Computer Graphics

Painting by numbers

Elmar Eisemann

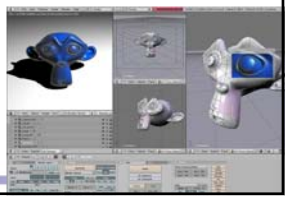
Delft University of Technology (TU Delft)

Introduction

- Computer Graphics
  - part of computer science
  - Concerning the manipulation and creation of visual and geometric information with a computer

Not: using Photoshop

Instead: making Photoshop

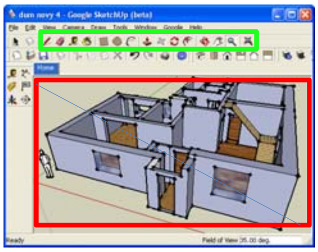


What is it about?

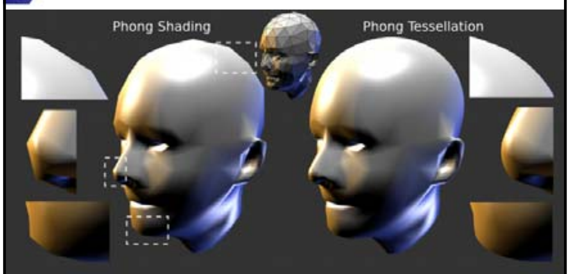
- Modeling - Making content
- Animation - Making movement
- Rendering - Making images

Modeling

- Create Models
  e.g.,
  - Geometry
  - Analysis
  - Representations



Modeling



What is Computer Graphics about?

- Modeling - Making content
- Animation - Making movement
- Rendering - Making images

Animation

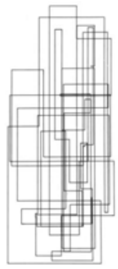
- Synthesize Movement
  e.g.,
  - Data analysis (e.g., PCA)
  - Data Interpolation
  - Differential analysis
  - Physics

What is Computer Graphics about?

- Modeling - Making content
- Animation - Making movement
- Rendering - Making images

Rendering

- Making images
  e.g.,
  - Physics
  - Math
  - User interfaces
  - Perception
  - Electrical Engineering
  - ...



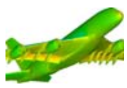
VERTICAL-HORIZONTAL NUMBER THREE (1964) BY A. MICHAEL NOLL

Introduction

- Computer graphics has many applications
  - Scientific Visualization
  - Architecture/Design
  - Training/Simulation
  - Remote Surgery
  - Movies
  - Games
  - ...

Ice Age, Blue Sky

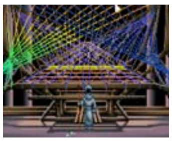
Ebert, Purdue University



Deutsches Zentrum für Luft und Raumfahrt

Introduction

- Graphics advances at an incredible pace

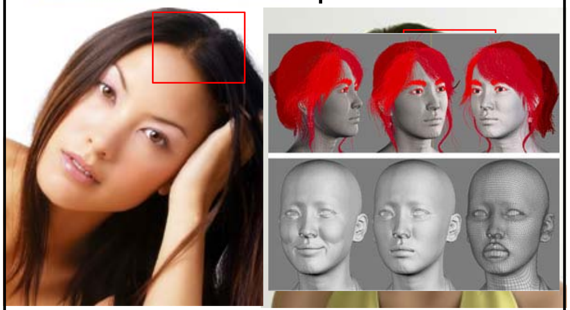


1978 – Space Invaders 1990 - Loom 2007 - Unreal

Photo or Computation?

Photo or Computation?

Photo or Computation?



The holy grail has been reached: computer-generated imagery of the human face that's indistinguishable from reality. – Charlie White

Can we do better?

- Yes, we can!

It should not…
…take months of work
…take hours of computation
…only result in
one view, pose, and light !

Extreme Computation Times

- Big Hero Six – copyright Disney

But is it really worth the effort?

Financial Facts

- To convince all those bankers… ;)

Financial Facts - Modeling

- GTA 4:
  - Roughly $ 90 Mio. in content production

Financial Facts - Animation

- Avatar:
  $ 460 Mio.

