Course on Virtual Reality

Vision

Aspects of Vision

- Human Factors
  - Anatomy of the Human Eye ✓
  - Perception: Depth Cues in Vision, ✓ Photometry
- Technology
  - Graphics Hardware
  - Head-Mounted Displays ✓
  - Room-Mounted Displays ✓ Stereo Techniques, Projectors, Screens, Brightness, …
- Algorithms
  - Geometrical Modeling
  - Rendering
    - Rendering Pipeline ✓
    - Stereo Projections
    - Viewer-Centered Projection
  - Global Illumination: Ray Tracing, Radiosity

Goal

Exact correspondence of visual perception in the real world and in the virtual world

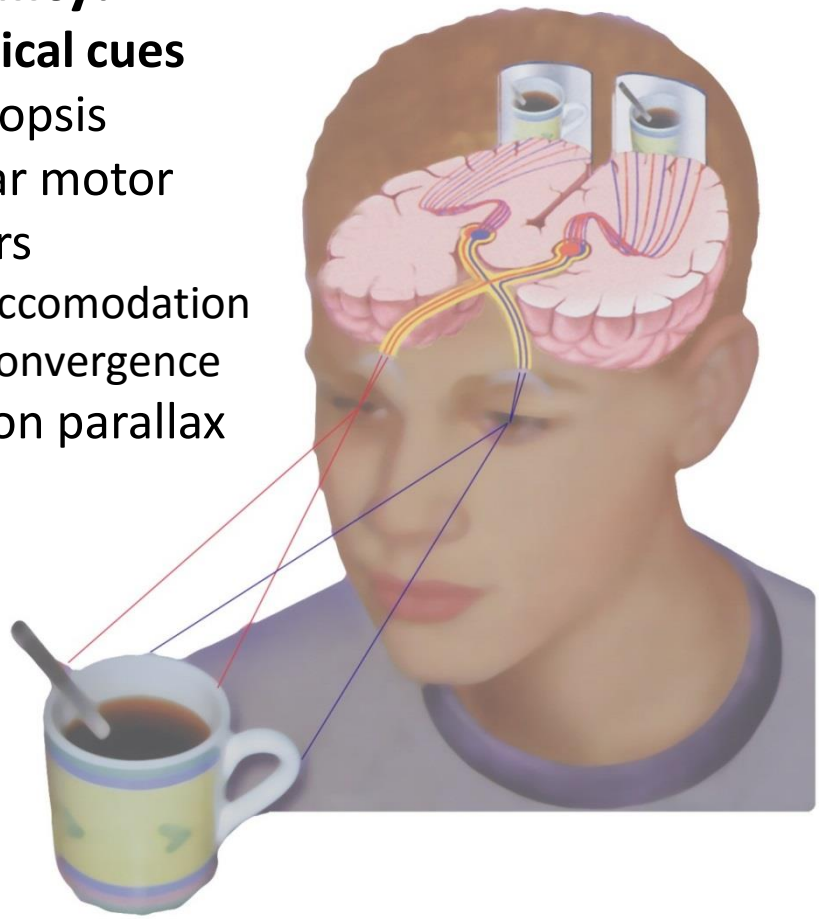
Physiological & Psychological Depth Cues

Traditional CG:

- **Psychological cues**
  - Perspective shortening
  - Occlusion of objects
  - Light and shadows
  - Texture gradients
  - Atmospheric perspective

Virtual Reality:

- **Physiological cues**
  - Stereopsis
  - Ocular motor factors
    - Accomodation
    - Convergence
  - Motion parallax

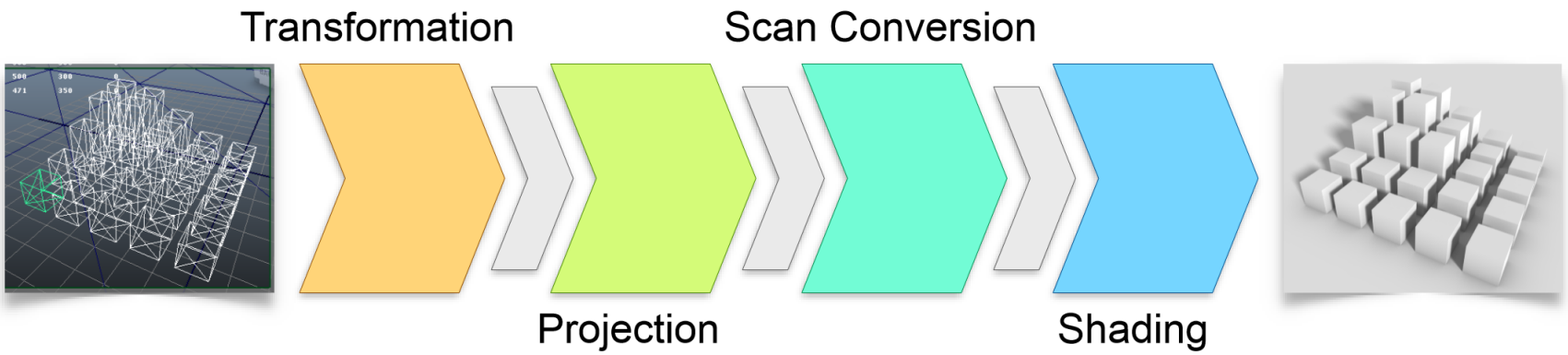


Addressing
Psychological Depth Cues and Real-Time Capabilities:

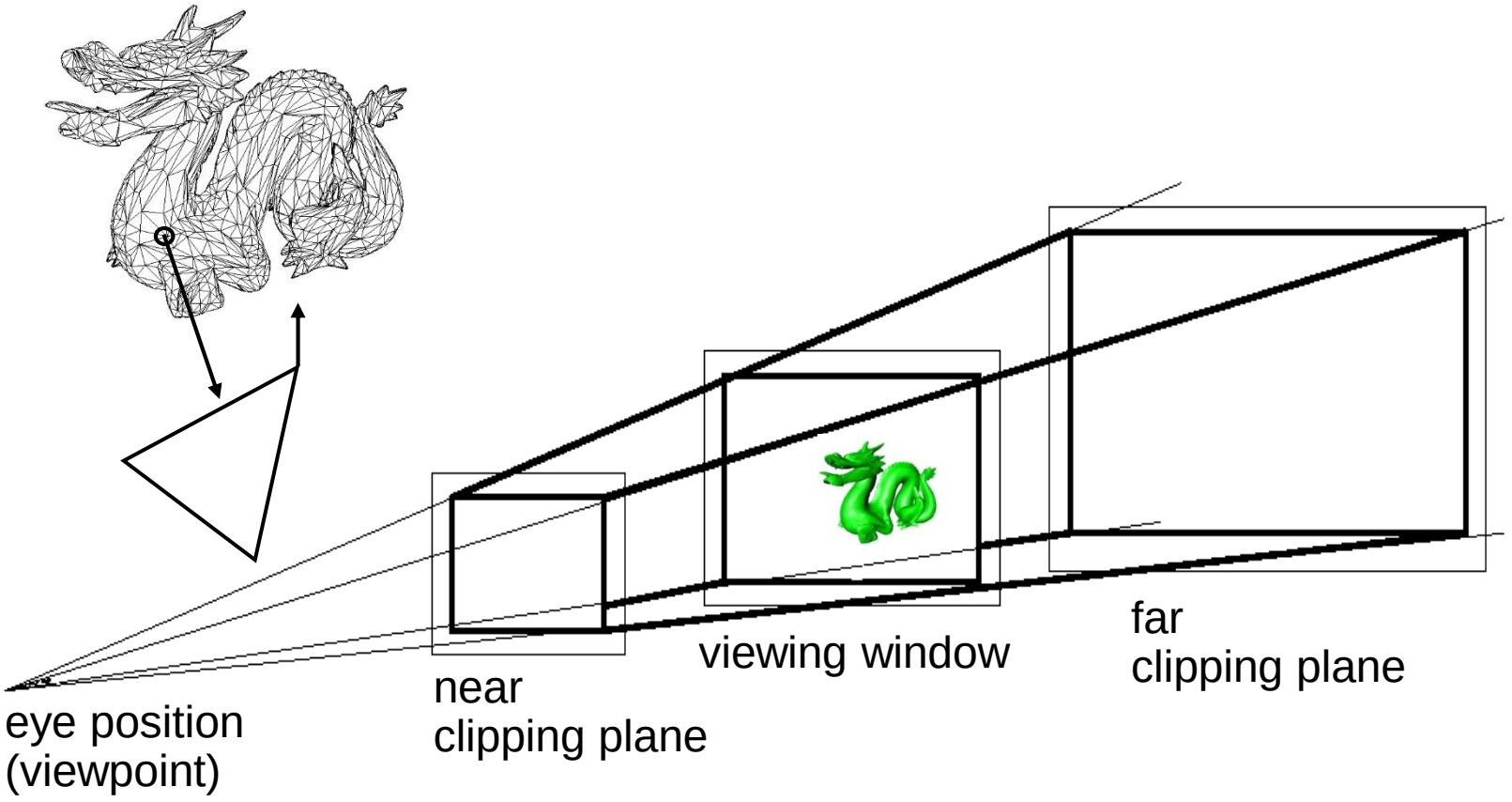
The Rendering Pipeline in Computer Graphics

Addressing the Psychological Depth Cues:

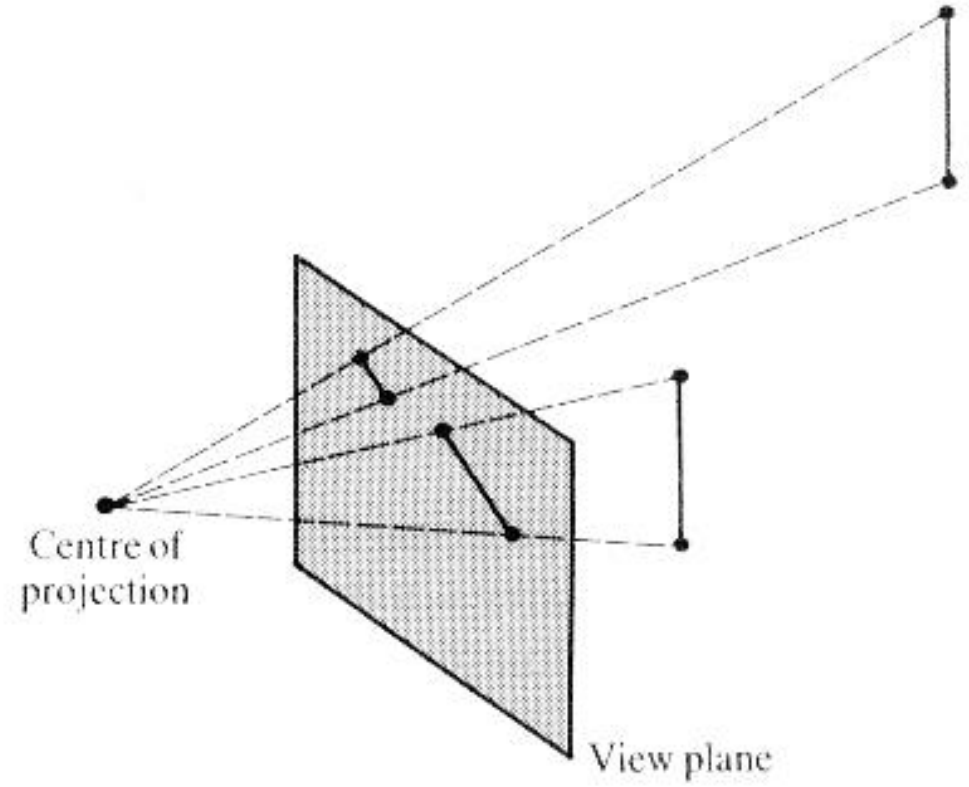
Projection → Perspective Shortening



View Volume


eye position
(viewpoint)
near
clipping plane
viewing window
far
clipping plane

Perspective Shortening as an Important Psyhological Depth Cue



Picture: Watt

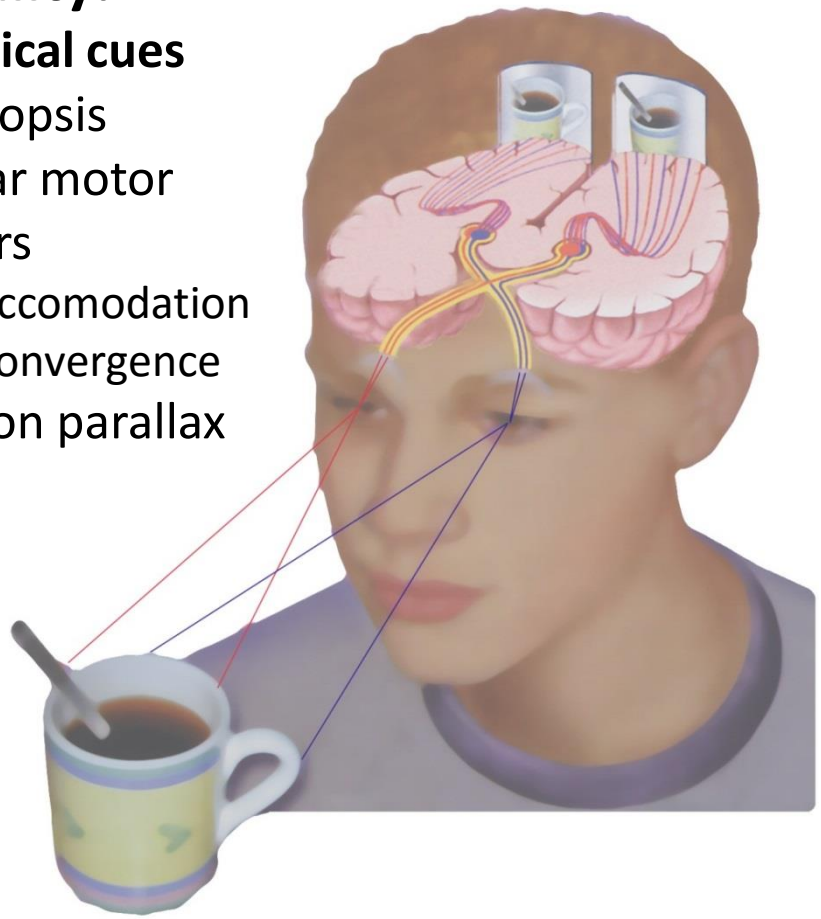
Physiological & Psychological Depth Cues

Traditional CG:

- **Psychological cues**
  - Perspective shortening
  - Occlusion of objects
  - Light and shadows
  - Texture gradients
  - Atmospheric perspective

Virtual Reality:

- **Physiological cues**
  - Stereopsis
  - Ocular motor factors
    - Accomodation
    - Convergence
  - Motion parallax



Addressing the Psychological Depth Cues:

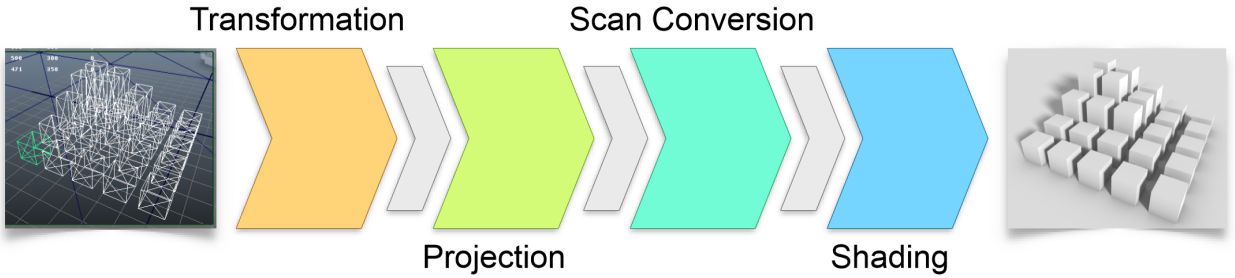
Reflection Model & Shading → Light & Shadows

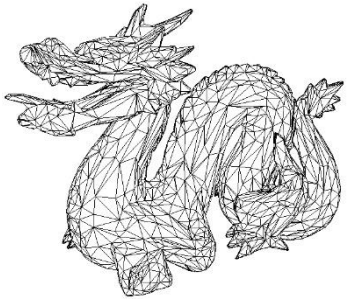
Addressing Real-Time Capabilities:

Local (Phong) Reflection Model

Shading → Gouraud & Phong

Rendering Pipeline Overview





Transformations:
position and orientation
of objects

Transformations:
- Local Coordinate Systems
- World Coordinate System
- View Coordinate System

Culling

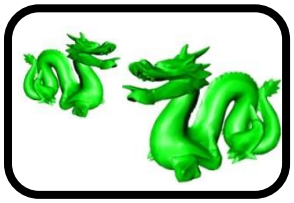
Projection

Clipping

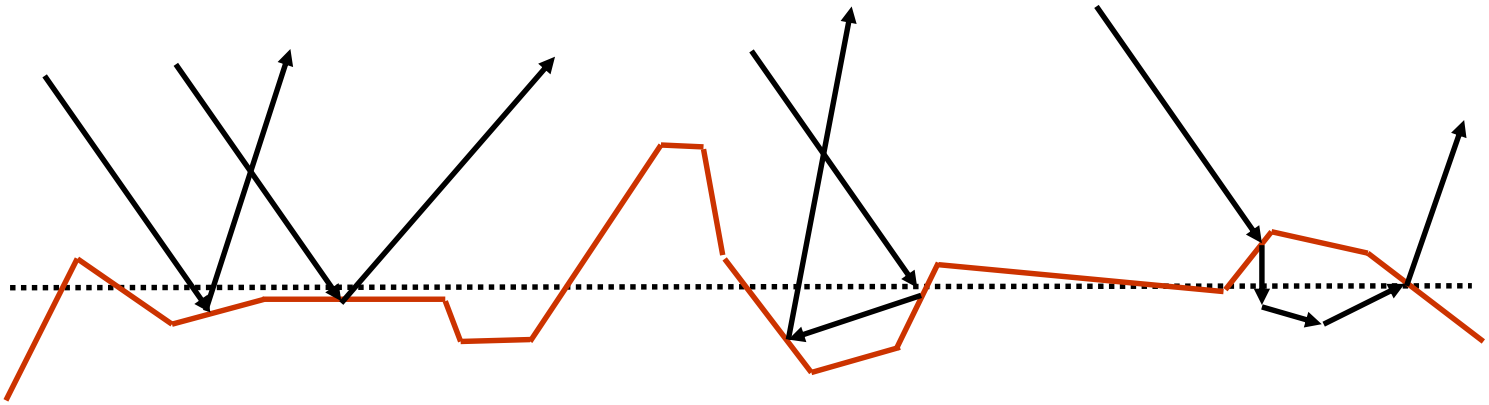
Shading

Hidden Surface
Removal

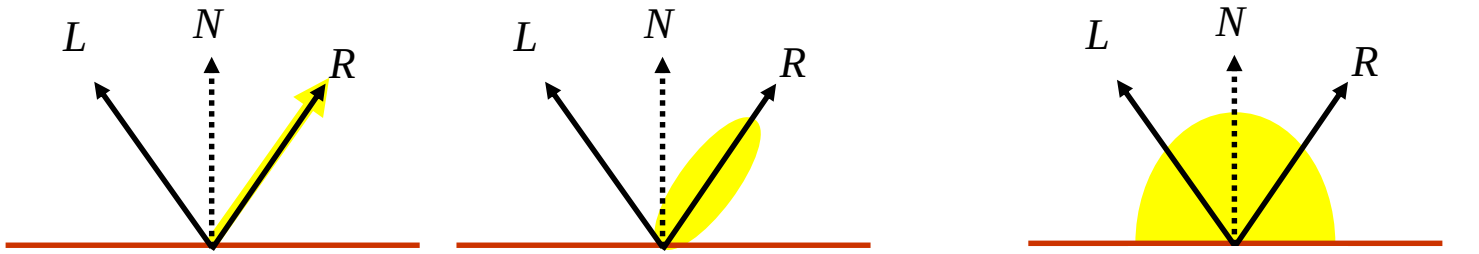
Scan
Conversion



Reflection on Surfaces



a) Direct reflection b) Multi-surface reflection c) Subsurface reflection

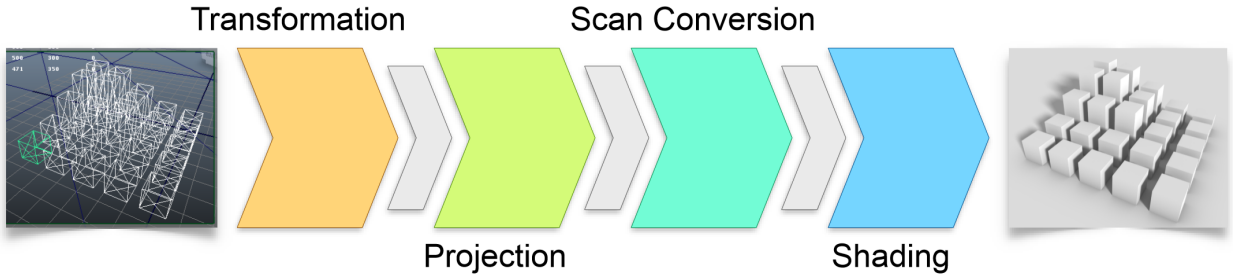


1) Mirror reflection
(perfect specular reflection)

