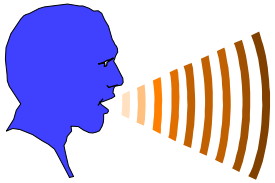
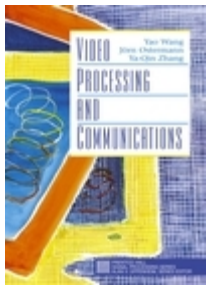





Video Processing & Communications

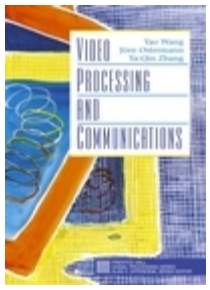
Two-Dimensional Motion Estimation (Part II: Advanced Techniques)

Yao Wang
Polytechnic University, Brooklyn, NY11201
http://eeweb.poly.edu/~yao



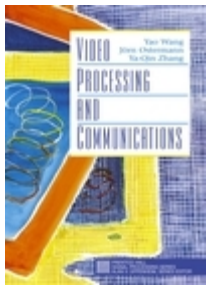
Outline

- Problems with EBMA
- Multiresolution motion estimation
  - Hierarchical block matching algorithm (HBMA)
- Deformable block matching algorithm (DBMA)
  - Node-based motion model
- Mesh-based motion estimation
  - Mesh-based motion representation
  - Mesh-based motion estimation
- Global motion estimation
  - Direct method
  - Indirect method
- Region-based motion estimation
- Summary



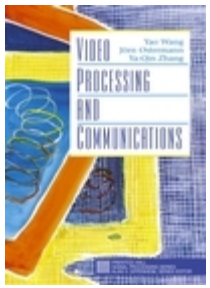
Problems with EBMA

- Blocking effect (discontinuity across block boundary) in the predicted image
  - Because the block-wise translation model is not accurate
  - Real motion in a block may be more complicated than translation
    - Fix: Deformable BMA
  - There may be multiple objects with different motions in a block
    - Fix:
      - region-based motion estimation
      - mesh-based using adaptive mesh
  - Intensity changes may be due to illumination effect
    - Should compensate for illumination effect before applying "constant intensity assumption"



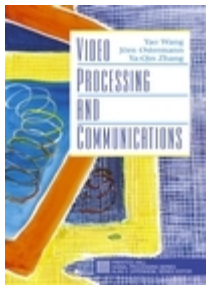
Problems with EBMA (Cntd)

- Motion field somewhat chaotic
  - because MVs are estimated independently from block to block
  - Fix:
    - Imposing smoothness constraint explicitly
    - Multi-resolution approach
    - Mesh-based motion estimation
- Wrong MV in the flat region
  - because motion is indeterminate when spatial gradient is near zero
  - Ideally, should use non-regular partition
  - Fix: region based motion estimation
- Requires tremendous computation!
  - Fix:
    - fast algorithms
    - Multi-resolution

