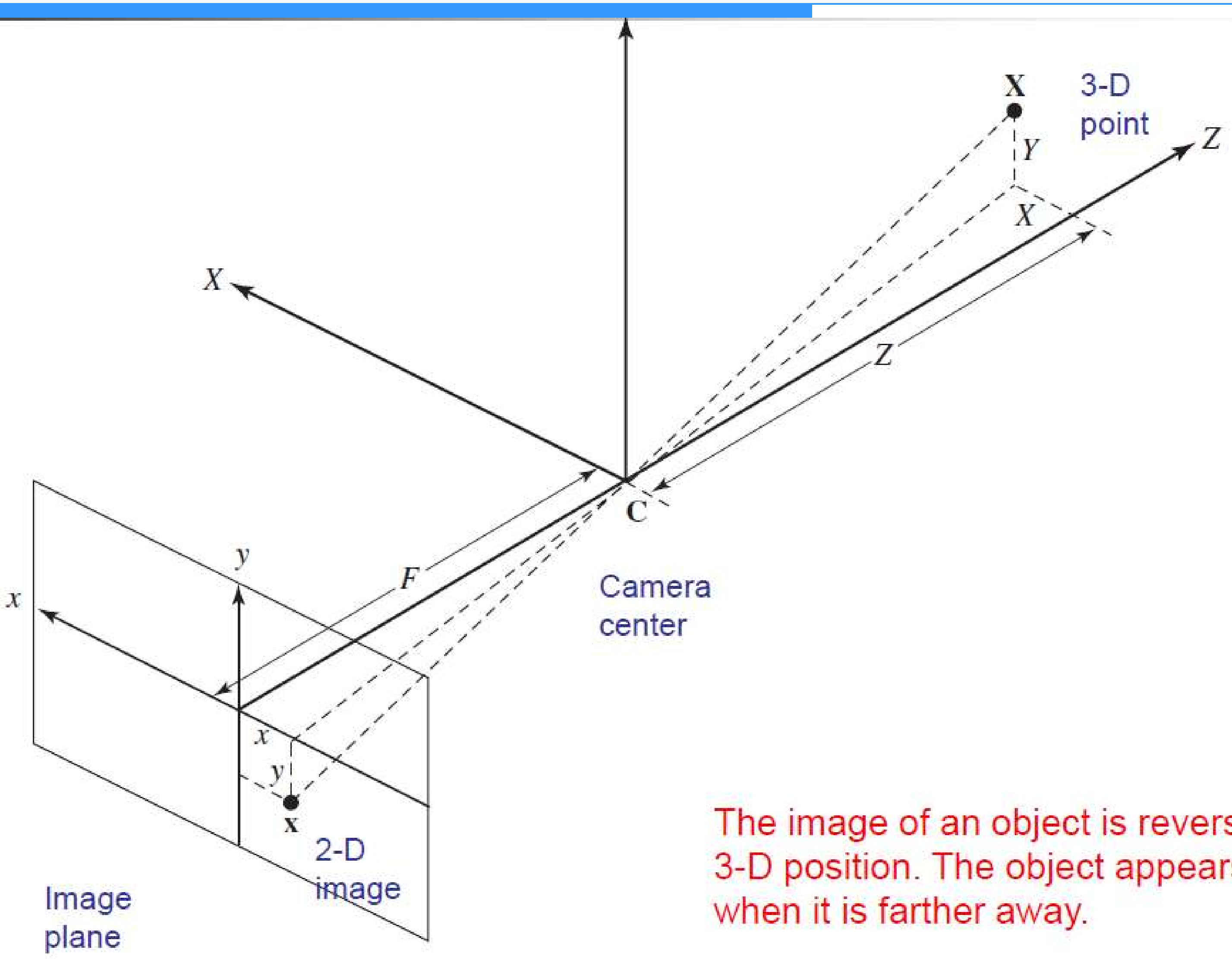
第13章 运动分析（Motion Analysis）

- ☐ 2-D motion model
- ☐ 2-D motion vs. optical flow
- ☐ Optical flow equation and ambiguity in motion estimation
- ☐ General methodologies in motion estimation
- ☐ Pixel-based motion estimation
- ☐ Block-based motion estimation
- ☐ Multiresolution motion estimation
- ☐ Phase Correlation Method
- ☐ Global motion estimation
- ☐ Region-based motion estimation
- ☐ Motion Segmentation

13.1 2-D Motion Model

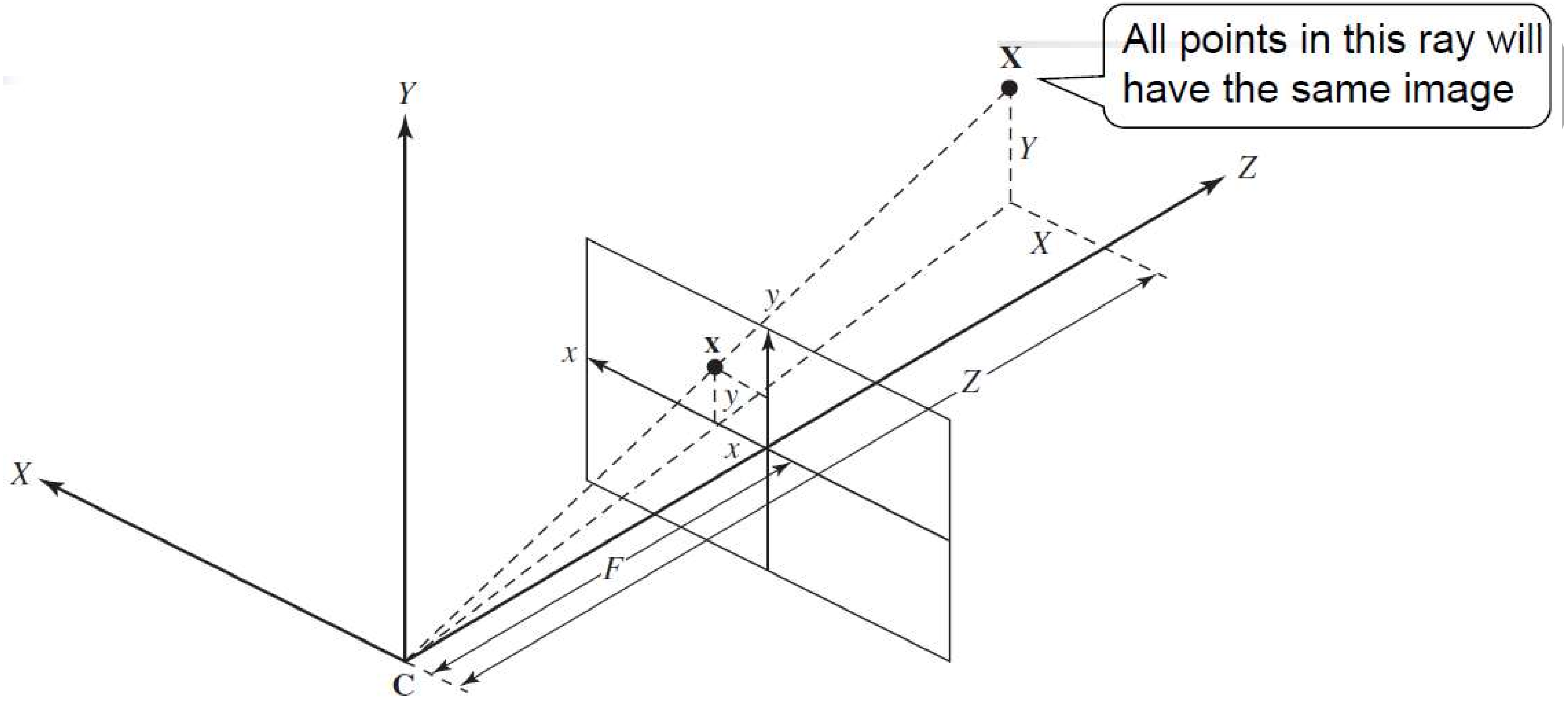
- Camera projection
- 3D motion
- Projection of 3-D motion
- 2D motion due to rigid object motion
  - Projective mapping
- Approximation of projective mapping
  - Affine model
  - Bilinear model

Pinhole Camera



The image of an object is reversed from its 3-D position. The object appears smaller when it is farther away.

Pinhole Camera Model: Perspective Projection

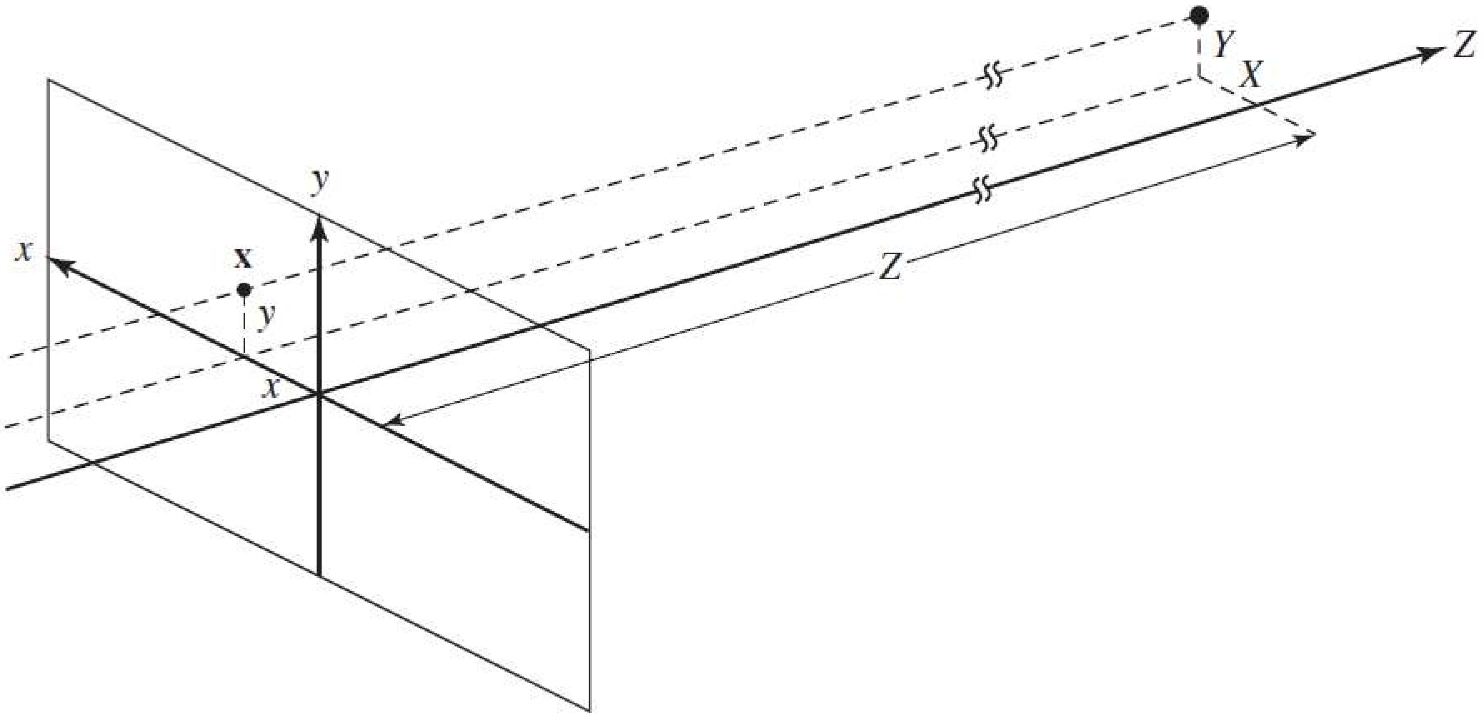


(a)

$$\frac{x}{F}=\frac{X}{Z},\frac{y}{F}=\frac{Y}{Z}\Rightarrow x=F\frac{X}{Z},y=F\frac{Y}{Z}$$

x, y are inversely related to Z

Approximate Model: Orthographic Projection



(b)

When the object is very far $(Z \to \infty)$

$x = X, y = Y$

Can be used as long as the depth variation within the object is small compared to the distance of the object.

