Lecture 2
Time varying image analysis

- Video Source Model
- 3-D motion
- 2-D motion

Problems

- Visual surveillance
  - stationary camera watches a workspace -find moving objects and alert an operator
  - moving camera navigates a workspace - find moving objects and alert an operator
- Video coding
  - use image motion to perform more efficient coding of image sequences
- Navigation
  - camera moves through the world
    - estimate its trajectory use this to remove unwanted jitter from image sequence: Image stabilization and mosaicking
    - use this to control the movement of a robot through the world
- Structure estimation

Methods

- Motion detection
- Motion modeling
- Motion estimation
- Motion tracking
- Structure estimation
- Segmentation

Source modeling of a video shot

- Frame-to-frame variation in the intensity of images is due to
  - 3-D *camera* motion; e.g., zoom, tilt, and pan, etc.
  - *object* motion
    - rigid motion; e.g., local translation and rotation
    - deformable motion
  - photometric effects of 3-D motion and change in scene illumination

Motion detection (1/2)

- Frame differencing
  - subtract, on a pixel by pixel basis, consecutive frames in a motion sequence. High differences indicate change between the frames due to either motion or changes in illumination
- Problems
  - noise in images can give high differences where there is no motion
    - compare neighborhoods rather than points
  - as objects move, their homogeneous interiors don't result in changing image intensities over short time periods
    - motion detected only at boundaries
    - requires subsequent grouping of moving pixels into objects

Motion detection (2/2)

- Background subtraction
  - create an image of the stationary background by averaging a long sequence
    - for any pixel, most measurements will be from the background
    - computing the median measurements, for example, at each pixel, will with high probability assign that pixel the true background intensity – fixed threshold on differencing used to find "foreground" pixels
    - can also compute a distribution of background pixels by fitting a mixture of Gaussians to set of intensities and assuming large population is the background - adaptive thresholding to find foreground pixels
  - difference a frame from the known background frame
    - even for interior points of homogeneous objects, likely to detect a difference
    - this will also detect objects that are stationary but different from the background
    - typical algorithm used in surveillance systems
- Algorithms such as these only work if the camera is stationary and objects are moving against a fixed bg

3-D and 2-D Motion

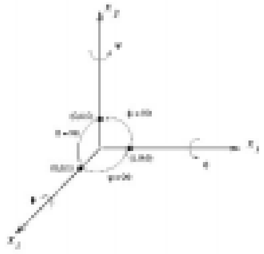
- 3-D Motion Modeling
  - Cartesian vs. Homogeneous Coordinates
  - Modeling Rigid Motion and Rotation Matrix
  - Modeling Deformable Motion
- Camera Modeling and Image Formation
  - Projective Camera (Perspective Projection)
  - Affine Camera (Weak-Perspective and Orthographic)
- 2-D motion estimation
- 3-D motion estimation
- Structure from motion
- Tracking

Rigid Motion - Cartesian coordinates

$$\begin{bmatrix} X1' \\ X2' \\ X3' \end{bmatrix} = \begin{bmatrix} r_{11} r_{21} r_{13} \\ r_{21} r_{22} r_{23} \\ r_{31} r_{32} r_{33} \end{bmatrix} \begin{bmatrix} X1 \\ X2 \\ X3 \end{bmatrix} + \begin{bmatrix} T1 \\ T2 \\ T3 \end{bmatrix}$$

- Rotation by Euler angles about coordinate axes
- Rotation by a solid angle about an arbitrary axis

Rigid Motion - Rotation by Euler angles



The matrices for clockwise rotation

$$\mathbf{R}_\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$\mathbf{R}_\psi = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix}$$

$$\mathbf{R}_\varphi = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Eulerian angles of rotation, θ, φ, ϕ.

