are created and textured in the local coordinate system of the object, with the result that texture filtering is simplified and improved. Visibility is calculated in screen space using stochastic point sampling with a z buffer. There are no clipping or inverse perspective calculations. Geometric and texture locality are exploited to minimize paging and to support models that contain arbitrarily many primitives.

Source: The Reyes Image Rendering Architecture By Cook et al. (1987)

- Renderman constraints on the view-reference coordinate system, requiring that the origin be at the center of projection and the view plane normal be the z axis, called eye-coordinate system.
- Reyes chops all objects into micropolygons: small constant-shaded quadrilaterals approximately ½ pixel on a side, called dicing. Dicing is performed prior to perspective transformation based on an estimate of the size of the resulting micropolygons after projection.

Source: Computer Graphics: Principles and Practice By Foley et al. (1995)

Cg Shader

- Do more than just shading. CG can perform physical simulation, compositing, and other shading tasks.
- Cg enables parallel processing.
- Prior to 2001, graphics hardware was hard-wired (algorithms fixed within the hardware)
- Cg gives the advantage of a high-level language such as C while delivering the performance of low-level assembly code.

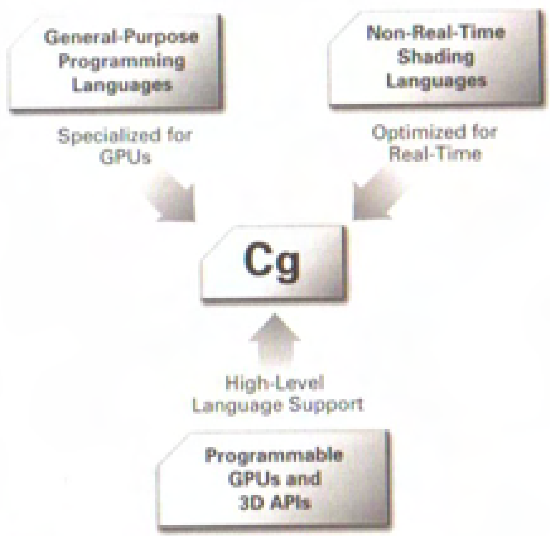


Figure 1-10. Sources of Cg's Technology Heritage

1.3 Cg's Historical Development

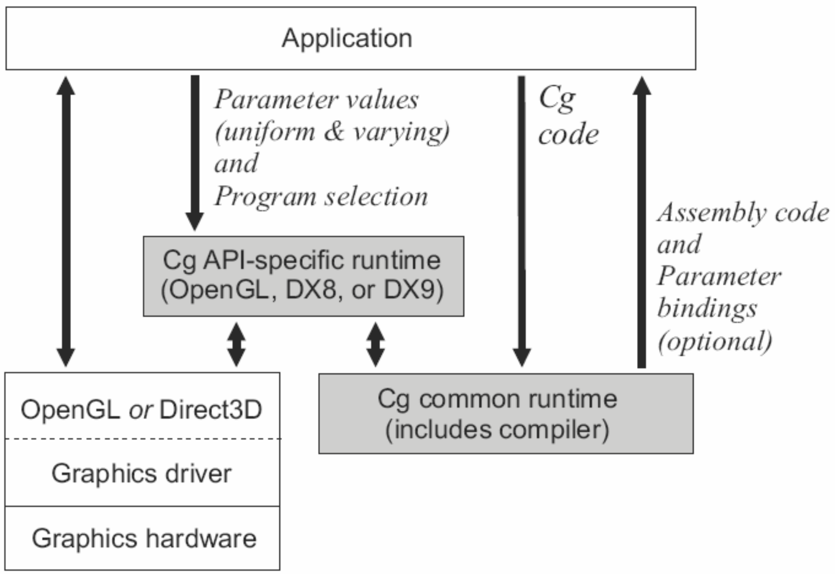


Figure 3: *Cg system architecture*

Cg Development

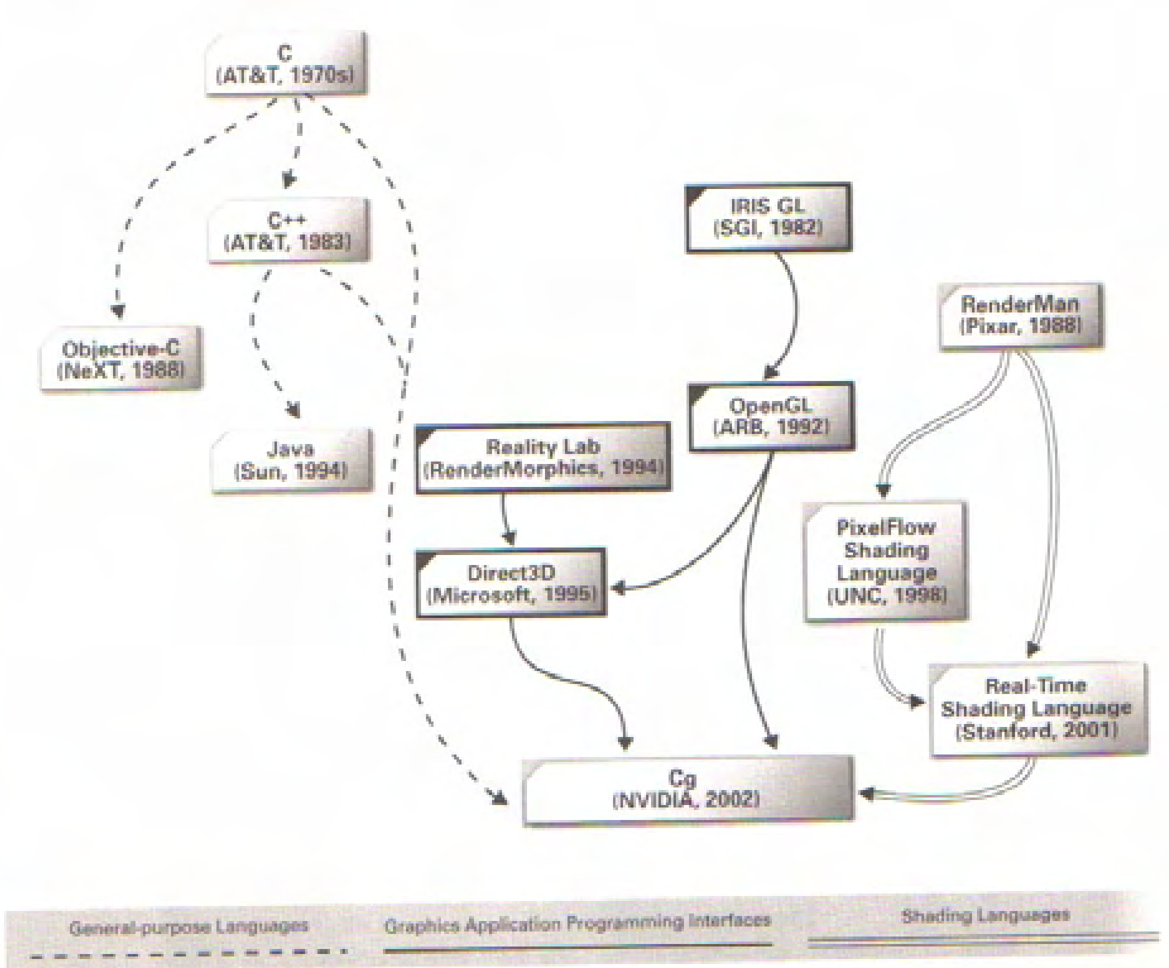
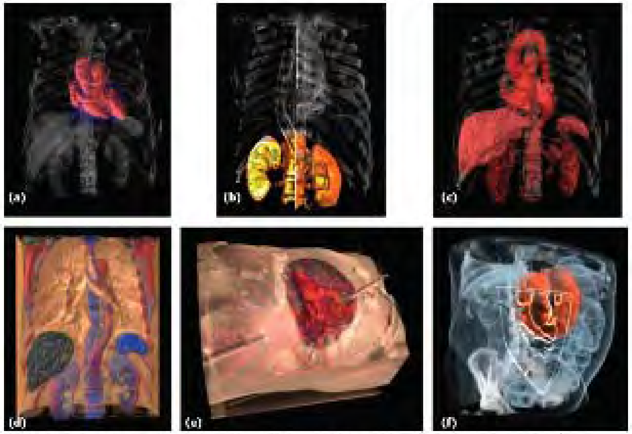


Figure 1-11. Inspirations for Cg's Development

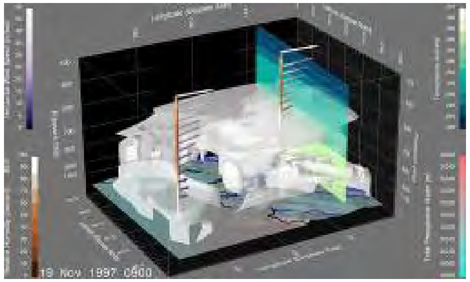
- Is there a need for Interactive NPR (non-photorealistic rendering) or PR (photorealistic rendering)?

- Diversity of Image/Rendering Styles
  - Traditional Art
  - InfoGraphics
  - Medical Imaging
  - Animated Motion Picture
  - Photorealistic/NPR Motion Picture
  - Scientific Visualization

