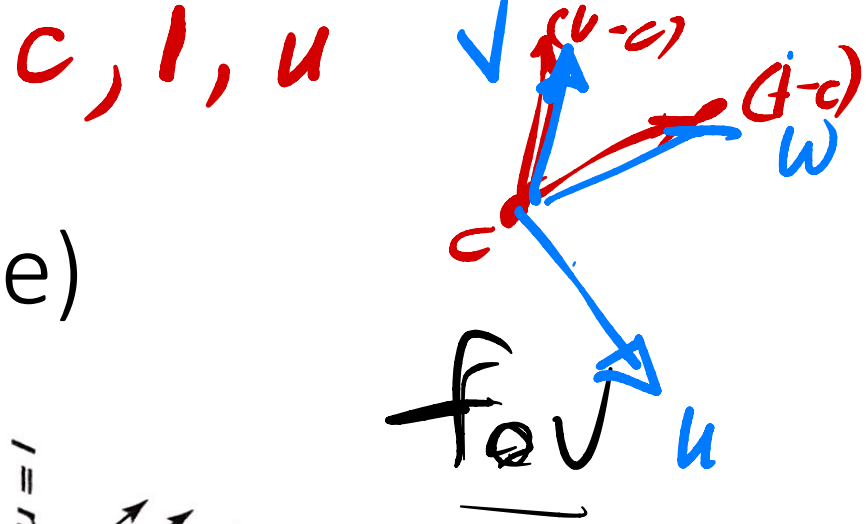
16 – more texture

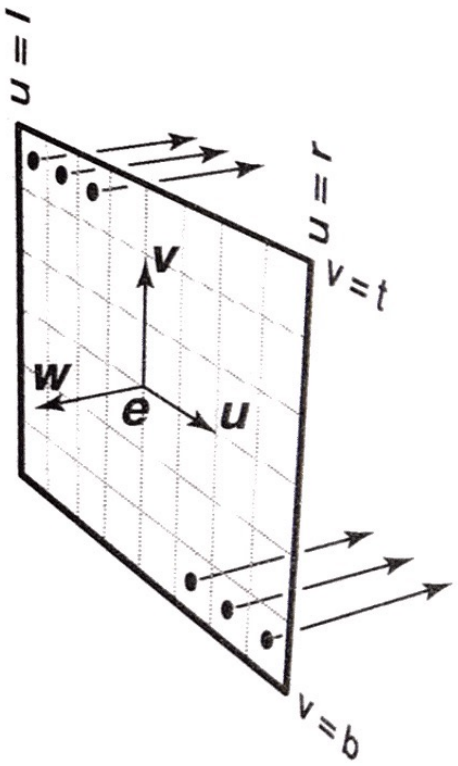
Bump Mapping and Environment Mapping

Basic Algorithm

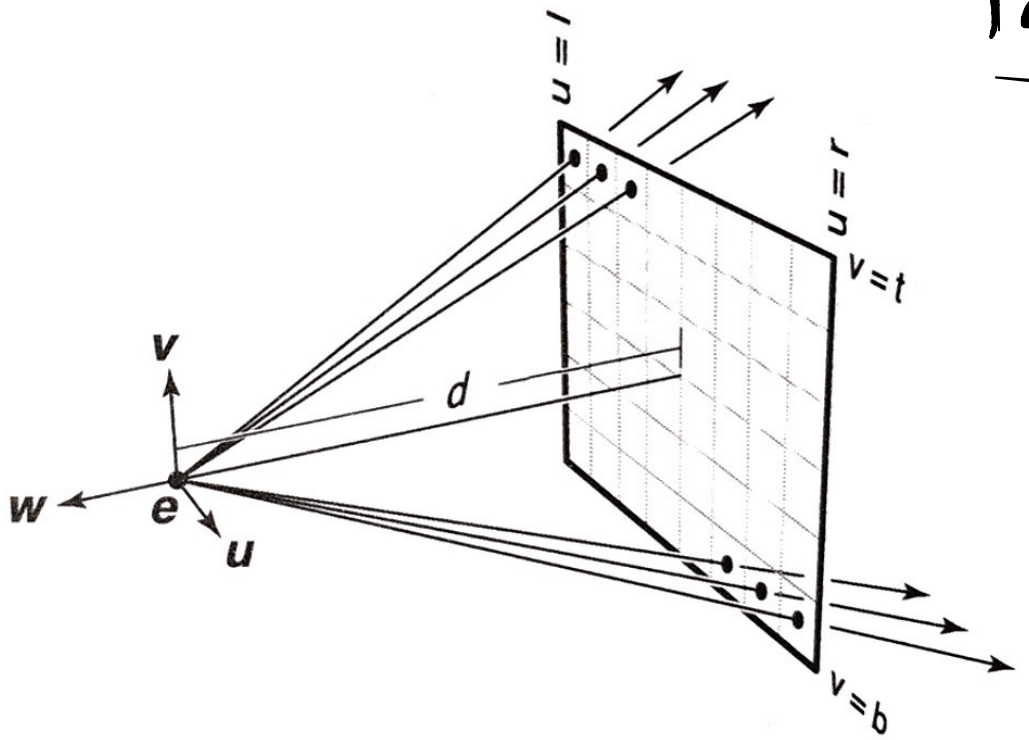
for each pixel (x_s, y_s)
  create a Ray R from eye through (x_s, y_s)
  for each object O_i in scene
    if R intersects O_j & it's the closest so far
      record this intersection
  shade pixel based on nearest intersection
    (recursively for ref & transmition)

Eye Rays: Depends on Projection (Orthographic, Perspective, Oblique)



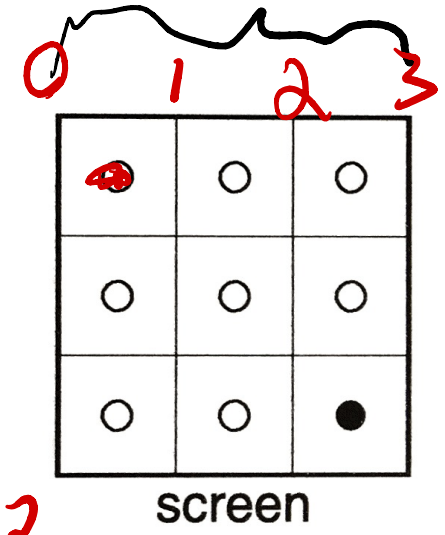


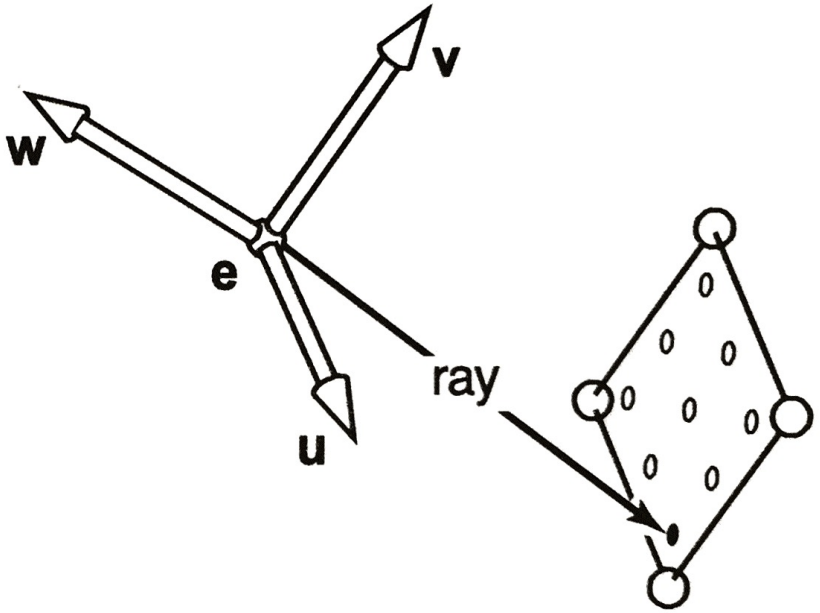
Parallel projection
same direction, different origins



