Computer Vision for HCI

Motion

Motion

• Changing scene may be observed in a sequence of images
• Changing pixels in image sequence provide important features for object detection and activity recognition
General Cases of Motion

- Still camera, single moving object, constant background
  - Simplest case, motion sensors for security
- Still camera, several moving objects, constant background
  - Tracking, multi-person event analysis
- Moving camera, relatively constant scene
  - Egomotion, panning to provide wider panoramic view
- Moving camera, several moving objects
  - Most difficult, robot navigating through heavy traffic

Image Differencing

- Simplest means of motion detection
- Detect absolute value of difference between frame t-1 and t
  - Threshold result
- Only motion “presence”, no direction

Frame t-1 | Frame t | Difference image
Single, Constant Threshold?

- Image differencing:

\[ \Delta I = \begin{cases} 
1 & \mid I_t - I_{t-1} \mid \geq \tau \\
0 & \text{else}
\end{cases} \]

- Appropriate threshold perhaps not a constant value
  - Set too low and noise will appear in the difference image
  - Set too high and will not work in all situations

Perception: Weber’s Law

- **Weber’s Law**: For humans, the ratio of just noticeable difference (\(\delta L\)) and luminance (\(L\)) is constant (i.e., \(\delta L = kL\))
  - The greater the luminance, the more luminance change that is needed to perceive a change (or, can see smaller luminance changes in lower luminance areas)
  - Better/Other models exist but Weber’s Law suffices as approximation

Image Difference Algorithm:

Compute Difference Image \(\delta I = \mid I_t - I_{t-1} \mid\)

Smooth \(I_{t-1}\) with Gaussian kernel to receive \(\tilde{I}\)

Apply Weber’s Law \(\Delta I = \begin{cases} 
1 & \delta I \geq k\tilde{I} \\
0 & \text{else}
\end{cases} \)
Optic Flow

- Image sequence \( f(x, y, t) \)
- Assume with small motion change (of patch), no change in gray levels (brightness constancy constraint) \( f(x, y, t) = f(x + dx, y + dy, t + dt) \)

\[
\begin{align*}
\partial f \over \partial x & \quad dx + \partial f \over \partial y \quad dy + \partial f \over \partial t \quad dt \\
\text{Divide by } dt & \\
\partial f \over \partial x \quad dx + \partial f \over \partial y \quad dy + \partial f \over \partial t \quad dt = 0
\end{align*}
\]
Optic Flow

\[
\begin{bmatrix}
\frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} & \frac{\partial f}{\partial t}
\end{bmatrix} = 0
\]

Gradients can be computed from images (keep proper gradient normalization factor!!!)

\[
\frac{\partial f}{\partial x} = f_x \Rightarrow \begin{bmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{bmatrix} / 8
\]

\[
\frac{\partial f}{\partial y} = f_y \Rightarrow \begin{bmatrix}
-1 & -2 & -1 \\
0 & 0 & 0 \\
1 & 2 & 1
\end{bmatrix} / 8
\]

Compute in \( I_t \)

\[
\frac{\partial f}{\partial t} = f_t \Rightarrow F(I_t - I_{t-1})
\]

e.g., \( F(.) \) could be take 3x3 smoothing filter applied to each image before taking the diff at center pixel locations

Optic Flow

****For motion from \( I_{t-1} \) to \( I_t \)

\[
f_x = [0(1) + 0(0) + 1(1) + 0(-2) + 1(0) + 1(2) + 1(-1) + 1(0) + 1(1)] / 8 = 3/8
\]

\[
f_y = [0(1) + 0(-2) + 1(-1) + 0(0) + 1(0) + 1(0) + 1(1) + 1(2) + 1(1)] / 8 = 3/8
\]

\[
f_t = F(I_t - I_{t-1})
\]

or

\[
f_t = F(smooth(I_t) - smooth(I_{t-1}))
\]
Optic Flow

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

or

\[
f_u + f_v + f_t = 0
\]

Goal is to compute image motion

But only one equation for two unknowns!