Goal? Effective visual communication between producers and users

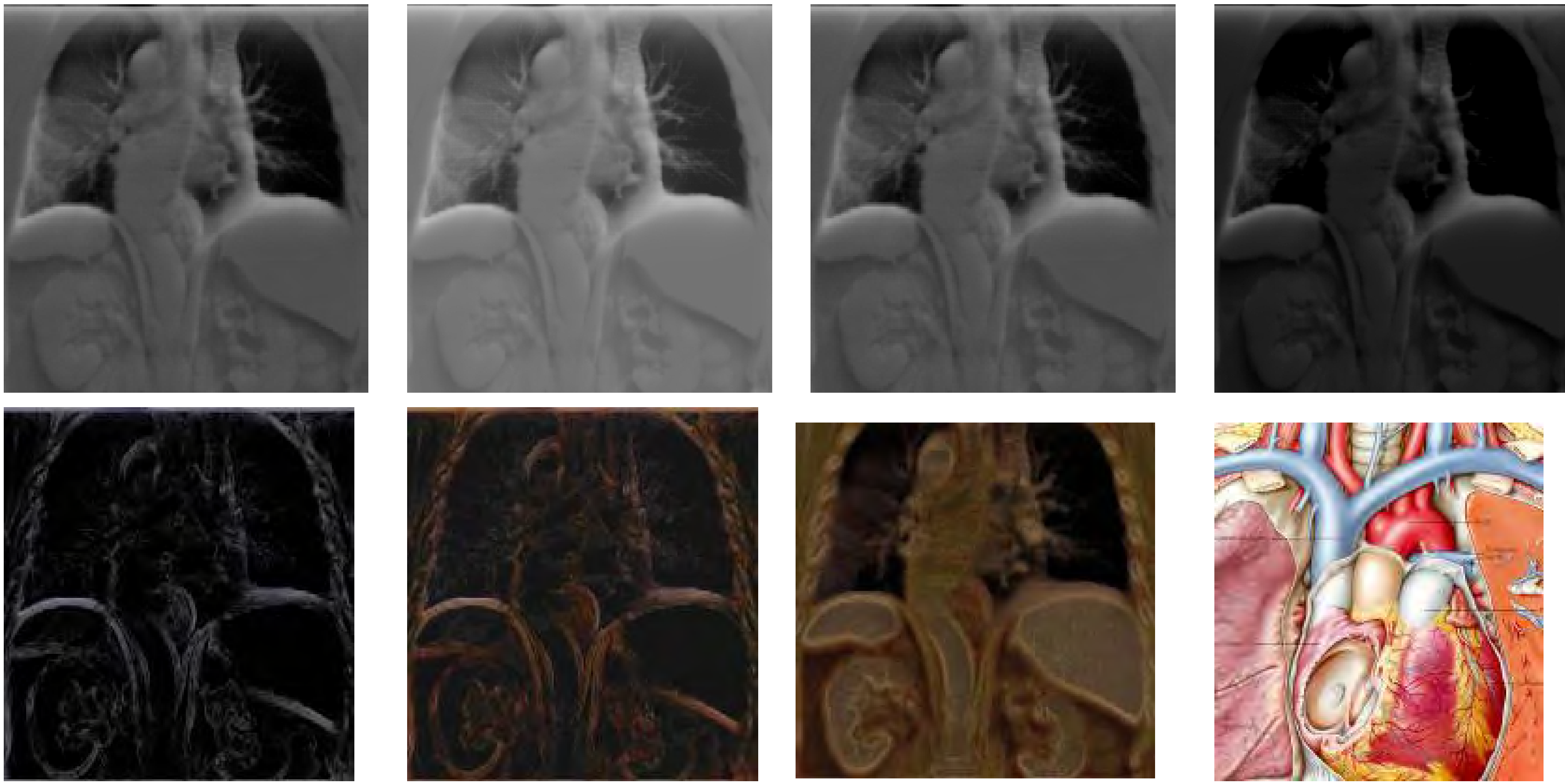


11 Abdomen CT data set illustrations: (a) heart chambers; (b) kidney structure; (c) circulatory system, liver, spleen, and kidneys; (d) false-colored chest and abdominal cavity illustration; (e) heart chambers surgical illustration; and (f) colon illustration with bent region highlighted and cut with clipping planes.



Abdominal CT Results

Which image should be used for cancer diagnosis of our patient?

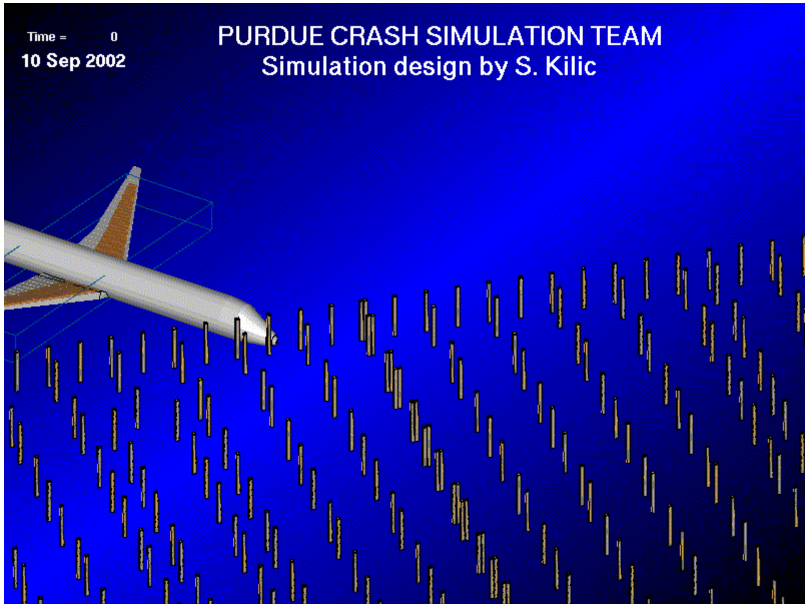


Are we going to have the same answer? How to evaluate the validity of everyone’s decision that consist of bias, knowledge, and physiological noise? Do image rendering styles matter?

Does data size matter to analyze 3D images?

