CVPR-99 Tutorial on VIDEO COMPUTING

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

Course Contents

- Introduction
- Part I Measurement of Image Motion
- Part II Change Detection and Tracking
- Part III Video Understanding
- Part IV Video Phones and MPEG-4

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

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

PART I

Measurement of Motion

Contents

- Image Motion Models
- Optical Flow Methods
  - Horn & Schunck
  - Lucas and Kanade
  - Anandan et al
  - Szeliski
  - Mann & Picard
- Video Mosaics

3-D Rigid Motion

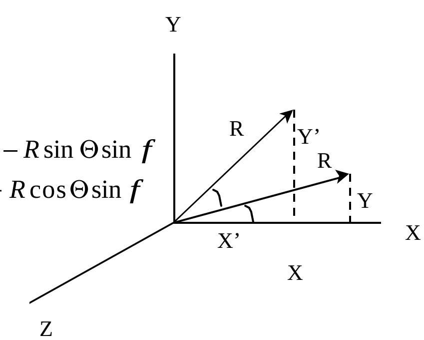
$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

Rotation matrix (9 unknowns)

Translation (3 unknowns)

Rotation

$$X = R \cos f$$
$$Y = R \sin f$$



$$X' = R\cos(\Theta + f) = R\cos\Theta\cos f - R\sin\Theta\sin f$$
$$Y' = R\sin(\Theta + f) = R\sin\Theta\cos f + R\cos\Theta\sin f$$

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

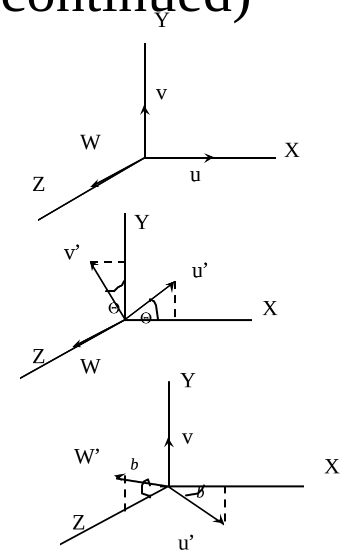
$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Rotation (continued)

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

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

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



Euler Angles

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

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

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

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.)

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

$$x' = x - ay + bZ + T_X$$
$$y' = ax + y - gZ + T_Y$$

Perspective Projection

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

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

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

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

Plane+Perspective(projective)

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

equation of a plane

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

3d rigid motion

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

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

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

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

Rigid

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

Affine

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

Projective

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

Biquadratic

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

Bilinear

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

Pseudo Perspective

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

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

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

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

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

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

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

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)

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

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

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

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

Plane+orthographic(Affine)

$$Z = a + bX + cY$$

$$u = V_1 + \Omega_2 Z - \Omega_3 y$$

$$v = V_2 + \Omega_3 x - \Omega_1 Z$$

$$u = b_1 + a_1 x + a_2 y$$

$$v = b_2 + a_3 x + a_4 y$$

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

$$b_1 = V_1 + a\Omega_2$$

$$a_1 = b\Omega_2$$

$$a_2 = c\Omega_2 - \Omega_3$$

$$b_2 = V_2 - a\Omega_1$$

$$a_3 = \Omega_3 - b\Omega_1$$

$$a_4 = -c\Omega_1$$

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

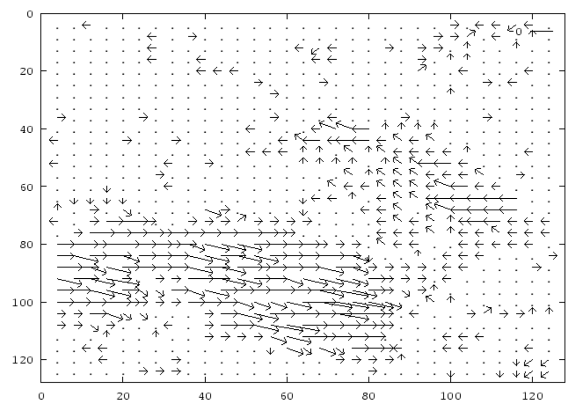
- Local Motion (Optical Flow)
- Global Motion (Frame Alignment)

Computing Optical Flow

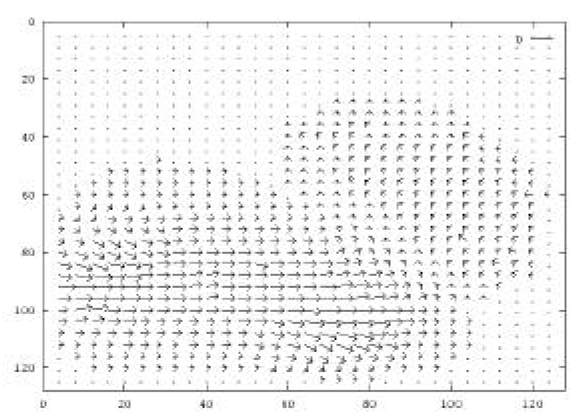
Image from Hamburg Taxi seq

Image from Hamburg Taxi seq

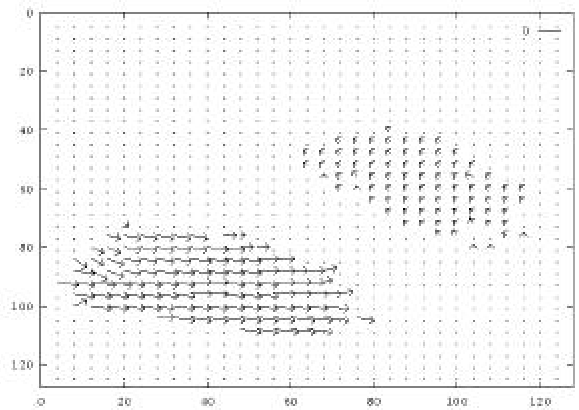
Fleet & Jepson optical flow



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

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

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

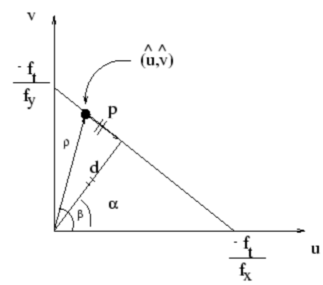
$$f_x u+f_y v+f_t=0$$ brightness constancy eq

Interpretation of optical flow eq

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

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

Equation of st.line



d=normal flow

p=parallel flow

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

Horn&Schunck (contd)

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

variational calculus

min

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

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

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

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

discrete version

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

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

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

$$D = l + f_x^2 + f_y^2$$

Algorithm-1

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

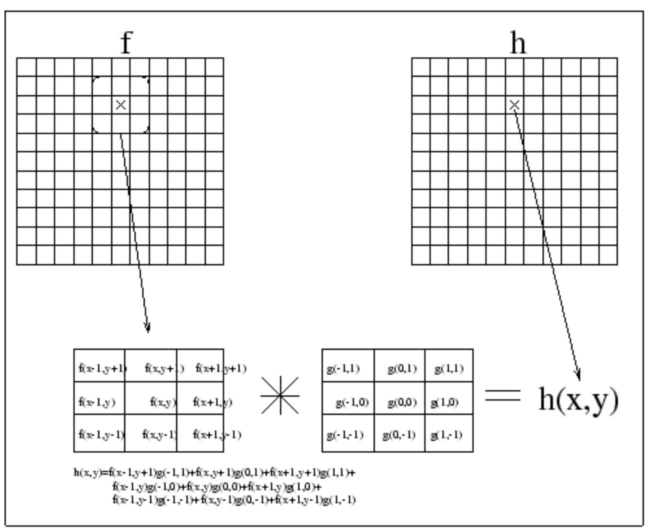
$$u^K = u_{av}^{k-1} - f_x \frac{P}{D}$$

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

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

$$D = l + f_x^2 + f_y^2$$

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

(a)

frame-1

-1	1
-1	1

frame-2

-1	1
-1	1

(b)

frame-1

1	1
-1	-1

frame-2

1	1
-1	-1

(c)

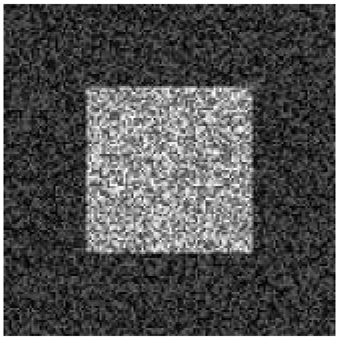
frame-1

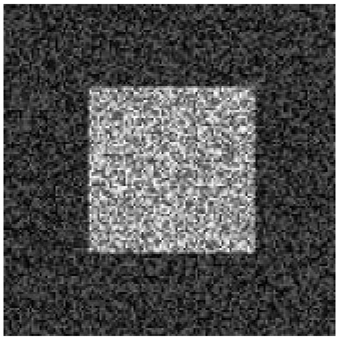
1	1
1	1

frame-2

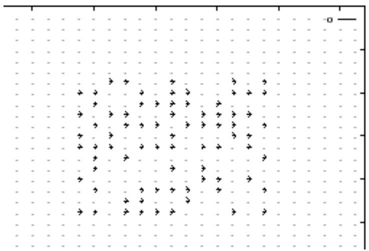
-1	-1
-1	-1

Synthetic Images

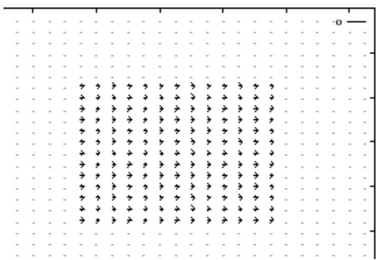




Results



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.