3D Motion Model

Translation

$$X' = X + T$$

Rotation

$$[R] = [R_z] \bullet [R_y] \bullet [R_x]$$

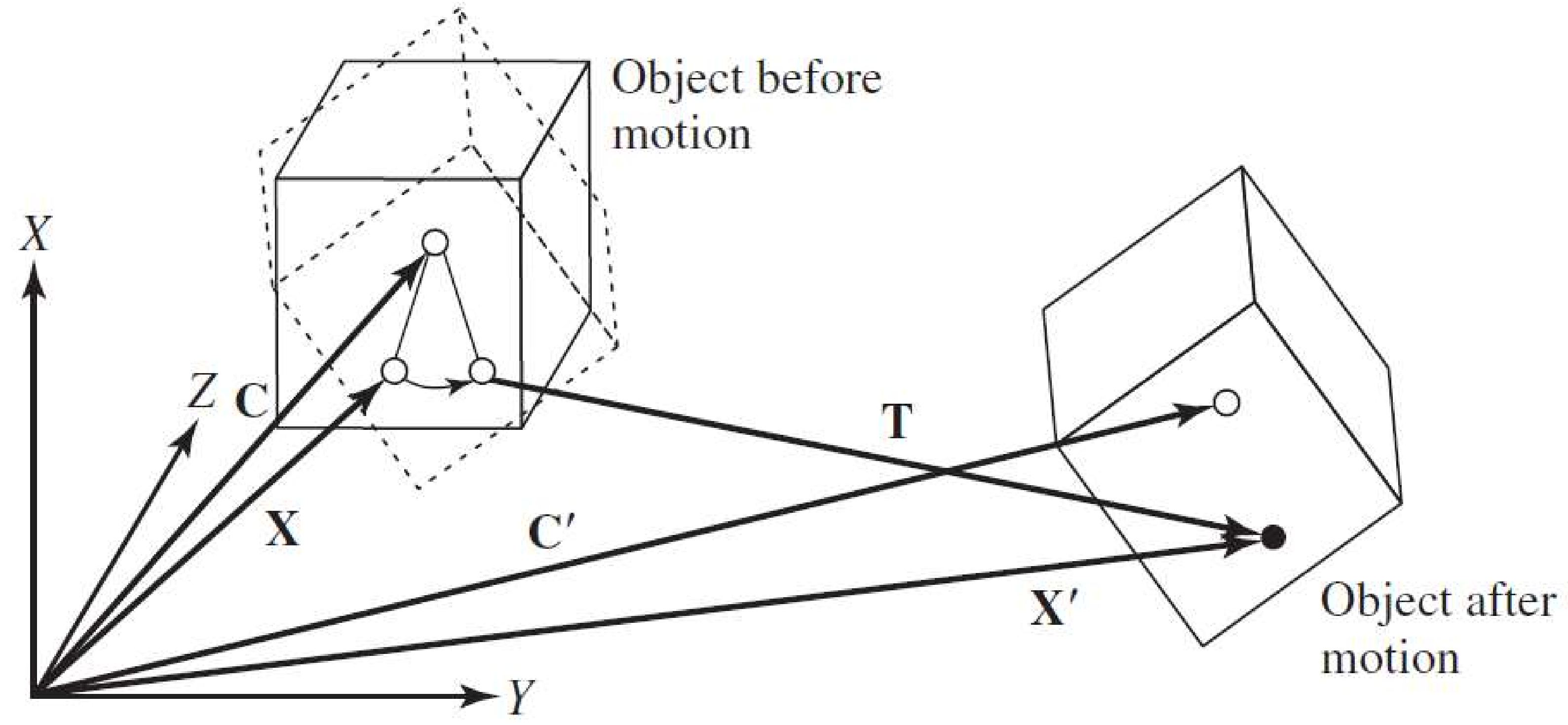
$$[R_x] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix}$$

$$[R_y] = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix}$$

$$[R_z] = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[R] \approx [R'] = \begin{bmatrix} 1 & -\theta_z & \theta_y \\ \theta_z & 1 & -\theta_x \\ -\theta_y & \theta_x & 1 \end{bmatrix}$$

Rigid Object Motion



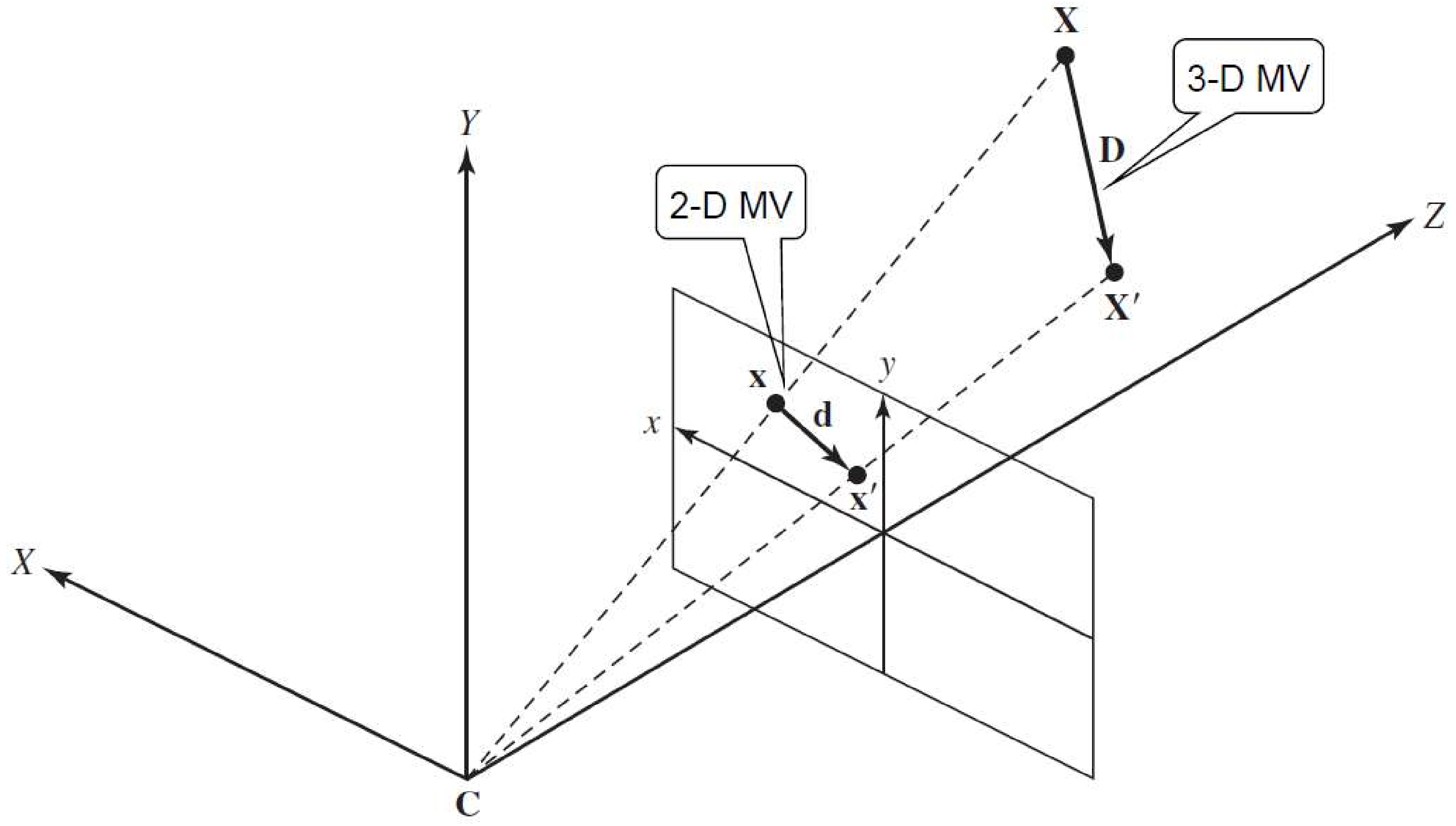
Rotation and translation wrt. the object center:

$X' = [R](X - C) + T + C; \quad [R]: \theta_x, \theta_y, \theta_z; \quad T: T_x, T_y, T_z$

Flexible Object Motion

- Two ways to describe
  - Decompose into multiple, but connected rigid sub-objects
  - Global motion plus local motion in sub-objects
    - ✓ Eg., human body consists of many parts each undergo a rigid motion

3-D Motion -> 2-D Motion



MV: motion vector

Definition and Notation

3D motion vector

$$D(X;t_1,t_2) = X' - X = [D_X, D_Y, D_Z]^T$$

2D motion vector

$$d(x;t_1,t_2) = x' - x = [d_x, d_y]^T$$

Mapping function

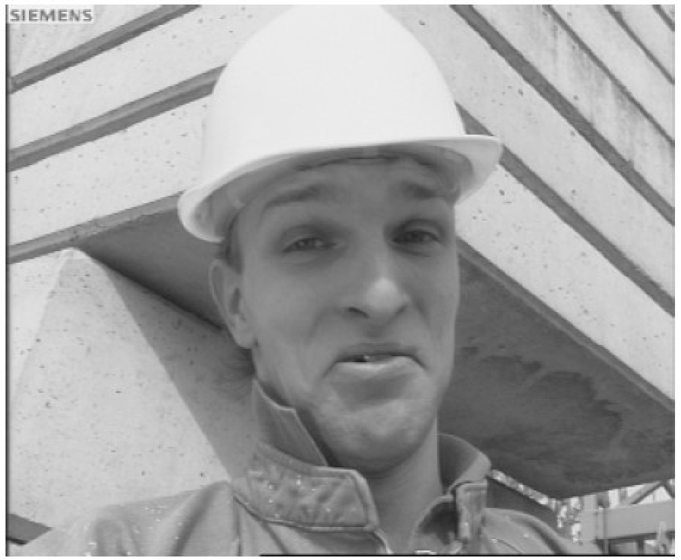
$$w(x;t_1,t_2) = x'$$

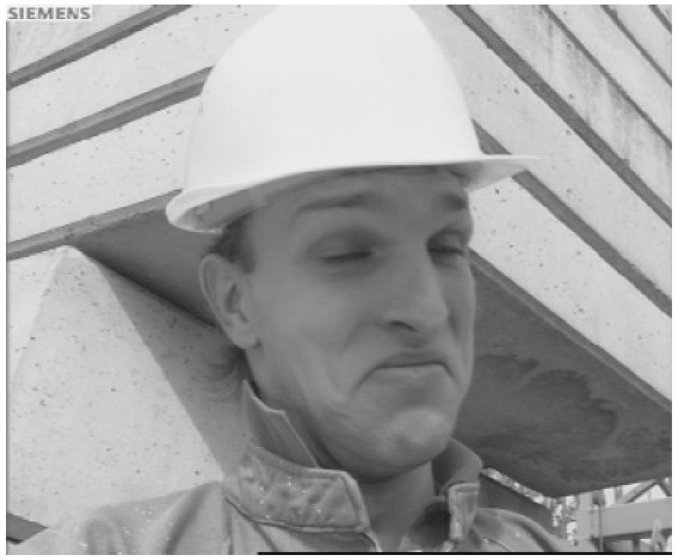
$$w(x) = x + d(x)$$

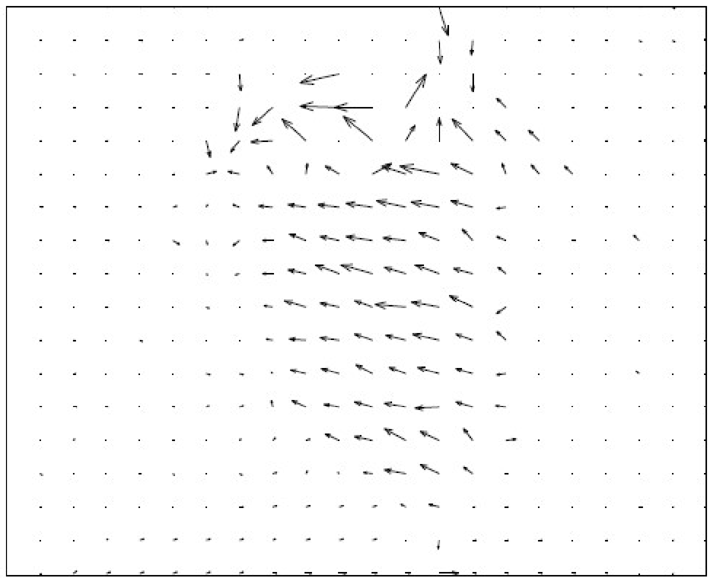
Flow vector

$$V = \frac{\partial d}{\partial t} = \left[\frac{\partial d_x}{\partial t}, \frac{\partial d_y}{\partial t}\right]^T$$