Perspective projection
same origin, different directions

FoV = FoVy

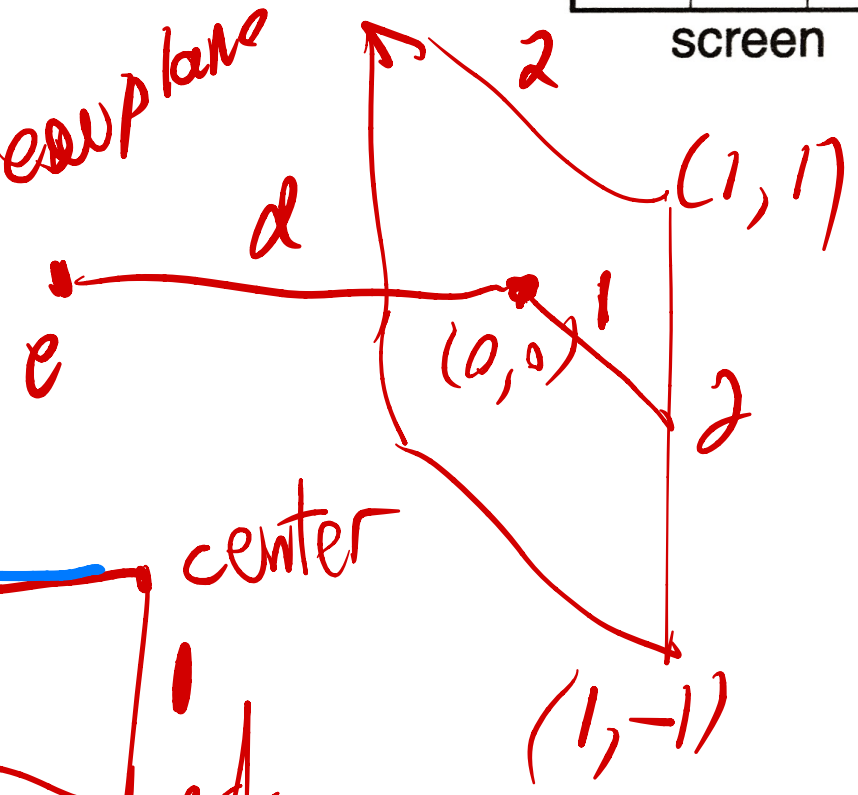




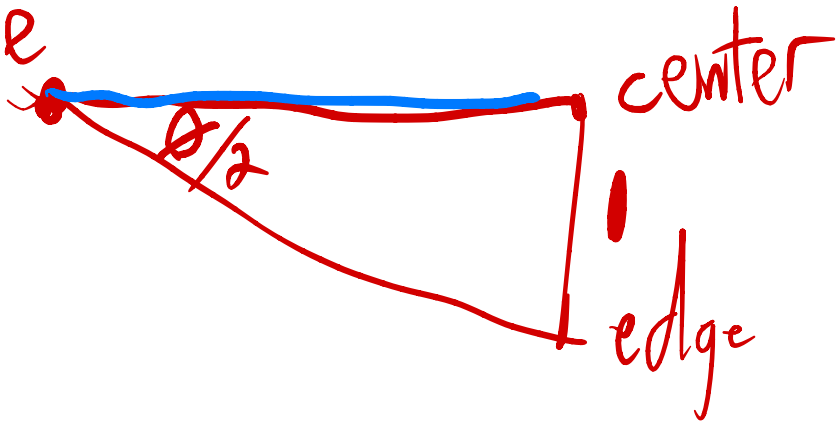
Parametric eq'n:

