16 – more texture

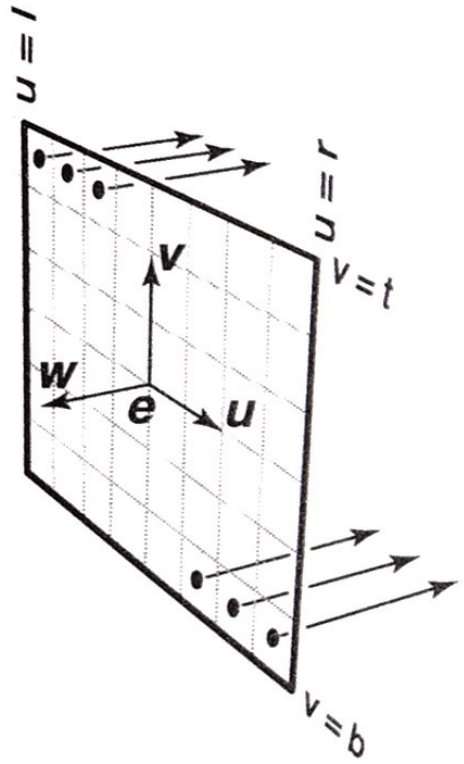
Bump Mapping and Environment Mapping

Basic Algorithm

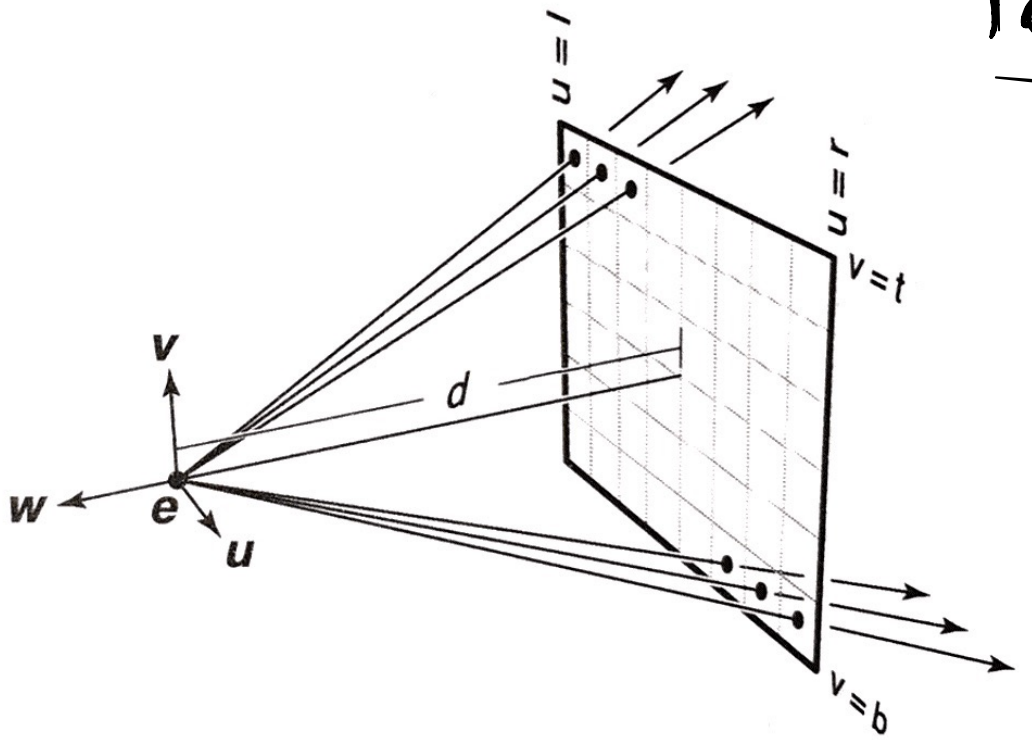
for each pixel (x_s, y_s)
    create a Ray R from eye through (x_s, y_s)
    for each object O_i in scene
        if R intersects O_j & it's the closest so far
            record this intersection
    shade pixel based on nearest intersection
        (recursively for ref & transmition)

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

fov



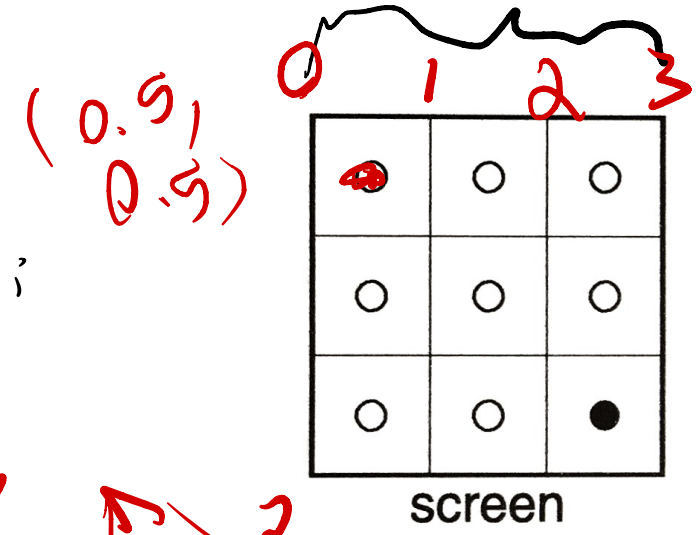
Parallel projection
same direction, different origins