Rigid Motion - Small angle approximation

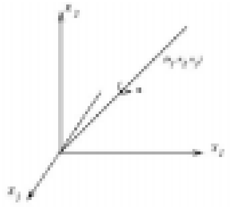
- For small rotations, $\cos \Delta\Psi \approx 1$ and $\sin \Delta\Psi \approx \Delta\Psi$, etc.,

$$\mathbf{R}_\psi = \begin{bmatrix} 1 & 0 & \Delta\psi \\ 0 & 1 & 0 \\ -\Delta\psi & 0 & 1 \end{bmatrix}$$

- Then, the composite rotation matrix $\mathbf{R}$ is given by:

$$\mathbf{R} = \mathbf{R}_\varphi \mathbf{R}_\theta \mathbf{R}_\psi = \begin{bmatrix} 1 & -\Delta\varphi & \Delta\psi \\ \Delta\varphi & 1 & -\Delta\theta \\ -\Delta\psi & \Delta\theta & 1 \end{bmatrix}$$

Rigid Motion - Rotation by a solid angle about an arbitrary axis



- Rotation by the angle α about an arbitrary axis (n_1, n_2, n_3)

Δα

$$\mathbf{R} = \begin{bmatrix} 1 & -n_3 \sin\alpha & +n_2 \sin\alpha \\ +n_3 \sin\alpha & 1 & -n_1 \sin\alpha \\ -n_2 \sin\alpha & +n_1 \sin\alpha & 1 \end{bmatrix}$$

Rigid Motion - Homogeneous Coordinates

- The affine transformation $\mathbf{X}' = \mathbf{R}\,\mathbf{X} + \mathbf{T}$ in the Cartesian coordinates can be expressed as a linear transformation in the homogeneous coordinates
- $\mathbf{X_h}' = \mathcal{R}\,\mathbf{X_h}$

Deformable Motion

- **X' = (R+D) X + T**

3-D Velocity model

$$\begin{bmatrix} X_1' \\ X_2' \\ X_3' \end{bmatrix} = \begin{bmatrix} 1 & -\Delta\phi & \Delta\psi \\ \Delta\phi & 1 & -\Delta\theta \\ -\Delta\psi & \Delta\theta & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$

$$\begin{bmatrix} X_1'-X_1 \\ X_2'-X_2 \\ X_3'-X_3 \end{bmatrix} = \begin{bmatrix} 0 & -\Delta\phi & \Delta\psi \\ \Delta\phi & 0 & -\Delta\theta \\ -\Delta\psi & \Delta\theta & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$

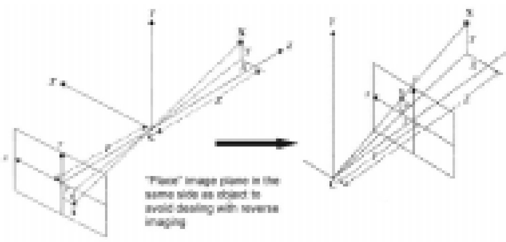
$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \\ \dot{X}_3 \end{bmatrix} = \begin{bmatrix} 0 & \Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} + \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix}$$

Geometric image formation(1/3)

- Imaging systems capture 2-D projections of a time-varying 3-D scene at certain intervals of time.

- ***P:*** $R^4 \rightarrow R^2 \times Z$
- $(X_1,X_2,X_3,t) \rightarrow (x_1,x_2,k)$

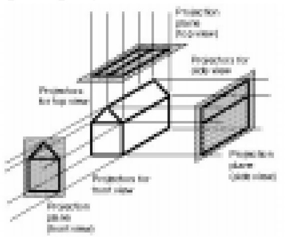
Geometric image formation(2/3)



$x = F\frac{X}{\ }$ $\qquad$ $y = F\frac{Y}{\ }$

Geometric image formation(3/3)

- Orthographic projection



$x = X$

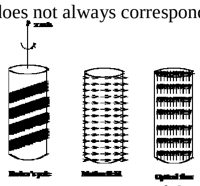
$y = Y$

Optical flow/correspondence

- Observable variations in the 2-D image intensity pattern (also called the apparent 2-D motion).
- *Optical flow* refers to a dense vector field indicating rate of change of intensity variations in x_1 and x_2 directions.
- *Dense correspondence field* refers to a set of frame-to-frame motion vectors indicating matching pixel pairs.