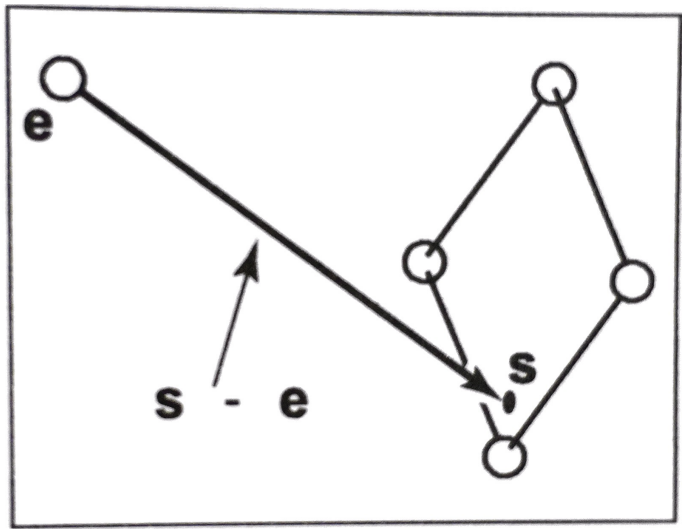
$$p(t) = e + t(s - e)$$

1) fov θ
2) size of viewplane



$$d = \frac{1}{\tan(\frac{\theta}{2})}$$





$$u_s = -1 + \frac{2i}{w}$$

$$v_s = -1 + \frac{2j}{h}$$

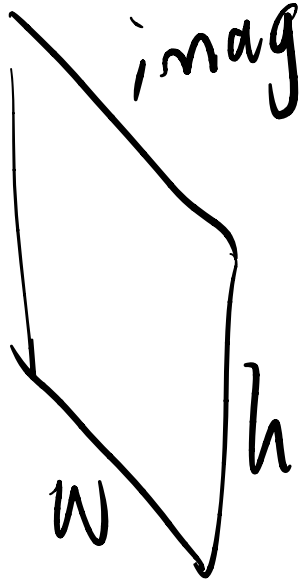
u & v

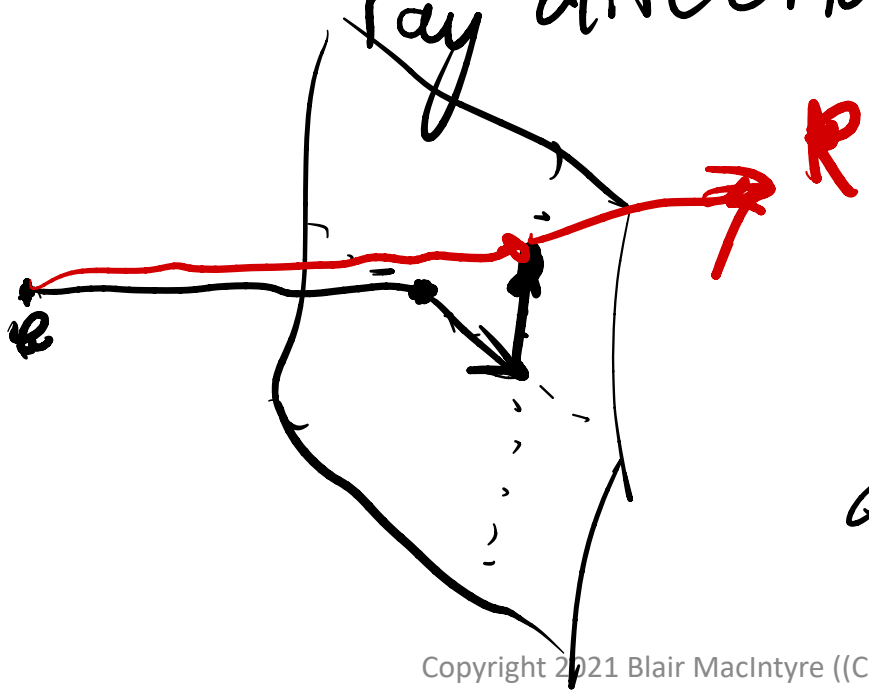
$-1 \le i$ range

u

v

w





for

$$\text{ray direction} = \frac{s - e}{|s - e|}$$

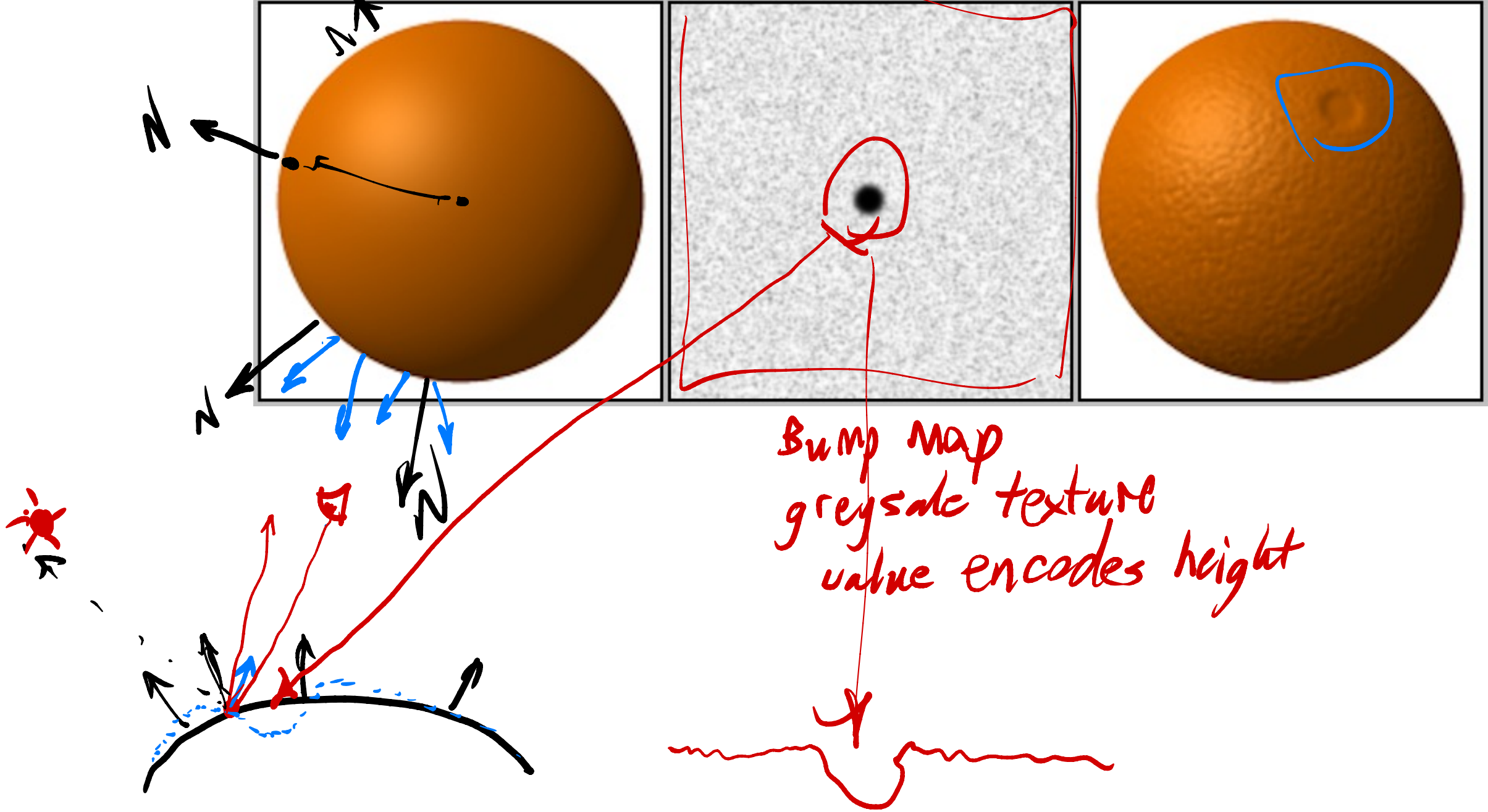
$$= -dw + u_s u + v_s v$$

$$\text{origin} = e$$

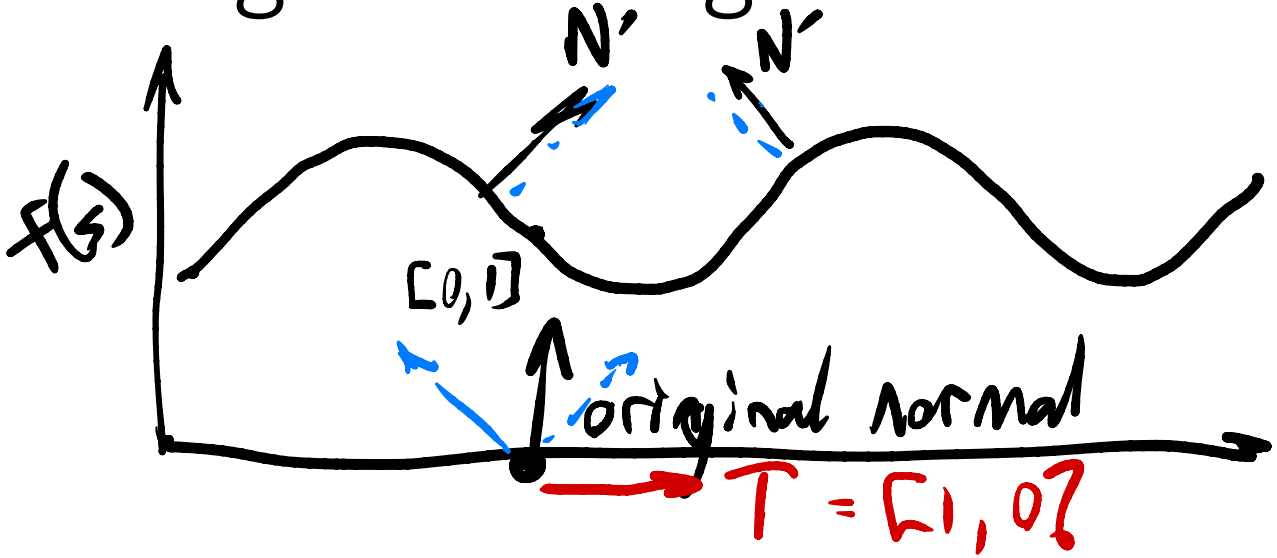
Bump Mapping

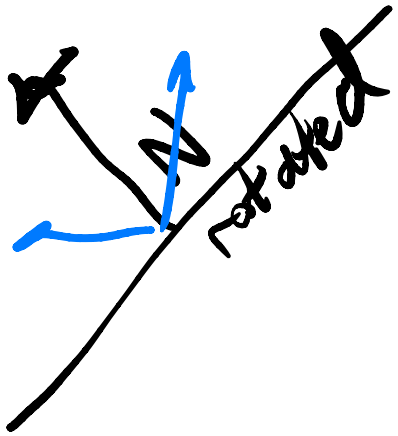