2) Directed diffuse reflection

3) Diffuse Reflection

Light and Reflection



Local reflection (in real time CG and Virtual Reality):

- Interaction between object surfaces and light sources

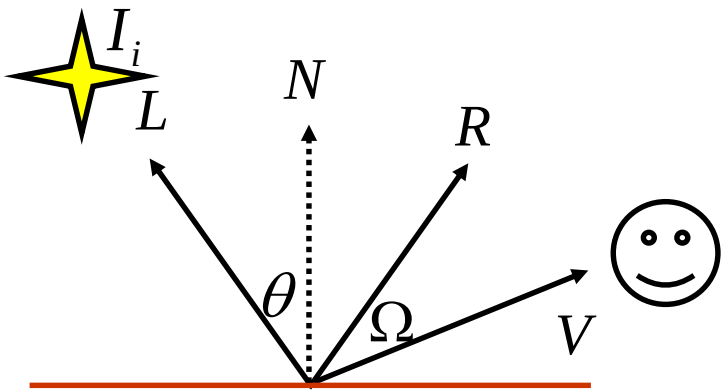
Global reflection (in photorealistic CG):

- Interaction between object surfaces and light sources
- Interaction between object surfaces and surfaces of other objects
- ➢ Ray Tracing (later in this lecture)
- ➢ Radiosity (see computer graphics lectures, e.g., Kobbelt – Informatik VIII)

The Phong Reflection Model

Linear combination from 3 components:

- Diffuse
- Specular
- Ambient



Reflection coefficients: k_d, k_s, k_a

Diffuse component: $I_d = I_i k_d \cos\theta = I_i k_d (L \cdot N)$ $\quad I_d = k_d \sum_n I_{i,n} (L_n \cdot N)$

Specular component: $I_s = I_i k_s \cos^n \Omega = I_i k_s (R \cdot V)^n$

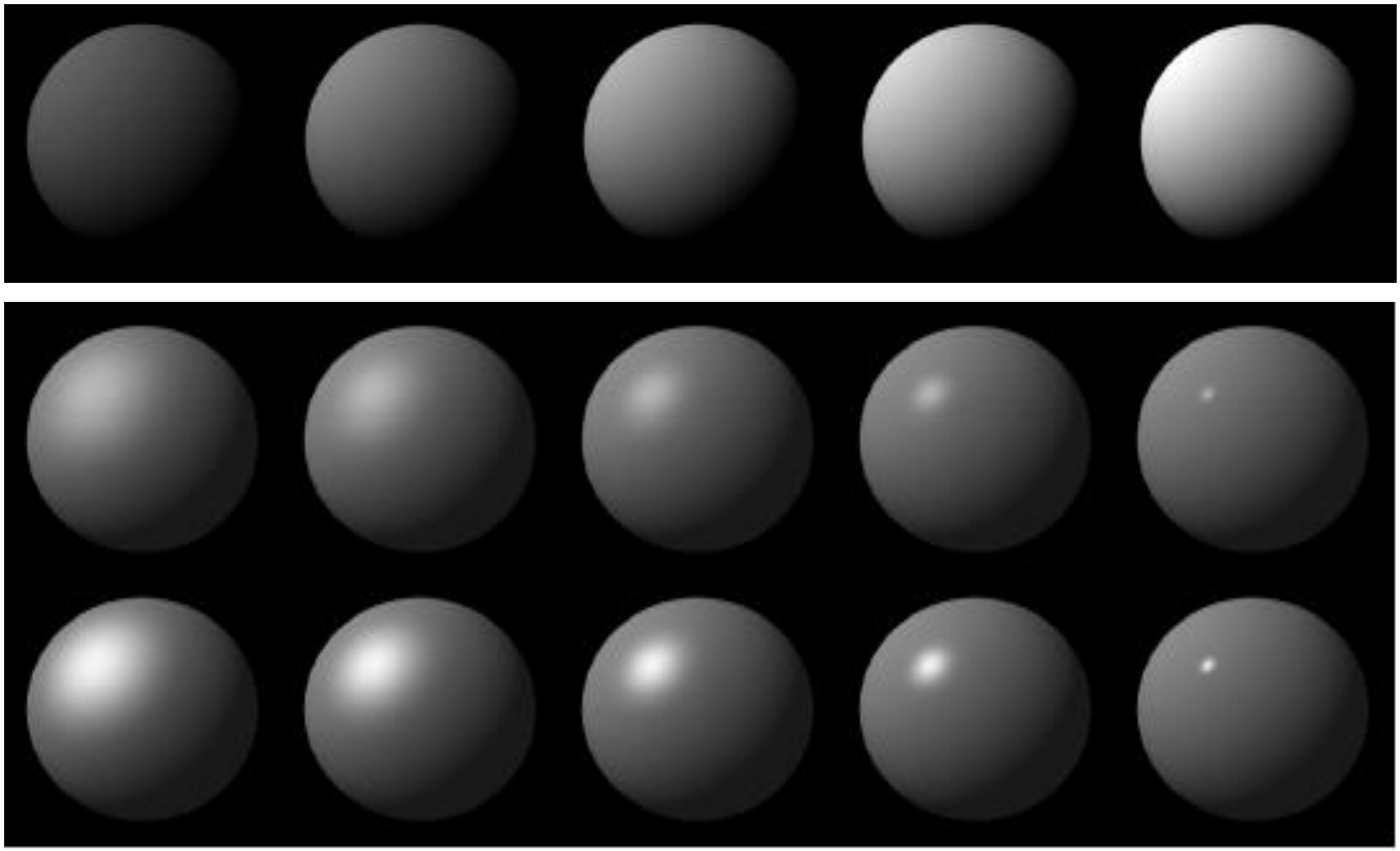
n : Index for surface roughness

Perfect mirror: $n \to \infty$ (Ray Tracing: Recursion)

Ambient component: $I_g = I_a k_a$

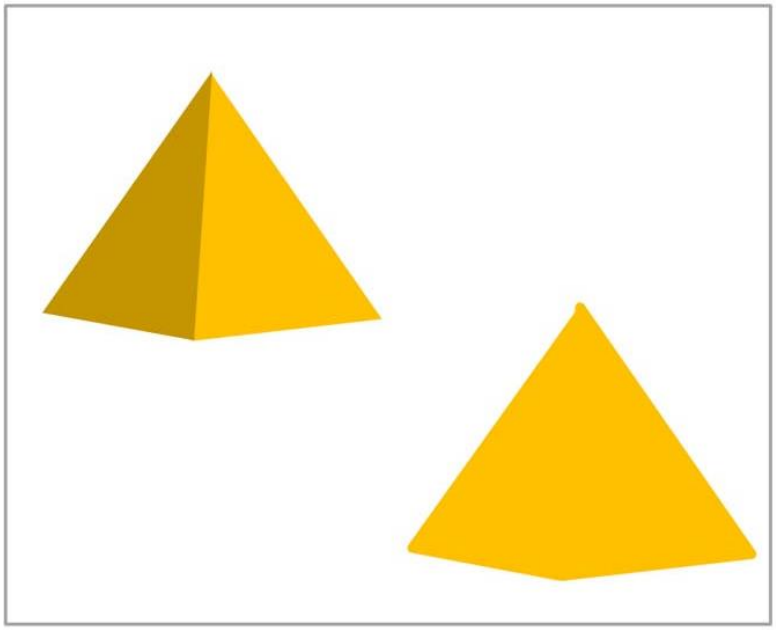
Overall intensity: $I = I_a k_a + I_i (k_d (L \cdot N) + k_s (R \cdot V)^n)$

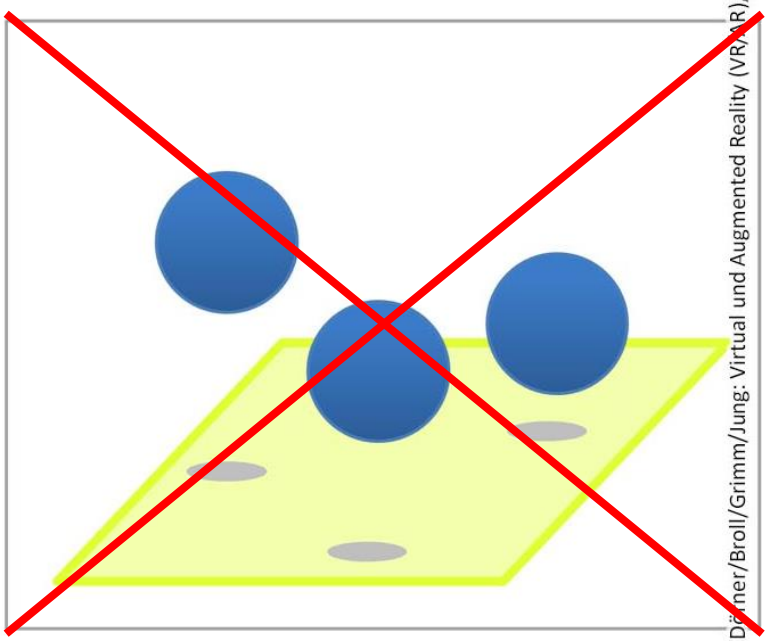
Diffuse & Specular



Pictures: Westermann

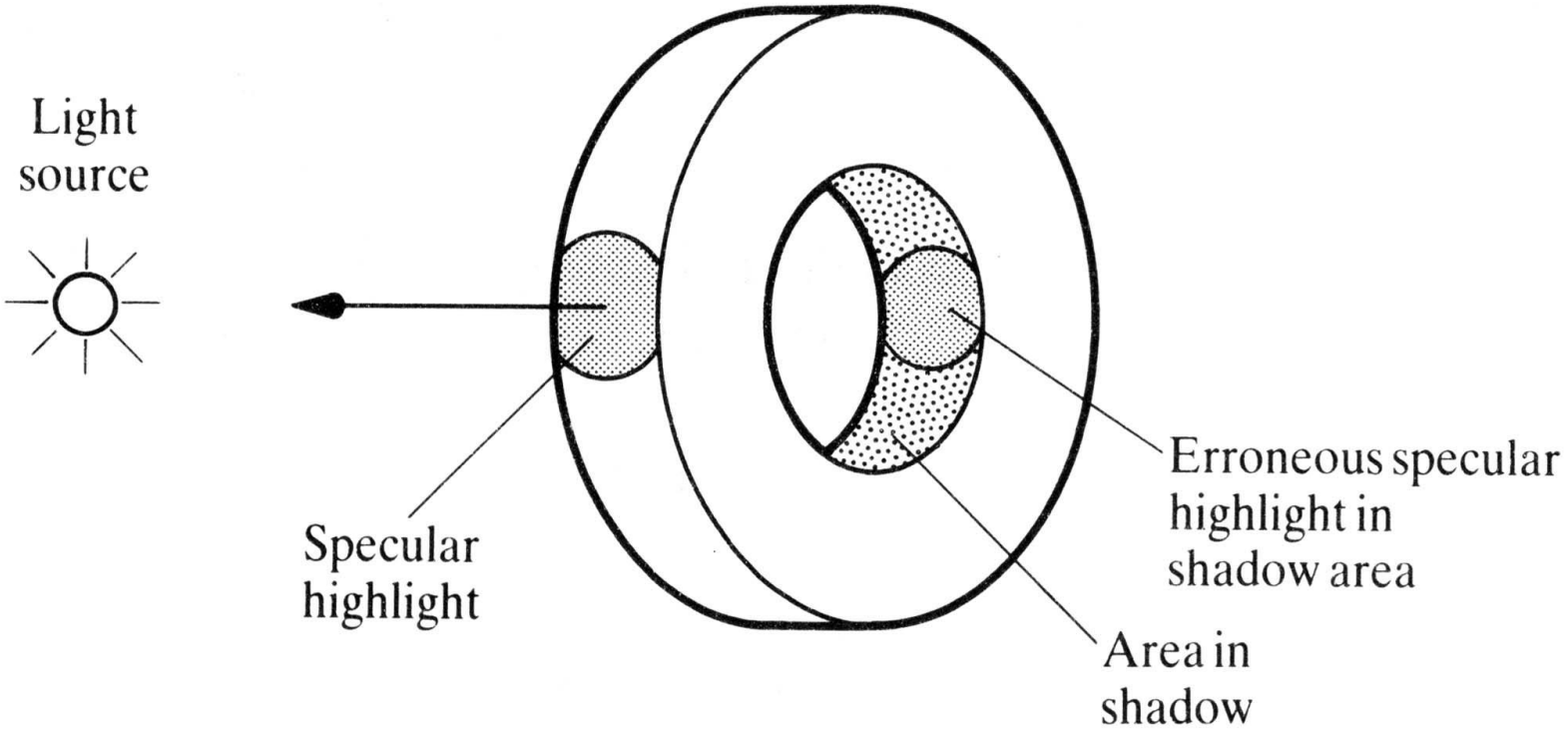
Light & Shadows





: Dörner/Broll/Grimm/Jung: Virtual und Augmented Reality (VR/AR),

Phong Reflection Model - Errors



Picture: Watt

What is Shading?

Shading

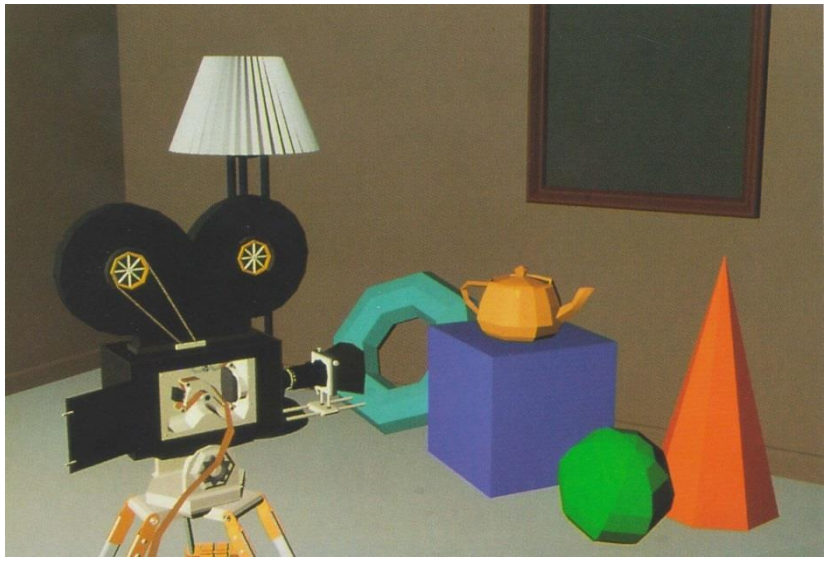
Definition: (Incremental, interpolative)
Application of a reflection model on polygons by calculation of intensities at polygon vertices and interpolation of these values for the inner points

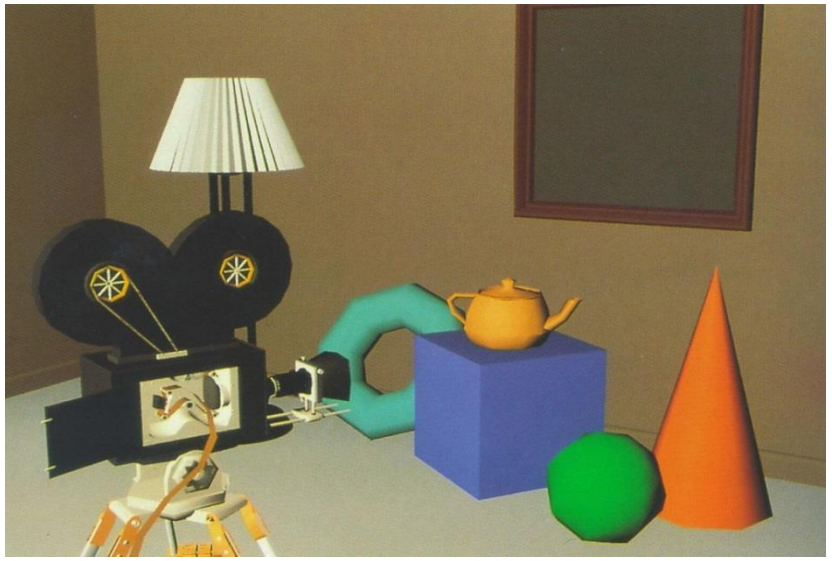
CG: Phong reflection model

1. Flat Shading
2. Gouraud Shading
3. Phong Shading

Flat Shading and Gouraud Shading

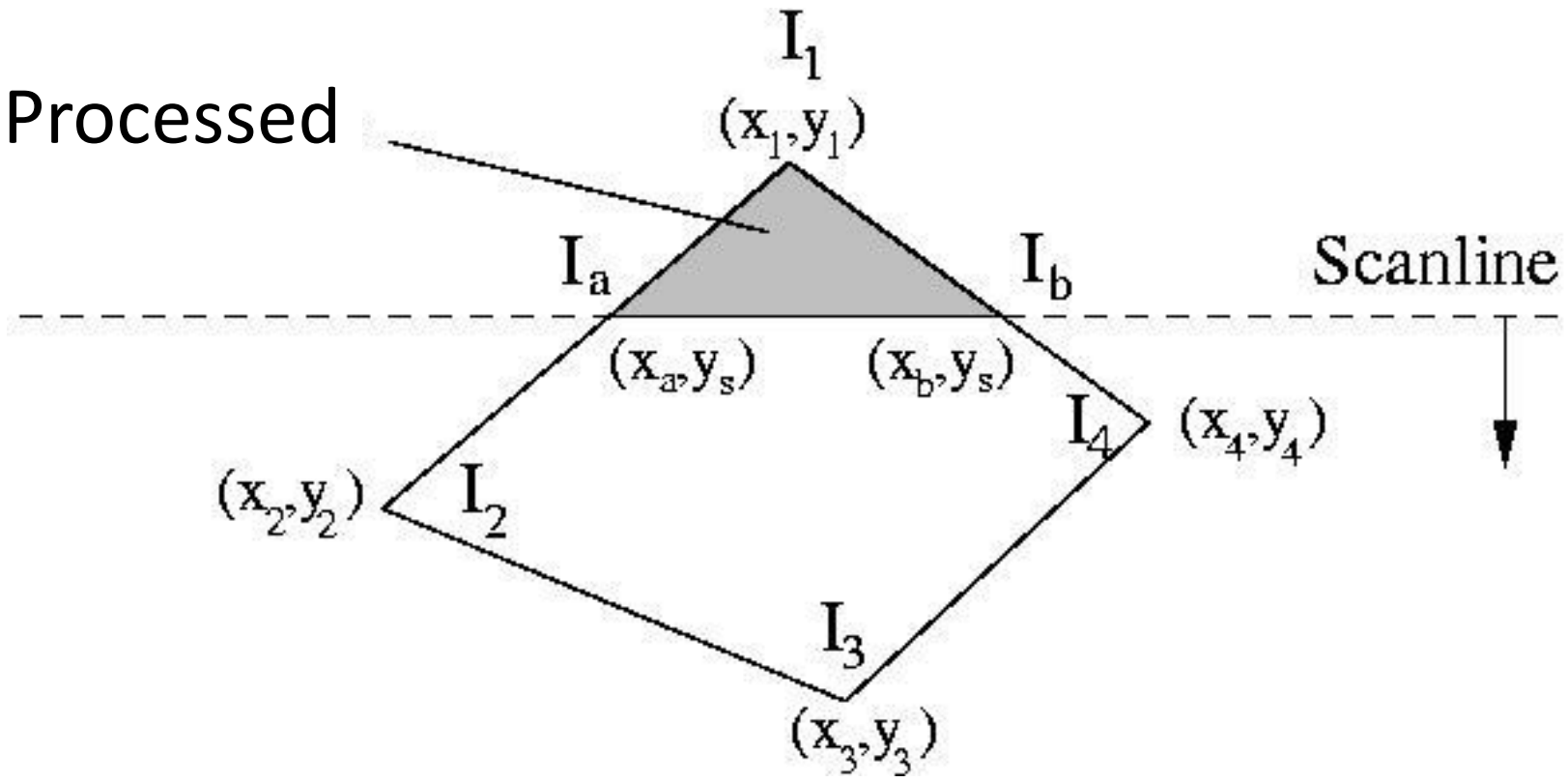
- **Flat Shading**
  No interpolation, all inner points of a polygon get the same intensity
- **Gouraud Shading**
  (Bilinear) interpolation of the inner points from the vertex intensities





Pictures: Foley et al.

