CSE 559A: Computer Vision

Fall 2017: T-R: 11:30-1pm @ Lopata 101

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Staff: Abby Stylianou (abby@wustl.edu), Jarett Gross (jarett@wustl.edu)

http://www.cse.wustl.edu/~ayan/courses/cse559a/

Sep 28, 2017
Almost all PSET 1 submissions in! PSET 2 Out.
- Will post PSET 1 prob6 solution for use in PSET 2 (soon, on BB).

Same Monday Office Hours Location for next week.

Recitation Oct 6. Strongly suggest you try all problems in the HW before then.

PSET 2: Last two problems for normal to depth
- Un-comment line in code to use your own normal estimates
PHOTOMETRIC STEREO++

- Robust Photometric Stereo
  In the presence of shadows, specular highlights
    - Simple option: For each pixel, just drop the darkest n and brightest n pixels.

Robust Photometric Stereo via Low-Rank Matrix Completion and Recovery

Wu et al., PAMI 2011 / ACCV 2010
PHOTOMETRIC STEREO++

\[ I_1 = \ell_1^T (\rho \, \hat{n}) \]
\[ I_2 = \ell_2^T (\rho \, \hat{n}) \]
\[ I_3 = \ell_3^T (\rho \, \hat{n}) \]

Three measurements, three images, only works for static scenes!
PHOTOMETRIC STEREO++

\[ I_R = \ell_R^T(\rho_R \hat{n}) \]
\[ I_G = \ell_G^T(\rho_G \hat{n}) \]
\[ I_B = \ell_B^T(\rho_B \hat{n}) \]

Multiplex in color!!

Three observations, five unknowns.

(so close......)
PHOTOMETRIC STEREO++

Solutions?

- More colors, hyperspectral imaging?
  Every additional channel, adds another unknown.
  (But with narrow wavelength bands, you can assume albedo
  of neighboring bands vary smoothly).

- What if I knew albedo (or even just albedo chromaticity)?

  - Want to capture shape of object, let's paint it's surface with
    known color paint.

\[
I_R = \ell_R^T (\rho_R \hat{n}) \\
I_G = \ell_G^T (\rho_G \hat{n}) \\
I_B = \ell_B^T (\rho_B \hat{n})
\]

Multiplex in color!!

(PS: this wouldn't work, not lambertian)

Three observations, five unknowns.

(so close ......)
PHOTOMETRIC STEREO++

Solutions?

- More colors, hyperspectral imaging?
  Every additional channel, adds another unknown.
  (But with narrow wavelength bands, you can assume albedo
  of neighboring bands vary smoothly).

- What if I knew albedo (or even just albedo chromaticity) ?

  - Want to capture shape of object, let's paint it's surface with
    known color paint.
    (actually, used sometimes ... use powder instead of paint)

  - Pull something with known albedo tightly over object

\[ I_B = \ell_B^T (\rho_B \hat{n}) \]

Multiplex in color !!

Three observations, five unknowns.

(so close ....)
Photometric Stereo++

The GelSight Benchtop Scanner can quickly capture the surface geometry of almost any object regardless of its optical properties. This capability is due to the GelSight sensor, a small elastomer with a reflective coating on one side. The unique properties of the GelSight sensor allow for the influence of the optical characteristics of the material on the measurement, thereby ensuring repeatability and consistent performance, even on optically complex surfaces.
Solutions?

- More colors, hyperspectral imaging?
  Every additional channel, adds another unknown.
  (But with narrow wavelength bands, you can assume albedo
  of neighboring bands vary smoothly).

- What if I knew albedo (or even just albedo chromaticity)?

  - Want to capture shape of object, let's paint it's surface with
    known color paint.
    (actually, used sometimes ... use powder instead of paint)

  - Pull something with known albedo tightly over object

- Make assumptions: albedo is smooth, shape is smooth,
  solve with a prior / regularizer!

\[
I_R = \ell_R^T (\rho_R, \hat{n}) \\
I_G = \ell_G^T (\rho_G, \hat{n}) \\
I_B = \ell_B^T (\rho_B, \hat{n})
\]

Multiplex in color!!

Three observations, five unknowns.

(so close ......)
Multiple Point Light Sources, all at infinity, on at the same time (not PS)

\[ I = \ell_1^T (\rho \hat{n}) + \ell_2^T (\rho \hat{n}) + \ell_3^T (\rho \hat{n}) \]
GENERAL SHADING

General Illumination from Infinity

From infinity implies that the angles of light to the point won't depend on the point's position

Two points with the same surface normal and same albedo will reflect the same light
GENERAL SHADING

Lambertian Calibration Target

Environment as seen on a Glass Sphere
GENERAL SHADING

For known albedo, gives us total irradiance for every normal.

\[ L_o(\theta_o, \phi_o) = K_c \int L_i(\theta_i, \phi_i) \cos \theta_i d\omega_i \]
GENERAL SHADING

\[ L_o(\theta_o, \phi_o) = K_t \int L_i(\theta_i, \phi_i) \cos \theta_i d\omega_i \]

Instead of \(<n,l>\), now have a 'lookup table' for each normal.

