A Tutorial on VIDEO COMPUTING

Mubarak Shah
School Of Computer Science
University of Central Florida
Orlando, FL 32816
Shah@cs.ucf.edu
Http://cs.ucf.edu/~vision/

Multimedia

- Text
- Graphics
- Audio
- Images
- Video

Imaging Configurations

- Stationary camera stationary objects
- Stationary camera moving objects
- Moving camera stationary objects
- Moving camera moving objects

Video

- sequence of images
- clip
- mosaic
- key frames

Steps in Video Computing

- Acquire (CCD arrays/synthesize (graphics))
- Process (image processing)
- Analyze (computer vision)
- Transmit (compression/networking)
- Store (compression/databases)
- Retrieve (computer vision/databases)
- Browse (computer vision/databases)
- Visualize (graphics)

Computer Vision

- Measurement of Motion
  - 2-D Motion
    - optical flow
    - point correspondences
  - 3-D Motion
    - structure from motion (sfm)
    - compute 3D translation, 3D rotation
    - shape from motion (depth)

Computer Vision (contd.)

- Scene Change Detection
  - consecutive frame differencing
  - background differencing
    - median filter
    - pfinder
    - W4
    - Mixture of Gaussians

Computer Vision (contd.)

- Tracking
  - people
  - vehicles
  - animals

Computer Vision (contd.)

- Video Recognition
  - activity recognition
  - gesture recognition
  - facial expression recognition
  - lipreading
- Video Segmentation
  - shots
  - scenes
  - stories
  - key frames

Image Processing

- Filtering
- Compression
  - MPEG-1
  - MPEG-2
  - MPEG-4
  - MPEG-7 (Multimedia Content Description Interface)

Databases

- Storage
- Retrieval
- Video on demand
- Browsing
  - skim
  - abstract
  - key frames
  - mosaics

Networking

- Transmission
- ATM

Computer Graphics

- Visualization
- Image-based Rendering and Modeling
- Augmented Reality

Video Computing

- Computer Vision
- Image Processing
- Computer Graphics
- Databases
- Networks

PART I

Measurement of Motion

Contents

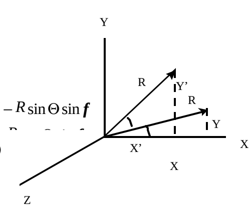
3-D Rigid Motion

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}}_{\text{Rotation matrix (9 unknowns)}}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \underbrace{\begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}}_{\text{Translation (3 unknowns)}}$$

Rotation

$$X = R\cos \boldsymbol{f}$$

$$Y = R\sin \boldsymbol{f}$$



$$X' = R\cos(\Theta + \boldsymbol{f}) = R\cos\Theta\cos\boldsymbol{f} - R\sin\Theta\sin\boldsymbol{f}$$

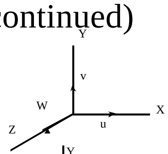
$$X' = X\cos\Theta - Y\sin\Theta$$

$$Y' = X\sin\Theta + Y\cos\Theta$$

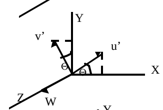
$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix}$$

Rotation (continued)

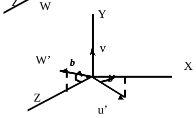
$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$$R = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$$R = \begin{bmatrix} \cos\boldsymbol{b} & 0 & -\sin\boldsymbol{b} \\ 0 & 1 & 0 \\ \sin\boldsymbol{b} & 0 & \cos\boldsymbol{b} \end{bmatrix}$$



Euler Angles

$$R = R_Z^{\boldsymbol{a}} R_Y^{\boldsymbol{b}} R_X^{\boldsymbol{g}} = \begin{bmatrix} \cos\boldsymbol{a}\cos\boldsymbol{b} & \cos\boldsymbol{a}\sin\boldsymbol{b}\sin\boldsymbol{g} - \sin\boldsymbol{a}\cos\boldsymbol{g} & \cos\boldsymbol{a}\sin\boldsymbol{b}\cos\boldsymbol{g} + \sin\boldsymbol{a}\sin\boldsymbol{g} \\ \sin\boldsymbol{a}\cos\boldsymbol{b} & \sin\boldsymbol{a}\sin\boldsymbol{b}\sin\boldsymbol{g} + \cos\boldsymbol{a}\cos\boldsymbol{g} & \sin\boldsymbol{a}\sin\boldsymbol{b}\cos\boldsymbol{g} - \cos\boldsymbol{a}\sin\boldsymbol{g} \\ -\sin\boldsymbol{b} & \cos\boldsymbol{b}\sin\boldsymbol{g} & \cos\boldsymbol{b}\cos\boldsymbol{g} \end{bmatrix}$$

if angles are small($\cos\Theta \approx 1$) $\sin\Theta \approx \Theta$

$$R = \begin{bmatrix} 1 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 1 & -\boldsymbol{g} \\ \boldsymbol{b} & \boldsymbol{g} & 1 \end{bmatrix}$$

Displacement Model

Orthographic Projection

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

(x,y)=image coordinates,
(X,Y,Z)=world coordinates

$$x' = r_{11}x + r_{12}y + (r_{13}Z + T_X)$$
$$y' = r_{21}x + r_{22}y + (r_{23}Z + T_Y)$$

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

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}$$ Affine Transformation

Orthographic Projection (contd.)

$$\begin{bmatrix} X' \\ Y \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} 1 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 1 & \boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$x' = x - \boldsymbol{a}y + \boldsymbol{b}Z + T_X$$
$$y' = \boldsymbol{a}x + y - \boldsymbol{g}Z + T_Y$$

Perspective Projection

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$x' = \frac{X'}{Z'} \qquad y' = \frac{Y'}{Z'}$$ focal length = -1

scale ambiguity

$$x' = \frac{r_{11}x + r_{12}y + r_{13} + \frac{T_X}{Z}}{r_{31}x + r_{32}y + r_{33} + \frac{T_Z}{Z}}$$

$$y' = \frac{r_{21}x + r_{22}y + r_{23} + \frac{T_Y}{Z}}{r_{31}x + r_{32}y + r_{33} + \frac{T_Z}{Z}}$$

Plane+Perspective(projective)

equation of a plane

$$\begin{bmatrix} a & b & c \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = 1$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T\begin{bmatrix} a & b & c \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = A\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad A = R + T\begin{bmatrix} a & b & c \end{bmatrix}$$

3d rigid motion

Plane+perspective (contd.)

$$x' = \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1}$$

scale ambiguity

$$y' = \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1}$$

find a's by least squares

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Displacement Models

Translation

$$x' = x + b_1 \qquad y' = y + b_2$$

Biquadratic

$$x' = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 y^2 + a_6 xy$$
$$y' = a_7 + a_8 x + a_9 y + a_{10} x^2 + a_{11} y^2 + a_{12} xy$$

Rigid

$$x' = x\cos\theta - y\sin\theta + b_1$$
$$y' = x\sin\theta + y\cos\theta + b_2$$

Bilinear

$$x' = a_1 + a_2 x + a_3 y + a_4 xy$$
$$y' = a_5 + a_6 x + a_7 y + a_8 xy$$

Affine

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

Pseudo Perspective

$$x' = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy$$
$$y' = a_6 + a_7 x + a_8 y + a_4 xy + a_5 y^2$$

Projective

$$x' = \frac{a_1 x + a_2 y + b_1}{c_1 x + c_2 y + 1}$$
$$y' = \frac{a_3 x + a_4 y + b_1}{c_1 x + c_2 y + 1}$$

Displacement Models (contd)

- Translation
  - simple
  - used in block matching
  - no zoom, no rotation, no pan and tilt
- Rigid
  - rotation and translation
  - no zoom, no pan and tilt

Displacement Models (contd)

- Affine
  - rotation about optical axis only
  - can not capture pan and tilt
  - orthographic projection
- Projective
  - exact eight parameters (3 rotations, 3 translations and 2 scalings)
  - difficult to estimate

Displacement Models (contd)

- Biquadratic
  - obtained by second order Taylor series
  - 12 parameters
- Bilinear
  - obtained from biquadratic model by removing square terms
  - most widely used
  - not related to any physical 3D motion
- Pseudo-perspective
  - obtained by removing two square terms and constraining four remaining to 2 degrees of freedom

Instantaneous Velocity Model

3-D Rigid Motion

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} 1 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 1 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$\begin{bmatrix} X'-X \\ Y'-Y \\ Z'-Z \end{bmatrix} = \begin{bmatrix} 0 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 0 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

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

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \left(\begin{bmatrix} 0 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 0 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

Orthographic Projection

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

$$u = \dot{x} = V_1 + \Omega_2 Z - \Omega_3 y$$
$$v = \dot{y} = V_2 + \Omega_3 x - \Omega_1 Z$$

(u,v) is optical flow

Perspective Projection (arbitrary flow)

$$x = \frac{fX}{Z} \qquad u = \dot{x} = \frac{fZ\dot{X} - fX\dot{Z}}{Z^2} = f\frac{\dot{X}}{Z} - x\frac{\dot{Z}}{Z}$$

$$y = \frac{fY}{Z} \qquad v = \dot{y} = \frac{fZ\dot{Y} - fY\dot{Z}}{Z^2} = f\frac{\dot{Y}}{Z} - y\frac{\dot{Z}}{Z}$$

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

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

Plane+orthographic(Affine)

$$Z = a + bX + cY$$

$$u = V^1 + \Omega^2 Z - \Omega^3 y$$

$$u = b^1 + a^1 x + a^2 y$$

$$\mathbf{u} = \mathbf{A}\,\mathbf{x} + \mathbf{b}$$

$$b_1 = V_1 + a\Omega_2$$
$$a_1 = b\Omega_2$$
$$a_2 = c\Omega_2 - \Omega_3$$

$$b_2 = V_2 - a\Omega_1$$
$$a_3 = \Omega_3 - b\Omega_1$$
$$a_4 = -c\Omega_1$$

Plane+Perspective (pseudo perspective)

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

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

$$Z = a + bX + cY$$

$$\frac{1}{Z} = \frac{1}{a} - \frac{b}{a}x - \frac{c}{a}y$$

$$u = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy$$

$$v = a_{64} + a_7 x + a_8 y + a_4 xy + a_5 y^2$$

Measurement of Image Motion

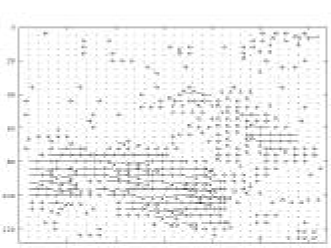
- Local Motion (Optical Flow)
- Global Motion (Frame Alignment)

Computing Optical Flow

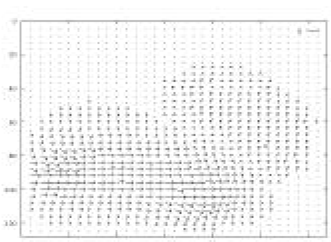
Image from Hamburg Taxi seq

Image from Hamburg Taxi seq

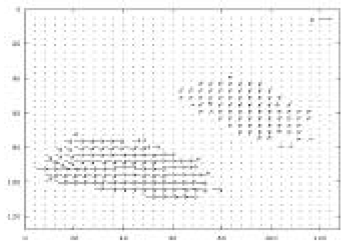
Fleet & Jepson optical flow



Horn & Schunck optical flow



Tian & Shah optical flow



Horn&Schunck Optical Flow

$$f(x,y,t)=f(x+dx,y+dy,t+dt)$$

Taylor Series

$$f(x,y,t)=f(x,y,t)+\frac{\partial f}{\partial x}dx+\frac{\partial f}{\partial y}dy+\frac{\partial f}{\partial t}dt$$