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

Important Issues

- What motion model?
- What function to be minimized?
- What minimization method?

Minimization Methods

- Least Squares fit
- Weighted Least Squares fit
- Newton-Raphson
- Gradient Descent
- Levenberg-Marquadet

Lucas & Kanade (Least Squares)

- Optical flow eq

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

- Consider 3 by 3 window

$$f_{x1} u + f_{y1} v = -f_{t1}$$
$$\vdots$$
$$f_{x9} u + f_{y9} v = -f_{t9}$$

$$\begin{bmatrix} f_{x1} & f_{y1} \\ \vdots & \vdots \\ & \\ f_{x9} & f_{y9} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -f_{t1} \\ \vdots \\ \\ -f_{t9} \end{bmatrix}$$

Lucas & Kanade

$$\mathbf{Au} = \mathbf{f_t}$$

$$\mathbf{A^T A u} = \mathbf{A^T f_t}$$

$$\mathbf{u} = (\mathbf{A^T A})^{-1} \mathbf{A^T f_t}$$

$$\min \sum_{i=-2}^{2} \sum_{j=-2}^{2} (f_{xi} u + f_{yi} v + f_{ti})^2$$

Lucas & Kanade

$$\min \sum_{i=-2}^{2} \sum_{j=-2}^{2} (f_{xi}u + f_{yi}v + f_{ti})^2$$

$$\Downarrow$$

$$\sum (f_{xi}u + f_{yi}v + f_{ti}) f_{xi} = 0$$

$$\sum (f_{xi}u + f_{yi}v + f_{ti}) f_{yi} = 0$$

Lucas & Kanade

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum f_{xi}^2 & \sum f_{xi} f_{yi} \\ \sum f_{xi} f_{yi} & \sum f_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum f_{xi} f_{ti} \\ -\sum f_{yi} f_{ti} \end{bmatrix}$$

Lucas & Kanade

$$\min \sum_{i=-2}^{2} \sum_{j=-2}^{2} w_i (f_{xi} u + f_{yi} v + f_{ti})^2$$

⇓

$$\mathbf{WAu} = \mathbf{Wf}_{\mathbf{t}}$$

$$\mathbf{A}^{\mathbf{T}}\mathbf{WAu} = \mathbf{A}^{\mathbf{T}}\mathbf{Wf}_{\mathbf{t}}$$

$$\mathbf{u} = (\mathbf{A}^{\mathbf{T}}\mathbf{WA})^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{Wf}_{\mathbf{t}}$$

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

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

$$E(da) = \sum_{x} (f_t + f_{\mathbf{x}}^T du)^2$$

$$E(da) = \sum_{x} (f_t + f_{\mathbf{x}}^T \mathbf{X} da \;)^2$$

Anandan

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

Basic Components

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

Szeliski (Levenberg-Marquadet)

$$E = \sum [f(x', y') - f(x, y)]^2 = \sum e^2$$

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

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

Szeliski (Levenberg-Marquadet)

Motion Vector:

$$\mathbf{m} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & b_1 & b_2 & c_1 & c_2 \end{bmatrix}^T$$

Szeliski (Levenberg-Marquadet)

Hessian $a_{kl} = \sum \frac{\partial e}{\partial m_k} \frac{\partial e}{\partial m_l}$

$b_k = -\sum e \frac{\partial e}{\partial m_k}$

gradient

$$\Delta m = (A + lI)^{-1} b$$

Szeliski (Levenberg-Marquadet)

- For each pixel I at (x_i, y_i)
  - Compute (x', y') using projective transform.
  - Compute $e = f(x', y') - f(x, y)$
  - Compute $\frac{\partial e}{\partial m_k} = \frac{\partial f}{\partial x'} \frac{\partial x'}{\partial m_k} + \frac{\partial f}{\partial y'} \frac{\partial y'}{\partial m_k}$

Szeliski (Levenberg-Marquadet)

-Compute A **and** b

-Solve system

$$(A - lI)\Delta m = b$$

-Update

$$m^{t+1} = m^t + \Delta m$$

Szeliski (Levenberg-Marquadet)

• check if error has decreased, if not increase l and compute a new Δm

• Continue iteration until error is below threshold.

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)

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

⇓ **minimize**

Projective Flow (weighted)

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

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

$$f^t = [f_x x, f_x y, f_x, f_y x, f_y y, f_y, x f_t - x^2 f_x - xy f_y, y f_t - xy f_x - y^2 f_y]$$

Projective Flow (unweighted)

Bilnear

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

⇓ 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

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

⇓ **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)

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

⇓ **Minimize**

Bilinear and Pseudo-Perspective

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

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

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

$$c_1 = x^2 f_x + xy f_x$$

$$c_2 = xy f_x + y^2 f_y$$

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 (x'_k, y'_k)
  - 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

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

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

- 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.

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

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

Mixture of Gaussians

Grimson

Algorithm

- Learn background model by watching 30 second video
- Detect moving object by measuring deviations from background model, and applying connected component to foreground pixels.
- Predict position of a region in the next frame using Kalman filter
- Update background and blob statistics

Summary

- Each pixel is an independent statistical process, which may be combination of several processes.
  – Swaying branches of tree result in a bimodal behavior of pixel intensity.
- The intensity is fit with a mixture of K Gaussians.

$$\Pr(X_t) = \sum_{j=1}^{K} \frac{w_j}{(2p)^{\frac{m}{2}} \, |\Sigma_j|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - m_j)^T \Sigma_j^{-1} (X_t - m_j)}$$

Mixture of Gaussians

- The K distributions are stored in descending order of the term $\frac{w_j}{s_j}$
- Out of “k” distributions, the first B are selected

$$B = \arg\min_b \left[\frac{\sum_{j=1}^{b} w_j}{\sum_{j=1}^{K} w_j} > T \right]$$

Learning Background Model

- Every new pixel is checked against all existing distributions. The match is the first distribution such that the pixel value lies within 2 standard deviations of mean.

- If no match, introduce new distribution.

Updating

- The mean and s.d. of unmatched distributions remain unchanged. For the matched distributions they are updated as:

$$m_{j,t} = (1 - r)m_{j,t-1} + rX_t$$

$$s_{j,t} = (1 - r)s^{2}_{j,t-1} + r(X_t - m_{j,t})^T (X_t - m_{j,t})$$

- The weights are adjusted:

$$w_{j,t} = (1 - a)w_{j,t-1} + a(M_{j,t})$$

Segmenting Background

- Any pixel that is more than 2 sd from all the distributions is marked as a part of foreground-moving object.
- Such pixels are then clustered into connected components.