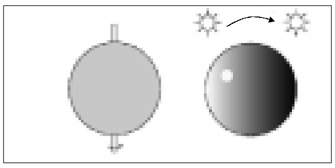
Optical Flow Vs. Motion Field (1/2)

- Optical flow does not always correspond to motion field



- Optical flow is an approximation of the motion field. The error is small at points with high spatial gradient under some simplifying assumptions

Optical Flow Vs. Motion Field (2/2)



Optical flow (observed/apparent 2-D motion) may not be the same as the actual projected 2-D motion.

2-D motion field

- Magnitude and direction representation

Optical flow equaltion

$$f(x+d_x, y+d_y, t+d_t) = f(x,y,t)$$ Const luminance assumpt.

$$LHS \approx f(x,y,t) + \frac{\partial f}{\partial x} d_x + \frac{\partial f}{\partial y} d_y + \frac{\partial f}{\partial t} d_t$$ Taylor's expansion

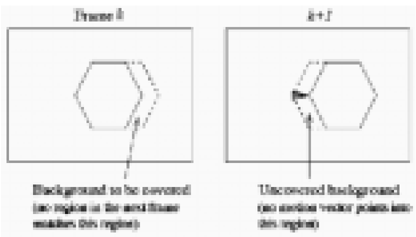
$$\Rightarrow \frac{\partial f}{\partial x} d_x + \frac{\partial f}{\partial y} d_y + \frac{\partial f}{\partial t} d_t = 0$$

$$\Rightarrow \boxed{\frac{\partial f}{\partial x} v_x + \frac{\partial f}{\partial y} v_y + \frac{\partial f}{\partial t} = 0, \text{ or } (\nabla f)^T \mathbf{v} + \frac{\partial f}{\partial t} = 0}$$

with $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$ being spatial gradient vector of f(x, y, t)

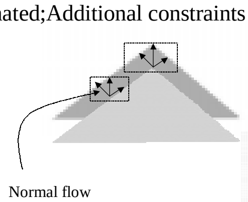
Ambiguity in motion estimation (1/2)

- *Existence of a solution:* Uniform, covered/uncovered background



Ambiguity in motion estimation (2/2)

Uniqueness of a solution: One equation for two unknowns; Aperture problem; Normal flow can be estimated;Additional constraints



$$v_\perp(\mathbf{x},t) = \frac{-\frac{\partial s(\mathbf{x},t)}{\partial t}}{\|\nabla s(\mathbf{x},t)\|}$$

Additional constraints

- OFE may be imposed on each color channel separately.
  - The displacement vector can then possibly be constrained in three different directions, if the direction of the spatial gradient at each band is different.
- Smoothness constraint

Displaced Frame Difference

- The DFD between two frames at t and t+Δt

$$dfd(\mathbf{x},\hat{\mathbf{d}}) = I(\mathbf{x}+\mathbf{d}(\mathbf{x}), t+\Delta t) - I(\mathbf{x},\mathbf{t})$$

Intensity displacement

Relation between DFE and OFE

- Use Taylor series expansion (small **d** and Δt)

$$I(\mathbf{x}+\mathbf{d}(\mathbf{x}), t+\Delta t) = I(x,t) + \frac{\delta I(\mathbf{x},t)}{\delta x_1} d_1(\mathbf{x}) + \frac{\delta I(\mathbf{x},t)}{\delta x_2} d_2(\mathbf{x}) + \frac{\delta I(\mathbf{x},t)}{\delta t}\Delta t$$

OFE and DFD constraints

- In practice, neither the DFD nor the error in the OFE is exactly zero because
  - there is observation noise
  - scene illumination may vary
  - there are occlusion regions
  - there are interpolation errors
- Therefore one minimizes absolute or the square value of DFD or LHS of OFE

General methods in motion estimation

- Feature based
  - Establish correspondence
  - Estimate parameters
- Intensity based
  - Apply optical flow equation or its variants
  - Good for multiple objects
  - Used in video coding

Key issues in motion estimation

- How to parameterize underlying motion field?
- What estimation criteria?
- How to search for optimal parameters?

