CAP 5415 Computer Vision
Fall 2005

Dr. Alper Yilmaz
Univ. of Central Florida
www.cs.ucf.edu/courses/cap5415/fall2005
Office: CSB 250

Recap
Motion

- Brightness constancy constraint

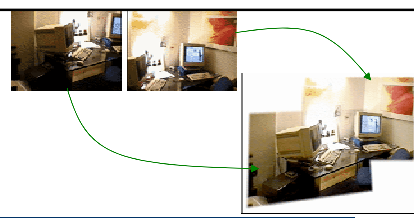
$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

- Optical flow equation, brightness constancy equation
  - Normal flow **can be** computed
  - Parallel flow **cannot**

$$uI_x + vI_y + I_t = 0 \qquad v = u\frac{I_x}{I_y} + \frac{I_t}{I_y} = 0 \text{ line equation}$$

Recap
Computing Optical Flow

- Define energy function
  - Brightness constancy + smooth solution (Horn&Schuck)
  - Common flow (Schuck)
  - Brightness constancy (Lucas&Kanade)
    - Least squares fitting



Recap
Global Motion

- Common motion of pixels observed in frame
  - Camera motion or rigid object motion
- Affine Model
  - Affine Transformation
  - Affine Motion

Alper Yilmaz, Fall 2005 UCF

Recap
Affine Transformation

- Direct relation between pixel positions

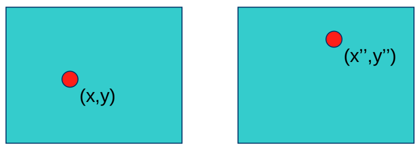


image at time t   image at time t+1

$$x'' = a_1 x + a_2 y + b_1$$
$$y'' = a_3 x + a_4 y + b_2$$

Recap
Affine Motion

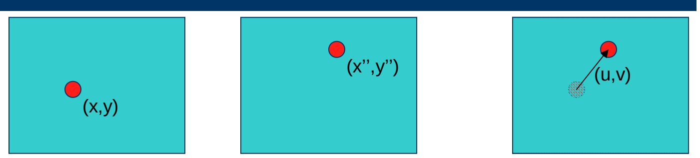


image at time t   image at time t+1

$$u(x,y) = a_1 x + a_2 y + b_1$$
$$u = x'' - x$$
$$x'' - x = a_1 x + a_2 y + b_1$$
$$\boxed{x'' = (a_1 + 1)x + a_2 y + b_1}$$

$$v(x,y) = a_3 x + a_4 y + b_2$$
$$v = y'' - y$$
$$\boxed{y'' = a_3 x + (a_4 + 1)y + b_2}$$

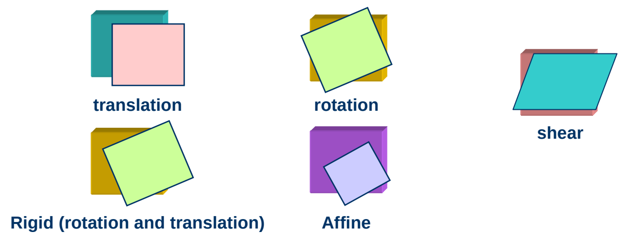
Recap
Affine Motion and Transformation

- Transformation

$$x'' = a_1 x + a_2 y + b_1$$
$$y'' = a_3 x + a_4 y + b_2$$

Motion

$$x'' = (a_1 + 1)x + a_2 y + b_1$$
$$y'' = a_3 x + (a_4 + 1)y + b_2$$



James R. Bergen, P. Anandan, Keith J. Hanna, Rajesh Hingorani: “Hierarchical Model-Based Motion Estimation,” ECCV 1992: 237-252

Recap
Anandan’s Approach (Affine Motion)

