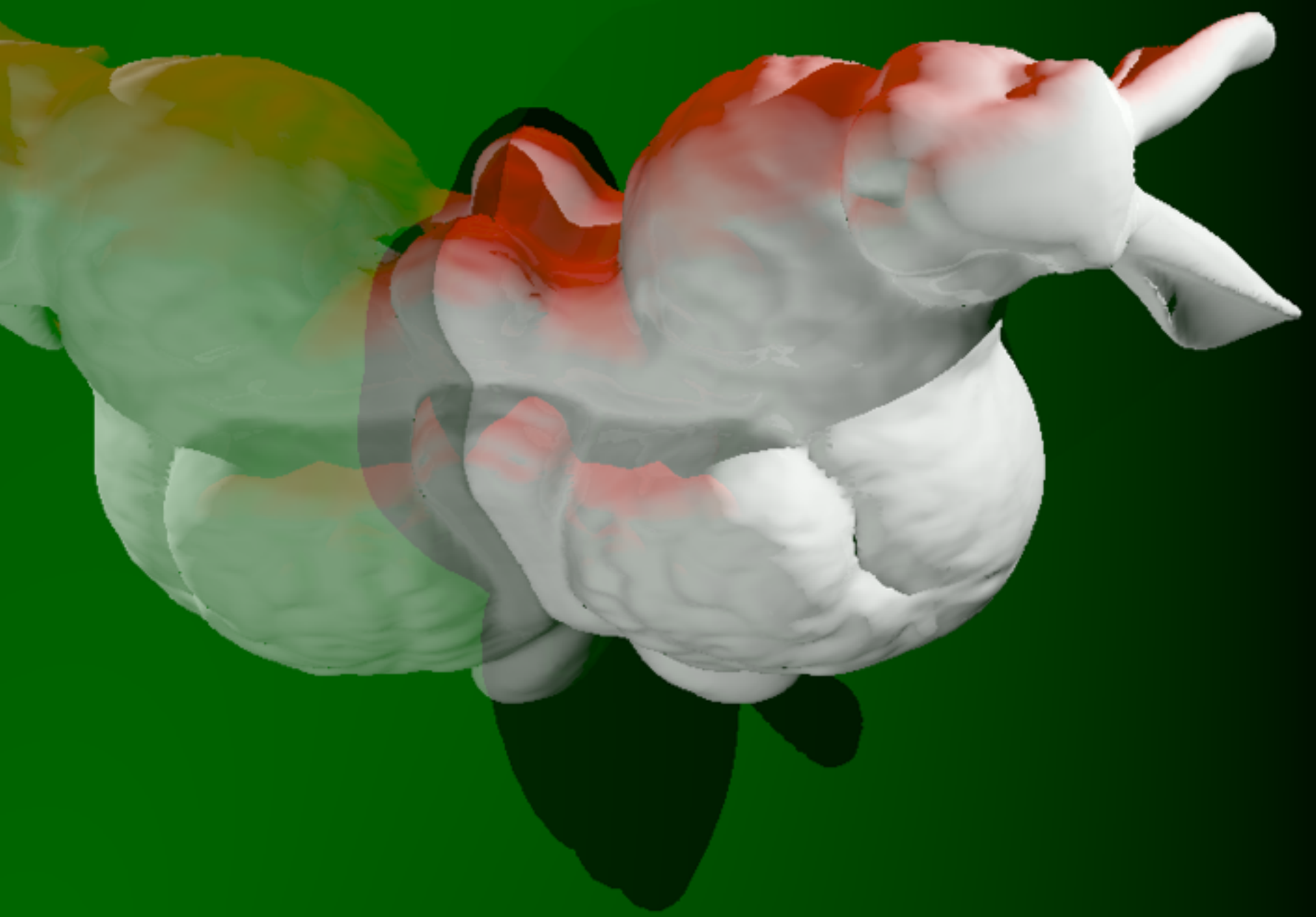
Texture Mapping



Overview

1. Simple Shapes
   - rectangle, disc, sphere, cylinder
2. Triangle Meshes
3. Ray Tracer API

Texture

- 2D bitmap mapped onto a 3D object
- function $[0,1] \times [0,1] \rightarrow$ Colour

- (0,0) is the bottom left corner
- (1,1) is the top right corner

where [0,1] is the interval from 0 to 1

Texture

- 2D bitmap mapped onto a 3D object
- function $[0,1] \times [0,1] \rightarrow$ Colour

  normalised texture coordinates

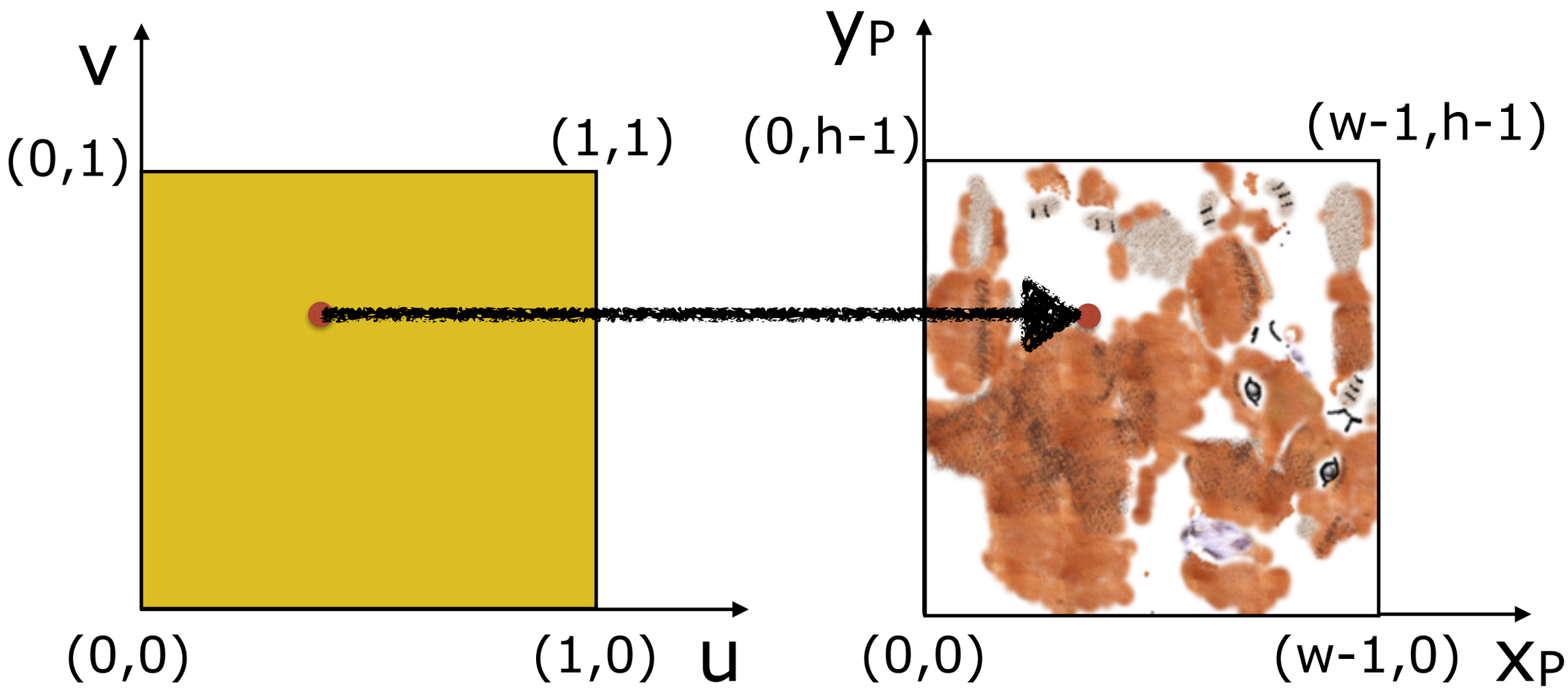
- (0,0) is the bottom left corner
- (1,1) is the top right corner

where [0,1] is the interval from 0 to 1

Texture Coordinates

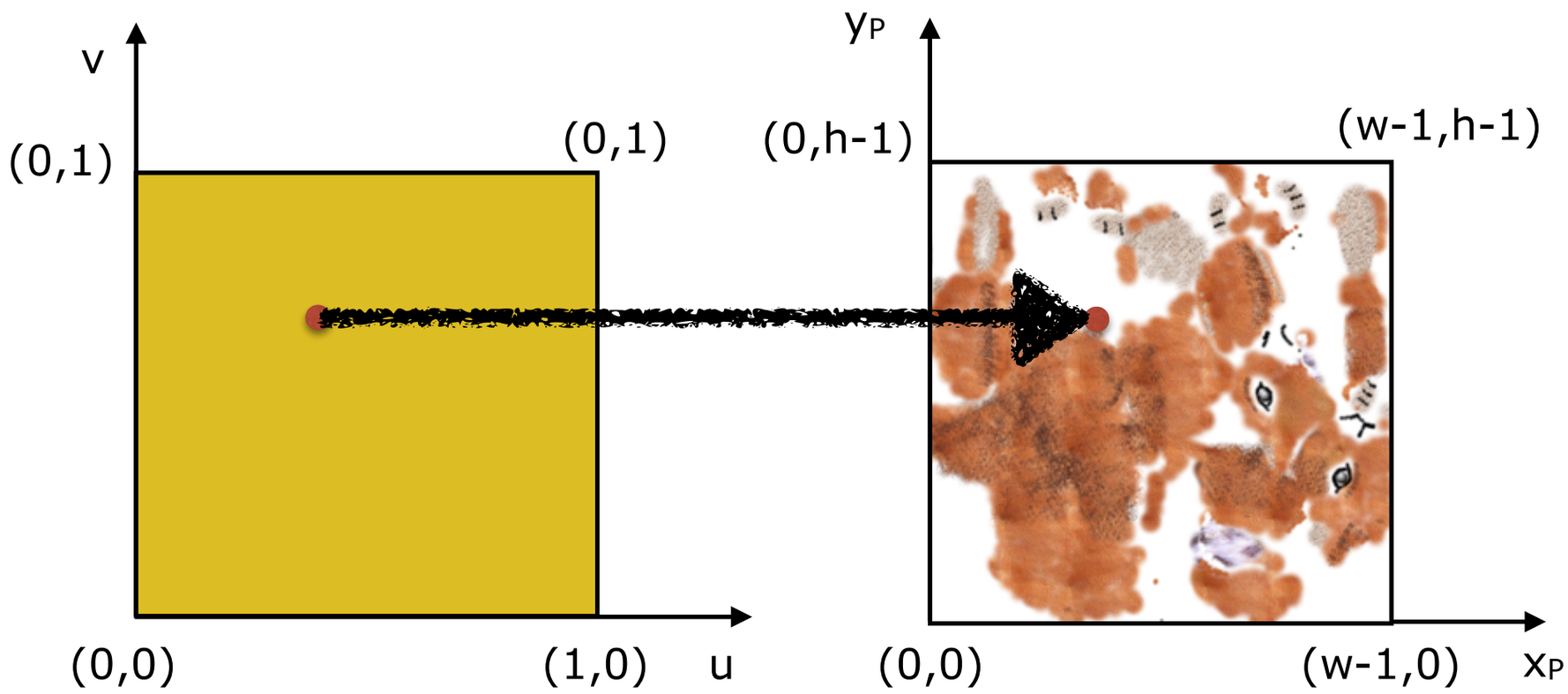
Texture coordinates
$[0,1] \times [0,1]$

Bitmap picture
width w, height h



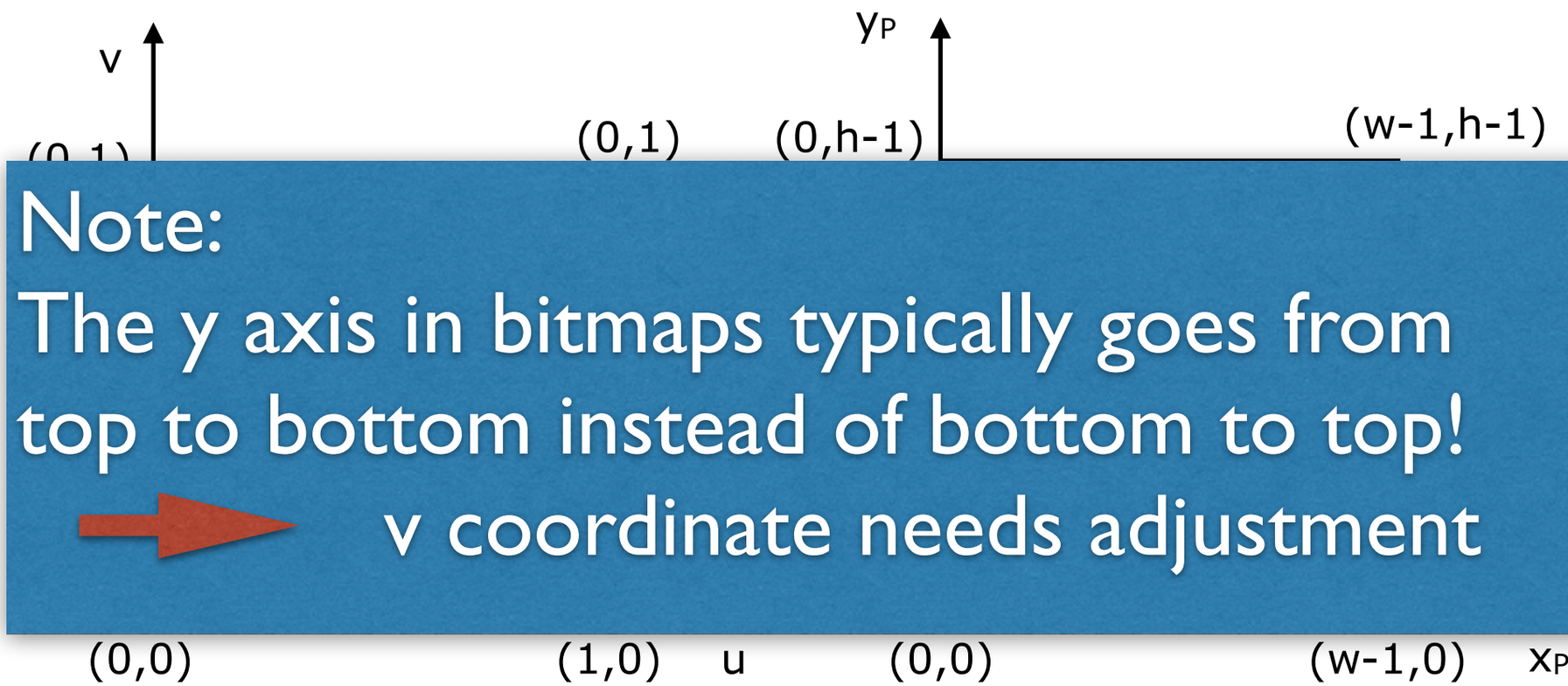
Convert Texture Coordinates

$$(u,v) \longrightarrow \Big(u(w-1), v(h-1)\Big)$$



Convert Texture Coordinates

$$(u,v) \longrightarrow \Big(u(w-1), v(h-1)\Big)$$

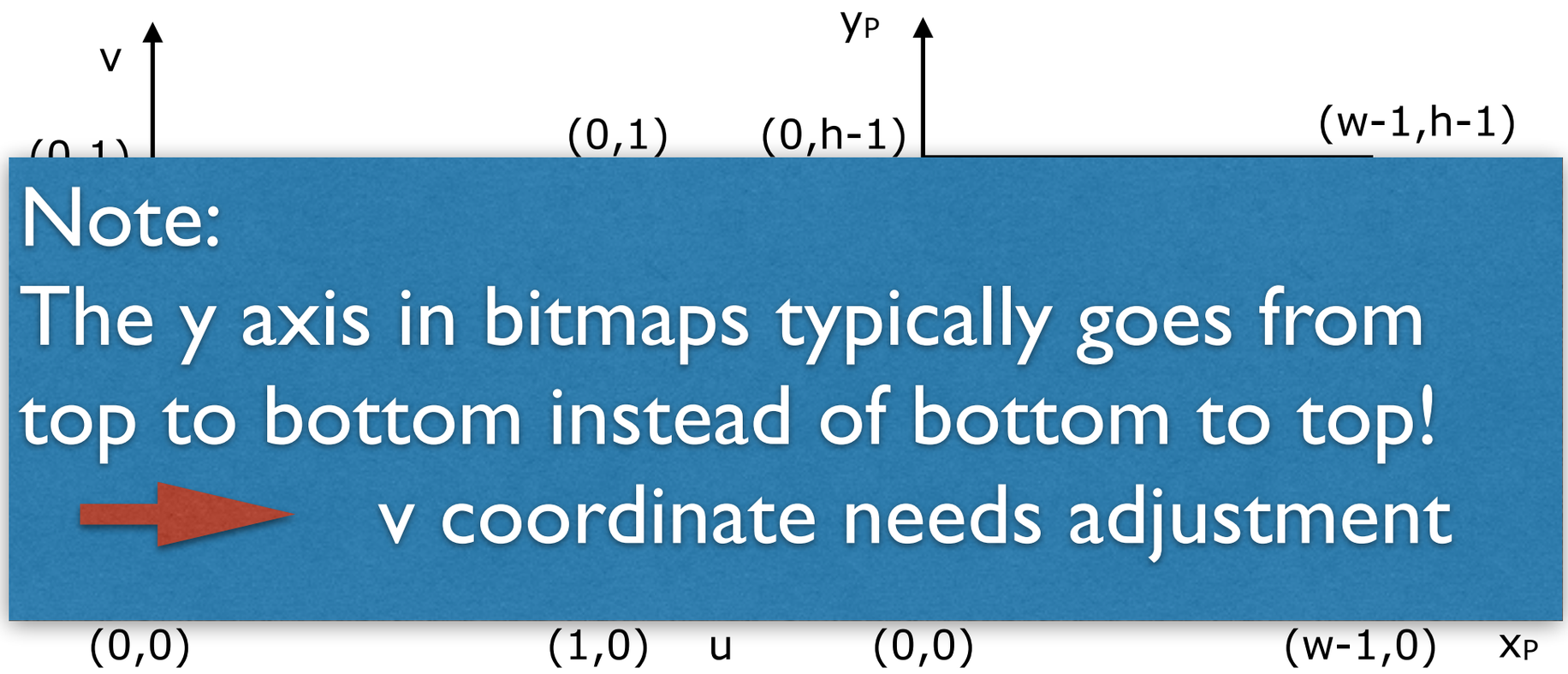


Note:
The y axis in bitmaps typically goes from top to bottom instead of bottom to top!