From this

to this



High level algorithm





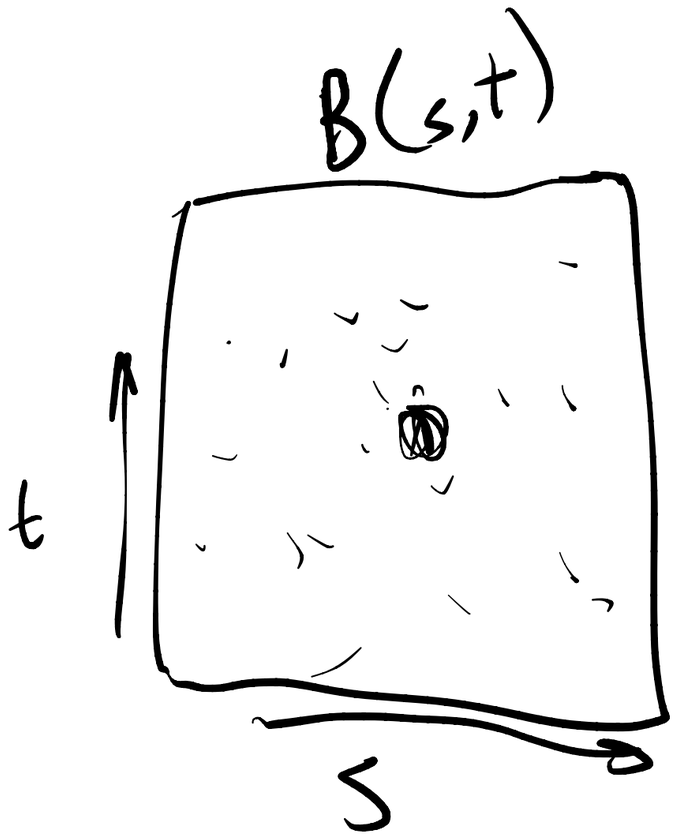
$N = N - f(s)[1,0]$

T

$f(s) = [1 \ldots 1]$

$0 \ldots 255 \rightarrow -1 \ldots 1$

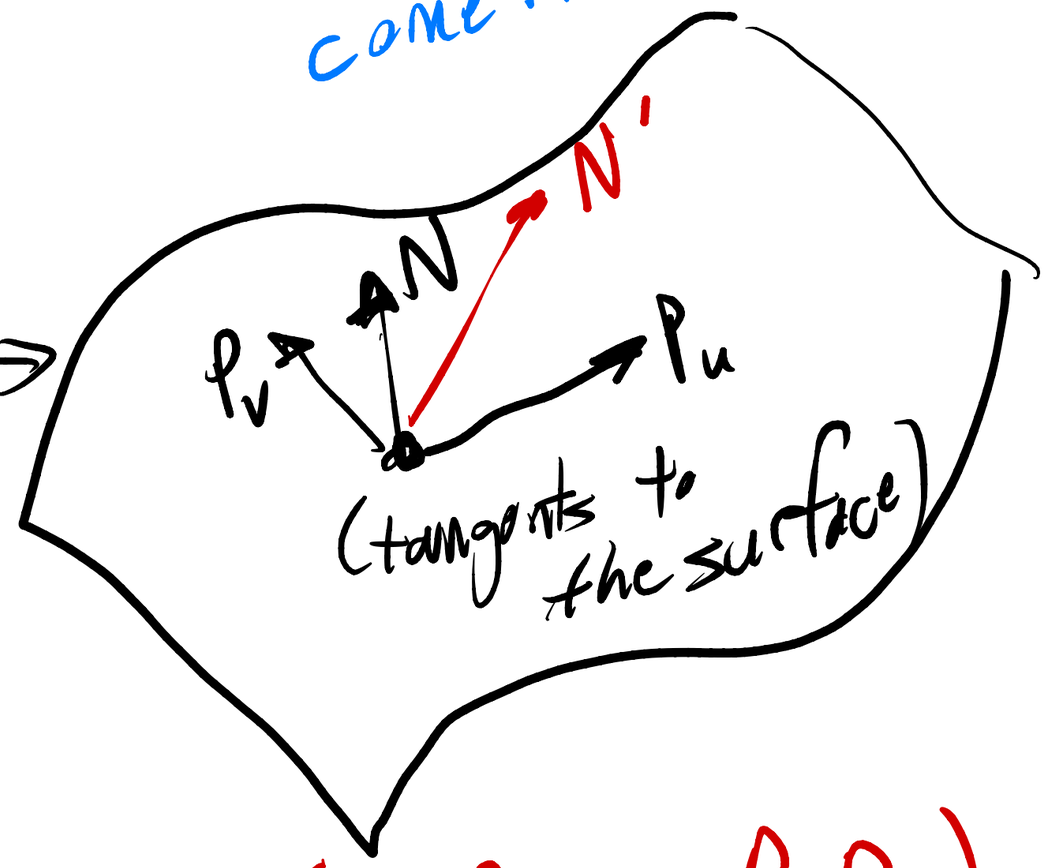
texture does not encode slope.



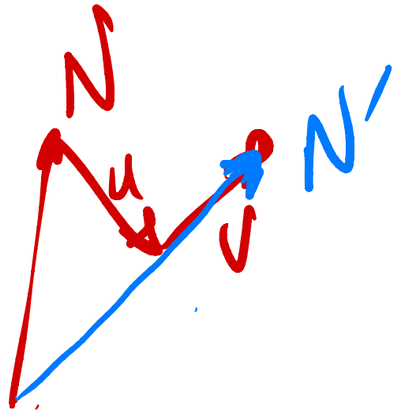
$$B_s = \frac{\partial B}{\partial s}$$

$$B_T = \frac{\partial B}{\partial T}$$

where do P_V & P_u ... ?

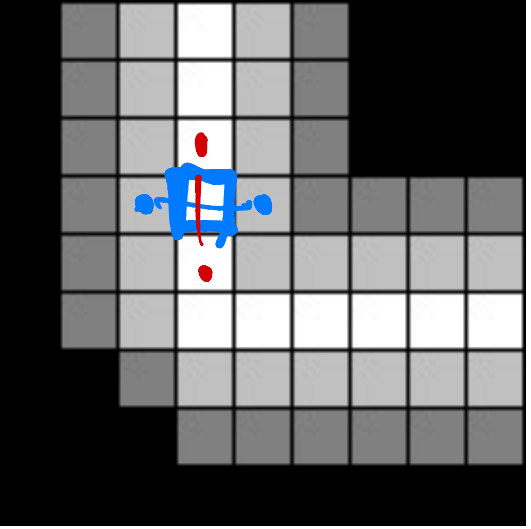


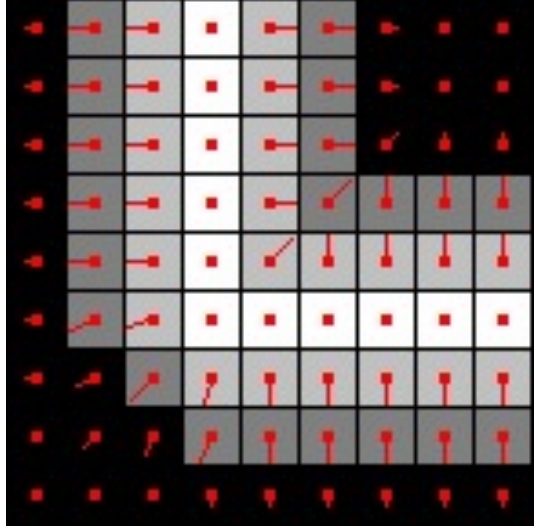
come from



$$N = N' - (B_s P_u + B_t P_V)$$

slopes (-1 ... 1)