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

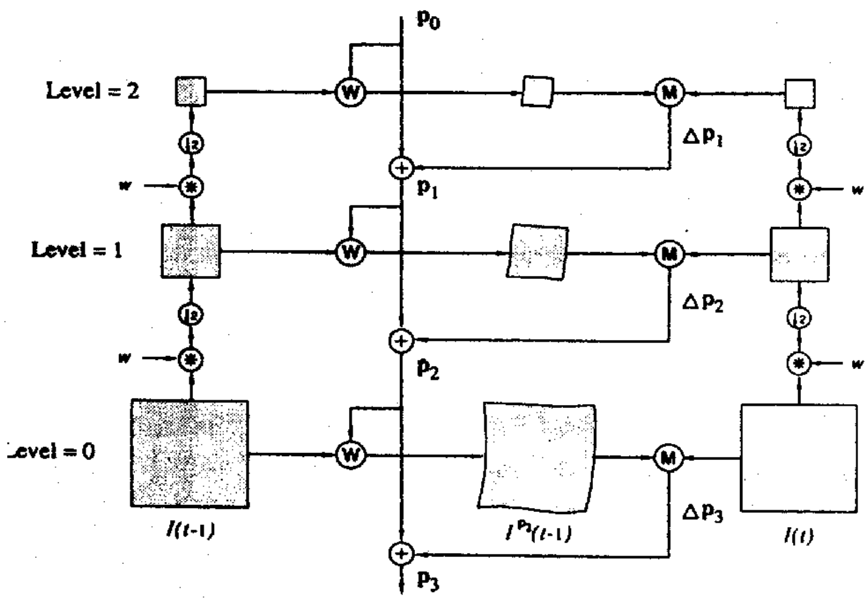
$$\begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ b_1 \\ a_3 \\ a_4 \\ b_2 \end{bmatrix}$$

$$\mathbf{u} = \mathbf{Xa}$$
$$\Delta I^T \mathbf{u} + I_t = 0$$
$$\Delta I^T \mathbf{Xa} + I_t = 0$$

minimize $E = \sum \left(\Delta I^T \mathbf{Xa} + I_t \right)^2$

$$\frac{\partial E}{\partial \mathbf{a}} = 2 \sum (\Delta I^T \mathbf{X})^T (I_t + \Delta I^T \mathbf{Xa}) = 0$$

$$\sum_{\text{all pixels}} \mathbf{X}^T \Delta I \Delta I^T \mathbf{Xa} = - \sum_{\text{all pixels}} \mathbf{X}^T \Delta I \; I_t$$

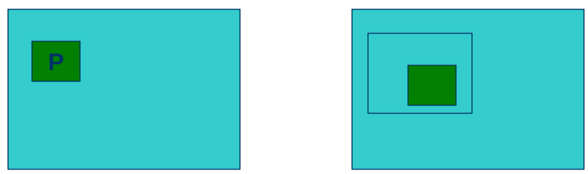


Block Based Optical Flow

Alper Yilmaz, Fall 2005 UCF

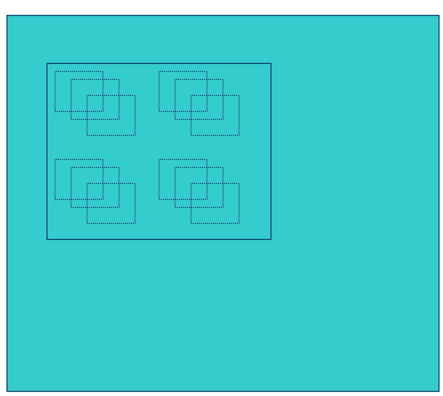
Block Based

- Select a patch **P** image at time t
- Search for **P** at frame t+1 in a larger neighborhood



Alper Yilmaz, Fall 2005 UCF

Block Search



- Search for patch in overlapping windows
- Compute similarity of intensity values between original patch and search patch
- Select the location with highest similarity
- Distance vector between centroids give optical flow vector

Alper Yilmaz, Fall 2005 UCF

Computing Similarity

$$\text{SSD} = \sum\sum(I_{t+1} - I_t)^2$$

Sum of square differences

$$\text{AD} = \sum\sum|(I_{t+1} - I_t)|$$

Absolute difference

$$\text{CC} = \sum\sum I_{t+1} I_t$$

Cross correlation

$$\text{NC} = \frac{\sum\sum(I_{t+1}.I_t)}{\sqrt{\sum\sum I_t.I_t}}$$

Normalized correlation

$$\text{MC} = \frac{1}{64\sigma_{t+1}\sigma_t}\sum\sum(I_{t+1} - \mu_{t+1})(I_t - \mu_t)$$

