Recap

CS 148: Summer 2016
Introduction of Graphics and Imaging

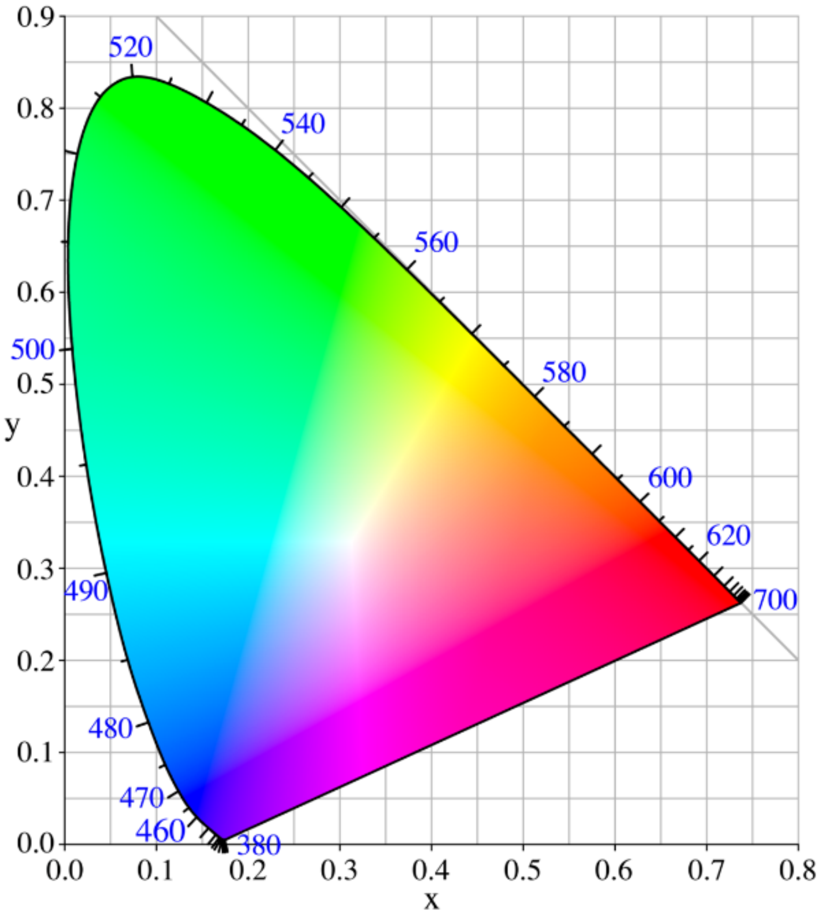
Zahid Hossain

Announcements

- Demo Day: 11th Aug, 1 PM – 7 PM (Bishop Auditorium)
    - Please fill out the sign-up sheet (See Piazza)
    - 4 mins presentation: 1 min Q/A
    - SCPD Students: sends us Youtube video by 10th 11:59 PM.
    - SCPD and students with special circumstances are not required to attend the full session.
- Report Deadline: 12th Aug before 11:59 PM.
    - Read the guidelines in the website https://mdzahidh.github.io/cs148/final/
    - Report format: Either a website or a PDF
- Please bring display adapters - just in case !
    - We will most likely have VGA only !
- Final Project Score Distribution (Course Grade)
    - 5% Milestone, 10% Demo, 5% Report (Total: 20%)

Week 1

Chromaticity Diagram



Projection of X,Y,Z on the plane

$$X + Y + Z = 1$$

$$x = \frac{X}{X + Y + Z}$$

$$y = \frac{Y}{X + Y + Z}$$

$$z = \frac{Z}{X + Y + Z}$$

z is redundant because

$$z = 1 - x - y$$

Color Spaces So Far

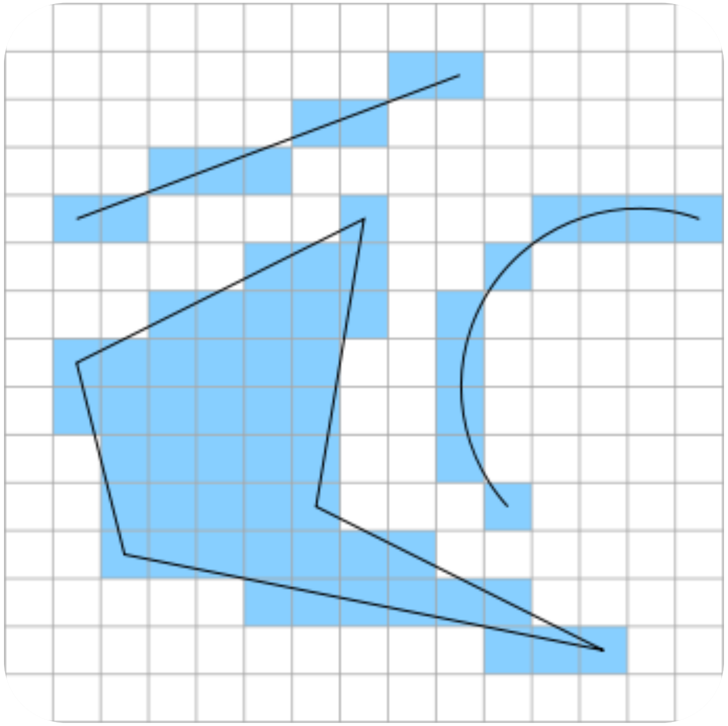
Color Space	Continuous	Perceptually Uniform
RGB	Yes	No
XYZ	Yes	No
Munsell	No	Yes
L*a*b	Yes	Yes
HSV	Yes	No

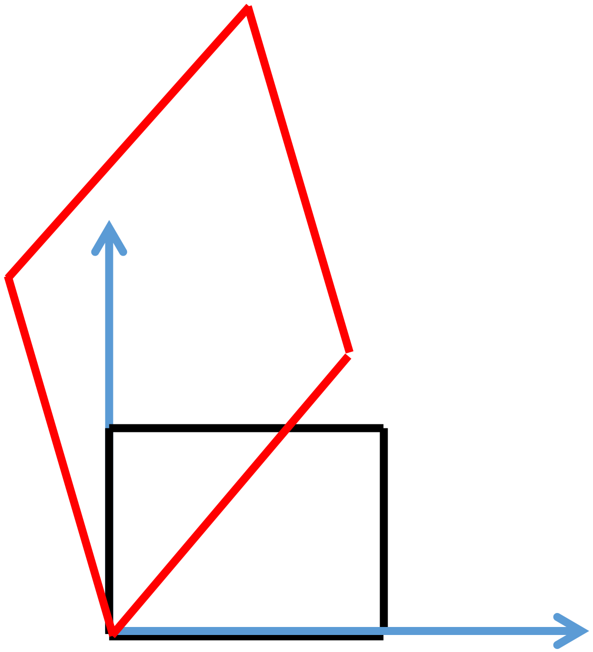
Other Spaces: CMYK

No black

Max black

http://en.wikipedia.org/wiki/CMYK_color_model





Week 2

Rasterization and Transformations

Midpoint Algorithm: Idea

$$y = \frac{\Delta y}{\Delta x} x + B$$

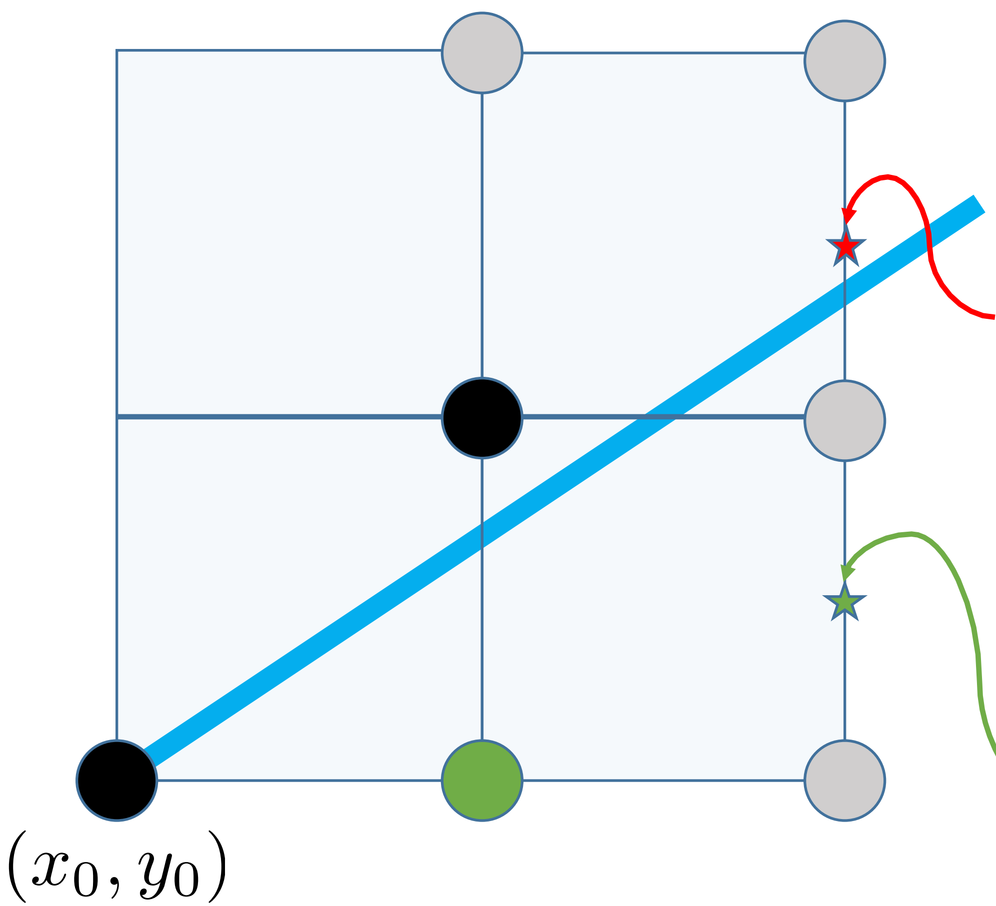
$$\implies \Delta x \cdot y = \Delta y \cdot x + \Delta x \cdot B$$

$$\implies 0 = \Delta y \cdot x - \Delta x \cdot y + \Delta x \cdot B$$

Equation of a line: Implicit Form

$$F(x, y) = \underbrace{\Delta y}_{a} \cdot x + \underbrace{(-\Delta x)}_{b} \cdot y + \underbrace{\Delta x \cdot B}_{c}$$

Midpoint Algorithm: Idea



Decision variable update

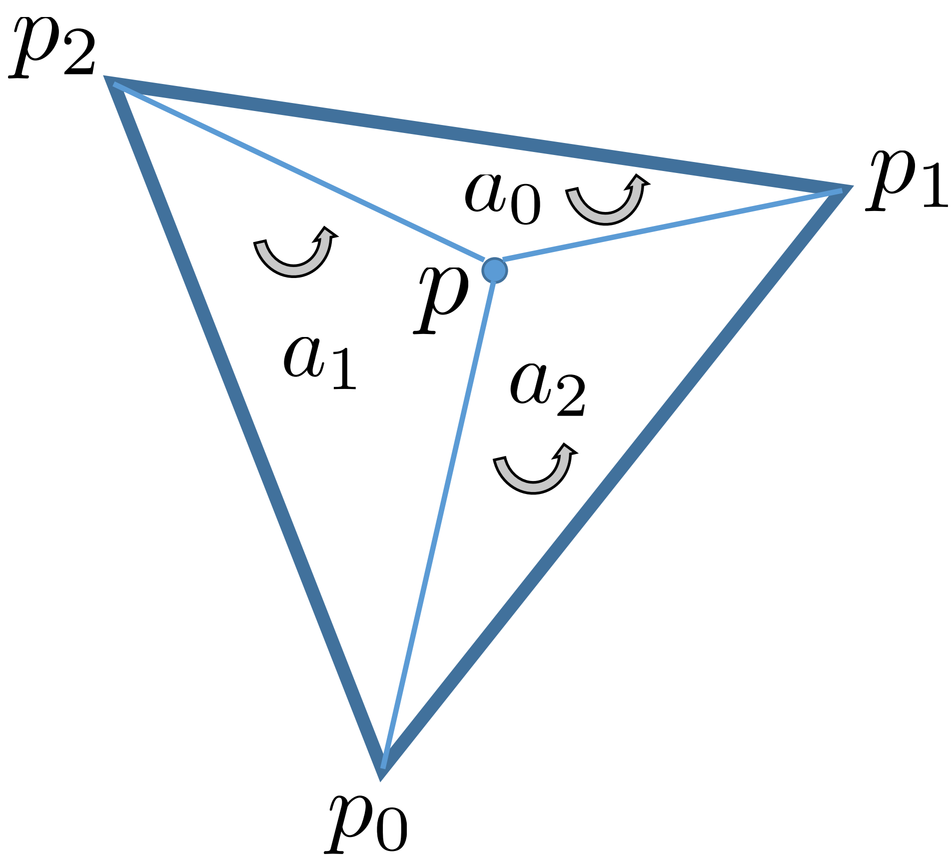
When NE was chosen

$$d = F\left(x_0 + 2, y_0 + \frac{3}{2}\right)$$
$$= a \cdot (x_0 + 2) + b \cdot \left(y_0 + \frac{3}{2}\right) + c$$
$$= d_{old} + \underbrace{(a + b)}_{\Delta d_{NE}}$$

If E were chosen instead

$$d = d_{old} + \underbrace{a}_{\Delta d_E}$$

Barycentric Coordinates



$$A = Area(p_0, p_1, p_2)$$

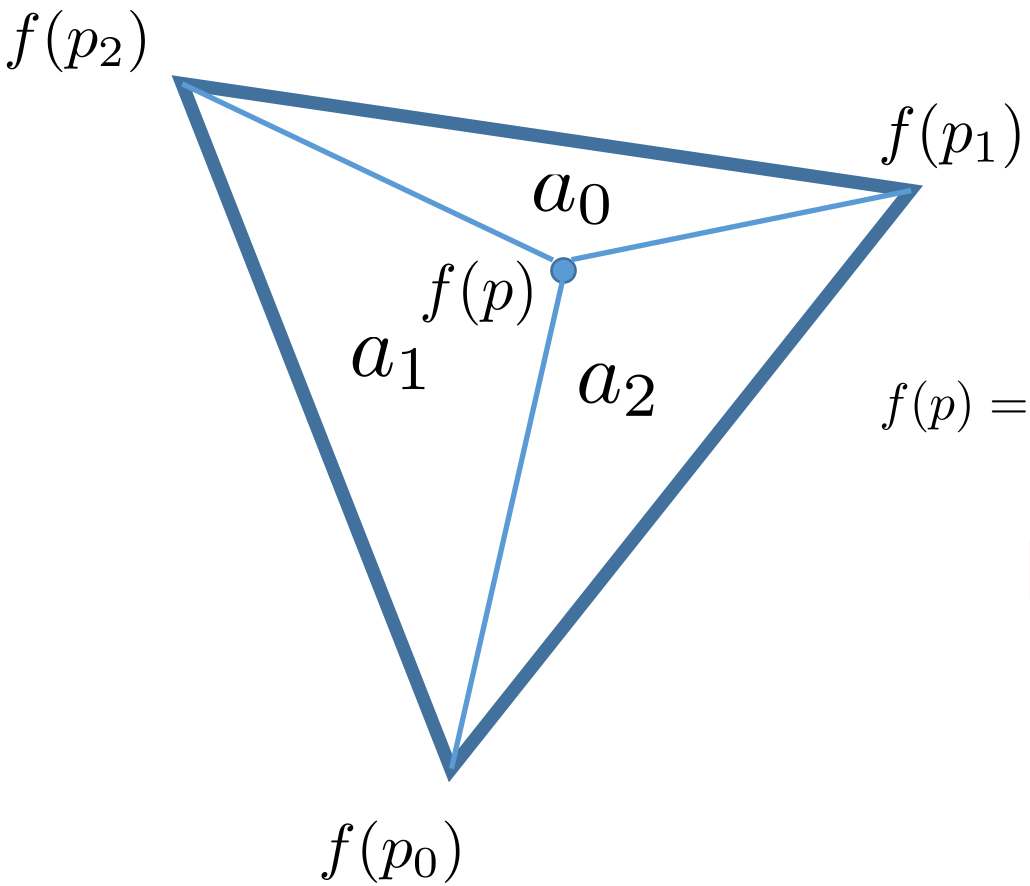
$$a_0 = \frac{Area(p, p_1, p_2)}{A}$$

$$a_1 = \frac{Area(p, p_2, p_0)}{A}$$

$$a_2 = \frac{Area(p, p_1, p_2)}{A}$$

$$a_0 + a_1 + a_2 = 1$$

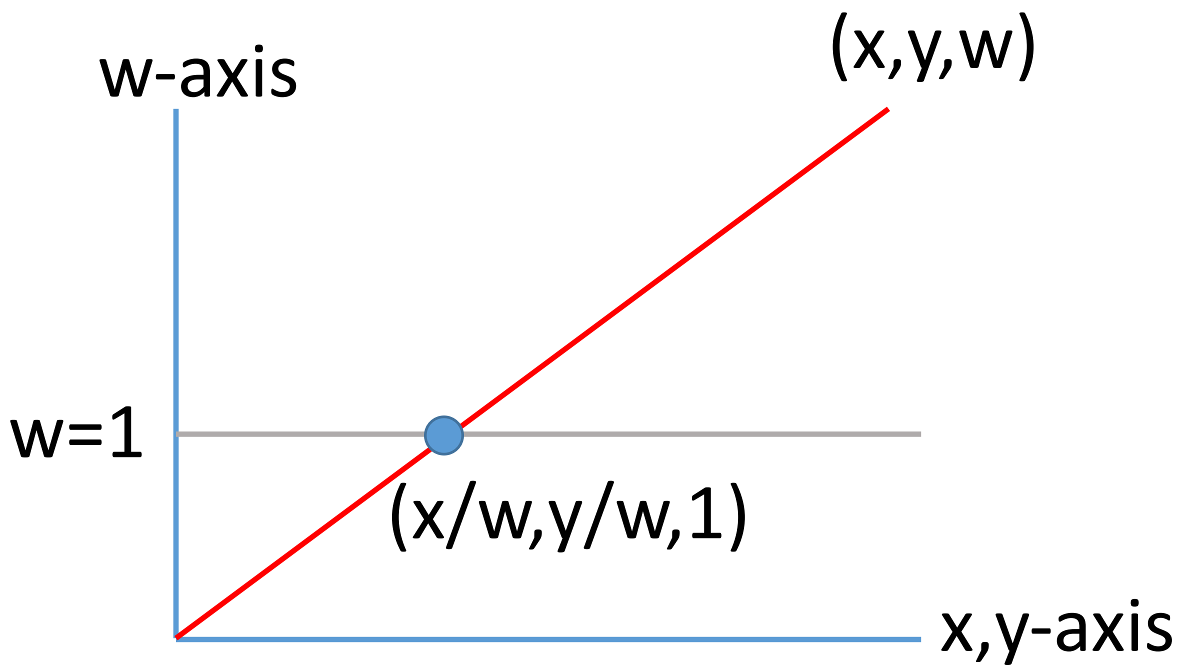
Barycentric Interpolation



$$f(p) = a_0 f(p_0) + a_1 f(p_1) + a_2 f(p_2)$$

Linear

Homogenous Coordinates



$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \equiv \begin{bmatrix} x/w \\ y/w \\ z/w \\ 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} x/w \\ y/w \\ z/w \end{bmatrix}$$

Homogenous Coordinates

$$\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} ?$$

Represents a Vector!

(Homogenous Coordinates express both Vectors and Points)

Back to Translation

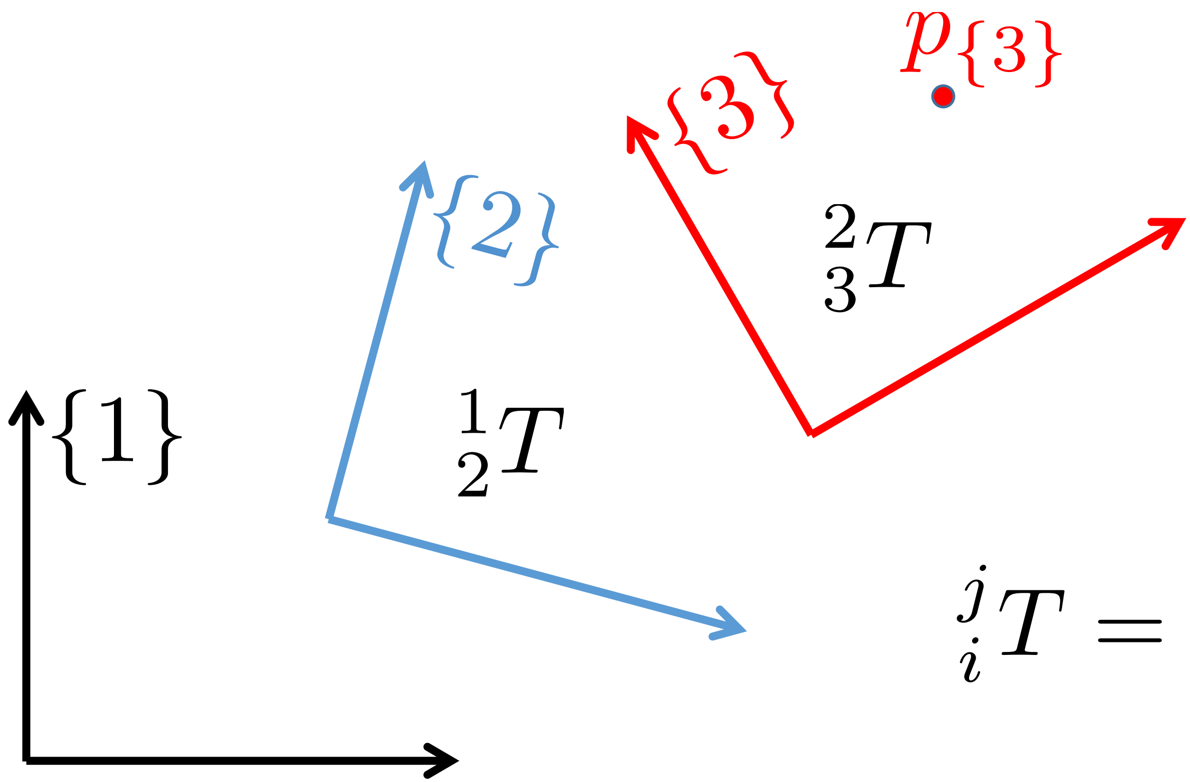
$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} A & B & C & bx \\ D & E & F & by \\ G & H & I & bz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Homogenous Coordinate

Homogenous Coordinate

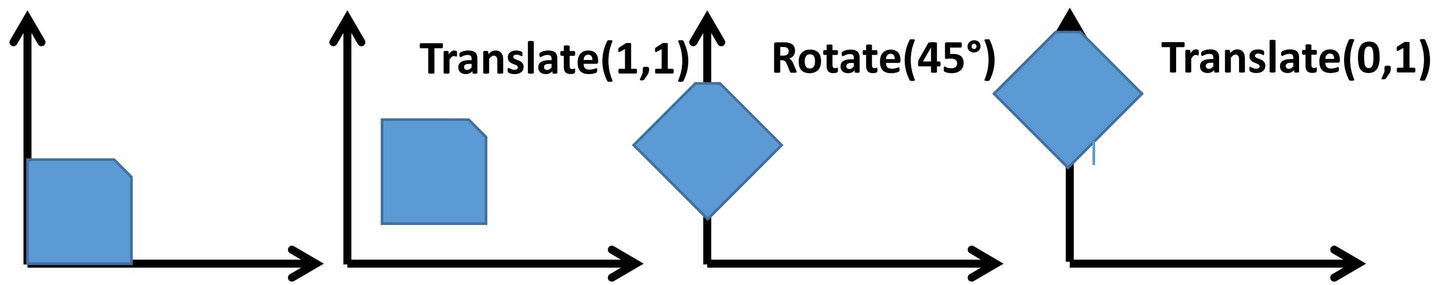
Coordinate System: Hierarchy



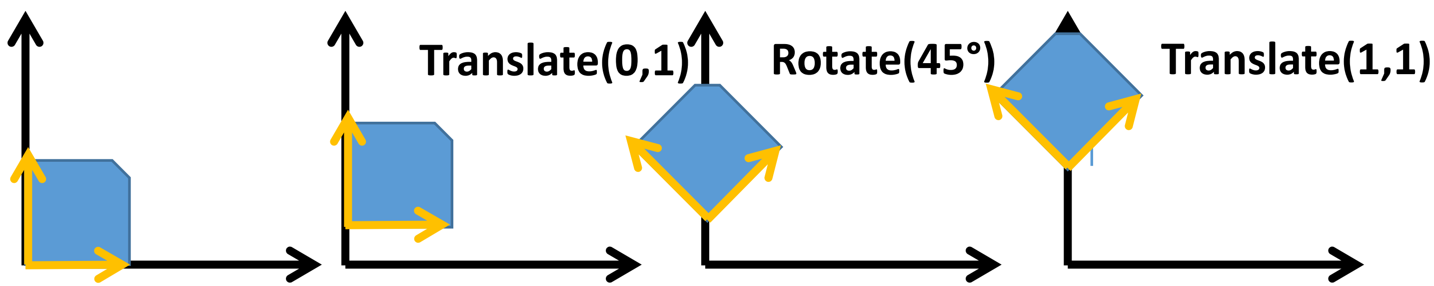
$$ {}^{j}_{i}T = {}^{i}_{j}T^{-1} $$

$$ p_{\{1\}} = {}^{1}_{2}T \; {}^{2}_{3}T \; p_{\{3\}} $$

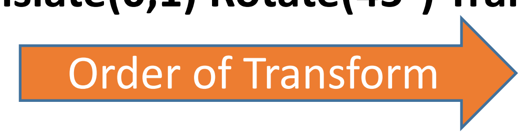
Transformation Interpretation



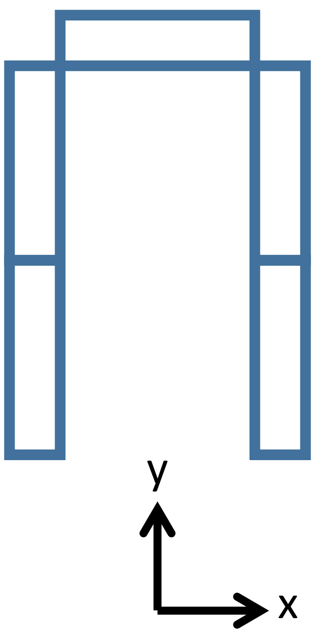
Combined = Translate(0,1) Rotate(45°) Translate(1,1)

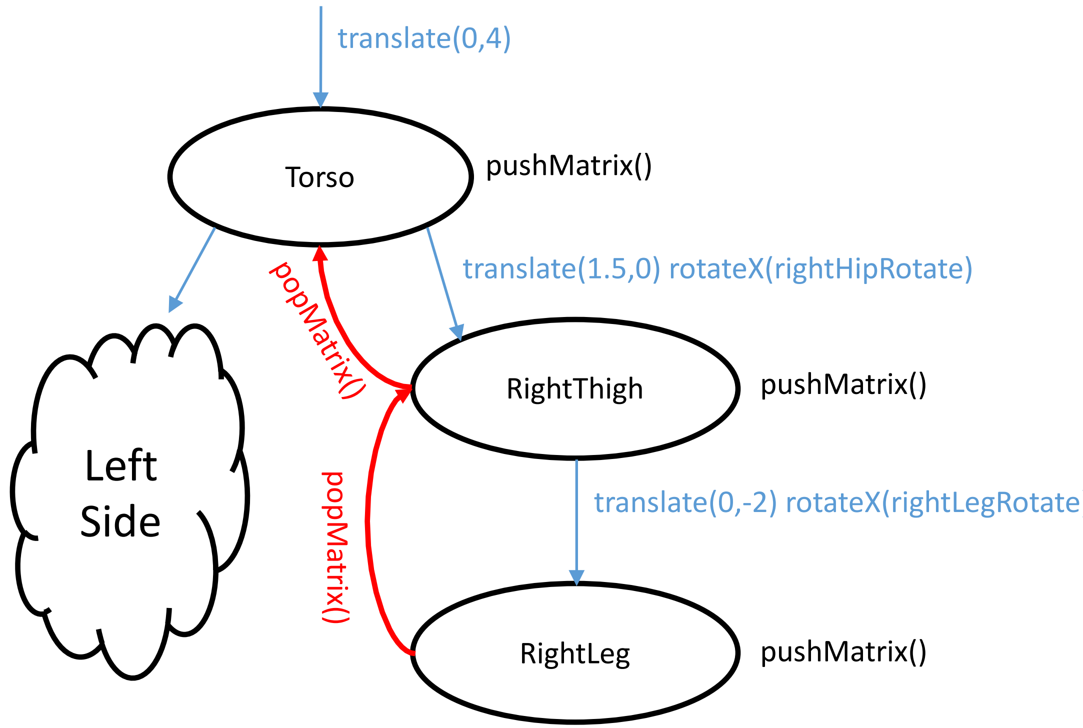


Combined = Translate(0,1) Rotate(45°) Translate(1,1)

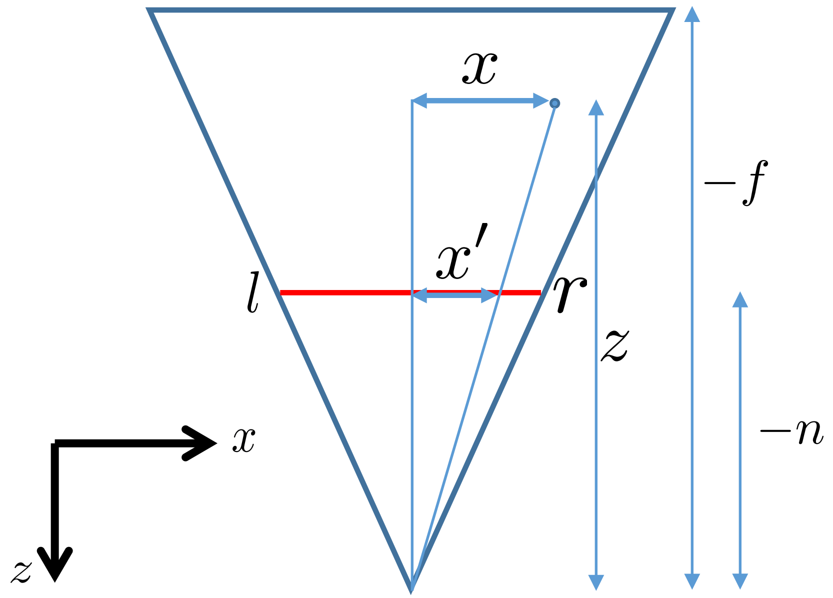


Hierarchical Modeling





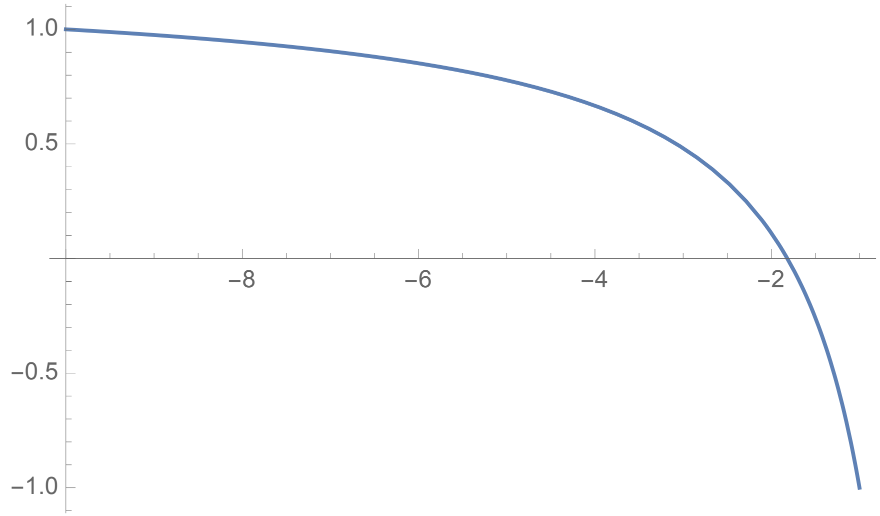
OpenGL Projection Matrix



$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{w} \end{bmatrix} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{f+n}{n-f} & \frac{2fn}{n-f} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