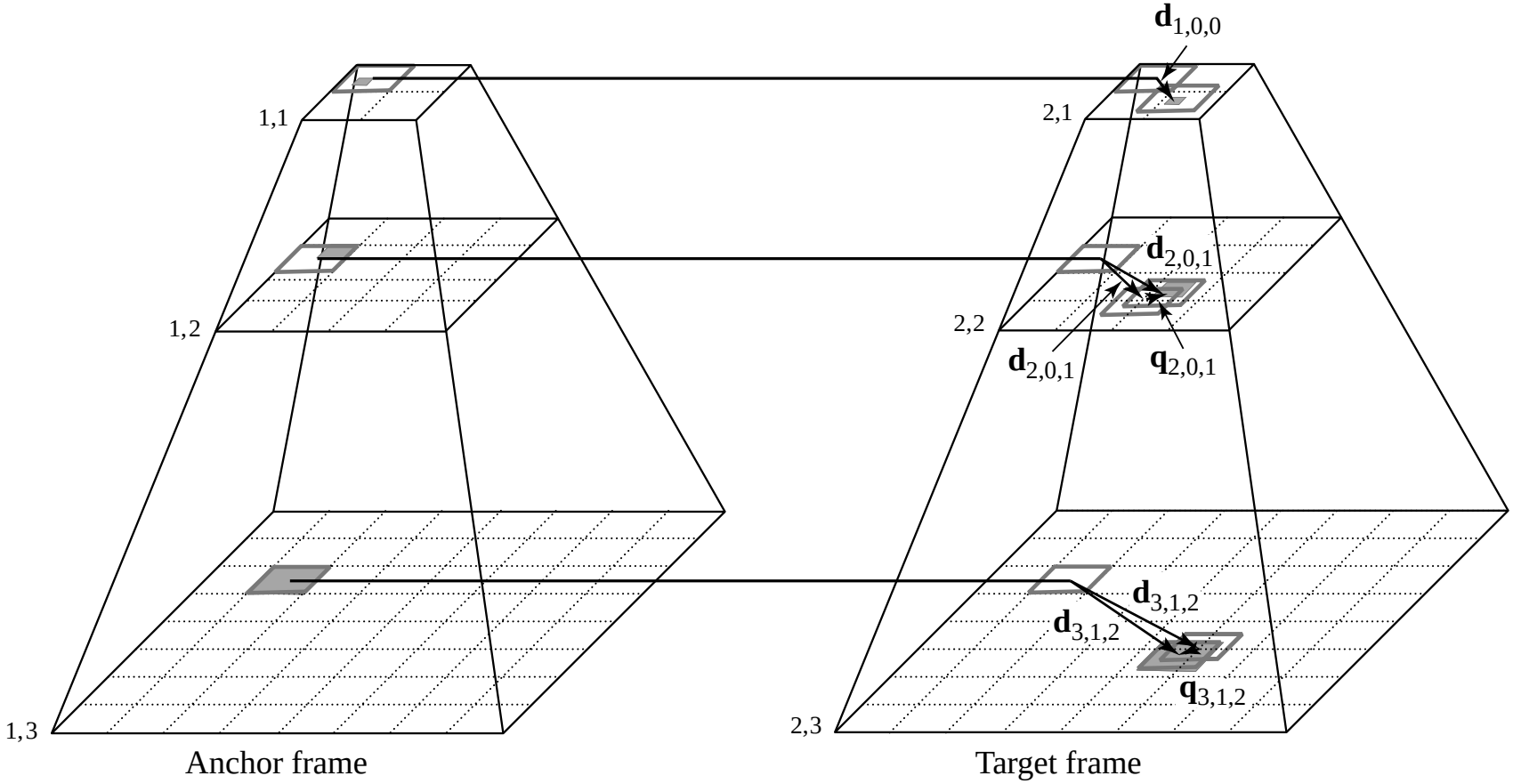


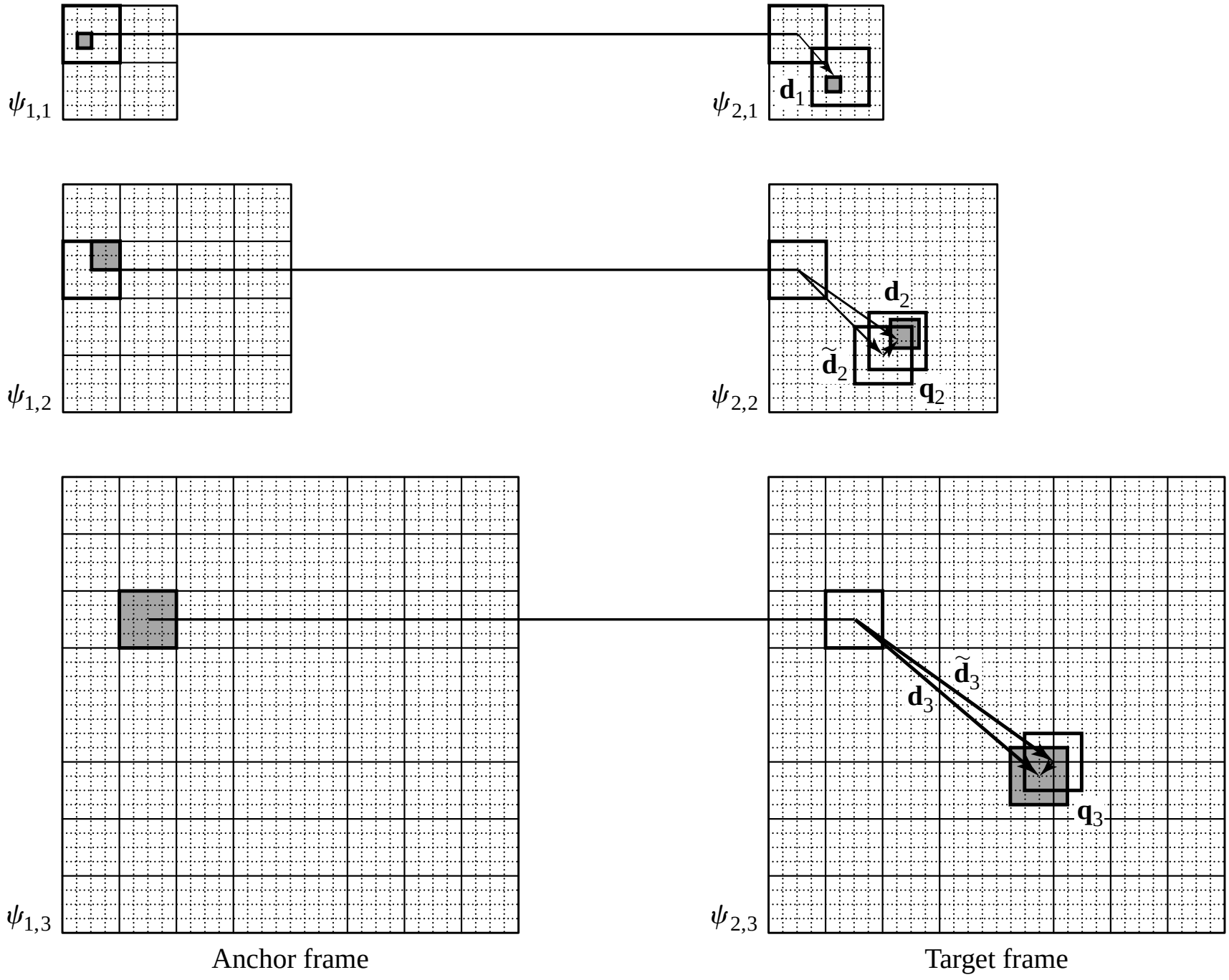
Multi-resolution Motion Estimation

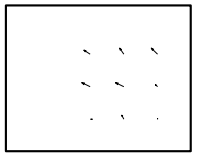
- Problems with BMA
  - Unless exhaustive search is used, the solution may not be global minimum
  - Exhaustive search requires extremely large computation
  - Block wise translation motion model is not always appropriate
- Multiresolution approach
  - Aim to solve the first two problems
  - First estimate the motion in a coarse resolution over low-pass filtered, down-sampled image pair
    - Can usually lead to a solution close to the true motion field
  - Then modify the initial solution in successively finer resolution within a small search range
    - Reduce the computation
  - Can be applied to different motion representations, but we will focus on its application to BMA



Hierarchical Block Matching Algorithm (HBMA)

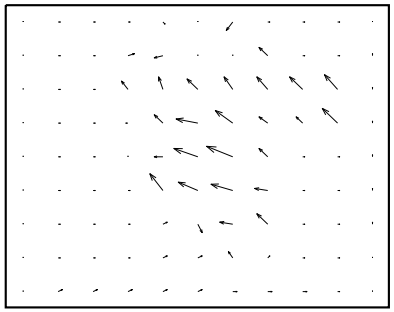



$\psi_{1,1}$
$\psi_{2,1}$
$\mathbf{d}_1$
$\psi_{1,2}$
$\psi_{2,2}$
$\mathbf{d}_2$
$\tilde{\mathbf{d}}_2$
$\mathbf{q}_2$
$\psi_{1,3}$
$\psi_{2,3}$
$\tilde{\mathbf{d}}_3$
$\mathbf{d}_3$
$\mathbf{q}_3$
Anchor frame
Target frame



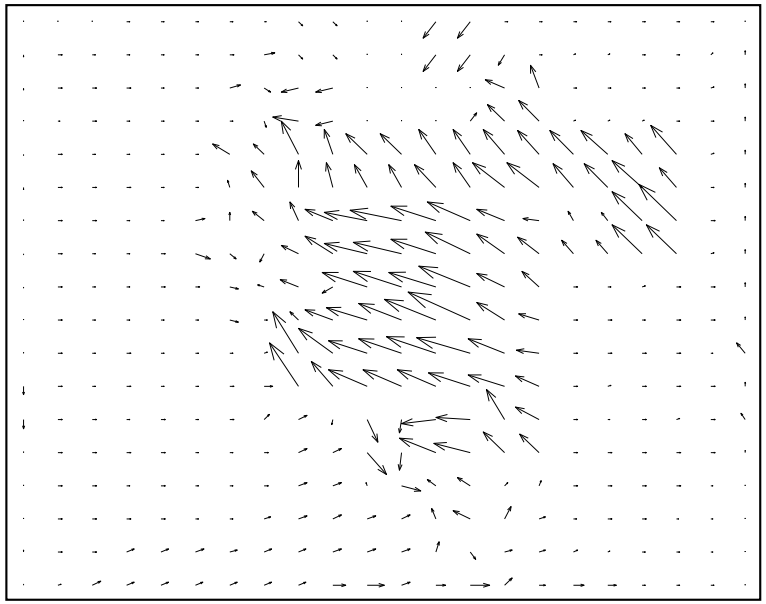
(a)

(b)



(c)

(d)



(e)

(f)

Predicted anchor frame (29.32dB)