→ v coordinate needs adjustment

Convert Texture Coordinates

$$(u,v) \longrightarrow \Big(u(w-1),(1-v)(h-1)\Big)$$



Note:
The y axis in bitmaps typically goes from top to bottom instead of bottom to top!
→ v coordinate needs adjustment

More Than Just Colour

- Simple texture

$$[0,1] \times [0,1] \rightarrow \text{Colour}$$

- We generalise this

$$[0,1] \times [0,1] \rightarrow \text{Material}$$

More Than Just Colour

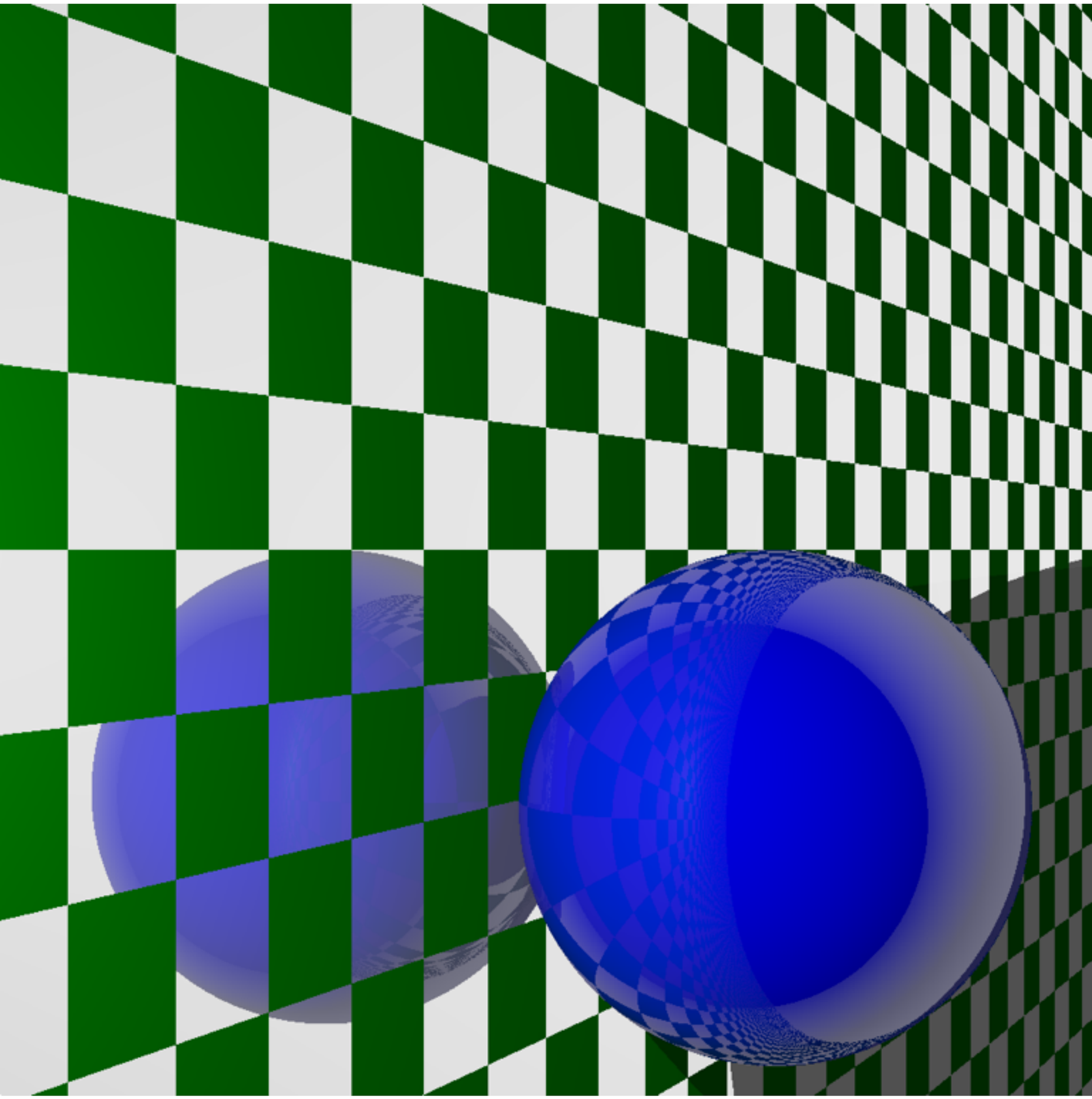
- Simple texture

$$[0,1] \times [0,1] \rightarrow \text{Colour}$$

- We generalise this

$$[0,1] \times [0,1] \rightarrow \text{Material}$$

contains colour and reflection factor



Load Texture from File

```
let loadTexture (file : string)
        : float -> float -> material =
  let img = new Bitmap(file)
  let widthf  = float (img.Width - 1)
  let heightf = float (img.Height - 1)
  let texture (u : float) (v : float) =
    let color = img.GetPixel
                  (int (widthf * u),
                   int (heightf * (1.0 - v)))
    mkMaterial (fromColor color) 0.0
  texture
```

Ray Tracing With Textures

Without textures:

1. find intersection of ray with shape
2. determine the colour of the pixel based on the **(single) colour of the shape** + lighting and reflection

With textures:

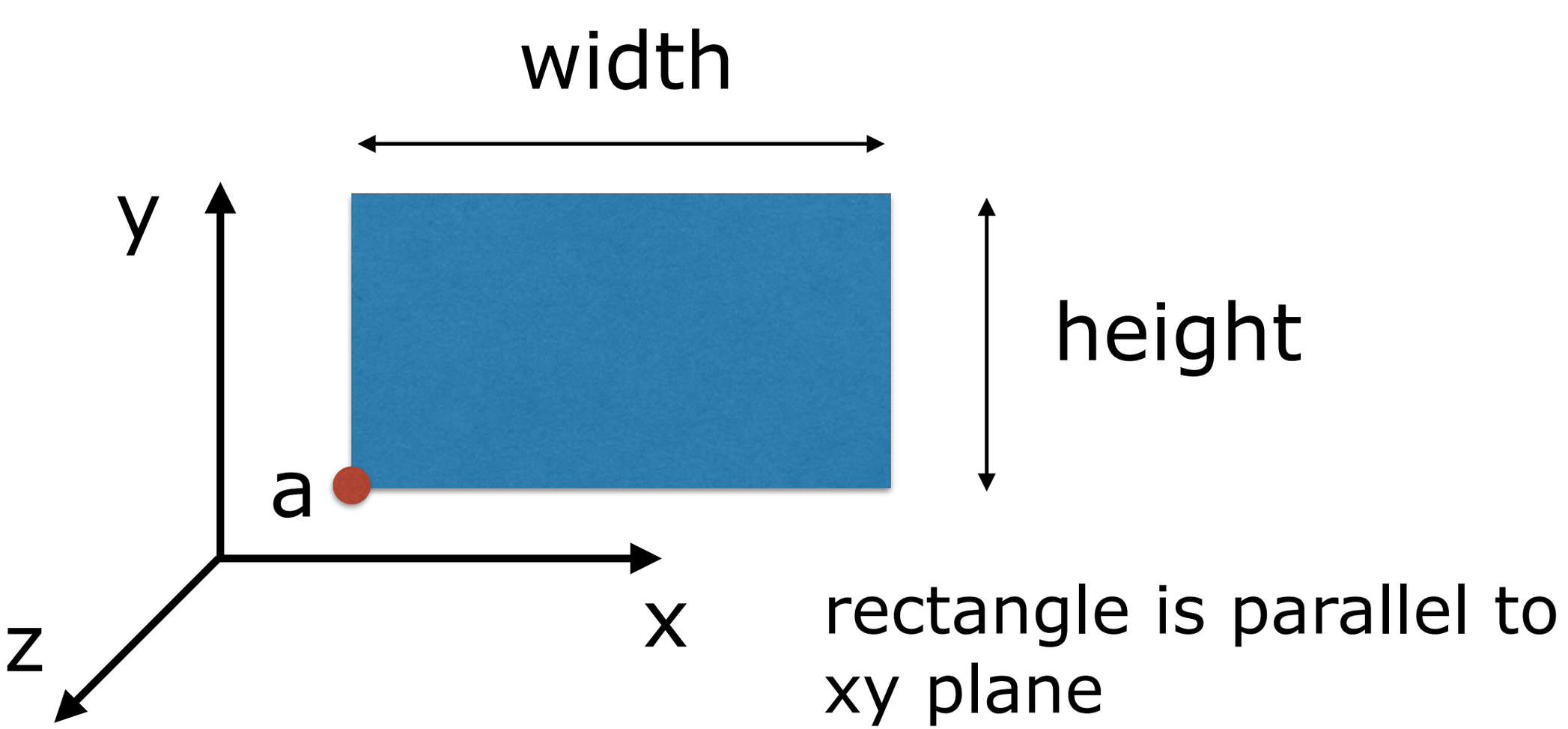
1. find intersection of ray with shape
2. determine (u,v) coordinates of intersection point
3. determine the colour of the pixel based on the **colour of the texture at (u,v)** + lighting and reflection

IT UNIVERSITY OF COPENHAGEN

Questions

Rectangle

- We only consider rectangles that are axis aligned
- They can be transformed later

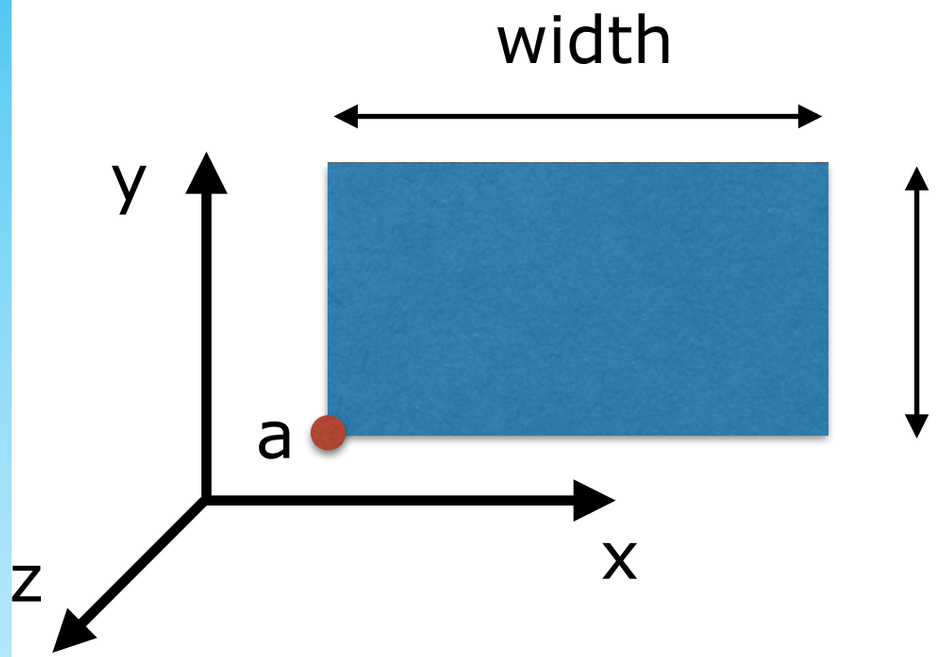


rectangle is parallel to xy plane

Hit function

- check intersection with the plane the rectangle is in (equation: $p_z = a_z$)
- check that the intersection point p (if any) is in the rectangle:

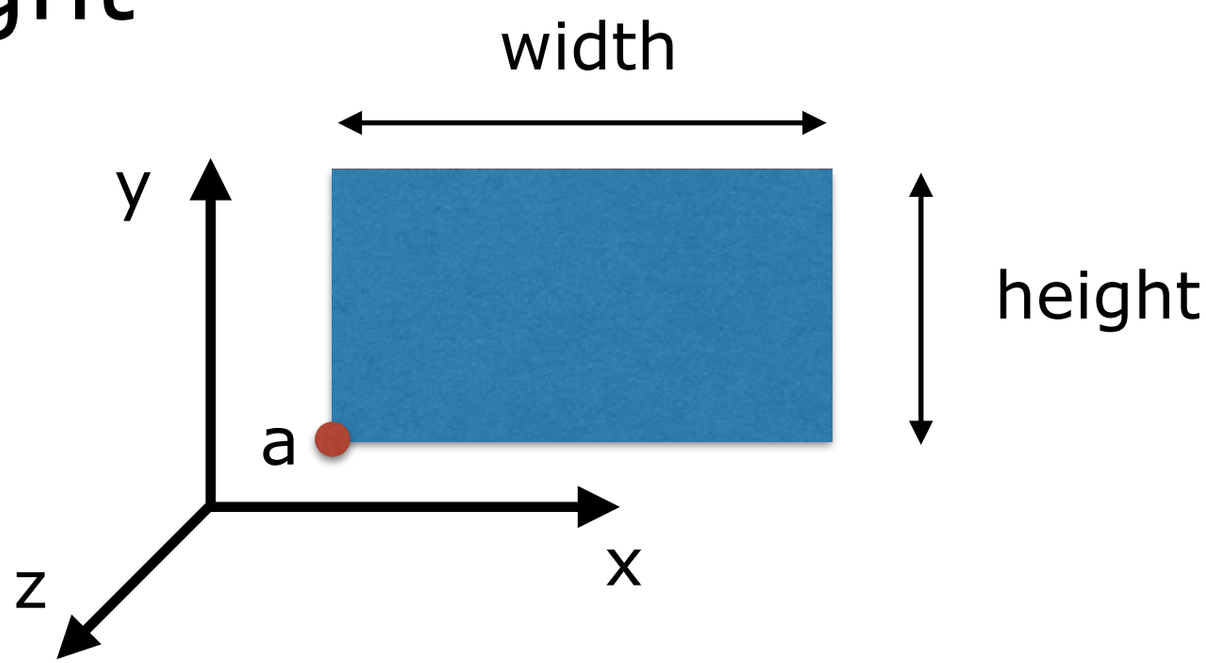
$$a_x \leq p_x \leq a_x + \text{width}$$
$$a_y \leq p_y \leq a_y + \text{height}$$



normal vector: [0,0,1]