- Physics-based simulation based on actual data from Pentagon, USA
- Handling tera-bytes of data

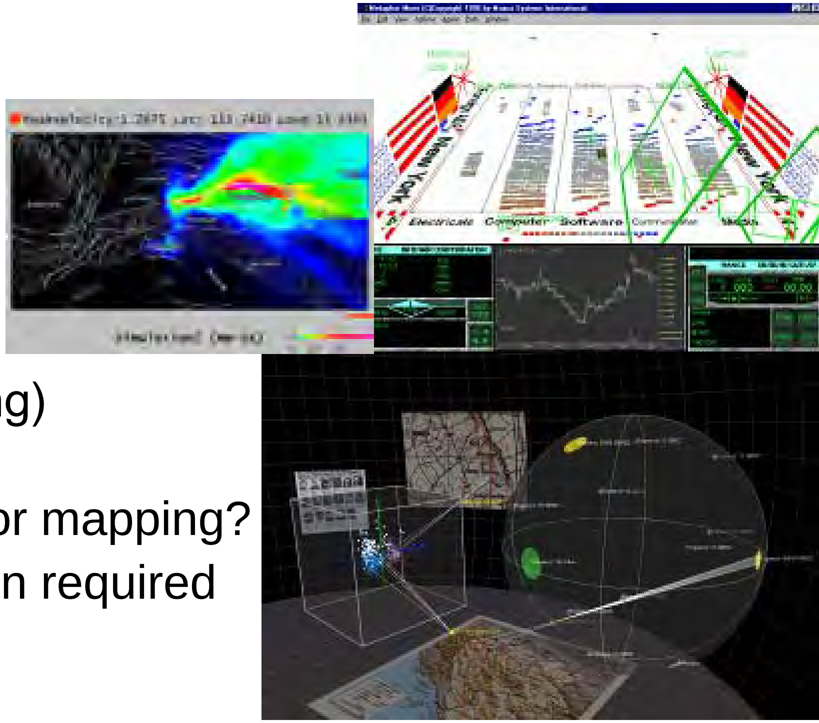
- Is it necessary to add high-quality photorealism or 3D interactivity for visual analysis?



Simulation of 9/11 attack (Hoffman et al, 2002)

Time/accuracy-critical mission, multiple dimension mapping

1. military simulation
2. air-traffic control
3. national security
4. natural disaster (earthquake, Katrina)
5. stock market
6. medical imaging
7. feature film/video games (release timing)



- How many factors? Multidimensional color mapping?
- Extreme measurement techniques is often required

  I.e) identify 6 pixel levels

- Each data point represent such as stock quote, virus infection, cancer positioning, impact of turbulence in airplane and so on.

Misinterpretation and misrepresentation of image rendering/interaction could severely affect a doctor's judgment and as a consequence, it could jeopardize a patient’s life. Or crash air jet fighter.

Different Realism/Effectiveness in Graphics

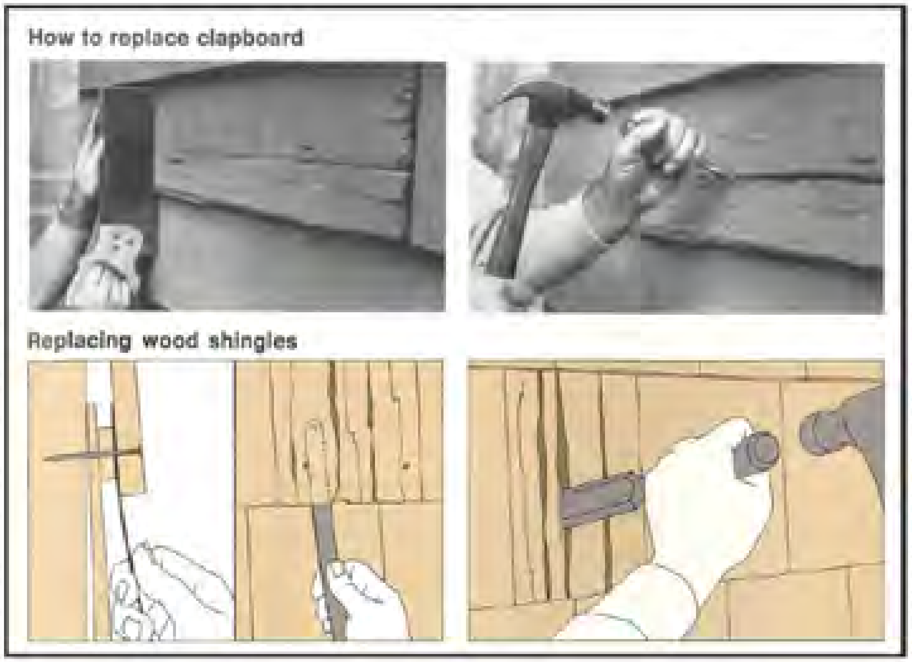
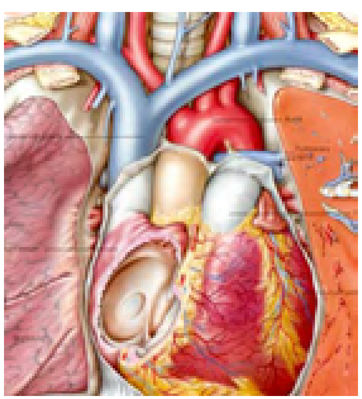
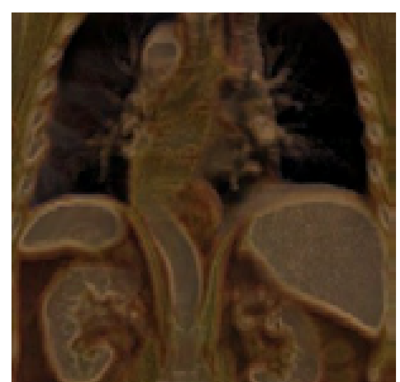


Figure 2: Functional realism in technical illustrations. Adapted from [18].

