Motion

Computer Vision I
CSE252A
Lecture 18

Announcements

• HW3 is due today
• HW4 will be assigned, due Friday Dec. 4

Motion Field Equation

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

• \(T\): Components of 3-D linear motion
• \(\omega\): Angular velocity vector
• \((u,v)\): Image point coordinates
• \(Z\): depth
• \(f\): focal length

Forward Translation & Focus of Expansion

[ Gibson, 1950 ]

Pure Rotation: Motion Field on Sphere
Motion Field Equation: Estimate Depth

\[ \begin{align*}
\frac{du}{Z} &= T_u - T_f f - \omega_y f + \omega_x v - \omega_z \frac{u^2}{f} \\
\frac{dv}{Z} &= T_v - T_f f + \omega_y f - \omega_x u - \omega_z \frac{v^2}{f}
\end{align*} \]

If \( T, \omega_y \), 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 \((u,v)\).

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.

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

Mathematical formulation

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

Consider scene (or camera) to be moving, so \((x, y)\) is a function of time (i.e., \(x(t), y(t)\)), and point is moving with velocity \((\frac{dx}{dt}, \frac{dy}{dt})\)

**Brightness constancy assumption:**

\[ I(x + \frac{dx}{dt}, y + \frac{dy}{dt}, t + \frac{dt}{dt}) = I(x, y, t) \quad \Rightarrow \quad \frac{dI}{dt} = 0 \]

**Optical flow constraint equation:**

\[ \frac{dl}{dt} = \frac{ds}{dx} \frac{dx}{dt} + \frac{ds}{dy} \frac{dy}{dt} + \frac{ds}{dt} = 0 \]

Optical Flow Constraint Equation

- Assume brightness of patch remains same in both images:
  \[ E(x + u \Delta t, y + v \Delta t, t + \Delta t) = E(x, y, t) \]
  \[ \frac{\partial E}{\partial x} + \frac{\partial E}{\partial y} + \frac{\partial E}{\partial t} = E(x, y, t) \]
Optical Flow Constraint Equation

\[ \frac{\partial E}{\partial x} + \frac{\partial E}{\partial y} + \frac{\partial E}{\partial t} = 0 \]

Divide by \( \partial t \) and take the limit \( \delta t \to 0 \)

Constraint Equation

\[ E_x u + E_y v + E_t = 0 \]

NOTE: \((u, v)\) must lie on a straight line

We can compute \( E_x, E_y, E_t \) from the images using gradient operators!

But, \((u, v)\) cannot be found uniquely with this constraint!

Measurements

Flow vector

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

Normal Flow

Illusion Works Barber Pole Illusion

Edge

Large gradients, all the same

Large \( \lambda_1 \), small \( \lambda_2 \)
Low texture region

\[ \sum \nabla I(\nabla I)^T \]
- gradients have small magnitude
- small \( \lambda_1 \), small \( \lambda_2 \)

High textured region

\[ \sum \nabla I(\nabla I)^T \]
- gradients are different, large magnitudes
- large \( \lambda_1 \), large \( \lambda_2 \)

Some variants

- 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?

Pyramid / “Coarse-to-fine”

Coarse-to-fine optical flow estimation

- Gaussian pyramid of image \( I \)
- Gaussian pyramid of image \( H \)
- run iterative L-K
- warp & upsample
- run iterative L-K
Multi-resolution Lucas Kanade Algorithm

- Compute ‘simple’ LK at highest level
- At level \( i \)
  - Take flow \( u_i, v_i \) from level \( i-1 \)
  - Bilinear interpolate \( I \) to create \( u_i', v_i' \)
    matrices of twice resolution for level \( i \)
  - Multiply \( u_i', v_i' \) by 2
  - Compute \( I \) from a block displaced by \( u_i', v_i' \)
  - Apply LK to get \( u', v' \) (the correction in flow)
- Add corrections \( u = u_i' + u_c, v = v_i' + v_c \)

Motion Model Example: Affine Motion

\[
\begin{bmatrix}
\delta x \\
\delta y
\end{bmatrix} = \begin{bmatrix}
a & b \\
c & d
\end{bmatrix}\begin{bmatrix}
x \\
y
\end{bmatrix}
\]

Visual Tracking

Computer Vision I
CSE252A
Lecture 18
Main tracking notions

- **State**: usually a finite number of parameters (a vector) that characterizes the "state" (e.g., location, size, pose, deformation of thing being tracked).
- **Dynamics**: How does the state change over time? How is that change constrained?
- **Representation**: How do you represent the thing being tracked?
- **Prediction**: Given the state at time $t-1$, what is an estimate of the state at time $t$?
- **Correction**: Given the predicted state at time $t$ and a measurement at time $t$, update the state.
- **Initialization**: What is the state at time $t=0$?

What is state?

- 2-D image location, $\Phi=(u,v)$
- Image location + scale $\Phi=(u,v,s)$
- Image location + scale + orientation $\Phi=(u,v,s,\theta)$
- Affine transformation
- 3-D pose
- 3-D pose plus internal shape parameters (some may be discrete).
  - e.g., for a face, 3-D pose + facial expression using FACS + eye state (open/closed).
- Collections of control points specifying a spline
- Above, but for multiple objects (e.g., tracking a formation of airplanes).
- Augment above with temporal derivatives $(\dot{\Phi}, \ddot{\Phi})$

State Examples:

- object is ball, state is 3D position + velocity, measurements are stereo pairs
- object is person, state is body configuration, measurements are frames
- What is state here?

