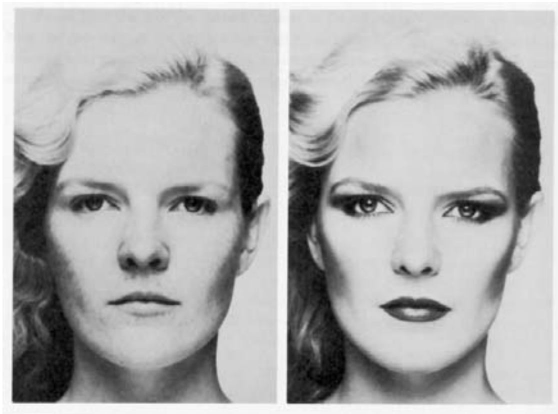
Photometric Stereo

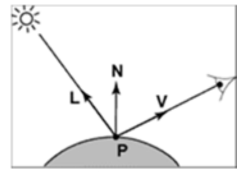


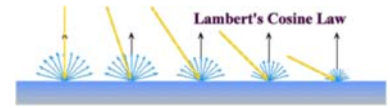
Merle Norman Cosmetics, Los Angeles

Readings

- Optional: Woodham's original photometric stereo paper
  - http://www.cs.ubc.ca/~woodham/papers/Woodham80c.pdf

Diffuse reflection





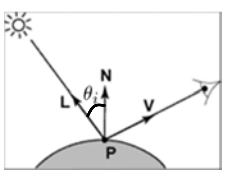
$$R_e = k_d \mathbf{N} \cdot \mathbf{L} R_i$$

image intensity of **P** ⟶ $I = k_d \mathbf{N} \cdot \mathbf{L}$

Simplifying assumptions

- $I = R_e$: camera response function f is the identity function:
  – can always achieve this in practice by solving for f and applying f^{-1} to each pixel in the image
- $R_i = 1$: light source intensity is 1
  – can achieve this by dividing each pixel in the image by R_i

Shape from shading



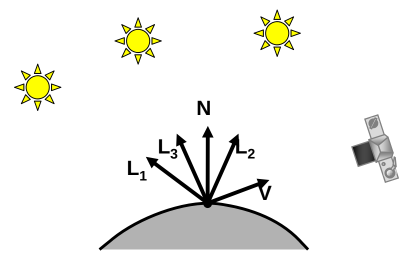
Suppose $k_d = 1$

$$\begin{aligned} I &= k_d \mathbf{N} \cdot \mathbf{L} \\ &= \mathbf{N} \cdot \mathbf{L} \\ &= cos\ \theta_i \end{aligned}$$

You can directly measure angle between normal and light source

- Not quite enough information to compute surface shape
- But can be if you add some additional info, for example
  - assume a few of the normals are known (e.g., along silhouette)
  - constraints on neighboring normals—"integrability"
  - smoothness
- Hard to get it to work well in practice
  - plus, how many real objects have constant albedo?

Photometric stereo



$$\begin{aligned} I_1 &= k_d \mathbf{N} \cdot \mathbf{L_1} \\ I_2 &= k_d \mathbf{N} \cdot \mathbf{L_2} \\ I_3 &= k_d \mathbf{N} \cdot \mathbf{L_3} \end{aligned}$$

Can write this as a matrix equation:

$$\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} = k_d \mathbf{N}^T \begin{bmatrix} \mathbf{L_1} & \mathbf{L_2} & \mathbf{L_3} \end{bmatrix}$$

Solving the equations

$$\underbrace{\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix}}_{\underset{1\times 3}{\mathbf{I}}} = \underbrace{k_d \mathbf{N}^T}_{\underset{1\times 3}{\mathbf{G}}} \underbrace{\begin{bmatrix} \mathbf{L_1} & \mathbf{L_2} & \mathbf{L_3} \end{bmatrix}}_{\underset{3\times 3}{\mathcal{L}}}$$

$$\mathbf{G} = \mathbf{I}\mathbf{L}^{-1}$$

$$k_d = \|\mathbf{G}\|$$
$$\mathbf{N} = \frac{1}{k_d}\mathbf{G}$$

More than three lights

Get better results by using more lights

$$\begin{bmatrix} I_1 & \dots & I_n \end{bmatrix} = k_d \mathbf{N}^T \begin{bmatrix} \mathbf{L_1} & \dots & \mathbf{L_n} \end{bmatrix}$$

