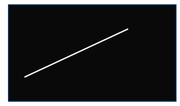
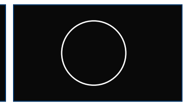
Mid Term 2

- Hough Transform
  - Line, circle fitting
  - Generalized Hough transform
- Interest point, corner detectors
- Pixel based optical flow
- Token based optical flow
- Global motion
- Shape from motion
- Geometry of a stereo camera pair
- Stereopsis
- Epipoloar Geometry

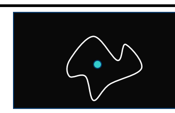
- Hough Transform
- Interest point, corner detectors
- Pixel based optical flow
- Token based optical flow
- Global motion





Hough Transform

- Line fitting $x\cos\theta + y\sin\theta = \rho$
  - 2D accumulator array **A**. Fix θ compute ρ.
  - Increment (θ,ρ) entry of **A**
- Circle fitting $(x - x_o)^2 + (y - y_o)^2 - r^2 = 0$
  - 3D accumulator array **A**. Fix x_0, y_0 compute r.
  - Increment (x_0,y_0,r) entry of **A**
- Practical circle fitting
  - Compute gradient direction at an edge point (θ)
  - Fix r compute $x_0 = x - r\cos\theta \quad y_0 = y - r\sin\theta$
  - Increment (x_0,y_0,r) entry of **A**



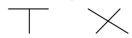
Generalized Hough Transform

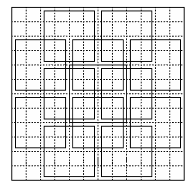
- For shapes with ***no*** analytical expression
- Requires learning of shape
  - For each edge compute direction θ and distance r from centroid
  - Construct a table indexed by θ (r-table)

$\theta 1$	r1, r2, r3 …
$\theta 2$	r14, r21, r23 …
$\theta 3$	r41, r42, r33 …
$\theta 4$	r10, r12, r13 …

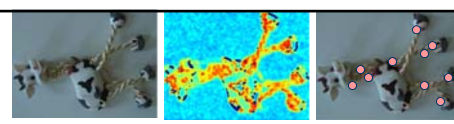
- Shape fitting (detecting)
  - Construct 2D accumulator array for (x_0,y_0)
  - Compute gradient direction θ for each edge point
  - Go to corresponding row of r-table
  - Compute possible x_0 and y_0 from (θ,r) pair
  - Increment accumulator array

Interest Points and Corner

- High texture variation around a pixel
  - T-joints, cross joints etc. 
- Movarec’s interest operator
  - Select 12x12 neighborhood around a pixel
  - Compute intensity variation v in overlapping 4x4 neighborhoods
  - If v for central 4x4 is equal or higher to v all other 4x4, mark pixel as interest point



Alper Yilmaz, Fall 2005 UCF

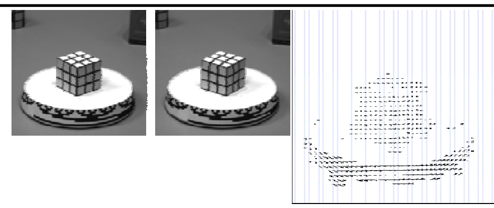


Harris Corner detector

- Smooth image (gaussian filter)
- Compute image derivatives I_x and I_y
- Smooth image gradients (gaussian filter)
- Construct gradient matrix in a neighborhood
  - Find eigenvalues of M
  - Save smallest eigenvalue in a corner strength array **A**
  - Perform non-maximum to **A** and apply threshold to mark corners

$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Alper Yilmaz, Fall 2005 UCF



Optical Flow

- Brightness constancy

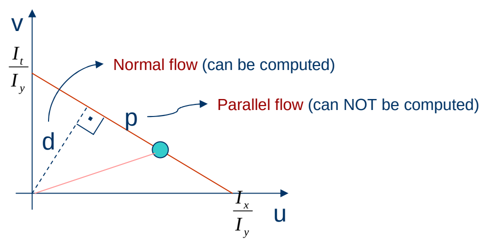
$$I(x,y,t) = I(x+\Delta x, y+\Delta y, t+\Delta t)$$

