CSC317 Computer Graphics
Tutorial 10

November 29, 2023

Final Review Session

- Final Exam: December 6 @ 11:00 am (during tutorial)
- Format: similar to midterm

Lecture 1: Raster Images

Gamma Correction

Gamma

$$\text{displayed intensity} = (\text{maximum intensity})a^{\gamma}$$

… of display

Amplitude from Image [0,1]

Lecture 1: Raster Images

Gamma Correction

Measure: Find image amplitude that = ½ display brightness

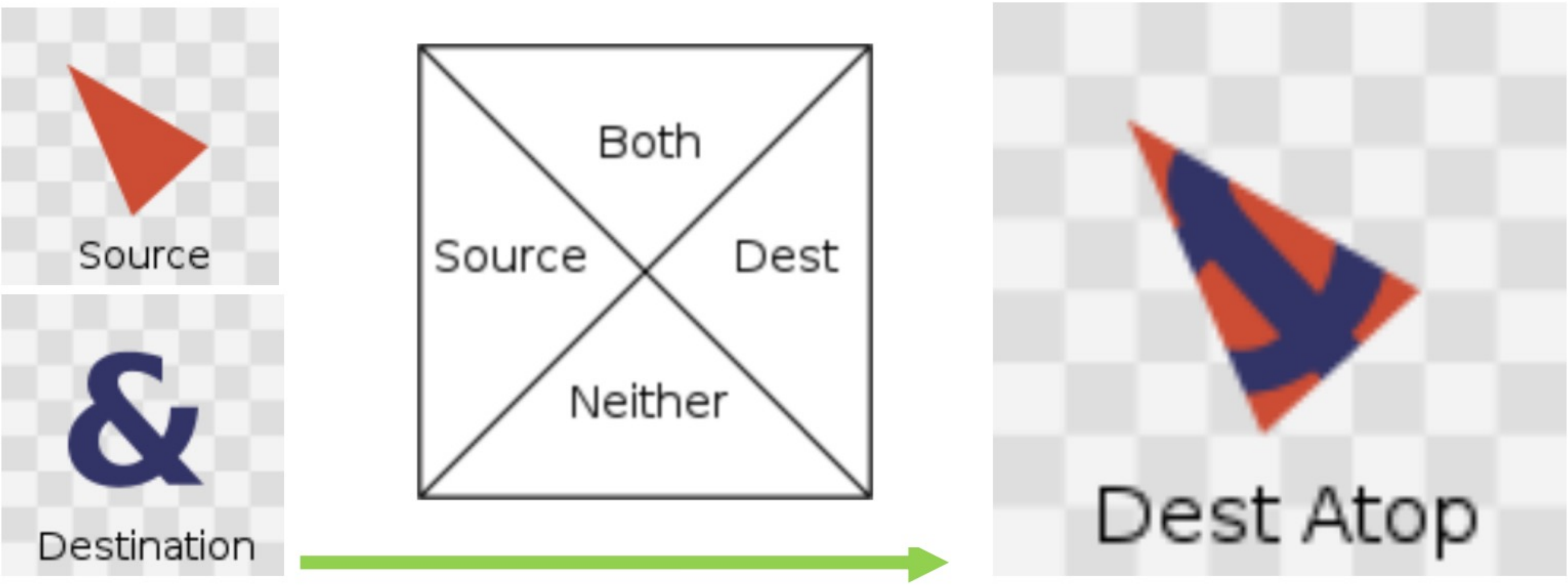
$$0.5 = a^\gamma$$

Fit model

$$\gamma = \frac{\ln 0.5}{\ln a}$$

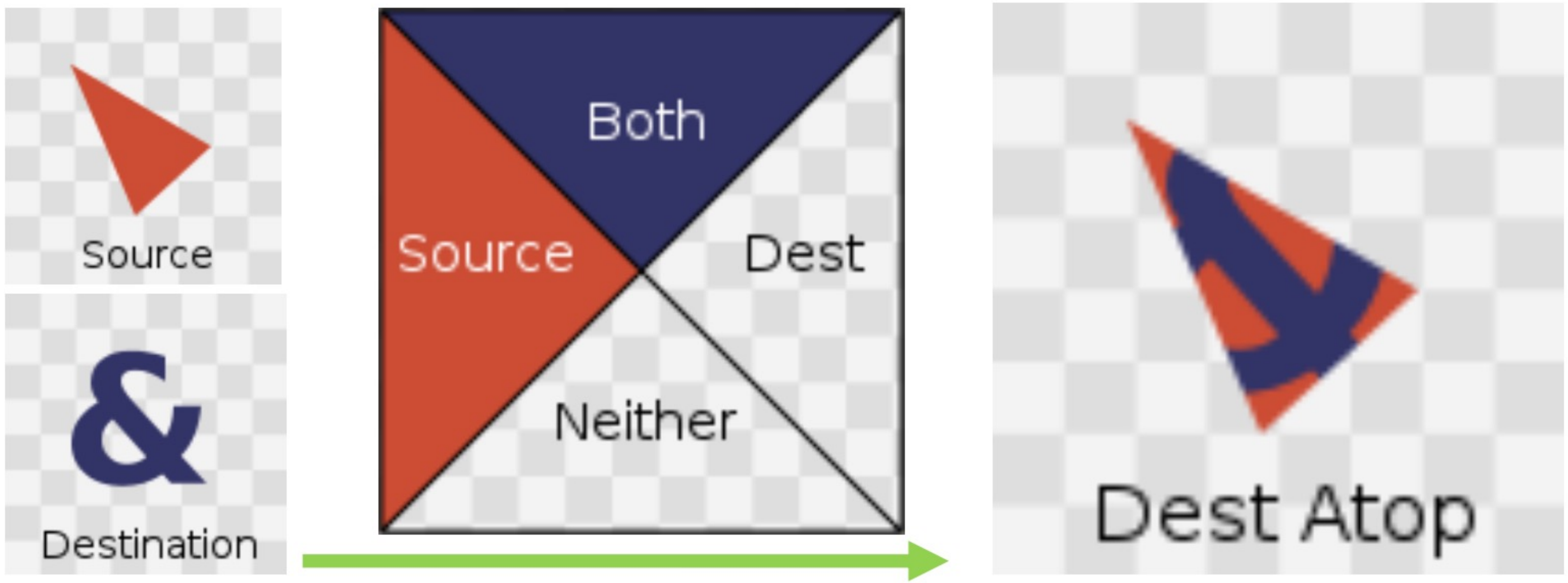
Lecture 1: Raster Images

Image Composition



Lecture 1: Raster Images

Image Composition



Lecture 1: Raster Images

Image Composition

iii– [2 MARKS] If all pixels in A are pure red, all pixels in B are pure blue, and $\alpha_A = 0.75$ and $\alpha_B = 1$. What will be the RGBα values of A over B:

$r_o =$

$g_o =$

$b_o =$

$\alpha_o =$

Lecture 1: Raster Images

Image Indexing

ii– [4 MARKS] Using a 1D array, `unsigned char rgb[]`, we can store pixels values running first over RGB channels, then across the width, and finally over the height.

Write a C++ routine to create a new image and loop over all pixel values and set them to 100% red, 20% green, and 0% blue.

Lecture 1: Raster Images

Image Color Space Conversion

i– [2 MARKS]

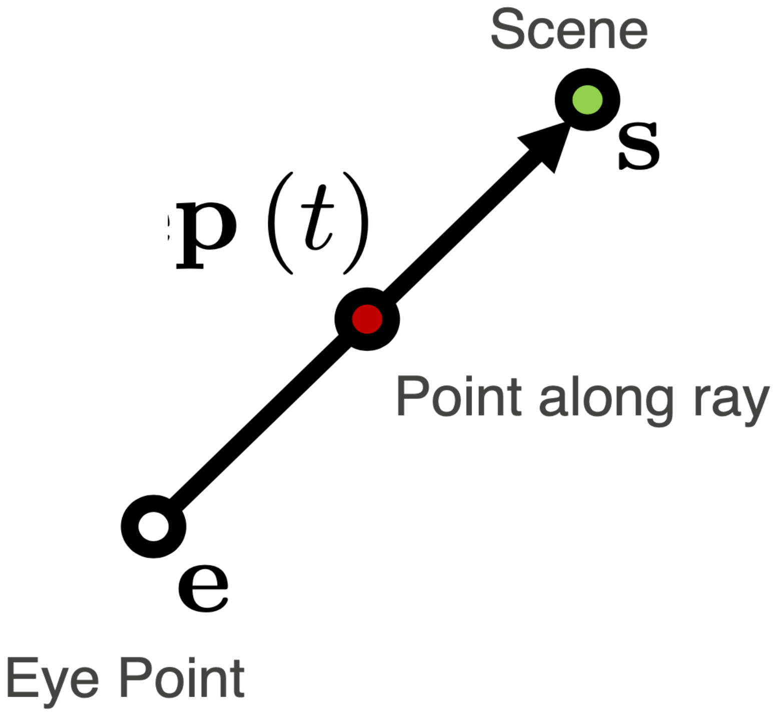
Oops! We have the correct code for converting an RGB image to a hue-saturation-value (HSV) image but the left-hand sides are missing. Write the *h*, *s* or *v* in the correct places on the last 3 lines below:

```
void rgb_to_hsv(
  const double r, const double g, const double b,
  double & h, double & s, double & v)
{
  const double M = std::max(std::max(r,g),b);
  const double m = std::min(std::min(r,g),b);
  // chroma
  const double C = M-m;
  double z = 0;
  if(M == r)
  {
    z = std::fmod((g-b)/C,6);
  }else if(M == g)
  {
    z = (b-r)/C + 2;
  }else //(M == b)
  {
    z = (r-g)/C+4;
  }
  ____ = M;
  ____ = M==0 ? 0 : C/M;
  ____ = 60*z;
}
```