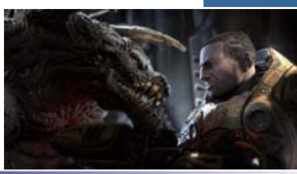
~$50.000 / second
~$ 2.000 / image

Animation:
~$ 700 / image
(non-confirmed)

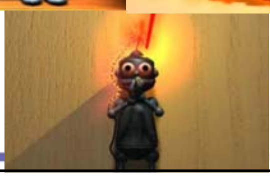
Financial Facts - Rendering

- Despicable me:
  500.000 € for electricity

What do we get for it ?

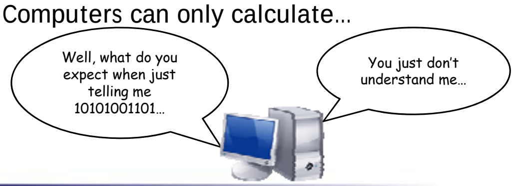


What do we get for it ?



How to produce an image?

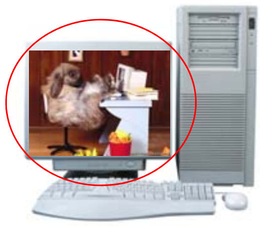
- Computers can only calculate…



Today

- How to make images on a computer?

Making images with a Computer



Pixels – Picture elements

Pixels – Picture elements

- A colored pixel has typically
  Red Green Blue values.

123

- We color by numbers…
  sounds simple…
  but choosing the values can be difficult

Producing Images in the Real World

- Albrecht Dürer, 16th century

Producing Images in the Real World

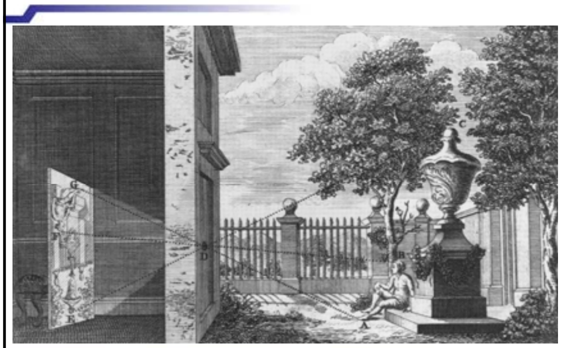
- Albrecht Dürer, 16th century

Producing Images in the Real World

- Albrecht Dürer, 16th century

Producing Images in the Real World

Producing Images in the Real World



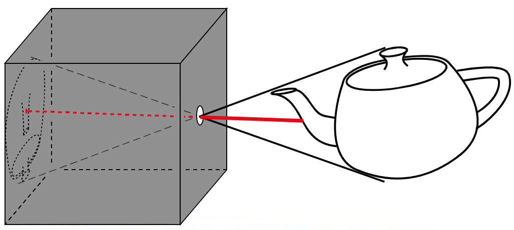
Camera obscura

Producing Images in the Real World

- A photo of such a camera [Abellardo]

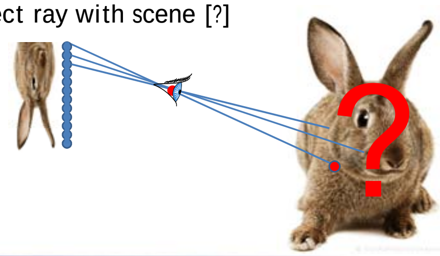
Pinhole camera

- Box with hole
- Perfect image for “point-sized” hole



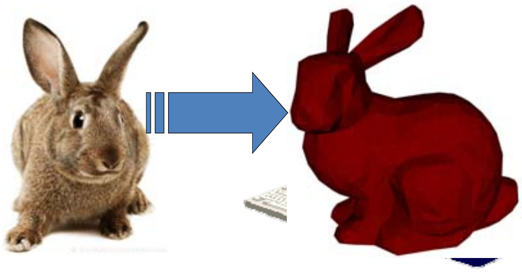
Virtual Camera

- Take a pixel on the image in the virtual world
- Compute ray through pixel and camera center
- Intersect ray with scene [?]