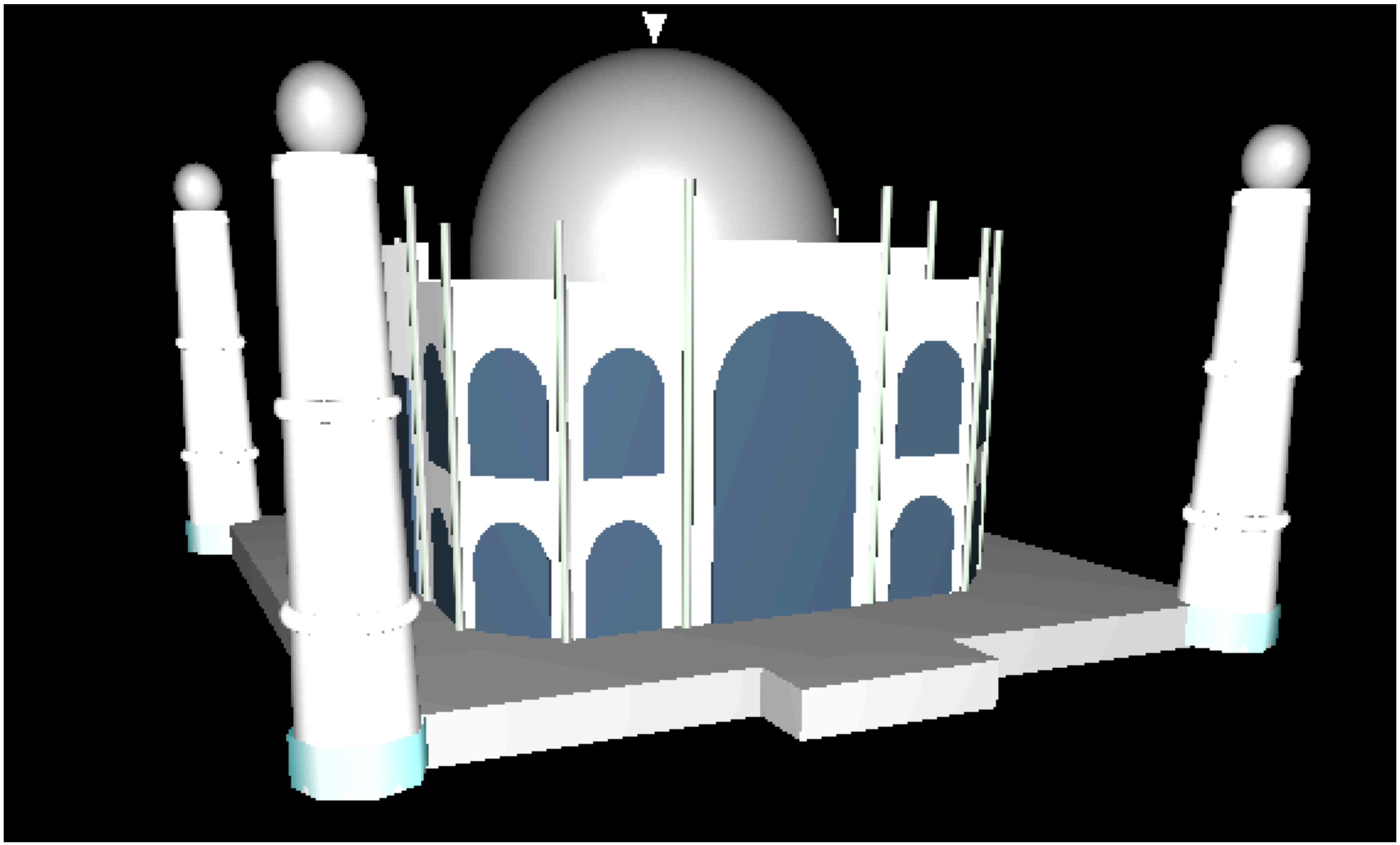
Non-Linearity in Z

$$z' = -\frac{f+n}{n-f} - \left(\frac{2fn}{n-f}\right)\frac{1}{z}$$



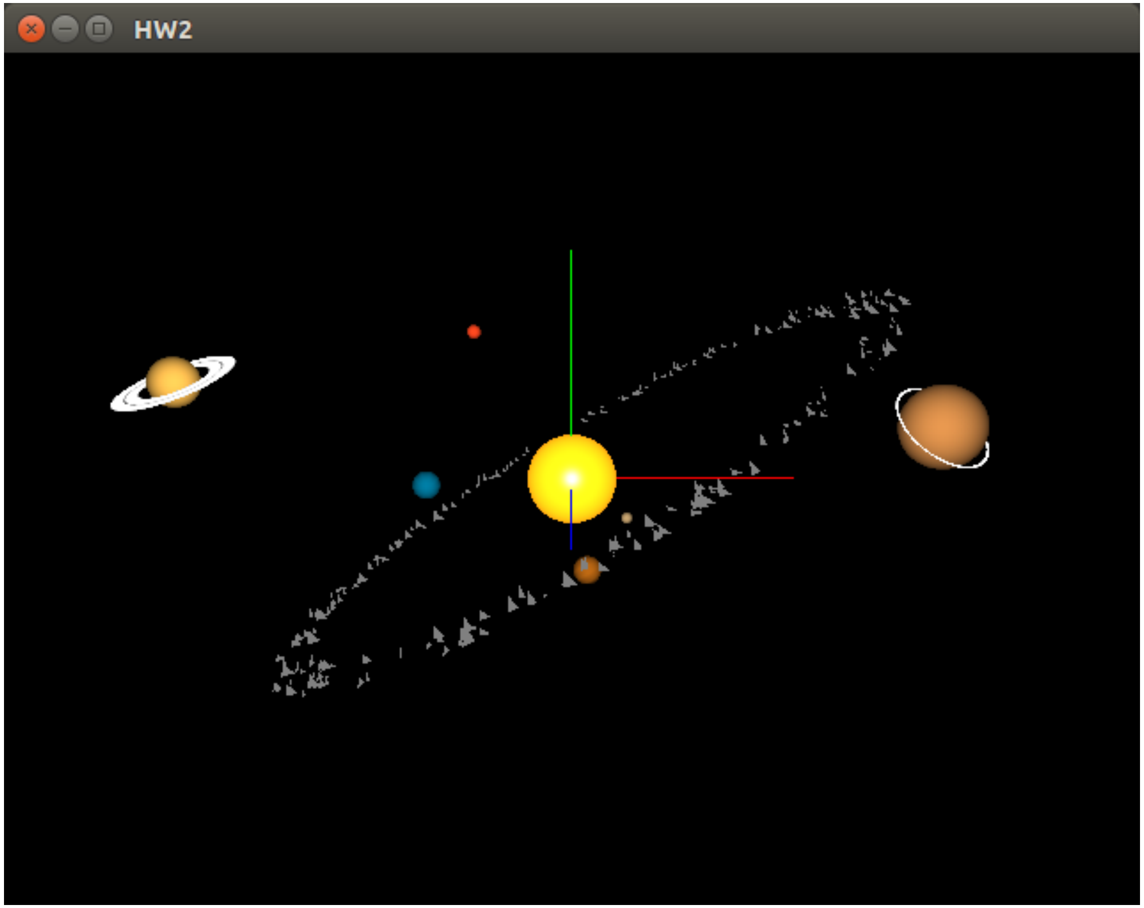
1. **Monotonic: values keeps increasing as z goes in –ve direction**
2. **Resolution decreases as z decrease (along –ve) or depth increases**

Best Extra Credit from HW2



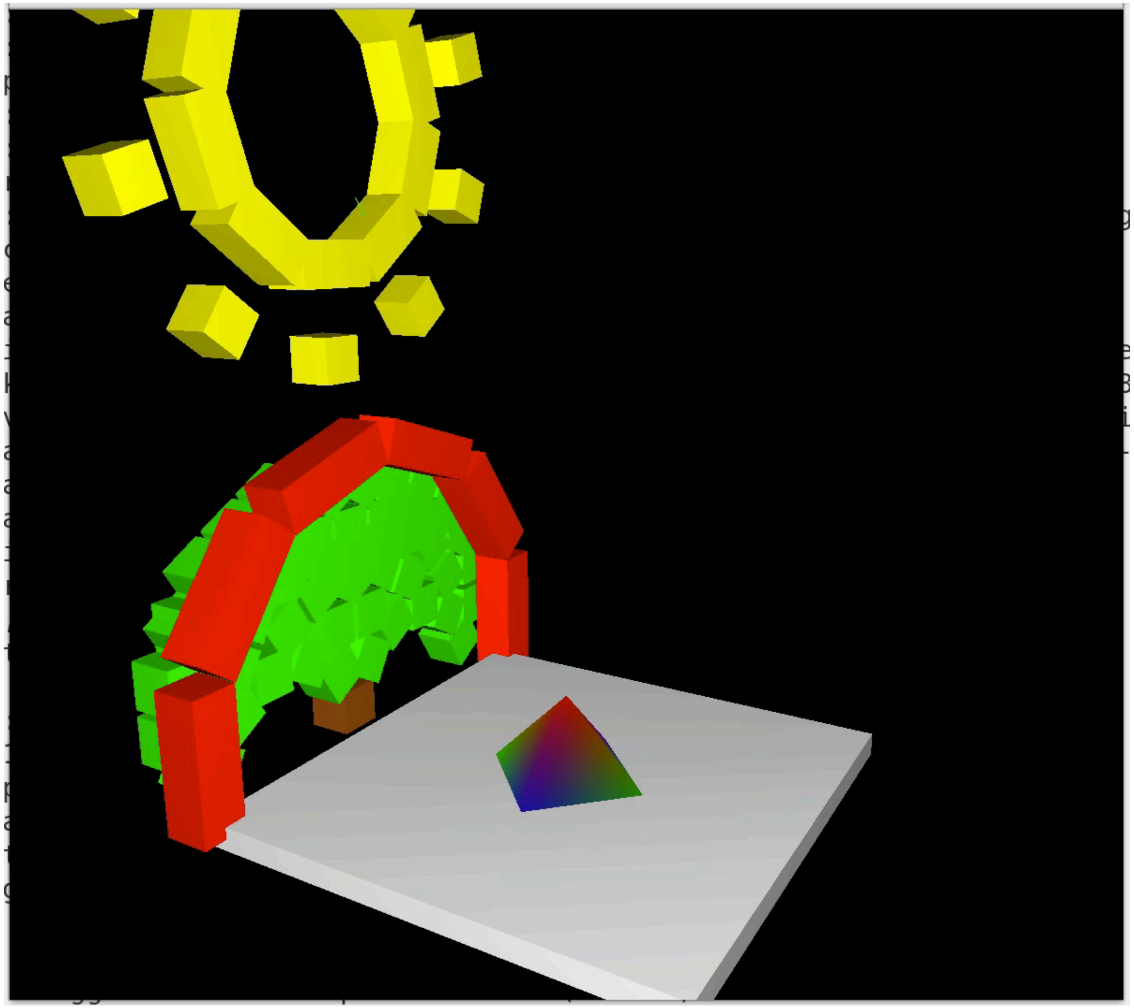
Taj Mahal – Sagar Chordia

Best Extra Credit from HW2

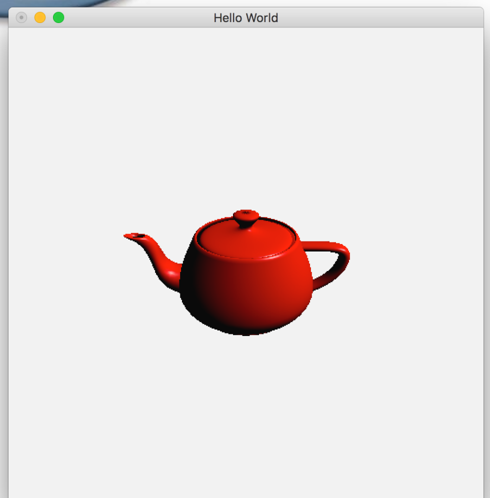


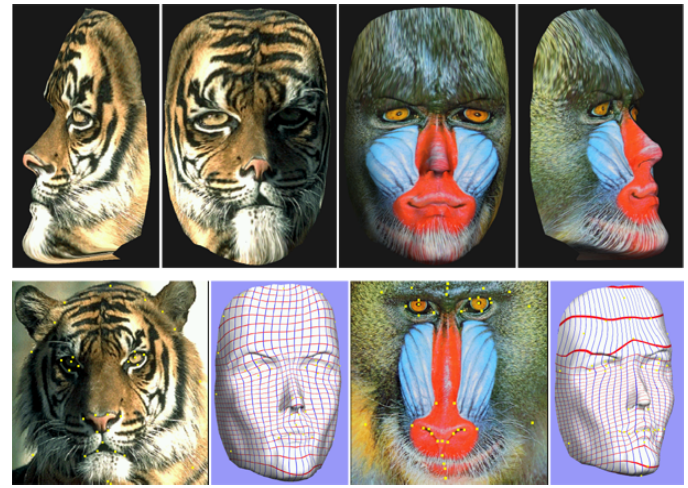
Solar System – Alexandar Schaub

Best Extra Credit from HW2



Sun and the Tree - Eugene Low

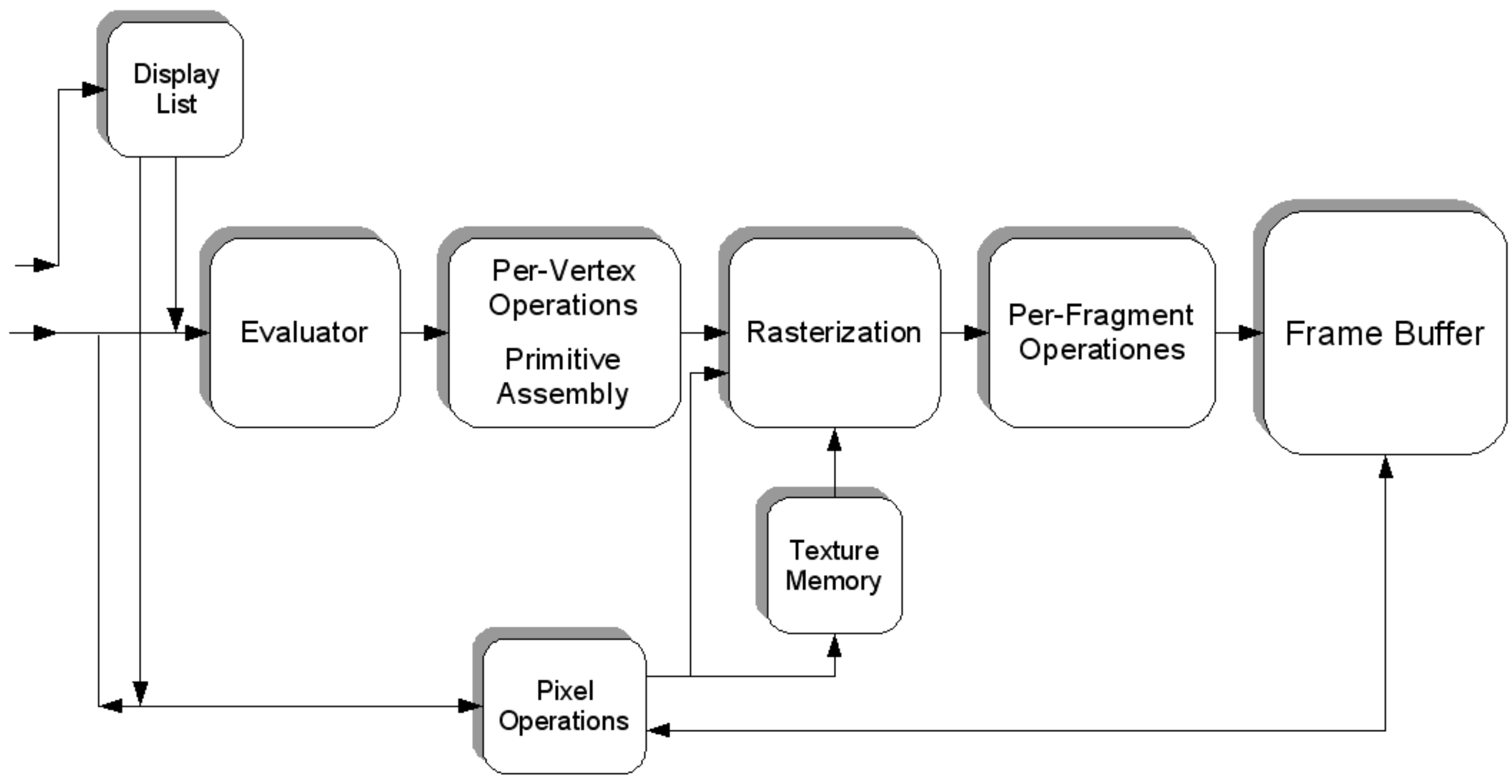




Week 3

OpenGL 1.x and Textures

OpenGL 1.x Pipeline (Simplified)



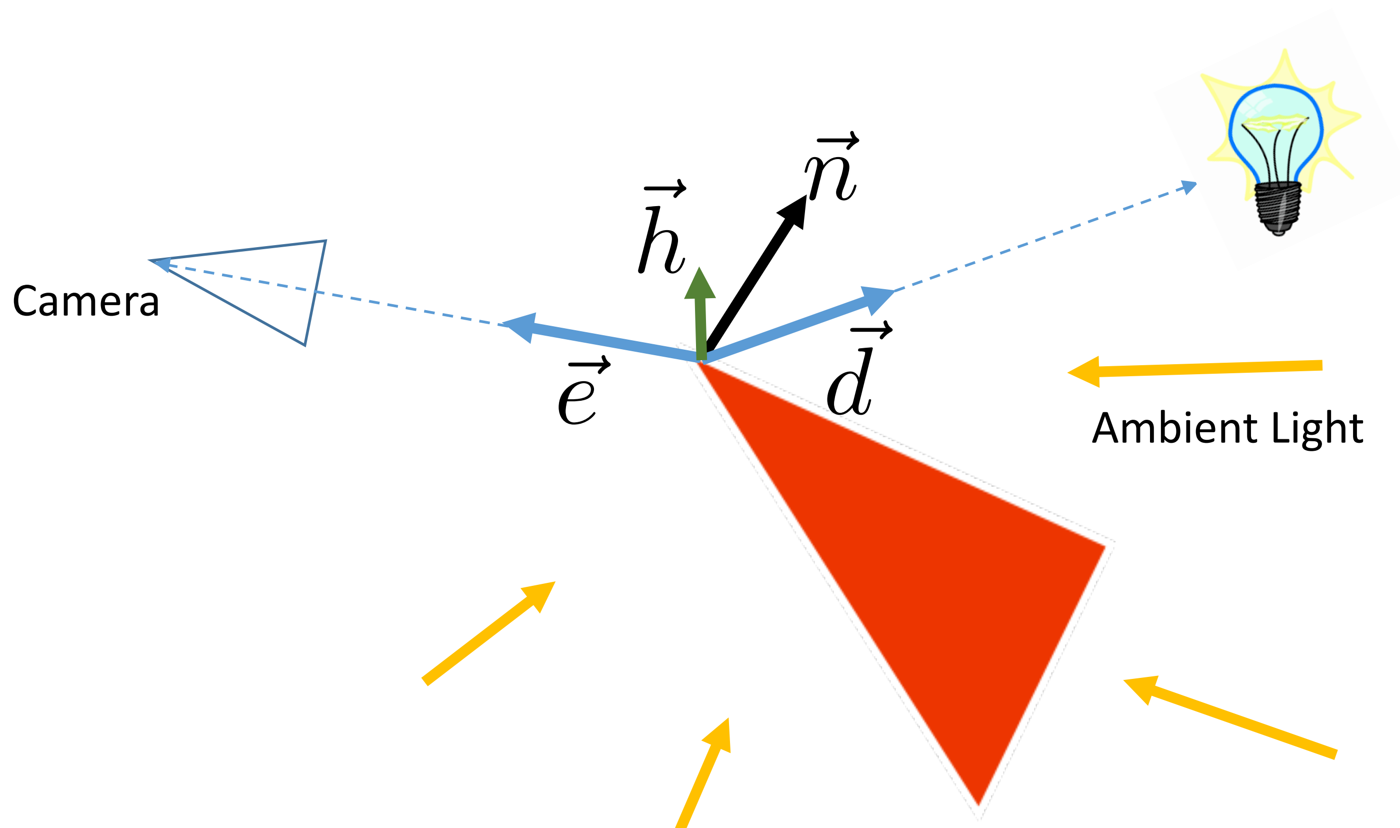
http://upload.wikimedia.org/wikipedia/commons/b/bb/Pipeline_OpenGL_%28en%29.png

OpenGL State Machine

Set State	Get State
`glColor3f(...)`	`glGetFloatv(...)`
`glEnable(...)`	`glIsEnabled(...)`
`glLineStipple(...)`	`glGetLineStipple(...)`

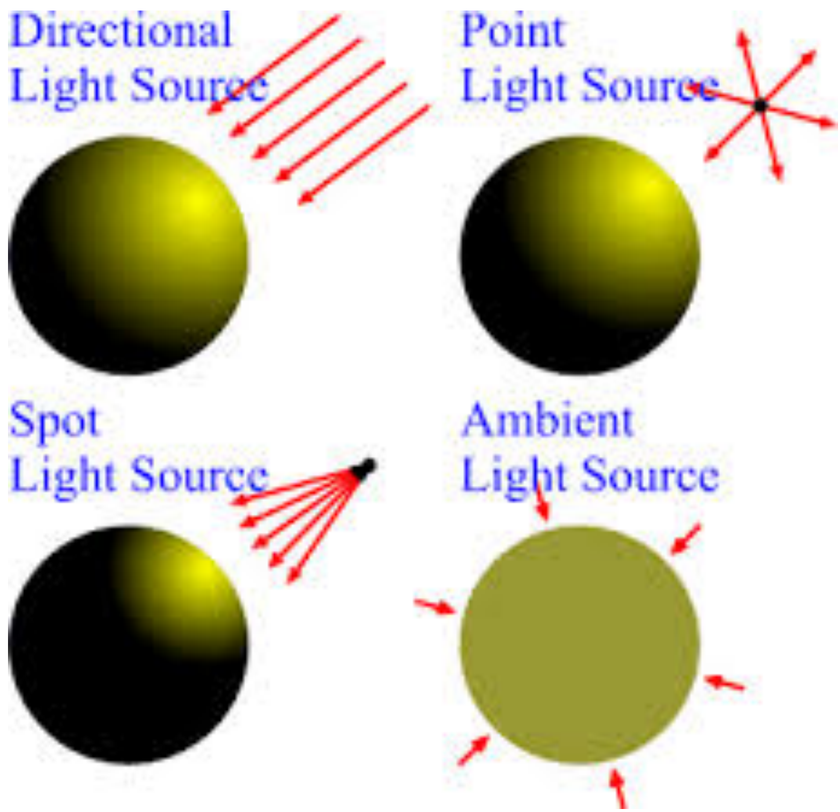
Efficiently managing state changes is a major implementation challenge

Vertex Lighting



$$
\begin{aligned}
\text{Color} = &(\text{DiffuseFactor} \cdot \text{DiffuseColor}) + \\
&\boxed{([\text{DiffuseFactor} > 0]} \cdot \text{SpecularFactor} \cdot \text{SpecularColor}) + \\
&\text{AmbientColor}
\end{aligned}
$$

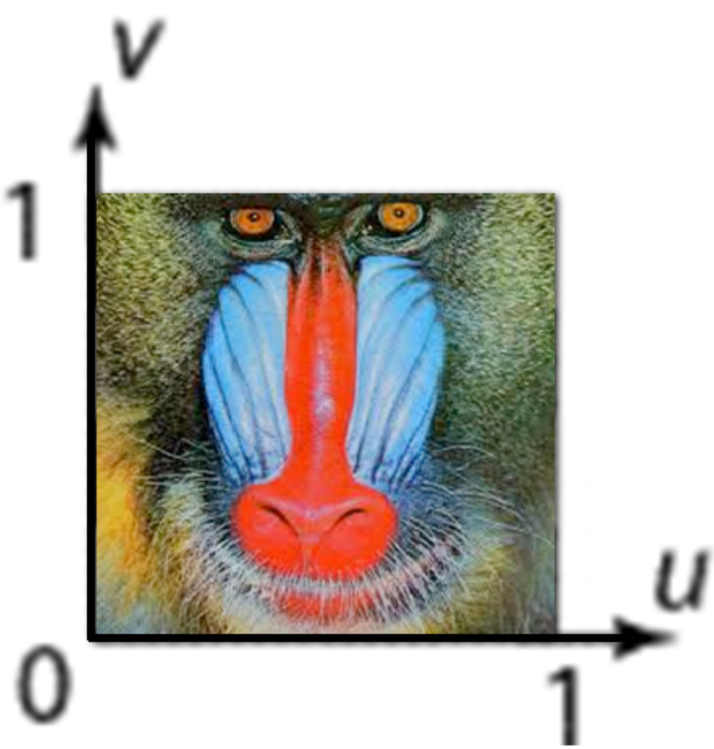
Vertex Lighting: Types

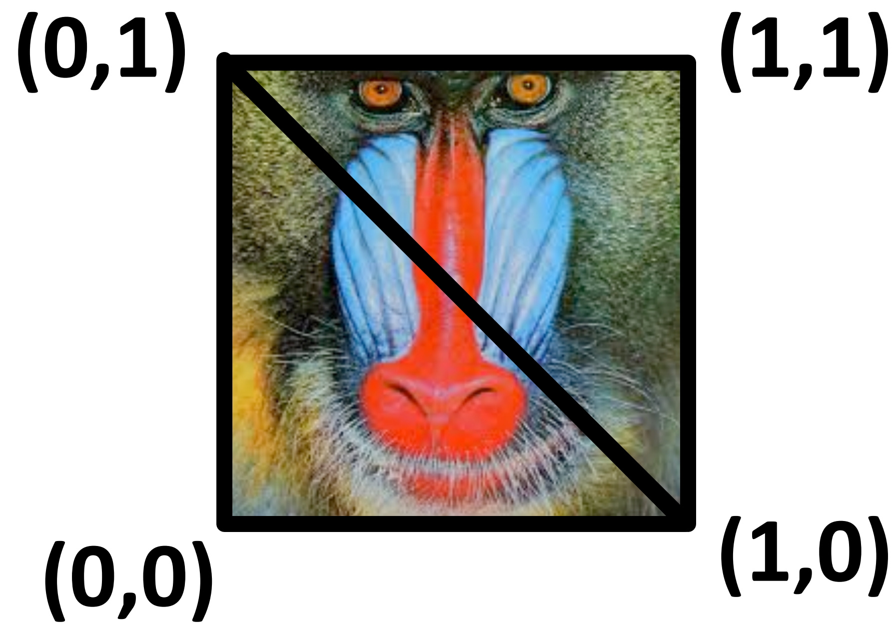


http://www.computing.northampton.ac.uk/~gary/csy3019/images3d/lightSources.gif

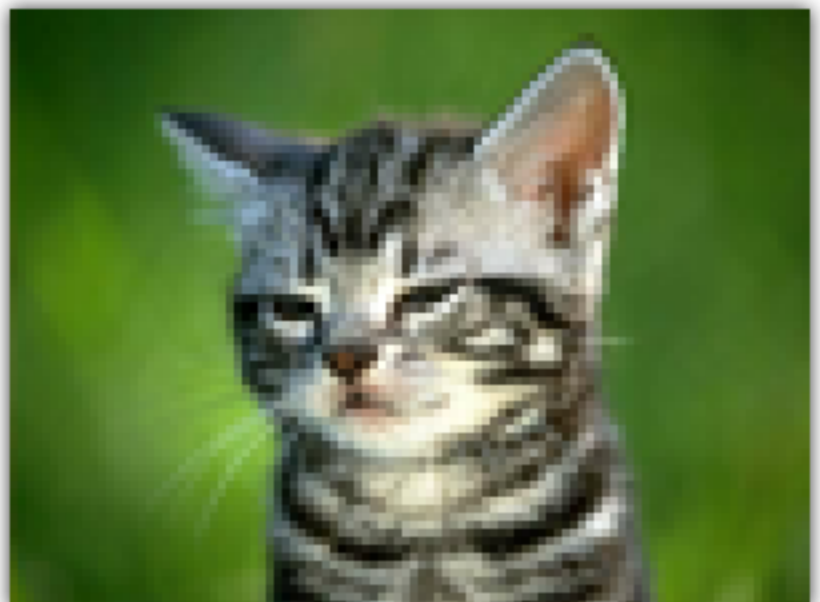
Texture Correspondence

- A texture map is defied in its own 2D coordinate system, parameterized by (u, v)
- Establish a correspondence by assigning (u, v) coordinates to triangle vertices





Nearest Neighbor Vs Bilinear



GL_NEAREST

GL_LINEAR

Perspective Correct Interpolation

Interpolation of the same two points in screen space (after projection)

$$P_x^s(s) = (1-s)\frac{dx_1}{z_1} + s\frac{dx_2}{z_2}$$

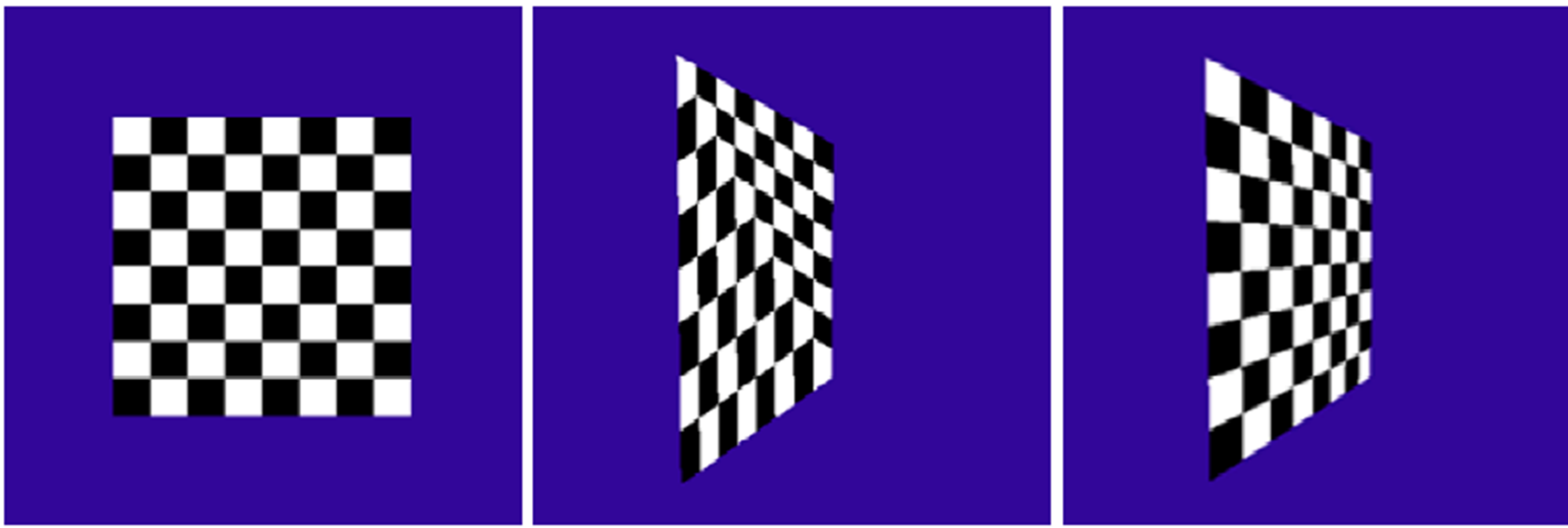
Screen space point and world-space point after projection must match

$$d\frac{(1-t)x_1 + tx_2}{(1-t)z_1 + tz_2} = (1-s)\frac{dx_1}{z_1} + s\frac{dx_2}{z_2}$$