Kanade

Summary

- Very similar to k-Gaussian with following differences:
  - uses only single Gaussian
  - uses gray level images, the mean and variance are scalar values

Algorithm

- Learn background model by watching 30 second video
- Detect moving object by measuring deviations from background model, and applying connected component to foreground pixels.
- Update background and region statistics

Detection

- During detection if intensity value is more than two sigma away from the background it is considered foreground:
  - keep original mean and variance
  - track the object with new mean and variance
  - if new mean and variance persists for sometime, then substitute the new mean and variance as the background model
  - Object is no longer visible, it is incorporated as part of background

W4 (Who, When, Where, What)

Davis

W4

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

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

- 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

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

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

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

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

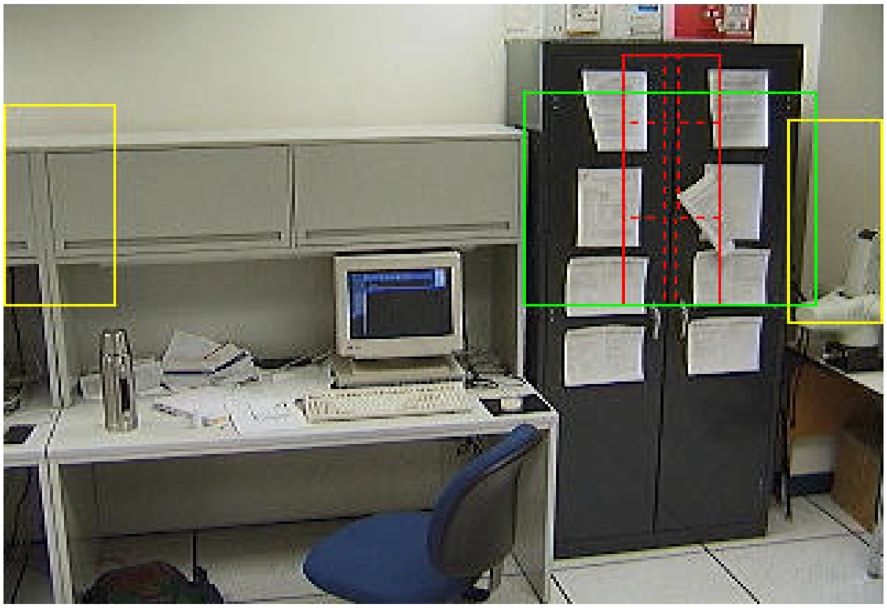
Possible Actions

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

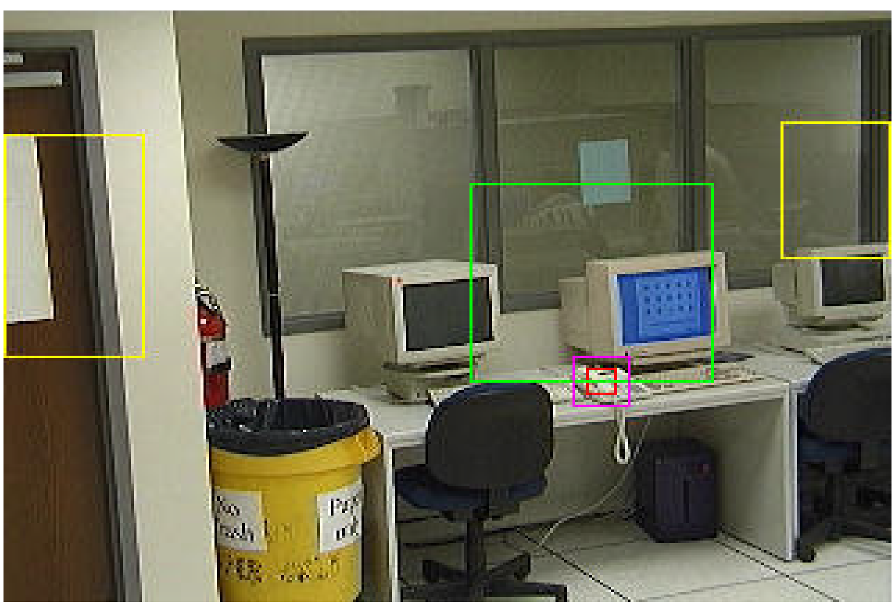
Prior Knowledge

- 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

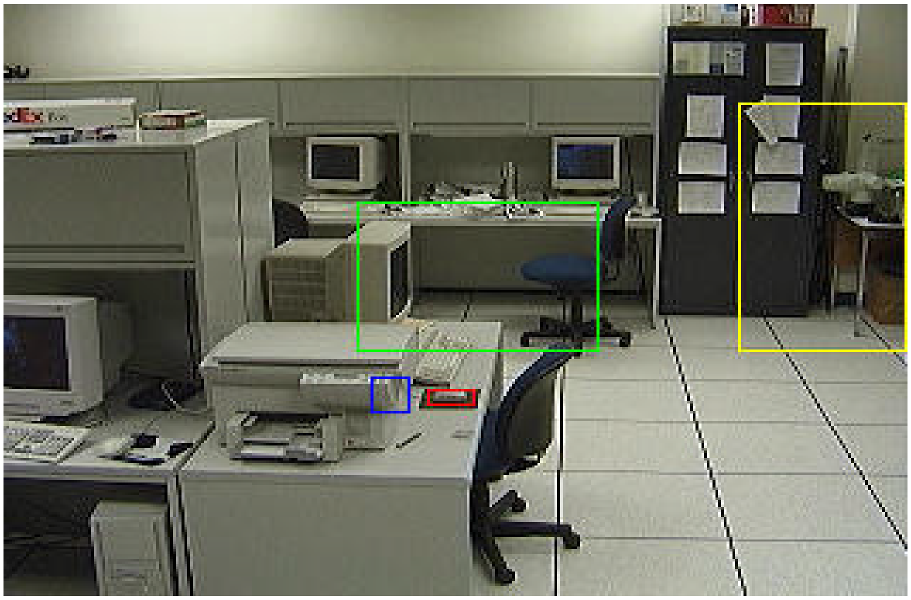
Layout of Scene 1



Layout of Scene 2



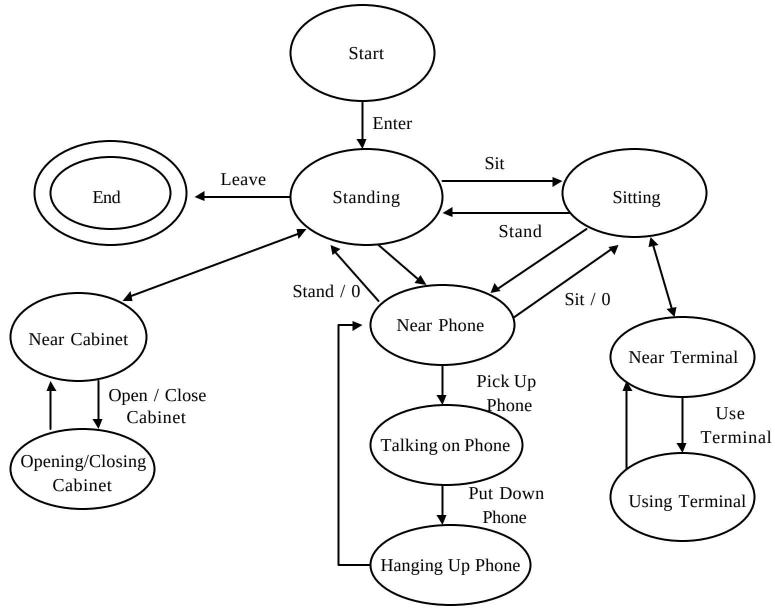
Layout of Scene 4



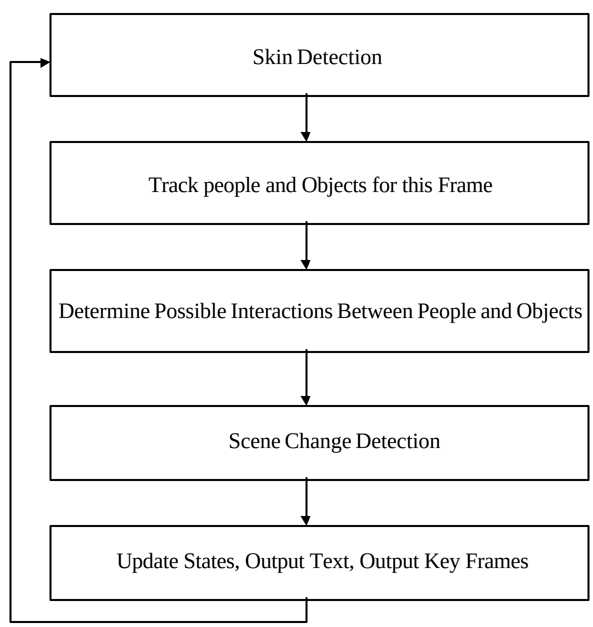
Major Components

- 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

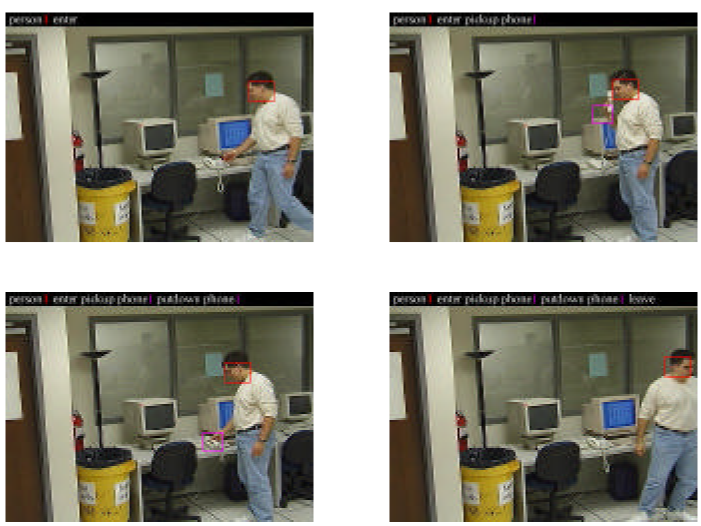
- <u>"Enter" key frames</u>: as the person leaves the entrance/exit area
- <u>"Leave" key frames</u>: as the person enters the entrance/exit area
- <u>"Standing/Sitting" key frames</u>: after the tracking box has stopped moving up or down respectively
- <u>"Open/Close" key frames</u>: when the % of changed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