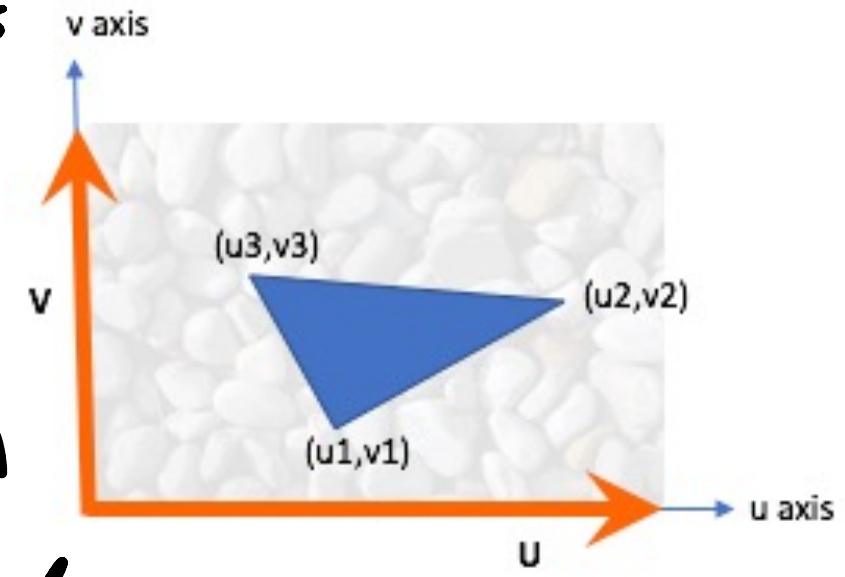




x_gradient = pixel(x-1, y) - pixel(x+1, y)
y_gradient = pixel(x, y-1) - pixel(x, y+1)

where do we get Bs & Bt?

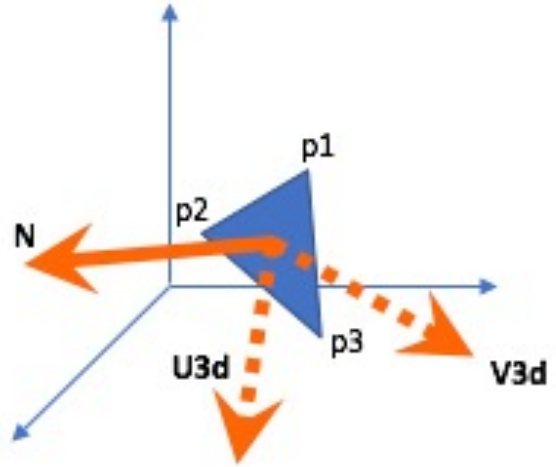
For Triangles



A triangle's *texture coordinates*.

where do Pu x Pv come from?

want a basis vectors that align with u & v in the texture



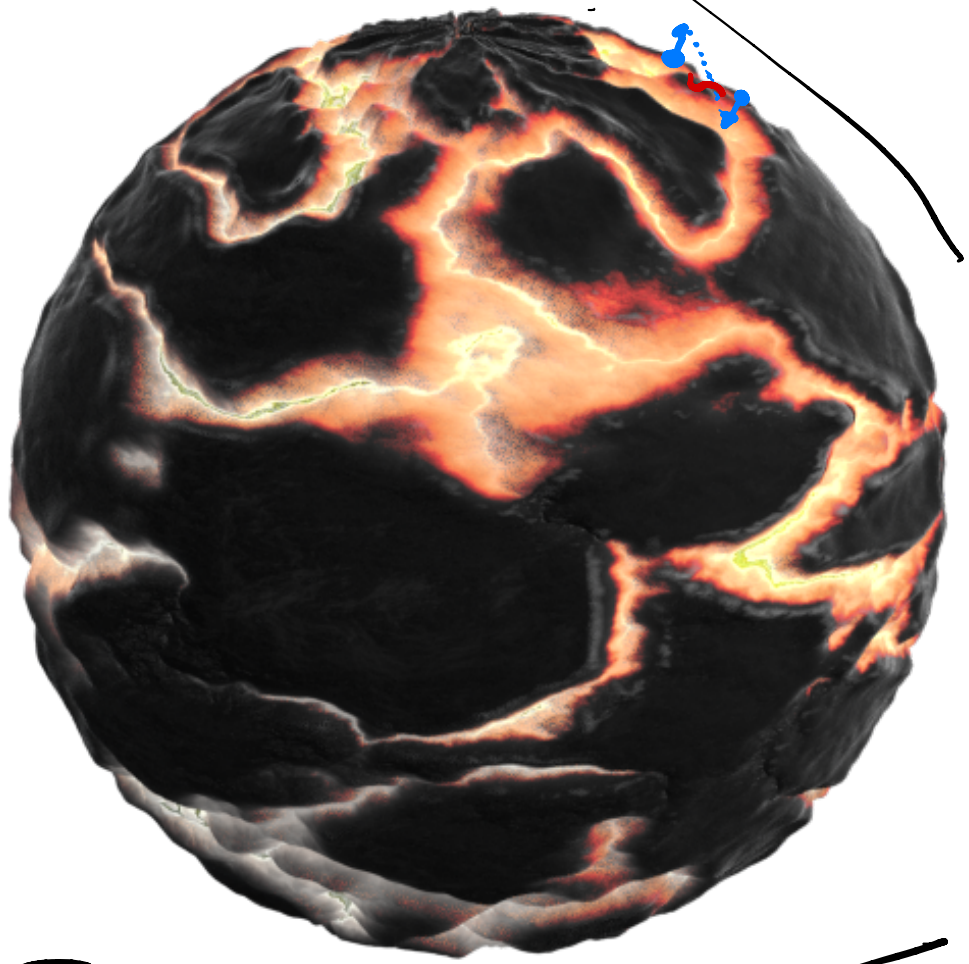
3D object space.

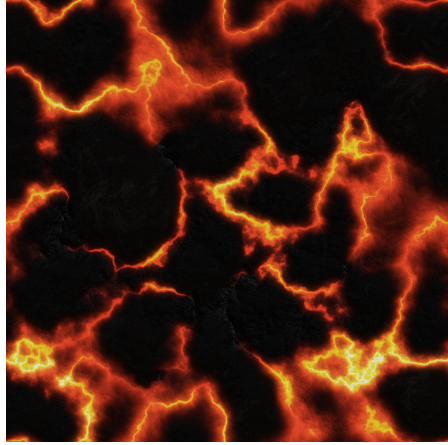
https://csawesome.runestone.academy/runestone/books/published/learnwebgl2/11_surface_properties/10_bump_maps.html

