CS664 Lecture #18: Motion

Announcements

- Most paper choices were fine
  - Please be sure to email me for approval, if you haven't already
  - This is intended to help you, especially with the final project
  - Use the computer vision bibliography
  - Some useful papers now on the web site
- Next quiz a week from today
  - Thursday 11/3, coverage through today
- PS#2 due November 8

Next Topic: Motion

- Motion and stereo are closely related
  - "Correspondence problem", or "visual correspondence": what went where?
  - Can be solved sparsely or densely
    - Except for wide-baseline stereo, in both cases sparse solutions are now rare
- Comparison:
  - Stereo uses a 1D label set, motion has 2D
  - Stereo involves bigger changes in appearance
  - Motion has an elegant formulation as a continuous problem

Some cool motion illusions

- There are lots of examples that show the complexity of human motion perception
  - Examples: http://www.cogsci.uci.edu/~ddhoff/Applets/index.html
    - Especially cool: pulsating square, moving disk

Continuous view of motion

- Our “data cost” will be the difference between the old pixel intensity and the new pixel intensity
  - New and old pixel are related by the motion
- Instead of search, we can directly solve for a data cost
  - We consider a series of images as samples of a function I(x,y,t)
    - What are we assuming?
- No explicit search over (many) labels

Constant brightness assumption

- Constant brightness assumption: $\frac{dI}{dt} = 0$
  - Expand out via chain rule:

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

  - By convention we will write:

$$u(x,y) = \frac{dx}{dt}, \quad v(x,y) = \frac{dy}{dt}$$

Example

- $I(x,y) \approx J(x+u(x,y),\ y+v(x,y))$

1	2	3	4
5	6	7	8
9	10	11	12
15	16	13	14

I

5	6	7	8
1	2	3	4
12	11	10	9
13	14	15	16

J

0	0	0	0
0	0	0	0
3	1	-1	-3
2	2	-2	-2

u

1	1	1	1
-1	-1	-1	-1
0	0	0	0
0	0	0	0

v

Optical Flow Constraint Equation

- We can measure the partial derivatives with respect to space and time, we are looking for u and v. Chain rule says:

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

- Equivalently,

$$\nabla I \cdot [u \; v] = -\frac{\partial I}{\partial t}$$

  - Where ∇I is the intensity gradient

OFCE consequences

- Only measure the projection of the true motion (u,v) on the intensity gradient ∇I
  - Hence, ambiguous
- This is the famous **aperture problem**
- How do we solve it?
  - Use neighboring pixels!

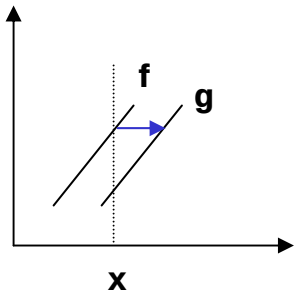
Gradient Constraint

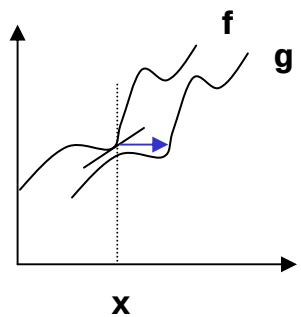
- One-dimensional example – linearization
  - Estimate displacement d using derivative
    - Two functions f(x) and g(x)=f(x-d)
  - Taylor series expansion

    $f(x-d) = f(x) - d\, f'(x) + E$
    - Where f' denotes derivative
  - Now write difference as