Sample of Motion Field

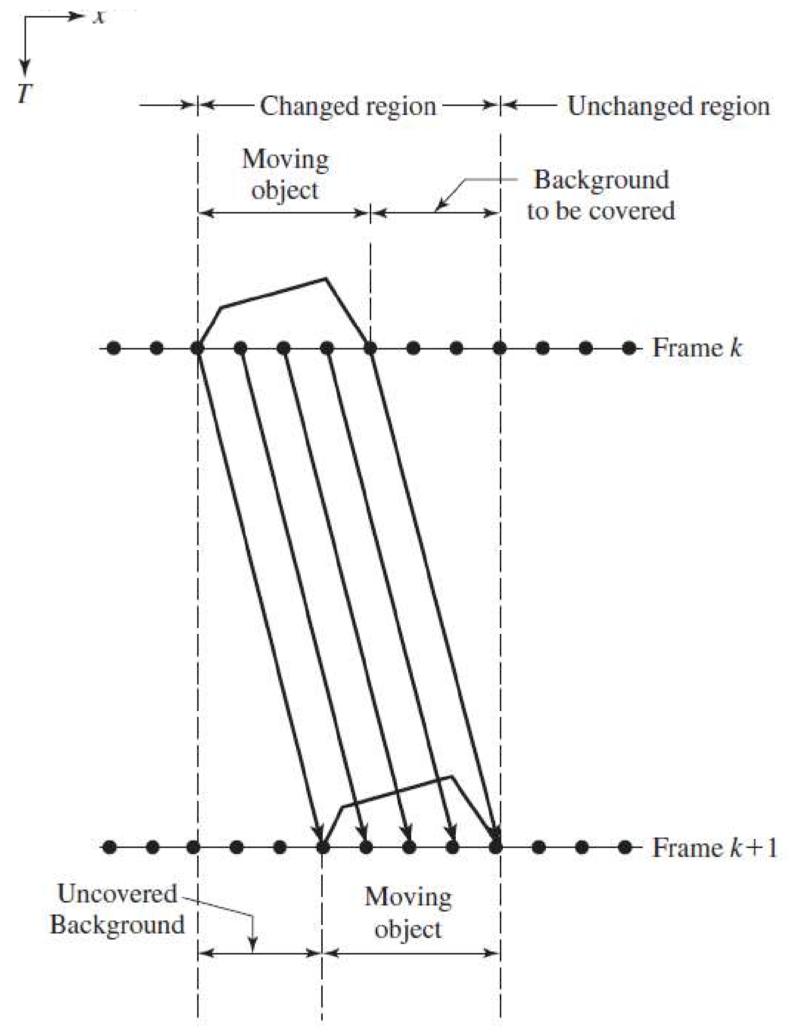




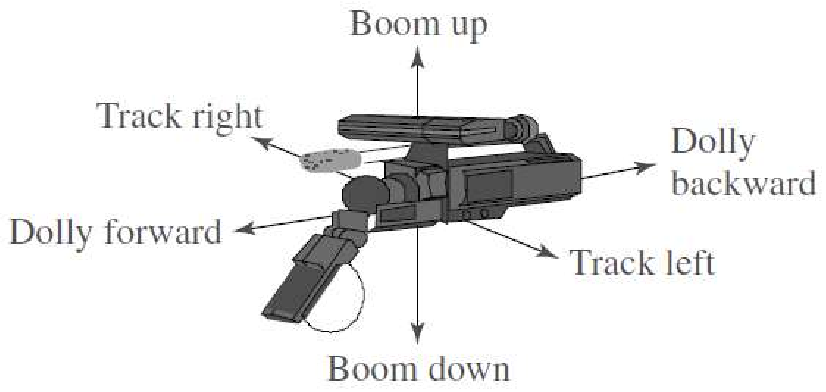


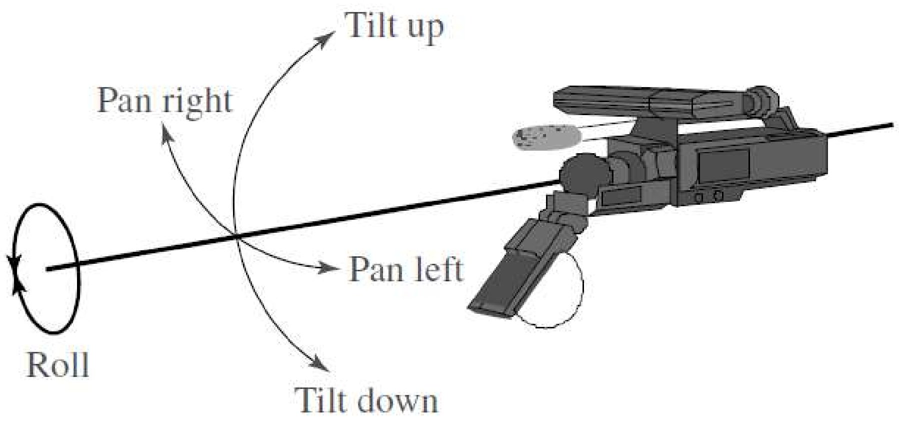
Occlusion Effect

Motion is undefined in occluded regions



Typical Camera Motions





Camera Translation (Track and Boom)

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ 0 \end{bmatrix} \Leftrightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} FT_x / Z \\ FT_y / Z \end{bmatrix}$$

When $\Delta Z \ll \bar{Z}$

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}, t_x = \frac{FT_x}{\bar{Z}}, t_y = \frac{FT_y}{\bar{Z}}$$

Camera Pan and Tilt

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = [R_x][R_y] \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad [R_x][R_y] = \begin{bmatrix} 1 & 0 & \theta_y \\ 0 & 1 & -\theta_x \\ -\theta_y & \theta_x & 1 \end{bmatrix}$$

If $Y\theta_x \ll Z, X\theta_y \ll Z$, then $Z' \approx Z$

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} \theta_y F \\ -\theta_x F \end{bmatrix}$$

Camera Zoom and Roll

Zoom

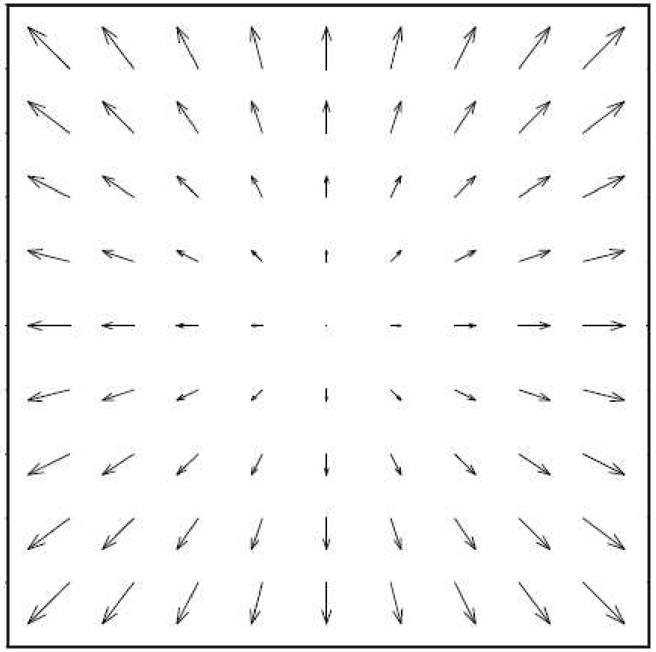
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \rho x \\ \rho y \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} (1-\rho)x \\ (1-\rho)y \end{bmatrix} \qquad (\rho = F'/F)$$

Roll

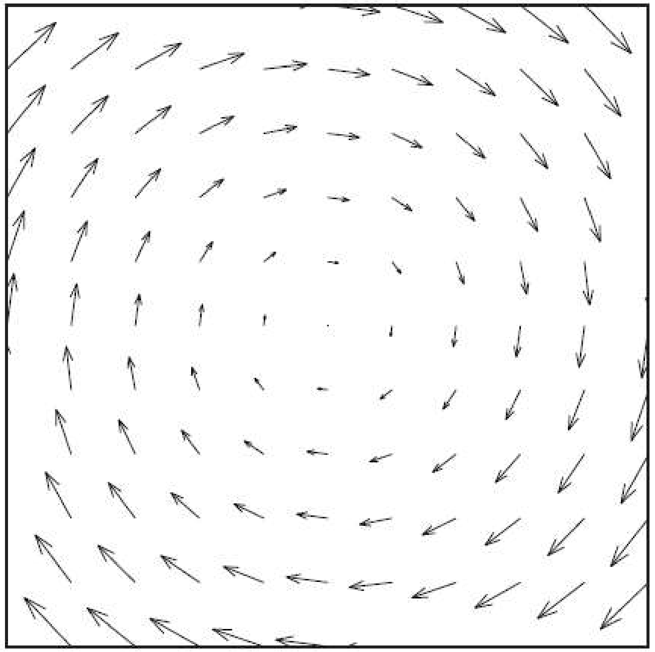
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta_z & -\sin\theta_z \\ \sin\theta_z & \cos\theta_z \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \approx \begin{bmatrix} 1 & -\theta_z \\ \theta_z & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} -\theta_z y \\ \theta_z x \end{bmatrix}$$

2-D Motion Corresponding to Camera Motion


(a)

Camera zoom


(b)

Camera rotation around Z-axis (roll)

Four-Parameter Model

Consider a camera that goes through translation, pan, tilt, zoom, and rotation in sequence

Geometric Mapping

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \rho \begin{bmatrix} \cos\theta_z & -\sin\theta_z \\ \sin\theta_z & \cos\theta_z \end{bmatrix} \begin{bmatrix} x + \theta_y F + t_x \\ y - \theta_x F + t_y \end{bmatrix}$$

$$= \begin{bmatrix} c_1 & -c_2 \\ c_2 & c_1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c_3 \\ c_4 \end{bmatrix}$$

This mapping function has four parameters, and is a special case of the affine mapping, which has 6 parameters.

2-D Motion Corresponding to Rigid Object Motion

- General case:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

$$\xrightarrow[\text{Perspective Projection}]{}$$

$$x' = F\frac{(r_1x + r_2y + r_3F)Z + T_xF}{(r_7x + r_8y + r_9F)Z + T_zF}$$

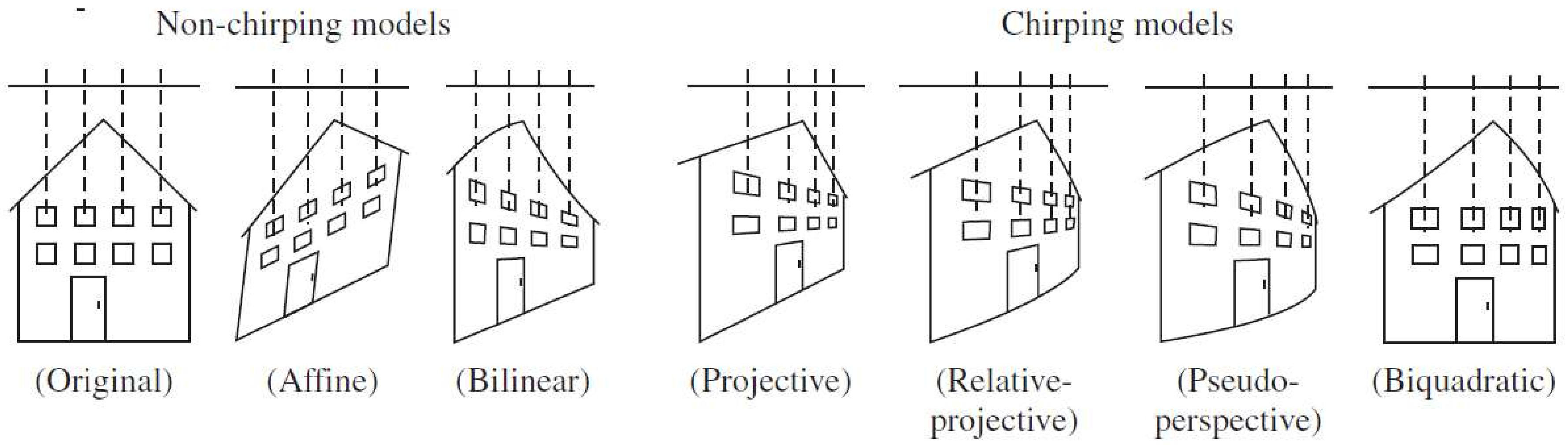
$$y' = F\frac{(r_4x + r_5y + r_6F)Z + T_yF}{(r_7x + r_8y + r_9F)Z + T_zF}$$

- Projective mapping:

When the object surface is planar $(Z = aX + bY + c)$:

$$x' = \frac{a_0 + a_1x + a_2y}{1 + c_1x + c_2y}, \quad y' = \frac{b_0 + b_1x + b_2y}{1 + c_1x + c_2y}$$

Projective Mapping



Two features of projective mapping:

- Chirping: increasing perceived spatial frequency for far away objects
- Converging (Keystone): parallel lines converge in distance

Affine and Bilinear Model

- Affine (6 parameters):

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

  - Good for mapping triangles to triangles

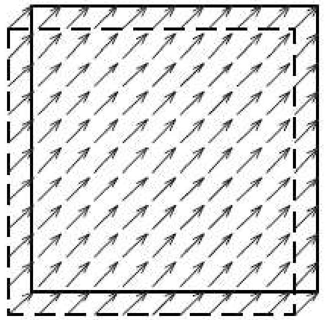
- Bilinear (8 parameters):

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

  - Good for mapping blocks to quadrangles

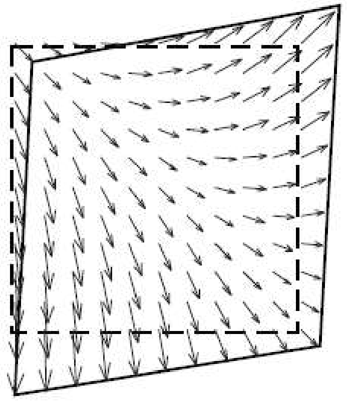
Motion Field Corresponding to Different 2-D Motion Models

Translation



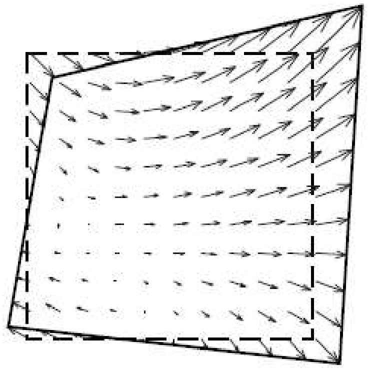
(a)