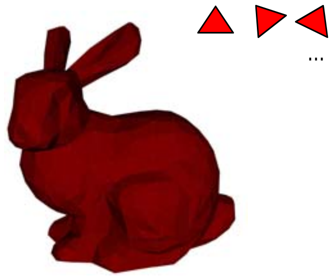
Models

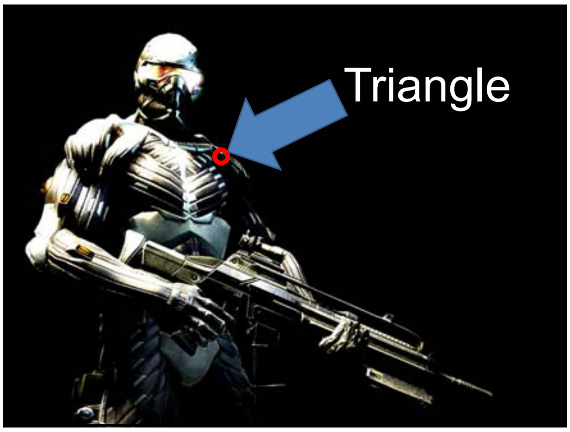
- Representing reality



Models

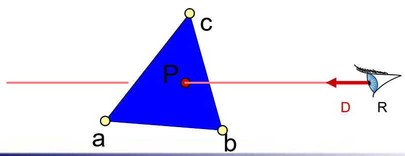
- Models are typically lists of triangles





Solve Equations

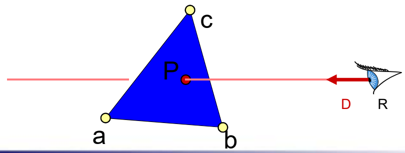
- R+tD= a+β (b-a)+γ (c-a)



Solve Equations

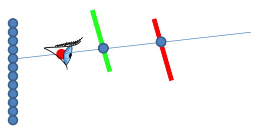
- $R_x+tD_x= a_x+\beta (b_x-a_x)+\gamma (c_x-a_x)$
- $R_y+tD_y= a_y+\beta (b_y-a_y)+\gamma (c_y-a_y)$
- $R_z+tD_z= a_z+\beta (b_z-a_z)+\gamma (c_z-a_z)$

$$\begin{bmatrix} a_x-b_x & a_x-c_x & D_x \\ a_y-b_y & a_y-c_y & D_y \\ a_z-b_z & a_z-c_z & D_z \end{bmatrix}\begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix}=\begin{bmatrix} a_x-R_x \\ a_y-R_y \\ a_z-R_z \end{bmatrix}$$



Produce Final Image

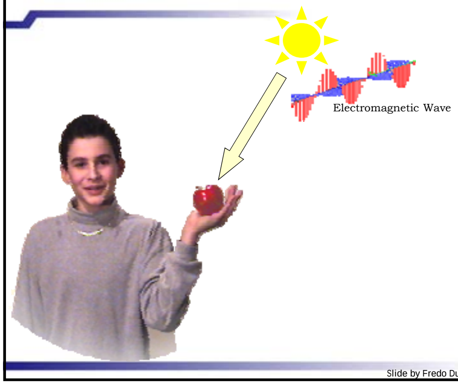
- Keep the closest intersection point



Ray Tracing - Recap

```
For each pixel
    Distance=MAX
    Color=0
    Ray=computeRay(pixel)
    For each triangle
        (CurrColor,CurrDistance)=computeIntersection(Ray)
        If (CurrDistance<Distance)
            Distance=CurrDistance
            Color=CurrColor
```

What is Color?



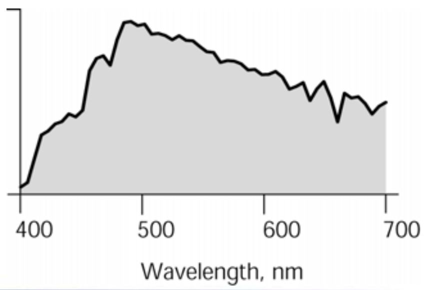
Slide by Fredo Durand

How do we see?