Cumbersome, so sometimes approximated using "spherical harmonics"

\[ I = \rho \, \hat{n}^T L \, \hat{n} \]

\[ L_o(\theta_o, \phi_o) = K_c \int L_i(\theta_i, \phi_i) \cos \theta_i d\omega_i \]

Lambertian Calibration Target
GENERAL SHADING

Can apply the same idea to handle non-lambertian shading

- Put sphere of same material as object in to the scene.
  (assume constant BRDF on target)

- Assume both light and camera far away from the object.

- Then same normal on both sphere and object will produce the same intensity.
Want this to work on natural images taken in natural illumination....

- General unknown illumination environment
- General unknown shape
- General unknown albedo
  Object possibly not Lambertian
SIRFS
shape, illumination, and reflectance from shading

\[ Z \quad \text{shape / depth} \]
\[ S(Z, L) \quad \text{log-shading of } Z \text{ and } L \]
\[ L \quad \text{illumination} \]
\[ R \quad \text{log-reflectance} \]
\[ I = R + S(Z, L) \quad \text{Input log-image} \]

minimize \[ Z, R, L \]
\[ g(R) + f(Z) + h(L) \]

Suggested Reading:

Barron & Malik,
"Shape, Reflectance, and Illumination from Shading,"
PAMI 2015.
Remember, the pinhole camera

\[(x, y, z) \Rightarrow \left( -f \frac{x}{z}, -f \frac{y}{z} \right)\]
GEOMETRY

$3D \ (x, y, z) \Rightarrow \left( -f \frac{x}{z}, -f \frac{y}{z} \right) \ 2D$

- The division is annoying, makes projection non-linear.
- Can no longer use matrices / linear operations to relate co-ordinates.
- But we like matrix operations!

**Solution:** Homogeneous Co-ordinates
HOMOGENEOUS CO-ORDINATES

Book-keeping trick!

- 2D Cartesian Co-ordinates: \((x, y)\)
- 2D Homogeneous Co-ordinates: \((\alpha x, \alpha y, \alpha)\)
- Cartesian to Homogeneous: \((x, y) \rightarrow (\alpha x, \alpha y, \alpha)\)
  - When \(\alpha = 1\), this is called "augmented": \((x, y, 1)\)
- Homogeneous to Cartesian: \((x', y', \alpha) \rightarrow \left(\frac{x'}{\alpha}, \frac{y'}{\alpha}\right)\)
- A whole family of homogeneous co-ordinates map to the same cartesian co-ordinate
  - *Over-parameterization* of a 2D point
  - Denote this equality by \(\sim\): \((\alpha_1 x, \alpha_1 y, \alpha_1) \sim (\alpha_2 x, \alpha_2 y, \alpha_2)\)
- Space of 2D Homogeneous co-ordinates denoted as \(\mathbb{P}^2 = \mathbb{R}^3 - (0, 0, 0)\)
- Note that \((x, y, 0)\) is defined. In cartesian co-ordinates, it is the point at infinity along the line joining \((0, 0)\) to \((x, y)\).
- 3D Homogeneous Co-ordinates: \((x, y, z) \Rightarrow (\alpha x, \alpha y, \alpha z, \alpha)\)
Homogeneous co-ordinates
HOMOGENEOUS CO-ORDINATES

\[
\left( -f \frac{x}{z}, -f \frac{y}{z} \right) \leftrightarrow (x, y, z)
\]

\[
\begin{bmatrix}
  a \\
  b \\
  c
\end{bmatrix}
= 
\begin{bmatrix}
  -f & 0 & 0 & 0 \\
  0 & -f & 0 & 0 \\
  0 & 0 & 1 & 0 \\
\end{bmatrix}
\begin{bmatrix}
  \alpha x \\
  \alpha y \\
  \alpha z \\
  \alpha
\end{bmatrix}
\]

Homogeneous co-ordinates

\[
\frac{a}{c} = -f \frac{x}{z} \quad \frac{b}{c} = -f \frac{y}{z}
\]

Works for all non-zero values of \( \alpha \)
Homogeneous Co-ordinates

- Turned non-linear perspective projection into a non-linear operation.
- Here's a different projection matrix:

\[
P_{2d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} P_{3d}
\]

What does this represent?

\((x, y, z) \rightarrow (x, y)\)

Orthographic Projection
HOMOGENEOUS CO-ORDINATES

Perspective Projection

Orthographic Projection

Preserves parallel lines
Doesn't really correspond to a real camera
HOMOGENEOUS CO-ORDINATES

- Also useful to represent translation, rotation, skew in addition to projection
- Learn to chain together all these operations to:
  - Relate points in 3D to points in image
  - Verify angles, metric lengths from calibration targets, ...
  - Relate points in two images from different cameras