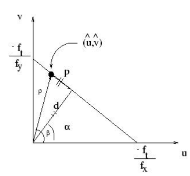
$$f_x dx+f_y dy+f_t dt=0$$

$$f_x u+f_y v+f_t=0$$ brightness constancy eq

Interpretation of optical flow eq

$$f_x u+f_y v+f_t=0$$

$$v=-\frac{f_x}{f_y}u-\frac{f_t}{f_y}$$



d=normal flow

p=parallel flow

$$d=\frac{f_t}{\sqrt{f_x^2+f_y^2}}$$

Equation of st.line

Horn&Schunck (contd)

$$\iint\{(f_x u+f_y v+f_t)^2+\lambda(u_x^2+u_y^2+v_x^2+v_y^2)\}dx\,dy$$ variational calculus

min

$$(f_x u+f_y v+f_t)f_x+\lambda(\Delta^2 u)=0$$

$$(f_x u+f_y v+f_t)f_y+\lambda(\Delta^2 v)=0$$

$$u=u_{av}-f_x\frac{P}{D}$$

$$v=v_{av}-f_y\frac{P}{D}$$

discrete version

$$(f_x u+f_y v+f_t)f_x+\lambda(u-u_{av})=0$$

$$(f_x u+f_y v+f_t)f_y+\lambda(v-v_{av})=0$$

$$P=f_x u_{av}+f_y v_{av}+f_t$$

$$D=\lambda+f_x^2+f_y^2$$

Algorithm-1

- k=0
- Initialize u^K v^K
- Repeat until some error measure is satisfied

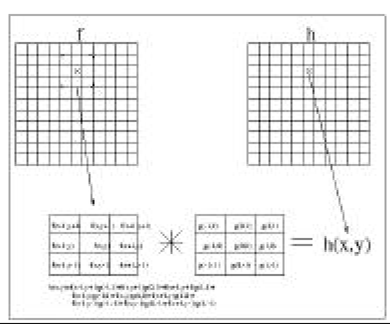
$$u^K=u_{av}^{k-1}-f_x\frac{P}{D}$$

$$v=v_{av}^{K-1}-f_y\frac{P}{D}$$

$$P=f_x u_{av}+f_y v_{av}+f_t$$

$$D=\lambda+f_x^2+f_y^2$$

Convolution

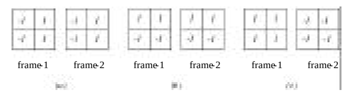


Convolution (contd)

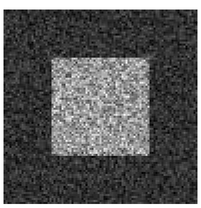
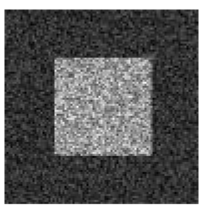
$$h(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x+i, y+j) g(i,j)$$

$$h(x,y) = f(x,y) * g(x,y)$$

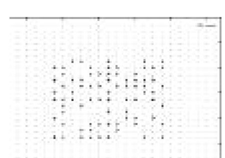
Derivative Masks

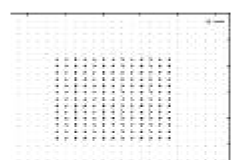


Synthetic Images

Results


One iteration


10 iterations

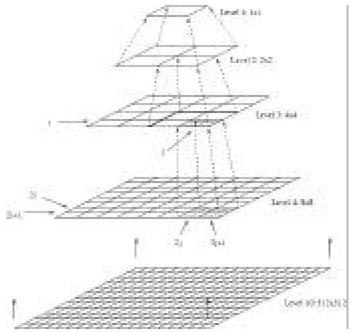
Comments

- Algorithm-1 works only for small motion.
- If object moves faster, the brightness changes rapidly, 2x2 or 3x3 masks fail to estimate spatiotemporal derivatives.
- Pyramids can be used to compute large optical flow vectors.

Pyramids

- Very useful for representing images.
- Pyramid is built by using multiple copies of image.
- Each level in the pyramid is 1/4 of the size of previous level.
- The lowest level is of the highest resolution.
- The highest level is of the lowest resolution.

Pyramid

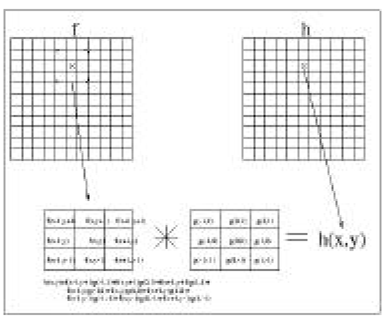


Gaussian Pyramids

$$g_l(i,j)=\sum_{m=-2}^{2}\sum_{n=-2}^{2}w(m,n)g_{l-1}(2i+m,2j+n)$$

$$g_l = REDUCE[g_{l-1}]$$

Convolution



Gaussian Pyramids

$$g_{l,n}(i,j)=\sum_{p=-2}^{2}\sum_{q=-2}^{2}w(p,q)g_{l,n-1}\left(\frac{i-p}{2},\frac{j-q}{2}\right)$$

$$g_{l,n} = EXPAND[g_{l,n-1}]$$

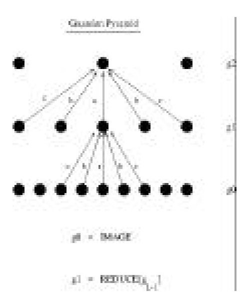
Reduce (1D)

$$g_l(i)=\sum_{m=-2}^{2}\hat{w}(m)g_{l-1}(2i+m)$$

$$g_l(2)=\hat{w}(-2)g_{l-1}(4-2)+\hat{w}(-1)g_{l-1}\hat{w}(4-1)+$$
$$\hat{w}(0)g_{l-1}(4)+\hat{w}(1)g_{l-1}(4+1)+\hat{w}(2)g_{l-1}(4+2)$$

$$g_l(2)=\hat{w}(-2)g_{l-1}(2)+\hat{w}(-1)g_{l-1}\hat{w}(3)+$$
$$\hat{w}(0)g_{l-1}(4)+\hat{w}(1)g_{l-1}(5)+\hat{w}(2)g_{l-1}(6)$$

Reduce



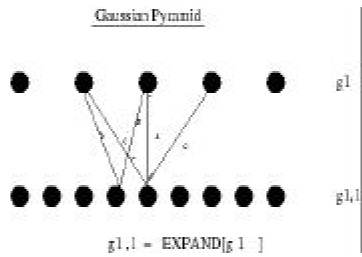
Expand (1D)

$$g_{l,n}(i) = \sum_{p=-2}^{2} \hat{w}(p) g_{l,n-1}(\frac{i-p}{2})$$

$$g_{l,n}(4) = \hat{w}(-2) g_{l,n-1}(\frac{4-2}{2}) + \hat{w}(-1) g_{l,n-1}(\frac{4-1}{2}) +$$
$$\hat{w}(0) g_{l,n-1}(\frac{4}{2}) + \hat{w}(1) g_{l,n-1}(\frac{4+1}{1}) + \hat{w}(2) g_{l,n-1}(\frac{4+2}{2})$$

$$g_{l,n}(4) = \hat{w}(-2) g_{l,n-1}(1) + \hat{w}(0) g_{l,n-1}(2) + \hat{w}(2) g_{l,n-1}(3)$$

Expand



Convolution Mask

$$[w(-2), w(-1), w(0), w(1), w(2)]$$

Convolution Mask

- Separable

$$w(m,n) = \hat{w}(m)\hat{w}(n)$$

•Symmetric

$$\hat{w}(i) = \hat{w}(-i)$$

$$[c, b, a, b, c]$$

Convolution Mask

- The sum of mask should be 1.

$$a + 2b + 2c = 1$$

•All nodes at a given level must contribute the same total weight to the nodes at the next higher level.

$$a + 2c = 2b$$

Convolution Mask

$$\hat{w}(0) = a$$

$$\hat{w}(-1) = \hat{w}(1) = \frac{1}{4}$$

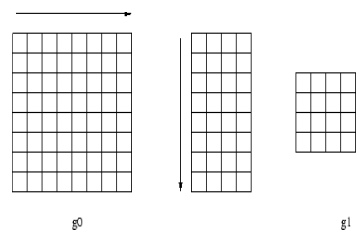
$$\hat{w}(-2) = \hat{w}(2) = \frac{1}{4} - \frac{a}{2}$$

a=.4 GAUSSIAN, a=.5 TRINGULAR

Gaussian

$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

Separability



Algorithm

- Apply 1 -D mask to alternate pixels along each row of image.
- Apply 1 -D mask to each pixel along alternate columns of resultant image from previous step.

Gaussian Pyramid

Laplacian Pyramids

- Similar to edge detected images.
- Most pixels are zero.
- Can be used for image compression.

$$L_1 = g_1 - EXPAND[g_2]$$

$$L_2 = g_2 - EXPAND[g_3]$$

$$L_3 = g_3 - EXPAND\ [g_4]$$

Coding using Laplacian Pyramid

•Compute Gaussian pyramid

$$g_1, g_2, g_3, g_4$$

•Compute Laplacian pyramid

$$L^1 = g^1 - EXPAND[g^2]$$
$$L^2 = g^2 - EXPAND[g^3]$$
$$L^3 = g^3 - EXPAND[g^4]$$

•Code Laplacian pyramid

Decoding using Laplacian pyramid

• Decode Laplacian pyramid.
• Compute Gaussian pyramid from Laplacian pyramid.

$$g_3 = EXPAND[g_4] + L_3$$
$$g_2 = EXPAND[g_3] + L_2$$
$$g_1 = EXPAND[g_2] + L_1$$

- is reconstructed image.

Algorithm

- Generate Laplacian pyramid Lo of orange image.
- Generate Laplacian pyramid La of apple image.
- Generate Laplacian pyramid Lc by copying left half of nodes at each level from apple and right half of nodes from orange pyramids.
- Reconstruct combined image from Lc.

Algorithm-2 (Optical Flow)

- Create Gaussian pyramid of both frames.
- Repeat
  - apply algorithm-1 at the current level of pyramid.
  - propagate flow by using bilinear interpolation to the next level, where it is used as an initial estimate.
  - Go back to step 2

Horn&Schunck Method

- Good only for translation model.
- Oversmoothing of boundaries.
- Does not work well for real sequences.

Other Optical Flow Methods

Anandan

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

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

Anandan

$$\mathbf{u}(\mathbf{x}) = \mathbf{X}(\mathbf{x})\mathbf{a}$$

$$E(\boldsymbol{d}a) = \sum_{x} (f_t + f_{\mathbf{x}}^T \boldsymbol{d}u)^2$$

$$E(\boldsymbol{d}a) = \sum_{x} (f_t + f_{\mathbf{x}}^T \mathbf{X}\boldsymbol{d}a\ )^2$$

Anandan

$$\left[\sum \mathbf{X}^{\mathrm{T}} (\mathbf{f}_{\mathbf{X}})(\mathbf{f}_{\mathbf{x}})^T \mathbf{X}\right] \boldsymbol{d}\mathbf{a} = -\sum \mathbf{X}^T \mathbf{f}_{\mathbf{X}} f_t$$

Basic Components

- Pyramid construction
- Motion estimation
- Image warping
- Coarse-to-fine refinement

Image Warping

$$X' = X - U = X - (AX + b)$$
$$X' = (I - A)X - b$$
$$X' = A'X - b$$
$$X' + b = A'X$$
$$(A')^{-1}(X' + b) = X$$

Image at time t: X
Image at time t-1: X'