After algebra

$$t = \frac{sz_1}{z_2 + s(z_1 - z_2)}$$

Perspective Correct Interpolation

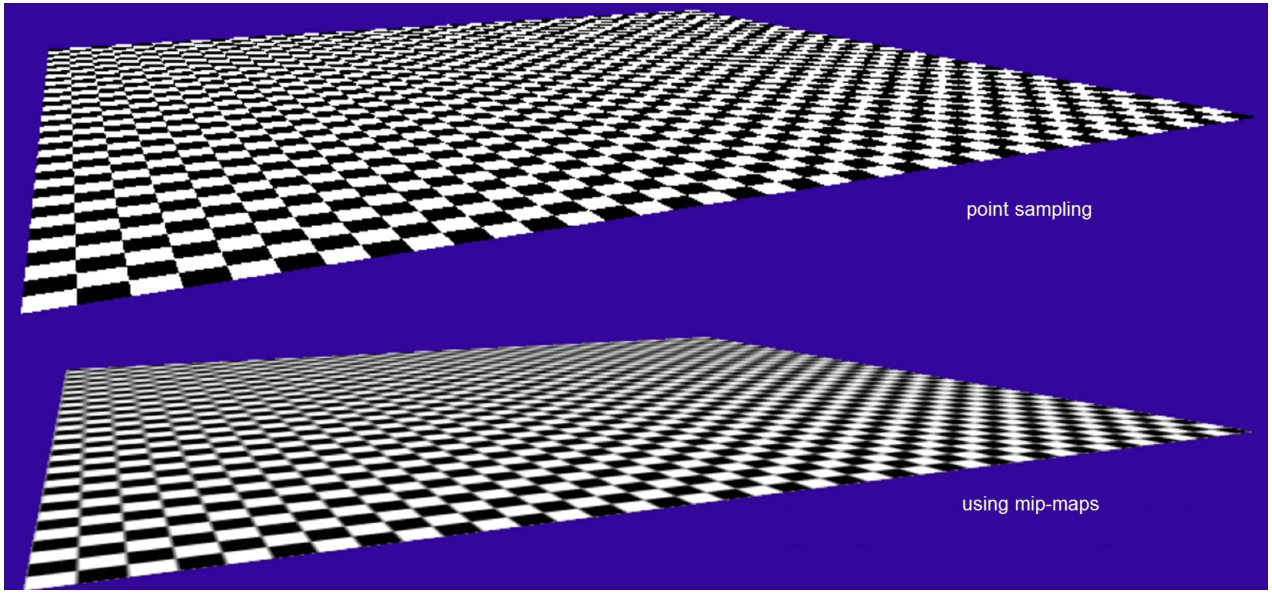


texture source

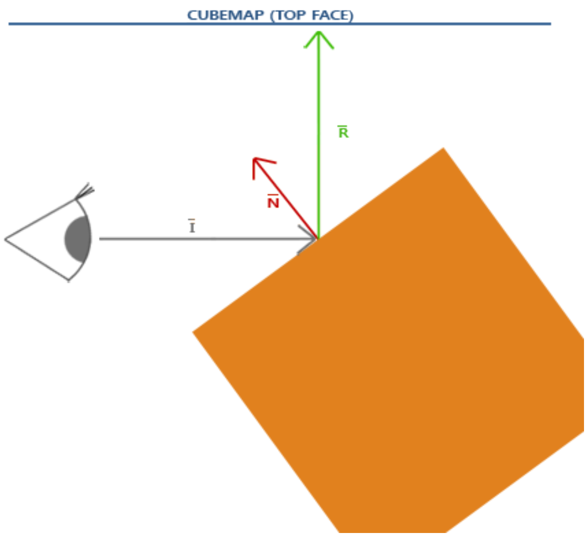
results without
perspective correct interpolation

results with
perspective correct interpolation

Mipmaps



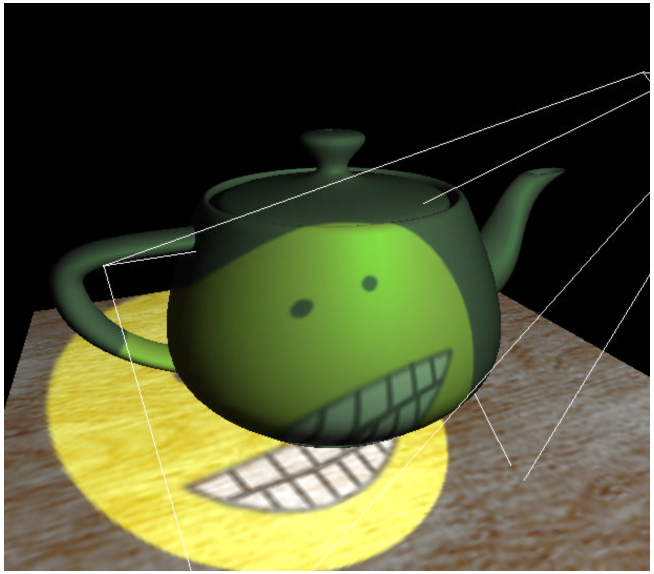
Cube Mapping

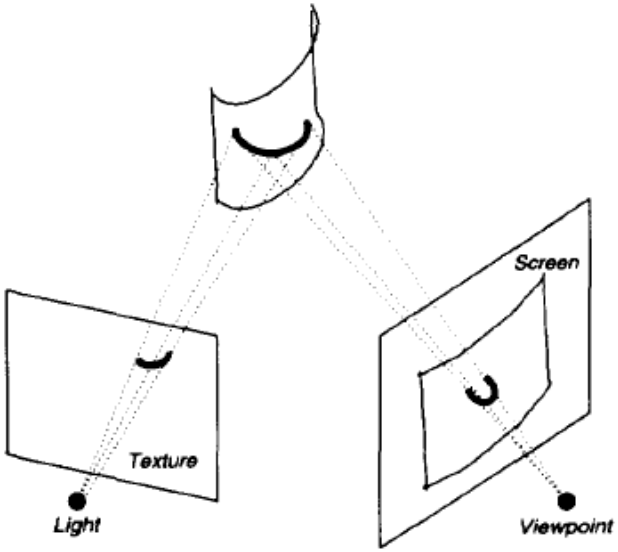



```
glTexGenfv(GL_S, GL_TEXTURE_GEN_MODE, GL_REFLECTION_MAP_EXT);
glTexGenfv(GL_T, GL_TEXTURE_GEN_MODE, GL_REFLECTION_MAP_EXT);
glTexGenfv(GL_R, GL_TEXTURE_GEN_MODE, GL_REFLECTION_MAP_EXT);
glEnable(GL_TEXTURE_GEN_S);
glEnable(GL_TEXTURE_GEN_T);
glEnable(GL_TEXTURE_GEN_R);
```

http://learnopengl.com/#!Advanced-OpenGL/Cubemaps

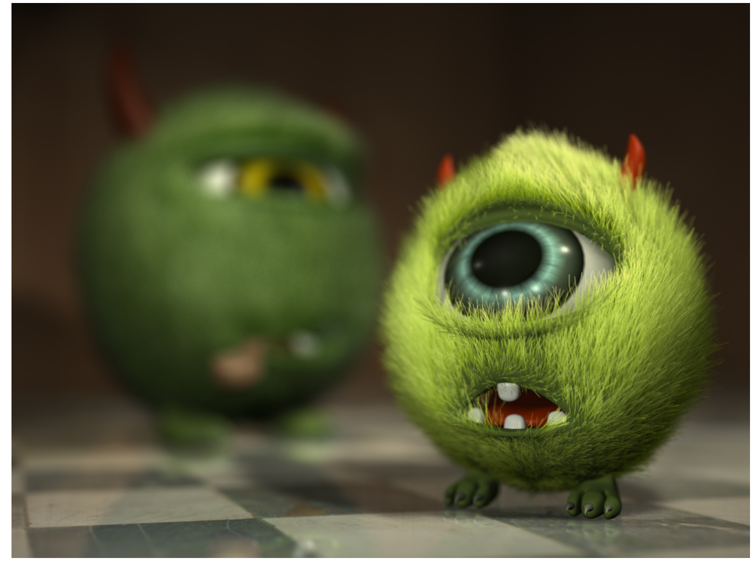
Projective Texturing





- Treat light Source as a
- Render the scene normally from the actual camera

http://www.nvidia.com/object/Projective_Texture_Mapping.html
Segal et. al. SIGGRAPH'92

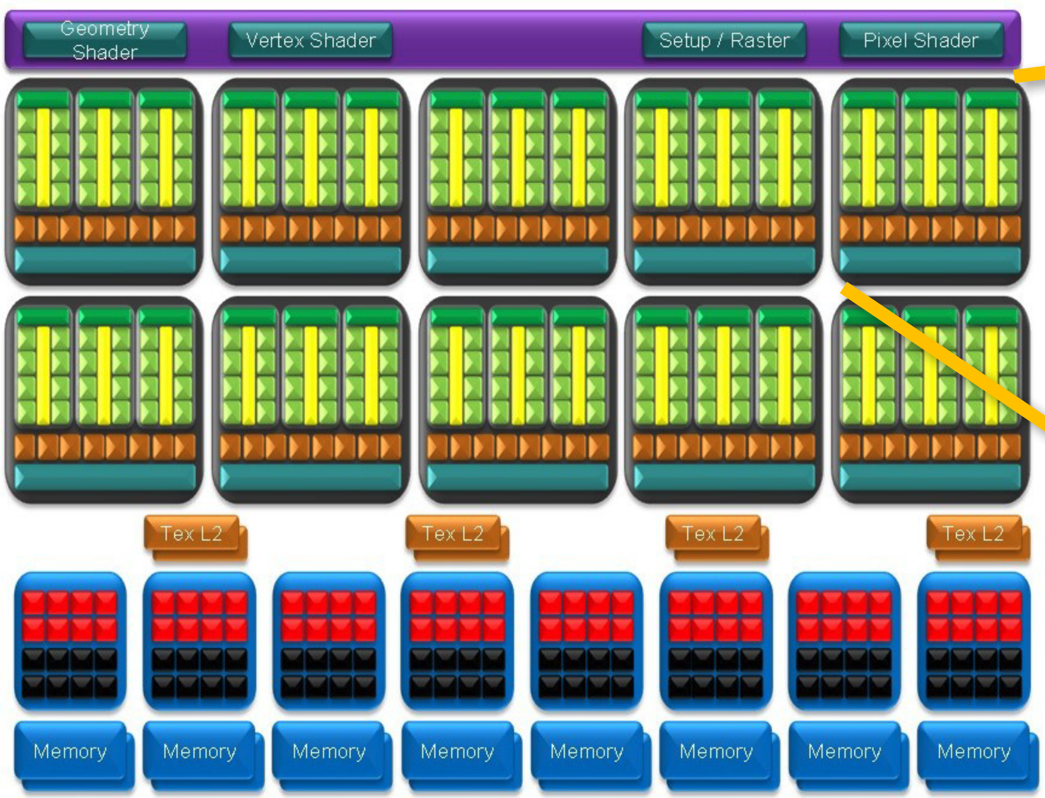


Week 4

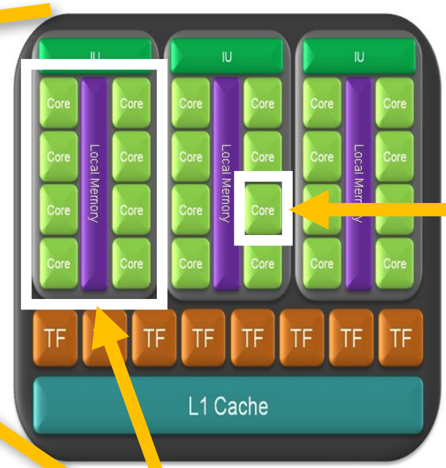
GPU, Shaders, OpenGL 3.0 and Rendering

Modern GPUs

GeForce GTX 280 Graphics Processing Architecture



TPC: Thread Processing Cluster



SM: Streaming Multi-Processors

SP: Streaming Processor

Unified Architecture!

Geforce GTX 200 Technical Brief by nVidia

Key Differences: GPU vs CPU

NV GTX 1060	Intel i7-4790K	Intel Xeon E5-2699
Cores: 1280	Cores: 4 (8 Threads)	Cores: 18 (36 Threads)
Clock: 1.5 – 1.7 GHz	Clock: 4 – 4.4 GHz	Clock: 2.3 – 3.6 GHz.
Power: 400 W (120W)	Power: 88W	Power: 145W
Mem BW: 192 GB/sec	Mem BW: 25.6 GB/sec	Memory BW: 68 GB/sec

GPU	CPU
Many Simple Cores	**Few Complex Cores**
Slower Clock	**Fast Clock**
Power Hungry	**Power Efficient**
High Mem Bandwidth	**Lower Mem Bandwidth**
High-Throughput	**Low Latency**

Key Differences: GPU vs CPU

NV GTX 1060

Cores: 1280

Clock: 1.5 – 1.7 GHz

Power: 400 W (120W)

Me

GTX 1080: 2560 Cores (1.6-1.7 GHz), 320 GB/sec

Tesla P100 (Pascal): 3584 Cores (1.3 – 1.4 GHz), 720GB/sec

Slower Clock

Power Hungry

High Mem Bandwidth

High-Throughput

GPU

Intel i7-4790K

Cores: 4 (8 Threads)

Clock: 4 – 4.4 GHz

Power: 88W

Intel Xeon E5-2699

Cores: 18 (36 Threads)

Clock: 2.3 – 3.6 GHz.

Power: 145W

Fast Clock

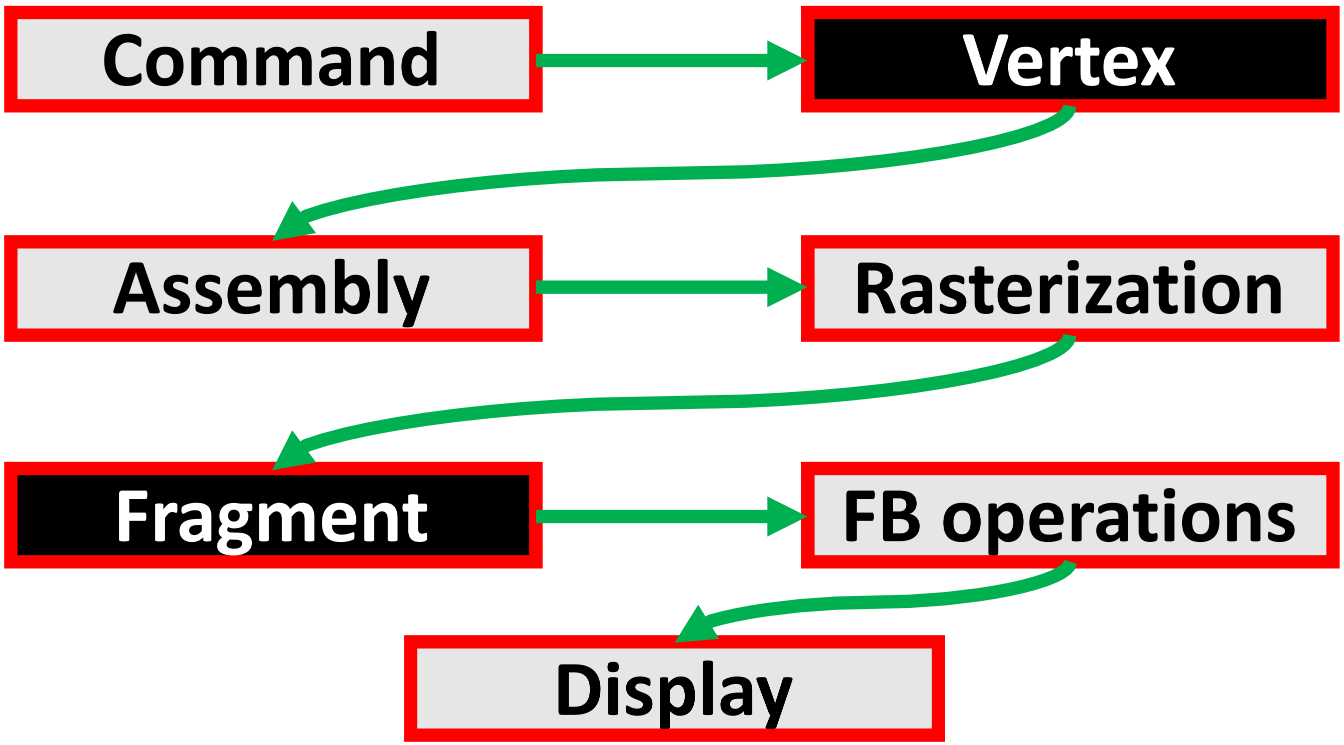
Power Efficient

Lower Mem Bandwidth

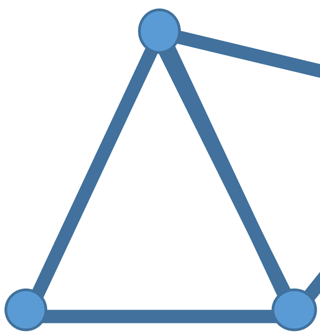
Low Latency

CPU

Simplistic Programmable Pipeline > 2.0



Vertex Program/Shader



Input: Single Vertex with attributes, e.g.

- Position
- Normal
- Texture Coordinate
- Other programmer defined attribs

Vertex program

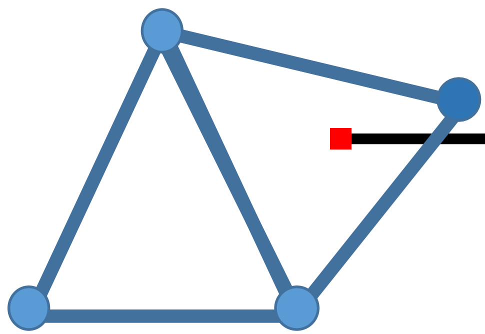
Input: Uniforms - remain constant during the entire draw call, e.g.

- Model matrix
- View/Camera matrix
- Projection matrix
- Other programmer defined parameters

Output:

- Transformed positions (NDC)
- Point size (optional)
- Other programmer defined **varying** data (input for Fragment program)

Fragment Program/Shader



Input: Single rasterized fragment with interpolated "out" attributes from the Vertex *Program*.

Fragment Program

Input: Uniforms - remain constant during the entire draw call, e.g.

- Material properties
- Textures (1D, 2D, 3D)
- Light positions
- etc - whatever else you wish

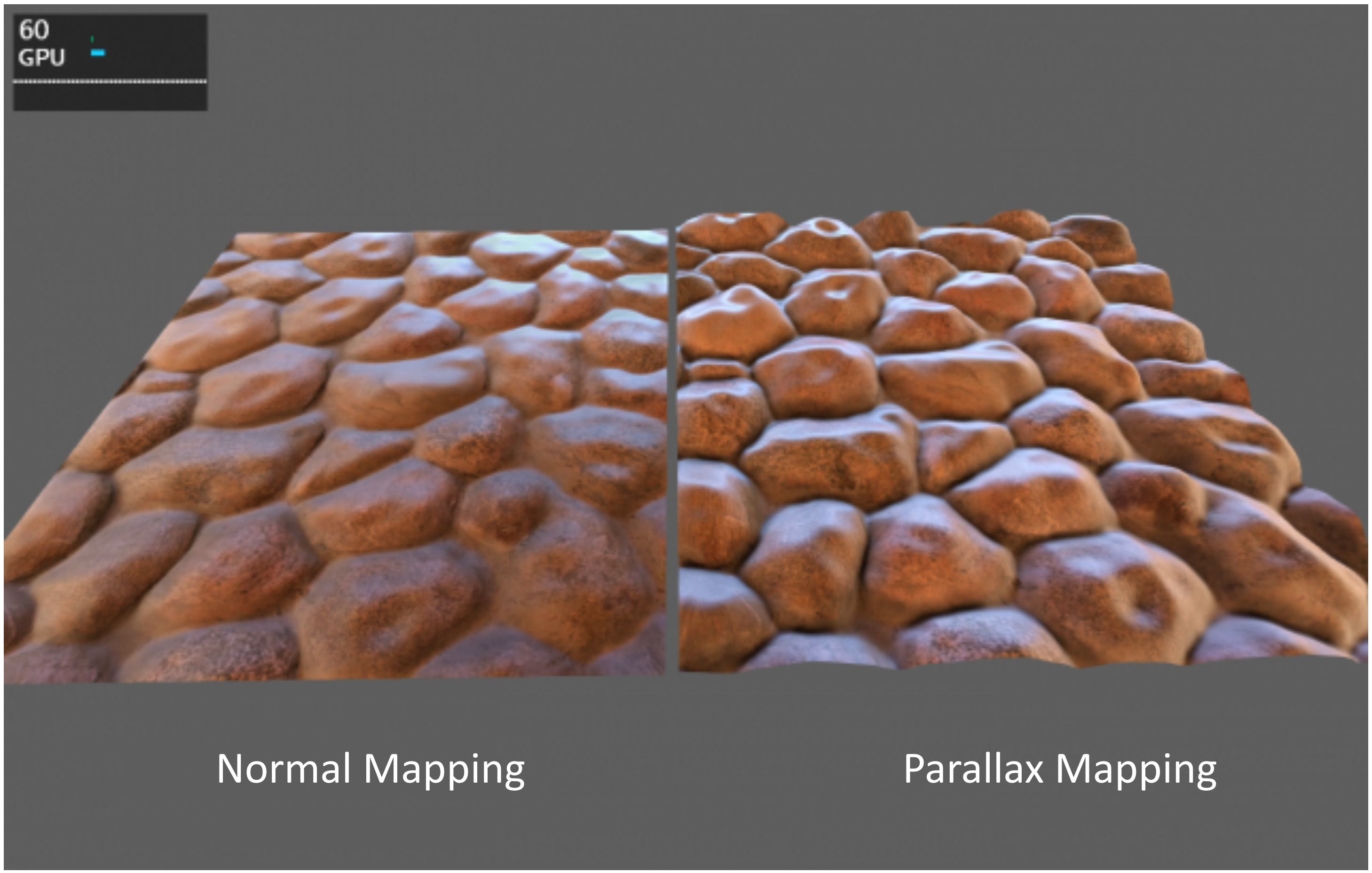
Output:

- Final color (with maybe Alpha)
- Depth value (optional)
- **Discard** – discard a fragment from further processing

Shader Considerations

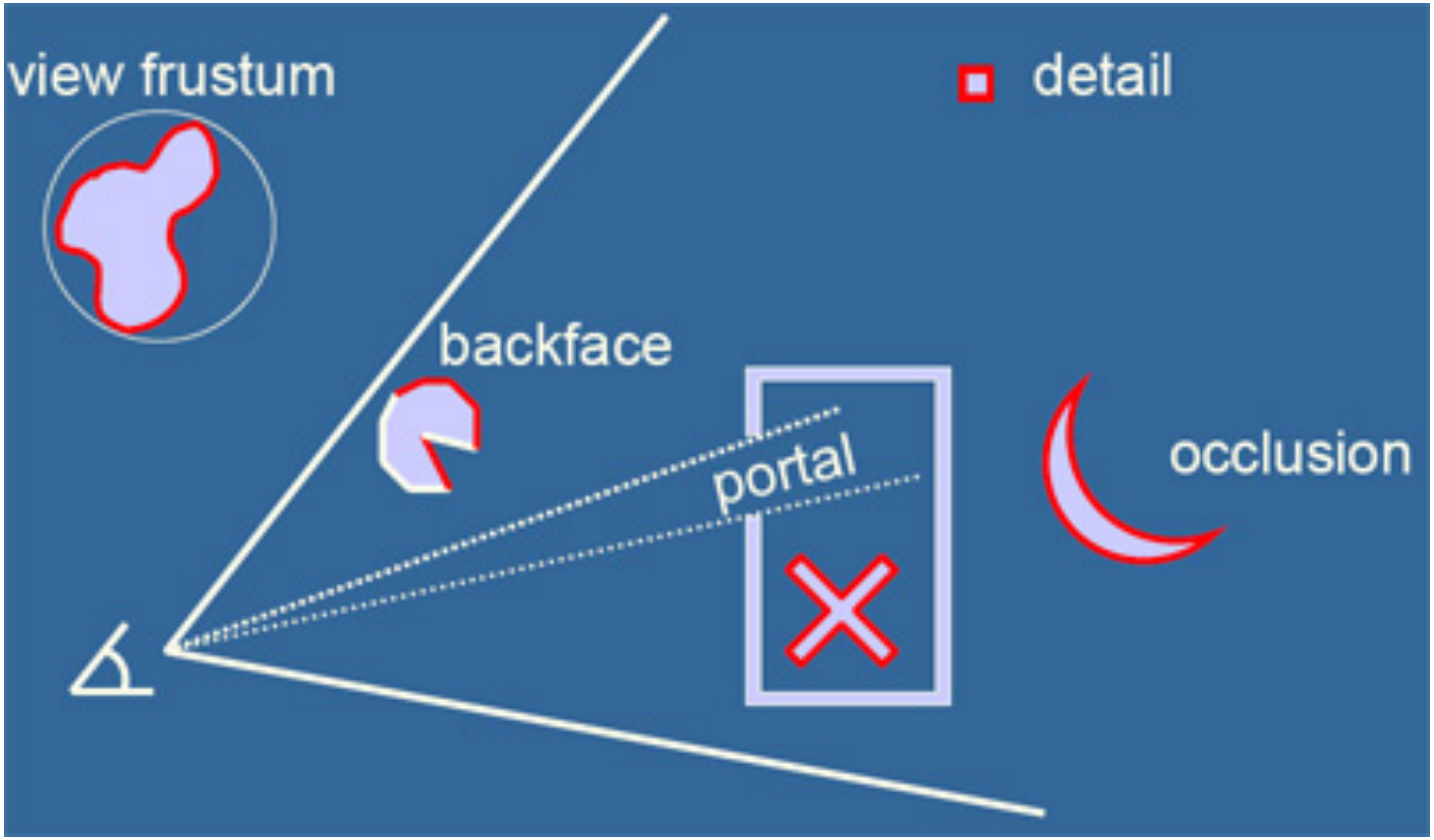
- Single-precision arithmetic
  - > GTX 200 supports Double-precision
- **Branching, loops expensive**
- No access to neighboring fragments/vertices
- Limited instruction/program size

Parallax Mapping

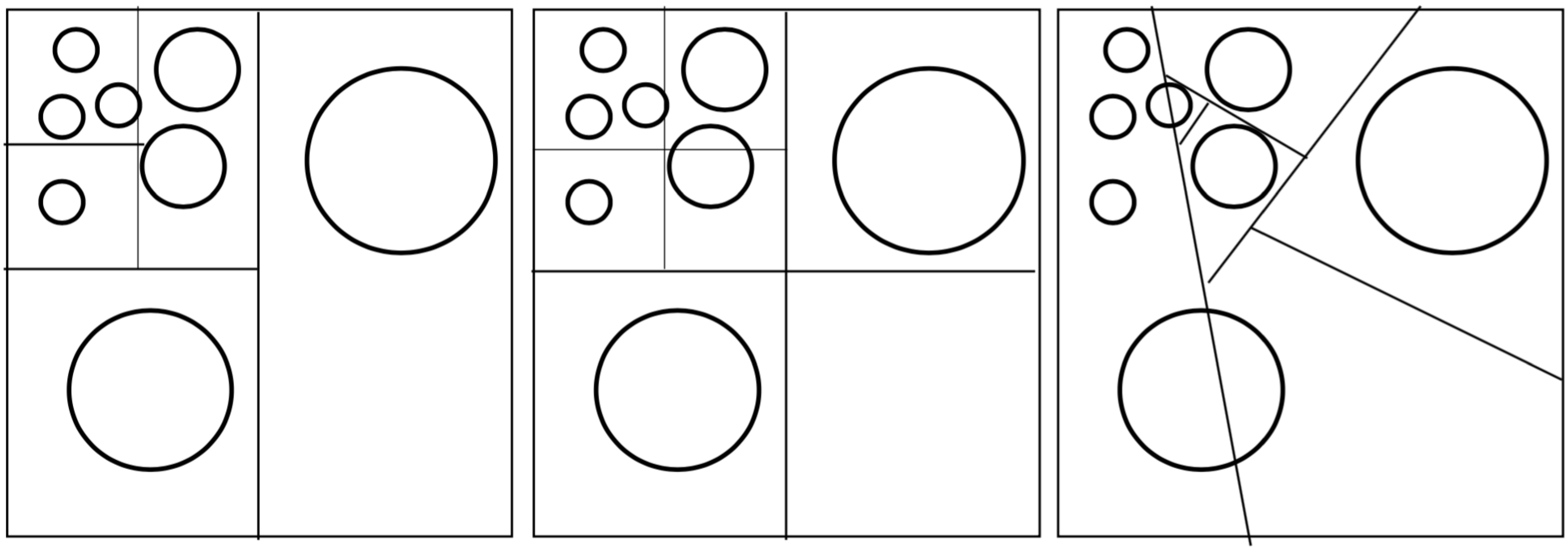


https://vvvv.org/sites/default/files/screenshot1382763194.png

Summary of Culling Techniques



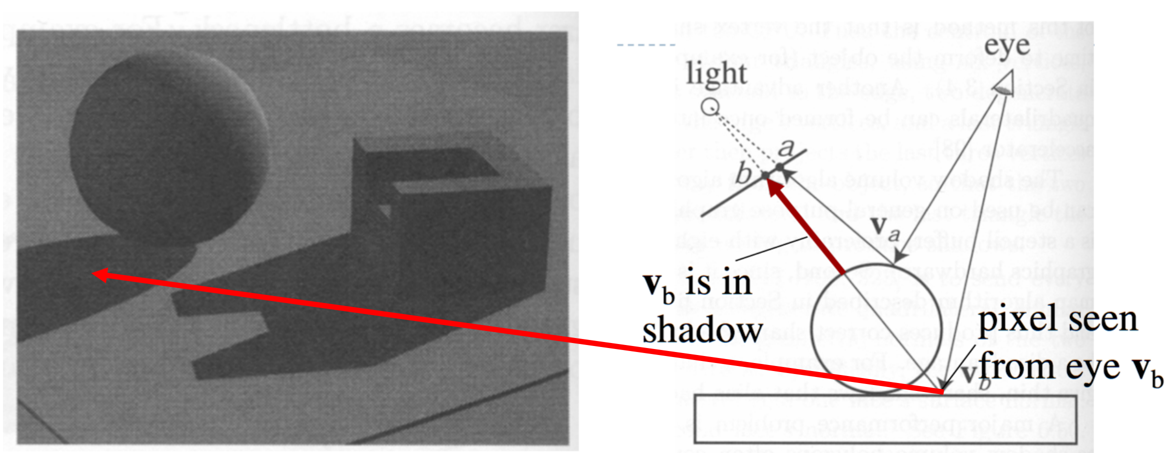
Spatial Hierarchies: Variations



kd-tree   oct-tree   bsp-tree

Shadow Mapping

- **First Pass**
  - Render the Scene from the light Source
    - Pretend the light is the "camera"
  - Store the depth buffer as a texture
    - Heightfield – tells us the "distance" of the nearest points from the light source.
- **Second Pass**
  - Project the depth buffer texture from the light's P.O.V
  - Render the scene from the camera position
  - Compare fragment's depth (projected r texture coordinate) to the depth stored in texture



Projective Texturing

- Map NDC (-1 , 1) to Texture Coordinate space (0-1)
  - Scale and add Bias

$$\begin{bmatrix} s'' \\ t'' \\ r'' \\ q'' \end{bmatrix}_{\text{TextureSpace}} = \begin{bmatrix} 0.5 & 0 & 0 & 0.5 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s' \\ t' \\ r' \\ q' \end{bmatrix}_{\text{NDC}}$$

Final texture coordinates after perspective-correct interpolation of (s'', t'', r'', q'')

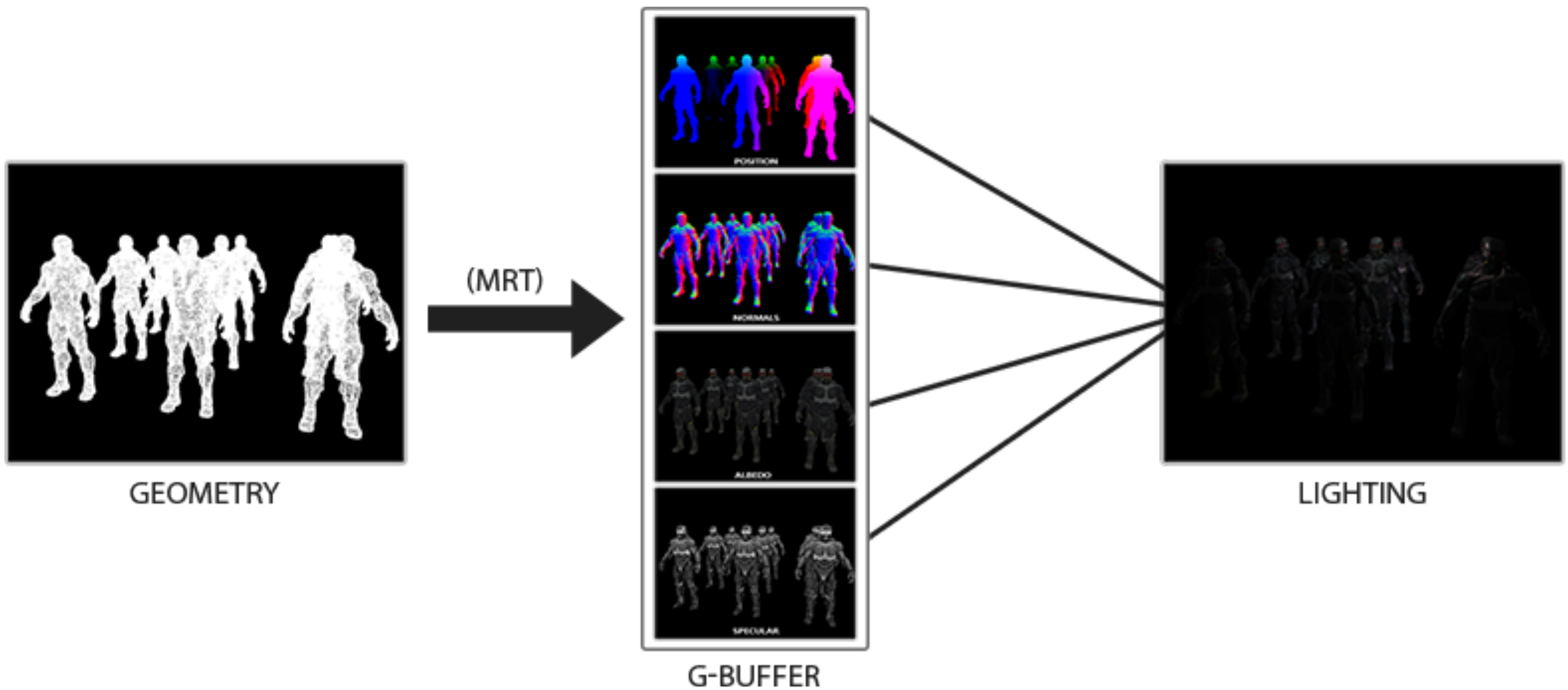
$$\left(\frac{s''}{q''}, \frac{t''}{q''}, \boxed{\frac{r''}{q''}} \right)$$