- Graphics categorization -
  - Physical realism
  - Photorealisitc realisim (psychophysics-based)
  - Functional realism

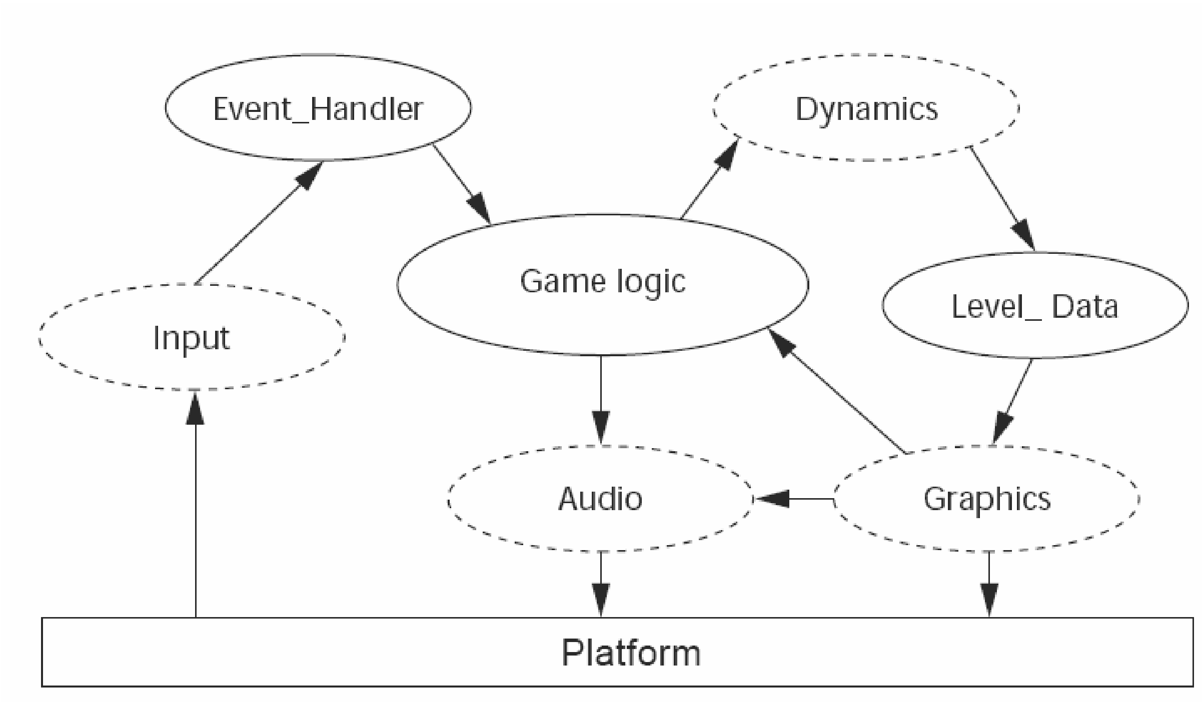




David Ebert (2001)

Game & VR Simulation

Game Engine



Source: Designing a PC Game Engine by Bishop et al (1999) IEEE CG & A

VR system

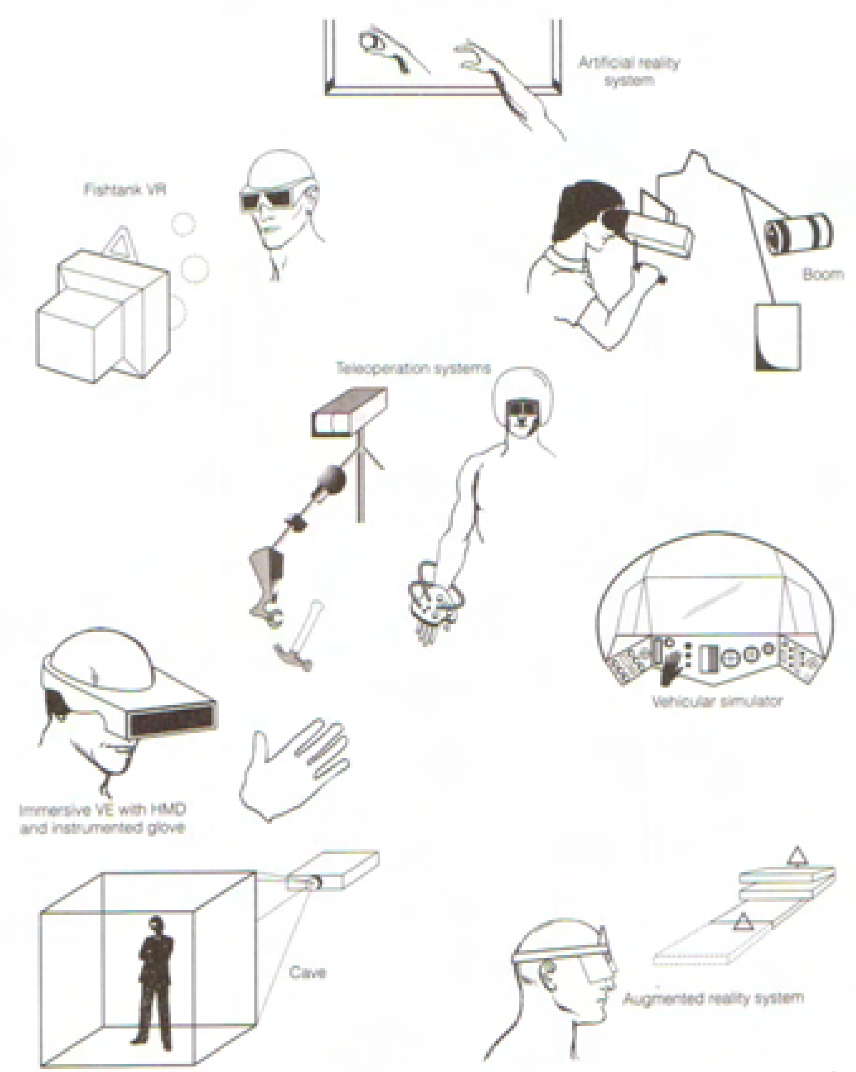


Figure 1.3 Types or "flavors" of virtual environments. (E.H.H. Stuart)