Affine



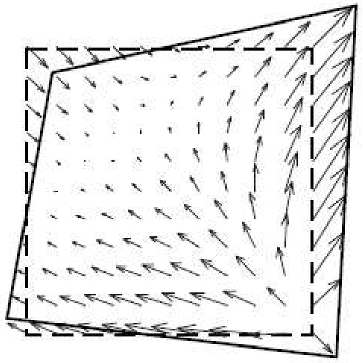
(b)

Bilinear



(c)

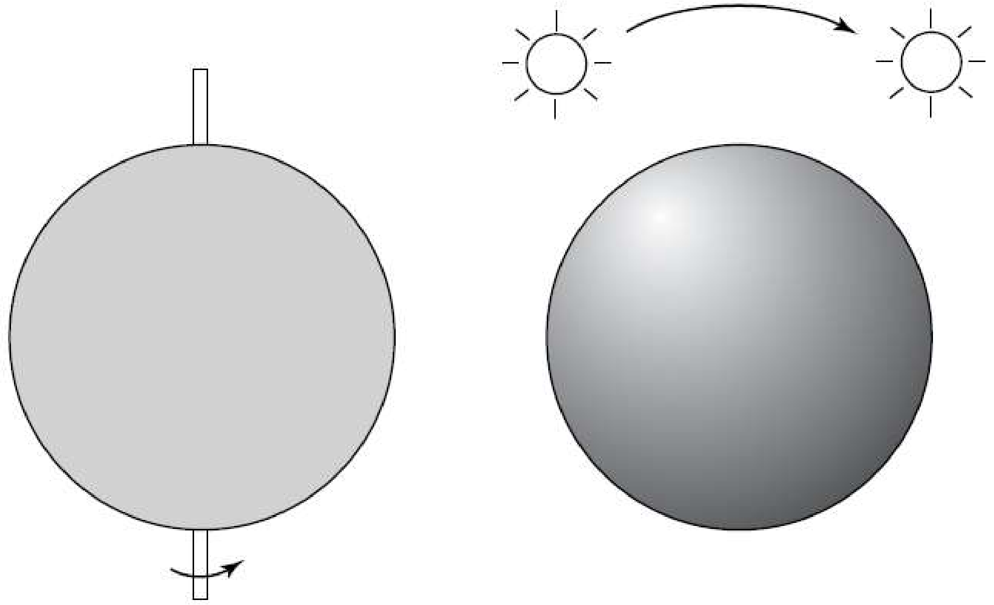
Perspective



(d)

13.2 2-D Motion vs. Optical Flow

- 2-D Motion: Projection of 3-D motion, depending on 3D object motion and projection operator
- Optical flow: "Perceived" 2-D motion based on changes in image pattern, also depends on illumination and object surface texture



On the left, a sphere is rotating under a constant ambient illumination, but the observed image does not change.

On the right, a point light source is rotating around a stationary sphere, causing the highlight point on the sphere to rotate.

13.3 Optical Flow Equation

- When illumination condition is unknown, the best one can do it is to estimate optical flow.
- Constant intensity assumption -> Optical flow equation

Under "constant intensity assumption":

$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t)$$

But, using Taylor's expansion :

$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t) + \frac{\partial \psi}{\partial x} d_x + \frac{\partial \psi}{\partial y} d_y + \frac{\partial \psi}{\partial y} d_t$$

Compare the above two, we have the optical flow equation :

$$\frac{\partial \psi}{\partial x} d_x + \frac{\partial \psi}{\partial y} d_y + \frac{\partial \psi}{\partial t} d_t = 0 \quad \text{or} \quad \frac{\partial \psi}{\partial x} v_x + \frac{\partial \psi}{\partial y} v_y + \frac{\partial \psi}{\partial t} = 0 \ \text{ or } \ \nabla \psi^T \mathbf{v} + \frac{\partial \psi}{\partial t} = 0$$

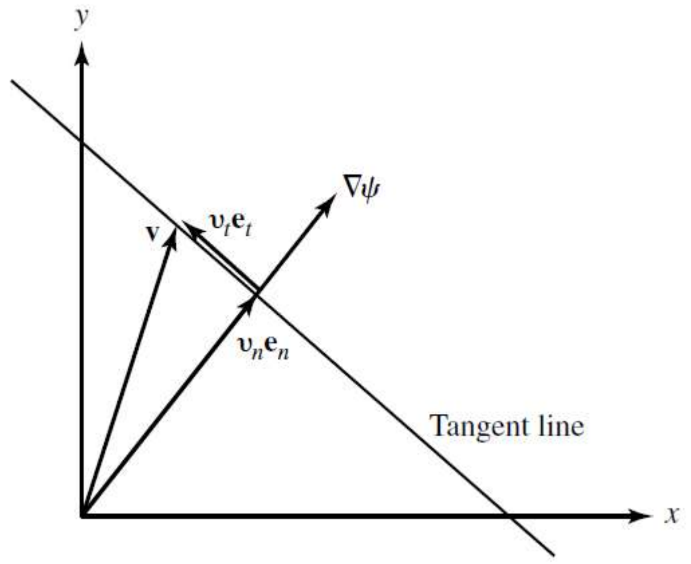
How to use the optical equation

$$\begin{aligned} f_t \approx{} & \frac{1}{4}\left[f(x,y,t+1)+f(x+1,y,t+1)+f(x,y+1,t+1)+f(x+1,y+1,t+1)\right] \\ & -\frac{1}{4}\left[f(x,y,t)+f(x+1,y,t)+f(x,y+1,t)+f(x+1,y+1,t)\right] \end{aligned}$$

$$\begin{aligned} f_x \approx{} & \frac{1}{4}\left[f(x+1,y,t)+f(x+1,y+1,t)+f(x+1,y,t+1)+f(x+1,y+1,t+1)\right] \\ & -\frac{1}{4}\left[f(x,y,t)+f(x,y+1,t)+f(x,y,t+1)+f(x,y+1,t+1)\right] \end{aligned}$$

Ambiguities in Motion Estimation

- Optical flow equation only constrains the flow vector in the gradient direction v_n
- The flow vector in the tangent direction (v_t) is under-determined
- In regions with constant brightness ($\nabla \psi = 0$), the flow is indeterminate
  - Motion estimation is unreliable in regions with flat texture, more reliable near edges

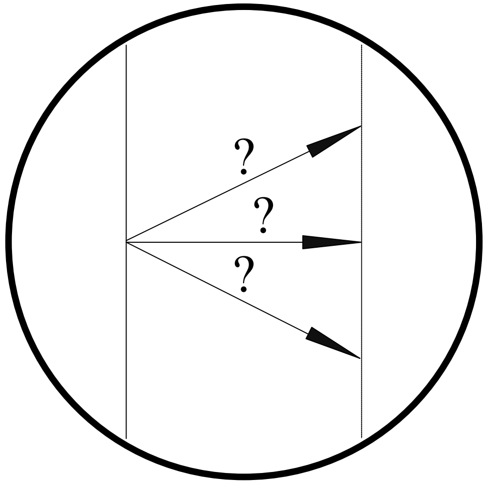


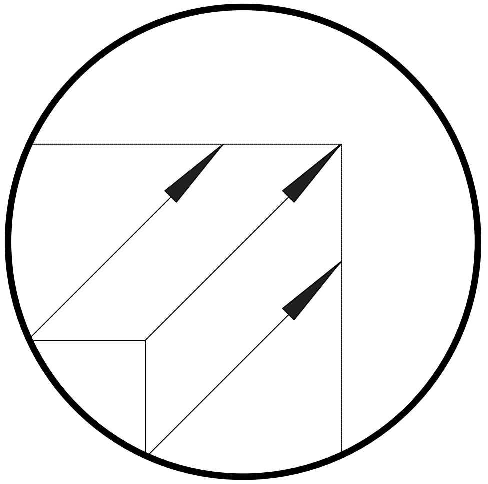
$$\nabla\psi^T\mathbf{v} + \frac{\partial\psi}{\partial t} = 0$$

$$\mathbf{v} = v_n\mathbf{e}_n + v_t\mathbf{e}_t$$

$$v_n \left\|\nabla \psi\right\| + \frac{\partial \psi}{\partial t} = 0$$

The Aperture Problem



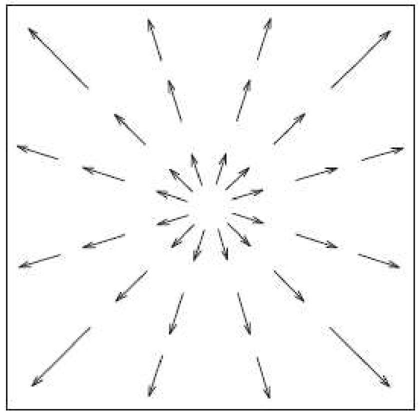


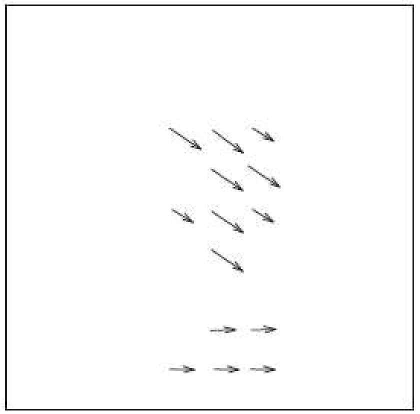
13.4 General Considerations for Motion Estimation

- Two categories of approaches:
  - Feature based (more often used in object tracking, 3D reconstruction from 2D)
  - Intensity based (based on constant intensity assumption) (more often used for motion compensated prediction, required in video coding, frame interpolation) -> Our focus
- Three important questions
  - How to represent the motion field?
  - What criteria to use to estimate motion parameters?
  - How to search motion parameters?

Motion Representation

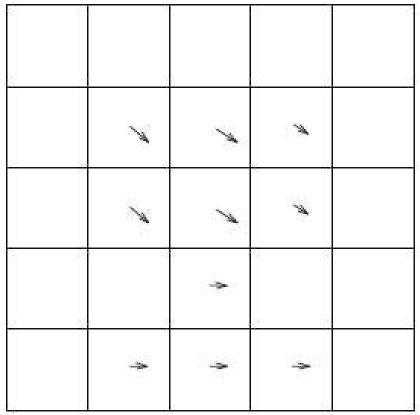
Global:
Entire motion field is represented by a few global parameters

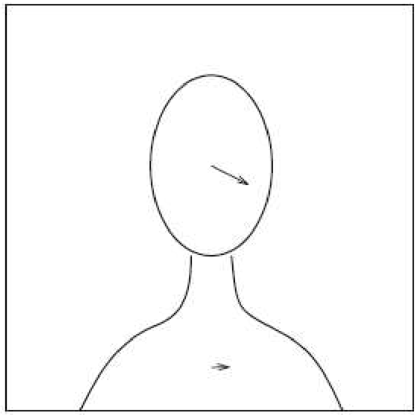

(a)


(b)

Pixel-based:
One MV at each pixel, with some smoothness constraint between adjacent MVs.

Block-based:
Entire frame is divided into blocks, and motion in each block is characterized by a few parameters.

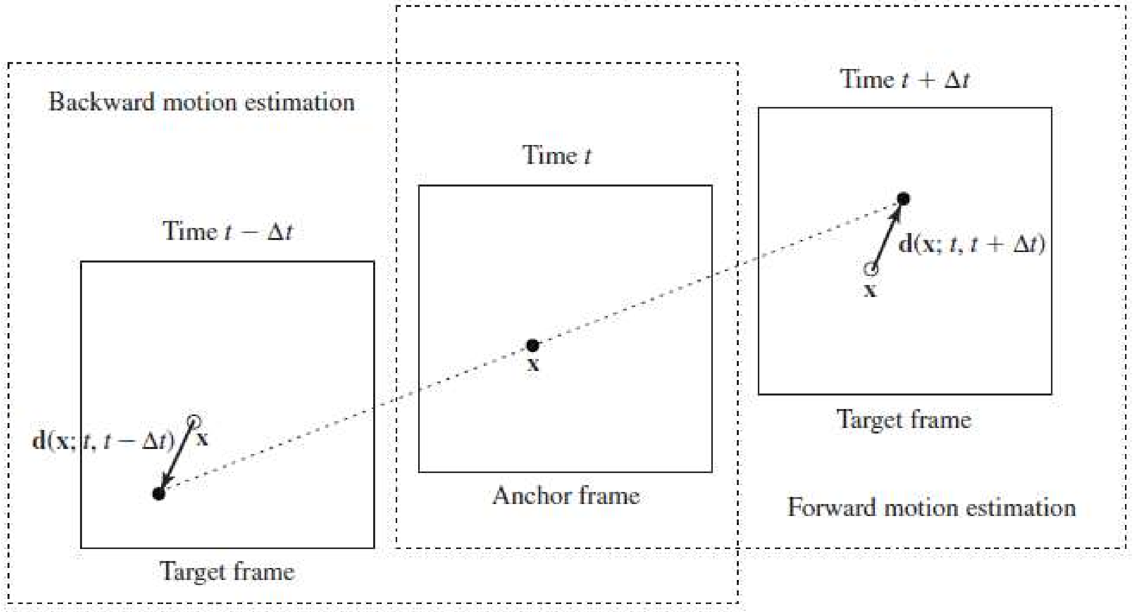

(c)


(d)

Region-based:
Entire frame is divided into regions, each region corresponding to an object or sub-object with consistent motion, represented by a few parameters.