Lecture 2: Ray Casting

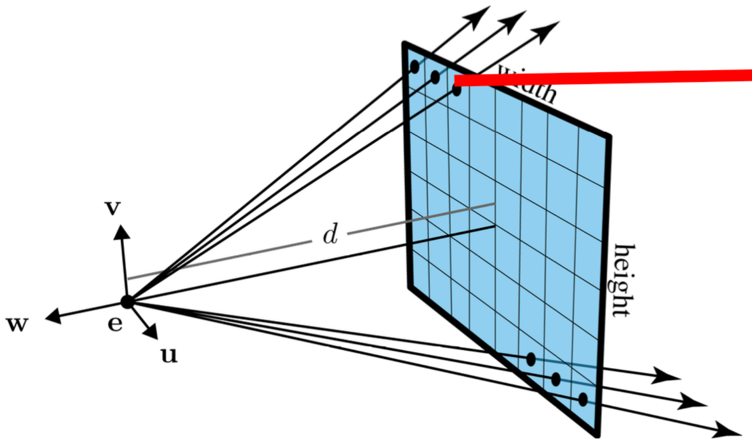
Ray Definition

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$



Lecture 2: Ray Casting

Ray Definition



$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

$$\mathbf{p}(t) = t \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{bmatrix} \begin{bmatrix} u(i) \\ v(j) \\ -d \end{bmatrix} + \mathbf{e}$$

Camera Transformation Matrix

Lecture 2: Ray Casting

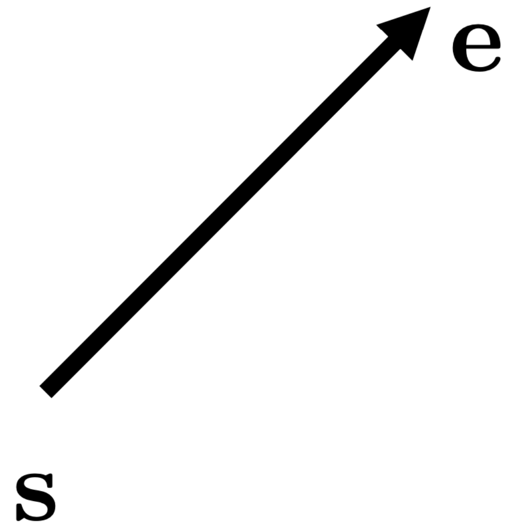
Intersection Tests

Ray-Plane Intersection

$$\mathbf{n}^T \mathbf{p}\left(t\right) - q = 0$$

$$\boxed{t = \frac{q - \mathbf{n}^T \mathbf{e}}{\mathbf{n}^T \left(\mathbf{s} - \mathbf{e}\right)}}$$

$$\mathbf{p}(t) = \mathbf{e} + t(\mathbf{s} - \mathbf{e})$$

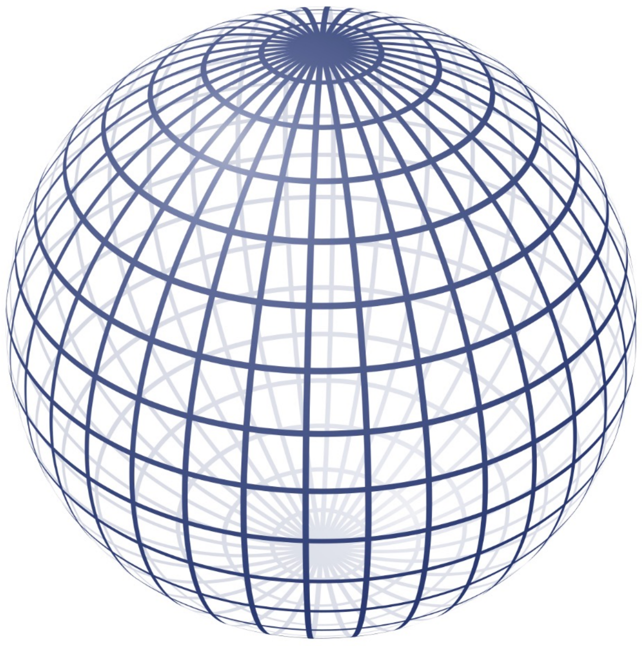


Lecture 2: Ray Casting

Intersection Tests

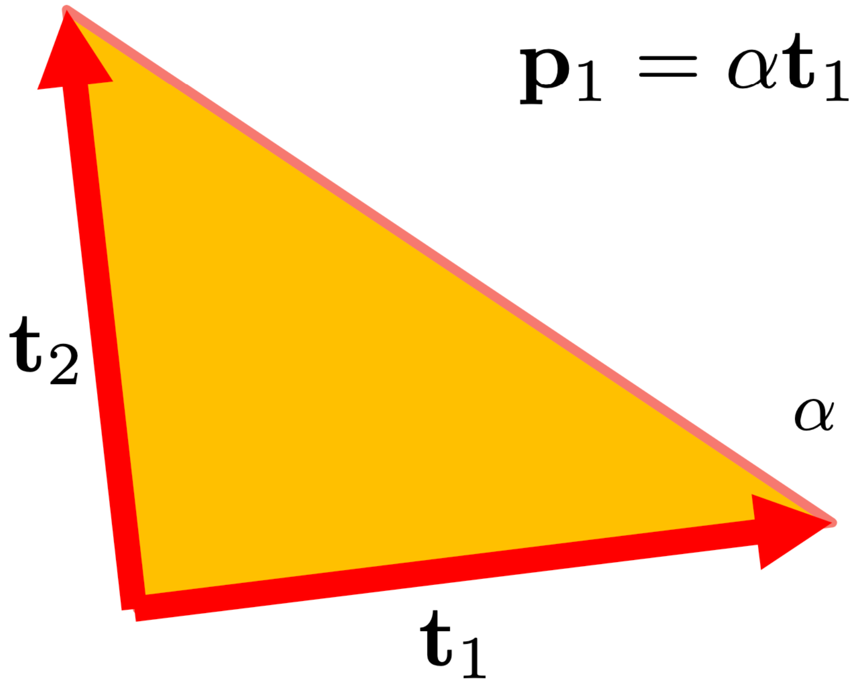
$$\mathbf{p}(t)^T \mathbf{p}(t) - r^2 = 0$$

$$\boxed{\begin{aligned} & a \cdot t^2 + b \cdot t + c = 0 \\ & a = (\mathbf{s} - \mathbf{e})^T (\mathbf{s} - \mathbf{e}) \\ & b = 2\mathbf{e}^T (\mathbf{s} - \mathbf{e}) \\ & c = \mathbf{e}^T \mathbf{e} - r^2 \end{aligned}}$$



Lecture 2: Ray Casting

Intersection Tests



$$\mathbf{p}_1 = \alpha\mathbf{t}_1 + \beta\mathbf{t}_2$$

$$\alpha \geqslant 0$$

$$\beta \geqslant 0$$

$$\alpha + \beta \leqslant 1$$

Lecture 2: Ray Casting

Intersection Tests

$$\mathbf{p}\left(t\right) = \alpha\mathbf{t}_1 + \beta\mathbf{t}_2$$

$$\mathbf{e} + t\left(\mathbf{s} - \mathbf{e}\right) = \alpha\mathbf{t}_1 + \beta\mathbf{t}_2$$

$$\alpha\mathbf{t}_1 + \beta\mathbf{t}_2 - t\left(\mathbf{s} - \mathbf{e}\right) = \mathbf{e}$$

Lecture 2: Ray Casting

Intersection Tests

$$\alpha \mathbf{t}_1 + \beta \mathbf{t}_2 - t\left(\mathbf{s} - \mathbf{e}\right) = \mathbf{e}$$

$$\begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & -\left(\mathbf{s} - \mathbf{e}\right) \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ t \end{bmatrix} = \mathbf{e}$$

Check t, α and β

Lecture 2: Ray Casting

Intersection Tests

Consider a ray emanating from a 3D position $\mathbf{e} \in \mathbb{R}^3$ in the direction defined by a 3D vector $\mathbf{d} \in \mathbb{R}^3$ and a sphere centred at $\mathbf{c} \in \mathbb{R}^3$ with radius r.

i– [3 MARKS] Using the variables above, solve for *parametric distances* t for any/all candidate ray-sphere intersections.

ii– How can we determine if the ray *never* hits the sphere? [1 MARK]

iii– [1 MARK] Given a ray-sphere hit at *parametric distance* t_{hit}, what is the *unit normal* of the sphere at the intersection point?