Aggregate Optic Flow

One solution: Over-constrain by checking over small “patch” of pixels

\[
\begin{align*}
\frac{dx}{dt} &= u \\
\frac{dy}{dt} &= v
\end{align*}
\]

\[
E = \sum_i (f_{xi}u + f_{yi}v + f_{ti})^2
\]

\[
\frac{\partial E}{\partial u} = \sum_i (f_{xi}^2 u + f_{xi}f_{yi}v + f_{xi}f_t) = 0
\]

\[
\frac{\partial E}{\partial v} = \sum_i (f_{yi}f_{xi}u + f_{yi}^2 v + f_{yi}f_t) = 0
\]

\[
\begin{bmatrix}
\sum_i f_{xi}^2 & \sum_i f_{xi}f_{yi} \\
\sum_i f_{yi}f_{xi} & \sum_i f_{yi}^2
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix} = -\begin{bmatrix}
\sum_i f_{xi}f_{ti} \\
\sum_i f_{yi}f_{ti}
\end{bmatrix}
\]

Perform least squares to solve
"Can I do anything with the equation itself?"

\[ f_x u + f_y v + f_t = 0 \]

Consider Normal Form of Straight Line Equation:

\[
\frac{A}{\sqrt{A^2 + B^2}} x + \frac{B}{\sqrt{A^2 + B^2}} y + \frac{C}{\sqrt{A^2 + B^2}} = 0
\]

\[
\begin{pmatrix}
A \\
B \\
-C \\
\end{pmatrix}
\sqrt{A^2 + B^2}
\]

\[
\begin{pmatrix}
\frac{A}{\sqrt{A^2 + B^2}} \\
\frac{B}{\sqrt{A^2 + B^2}} \\
\end{pmatrix}
\]

"Normal Optic Flow"

\[ f_x u + f_y v + f_t = 0 \]

\[
\frac{f_x}{\sqrt{f_x^2 + f_y^2}} u + \frac{f_y}{\sqrt{f_x^2 + f_y^2}} v + \frac{f_t}{\sqrt{f_x^2 + f_y^2}} = 0
\]

Can determine only one component of flow:

Magnitude of flow in the direction of the brightness gradient (perpendicular to brightness contour)

"Normal Flow"
Aperture Problem
Barber Pole Illusion

Weighted Aggregate Normal Flow

- Weight the computation of the normal flow magnitude by the strength of gradients in local region (smoothes result)

\[ NF_{mag} = \frac{-f_i}{\sqrt{f_x^2 + f_y^2}} \]

\[ wNF_{mag} = \frac{-\sum_l (\sqrt{f_{x_l}^2 + f_{y_l}^2}) \cdot f_{il}}{\sum_l (\sqrt{f_{x_l}^2 + f_{y_l}^2}) \cdot \sqrt{f_{x_l}^2 + f_{y_l}^2}} = \frac{-\sum_l (\sqrt{f_{x_l}^2 + f_{y_l}^2}) \cdot f_{il}}{\sum_l (\sqrt{f_{x_l}^2 + f_{y_l}^2})^2} \]
Normal Flow vs. Desired Optic Flow for Moving Box

Motion
Hierarchical Motion Estimation
Optical Flow

- Assumptions for computation of optical flow: \( f_x u + f_y v + f_t = 0 \)
  - Small motion change (of patch)
  - No change in gray levels after moving (brightness constancy constraint)
- Not entirely realistic for real-world motions in video
  - Usually larger motion (even at 30+ frames/sec)

Hierarchical Motion Estimation

- Bergen J., and Hingorani R. “Hierarchical Motion-Based Frame Rate Conversion”, 1990
- Key feature of framework
  - Coarse-to-fine refinement strategy
    - Local model (used in estimation process)
    - Global model (constrains overall structure of motion)
Hierarchical Motion Estimation

- Key element is use of multi-resolution image representation to allow optical flow constraint equation even when motion is fairly large.