Compare this with depth

Deferred Rendering

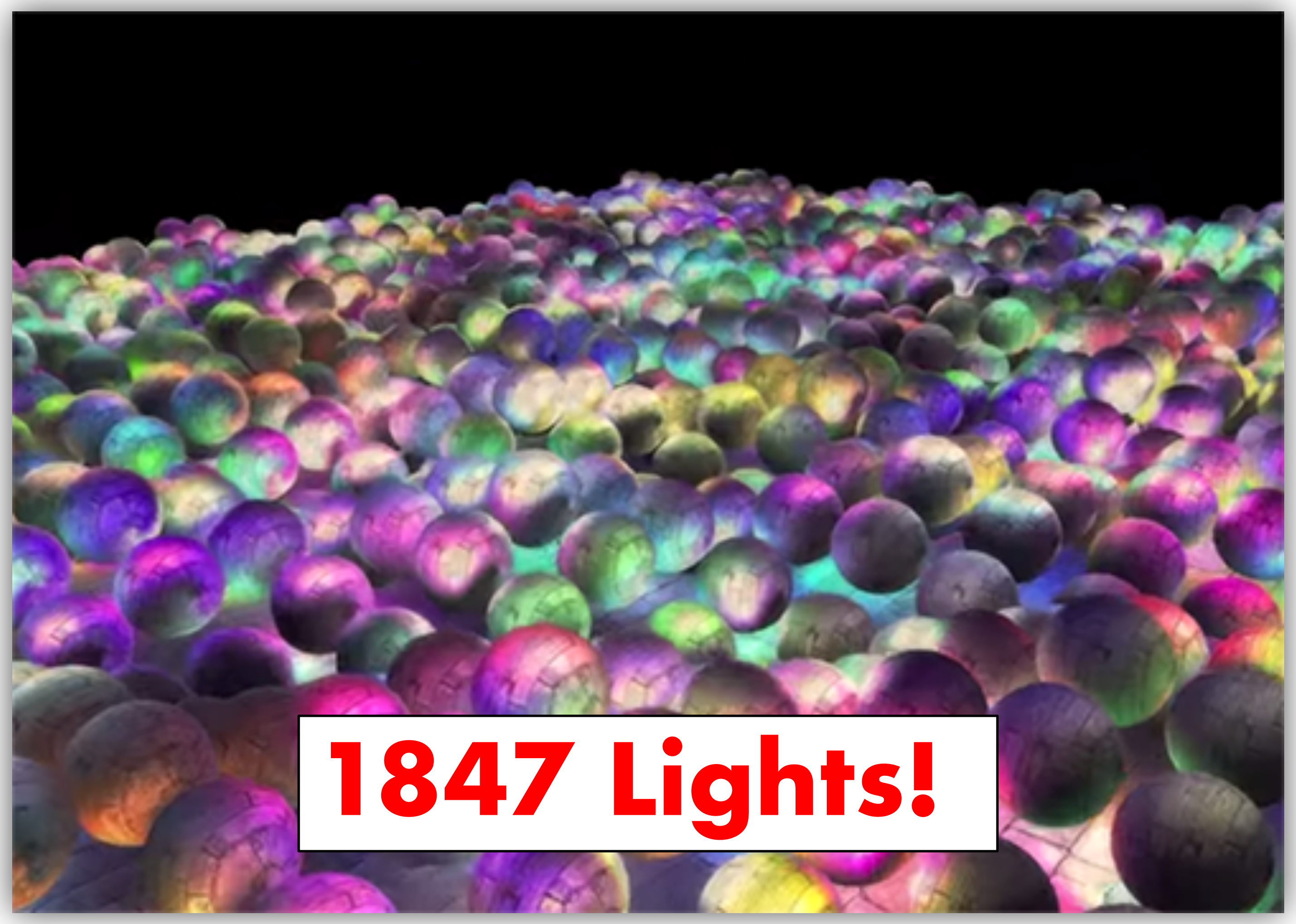
Two Pass

1. Geometry Pass
2. Lighting Pass



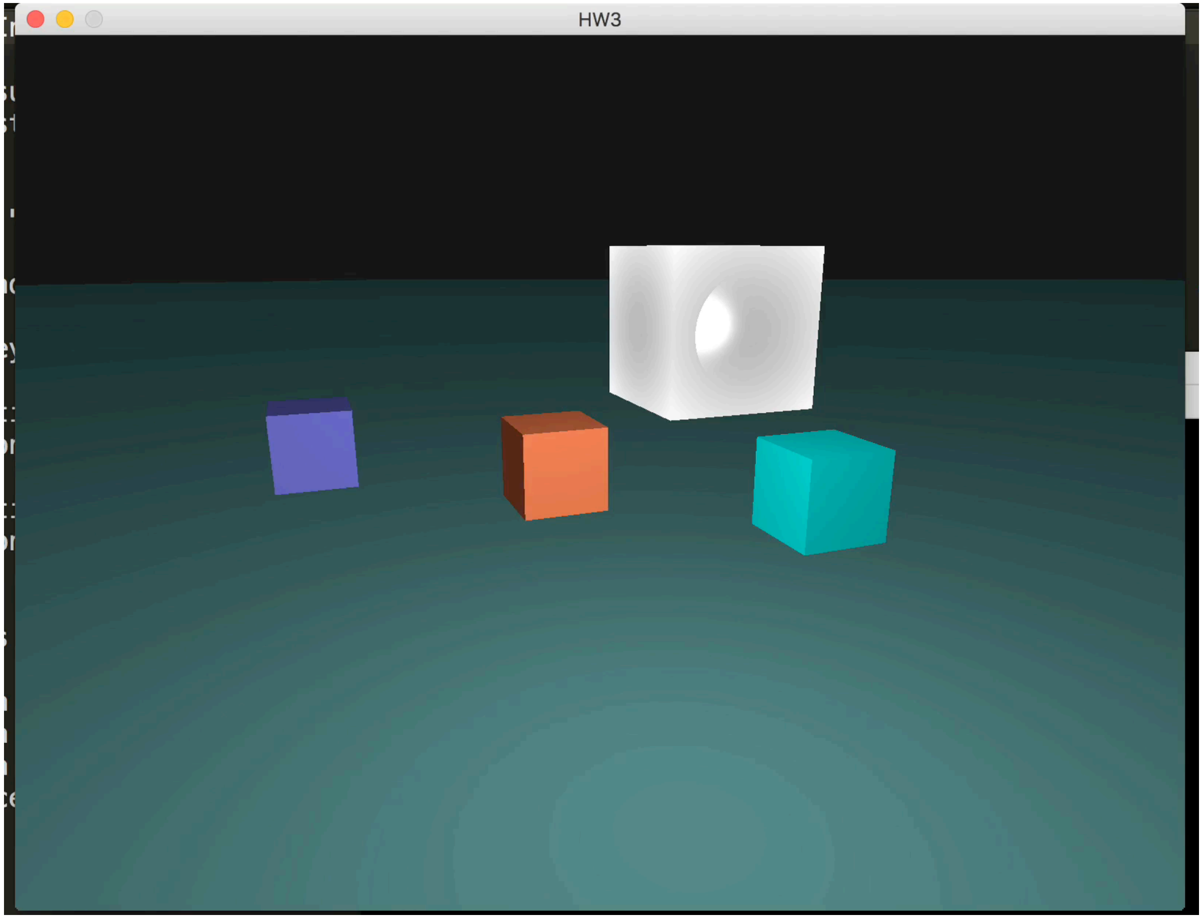
http://learnopengl.com/#!Advanced-Lighting/Deferred-Shading

Deferred Rendering: Lots of Light



http://learnopengl.com/#!Advanced-Lighting/Deferred-Shading

Best Extra Credits HW3

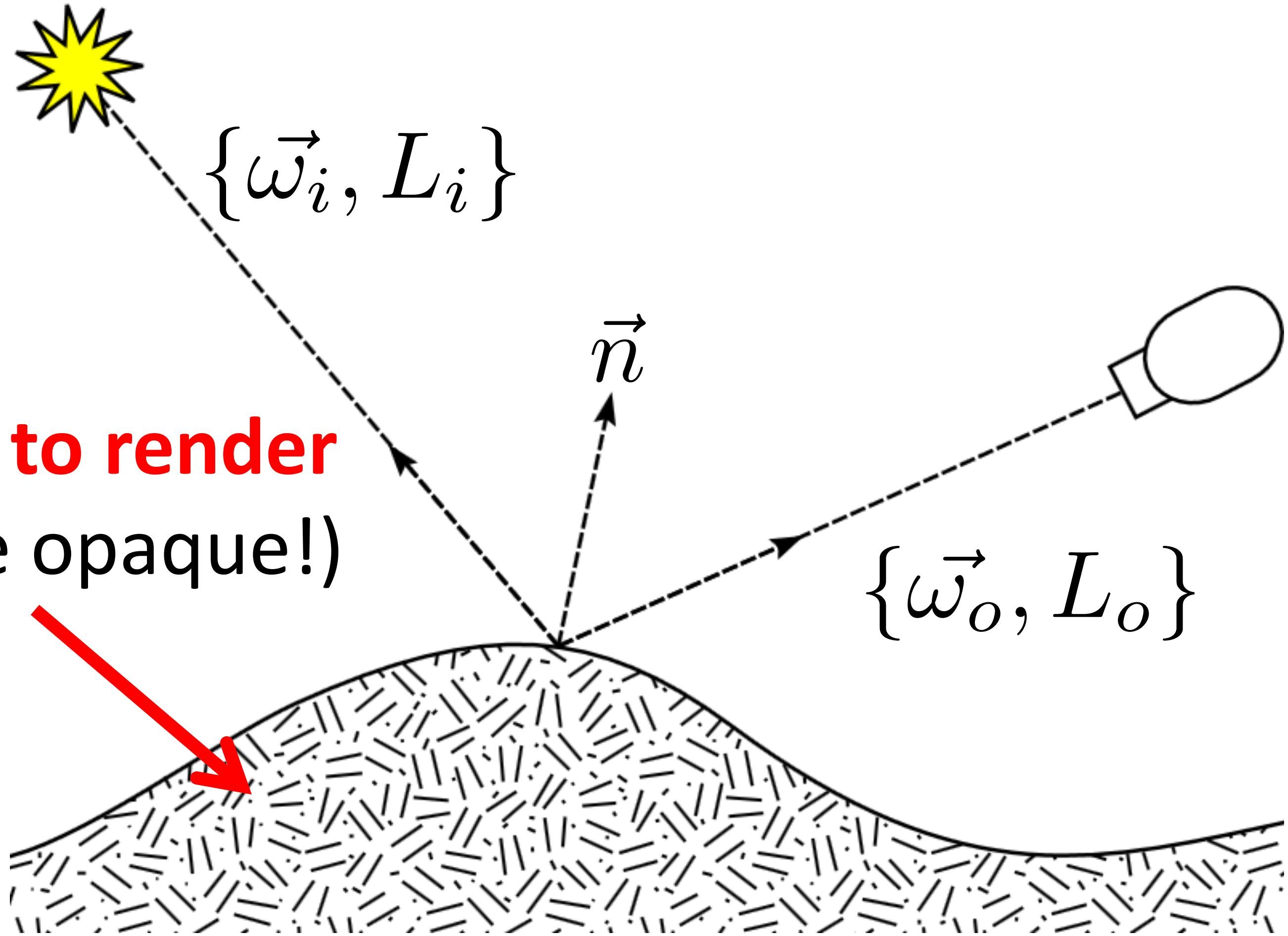


Light Box - Danny Diekroeger

Week 5

Materials & Ray Tracing

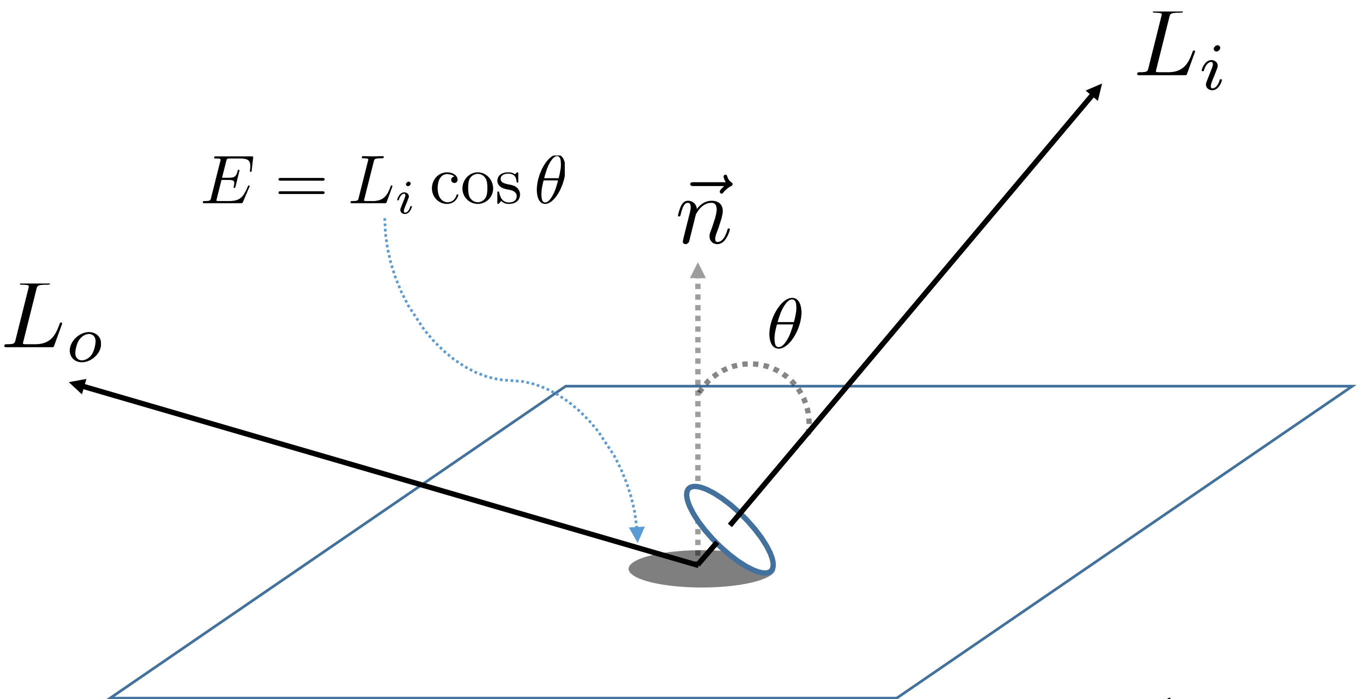
Basic Shading Model



Surface to render
(assume opaque!)

http://upload.wikimedia.org/wikipedia/commons/e/ed/BRDF_Diagram.svg

Basic Radiometry



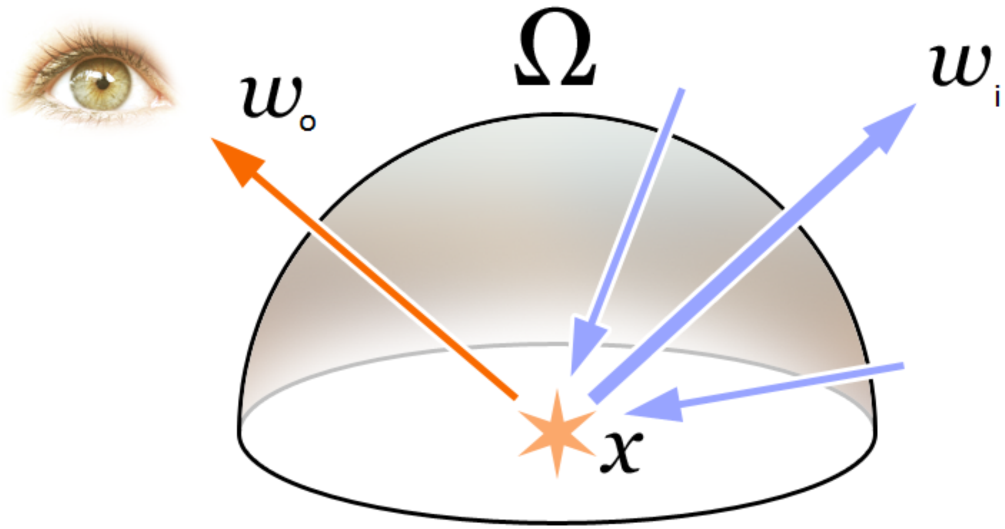
$$L_o \propto E$$

$$L_o = f\left(\ldots\right) L_i \cos\theta$$

$$f\left(\vec{\omega_i}, \vec{\omega_o}\right)$$

$$f\left(\vec{\omega_i}, \vec{\omega_o}\right) = \frac{L_o}{E} = \frac{L_o}{L_i \cos\theta}$$

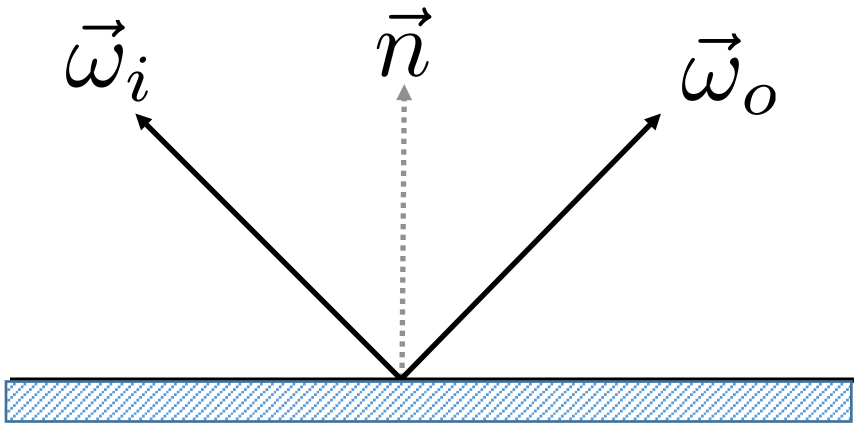
Rendering Equation



$$L_o(\vec{\omega}_o) = L_e(\vec{\omega}_o) + \int_\Omega f\,(\vec{\omega}_i, \vec{\omega}_o)\, L_i(\vec{\omega}_i)\, \underbrace{(\vec{n} \cdot \vec{w}_i)}_{\cos\theta}\, d\vec{\omega}_i$$

Sum over all incoming light

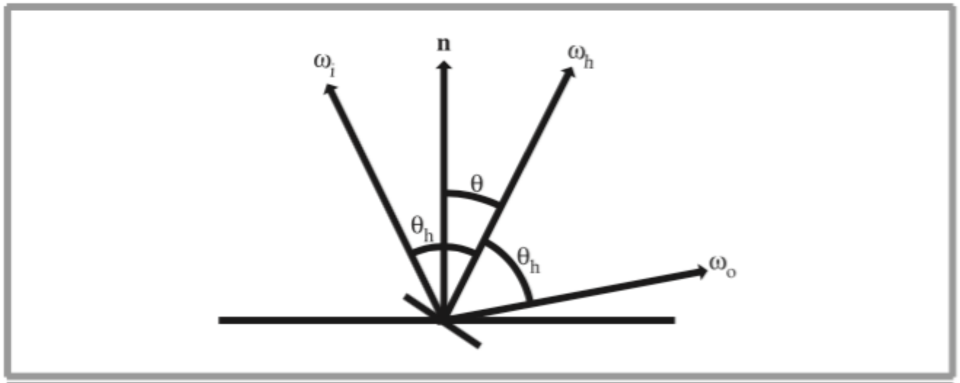
Reflection BRDF



$$L_o(\vec{\omega}_o) = \int_{\Omega} f\left(\vec{\omega_i}, \vec{\omega_o}\right) L_i(\vec{\omega}_i)(\vec{n} \cdot \vec{w_i}) d\vec{\omega}_i$$

$$f\left(\vec{\omega_i}, \vec{\omega_o}\right) = \frac{\delta\left(\vec{\omega_i} - r(\vec{\omega_o})\right)}{\vec{n} \cdot \vec{\omega_i}}$$

Microfacet: Torrance-Sparrow



Key idea:

For a given $\vec{\omega}_i, \vec{\omega}_o$, all the microfacets with orientation $\vec{\omega}_h = \underbrace{\widehat{\vec{\omega}_i + \vec{\omega}_o}}_{\text{half angle}}$ will reflect

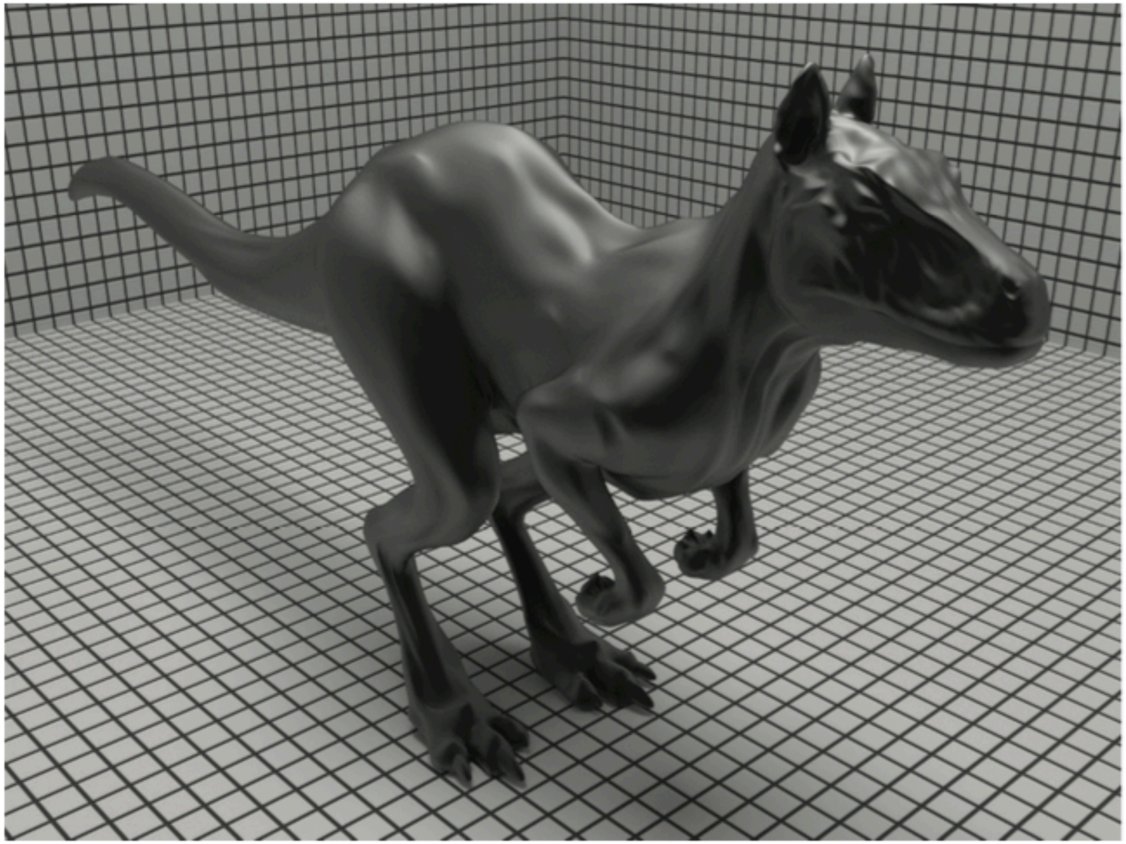
$D\left(\vec{\omega}_h\right)$: Distribution of microfacets with orientation $\vec{\omega}_h$

$$f\left(\vec{\omega}_i, \vec{\omega}_o\right) = \frac{\mathrm{D}(\vec{\omega}_h)}{4\left(\vec{n} \cdot \vec{\omega}_o\right)\left(\vec{n} \cdot \vec{\omega}_i\right)}$$

$$D\left(\vec{\omega_h}\right) = \frac{e+2}{2\pi}\left(\vec{\omega_h} \cdot \vec{n}\right)^e$$

Blinn's Distribution

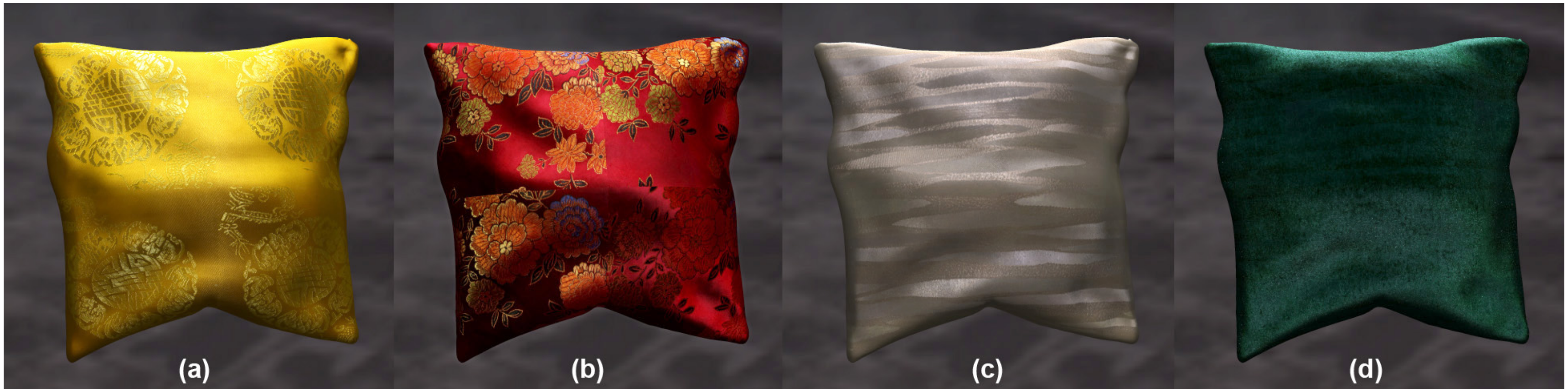
Microfacet: Torrance-Sparrow



With Blinn's distribution

Has a nice parameter to vary from "smooth" to "rough"

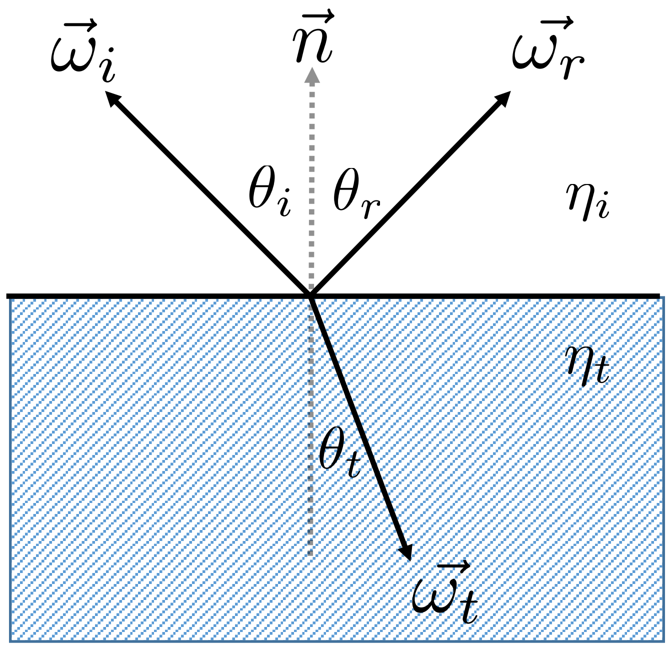
Extensions of BRDFs



Spatially-varying BRDF (SVBRDF)

http://research.microsoft.com/en-us/um/people/johnsny/images/aniso_pillows.bmp

Fresnel Term



For Dielectrics

$$r_\parallel = \frac{\eta_t \cos(\theta_i) - \eta_i \cos(\theta_t)}{\eta_t \cos(\theta_i) + \eta_i \cos(\theta_t)}$$

$$r_\perp = \frac{\eta_i \cos(\theta_i) - \eta_t \cos(\theta_t)}{\eta_i \cos(\theta_i) + \eta_t \cos(\theta_t)}$$

For unpolarized light

$$F_r = \frac{1}{2}\left(r_\parallel^2 + r_\perp^2\right)$$

$$F_t = 1 - F_r$$

Schlick's Approximation

$$F_r = F_0 + (1 - F_0)(1 - \cos\theta_i)^5$$

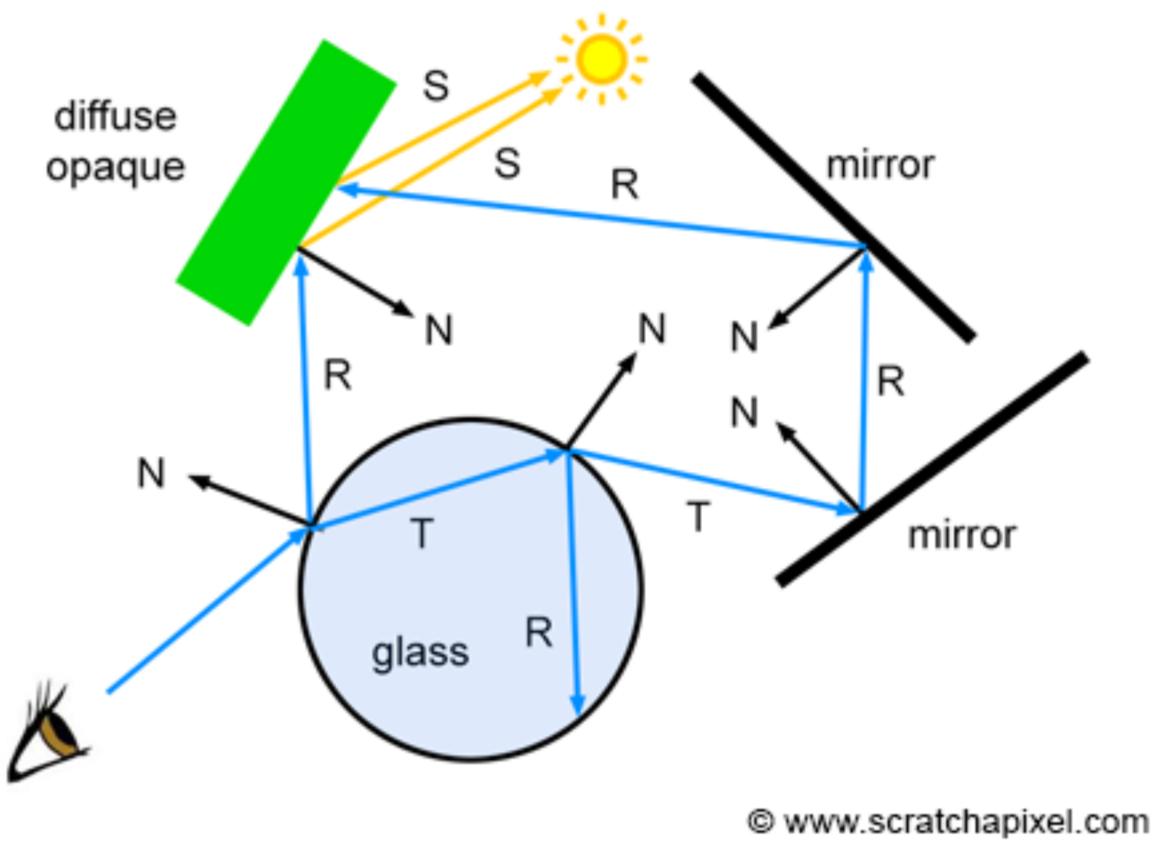
$$F_0 = \left(\frac{\eta_t - \eta_i}{\eta_t + \eta_i}\right)^2$$

Extensions of BRDFs

B Subsurface Scattering RDF (BSSRDF)

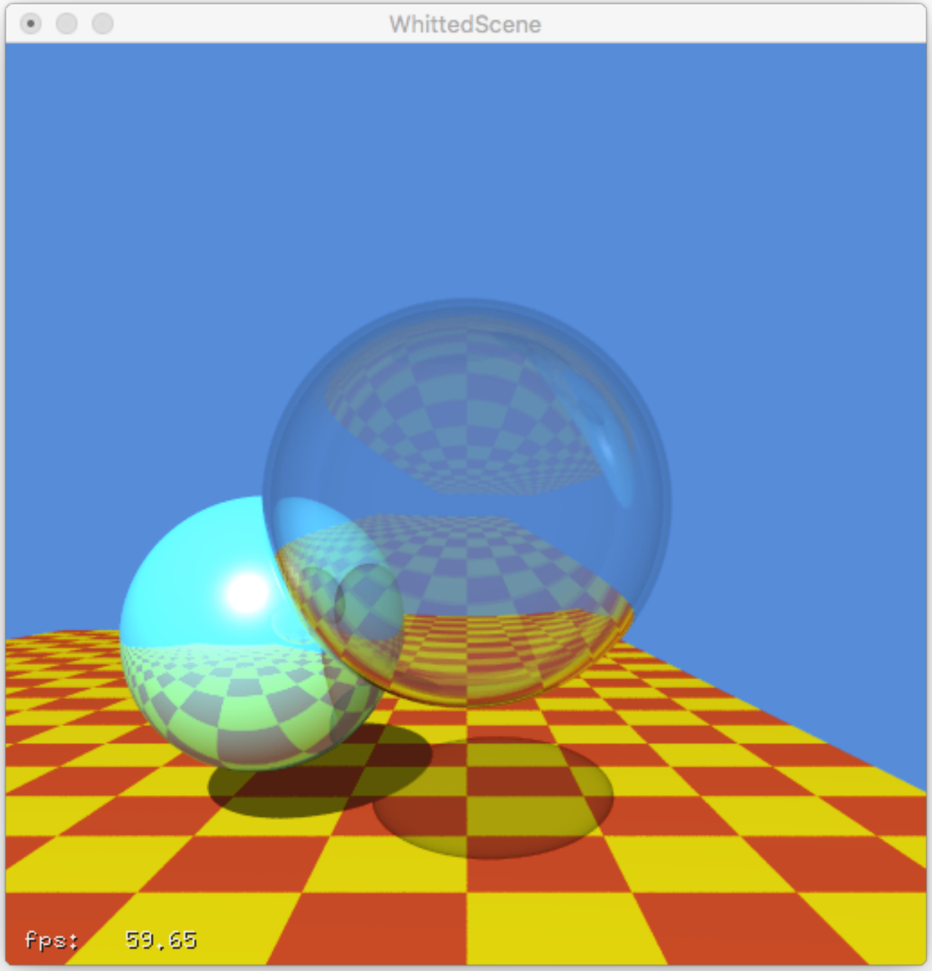
http://graphics.ucsd.edu/~henrik/images/imgs/diana_bssrdf.jpg

Recursive Ray Tracing



http://www.scratchapixel.com/images/upload/ray-tracing-refresher/rt-whitted-example.png?