Example: Three-level HBMA

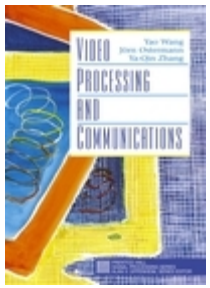
Motion field

target frame

Example: Half-pel EBMA

Predicted anchor frame (29.86dB)

anchor frame

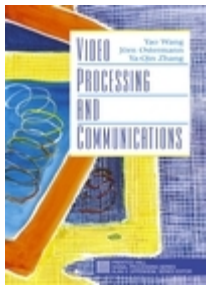


Computation Requirement of HBMA

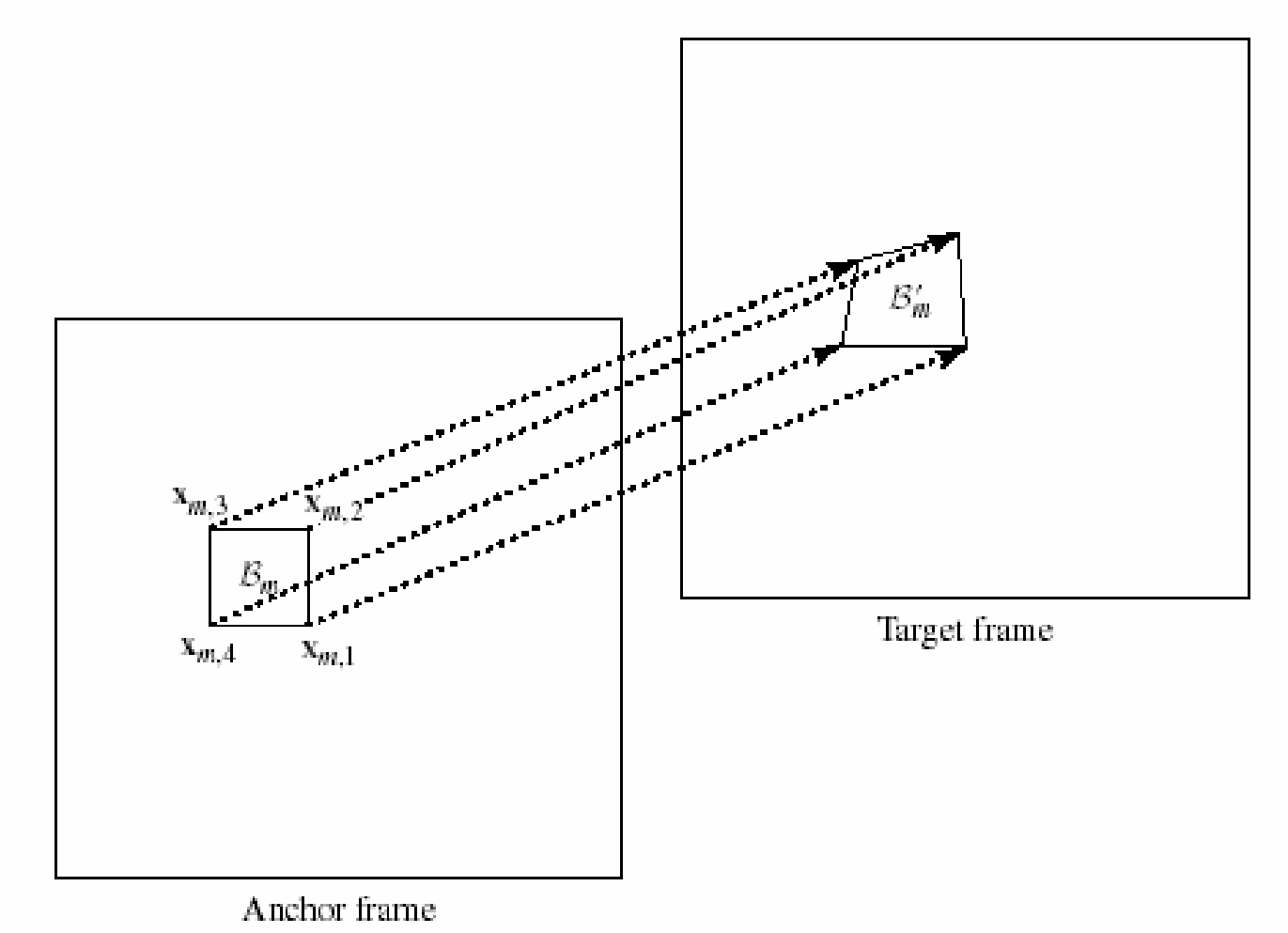
- Assumption
  - Image size: MxM; Block size: NxN at every level; Levels: L
  - Search range:
    - 1st level: R/2^(L-1) (Equivalent to R in L-th level)
    - Other levels: R/2^(L-1) (can be smaller)
- Operation counts for EBMA
  - image size M, block size N, search range R
  - # operations: $M^2(2R+1)^2$
- Operation counts at l-th level (Image size: M/2^(L-l))

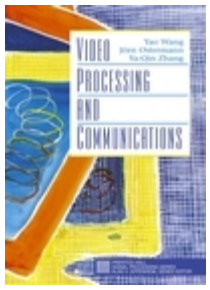
  $$\left(M/2^{L-l}\right)^2\left(2R/2^{L-1}+1\right)^2$$
- Total operation count

  $$\sum_{l=1}^{L}\left(M/2^{L-l}\right)^2\left(2R/2^{L-1}+1\right)^2 \approx \frac{1}{3}4^{-(L-2)}4M^2R^2$$
- Saving factor: $3\cdot 4^{(L-2)} = 3(L=2); 12(L=3)$



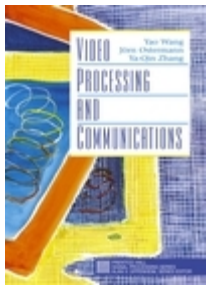
Deformable Block Matching Algorithm





Overview of DBMA

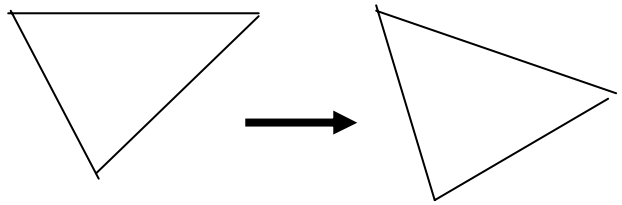
- Three steps:
  - Partition the anchor frame into regular blocks
  - Model the motion in each block by a more complex motion
    - The 2-D motion caused by a flat surface patch undergoing rigid 3-D motion can be approximated well by projective mapping
    - Projective Mapping can be approximated by affine mapping and bilinear mapping
    - Various possible mappings can be described by a node-based motion model
  - Estimate the motion parameters block by block independently
    - Discontinuity problem cross block boundaries still remain
- Still cannot solve the problem of multiple motions within a block or changes due to illumination effect!