Results

Key Frames Sequence 1 (350 frames), Part 1

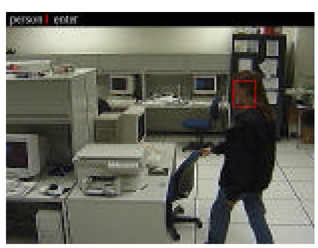
Key Frames Sequence 1 (350 frames), Part 2

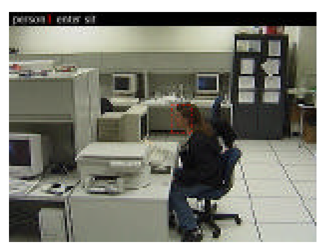
Key Frames Sequence 2 (200 frames)

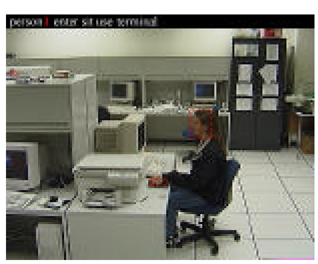


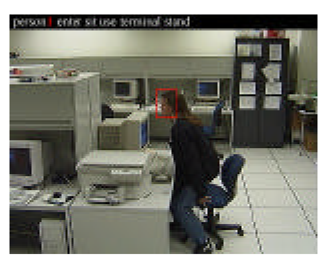
Key Frames Sequence 3 (200 frames)

Key Frames Sequence 4 (399 frames), Part 1

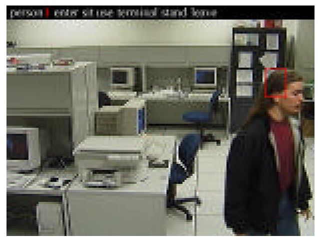








Key Frames Sequence 4 (399 frames), Part 2



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

Model-Based Human Activity Recognition

Approach

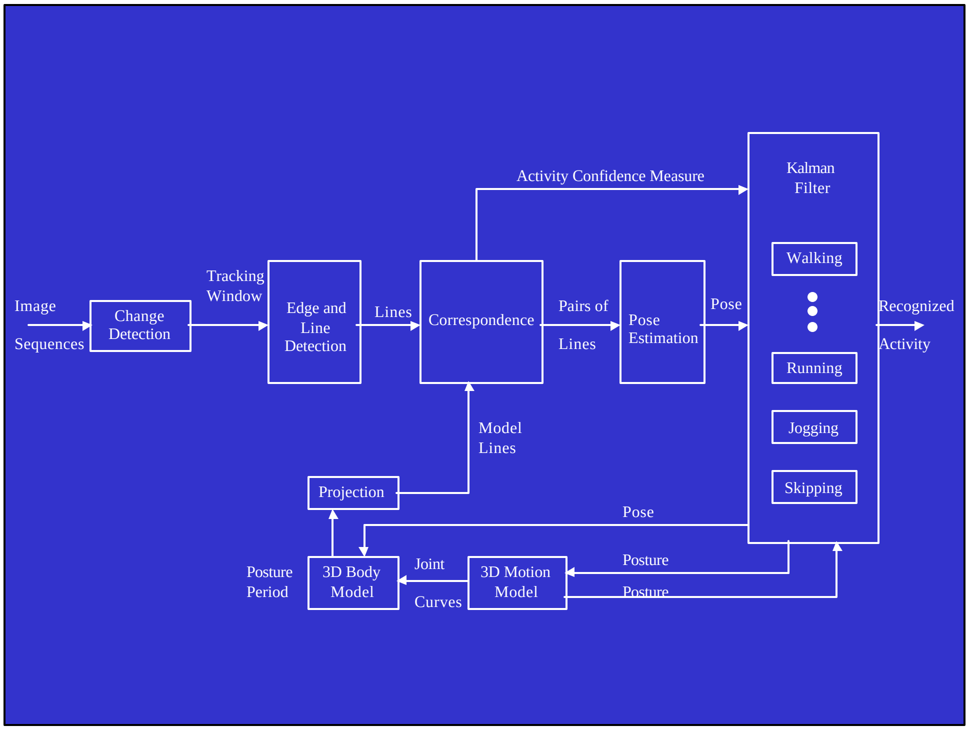
- Activity Detection
- Activity Recognition
- Activity Modeling

3-D Body Model

- 14 Cylinders
  - Head (1), Torso (1), Upperarms (2), Forearms(2), Hands (2), Thighs (2), Calves (2) and Feet.
- 2 Parameters
  - Length
  - Circular Crossections
- The center of Torso is the origin of 3D coordinate system.

Modeling Activities

- Joint Curves at Shoulder, Elbow, Hip and Knee.
- Walking Curves from Rohr.
- Running, Skipping and Jogging Curves from Goddard.
- One Cycle of a joint curve is polynomial curve.



Change Detection

- Motion arises from a relative displacement between the sensor and the scene.
- The accumulative differences method is used.
- Algorithm for change detection:
  - Divide the image into 5X5 blocks
  - if DP>T, set block “k” to all ones

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

Change Detection

- Connected component analysis on the changed pixels.
- A merging phase to combine overlapping regions.
- Area thresholding to reject small regions.

Line Correspondence

- We match 2D lines from the model projection to the scene lines.
- The line is represented by a vector [a,b,y,l].
- Representation:
  - A line from the model, by vector m0
  - A line from the scene by the vector ri
- For an ideal match between a model line and scene line ri-m=0
  - A Mahalanobis Distance is computed

Pose Estimation

- For any 3D point there is a transformation that will give the 2D image point:

$$m = \tilde{P}M$$

- For every 3D point and its corresponding 2D point we have:

$$\mathbf{q}_1^T M_i - u_i \mathbf{q}_3^T M_i + q_{14} - u_i q_{34} = 0$$
$$\mathbf{q}_2^T M_i - v_i \mathbf{q}_3^T M_i + q_{24} - v_i q_{34} = 0$$

Pose Estimation

- For N points, we have 2N equations:

$$\mathbf{Aq} = \mathbf{0}$$

$$\mathbf{A} = \begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_i X_i & -u_i Y_i & -u_i Z_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_i X_i & -v_i Y_i & -v_i Z & v_i \end{bmatrix}$$
$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_1^T & q_{14} & \mathbf{q}_2^T & q_{24} & \mathbf{q}_3^T & q_{34} \end{bmatrix}$$

- Solve to estimate P.

Activity Recognition Using Kalman Filter

- **1 State Vector**

$$(X, Y, Z, R_x, R_y, R_z, \dot{X}, \dot{Y}, \dot{Z}, \dot{R}_x, \dot{R}_y, \dot{R}_z)$$

- 2 **Start a Kalman Filter for each possible activity.**

- **3 Continuous processing per filter**

Activity Recognition Using Kalman Filter

-i **Predict next sate, and compute covariance matrix.**

$$\mathbf{a}_i' = f\mathbf{a}_i'^{t-1}$$

$$\mathbf{P}_\mathbf{i}' = f\mathbf{P}_i^{t-1} f^T + \mathbf{Q}$$

-ii **Use predicted state to generate 3D model based upon filter's corresponding activity.**

Activity Recognition Using Kalman Filter

- iii **Do projection, line correspondence and pose estimation to obtain measurement vector and error measure**

$$\mathbf{a}_i^t = \mathbf{a}_i' + \mathbf{K}_i^t(\mathbf{r}_t - \mathbf{H}\mathbf{a}_i')$$

$$\mathbf{K}_i^t = \mathbf{P}_i'\mathbf{H}^T(\mathbf{H}\mathbf{P}_i'\mathbf{H}^T + \mathbf{R}_t)^{-1}$$

$$\mathbf{P}_i^t = \mathbf{P}_i' - (\mathbf{K}_i^t\mathbf{H}\mathbf{P}_i')$$

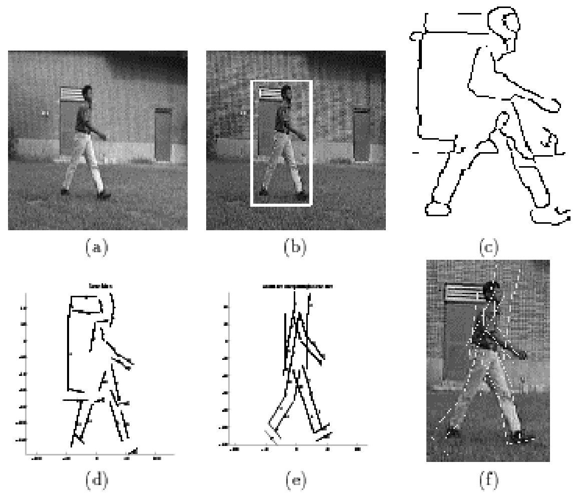
- iv **If error is below threshold, then update the state vector.**