- We have two eyes…

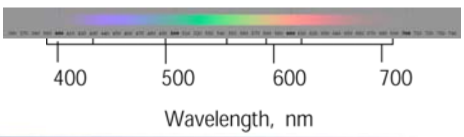
Color - Physical Definition

- Color = Distribution of power over a spectrum

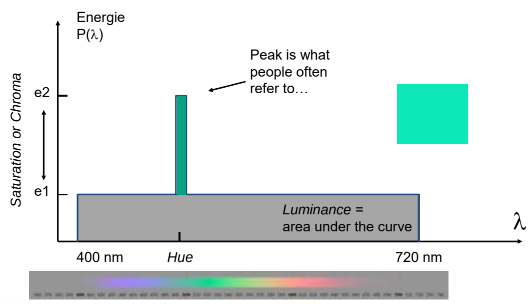


Light Spectrum

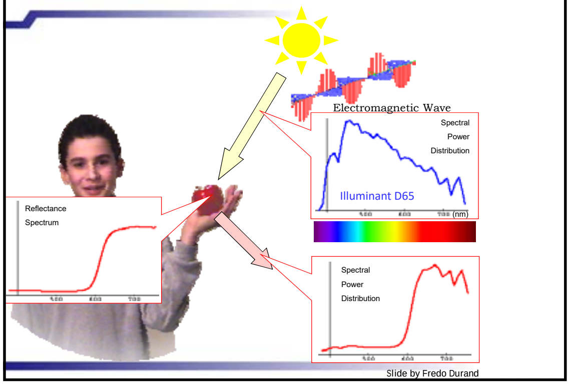
- Visible color between 380 nm (violette) and 720 nm (red)
- Outside visible range
  - Below 380 nm : ultra-violet
  - Above 720 nm : infra-red



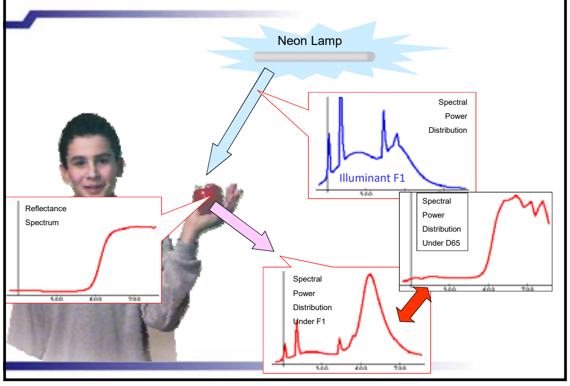
Simple example



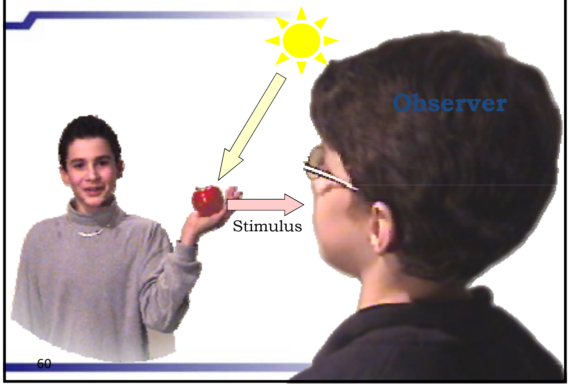
What is Color?



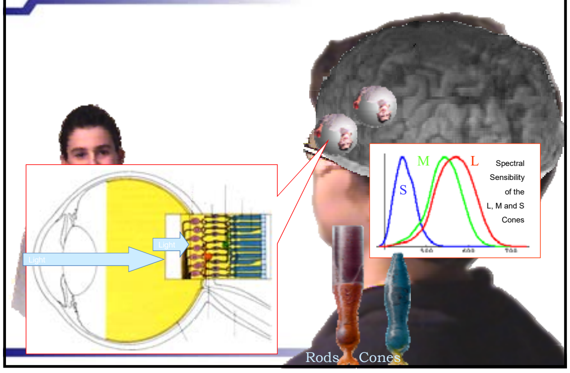
What is Color?



What do we see ?



Eye Biology



Eye Biology

- Cones :
  - Chromatic perception (3 types-LMS)
  - Concentrated in center of retina
  - 6 to 7 million in retinal center
    - 3 times full HD

- Rods :
  - Achromatic perception
  - Low-light vision

Night Vision

- During night, all cats are gray… Cones shut off!

Night Vision

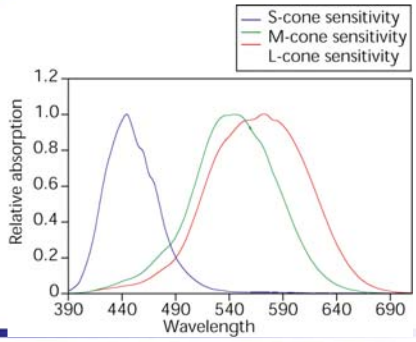
- Who sees noise in the night?