Affine and Bilinear Model

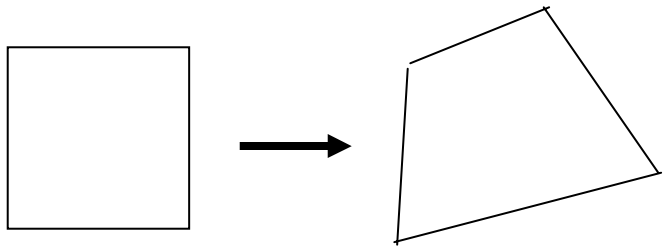
- Affine (6 parameters):
  - Good for mapping triangles to triangles

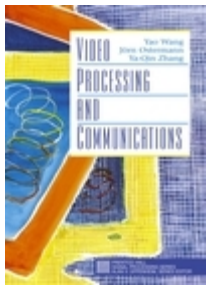
$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y \\ b_0 + b_1 x + b_2 y \end{bmatrix}$$



- Bilinear (8 parameters):
  - Good for mapping blocks to quadrangles

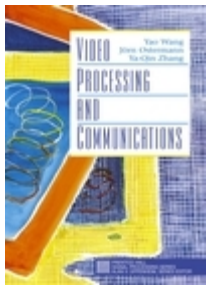
$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y + a_3 xy \\ b_0 + b_1 x + b_2 y + b_3 xy \end{bmatrix}$$





Difficulties in Estimating Affine and Bilinear Motion Parameters

- The coefficients need floating point precision
- The coefficients have different influence on the estimated motion
  - 0-th order coefficients (a0,b0) represent the translation component
  - Other coefficients' influence depend on the pixel coordinates



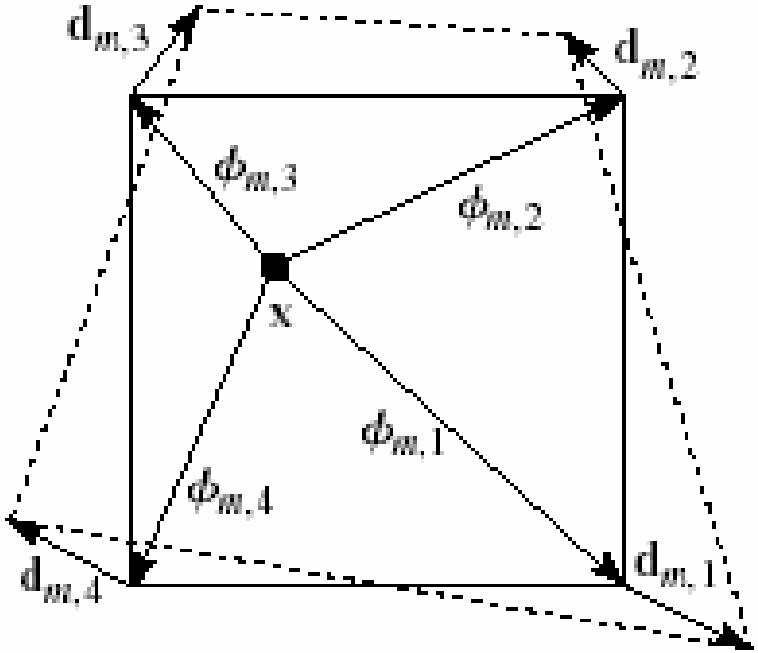
Node-Based Motion Model

Control nodes in this example: Block corners

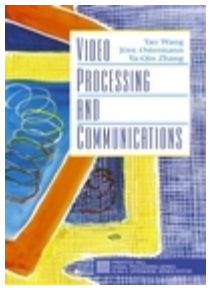
Motion in other points are interpolated from the nodal MVs $\mathbf{d}_{m,k}$

Control node MVs can be described with integer or half-pel accuracy, all have same importance

Translation, affine, and bilinear are special case of this model



$$\mathbf{d}_m(\mathbf{x}) = \sum_{k=1}^{K} \phi_{m,k}(\mathbf{x})\mathbf{d}_{m,k}, \quad \mathbf{x} \in \mathcal{B}_m.$$



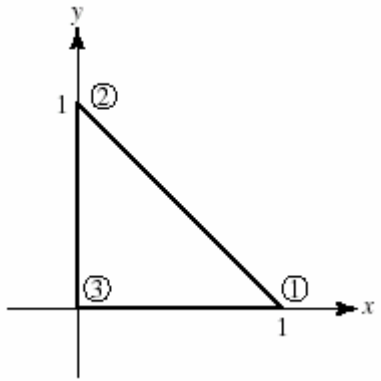
Interpolation Kernels

- Requirement:

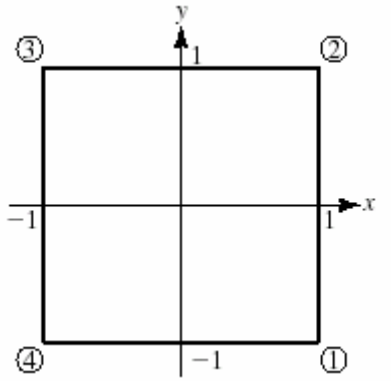
$$0 \le \phi_{m,k}(\mathbf{x}) \le 1, \quad \sum_k \phi_{m,k}(\mathbf{x}) = 1, \quad \forall\, \mathbf{x} \in \mathcal{B}_{1,m}, \quad \phi_{m,k}(\mathbf{x}_l) = \delta_{k,l} = \begin{cases} 1 & k = l, \\ 0 & k \ne l. \end{cases}$$

- For standard triangular element:

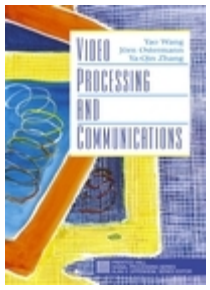
$$\phi_1^t(x, y) = x, \quad \phi_2^t(x, y) = y, \quad \phi_3^t(x, y) = 1 - x - y.$$



- For standard quadrilateral element:



$$\phi_1^q(x, y) = (1 + x)(1 - y)/4, \quad \phi_2^q(x, y) = (1 + x)(1 + y)/4,$$
$$\phi_3^q(x, y) = (1 - x)(1 + y)/4, \quad \phi_4^q(x, y) = (1 - x)(1 - y)/4.$$



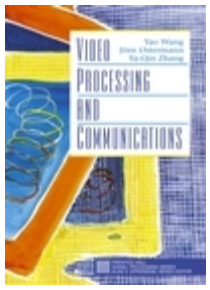
Estimation of Nodal Motions

- Objective function:

$$E(\mathbf{a}) = \sum_{\mathbf{x}\in\mathcal{B}} |\psi_2(\mathbf{w}(\mathbf{x};\mathbf{a})) - \psi_1(\mathbf{x})|^p, \qquad \mathbf{a} = [\mathbf{d}_k, k \in \mathcal{K}]$$

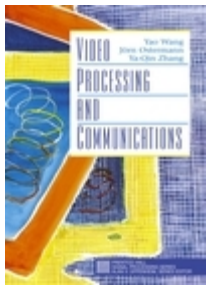
$$\mathbf{w}(\mathbf{x};\mathbf{a}) = \mathbf{x} + \sum_{k\in\mathcal{K}} \phi_k(\mathbf{x})\mathbf{d}_k.$$