Example: Blob Tracker

- From input image $I(u,v)$ (color?) at time $t$, create a binary image by applying a function $I(I(u,v))$.
- Clean up binary image using morphological operators
- Perform connected component exploration to find “blobs” – connected regions.
- Compute their moments (mean and covariance of coordinates of region), and use as state
- Using state estimate from $t-1$ and perform “data association” to identify state in from $t$. 
Blob Tracking in IR Images

- Threshold about body temperature
- Connected component analysis
- Position, scale, orientation of regions
- Temporal coherence

Tracking: Probabilistic framework

- Very general model:
  - We assume there are moving objects, which have an underlying state \( X \)
  - There are measurements \( Y \), some of which are functions of this state
  - There is a clock
    - at each tick, the state changes
    - at each tick, we get a new observation

Tracking State

\[
\begin{align*}
X_0 & \rightarrow X_1 & \cdots & \rightarrow X_{i-1} & \rightarrow X_i & \rightarrow X_{i+1} \\
Y_0 & \rightarrow Y_1 & \rightarrow Y_{i-1} & \rightarrow Y_i & \rightarrow Y_{i+1}
\end{align*}
\]

- Instead of "knowing state" at each instant, we treat the state as random variables \( X_t \) characterized by a pdf \( P(X_t) \) or perhaps conditioned on other Random Variables e.g., \( P(X_t | X_j, Y_j) \), etc.
- The observation (measurement) \( Y_t \) is a random variable conditioned on the state \( P(Y_t | X_t) \)
- Generally, we don’t observe the state – it’s hidden.

Simplifying Assumptions

- Only the immediate past matters: formally, we require
  \[
P(X_i | X_1, \ldots, X_{i-1}) = P(X_i | X_{i-1})
  \]

- Measurements depend only on the current state: we assume that \( Y_t \) is conditionally independent of all other measurements given \( X_t \). This means that
  \[
P(Y_1, \ldots, Y_{i-1}, Y_i | X_t) = P(Y_i | X_t) P(Y_1, \ldots, Y_{i-1} | X_t)
  \]

Three main steps

- Prediction: we have seen \( y_0, \ldots, y_{i-1} \) — what state does this set of measurements predict for the \( i \)th frame? to solve this problem, we need to obtain a representation of \( P(X_i | Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}) \).
- Data association: Some of the measurements obtained from the \( i \)th frame may tell us about the object’s state. Typically, we use \( P(X_i | Y_0 = y_0, \ldots, Y_{i-1} = y_{i-1}) \) to identify these measurements.
- Correction: now that we have \( y_i \) — the relevant measurements — we need to compute a representation of \( P(X_i | Y_0 = y_0, \ldots, Y_{i} = y_i) \).

Tracking as induction

- Assume data association is done
  - Sometimes challenging in cluttered scenes. See work by Christopher Rasmussen on Joint Probabilistic Data Association Filters (JPDAF).
- Do correction for the 0’th frame
- Assume we have corrected estimate for i’th frame
  - show we can do prediction for i+1 frame, correction for i+1 frame
Base case

$P(y \mid x)$ is our observation model. Given the state $x$, what is the observation. For example, $P(y \mid x)$ might be a Gaussian with mean $x$.

Firstly, we assume that we have $P(x_0)$.

$$P(x_0) = \frac{P(y_0 \mid x_0)P(x_0)}{P(y_0)}$$

$$= \frac{P(y_0 \mid x_0)P(x_0)}{\int P(y_0 \mid x_0)P(x_0)dx}$$

$$\propto P(y_0 \mid x_0)P(x_0)$$

Induction step

**Prediction**

Prediction involves representing

$$P(x_t \mid y_0, \ldots, y_{t-1})$$

Our independence assumptions make it possible to write

$$P(x_t \mid y_0, \ldots, y_{t-1}) = \int P(x_t \mid x_{t-1}, y_0, \ldots, y_{t-1})dx_{t-1}$$

$$= \int P(x_t \mid x_{t-1}, y_0, \ldots, y_{t-1})P(x_{t-1} \mid y_0, \ldots, y_{t-1})dx_{t-1}$$

$$= \int P(x_t \mid x_{t-1})P(x_{t-1} \mid y_0, \ldots, y_{t-1})dx_{t-1}$$

**Correction**

Correction involves obtaining a representation of

$$P(x_t \mid y_0, \ldots, y_t)$$

Our independence assumptions make it possible to write

$$P(x_t \mid y_0, \ldots, y_t) = \frac{P(x_t \mid y_0, \ldots, y_t)}{P(y_t \mid y_0, \ldots, y_t)}$$

$$= \int P(x_t \mid y_0, \ldots, y_t)P(x_t \mid y_0, \ldots, y_t)dx_t$$

$$= \int P(y_t \mid x_t)P(x_t \mid y_0, \ldots, y_t)dx_t$$