Image Warping

$$X' \rightarrow X''$$

$$(A')^{-1}(X' + b) = X''$$
$$(X' + b) = (A')X''$$
$$X' = (A')X'' - b$$

Bi-linear Interpolation

Four nearest points of (x,y) are:

$$(\underline{x}, \underline{y}), (\overline{x}, \underline{y}), (\underline{x}, \overline{y}), (\overline{x}, \overline{y})$$

$$\underline{x} = \mathrm{int}(x)$$
$$\underline{y} = \mathrm{int}(y)$$
$$\overline{x} = x + 1$$
$$\overline{y} = y + 1$$

$$f(x',y') = \overline{\boldsymbol{e}_x}\,\overline{\boldsymbol{e}_y}\, f(\underline{x},\underline{y}) + \underline{\boldsymbol{e}_x}\,\overline{\boldsymbol{e}_y}\, f(\overline{x},\underline{y}) +$$
$$\overline{\boldsymbol{e}_x}\,\underline{\boldsymbol{e}_y}\, f(\underline{x},\overline{y}) + \underline{\boldsymbol{e}_x}\,\underline{\boldsymbol{e}_y}\, f(\overline{x},\overline{y})$$

$$\overline{\boldsymbol{e}_x} = \overline{x} - x$$

$$\overline{\boldsymbol{e}_y} = \overline{y} - y$$

$$\underline{\boldsymbol{e}_x} = x - \underline{x}$$

$$\underline{\boldsymbol{e}_y} = y - \underline{y}$$

Mann & Picard

Projective Flow (weighted)

$$u_f f_x + v_f f_y + f_t = 0$$

$$\mathbf{u}_m^T \mathbf{f}_\mathbf{x} + f_t = 0$$

$$\mathbf{x}' = \frac{A\mathbf{x} + \mathbf{b}}{\mathbf{C}^\mathbf{T}\mathrm{x} + 1}$$

Projective Flow (weighted)

$$\boldsymbol{e}_{flow} = \sum (\mathbf{u}_\mathbf{m}^\mathbf{T} \mathbf{f}_\mathbf{X} + f_\mathbf{t})^2$$

$\Downarrow$ **minimize**

Projective Flow (weighted)

$$(\sum \boldsymbol{f}\boldsymbol{f}^T)\mathbf{a} = \sum (\mathbf{x}^\mathbf{T}\mathbf{f}_x - f_t)\boldsymbol{f}$$

$$a = [a_{11}, a_{12}, b_1, a_{21}, a_{22}, b_2, c_1, c_2]^T$$

Projective Flow (unweighted)

Bilnear

$$\mathbf{x}' = \frac{A\mathbf{x} + \mathbf{b}}{\mathbf{C}^T\mathbf{x} + 1}$$

Taylor Series

$$u_m + x' = a_1 + a_2 x + a_3 y + a_4 xy$$
$$v_m + y' = a_5 + a_6 x + a_7 y + a_8 xy$$

Pseudo-Perspective

$$\mathbf{x}' = \frac{A\mathbf{x} + \mathbf{b}}{\mathbf{C}^T\mathbf{x} + 1}$$

Taylor Series

$$x' + u_m = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy$$
$$y' + v_m = a_6 + a_7 x + a_8 y + a_4 xy + a_5 y^2$$

Projective Flow (unweighted)

$$\boldsymbol{e}_{flow} = \sum (\mathbf{u}_\mathbf{m}^\mathbf{T}\mathbf{f}_\mathbf{X} + f_\mathbf{t})^2$$

Minimize

Bilinear and Pseudo-Perspective

$$(\sum \Phi\Phi^T)\mathbf{q} = -\sum f_t\Phi$$

$$\Phi^T = [f_x(xy, x, y, 1), \quad f_y(xy, x, y, 1)]$$ **bilinear**

$$\Phi^T = [f_x(x, y, 1) \quad f_y(x, y, 1) \quad c_1 \quad c_2]$$ **Pseudo p**

$$c_1 = x^2 f_x + xyf_x$$
$$c_2 = xyf_x + y^2 f_y$$

Algorithm

- Estimate "q" (using approximate model, e.g. bilinear model).
- Relate "q" to "p"
  - select four points S1, S2, S3, S4
  - apply approximate model using "q" to compute
  - estimate exact "p":

True Projective

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = \begin{bmatrix} x_k & y_k & 1 & 0 & 0 & 0 & -x_k x'_k & -y_k x'_k \\ 0 & 0 & 0 & x_k & y_k & 1 & -x_k y'_k & -y_k y'_k \end{bmatrix} \mathbf{a}$$

$$\mathbf{a} = [a_1 \quad a_2 \quad b_1 \quad a_3 \quad a_4 \quad b_2 \quad c_1 \quad c_1]^T$$

Final Algorithm

- A Gaussian pyramid of three or four levels is constructed for each frame in the sequence.
- The parameters “p” are estimated at the top level of the pyramid, between the two lowest resolution images, “g” and “h”, using algorithm-1 (see figure).

Final Algorithm

- The estimated “p” is applied to the next higher resolution image in the pyramid, to make images at that level nearly congruent.
- The process continues down the pyramid until the highest resolution image in the pyramid is reached.

Video Mosaics

- Mosaic aligns different pieces of a scene into a larger piece, and seamlessly blend them.
  - High resolution image from low resolution images
  - Increased filed of view

Steps in Generating A Mosaic

- Take pictures
- Pick reference image
- Determine transformation between frames
- Warp all images to the same reference view

Applications of Mosaics

- Virtual Environments
- Computer Games
- Movie Special Effects
- Video Compression

Webpages

- http://n1nlf1.eecg.toronto.edu/tip.ps.gz
  Video Orbits of the projective group, S. Mann and R. Picard.
- http://wearcam.org/pencigraphy (C code for generating mosaics)

Webpages

- http://ww-bcs.mit.edu/people/adelson/papers.html
  - The Laplacian Pyramid as a compact code, Burt and Adelson, IEEE Trans on Communication, 1983.
- J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical Model-Based Motion Estimation", ECCV-92, pp 237-22.

Webpages

- http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/v-source.html (c code for several optical flow algorithms)
- ftp://csd.uwo.ca/pub/vision

  Performance of optical flow techniques (paper)

Barron, Fleet and Beauchermin

Webpages

- http://www.wisdom.weizmann.ac.il/~irani/abstracts/mosaics.html ("Efficient representations of video sequences and their applications", Michal Irani, P. Anandan, Jim Bergen, Rakesh Kumar, and Steve Hsu)
- R. Szeliski. "Video mosaics for virtual environments", IEEE Computer Graphics and Applications, pages,22-30, March 1996.

Part II

Change Detection and Tracking

Contents

- Change Detection
- Pfinder
- Mixture of Gaussians
- Kanade
- W4
- Tracking People Using Color
- Kalman Filter

Change Detection

Main Points

- Detect pixels which are changing due to motion of objects.
- Not necessarily measure motion (optical flow), only detect motion.
- A set of connected pixels which are changing may correspond to moving object.

Picture Difference

$$D_i(x,y) = \begin{cases} 1 & if \quad DP(x,y) > T \\ 0 & \ldots\ldots otherwise \end{cases}$$

$$DP(x,y) = | f_i(x,y) - f_{i-1}(x,y) |$$

$$DP(x,y) = \sum_{i=-m}^{m} \sum_{j=-m}^{m} | f_i(x+i, y+j) - f_{i-1}(x+i, y+j) |$$

$$DP(x,y) = \sum_{i=-m}^{m} \sum_{i=-m}^{m} \sum_{k=-m}^{m} | f_i(x+i, y+j) - f_{i+k}(x+i, y+j) |$$

Background Image

- The first image of a sequence without any moving objects, is background image.
- Median filter

$$B(x,y) = median\ (f_1(x,y), \ldots, f_n(x,y))$$

PFINDER

Pentland

Pfinder

- Segment a human from an arbitrary complex background.
- It only works for single person situations.
- All approaches based on background modeling work only for fixed cameras.

Algorithm

- Learn background model by watching 30 second video
- Detect moving object by measuring deviations from background model
- Segment moving blob into smaller blobs by minimizing covariance of a blob
- Predict position of a blob in the next frame using Kalman filter
- Assign each pixel in the new frame to a class with max likelihood.
- Update background and blob statistics

Learning Background Image

- Each pixel in the background has associated mean color value and a covariance matrix.
- The color distribution for each pixel is described by Gaussian.
- YUV color space is used.

Detecting Moving Objects

- After background model has been learned, Pfinder watches for large deviations from the model.
- Deviations are measured in terms of Mahalanobis distance in color.
- If the distance is sufficient then the process of building a blob model is started.

Detecting Moving Objects

- For each of k blob in the image, log-likelihood is computed

$$d_k = -.5(y - \boldsymbol{m}_k)^T K_k^{-1}(y - \boldsymbol{m}_k) - .5\ln|K_k| - .5m\ln(2\lambda)$$

- Log likelihood values are used to classify pixels

$$s(x, y) = \arg\max_k (d_k(x, y))$$

Updating

•The statistical model for the background is updated.

$$K^t = E[(y - \boldsymbol{m}^t)(y - \boldsymbol{m}^t)^T]$$
$$\boldsymbol{m}^t = (1 - \boldsymbol{a})\boldsymbol{m}^{t-1} + \boldsymbol{a}y$$

• The statistics of each blob (mean and covariance) are re-computed.

W4 (Who, When, Where, What)

Davis

W4

• Compute “minimum”(M(x)), “maximum” (N(x)), and “largest absolute difference” (L(x)).

$$D_i(x, y) = \begin{cases} 1 \quad if \quad |M(x, y) - f_i(x, y)| > L(x, y) or \\ \quad |N(x, y) - f_i(x, y)| > L(x, y) \\ 0 \quad \dots \quad otherwise \end{cases}$$

- Theoretically, the performance of this tracker should be worse than others.
- Even if one value is far away from the mean, then that value will result in an abnormally high value of L.
- Having short training time is better for this tracker.

Limitations

- Multiple people
- Occlusion
- Shadows
- Slow moving people
- Multiple processes (swaying of trees..)

Webpage

- Http://www.cs.cmu.edu/~vsam (DARPA Visual Surveillance and Monitoring program)

Skin Detection

Kjeldsen and Kender

Training

- Crop skin regions in the training images.
- Build histogram of training images.
- Ideally this histogram should be bi-modal, one peak corresponding to the skin pixels, other to the non-skin pixels.
- Practically there may be several peaks corresponding to skin, and non-skin pixels.

Training

- Apply threshold to skin peaks to remove small peaks.
- Label all gray levels (colors) under skin peaks as "skin", and the remaining gray levels as "non-skin".
- Generate a look-up table for all possible colors in the image, and assign "skin" or "non-skin" label.

Detection

- For each pixel in the image, determine its label from the "look-up table" generated during training.

Building Histogram

- Instead of incrementing the pixel counts in a particular histogram bin:
  - for skin pixel increment the bins centered around the given value by a Gaussian function.
  - For non-skin pixels decrement the bins centered around the given value by a smaller Gaussian function.