- Search method:
  - Exhaustive search:
    - search K nodal MVs simultaneously in integer or half-pel accuracy, may not be feasible in practice
  - Gradient descent approach:
    - See textbook for the Newton-Raphson update algorithm
    - Solution depends on the initial solution. A good initial solution is the translation MV found using EBMA

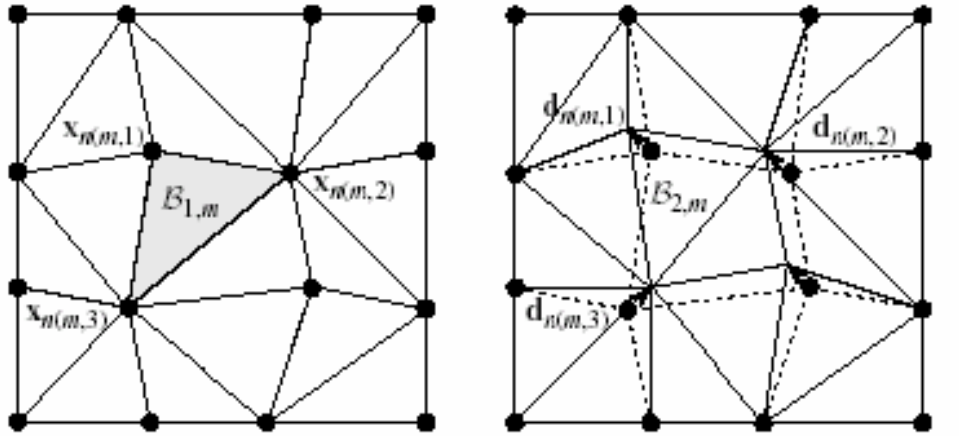


Problems with DBMA

- Motion discontinuity across block boundaries, because nodal MVs are estimated independently from block to block
  - Fix: mesh-based motion estimation
  - First apply EBMA to all blocks
- Cannot do well on blocks with multiple moving objects or changes due to illumination effect
  - Three mode method
    - First apply EBMA to all blocks
    - Blocks with small EBMA errors have *translational motion*
    - Blocks with large EBMA errors may have *non-translational motion*
      - First apply DBMA to these blocks
      - Blocks still having errors are *non-motion compensable*
    - [Ref] O. Lee and Y. Wang, Motion compensated prediction using nodal based deformable block matching. J. Visual Communications and Image Representation (March 1995), 6:26-34



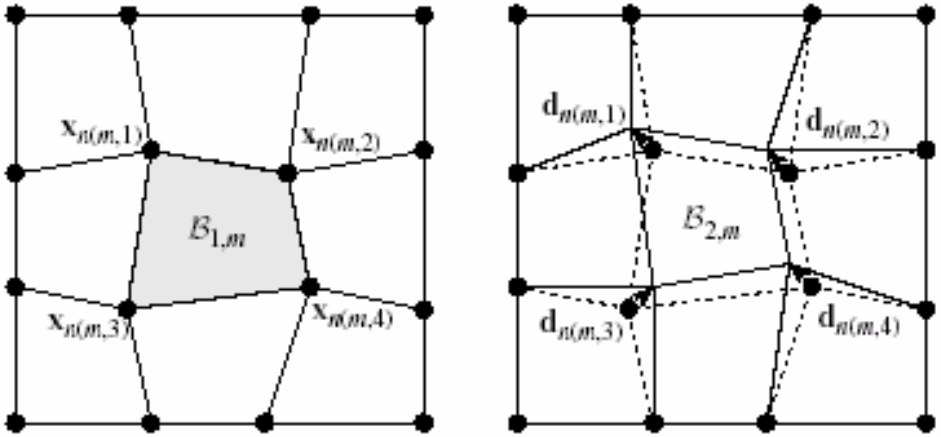
Mesh-Based Motion Estimation: Overview



(a)

$$\mathbf{d}_m(\mathbf{x}) = \sum_{k \in \mathcal{K}} \phi_{m,k}(\mathbf{x}) \mathbf{d}_{n(m,k)}, \quad \mathbf{x} \in \mathcal{B}_{1,m},$$

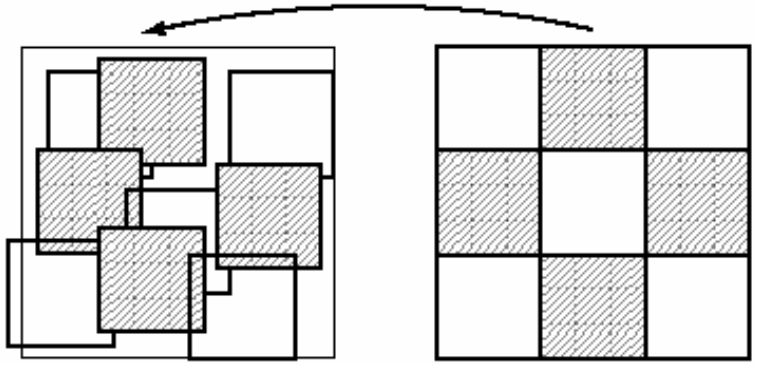
(a) Using a triangular mesh



(b)

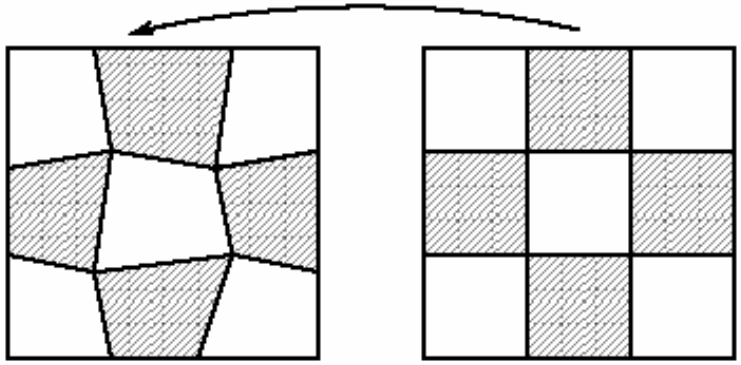
(b) Using a quadrilateral mesh

Mesh-based vs. block-based motion estimation



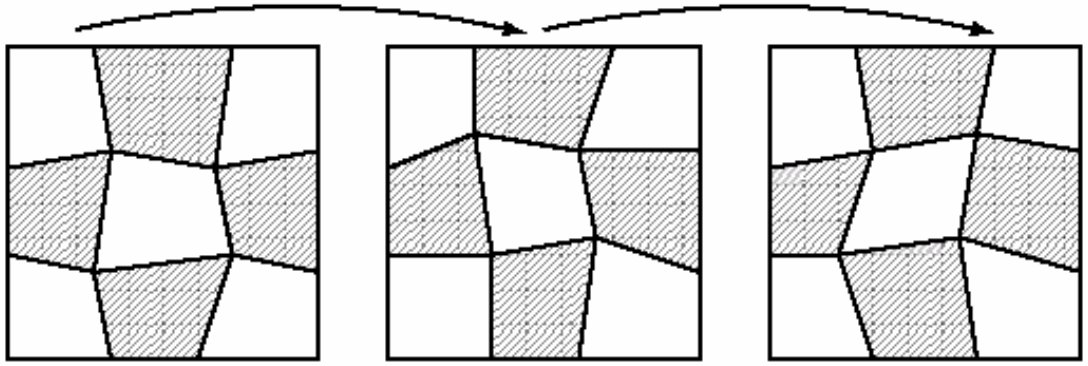
(a)