During the day

- Rods saturate!
- All that we have left are cones…

- For “color”, cones are the interesting case…

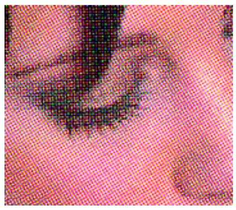
3 Cone types



3 Cone types

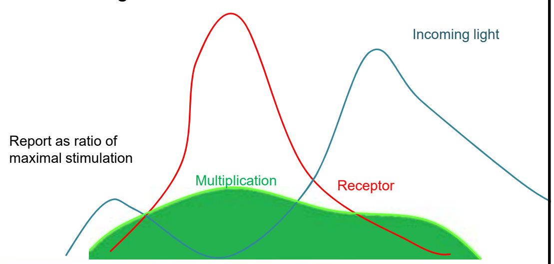
- 3 cone types explain why most display systems rely on 3 components: Red Green Blue
  or Cyan Magenta Yellow


RGB screen


CMY print

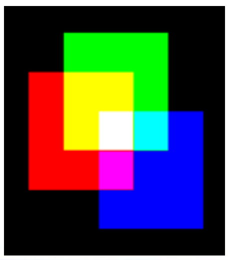
Receptor and Incoming Light

- Multiply incoming light and receptor and integrate

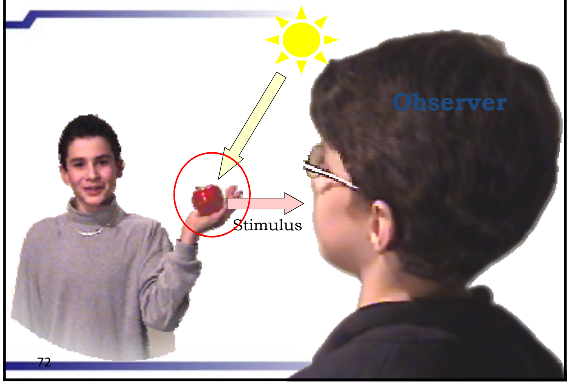


Eye Biology

- The eye “adds energies”
- Cone information is combined

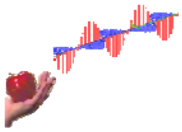


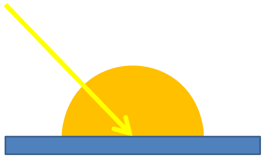
What do we see ?



Reflection

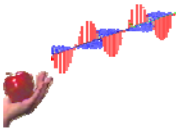
- What happens when the light hits a surface?

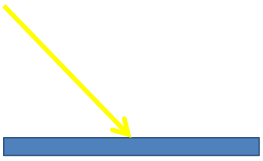




Reflection

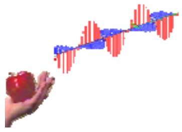
- What happens when the light hits a surface?

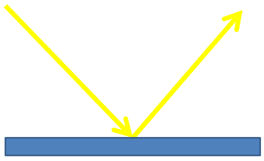




Reflection

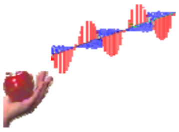
- What happens when the light hits a surface?

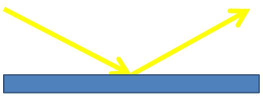




Reflection

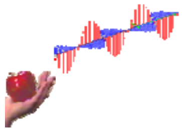
- What happens when the light hits a surface?

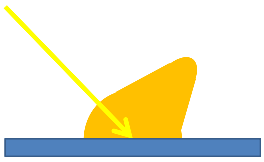




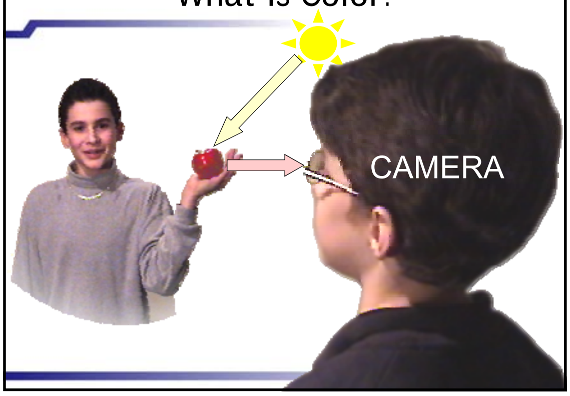
Reflection

- What happens when the light hits a surface?