Displacement Reduction in Pyramid

Can compute optical flow at this level

Δx = 1 pixel
Δx = 2 pixels
Δx = 4 pixels
Δx = 8 pixels

Gaussian pyramid of image \( I_{t-1} \)
Gaussian pyramid of image \( I_t \)
Smoothness

- Pyramid construction smoothens out discontinuities to provide better gradient-based estimations
  - Smooths over boundaries, but dealt with in later estimations

Displacement Reduction in Pyramid

- Can compute optical flow at top level of pyramid but …
  - Low image resolution (reduced size), so not true answer for original sized image
  - Motion may start at lower (larger) levels
- Provides result for initial estimate (to be refined)
Hierarchical Motion Estimation

• Overview of approach for pair of images \((I_{t-1}, I_t)\)
  – Generate multi-resolution Gaussian pyramid of images
    • Iteratively reduce image
  – Perform local estimation of displacements
    • Compute optical flow
    • Start at smallest pyramid level
  – Expand flow and warp image at next level, then compute local flow estimation
    • Compensates for previously estimated displacements
  – Iteratively refine global optical flow

Course-to-Fine Motion Estimation
Computing Optic Flow

Aggregate flow method

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

\[
E = \sum_i (f_{xi}u + f_{yi}v + f_{ti})^2
\]

\[
\frac{\partial E}{\partial u} = \sum_i (f_{xi}^2u + f_{xi}f_{yi}v + f_{xi}f_{ti}) = 0
\]

\[
\frac{\partial E}{\partial v} = \sum_i (f_{yi}f_{xi}u + f_{yi}^2v + f_{yi}f_{ti}) = 0
\]

\[
\begin{bmatrix}
\sum_i f_{xi}^2 & \sum_i f_{xi}f_{yi} \\
\sum_i f_{yi}f_{xi} & \sum_i f_{yi}^2
\end{bmatrix}
\begin{bmatrix}
u_x \\ v_y
\end{bmatrix}
= -
\begin{bmatrix}
\sum_i f_{xi}f_{ti} \\
\sum_i f_{yi}f_{ti}
\end{bmatrix}
\]

Least squares

Expanding & Scaling Motion

- The flow \((u,v)\) is expanded to match the size of the next (bigger) pyramid image
  - Using 2x spatial expand operations (see upsampling in pyramid lecture)
  - Also **must expand the flow magnitude**, as the vectors are in pixel units (and image was just doubled in size)

\[
u' = \text{expand}(u) \times 2
\]

\[
v' = \text{expand}(v) \times 2
\]
Image Warping

• Given image $I_{t-1}$ at next (larger) pyramid level and the corresponding $(u', v')$ motion estimate for that level, use motion vectors to warp $I_{t-1}$ into $W$

• $W$ and the corresponding level $I_t$ should appear to “be close”
  – Enough to compute valid optical flow

Matlab

```matlab
function [warpIm]=warp(Im, u, v)

[M, N]=size(Im);

[x, y]=meshgrid(1:N,1:M);

warpIm=interp2(x, y, Im, x-u, y-v);

% Matlab returns a NaN in places outside the first image
I=find(isnan(warpIm));
warpIm(I)=zeros(size(I));
```
Updating Motion

• Compute new, local motion flow at this level
  – From $W$ to $I_t$
• Refine previous estimate (that was expanded/scaled and used from previous level for $W$)
  \[
  u = u' + \text{local}_u \\
  v = v' + \text{local}_v
  \]

Other 2-D Motion Approaches

• Many, many, many other approaches
  – Add smoothness terms to error function
  – Robust statistics
  – …
Motion

BONUS MATERIAL:
3-D Motion with Images and Model

3-D Motion

• Infer 3-D motion of object from 2-D image properties and 3-D model of object
• Instantaneous method
  – Optical flow used to recover 3-D motion and depth values
3-D Motion of Point

\[ \tilde{V} = \tilde{T} + \tilde{\omega} \times \tilde{r} \]

\[ = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} + \begin{bmatrix} i & j & k \\ \omega_x & \omega_y & \omega_z \\ X & Y & Z \end{bmatrix} \]

\[ = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \begin{bmatrix} \omega_Z - Y \omega_z \\ X \omega_x - \omega_Z \\ \omega Y - X \omega_x \end{bmatrix} \]

\[ \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} T_x + \omega Y - X \omega_x \\ T_y + X \omega_x - \omega Z \\ T_z + \omega Z - Y \omega_z \end{bmatrix} \]