Today: nVidia OptiX

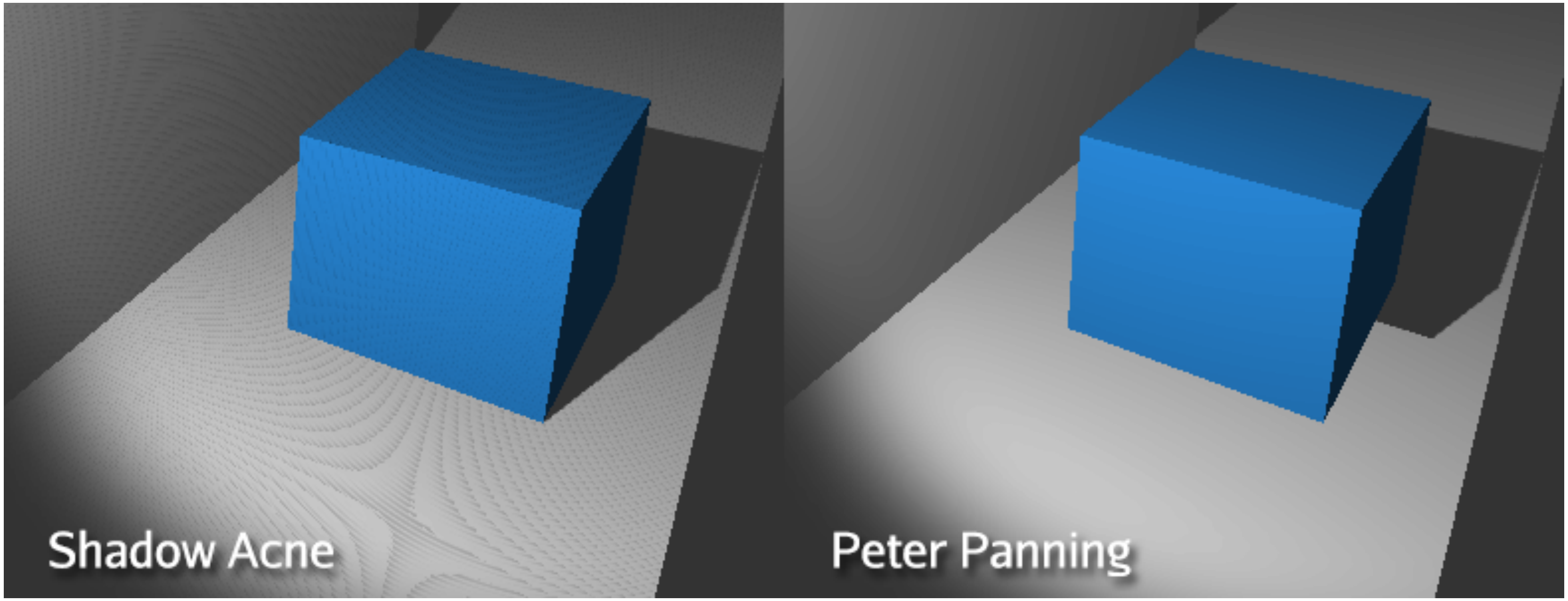


Render Time: 16 ms ! (60 fps): my Macbook Pro nVidia GTX M750

Today: Realism

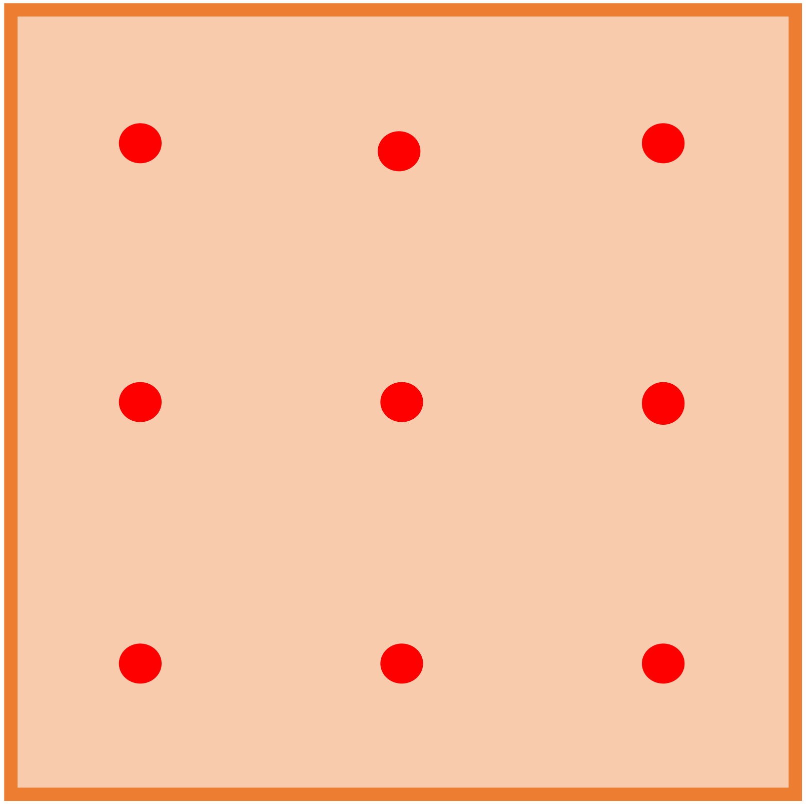
http://images.povcomp.com/entries/images/105_main.jpg

Shadow Acne / Peter Panning



Real life issues !

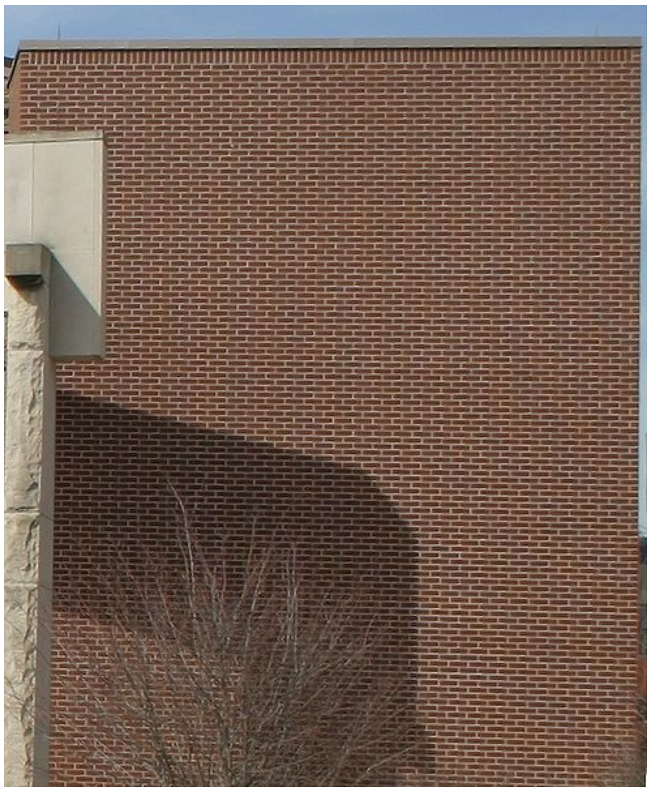
Distribution Antialiasing

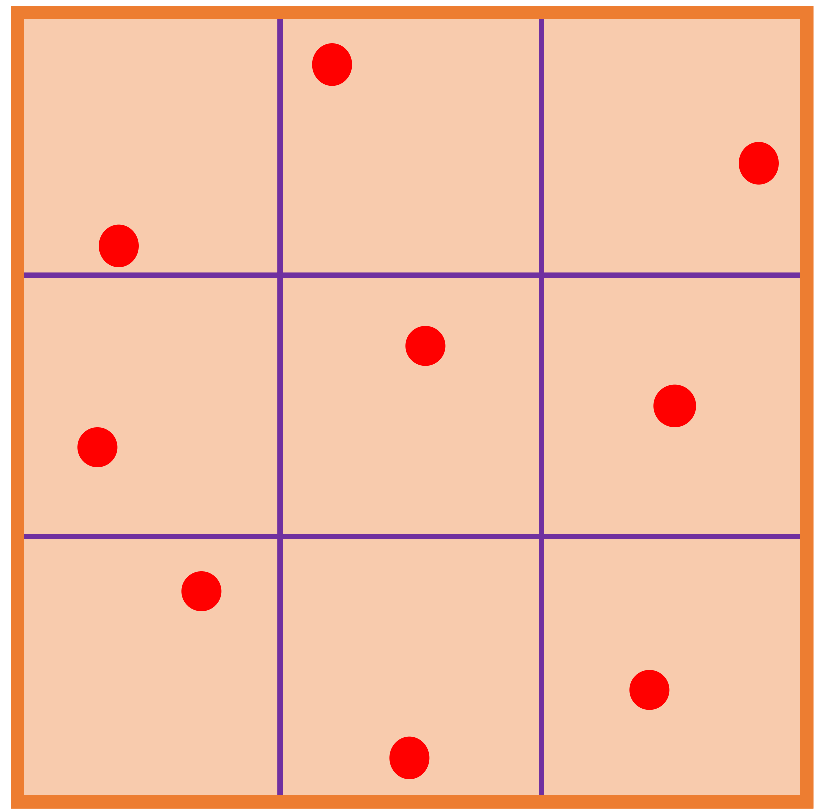


Patterned Noise

http://upload.wikimedia.org/wikipedia/commons/f/fb/Moire_pattern_of_bricks_small.jpg

Random Sampling and Jittering

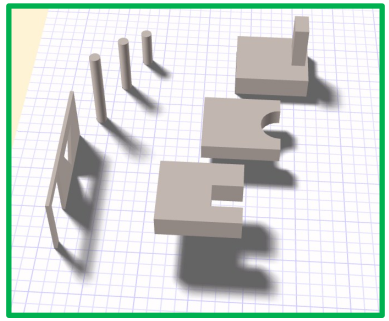




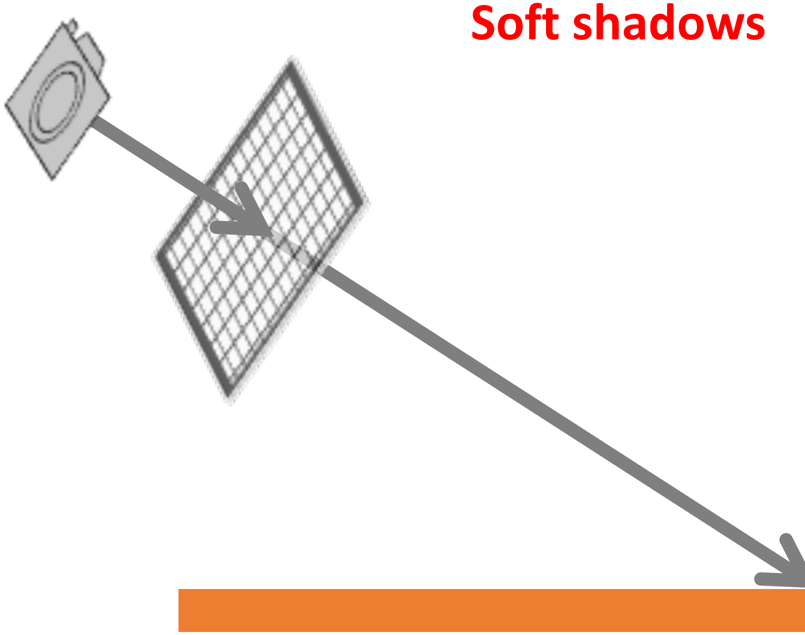
We are less sensitive to granular noise

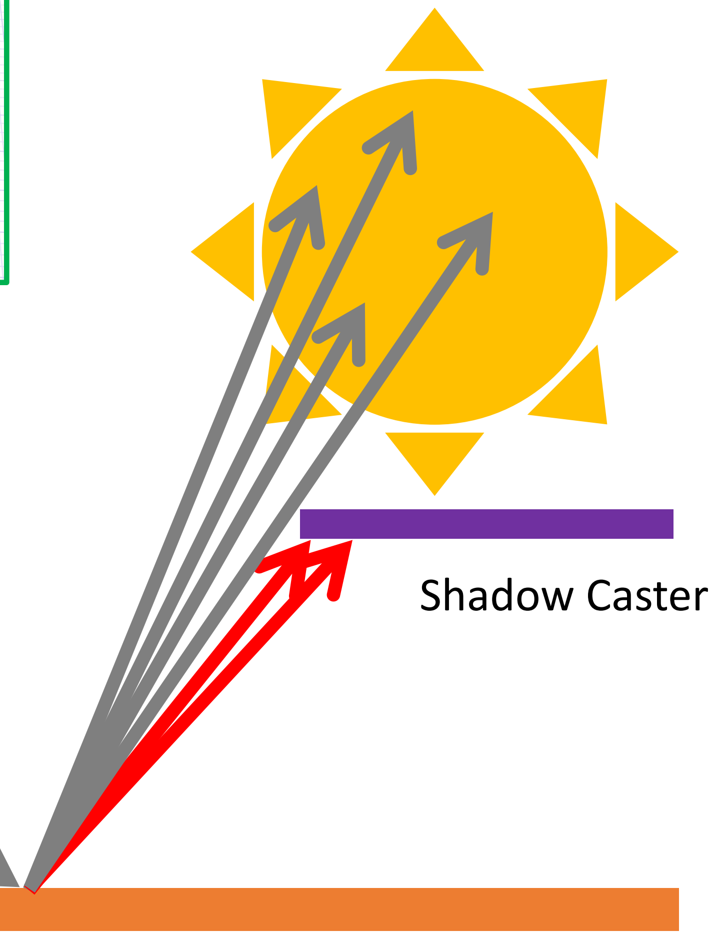
http://en.wikipedia.org/wiki/File:Moire_pattern_of_bricks.jpg