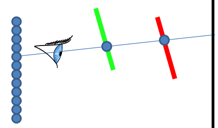




What is Color?



Ray Tracing - Cost



```
For each pixel
    Distance=MAX
    Color=0
    R=computeRay(pixel)
    For each triangle
        (CurrColor,CurrDistance)=testIntersection(R)
        If (CurrDistance<Distance)
            Distance=CurrDistance
            Color=CurrColor
```

Performance Analysis

- Stupid implementation:
- Ray Tracing:
  Cost = Pixels * Triangles

e.g., 100.000 triangles and a 1000^2 screen:
  Raytracing: 100.000 * 1.000.000 = 10^11

Performance Analysis

- Smart implementation:
- Ray Tracing:
  Cost = Pixels * log(Triangles)
  + building a structure

e.g., 100.000 triangles and a 1000^2 screen:
  Raytracing: 1.000.000 * 5 + X = 5 * 10^6

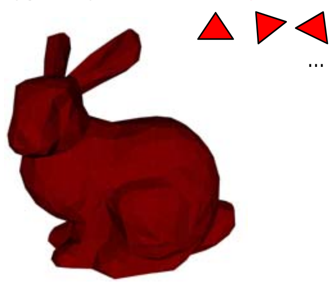
What about real-time (30 Images/Sec)

“building a structure” is slow
  (if you are not hypersmart…
  and even then it might be…
  but things can always change)

Alternative approach:
  Rasterization via the Graphics Pipeline

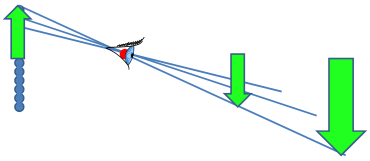
Simplified Graphics Pipeline

- Models are typically lists of triangles



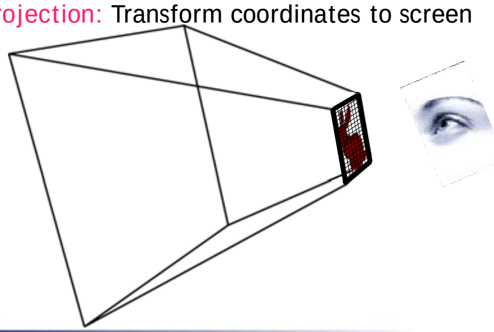
Virtual Camera

- Camera Plane in front of the eye



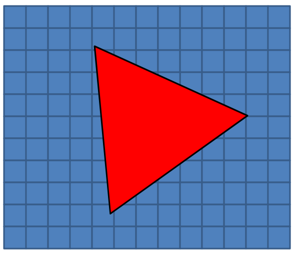
Simplified Graphics Pipeline

- Projection: Transform coordinates to screen



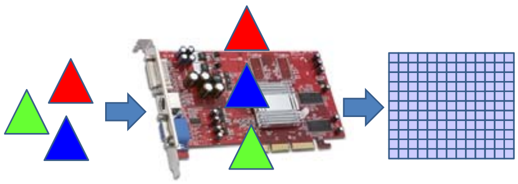
Simplified Graphics Pipeline

- Rasterization: Fill screen pixels



Simplified Graphics Pipeline

- Highly parallelizable ⇒ GPUs

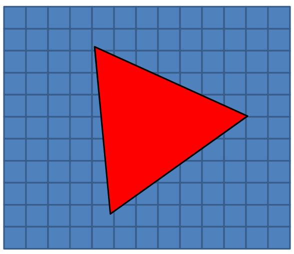


128 processors working in parallel (in 2007)

…2015 thousands and more…

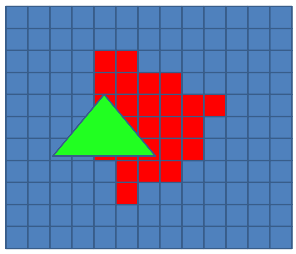
Simplified Graphics Pipeline

- Catch: Let’s look at a second triangle…



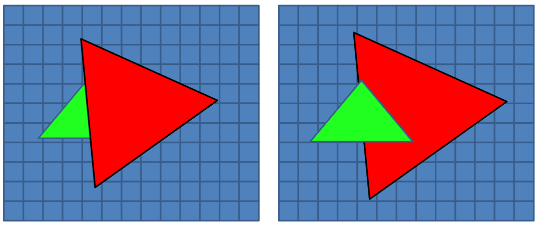
Simplified Graphics Pipeline

- Catch: Let’s look at a second triangle…



Simplified Graphics Pipeline

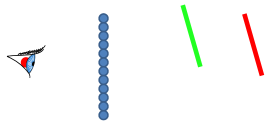
- Catch: Triangle drawing order changes result