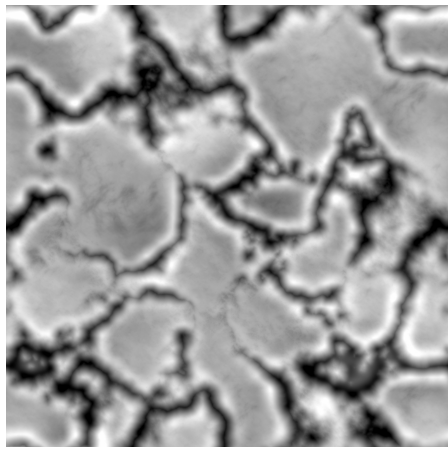
also use Norm Map in Vertex shader

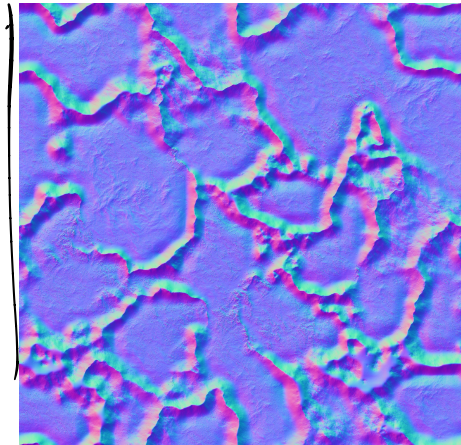
bump map

Color









0 ... 255

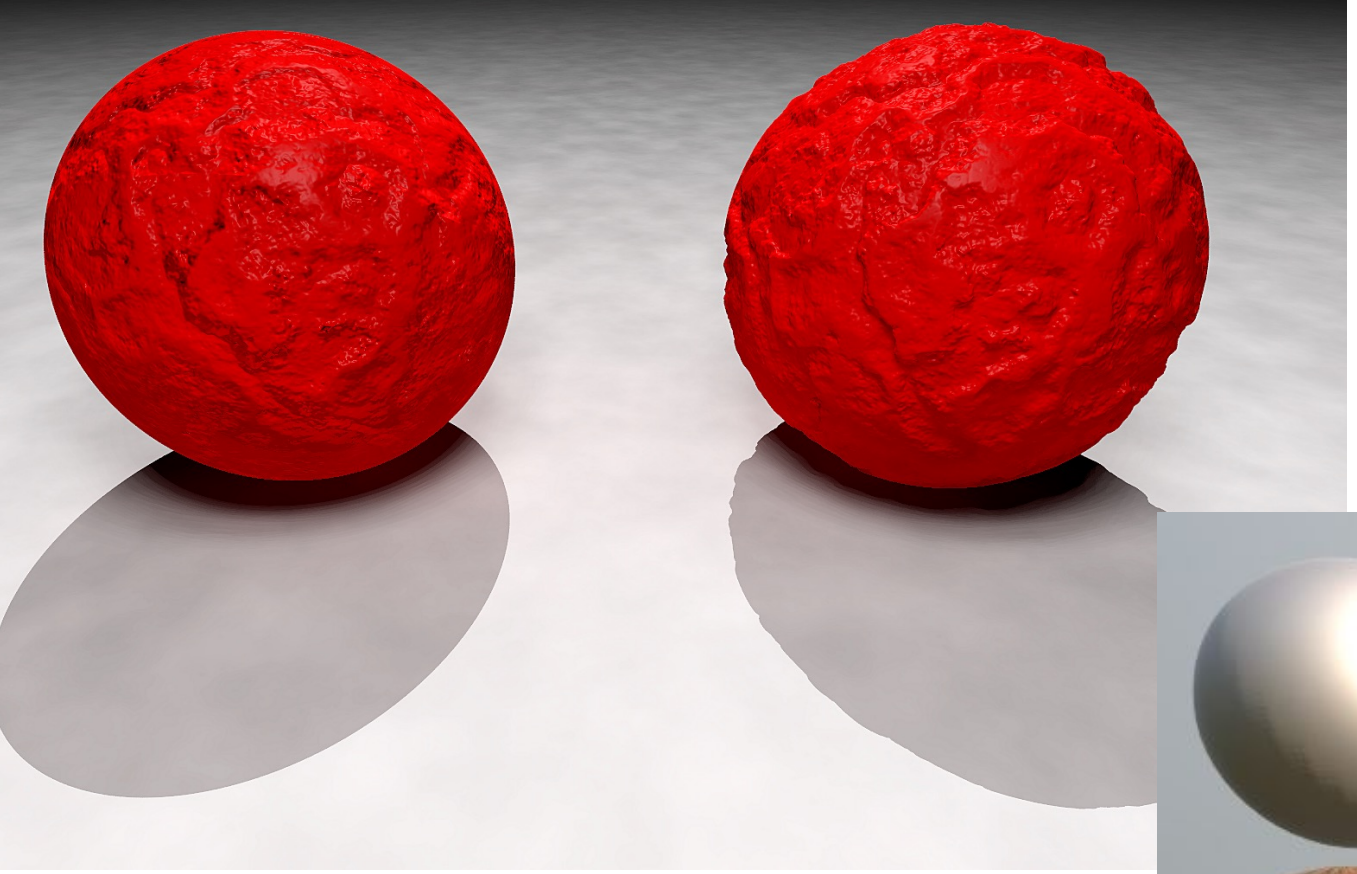
↓

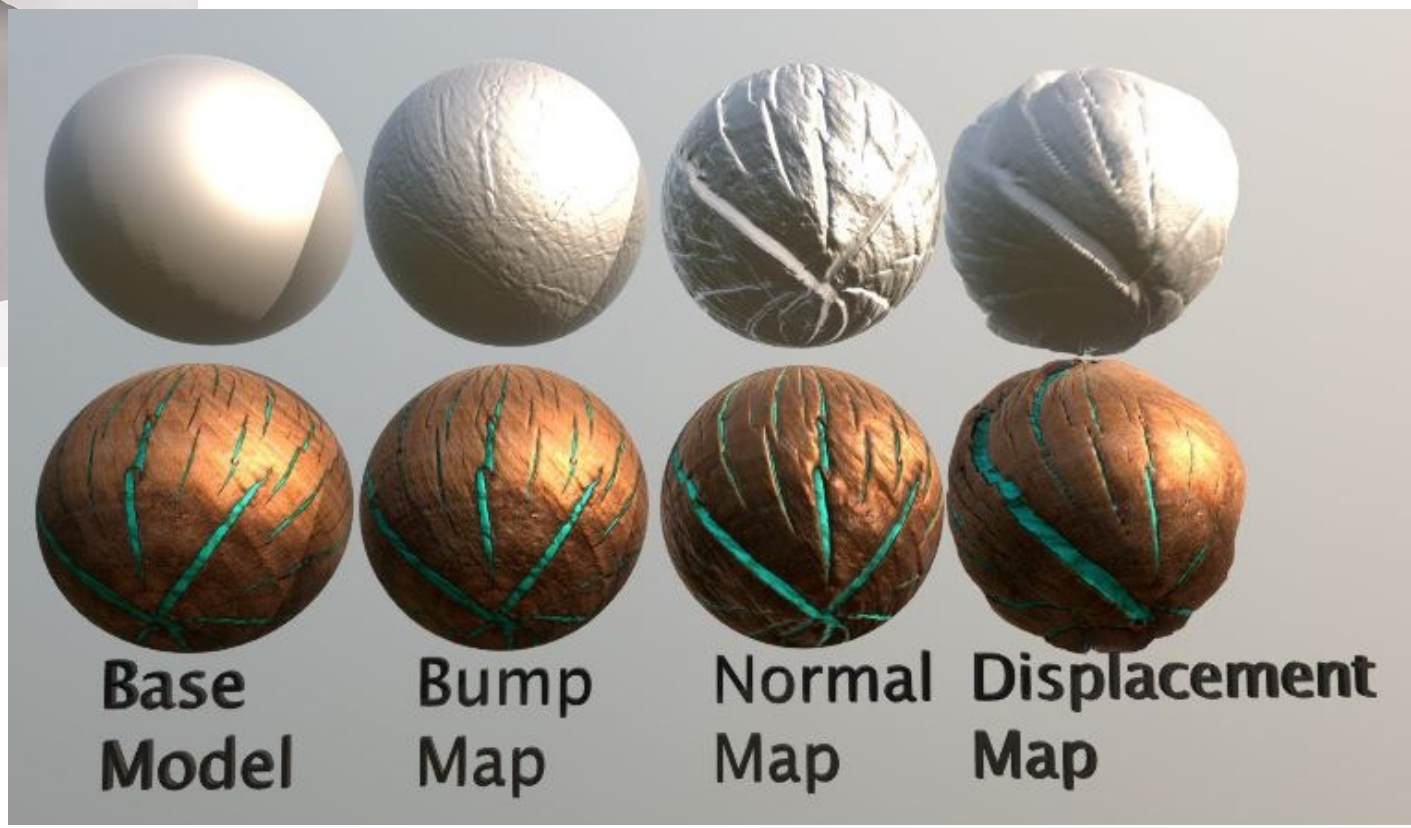
-1 ... 1 for x, y

RGB of the normal is the normal

"(x, y, 1)" → (x, y, z)