Activity Recognition Using Kalman Filter

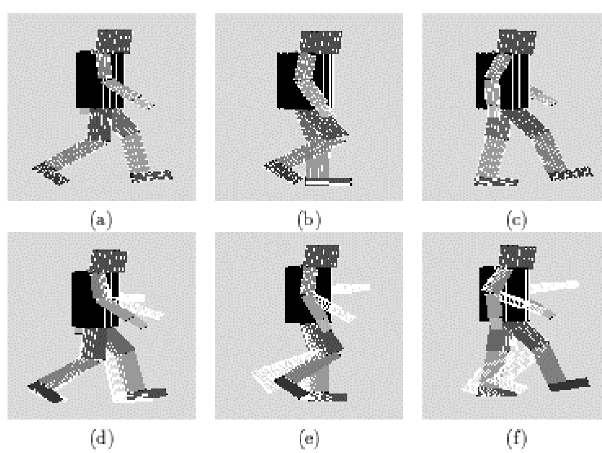
- v **If error is above threshold, the filter's corresponding activity is incorrect, and filter is stopped.**

- **Stop processing when all but one filter have ceased. Remaining filter gives the recognized activity.**

Results



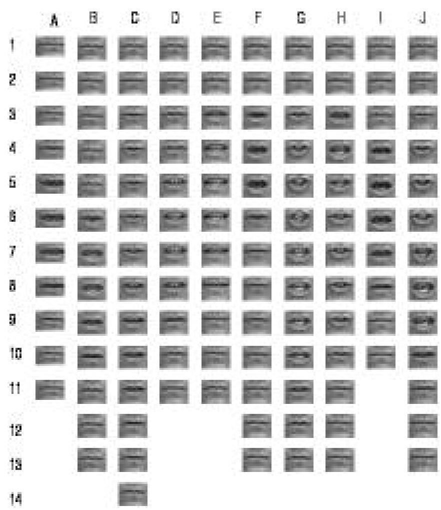
Results



SHOW VIDEO CLIP

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 w , collect vectors into matrix T

$$T_w = \left[u^1, u^2, \ldots u^K\right]$$

- Create sample matrix A:

$$A = \left[T_1, T_2, \ldots T_e\right]$$

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

The Most Expressive Features

- f is an orthonormal basis of the sample matrix.

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

$$u = \sum_{n=1}^{Q} a_{nf_n} = fa$$

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

$$a_n = u^T f_n$$

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:

$$a_i^x = u_x^T . f_i$$

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

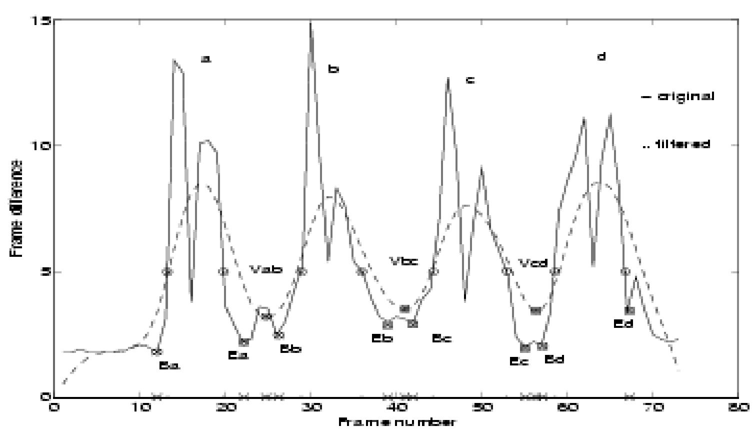
- Determine best match by

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.

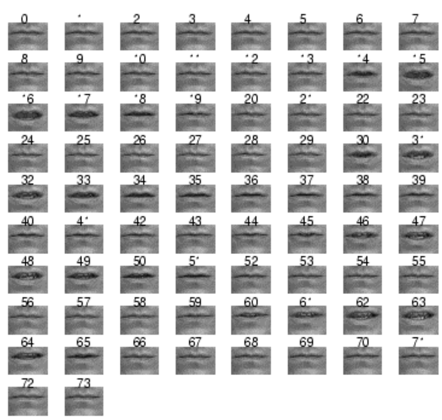
- Locate beginning and ending frames.

A 12-22

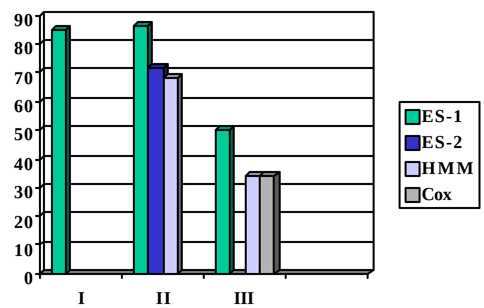
B 26-39

C 42-55

D 57-67



Results

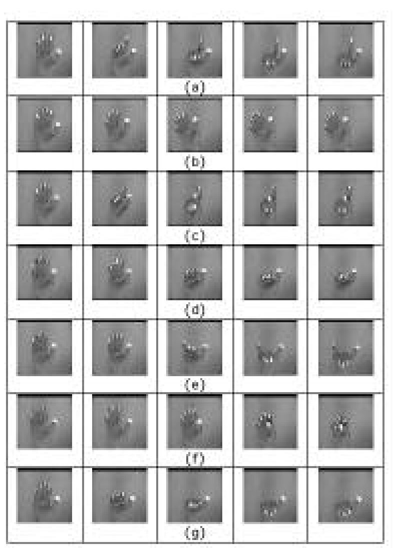


I: “A” to “J” one speaker, 10 training seqs
II. “A” to “M”, one speaker, 10 training seqs
III. “A” to “Z”, ten speakers, two training seqs/letter/person

Show Video Clip

Hand Gesture Recognition

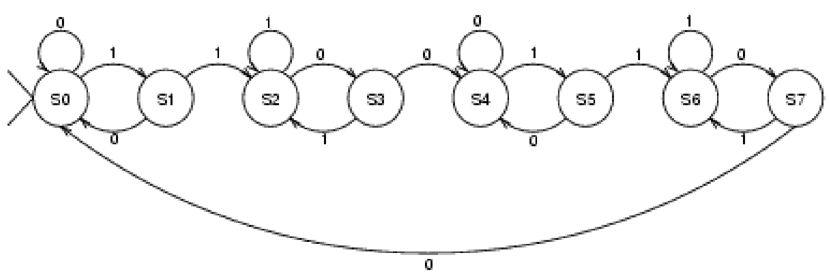
Seven Gestures



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.

