Optical Flow and Motion

Introduction to Computer Vision I
CSE 152A
Lecture 12
Announcements

• Assignment 3 is due Feb 24, 11:59 PM
• Quiz 3 is Feb 26
• Reading:
  – Szeliski
    • Sections 9.1 (intro), 9.1.1, 9.1.3, 9.2 (intro), and 9.4 (intro)
  – Introductory Techniques for 3-D Computer Vision, Trucco and Verri (Course Reserves)
    • Chapter 8: Motion
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 3D motion & scene structure?
Motion

- https://www.youtube.com/watch?v=TKsVVmOGV9I
Structure-from-Motion (SFM)

Given two or more images or video without any information on camera position/motion as input, estimate camera motion and 3-D structure of a scene

Two Approaches
1. Discrete motion (wide baseline)
2. Continuous (Infinitesimal) motion usually from video
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

- Does not indicate which way objects are moving
The Motion Field

Where in the image did a point move?

Down and left
Motion field

- The motion field is the projection of the 3D scene motion into the image
Motion blur.
Usually in direction of motion field
The Motion Field
What causes a motion field?

1. Camera moves (translates, rotates)
2. Objects in scene move rigidly
3. Objects articulate (pliers, humans, animals)
4. Objects bend and deform (fish)
5. Blowing smoke, clouds
6. Multiple movements
An example motion field: Camera moving straight along optical axis

The “instantaneous” velocity of all points in an image

LOOMING

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
Rigid Motion: General Case

Position and orientation of a rigid body
Rotation Matrix & Translation vector

\[ P = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \]

\[ Z' = C_Z - Z \]

Rigid Motion:
Velocity Vector: \( T \)
Angular Velocity Vector: \( \omega \) (or \( \Omega \))

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

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

Let \((x,y,z)\) be functions of time \((x(t), y(t), z(t))\):

\[
\begin{bmatrix}
  \dot{u} \\
  \dot{v}
\end{bmatrix} = \frac{f}{z} \begin{bmatrix}
  \dot{x} \\
  \dot{y}
\end{bmatrix} - \frac{z}{z^2} \begin{bmatrix}
  x \\
  y
\end{bmatrix}
\]

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

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

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

- **Image**
  - \((u,v)\): Image point coordinates
  - \((\dot{u}, \dot{v})\): Image point velocity

- **Camera**
  - \(\mathbf{T}\): Components of 3-D linear motion
  - \(\omega\): Angular velocity vector
  - \(f\): focal length

- **Scene**
  - \(z\): depth
Pure Translation

If camera is just translating with velocity \((T_x, T_y, T_z)\), there’s no rotation:

\[
\omega = 0
\]

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

- \(\dot{u}, \dot{v}\) is inversely proportional to \(Z\) (remember Euclid)
- Focus of expansion is located at \((u,v)\) where \(\dot{u} = \dot{v} = 0\)
Forward Translation & Focus of Expansion

[Gibson, 1950]
Focus of Expansion (FOE)

\[
\dot{u} = \frac{T_z u - T_x f}{Z} \\
\dot{v} = \frac{T_z v - T_y f}{Z}
\]

- Focus of expansion is located at \((u,v)\) where \(\dot{u} = \dot{v} = 0\)

\[
T_z u - T_x f = 0 \\
T_z v - T_y f = 0
\]

- Solve for \(u,v\)

\[
u = f \frac{T_x}{T_z} \\
v = f \frac{T_y}{T_z}
\]

Insight: The FOE is the perspective projection of the linear velocity vector \((T_x, T_y, T_z)\).
Pure Translation

- Radial about FOE
- Parallel (FOE point at infinity)
  \[ T_z = 0 \]
- Motion parallel to image plane
Sideways Translation

[Ref: Gibson, 1950]

Parallel

(FOE point at infinity)

\[ T_z = 0 \]

Motion parallel to image plane
Pure Rotation: $T=0$

\[
\begin{align*}
\dot{u} &= \frac{T_{xu} - T_{xf}}{f} - \omega_y f + \omega_z v + \frac{\omega_x u v}{f} - \frac{\omega_y u^2}{f} \\
\dot{v} &= \frac{T_{xy} - T_{vf}}{Z} + \omega_x f - \omega_z u + \frac{\omega_y u v}{f} - \frac{\omega_x v^2}{f}
\end{align*}
\]

- 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

\[ \omega = (0, 0, 1)^T \]
Pure translation and pure rotation: Motion Field on Sphere
Motion Field Equation: Estimate Depth

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

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 \( \dot{u}, \dot{v} \) at \((u,v)\).

\[
Z = \frac{T_z u - T_x f}{\dot{u} + \omega_y f - \omega_z v - \frac{\omega_x uv}{f} + \frac{\omega_y u^2}{f}}
\]

Inversely proportional to image velocity
Measuring Motion
Optical Flow Field
Estimating the motion field from images

1. Feature-based (Sect. 8.4.2 of Trucco & Verri)
   1. Detect (corner-like) features in an image
   2. Search for the same features nearby (feature tracking)

2. Differential techniques (Sect. 8.4.1)
Optical Flow

• Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene.