Motion models

- Parametric models
  - – Translational motion
  - – Affine motion
  - – Perspective motion
  - – Bilinear/quadratic motion
- Quasi-parametric models
- Non-parametric models

Parametric motion models

- Affine motion model (6-parameters)

$$x_1' = a_1 x_1 + a_2 x_2 + a_3$$
$$x_2' = a_4 x_1 + a_5 x_2 + a_6$$

- Perspective motion model (8-parameters)

$$x_1' = \frac{a_1 x_1 + a_2 x_2 + a_3}{a_7 x_1 + a_8 x_2 + 1} \qquad x_2' = \frac{a_4 x_1 + a_5 x_2 + a_6}{a_7 x_1 + a_8 x_2 + 1}$$

Bilinear motion model (8-parameters)

$$x_1' = a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 + a_4$$
$$x_2' = a_5 x_1 + a_6 x_2 + a_7 x_1 x_2 + a_8$$

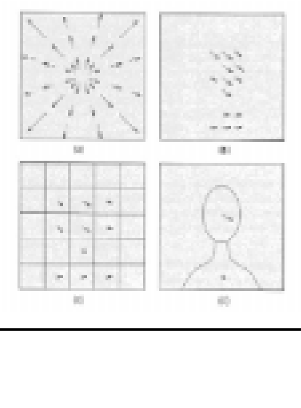
Motion representations (1/4)

- Pixel based representation
  - Specify MV for each pixel
  - Computationally expensive
  - Need additional constraints
- Global motionrepresentation
  - Good if major motion is camera motion
- Region based representation
  - One set of motion param for each region
  - Need iterative segmentation and estimation

Motion representations (2/4)

- Block based representation
  - Fixed partitioning into blocks and characterize each with simple model
  - Pro: Good comprpmize between accuracy and complexity; Con: False discontinuties btw blocks
- Mesh based representation
  - Partition image into polygons
  - Pro: Provide continuos motion, Con Problem with object boundaries

Motion representations (3/4)



Motion representations (4/4)

Nonparametric 2-D motion estimation

- Methods based on OFE
- Phase correleation method
- Block matching method
- Pel-recursive methods
- Bayesian methods

Methods using OFE – Horn-Schunck Method

- Two criteria:
  - Optical flow is smooth, $\mathbf{E}_s(u,v)$
  - Small error in optical flow constraint equation, $\mathbf{E}_{of}(v_x,v_y)$
- Minimize a combined error functional

$$\mathbf{E}_c(u,v) = \mathbf{E}^2_{of}(u,v) + \lambda\, \mathbf{E}^2_s(u,v)$$

  λ is a weighting parameter
- Solved iteratively

Error functional

- OF cost

$$\mathrm{E}of = \frac{\delta\ I}{\delta\ x}u + \frac{\delta\ I}{\delta\ y}v + \frac{\delta\ I}{\delta\ t}$$

- Smoothness cost

$$\mathrm{E}^2{}_s = (\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2$$

$$\mathrm{E}^2{}_s = \|\nabla u\|^2 + \|\nabla v\|^2$$

Horn-Schunck Algorithm : Discrete Case

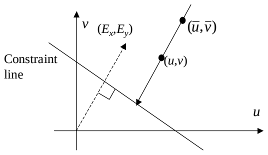
- Derivatives (and error functionals) are approximated by difference operators
- Leads to an iterative solution:

$$u_{ij}^{n+1} = \bar{u}_{ij}^{n} - \alpha E_x \qquad \alpha = \frac{E_x \bar{u}_{ij}^{n} + E_y \bar{v}_{ij}^{n} + E_t}{1 + \lambda (E_x^2 + E_y^2)}$$

$$v_{ij}^{n+1} = \bar{v}_{ij}^{n} - \alpha E_y$$

$\bar{u}, \bar{v}$ is the average of values of neighbors

Intuition of the Iterative Scheme



The new value of (u,v) at a point is equal to the average of surrounding values minus an adjustment in the direction of the brightness gradient