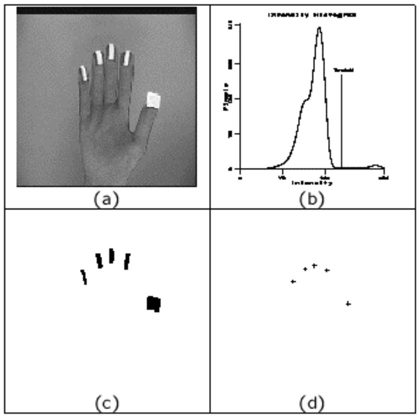
Finite State Machine



Main Steps

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

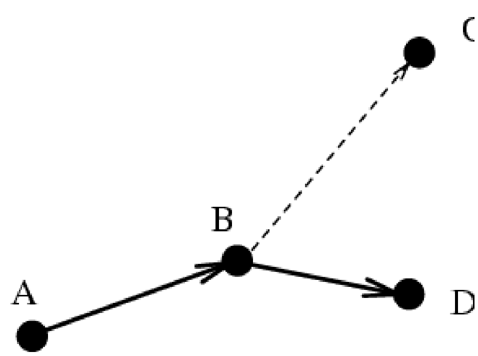
Detecting Fingertips



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

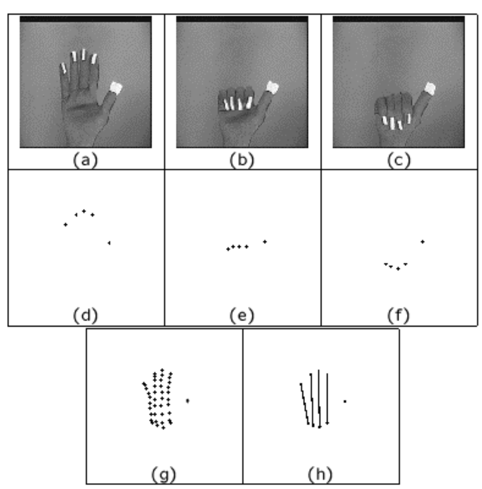


Proximal Uniformity Constraint

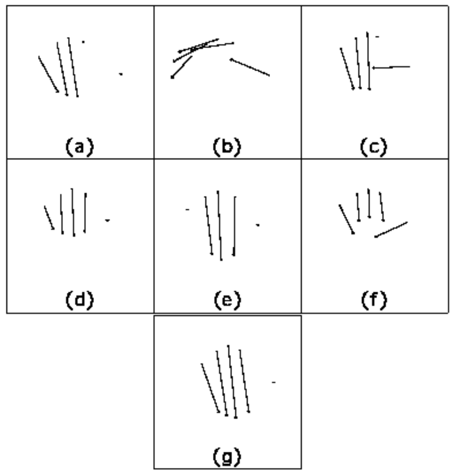
Establish correspondence by minimizing:

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

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.

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

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

Central Moments

$$m_{pq} = \int\int (x-\bar{x})^p (y-\bar{y})^q \, r(x,y) \; d(x-\bar{x}) d(y-\bar{y})$$

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

Moments

Hu Momens

$$u_1 = m_{20} + m_{02}$$

$$u2 = (m_{20} - m_{02})^2 + m_{11}{}^2$$

$$u_3 = (m_{30} - 3m_{12})^2 + (3m_{12} - m_{03})^2$$

$$u_4$$

$$\vdots$$

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.

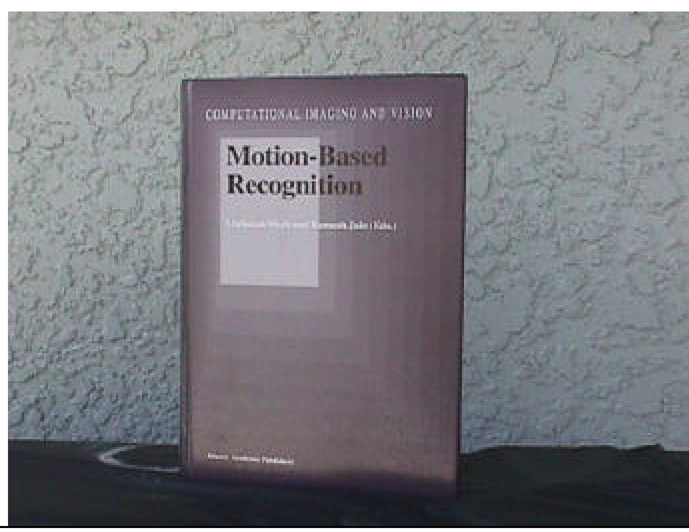
Papers

- 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

Video Compression

- Video compression is important.
- MPEG compression is domain independent, uses 2D block motion.
- Compression ratio in MPEG is limited.
- Model-Based compression can be used to achieve compression of up to 250kb/s.

Model-Based Compression

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

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.
- Another face model was created by sticking 182 color dots on the face, and capturing dots by six cameras.

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

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

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

$$\Big(f\big(\frac{V_2}{Z}-\Omega_1\big)+\Omega_3 x-\frac{V_3}{Z}y+\frac{\Omega_2}{f}xy-\frac{\Omega_1}{f}y^2\Big)+f_t=0$$

$$\big(f_x\frac{f}{Z}\big)V_1+\big(f_y\frac{f}{Z}\big)V_2+\big(\frac{f}{Z}(f_x x-f_y y)\big)V_3+$$

$$\big(-f_x\frac{xy}{f}+f_y\frac{y^2}{f}-f_y f\big)\Omega_1+\big(f_x f+f_x\frac{x^2}{f}+f_y\frac{xy}{f}\big)\Omega_2+$$

$$(f_x y+f_y x)\Omega_3=-f_t$$

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

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

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

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

3-D Rigid Motion

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

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

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

3-D Rigid+Non-rigid Motion

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

Facial expressions

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

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

$$\Phi = (f_1, f_2, \dots, f_m)^T$$

3-D Rigid+Non-rigid Motion

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

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

3-D Rigid+Non-rigid Motion

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

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

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

3-D Rigid+Non-rigid Motion

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

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

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

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)

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

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

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

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

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

- Extract texture.
- Create new expressions using morphing.

3D Rigid Transformation

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

Camera coordinates

Wireframe coordinates

$$x_i'^k = f^k \frac{X_i'^k}{Z_i'^k}, \quad y_i'^k = f^k \frac{Y_i'^k}{Z_i'^k}$$ perspective

3D Rigid Transformation

$$x_i'^k = f^k \frac{X_i'^k}{Z_i'^k}, \quad y_i'^k = f^k \frac{Y_i'^k}{Z_i'^k}$$

$$x_i'^k = f_k \frac{r_{11}^k X_i + r_{12}^k Y_i + r_{13}^k Z_i + T_X^k}{r_{31}^k X_i + r_{32}^k Y_i + r_{33}^k Z_i + T_Z^k}$$

$$y_i'^k = f_k \frac{r_{21}^k X_i + r_{22}^k Y_i + r_{23}^k Z_i + T_Y^k}{r_{31}^k X_i + r_{32}^k Y_i + r_{33}^k Z_i + T_Z^k}$$

Model Fitting

$$x_i'^k = f_k \frac{\mathbf{r}_x^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_X^k}{\mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Z^k}$$

$$y_i'^k = f_k \frac{\mathbf{r}_y^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Y^k}{\mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Z^k}$$

Model Fitting

$$x_i'^k = f_k \frac{\mathbf{r}_x^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_X^k}{\mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Z^k}$$

$$y_i'^k = f_k \frac{\mathbf{r}_y^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Y^k}{\mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Z^k}$$

$$x_i'^k = s_k \frac{\mathbf{r}_x^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_X^k}{1 + h^k \mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}}}$$

$$y_i'^k = s_k \frac{\mathbf{r}_y^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Y^k}{1 + h^k \mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}}}$$

Model Fitting

$$x_i'^{k} = s_k \frac{\mathbf{r}_x^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_X^k}{1 + h^k \mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}}}$$

$$y_i'^{k} = s_k \frac{\mathbf{r}_y^{\mathbf{k}} \mathbf{p}_{\mathbf{i}} + T_Y^k}{1 + h^k \mathbf{r}_z^{\mathbf{k}} \mathbf{p}_{\mathbf{i}}}$$

$$w_i^k (x_i'^k + x_i'^k h^k (\mathbf{r}_{\mathbf{z}}^{\mathbf{k}} . \mathbf{p}_{\mathbf{i}}) - s^k (\mathbf{r}_X^{\mathbf{k}} . \mathbf{p}_{\mathbf{i}} + T_X^k) = 0$$

$$w_i^k (y_i'^k + y_i'^k h^k (\mathbf{r}_{\mathbf{z}}^{\mathbf{k}} . \mathbf{p}_{\mathbf{i}}) - s^k (\mathbf{r}_Y^{\mathbf{k}} . \mathbf{p}_{\mathbf{i}} + T_Y^k) = 0$$

Model Fitting

- Solve for unknowns in five steps:

$$s^k ; \mathbf{p}_i ; \mathbf{R}^k ; T_X^k , T_Y^k ; h^k$$

- Use linear least squares fit.

- When solving for an unknown, assume other parameters are known.

Interpolation

- Use initial set of coordinates for the feature points (13 points), to deform the remaining vertices using interpolation.

Interpolation

$$f(\mathbf{p}) = \sum_i c_i f(\| \mathbf{p} - \mathbf{p}_i \|) + \mathbf{M}\mathbf{p} + \mathbf{t}$$

$$u_i = \mathbf{p}_i - \mathbf{p}_\mathbf{i}^\mathbf{0}, u_i = f(\mathbf{p}_i)$$

$$\sum_i c_i = 0, \sum_i c_i \mathbf{p}_i = 0$$

$$f(r) = e^{\frac{-r}{64}}$$

Texture Extraction

$$T(\mathbf{p}) = \frac{\sum_k m^k(\mathbf{p}) I^k(x^k, y^k)}{\sum_k m^k(\mathbf{p})}$$

I^k is k-th image

m^k is weight

Weights

- Self-occlusion
- Smoothness
- Positional certainty
- View similarity

Texture Extraction

•Visibility map $F^k(u,v)$ is set to 1 if the corresponding point p is visible in k-th image, and zero otherwise.