Gouraud Shading – Bilinear Interpolation → Real-Time CG in VR



Gouraud Shading – Diffuse Component

- Calculate the vertex normal vector as average from the adjacent polygon normal vectors (offline!)
- Calculate the vertex intensities according to Phong model
- Interpolation process (integrated in scan conversion) in scan line order
  - Interpolate intensities at the intersection points of scan line and polygon edges from the vertex intensities

$$I_a = \frac{1}{y_1 - y_2}\left(I_1(y_s - y_2) + I_2(y_1 - y_s)\right)$$

$$I_b = \frac{1}{y_1 - y_4}\left(I_1(y_s - y_4) + I_4(y_1 - y_s)\right)$$

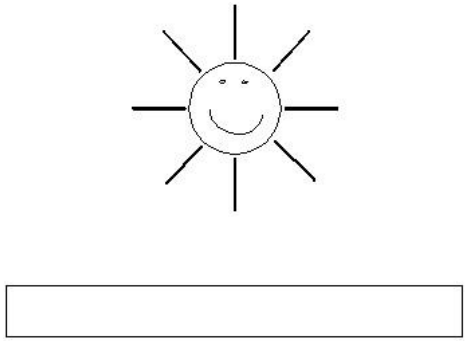
  - Interpolate intensities for the inner points along the scan line from the intensities at the intersection points

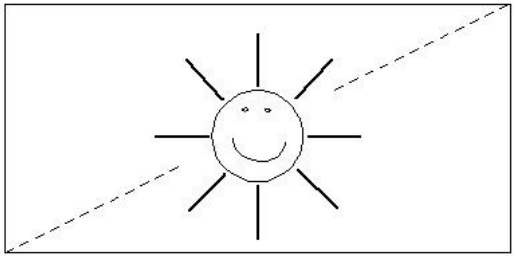
$$I_s = \frac{1}{x_b - x_a}\left(I_a(x_b - x_s) + I_b(x_s - x_a)\right)$$

Has to be calculated for every pixel: incremental approach, see literature

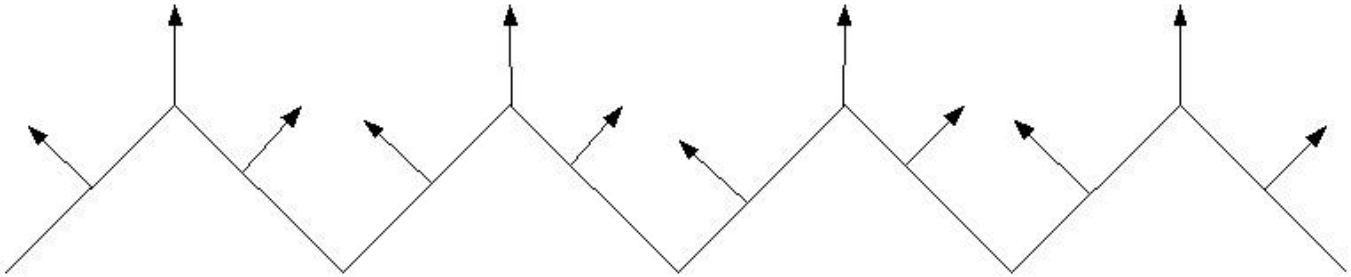
Gouraud Shading – Some Problems

- No highlights in the middle of polygons (example: virtual table)



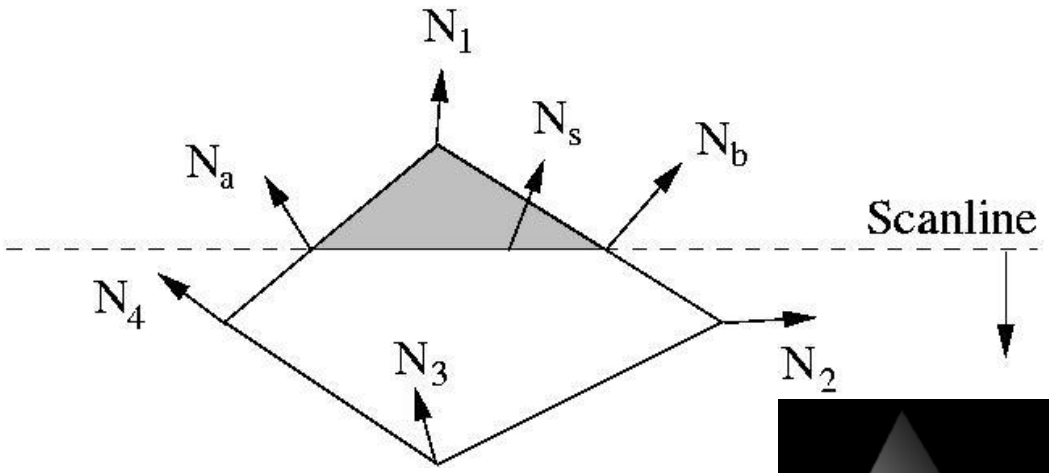


- Corrugations are smoothed out

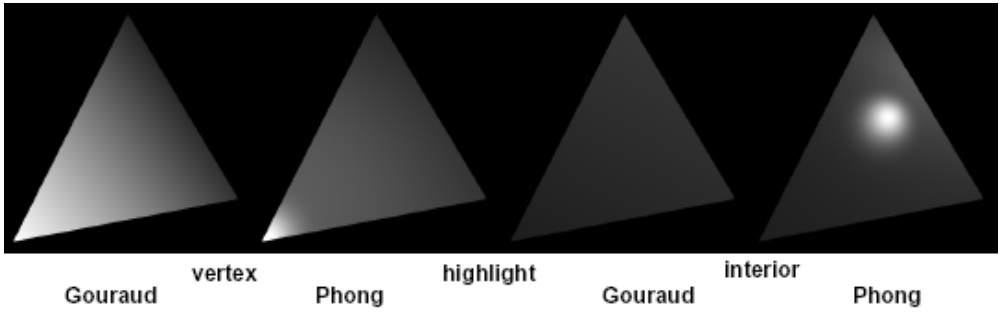


Phong Shading

- Interpolation of vertex normal vectors instead of intensities
- Get an individual normal vector for each pixel as an approximation of the “real” normal vector on a curved surface

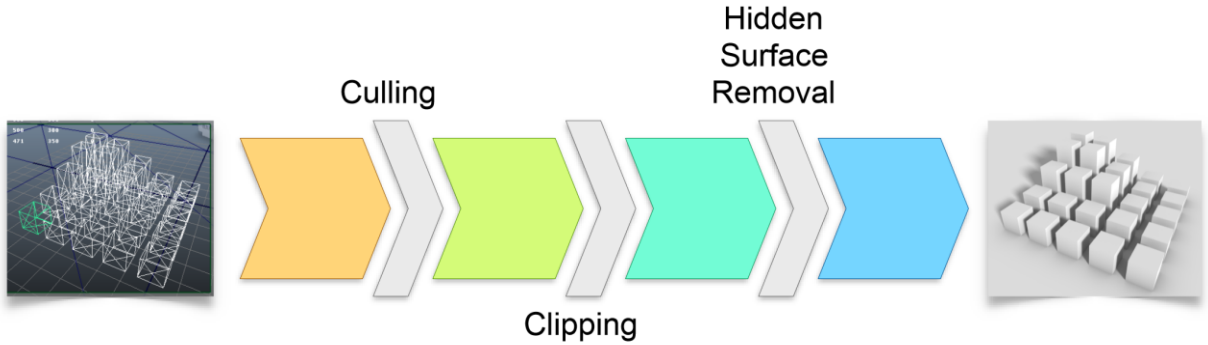


Picture: Westermann



Addressing the Psychological Depth Cues:

Hidden Surface Removal → Occlusion



Addressing Real-Time Capabilities:

Backface Culling

Hidden Surface Removal → Z-Buffering

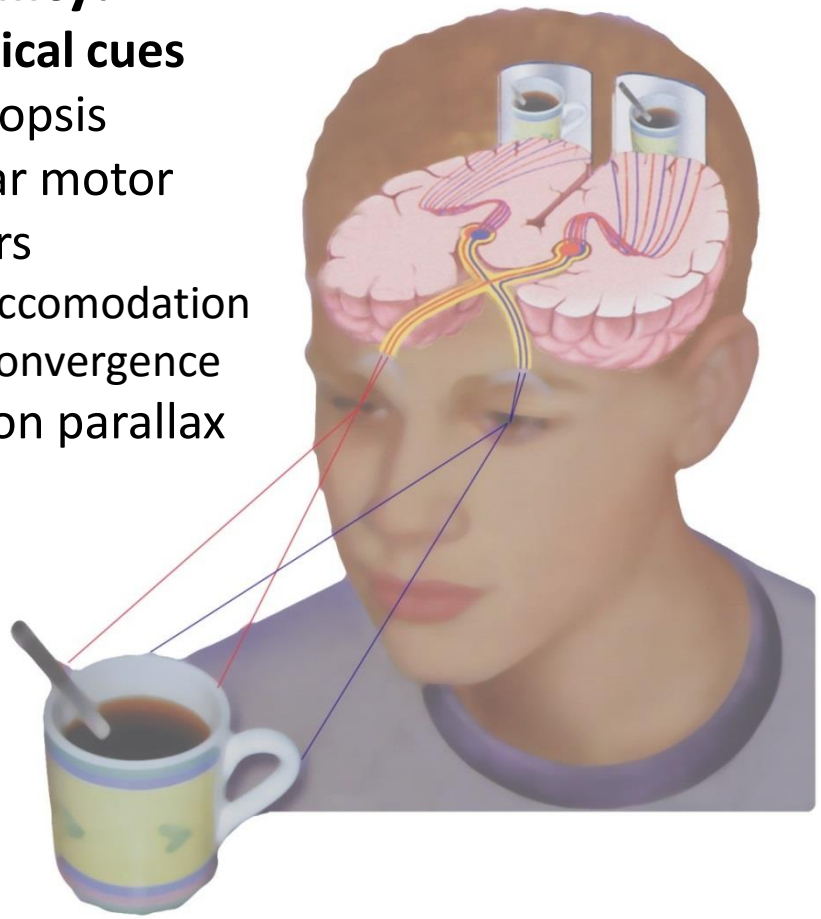
Physiological & Psychological Depth Cues

Traditional CG:

- **Psychological cues**
  - Perspective shortening
  - Occlusion of objects
  - Light and shadows
  - Texture gradients
  - Atmospheric perspective

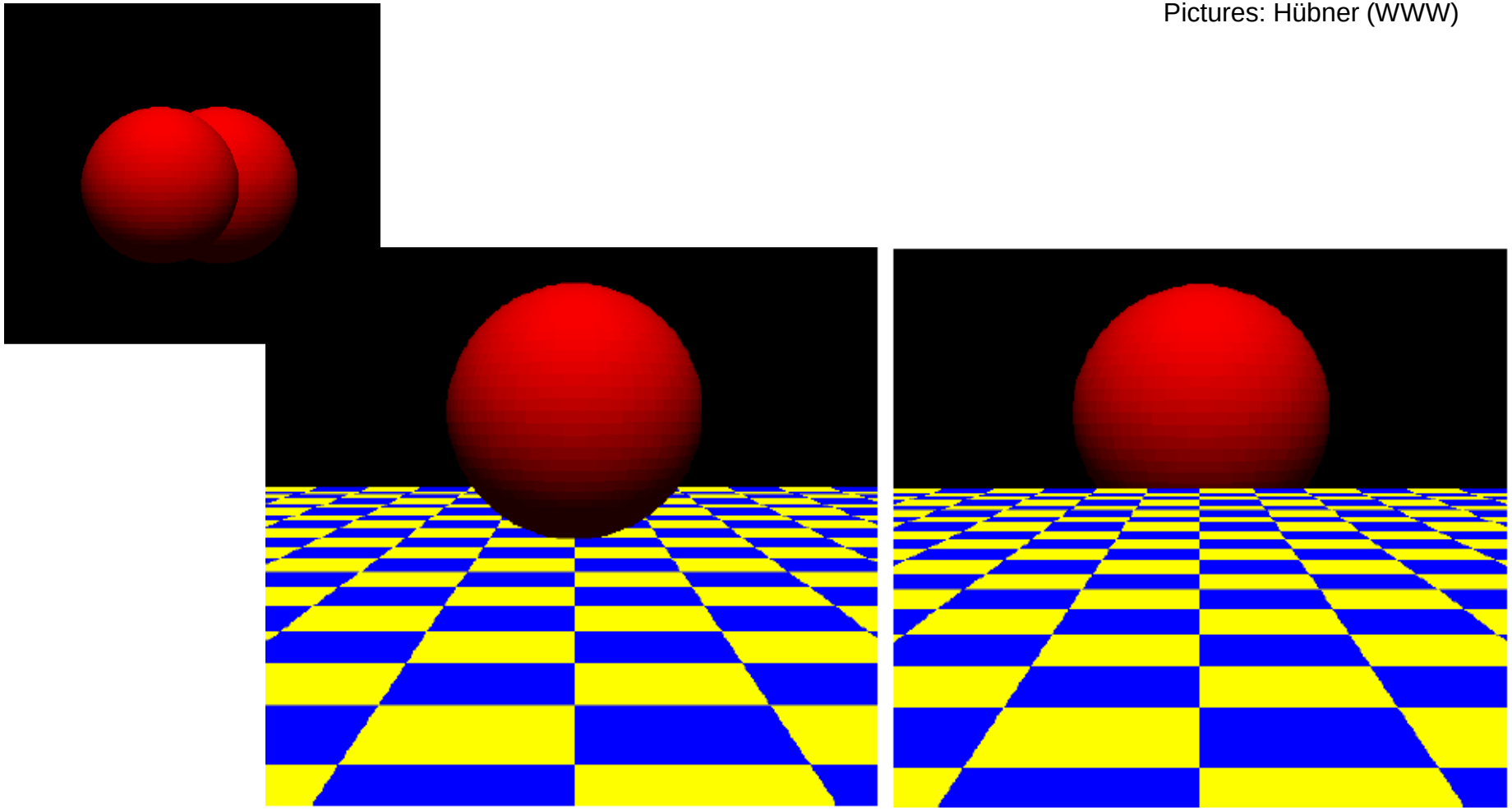
Virtual Reality:

- **Physiological cues**
  - Stereopsis
  - Ocular motor factors
    - Accomodation
    - Convergence
  - Motion parallax

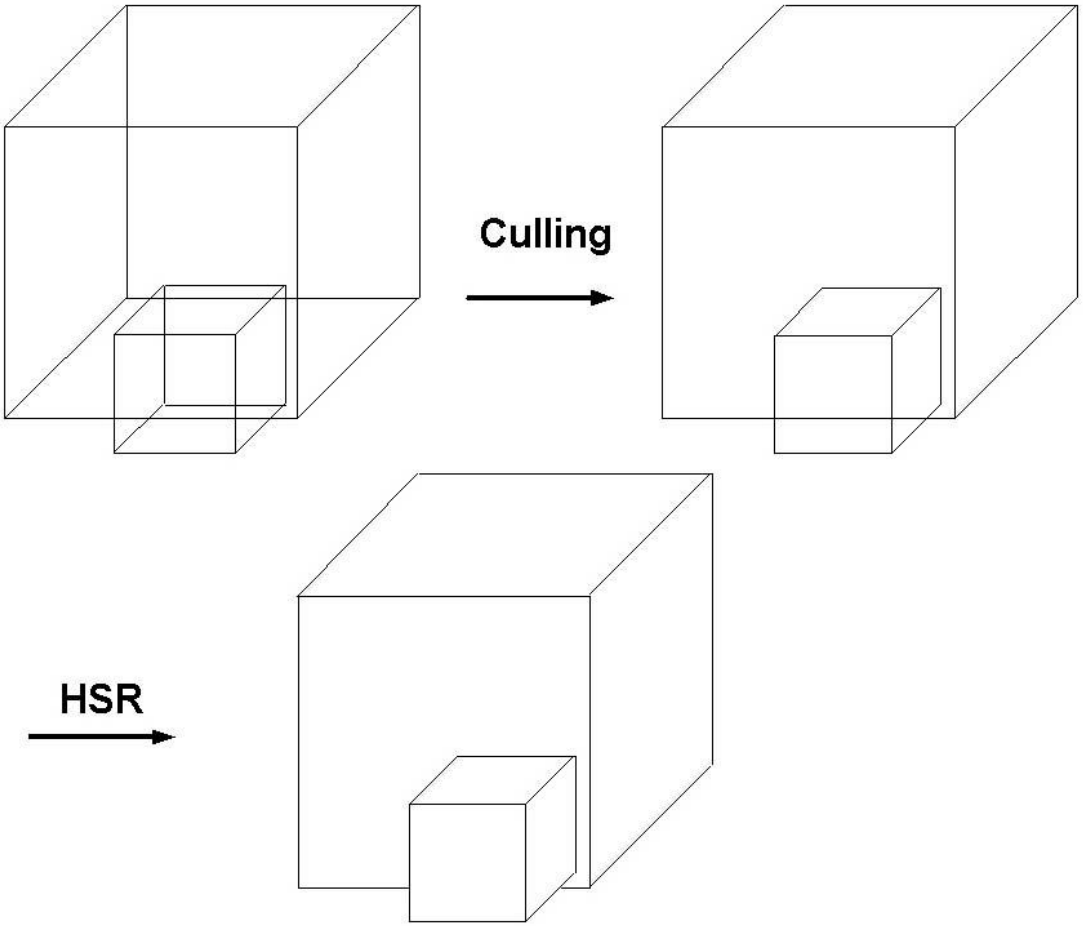


Occlusion

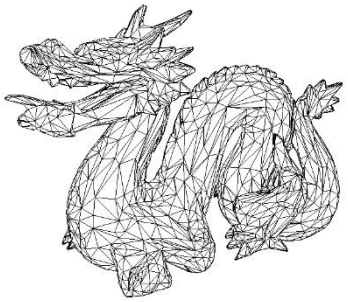
Pictures: Hübner (WWW)



Culling versus Hidden Surface Removal



Rendering Pipeline Overview



Transformations:
position and orientation
of objects

Transformations:
- Local Coordinate Systems
- World Coordinate System
- View Coordinate System

Culling

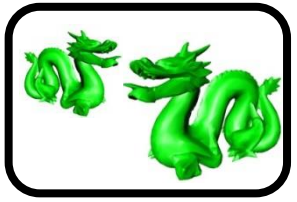
Projection

Clipping

Shading

Hidden Surface
Removal

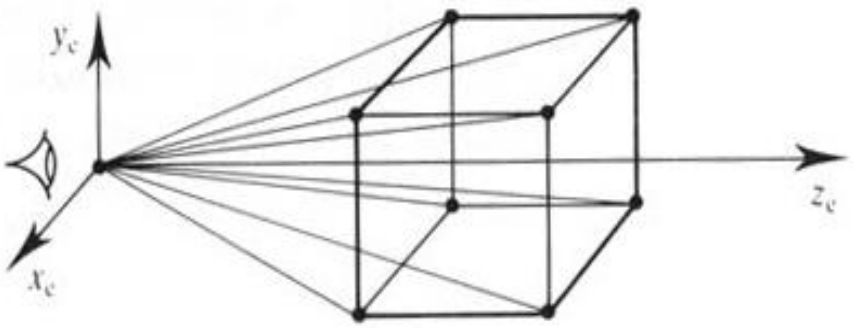
Scan
Conversion

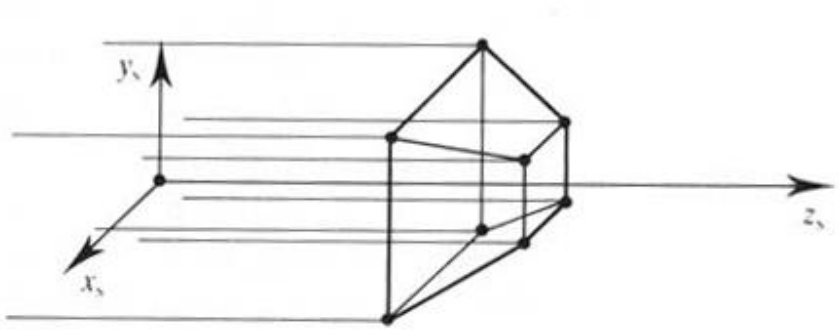


Z-Buffering: Occlusion as a Psychological Depth Cue

Z-Buffer

- Special memory on graphics hardware, 1 entry for every pixel
- Updated with the highest z value of 3D object points that cover its pixel
- Accuracy depends on
  - Length of memory words („depth“) (usually 24 bits/pixel)
  - Z-value of front and back clipping plane
- Compare points lying on the same projector (OpenGL: parallel projection!)