Tracking People Using Color

Fieguth and Terzopoulos

- Computer mean color vector for each sub region.

$$(r_i, g_i, b_i) = \frac{1}{|R_i|} \sum_{(x,y) \in R_i} (r(x,y), g(x,y), b(x,y))$$

Fieguth and Terzopoulos

- Compute goodness of fit.

$$\Psi_i = \frac{\max\left\{\frac{r_i}{\bar{r}_i}, \frac{g_i}{\bar{g}_i}, \frac{b_i}{\bar{b}_i}\right\}}{\min\left\{\frac{r_i}{\bar{r}_i}, \frac{g_i}{\bar{g}_i}, \frac{b_i}{\bar{b}_i}\right\}}$$

Target Measurement

Fieguth and Terzopoulos

- Tracking

$$\Psi(x_H, y_H) = \sum_{i=1}^{N} \frac{\Psi_i(x_H + x_i, y_H + y_i)}{N}$$

$$(\hat{x}, \hat{y}) = \arg_{(x_H, y_H)} \min\{\ \Psi(x_H, y_H)\}$$

Fieguth and Terzopoulos

- Non-linear velocity estimator

$$if \quad (\boldsymbol{r}(f).\boldsymbol{r}(f-1) > 0) \quad v(f) \quad += \quad \boldsymbol{d}\frac{\operatorname{sgn}(\boldsymbol{r}(f))}{\Delta t}$$

$$if \quad (\boldsymbol{r}(f).v(f-1) < 0) \quad v(f) \quad += \quad \boldsymbol{d}\frac{\operatorname{sgn}(\boldsymbol{r}(f))}{\Delta t}$$

$$if \quad (\boldsymbol{r}(f) = 0) \quad v(f) \quad += \quad \boldsymbol{d}\frac{\operatorname{sgn}(v(f))}{2\Delta t}$$

Bibliography

- .J. K. Aggarwal and Q. Cai, “Human Motion Analysis: A Review”, *Computer Vision and Image Understanding*, Vol. 73, No. 3, March, pp. 428-440, 1999
- .Azarbayejani, C. Wren and A. Pentland, “Real-Time 3D Tracking of the Human Body”, MIT Media Laboratory, Perceptual Computing Section, TR No. 374, May 1996
- .W.E.L. Grimson *et. al.*, “Using Adaptive Tracking to Classify and Monitor Activities in a Site”, *Proceedings of Computer Vision and Pattern Recognition,* Santa Barbara, June 23-25, 1998, pp. 22-29

Bibliography

- .Takeo Kanade *et. al*. “Advances in Cooperative Multi-Sensor Video Surveillance”, *Proceedings of Image Understanding workshop*, Monterey California, Nov 20-23, 1998, pp. 3-24
- .Haritaoglu I., Harwood D, Davis L, “W^4 - Who, Where, When, What: A Real Time System for Detecting and Tracking People”, *International Face and Gesture Recognition Conference,* 1998
- .Paul Fieguth , Demetri Terzopoulos, “Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates”, *CVPR 1997,* pp. 21-27

Part III

VIDEO UNDERSTANDING

Contents

- Monitoring Human Behavior In an Office
- Model-Based Human Activities Recognition
- Visual Lipreading
- Hand Gesture Recognition
- Action Recognition using temporal templates

Monitoring Human Behavior In an Office Environment

Goals of the System

- Recognize human actions in a room for which **prior knowledge** is available.
- Handle multiple people
- Provide a textual description of each action
- Extract "key frames" for each action

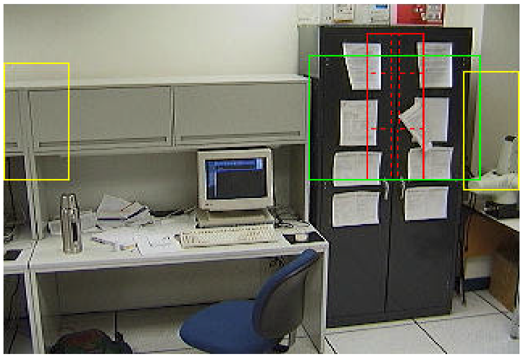
Possible Actions

- **Enter**
- **Leave**
- **Sitting** or **Standing**
- **Picking Up Object**
- **Put Down Object**
- **…..**

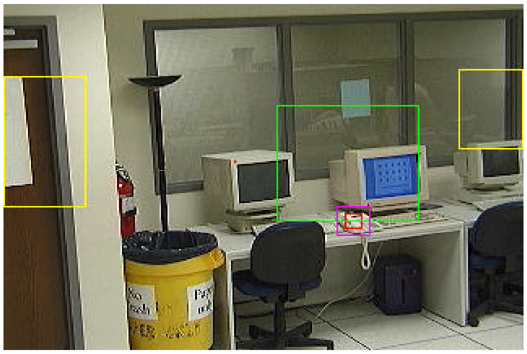
Prior Knowledge

- Spatial layout of the scene:
  - Location of **entrances** and **exits**
  - Location of **objects** and some information about how they are use
- Context can then be used to improve recognition and save computation

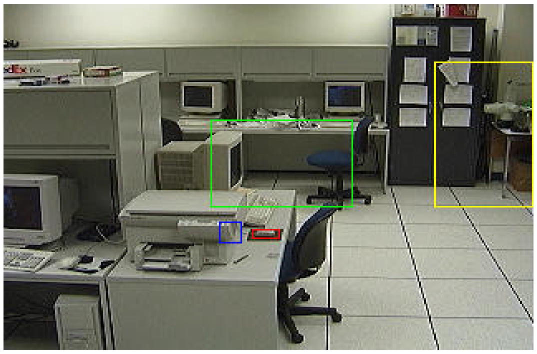
Layout of Scene 1



Layout of Scene 2



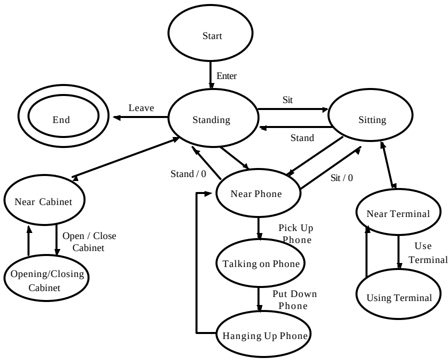
Layout of Scene 4



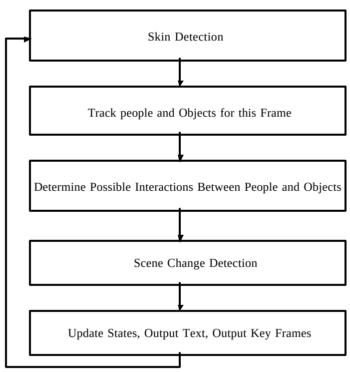
Major Components

- Skin Detection
- Tracking
- Scene Change Detection
- Action Recognition

State Model For Action Recognition



Flow of the System



Key Frames

- Why get key frames?
  - Key frames take less space to store
  - Key frames take less time to transmit
  - Key frames can be viewed more quickly
- We use heuristics to determine when key frames are taken
  - Some are taken before the action occurs
  - Some are taken after the action occurs

Key Frames

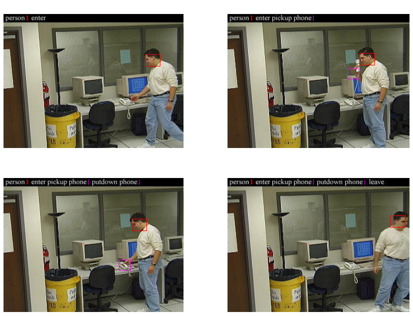
- <u>“Enter” key frames</u>: as the person leaves the entrance/exit area
- <u>“Leave” key frames</u>: as the person enters the entrance/exit area
- <u>“Standing/Sitting” key frames</u>: after the tracking box has stopped moving up or down respectively
- <u>“Open/Close” key frames</u>: when the % of changed pixels stabilizes

Results

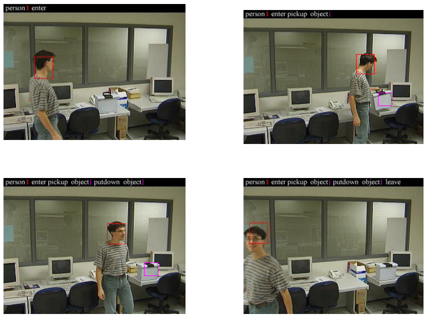
Key Frames Sequence 1 (350 frames), Part 1

Key Frames Sequence 1 (350 frames), Part 2

Key Frames Sequence 2 (200 frames)

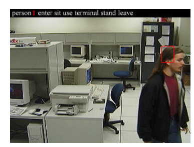


Key Frames Sequence 3 (200 frames)



Key Frames Sequence 4 (399 frames), Part 1

Key Frames Sequence 4 (399 frames), Part 2

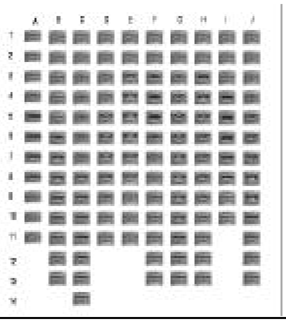


Generalizations

- Increased field of view
  - Arbitrary positioned un-calibrated cameras
- Activity Recognition without a priori knowledge
  - Automatically learn activities by observing
  - Determine which objects persons interact with frequently
  - Separate out object motion from human motion, to determine objects being interacted with
- Real-time implementation

Visual Lipreading

Image Sequences of "A" to "J"



Particulars

- Problem: Pattern differ spatially
- Solution: Spatial registration using SSD
- Problem : Articulations vary in length, and thus, in number of frames.
- Solution: Dynamic programming for temporal warping of sequences.
- Problem: Features should have compact representation.
- Solution: Principle Component Analysis.

Feature Subspace Generation

- Generate a lower dimension subspace onto which image sequences are projected to produce a vector of coefficients.
- Components
  - Sample Matrix
  - Most Expressive Features

Generating the Sample Matrix

- Consider e letters, each of which has a training set of K sequences. Each sequence is compose of images:

$$I_1, I_2, \ldots, I_P$$

- Collect all gray-level pixels from all images in a sequence into a vector:

$$u = (I_1(1,1), \ldots, I_1(M,N), I_2(1,1), \ldots, I_2(M,N), \ldots I_P(1,1), \ldots, I_P(M,N))$$

. Generating the Sample Matrix

- For letter $\boldsymbol{w}$, collect vectors into matrix T

$$T_{\boldsymbol{w}} = \left[u^1, u^2, \ldots u^K\right]$$

- Create sample matrix A:

$$A = \left[T_1, T_2, \ldots T_{\boldsymbol{e}}\right]$$

•The eigenvectors of a matrix $L = AA^T$ are defined as:

The Most Expressive Features

- $\boldsymbol{f}$ is an orthonormal basis of the sample matrix.

•Any image sequence, u, can be represented as:

$$u = \sum_{n=1}^{Q} a_{n\boldsymbol{f}_n} = \boldsymbol{f}a$$

- Use Q most significant eigenvectors.
- The linear coefficients can be computed as:

$$a_n = u^T \boldsymbol{f}_n$$

Training Process

- Model Generation
  - Warp all the training sequences to a fixed length.
  - Perform spatial registration (SSD).
  - Perform PCA.
  - Select Q most significant eigensequences, and compute coefficient vectors “a”.
  - Compute mean coefficient vector for each letter.

Recognition

- Warp the unknown sequence.
- Perform spatial registration.
- Compute:

$$a_i^x = u_x^T . \boldsymbol{f}_i$$

$$d^w = \| a^w - a^x \|$$

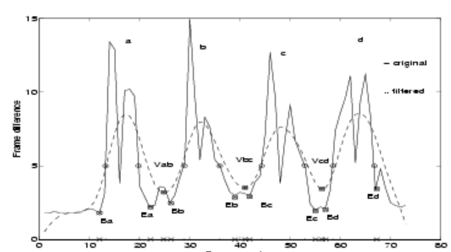
- Determine best match by

Extracting letters from Connected Sequences

- Average absolute intensity difference function

$$f(n) = \frac{1}{MN} \sum_{x=1}^{M} \sum_{y=1}^{N} \| I_n(x,y) - I_{n-1}(x,y) \|$$

- f is smoothed to obtain g.
- Articulation intervals correspond to peaks and non-articulation intervals correspond to valleys in “g”.



Extracting letters from Connected Sequences

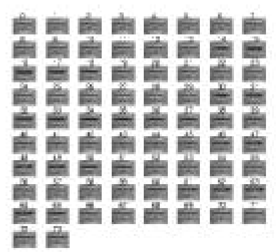
- Detect valleys in g.
- From valley locations in g, find locations where f crosses high threshold.
- Locate beginning and ending frames.