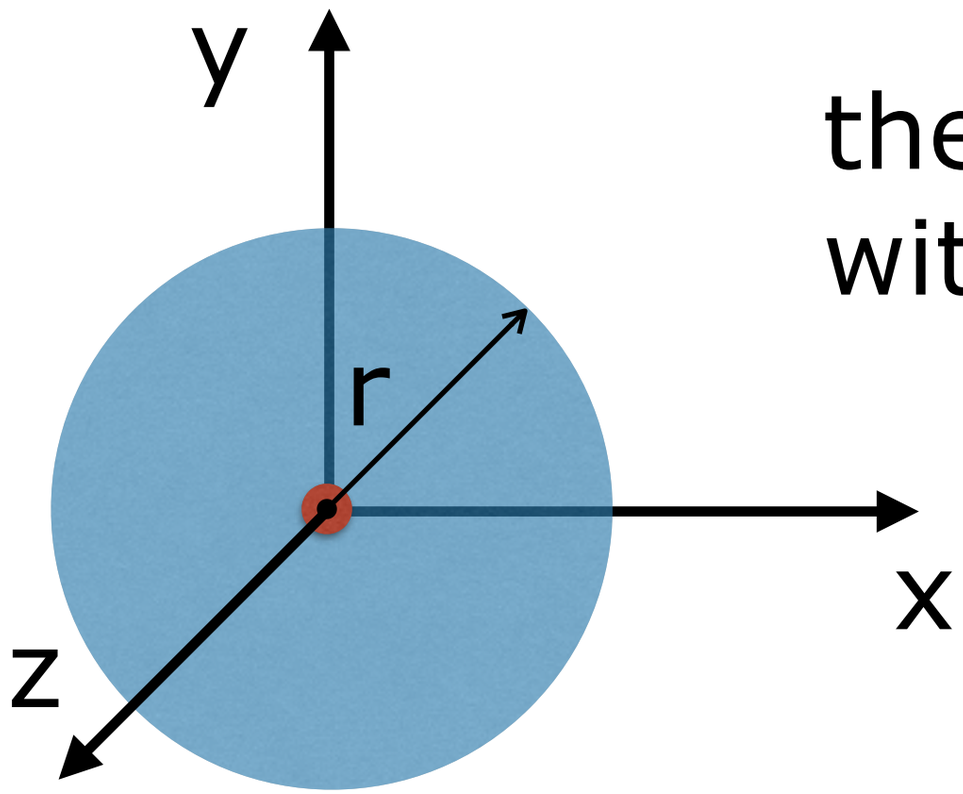
Texture Mapping

- recall: $[0,1] \times [0,1] \rightarrow$ Colour
- we need to map the intersection point p to u-v coordinates in $[0,1] \times [0,1]$
- $u = (p_x - a_x)/\text{width}$
- $v = (p_y - a_y)/\text{height}$



Discs

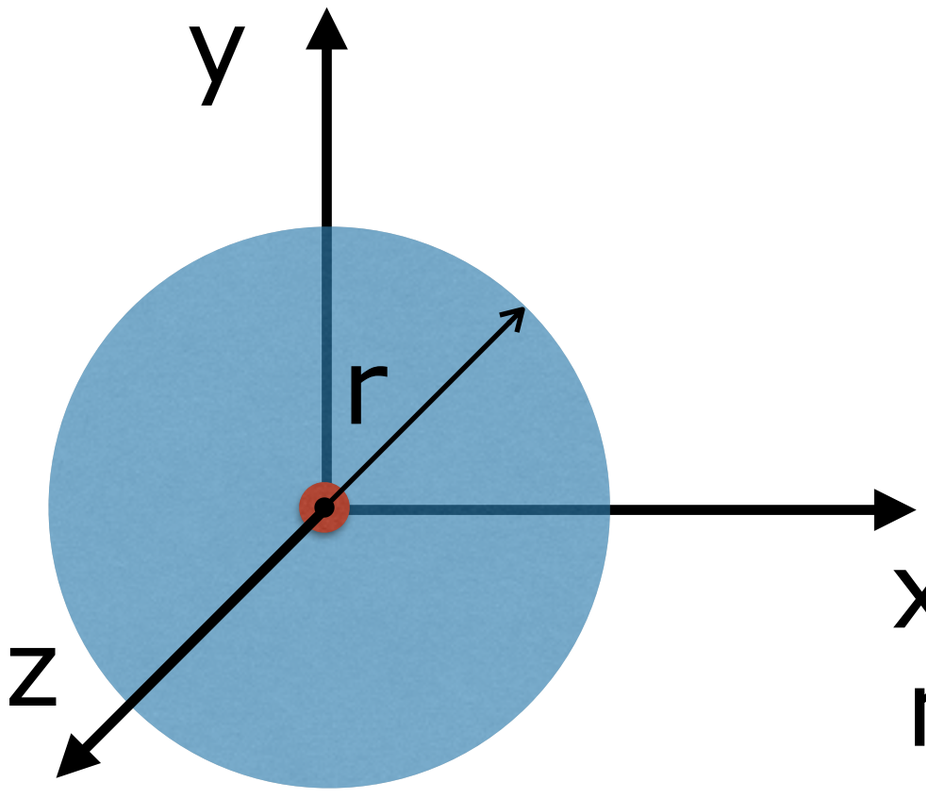
- We only consider discs that are axis aligned
- They can be transformed later



the disc is in xy plane with center (0,0,0)

Hit function

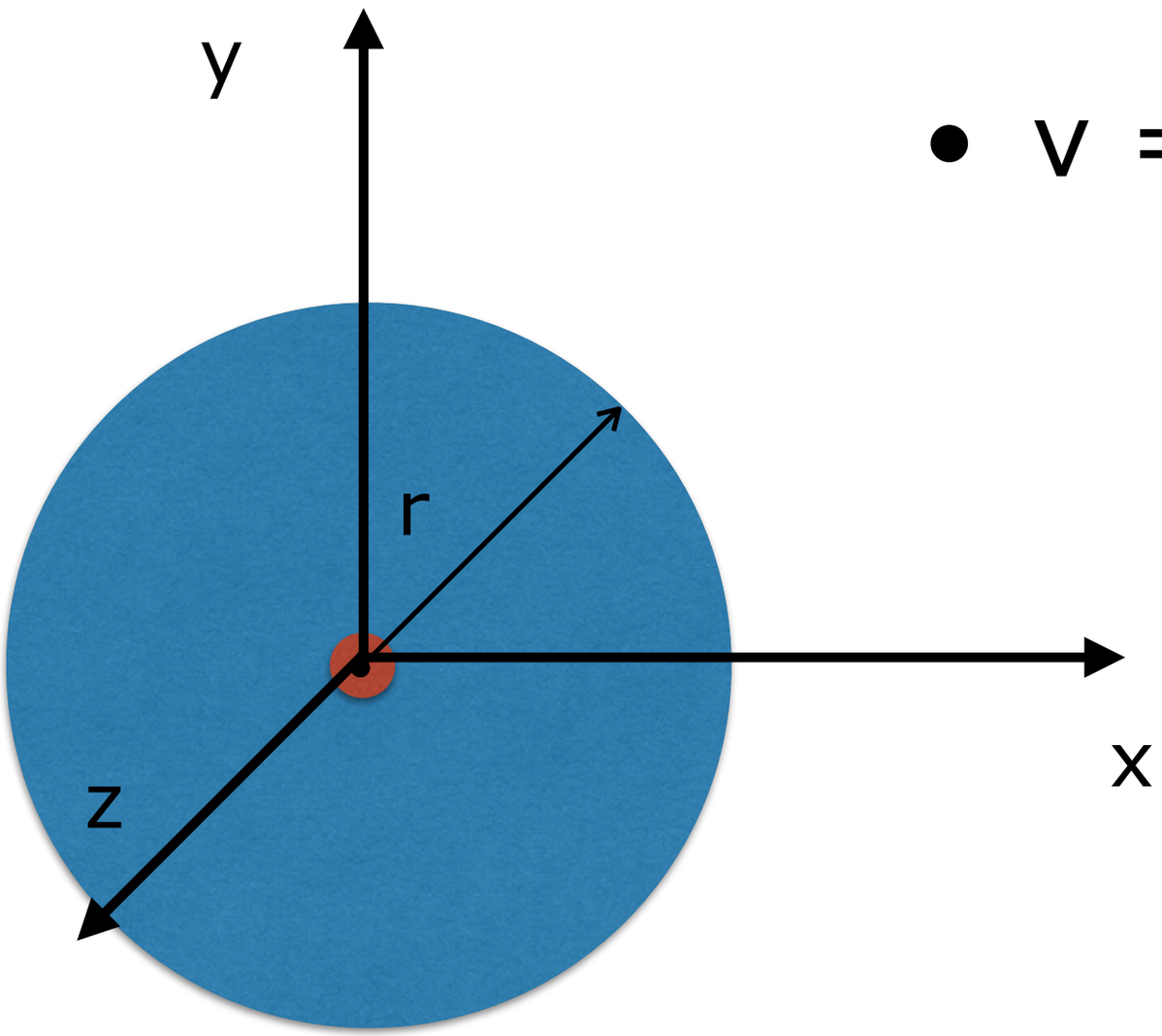
- check intersection with the plane the disc is in: $p_z = 0$
- check that the intersection point p (if any) is in the disc:



$$p_x^2 + p_y^2 \leq r^2$$

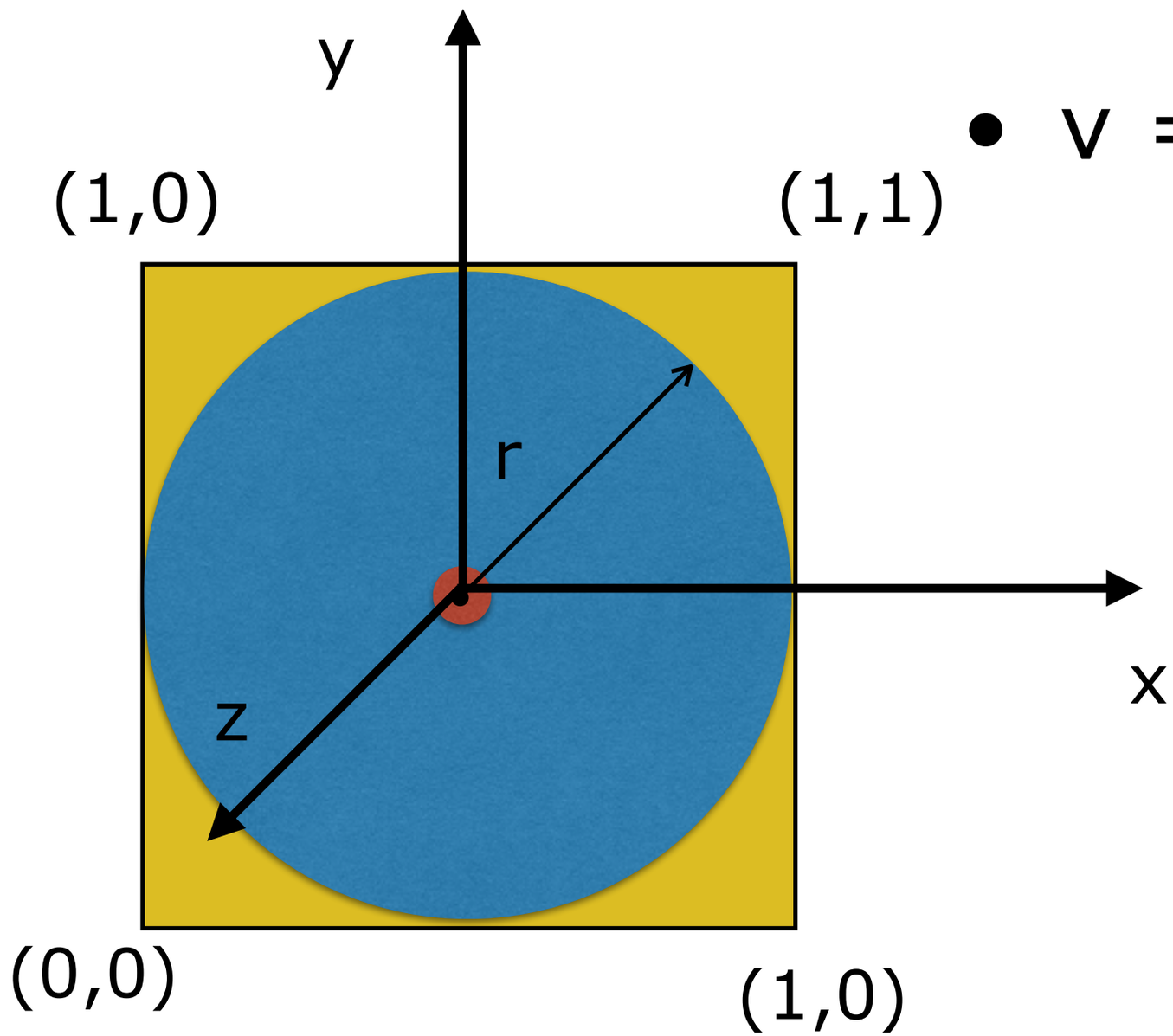
normal vector: (0,0,1)