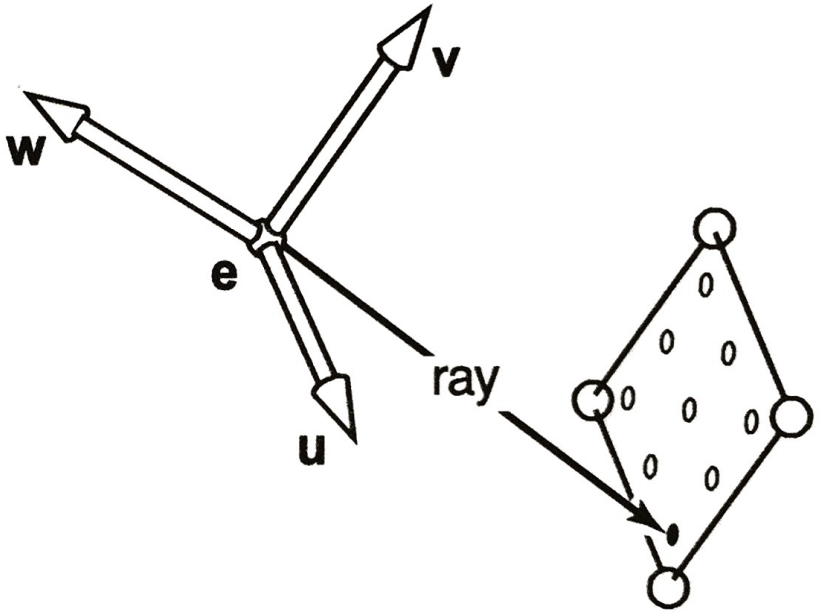


Perspective projection
same origin, different directions

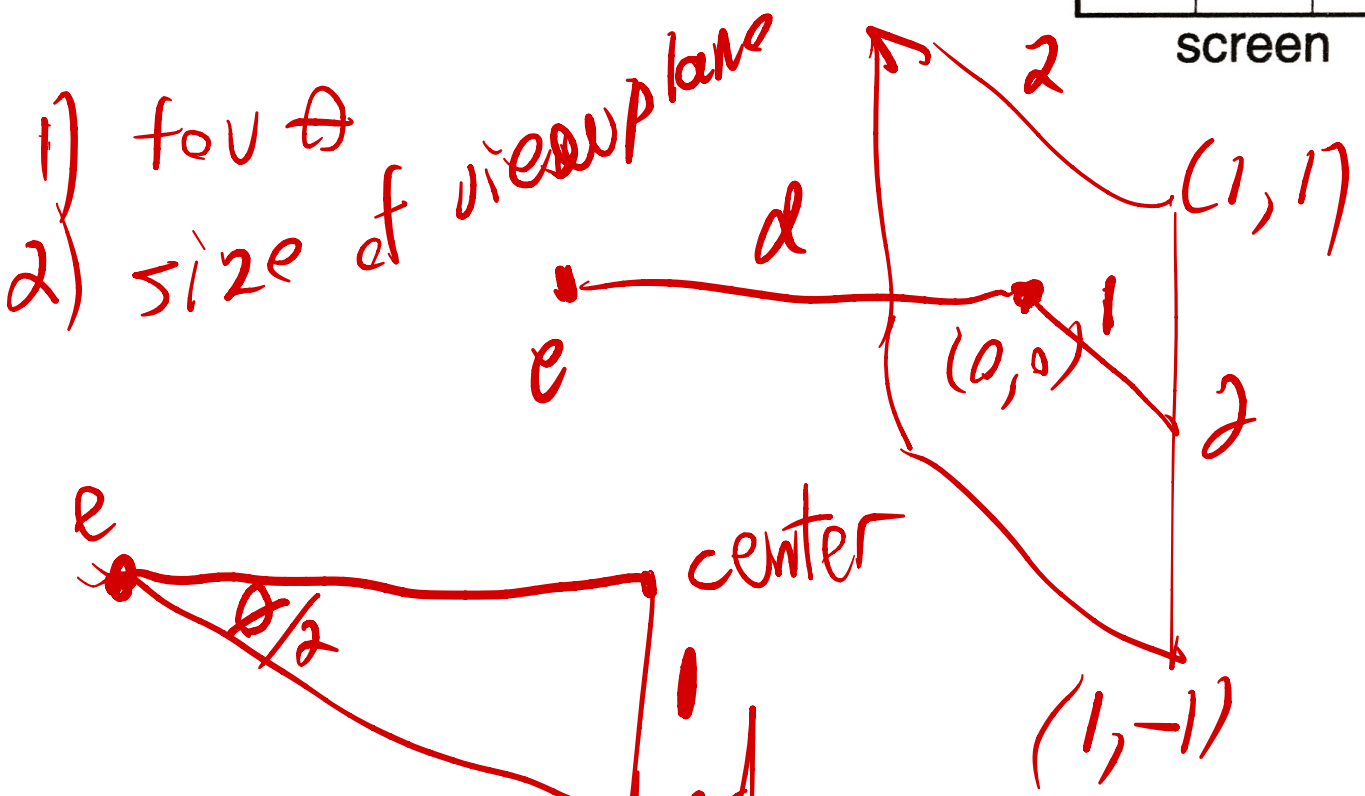
Parametric eq'n:

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

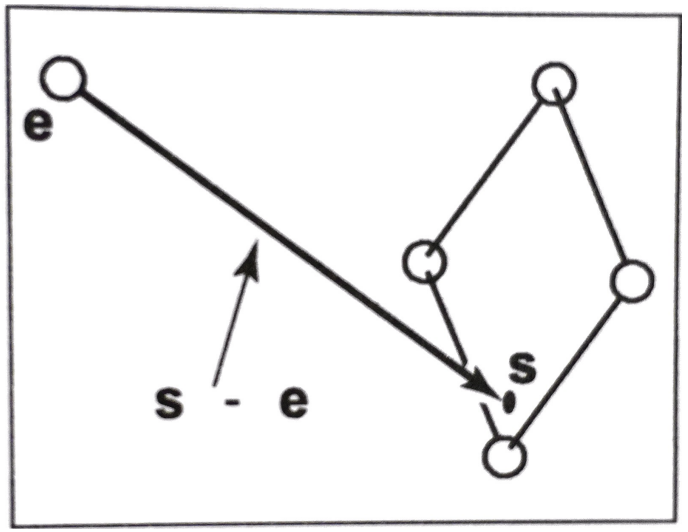




1) fov θ
2) size of viewplane



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



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

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

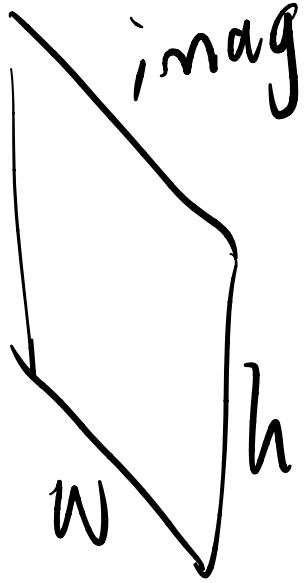
u & v

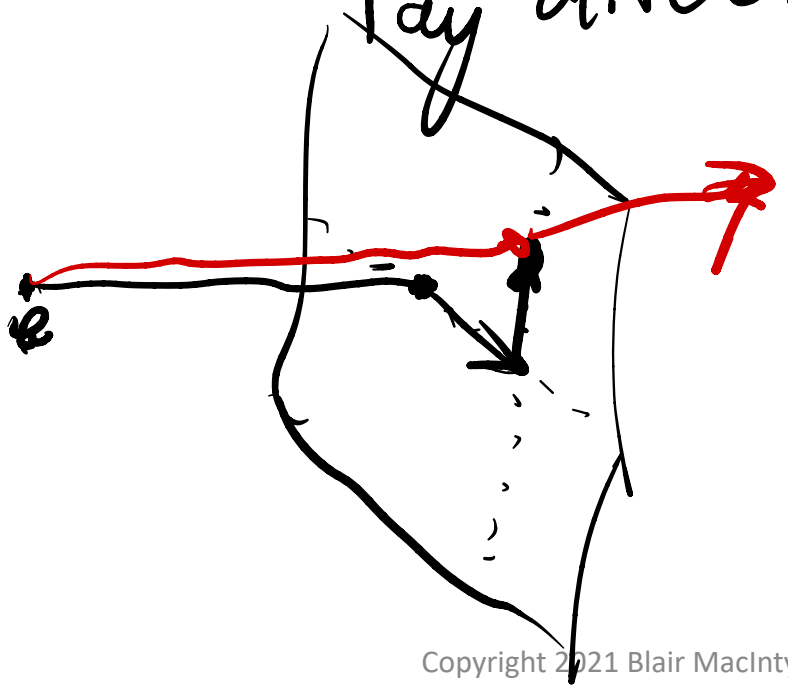
$-1 \le i$ range

u

v

w





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

