From Optical Flow to Tracking and on to Recognition

Introduction to Computer Vision
CSE 152
Lecture 16

• HW3 due on wed
Measuring Motion

Optical Flow Constraint Equation

Optical Flow

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

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

\[
\frac{dx}{dt} \frac{\partial I}{\partial x} + \frac{dy}{dt} \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0
\]
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 \( u = \frac{dx}{dt}, v = \frac{dy}{dt} \)
- One equation, two unknowns \( \rightarrow \) Can’t solve it

Aperture Effect

Which way did the center point move?
- Up and to the right?
- Down and to the right?
- Straight up?

This ambiguity is called the Aperture Effect
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 ≠ Motion Field

Motion field exists but no optical flow
No motion field but shading changes
Lucas-Kanade Motivation

- Optical flow cannot be estimated using information at a single pixel location because we only have one equation and two unknowns, called the aperture effect.
- Instead of computing flow using a single pixel, use the information from a patch around the pixel.
  - But, make an assumption: the flow for every point in the patch is the same.

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.

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

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

Rewrite as

\[
\begin{bmatrix}
\sum I_x^2 & \sum I_xI_y \\
\sum I_xI_y & \sum I_y^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix} = -\left( \frac{\sum I_xI_t}{\sum I_yI_t} \right)
\]

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

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

Let \( M = \sum (\nabla I)(\nabla I)^T \) and \( \mathbf{b} = [\sum I_xI_x, -\sum I_xI_y] \)

- So, the optical flow \( U = (u,v) \) can be written as
  \[ MU = \mathbf{b} \]

- And optical flow is just \( U = M^{-1}\mathbf{b} \)

Lucas-Kanade: Singularities & Aperture Problem

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

- \( M \) 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 textured 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

$M = \sum (\nabla I) (\nabla I)^T$

- 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 $\lambda_2$
Revisiting the small motion assumption

- Is this motion small enough?
  - Probably not—it’s much larger than one pixel (2nd 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
  \textit{(easier said than done)}
- Refine estimate by repeating the process
Pyramid / “Coarse-to-fine”

Coarse-to-fine optical flow estimation

Gaussian pyramid of image H
Gaussian pyramid of image I

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 $u_{i-1}, v_{i-1}$ 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 $f_i$ 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. $u_i = u_i^* + u_i'$, $v_i = v_i^* + v_i'$.
Visual Tracking

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

Main Challenges
1. 3-D Pose Variation
2. Occlusion of the target
3. Illumination variation
4. Camera jitter
5. Expression variation etc.

[ Ho, Lee, Kriegman ]

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 changed 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 $t$?
- **Data Association**: Which measurements correspond to which object?
- **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 + eye state (open/closed)
- Collections of control points specifying a spline
- Same as above, but for multiple objects (e.g. tracking a formation of airplanes)
- Augment above with temporal derivatives $(\phi, \dot{\phi})$

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 $I(u,v)$ at time $t$, create a binary image by applying a function $f(I(u,v))$.
- Clean up binary image using filters called morphological operators
- Perform connected component exploration to find “blobs” – connected regions.
- Compute moments of each blob (mean and covariance of coordinates of region) and use as the state
- Using state estimate from $t-1$, perform “data association” to identify state in time $t$

Blob Tracking in IR Images

- Threshold about body temperature
- Connected component analysis to find blobs
- Position, scale, orientation of blobs
- Temporal coherence
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 $P(X_i | Y_0 = y_0, \ldots, Y_i = y_i)$.

We can try to express these conditional distributions parametrically, sample the distribution, or estimate the mode.

Recognition

Given a database of objects and an image determine what, if any of the objects are present in the image.
Recognition

Given a database of objects and an image determine what, if any of the objects are present in the image.
Where are the coral heads and which ones are healthy and which are bleached?

Input Image

Segmented/labeled Image

How many visual object categories are there?

~10,000 to 30,000
Specific recognition tasks
Scene categorization

- outdoor/indoor
- city/forest/factory/etc.

Image annotation/tagging

- street
- people
- building
- mountain
- ...
Object detection

- find pedestrians

Image parsing/Semantic Segmentation

- sky
- mountain
- building
- tree
- banner
- street lamp
- market
- people
Object Recognition: The Problem

Given: A database $D$ of “known” objects and an image $I$:

1. Determine which (if any) objects in $D$ appear in $I$
2. Determine the pose (rotation and translation) of the object

WHAT AND WHERE!!!

Within-class variations
Recognition Challenges

• Within-class variability
  – Different objects within the class have different shapes, materials, or color patterns
  – Deformable
  – Articulated
• Pose variability:
  – 2-D Image transformation (translation, rotation, scale)
  – 3-D Pose Variability and perspective distortion
• Lighting
  – Direction, number of sources
  – Color
  – Shadows
• Occlusion – partial occlusion, self occlusion
• Clutter in background -> false positives

Sketch of a Pattern Recognition Architecture
Sliding window approach to face detection

Example: Face Detection

- Scan window over image.
- Classify window as either:
  - Face
  - Non-face

See for example, Viola-Jones face detector in OpenCV
Evaluating a binary classifier

- For a detector, there are two types of errors:
  - False Positives, False accept (e.g., non-face is detected as a face)
  - False Negatives, False Reject (e.g., face is missed)
- ROC Curve (Receiver Operator Characteristic)-Plot of tradeoff between False Positives and false negatives

\[
\text{Precision} = \frac{tp}{tp + fp} \\
\text{Recall} = \frac{tp}{tp + fn}
\]

• See also definitions of precision and recall
• https://en.wikipedia.org/wiki/Precision_and_recall

Evaluating Multi-class classifiers

CIFAR 10
60,000 32x32 color images in 10 classes

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

CSE152, Spring 2019
Intro Computer Vision
Evaluating Multi-class classifiers

- Overall accuracy
- Confusion Matrix – Example from Coral Reef Classification

<table>
<thead>
<tr>
<th>Ground Truth</th>
<th>CCA</th>
<th>Turf</th>
<th>Macro</th>
<th>Sand</th>
<th>Acrop</th>
<th>Pavon</th>
<th>Monti</th>
<th>Pocill</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCA</td>
<td>.89</td>
<td>.04</td>
<td>.01</td>
<td>.02</td>
<td>.02</td>
<td>.95</td>
<td>.01</td>
<td>.01</td>
<td>.02</td>
</tr>
<tr>
<td>Turf</td>
<td>.40</td>
<td>.46</td>
<td>.01</td>
<td>.03</td>
<td>.06</td>
<td>1.0</td>
<td>.01</td>
<td>.01</td>
<td>.03</td>
</tr>
<tr>
<td>Macro</td>
<td>.69</td>
<td>.05</td>
<td>.19</td>
<td>.02</td>
<td>.02</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.02</td>
</tr>
<tr>
<td>Sand</td>
<td>.15</td>
<td>.01</td>
<td>.83</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>Acrop</td>
<td>.13</td>
<td>.14</td>
<td>.01</td>
<td>.02</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>Pavon</td>
<td>.25</td>
<td>.06</td>
<td>.01</td>
<td>.03</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>Monti</td>
<td>.41</td>
<td>.03</td>
<td>.01</td>
<td>.02</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>Pocill</td>
<td>.34</td>
<td>.03</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
<tr>
<td>Port</td>
<td>.20</td>
<td>.02</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
<td>.01</td>
</tr>
</tbody>
</table>

2008, 2009 ⇒ 2010 (83.1%)

CCA Turf Macro Sand Acrop Pavon Monti Pocill Port

Ground Truth

Estimated

CSE152, Spring 2019
Intro Computer Vision