Motion

Introduction to Computer Vision
CSE 152
Lecture 14

• HW3 assigned

• Reading from Trucco & Verri is on Ereserves
Motion

- https://www.youtube.com/watch?v=TKsVVMoGV9I
Estimate 3D structure from images
Structure-from-Motion (SFM)

Given two or more images or video w/o 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

Random Dot Kinnetogram

- Motion reveals structure

- Ego motion
  - https://www.youtube.com/watch?v=y4JRamQvKrM

- 3D structure from motion + ambiguity
  - https://www.youtube.com/watch?v=DLBkwig3M2U
Small Motion

Motion

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

What causes a motion field?

1. Camera moves (translates, rotates)
2. Object’s 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: \( \mathbf{R} \)
Translation vector: \( \mathbf{t} \)

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

\[ \mathbf{Z}' = C_{Z} - Z \]

Rigid Motion:
Velocity vector: \( \mathbf{T} \)
Angular velocity vector: \( \mathbf{\omega} \)

\[ \dot{\mathbf{P}} = \mathbf{T} + \mathbf{\omega} \times \mathbf{P} \]

I know, using \( \mathbf{T} \) for velocity is poor notation, but it’s consistent with Trucco and Verri textbook
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 \((x(t), y(t), z(t))\):

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

\[
= \frac{f}{z} \begin{bmatrix}
\dot{x} \\
\dot{y}
\end{bmatrix} - \frac{1}{z} \begin{bmatrix}
u \\
v
\end{bmatrix}
\]

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

---

Motion Field Equation

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

- Image
  - \((u,v)\): Image point coordinates
  - \((u,v)\): Image point velocity

- Camera
  - \(T\): 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: \[\omega = 0\]

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

- \(\dot{u}, \dot{v}\) 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)**

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

- Focus of expansion is located at \((u,v)\) where \(\hat{u} = \hat{v} = 0\)
  
  \[ T_z u - T_x f = 0 \]
  \[ T_z v - T_y f = 0 \]

- 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

[Gibson, 1950]

Parallel
(FOE point at infinity)

\[ T_Z = 0 \]

Motion parallel to image plane

---

**Pure Rotation: T=0**

\[
\begin{align*}
u &= \frac{x}{f} + \omega_y f - \frac{\omega_z u^2}{f} - \omega_v u - \frac{\omega_u v}{f} \\
v &= \frac{y}{f} + \omega_x f - \omega_z u - \frac{\omega_v v^2}{f} - \omega_u v
\end{align*}
\]

- Independent of \( T_x, T_y, T_z \)
- Independent of \( Z \)
- Only function of image plane position \((u,v), f\) and \( \omega \)
Rotational Motion Field

The “instantaneous” velocity of points in an image

Pure Rotation

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

Motion Field Equation: Depth Estimation

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 \((u, v)\).

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

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

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

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 Features (corners) 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’ll see, it’s not but more on that later.
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 called brightness constancy
  – small motion: points do not move very far

Warning

• Notation shift:
  • (x,y) will be image positions
  • (u,v) will be image velocities

• Why? Notation in Trucco & Verri
Optical Flow Constraint Equation

- Assume brightness of patch remains same in both images:
  \[ I(x + u \, \delta t, y + v \, \delta t, t + \delta t) = I(x, y, t) \]

- Assume small motion: (Taylor series expansion of LHS about \((x, y, t)\) and retain first order terms)
  \[ I(x, y, t) + \delta x \, \frac{\partial I}{\partial x} + \delta y \, \frac{\partial I}{\partial y} + \delta t \, \frac{\partial I}{\partial t} = I(x, y, t) \]

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

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

Optical Flow Constraint Equation

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

- Assume small \(\delta t\), 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 \( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t} \)
  - \( \frac{\partial I}{\partial x} \) Convolve image with \([-1, 0, 1]\)
  - \( \frac{\partial I}{\partial y} \) Convolve image with \([-1, 0, 1]^T\)
  - \( \frac{\partial I}{\partial t} \) 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 \(\rightarrow\) Can’t solve it

---

Aperture Problem and Normal Flow

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

We want to estimate Flow vector
\[
\begin{align*}
\frac{dx}{dt} = u, \\
\frac{dy}{dt} = v
\end{align*}
\]

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:
\[
\begin{align*}
\begin{bmatrix} u \end{bmatrix} & = - \frac{I_t}{|\nabla I|} \nabla I \\
\begin{bmatrix} v \end{bmatrix} & = \frac{I_t}{|\nabla I|} \nabla I
\end{align*}
\]

The component of the optical flow in the direction of the image gradient.
Barber Pole Illusion

Optical flow field isn’t 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
Apparently an aperture problem

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.

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

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

Rewrite as

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

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

\[
\left( \sum \nabla I \nabla I^T \right) \ddot{\mathbf{u}} = - \sum \nabla II_t
\]
Lukas-Kanade (cont.)

Let \( M = \sum (\nabla I)(\nabla I)^\top \) and \( b = \left[ -\sum I_i I_i \right] \)

- So, the optical flow \( U = (u,v) \) can be written as
  \[ MU = b \]
- And optical is just \( U = M^{-1}b \)

Lukas-Kanade: Singularities & Aperture Problem

Let \( M = \sum (\nabla I)(\nabla I)^\top \) and \( b = \left[ -\sum I_i I_i \right] \)

- 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)
    - Aperature problem still exits
- \( M \) is full rank for corners and texture regions
Edge

- 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 $\lambda_2$
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

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