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

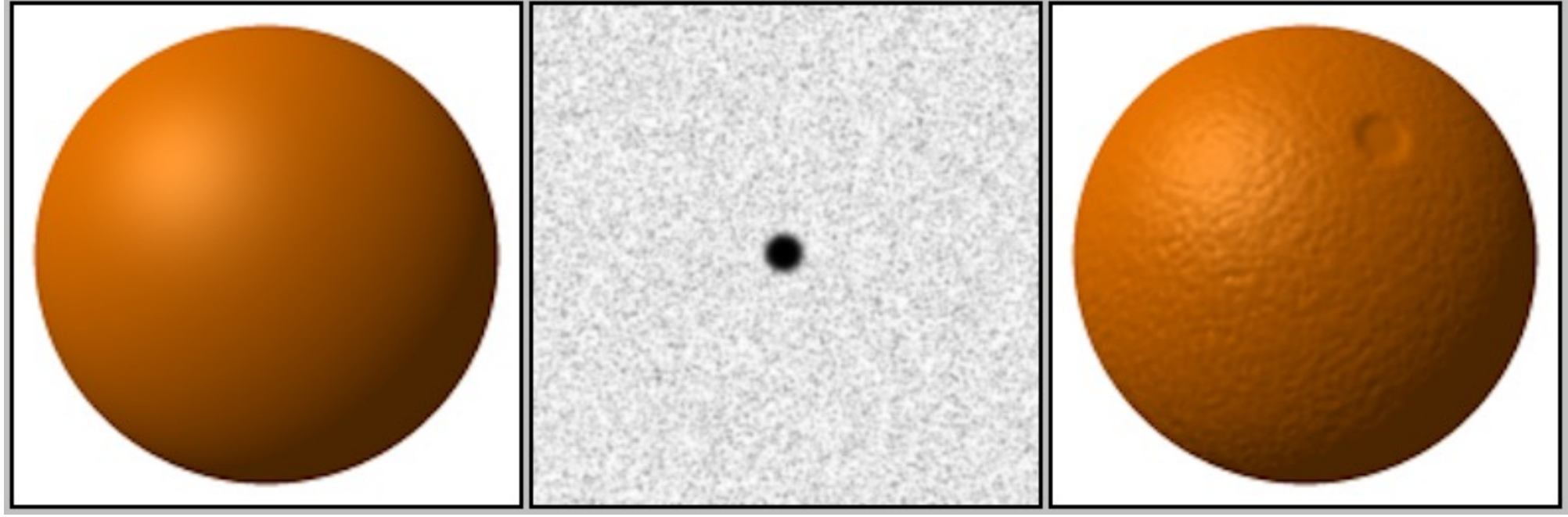
for

origin = e

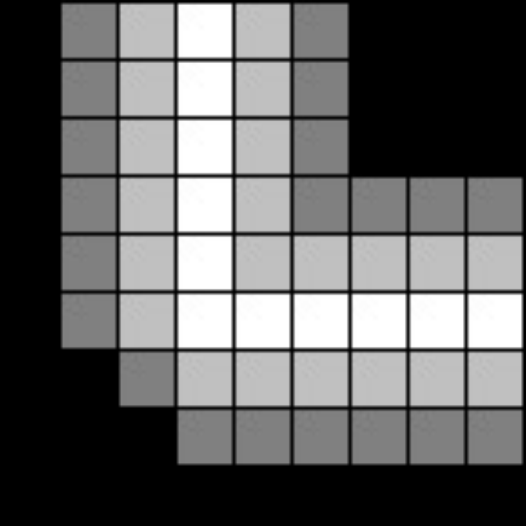
Bump Mapping

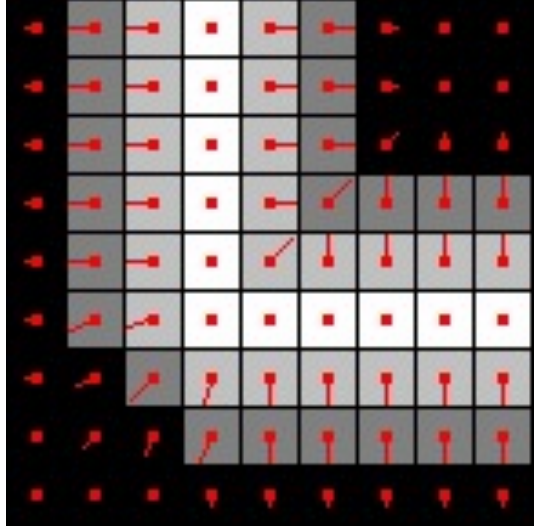
From this

to this

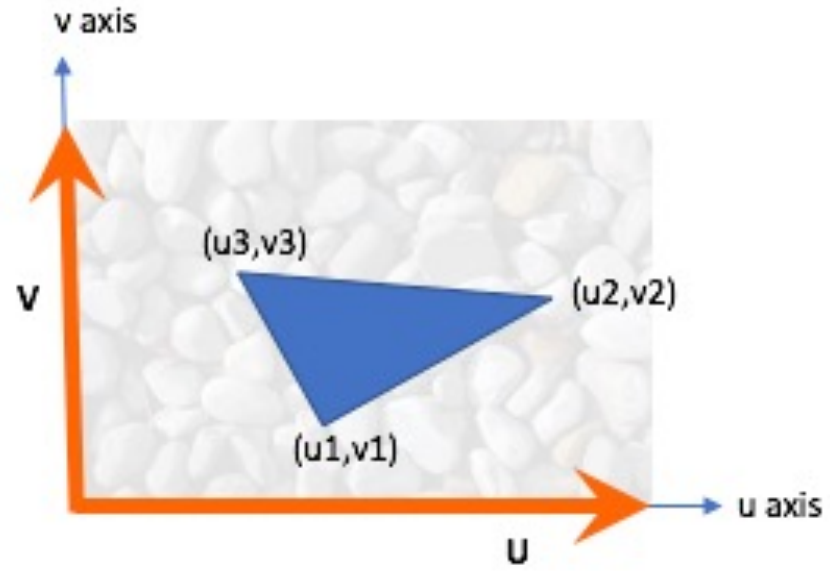


High level algorithm

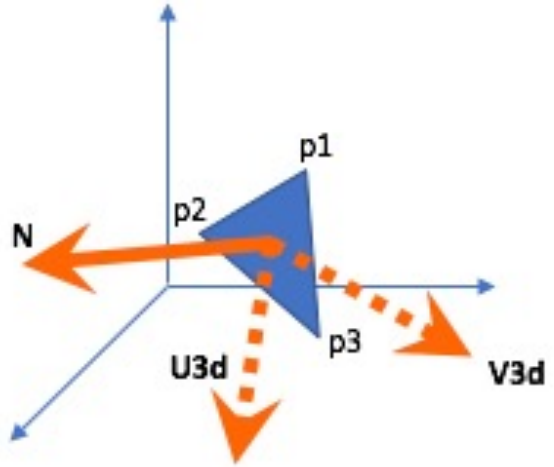




x_gradient = pixel(x-1, y) - pixel(x+1, y)
y_gradient = pixel(x, y-1) - pixel(x, y+1)