Texture Mapping



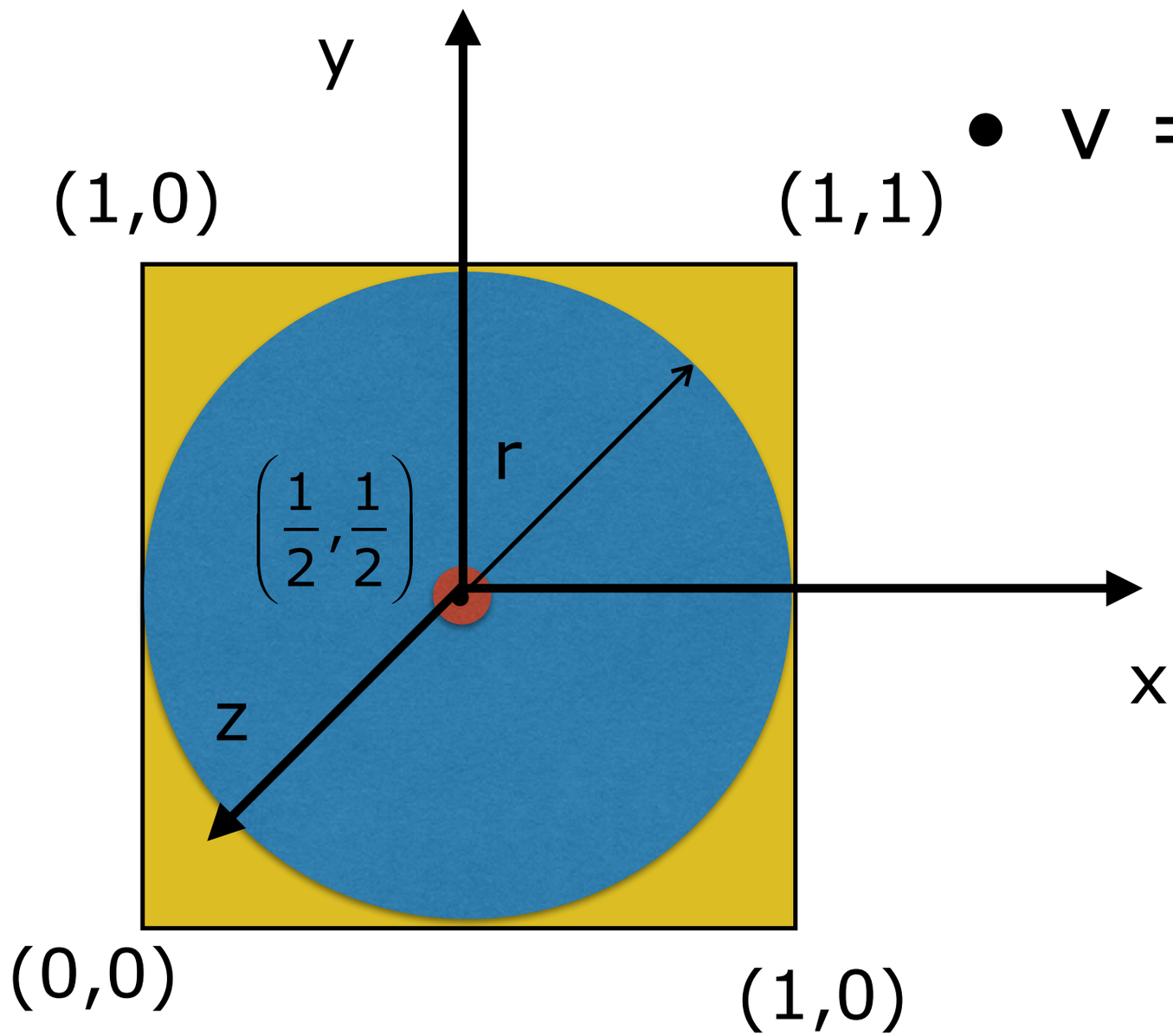
- $u = (p_x+r)/2r$
- $v = (p_y+r)/2r$

Texture Mapping



- $u = (p_x+r)/2r$
- $v = (p_y+r)/2r$

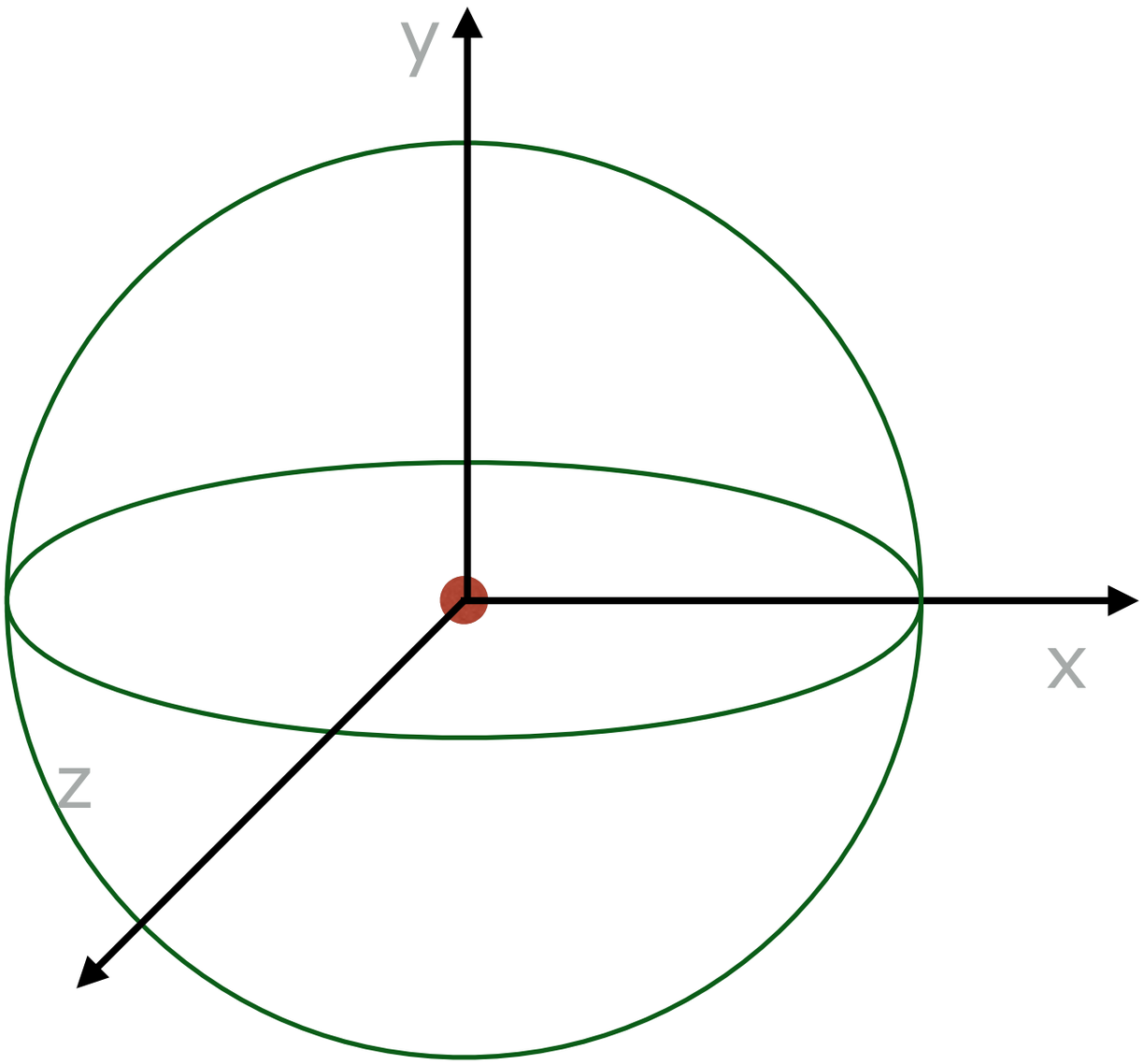
Texture Mapping



- $u = (p_x+r)/2r$
- $v = (p_y+r)/2r$

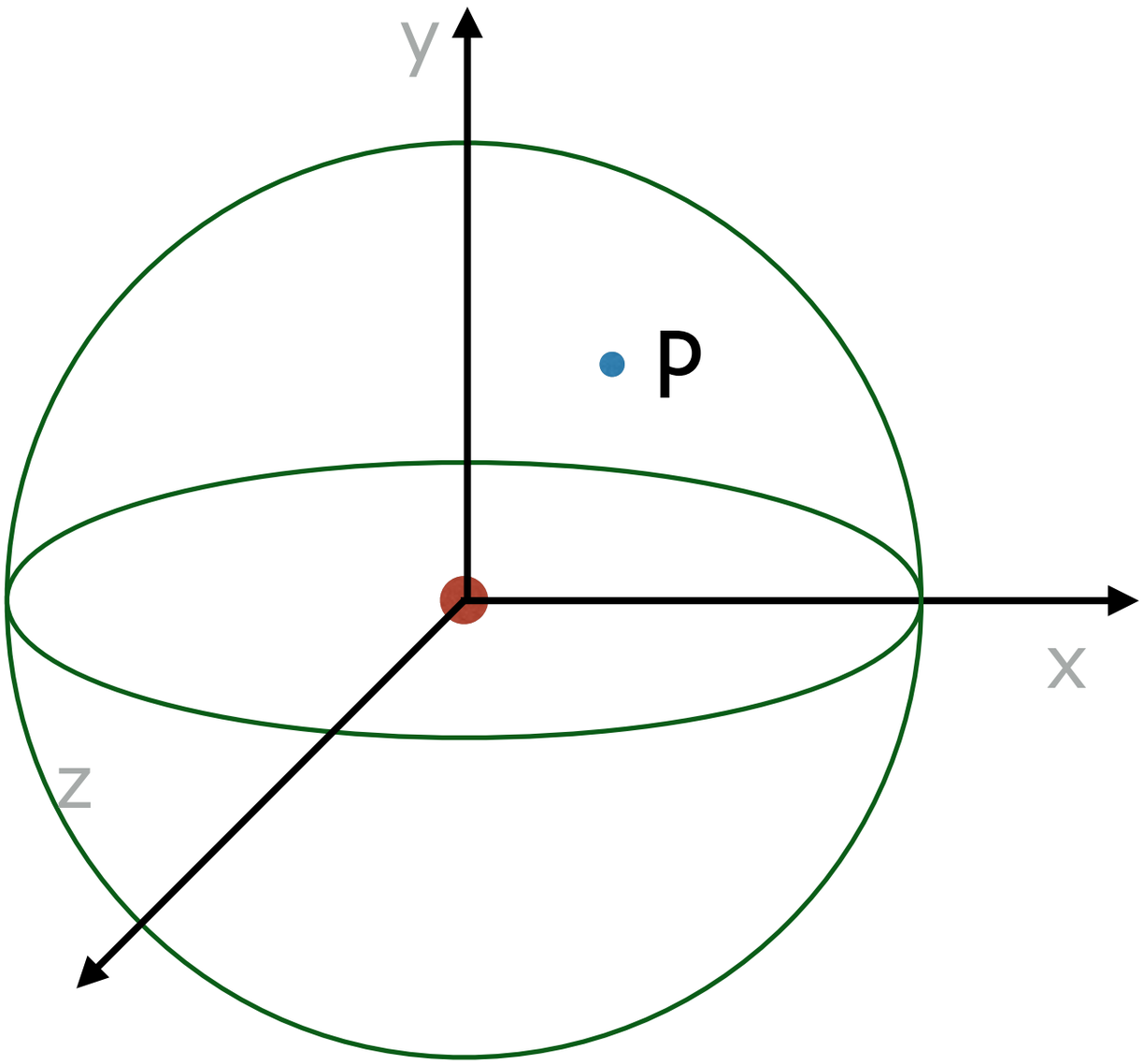
Texturing a Sphere

Sphere with radius r



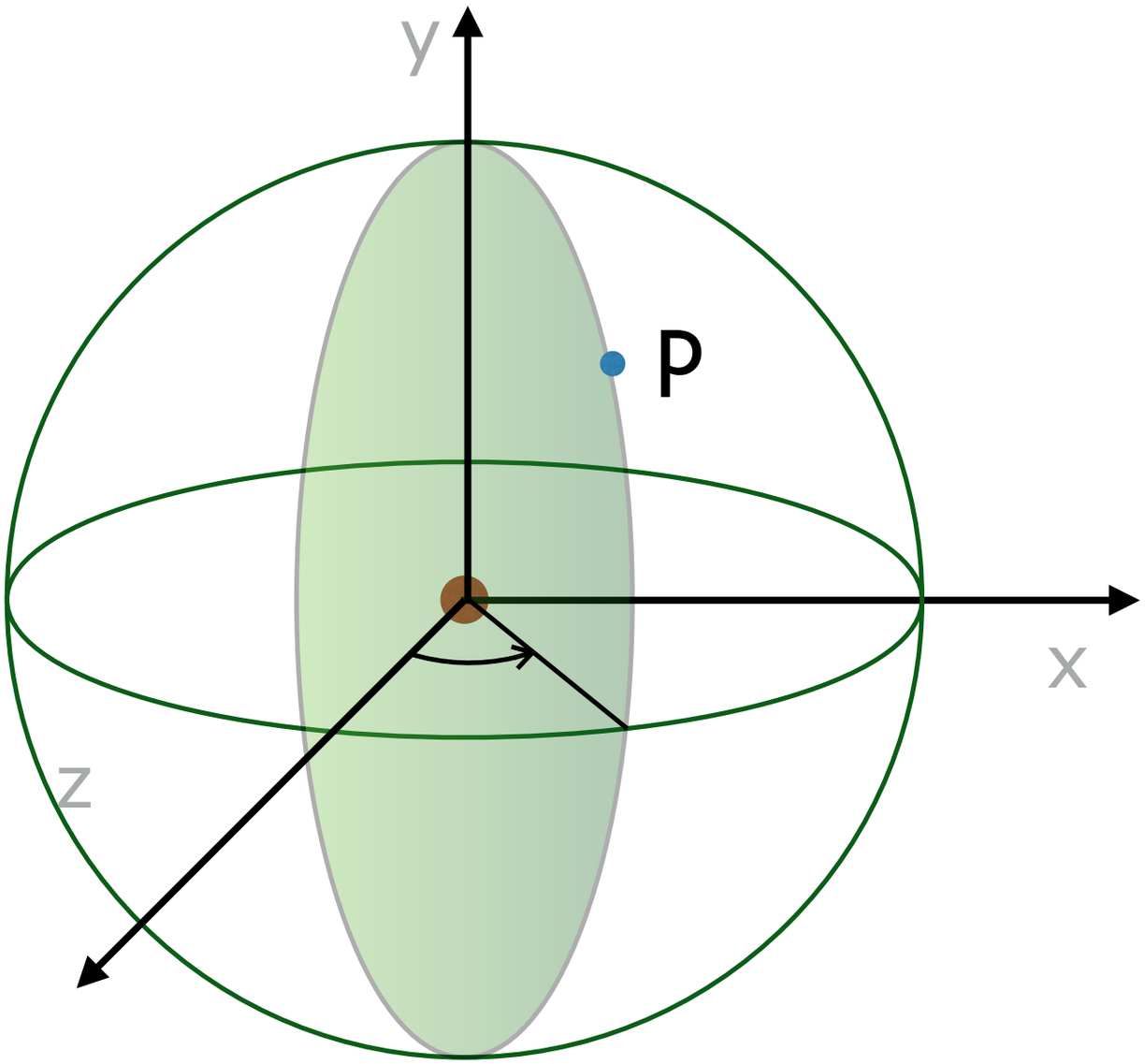
Texturing a Sphere

Sphere with radius r



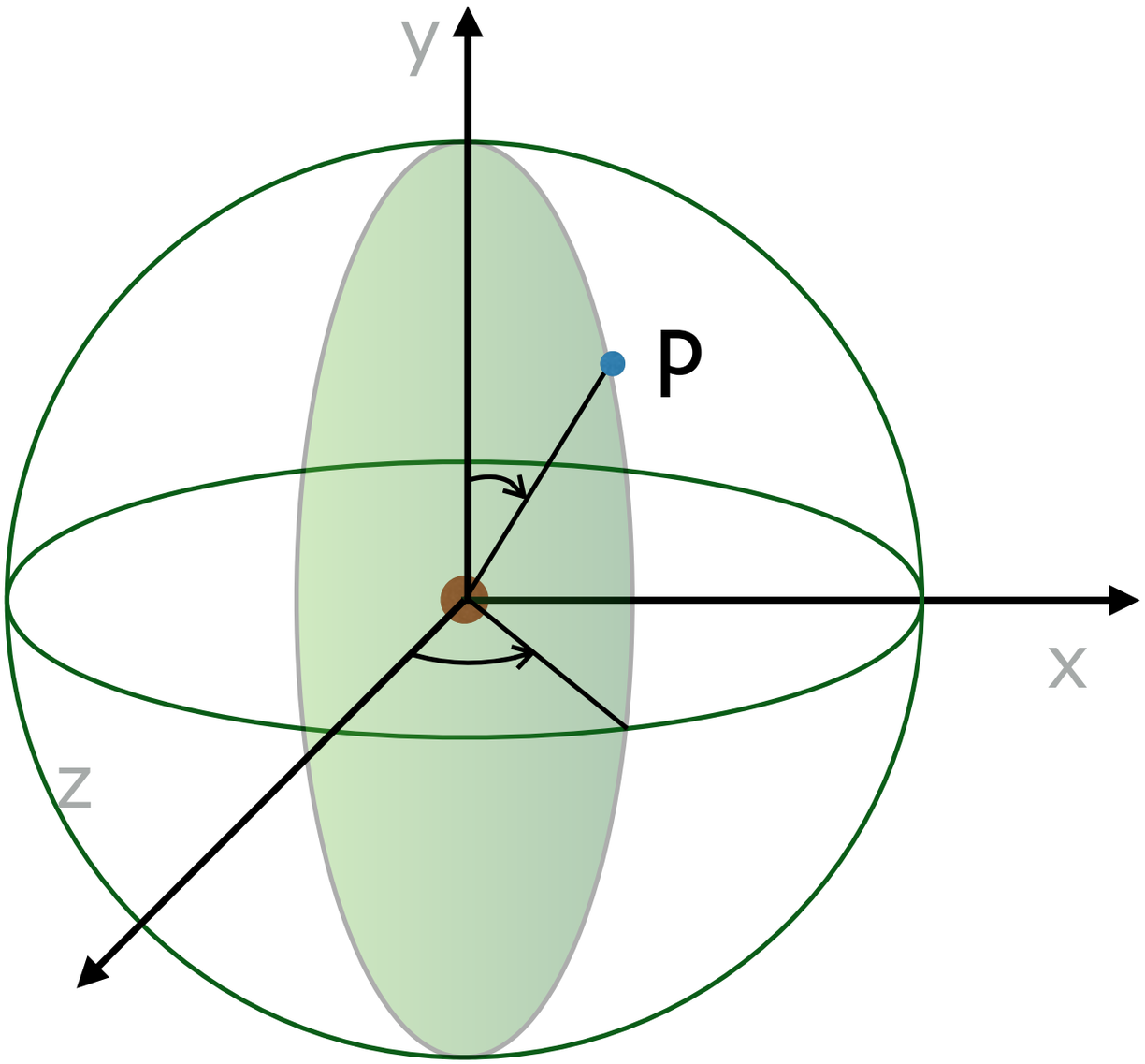
Texturing a Sphere

Sphere with radius r



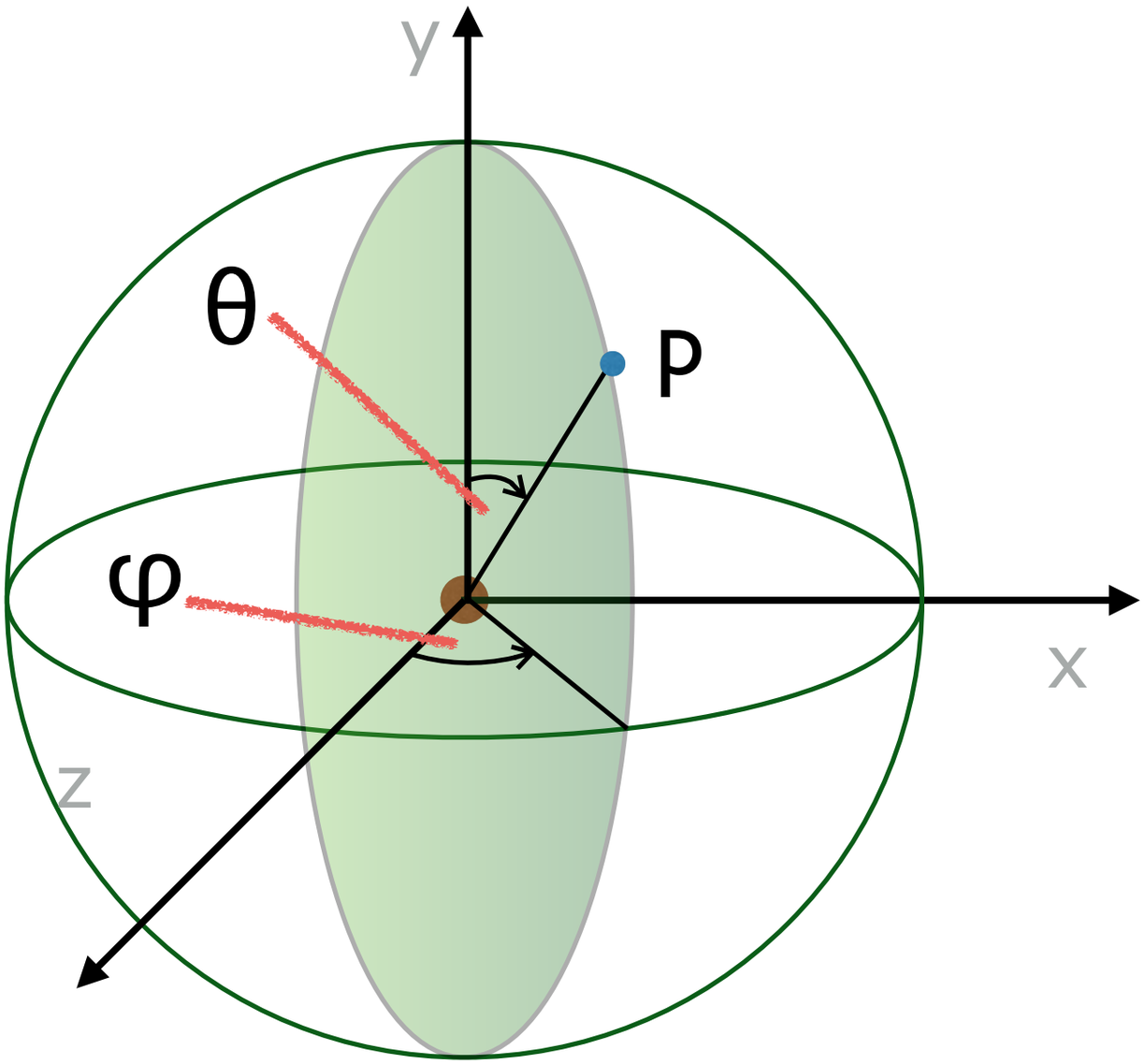
Texturing a Sphere

Sphere with radius r



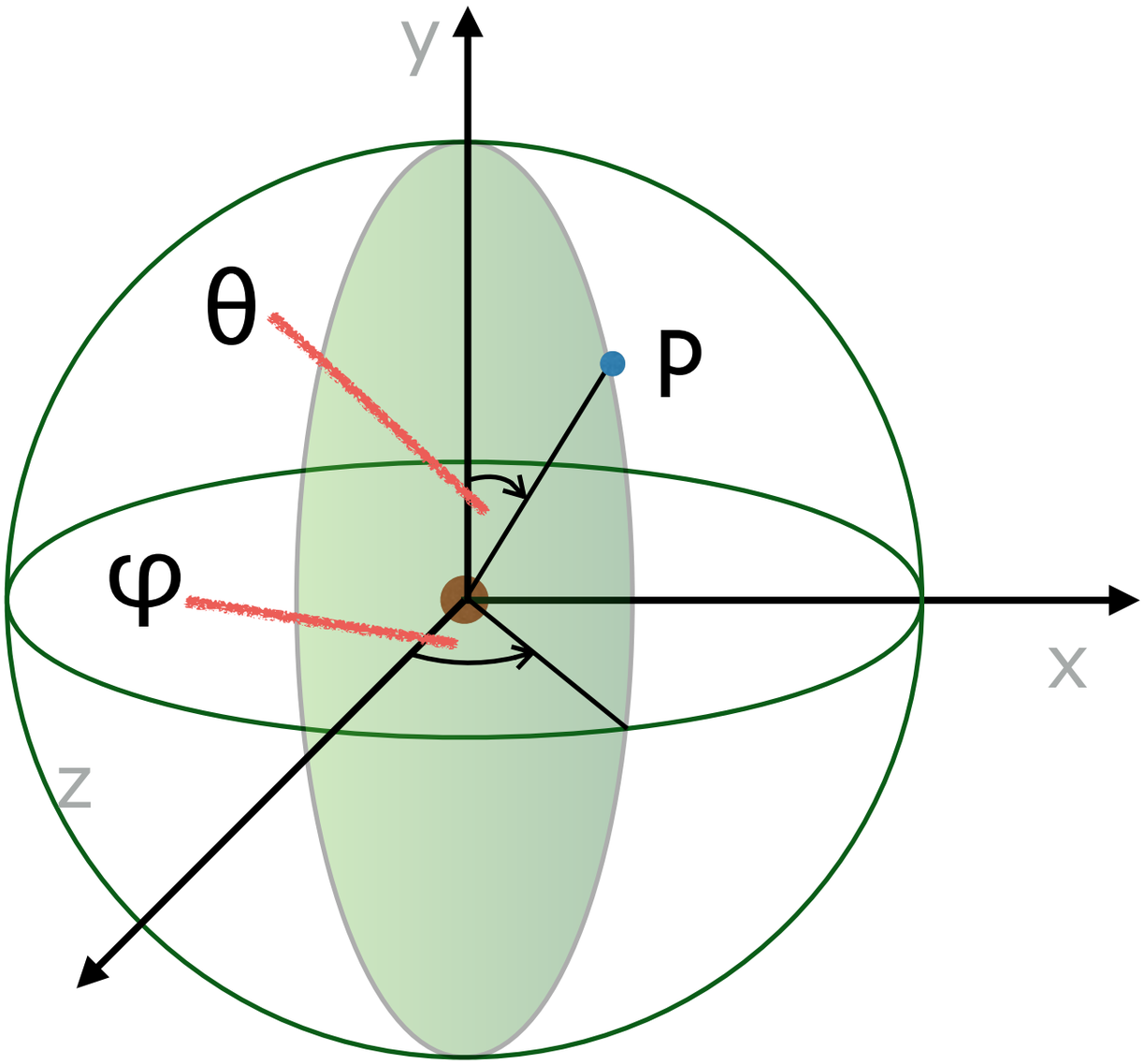