Other representation: mesh-based (control grid) (to be discussed later)

Notations



Backward motion estimation
Time $t - \Delta t$
Time t
Time $t + \Delta t$
$\mathbf{d}(\mathbf{x}; t, t - \Delta t)$
$\mathbf{x}$
$\mathbf{x}$
$\mathbf{d}(\mathbf{x}; t, t + \Delta t)$
$\mathbf{x}$
Target frame
Anchor frame
Target frame
Forward motion estimation

Anchor frame: $\psi_1(\mathbf{x})$

Target frame: $\psi_2(\mathbf{x})$

Motion parameters: $\mathbf{a}$

Motion vector at a pixel in the anchor frame: $\mathbf{d}(\mathbf{x})$

Motion field: $\mathbf{d}(\mathbf{x};\mathbf{a}), \mathbf{x} \in \Lambda$

Mapping function:

$$\mathbf{w}(\mathbf{x};\mathbf{a}) = \mathbf{x} + \mathbf{d}(\mathbf{x};\mathbf{a}), \mathbf{x} \in \Lambda$$

displaced frame difference (DFD)

$$E_{\mathrm{DFD}}(\mathbf{a}) = \sum_{x \in \Lambda} \left| \psi_2(\mathbf{x} + \mathbf{d}(\mathbf{x};\mathbf{a})) - \psi_1(\mathbf{x}) \right|^p$$

Motion Estimation Criterion

- To minimize the displaced frame difference (DFD)

$$E_{\mathrm{DFD}}(\mathbf{a}) = \sum_{x\in\Lambda} \left|\psi_2(\mathbf{x}+\mathbf{d}(\mathbf{x};\mathbf{a})) - \psi_1(\mathbf{x})\right|^p \rightarrow \min$$

$$p = 1 : \mathrm{MAD}; \quad P = 2 : \mathrm{MSE}$$

- To satisfy the optical flow equation

$$E_{\mathrm{OF}}(\mathbf{a}) = \sum_{x\in\Lambda} \left|(\nabla\psi_2(\mathbf{x}))^T \mathbf{d}(\mathbf{x};\mathbf{a}) + \psi_2(\mathbf{x}) - \psi_1(\mathbf{x})\right|^p \rightarrow \min$$

- To impose additional smoothness constraint using regularization technique (important in pixel- and block-based representation)

$$E_s(\mathbf{a}) = \sum_{\mathbf{x}\in\Lambda} \sum_{\mathbf{y}\in N_x} \left\|\mathbf{d}(\mathbf{x};\mathbf{a}) - \mathbf{d}(\mathbf{y};\mathbf{a})\right\|^2$$

$$w_{DFD} E_{\mathrm{DFD}}(\mathbf{a}) + w_s E_s(\mathbf{a}) \rightarrow \min$$

- Bayesian (MAP) criterion: to maximize the a posteriori probability

$$P(D = \mathbf{d} | \psi_2, \psi_1) \rightarrow \max$$

Relation Among Different Criteria

- OF criterion is good only if motion is small.
- OF criterion can often yield closed-form solution as the objective function is quadratic in MVs.
- When the motion is not small, can iterate the solution based on the OF criterion to satisfy the DFD criterion.
- Bayesian criterion can be reduced to the DFD criterion plus motion smoothness constraint

Optimization Methods

- Exhaustive search
  - Typically used for the DFD criterion with p=1 (MAD)
  - Guarantees reaching the **global** optimal
  - Computation required may be unacceptable when number of parameters to search simultaneously is large!
  - Fast search algorithms reach sub-optimal solution in shorter time
- Gradient-based search
  - Typically used for the DFD or OF criterion with p=2 (MSE)
    - ✓ The gradient can often be calculated analytically
    - ✓ When used with the OF criterion, closed-form solution may be obtained
  - Reaches the local optimal point closest to the initial solution
- Multi-resolution search
  - Search from coarse to fine resolution, faster than exhaustive search
  - Avoid being trapped into a local minimum

13.5 Pixel-Based Motion Estimation

- Horn-Schunck method
  - OF + smoothness criterion
- Multipoint neighborhood method
  - Assuming every pixel in a small block surrounding a pixel has the same MV
- Pel-recursive method
  - MV for a current pel is updated from those of its previous pels, so that the MV does not need to be coded
  - Developed for early generation of video coder

Horn-Schunck method

OF + smoothness criterion

$$E(V(X)) = \sum_{X\in\Delta} \left(\frac{\partial\psi}{\partial x} v_x + \frac{\partial\psi}{\partial y} v_y + \frac{\partial\psi}{\partial t}\right)^2 + w_s\left(\left\|\nabla v_x\right\|^2 + \left\|\nabla v_y\right\|^2\right)$$

$$\nabla v_x = [v_x(x,y) - v_x(x-1,y), v_x(x,y) - v_x(x,y-1)]^T$$

$$\nabla v_y = [v_y(x,y) - v_y(x-1,y), v_y(x,y) - v_y(x,y-1)]^T$$

Note that the method used for calculating the gradient functions can have profound impact on the accuracy and robustness of the ME method

Gaussian pre-filter followed by a central difference generally leads to better results

Multipoint Neighborhood Method

- Estimate the MV at each pixel independently, by minimizing the DFD error over a neighborhood surrounding this pixel
- Every pixel in the neighborhood is assumed to have the same MV
- Minimizing function:

$$E_{\mathrm{DFD}}(\mathbf{d}_\mathrm{n}) = \sum_{\mathbf{x} \in B(\mathbf{x}_n)} w(\mathbf{x}) \left| \psi_2(\mathbf{x}+\mathbf{d}_n) - \psi_1(\mathbf{x}) \right|^2 \rightarrow \min$$

- Optimization method:
  - Exhaustive search (feasible as one only needs to search one MV at a time)
    - ✓ Need to select appropriate search range and search step-size
  - Gradient-based method

13.6 Block-Based Motion Estimation: Overview

- Assume all pixels in a block undergo a coherent motion, and search for the motion parameters for each block independently
- Block matching algorithm (BMA): assume translational motion, 1 MV per block (2 parameter)
  - Exhaustive BMA (EBMA)
  - Fast algorithms
- Deformable block matching algorithm (DBMA): allow more complex motion (affine, bilinear)

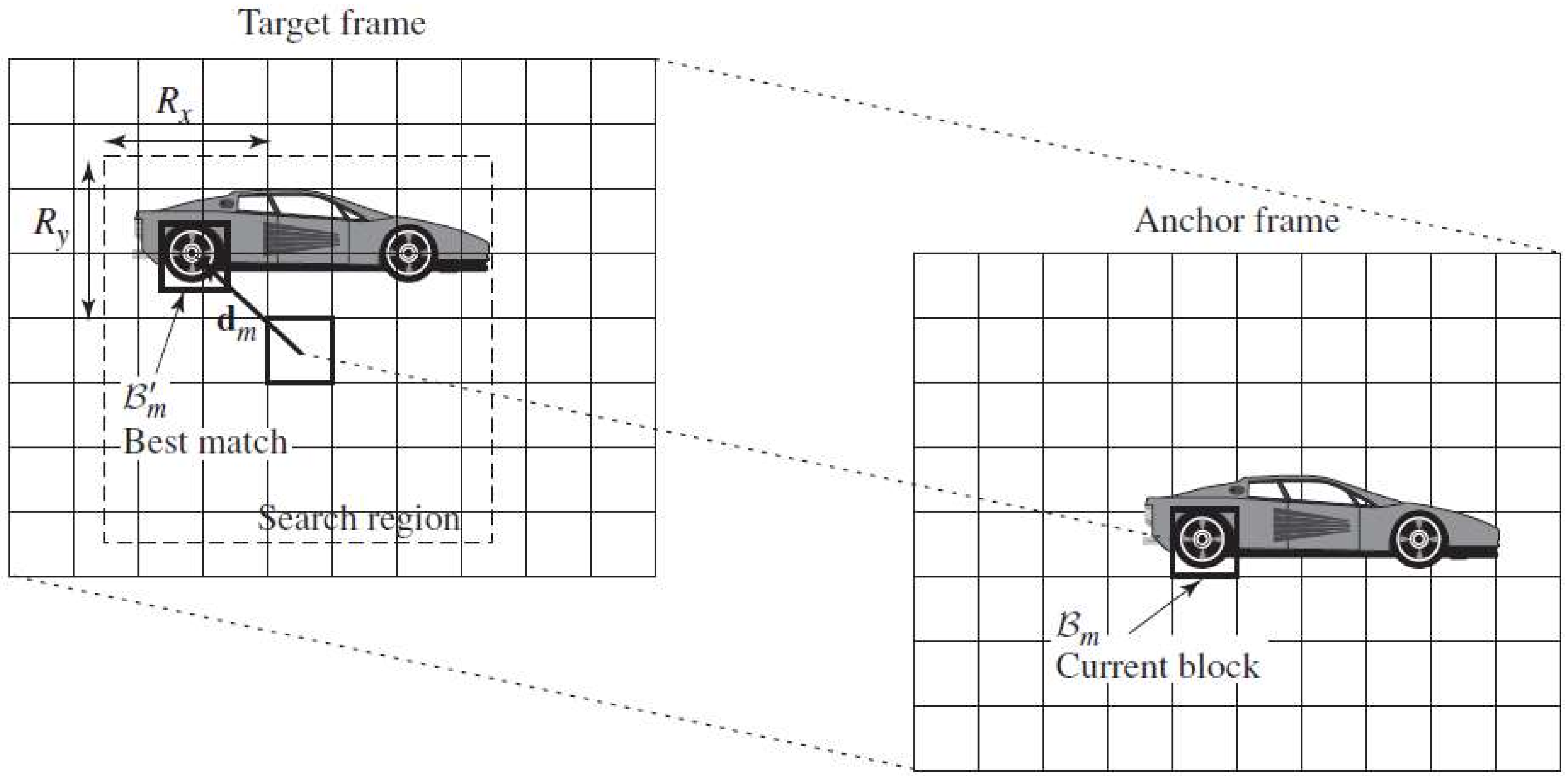
Block Matching Algorithm

- Overview:
  - Assume all pixels in a block undergo a translation, denoted by a single MV
  - Estimate the MV for each block independently, by minimizing the DFD error over this block
- Minimizing function:

$$E_{\mathrm{DFD}}(\mathbf{d}_{\mathrm{m}}) = \sum_{\mathbf{x}\in B_m} \left|\psi_2(\mathbf{x}+\mathbf{d}_m) - \psi_1(\mathbf{x})\right|^p \rightarrow \min$$

- Optimization method:
  - Exhaustive search (feasible as one only needs to search one MV at a time), using MAD criterion (p=1)
  - Fast search algorithms
  - Integer vs. fractional pel accuracy search

Exhaustive Block Matching Algorithm (EBMA)