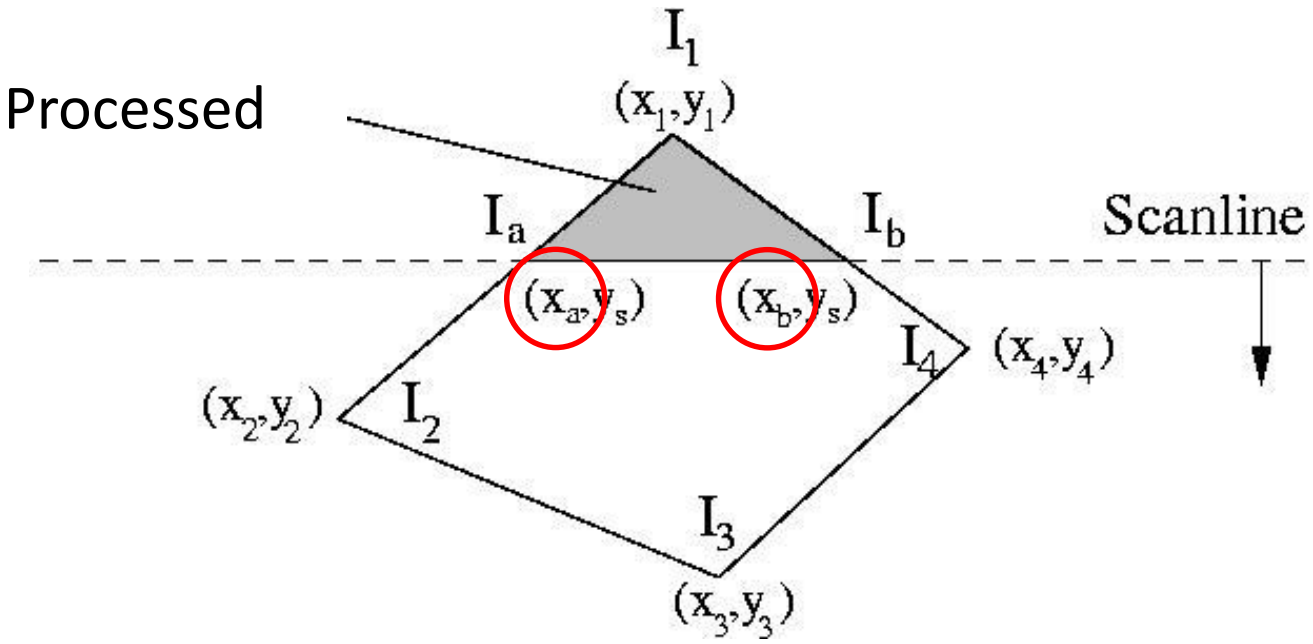


Addressing Real-Time in Virtual Reality: The Pixel Loop

Simple Reflection Model, Shading and Scan Conversion

applied to Polygons

Scan Conversion



- **Scan Conversion** or **Rasterization** determines the actual pixels of the view window for which intensity values are required.
- Rasterization of solid objects:
  Polygon fill via horizontal line segments (“scanline order”)
- Required: Right/Left border of a segment, i.e. the intersection point of scanline and polygon edges → Digital Differential Analyzer

Rasterization of Lines – Brute Force Approach

Rendering of lines which are non-parallel to the screen axes:

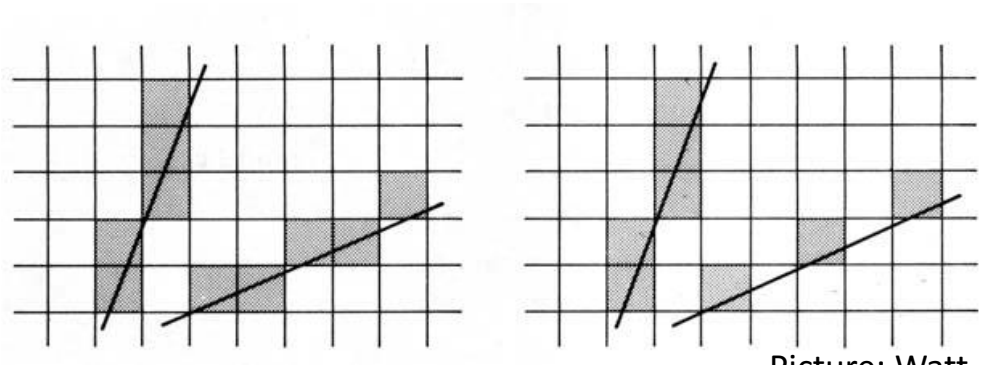
- Which pixels should be set?
- “Wireframe”: Avoid gaps!

Simple strategy: Use DDA algorithm

- Follow the line horizontally or vertically, according to slope
- Set the pixel closest to the line

/* Let (x_s, y_s), (x_e, y_e) be start and end vertex of a polygon edge, $y_e > y_s$ */

$x := x_s$
$m := (x_e - x_s)/(y_e - y_s)$
For $y := y_s$ to y_e do
    output $(round(x), y)$
    $x := x + m$



Picture: Watt

The Bresenham Algorithm from 1962 - Idea

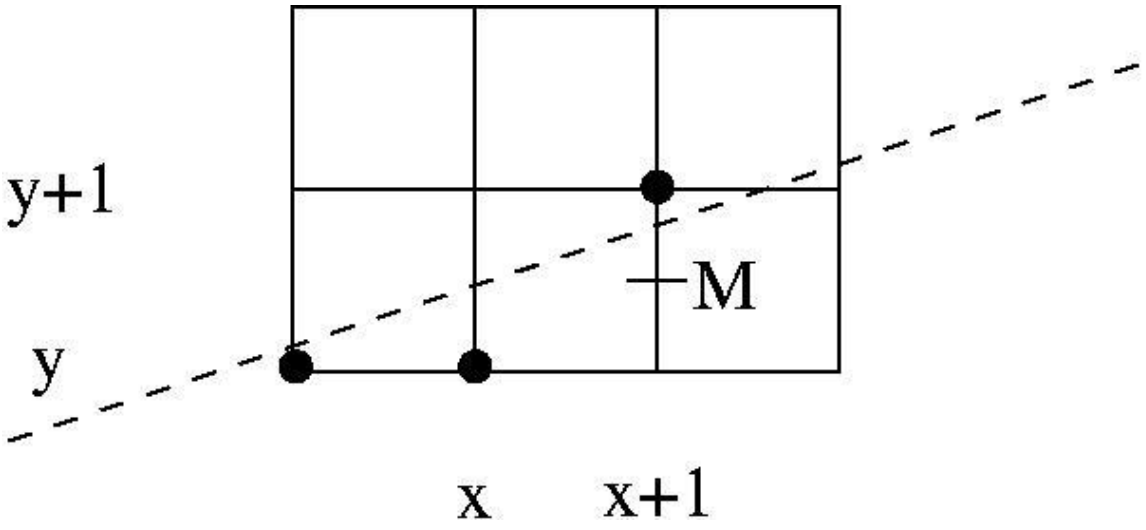
- Goal:
  Replace Float by Integer operations (faster)
- Idea:
  After pixel (x, y) has been set, there are 2 possibilities for the next step:
  - E (East) case: Set pixel (x+1, y)
  - NE (North East) case : Set pixel (x+1, y+1)
- Let the sign of a decision variable d decide about E or NE
- Incrementally modify d from step to step

The Bresenham Algorithm – Decision Variable

Line equation: $$y = \frac{\Delta y}{\Delta x}(x - x_1) + y_1$$

Implicit: $$F(x, y) = 0 \text{ mit } F(x, y) = \frac{\Delta y}{\Delta x}(x - x_1) - (y - y_1)$$

Decision variable d: $$F\left(x + 1, y + \frac{1}{2}\right) =: F(M)$$



The Bresenham Algorithm – Initialization

Initialize *d*:

$$d = F\left(x_1 + 1, y_1 + \frac{1}{2}\right) = \frac{\Delta y}{\Delta x} - \frac{1}{2} \quad |\cdot 2\Delta x \text{ (does not change the sign of } d\text{)}$$

$$\Rightarrow d = 2\Delta y - \Delta x \quad \text{INTEGER!}$$

The Bresenham Algorithm – East and North East

Two cases for M:

- M is below the line $\Rightarrow d > 0\ (NE)$

$$x := x+1, y := y+1$$

$$d_{NEW} := F\left(x+2, y+\frac{3}{2}\right) = \frac{\Delta y}{\Delta x}(x+2-x_1) - \left(y+\frac{3}{2}-y_1\right) =$$

$$\frac{\Delta y}{\Delta x}(x+1-x_1) + \frac{\Delta y}{\Delta x} - \left(y+\frac{1}{2}-y_1\right) - 1 = d_{OLD} + \frac{\Delta y}{\Delta x} - 1$$

- M is above the line $\Rightarrow d \leq 0\ (E)$

$$x := x+1, y := y$$

$$d_{NEW} := F\left(x+2, y+\frac{1}{2}\right) = d_{OLD} + \frac{\Delta y}{\Delta x}$$

The Bresenham Algorithm – Pseude Code

/* Let (x_s, y_s) ,(x_e, y_e) be start and end vertex of a polygon edge, $y_e > y_s$/*

$x := x_1$,y $:= y_1, \Delta x := x_2 - x_1, \Delta y := y_2 - y_1$

$d := 2\Delta y - \Delta x$

$incrE := 2\Delta y$ /* Δy / Δx multiplied with 2 Δx */

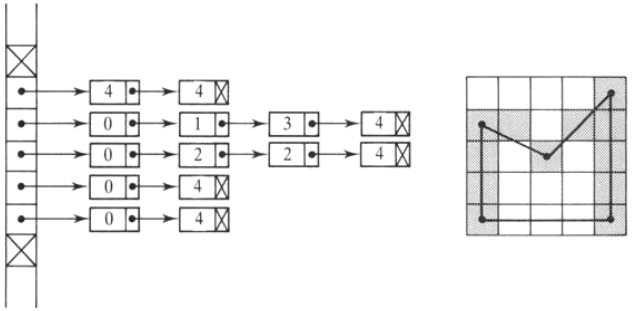
$incrNE := 2(\Delta y - \Delta x)$ /* Δy / Δx -1 multiplied with 2 Δx */

The Bresenham Algorithm – Pseude Code (cont.)

While $(x < x_2)$
    $x := x + 1$
    if $d > 0$ then
        $y := y + 1$
        $d := d + incrNE$
    else
        $d := d + incrE$
    endif
    output(x, y)
endwhile

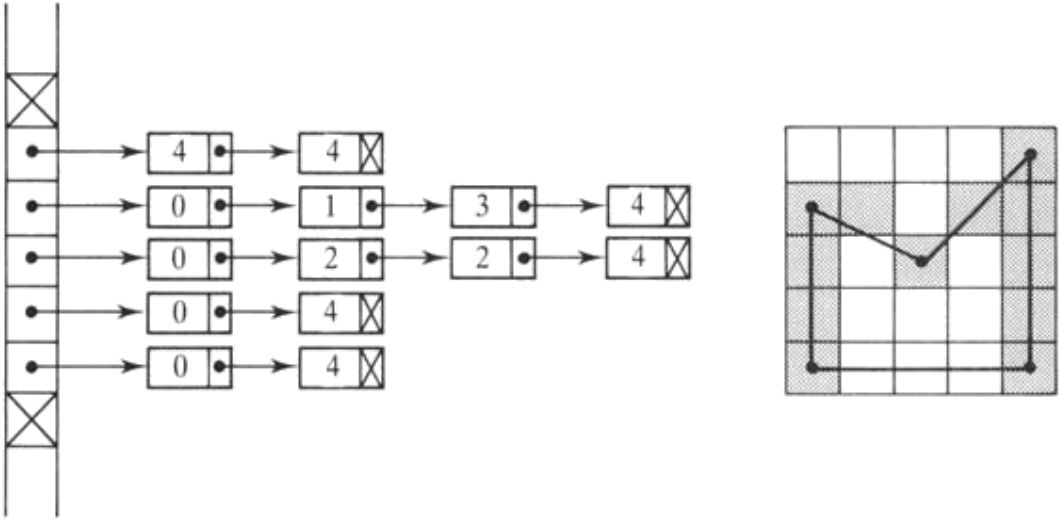
Rendering (Pixel) Loop

For every polygon
  For each edge (x_s, y_s), (x_e, y_e) of the polygon /* $y_e > y_s$ */
    $x := x_s$ /*DDA */
    $m := \frac{x_e - x_s}{y_e - y_s}$
    For $y := y_s$ to y_e do
      insert $round(x)$ into linear list $A[y]$ /* insertion sort */
      $x := x + m$
    endfor
  endfor
  For $y := y_{min}$ to y_{max} do
    For every pair (x_i, x_{i+1}) in $A[y]$ do
      if z-value(x, y) > z-buffer(x, y) then
        update (z-buffer)
        $I_d = k_d \sum_n I_{i,n}(L_n \cdot N)$ /* diffuse illumination, explicit */
        set_framebuffer (x, y, I_d)
      endif
    endfor
  endfor
endfor



Polygon Edge List

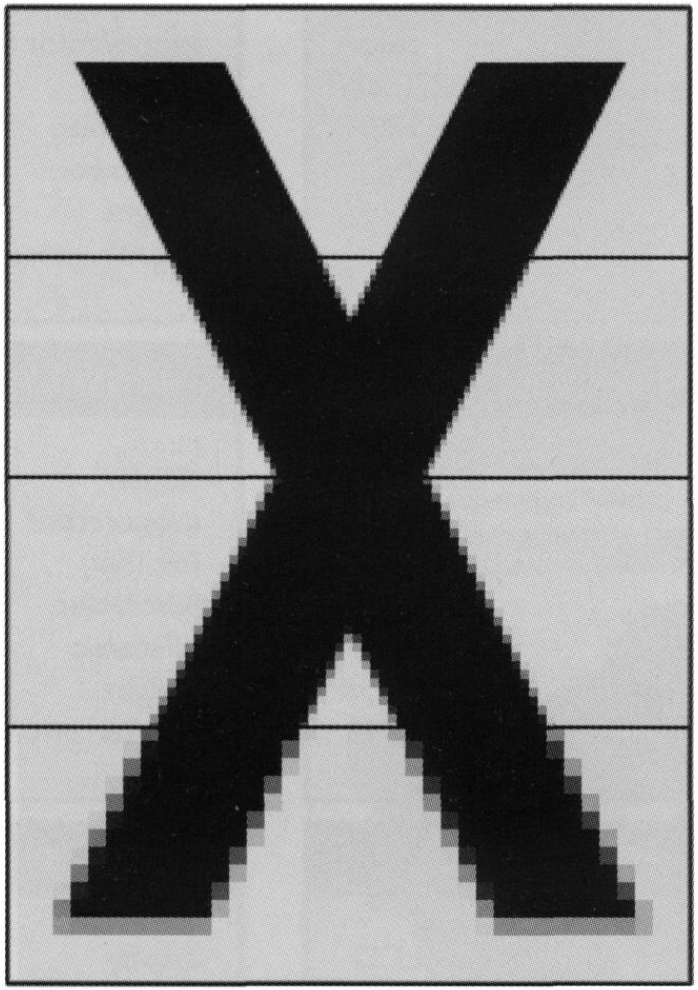
- Data structure:
  Edge list for each polygon by an array of linked lists, one element for each scan line



- Closed polygons: Even number of list elements
- Also works for concave polygons with holes
- Insertion sort: Sort while inserting elements into the list
- **Convex polygons (e.g., triangles): 2 entries per array element only**

Picture: Watt

Display Resolution



Today:

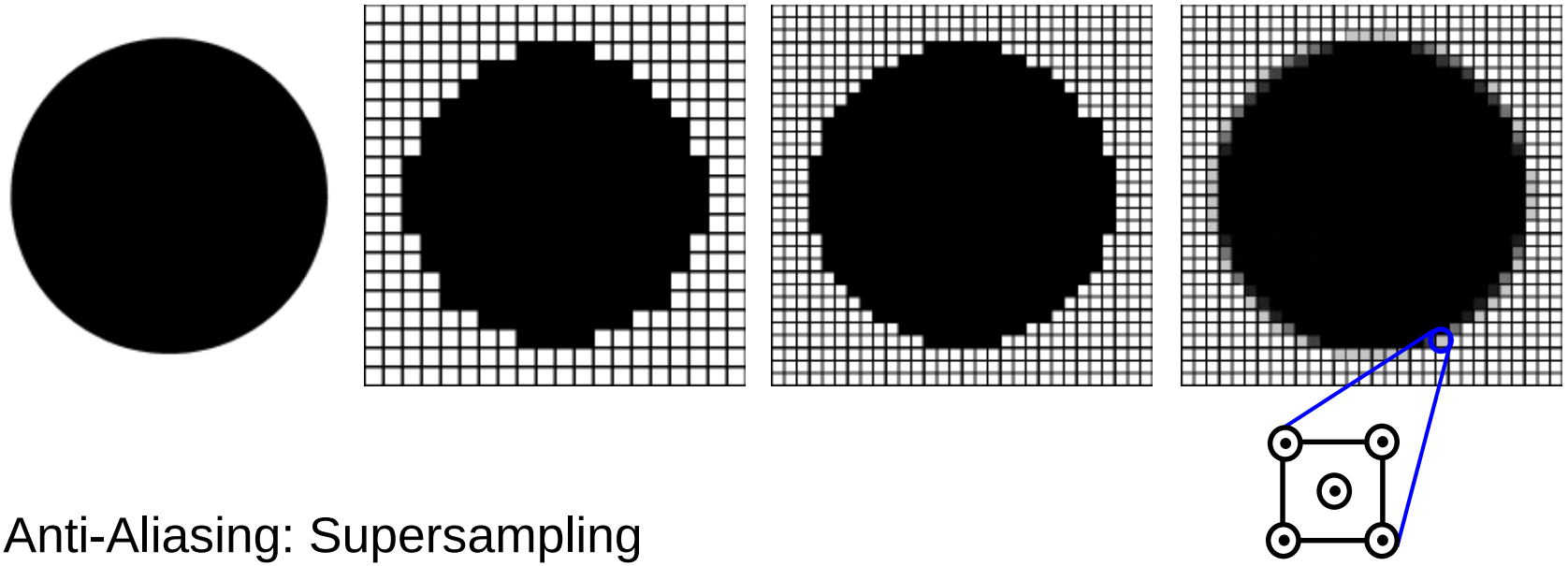
- SXGA, 1280x1024
- UXGA, 1600x1200
- WUXGA, 1920x1200
- 4K, 4096x2160

See also:

- Color depth
- Anti-Aliasing

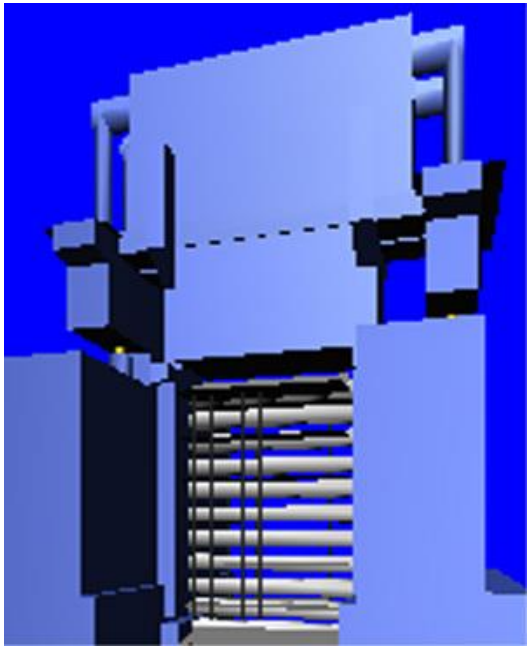
Picture: Burdea et al.

