Announcements

- Assignment 3: Due date extended to 5:00 on Wed (tomorrow).

- Today
  - Review of Photometric Stereo
  - Discrete Structure from Motion.

Reflectance Map

Now if BRDF and light source direction/strength are known, then for each image point:

1. Image intensity is a function of only the direction of the surface normal.
2. In gradient space, we have $E(x,y) = R(p,q)$ where $E$ is measured, $(p,q)$ is unknown, but form of function $R(p,q)$ is known.

Coordinate system

Two Light Sources

Two reflectance maps

Third image would disambiguate match

Plastic Baby Doll: Normal Field
Recovering the surface \( z(x,y) \)

Many methods: Simplest approach
1. From estimate \( \mathbf{n} = (n_x, n_y, n_z), p=n_x/n_z, q=n_y/n_z \)
2. Integrate \( p = dz/dx \) along a row \((x,0)\) to get \( z(x,0) \)
3. Then integrate \( q = dz/dy \) along each column starting with value of first row to get \( z(x,y) \)

\[ z(x,0) \]

If \( f(x,y) \) is the height function, we expect that

In terms of estimated gradient space \( (p,q) \), this means:

But since \( p \) and \( q \) were estimated independently at each point as intersection of curves on three reflectance maps, equality is not going to exactly hold

Lambertian Surface

At image location \((u,v)\), the intensity of a pixel \( x(u,v) \) is:

\[ e(u,v) = \mathbf{a}(u,v) \cdot \mathbf{n}(u,v) \cdot s_0 \cdot s \]

where
- \( \mathbf{a}(u,v) \) is the albedo of the surface projecting to \((u,v)\).
- \( \mathbf{n}(u,v) \) is the direction of the surface normal.
- \( s_0 \) is the light source intensity.
- \( s \) is the direction to the light source.

Important Special Case: Lambertian Photometric Stereo

- If the light sources \( s_1, s_2, \) and \( s_3 \) are known, then we can recover \( b \) from as few as three images. (Photometric Stereo: Silver 80, Woodham81).

\[ \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix} = b^T \begin{bmatrix} s_1 & s_2 & s_3 \end{bmatrix} \]

- i.e., we measure \( e_1, e_2, \) and \( e_3 \) and we know \( s_1, s_2, \) and \( s_3 \). We can then solve for \( b \) by solving a linear system.

\[ b^T = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix} \begin{bmatrix} s_1 & s_2 & s_3 \end{bmatrix}^{-1} \]

- Normal is: \( \mathbf{n} = b/|b| \), albedo is: \( |b| \)

Bas-Relief Ambiguity

Light Sources \( (s_1, s_2, s_3) \) are unknown

Motion
Structure-from-Motion (SFM)

Goal: Take as input two or more images or video w/o any information on camera position/motion, and estimate camera position and 3-D structure of scene.

**Two Approaches**
1. Discrete motion (wide baseline)
   1. Orthographic (affine) vs. Perspective
   2. Two view vs. Multi-view
   3. Calibrated vs. Uncalibrated
2. Continuous (Infinitesimal) motion

Discrete Motion: Some Counting

Consider $M$ images of $N$ points, how many unknowns?

1. Affix coordinate system to location of first camera location: $(M-1)*6$ Unknowns
2. 3-D Structure: $3*N$ Unknowns
3. Can only recover structure and motion up to scale. Why?

Total number of unknowns: $(M-1)*6+3*N-1$
Total number of measurements: $2*M*N$
Solution is possible when $(M-1)*6+3*N-1 \leq 2*M*N$

Epipolar Constraint: Calibrated Case

$$\mathbf{p}^T (\mathbf{O} \times \mathbf{O'}) = 0$$

where $\mathbf{p} = [u, v, 1]^T$

$$\mathbf{p'}^T \mathbf{E} \mathbf{p} = 0$$

Essential Matrix (Longuet-Higgins, 1981)

The Eight-Point Algorithm (Longuet-Higgins, 1981)