Texturing a Sphere

Sphere with radius r



Texturing a Sphere

Sphere with radius r

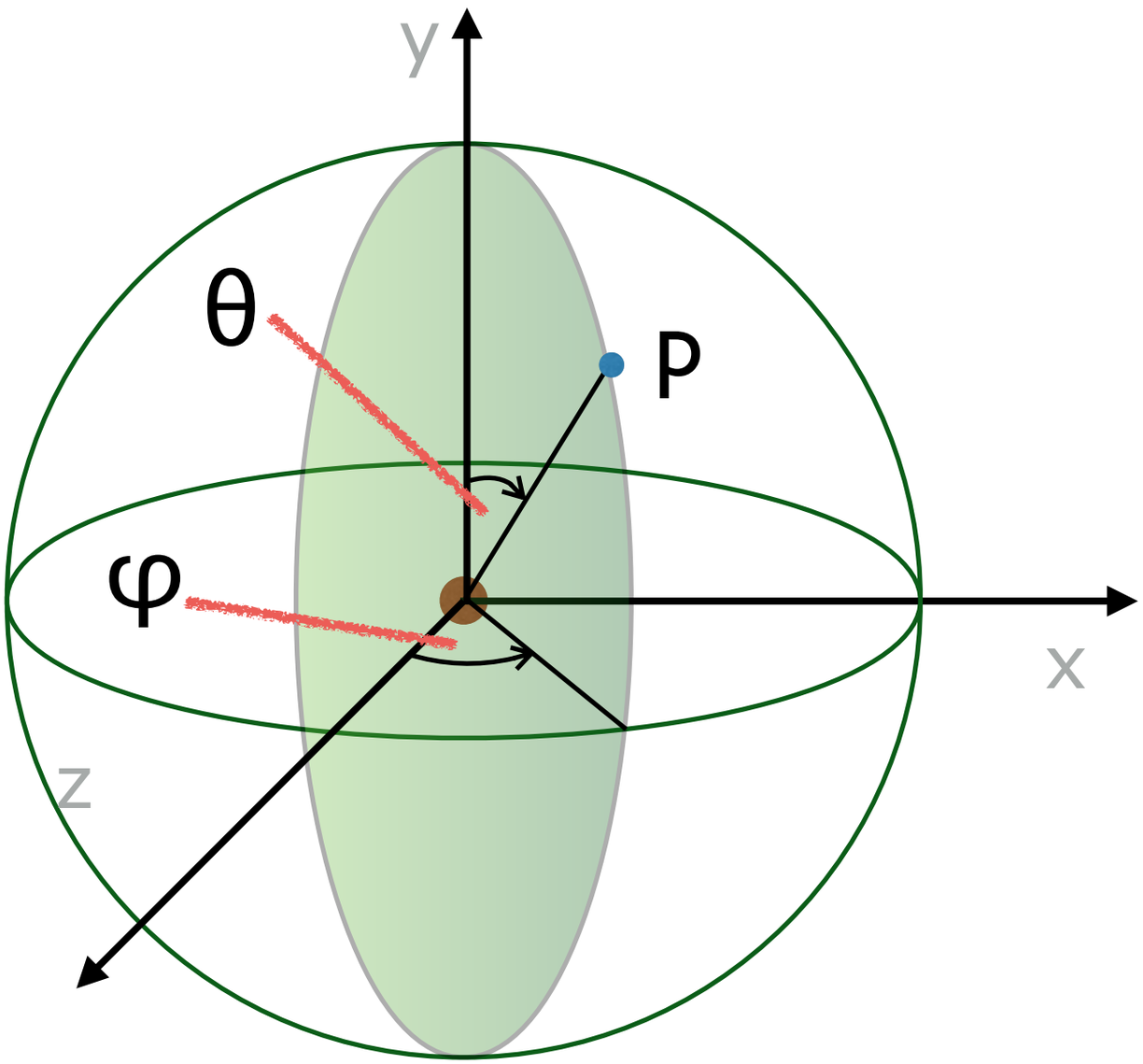


$u = \varphi/2\pi$

$v = 1-\theta/\pi$

Texturing a Sphere

Sphere with radius r



$n = p/r$

$\theta = \text{acos}(n_y)$

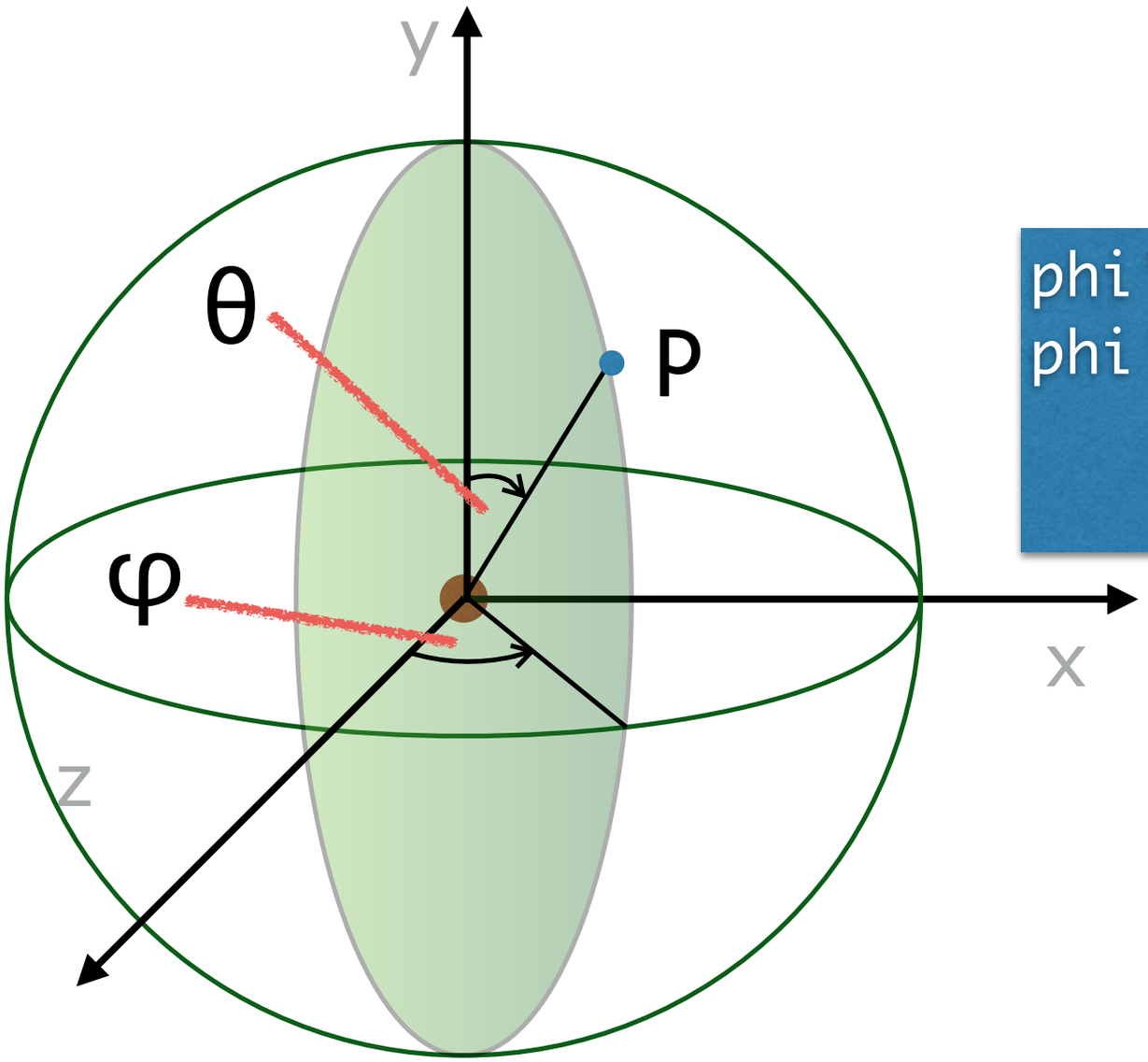
$\varphi = \tan^{-1}(n_x/n_z)$

$u = \varphi/2\pi$

$v = 1-\theta/\pi$

Texturing a Sphere

Sphere with radius r



$n = p/r$

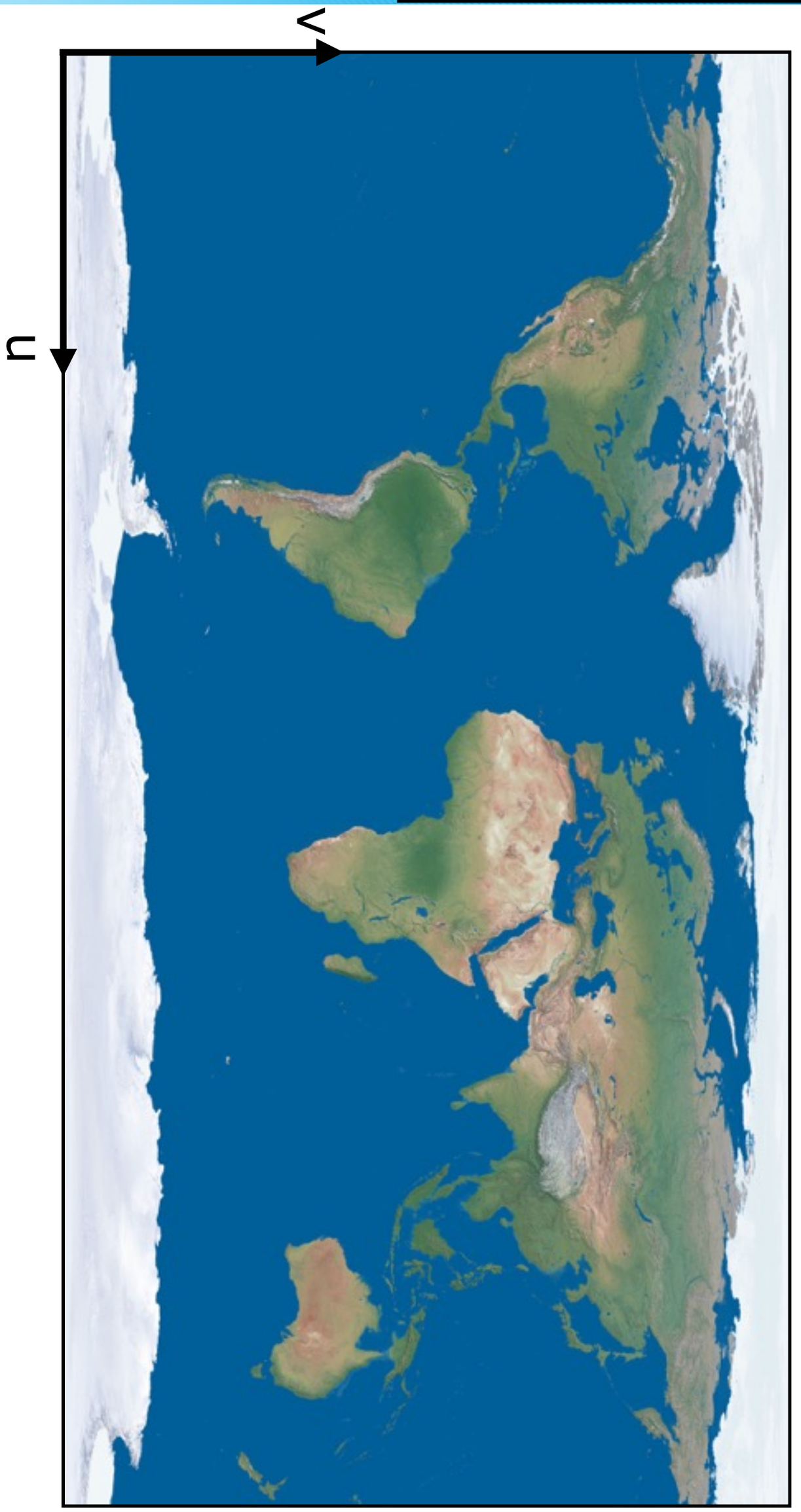
$\theta = \text{acos}(n_y)$

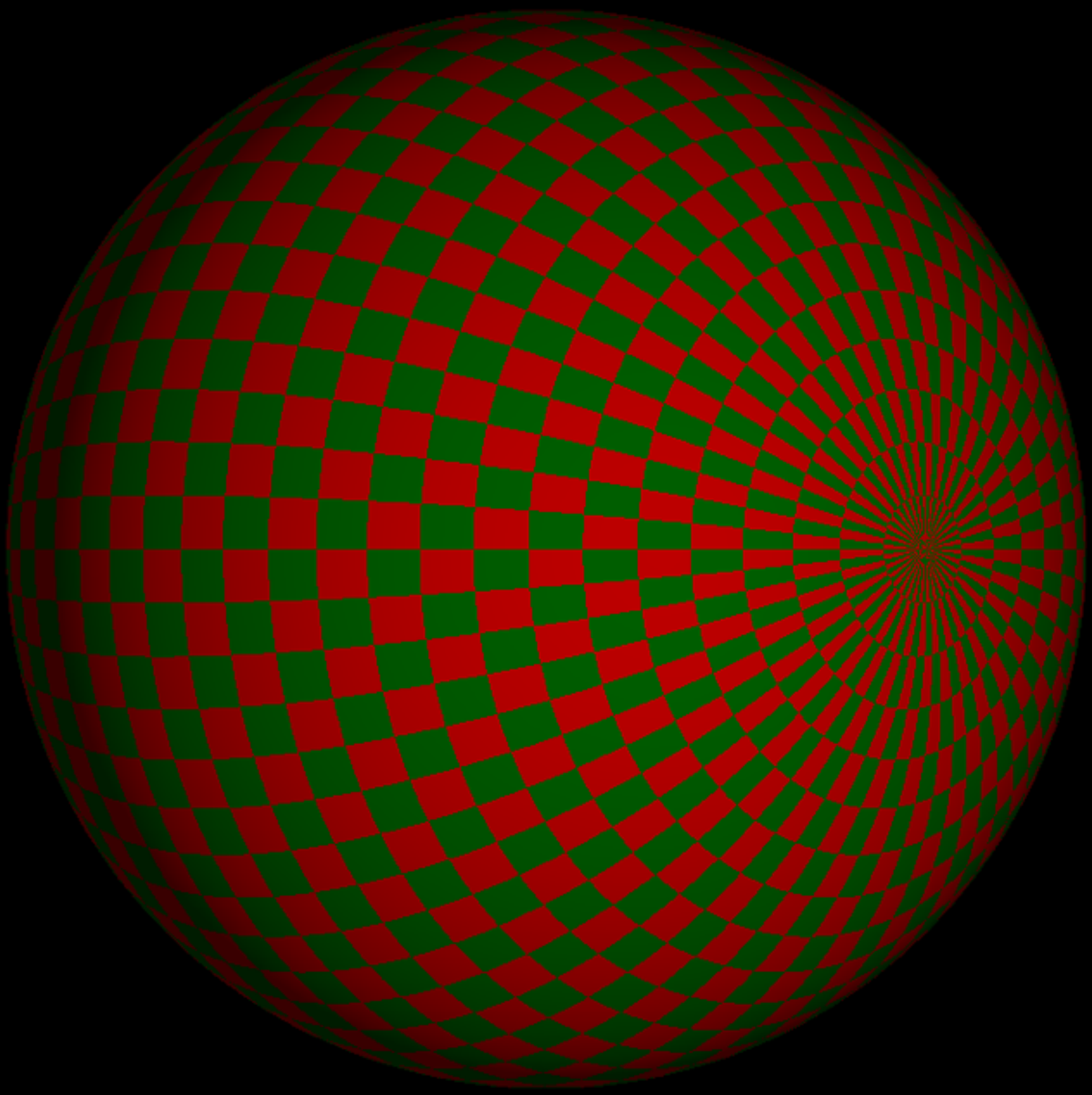
$\varphi = \tan^{-1}(n_x/n_z)$

```
phi' = atan2(nx,nz)
phi  = if phi' < 0
       then phi' + 2 * pi
       else phi'
```