Lecture 3: Ray Tracing

- Lights

Two types of lights:

Directional Light:

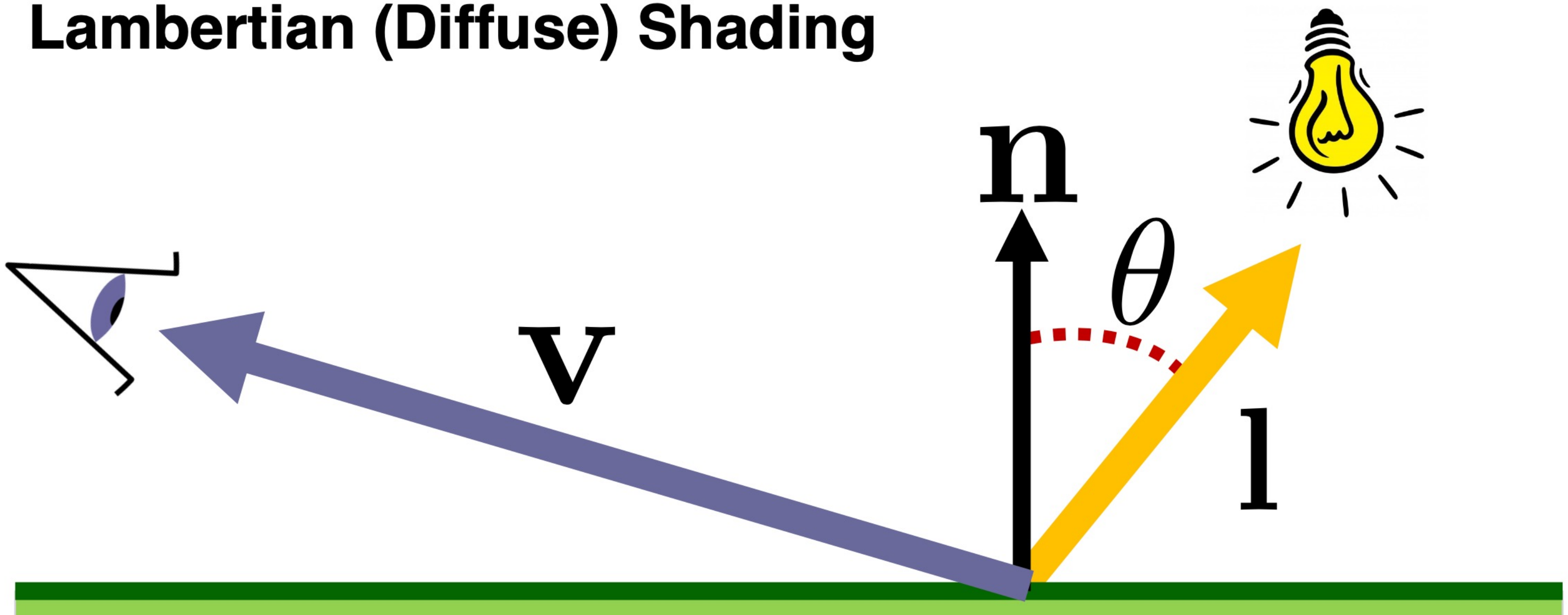
Direction of light does not depend on the position of the object. **Light is very far away**

Point Light

Direction of light depends on position of object relative to light.

Lecture 3: Ray Tracing

Lambertian (Diffuse) Shading



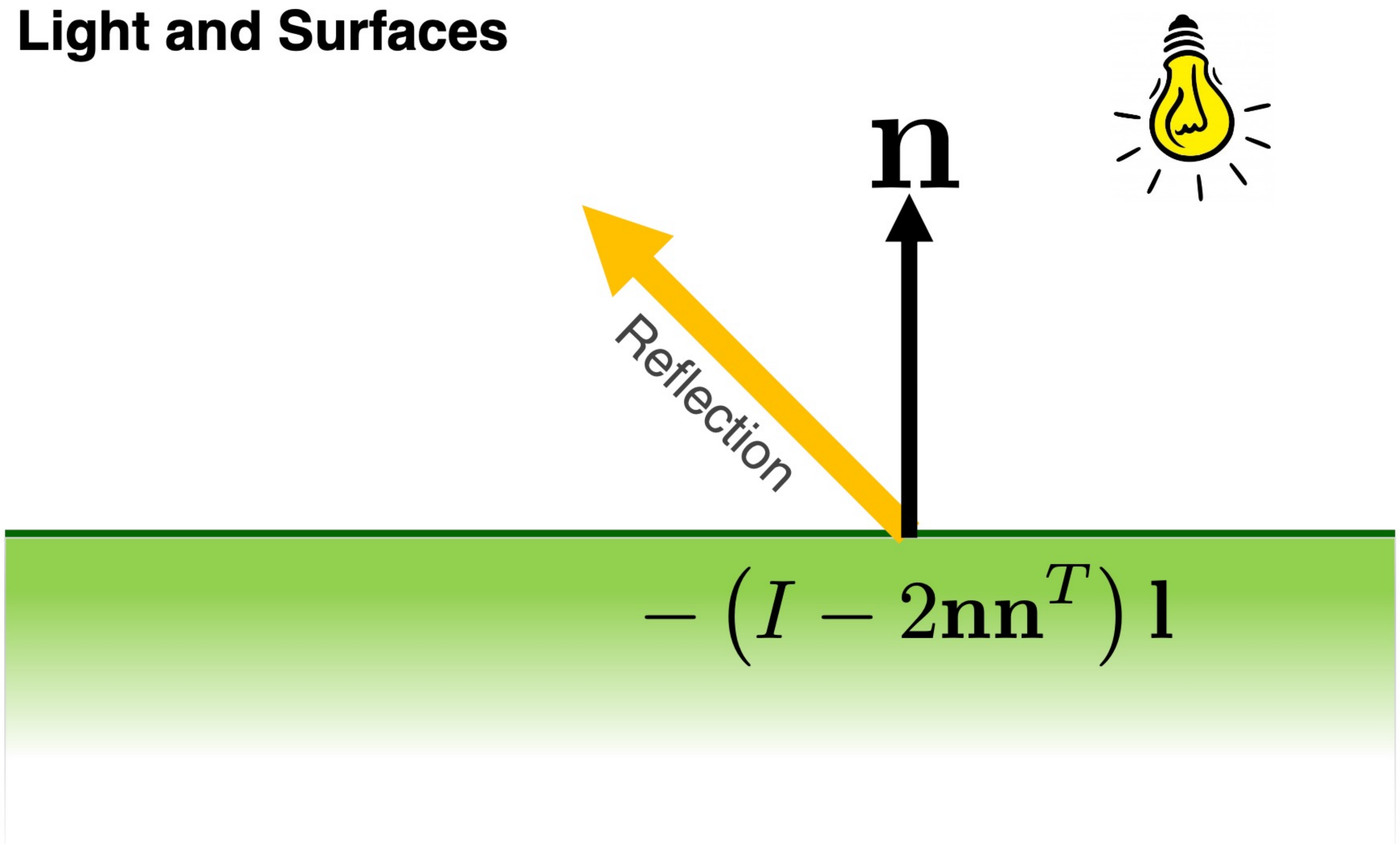
$$L = k_d \max\left(0, \mathbf{n}^T \mathbf{l}\right)$$

Lecture 3: Ray Tracing

- Full Blinn-Phong Model

$$L = k_a + k_d \max\left(0, \mathbf{n}^T \mathbf{l}\right) + k_s \max\left(0, \cos(\theta_{\mathbf{h}}^{\mathbf{n}})\right)^p$$

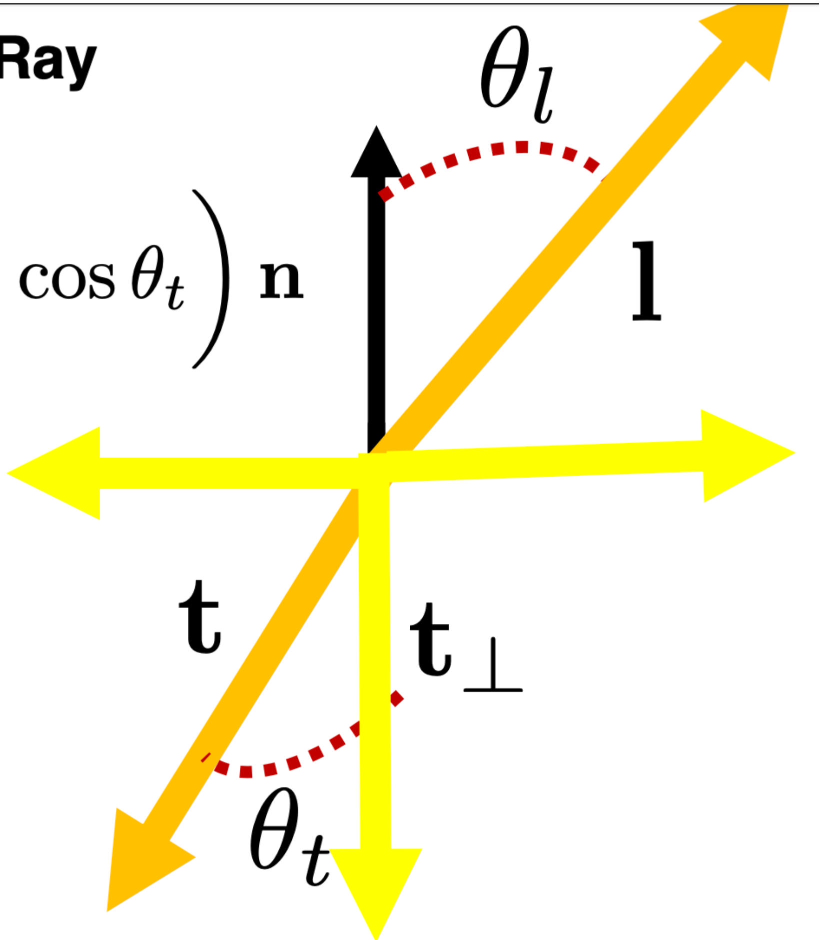
Lecture 3: Ray Tracing



Lecture 3: Ray Tracing

Computing the Refracted Ray

$$\mathbf{t} = -\frac{c_l}{c_t}\mathbf{l} + \left(\frac{c_l}{c_t}\cos\theta_1 - \cos\theta_t\right)\mathbf{n}$$