Let $\mathbf{F}$ denote the Essential Matrix Here

$$\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = 0$$

Set $F_{33}$ to 1

Sketch of Two View SFM Algorithm

Input: Two images
1. Detect feature points
2. Find 8 matching feature points (easier said than done)
3. Compute the Essential Matrix $\mathbf{E}$ using Normalized 8-point Algorithm
4. Computer $\mathbf{R}$ and $\mathbf{T}$ (recall that $\mathbf{E} = \mathbf{S}\mathbf{R}$ where $\mathbf{S}$ is skew symmetric matrix)
5. Perform stereo matching using recovered epipolar geometry expressed via $\mathbf{E}$.
6. Reconstruct 3-D geometry of corresponding points.

Feature points

Select strongest features (e.g. 1000/image)
Finding Corners

Intuition:
- Right at corner, gradient is ill defined.
- Near corner, gradient has two different values.

$\nabla^2 I = \text{corner}$

Feature matching

Evaluate normalized cross correlation (or sum of squared differences) for all features with similar coordinates:

$\{x, y\} \approx \{x - \Delta x, y - \Delta y\}$

Keep mutual best matches

Still many wrong matches!

Detecting Feature points

(e.g. Harris & Stephens '88; Shi & Tomasi '94)

Find points that differ as much as possible from all neighboring points

$SSD \approx \Delta^T M \Delta$

$M = \iint \left| \frac{\partial}{\partial x} I(x, y) \right|^2 + \left| \frac{\partial}{\partial y} I(x, y) \right|^2 dxdy$

$M$ should have large eigenvalues

Feature = local maxima of $F(\lambda_1, \lambda_2)$

Comments

- Greedy Algorithm:
  - Given feature in one image, find best match in second image irrespective of other matches.
  - OK for small motions, little rotation, small search window.
- Otherwise:
  - Must compare descriptor over rotation.
  - Can’t consider $O(n^2)$ potential pairings (way too many), so
    - Manual correspondence (e.g., façade, photogrametry).
    - Use random sampling (RANSAC).
    - More descriptive features (line segments, larger regions, color).
    - Use video sequence to track, but perform SFM w/ first and last image.

Continuous Motion

- Consider a video camera moving continuously along a trajectory (rotating & translating).
  - How do points in the image move?
  - What does that tell us about the 3-D motion & scene structure?

Motion

Some problems of motion:
1. Correspondence: Where have elements of the image moved between image frames
2. Reconstruction: Given correspondence, what is 3-D geometry of scene
3. Segmentation: What are regions of image corresponding to different moving objects
4. Tracking: Where have objects moved in the image? related to correspondence and segmentation.

Variations:
- Small motion (video),
- Wide-baseline (multi-view)
Motion

“When objects move at equal speed, those more remote seem to move more slowly.”
- Euclid, 300 BC

Simplest Idea for video processing

Image Differences

• Given image $I(u,v,t)$ and $I(u,v, t+\delta t)$, compute $I(u,v, t+\delta t) - I(u,v, t)$.
• This is partial derivative: $\frac{\partial I}{\partial t}$
• At object boundaries, $\frac{\partial I}{\partial t}$ is large and is a cue for segmentation
• Doesn’t tell which way stuff is moving

Background Subtraction

• Gather image $I(x,y,t_0)$ of background without objects of interest (perhaps computed over average over many images).
• At time $t$, pixels where $|I(x,y,t)-I(x,y,t_0)| > \tau$ are labeled as coming from foreground objects

The Motion Field

Where in the image did a point move?

• Camera moves (translates, rotates)
• Object’s in scene move rigidly
• Objects articulate (pliers, humans, animals)
• Objects bend and deform (fish)
• Blowing smoke, clouds
Motion Field Yields 3-D Motion Information

The "instantaneous" velocity of points in an image

The Focus of Expansion (FOE)

Intersection of velocity vector with image plane

With just this information it is possible to calculate:
1. Direction of motion
2. Time to collision

Is motion estimation inherent in humans?

