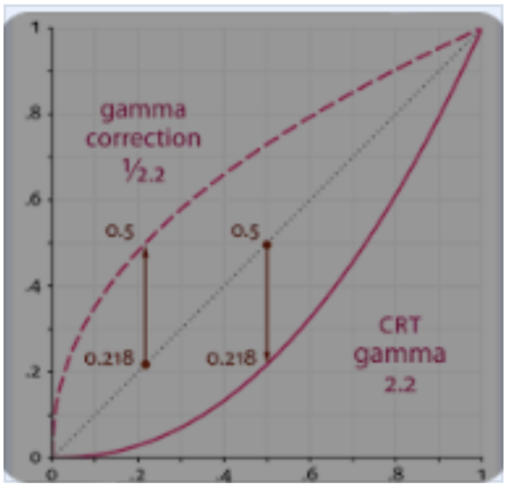
final

gamma correction:

- 1/gamma
- if image is stored (aşağıda kalan curve gibi depolanıyor) in a device and will be displayed input^(1/2.2) (üstte kalan curve ile linear hale getiriliyor) , high gamma means darker, but overall it makes the image brighter(grafikteki yüksek değerleri açık renklerinin value'larını artırıyor)
- the light/luminance that is perceived by eye is calculated input^(2.2), grafiğin üstteki curve'ü gibi, koyu renklerdeki değişimi daha fazla görürüz.



x(t) = 2 + 7t
y(t) = 1 + 2t
z(t) = 3 – 5t

- ***r**(t)* = ***o*** + *t**d*** where ***o*** = *(2, 1, 3)* and ***d*** = *(7, 2, -5)*

to a point inside a triangle,find its barycentric coordinates, if all are positive,it is inside

- Vectors can be rotated and scaled but translating a vector does not change it (vectors last homogeneous value is 0)

the cohen–sutherland algorithm

- iteration follows a fixed order(left, right, bottom, top)
- the out code will have 4 bits for two-dimensional clipping, or 6 bits in the three-dimensional case.
- the cohen–sutherland algorithm can be only be used on a rectangular clip window

liang-barsky algorithm

1. t_ent=0, t_leav=1
2. check if potentially entering of leaving from only the difference of the points of lines (do not check the visible region)

3. compute t
4. if entering check t<t_leav, if leaving check t>t_ent (not visible if not true)
5. update t_ent if new t is entering and greater, t_leav if new t is leaving and smaller

- more efficient than cohen–sutherland algorithm (when dimensions increases much more efficient)
- pk = the difference in line points qk= the difference between edge and point

sutherland hodgeman algorithm

- initially= all vertices
- both outside -> add nothing
- both inside -> add v2
- inside to outside -> add v2'
- outside to inside -> add v1' and v2

rasterization - mid point line algorithm

1. find the implicit line equation (with positive m, y positive x negative coefficient)
2. find the coordinates of mid point of the next pixel (x+1, y+0.5)
3. if midpoint is below the line, the result is negative, choose upper pixel
4. the upper pixel's midpoint is selected

- if(m>1): swap roles, x positive coefficient in implicit eq

interpolating attributes

yakın olan noktaya ters orantı yapacak şekilde topla

triangle rasterization

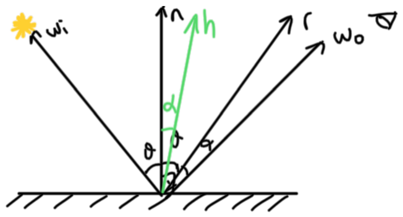
1. plot each edge using mid point algorithm
2. check the points inside the bounding box using implicit line equation
3. edges are not in order to counter clock wise, reverse order (in clockwise)
4. if point is negative in all three equation, it is inside the triangle

interploting triangle attributes

1. calculate implicit edge equations counter wise
2. get the ratio of f(point)/f(karşıdaki vertex) -> they become alpha beta gamma (must bu positive)
3. c= f(point)/f(karşıdaki vertex) * c_karşıdaki + .. (sum them up)

shadow map technique more suitable for spot lights. The *Stencil Shadow Volume* is an interesting technique that provides a straightforward solution to the problem of point lights.

The key idea is that **when an object is inside the volume (and therefore in shadow) the front polygons of the volume win the depth test against the polygons of the object and the back polygons of the volume fail the same test**.



$$\text{radiance}(L)\begin{cases} L_a = \text{ambient shading} = I\,k_a \\ L_d = \text{diffuse shading} = I \cdot k_d \cdot \max(0, w_i \cdot n) \\ L_s = \text{specular shading} = I \cdot k_s \cdot \max(0, n \cdot h) \end{cases}$$

R = reflectance $[0,1]$

shadow rays = from intersection point to every light source