Lecture 3: Ray Tracing

Consider a point $\mathbf{p} \in \mathbb{R}^3$ on a surface with unit normal $\hat{\mathbf{n}} \in \mathbb{R}^3$, viewed from a ray emanating from the point $\mathbf{e} \in \mathbb{R}^3$ and a point light located at $\boldsymbol{\ell} \in \mathbb{R}^3$.

i– [3 MARKS] Define the specular term of Blinn-Phong shading (label any new terms), making use of the specular exponent α:

ii– In words, describe what will happen to the appearance of the shape if the specular exponent α is *decreased*? [2 MARKS]

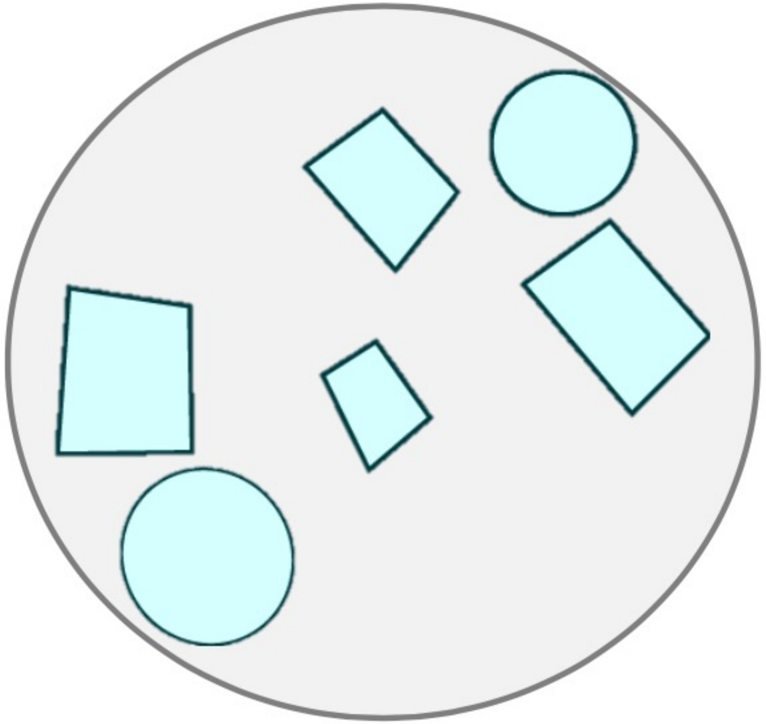
Lecture 4: Bounding Volume Hierarchy

- Bounding Volume (Sphere)

Parameters of a Sphere:

1. Center = $\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}^i$
2. Radius = $r = \max\left(\mathbf{v}^i - \mathbf{c}\right)$

$$\mathbf{v}^i \in \text{Vertices}$$



Lecture 4: Bounding Volume Hierarchy

- Bounding Volume (Axis-Aligned Bounding Box (AABB))