A 12-22

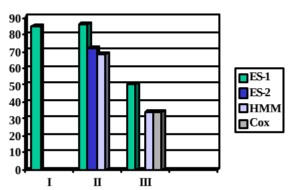
B 26-39

C 42-55

D 57-67



Results

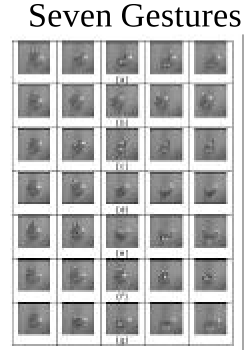


I: “A” to “J” one speaker, 10 training seqs
II. “A” to “M”, one speaker, 10 training seqs
III. “A” to “Z”, ten speakers, two training seqs/letter/person

Show Video Clip

Hand Gesture Recognition

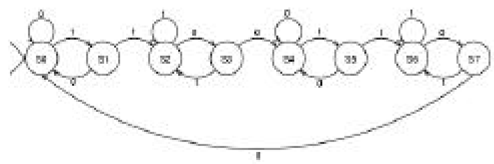
Seven Gestures



Gesture Phases

- Hand fixed in the start position.
- Fingers or hand move smoothly to gesture position.
- Hand fixed in gesture position.
- Fingers or hand return smoothly to start position.

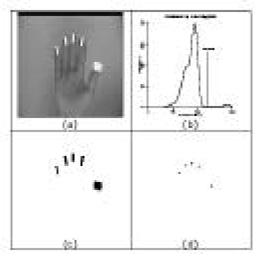
Finite State Machine



Main Steps

- Detect fingertips.
- Create fingertip trajectories using motion correspondence of fingertip points.
- Fit vectors and assign motion code to unknown gesture.
- Match.

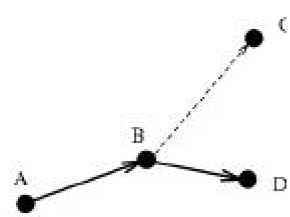
Detecting Fingertips



Proximal Uniformity Constraint

- Most objects in the real world follow smooth paths and cover small distance in a small time.
  - Given a location of point in a frame, its location in the next fame lies in the proximity of its previous location.
  - The resulting trajectories are smooth and uniform.

Proximal Uniformity Constraint

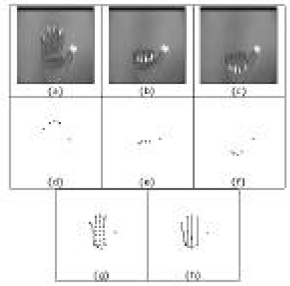


Proximal Uniformity Constraint

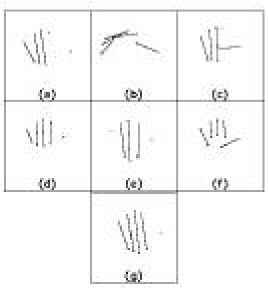
Establish correspondence by minimizing:

$$\boldsymbol{d}(X_p^{k-1}, X_q^k, X_r^{k+1}) = \frac{\| \overline{X_p^{k-1} X_q^k} - \overline{X_q^k X_r^{k+1}} \|}{\sum_{x=1}^{m}\sum_{z=1}^{m} \| \overline{X_x^{k-1} X_y^k} - \overline{X_y^k X_z^{k+1}} \|} + \frac{\| \overline{X_q^k X_r^{k+1}} \|}{\sum_{x=1}^{m}\sum_{z=1}^{m} \| \overline{X_y^k X_z^{k+1}} \|}$$

Vector Extraction



Vector Representation of Gestures



Results

Results

Run	Frames	L	R	U	D	T	G	S
1	200	√	√	√	√	√	√	√
2	250	√	√	√	√	√	√	√
3	250	√	√	√	X	√	√	√
4	250	√	√	√	√	√	√	√
5	300	√	√	√	√	√	√	√
6	300	√	√	√	√	√	√	√
7	300	√	√	√	√	√	√	√
8	300	√	√	√	√	√	√	√
9	300	√	√	√	√	*	*	*
10	300	√	√	√	√	√	√	√

L = Left, R = Right, U = Up, D = Down, T = Rotate, G = Grab, S = Stop, √ - Recognized, X - Not Recognized, * - Error in Sequence.

Virtual 3-D Blackboard:

Finger Tracking with a Single Camera

Andrew Wu

REU 1999

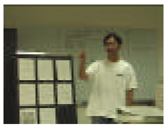
awu@uiuc.edu http://www.cs.ucf.edu/~vision

(go to REU99)

Project Goals

- Using computer vision, implement a virtual 3-D blackboard
- Program will parse 2-D image input, recording corresponding 3-D motion of a user's fingertip
- Motion of user's finger will be recognized as a certain type of 3-D gesture

Sample Picture

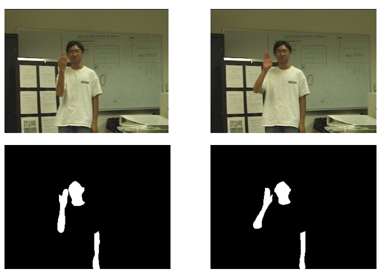


- •Single color camera
- •Static background
- •One person in picture
- •Consistent lighting

Skin Detection

- Color Predicate
  - – Skin-tones in RGB space marked by computer program
  - – Trained on several color images with hand-drawn binary masks
  - – Color Predicate data structure saved

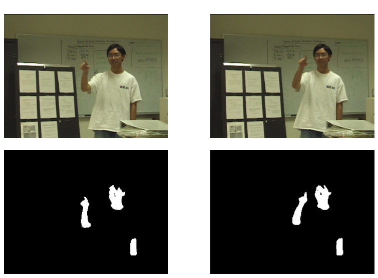
Example training images



Using the Color Predicate

- Check RGB values of every pixel in input image
- If RGB value satisfies Color Predicate, output as true in output binary image
- Median-filter binary output to remove noise and outliers

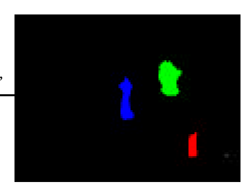
Results of skin detection



Separating regions

- Next step: demarcate connected skin regions
- Simple 8-connectivity algorithm that grows regions
- Cull three largest regions (presumably the head and two arms)

Three largest regions, pseudo-colored



Separating regions

•Find centroids for largest regions (regions alpha blended with original color image for effect)

centroids

Finding the arm

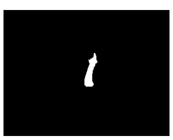
- Assume fastest moving centroid belongs to gesticulating arm
- Find largest delta between two skin frames

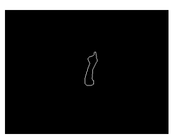
Difference picture between temporally proximal skin images

Difference picture for region of largest centroid delta, only

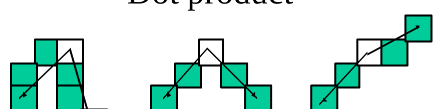
Outlining the arm

- Goal: find perimeter of isolated arm segment
- Assume contiguous region
- If pixel has 4-connectivity with black region, pixel is on the periphery





Dot product



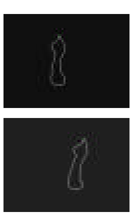
A dot B > 0 | A dot B = 0 | A dot B < 0

•Method: find two vectors between the current pixel and the pixels N steps away (N=3, 2, 2 above)

•Repeat procedure for all pixels in outline, searching for maximum dot product.

•Idea: largest dot product will be formed by two vectors extending in both directions from the fingertip

Results of dot product approach



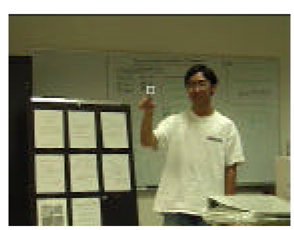
•Very good output

•Found finger in all cases when given proper outline of arm

•Perhaps can detect absence of finger from skin outline

•For test data, found best value of N to be 3 pixels (N being number of pixels to step away for vector calculation)

Video



•Program run on 7 continuous frames

•In sum: finds skin from color, arm from centroid speed, then finger from dot product of pixel outline

•Tracks finger fairly well

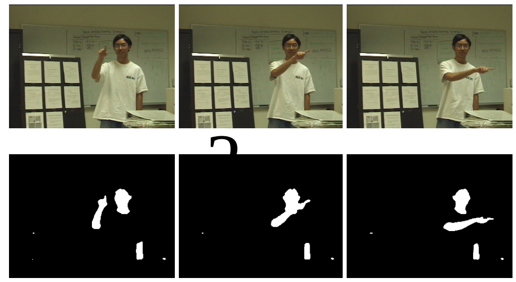
Next issue: Occlusion

•Comparatively easy to handle simple situations where hands and face are far apart (no occlusion)

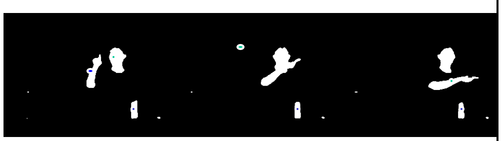
•Problems appear when one hand blocks face:

- -How to distinguish body parts?
- -Where are the centroids?
- -Can we find the finger?

Example images



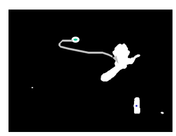
The problem



•Since the arm finding algorithm looks for region whose centroid has moved the most, the algorithm gets confused when centroids disappear and reappear

•Above, 3 centroids become 2 centroids, become 3

Current solution



•When centroids gets "lost", missing centroid is assumed to have been assimilated by the largest contiguous region

•So, place "ghost" centroid marker on top of centroid of largest region

•When centroid reappears, each centroid accesses the next frame to find closest future centroid.

•Also, ensure a one-to-one mapping (two centroids in one frame should not map to only one centroid in next)

Video

Tracking through occlusion, a demo

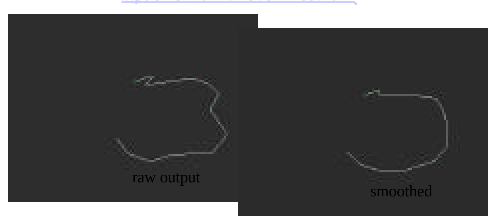
http:../sony/public_html/move.html

..\public_html\move.html

Updated images

http://www.cs.ucf.edu/~aw47967/move-fixed.html

..\public_html\move-fixed.html



Body Tracking

Why track the body?

- Helps us derive 3-D information
- Useful in correcting errors in finger tracking

Quo vadis?

("Where are you headed?", or Head Tracking)

Highest centroid (lowest Y) that does not belong to the arm should correspond to the head

- When centroids merge, do not track head, rather use previous data
- Simple approach, but works well

Approximating the shoulder

Need to guess shoulder location -- actual coordinates unimportant

- Relative distance to head should be consistent
- Assume shoulder is a certain distance from centroid of head

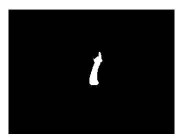
Procedure: Find approximate radius of head region

Shoulder <x,y> = Head Centroid <x,y> + <radius, radius * 1.618…>

(1.618… = (1 + sqrt(5))/2)

Finding the Elbow

For simplicity, assume elbow is part of skin image. If necessary, other techniques are applicable to images where this is not true



Erode arm region by several pixels

Using known finger location, find point on opposite side of arm

(that is, maximize distance between known finger point and unknown elbow position)

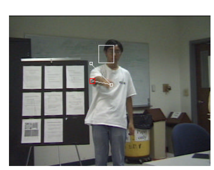
Results of Body Tracking

skeleton, superimposed

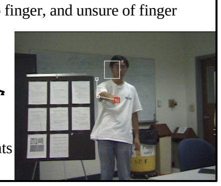
Improving the tracker

When finger occludes hand, tracker sometimes marks elbow as fingertip, because hand location has low curvature

So, if 'bicep' length is clearly greater than arm length, distance from shoulder to elbow is greater than distance from shoulder to finger, and unsure of finger location...