(a) block-based backward ME



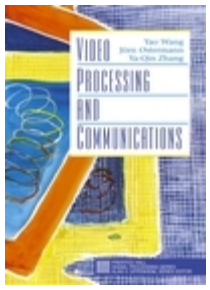
(b)

(b) mesh-based backward ME



(c)

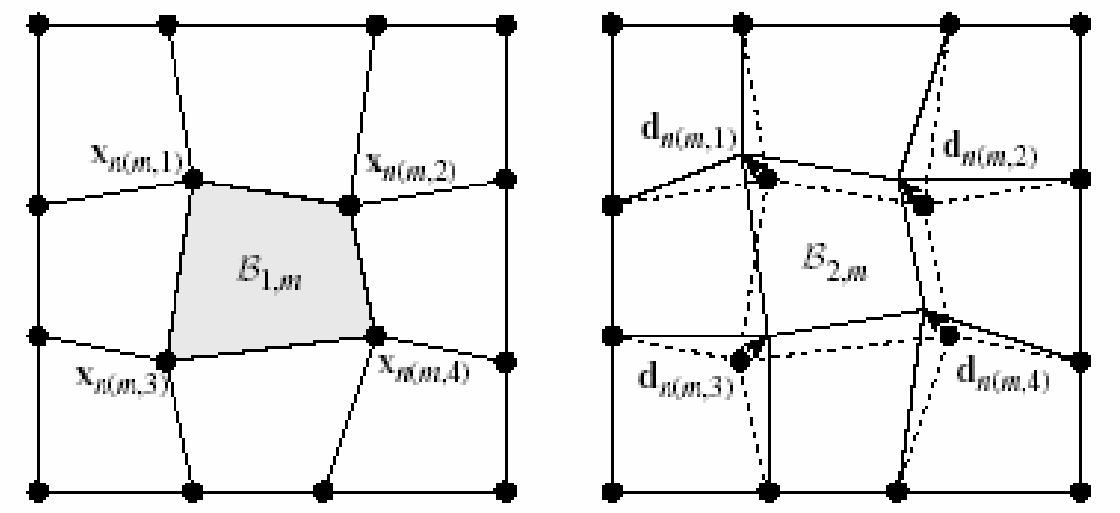
(c) mesh-based forward ME



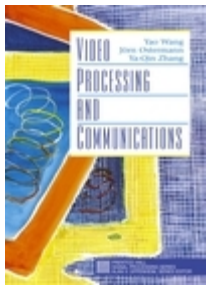
Mesh-Based Motion Model

- The motion in each element is interpolated from nodal MVs

$$\mathbf{d}_m(\mathbf{x}) = \sum_{k \in \mathcal{K}} \phi_{m,k}(\mathbf{x}) \mathbf{d}_{n(m,k)}, \quad \mathbf{x} \in \mathcal{B}_{1,m},$$

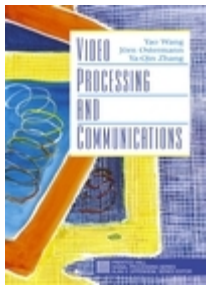


- Mesh-based vs. node-based model:
  - Mesh-based: Each node has a single MV, which influences the motion of all four adjacent elements
  - Node-based: Each node can have four different MVs depend on within with element it is considered



Mesh Generation and Motion Estimation

- Two problems:
  - Given a mesh in the anchor frame, determine nodal positions in the target frame – Motion estimation
  - Set up the mesh in the anchor frame, so that the mesh conforms with object boundaries – Mesh generation
    - Backward ME: can use either regular mesh or object adaptive mesh at each new frame
      - Motion estimation is easier with a regular mesh, but adaptive mesh can yield more accurate result
    - Forward ME:
      - only need to establish a mesh for the initial frame. Meshes in the following frames depend on the nodal MVs between successive frames.
      - To accommodate appearing/disappearing objects, the mesh geometry needs to be updated.
  - Only discuss motion estimation problem here



Estimation of Nodal Motion

- Unlike DBMA, all nodal MVs should be estimated simultaneously

$$E(\mathbf{d}_n, n \in \mathcal{N}) = \sum_{m\in\mathcal{M}} \sum_{\mathbf{x}\in\mathcal{B}_{1,m}} |\psi_2(\mathbf{w}_m(\mathbf{x})) - \psi_1(\mathbf{x})|^p,$$

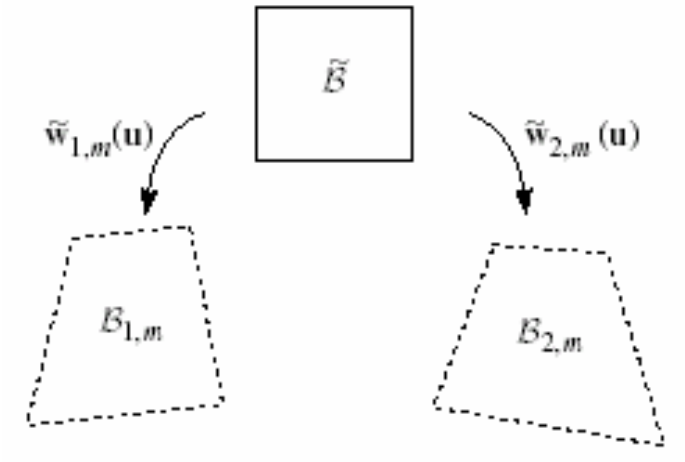
$$\mathbf{w}_m(\mathbf{x}) = \mathbf{x} + \sum_{k\in\mathcal{K}} \phi_{m,k}(\mathbf{x})\mathbf{d}_{n(m,k)}, \quad \mathbf{x} \in \mathcal{B}_{1,m}.$$

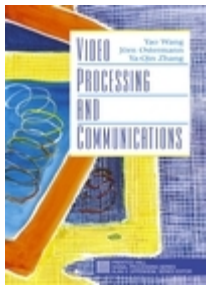
- Unless the anchor frame uses a regular mesh, the interpolation kernels are complicated
- Using a mapping to master element:

$$E(\mathbf{d}_n, n \in \mathcal{N}) = \sum_{m\in\mathcal{M}} \sum_{\mathbf{u}\in\tilde{\mathcal{B}}} |\tilde{e}_m(\mathbf{u})|^p |J_m(\mathbf{u})|,$$

$$\tilde{e}_m(\mathbf{u}) = \psi_2(\tilde{\mathbf{w}}_{2,m}(\mathbf{u})) - \psi_1(\tilde{\mathbf{w}}_{1,m}(\mathbf{u}))$$