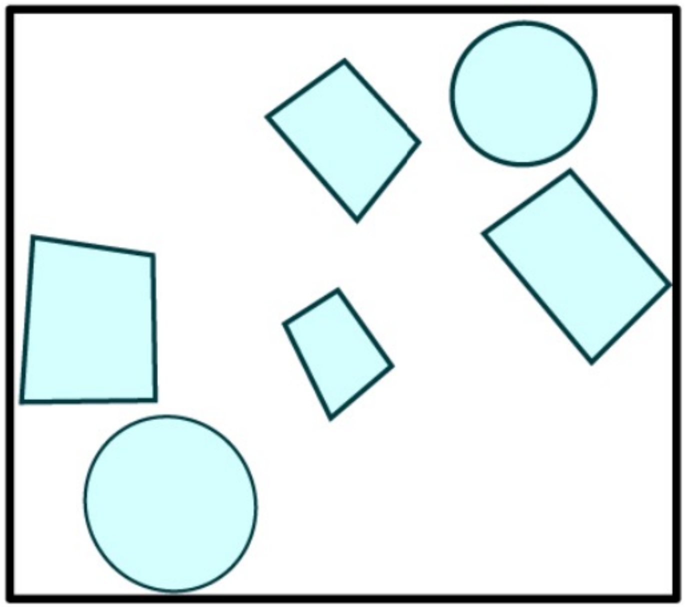
$$x_{min} = \min\left(v_x^i\right)$$

$$x_{max} = \max\left(v_x^i\right)$$

$$y_{min} = \min\left(v_y^i\right)$$

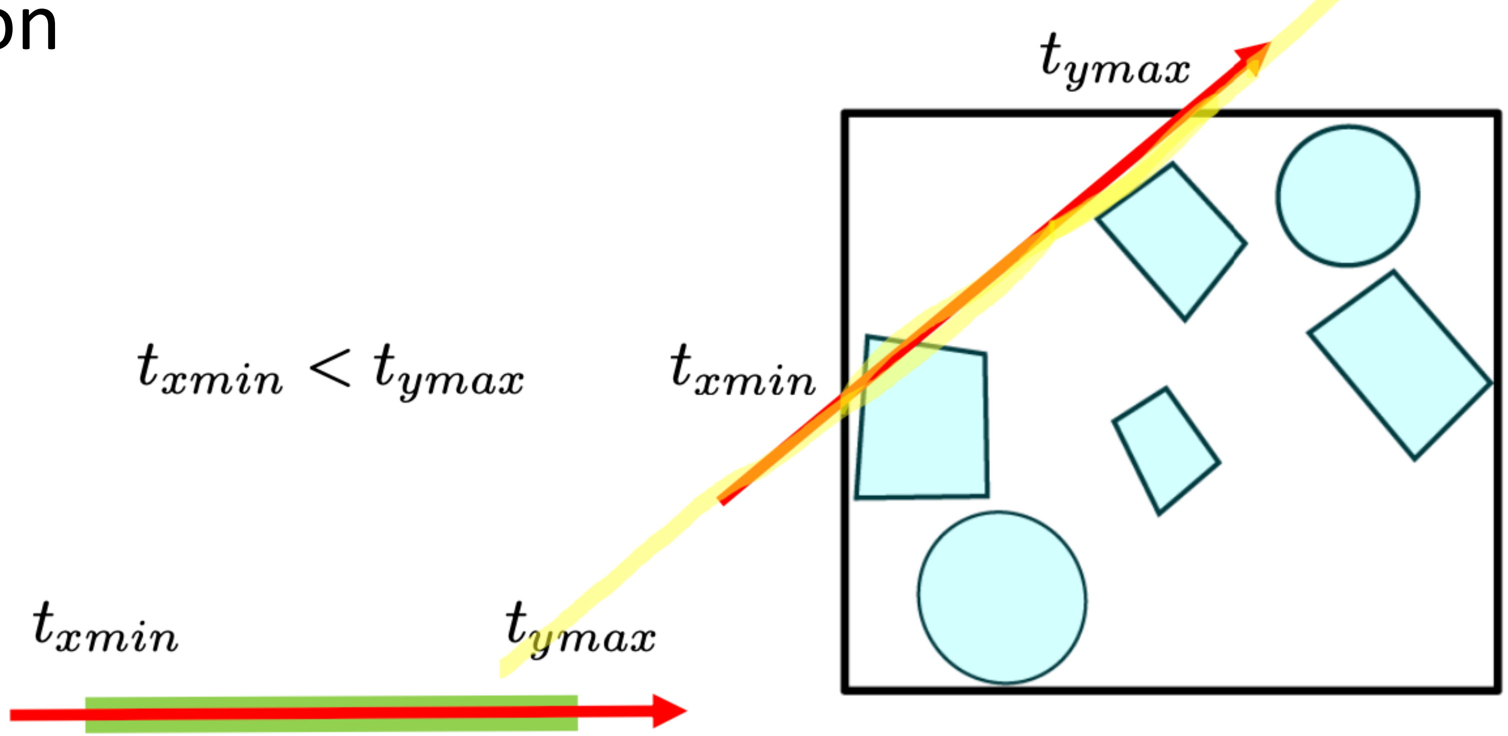
$$y_{max} = \max\left(v_y^i\right)$$

$$\mathbf{v}^i \in \text{Vertices}$$



Lecture 4: Bounding Volume Hierarchy

- Bounding Volume (Axis-Aligned Bounding Box (AABB)) Intersection

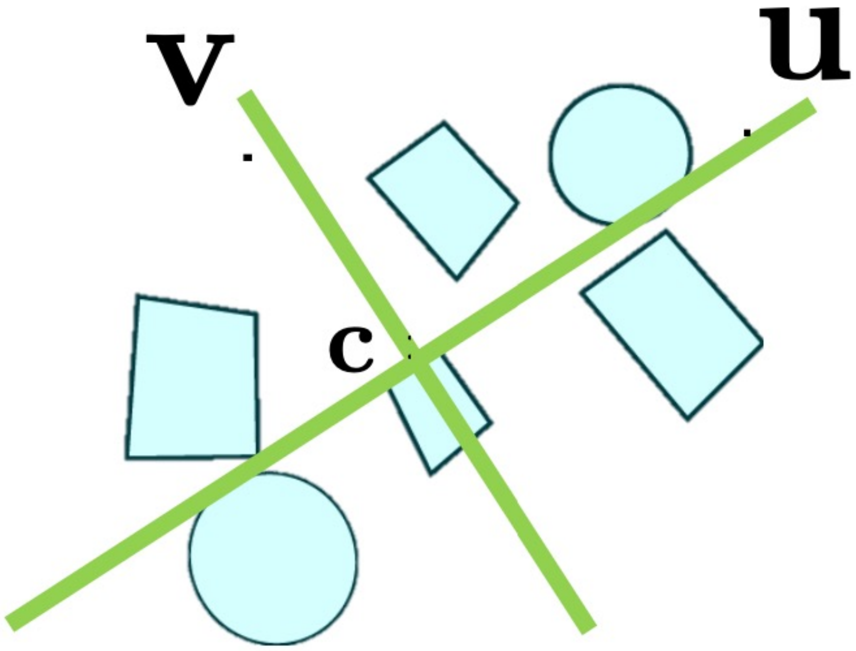


Lecture 4: Bounding Volume Hierarchy

- Bounding Volume (Object-Oriented Bounding Box (OOBB))

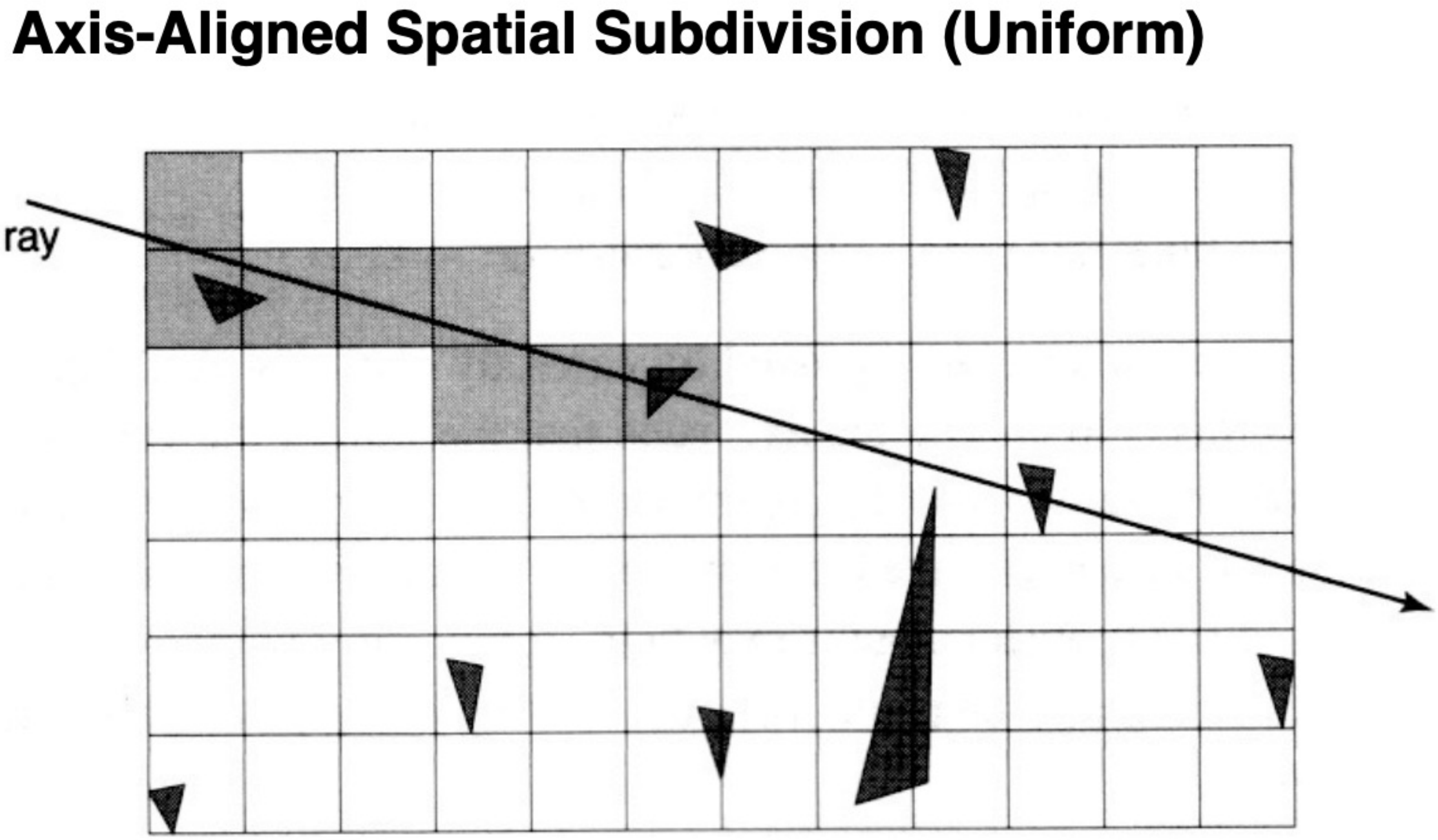
$$\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}^i$$

$$\begin{bmatrix} \mathbf{u} & \mathbf{v} \end{bmatrix}$$

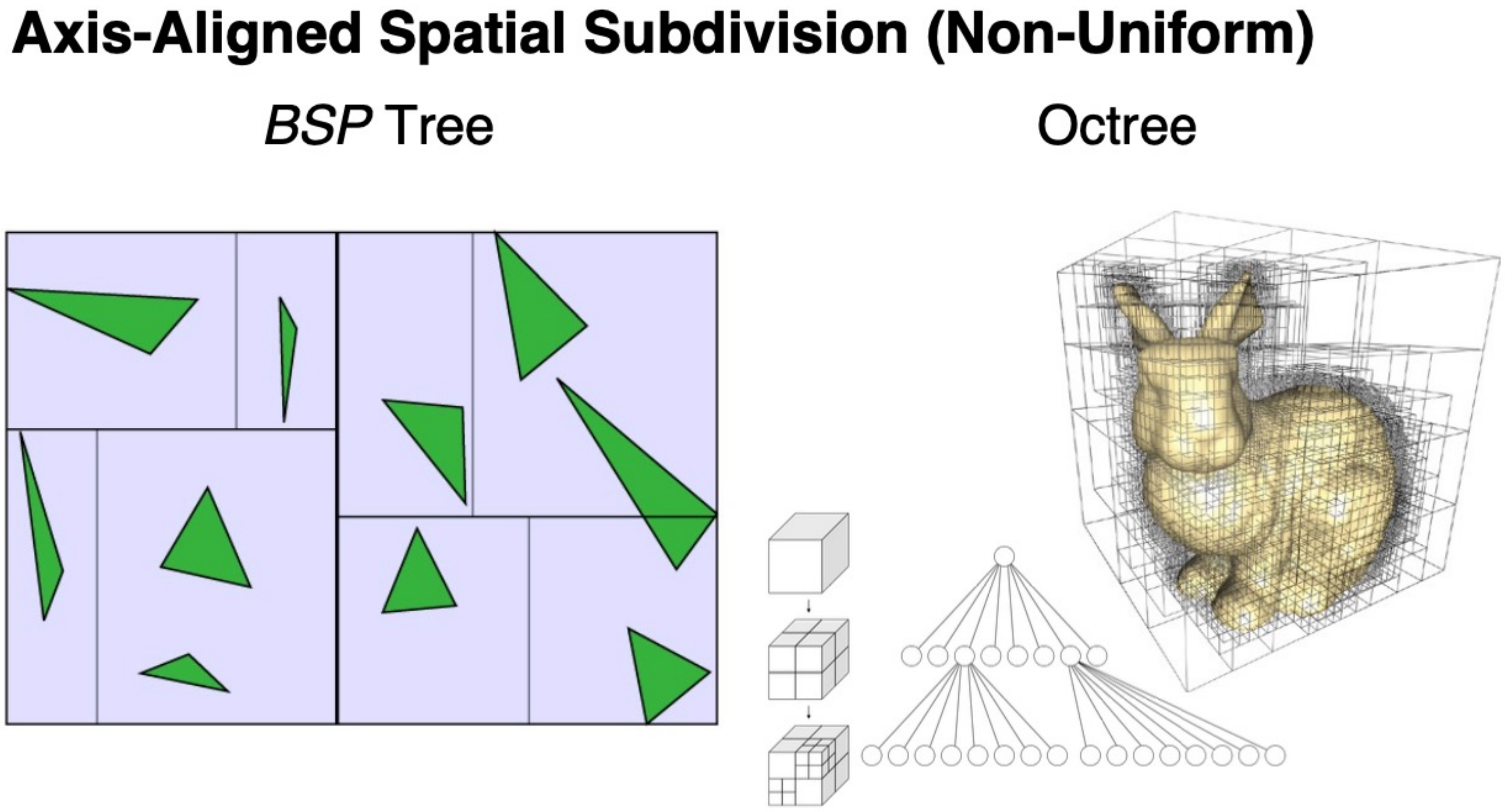


Find directions of maximum and minimum variance

Lecture 4: Bounding Volume Hierarchy



Lecture 4: Bounding Volume Hierarchy



Lecture 4: Bounding Volume Hierarchy

Consider a set of n objects in space.

i– [1 MARK] Using big-$\mathcal{O}$ notation. How many leaf-nodes could an axis-aligned bounding box tree have?