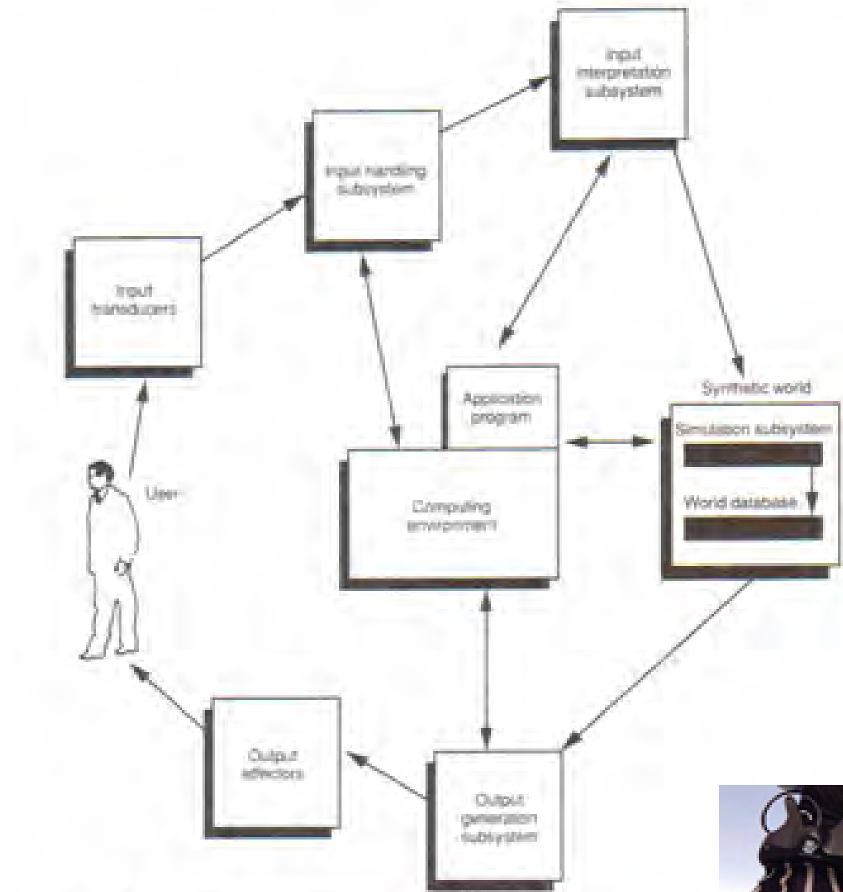
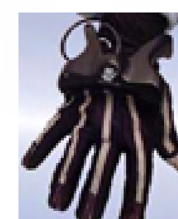


Figure 1.1 Virtual environment system, with coupled simulation



Source: Design of Virtual Environments by Rory Stuart

VR Application

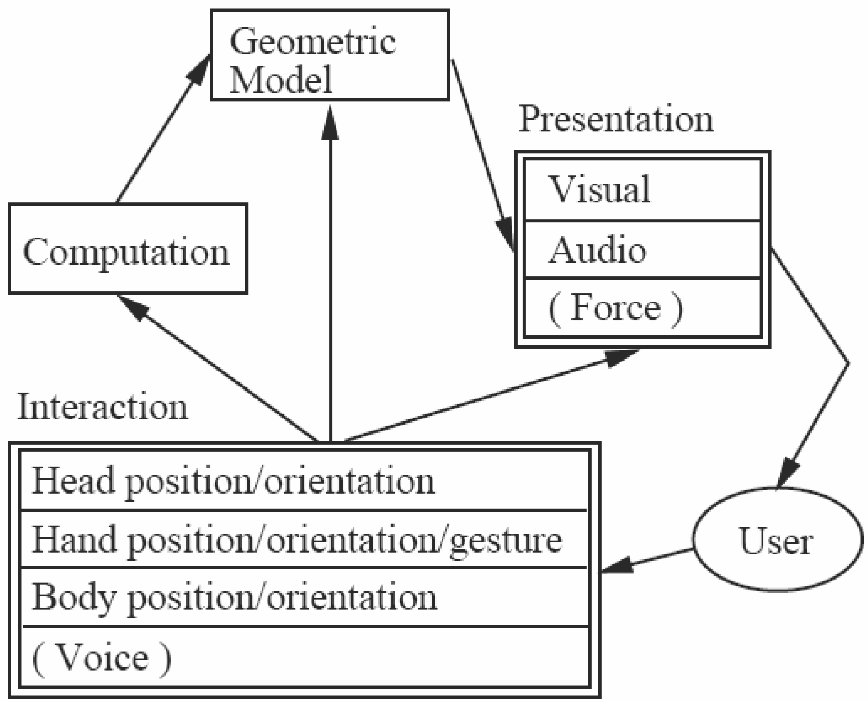


Figure 1: Decoupled Simulation Model.