Demo

Rigid Motion and the Motion Field

Rigid Motion: General Case

Position & Orientation

Rotation Matrix & Translation vector

\[ \dot{p} = T + \omega \times p \]

General Motion

\[
\begin{bmatrix}
\dot{u} \\
\dot{v}
\end{bmatrix} = f \begin{bmatrix}
\dot{x} \\
\dot{y}
\end{bmatrix} \]

\[
\begin{bmatrix}
\dot{u} \\
\dot{v}
\end{bmatrix} = f \begin{bmatrix}
\dot{z} \\
\dot{y}
\end{bmatrix} - \frac{z}{f} \begin{bmatrix}
x \\
y
\end{bmatrix}
\]

Substitute \( \dot{p} = T + \omega \times p \) where \( p = (x, y, z) \)

Motion Field Equation

\[
\begin{align*}
\dot{u} &= \frac{T_z - T_y f}{Z} - \omega_y f - \omega_z v + \frac{\omega_y u v}{f} - \frac{\omega_z u^2}{f} \\
\dot{v} &= \frac{T_x - T_z f}{Z} + \omega_z f - \omega_x v - \frac{\omega_x u v}{f} - \frac{\omega_z y^2}{f}
\end{align*}
\]

- \( T \): Components of 3-D linear motion
- \( \omega \): Angular velocity vector
- \( (u,v) \): Image point coordinates
- \( Z \): depth
- \( f \): focal length
Pure Translation
\[ \begin{align*}
\dot{u} &= \frac{T_u - T_s f}{Z} - \omega_y f + \frac{\omega_z v}{f} - \frac{\omega_z u^2}{f} \\
\dot{v} &= \frac{T_v - T_s f}{Z} + \omega_x f - \frac{\omega_x v^2}{f} + \frac{\omega_x u^2}{f}
\end{align*} \]
\[ \omega = 0 \]

Pure Translation
- Parallel (FOE point at infinity)
- Radial about FOE
- Motion parallel to image plane

Forward Translation & Focus of Expansion
[Gibson, 1950]

Pure Rotation: T=0
\[ \begin{align*}
\dot{u} &= \frac{T_u - T_s f}{Z} - \omega_y f + \frac{\omega_z v}{f} - \frac{\omega_z u^2}{f} \\
\dot{v} &= \frac{T_v - T_s f}{Z} + \omega_x f - \frac{\omega_x v^2}{f} + \frac{\omega_x u^2}{f}
\end{align*} \]
- Independent of \( T_x, T_y, T_z \)
- Independent of Z
- Only function of \((u,v), f\) and \(\omega\)

Rotational MOTION FIELD
The “instantaneous” velocity of points in an image

Pure Rotation
\[ \omega = (0,0,1)^T \]

Motion Field Equation: Estimate Depth
\[ \begin{align*}
\dot{u} &= \frac{T_u - T_s f}{Z} - \omega_y f + \frac{\omega_z v}{f} - \frac{\omega_z u^2}{f} \\
\dot{v} &= \frac{T_v - T_s f}{Z} + \omega_x f - \frac{\omega_x v^2}{f} + \frac{\omega_x u^2}{f}
\end{align*} \]
If \( T, \omega, \) and \( f \) are known or measured, then for each image point \((u,v)\), one can solve for the depth \( Z \) given measured motion (\( \frac{du}{dt}, \frac{dv}{dt} \)) at \((u,v)\).
Optical Flow

Optical Flow: Where do pixels move to?

Definition of optical flow

OPTICAL FLOW = apparent motion of brightness patterns

Ideally, the optical flow is the projection of the three-dimensional velocity vectors on the image

Mathematical formulation

[Note change of notation: image coordinates now (x,y), not (u,v)]

\[ I(x,y,t) = \text{brightness at image point } (x,y) \text{ at time } t \]

Consider scene (or camera) to be moving, so \( x(t), \ y(t) \)

Brightness constancy assumption:
\[ I(x + \frac{dx}{dt} \partial_x, y + \frac{dy}{dt} \partial_y, t + \frac{dt}{dt}) = I(x, y, t) \quad \Rightarrow \quad \frac{dI}{dt} = 0 \]