Soft Shadows



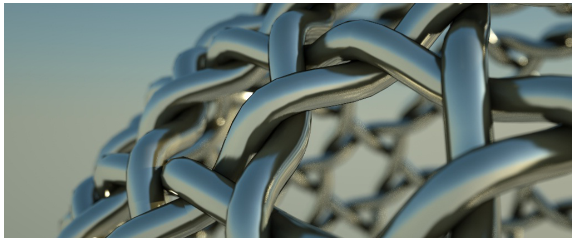
Soft shadows

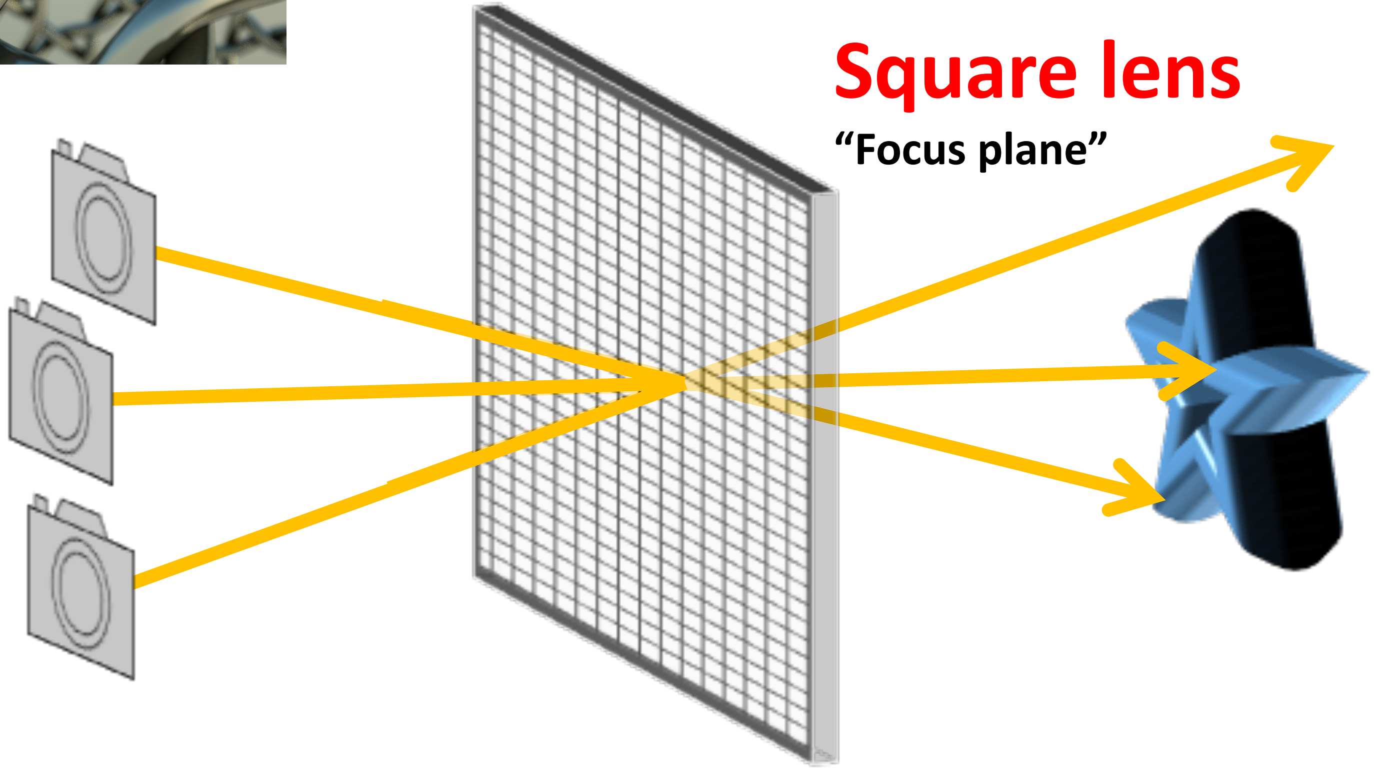




Shadow computed per ray: Average intensity

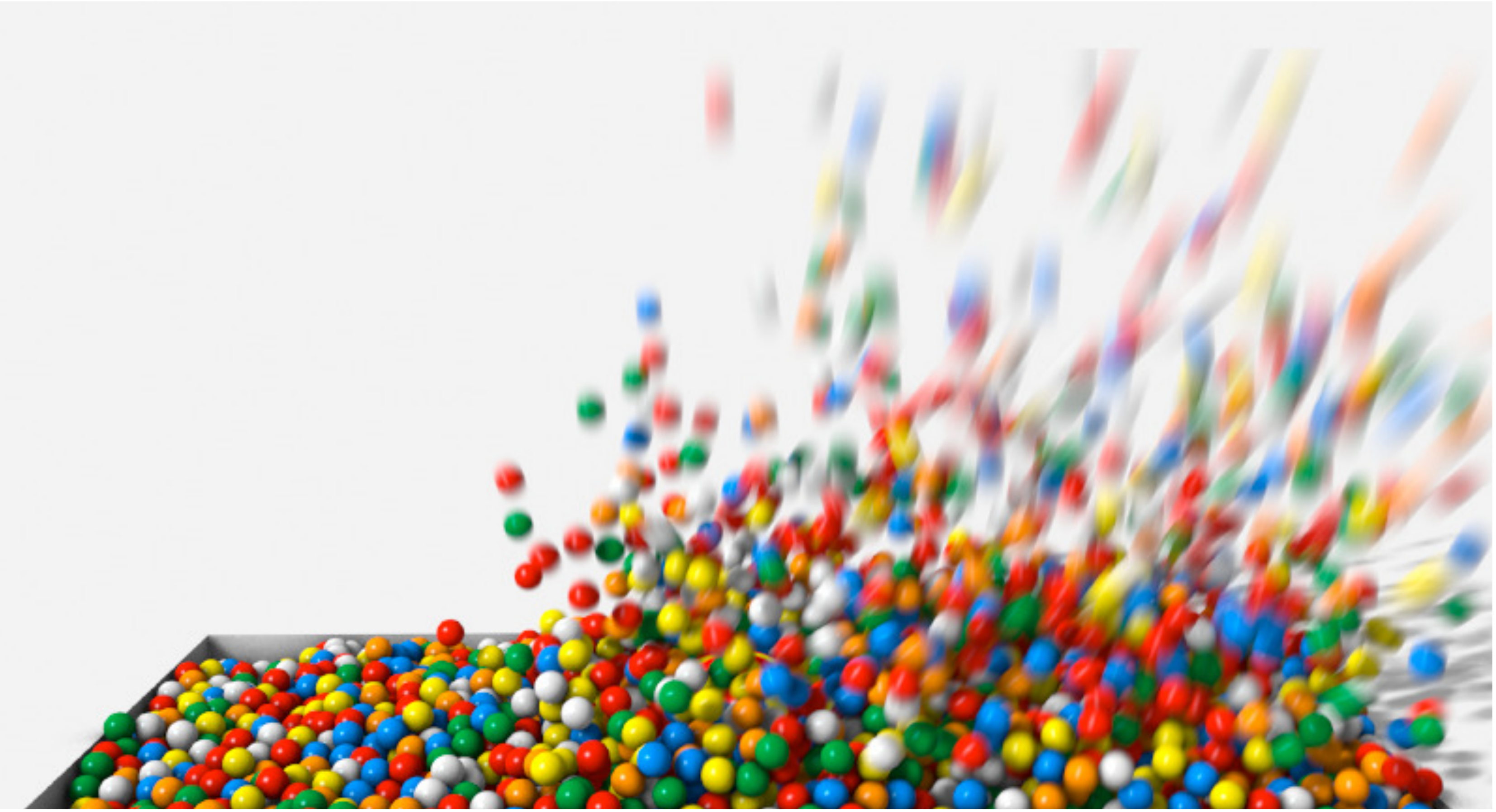
Depth of Field





Randomly sample eye positions

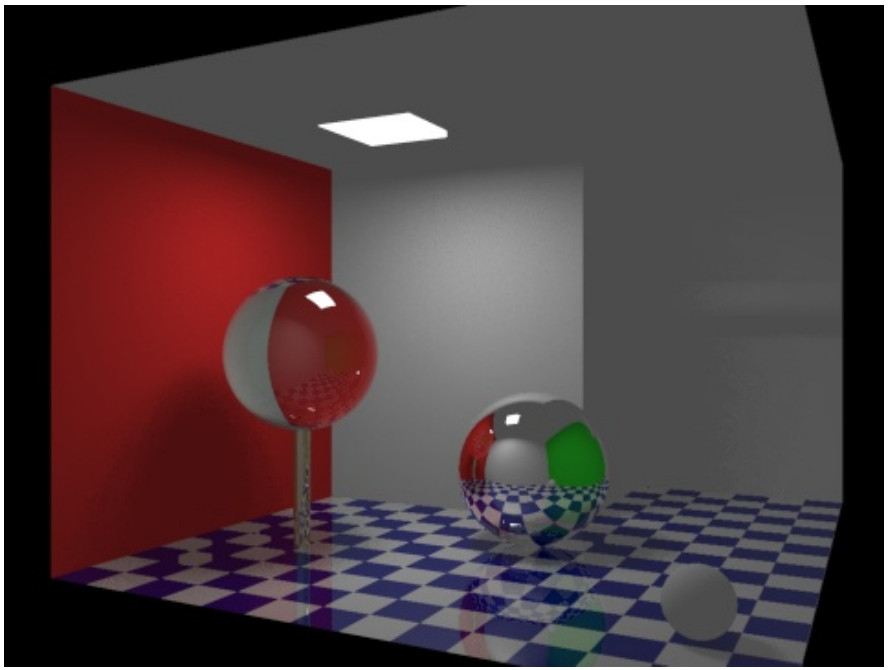
Motion Blur

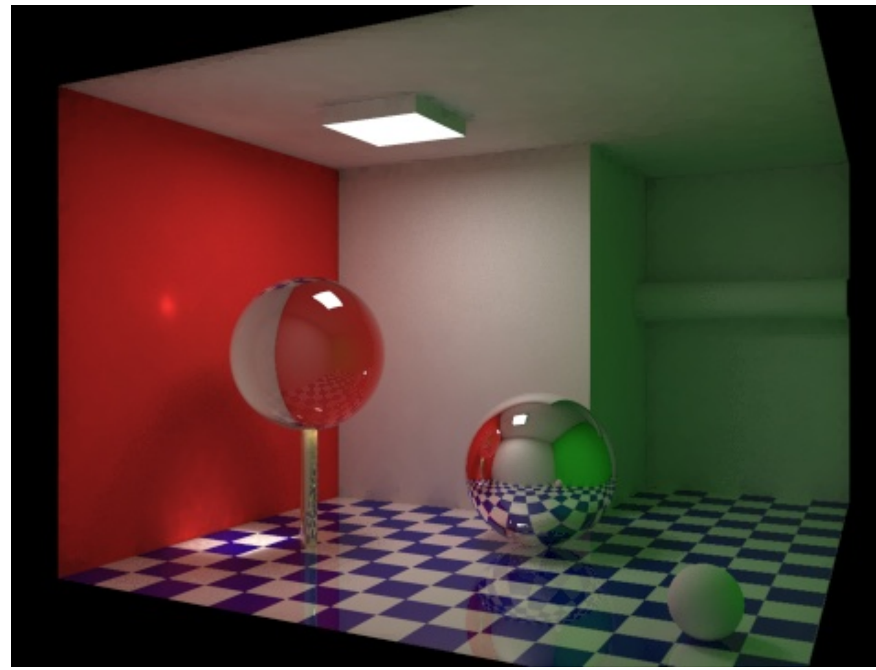


http://www.matkovic.com/anto/3dl-test-balls-01.jpg

Randomly sample positions

Global Illumination



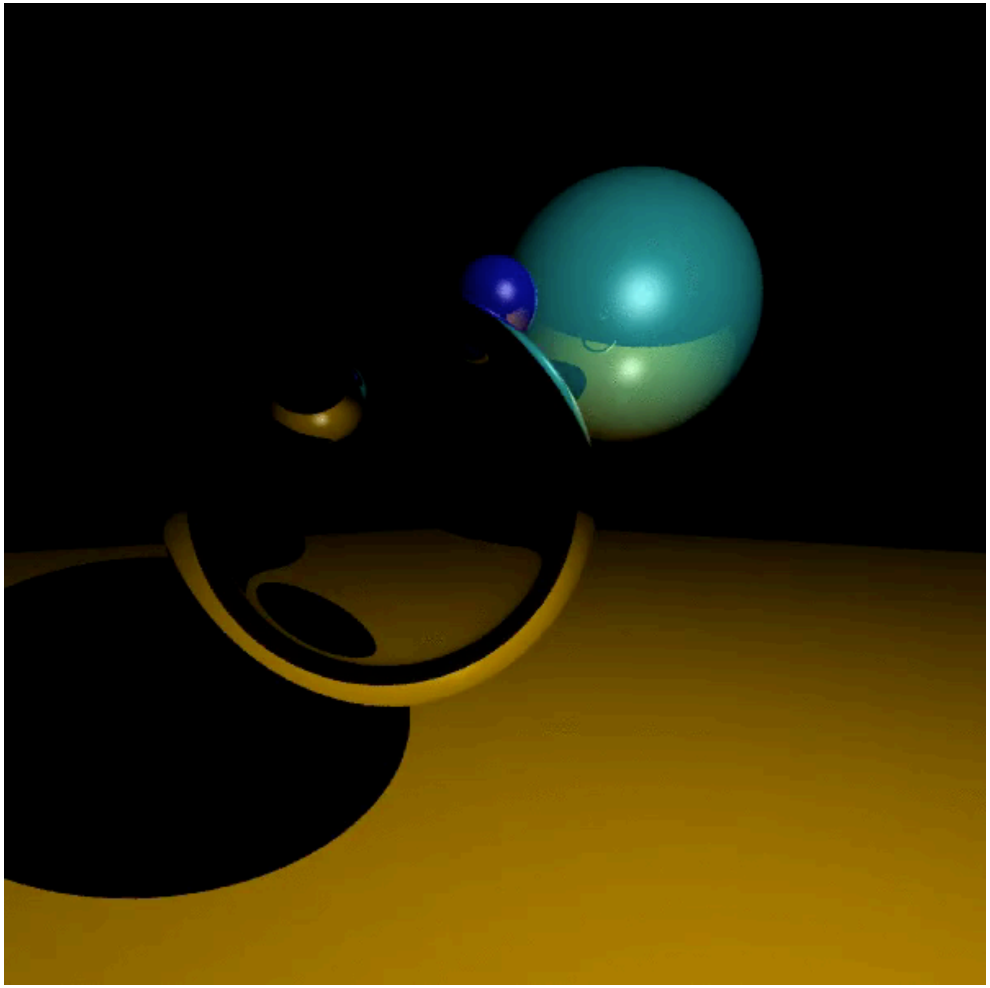


Account for indirect lighting

http://en.wikipedia.org/wiki/Global_illumination

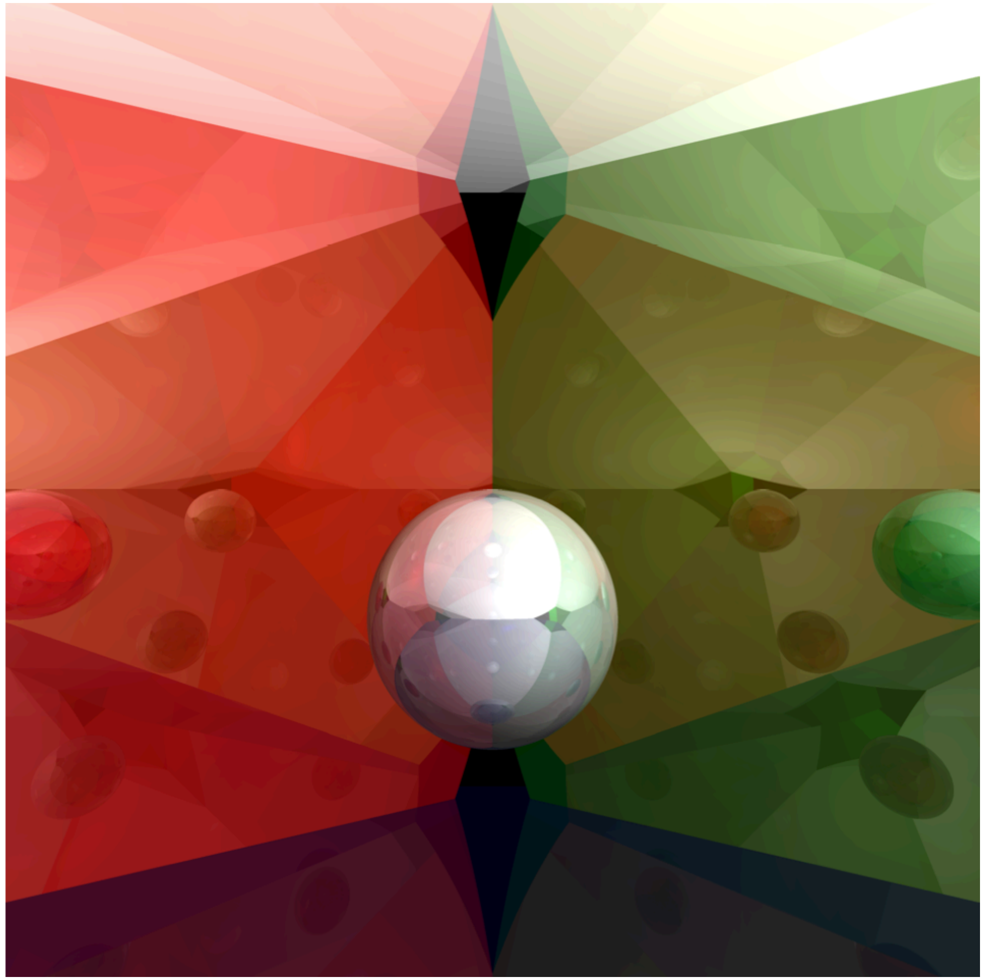
Bi-directional Path Tracing

Best Extra Credits: HW4



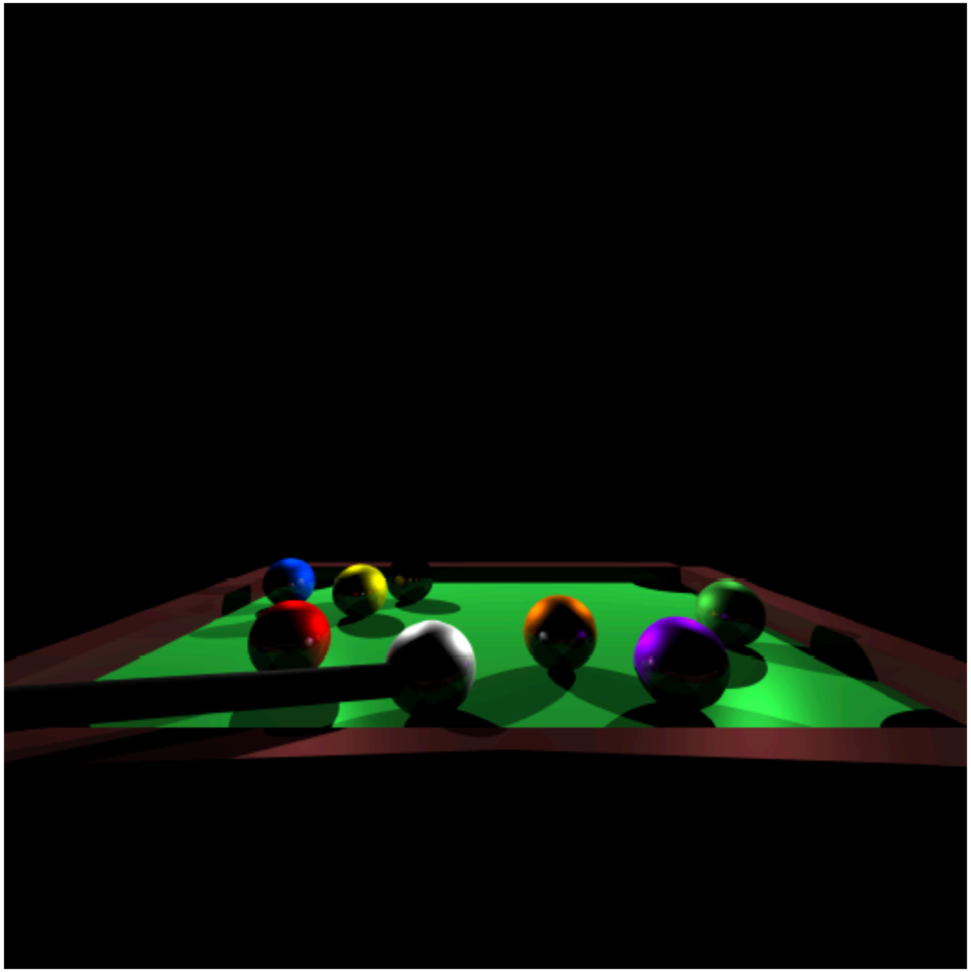
Gardar Sigurdsson

Best Extra Credits: HW4



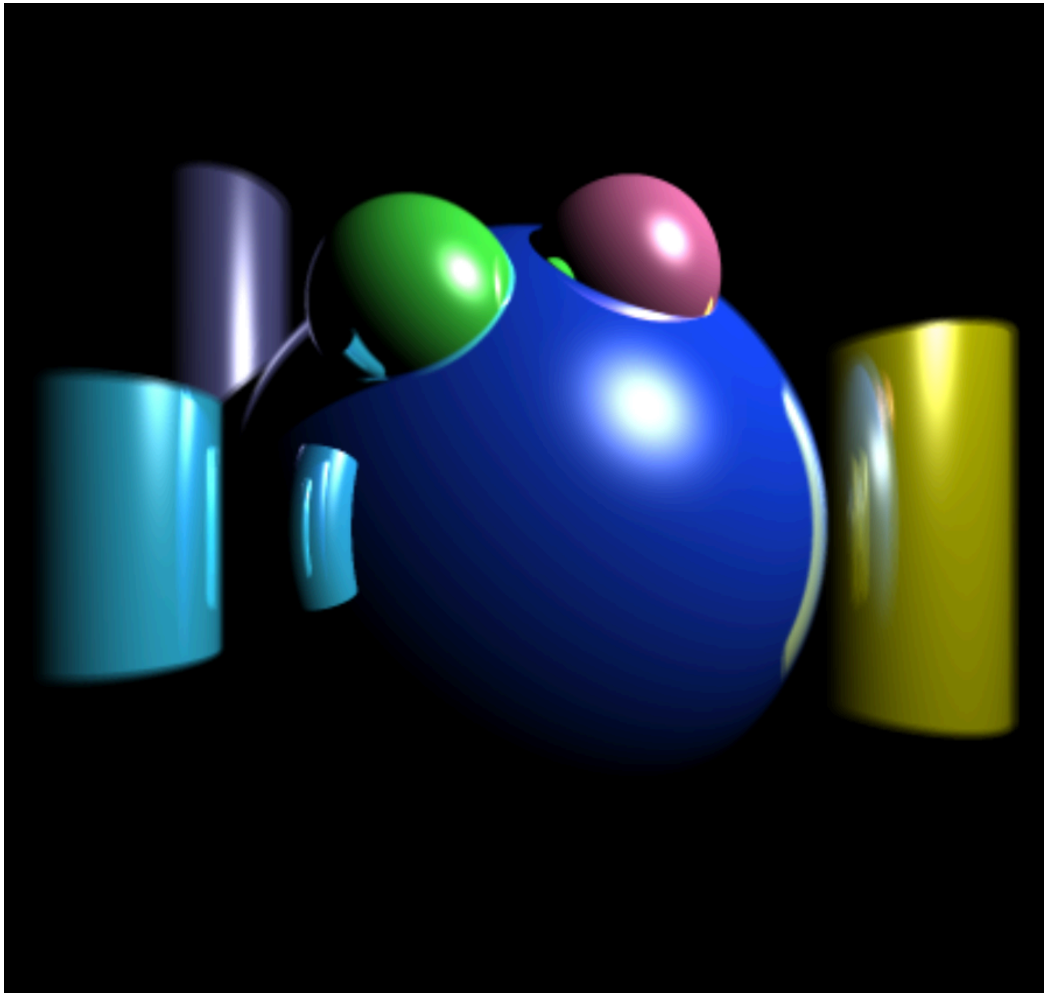
Veronica Kim

Best Extra Credits: HW4



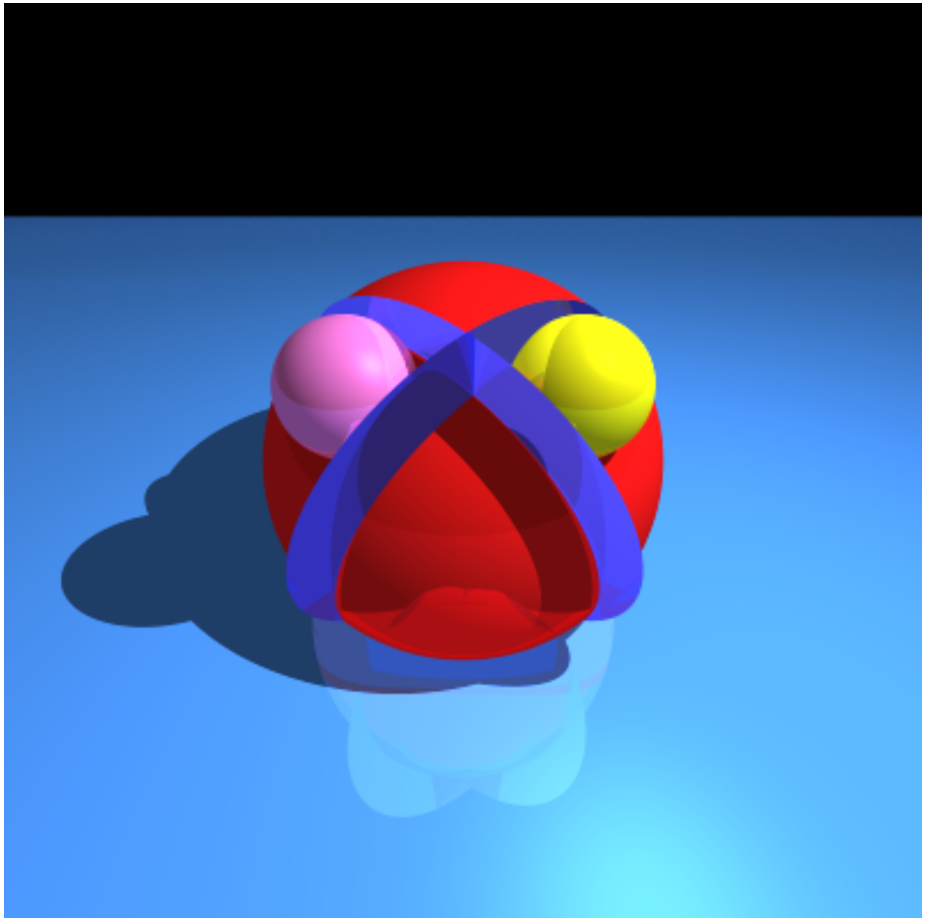
Alexander Schaub

Best Extra Credits: HW4



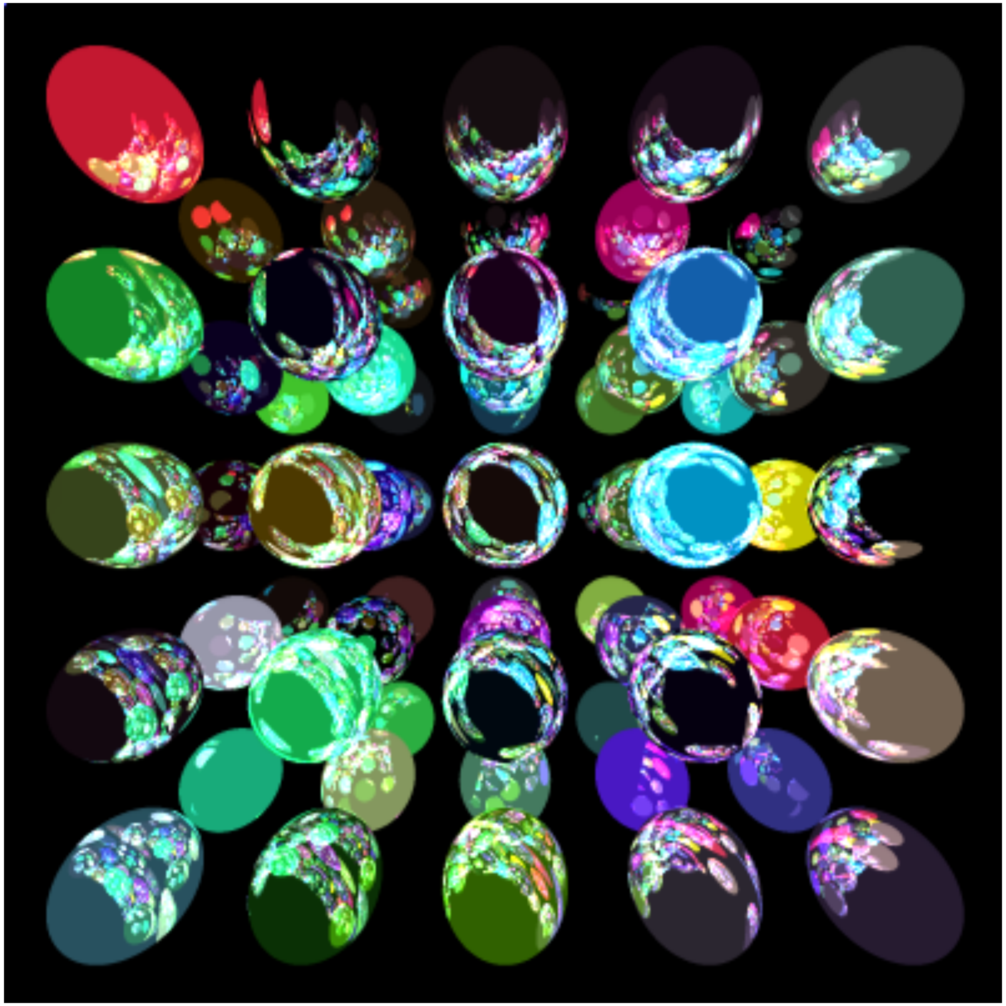
Benjamin Harry

Best Extra Credits: HW4



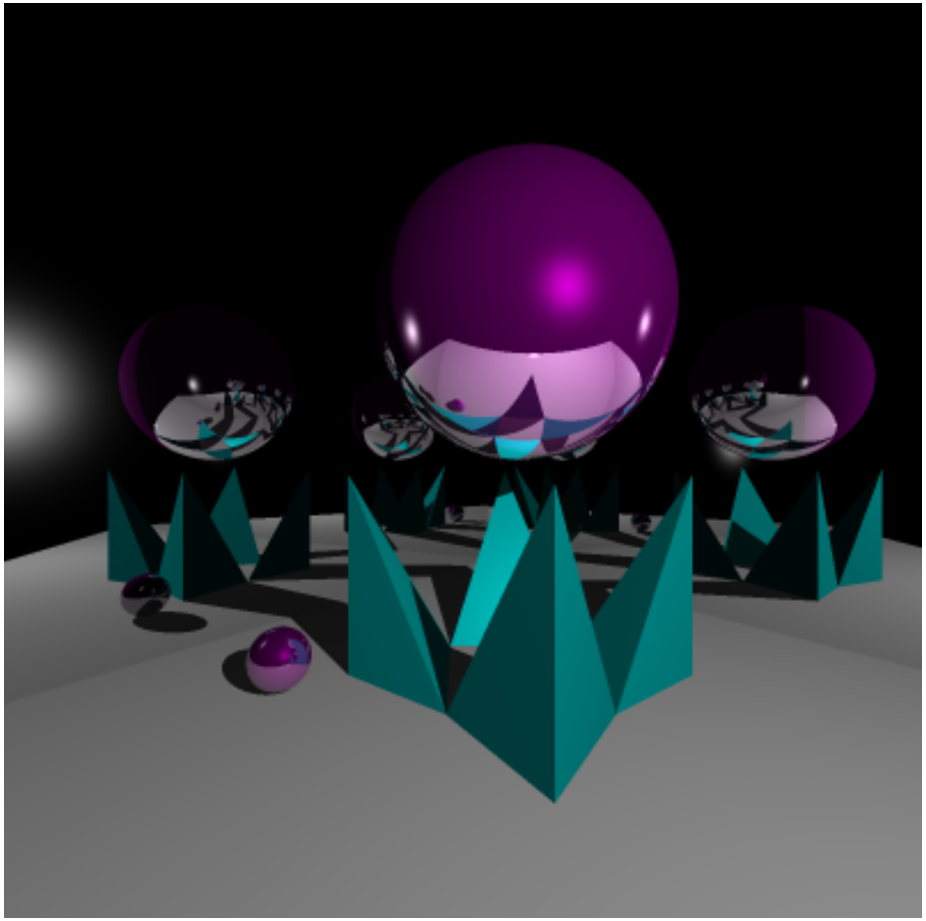
Sumant Sharma

Best Extra Credits: HW4



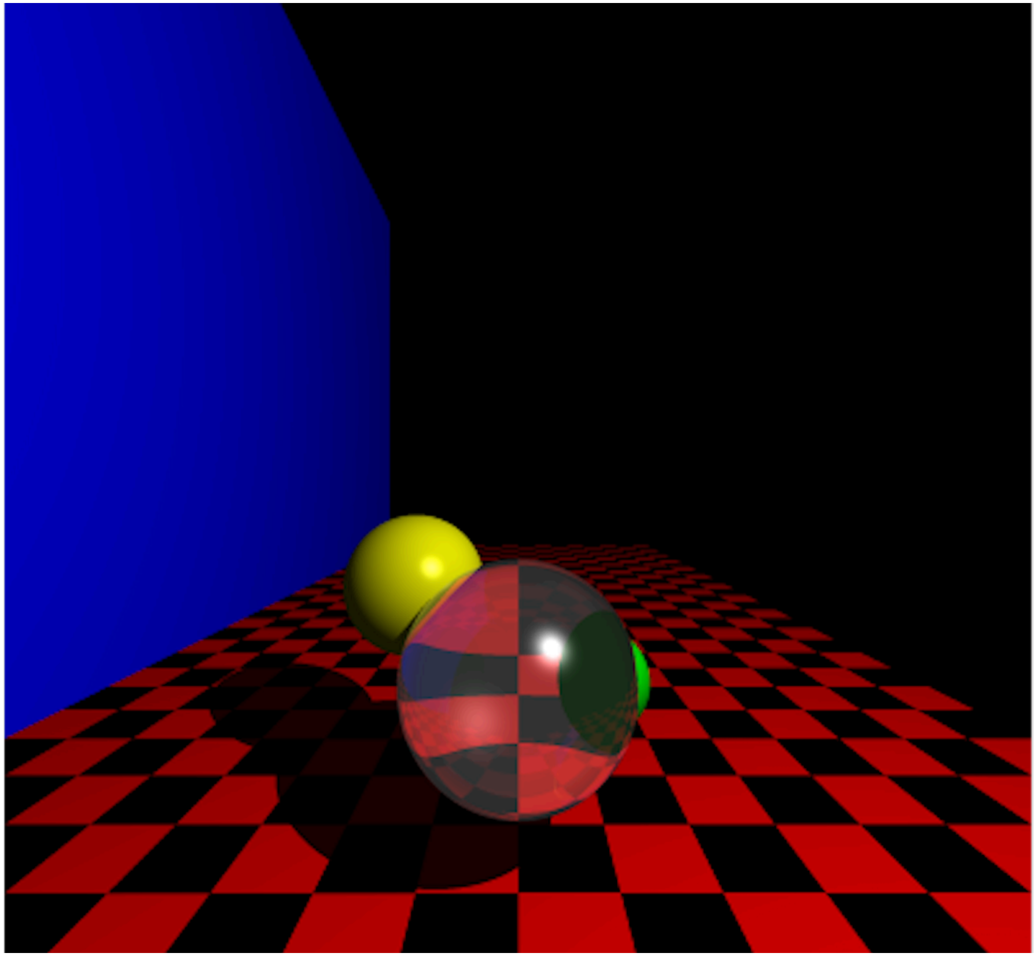
Wenjie Zheng

Best Extra Credits: HW4



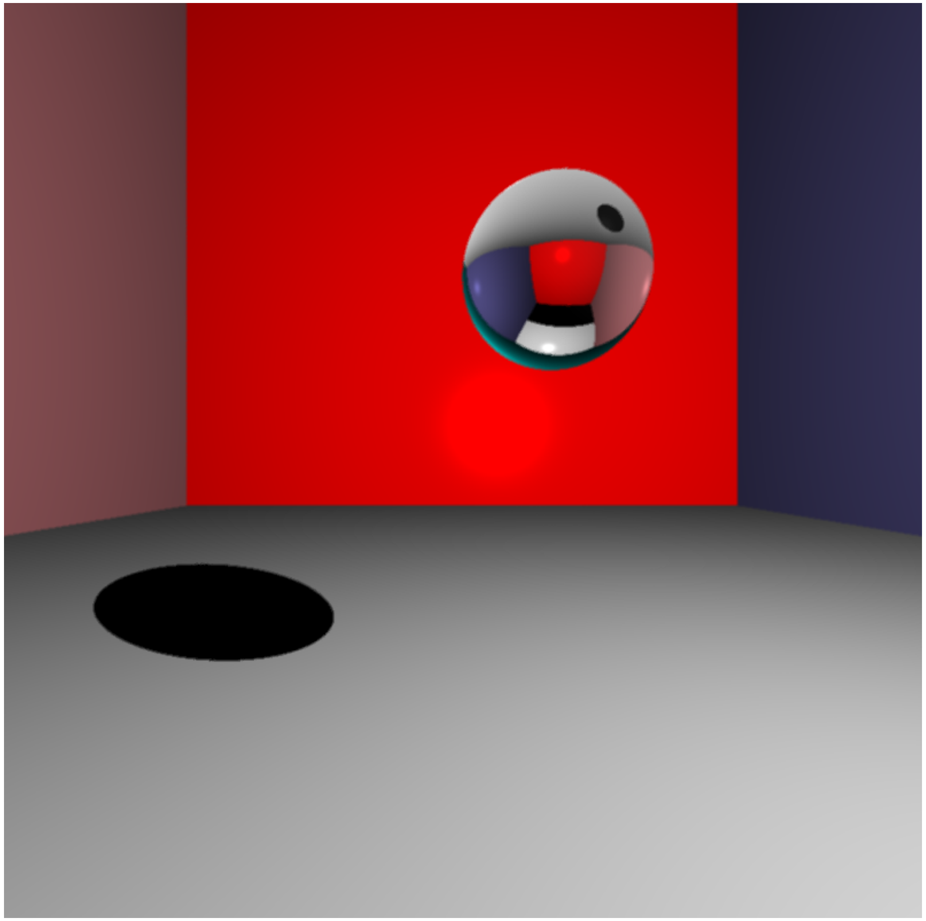
Shalom Rottman-Yang

Best Extra Credits: HW4

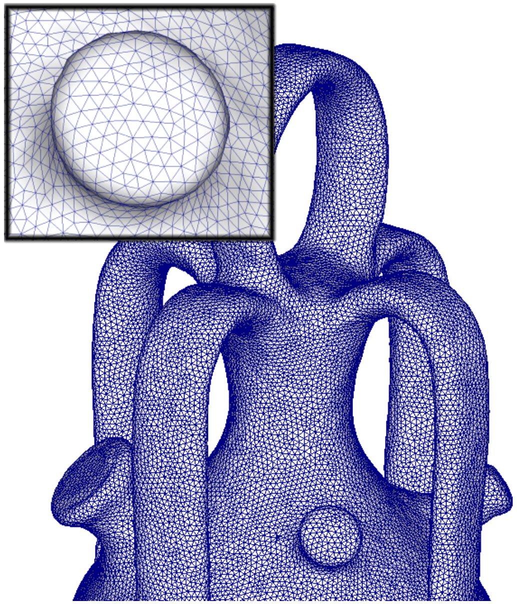


Xuanyu Zhou

Best Extra Credits: HW4



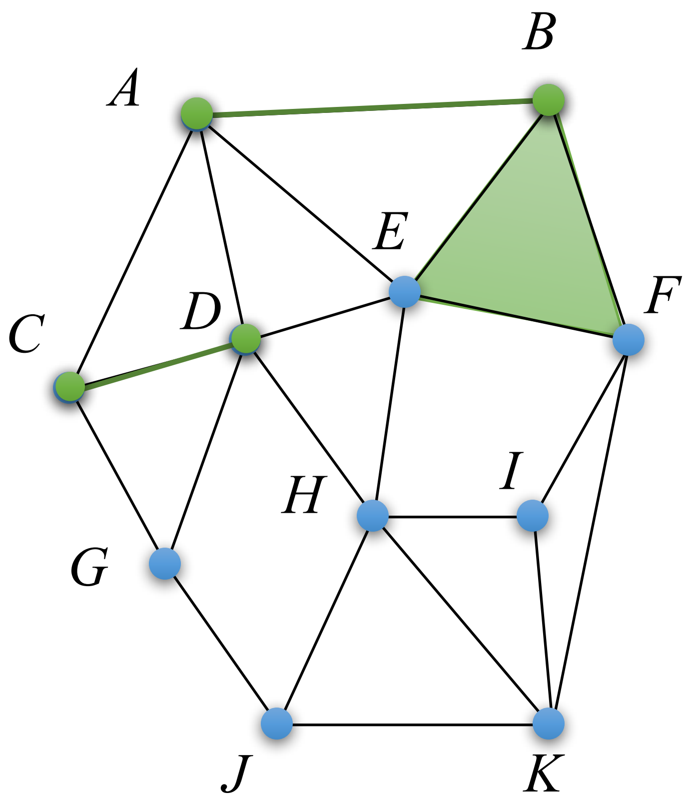
Jean-Paul Schmetz



Week 6

Geometry Processing and Animation

Triangle Mesh as Graph



G = graph = <V, E>

V = vertices = {A, B, C, ..., K}

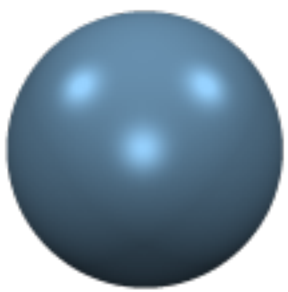
E = edges = {(AB), (AE), (CD), ...}

F = faces = {(ABE), ($DHJG$), ...}

Global Topology: Genus

Genus:

Half the maximal number of closed paths that do not disconnect the mesh (= the number of holes)



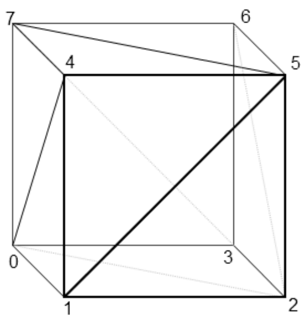
Genus 0

Genus 1

Genus 2

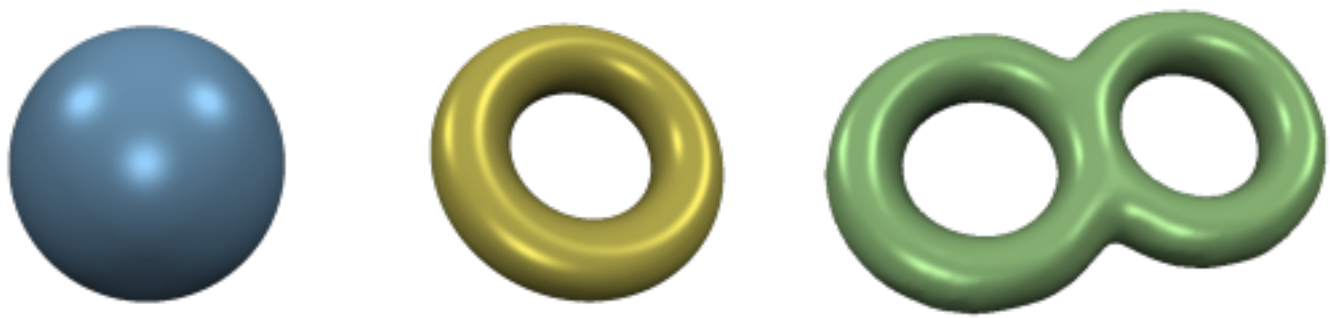
Euler-Poincaré formula

$$V + F - E = 2(1 - g)$$



$$V = 8, \quad E = 18, \quad F = 12$$

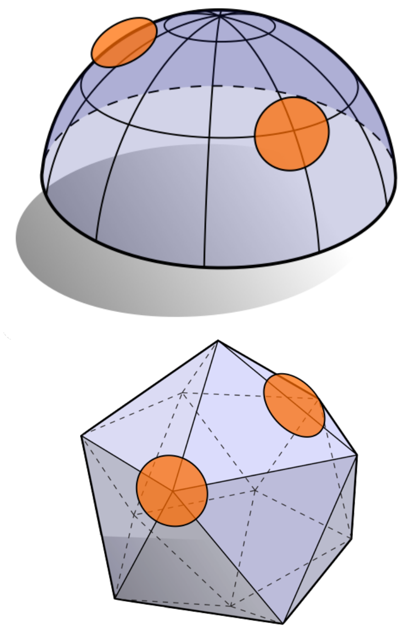
$$8 + 12 - 18 = 2 \implies g = 0$$



$g = 0$ $g = 1$ $g = 2$

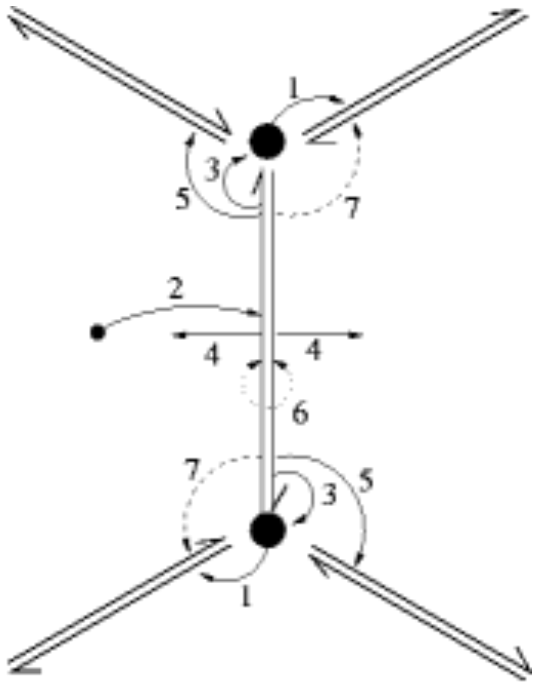
Manifold

- A edge connects exactly two faces
- An edge connects exactly two vertices
- A face consists of a ring of edges and vertices
- A vertex consists of a ring of edges and vertices



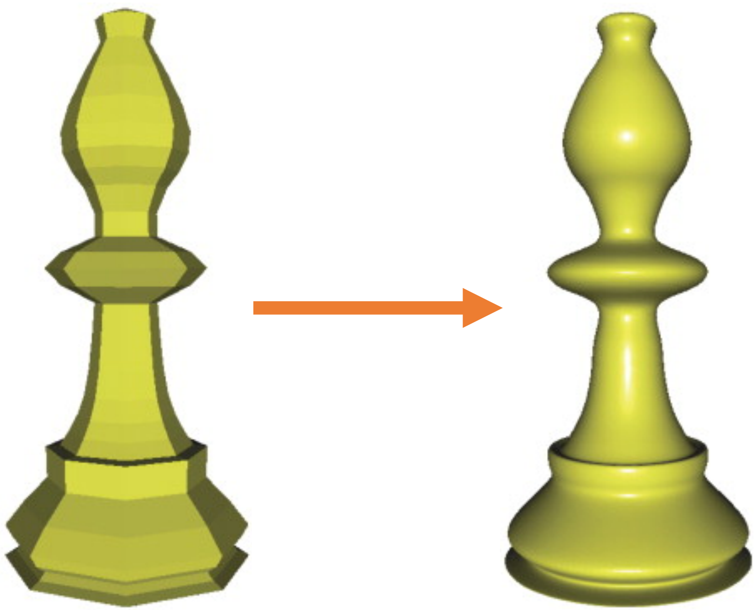
Halfedge Data Structure

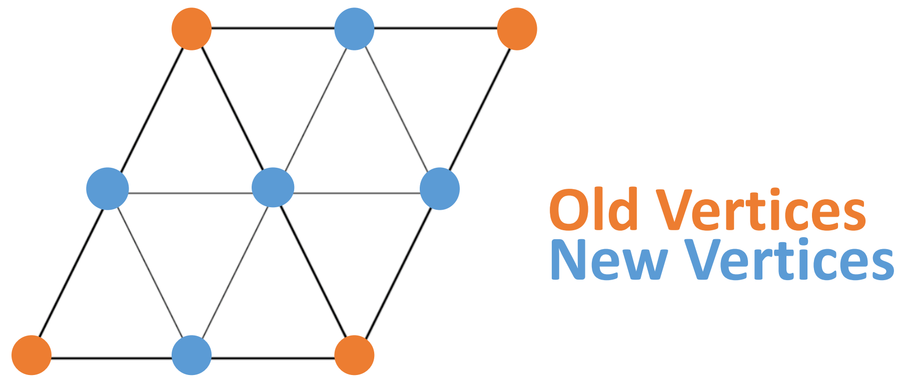
- Vertex Stores
  - Position
  - **One outgoing** halfedge (1)
- A Face Stores
  - 1 halfedge bounding to it (2)
- A halfedge stores
  - The vertex it **points to** (3)
  - The face is belongs to (4)
  - The next halfedge inside the face (5)
  - The opposite halfedge (6)
  - (optional) previous half edge in the face(7)



http://www.openmesh.org/Documentation/OpenMesh-Doc-Latest/a00016.html

Loop Subdivision

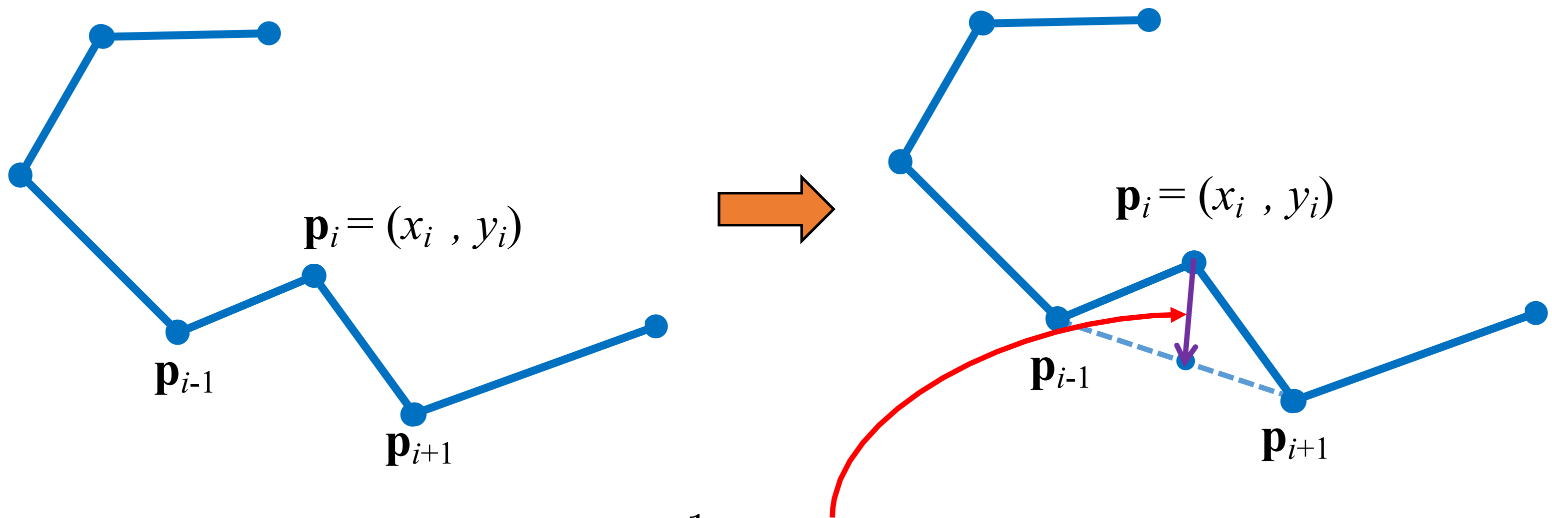




Provable smoothness and regularities

Laplacian Smoothing: 1D

An easier problem: How to smooth a curve?



$$\vec{dp}_i = \frac{1}{2}\left(P_{i-1} + P_{i+1}\right) - P_i$$

$$\vec{dp}_i = \frac{1}{2}L(P_i)$$

$$\text{Where, } L(P_i) = P_{i-1} - 2P_i + P_{i+1}$$

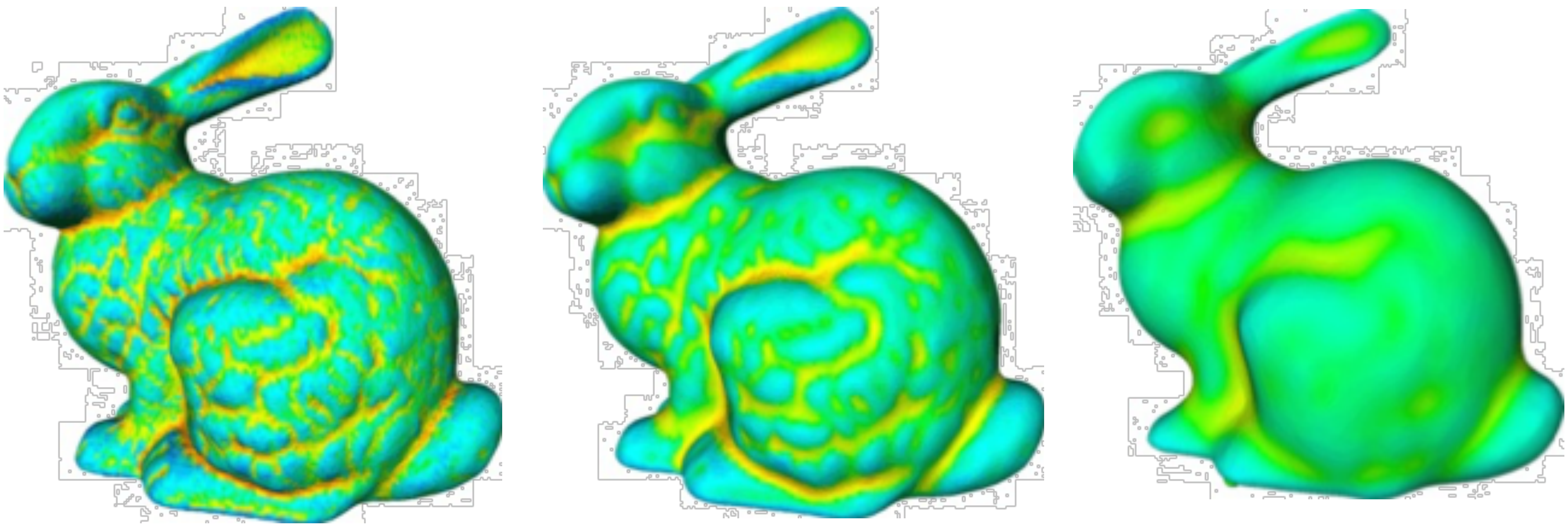
Laplacian Smoothing

$$\text{Where, } L(P_i) = P_{i-1} - 2P_i + P_{i+1}$$

$$L\left(\mathbf{P}^{(n)}\right) = -\underbrace{\begin{bmatrix} 2 & -1 & \dots & \dots & -1 \\ \dots & -1 & 2 & -1 & \dots \\ \vdots & & & & \vdots \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \end{bmatrix}}_{\mathbf{P}^{(n)}}$$