…swap finger and elbow points



Judging tracker accuracy

Based on magnitude and sign of dot product, we can assign a “confidence” value to each data set recorded

Graphically, these confidence values are represented as:

- •green – high confidence
- •dark green – lesser confidence
- •red -- low degree of confidence

Movie

2-D Body tracking with graphical tracker confidence display, the movie

..\public_html\tracker-tracking.html

Leaving Flatland

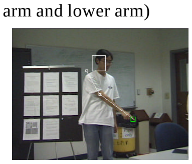
Armed with body data, we can begin to derive 3-D information

•First, assume orthographic projection

•Need to record “true” pixel length of Humerus, Radius (upper arm and lower arm)

•Assume that at some point in footage, arm is fully extended

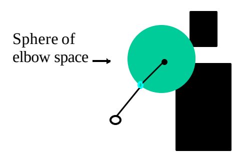
•Store largest lengths



Spherical kinematics

Humanoid bones have constant length

Thus, for example, since distance from elbow to shoulder is fixed, elbow can only move tangent to a certain sphere (sphere centered on shoulder)



Collapsing the hemisphere

Ignoring the half of the sphere that lies behind the plane of the body, we are left with the surface of one hemisphere

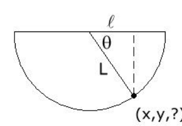
If we look at the hemisphere along the polar axis, we can see the entire surface of the hemisphere

That is, along the polar axis, we can collapse the hemisphere from 3-D to 2-D without loss of information, as well as recover the hemisphere from its 2-D projection.

Math

L = r = radius of sphere

l = length of projection of 3-D line onto 2-D circle



(View from above)

Geometrical:

Algebraic:

cos(theta) = l / L

theta = acos(l/L)

z = L sin(theta)

z = L sin acos(l/L)

x^2 + y^2 + z^2 = r^2

z^2 = r^2 - (x^2 + y^2)

z = sqrt(r^2 - l^2)

Relative Z

From shoulder point and elbow point, we can calculate relative Z from shoulder to elbow

Using a similar line of reasoning, we can deduce the relative Z coordinate of finger compared to Z of elbow

Setting Z coordinate of shoulder to be 0,

elbow.Z = 0 + relative Z from shoulder to elbow

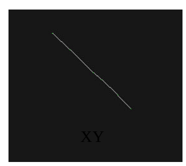
finger.Z = elbow.Z + relative Z from elbow to finger

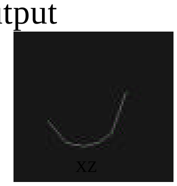
Movie

3-D finger tracking
of a semi-circle

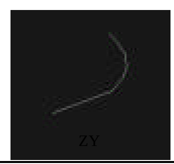
..\public_html\montage-semi.html

Graph output





Graphs of semi-circle movement, from varying viewpoints



Movie

3-D finger tracking of
circle motion

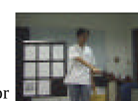
montage-circle.html

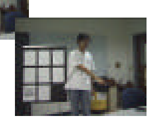
..\public_html\montage-circle.html

Tracker improvements

Problem:

Arm not identified properly when little or no motion in frame

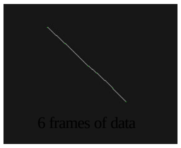


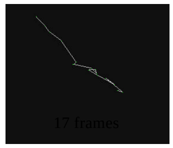


Approach:

Set arbitrary threshold for minimum amount of movement. If threshold not reached, use previously found arm centroid

Comparison

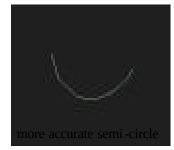




•Same data set, from semi-circle motion

•Extra data shows how local tracking errors can be quite significant

Comparison of accuracy



6 frames, → 17 frames,

raw, uneducated shoulder guess → adjusted shoulder position

Model of Error

- Previously did not use “confidence” recorded
- If we can estimate error, we can use estimate to adjust level of smoothing
- High Confidence means Low Error

Error and Confidence

- Estimated Confidence = Linear Combination of 4 Factors
- First Factor:
  Dot Product

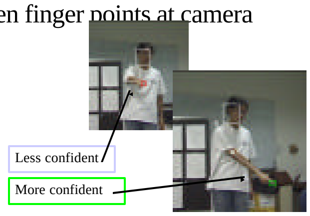
Higher dot product (DP) means Higher confidence

Confidence from 3D

- We know that certain 3D positions of the finger are harder to detect
- Worst case: when finger points at camera

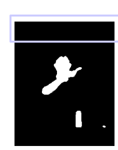
Generalization:

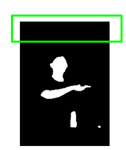
The length of the lower arm is proportional to our confidence in tracking



Next Factor: Occlusion

- When there is occlusion, we are less sure of the tracking
- Thus, the number of centroids is proportional to confidence



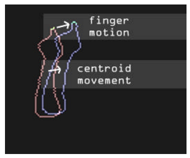


Movement Disparity

- We expect that the finger will move as the (centroid of the) arm moves

Note how both vectors (white arrows) are very similar

Less disparity means greater confidence



(Disparity = magnitude of the difference of both vectors)

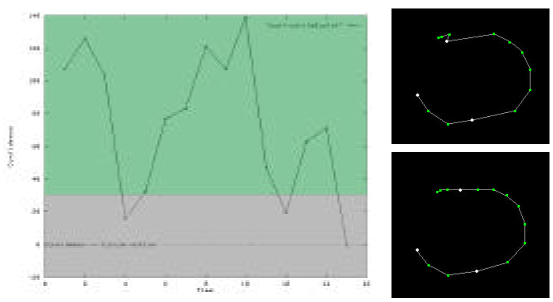
Linear Combination

- Confidence =
  C_1 * dot product +
  C_2 * length of lower arm +
  C_3 * number of centroids +
  C_4 * movement disparity

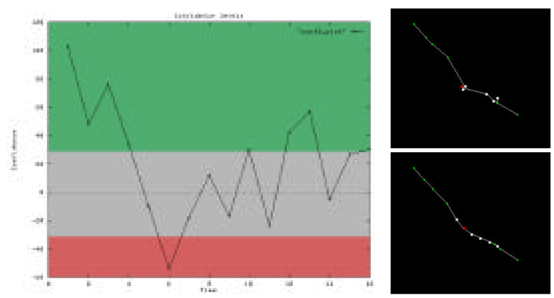
$C_1 = 8$, $C_2 = 1$, $C_3 = 6$, $C_4 = -2$

Constants chosen by hand to match error determined by (human) visual inspection

Graph of Confidence - Circle



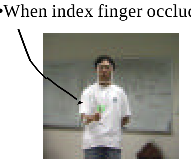
Confidence - Semicircle



Improving the User

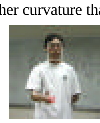
•Low-tech improvement: have user stick out thumb while gesturing

•When index finger occluded, thumb is not...



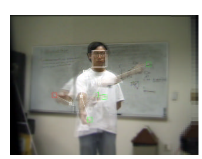
user points toward camera

…but index finger still has higher curvature than thumb:

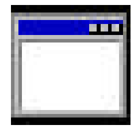


Saddle Point movie

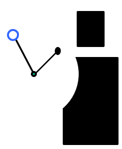
..\public_html\move-sad.html



..\public_html\move-spire.html



Fin



Action Recognition Using Temporal Templates

Jim Davis and Aaron Bobick

Main Points

- Compute a sequence of difference pictures from a sequence of images.
- Compute Motion Energy Images (MEI) and Motion History Images (MHI) from difference pictures.
- Compute Hu moments of MEI and MHI.
- Perform recognition using Hu moments.

MEI and MHI

Motion-Energy Images (MEI)

$$E_t(x, y, t) = \bigcup_{i=0}^{t-1} D(x, y, t-i)$$ **Difference Pictures**

Motion History Images (MHI)

$$H_t(x, y, t) = \begin{cases} t & if D(x, y, t) = 1 \\ \max(0, H_t(x, y, t-1) - 1) & otherwise \end{cases}$$

Moments

General Moments

$$m_{pq} = \int_{-}\int x^p y^q \boldsymbol{r}(x, y)dxdy$$

Central Moments

$$\boldsymbol{m}_{pq} = \int\int (x-\overline{x})^p (y-\overline{y})^q \boldsymbol{r}(x, y) \; d(x-\overline{x})d(y-\overline{y})$$

$$\overline{x} = \frac{m_{10}}{m_{00}}, \overline{y} = \frac{m_{01}}{m_{00}}$$

Moments

Hu Momens

$$\boldsymbol{u}_1 = \boldsymbol{m}_{20} + \boldsymbol{m}_{02}$$

$$\boldsymbol{u}2 = (\boldsymbol{m}_{20} - \boldsymbol{m}_{02})^2 + \boldsymbol{m}_{11}{}^2$$

$$\boldsymbol{u}_3 = (\boldsymbol{m}_{30} - 3\boldsymbol{m}_{12})^2 + (3\boldsymbol{m}_{12} - \boldsymbol{m}_{03})^2$$

$$\boldsymbol{u}_4$$

$$\vdots$$

Webpage

- http://vismod.www.media.mit.edu/vismod/demos/actions/mhi_generation.mov
- http://www.cs.ucf.edu/~ayers/research.html

Papers

- Claudette Cedras and Mubarak Shah, “Motion-Based Recognition: A survey”, Image and Vision Computing, March 1995.
- Jim Davis and Mubarak Shah, “Visual Gesture Recognition”, IEE Proc. Vis Image Signal Processing, October 1993.

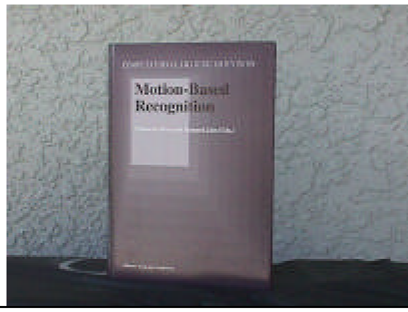
Papers

- Li Nan, Shawn Dettmer, and Mubarak Shah, “Visual Lipreading”, Workshop on Face and Gesture Recognition, Zurich, 1995.
- Doug Ayers and Mubarak Shah, “Recognizing Human Activities In an Office Environment”, Workshop on Applications of Computer Vision, October, 1998.

Book

- Mubarak Shah and Ramesh Jain, “**Motion-Based Recognition**”, Kluwer Academic Publishers, 1997 ISBN 0-7923-4618-1.

Book



Contents

- Mubarak Shah and Ramesh Jain, "Visual Recognition of Activities, Gestures, Facial Expressions and Speech: An Introduction and a Perspective"
- Human Activity Recognition
  - Y. Yacoob and L. Davis, "Estimating Image Motion Using Temporal Multi-Scale Models of Flow and Acceleration
  - A. Baumberg and D. Hogg, "Learning Deformable Models for Tracking the Human Body
  - S. Seitz and C. Dyer, "Cyclic Motion Analysis Using the Period Trace"

Contents (contd.)

  - R. Pollana and R. Nelson, "Temporal Texture and Activity Recognition"
  - A. Bobick and J. Davis, "Action Recognition Using Temporal Templates"
  - N. Goddard, "Human Activity Recognition"
  - K. Rohr, "Human Movement Analysis Based on Explicit Motion Models"

Contents (contd.)

- Gesture Recognition and Facial Expression Recognition
  - A. Bobick and A. Wilson, "State-Based Recognition of Gestures"
  - T. Starner and A. Pentland, "Real-Time American Sign Language Recognition from Video Using Hidden Markov Models"
  - M. Black , Y. Yacoob and S. Ju, "Recognizing Human Motion Using Parameterized Models of Optical Flow"

Contents (contd.)

  - I. Essa and A. Pentland, "Facial Expression Recognition Using Image Motion"
- Lipreading
  - C. Bregler and S. Omohumdro, "Learning Visual Models for Lipreading"
  - A. Goldschen, O. Garcia and E. Petajan, "Continuous Automatic Speech Recognition by Lipreading"
  - N. Li, S. Dettmer and M. Shah, "Visually Recognizing Speech Using Eigensequences"

Part IV

Video Phones and MPEG-4

MPEG-1 & MPEG -2 Artifacts

- Blockiness
  - poor motion estimation
  - seen during dissolves and fades
- Mosquito Noises
  - edges of objects (high frequency DCT terms)
- Dirty Window
  - streaks or noise remain stationary while objects move

MPEG-1 & MPEG -2 Artifacts

- Wavy Noise
  - seen during pans across crowds
  - coarsely quantized high frequency terms cause errors

Where MPEG-2 will fail?