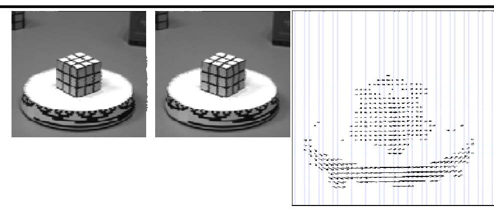
$$uI_x + vI_y + I_t = 0$$

Taylor series expansion

- For given I_x, I_y and I_t there are a set of (u,v) pairs

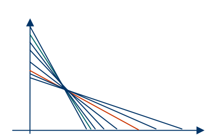
$$v = u\frac{I_x}{I_y} + \frac{I_t}{I_y}$$

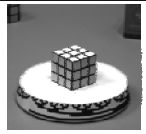
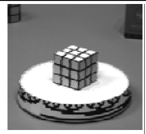
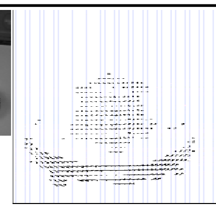




Optical Flow

- Horn & Schunck (1981)
  - Regularization of optical flow (defines 2 energy terms)
    - Brightness constraint energy $E_B(x,y) = (uI_x + vI_y + I_t)^2$
    - Smoothness energy $E_S(x,y) = (u_x^2 + u_y^2 + v_x^2 + v_y^2)$
  - Minimize $E(x,y) = \int \left(E_B(x,y) + \lambda E_S(x,y)\right) dxdy$ iteratively
- Schunk (1989)
  - Neighboring pixels move with same motion
  - Form intersecting lines in (u,v) space
  - Biggest cluster of intersection is the optical flow



Optical Flow

- Lucas & Kanade (1981)
  - Least squares method. Minimize $E = \sum (uI_x + vI_y + I_t)^2$
  - Take derivatives wrt. *u* and *v* equal it to 0.
    - 2 unknowns 2 equations (Lecture 14 slides 26-28)
  - Compute unknowns using least squares.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{\sum I_x^2 \sum I_y^2 - \left(\sum I_x I_y\right)^2} \begin{bmatrix} \sum I_y^2 & -\sum I_x I_y \\ -\sum I_x I_y & \sum I_x^2 \end{bmatrix} \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$$

Optical Flow

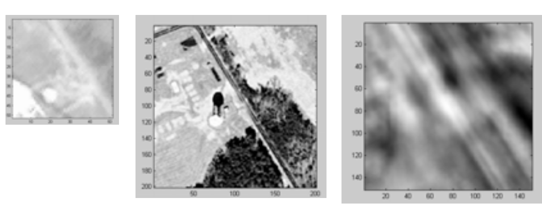
- Optical methods work only for small motion.
- If object moves faster, the brightness changes rapidly, derivative masks fail to estimate spatiotemporal derivatives.
- Gaussian pyramids can be used to compute large optical flow vectors.

Block Based Optical Flow

- First approach
  - Find tokens in 1st image
  - Search similar tokens in 2nd image using correlation methods
- Second approach
  - Find tokens (corners etc.) in both images
  - Find correspondence between tokens by enforcing constraints, such as, maximum speed, common motion, minimum velocity, consistent match

Block Based Optical Flow First Approach

- Select a patch **P** image at time t
- Search for **P** at frame t+1 in a larger neighborhood
  - Compute similarity between original patch and search patch
  - Construct a correlation surface
  - Select maximum



Block Based Optical Flow
Issues With Correlation

- Patch Size
- Search Area
- How many peaks
- Computationally expensive
  - Same operations in Fourier domain takes less time
    - Take FFT of image patch and search area
    - Multiply Fourier coefficients to construct corr. surface
    - Find maximum
- Should use pyramids here too for large displacements

Block Based Optical Flow
Second Approach

- Find initial correspondences using correlation
- Compute costs w_{ij} for each pair of points a_i, b_j
- Construct a bipartite graph based on computed costs
- Prune all edges having weights exceeding certain threshold
  - Define cost matrix
- Find the minimum matching of constructed graph.
  - Hungarian Algorithm
  - Greedy search

$$u(x,y) = a_1 x + a_2 y + b_1$$
$$v(x,y) = a_3 x + a_4 y + b_2$$

Global Motion (Anandan's Approach)