Optimization - Quality

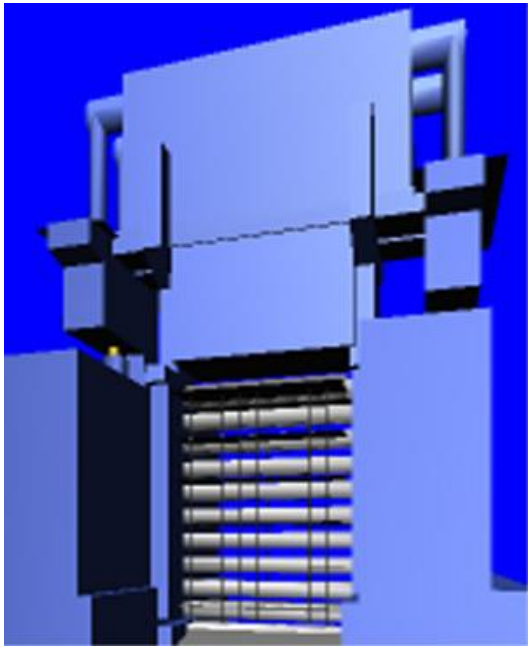


- Anti-Aliasing: Supersampling
  - Compute image @ “virtually” higher resolution
  - Treat pixels as a plane
  - Every pixel has multiple subpixels
  - Averaging the intensities of single subpixels leads to pixel color

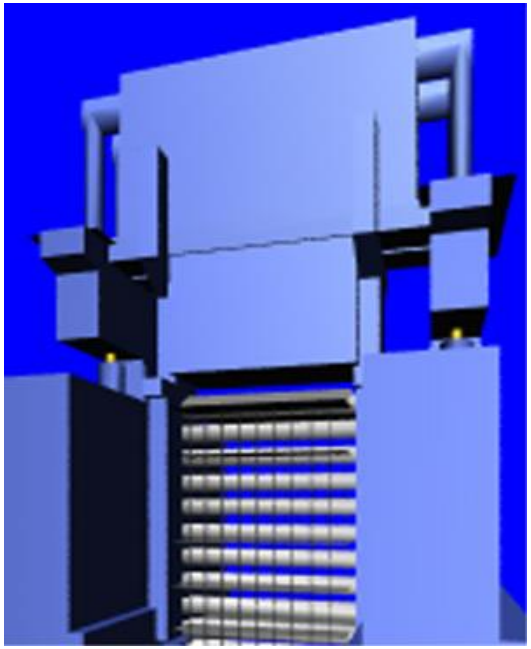
Anti-Aliasing



1 sample



4 samples

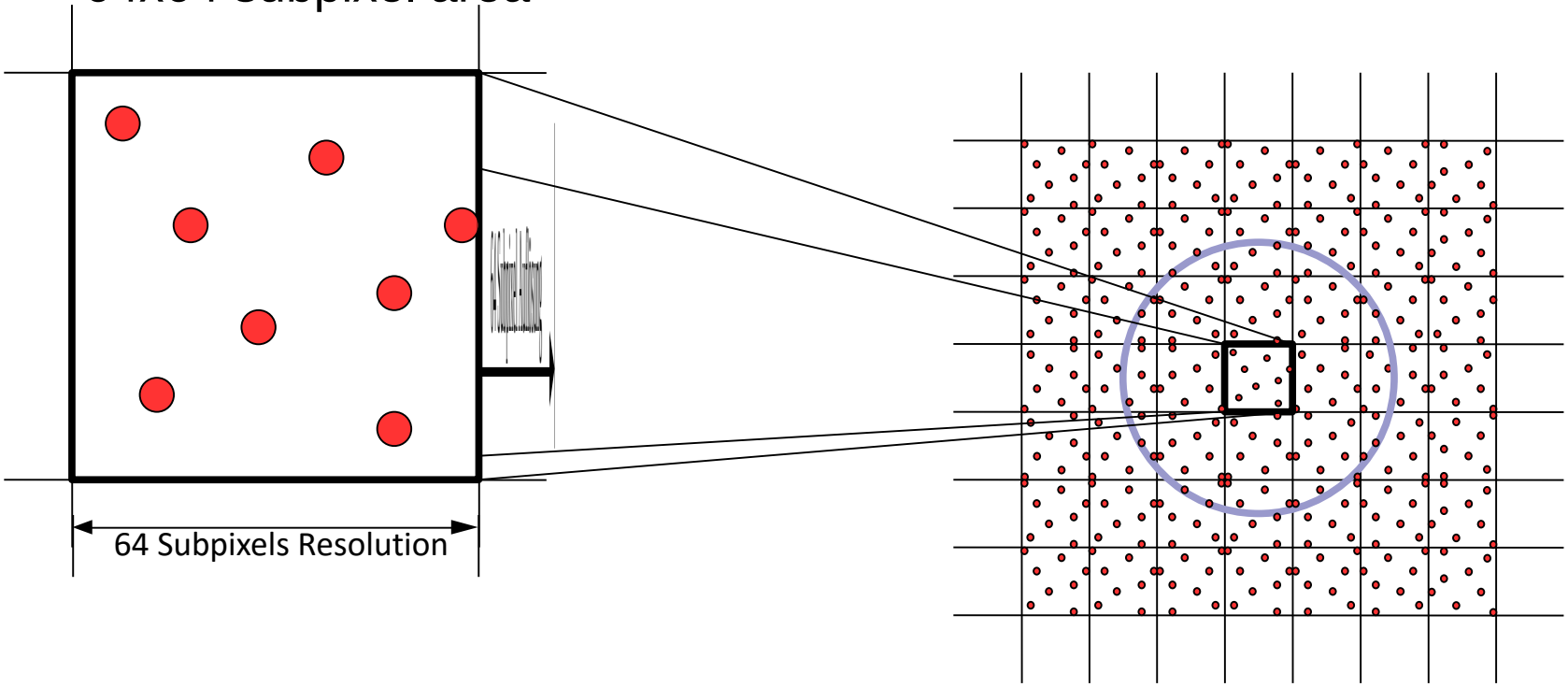


16 samples

Pictures: IT Center RWTH

Multisampling on XVR 4000

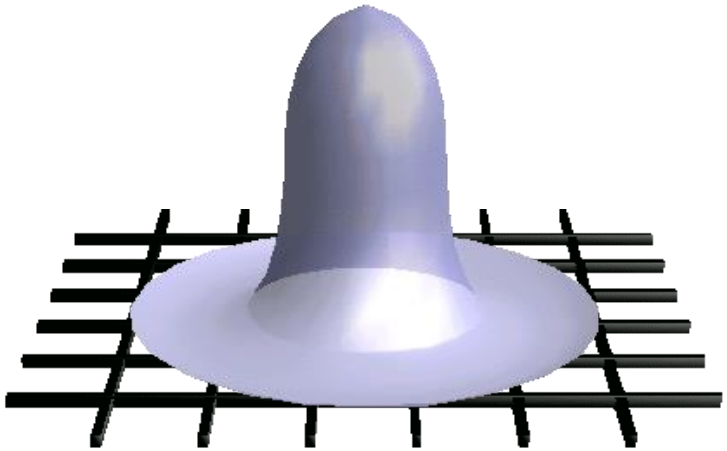
1. 1-16 Samples, random distribution that changes with each pixel, calculated in 64x64 subpixel area
2. Samples are collected and filtered within an area shaped as circle and 5x5 pixels large



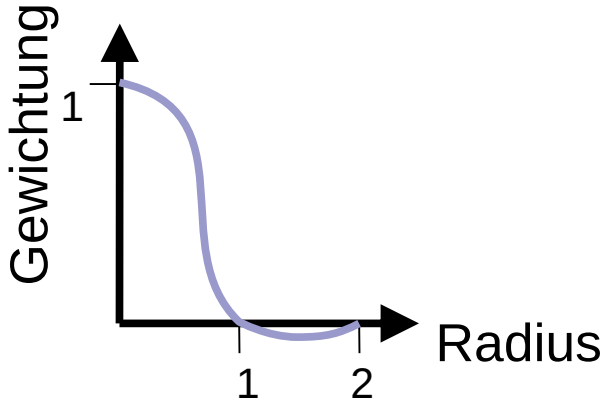
Courtesy of C. Gonzalez, Sun Microsystems

Multisampling on XVR 4000 (cont.)

3. All samples within the circle are weighted, normalized and accumulated



4. The filter curve is programmable. User can adapt filter characteristics to the application.



Courtesy of C. Gonzalez, Sun Microsystems

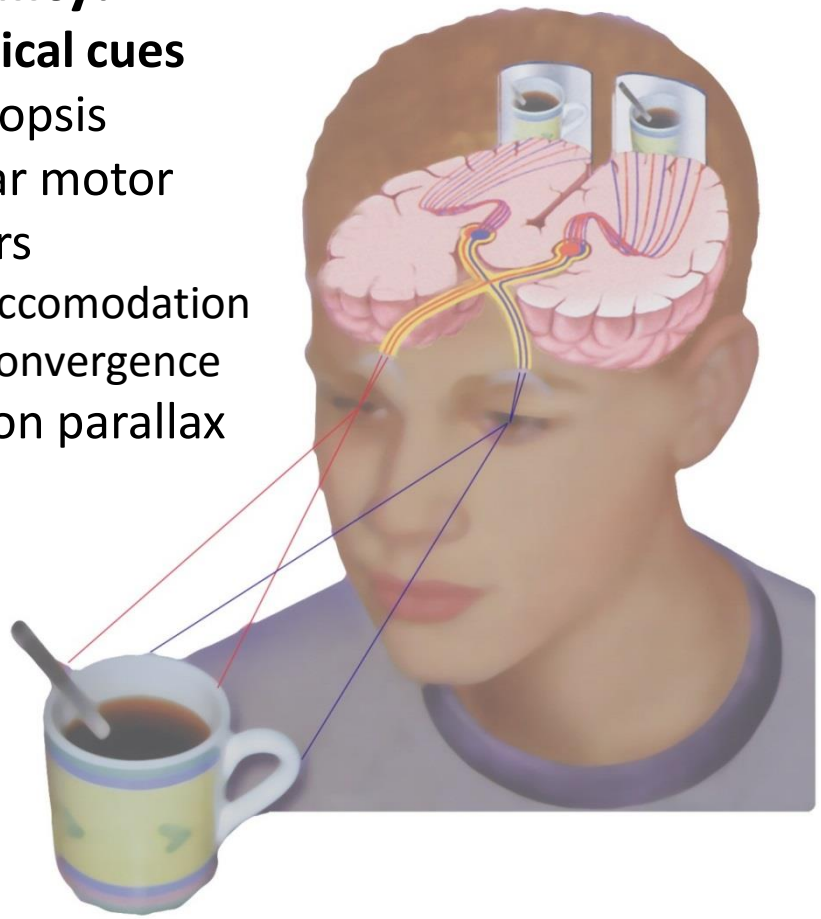
Physiological & Psychological Depth Cues

Traditional CG:

- **Psychological cues**
  - Perspective shortening
  - Occlusion of objects
  - Light and shadows
  - Texture gradients
  - Atmospheric perspective

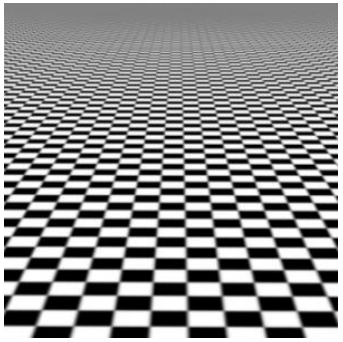
Virtual Reality:

- **Physiological cues**
  - Stereopsis
  - Ocular motor factors
    - Accomodation
    - Convergence
  - Motion parallax



Addressing the Psychological Depth Cues:

Texture Gradient

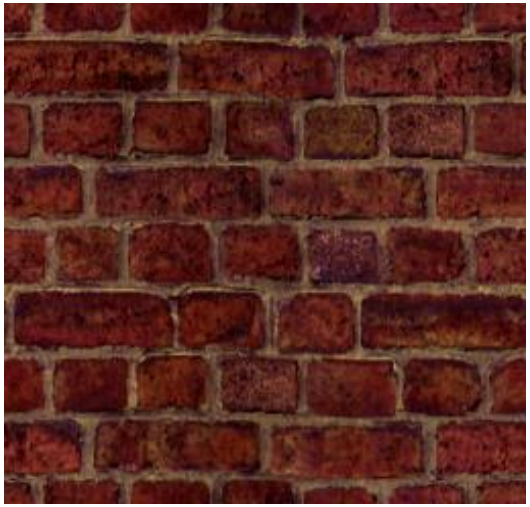


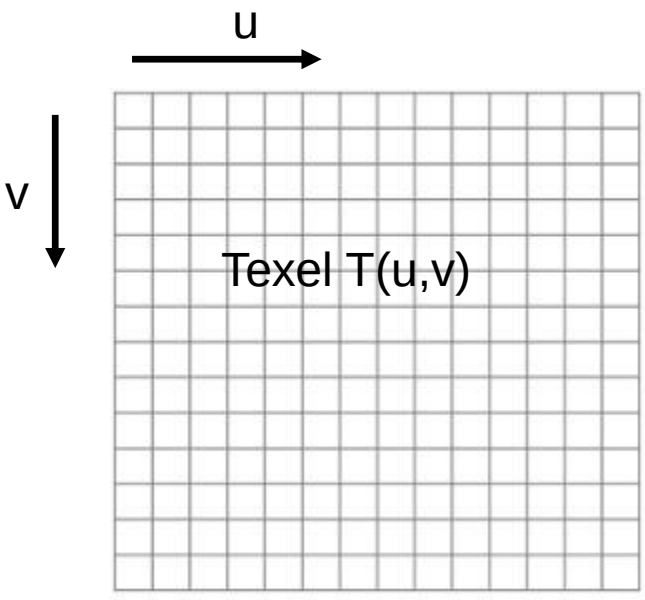
Addressing Real-Time Capabilities and Visual Realism in Virtual Reality

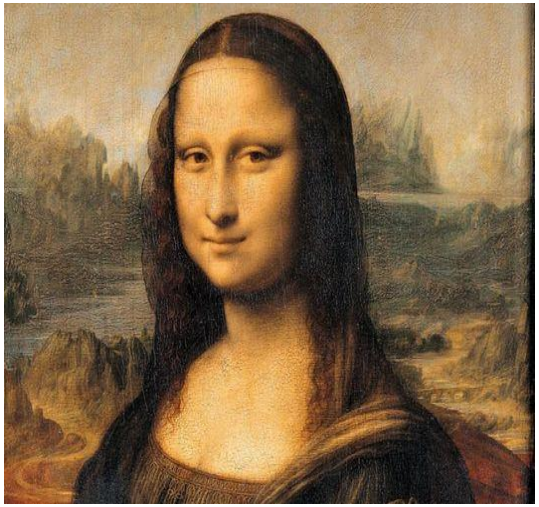
→ Texture Mapping

What is Texture Mapping?

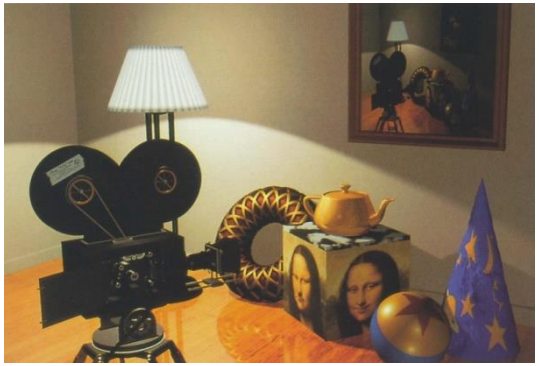
Texture Mappping: Usage of textures in CG (Catmull 1974)







Texture Map



Motivation

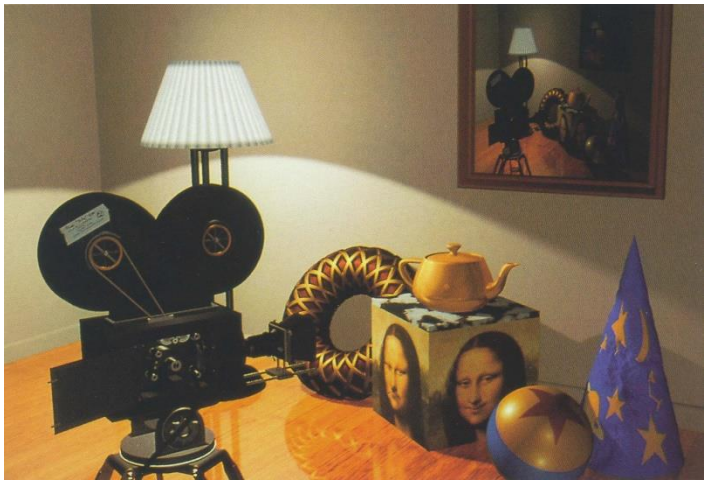
Increase realism of virtual scenes:

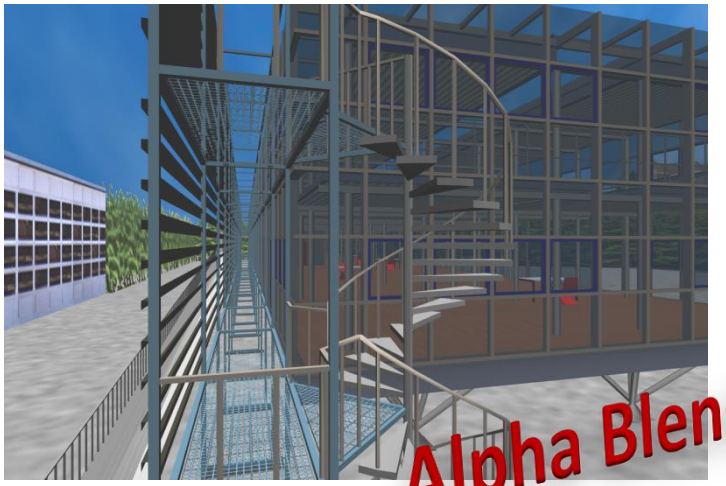
- Integrate real photos into the virtual scene
- Render the structure of surfaces without increasing geometry complexity
- Simulate mirror reflections without ray tracing
- Render semi-transparent surfaces (e.g., glass)

Scientific Visualization

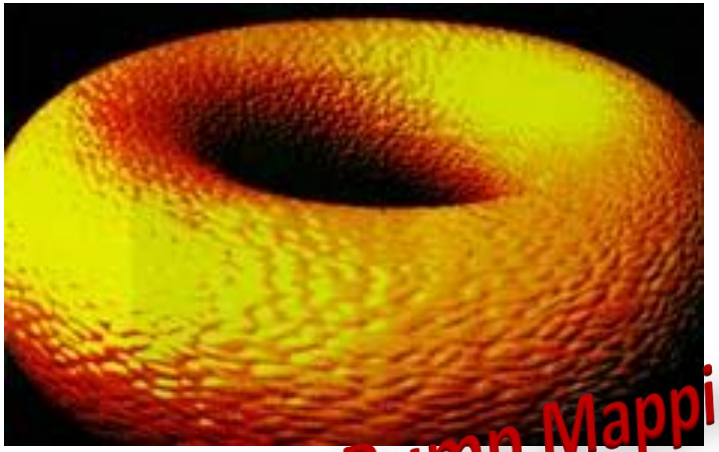
- LIC (linear integral convolution)
- Imaging in medicine: Render volume by textures

Modulation of Color, Transparency, Normals, Specular Components





Alpha Blending



Bump Mapping

Environmental Mapping

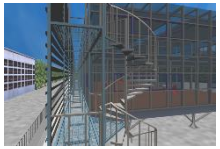
Texture Mapping: Parameter Modulation

Parameter Modulation:

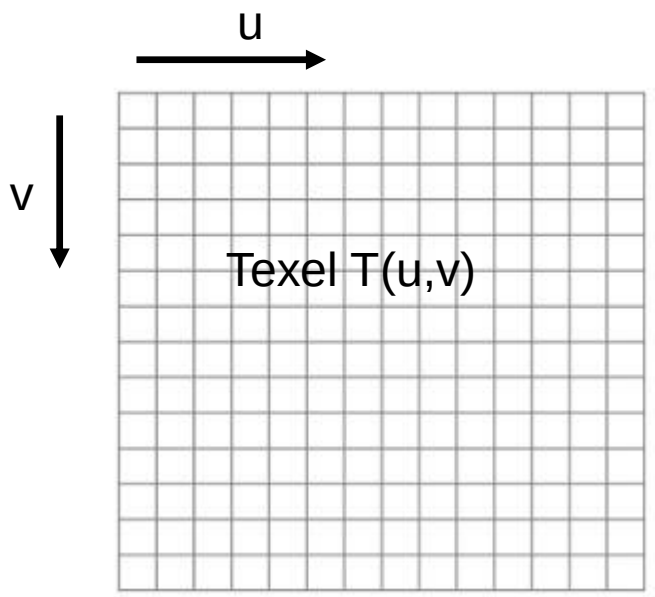
- Color
  - Photo
  - Functional or procedural
  - Randomized

- Specular and diffuse component (Environment Mapping)

- Transparency (α- Blending)



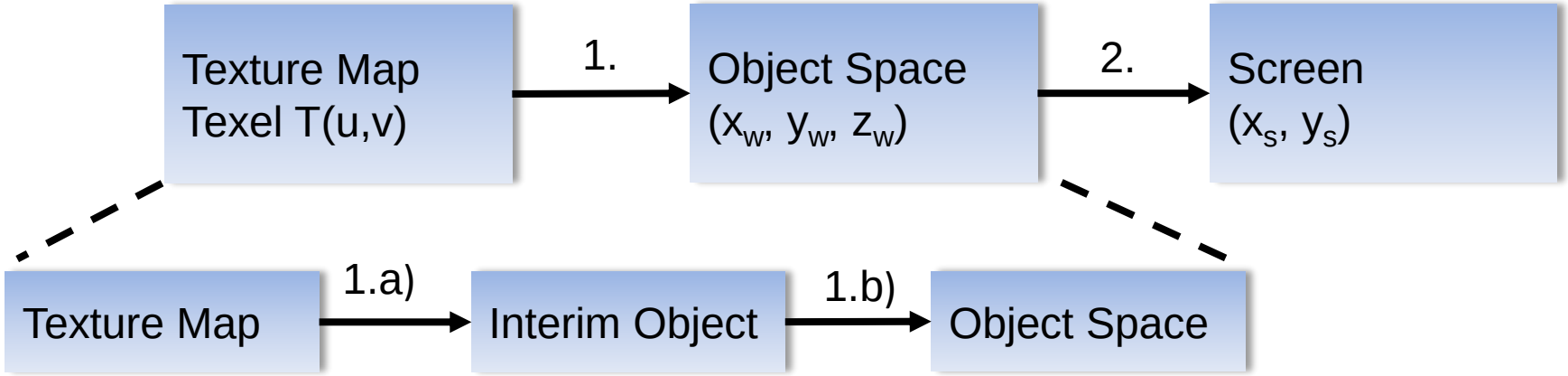
- Perturbation of the normal vector (Bump Mapping)