As for ray tracing: need the closest triangle

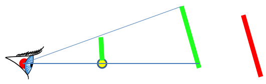
Simplified Graphics Pipeline

- Depth Test: Avoid sorting!
- Store a depth in each pixel



Simplified Graphics Pipeline

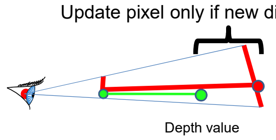
- Depth Test: Avoid sorting!
- Store a depth in each pixel



Simplified Graphics Pipeline

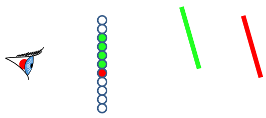
- Depth Test: Avoid sorting!
- Store a depth in each pixel

Compare new distance to stored distance
Update pixel only if new distance is nearer



Simplified Graphics Pipeline

- Depth Test: Avoid sorting!
- Store a depth in each pixel



Cost of Rasterization

Algorithm:
For each triangle
  projTri=projectTriangle(triangle)
  fillPixels(projTri)

Cost = Triangles + “drawn pixels”

Performance Analysis

Ray Tracing:
Cost = Pixels * log(Triangles) + structure
vs.
Rasterization:
Cost = Triangles + “drawn pixels”

e.g., 100.000 triangles and a 1000^2 screen:
Raytracing: X+5 * 1.000.000
Rasterization: 100.000 + “drawn pixels”
Raytracing/Rasterization : ~50

Performance Analysis

Ray Tracing:
Cost = Pixels * log(Triangles) + structure
vs.
Rasterization:
Cost = Triangles + “drawn pixels”

e.g., 100.000.000 triangles and a 1000^2 screen:
Raytracing: X+ 8 * 1.000.000
Rasterization: 100.000.000 + “drawn pixels”
Raytracing/Rasterization: ~0.1

… but Rasterization can be made smarter too…

What complexity do we work with?

- Today’s Games:
  - 200.000 triangles
- Today’s Movies
  - more than 1 Billion


Depth Buffering [1974]

- But only applied much later… Why?

Memory requirements
320x200 pixel -> 200 KB of memory !
2000x1000 pixel -> 6 MB of memory !

1974 : $314,573 /MB
1986 : $ 300 /MB
1993: $28 /MB (Nvidia)

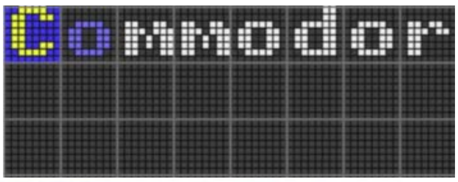
Bill Gates: 640 KB ought to be enough for everyone!

A Short History of Video Memory

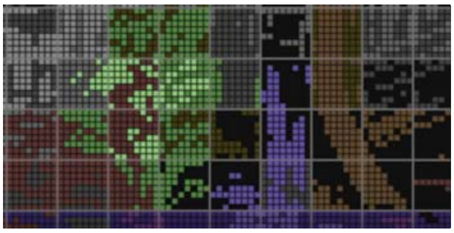
- Old days: Shared with CPU
- C64 = 64 KB of memory
- Example:
  - 320 x 200 Pixels at 1 bit is 8 KB
  - 320 x 200 Pixels at 4 bit (16 Colors) is 32 KB

Color Cells

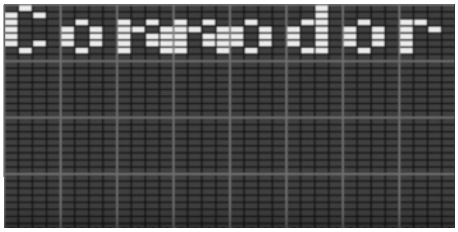
- Create 8x8 cells
- 2 colors/cell (foreground/background)
  1 Byte extra information (2*16 colors) per cell
- 9 KB for entire screen