- Motions which are not translation
  - zooms
  - rotations
  - non-rigid (smoke)
  - dissolves
- Others
  - shadows
  - scene cuts
  - changes in brightness

Video Compression At Low Bitrate

- The quality of block-based coding video (MPEG-1 & MPEG-2) at low bitrate, e.g., 10 kbps is very poor.
  - Decompressed images suffer from blockiness artifacts
  - Block matching does not account for rotation, scaling and shear

Model-Based Video Coding

Model-Based Compression

- Object-based
- Knowledge-based
- Semantic-based

Model-Based Compression

- Analysis
- Synthesis
- Coding

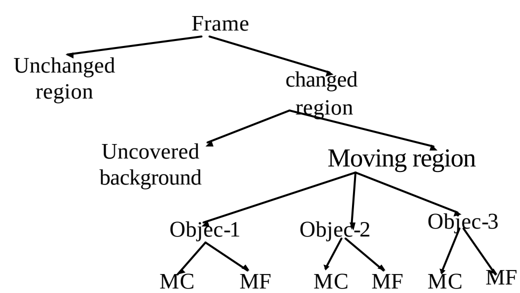
Video Compression

- MC/DCT
  - Source Model: translation motion only
  - Encoded Information: Motion vectors and color of blocks
- Object-Based
  - Source Model: moving unknown objects
    - translation only
    - affine
    - affine with triangular mesh
  - Encoded Information: Shape, motion, color of each moving object

Video Compression

- Knowledge-Based
  - Source Model: Moving known objects
  - Encoded Information: Shape, motion and color of known objects
- Semantic
  - Source Model: Facial Expressions
  - Encoded Information: Action units

Object-Based Coding



Contents

- Estimation using rigid+non-rigid motion model
- Making Faces (SIGGRAPH-98)
- Synthesizing Realistic Facial Expressions from Photographs (SIGGRAPH-98)
- MPEG-4

Model-Based Image Coding

- The transmitter and receiver both posses the same 3D face model and texture images.
- During the session, at the transmitter the facial motion parameters: global and local, are extracted.
- At the receiver the image is synthesized using estimated motion parameters.
- The difference between synthesized and actual image can be transmitted as residuals.

Face Model

- Candide model has 108 nodes, 184 polygons.
- Candide is a generic head and shoulder model. It needs to be conformed to a particular person's face.
- Cyberware scan gives head model consisting of 460,000 polygons.

Wireframe Model Fitting

- Fit orthographic projection of wireframe to the frontal view of speaker using Affine transformation.
- Locate four features in the image and the projection of model.
- Find parameters of Affine using least squares fit.
- Apply Affine to all vertices, and scale depth.

Synthesis

- Collapse initial wire frame onto the image to obtain a collection of triangles.
- Map observed texture in the first frame into respective triangles.
- Rotate and translate the initial wire frame according to global and local motion, and collapse onto the next frame.
- Map texture within each triangle from first frame to the next frame by interpolation.

Video Phones

Motion Estimation

Perspective Projection (optical flow)

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

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

Optical Flow Constraint Eq

$$f_x u + f_y v + f_t = 0$$

$$f_x\left(f\left(\frac{V_1}{Z}+\Omega_2\right)-\frac{V_3}{Z}x-\Omega_3 y-\frac{\Omega_1}{f}xy+\frac{\Omega_2}{f}x^2\right)+f_y$$
$$\left(f\left(\frac{V_2}{Z}-\Omega_1\right)+\Omega_3 x-\frac{V_3}{Z}y+\frac{\Omega_2}{f}xy-\frac{\Omega_1}{f}y^2\right)+f_t=0$$

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

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

$$\mathbf{Ax}=\mathbf{b}$$ Solve by Least Squares

$$\mathbf{x}=(V_1,V_2,V_3,\Omega_1,\Omega_2,\Omega_3)$$

$$A=\begin{bmatrix} & & \vdots & & & \\ (f_x\frac{f}{Z}) & (f_y\frac{f}{Z}) & (\frac{f}{Z}(f_x x-f_y y) & (-f_x\frac{xy}{f}+f_y\frac{y^2}{f}-f_y f) & (f_x f+f_x\frac{x^2}{f}+f_y\frac{xy}{f}) & (f_x y+f_y x) \\ & & \vdots & & & \end{bmatrix}$$

Comments

- This is a simpler (linear) problem than sfm because depth is assumed to be known.
- Since no optical flow is computed, this is called “direct method”.
- Only spatiotemporal derivatives are computed from the images.

Problem

- We have used 3D rigid motion, but face is not purely rigid!
- Facial expressions produce non-rigid motion.
- Use global rigid motion and non-rigid deformations.

3-D Rigid Motion

$$\begin{bmatrix}X'\\Y'\\Z'\end{bmatrix}=\begin{bmatrix}1&-\boldsymbol{a}&\boldsymbol{b}\\\boldsymbol{a}&1&-\boldsymbol{g}\\-\boldsymbol{b}&\boldsymbol{g}&1\end{bmatrix}\begin{bmatrix}X\\Y\\Z\end{bmatrix}+\begin{bmatrix}T_X\\T_Y\\T_Z\end{bmatrix}$$

$$\begin{bmatrix}X'\\Y'\\Z'\end{bmatrix}=\left(\begin{bmatrix}0&-\boldsymbol{a}&\boldsymbol{b}\\\boldsymbol{a}&0&-\boldsymbol{g}\\-\boldsymbol{b}&\boldsymbol{g}&0\end{bmatrix}+\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\right)\begin{bmatrix}X\\Y\\Z\end{bmatrix}+\begin{bmatrix}T_X\\T_Y\\T_Z\end{bmatrix}$$

3-D Rigid Motion

$$\begin{bmatrix} X'-X \\ Y'-Y \\ Z'-Z \end{bmatrix} = \begin{bmatrix} 0 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 0 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

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

3-D Rigid+Non-rigid Motion

$$\mathbf{X'} = \mathbf{RX} + \mathbf{T} + \mathbf{E}\Phi$$

Facial expressions

$$\mathbf{E} = \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1m} \\ e_{21} & e_{22} & \cdots & e_{2m} \\ e_{31} & e_{32} & \cdots & e_{3m} \end{bmatrix}$$

Action Units:
- -opening of a mouth
- -closing of eyes
- -raising of eyebrows

$$\Phi = (\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_m)^T$$

3-D Rigid+Non-rigid Motion

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} 1 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 1 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i \\ T_Y + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i \\ T_Z + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i \end{bmatrix}$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \left(\begin{bmatrix} 0 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 0 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i \\ T_Y + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i \\ T_Z + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i \end{bmatrix}$$

3-D Rigid+Non-rigid Motion

$$\begin{bmatrix} X'-X \\ Y'-Y \\ Z'-Z \end{bmatrix} = \begin{bmatrix} 0 & -\boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{a} & 0 & -\boldsymbol{g} \\ -\boldsymbol{b} & \boldsymbol{g} & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i \\ T_Y + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i \\ T_Z + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i \end{bmatrix}$$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i \\ T_Y + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i \\ T_Z + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i \end{bmatrix}$$

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

3-D Rigid+Non-rigid Motion

$$\dot{X} = -\Omega_3 Y + \Omega_2 Z + V_1 + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i$$

$$\dot{Y} = \Omega_3 X - \Omega_1 Z + V_2 + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i$$

$$\dot{Z} = -\Omega_2 X + \Omega_1 Z + V_3 + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i$$

Perspective Projection (arbitrary flow)

$$x = \frac{fX}{Z}$$

$$y = \frac{fY}{Z}$$

$$u = \dot{x} = \frac{fZ\dot{X} - fX\dot{Z}}{Z^2} = f\frac{\dot{X}}{Z} - x\frac{\dot{Z}}{Z}$$

$$v = \dot{y} = \frac{fZ\dot{Y} - fY\dot{Z}}{Z^2} = f\frac{\dot{Y}}{Z} - y\frac{\dot{Z}}{Z}$$

Perspective Projection (arbitrary flow)

$$u = \dot{x} = \frac{fZ\dot{X} - fX\dot{Z}}{Z^2} = f\frac{\dot{X}}{Z} - x\frac{\dot{Z}}{Z}$$

$$v = \dot{y} = \frac{fZ\dot{Y} - fY\dot{Z}}{Z^2} = f\frac{\dot{Y}}{Z} - y\frac{\dot{Z}}{Z}$$

$$u = f\left(\frac{V_1 + \sum_{i=1}^{m} e_{1i}\boldsymbol{f}_i}{Z} + \Omega_2\right) - \frac{V_3 + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i}{Z}x - \Omega_3 y - \frac{\Omega_1}{f}xy + \frac{\Omega_2}{f}x^2$$

$$v = f\left(\frac{V_2 + \sum_{i=1}^{m} e_{2i}\boldsymbol{f}_i}{Z} - \Omega_1\right) + \Omega_3 x - \frac{V_3 + \sum_{i=1}^{m} e_{3i}\boldsymbol{f}_i}{Z}y + \frac{\Omega_2}{f}xy - \frac{\Omega_1}{f}y^2$$

Optical Flow Constraint Eq

$$f_x u + f_y v + f_t = 0$$

$$\mathbf{Ax} = \mathbf{b}$$

Making Faces

Guenter et al
SIGGARPH'98

Making Faces

- System for capturing 3D geometry and color and shading (texture map).
- Six cameras capture 182 color dots on a face.
- 3D coordinates for each color dot are computed using pairs of images.
- Cyberware scanner is used to get dense wire frame model.

Making Faces

- Two models are related by a rigid transformation.
- Movement of each node in successive frames is computed by determining correspondence of nodes.

Synthesizing Realistic Facial Expressions from Photographs

Pighin et al

SIGGRAPH'98

Synthesizing Realistic Facial Expressions

- Select 13 feature points manually in face image corresponding to points in face model created with Alias.
- Estimate camera poses and deformed 3d model points.
- Use these deformed values to deform the remaining points on the mesh using interpolation.

Synthesizing Realistic Facial Expressions

- Introduce more points feature points (99) manually, and compute deformations as before by keeping the camera poses fixed.
- Use these deformed values to deform the remaining points on the mesh using interpolation as before.
- Extract texture.
- Create new expressions using morphing.

Show Video Clip.

MPEG-4

MPEG-4

- MPEG-4 will soon be international standard for true multimedia coding.
- MPEG-4 provides very low bitrate & error resilience for Internet and wireless.
- MPEG-4 can be carried in MPEG-2 systems layer.
- MPEG-4 text and graphics can be overlaid on MPEG-2 video for enhanced content: sports statistics and player trajectories.

MPEG-4

- Real audio and video objects
- Synthetic audio and video
- 2D and 3D graphics (based on VRML)

MPEG-4

- Traditional video coding is block-based.
- MPEG-4 provides object-based representation for better compression and functionalities.
- Objects are rendered after decoding object descriptions.
- Display of content layers can be selected at MPEG-4 terminal.

MPEG-4

- User can search or store objects for later use.
- Content does not depend on the display resolution.
- Network providers can re-purpose content for different networks and users.