Mutual correlation

μ, σ are region mean and stdv

Alper Yilmaz, Fall 2005 UCF

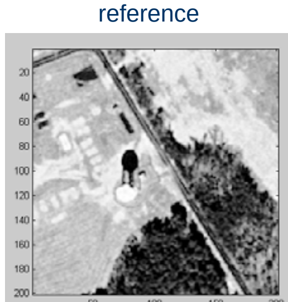
Correlation Surface

- Using Cross correlation

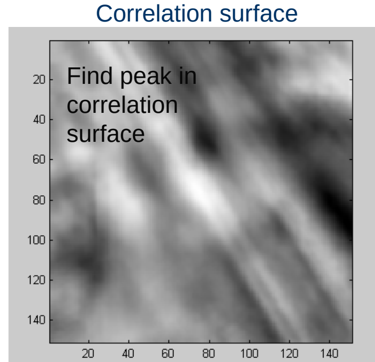
$$\text{CC} = \sum\sum I_{t+1} I_t$$

mission

reference



Correlation surface



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

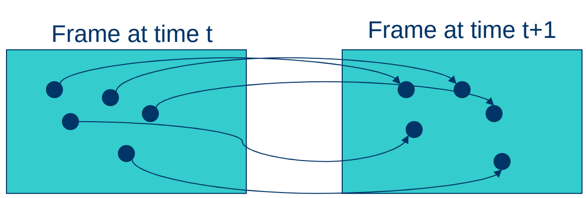
Token Based Optical Flow

Tokens

- Interest points
  - Movarec’s operator
- Corners
  - Harris corner detector
- Edges
  - Any edge detection algorithm

Overview

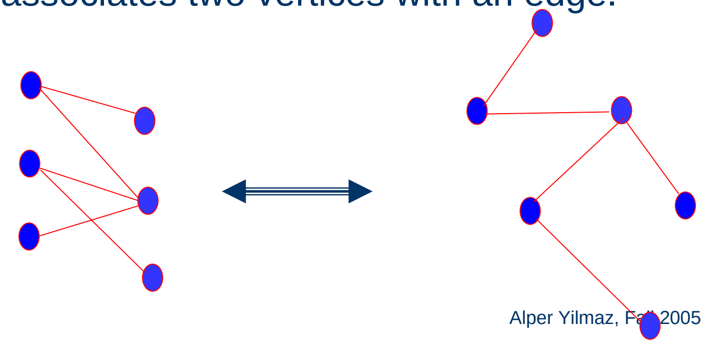
- Given two consecutive images
- Find tokens in both images
- Find token correspondences



Alper Yilmaz, Fall 2005 UCF

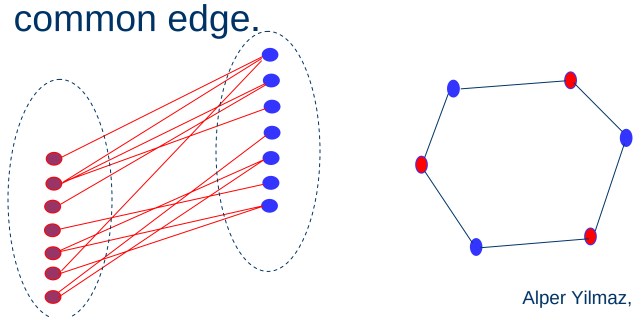
Representing the Correspondence

- A graph **G(V,E)** is a triple consisting of a vertex set **V** an edge set **E** and a *relation* that associates two vertices with an edge.