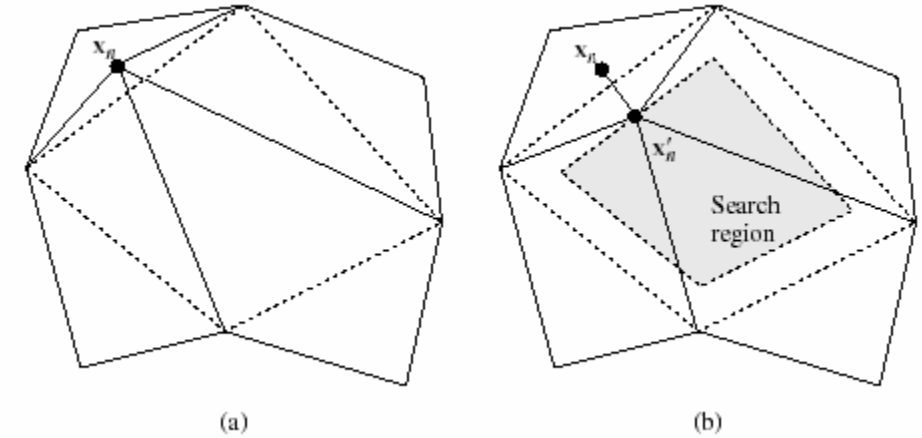
$$\tilde{\mathbf{w}}_{t,m}(\mathbf{u}) = \sum_{k\in\mathcal{K}} \tilde{\phi}_k(\mathbf{u})\mathbf{x}_{t,n(m,k)}, \quad \mathbf{u} \in \tilde{\mathcal{B}}, \quad t = 1, 2.$$





Estimation of Nodal Motion (cntd)

- Simplification:
  - Update one node at a time, minimizing DFD over all adjacent elements
    - Gradient descent method [Wang and Lee 1994]
    - Exhaustive search [Wang and Ostermann 1998]
  - Update order is important
    - First update those nodes where motion can be estimated accurately (near edges)
  - Motion of this node should be constrained not to cause excessively deformed elements



Motion field

target frame

Example: Half-pel EBMA

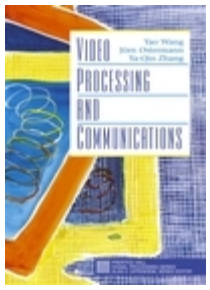
Predicted anchor frame (29.86dB)

anchor frame

EBMA vs. Mesh-based Motion Estimation

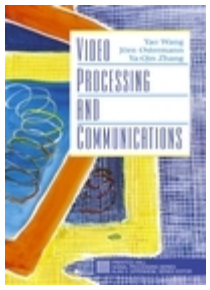
mesh-based method (29.72dB)

EBMA (29.86dB)



Global Motion Estimation

- Global motion:
  - Camera moving over a stationary scene
    - Most projected camera motions can be captured by affine mapping!
  - The scene moves in its entirety --- a rare event!
  - Typically, the scene can be decomposed into several major regions, each moving differently (region-based motion estimation)
- If there is indeed a global motion, or the region undergoing a coherent motion has been determined, we can determine the motion parameters
  - Direct estimation
  - Indirect estimation
- When a majority of pixels (but not all) undergo a coherent motion, one can iteratively determine the motion parameters and the pixels undergoing this motion
  - Robust estimator



Direct Estimation

- Parameterize the DFD error in terms of the motion parameters, and estimate these parameters by minimizing the DFD error

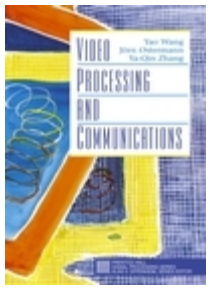
$$E_{\text{DFD}} = \sum_{n \in \mathcal{N}} w_n |\psi_2(\mathbf{x}_n + \mathbf{d}(\mathbf{x}_n; \mathbf{a})) - \psi_1(\mathbf{x}_n)|^p$$

Weighting w_n coefficients depend on the importance of pixel $\mathbf{x}_n$.

Ex: Affine motion:

$$\begin{bmatrix} d_x(\mathbf{x}_n;\mathbf{a}) \\ d_y(\mathbf{x}_n;\mathbf{a}) \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x_n + a_2 y_n \\ b_0 + b_1 x_n + b_2 y_n \end{bmatrix}, \quad \mathbf{a} = [a_0, a_1, a_2, b_0, b_1, b_2]^T$$

Exhaustive search or gradient descent method can be used to find **a** that minimizes E_{DFD}



Indirect Estimation

- First find the dense motion field using pixel-based or block-based approach (e.g. EBMA)
- Then parameterize the resulting motion field using the motion model through least squares fitting

$$E_{fit} = \sum w_n (\mathbf{d}(\mathbf{x}_n;\mathbf{a}) - \mathbf{d}_n)^2$$

Affine motion :

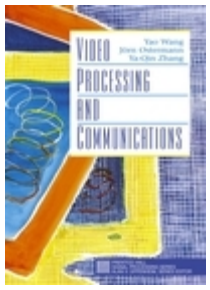
$$\mathbf{d}(\mathbf{x}_n;\mathbf{a}) = [\mathbf{A}_n]\mathbf{a},$$

$$[\mathbf{A}_n] = \begin{bmatrix} 1 & x_n & y_n & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_n & y_n \end{bmatrix}$$

$$\frac{\partial E_{fit}}{\partial \mathbf{a}} = \sum w_n [\mathbf{A}_n]^T ([\mathbf{A}_n]\mathbf{a} - \mathbf{d}_n) = 0$$

$$\mathbf{a} = \left(\sum w_n [\mathbf{A}_n]^T [\mathbf{A}_n]\right)^{-1} \left(\sum w_n [\mathbf{A}_n]^T \mathbf{d}_n\right)$$

Weighting w_n coefficients depend on the accuracy of estimated motion at $\mathbf{x}_n$.



Robust Estimator

Essence: iteratively removing "outlier" pixels.

1. Set the region to include all pixels in a frame
2. Apply the direct or indirect method method over all pixels in the region
3. Evaluate errors (E_{DFD} or E_{fit}) at all pixels in the region
4. Eliminate "outlier" pixels with large errors
5. Repeat steps 2-4 for the remaining pixels in the region

Details: Hard threshold vs. soft threshold. See textbook.