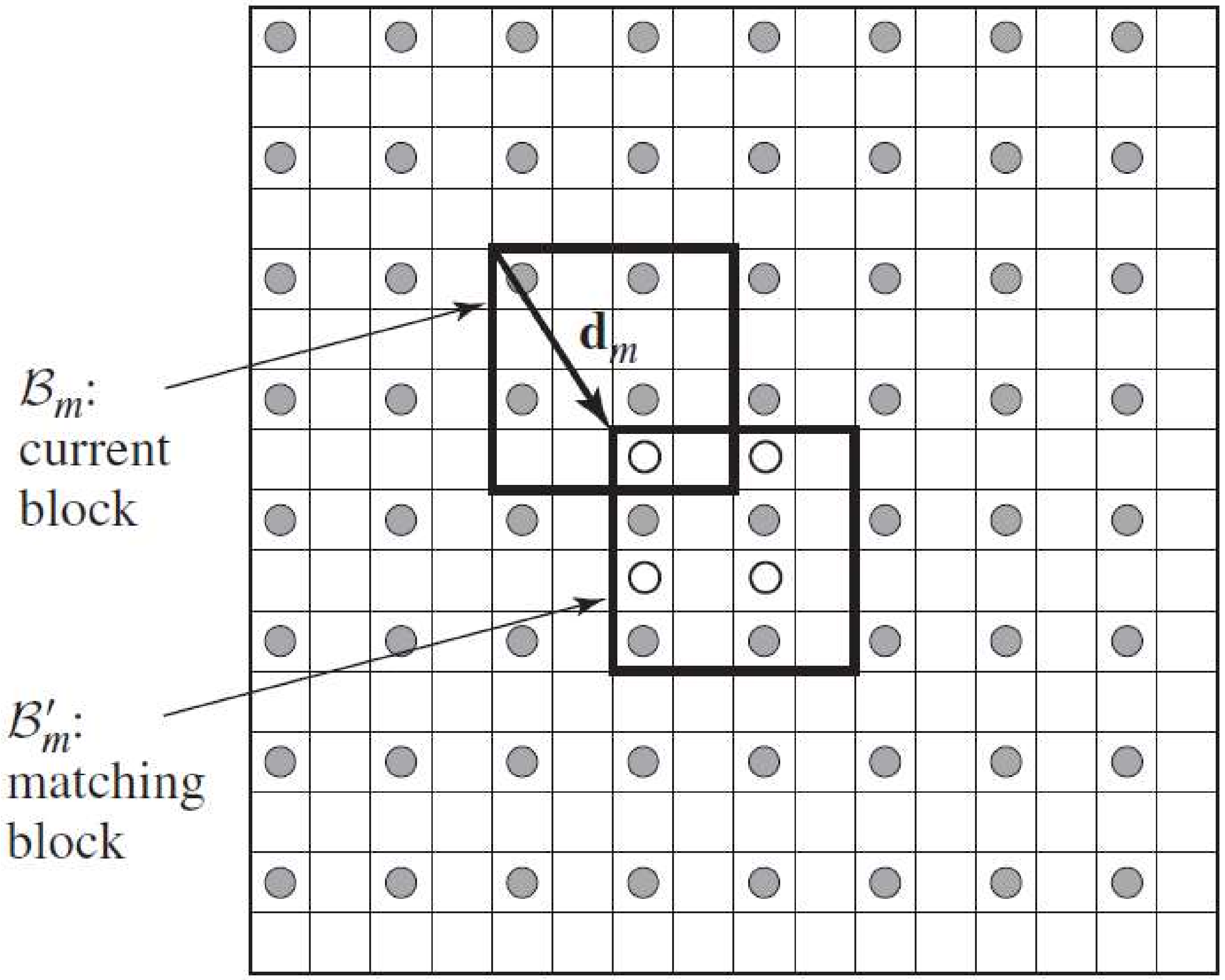
Complexity of Integer-Pel EBMA

- Assumption
  - Image size: MxM
  - Block size: NxN
  - Search range: (-R,R) in each dimension
  - Search stepsize: 1 pixel (assuming integer MV)
- Operation counts (1 operation=1 "-", 1 "abs", 1 "+"):
  - Each candidate position: N^2
  - Each block going through all candidates: $(2R+1)^2 N^2$
  - Entire frame: $(M/N)^2(2R+1)^2 N^2 = M^2(2R+1)^2$
    - ✓ Independent of block size!
- Example: M=512, N=16, R=16, 30 fps
  - Total operation count = 2.85x10^8/frame*30 frame/s =8.55x10^9/s
- Regular structure suitable for VLSI implementation
- Challenging for software-only implementation

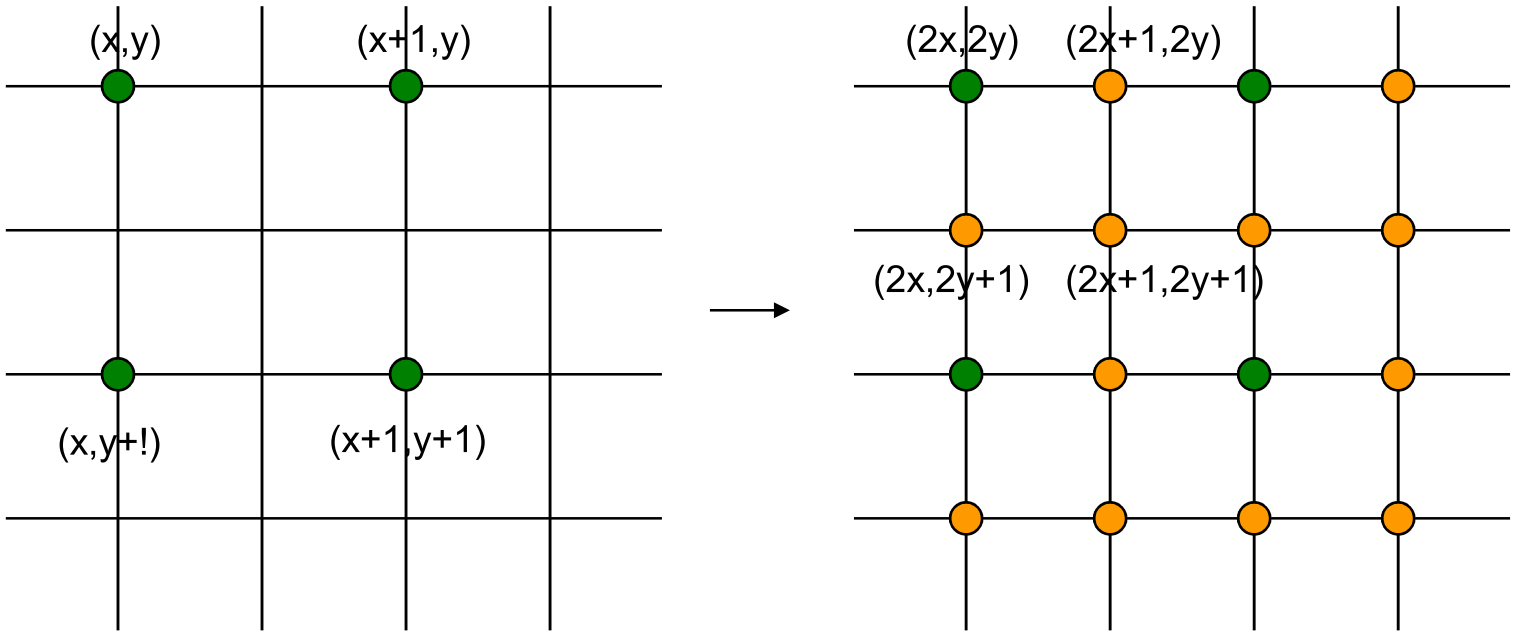
Fractional-Pixel Accuracy EBMA

- Real MV may not always be multiples of pixels. To allow sub-pixel MV, the search stepsize must be less than 1 pixel
- Half-pel EBMA: step-size=1/2 pixel in both dimension
- Difficulty:
  - Target frame only have integer pels
- Solution:
  - Interpolate the target frame by factor of two before searching
  - Bilinear interpolation is typically used
- Complexity:
  - 4 times of integer-pel, plus additional operations for interpolation.
- Fast algorithms:
  - Search in integer precisions first, then refine in a small search region in half-pel accuracy.

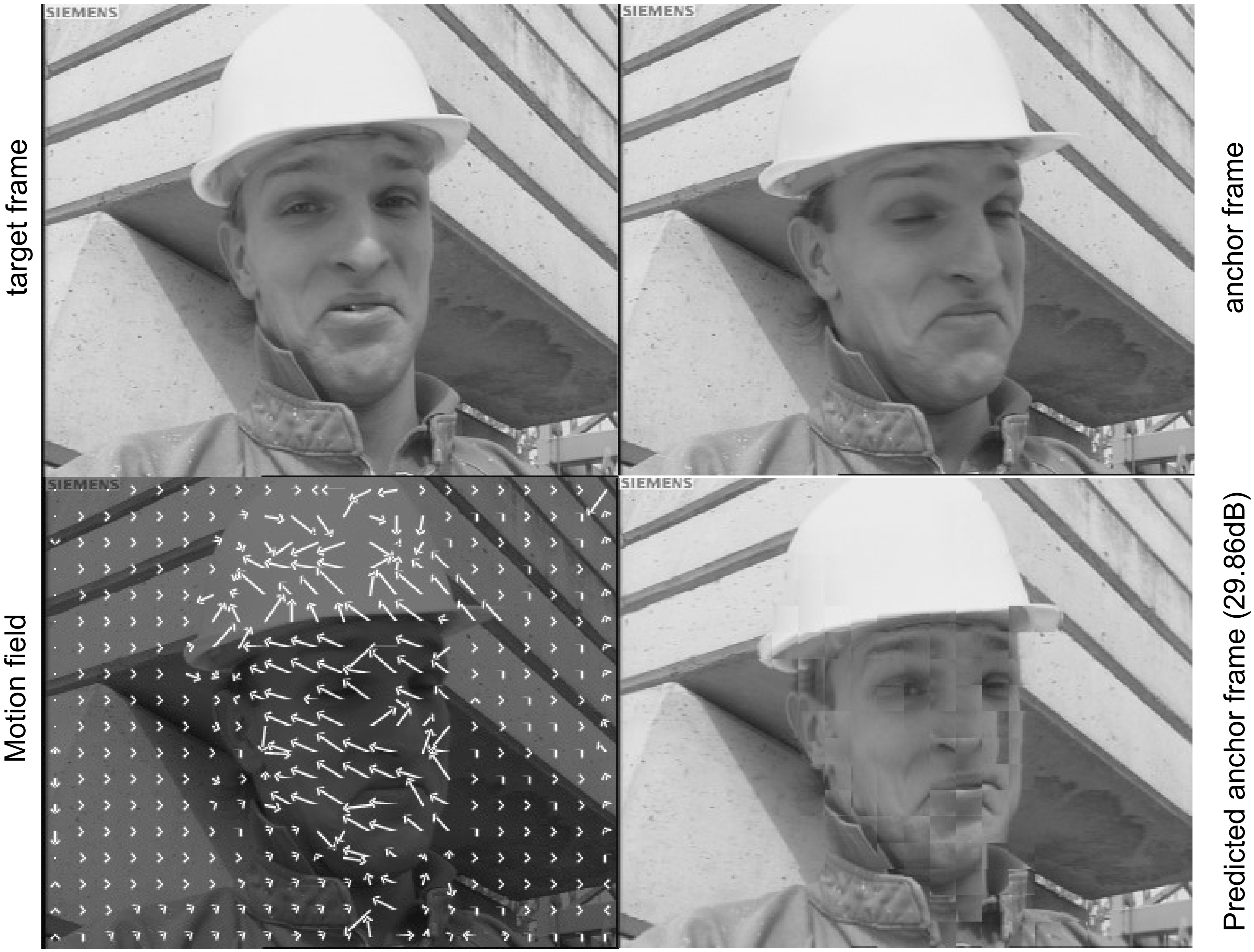
Half-Pel Accuracy EBMA



Bilinear Interpolation



O[2x,2y]=I[x,y]
O[2x+1,2y]=(I[x,y]+I[x+1,y])/2
O[2x,2y+1]=(I[x,y]+I[x+1,y])/2
O[2x+1,2y+1]=(I[x,y]+I[x+1,y]+I[x,y+1]+I[x+1,y+1])/4

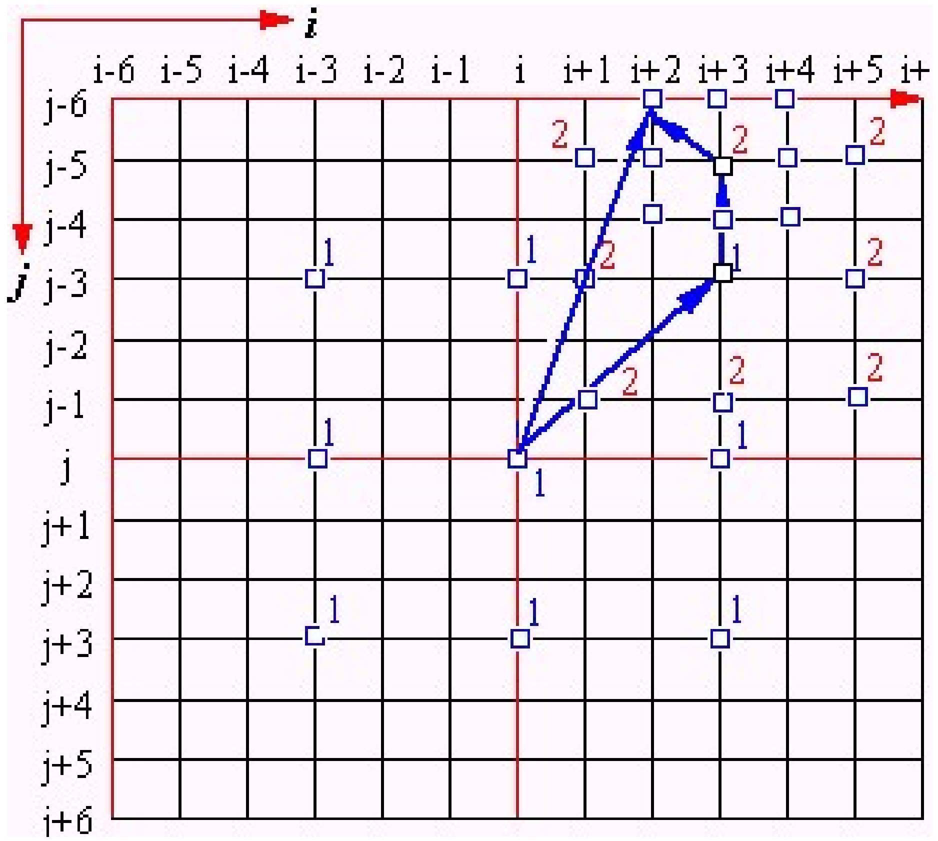


Example: Half-pel EBMA

Fast Algorithms for BMA

- Key idea to reduce the computation in EBMA:
  - Reduce # of search candidates:
    - ✓ Only search for those that are likely to produce small errors.
    - ✓ Predict possible remaining candidates, based on previous search result
  - Simplify the error measure (DFD) to reduce the computation involved for each candidate
- Classical fast algorithms
  - Three-step
  - 2D-log
- Many new fast algorithms have been developed since then
  - Some suitable for software implementation, others for VLSI implementation (memory access, etc)