Color Cells

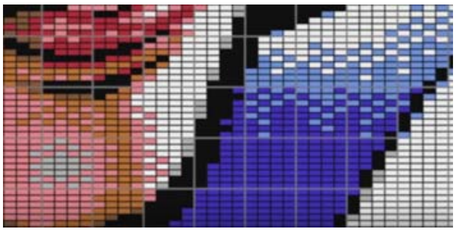


C64 Multi Color Mode



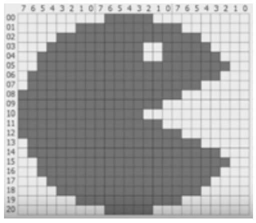
- Half the resolution, double the colors…

C64 Multi Color Mode



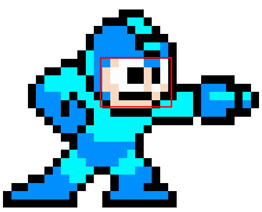
More Flexibility with Sprites

- Overlaid over the background graphics
- Often monochromatic (1 color + transparent)



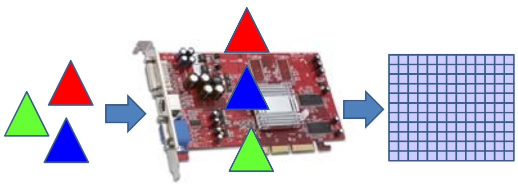
Sprites

- Nes: 4 colors per Sprite (one is transparent)



Simplified Graphics Pipeline

- Today: Up to 8 GB of RAM for Video Memory



Graphics Pipeline

- Local computations:

- Processor only knows its *current triangle* generally NOT enough

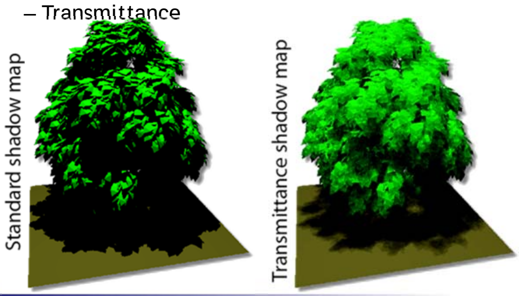
Non-Local Problems

- Challenges beyond local computations
  - Shadows

1996 – Super Mario 64

Non-Local Problems

- Challenges beyond local computations
  - Transmittance

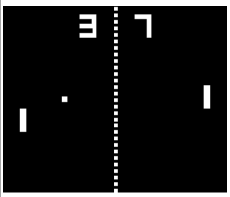


Non-Local Problems

- Challenges beyond local computations
  - Refraction/Translucency

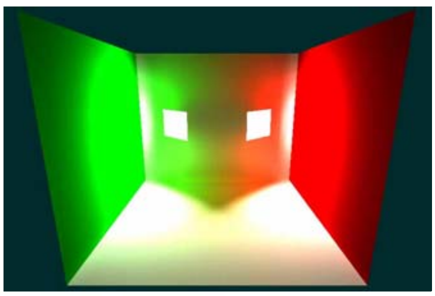
Non-Local Problems

- Challenges beyond local computations
  - Collision Detection



1972 - Pong

Global Illumination



Global Illumination

Non-Local Problems

- Challenges beyond local computations

Crysis (2007)

Resume

- Introduction to Graphics
  - What is Computer Graphics?
  - How do we BASICALLY perceive images/colors?
  - How to create images on a computer?
    - Ray tracing
    - Rasterization (+Depth Buffer and memory discussion)

Quiz for Today

- Why is rasterization usually faster than raytracing on today's game scenes?
- Name two physical phenomena that are difficult to reproduce with the standard graphics pipeline.
- Compute the intersection of the plane with normal 1/sqrt(2) * (1,1,0) through the origin and ray R(t):= (1,1,1) + t*(1,0,1).

Books

- Real-time Rendering
  by Tomas Akenine-Möller, Eric Haines,
  Naty Hoffman - Peters, Wellesley
- OpenGL Programming Guide
  Download by searching for "RED BOOK OpenGL"
- Real-Time Shadows
  by Elmar Eisemann, Michael Schwarz,
  Ulf Assarsson, Michael Wimmer
- Computer Graphics. Principles and Practice
  by James D. Foley, Andries VanDam,
  Steven K. Feiner

Thank you very much for your attention!

e.eisemann@tudelft.nl