    $f(x)-g(x) = d\, f'(x) + E$
  - Neglecting higher order terms

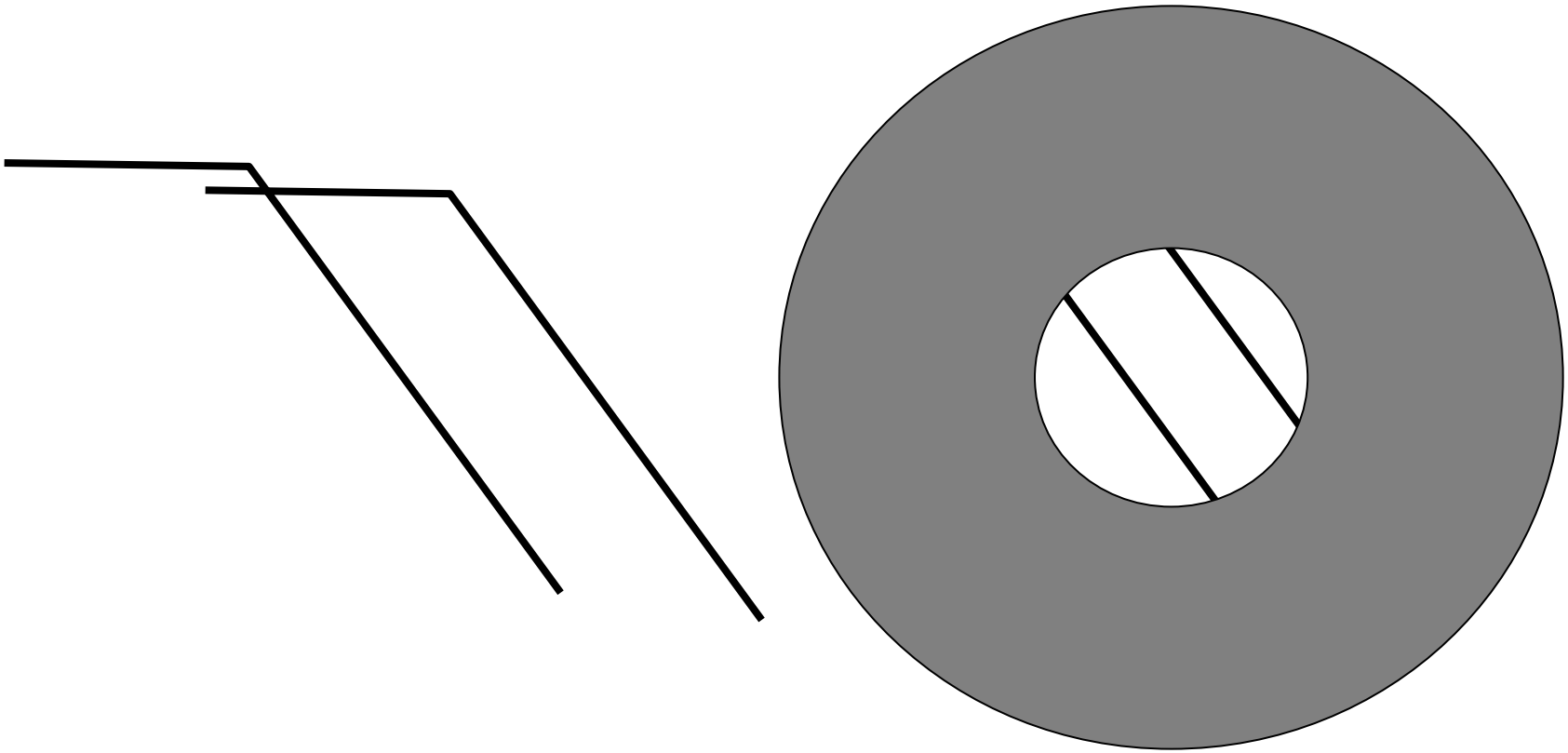
    $\delta = (f(x)-g(x))/f'(x)$
  - Note only for small d





Aperture Problem (Normal Flow)

- Can only measure motion in direction normal to edge (along gradient)

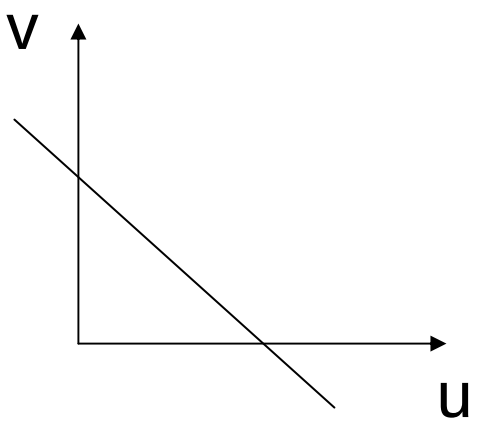


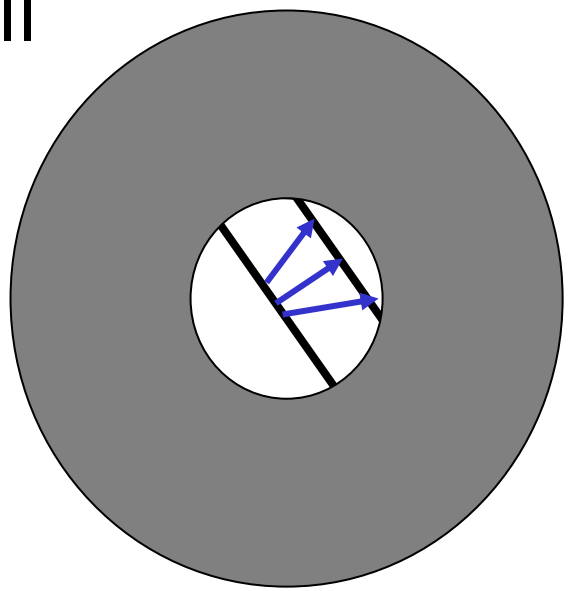
Aperture Problem (Normal Flow)

- Gradient constraint defines line in (u,v) space

$$\nabla I \cdot (\mathbf{u}, \mathbf{v}) \approx -I_t$$

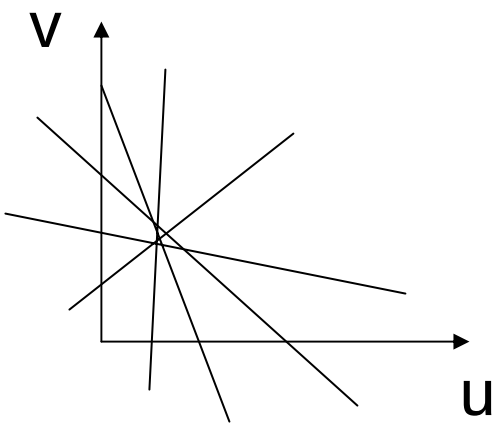
- Methods based solely on per pixel estimates don't work well