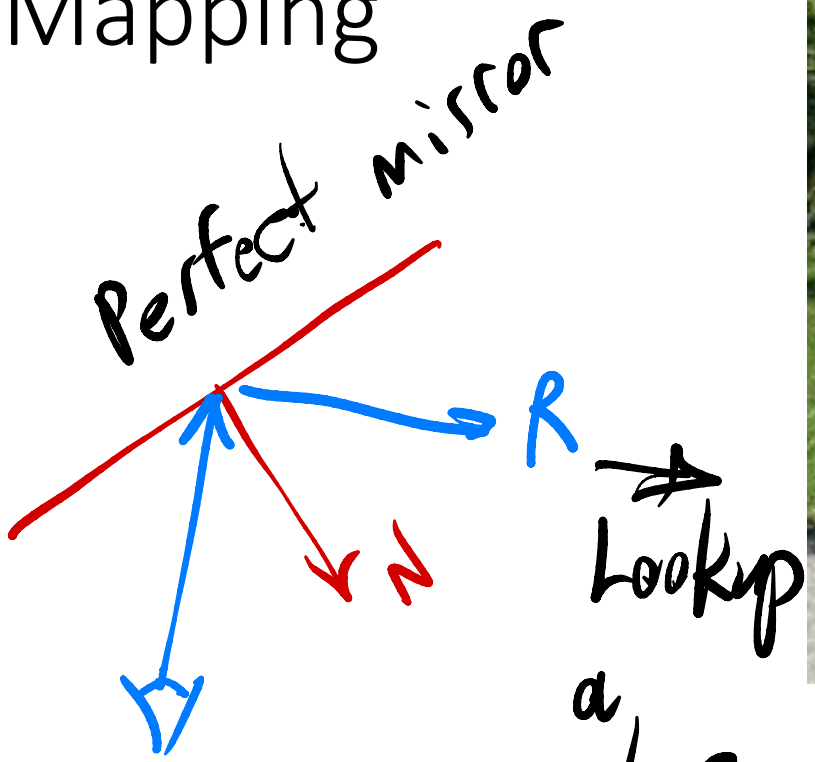
normal Map





https://spiderlili.com/2020/03/01/tech-art-tips-tricks-all-about-displacement-normal-bump-maps/

Environment Mapping



Lookup a color in

https://www.youtube.com/watch?app=desktop&v=xLPRHNiXE6w

image! of background

No self reflection!

Algorithm

NOT a Ray

1) calculate R (vector) from N & V

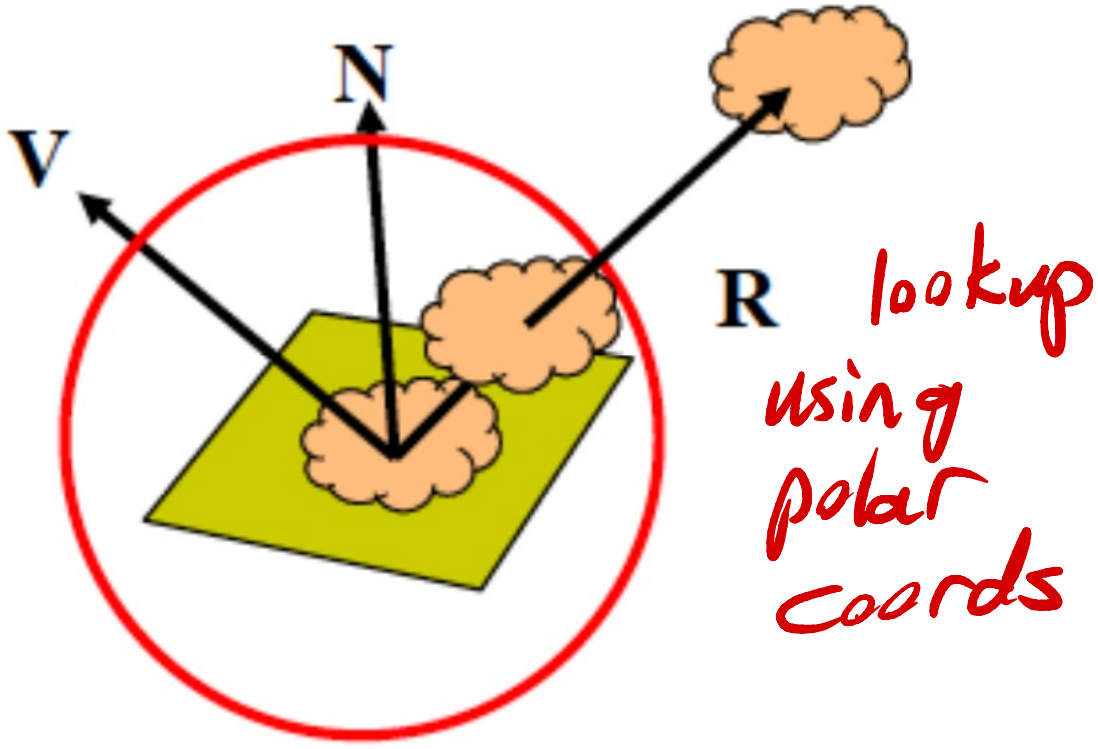
2) use R to look up color in env map.

Texture lookup

what are (u, v) ← texture coordinates

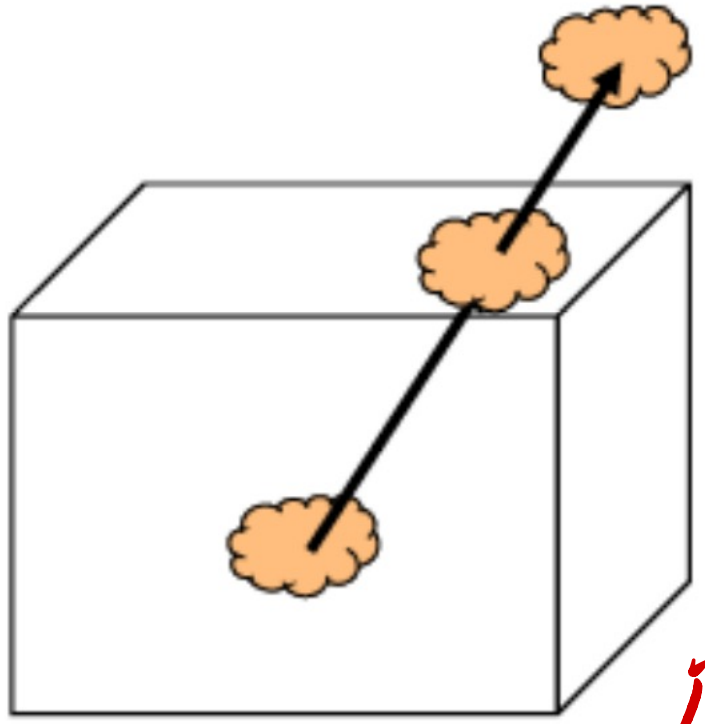
Two kinds of environment maps

a) Sphere around object (sphere map)



lookup using polar coords

b) Cube around object (cube map)