Pinhole Camera Model

- Rays connect image plane to object through pinhole
  - Object is imaged upside-down on image plane

\[ \frac{X}{Z} = \frac{x}{f} \rightarrow x = \frac{fX}{Z} \]

\[ \frac{Y}{Z} = \frac{y}{f} \rightarrow y = \frac{fY}{Z} \]
2-D Perspective Motion

2-D perspective motion (camera focal length $F$, “pinhole camera”)

\[
\frac{dx}{dt} = \frac{d}{dt} \left( \frac{FX}{Z} \right) = \frac{dX}{dt} \frac{F}{Z} - \frac{FX}{Z^2} \frac{dZ}{dt} = \frac{F}{Z} \left( \frac{dX}{dt} - \frac{X}{Z} \frac{dZ}{dt} \right) = \frac{F}{Z} \left( V_x - \frac{X}{Z} V_z \right)
\]

\[
\frac{dy}{dt} = \frac{d}{dt} \left( \frac{FY}{Z} \right) = \frac{dY}{dt} \frac{F}{Z} - \frac{FY}{Z^2} \frac{dZ}{dt} = \frac{F}{Z} \left( \frac{dY}{dt} - \frac{Y}{Z} \frac{dZ}{dt} \right) = \frac{F}{Z} \left( V_y - \frac{Y}{Z} V_z \right)
\]

Substitute $V_x, V_y, V_z$ from previous slide

\[
\frac{dx}{dt} = \frac{F}{Z} \left( V_x - \frac{X}{Z} V_z \right) = \frac{F}{Z} \left( \left[ T_x + \omega_z Z - Y \omega_y \right] - \frac{X}{Z} \left[ T_z + \omega_y Y - X \omega_x \right] \right)
\]

\[
\frac{dy}{dt} = \frac{F}{Z} \left( V_y - \frac{Y}{Z} V_z \right) = \frac{F}{Z} \left( \left[ T_y + X \omega_z - \omega_y Z \right] - \frac{Y}{Z} \left[ T_z + \omega_y Y - X \omega_x \right] \right)
\]

Optical Flow Constraint

Recall optical flow constraint equation

\[
f_x \frac{dx}{dt} + f_y \frac{dy}{dt} + f_i = 0
\]

Substitute derivatives from previous slide

\[
-f_i = f_x \frac{F}{Z} \left( \left[ T_x + \omega_z Z - Y \omega_y \right] - \frac{X}{Z} \left[ T_z + \omega_y Y - X \omega_x \right] \right)
\]

\[
+ f_y \frac{F}{Z} \left( \left[ T_y + X \omega_z - \omega_y Z \right] - \frac{Y}{Z} \left[ T_z + \omega_y Y - X \omega_x \right] \right)
\]
3-D Motion Parameters

\[-f_t = \left[ f_x \frac{F}{Z} \right] T_x + \left[ f_y \frac{F}{Z} \right] T_y - \left[ \frac{F}{Z^2} (f_z X + f_y Y) \right] T_z - \left[ \frac{F}{Z^2} (f_x X + f_y Z^2 + f_y^2) \right] \omega_x + \left[ \frac{F}{Z} (f_x Z^2 + f_y X^2 + f_y YX) \right] \omega_y - \left[ \frac{F}{Z} (f_y Y - f_y X) \right] \omega_z \]

\((X,Y,Z)\) points are known from 3-D model of object

\(f_x, f_y, f_t\) can be computed from images (projected locations)

Using least squares, compute \textit{translations} and \textit{rotations} (rigid object)

\[\]
Motion Templates

- From blurred video, easily recognize the activity
- Recognize holistic “patterns of motion”
  - No tracking of structural features (hands, elbows)

Representation Theory

- Decompose motion
- The “where”
  - Spatial pattern of “where” motion occurred
  - Motion energy image (MEI)
- The “how”
  - Progression of “how” the motion is moving
  - Motion history image (MHI)
Temporal Template

- MHI (and MEI) is a static image
  - Value at each pixel is some function of the motion at that pixel
  - MHI: pixel records temporal history
  - MEI: pixel records presence of motion
- TT = [MEI, MHI]

Cumulative Motion Images

- Cumulative motion presence
  - Image differencing
- Sweeps out particular region
- Shape can be used to suggest action and view
Across View Angle