Three-step Search Method



R_0: initial search step

Search step L

$$L = \lfloor \log_2 R_0 + 1 \rfloor$$

Total number:8L+1

For example

R=32

EBMA:4225 = $(2R+1)^2$

3Step:41 = 8*5+1

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"

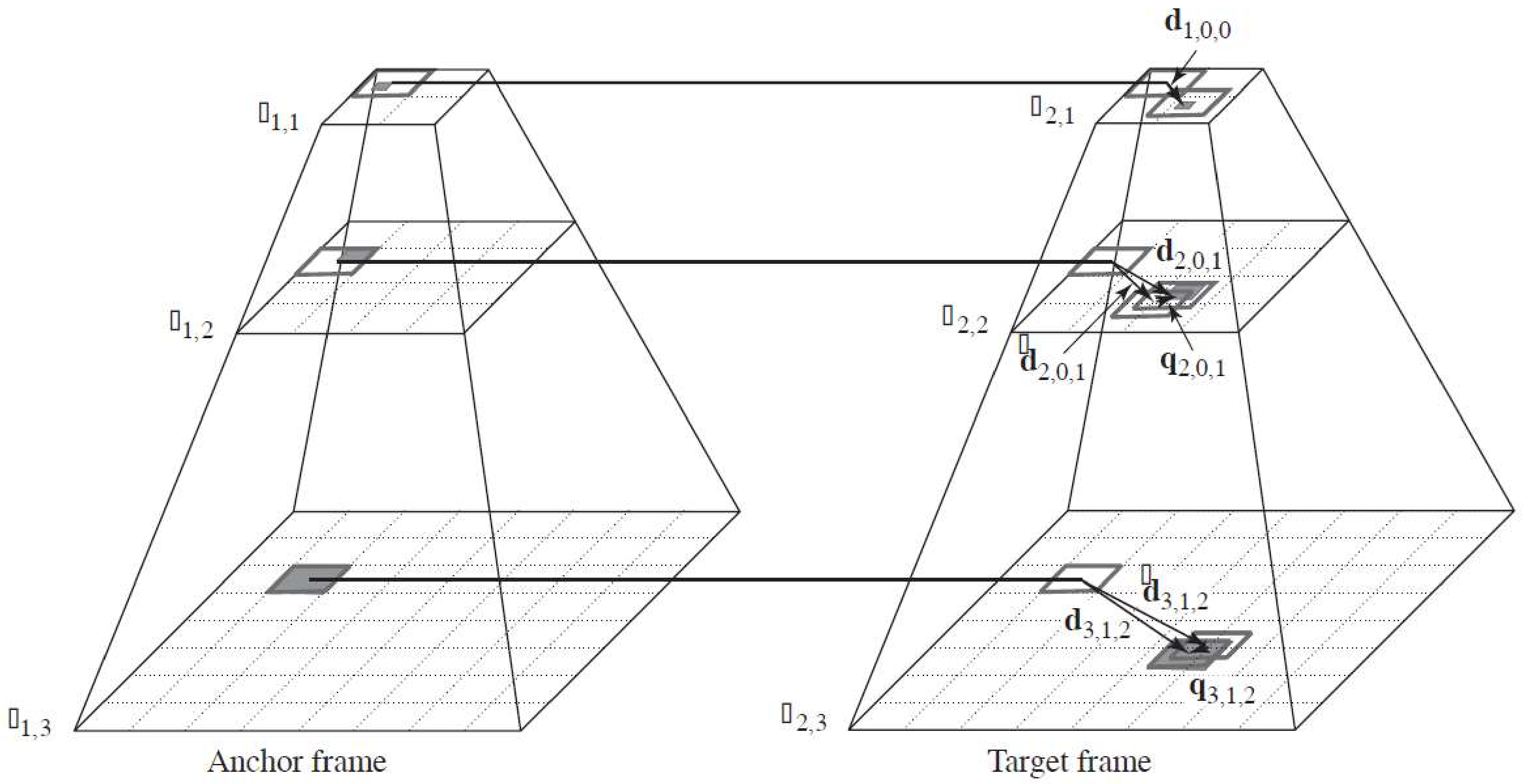
Problems with EBMA (Cntd)

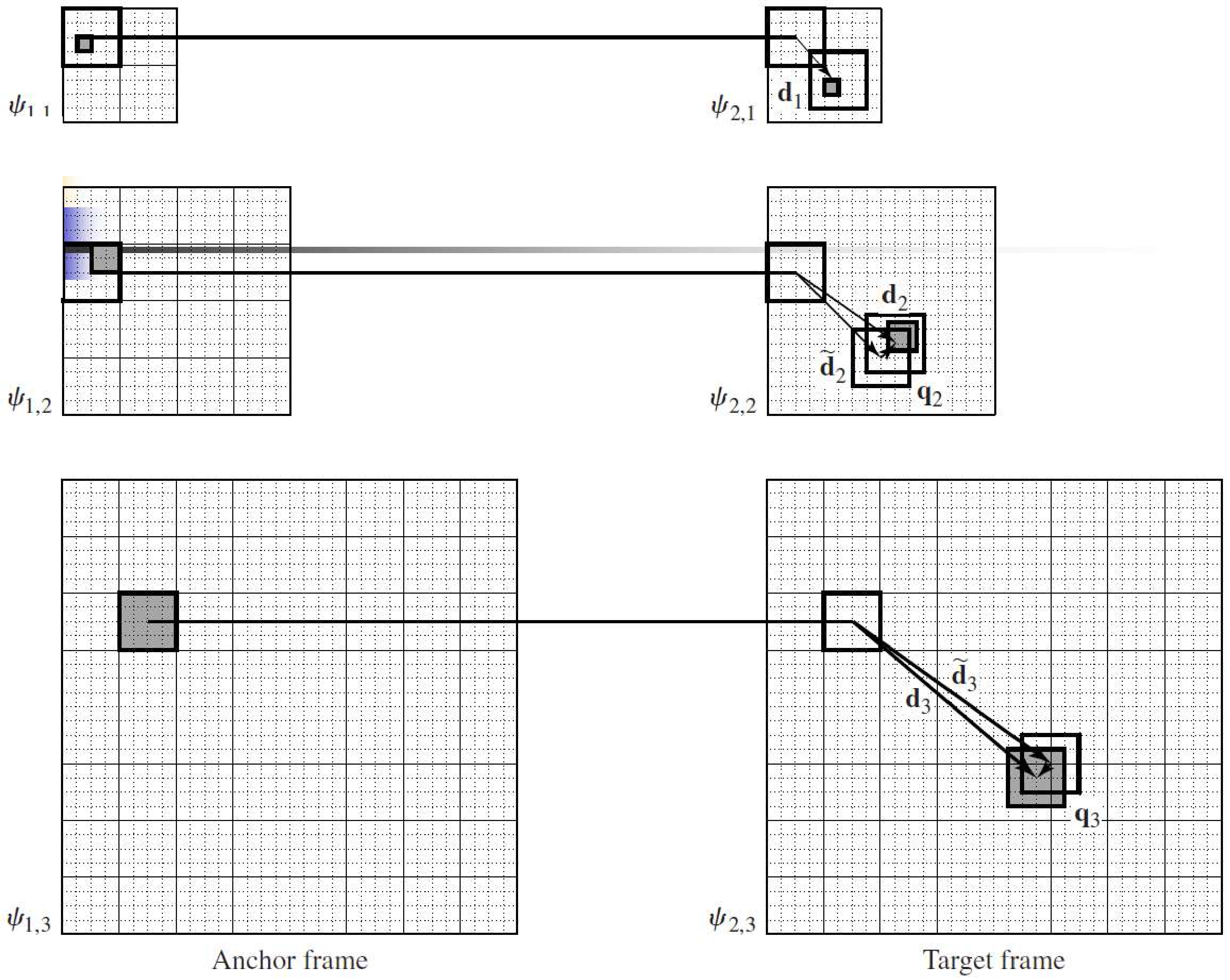
- Motion field somewhat chaotic
  - Because MVs are estimated independently from block to block
  - Fix:
    - ✓ Imposing smoothness constraint explicitly
    - ✓ Multi-resolution approach
    - ✓ Mesh-based motion estimation
- Wrong MV in the flat region
  - Because motion is indeterminate when spatial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

13.7 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

Hierarchical Block Matching Algorithm (HBMA)

Number of levels: L

lth level image: $\Psi_{t,l}(X), X \in \Lambda_l, t = 1,2$

Interpolation operator: $\tilde{d}_l(X) = \mathcal{U}(d_{l-1}(X))$

Error function: $\sum_{X \in \Lambda_l} | \Psi_{2,l}(X + \tilde{d}_l(X) + q_l(X)) - \Psi_{1,l}(X) |^p$

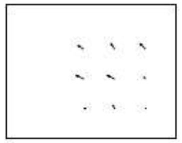
Update motion vector: $d_l(X) = \tilde{d}_l(X) + q_l(X)$

MV at lth level prediction:

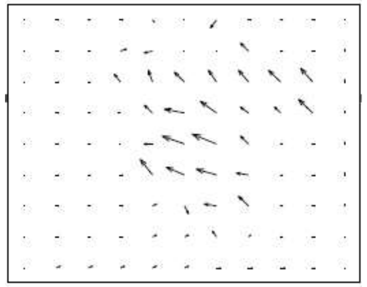
$$\tilde{d}_{l,m,n}(X) = \mathcal{U}(d_{l-1,\lfloor m/2 \rfloor,\lfloor n/2 \rfloor}(X)) = 2d_{l-1,\lfloor m/2 \rfloor,\lfloor n/2 \rfloor}(X)$$

Total motion:

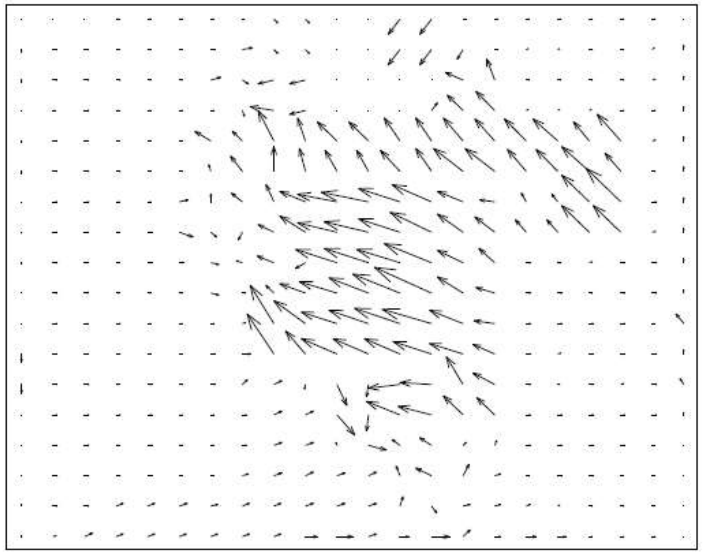
$$d_l(X) = q_L(X) + \mathcal{U}(q_{L-1}(X) + \mathcal{U}(q_{L-2}(X) \cdots + \mathcal{U}(q_1(X) + d_0(X)) \cdots))$$

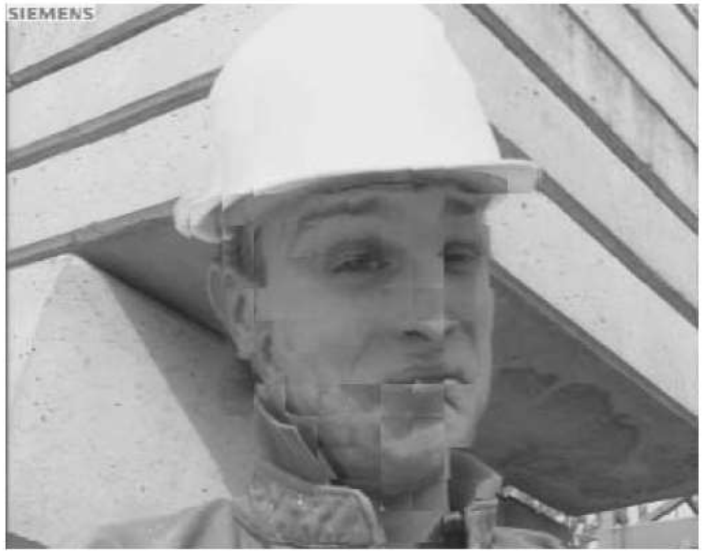

(a)

(b)


(c)

(d)