$u = \varphi/2\pi$

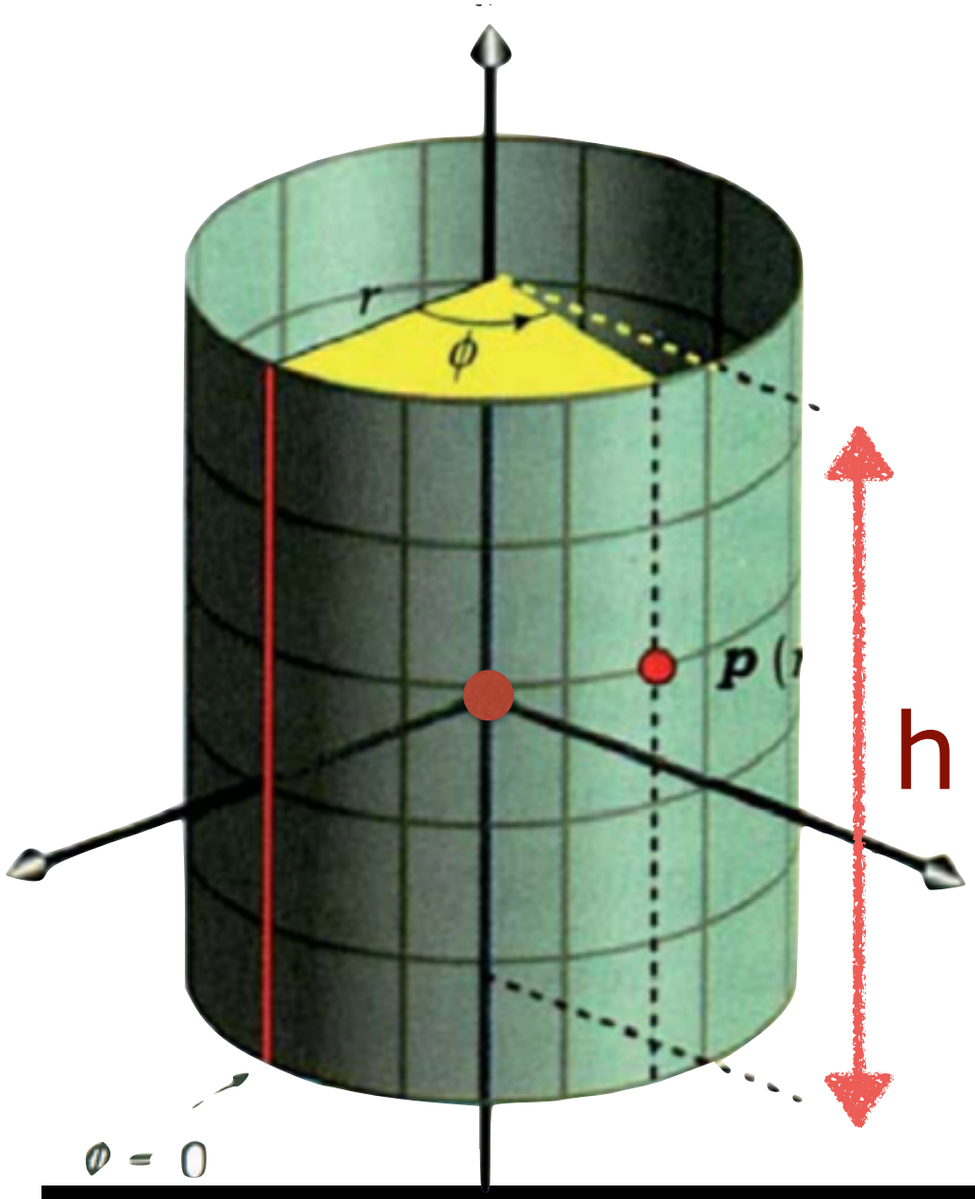
$v = 1-\theta/\pi$


v
u



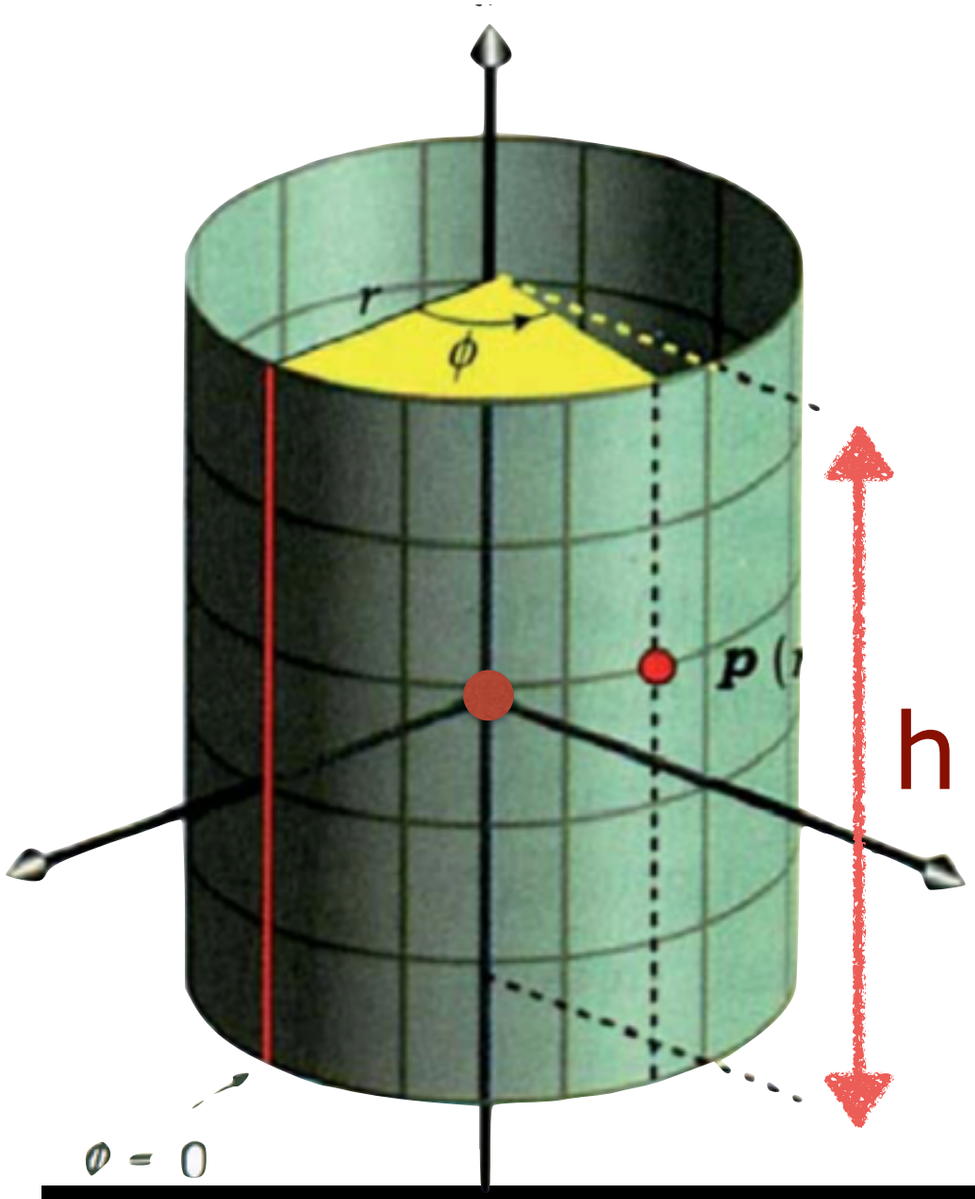
Open (hollow) Cylinder

- centre point (0,0,0)
- height h
- radius r



Open (hollow) Cylinder

- centre point (0,0,0)
- height h
- radius r



1. use equation for infinite cylinder

$$p_x^2 + p_z^2 = r^2$$

2. find intersections with ray

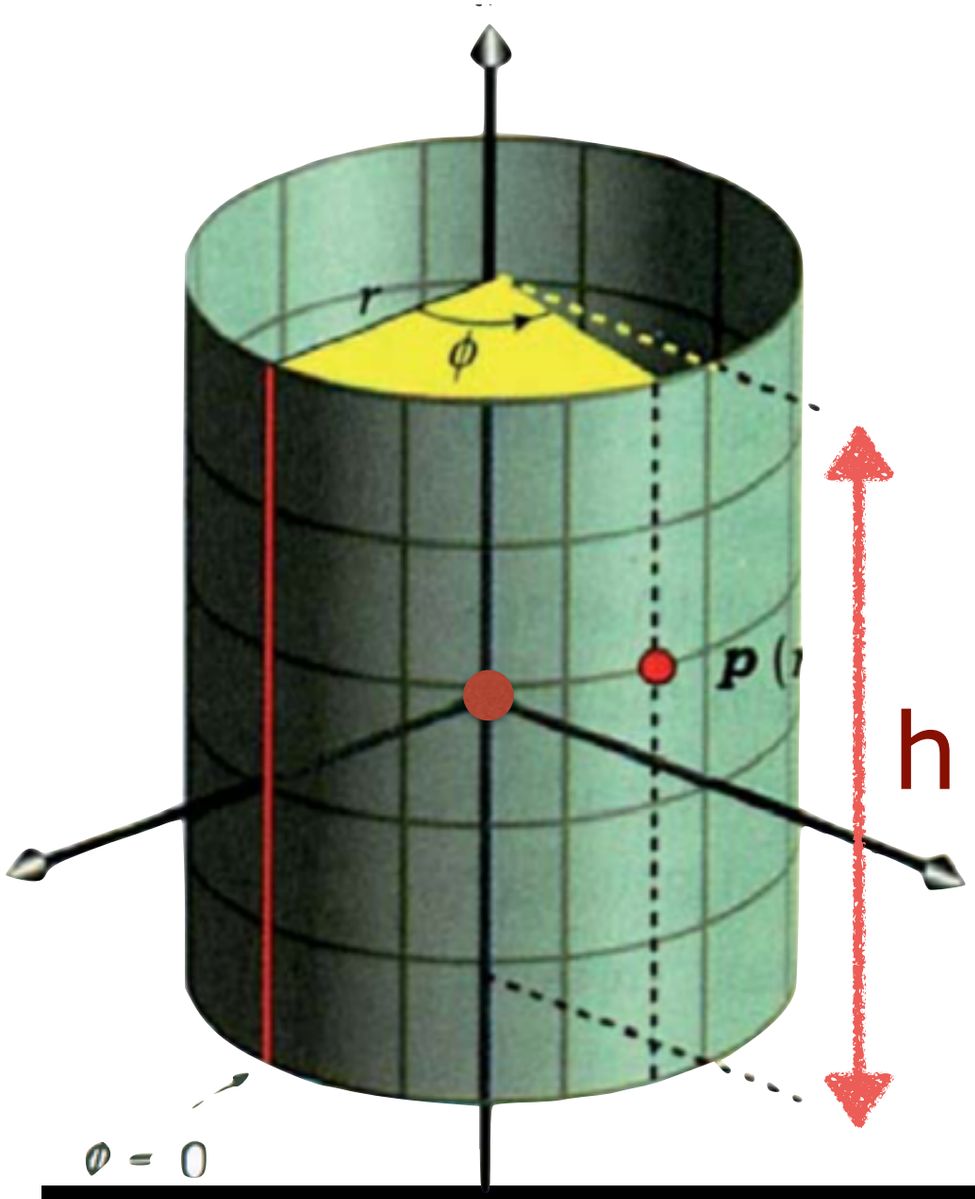
$$p = o + td$$

3. check that intersection point p satisfies

$$-h/2 \le p_y \le h/2$$

Open (hollow) Cylinder

- centre point (0,0,0)
- height h
- radius r



1. use equation for infinite cylinder

$$p_x^2 + p_z^2 = r^2$$

2. find intersections with ray

$$p = o + td$$

3. check that intersection point p satisfies

$$-h/2 \le p_y \le h/2$$

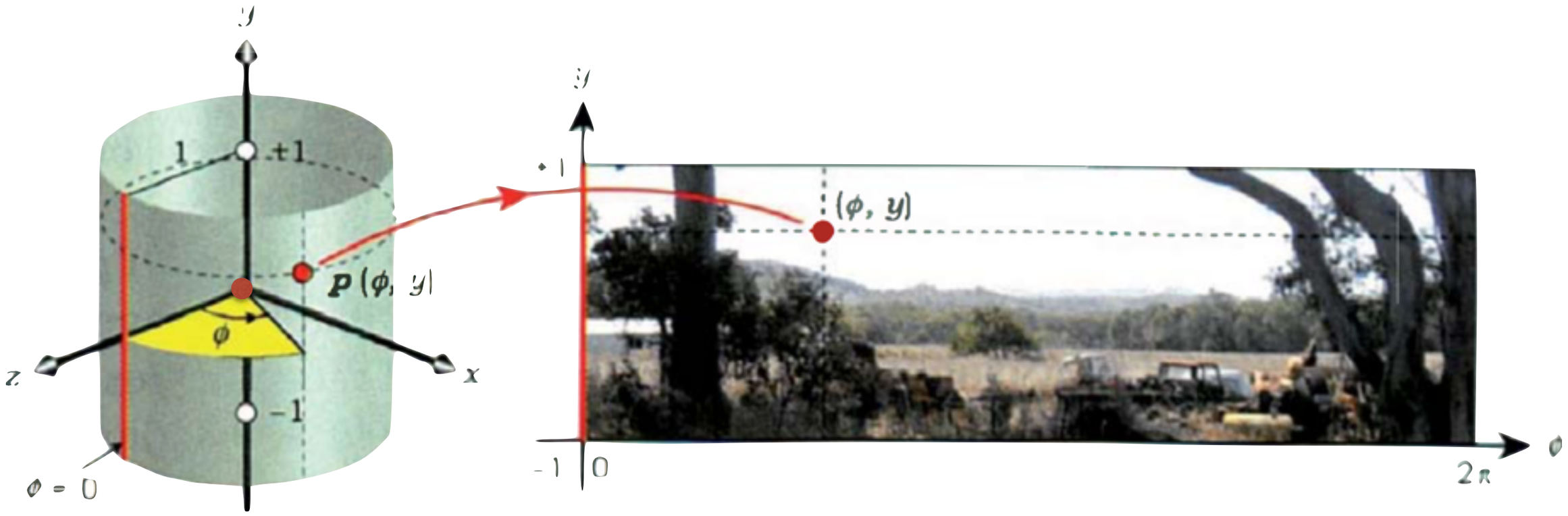
normal vector:

$$n = (p_x / r, 0, p_z / r)$$

Texture Mapping

$$u = \phi / 2\pi$$

$$v = \frac{p_y}{h} + \frac{1}{2}$$



Texture Mapping

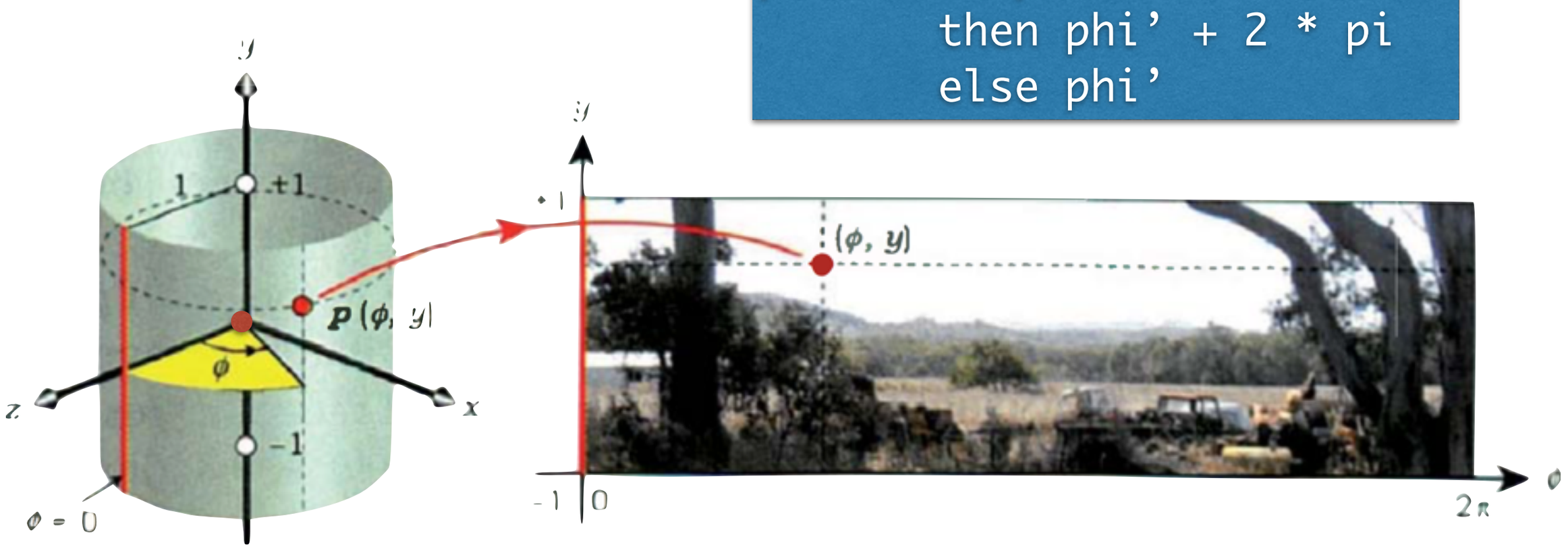
$$u = \phi / 2\pi$$

$$v = \frac{p_y}{h} + \frac{1}{2}$$

$$n = (p_x / r, 0, p_z / r)$$