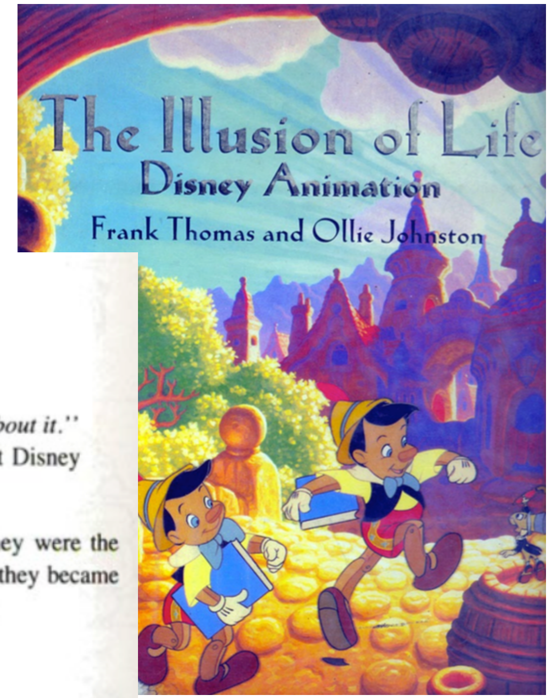
Recall: Adjacency Matrix for Mesh

Laplace Smoothing: On Mesh

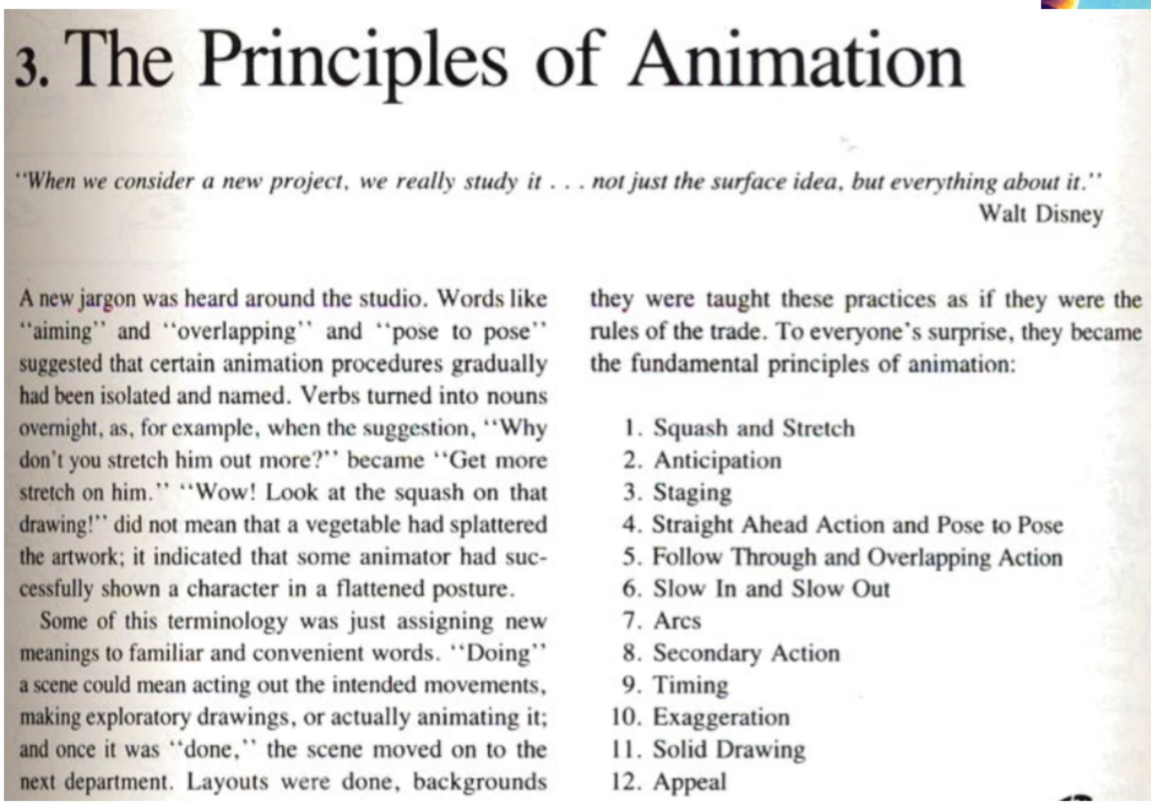


0 Iterations 5 Iterations 20 Iterations

Principles of Animation



1981



3. The Principles of Animation

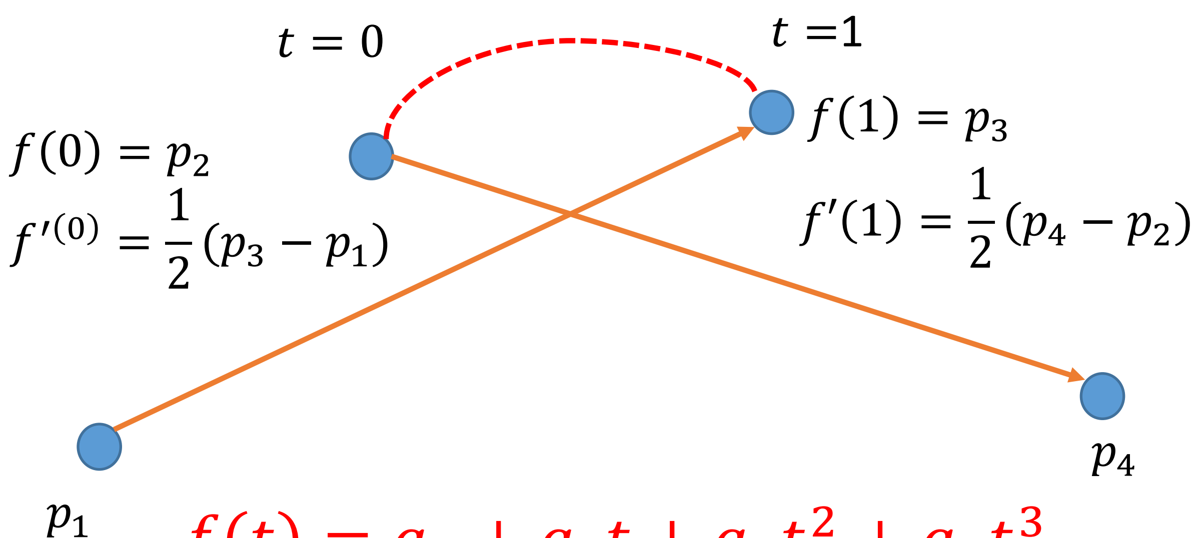
"When we consider a new project, we really study it . . . not just the surface idea, but everything about it."
Walt Disney

A new jargon was heard around the studio. Words like "aiming" and "overlapping" and "pose to pose" suggested that certain animation procedures gradually had been isolated and named. Verbs turned into nouns overnight, as, for example, when the suggestion, "Why don't you stretch him out more?" became "Get more stretch on him." "Wow! Look at the squash on that drawing!" did not mean that a vegetable had splattered the artwork; it indicated that some animator had successfully shown a character in a flattened posture.

Some of this terminology was just assigning new meanings to familiar and convenient words. "Doing" a scene could mean acting out the intended movements, making exploratory drawings, or actually animating it; and once it was "done," the scene moved on to the next department. Layouts were done, backgrounds they were taught these practices as if they were the rules of the trade. To everyone's surprise, they became the fundamental principles of animation:

1. Squash and Stretch
2. Anticipation
3. Staging
4. Straight Ahead Action and Pose to Pose
5. Follow Through and Overlapping Action
6. Slow In and Slow Out
7. Arcs
8. Secondary Action
9. Timing
10. Exaggeration
11. Solid Drawing
12. Appeal

Catmull-Rom Spline



$$f(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

$$f(0) = a_0 = p_2$$

$$f'(0) = a_1 = \frac{1}{2}(p_3 - p_1)$$

$$f(1) = a_0 + a_1 + a_2 + a_3 = p_3$$

$$f'(1) = a_1 + 2a_2 + 3a_3 = \frac{1}{2}(p_4 - p_2)$$

Bezier Curve

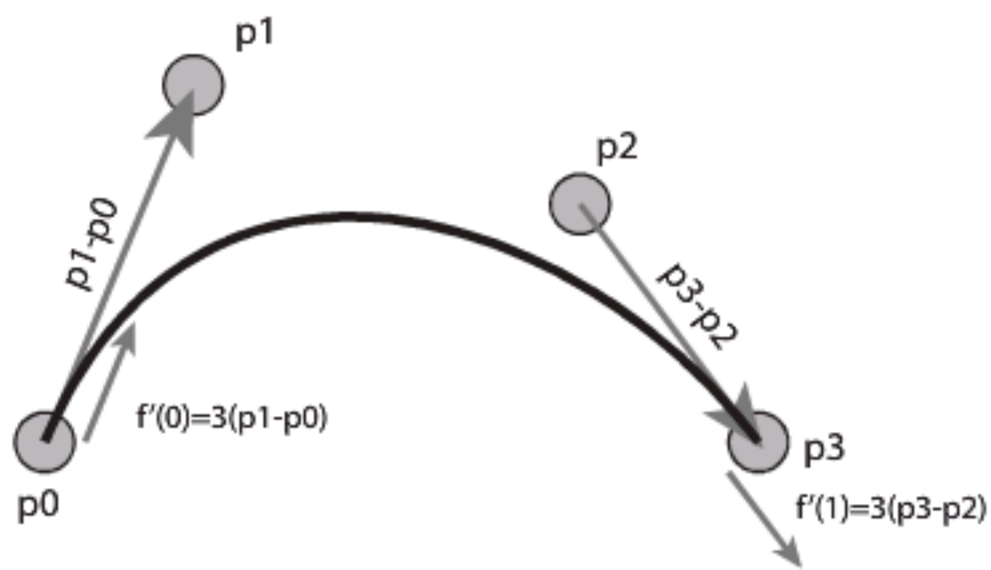
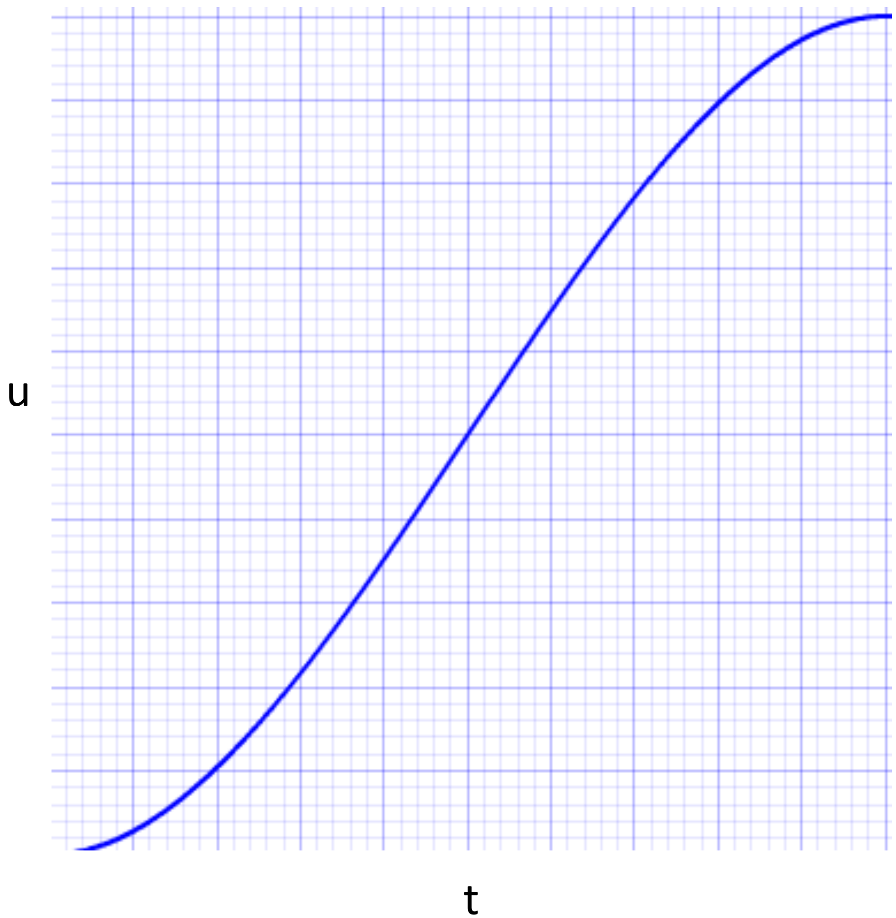


Figure 15.10. A cubic Bézier curve is controlled by four points. It interpolates the first and last, and the beginning and final derivatives are three times the vectors between the first two (or last two) points.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

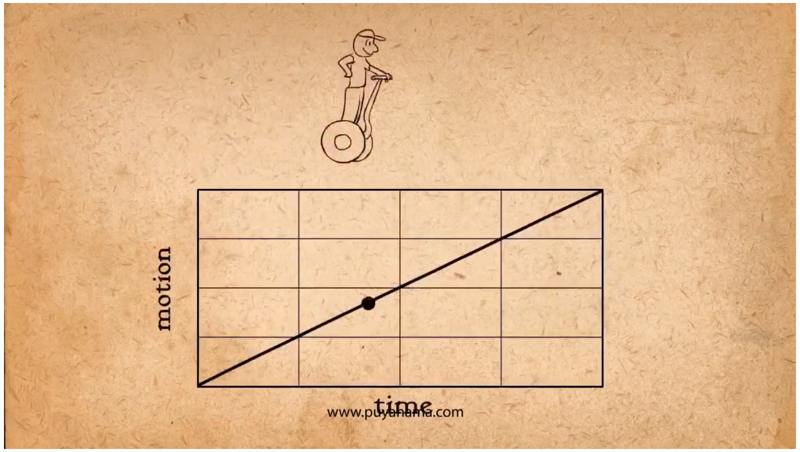
Fundamentals of Computer Graphics - Shirley Page:365

Slow-In-Slow-Out

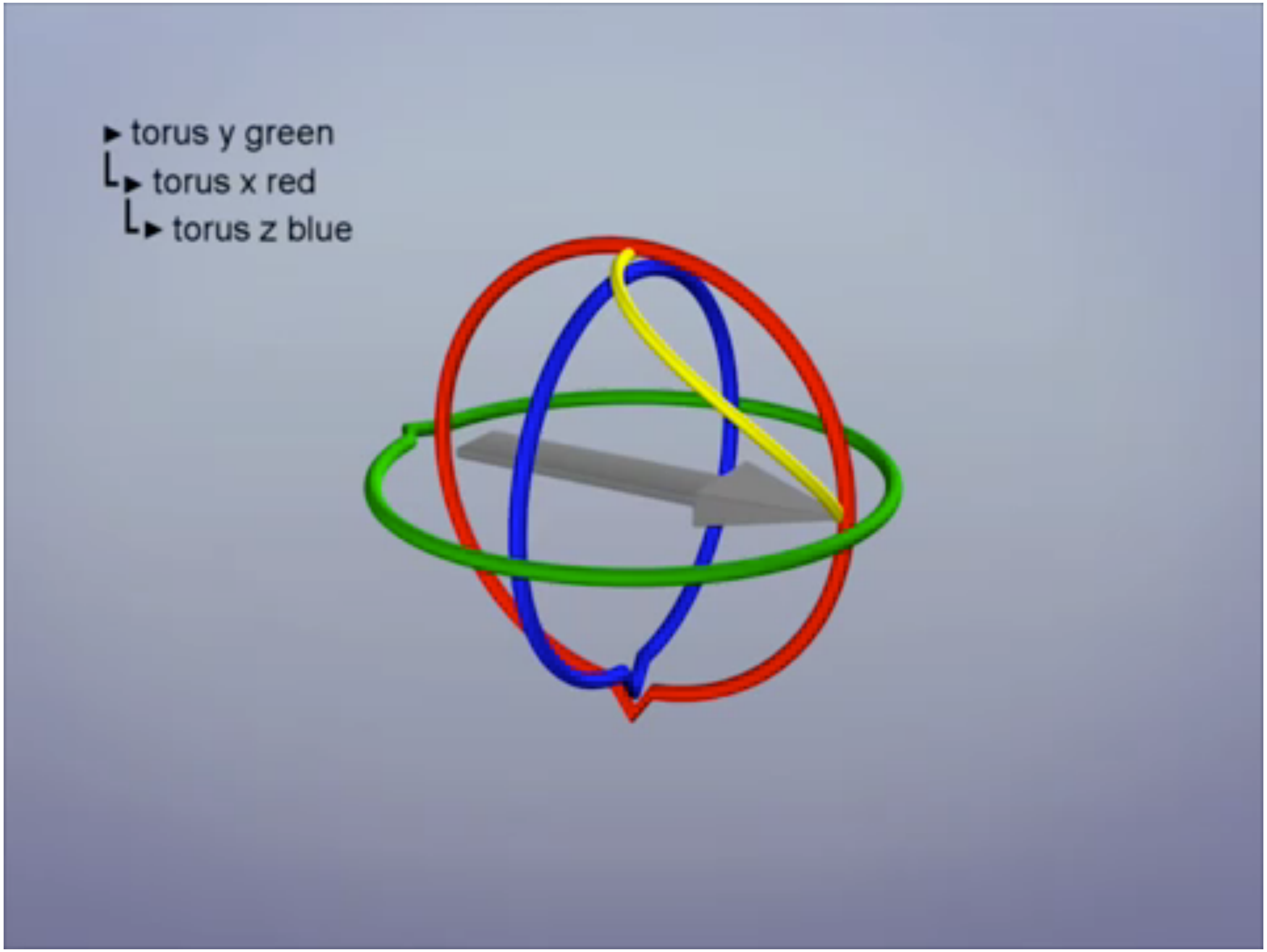


Replace an animation parameter t with $u(t)$

Bezier Curve is tunable way to implement this



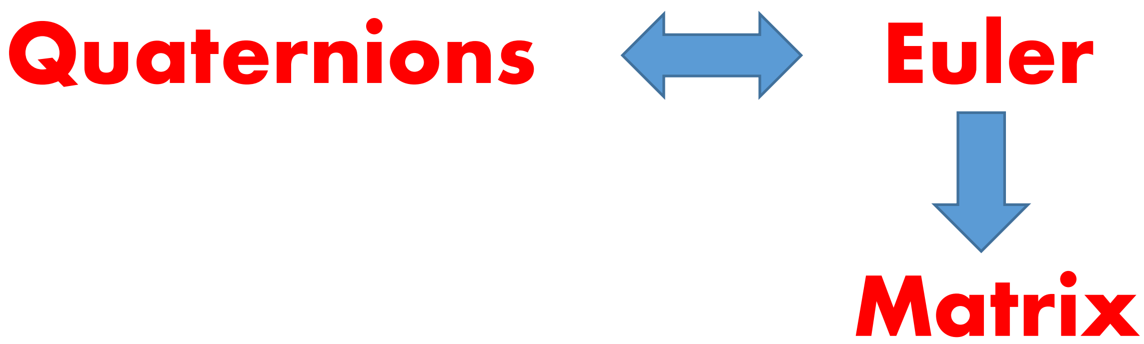
Gimbal Lock



Quaternions

$$q = [q_0, q_1, q_2, q_3]^T$$

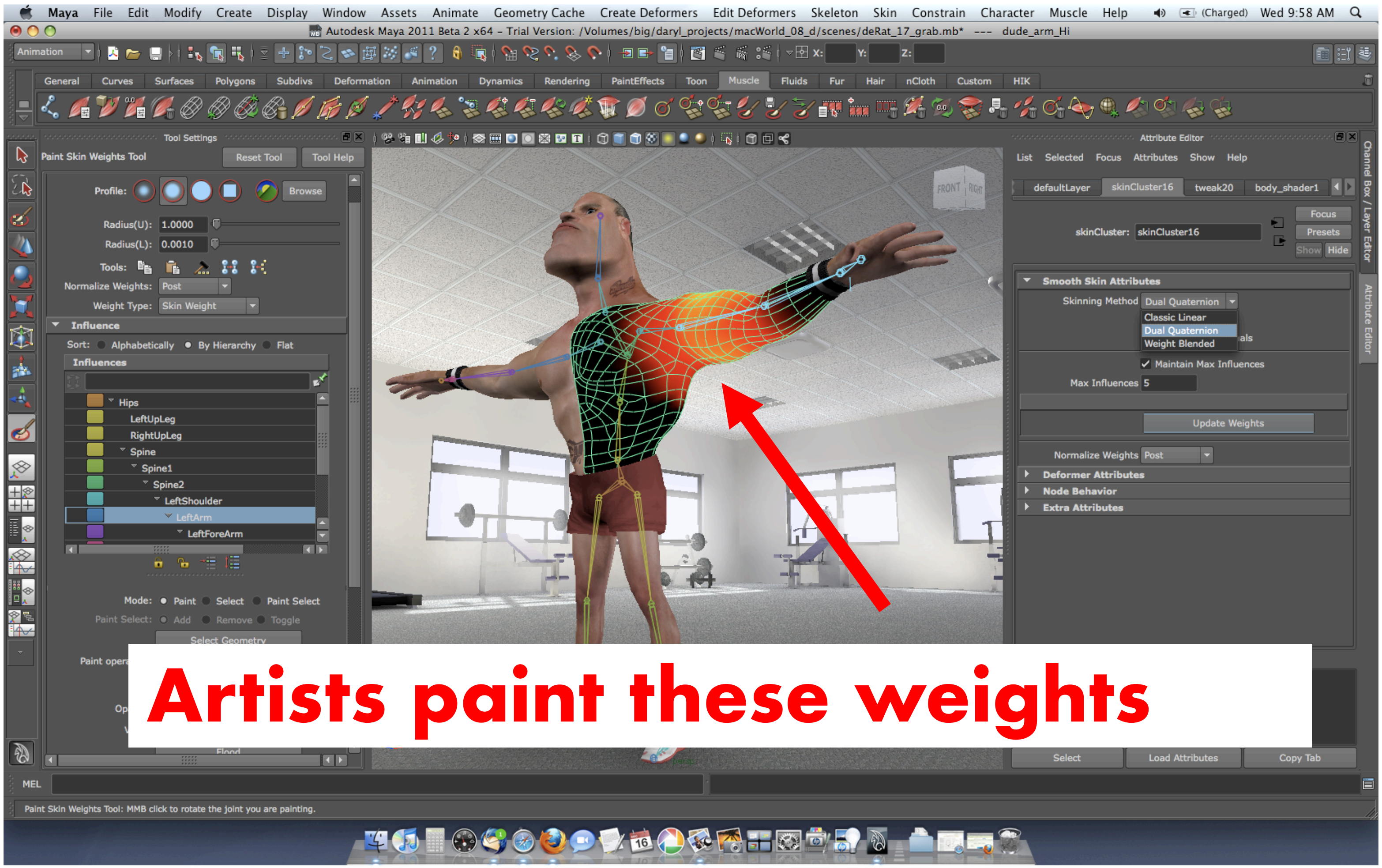
$$|q|^2 = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$$



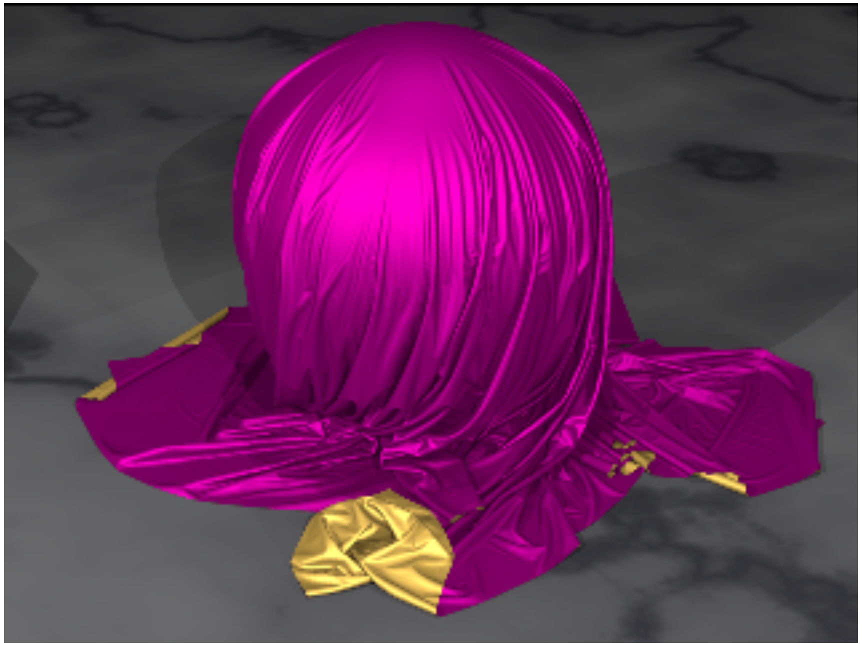
For conversion formulae

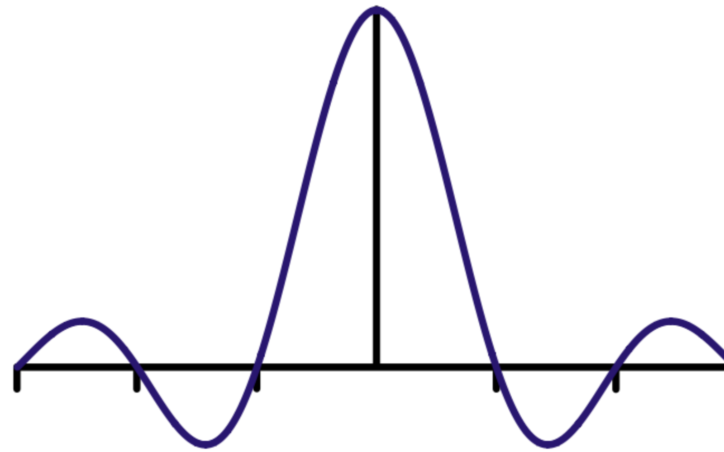
https://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles

Linear Blend Skinning



Maya

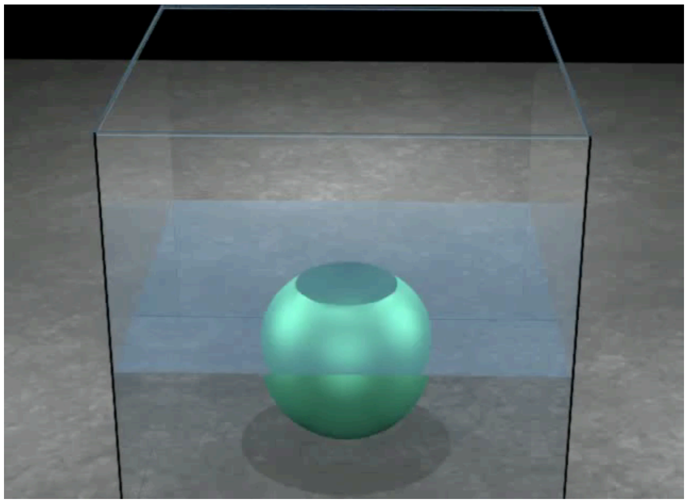


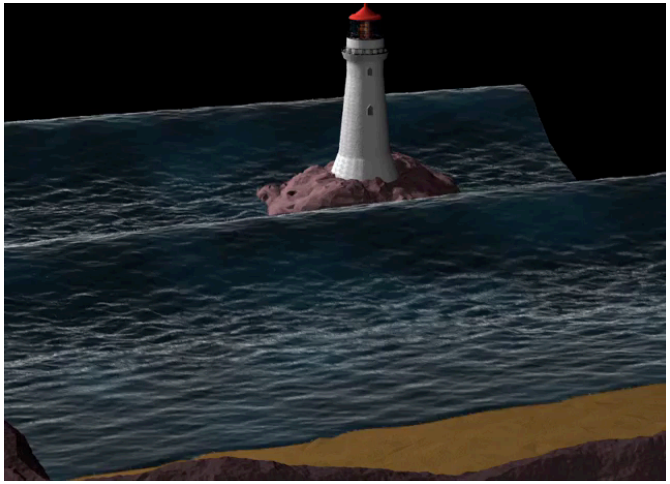


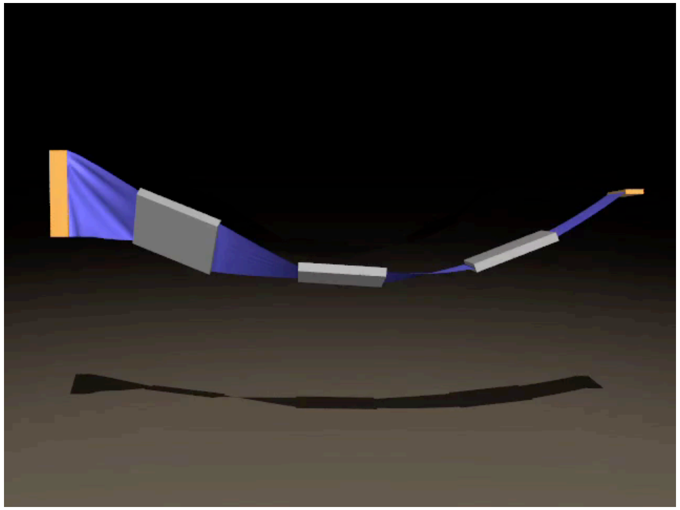
Week 7

Physically Based Animation and Sampling

Physically Based Simulations



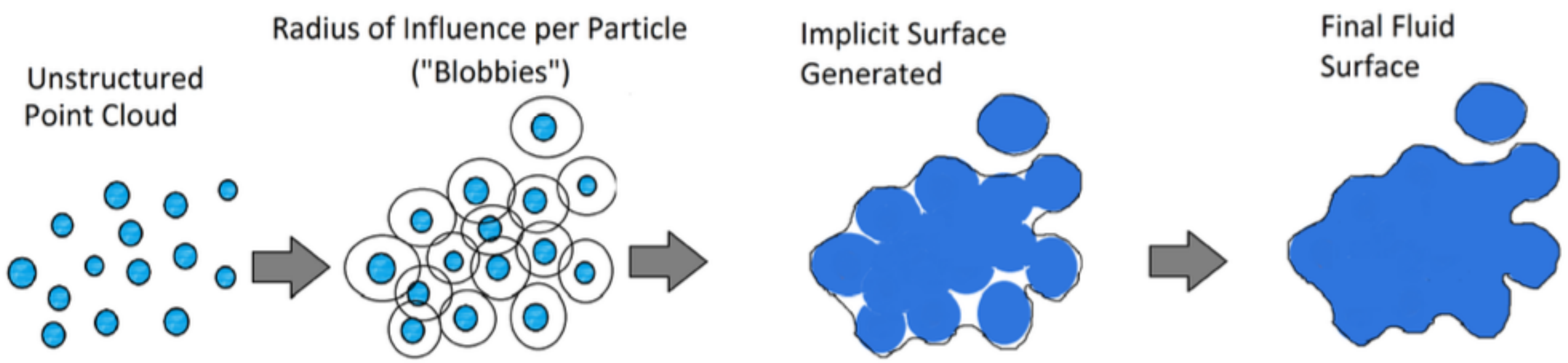




Lagrangian: Particle Based Fluid

Real fluid is not made of particles

Can't simulate infinite number of small particles



Eulerian: Grid Based Fluid

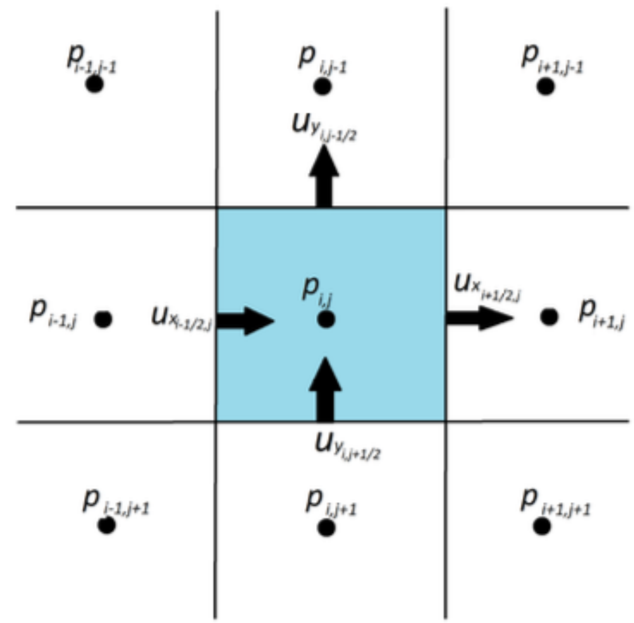
Choosing a time step

CFL condition:

$$\Delta t = \frac{\Delta h}{\vec{u}_{max}}$$

(constants out in front?)

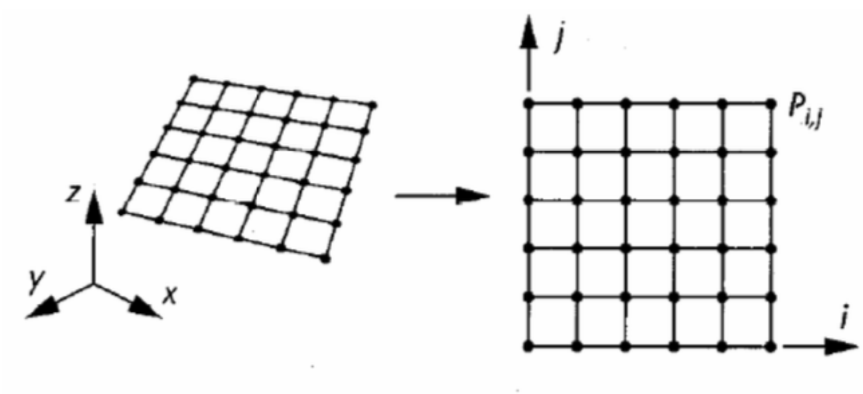
(adaptive time steps?)