• Ideally, the optical flow is the projection of the three-dimensional velocity vectors on the image. i.e., the motion field.
  – As we will see, it is not
Problem Definition: Optical Flow

- How to estimate pixel motion from image $H$ to image $I$?
  - Find pixel correspondences
    - Given a pixel in $H$, look for nearby pixels of the same color in $I$

- Key assumptions
  - color constancy: a point in $H$ looks “the same” in image $I$
    - For grayscale images, this is brightness constancy
  - small motion: points do not move very far
Warning

- Notation shift to match Trucco and Verri
  - \((x,y)\) will be image positions
  - \((u,v)\) will be image velocities
Optical Flow Constraint Equation

1. Assume brightness of patch remains the same in both images:

\[ I(x + u \delta t, y + v \delta t, t + \delta t) = I(x, y, t) \]

2. Assume small motion: (Taylor expansion of LHS up to first order)

\[ I(x, y, t) + \delta x \frac{\partial I}{\partial x} + \delta y \frac{\partial I}{\partial y} + \delta t \frac{\partial I}{\partial t} = I(x, y, t) \]
Optical Flow Constraint Equation

Optical Flow: Velocities \((u, v)\)

Displacement:
\[
(\delta x, \delta y) = (u \ \delta t, v \ \delta t)
\]

3. Subtracting \(I(x,y,t)\) from both sides and dividing by \(\delta t\)
\[
\frac{\delta x}{\delta t} \frac{\partial I}{\partial x} + \frac{\delta y}{\delta t} \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0
\]

4. Assume small interval, this becomes:
\[
\frac{dx}{dt} \frac{\partial I}{\partial x} + \frac{dy}{dt} \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0
\]
Solving for flow

Optical flow constraint equation :

\[
\frac{dx}{dt} \frac{\partial I}{\partial x} + \frac{dy}{dt} \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0
\]

- We can measure \( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t} \)
  - Convolve image with \([-1, 0, 1]\)
  - Convolve image with \([-1, 0, 1]^T\)
  - Consider stacking 3 images at \((t-1, t, t+1)\).
    Convolve with \([-1, 0, 1]\) over time

- We want to solve for \( \frac{dx}{dt}, \frac{dy}{dt} \)
- One equation, two unknowns \(\Rightarrow\) Cannot solve it
Aperture Problem and Normal Flow

We measure:

\[ I_x = \frac{\partial I}{\partial x}, \]
\[ I_y = \frac{\partial I}{\partial y}, \]
\[ I_t = \frac{\partial I}{\partial t} \]

We want to estimate Flow vector

\[ u = \frac{dx}{dt}, \]
\[ v = \frac{dy}{dt} \]

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

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
Barber Pole Illusion

Optical flow field is not always the same as the motion field

http://www.opticalillusion.net/optical-illusions/the-barber-pole-illusion/
Optical Flow $\neq$ Motion Field

- Motion field exists but no optical flow
- No motion field but shading changes
Lucas-Kanade: Integrate over a Patch

Assume a single velocity \((u, v)\) for pixels within an image patch \(\Omega\)

\[
E(u, v) = \sum_{x, y \in \Omega} \left( I_x(x, y)u + I_y(x, y)v + I_t \right)^2
\]

\(E(u, v)\) is minimized when partial derivatives equal zero.

\[
\frac{dE(u, v)}{du} = \sum 2I_x \left( I_x u + I_y v + I_t \right) = 0
\]

\[
\frac{dE(u, v)}{dv} = \sum 2I_y \left( I_x u + I_y v + I_t \right) = 0
\]

In matrix form:

\[
\begin{bmatrix}
\sum I_x^2 & \sum I_x I_y \\
\sum I_x I_y & \sum I_y^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t
\end{bmatrix}
\]

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

\[
\left( \sum \nabla I \nabla I^T \right) \vec{U} = -\sum \nabla I I_t
\]
Lukas-Kanade

Let \( M = \sum (\nabla I)(\nabla I)^T \) and \( b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix} \)