Texture Map

From Texture Map to Pixels

1. Surface parametrization
2. Projection (interpolation, see Shading)

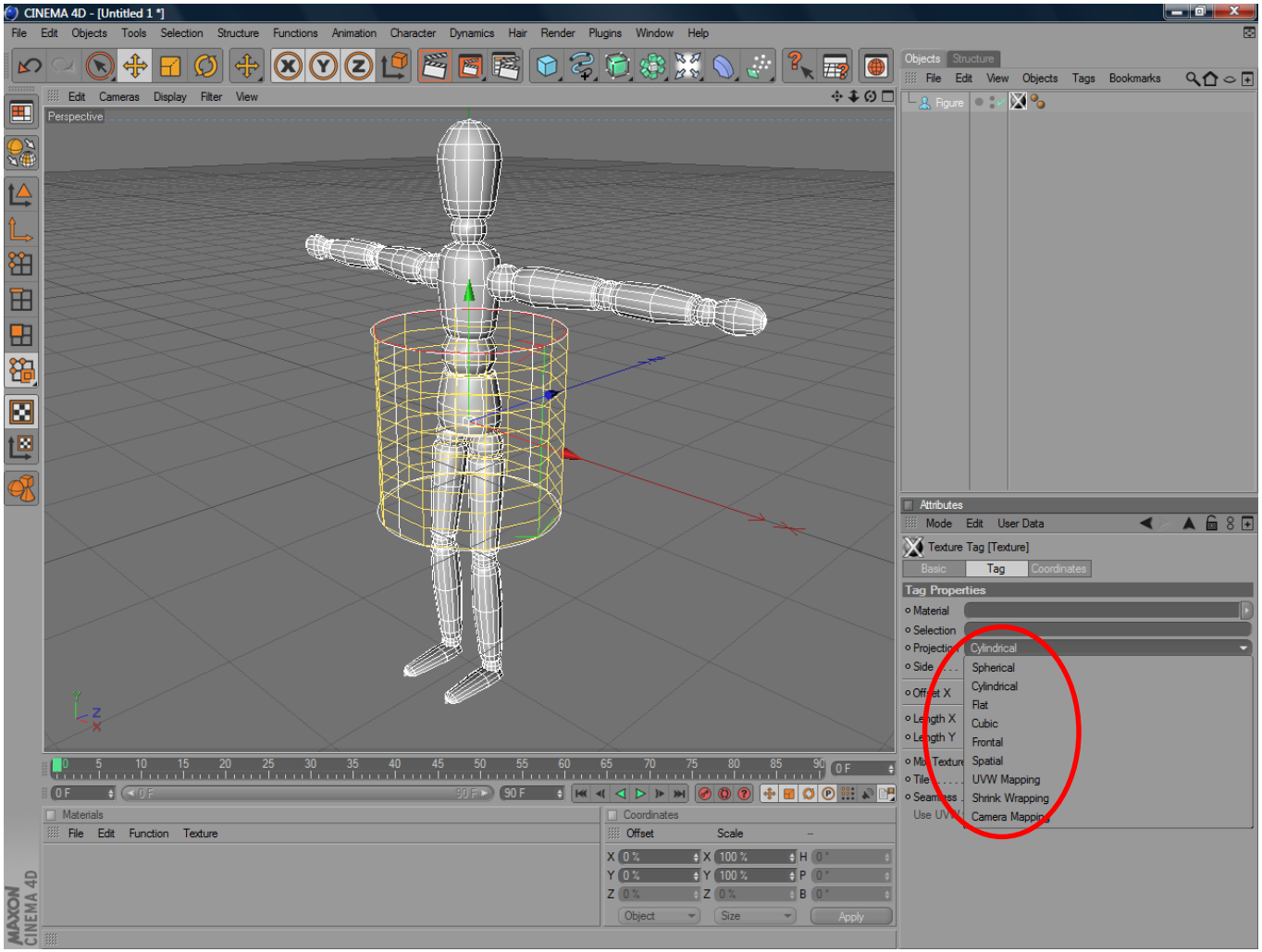


Use 2-phase parametrization to avoid distortions for polygon models:

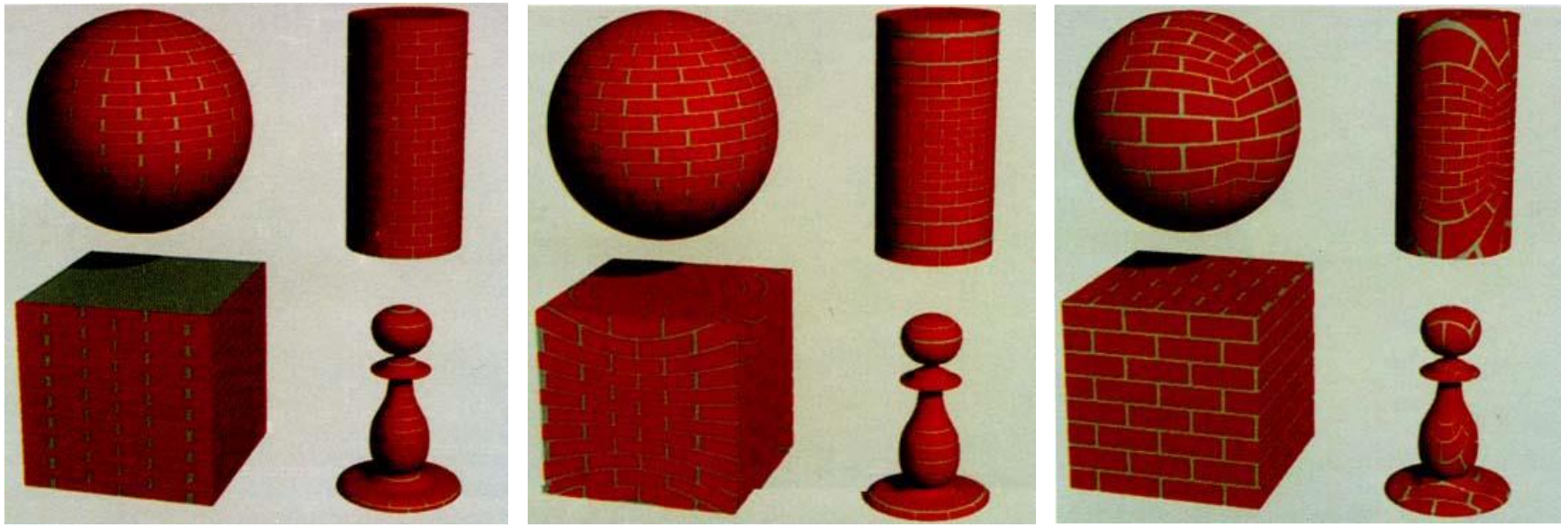
a) S-Mapping

b) O-Mapping

Texture Mapping User Interface in Cinema4D



S-Mapping – Effects of Different Interim objects

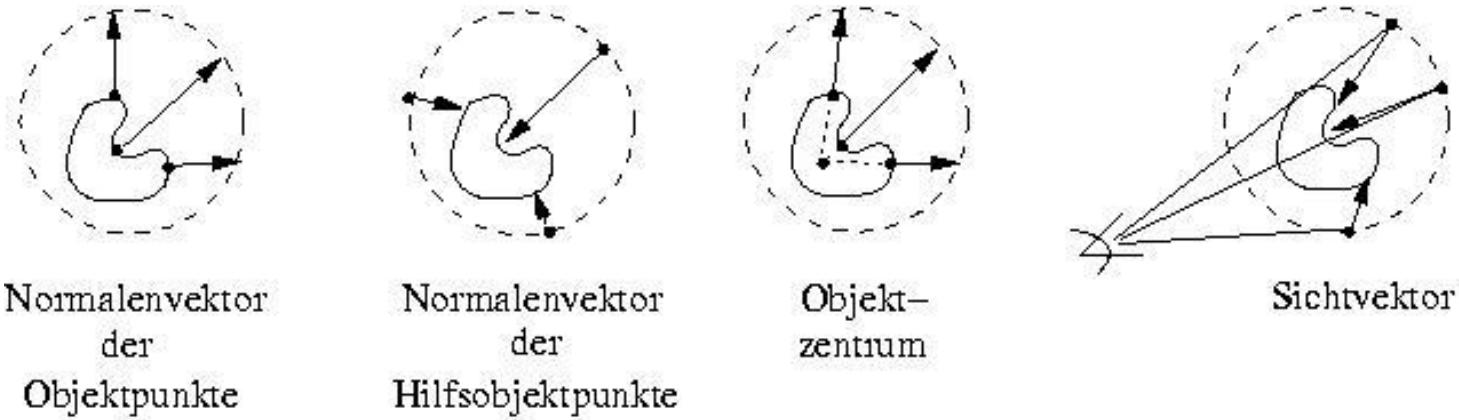


Picture: Watt

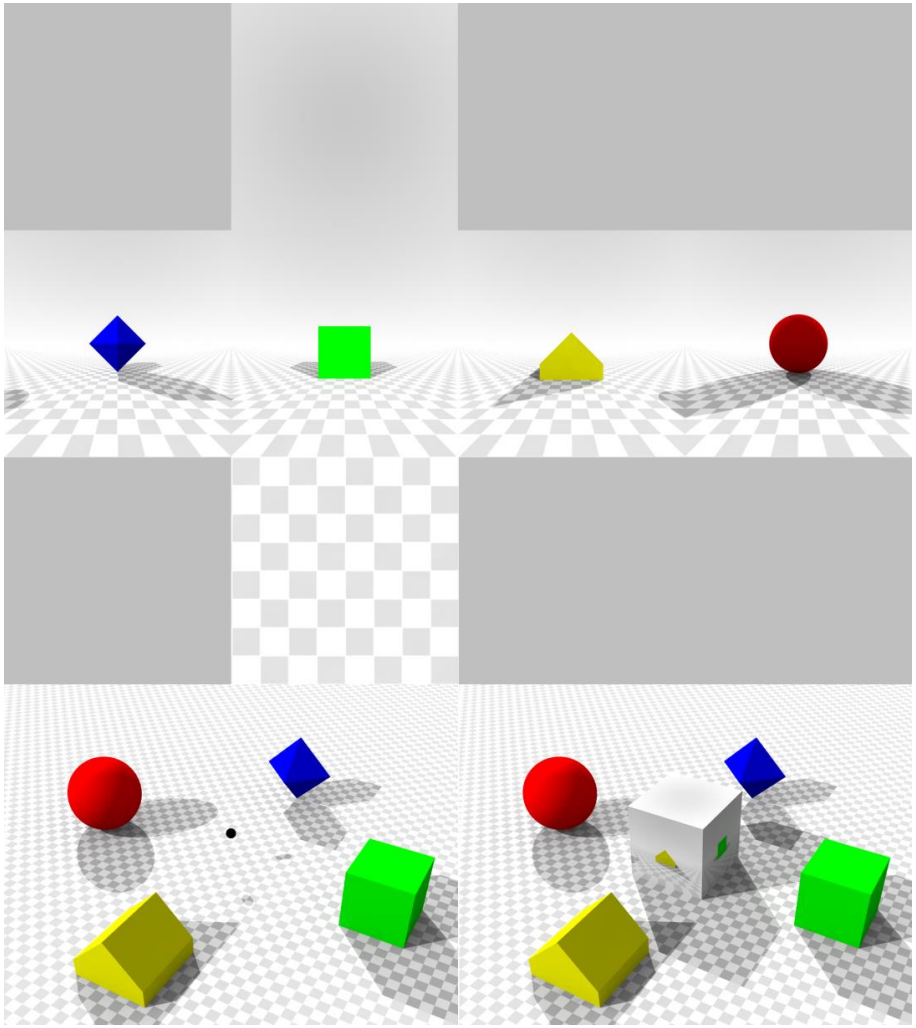
O-Mapping

$$T'(x_i, y_i, z_i) \rightarrow O(x_w, y_w, z_w)$$

- 4 strategies:
  1. Intersection point of the surface normal vector at every position (x_w, y_w, z_w) and T´
  2. Intersection point of the interim object at every position (x_i, y_i, z_i) with the object surface
  3. Projection of the object midpoint onto the interim surface
  4. Mirror reflection of a sight vector at the interim object. Texture seems to walk over the object surface when the viewer moves (Environment Mapping).

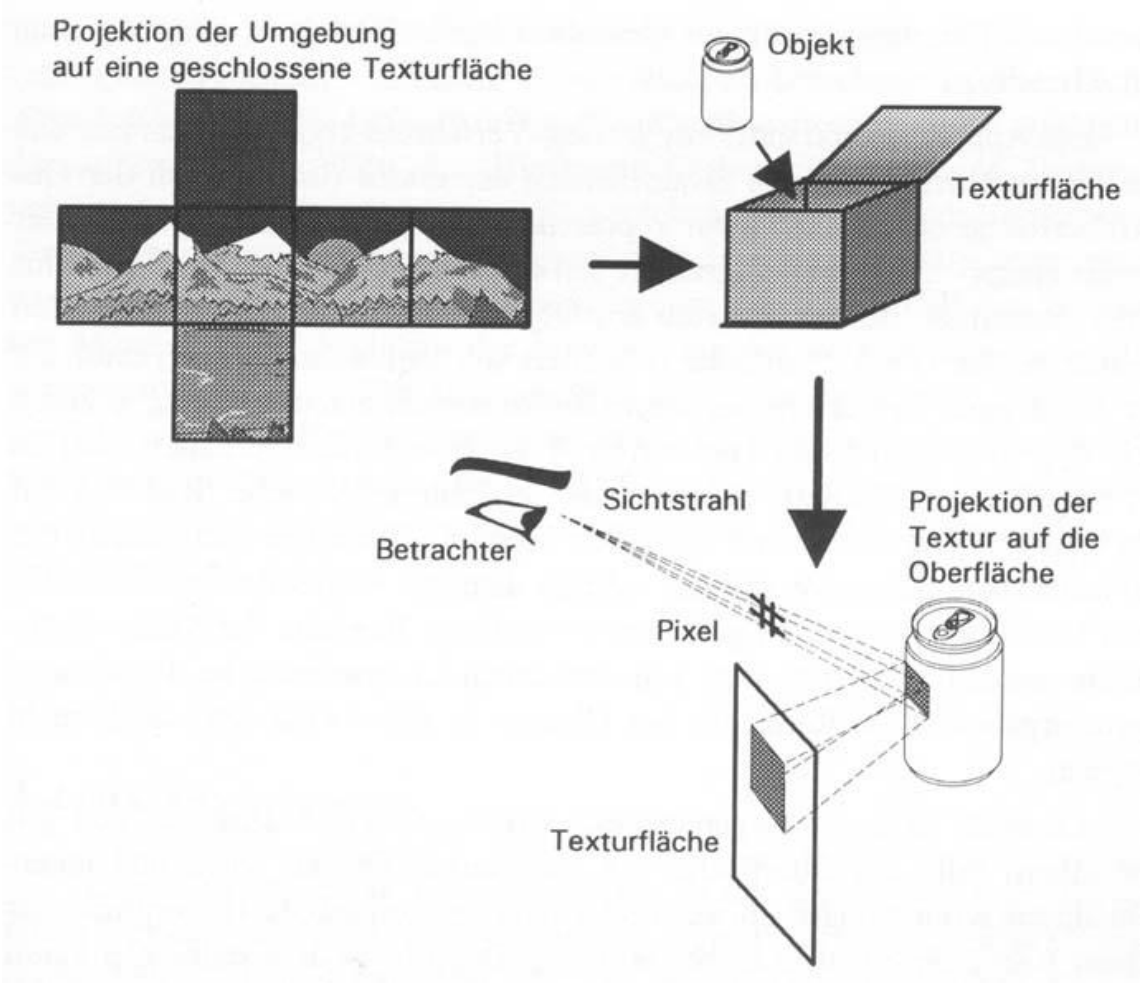


Cube Mapping



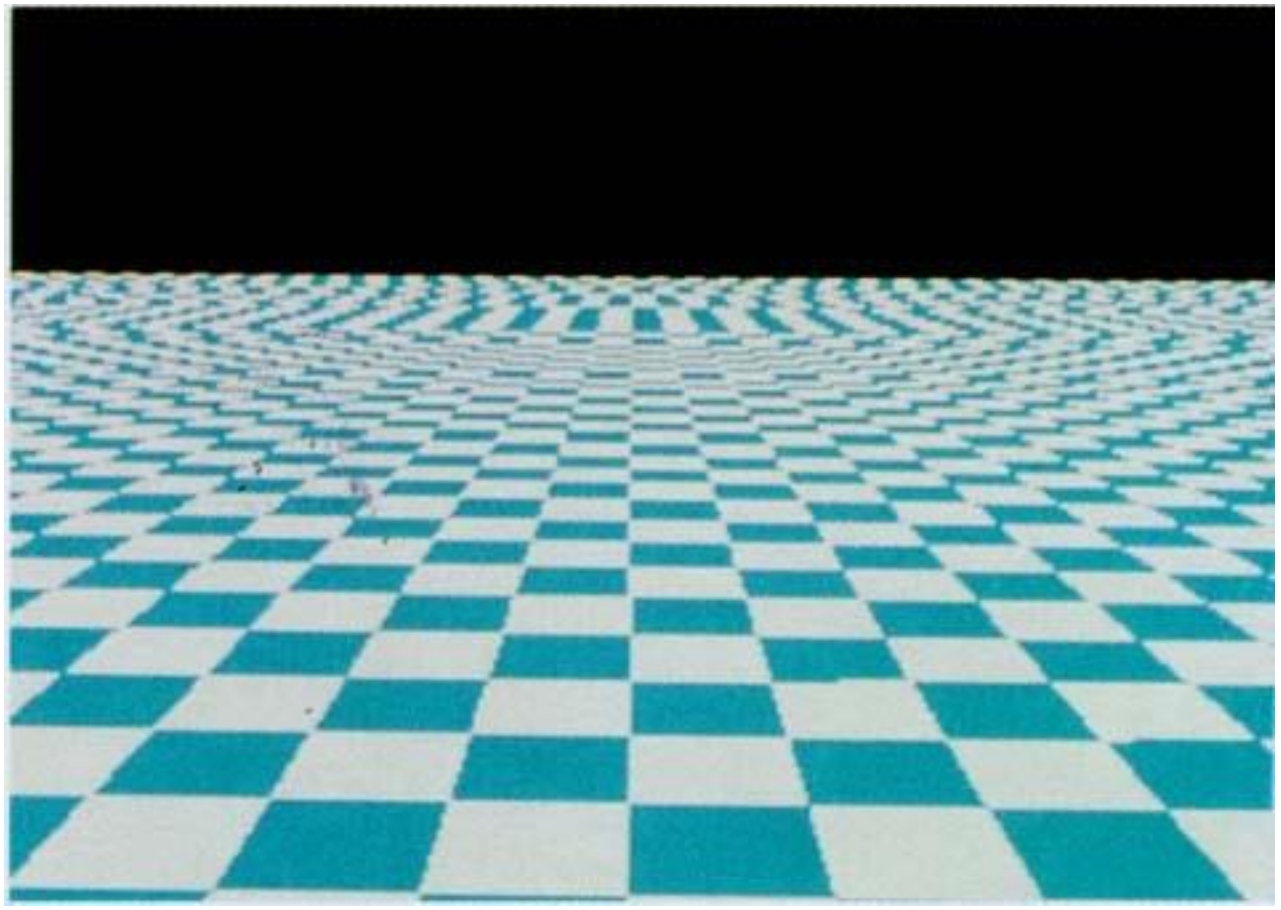
Picture: www

The Principle of Environment Mapping



Picture: Tönnies

Aliasing Example



Aliasing Effects

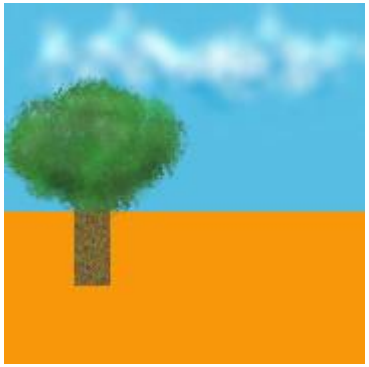
2 extreme situations:

1. Viewer is very close to the textured object: Only one texel for many pixels (tiling effect)
2. Textured object is very far from the viewer: Many texels for only one pixel (see example on last slide)

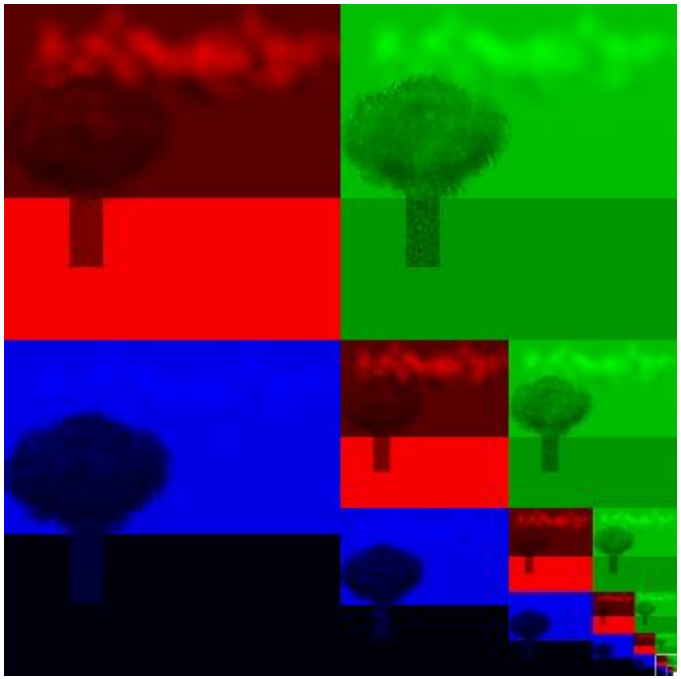
ad 1/2. Texture hierarchies and Mip Mapping