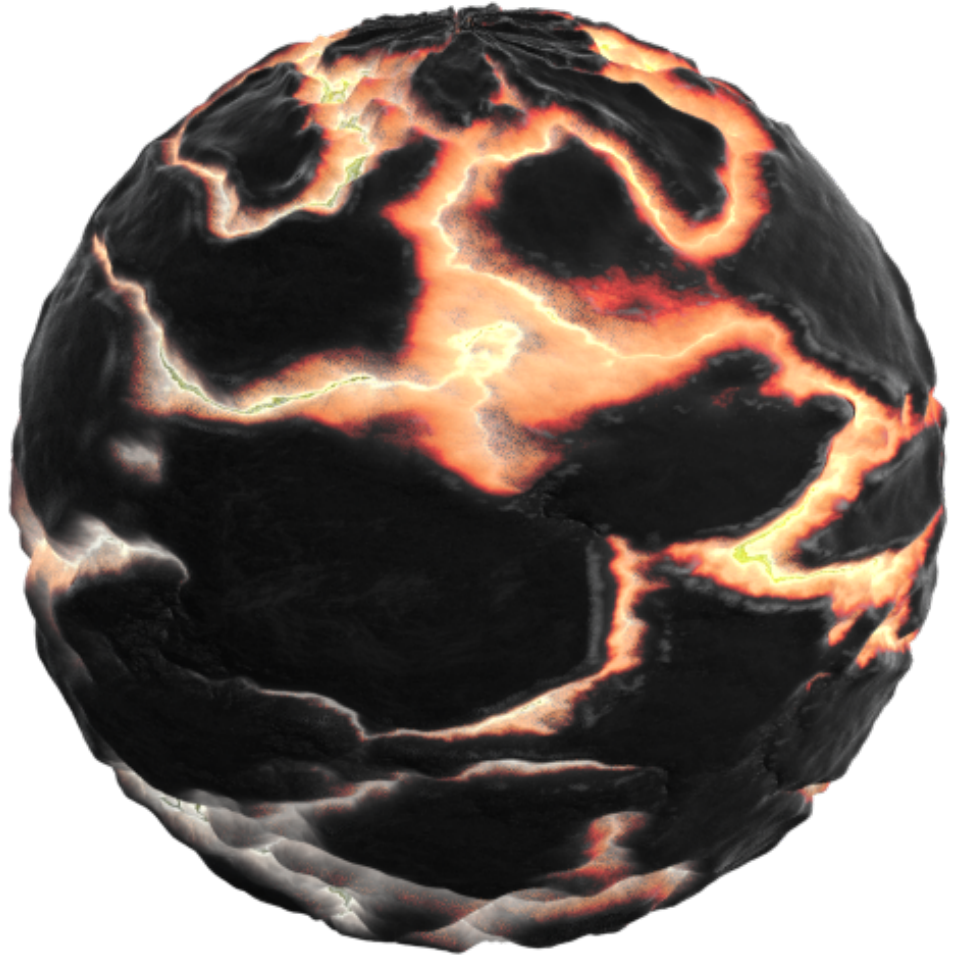


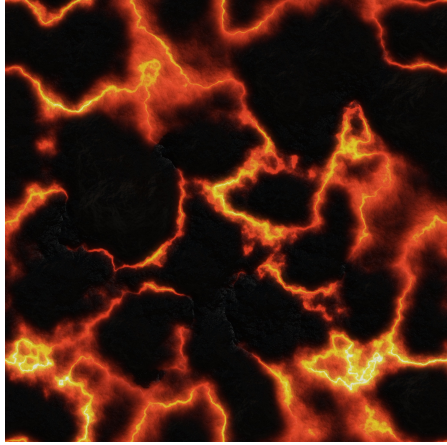
A triangle's *texture coordinates*.

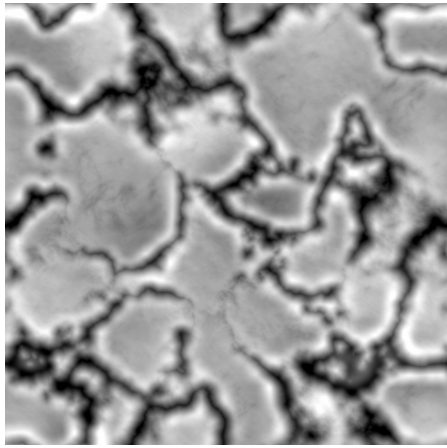


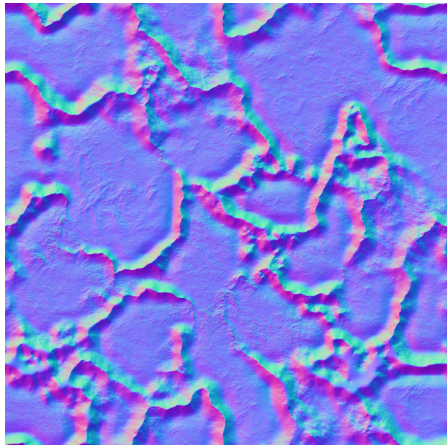