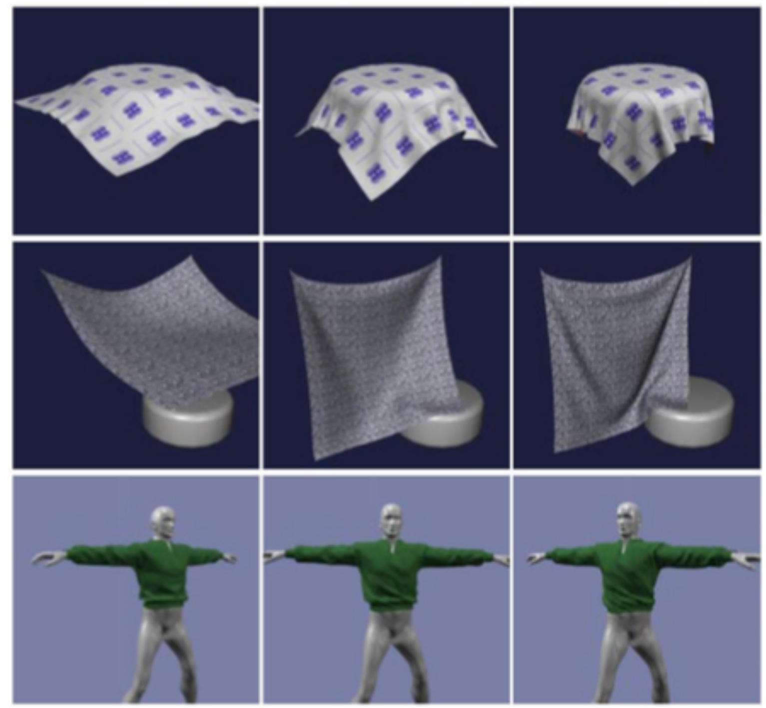


Fluid Sim: Comparison

	Particle-Based	Grid-Based
Speed?	**Faster**	Slower
Parallelization?	**Trivial**	Non-trivial
Accuracy?	Less accurate	**More accurate**
Visual appearance?	Worse	**Better**

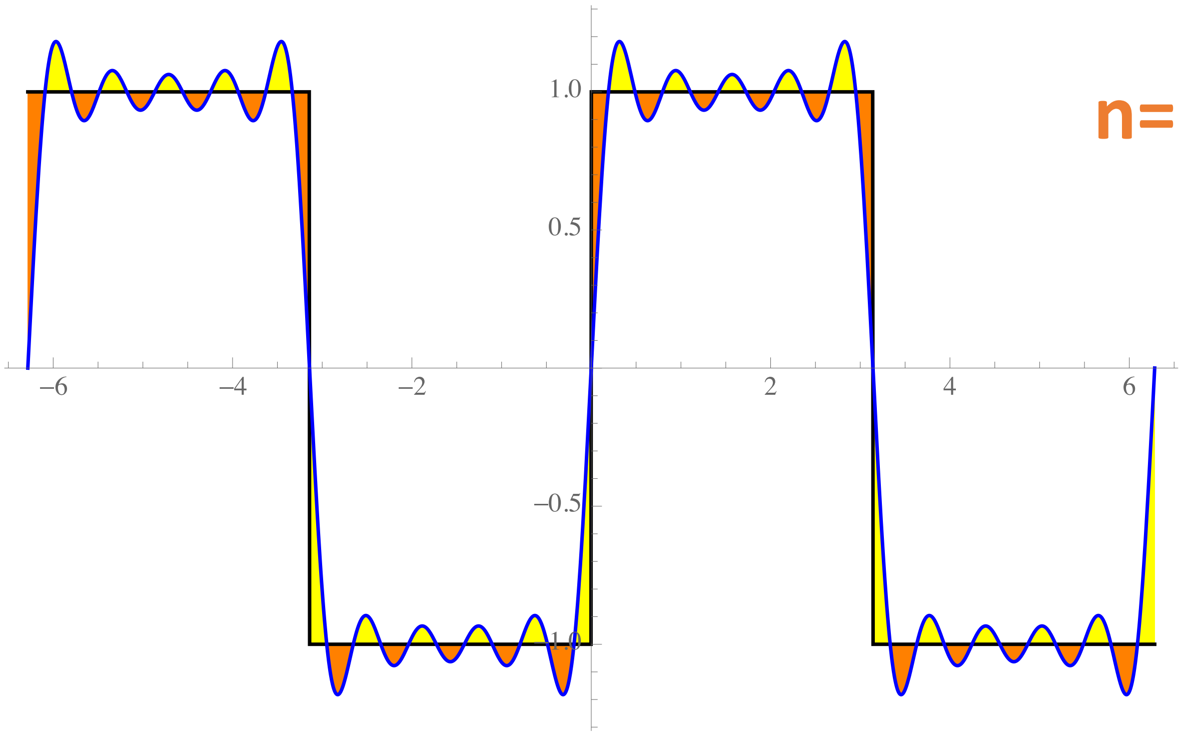
Cloth Simulation: Mass-Spring





Fourier Series

$$g(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega_o x) + \sum_{n=1}^{\infty} b_n \sin(n\omega_o x)$$



n=10

Fourier Series: Coefficients

$$g(x) = \sum_{n=-\infty}^{\infty} A_n e^{in\omega_0 x}$$

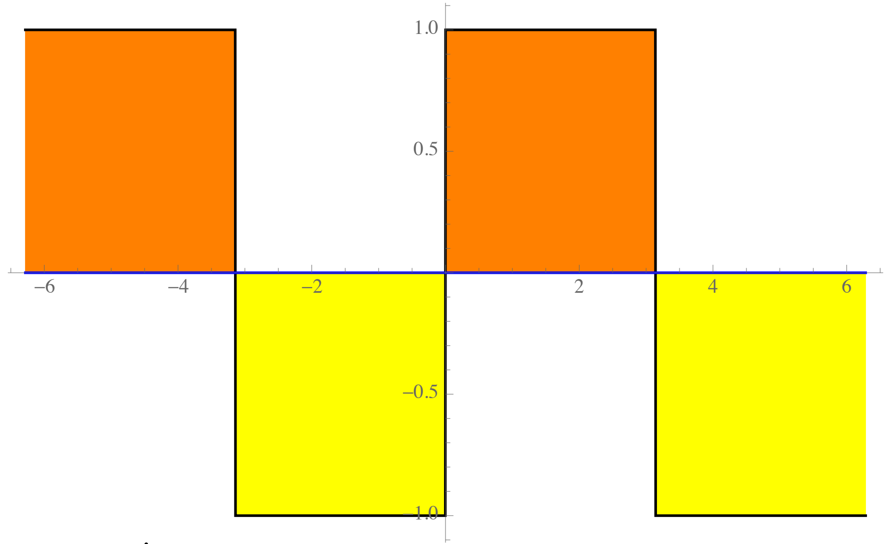
$$A_n = \frac{\langle g(x), e^{in\omega_0 x} \rangle}{\langle e^{in\omega_0 x}, e^{in\omega_0 x} \rangle} = \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} g(x) e^{-in\omega_0 x} dx$$

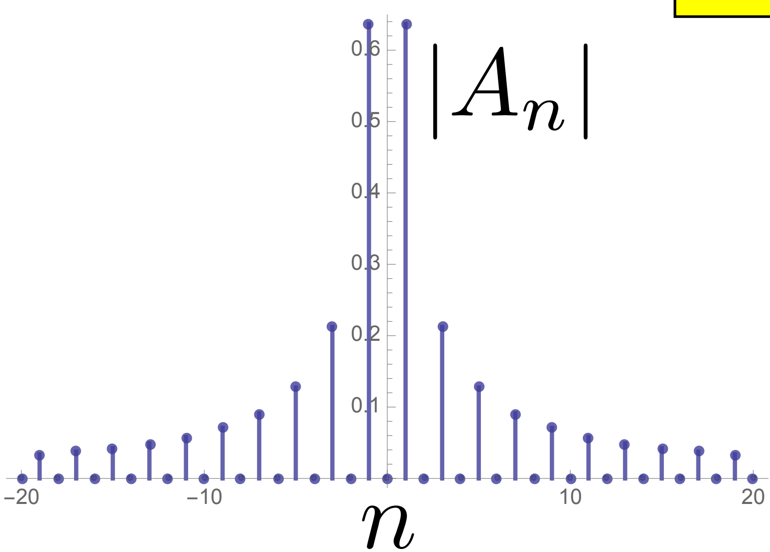
Fourier Transform: Operator Form

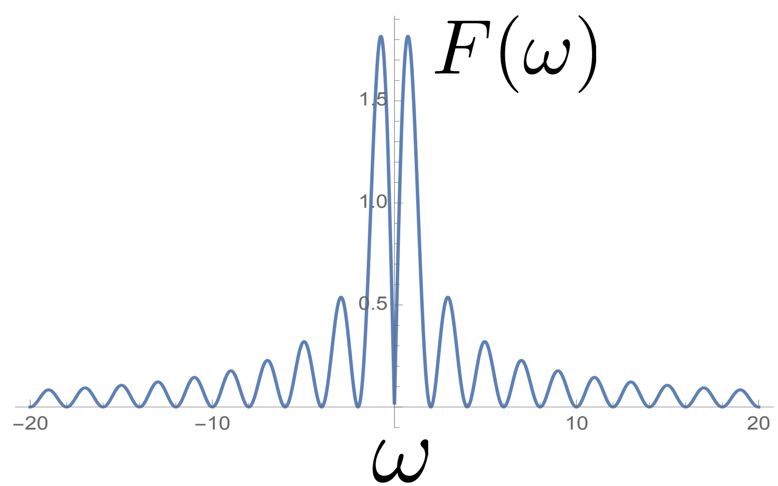
$$\mathcal{F}\left[g(x)\right] = \frac{1}{2\pi}\int_{-\infty}^{\infty} g(x)e^{-i\omega x}dx$$

$$\mathcal{F}^{-1}\left[F(\omega)\right] = g(x) = \int_{-\infty}^{\infty} F(\omega)e^{i\omega x}d\omega$$

Fourier Transform: Examples

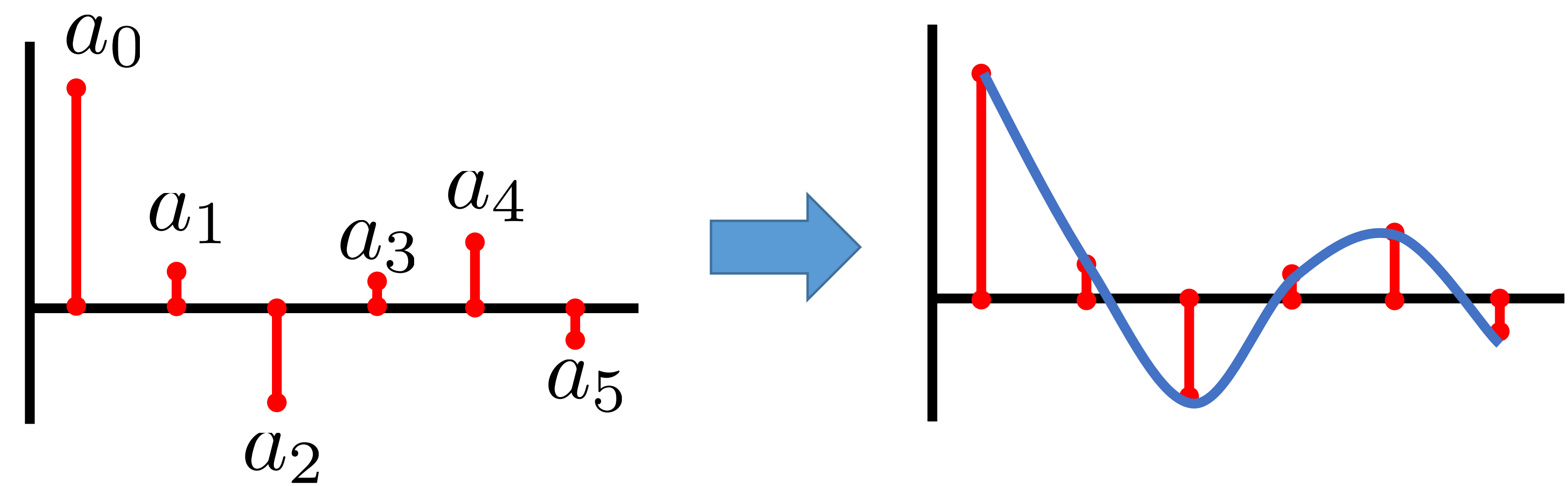






Convolution: Avenue to Continuous

Discrete Function | **Continuous Function**



$$\{a_0, a_1, a_2 \ldots a_n\}$$

$$\hat{g}(x) = \sum_i a_i k(x - i)$$

Fourier Transform and Convolution

$$g = f * k$$

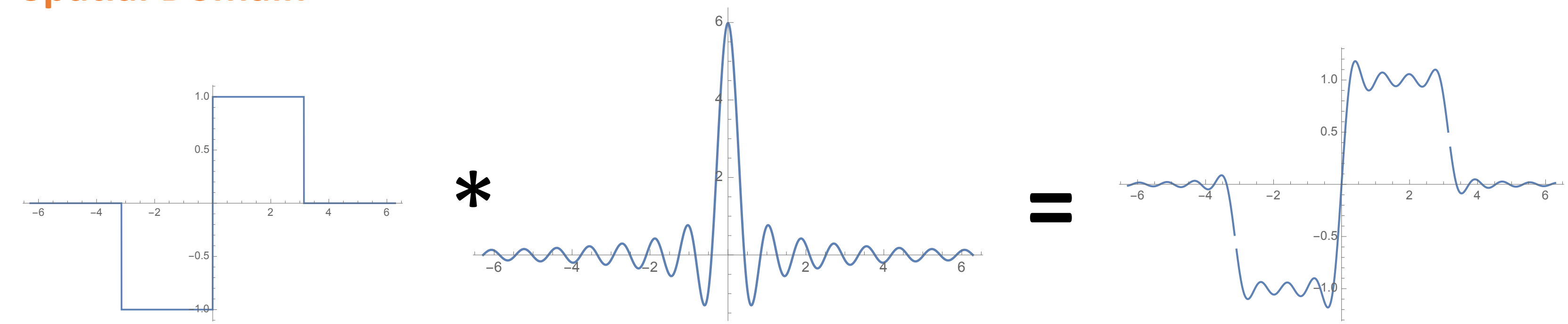
$$\implies \mathcal{F}[g] = \mathcal{F}[f] \cdot \mathcal{F}[k]$$

$$G(\omega) = F(\omega) \cdot K(\omega)$$

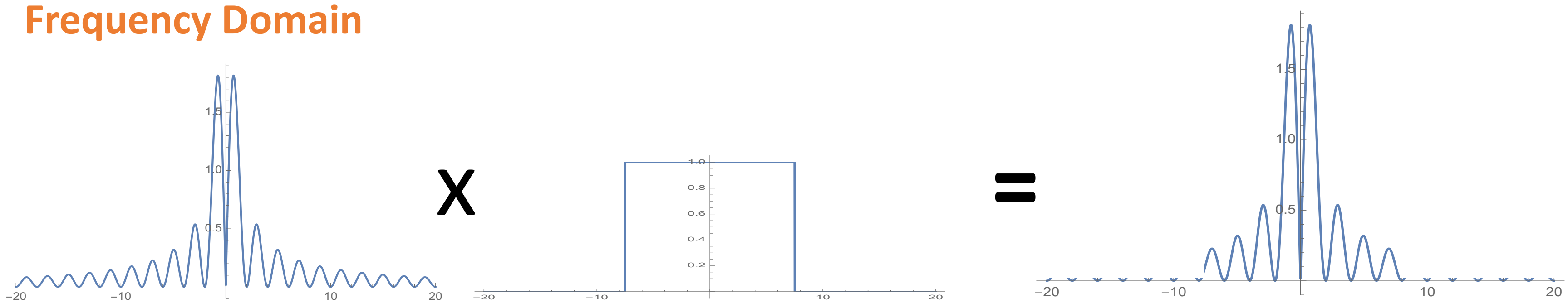
Convolution turns into simple multiplication in the Fourier/Frequency domain

Example

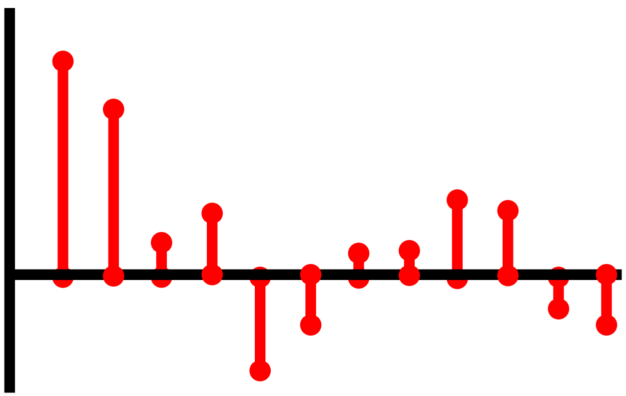
Spatial Domain

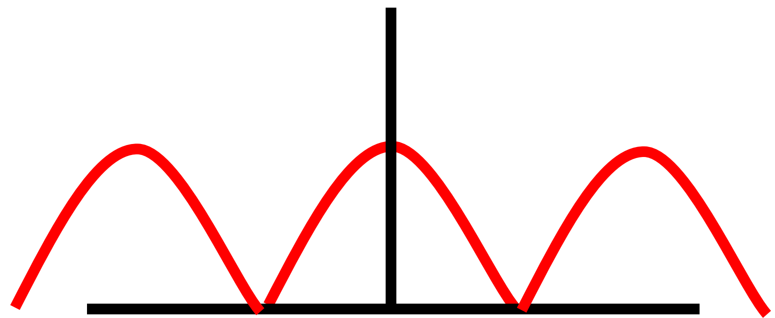


Frequency Domain



Optimal Sampling



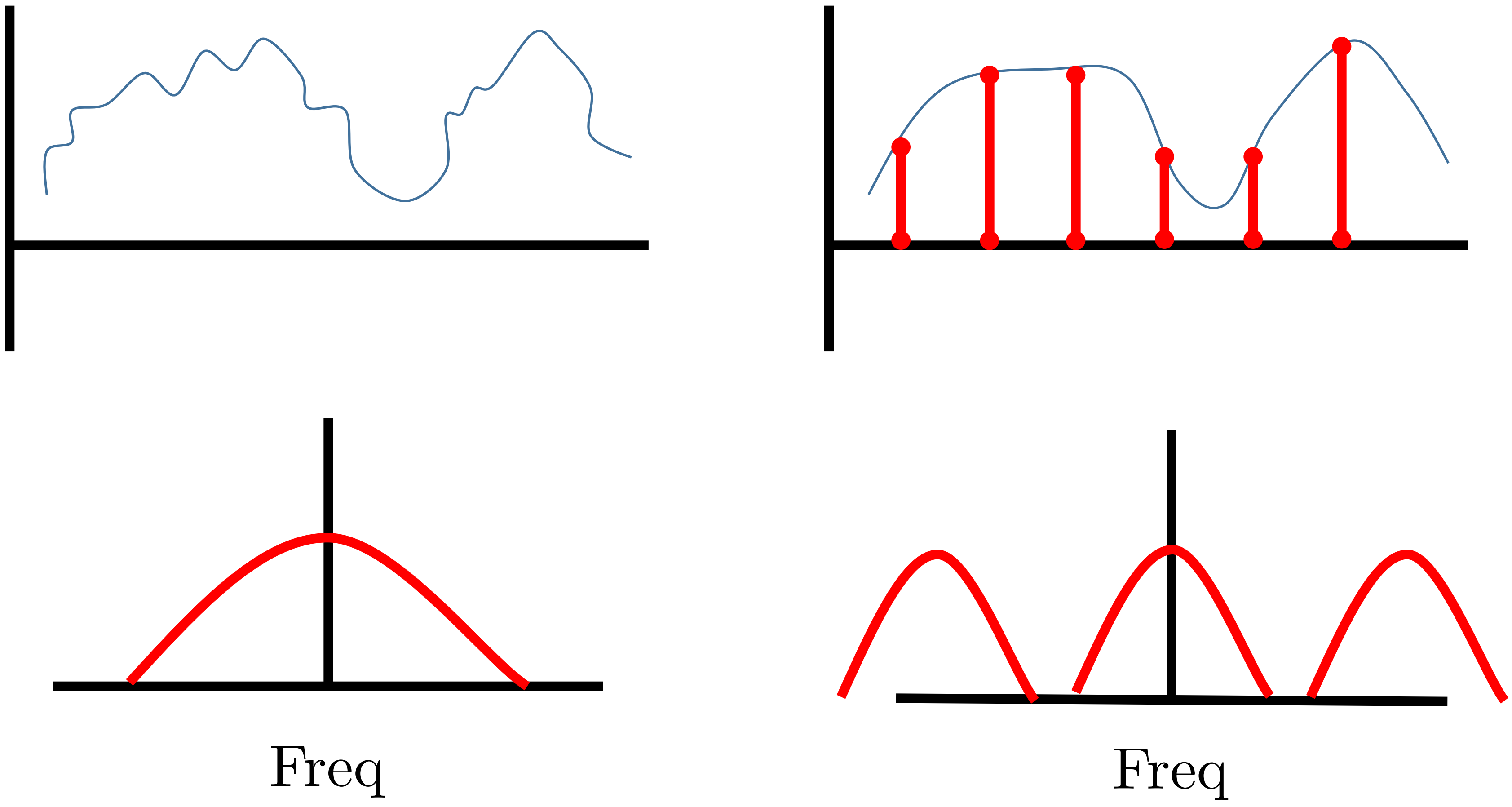


Copies of F(f) are just touching

$$\frac{1}{T} = \text{Sampling Rate} = 2 \times \text{Bandwidth}$$

Nyquist Citerion/Theorem

Sampling/Subsampling



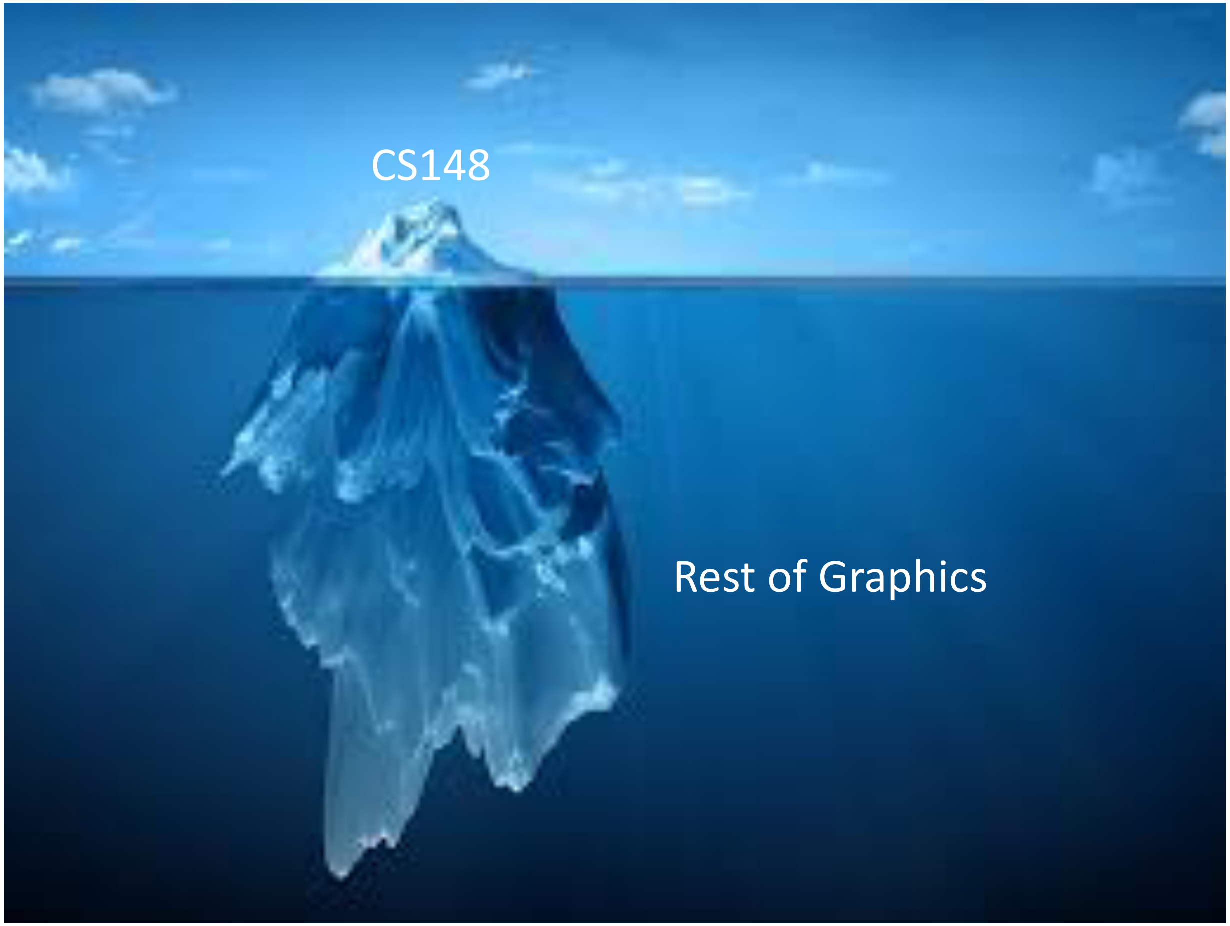
Sample after low-pass filtering: no aliasing

What's Next ?

What’s Next ?

- **CS248 - Make a game**
- CS268 – Geometric Algorithm
- **CS348a - Geometric Modeling**
- **CS348b - Photorealistic Images**
- CS348c - Physically based Simulation
- CS368 – Advanced Geometrical Algorithm

Graphics: A Humungous Field



Video Games

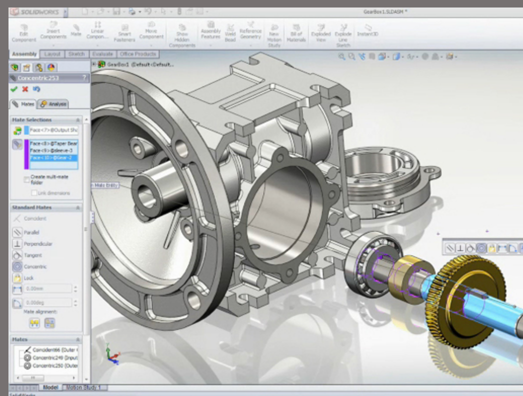
Gears of War 3 (2011), Unreal

Movies

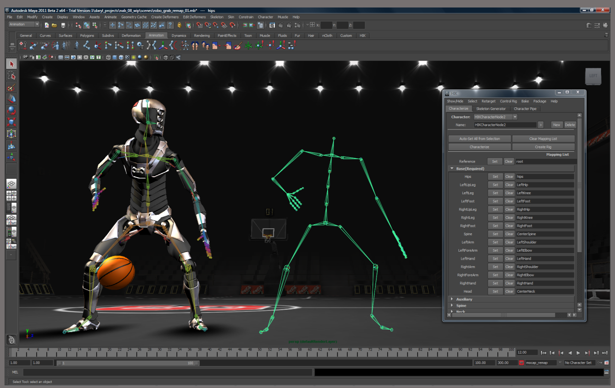
Day After Tomorrow (2004)

Academy Awards!

CADs, Content Creators

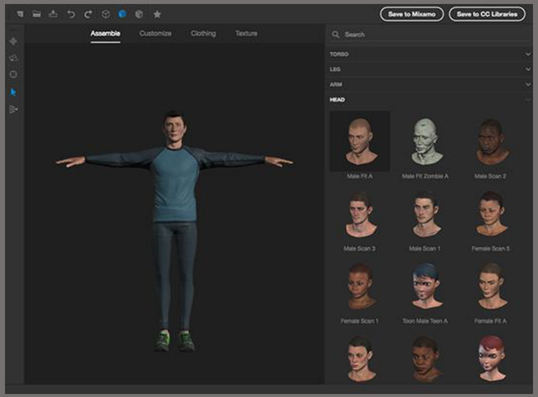


Solidworks, Dassault Systemes



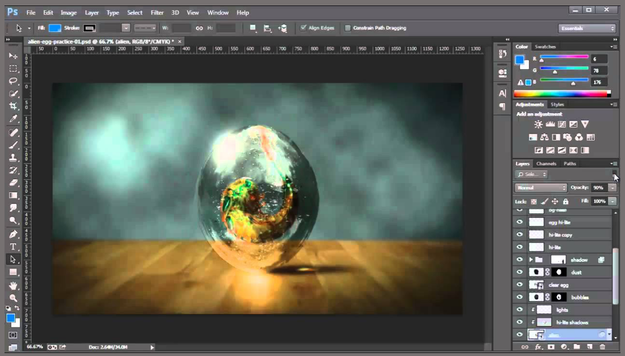
Maya, Autodesk

ZBrush, Pixologic



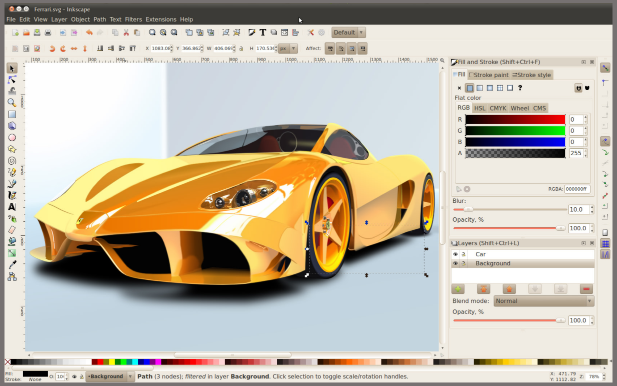
Fuse, Adobe

Computational Photography



Photoshop, Adobe

Aperture, Apple

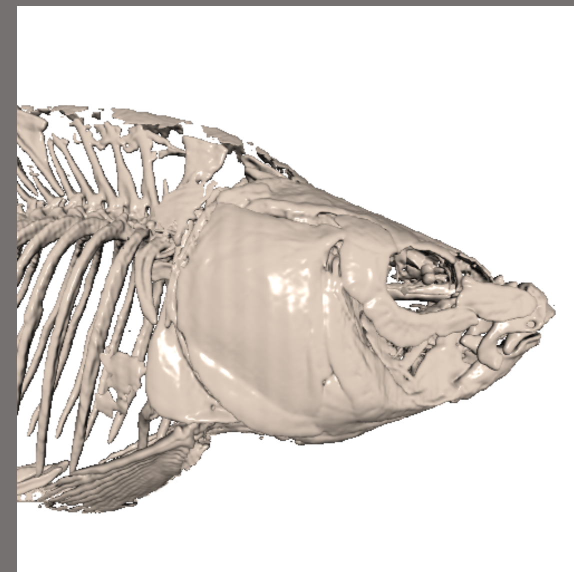


GIMP

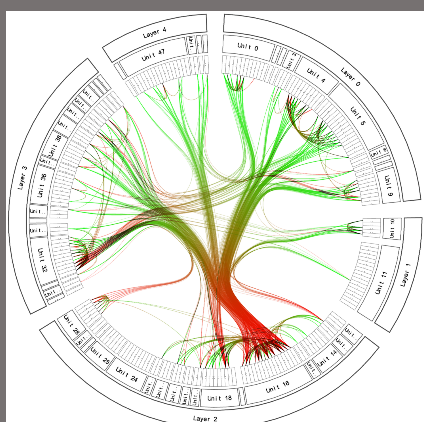
ILLUM, Lytro

Visualization

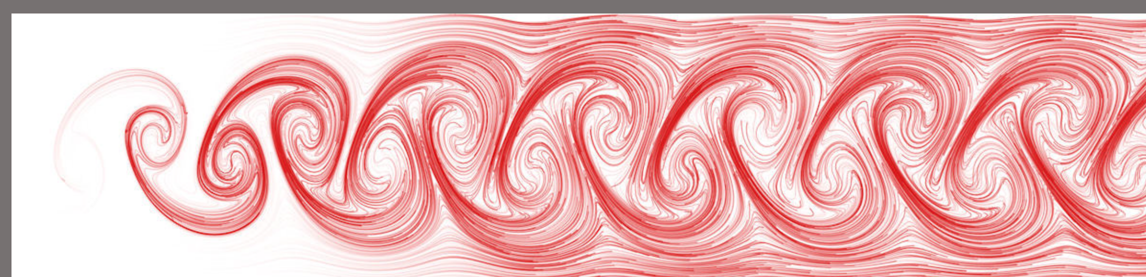
CT scan of Tamut, CNN 2014



Z. Hossain and T. Möller 2011



Hierarchical Edge Bundling



T. Weinkauf and H. Theisel 2010

*http://www.win.tue.nl/vis1/home/dholten/

Virtual and Augmented Reality

Occulus Rift, Occulus 2016

Hololens, Microsoft 2016

Special Thanks

- VMWare – for VMWare Fusion 8 License Keys
- Lecture Materials:
  - Dr. Justin Solomon (now a faculty member at MIT)
  - Katherine Breeden (at Stanford, PhD candidate)
  - Dr. Mirela Ben-Chen (now at Technion, Israel)
  - Dr. Ronald Fedkiw (regular professor of CS148)
  - Dr. Pat Hanrahan (past professor of CS148)
  - David Hyde (for Physically Based Animation)

CS 148: Introduction to Computer Graphics and Imaging

Zahid Hossain

PhD Candidate

Computer Science & Bioengineering

Teaching Assistants:

David Hyde

Minjae Lee

Jason Riggs