Source: The Decoupled Simulation Model for Virtual Reality System by Shaw et al. (1992)

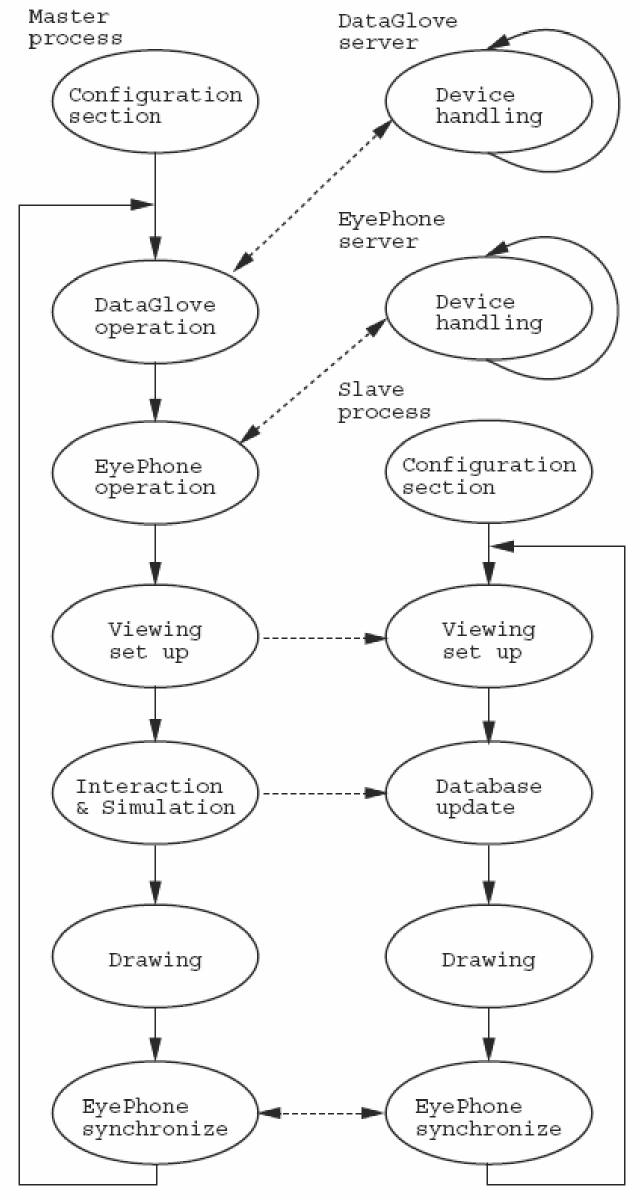


Figure 3: A typical flowchart for an MR application.

VR Input & output

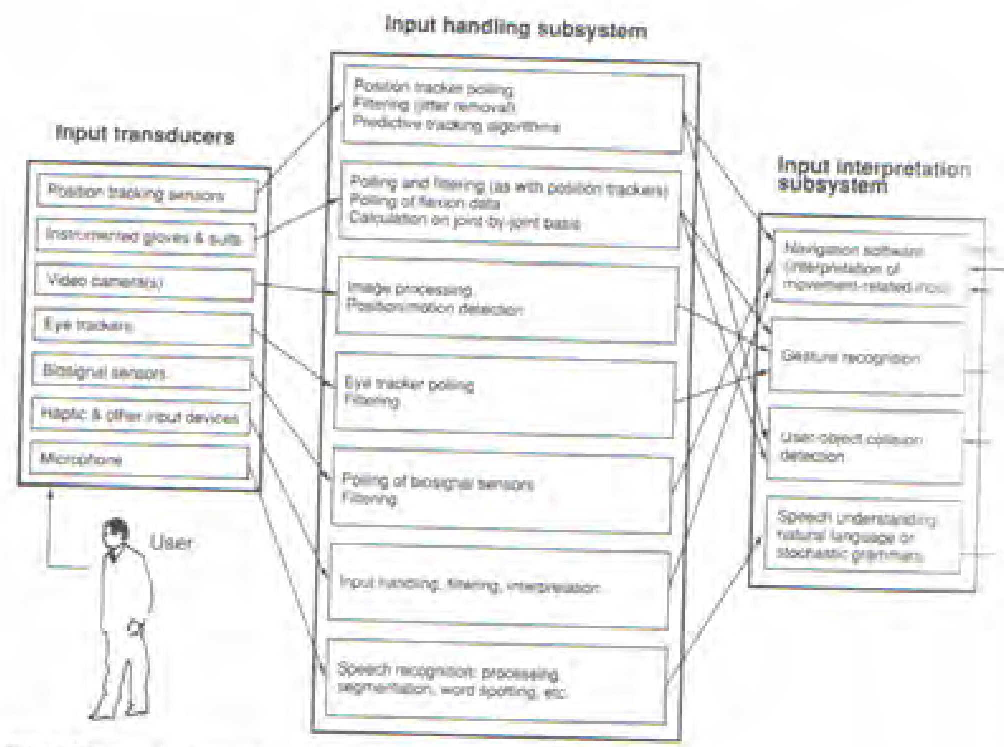


Figure 4.1 Input subsystem.