Least squares solution:

$$\begin{aligned} \mathbf{I} &= \mathbf{G}\mathbf{L} \\ \mathbf{I}\mathbf{L}^{\mathbf{T}} &= \mathbf{G}\mathbf{L}\mathbf{L}^{\mathbf{T}} \\ \mathbf{G} &= (\mathbf{I}\mathbf{L}^{\mathbf{T}})(\mathbf{L}\mathbf{L}^{\mathbf{T}})^{-1} \end{aligned}$$

Solve for N, k_d as before

What's the size of $\mathbf{LL^T}$?

Color images

The case of RGB images

- get three sets of equations, one per color channel:

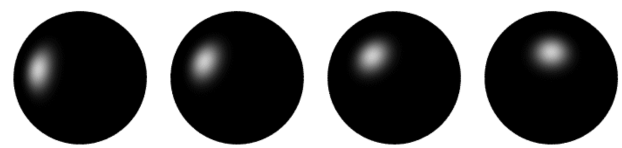
$$\begin{aligned} \mathbf{I}_R &= k_{dR}\mathbf{N}^T\mathcal{L} \quad \text{call this } \mathbf{J} \\ \mathbf{I}_G &= k_{dG}\mathbf{N}^T\mathcal{L} \\ \mathbf{I}_B &= k_{dB}\mathbf{N}^T\mathcal{L} \end{aligned}$$

- Simple solution: first solve for **N** using one channel
- Then substitute known **N** into above equations to get k_d s:

$$\begin{aligned} \mathbf{I}_R &= k_{dR}\mathbf{J} \\ \mathbf{J}\cdot\mathbf{I}_R &= k_{dR}\mathbf{J}\cdot\mathbf{J} \\ k_{dR} &= \frac{\mathbf{J}\cdot\mathbf{I}_R}{\mathbf{J}\cdot\mathbf{J}} \end{aligned}$$

Computing light source directions

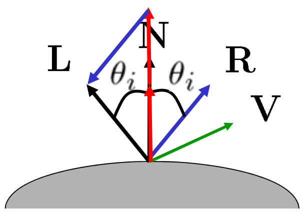
Trick: place a chrome sphere in the scene



- the location of the highlight tells you where the light source is

Recall the rule for specular reflection

For a perfect mirror, light is reflected about **N**



$$R_e = \begin{cases} R_i & \text{if } \mathbf{V} = \mathbf{R} \\ 0 & \text{otherwise} \end{cases}$$

2(R.N)N − R

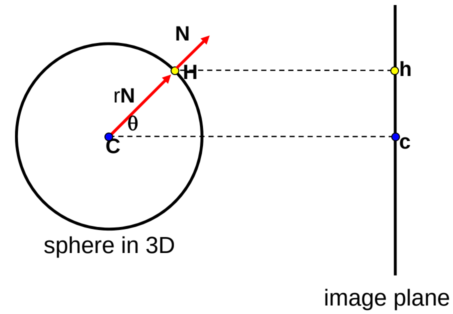
We see a highlight when **V** = **R**

- then L is given as follows:

$$\mathbf{L} = 2(\mathbf{N} \cdot \mathbf{R})\mathbf{N} - \mathbf{R}$$

Computing the light source direction

Chrome sphere that has a highlight at position **h** in the image

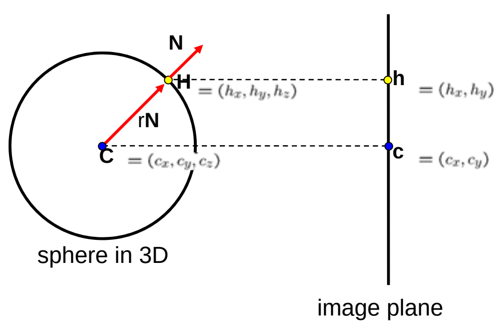


Can compute θ (and hence **N**) from this figure

Now just reflect **V** about **N** to obtain **L**

Computing the light source direction

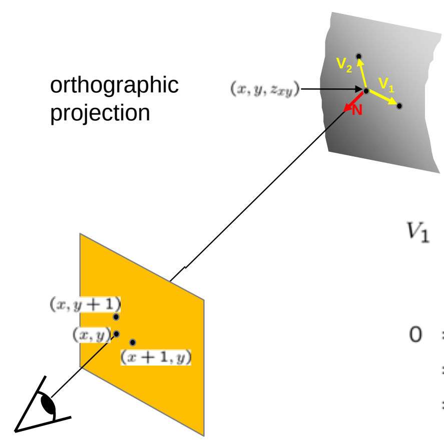
Chrome sphere that has a highlight at position **h** in the image



Can compute **N** by studying this figure

- Hints:
  - use this equation: $\|H - C\| = r$
  - can measure **c**, **h**, and r in the image
  - can choose $c_z = 0$

Depth from normals



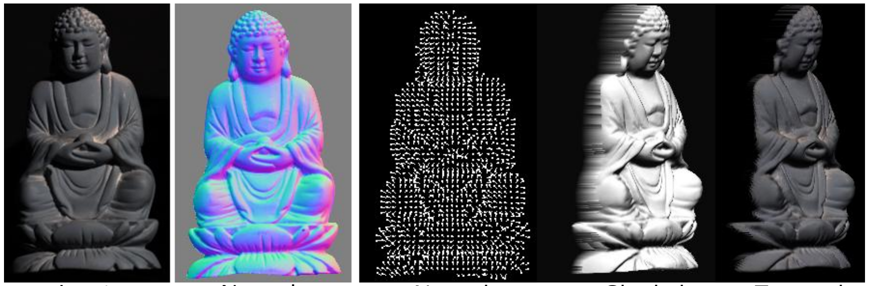
$$\begin{aligned} V_1 &= (x+1, y, z_{x+1,y}) - (x, y, z_{xy}) \\ &= (1, 0, z_{x+1,y} - z_{xy}) \end{aligned}$$

$$\begin{aligned} 0 &= N \cdot V_1 \\ &= (n_x, n_y, n_z) \cdot (1, 0, z_{x+1,y} - z_{xy}) \\ &= n_x + n_z(z_{x+1,y} - z_{xy}) \end{aligned}$$

Get a similar equation for $\mathbf{V_2}$

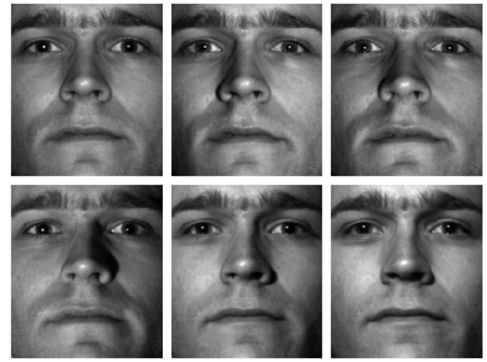
- Each normal gives us two linear constraints on z
- compute z values by solving a matrix equation

Results…



Input (1 of 12) | Normals | Normals | Shaded rendering | Textured rendering

Results…



from Athos Georghiades
http://cvc.yale.edu/people/Athos.html

Limitations

Big problems
- doesn’t work for shiny things, semi-translucent things
- shadows, inter-reflections

Smaller problems
- camera and lights have to be distant
- calibration requirements
  - measure light source directions, intensities
  - camera response function

Trick for handling shadows

Weight each equation by the pixel brightness:

$$I_i(I_i) \;=\; I_i[k_d \mathbf{N} \cdot \mathbf{L_i}]$$

Gives weighted least-squares matrix equation:

$$\left[\begin{array}{ccc} I_1^2 & \ldots & I_n^2 \end{array}\right] = k_d \mathbf{N}^T \left[\begin{array}{ccc} I_1\mathbf{L_1} & \ldots & I_n\mathbf{L_n} \end{array}\right]$$

Solve for N, k_d as before