• Positional certainty, $P^k(\mathbf{p})$ is define as a dot product of surface normal at p and the k-th direction of projection.

Texture Extraction

• View-independent texture mapping:

$$m^k(u,v) = F^k(u,v)P^k(\mathbf{p})$$

• View-dependent texture mapping:

$$m^k(u,v) = F^k(x^k, y^k)P^k(\mathbf{p})V^k(d)$$

$$V^k(\mathbf{d}) = \mathbf{d}.\mathbf{d}^k - \mathbf{d}^l.\mathbf{d}^{l+1}$$

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.

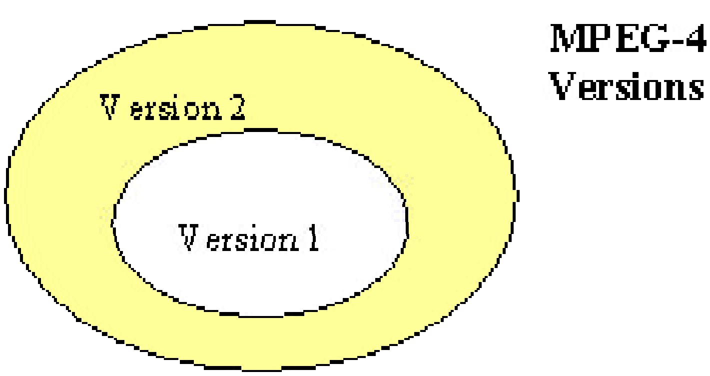
Scope & Features of MPEG-4

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

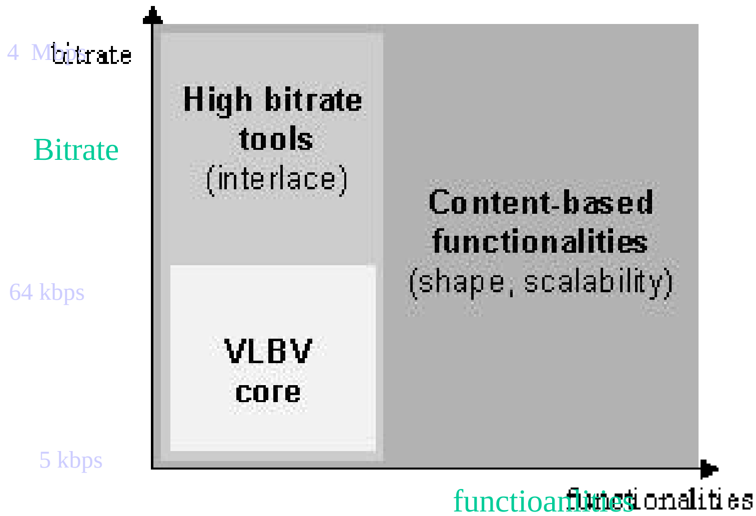
Media Objects

- 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



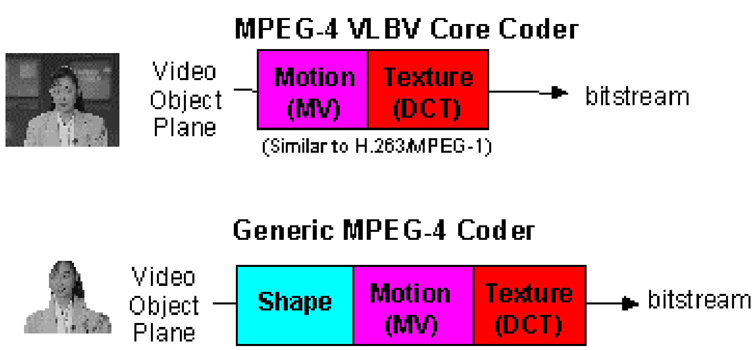
MPEG-4



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

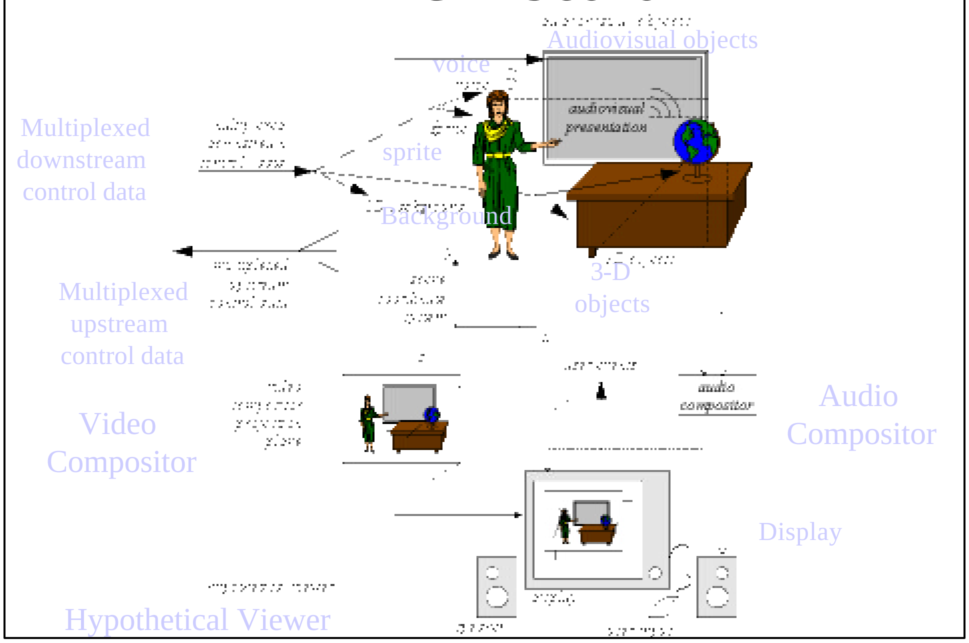
- Efficient representation of visual objects of arbitrary shape to support content-based functionalities
- Supports most functionalities of MPEG-1 and MPEG-2
  - rectangular sized images
  - several input formats
  - frame rates
  - bit rates
  - spatial, temporal and quality scalability



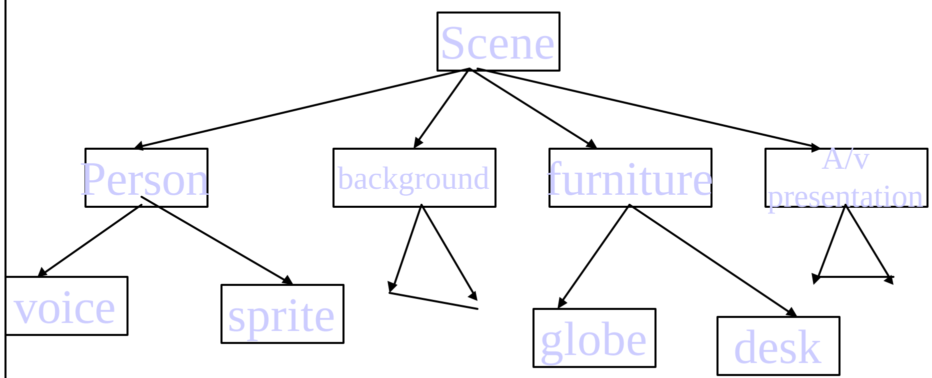
Object Composition

- Objects are organized in a scene graph.
- VRLM based binary format BIF is used to specify scene graph.
- 2-D and 3-D objects, transforms and properties are specified.
- MPEG-4 allows objects to be transmitted once, and displayed repeatedly in the scene after transformations.

MPEG-4 Scene



Scene Graph



Standardized Ways

- To represent “media object”
  - visual or audiovisual
  - synthetic or natural
- To multiplex and synchronize the data associated with media objects for transportation over the network
- Interact with audiovisual scene generated at the receiver’s end.