- So, the optical flow \( U = (u, v) \) can be written as \( MU = b \)
- And optical flow is just \( U = M^{-1}b \)
Lukas-Kanade: Singularities & Aperture Problem

Let \( M = \sum (\nabla I)(\nabla I)^T \) and \( b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix} \)

- Algorithm: At each pixel compute \( U \) by solving \( MU = b \)

- \( M \) is singular if
  - constant brightness in image: \( \nabla I = 0 \)
  - Window is one pixel
  - Along an edge (where the direction of \( \nabla I \) is the same (or zero) in the window)

- \( M \) is full rank for corners and texture regions
Edge

\[ M = \sum (\nabla I)(\nabla I)^T \]

- large gradients, all the same
- Eigenvalues of M: large \( \lambda_1 \), small \( \lambda_2 \)
Low texture region

- gradients have small magnitude
- Eigenvalues of $M$: small $\lambda_1$, small $\lambda_2$

$$M = \sum (\nabla I)(\nabla I)^T$$
High textured region

\[ M = \sum (\nabla I)(\nabla I)^T \]

- gradients are different, large magnitudes
- Eigenvalues of M: large \( \lambda_1 \), large \( \lambda_2 \)
Some variants

- Iterative refinement
- Coarse to fine (image pyramids)
- Local/global motion models
- Robust estimation
Revisiting the small motion assumption

• Is this motion small enough?
  – Probably not—it’s much larger than one pixel (2\textsuperscript{nd} order terms dominate)
  – How might we solve this problem?
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
Pyramid / “Coarse-to-fine”
Coarse-to-fine optical flow estimation

1. Gaussian pyramid of image J
2. Gaussian pyramid of image I
3. Run iterative L-K on image J
4. Warp & upsample
5. Run iterative L-K on image I
6. Repeat steps 3-5 for finer levels of the pyramid.
Coarse-to-Fine Estimation

\[ I_x \cdot u + I_y \cdot v + I_t \approx 0 \]

\( \implies \) small \( u \) and \( v \) ...

Pyramid of image J

Pyramid of image I
Multi-resolution Lucas Kanade Algorithm

- Compute ‘simple’ LK at highest level
- At level $i$
  - Take flow $u_{i-1}$, $v_{i-1}$ from level $i-1$
  - bilinear interpolate it to create $u_i^*$, $v_i^*$ matrices of twice resolution for level $i$
  - multiply $u_i^*$, $v_i^*$ by 2
  - compute $f_t$ from a block displaced by $u_i^*(x,y)$, $v_i^*(x,y)$
  - Apply LK to get $u_i'(x,y)$, $v_i'(x,y)$ (the correction in flow)
  - Add corrections $u_i$, $v_i'$, i.e. $u_i = u_i^* + u_i'$, $v_i = v_i^* + v_i'$. 
Parametric (Global) Motion Models

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

3D Models:
Instantaneous camera motion models
Homography+epipole
Plane+Parallax
Motion Model Example: Affine Motion

Affine: \[ A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad h = \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} \]
Visual Tracking

Optical flow is pixel-level tracking.
Now we consider tracking objects

Main Challenges
1. 3D pose variation
2. Target occlusion
3. Illumination variation
4. Camera jitter
5. Expression variation etc.

[ Ho, Lee, Kriegman ]
Main tracking notions

- **State** \( \phi(t) \): usually a finite number of parameters (a vector) that characterizes the “state” (e.g., location, size, pose, deformation of thing being tracked.

- **Dynamics** \( \dot{\phi}(t) \): How does the state change over time? How is that changed constrained?

- **Prediction**: Given the state \( \phi(t) \) at time \( t \), what is an estimate \( \phi_p(t + 1) \)? Use \( \phi(t) \) and \( \dot{\phi}(t) \).

- **Representation**: How do you represent the thing being tracked

- **Data Association**: Which measurements correspond to which object?

- **Correction**: Given the predicted state \( \phi_p(t + 1) \) at time \( t+1 \), and a measurement at time \( t+1 \), update the state \( \phi_c(t+1) = f(\phi_p(t + 1), M(t + 1)) \)

- **Initialization** – what is the state at time \( t=0 \)?
Tracking by detection

• Example: Structured Output Tracking with Kernels

https://youtu.be/gnT34hJwdjM
Next Lectures

• Recognition, detection, and classification
• Reading:
  – Szeliski
    • Sections 5.1 (intro), 5.1.1, 5.1.2, 5.1.4, 6.1, 6.2 (intro), and 6.2 (intro), 6.2.1, 6.3 (intro), 6.3.1, and 6.3.3