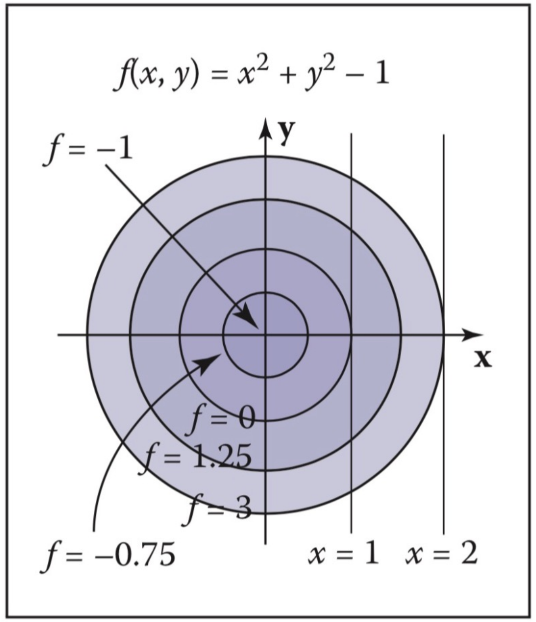
ii– [1 MARK] Using big-$\mathcal{O}$ notation, how deep could an axis-aligned bounding box tree be in the worst case?

iii– [2 MARKS] What strategy is used to prevent the worst case behaviour with respect to the depth of an AABB tree from occurring in practice?

Lecture 5: Meshes

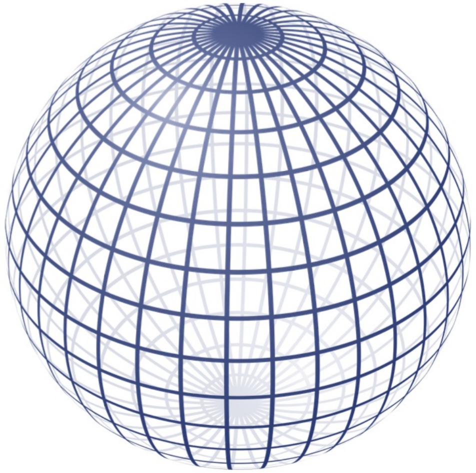
- Surface Representation

Implicit Surface



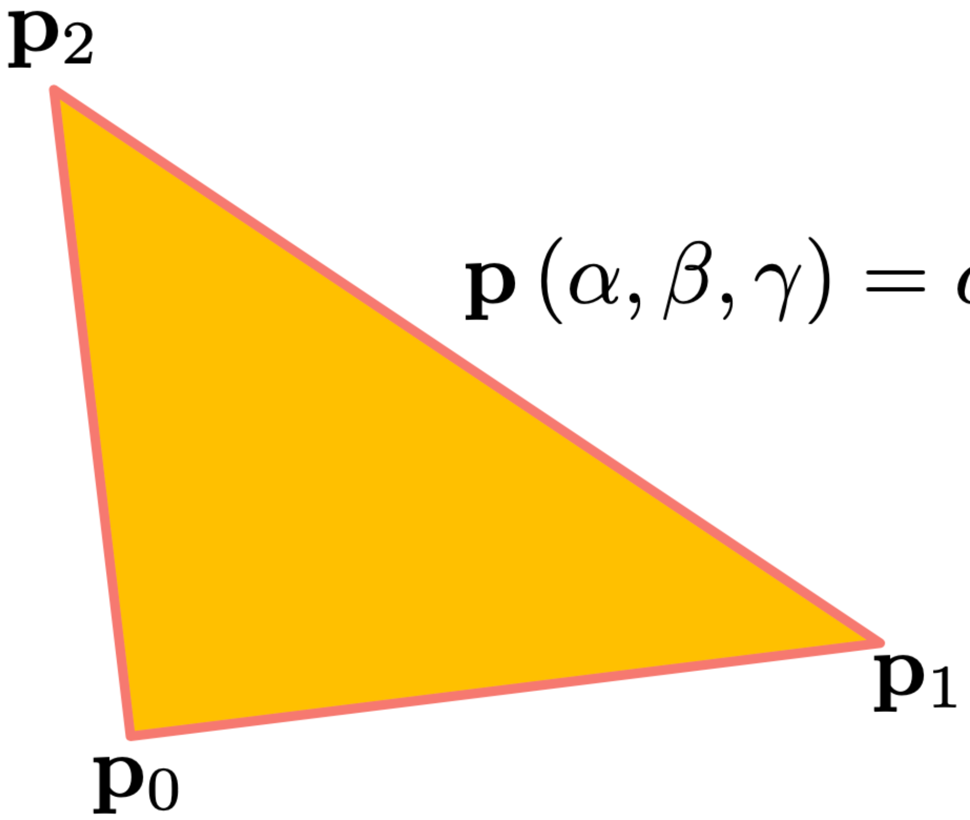
Parametric Surface

$$x = r\cos\phi\sin\theta,$$
$$y = r\sin\phi\sin\theta,$$
$$z = r\cos\theta.$$



Lecture 5: Meshes

- Barycentric Coordinates



$$\mathbf{p}(\alpha, \beta, \gamma) = \alpha\mathbf{p}_1 + \beta\mathbf{p}_2 + \gamma\mathbf{p}_0$$