3D object space.

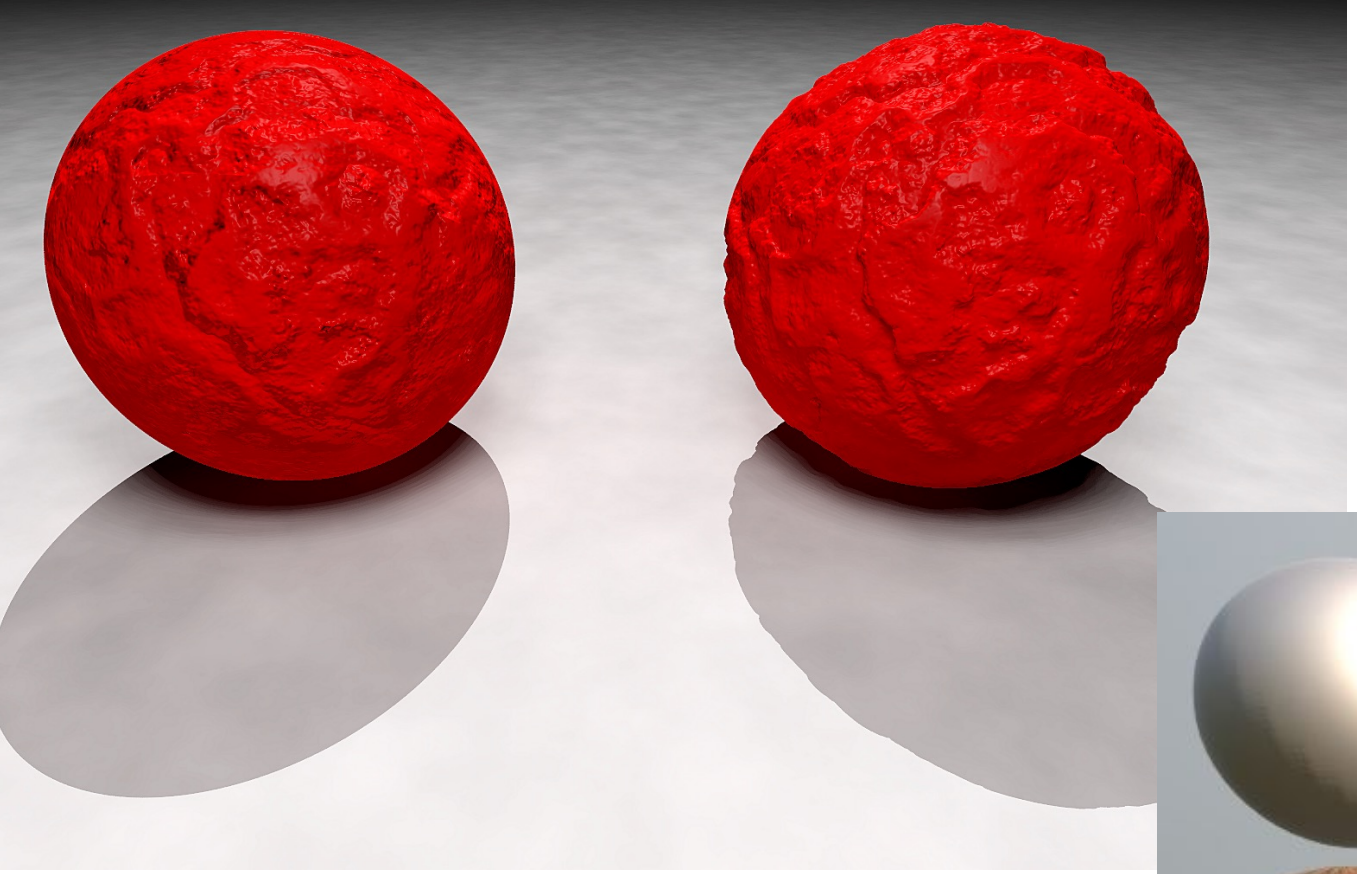
https://csawesome.runestone.academy/runestone/books/published/learnwebgl2/11_surface_properties/10_bump_maps.html