Horn - Schunck Algorithm (1/2)

begin
for $j := 1$ to N do for $i := 1$ to M do begin
calculate the values $E_x(i,j,t)$, $E_y(i,j,t)$, and $E_t(i,j,t)$ using a selected approximation formula;
{ special cases for image points at the image border have to be taken into account }
initialize the values $u(i,j)$ and $v(i,j)$ with zero
end {for};
choose a suitable weighting value λ; { e.g $\lambda = 10$ }
choose a suitable number $n_0 \geq 1$ of iterations; { $n_0 = 8$ }
$n := 1$; { iteration counter }
while $n \leq n_0$ **do begin**
for $j := 1$ to N **do** for $i := 1$ to M do begin
$\bar{u} := \frac{1}{4}\left(u(i-1,j)+u(i+1,j)+u(i,j-1)+u(i,j+1)\right)$;
$\bar{v} := \frac{1}{4}\left(v(i-1,j)+v(i+1,j)+v(i,j-1)+v(i,j+1)\right)$;
{ treat image points at the image border separately }
$$\alpha := \frac{E_x(i,j,t)\bar{u}+E_y(i,j,t)\bar{v}+E_t(i,j,t)}{1+\lambda\left(E_x^2(i,j,t)+E_y^2(i,j,t)\right)}\cdot\lambda\ ;$$
$u(i,j) := \bar{u} - \alpha \cdot E_x(i,j,t)$; $v(i,j) := \bar{v} - \alpha \cdot E_y(i,j,t)$
end {for};
$n := n+1$
end {while}
end;

Horn-Schunck Algorithm(2/2)

begin
for *j* := 1 to *N* **do** **for** *I*:= 1 to *M* **do** **begin**
calculate the values $E_x(i,j,t)$, $E_y(i,j,t)$ *and* $E_t(i,j,t)$ using a selected approx formula
initialize the values *u(I,j)* and *v(i,j)* to zero
end {for}
choose a suitable weighting value
choose a suitable number n_0 1 of iterations
n := 1
while *n* n_0 **do begin**
for *j* := 1 to *N* **do** **for** *i* := 1 to *M* **do** **begin**
compute *u*, *v*,
update *u(i,j)*, *v(i,j)*
end {for}
n := *n* + 1
end {while}
end

Finite difference methods

- Forward difference
- Backward difference
- Average difference
- Local average of the average differences
- Horn and Schunck proposed averaging four finite differences

$$\frac{\partial I}{\partial x} = \frac{1}{4}\left\{ I(x+1,y,t) - I(x,y,t) + I(x+1,y+1,t) - I(x,y+1,t) + \right.$$

Local polynomial fitting method

- Approximate I(x,y,t) locally by a linear combination of some low order polynomials

$$\hat{I}\ (x,y,t) = \sum_{i=0}^{N-1} a_i \phi_i (x,y,t)$$

- For N = 9 → Basis functions: 1,x,y,t,x^2,y^2,xy,xt,yt

Coefficient estimation

- Coefficients a_i are estimated using the least squares method

$$e^2 = \sum_{n1} \sum_{n2} \sum_{n3} (I\ (x,y,t) - \sum_{i=0}^{N-1} a_i \phi_i (x,y,t))^2 |_{x=n1, y=n2, t=n3}$$