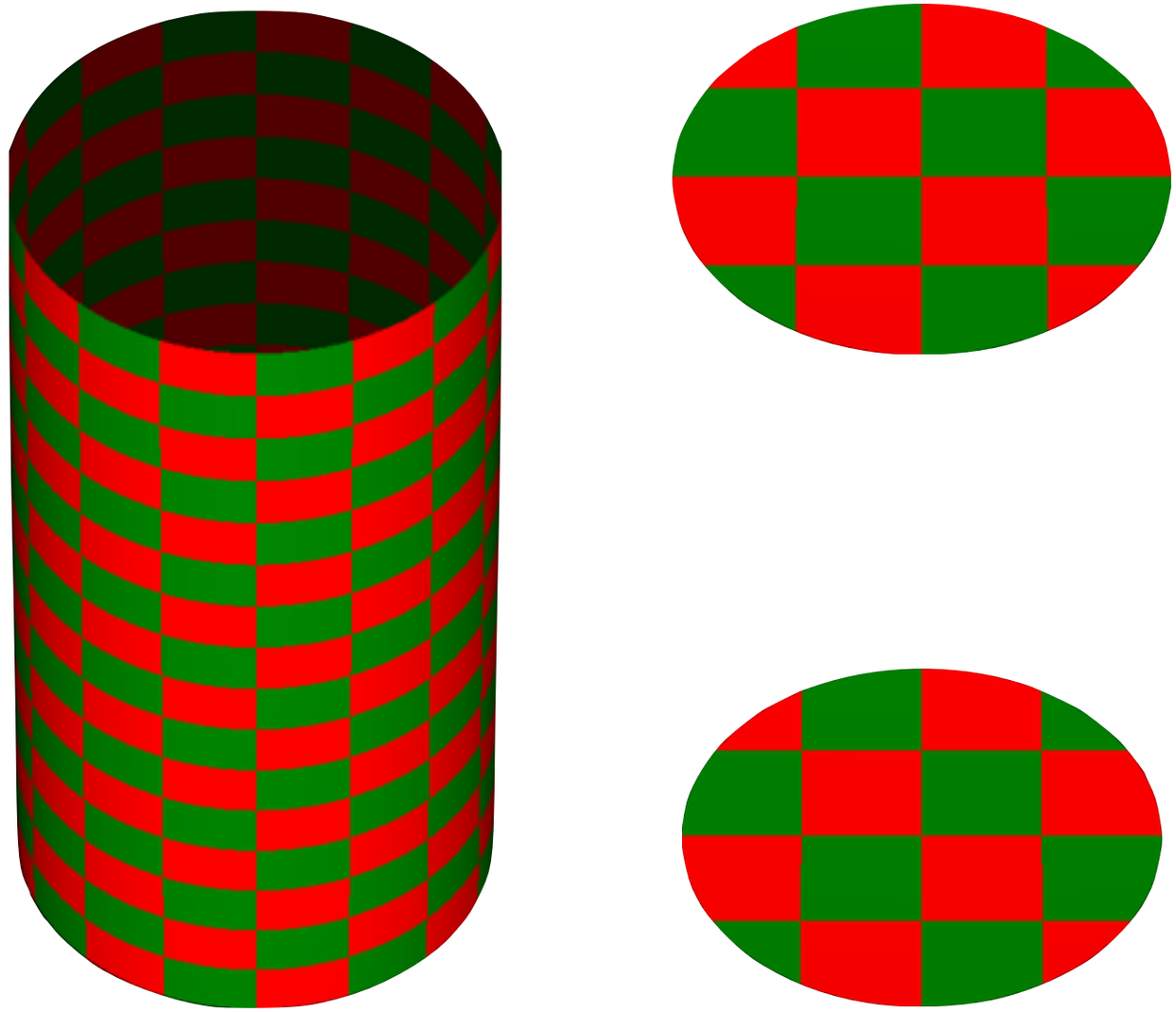
$$\phi = \tan^{-1}(n_x / n_z)$$

```
phi' = atan2(nx,nz)
phi  = if phi' < 0
       then phi' + 2 * pi
       else phi'
```



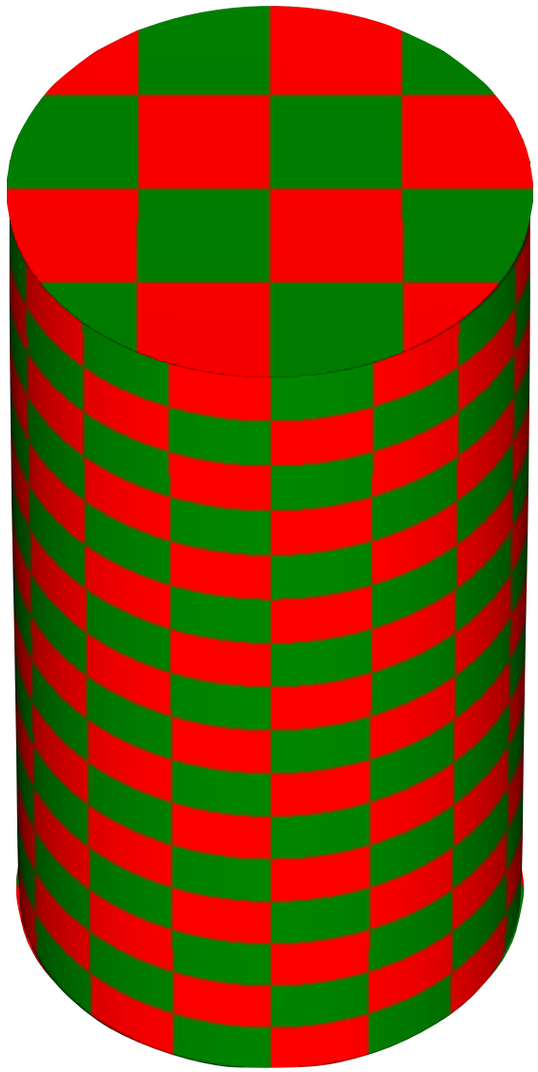
Solid Cylinder

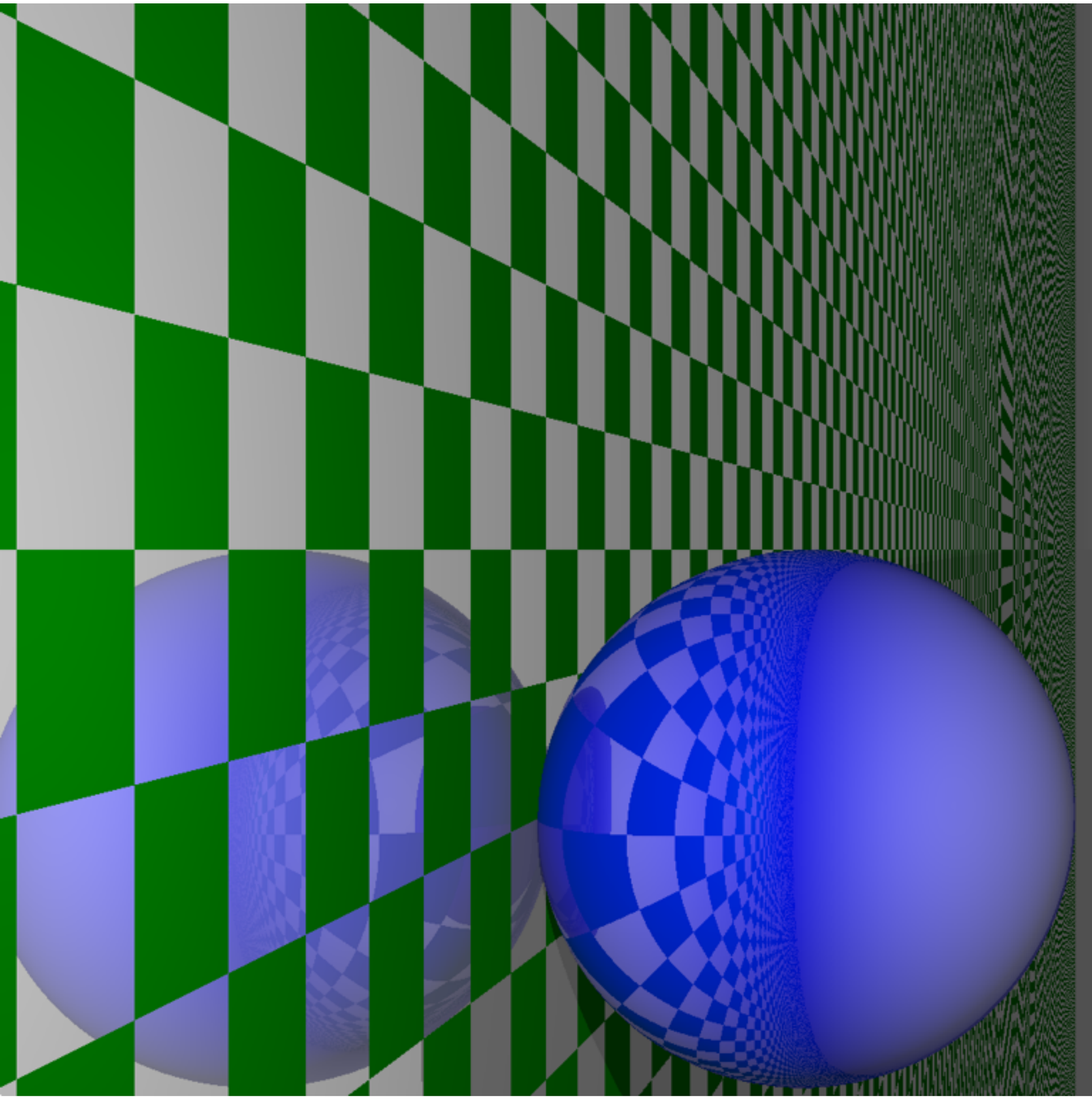
open cylinder + 2 discs



Solid Cylinder

open cylinder + 2 discs





Infinite Plane

- We only consider simple case:

  the x-y plane
- simple equation: $p_z = 0$
- (u,v) coordinates may be arbitrary (i.e. not necessarily in [0,1])

$$u = p_x$$

$$v = p_y$$

IT UNIVERSITY OF COPENHAGEN

Questions

IT UNIVERSITY OF COPENHAGEN

Triangle Meshes

Texture Coords in PLY Files

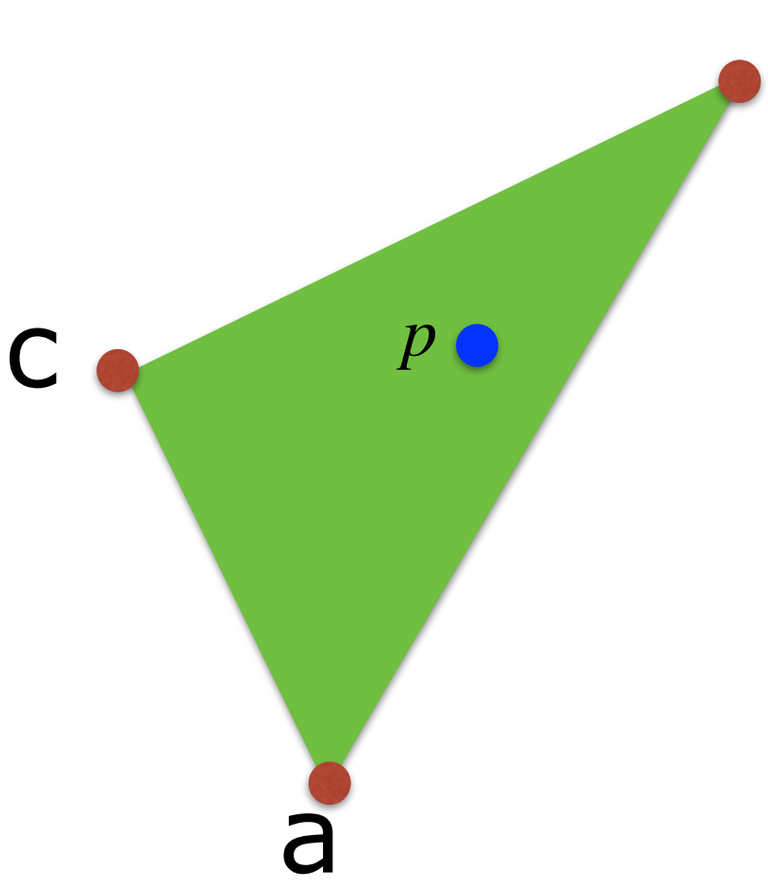
```
ply
format ascii 1.0
comment this is a bunny!!
element vertex 35947
comment has texture coords and normals
property float x
property float y
property float z
property float nx
property float ny
property float nz
property float u
property float v
property float fooness
property float barness
element face 69451
property list uint uint vertex_indices
end_header
…
```