Optical flow constraint equation:
\[ \frac{dI}{dt} = \frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial y} \frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \]
Solving for flow

Optical flow constraint equation:
\[
\frac{dl}{dt} = \frac{\partial l}{\partial x} \frac{dx}{dt} + \frac{\partial l}{\partial y} \frac{dy}{dt} + \frac{\partial l}{\partial t} = 0
\]

- We can measure \(\frac{\partial l}{\partial x}, \frac{\partial l}{\partial y}, \frac{\partial l}{\partial t}\)
- We want to solve for \(\frac{dx}{dt}, \frac{dy}{dt}\)
- One equation, two unknowns

Measurements

\(l_x = \frac{\partial l}{\partial x}\)
\(l_y = \frac{\partial l}{\partial y}\)
\(l_t = \frac{\partial l}{\partial t}\)

Flow vector

\(u = \frac{dx}{dt}\)
\(v = \frac{dy}{dt}\)

The component of the optical flow in the direction of the image gradient.

What is the correspondence of P & P’

Contour plots of image intensity in two images

Aperture Problem and Normal Flow

The gradient constraint:
\[
I_x u + I_y v + I_t = 0
\]
\[
\nabla I \cdot \vec{U} = 0
\]
Defines a line in the \((u,v)\) space

Normal Flow

Illusion Works Barber Pole Illusion

Two ways to get flow

1. Think globally, and regularize over image
2. Look over window and assume constant motion in the window
Lucas-Kanade: Integrate over a Patch

Assume a single velocity for all pixels within an image patch

\[ E(u, v) = \sum_{x, y} (I(x, y)u + I(x, y)v + I) \]

\[ \frac{dE(u, v)}{du} = 2I_x(u + I_x + I) = 0 \]
\[ \frac{dE(u, v)}{dv} = 2I_y(u + I_y + I) = 0 \]

Solve with:

\[ \begin{bmatrix} \frac{\sum I_x^2}{\sum I_x I_y} & \frac{\sum I_x}{\sum I_x I_y} \\ \frac{\sum I_y}{\sum I_x I_y} & \frac{\sum I_y^2}{\sum I_x I_y} \end{bmatrix} = \begin{bmatrix} \sum I_x I_x \\ \sum I_x I_y \end{bmatrix} \]

On the LHS: sum of the 2x2 outer product tensor of the gradient vector

\[ \left( \sum \nabla I \nabla I^T \right) \mathbf{u} = -\nabla I \]

Iterative Refinement

- Estimate velocity at each pixel using one iteration of Lucas and Kanade estimation
- Warp one image toward the other using the estimated flow field (easier said than done)
- Refine estimate by repeating the process

Some variants

- Coarse to fine (image pyramids)
- Local/global motion models
- Robust estimation

Limits of the (local) gradient method

1. Fails when intensity structure within window is poor
2. Fails when the displacement is large (typical operating range is motion of 1 pixel per iteration!)
   - Linearization of brightness is suitable only for small displacements
Also, brightness is not strictly constant in images
   - actually less problematic than it appears, since we can pre-filter images to make them look similar

Pyramid / “Coarse-to-fine”
Motion Model Example: Affine Motion

\[
\begin{align*}
\text{Affine:} & \quad \mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \\
& \quad \mathbf{h} = \begin{bmatrix} \delta x \\ \delta y \end{bmatrix}
\end{align*}
\]

Parametric (Global) Motion Models

**2D Models:**
- Translation
- Affine
- Quadratic
- Planar projective transform (Homography)

**3D Models:**
- Instantaneous camera motion models
- Homography+epipole
- Plane+Parallax

Robust Estimation

Quadratic 0 function gives too much weight to outliers.

\[
\begin{align*}
\rho(r; \sigma) &= \frac{r^2}{\sigma^2 + r^2} \\
\psi(r; \sigma) &= \frac{2r\sigma^2}{(\sigma^2 + r^2)^2}
\end{align*}
\]