$$\alpha \geqslant 0$$

$$\beta \geqslant 0$$

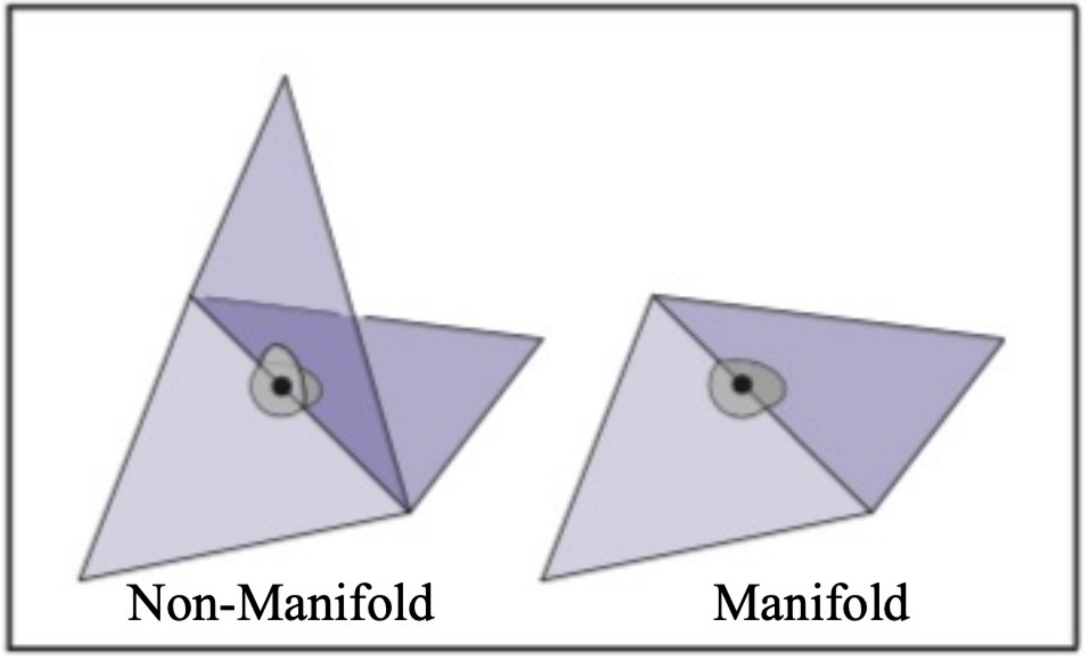
$$\alpha + \beta \leqslant 1$$

$$\gamma = 1 - \alpha - \beta$$

Lecture 5: Meshes

- Manifold

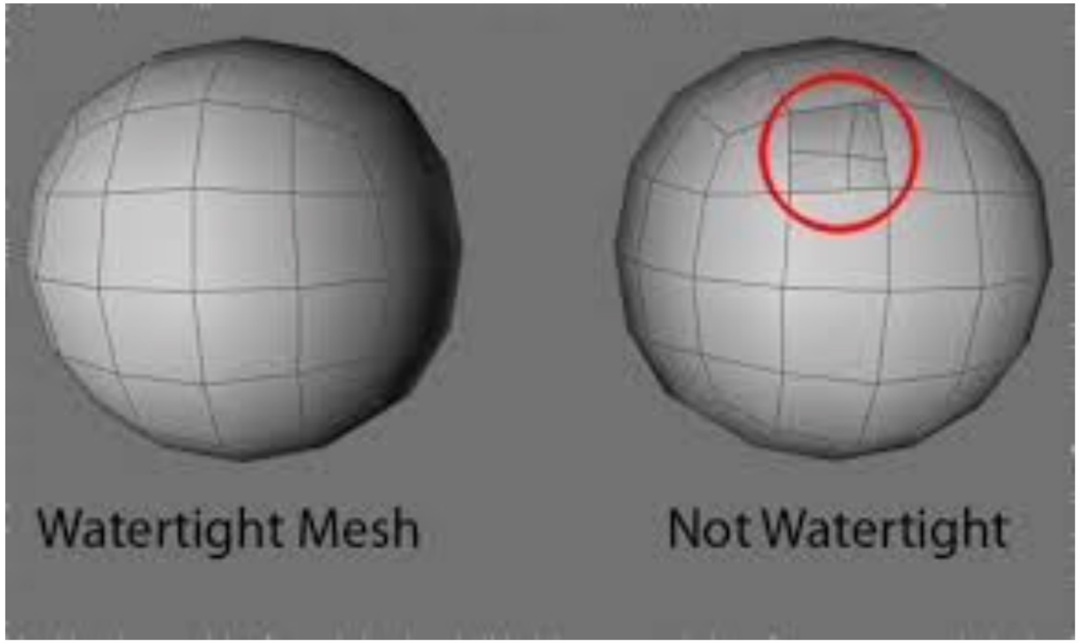
A *2-manifold* is a surface for which the neighbourhood around any point can be flattened onto the plane



Lecture 5: Meshes

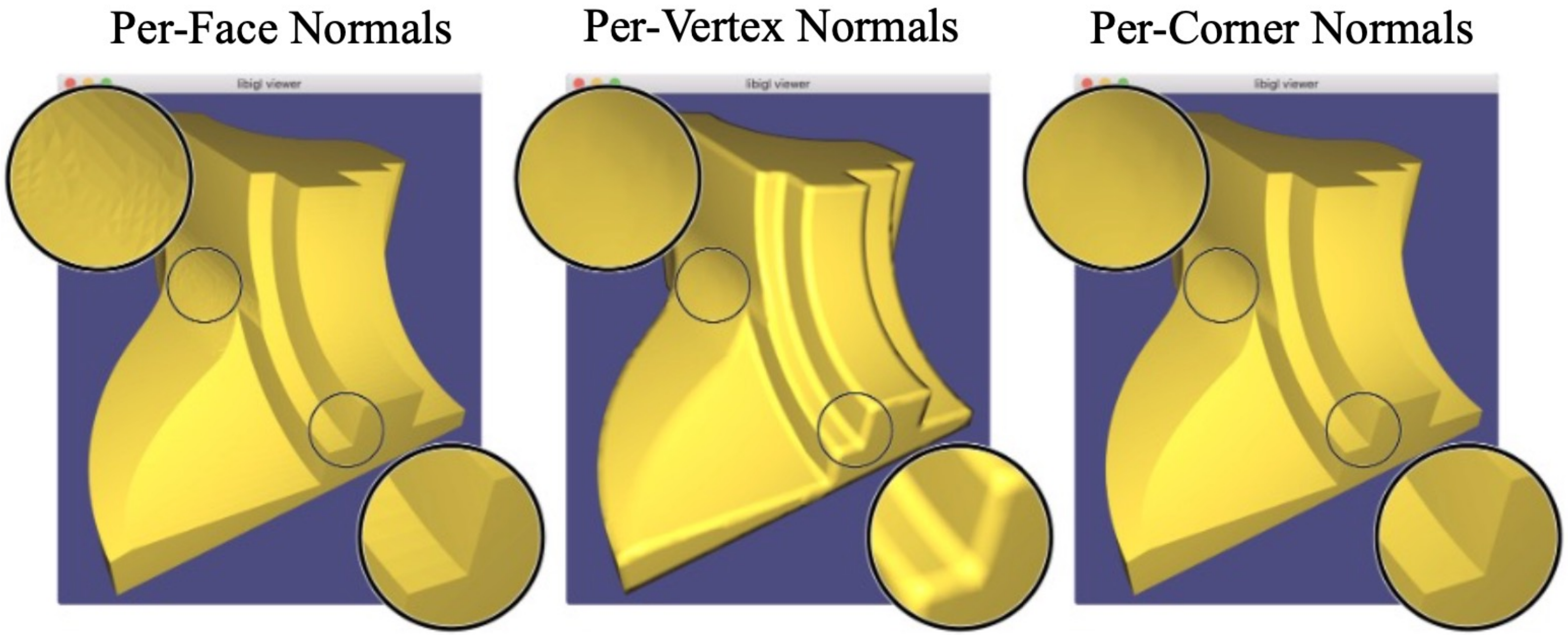
- Watertight

Watertight meshes have no holes



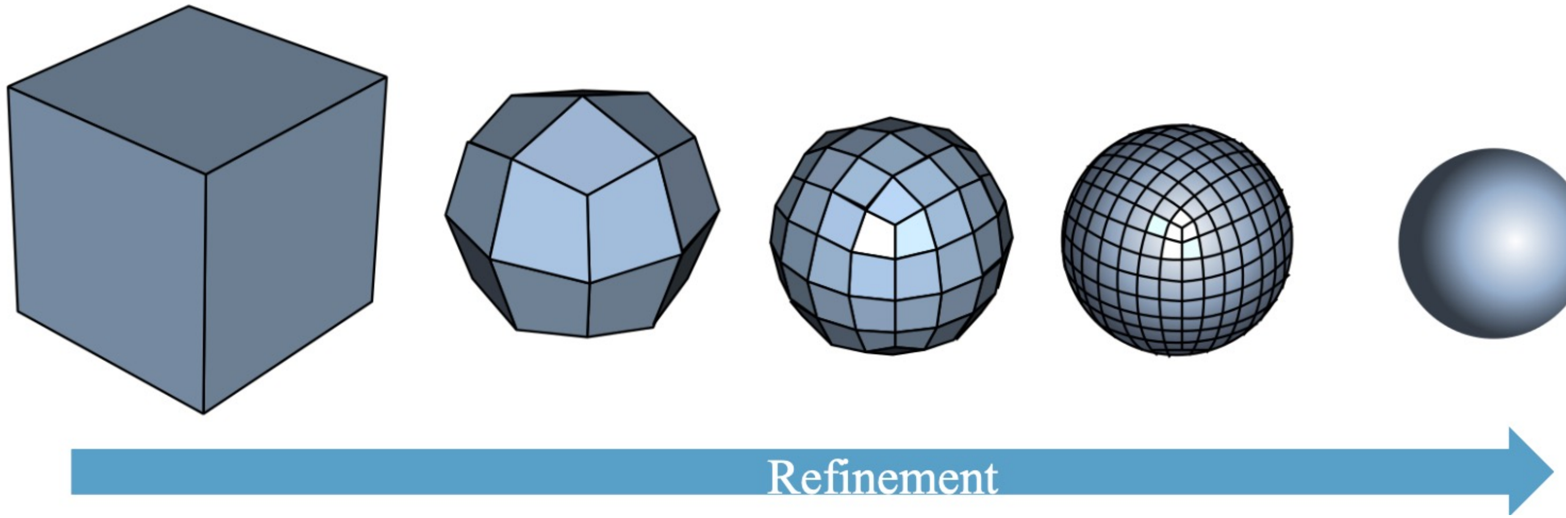
Lecture 5: Meshes

- Normals



Lecture 5: Meshes

- Mesh Subdivision (Catmull-Clark)



Lecture 5: Meshes

i– [1 MARK] When computing per-vertex normals, why is it beneficial to use area-weighting?

ii– [2 MARKS] Match each method to its best suited example.

per-face normals ∘	∘ a cube
per-vertex normals ∘	∘ a cylinder
per-corner normals ∘	∘ a sphere

Lecture 6: Shader Pipeline

- 2D Scaling

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

Lecture 6: Shader Pipeline

- 2D Rotation

$$\mathrm{rotate}(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

Lecture 6: Shader Pipeline

- 2D Translation

$$T \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \end{bmatrix}$$

Lecture 6: Shader Pipeline

- Homogeneous Coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow[\text{homogeneous coordinates}]{\text{Considered as a point in 3D}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Lecture 6: Shader Pipeline