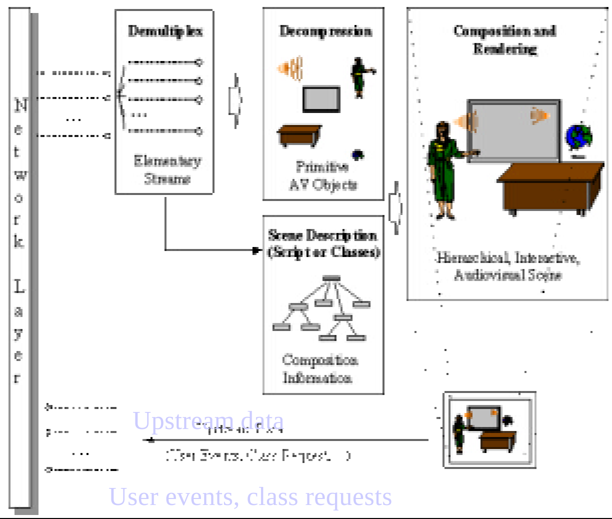
Standardized Ways To

- place a media objects anywhere in a given coordinate system;
- apply transforms to change the geometrical or acoustical appearances of media objects;
- group primitive media objects to form compound media objects;
- apply stream data to media objects to modify their attributes;
- change interactively user's viewing and listening points anywhere in the scene

Interaction with media objects

- change the viewing/listening point of the scene, e.g., by navigating through a scene;
- drag objects in the scene to a different position;
- trigger a cascade of events by clicking on specific objects, e.g., starting or sopping a video stream;
- select the desired language when multiple language tracks are available;
- more complex behavior

MPEG-4 Terminal



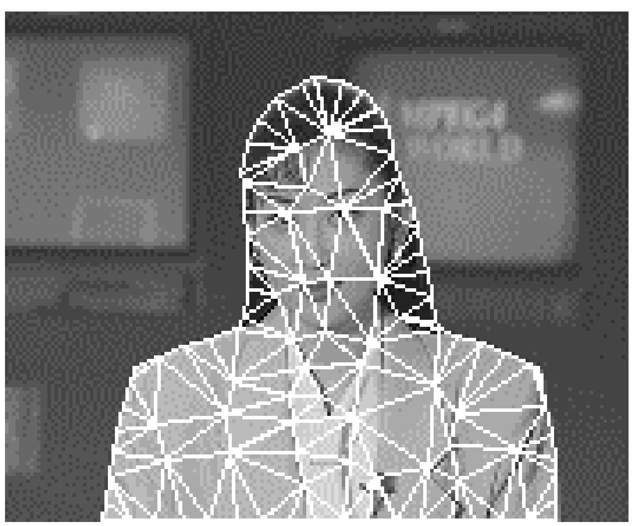
Textures, Images and Video

- Efficient compression of
  - images and video
  - textures for texture mapping on 2D and 3D meshes
  - implicit 2D meshes
  - time-varying geometry streams that animate meshes

Textures, Images and Video

- Efficient random access to all types of visual objects
- Extended manipulation functionalities for images and video sequences
- Content-based coding of images and video
- Content-based scalability of textures, images and video
- Spatial, temporal and quality scalability
- Error robustness and resilience

2-D Mesh Modeling



2-D Mesh Representation of Video Object

- Video Object Manipulation
  - Augmented Reality
  - Synthetic-object-transfiguration/animation
  - Spatio-temporal interpolation (e.g., frame rate up-conversion)
- Video Object Compression
  - transmit texture maps only at keyframes
  - animate texture maps for the intermediate frames

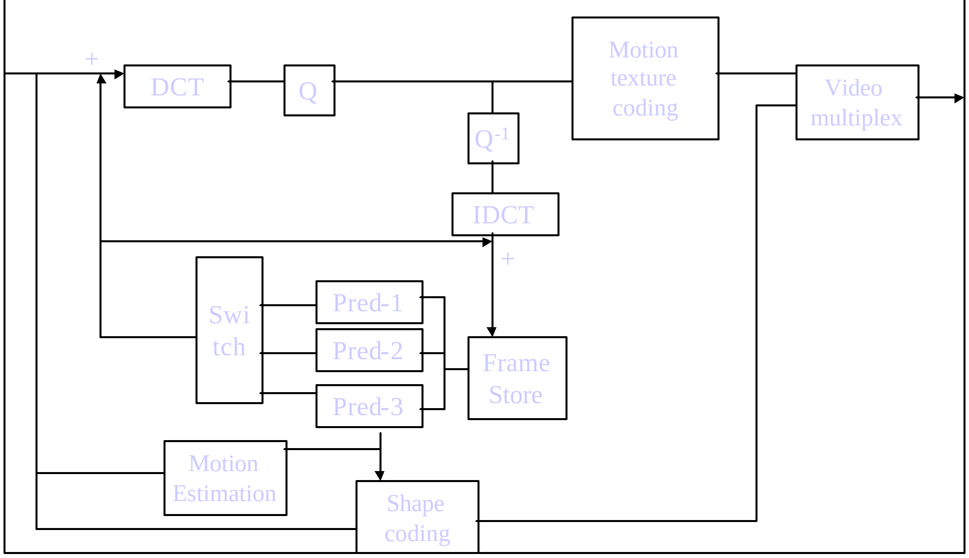
2-D Mesh Representation of Video Object

- Content-Based Indexing
  - Provides vertex-based object shape representation which is more efficient than the bitmap representation of shape-based object retrieval
  - Provides accurate object trajectory information that can be used to retrieve visual objects with specific motion
  - Animated key snapshots as visual synopsis of objects

MPEG-4 Video and Image Coding Scheme

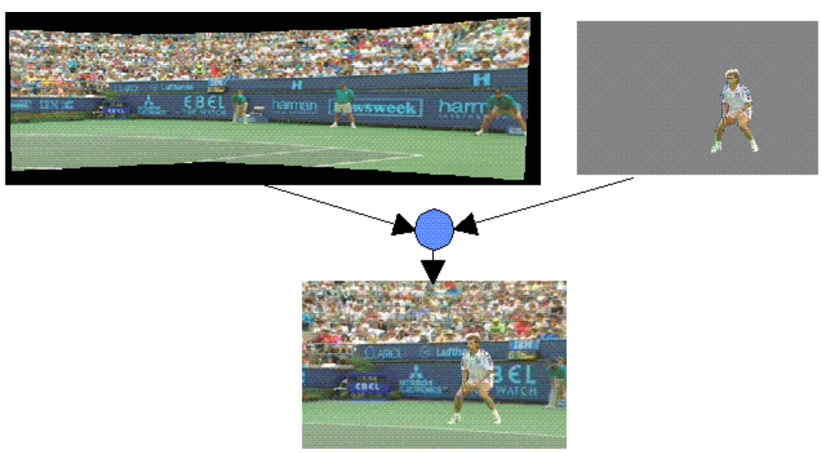
- Shape coding and motion compensation
- DCT-based texture coding
  - standard 8x8 and shape adpated DCT
- Motion compensation
  - local block based (8x8 or 16x16)
  - global (affine) for sprites

MPEG-4 Video Coder



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.



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.

Face Node

- FAP (Facial Animation Parameters) node
- Face Scene graph
- Face Definition Parameters (FDP)
- Face Interpolation Table (FIT)
- Face Animation Table (FAT)

Face Model

- Face model (3D) specified in VRLM, can be downloaded to the terminal with MPEG-4
- FAT maps FAPS to face model vertices.
- FAPS are quantized and differentially coded
- Typical compressed FAP bitrate is less than 2 kbps

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

Facial Animation Parameters (FAPS)

- 2 eyeball and 3 head rotations are represented using Euler angles
- Each FAP is expressed as a fraction of neutral face mouth width, mouth-nose distance, eye separation, or iris diameter.

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

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.

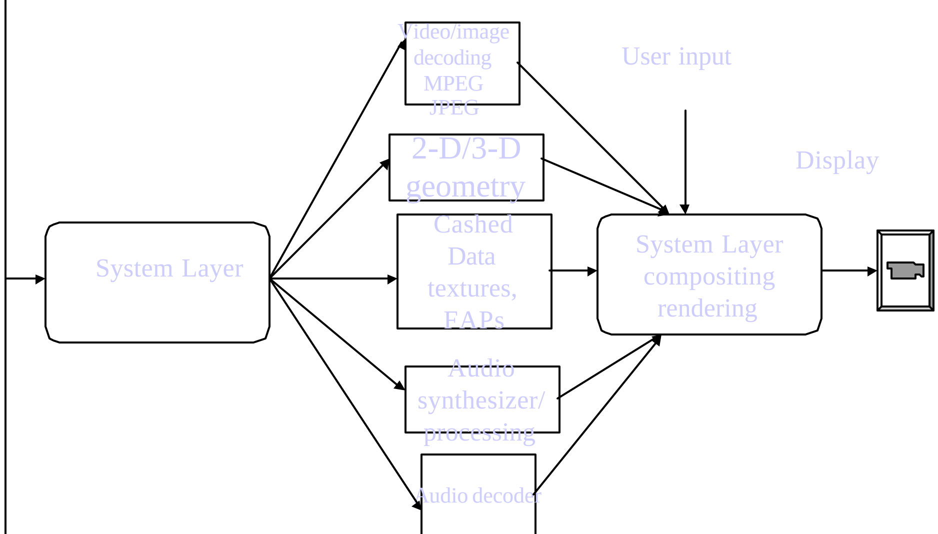
Facial Expressions

- 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-4

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