Photometric stereo
Announcements

• Project 3 released
  – Due Monday, April 22, by 11:59pm
  – To be done in groups of 2
  – Demo by Kai
Recap:
Lambertian (Diffuse) Reflectance

\[ I = k_d \mathbf{N} \cdot \mathbf{L} \]

- \( I \): observed image intensity
- \( k_d \): object albedo
- \( \mathbf{N} \): surface normal
- \( \mathbf{L} \): light source direction
Objects can have varying albedo and albedo varies with wavelength.

### Sample albedos

<table>
<thead>
<tr>
<th>Surface</th>
<th>Typical albedo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fresh asphalt</td>
<td>0.04[^4^]</td>
</tr>
<tr>
<td>Open ocean</td>
<td>0.06[^5^]</td>
</tr>
<tr>
<td>Worn asphalt</td>
<td>0.12[^4^]</td>
</tr>
<tr>
<td>Conifer forest (Summer)</td>
<td>0.08,[^6^] 0.09 to 0.15[^7^]</td>
</tr>
<tr>
<td>Deciduous trees</td>
<td>0.15 to 0.18[^7^]</td>
</tr>
<tr>
<td>Bare soil</td>
<td>0.17[^8^]</td>
</tr>
<tr>
<td>Green grass</td>
<td>0.25[^8^]</td>
</tr>
<tr>
<td>Desert sand</td>
<td>0.40[^9^]</td>
</tr>
<tr>
<td>New concrete</td>
<td>0.55[^8^]</td>
</tr>
<tr>
<td>Ocean ice</td>
<td>0.5–0.7[^8^]</td>
</tr>
<tr>
<td>Fresh snow</td>
<td>0.80–0.90[^8^]</td>
</tr>
</tbody>
</table>

Source:
Can we determine shape from lighting?

- Are these spheres?
  - Or just flat discs painted with varying albedo?
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?
  - But, deep learning can help

Suppose $k_d = 1$

\[
I = k_d N \cdot L
= N \cdot L
= \cos \theta_i
\]
Let’s take more than one photo!
Photometric stereo

Can write this as a matrix equation:

\[
\begin{bmatrix}
I_1 \\
I_2 \\
I_3
\end{bmatrix}
= k_d \begin{bmatrix}
L_1^T \\
L_2^T \\
L_3^T
\end{bmatrix} N
\]
Solving the equations

\[
\begin{bmatrix}
I_1 \\
I_2 \\
I_3
\end{bmatrix}
= 
\begin{bmatrix}
L_1^T \\
L_2^T \\
L_3^T
\end{bmatrix}
\begin{bmatrix}
k_d N
\end{bmatrix}
\]

\[
G = L^{-1} I
\]

\[
k_d = \|G\|
\]

\[
N = \frac{1}{k_d} G
\]

Solve one such linear system \textbf{per pixel} to solve for that pixel's surface normal
More than three lights

Get better results by using more lights

\[
\begin{bmatrix}
I_1 \\
\vdots \\
I_n
\end{bmatrix} = \begin{bmatrix}
L_1 \\
\vdots \\
L_n
\end{bmatrix} k_d N
\]

Least squares solution:

\[
\begin{align*}
I &= LG \\
L^T I &= L^T L G \\
G &= (L^T L)^{-1} (L^T I)
\end{align*}
\]

Solve for N, k_d as before

What's the size of \( L^T L \)?
Computing light source directions

Trick: place a chrome sphere in the scene

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

Recovered albedo

Recovered normal field

Forsyth & Ponce, Sec. 5.4
Depth from normals

- Solving the linear system per-pixel gives us an estimated surface normal for each pixel.

- How can we compute depth from normals?
  - Normals are like the “derivative” of the true depth.
Normal Integration

- Integrating a set of derivatives is easy in 1D
  - (similar to Euler’s method from diff. eq. class)

- Could just integrate normals in each column / row separately
- Instead, we formulate as a linear system and solve for depths that best agree with the surface normals
Depth from normals

Get a similar equation for $V_2$

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

\[
V_1 = (x + 1, y, z_{x+1,y}) - (x, y, z_{xy})
= (1, 0, z_{x+1,y} - z_{xy})
\]

\[
0 = N \cdot V_1
= (n_x, n_y, n_z) \cdot (1, 0, z_{x+1,y} - z_{xy})
= n_x + n_z(z_{x+1,y} - z_{xy})
\]
Results

from Athos Georghiades
Results
Extension

Photometric Stereo from Colored Lighting

Fig. 2. Applying the original algorithm to a face with white makeup. Top: example input frames from video of an actor smiling and grimacing. Bottom: the resulting integrated surfaces.

Video Normals from Colored Lights
Gabriel J. Brostow, Carlos Hernández, George Vogiatzis, Björn Stenger, Roberto Cipolla
Questions?
For now, ignore specular reflection
And Refraction…
And Interreflections...
And Subsurface Scattering…
Limitations

Bigger 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

Newer work addresses some of these issues

Some pointers for further reading:
• Hertzmann & Seitz, “Example-Based Photometric Stereo: Shape Reconstruction with General, Varying BRDFs.” IEEE Trans. PAMI 2005
Johnson and Adelson, 2009
Lights, camera, action
Figure 7: Comparison with the high-resolution result from the original retrographic sensor. (a) Rendering of the high-resolution $20 bill example from the original retrographic sensor with a close-up view. (b) Rendering of the captured geometry using our method.
Figure 9: Example geometry measured with the bench and portable configurations. Outer image: rendering under direct lighting. Inset: macro photograph of original sample. Scale shown in upper left. Color images are shown for context and are to similar, but not exact scale.
Sensing Surfaces with GelSight

https://www.youtube.com/watch?v=S7gXih4XS7A
InverseRenderNet: Learning single image inverse rendering

Ye Yu and William A. P. Smith
Department of Computer Science, University of York, UK
{yy1571,william.smith}@york.ac.uk

Figure 1: From a single image (col. 1), we estimate albedo and normal maps and illumination (col. 2-4); comparison multi-view stereo result from several hundred images (col. 5); re-rendering of our shape with frontal/estimated lighting (col. 6-7).
Questions?