Optical Flow and On to Recognition

Introduction to Computer Vision
CSE 152
Lecture 17

Announcements

• Assignment 4: Due Thursday
• Assignment 5: To be posted on Thursday
• Read: Trucco & Verri, Chapter 8 on Motion
• Final Exam: Wed, 6/9/04, 11:30-2:30, WLH 2207 (here)

Virtual Cinematography: Making 'The Matrix' Sequels
George Borshukov
VFX Technology Supervisor, ESC Entertainment
Friday, June 4, 2004
1:00 p.m. to 2:30 p.m.

The presentation will cover the key technologies that had to be developed and deployed to create the synthetic human sequences in the Matrix sequels including Universal Capture - image-based facial animation, realistic human face rendering, and use of measured BRDF in film production. It will also feature a breakdown of The Superpunch shot (pictured above) from "The Matrix Revolutions" (the bullet time punch that Neo delivers to Agent Smith during the film's last face-off). This difficult, important, expensive, and challenging shot was entirely computer generated and showcased the technological developments of 3.5+ years at their best by showing a full-frame close-up of a known human actor.

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, \left| \frac{\partial I}{\partial t} \right| is large and is cue for segmentation
• Doesn’t tell which way stuff is moving

Optical Flow:
Where do pixels move to?
The Motion Field

Rigid Motion: General Case

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

Position and orientation of a rigid body
Rotation Matrix & Translation vector
Angular Velocity Vector: \( \omega \) (or \( \Omega \))

Motion Field Equation

\[
\begin{align*}
\dot{u} &= \frac{T_u - T_z f}{Z} - \omega_z + \omega_{uv} - \frac{\omega_u u^2}{f} \\
\dot{v} &= \frac{T_v - T_z f}{Z} + \omega_z - \omega_{uv} - \frac{\omega_v v^2}{f}
\end{align*}
\]

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

Pure Translation

\[
\begin{align*}
\dot{u} &= \frac{T_u - T_z f}{Z} - \omega_z + \omega_{uv} - \frac{\omega_u u^2}{f} \\
\dot{v} &= 0
\end{align*}
\]

\[ \omega = 0 \]

Pure Rotation: \( T=0 \)

\[
\begin{align*}
\dot{u} &= \frac{S_u - T_z f}{Z} - \omega_z + \omega_{uv} - \frac{\omega_u u^2}{f} \\
\dot{v} &= \frac{S_v - T_z f}{Z} + \omega_z - \omega_{uv} - \frac{\omega_v v^2}{f}
\end{align*}
\]

- 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

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

Motion Field Equation: Estimate Depth

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

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 \((du/dt, dv/dt)\) at \((u,v)\).

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)

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

Mathematical formulation

[Note change of notation: image coordinates now \((x,y)\), not \((u,v)\)]

\( I(x,y,t) \) = brightness at image point \((x,y)\) at time \( t \)

Consider scene (or camera) to be moving, so \( x(t), y(t) \)

**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:**

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

Solving for flow

**Optical flow constraint equation:**

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

- We can measure \( \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t} \)
- We want to solve for \( \frac{dx}{dt}, \frac{dy}{dt} \)
- One equation, two unknowns
**Aperture Problem and Normal Flow**

 Measurements

\[ I_x = \frac{\partial I}{\partial x} \]

\[ I_y = \frac{\partial I}{\partial y} \]

\[ I_t = \frac{\partial I}{\partial t} \]

Flow vector

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

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

Normal Flow:

\[ u_x = -\frac{I_x}{\sqrt{I_x^2 + I_y^2}} \]

\[ v_y = \frac{I_y}{\sqrt{I_x^2 + I_y^2}} \]

The gradient constraint:

\[ I_x u + I_y v + I_t = 0 \]

\[ \nabla I \cdot \vec{U} = 0 \]

Defines a line in the \((x, y)\) space

---

**What is the correspondence of \(P\) & \(P'\)**

Contour plots of image intensity in two images

---

**Normal Flow**

Illusion Works Barber Pole Illusion

---

**Two ways to get flow**

1. Think globally, and regularize over image
2. Look over window and assume constant motion in the window

---

**Lucas-Kanade: Integrate over a Patch**

Assume a single velocity for all pixels within an image patch

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

\[ \frac{dE(u, v)}{du} = \sum_{(x, y) \in \Omega} 2I_x (I_x u + I_y v + I_t) = 0 \]

\[ \frac{dE(u, v)}{dv} = \sum_{(x, y) \in \Omega} 2I_y (I_x u + I_y v + I_t) = 0 \]

Solve with:

\[ \left( \sum I_x^2 \sum I_x I_y \right) u = -\left( \sum I_y I_t \right) \]

\[ \left( \sum I_x I_t \right) v \]

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

\[ \sum \nabla I(1) \cdot \nabla I(2) = -\nabla I \cdot H \]
Lucas-Kanade: Singularities and the Aperture Problem

Let \( M = \sum (\nabla I | \nabla I')' \) and \( b = \left[ \frac{\sum I_x}{\sum I} \right] \)

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

- \( M \) is singular if all gradient vectors point in the same direction
  - e.g., along an edge
  - of course, trivially singular if the summation is over a single pixel
  - i.e., only normal flow is available (aperture problem)

- Corners and textured areas are OK
- \( M \) is zero matrix in constant intensity region

Recognition

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

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!!!
Recognition Challenges

- **Within-class variability**: Different objects within the class have different shapes or different material characteristics
  - Deformable
  - Articulated
  - Compositional
- **Pose variability**:
  - 2-D Image transformation (translation, rotation, scale)
  - 3-D Pose Variability (perspective, orthographic projection)
- **Lighting**
  - Direction (multiple sources & type)
  - Color
  - Shadows
- **Occlusion** – partial
- **Clutter in background** -> false positives