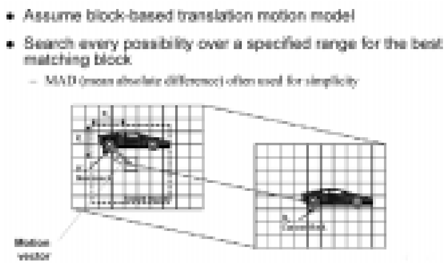
Lucas and Kanade

- Assumes that the motion vectors remain unchanged over a particular block of pixels
- Minimize

$$E = \sum (\frac{\delta I}{\delta x} u + \frac{\delta I}{\delta y} v + \frac{\delta I}{\delta t})^2$$

$$\begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \begin{bmatrix} \sum_{x \in B} \frac{\partial I}{\partial x}\frac{\partial I}{\partial x} & \sum_{x \in B} \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum_{x \in B} \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \sum_{x \in B} \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{x \in B} \frac{\partial I}{\partial x}\frac{\partial I}{\partial t} \\ -\sum_{x \in B} \frac{\partial I}{\partial y}\frac{\partial I}{\partial t} \end{bmatrix}$$

Block matching



Search procedures

- Usually the search area is limited to

$$-M_1 <= d_1 <= M_1$$
$$-M_2 <= d_2 <= M_2$$

- Methods
  - Full(exhaustive search)
  - Three-step search
  - Cross search

Exhaustive search: Pros and cons

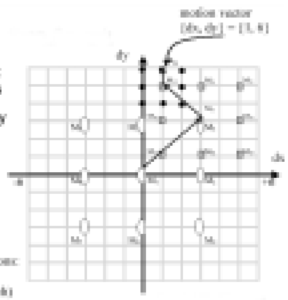
- Pros
  - Guaranteed optimal within search range
- Cons
  - Can only search among finitely many candidates
    - *What if the motion is "fractional"?*
  - High computation complexity
    - *On the order of [search-range-size * image-size] for 1-pixel step size*

➔ How to improve accuracy?
  - Include blocks at fractional translation as candidates – require interpolation

➔ How to improve speed?
  - Try to exclude unlikely candidates

3-step search

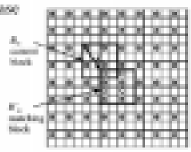
- Search candidates at 8 neighbour positions
- Step-size cut down by 2 after each iteration
  - Start with step size approx. half of max. search range



Total number of computations:
9 + 8×2 = 25 (3-step)
$(2R+1)^2$ = 169 (full search)

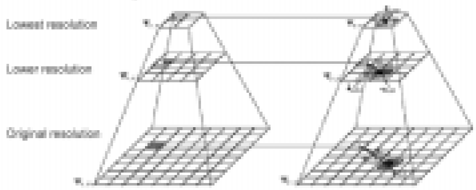
Fractional accuracy

- For motion accuracy of 1/K pixel
  - Upsample (interpolate) reference frame by a factor of K
  - Search for the best matching block in the upsampled reference frame
- Half-pel accuracy ~ K=2
  - Significant accuracy improvement over integer-pel (esp. for low-resolution)
  - Complexity increase

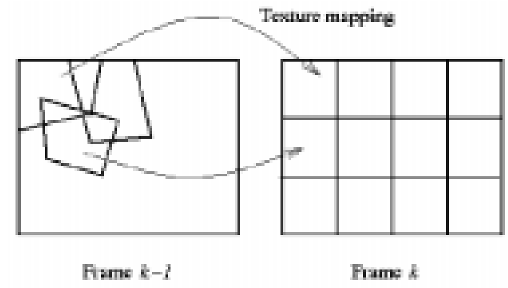


Hierarchical block matching

- Problem with fast search at full resolution
  - Small mis-alignment may give high displacement error (E_{DFD})
    - *esp. for texture and edge blocks*
- Hierarchical (multi-resolution) block matching
  - Match with coarse resolution to narrow down search range
  - Match with high resolution to refine motion estimation



Generalized block matching



Mesh modeling