- Common motion observed by most of the pixels in the frame
- Reasons are camera motion or motion of rigid scene
- Uses brightness constraint
- Common motion model is affine (among others)
  - Affine can handle translation, rotation, shear, scaling
- Minimize energy function $E = \sum_{\text{all pixels}} (uI_x + vI_y + I_t)^2$
  - Unknowns are $a_1 \ldots a_4, b_1, b_2$
  - Take derivative make it equal to zero

Alper Yilmaz, Fall 2005 UCF

Global Motion (Anandan's Approach)

$$E(\mathbf{a}) = \sum_{\text{all pixels}} (I_t + \Delta I^T \mathbf{X}\mathbf{a})^2$$

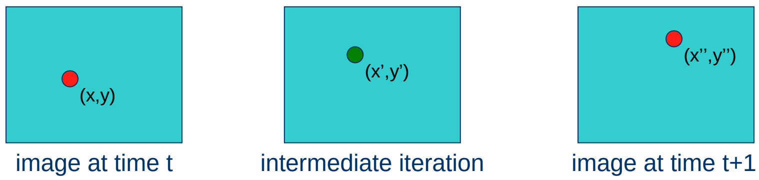
$$\frac{\partial E}{\partial \mathbf{a}} = 2 \sum_{\text{all pixels}} (\Delta I^T \mathbf{X})^T (I_t + \Delta I^T \mathbf{X}\mathbf{a}) = 0$$

$$\sum_{\text{all pixels}} \mathbf{X}^T \Delta I I_t + \sum_{\text{all pixels}} \mathbf{X}^T \Delta I \Delta I^T \mathbf{X}\mathbf{a} = 0$$

$$\underbrace{\sum_{\text{all pixels}} \mathbf{X}^T \Delta I \Delta I^T \mathbf{X}}_{A}\mathbf{a} = -\underbrace{\sum_{\text{all pixels}} \mathbf{X}^T \Delta I \; I_t}_{B} \quad \Rightarrow \quad \mathbf{a} = \mathbf{A}^{-1}\mathbf{B}$$

Alper Yilmaz, Fall 2005 UCF

Iterative Update of Affine Parameters



$$X' = \overbrace{(A_0 + I)}^{A_0'} X + B_0 \qquad X'' = \overbrace{(A_1 + I)}^{A_1'} X' + B_1$$

$$X'' = A_1' A_0' X + A_1' B_0 + B_1$$

$$\hat{A} = A_1' A_0'$$

$$\hat{B} = A_1' B_0 + B_1$$

Alper Yilmaz, Fall 2005 UCF

3D Motion

- Displacement model
- Velocity model

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \mathbf{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{T} = \underbrace{\begin{bmatrix} 1 & -\alpha & \beta \\ \alpha & 1 & -\gamma \\ -\beta & \gamma & 1 \end{bmatrix}}_{\text{Rotation matrix using Euler angles}} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix} \Rightarrow \begin{aligned} X' &= X - \alpha Y + \beta Z + T_X \\ Y' &= \alpha X + Y - \gamma Z + T_Y \\ Z' &= -\beta X + \gamma Y + Z + T_Z \end{aligned}$$

Rotation matrix
using Euler angles

Alper Yilmaz, Fall 2005 UCF

Displacement Model and Its Projection onto Image Space

$$X' = X - \alpha Y + \beta Z + T_X$$
$$Y' = \alpha X + Y - \gamma Z + T_Y$$
$$Z' = -\beta X + \gamma Y + Z + T_Z$$

Orthographic projection

$$x' = x - \alpha y + \beta Z + T_X$$
$$y' = \alpha x + y - \gamma Z + T_Y$$

Perspective projection

$$x' = \frac{X - \alpha Y + \beta Z + T_X}{-\beta X + \gamma Y + Z + T_Z}$$
$$y' = \frac{\alpha X + Y - \gamma Z + T_Y}{-\beta X + \gamma Y + Z + T_Z}$$

$$x' = \frac{x - \alpha y + \beta + \frac{T_X}{Z}}{-\beta x + \gamma y + 1 + \frac{T_Z}{Z}}$$
$$y' = \frac{\alpha x + y - \gamma + \frac{T_Y}{Z}}{-\beta x + \gamma y + 1 + \frac{T_Z}{Z}}$$