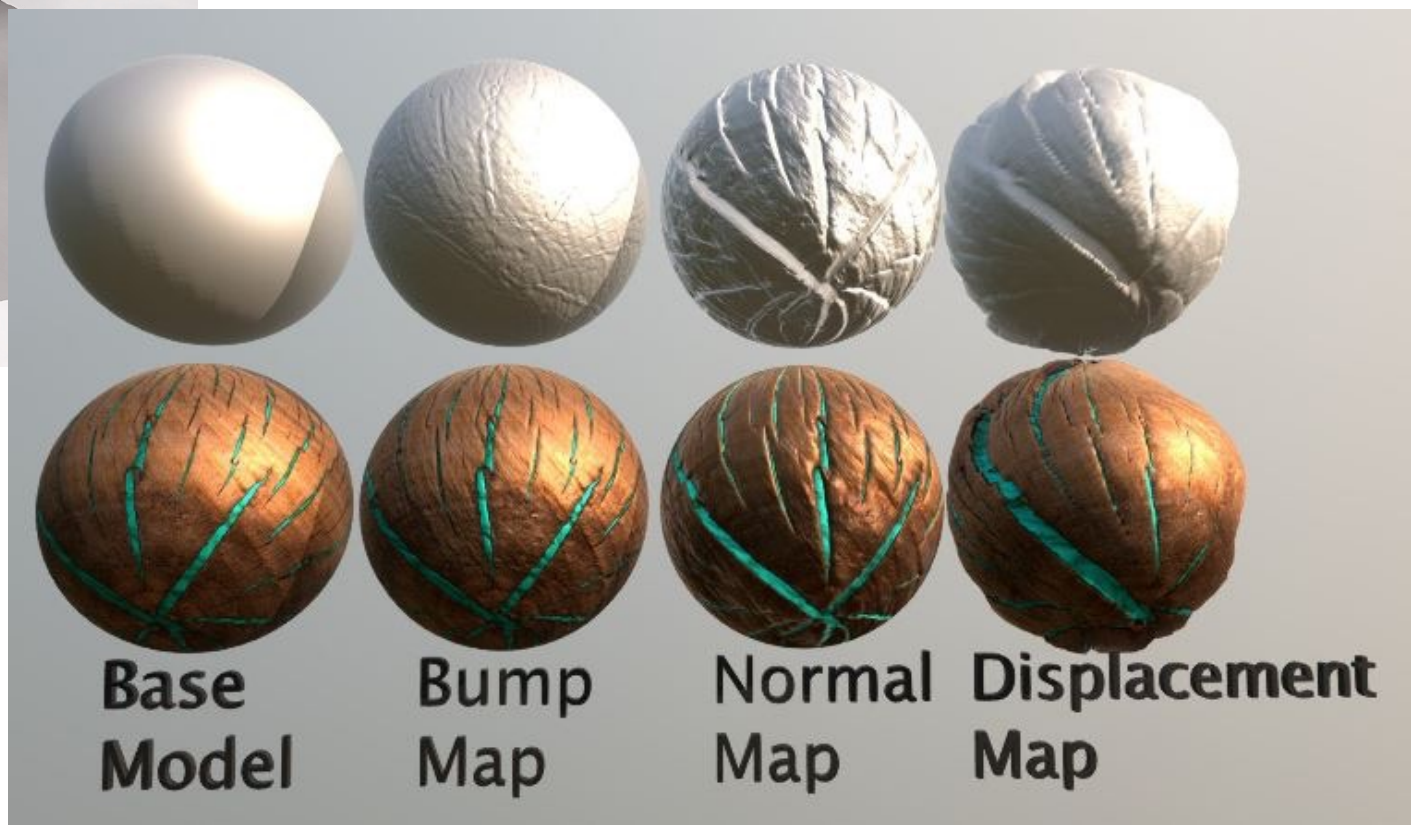












https://spiderlili.com/2020/03/01/tech-art-tips-tricks-all-about-displacement-normal-bump-maps/

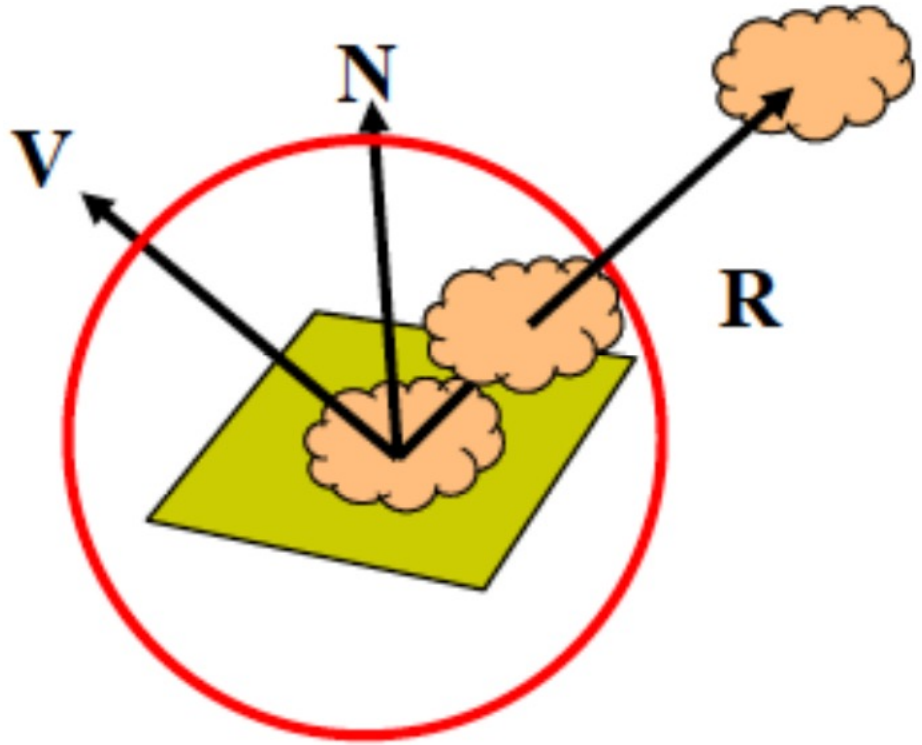
Environment Mapping

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