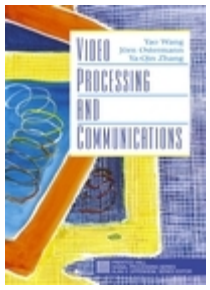
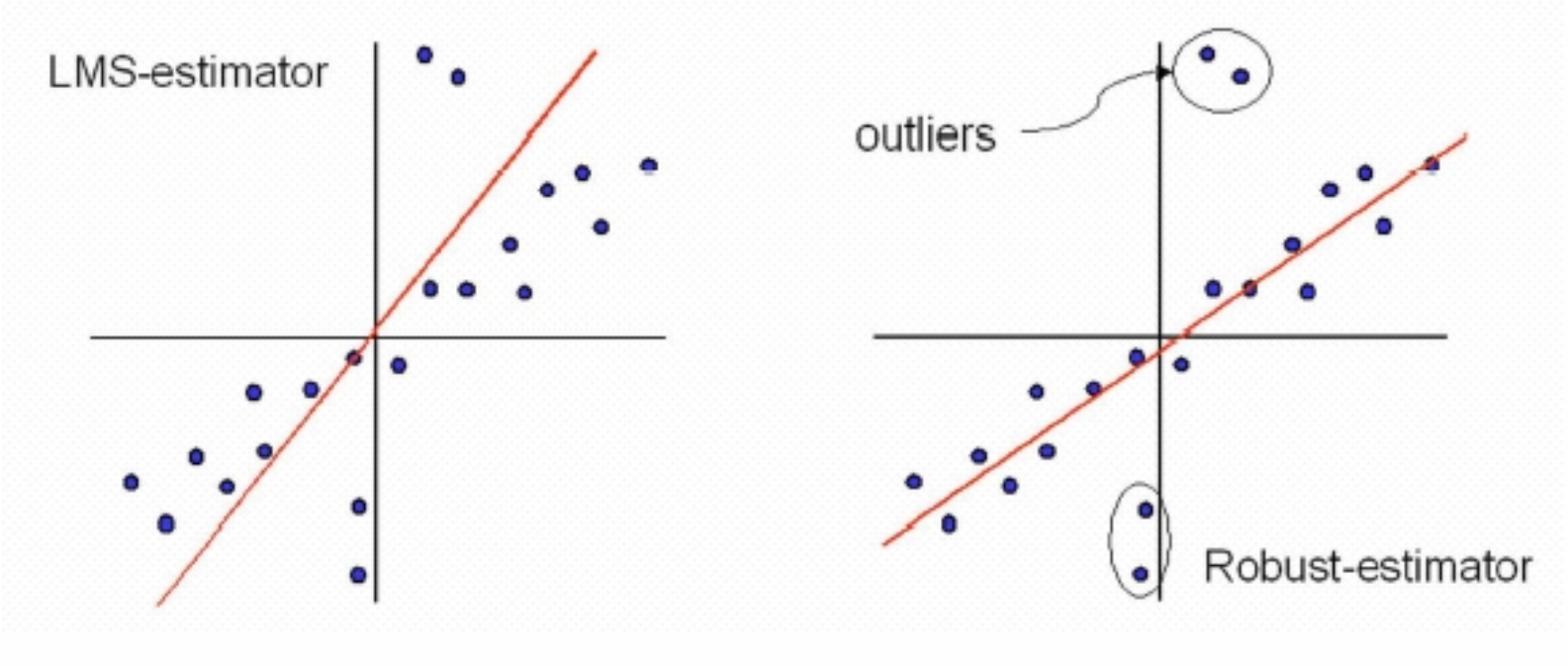
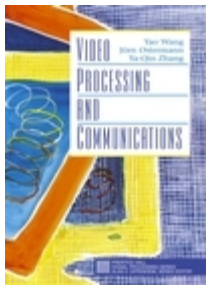


Illustration of Robust Estimator

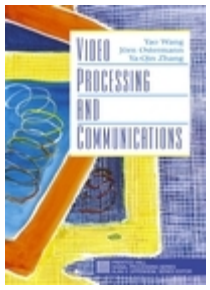


Fitting a line to the data points by using LMS and robust estimators. Courtesy of Fatih Porikli



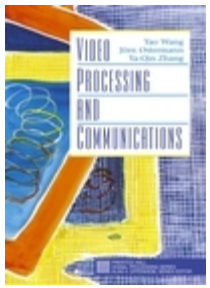
Region-Based Motion Estimation

- Assumption: the scene consists of multiple objects, with the region corresponding to each object (or sub-object) having a coherent motion
  - Physically more correct than block-based, mesh-based, global motion model
- Method:
  - Region First: Segment the frame into multiple regions based on texture/edges, then estimate motion in each region using the global motion estimation method
  - Motion First: Estimate a dense motion field, then segment the motion field so that motion in each region can be accurately modeled by a single set of parameters
  - Joint region-segmentation and motion estimation: iterate the two processes



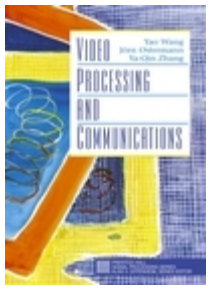
Summary

- Fundamentals:
  - Optical flow equation
    - Derived from constant intensity and small motion assumption
    - Ambiguity in motion estimation
  - How to represent motion:
    - Pixel-based, block-based, region-based, global, etc.
  - Estimation criterion:
    - DFD (constant intensity)
    - OF (constant intensity+small motion)
    - Bayesian (MAP, DFD+motion smoothness)
  - Search method:
    - Exhaustive search, gradient-descent, multi-resolution



Summary (Cntd)

- Basic techniques:
  - Pixel-based motion estimation
  - Block-based motion estimation
    - EBMA, integer-pel vs. half-pel accuracy, Fast algorithms
- More advanced techniques
  - Multiresolution approach
    - Avoid local minima, smooth motion field, reduced computation
  - Deformable block matching algorithm (DBMA)
    - To allow more complex motion within each block
  - Mesh-based motion estimation
    - To enforce continuity of motion across block boundaries
  - Global motion estimation
    - Good for estimating camera motion
  - Region-based motion estimation
    - More physically correct: allow different motion in each sub-object region
- Application in Video Coding



Homework

- Reading assignment
  - Read Secs. 6.5-6.10
  - Go through and verify the gradient descent algorithm presented for DBMA (Eqs. 6.5.2-6.5.6).
  - Go through the derivation of the objective function definition (Eq. 6.6.6-6.6.8) for mesh-based motion estimation carefully, and verify the gradient function given in Eq. 6.6.9.
- Assignment
  - Prob. 6.9, 6.10, 6.16, 6.15 (Computer assignment)
- Optional computer assignment
  - Assuming the motion between two frames can be approximated by an affine mapping,determine the affine parameters using the indirect method. First apply the HBMA (or EBMA) algorithm you implemented, to determine a block-wise motion field between two frames. Then determine the affine parameters using the weighted least squares method (Eq. 6.7.3). Show the predicted image based on the affine parameters and the associated prediction error (in terms of PSNR). Compared them to those obtained with the original block-based motion estimation. Note: You should apply you algorithm to two video frames experiencing predominantly camera motion. To test the accuracy of your algorithm, you may want to artificially generate a pair of frames, where one frame is the affine mapping of another.
  - Implement the direct method (Prob. 6.17), and compare the results.