Texture Coords in PLY Files

```
ply
format ascii 1.0
comment this is a bunny!!
element vertex 35947
comment has texture coords and normals
property float x
property float y
property float z
property float nx
property float ny
property float nz
property float u
property float v
property float fooness
property float barness
element face 69451
property list uint uint vertex_indices
end_header
…
```

texture coordinates

Recall



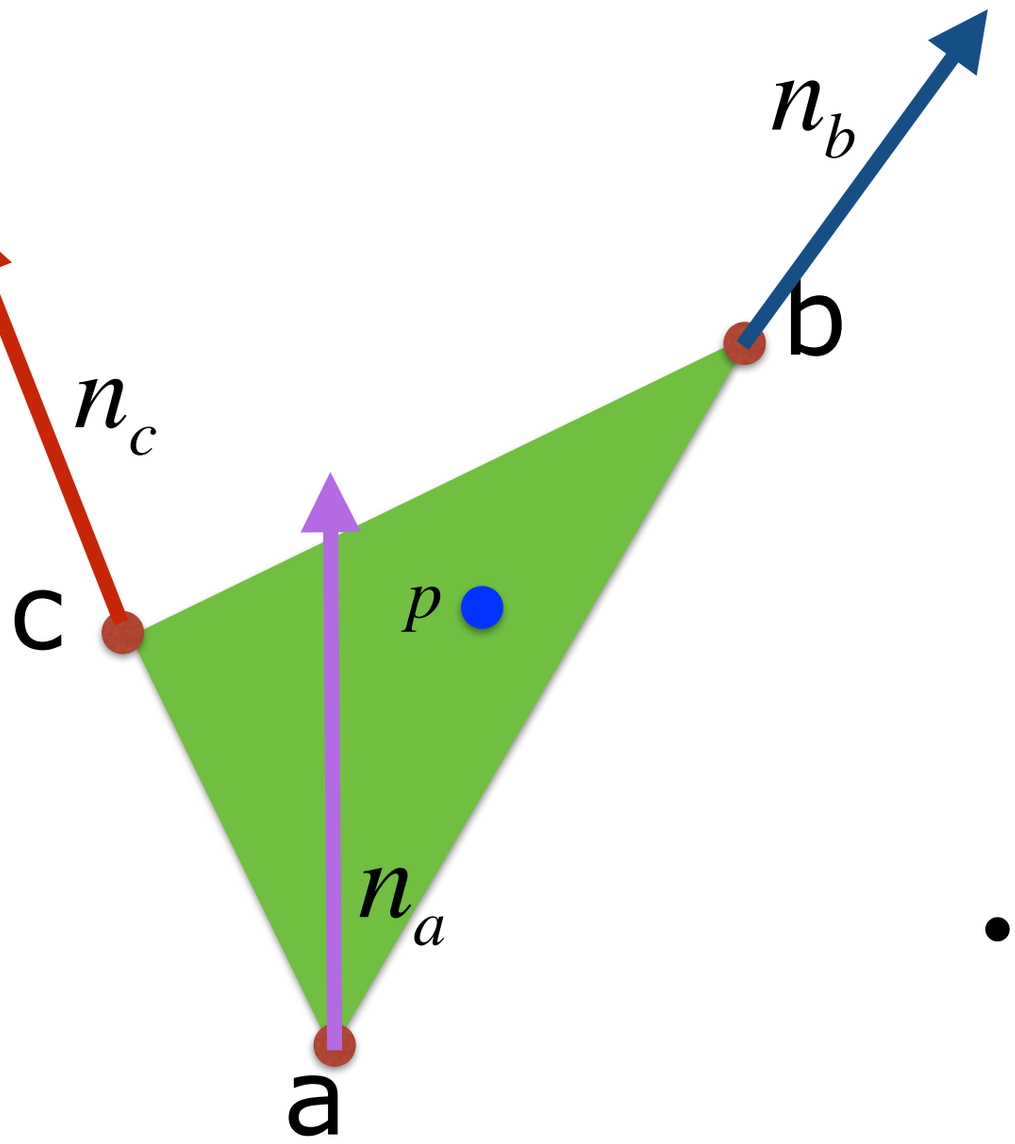
- Hit function calculates
  - distance t
  - coordinates β, γ

$$p = \alpha a + \beta b + \gamma c$$

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

- we used α,β,γ for interpolating normals
- texture coordinates are interpolated in the same way!

Recall



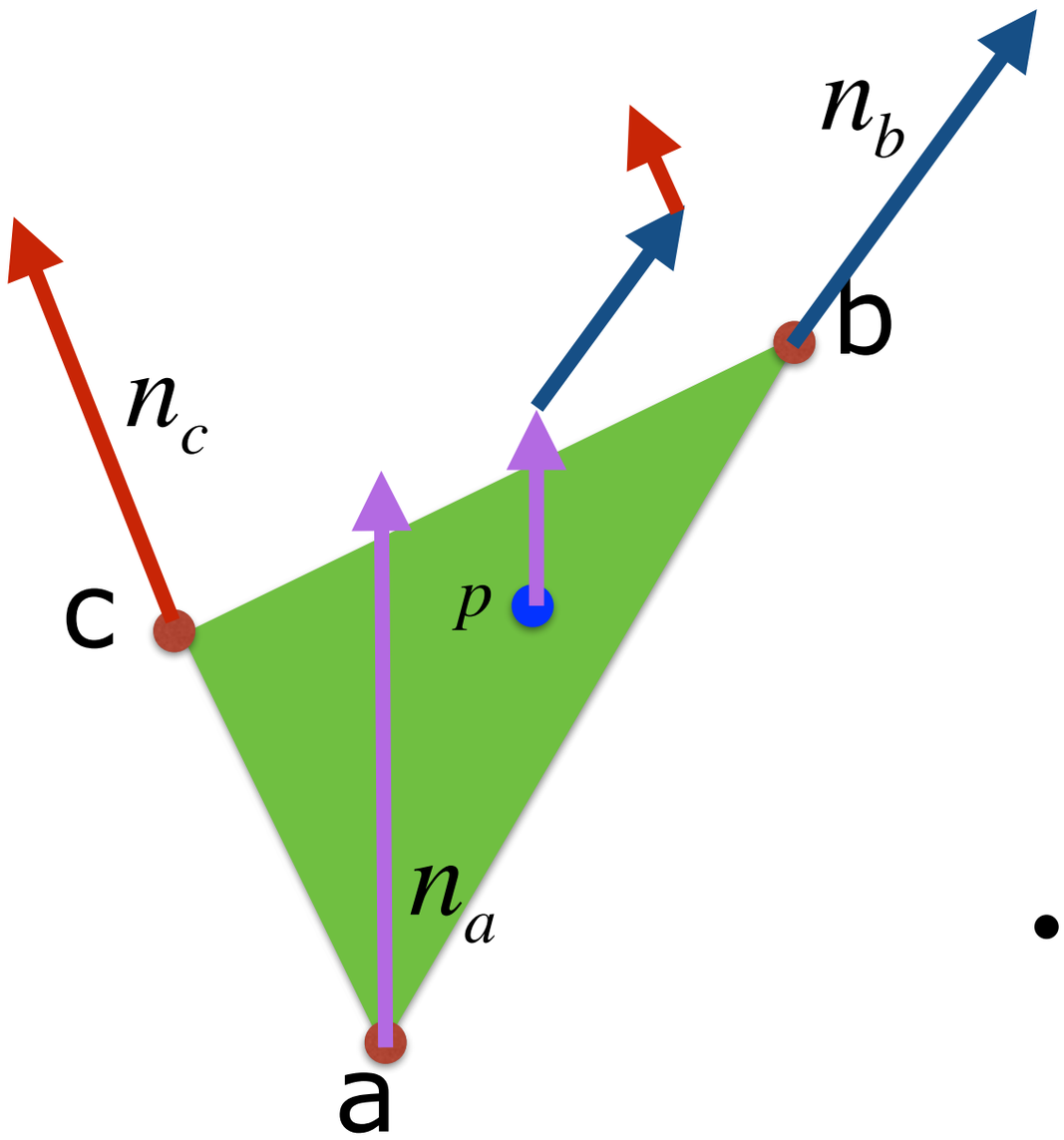
- Hit function calculates
  - distance t
  - coordinates β, γ

$$p = \alpha a + \beta b + \gamma c$$

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

- we used α,β,γ for interpolating normals
- texture coordinates are interpolated in the same way!

Recall



$$v = \alpha\, n_a + \beta\, n_b + \gamma\, n_c$$

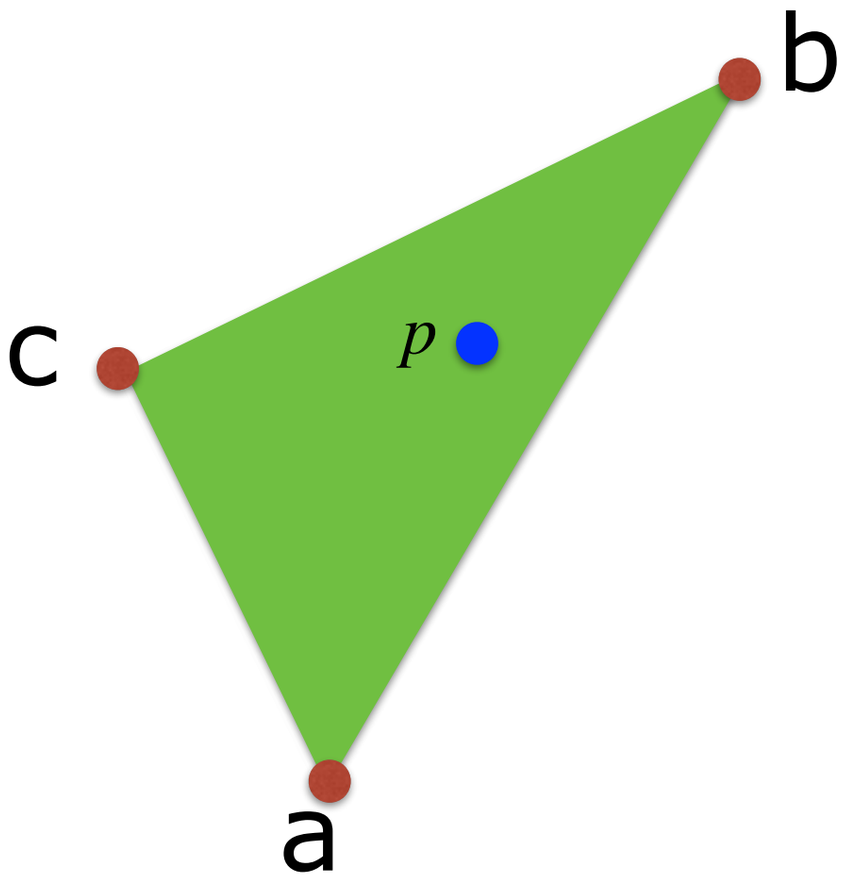
- Hit function calculates
  - distance t
  - coordinates β, γ

$$p = \alpha\, a + \beta\, b + \gamma\, c$$

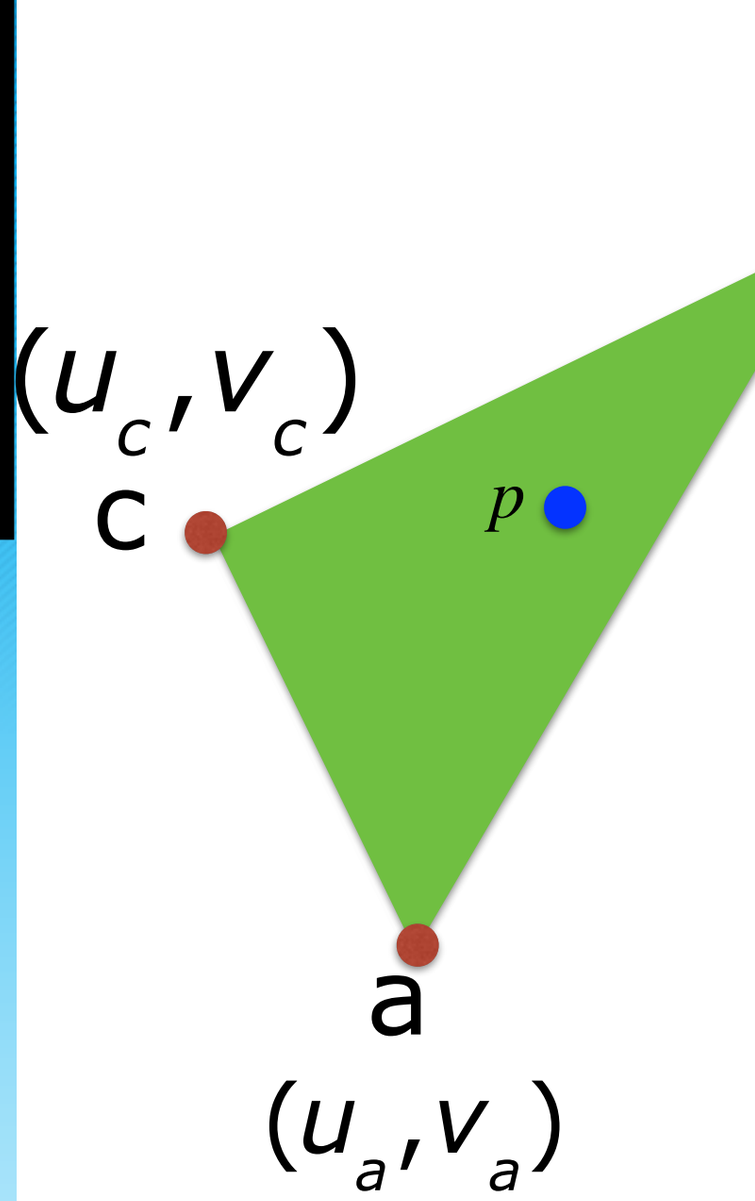
$$\alpha = 1 - \beta - \gamma$$

- we used α,β,γ for interpolating normals
- texture coordinates are interpolated in the same way!

Calculate Texture Coords



Calculate Texture Coords



$$v = \alpha v_a + \beta v_b + \gamma v_c$$

$$u = \alpha u_a + \beta u_b + \gamma u_c$$

Recap

1. find intersection of ray with a shape
2. calculate (u,v) coordinates of intersection point
3. lookup material at (u,v) coordinate
4. use the material to colour the pixel an to do calculate reflection

IT UNIVERSITY OF COPENHAGEN

API