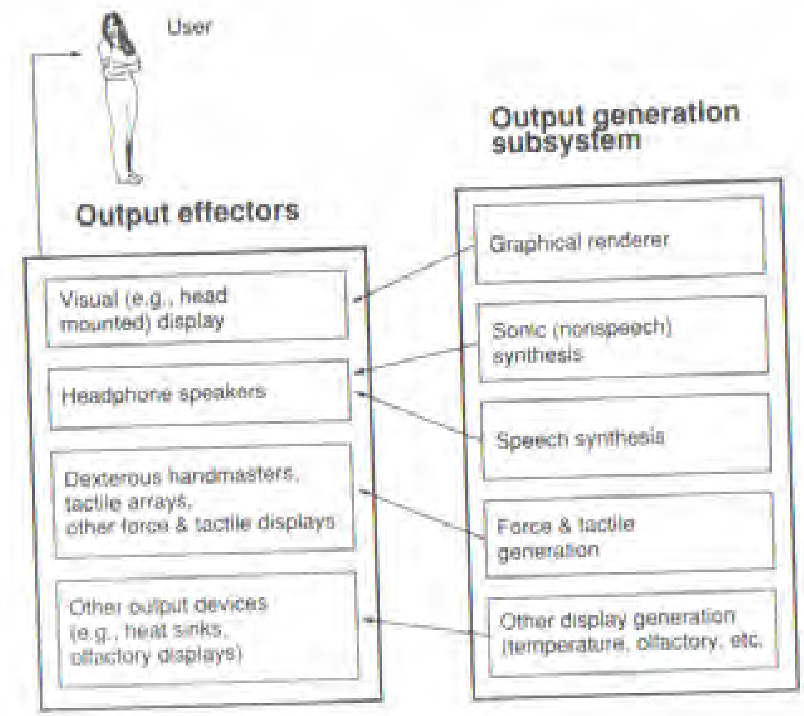


Figure 5.1 Output subsystem

Source: Design of Virtual Environments by Rory Stuart

Planning VR Project

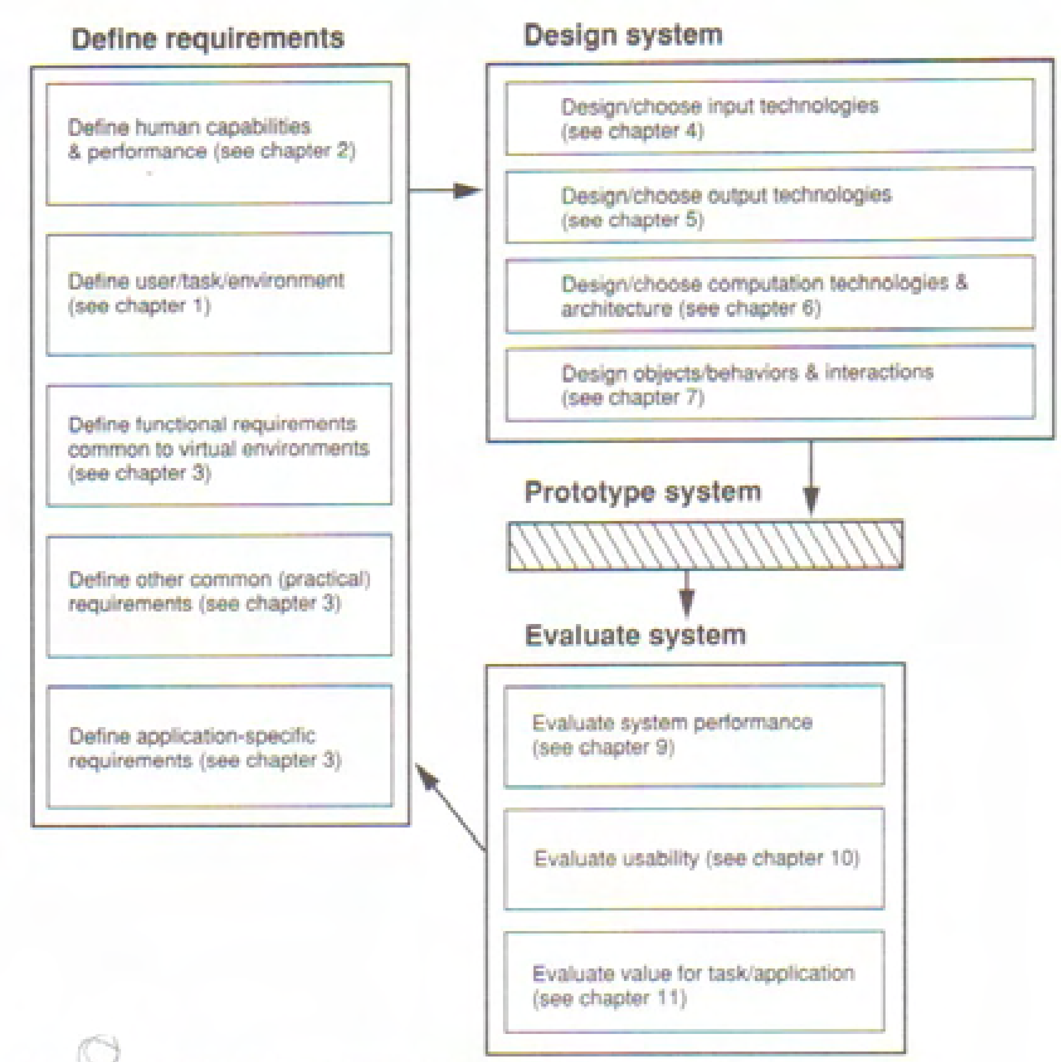


Figure 1.4 The iterative design process applied to the design of virtual environments.

Source: Design of Virtual Environments by Rory Stuart