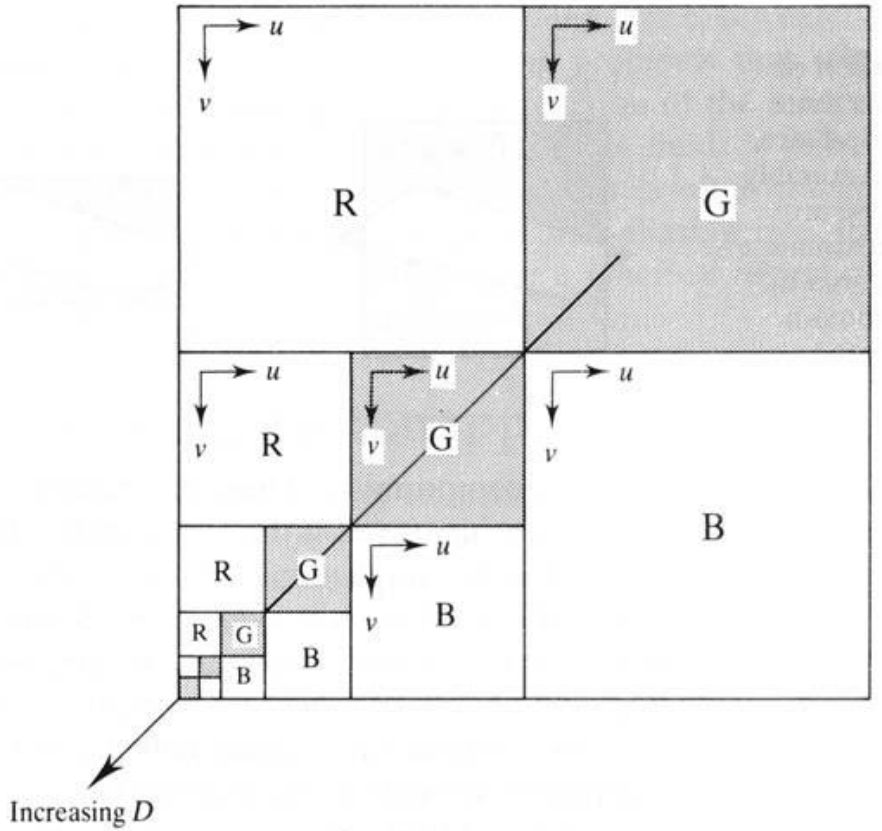
ad 2. Shrink Wrap Method and Inverse Mapping

Texture Hierarchies

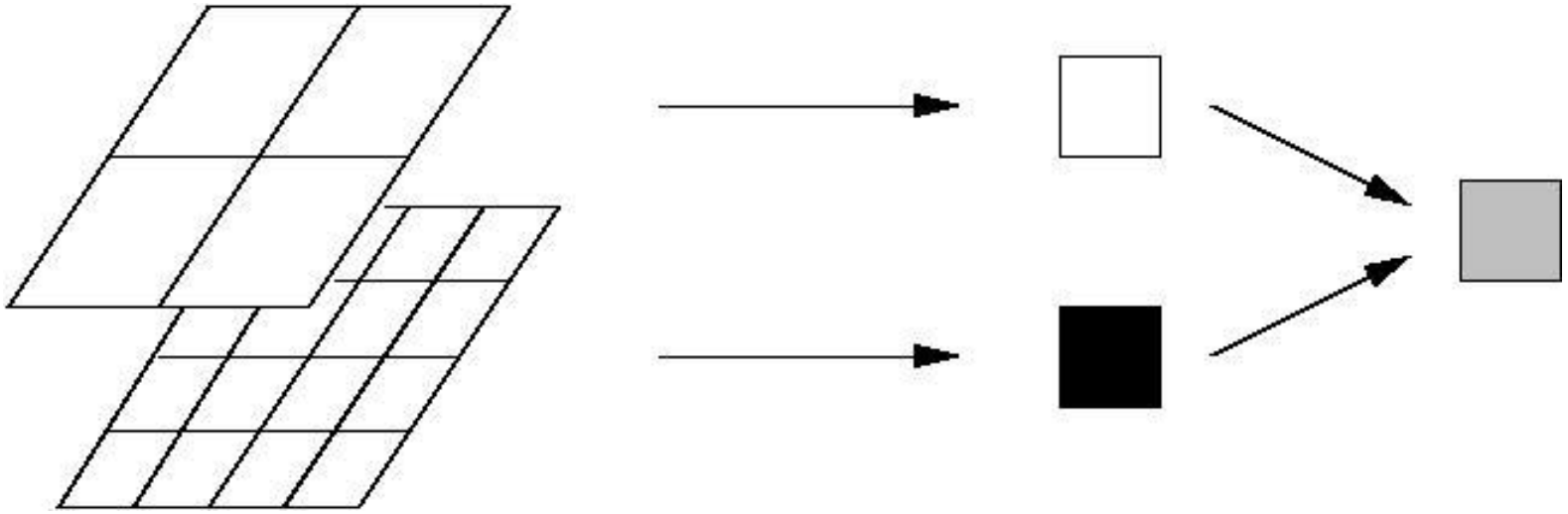


Picture: WWW



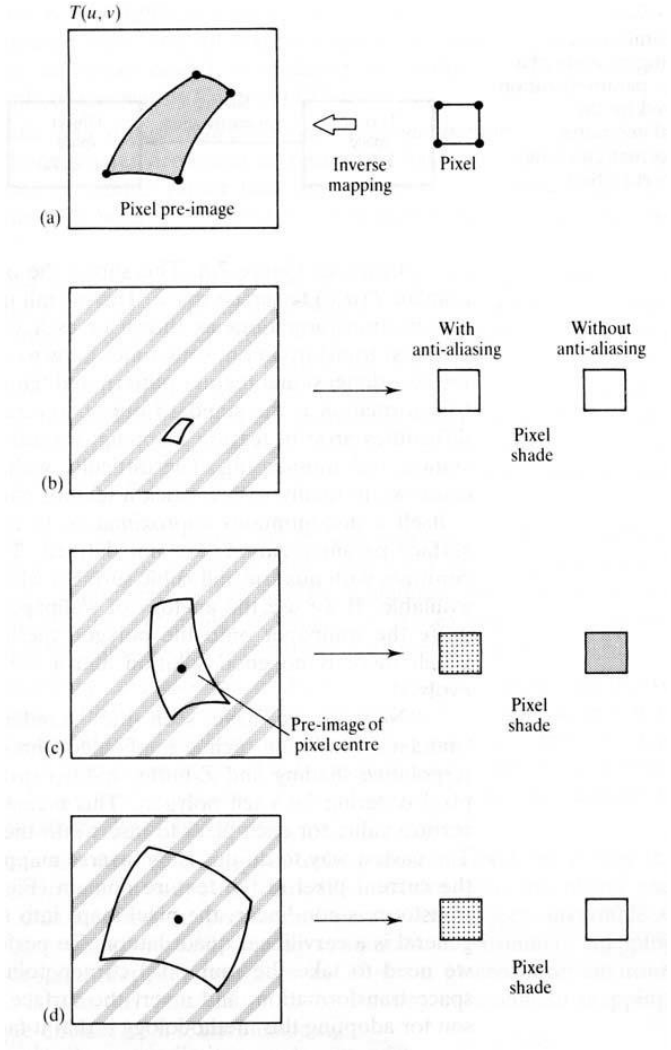


Mip Mapping – Trilinear Interpolation



Shrink Wrap Method & Inverse Mapping

- Start from the 4 pixel corners (Inverse Mapping)
- Transform all 4 corners into texture coordinates
- Integrate over the surface in the texture map that is covered by the pixel



The Classical Rendering Pipeline for Virtual Reality - Summary

- **Real-time:**
  - Polygonal representation → transformations on polygon vertices
  - Homogeneous coordinates → concatenation of transformations
  - Simple/local illumination/reflection model
  - Bilinear interpolation in shading
  - Backface culling → early elimination of non-visible polygons
  - Z-buffering → hardware-supported occlusion
  - Texture Mapping → details without geometry

- **Psychological depth cues:**
  - Projection → perspective shortening
  - Shading → light & shadows (limited)
  - Z-Buffering → occlusion
  - Texture Mapping → texture gradients

Goal

Exact correspondence
of visual perception
in the real world and in the virtual world

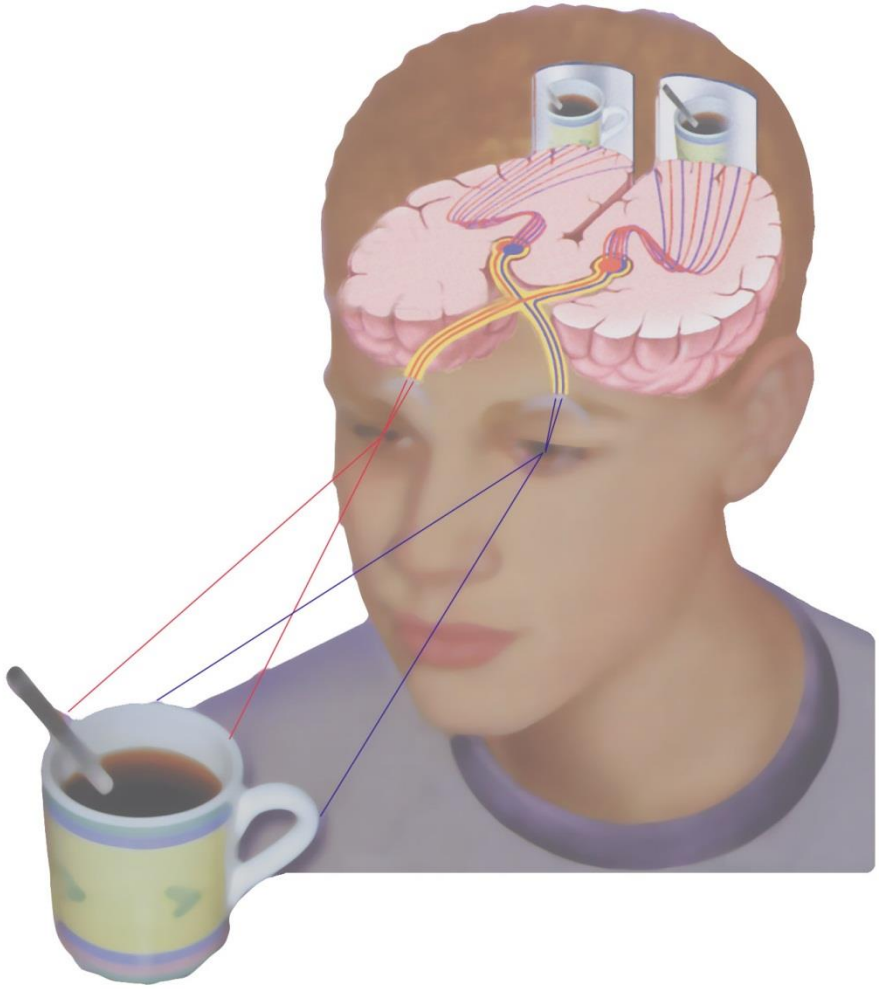
Limitations of Real-Time Computer Graphics

- Physical simulation of light
- Polygonal representation of objects
- Resolution (→ HW limitations)
  - Frame buffer
  - Color depth
  - Z-buffer

What about the physiological depth cues?

Physiological Depth Cues

- Stereopsis
- Ocular motor factors
  - Accomodation
  - Convergence
- Motion parallax



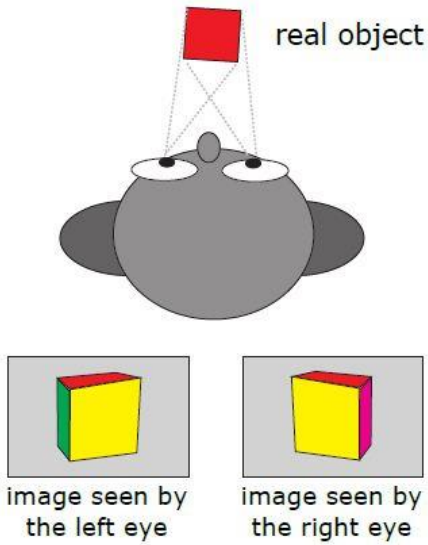
Aspects of Vision

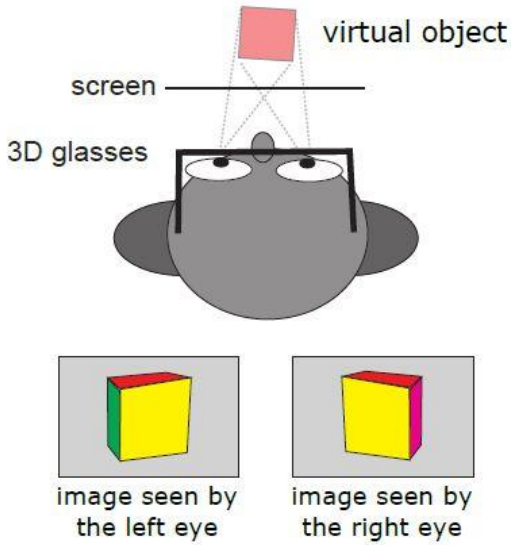
- Human Factors
  - Anatomy of the Human Eye ✓
  - Perception: Depth Cues in Vision, ✓ Photometry
- Technology
  - Graphics Hardware
  - Displays: Projectors, Screens, Brightness, …
  - Stereo Techniques
- Algorithms
  - Geometrical Modeling
  - Rendering
    - Rendering Pipeline ✓
    - Stereoscopic Projections
    - Viewer-Centered Projection
  - Global Illumination: Ray Tracing, Radiosity

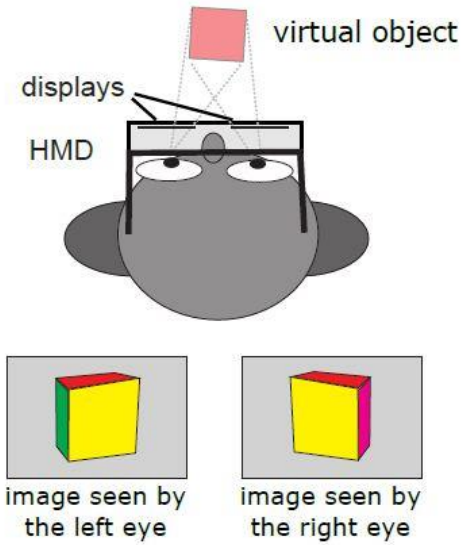
Addressing Stereo & Motion Parallax:

Stereoscopic, Viewer-Centered Projections

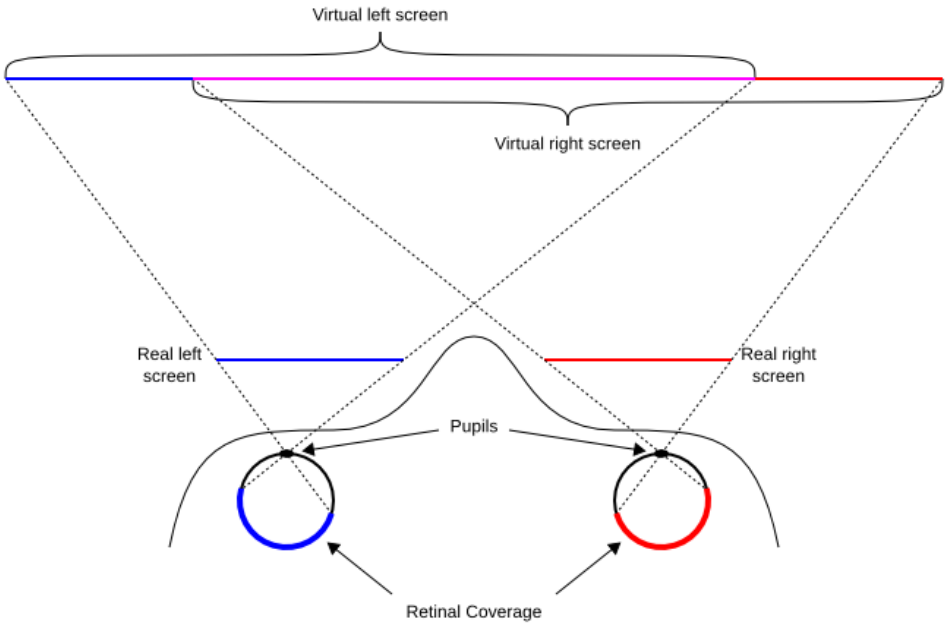
Stereo in Head-Mounted &Room-Mounted Displays







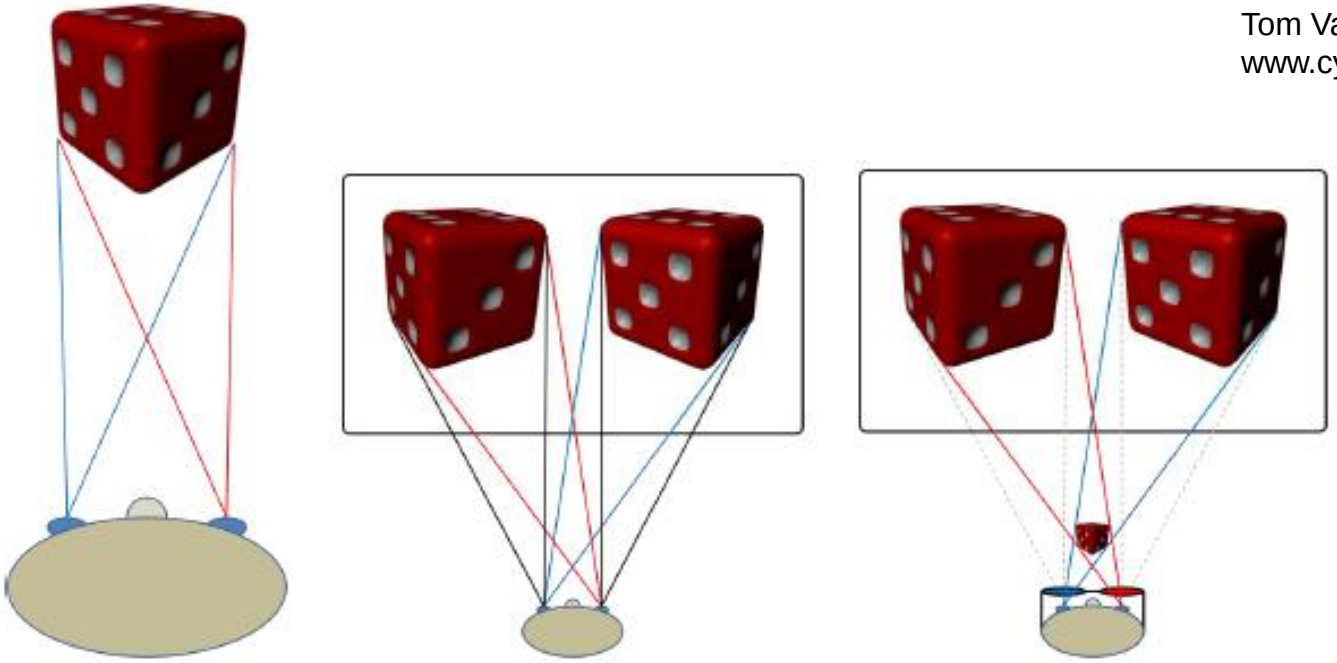
Stereo in Head-Mounted Displays



Oliver Kreylos, Doc-Ok.org

- Separate screens for left and right eye, screens move with user's head
- (Asymmetric) view volume does not change while user is moving
- Optics: Create larger virtual screens at a longer distance away to allow users to properly focus on those screens
- Proper calibration eyes/screens needed

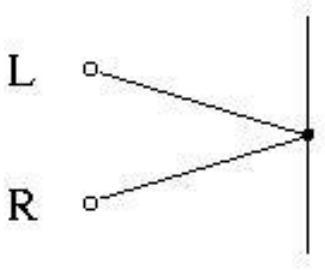
Stereo Parallax on Room-Mounted Displays



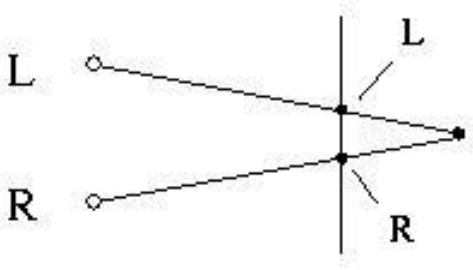
Tom Vaughan,
www.cyberlink.com

- Single screen for left and right eye, screen does not move with user's head
- Stereo glasses separate left and right image for the user's eyes
- (Asymmetric) view volume changes while user is moving → see later

