Shadows, etc.

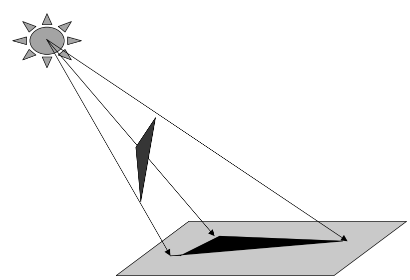
cheesy shadows

picture & demo

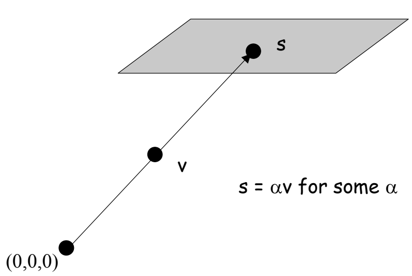
cheesy shadows

```
for each light {
  for each polygon in the scene {
     for each vertex of the polygon {
          project the vertex onto the floor
               with respect to the light
          }
     draw the projected polygon
  }
}
```

projection

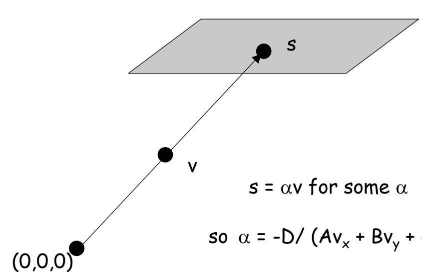


suppose we have a point light at the origin



$s = \alpha v$ for some α

assume the equation of the floor plane is $Ax+By+Cz+D=0$



$s = \alpha v$ for some α

so $\alpha = -D / (Av_x + Bv_y + Cv_z)$

using the pipeline

$s = -Dv/ (Av_x + Bv_y + Cv_z)$

$$\underbrace{\begin{pmatrix} D & 0 & 0 & 0 \\ 0 & D & 0 & 0 \\ 0 & 0 & D & 0 \\ -A & -B & -C & 0 \end{pmatrix}}_{M_{shadow}} \begin{pmatrix} v_x \\ v_y \\ v_z \\ 1 \end{pmatrix} \xrightarrow[\text{division}]{\text{perpective}} s$$

for point light at point p

$M(v-p) \rightarrow$ perspective division $\rightarrow$ add p

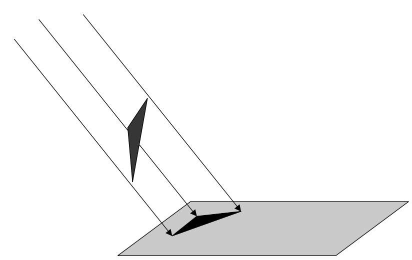
final shadow matrix for point light

$p \bullet q\, I - p^T q$

where $p=\langle p_x, p_y, p_z, 1\rangle$, $q=\langle A, B, C, D\rangle$

Exercise:
Use this equation to compute the shadow matrix when the light is at 0.
How does your answer compare with our derivation for this special case?

directional light



shadow matrix

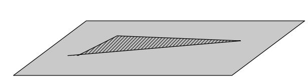
for a light with direction $\langle p_x, p_y, p_z\rangle$

$p \bullet q\, I - p^T q$

where $p=\langle p_x, p_y, p_z, 0\rangle$, $q=\langle A, B, C, D\rangle$

practical problems

drawing two polygon in same place



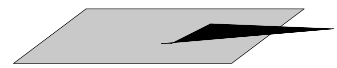
polygon offset

when enabled, a small offset is added to the depth values during scan conversion

see Woo p. 250

practical problems

drawing beyond the floor



stencil buffer

RGB	float	n bits
frame buffer	z-buffer (depth buffer)	stencil buffer

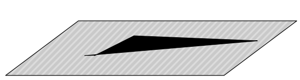
stencil buffer

initialize stencil buffer to all 0's
draw floor:
- whenever pixel is drawn, set corresponding position in stencil buffer to 1

draw shadow:
- draw shadow pixels provided the corresponding stencil values is 1

practical problems

what about ambient light?



alpha blending