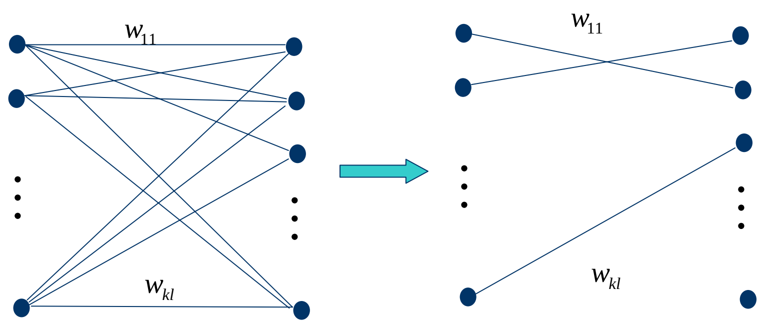
Representing the Correspondence

- **Bipartite graph:** A graph **G** is bipartite if its vertex set can be partitioned in two subsets such that no two vertex in same set have a common edge.



Finding Correspondence

- Finding matching: Matching is a **set of edges** such that **no** two of them have a **common** vertex



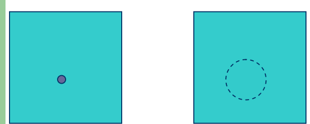
Alper Yilmaz, Fall 2005 UCF

Token Based Optical Flow

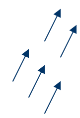
- Tokens corresponds to vertices in the bipartite graph
- Tokens at time instants *t* and *t*+1 form partite sets of graph.
- The cost of corresponding a point at instant *t* to a point at instant *t*+1 is the weight of edge between the corresponding vertices.

Alper Yilmaz, Fall 2005 UCF

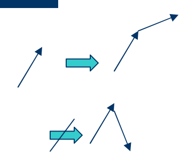
Defining Weights



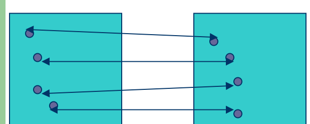
Maximum Speed



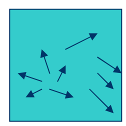
Common Motion



Minimum Velocity



Consistent Match



Model

Weights

geometric mean $GM(a,b)=\sqrt{ab}$

aritmetic mean $AM(a,b)=\frac{a+b}{2}$

Ullman: $w_{ij}=\|y_j-x_i\|$ (Absolute distance between points)

Sethi & Jain: $$w_{ij}=c\left[1-\frac{(y_j-x_i).v_{x_i}}{\|y_j-x_i\|\|v_{x_i}\|}\right]+(1-c)\left[1-\frac{GM(\|y_j-x_i\|,\|v_{x_i}\|)}{AM(\|y_j-x_i\|,\|v_{x_i}\|)}\right]$$

Rangarajan & Shah: $$w_{ij}=\frac{\|v_{x_i}-(y_j-x_i)\|}{\sum_{k=1}^{p}\sum_{l=1}^{q}\|v_{x_k}-(y_l-x_k)\|}+\frac{\|y_j-x_i\|}{\sum_{k=1}^{p}\sum_{l=1}^{q}\|y_l-x_p\|}$$

Algorithm

- Find initial correspondences using correlation
- Compute costs w_{ij} for each pair of points x_i, y_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

Greedy Algorithm

Algorithm A

1. For $k = 2$ to $n-1$ do
   (a) Construct M an $(m*m)$ matrix, with the points from kth frame along the rows and points from $(k+1)$th frame along the columns. Let $M[i,j] = \delta(X_p^{k-1} X_i^k X_j^{k+1})$, when $\Phi^{k-1}(p) = i$.
   (b) for $a = 1$ to m do
      i. Identify the minimum element $[i, l_i]$ in each row i of M.
      ii. Compute *priority matrix* B, such that $B[i, l_i] = \sum_{j=1, j\neq l_i}^{m} M[i,j] + \sum_{k=1, k\neq i}^{m} M[k, l_i]$ for each i.
      iii. Select $[i, l_i]$ pair with highest *priority* value $B[i, l_i]$, and make $\Phi^k(i) = l_i$.
      iv. Mask row i and column l_i from M.

Figure 5.8: Motion correspondence using multiple frames.