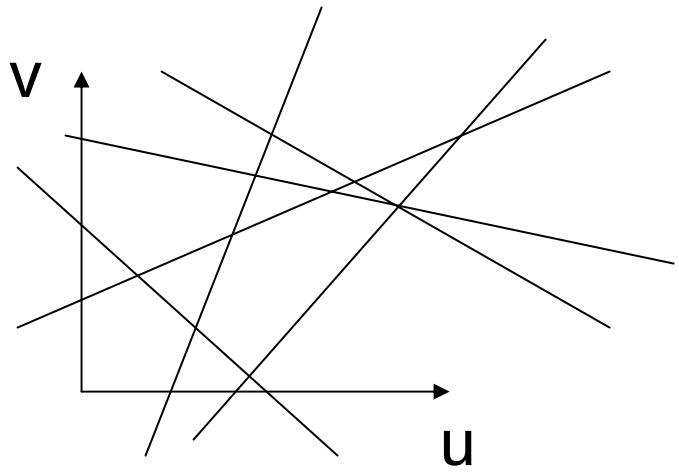




Combining Local Constraints

- Each pixel defines linear constraint on possible (u,v) displacement
  - For set of pixels with same displacement combine constraints to get estimate
  - For pixels with different displacements, somehow identify that is case





Lucas-Kanade

- Suppose that there is a single translational motion (u,v)
  - In a window, or over the entire image
- We can use least squares to solve this
- At each pixel, the OFCE says:

$$I_x(x_i, y_i) \cdot u + I_y(x_i, y_i) \cdot v = -I_t(x_i, y_i)$$

  - Here, ∇ I = $\langle I_x, I_y \rangle$ is the spatial gradient, and I_t is the temporal
  - These are the observations

Matrix form

- Write the set of these equations as

$$\begin{bmatrix} I_x(x_1,y_1) & I_y(x_1,y_1) \\ I_x(x_2,y_2) & I_y(x_2,y_2) \\ \vdots & \vdots \\ I_x(x_n,y_n) & I_y(x_n,y_n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -I_t(x_1,y_1) \\ -I_t(x_2,y_2) \\ \vdots \\ -I_t(x_n,y_n) \end{bmatrix}$$

$$S \cdot \begin{bmatrix} u \\ v \end{bmatrix} = -T$$

- With unrealistic assumptions, there will be a (u,v) that makes this equality true
  - What assumptions are these?

Least squares solution

- Multiplying by S^t we get the normal equations:

$$S^t S \cdot \begin{bmatrix} u \\ v \end{bmatrix} = -S^t T$$

$$S^t S = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \qquad S^t T = \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

  - You can easily solve this (inverting a 2x2 matrix has a simple closed form)

Easy and hard cases

- If the spatial gradients are parallel, or close to parallel, the matrix is singular
  - Multiple copies of the same equation obviously do not help you…
  - What does this mean in terms of the image?
    - Low texture is hard!
- The other extreme is when the nearby spatial gradients are orthogonal
  - I.e., we are near a corner
  - The matrix S^tS is also used for corner detection (KLT algorithm)

Application: focus of expansion

- From motion estimates you can compute the focus of expansion
  - Direction of heading, relative to camera
  - Compute by intersecting motion rays
  - Solve by some form of clustering
    - Hough transform voting scheme
- Useful for robotic applications
- Can also detect “looming”

Global motion methods

- Lucas-Kanade is a standard way to register images (i.e., for panoramas)
  - Almost always will need to deal with outliers
  - Good application of IRLS
    - Find LS fit
    - Pixels with high residuals violate the OFCE
    - Reduce their weight, solve again
- Can also estimate motion vectors that are parameterized over the image
  - Each vector fits some low-order model of how vectors change

Affine registration

- An affine motion maps between two arbitrary triangles
  - 6 parameters:

$$u(x,y) = a_1 + a_2 x + a_3 y$$

$$v(x,y) = a_4 + a_5 x + a_6 y$$

  - Note that translational motion is the special case where $a_2 = a_3 = a_5 = a_6 = 0$
  - Plugging this into OFCE we get:

$$I_x(x_i,y_i)\cdot(a_1 + a_2 x + a_3 y) + I_y(x_i,y_i)\cdot(a_4 + a_5 x + a_6 y) = -I_t(x_i,y_i)$$

Solving for affine motion

- More complex optimization problem
  - We need to find 6 parameters instead of 2
  - Can still use LS or IRLS
- This is what Birchfield & Tomasi do in their slanted surfaces paper
  - A triangle on the plane in the scene is mapped to a triangle on each image
  - You can view planes as affine transformations
    - Fairly reasonable model
  - No need for IRLS (why not?)