pick a side based on largest value of x, y, z in R

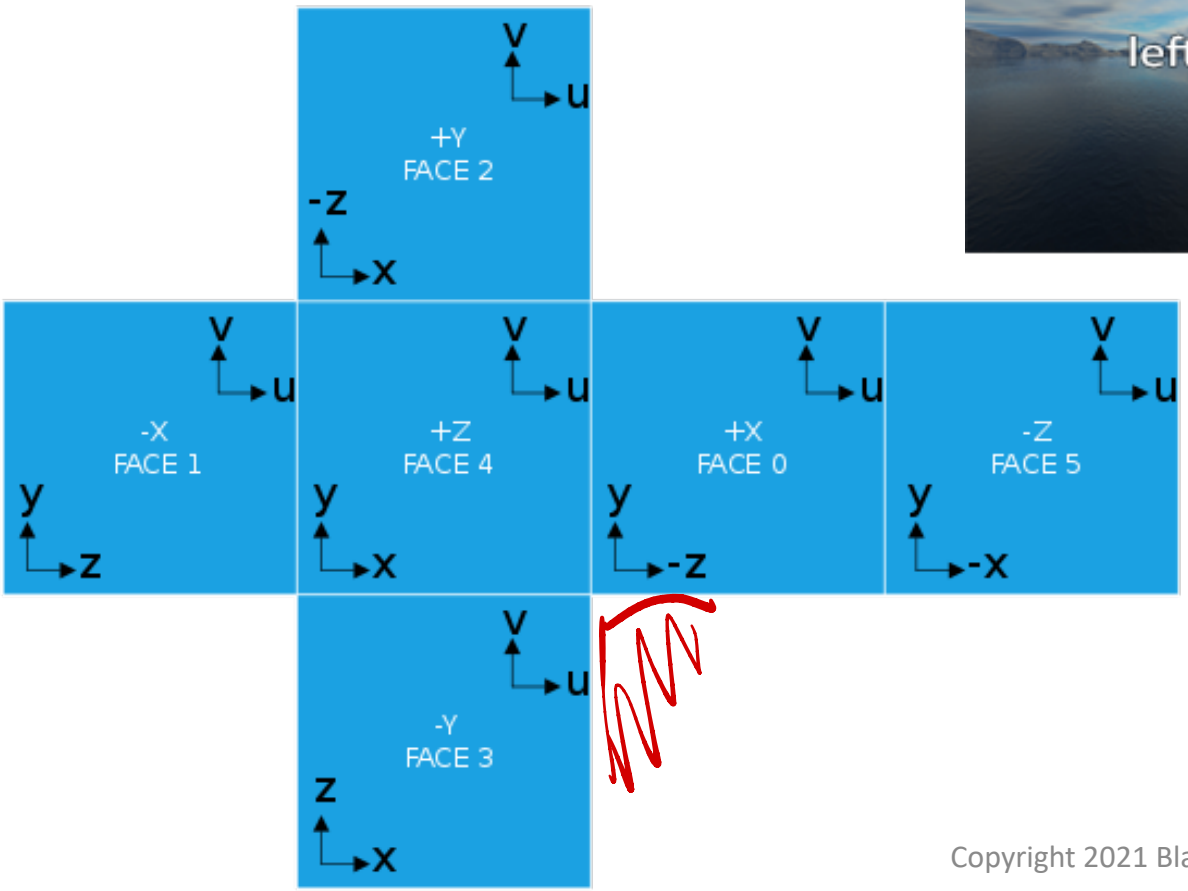
https://courses.engr.illinois.edu/cs418/fa2017/418-Lecture%2027%20-%20Environment%20Mapping.pdf

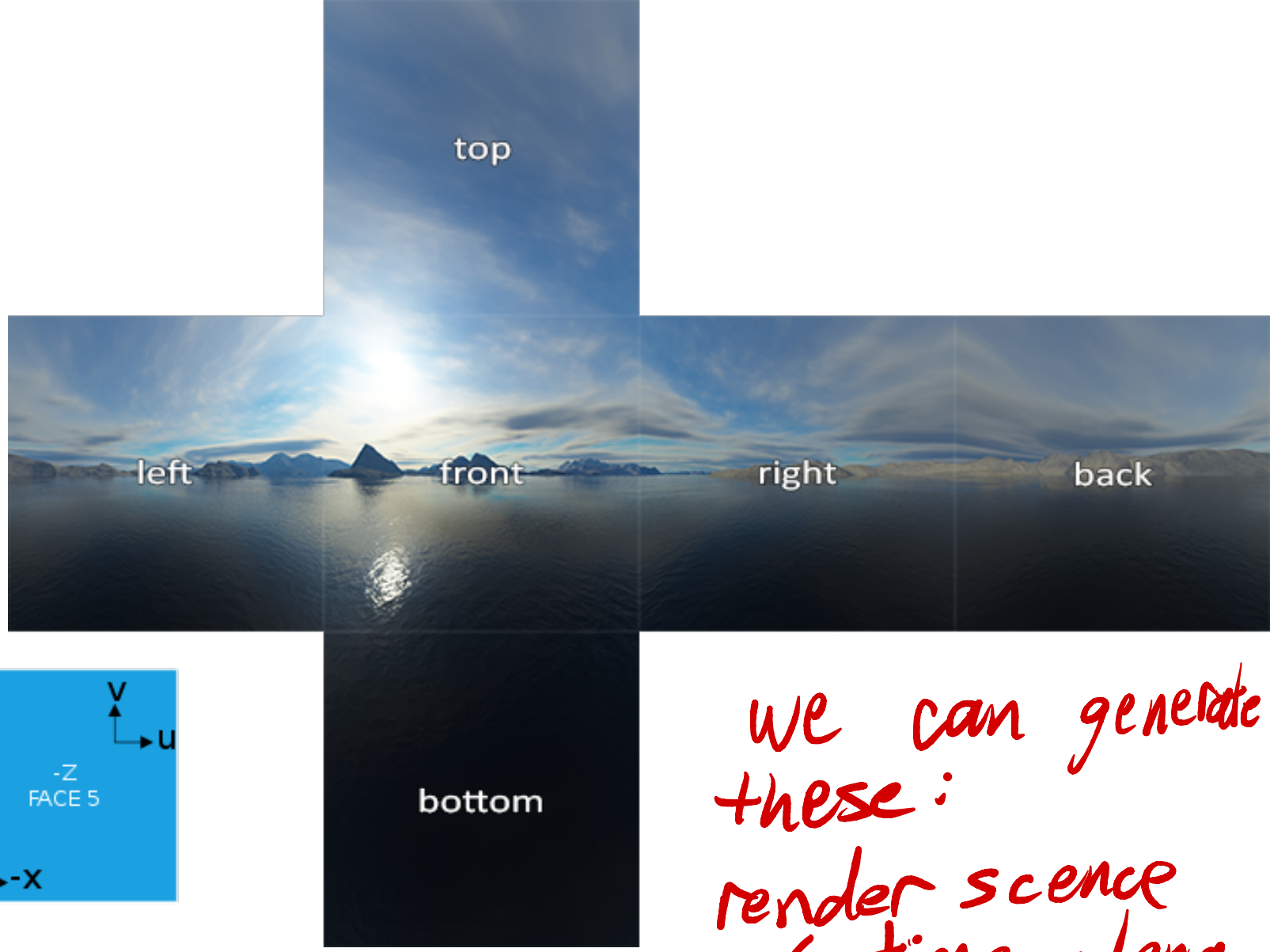
top & bottom are a single point

Sphere maps

https://aerotwist.com/tutorials/create-your-own-environment-maps/

Cube Map





we can generate these:
render scence 6 time along each ± axis