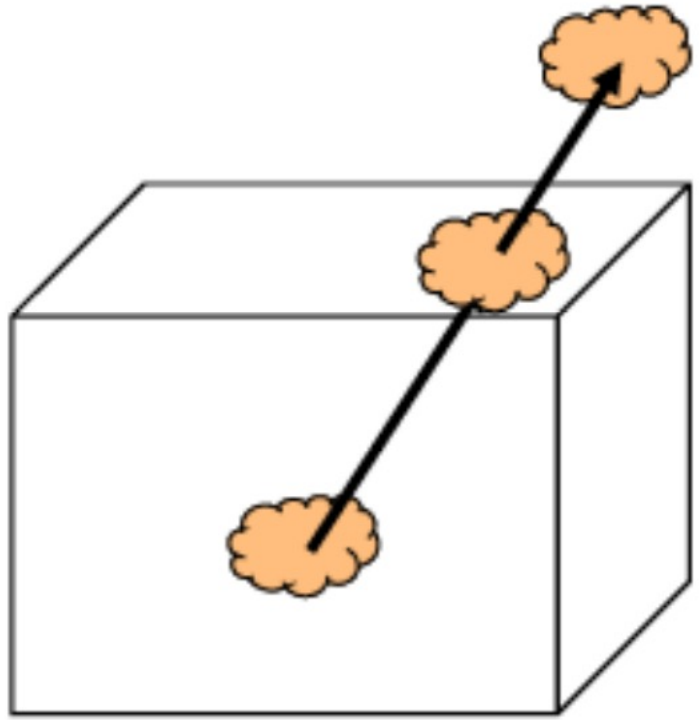
Algorithm

Two kinds of environment maps

a) Sphere around object (sphere map)



b) Cube around object (cube map)



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

Sphere maps

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

Cube Map