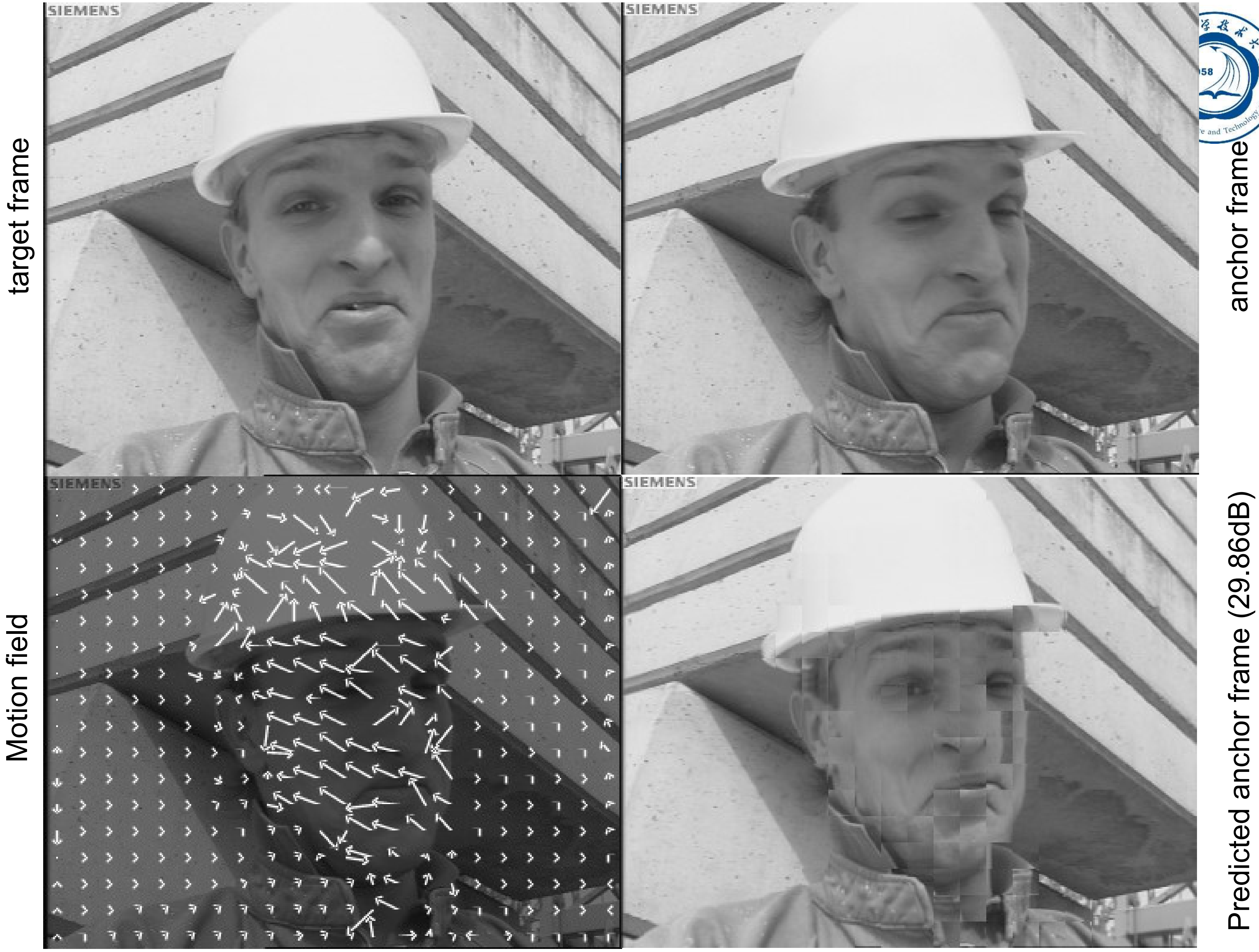

(e)



(f)

Predicted anchor frame (29.32dB)

Example: Three-level HBMA



Example: Half-pel EBMA

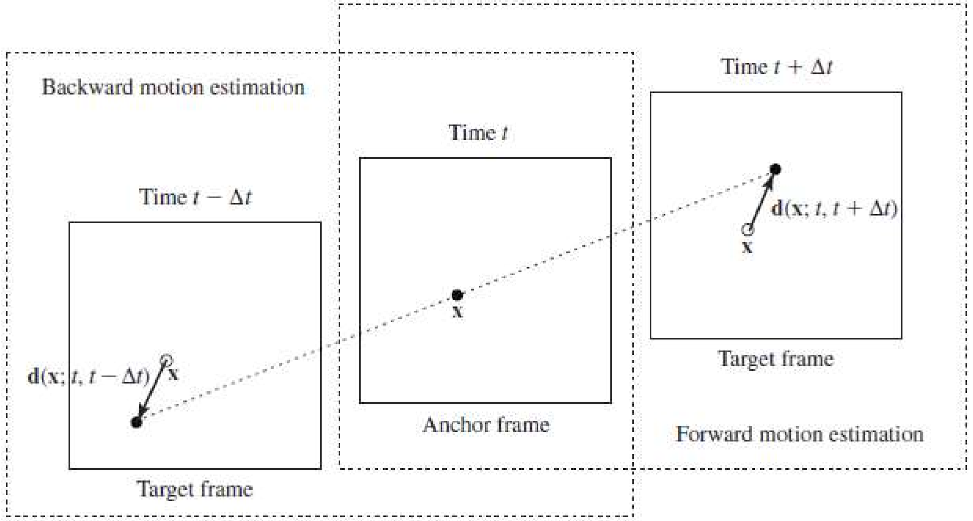
Computation Requirement of HBMA

- 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: $R/2^{(L-l)}$ (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: $3\cdot 4^{(L-2)}=3(L=2);12(L=3)$

13.8 Phase Correlation Method



Backward motion estimation
Time $t - \Delta t$
Time t
Time $t + \Delta t$
$\mathbf{d}(\mathbf{x}; t, t - \Delta t)$
$\mathbf{x}$
$\mathbf{d}(\mathbf{x}; t, t + \Delta t)$
Target frame
Anchor frame
Target frame
Forward motion estimation

To identify peaks in the phase correlation function(PCF)

$$\psi_1(X) = \psi_2(X + d)$$

$$\overline{\Psi}_1(f) = \overline{\Psi}_2(f) \bullet e^{j2\pi d^T f}$$

$$\tilde{\psi}(f) = \frac{\overline{\Psi}_1(f) \bullet \overline{\Psi}_2^*(f)}{|\overline{\Psi}_2(f) \bullet \overline{\Psi}_2^*(f)|} = e^{j2\pi d^T f}$$

$$PCF(X) = F^{-1}\{\tilde{\psi}(f)\} = \delta(X + d)$$

Note:

▪To reduce the effect of boundary: space domain weighing window function

▪To be extensively used in image registration

▪Advantage: To be insensible to illumination changes

13.9 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

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

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 $\mathbf{x}_n$.

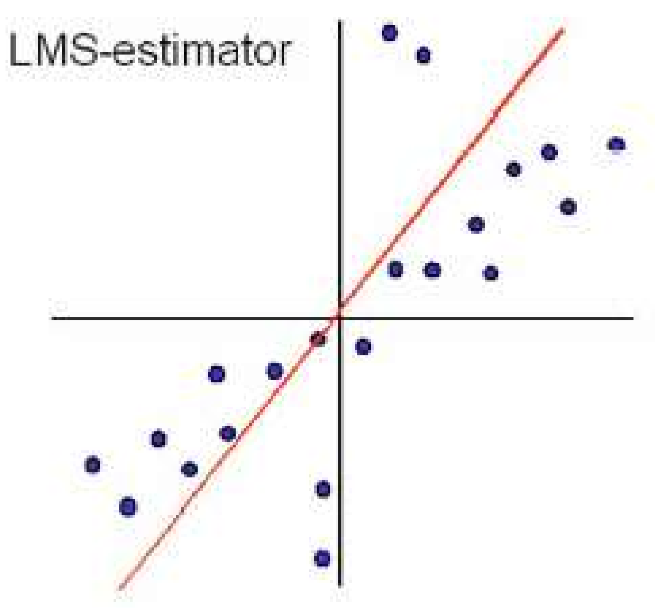
Robust Estimator

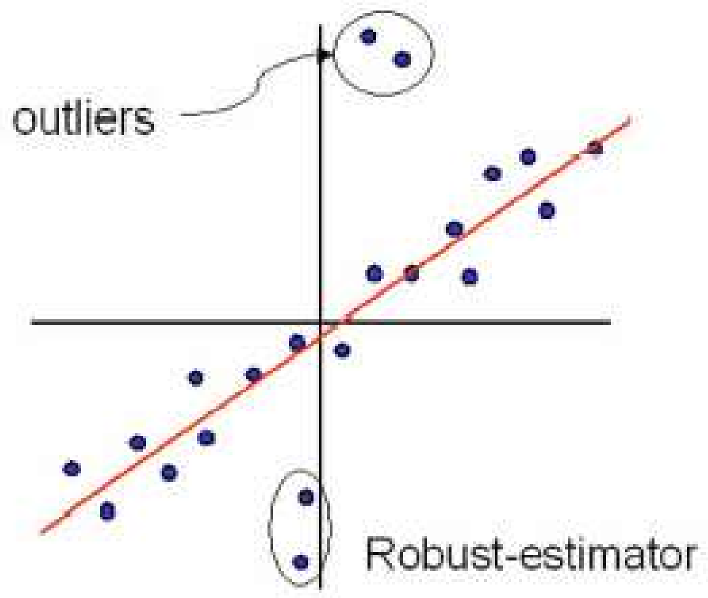
Essence: iteratively removing "outlier" pixels.

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

Illustration of Robust Estimator





Fitting a line to the data points by using LMS (least mean square) and robust estimators

13.10 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, or 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

13.11 Motion Segmentation

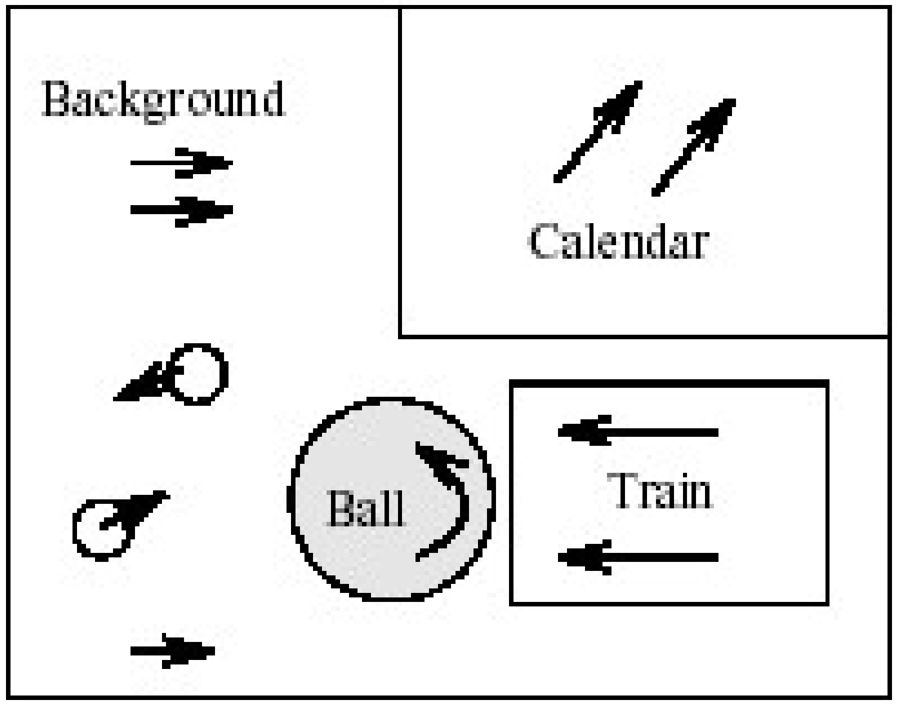
- WHY OBJECT MOTION SEGMENTATION
  - Help improve optical flow estimation with multiple motion
  - Help improve 3D motion and structure estimation
  - Object recognition
  - Object tracking
  - Object based video coding
  - Object based editing (synthetic transfiguration)

Image vs Motion Segmentation

- Segmentation is based on a feature (vector).
  - e.g. image segmentation usually based upon the grayscale, color or texture
- Application of standard image segmentation methods directly to motion segmentation (i.e. using the velocity vector as feature) may not be useful, since 3D motion usually generates spatially varying optical flow fields.
  - e.g. within a purely rotating object there is no flow at the center of rotation and the magnitude of the flow vectors increase as the distance of the points from the center of rotation increase.
- Motion segmentation needs to be based on some parametric description of the motion field.

2D Optical Flow Estimation and Segmentation

- A realistic scene generally contains multiple motion
- Smoothness constraints cannot be imposed across motion boundaries



Motion Segmentation Methods

- Direction method
  - Thresholding for change detection
- Segmentation based on motion model
  - Modified Hough Transform
  - Parameter clustering approach
  - Bayesian segmentation
  - Dominant motion estimation approach
  - Simultaneous estimation and segmentation

Summary

- Fundamentals:
  - 2-D Motion Corresponding to Camera Motion
    - ✓ Projective mapping, Affine mapping
  - 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

- 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
  - Phase Correlation Method
  - Global motion estimation
    - ✓ Good for estimating camera motion
  - Region-based motion estimation
    - ✓ More physically correct: allow different motion in each sub-object region
  - Motion Segmentation

Reference

- A. M. Tekalp. Digital Video Processing. 清华大学出版社，1998.
- Yao Wang etc. Video Processing and Communications. 清华大学出版社，2002.