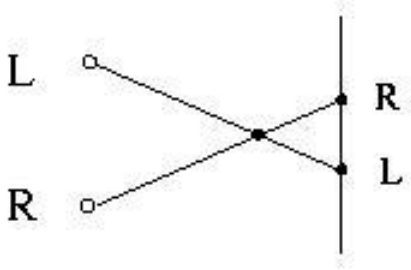
Stereograms



zero parallax

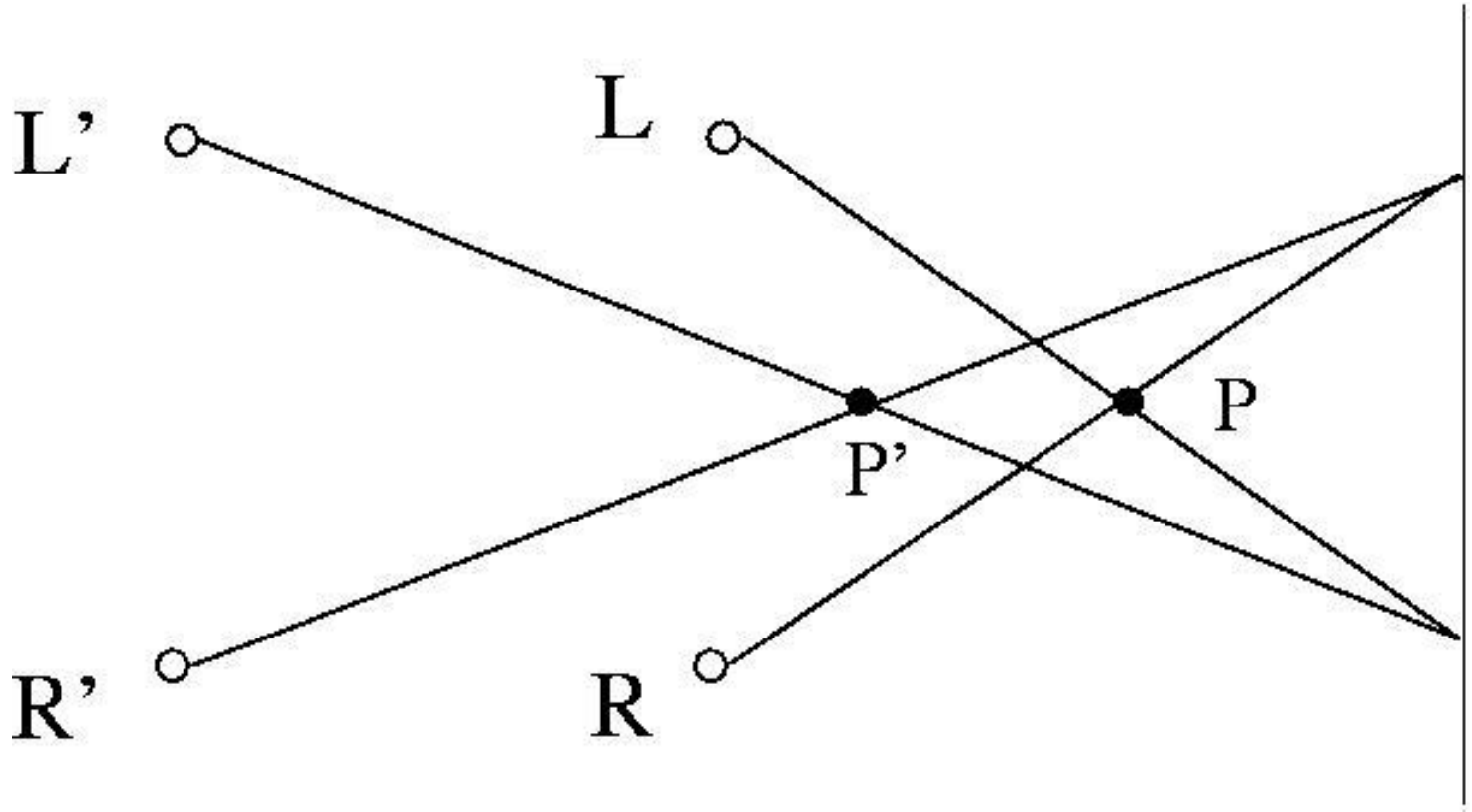


non-crossed
(positive) parallax

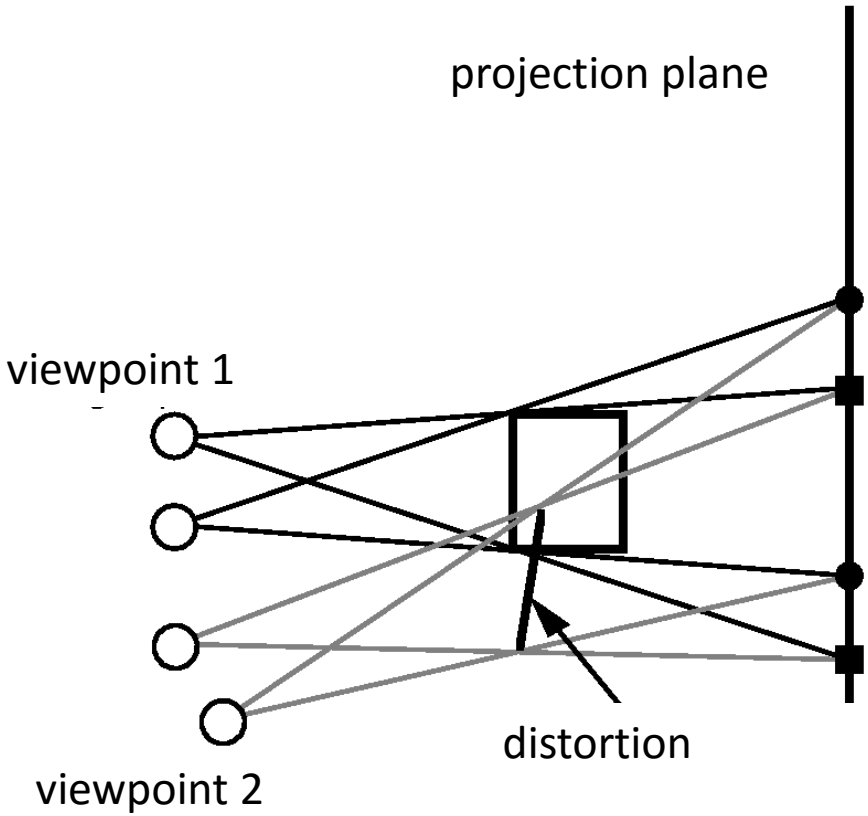


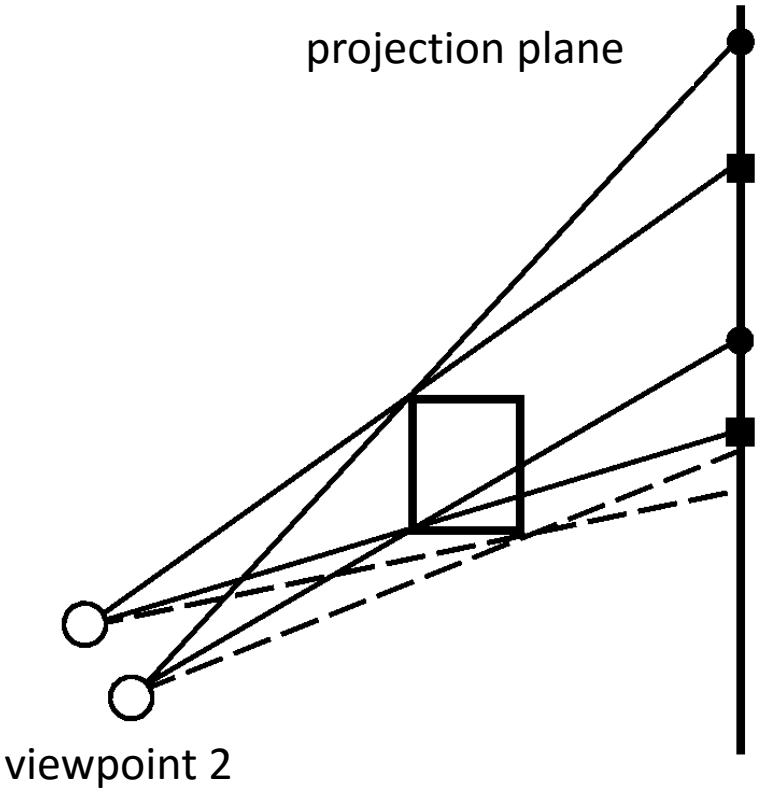
crossed
(negative) parallax

Distortions in Static Stereograms



Adaptation of projection to the viewpoint

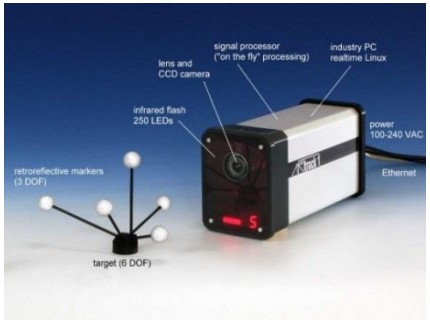


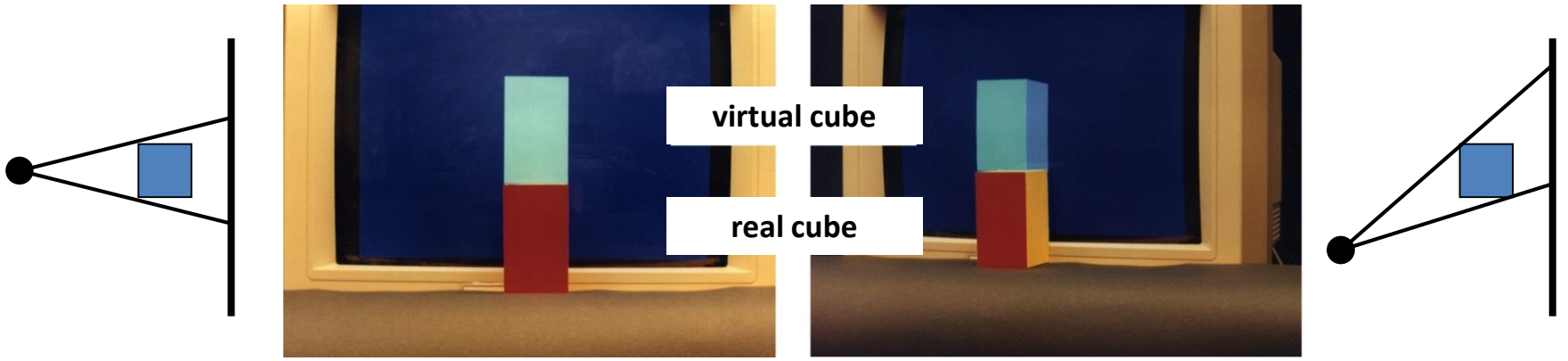


Motion Parallax & Viewer Centered Projection

Stereo parallax

Motion parallax



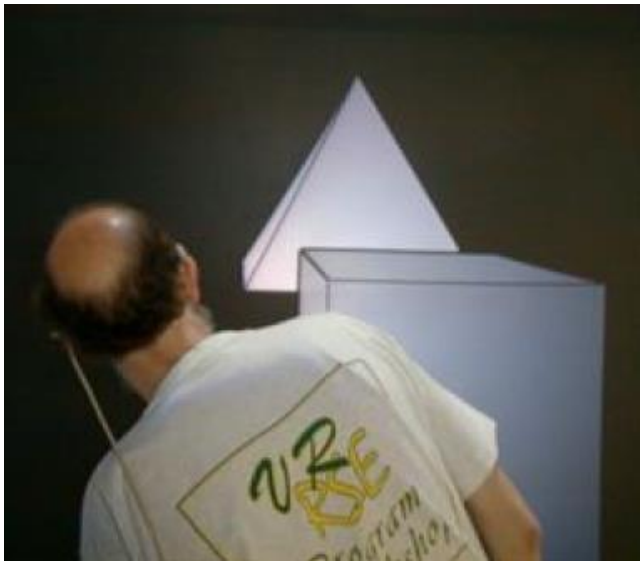


Video: Viewer Centered Projection

Video: Courtesy of VRVis, Vienna

The Effect of Motion Parallax

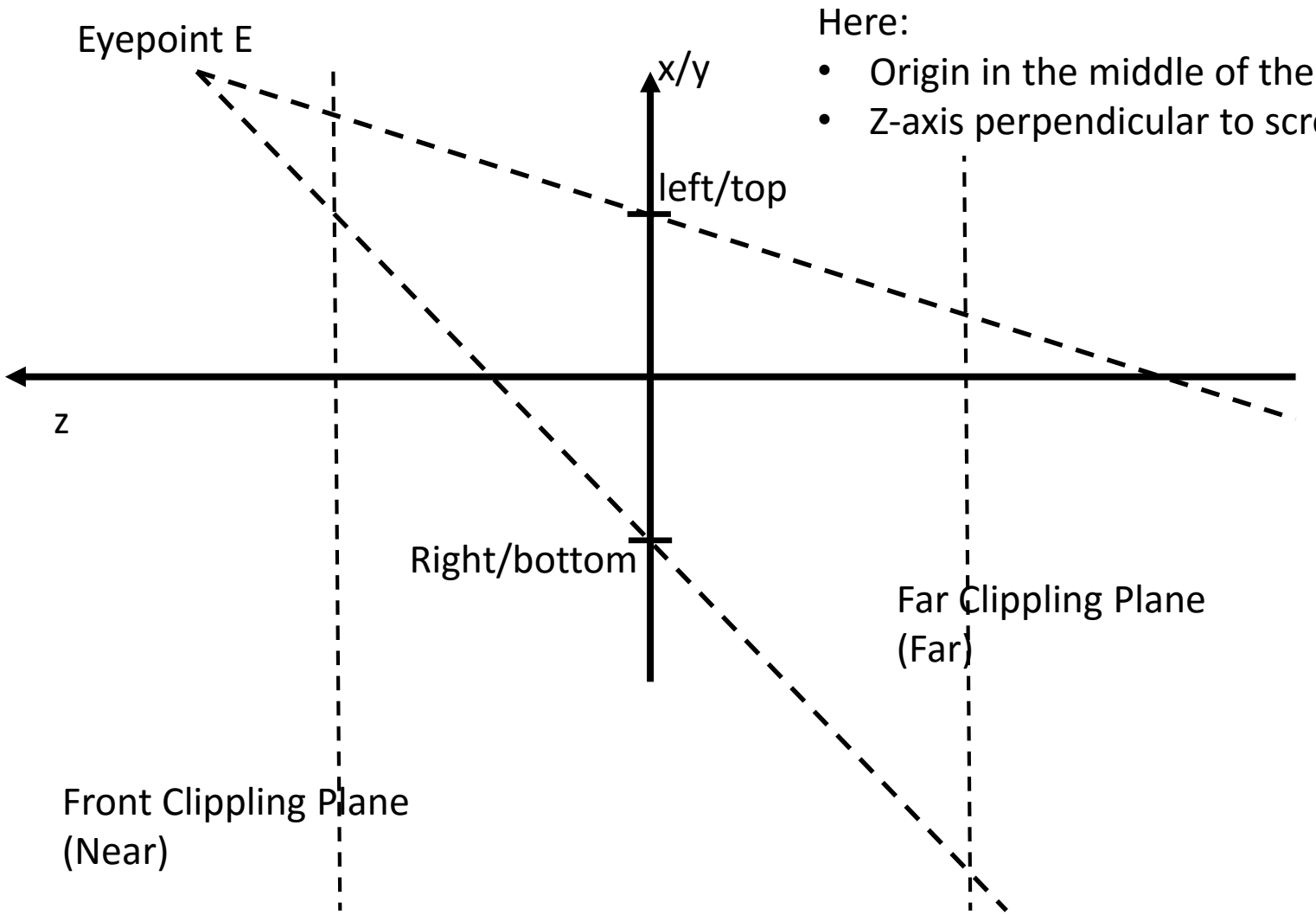
Courtesy of Bill Sherman



VR Needs Ego-Centric Perspective

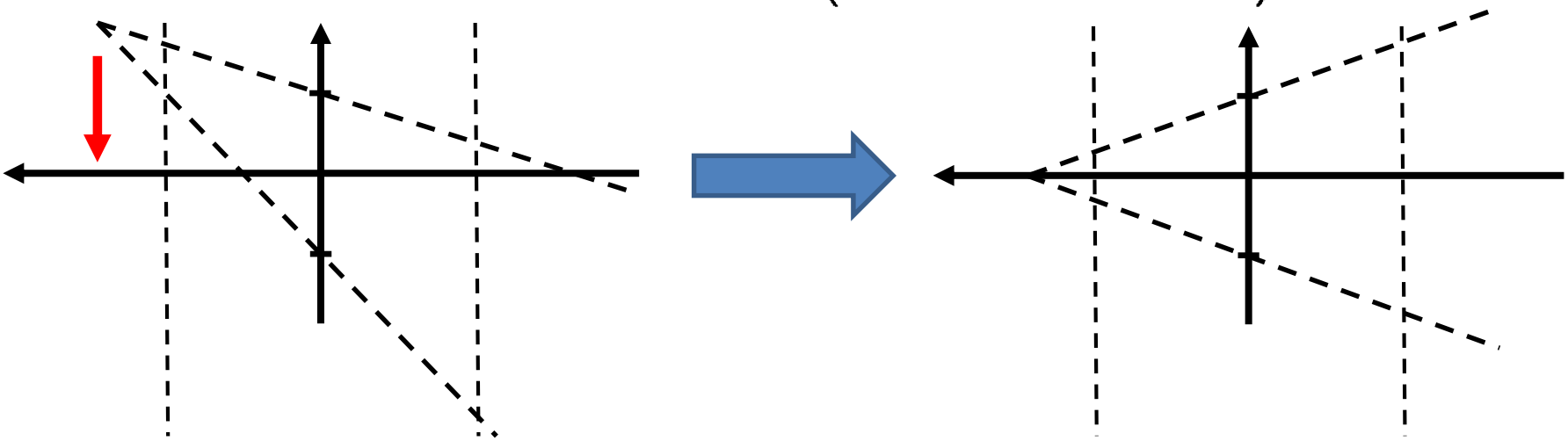
- Ego-centric perspective:
  - Projection is coupled to the user's viewpoint
    - Eye position = projection's center of origin
    - Motion parallax
  - Requires real-time head tracking
  - Latency can cause motion sickness
  - # exo-centric perspective

Ego-Centric Perspective in Room-Mounted Displays



Shearing the View Volume

$$P_1 = SH_{xy}(\frac{-E_x}{E_z}, \frac{-E_y}{E_z}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{-E_x}{E_z} & \frac{-E_y}{E_z} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



- Symmetric view volume
- Leaves image plane unchanged

➢ Standard perspective projection

The Significant Contribution of Motion Parallax

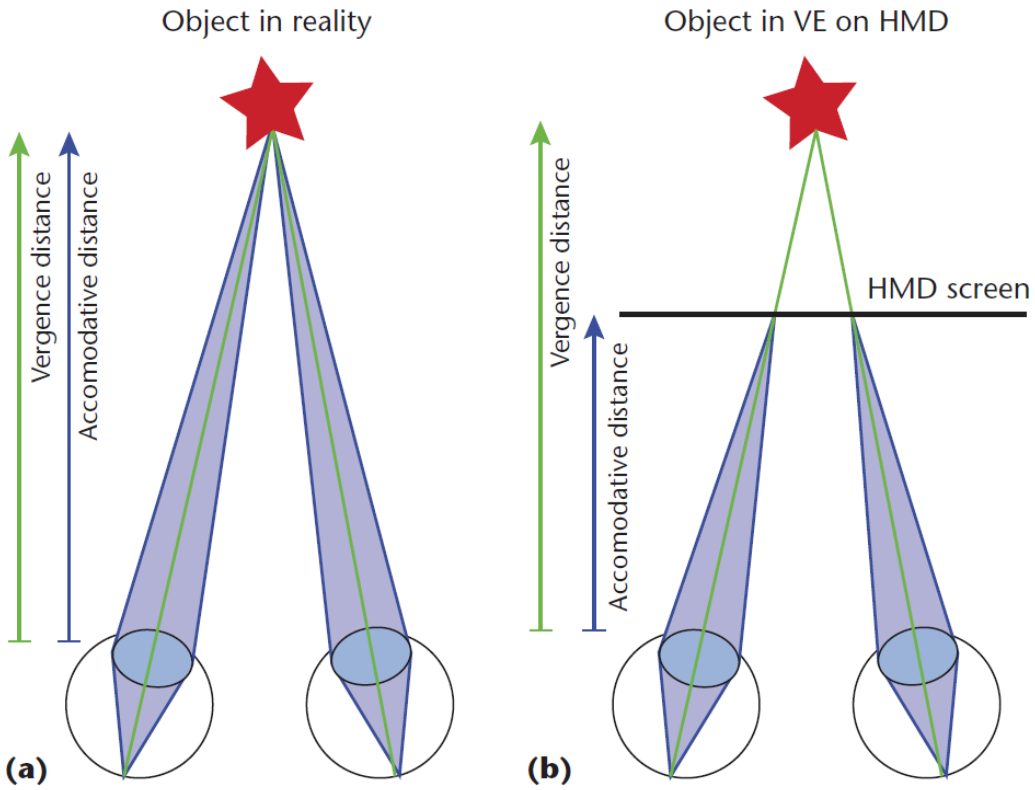
SIGGRAPH 2014: „The Box“ from The Creators Project (see YouTube)

Goal

Exact correspondence
of visual perception
in the real world and in the virtual world

Physiological Depth Cues with VCP

- Stereopsis ✓
- Motion parallax ✓
- Ocular motor factors
  - Accomodation
  - Convergence



Kieran Carnegie and Taehyun Rhee, CGA Journal 2015

Perception of Distances in VR

Distances are typically under-estimated in virtual environments

- **Influence candidates:**
  - Rendering quality
    - Illumination model
    - Positions of light sources
  - Rendering speed: Framerate
  - Display hardware characteristics / quality
    - Resolution / aliasing
    - Brightness uniformity
  - Accomodation / Convergence

- Distance estimation „nearly“ independent from rendering /display technology
- Many contradictory studies
- No explanation for under-estimation