Motion Energy Image (MEI)

- Cumulative motion images
  
  \[ E_{\tau}(x, y, t) = \bigcup_{i=0}^{\tau-1} D(x, y, t - i) \]

- Duration \( \tau \) defines temporal extent
Motion History Image (MHI)

- Represent “how” motion is moving
- Pixel intensity is function of temporal history at that point
- Simple replace-and-decay operator with timestamp $\tau$

$$MHI_{\delta}(x, y) = \begin{cases} 
\tau & \text{if } \Psi(I(x, y)) \neq 0 \\
0 & \text{else if } MHI_{\delta}(x, y) < \tau - \delta 
\end{cases}$$

Note: later will normalize MHI to values (0-1) for matching.

Silhouette Differences

Changing delay $\delta$

$\delta = 0.25$  $\delta = 0.5$  $\delta = 1$
vs. Image Differencing

Matching Templates

- First normalize template to (0-1)
  - See next slide!
- Invariant to viewing condition?
  - Want scale and translation invariance
- Feature vector of 7 Similitude moments
  - Vector for MEI
  - Vector for MHI
- Mahalanobis distance (squared) metric to models
  \[ MD = (x - m)^T K^{-1} (x - m) \]
MHI (Re)Normalization

Want “fade to black” for duration used

For any pixel having motion that occurred at time $t$:

$$\max\left(0, \frac{t - \lfloor \min(t) - \Delta t \rfloor}{\max(t) - \lfloor \min(t) - \Delta t \rfloor}\right)$$

$$\max(0, \frac{\text{frame} \# - \lfloor \min(\text{frame} \#) - 1 \rfloor}{\max(\text{frame} \#) - \lfloor \min(\text{frame} \#) - 1 \rfloor})$$

Aerobics Data

Duration

- Long
- Medium
- Short

The oldest frame # that still want to keep/display

The most current frame # (most recent)
MEIs and MHIs

Confusion Difficulties

Test input: Move 13 at 30°

Closest match: Move 6 at 0°

Correct match
Subject Variances and Motion Calculation

Test input: Move 16 at 30°
Closest match: Move 15 at 0°
Correct match

Bad motion calculation in test input, and different subject performances

Temporal Segmentation

- Maximum and minimum duration of actions
  \[ \tau_{\text{min}} - \tau_{\text{max}} \]
- Backward searching time window
  - Compute \( MHI_{r_{\text{max}}} (x, y, t) \)
  - Normalize (0-1), match to models
  - Compute \( MHI_{r-\Delta r} (x, y, t) \)
    - Just threshold previous version! (removes older parts)
  - Normalize (0-1), Match to models
  - Keep going until \( \tau - \Delta \tau = \tau_{\text{min}} \)
Motion Gradients

- Can perceive “direction of motion” in intensity fading of motion template
  - Downward, backward swipe
- Convolve gradient masks with template
  - Intensity gradient similar to motion flow (normal flow)

\[
F_x = \begin{bmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{bmatrix}
\]

\[
F_y = \begin{bmatrix}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & 2 & -1 \\
\end{bmatrix}
\]
Demonstration

Motion Orientation Histograms
Effect of Occlusion

Summary

- Changing pixels in image sequence provide important features for motion analysis
- Optic flow
  - Brightness constancy constraint
  - Under-constrained and aperture problem
  - Aggregate optic flow
- Normal flow
  - Magnitude of flow in the direction of the brightness gradient (perpendicular to brightness contour)
Summary (con’t)

• Multi-resolution motion estimation
  – Coarse-to-fine refinement strategy
• Permits use of optical flow constraint equation even when motion is fairly large
• Key steps
  – Generate multi-resolution Gaussian pyramid of images
  – Perform local estimation of motion displacements
  – Expand flow and warp image at next level, then compute local flow estimation
  – Update global motion estimation
• 3-D motion from images (and model)

Summary (con’t)

• Motion Templates
  – MEI and MHI
  – Temporal accumulator and decay operator
• Recognition I
  – Compute and match first seven scale- and translation-invariant moments from entire MHI (and MEI)
    • Also from a binary MHI
• Motion Gradients
  – Direction of motion from intensity gradient