Velocity Model in 3D Optical Flow

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} 1 & -\alpha & \beta \\ \alpha & 1 & -\gamma \\ -\beta & \gamma & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

Starting with the displacement

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \left(\begin{bmatrix} 0 & -\alpha & \beta \\ \alpha & 0 & -\gamma \\ -\beta & \gamma & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$\begin{bmatrix} X' - X \\ Y' - Y \\ Z' - Z \end{bmatrix} = \begin{bmatrix} 0 & -\alpha & \beta \\ \alpha & 0 & -\gamma \\ -\beta & \gamma & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

Velocity Model in 3D
Optical Flow

$$\begin{bmatrix} X'-X \\ Y'-Y \\ Z'-Z \end{bmatrix} = \begin{bmatrix} 0 & -\alpha & \beta \\ \alpha & 0 & -\gamma \\ -\beta & \gamma & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

rotational velocities

translational velocities

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad \Omega = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \end{bmatrix}$$

$$\dot{\mathbf{X}} = \Omega \times \mathbf{X} + \mathbf{V}$$

cross product

Alper Yilmaz, Fall 2005 UCF

$$\dot{X} = \Omega_2 Z - \Omega_3 Y + V_1$$
$$\dot{Y} = \Omega_3 X - \Omega_1 Z + V_2$$
$$\dot{Z} = \Omega_1 Y - \Omega_2 X + V_3$$

Velocity Model in 2D

- Orthographic projection

$$x = X \quad y = Y \;\Rightarrow\; u = \dot{x} = \dot{X} \quad v = \dot{y} = \dot{Y} \;\Rightarrow\; u = \Omega_2 Z - \Omega_3 y + V_1 \quad v = \Omega_3 x - \Omega_1 Z + V_2$$

- Perspective projection

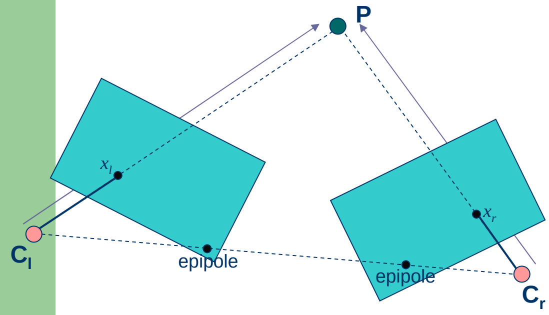
$$x = f\frac{X}{Z} \quad y = f\frac{Y}{Z} \;\Rightarrow\; u = \dot{x} \quad v = \dot{y} \;\Rightarrow$$

$$u = f(\frac{V_1}{Z} + \Omega_2) - \frac{V_3}{Z}x - \Omega_3 y - \frac{\Omega_1}{f}xy + \frac{\Omega_2}{f}x^2$$
$$v = f(\frac{V_2}{Z} - \Omega_1) + \Omega_3 x - \frac{V_3}{Z}y + \frac{\Omega_2}{f}xy - \frac{\Omega_1}{f}y^2$$

HOMEWORK

Alper Yilmaz, Fall 2005 UCF

Epipolar Geometry



$T = C_l C_r$

$P_l = PC_l$

$P_r = R(P_l - T)$

- Co-planarity condition
  - 3 vectors on a plane

$$A^T (B \times C) = 0$$

  - Let's define for epipolar plane

$$(P_l - T)^T (T \times P_l) = 0$$

$$(R^T P_r)^T (T \times P_l) = 0$$

$$P_r^T RSP_l = 0$$

Essential Matrix

- Related to camera extrinsic parameters
  - Rotation and translation

$$P_r^T RSP_l = 0$$

$$E = RS$$

- Captures relation between to camera coordinates

Fundamental Matrix

- From camera coordinates to image coordinates.
  - Intrinsic camera parameters (in homogenous coordinates)

$$x_r^T M_r^{-T} RSM_t^{-1} x_l = 0$$

$$E = RS$$

$$\left.\begin{aligned} F &= M_r^{-T} RSM_t^{-1} \\ F &= M_r^{-T} EM_t^{-1} \end{aligned}\right\} \text{Fundamental matrix}$$

Alper Yilmaz, Fall 2005 UCF