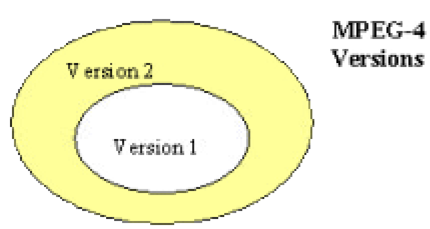
Scope & Features of MPEG-4

- Authors
  - reusability
  - flexibility
  - content owner rights
- Network providers
- End users

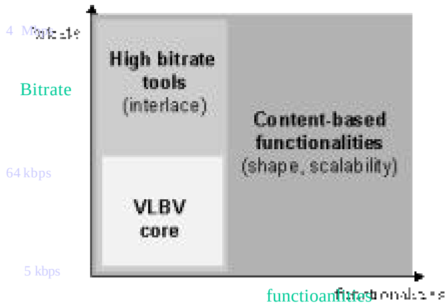
Media Objects

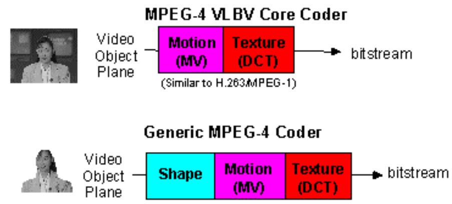
- Primitive Media Objects
- Compound Media Objects
- Examples
  - Still Images (e.g. fixed background)
  - Video objects (e.g., a talking person-without background)
  - Audio objects (e.g., the voice associated with that person)
  - etc

MPEG-4 Versions



MPEG-4

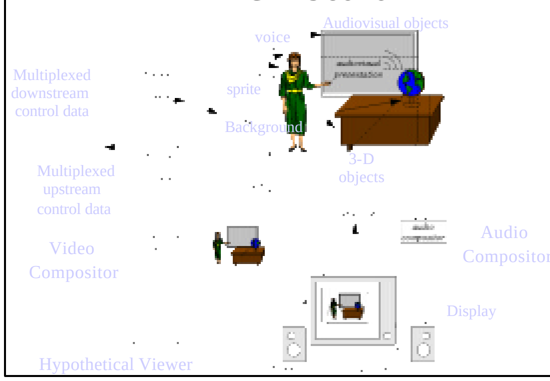




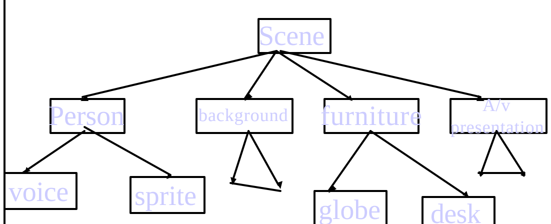
User Interactions

- Client Side
  - content manipulation done at client terminal
    - changing position of an object
    - making it visible or invisible
    - changing the font size of text
- Server Side
  - requires back channel

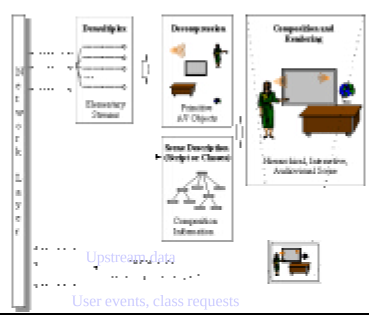
MPEG-4 Scene



Scene Graph



MPEG-4 Terminal

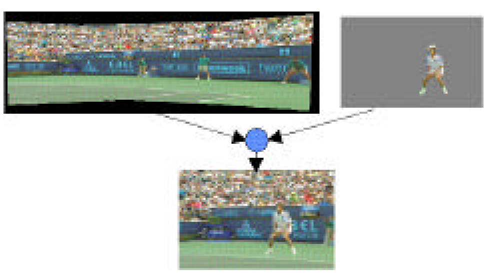


MPEG-4 Video and Image Coding Scheme

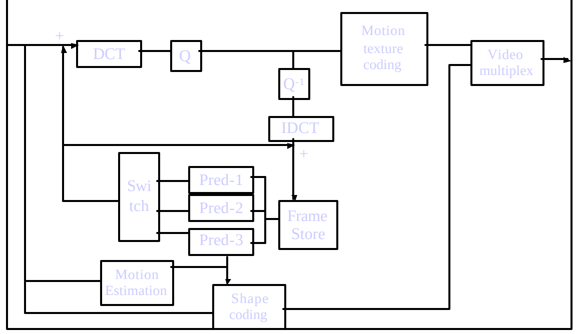
- Shape coding and motion compensation
- DCT-based texture coding
  - – standard 8x8 and shape adpated DCT
- Motion compensation
  - – local block based (8x8 or 16x16)
  - – global (affine) for sprites

Sprite Panorama

- First compute static "sprite" or "mosaic"
- Then transmit 8 or 6 global motion (camera) parameters for each frame to reconstruct the fame from the "sprite"
- Moving foreground is transmitted separately as an arbitrary-shape video object.



MPEG-4 Video Coder



Other Objects

- Text and graphics
- Talking synthetic head and associated text
- Synthetic sound

Face and Body Animtion

- Face animation is in MPEG-4 version 1.
- Body animation is in MPEG-4 version 2.
- Face animation parameters displace feature points from neutral position.
- Body animation parameters are joint angles.
- Face and body animation parameter sequences are compressed to low bit rate.
- Facial expressions: joy, sadness, anger, fear, disgust and surprise.

Neutral Face

- Face is gazing in the Z direction
- Face axes parallel to the world axes
- Pupil is 1/3 of iris in diameter
- Eyelids are tangent to the iris
- Upper and lower teeth are touching and mouth is closed
- Tongue is flat, and the tip of tongue is touching the boundary between upper and lower teeth

FAP Groups

Group	FAPS
Visemes & expressions	2
jaw, chin, inner lower-lip, corner lip, mid-lip	16
eyeballs, pupils, eyelids	12
eyebrow	8
cheeks	4
tongue	5
head rotation	3
outer lip position	10
nose	4
ears	4

Visemes and Expressions

- For each frame a weighted combination of two visemes and two facial expressions
- After FAPs are applied the decoder can interpret effect of visemes and expressions
- Definitions of visemes and expressions using FAPs can be downloaded

Phonemes and Visemes

- 56 phonemes
  - 37 consonants
  - 19 vowels/diphthongs
- 56 phonemes can be mapped to 35 visemes

56 Phonemes

Phone	Example	Phone	Example	Phone	Example	Phone	Example
aa	cot	ow	boat	g	gag	q	glottal stop
ac	bat	oy	boy	gcl	g-closure	r	red
ah	butt	oy	boy	hh	hay	s	sis
ao	about	uh	book	hv	Leheigh	sh	shoe
aw	bough	uw	boot	jh	judge	t	tot
ax	the	ux	beauty	k	kick	tcl	t-closure
axr	diner	b	bob	kcl	k-closur	th	thief
ay	bite	bcl	b-closure	l	led	v	very
eh	bet	ch	church	m	mom	w	wet
er	birrd	d	dad	n	non	y	yet
ey	bait	dcl	d-closure	ng	sing	z	zoo
ih	bit	dh	they	nx	flapped-n	zh	measure
ix	roses	dx	butter	p	pop	epi	epithetic closure
iy	beat	en	button	pcl	p-closur	h#	silence
		f	fief				

Visems

Viseme_select	phonemes	example
0	none	na
1	p, b, m	put, bed, mill
2	f, v	far, voice
3	T, D	think, that
4	t, d	tip, doll
5	k, g	call, gas
6	tS, dZ, S	chair, join, she
7	s, z	sir, zeal
8	n, l	lot, not
9	r	red
10	A:	car
11	e	bed
12	I	tip
13	O	top
14	U	book

Facial Expressions

- Joy
  - The eyebrows are relaxed. The mouth is open, and mouth corners pulled back toward ears.
- Sadness
  - The inner eyebrows are bent upward. The eyes are slightly closed. The mouth is relaxed.
- Anger
  - The inner eyebrows are pulled downward and together. The eyes are wide open. The lips are pressed against each other or opened to expose teeth.

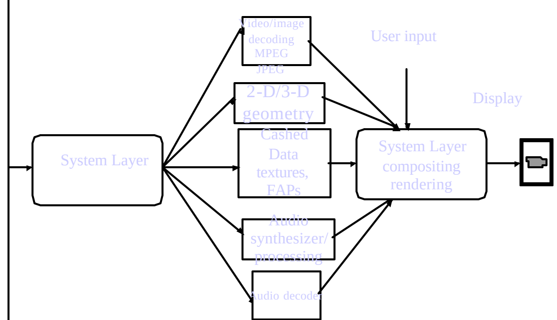
Facial Expressions

- Fear
  - The eyebrows are raised and pulled together. The inner eyebrows are bent upward. The eyes are tense and alert.
- Disgust
  - The eyebrows and eyelids are relaxed. The upper lip is raised and curled, often asymmetrically.
- Surprise
  - The eyebrows are raised. The upper eyelids are wide open, the lower relaxed. The jaw is open.

FAPs

- Speech recognition can use FAPs to increase recognition rate.
- FAPs can be used to animate face models by text to speech systems
- In HCI FAPs can be used to communicate speech, emotions, etc, in particular noisy environment.

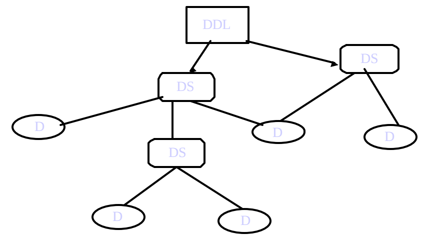
MPEG-4 Decoder

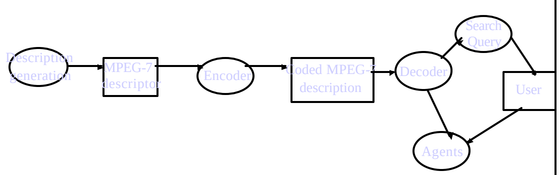


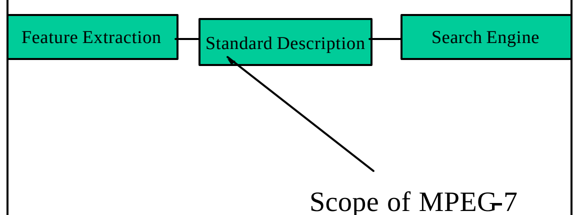
MPEG-7

- MPEG-7 will specify a standard set of descriptors that can be used to describe various types of multimedia information.
  - Descriptors
  - Description Scheme
  - Description Definition Language (DDL)

- MPEG-7 represents information about the content, not the content itself (“the bits about the bits”)







Different Types of Features

- Lower abstraction level
  - shape
  - size
  - texture
  - color
  - movement
  - position (where in the scene can the object be found)

Different Types of Features

- Audio
  - key
  - mood
  - tempo
  - tempo changes
  - position in sound space

Different Types of Features

- Highest Level Abstraction (semantic)
  - “This is a scene with a barking brown dog on the left and a blue ball that falls down on the right, with the sound of passing cars in the background.”

Other Type of Information

- The form
  - coding scheme (JPEG, MPEG-2)
  - size
- Conditions for accessing the material
- Links to other relevant material
- The context (e.g. Olympic 1996)

Search

- MPEG-7 data will be used to answer user queries.
- Music
  - Play a few notes on a keyboard and get in return a list of musical pieces containing required tune or images somehow matching the notes, e.g., in terms of emotions.

Search

- Graphics
  - Draw a few lines on a screen and get in return a set of images containing similar graphics, logos, ideograms,..
- Image
  - Define objects, including color patches or textures and get in return examples among which you select the interesting objects to compose your image.

Search

- Movement
  - On a given set of objects, describe movements and relations between objects and get in return a list of animations fulfilling the described temporal and spatial relations.
- Scenario
  - On a given content, describe actions and get a list of scenarios where similar actions happen.

Search

- Voice
  - Using an excerpt of Pavarotti’s voice, and getting a list of Pavarotti’s records, video clips, where Pavarotti is singing or video clips where Pavarotti is present

MPEG-4

- Go to http://www.cselt.it/mpeg

Conclusion

- Video Computing
  - Video Understanding
  - Video Tracking
  - Video Mosaics
  - Video Phones
  - Video Synthesis
  - Video Compression