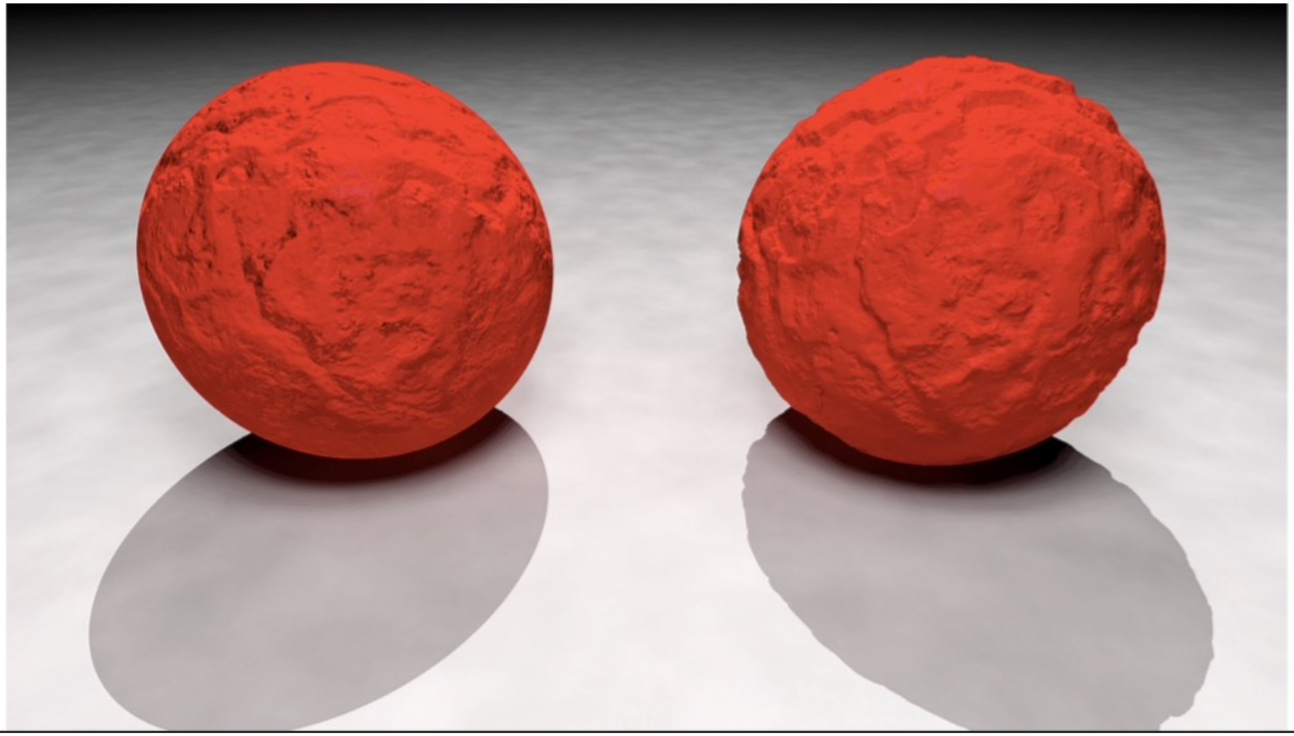
- Homogeneous Coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow[\text{homogeneous coordinates}]{\text{Considered as a } \underline{\textbf{\textit{vector}}} \text{ in 3D}} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$$

Lecture 6: Shader Pipeline

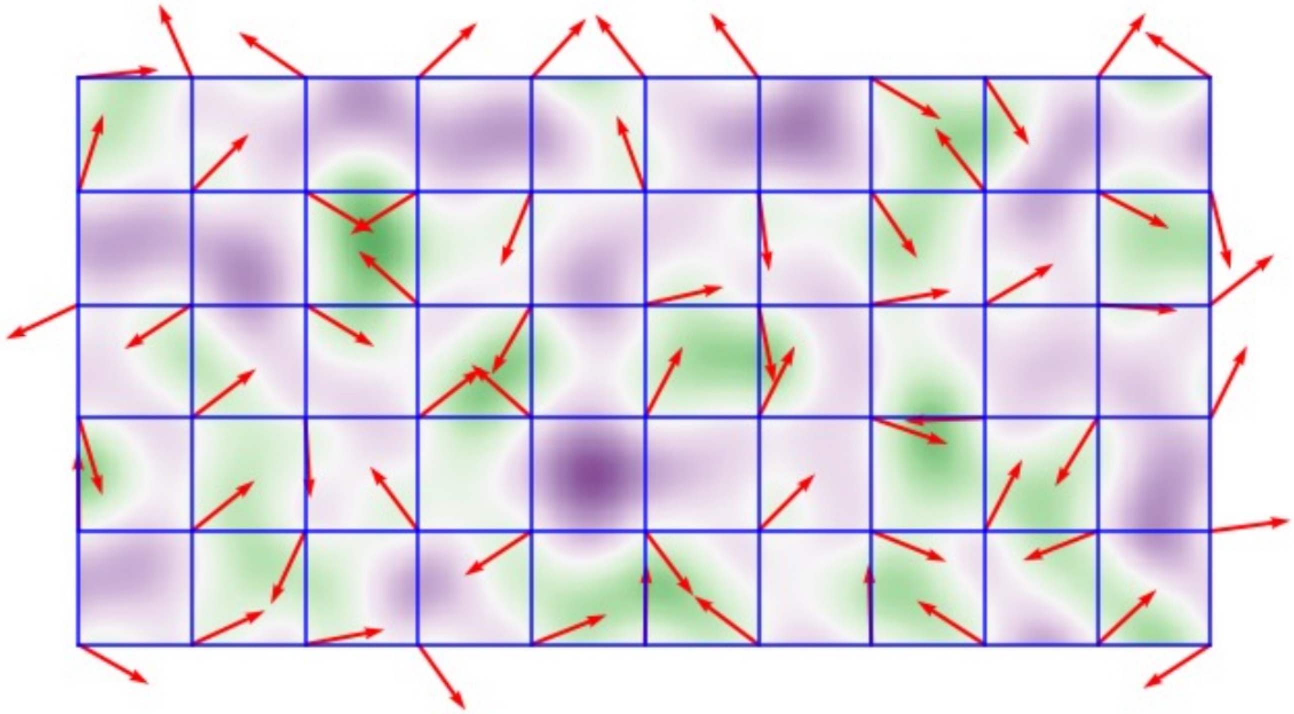
- Mapping

Normal Mapping vs. Displacement Mapping



Lecture 6: Shader Pipeline

- Perlin Noise



Lecture 6: Shader Pipeline

- Perlin Noise

Part (b) Perlin Noise [1 MARK]

If a 2D noise function noise(x,y) is too low frequency, how can easily get a higher frequency noise pattern?

Lecture 6: Shader Pipeline

- Shader

ii– [1 MARK] Circle one. Using per-vertex normals, where should Phong shading be implemented for best looking highlights?

vertex shader

tessellation control shader

tessellation eval shader

none (rasterization does shading automatically)

fragment shader

Lecture 7: Kinematics

Forward Kinematics

Kinematics – study of motion without consideration of what causes that motion

Forward Kinematics – Generate motion by setting all the bone positions by hand

To do this we need a more rigourous understanding of how our bone motions are represented

Lecture 7: Kinematics

- Inverse Kinematics

Posing all those bones can be tedious, wouldn’t it be great if you could just specify a few bones and the rest would be automatically computed ?

That’s what inverse kinematics does

Lecture 7: Kinematics

- Gradient Descent

Simple Gradient Descent Algorithm

While not at an optimal point

- Compute the gradient at current point (x)
- Move to new point $x = x - \boxed{h} \nabla f(x)$

Lecture 7: Kinematics

Part (b) Linear Blend Skinning [3 MARKS]

i– [2 MARKS] Consider two rotation matrices $\mathbf{R}_1 \in \mathbb{R}^{3\times3}$ and $\mathbf{R}_2^{3\times3}$. Will the weighted average of these matrices $(w\mathbf{R}_1 + (1-w) * \mathbf{R}_2)$ produce another rotation matrix? Prove or counter example.

ii– [1 MARK] Why is it important that linear blend skinning weights sum to one (partition unity)?

Lecture 8: Mass-Spring Systems

i– For a point-mass in 3D. What are the dimension of $\mathbf{f}$, m and $\mathbf{a}$ in Newton's second law $\mathbf{f} = m\mathbf{a}$? [2 MARKS]

$\mathbf{f} \in \mathbb{R}^{\_\_\_}$

$m \in \mathbb{R}^{\_\_\_}$

$\mathbf{a} \in \mathbb{R}^{\_\_\_}$

Lecture 8: Mass-Spring Systems

Part (a) Sparse Matrices [6 MARKS]

Consider a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ with maximum k non-zeros per row and a vector $\mathbf{x} \in \mathbb{R}^n$.

i– [1 MARK] Using big-$\mathcal{O}$ notation, what is the computational complexity of computing $\mathbf{Ax}$ using dense (full) matrix data-structures?

ii– [1 MARK] Using big-$\mathcal{O}$ notation, what is the computational complexity of computing $\mathbf{Ax}$ using a sparse matrix data-structure to store $\mathbf{A}$?

Tips

- State your assumptions
- Content: “at least 80% covering material after the midterm”
- Don’t “over think”