# Model-Based Video Coding

## Model-Based Compression

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

## Model-Based Compression

- Analysis
- Synthesis
- Coding

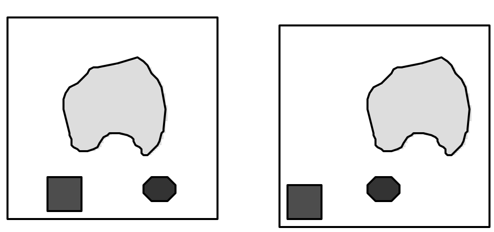
## Video Compression

- 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 Segmentation



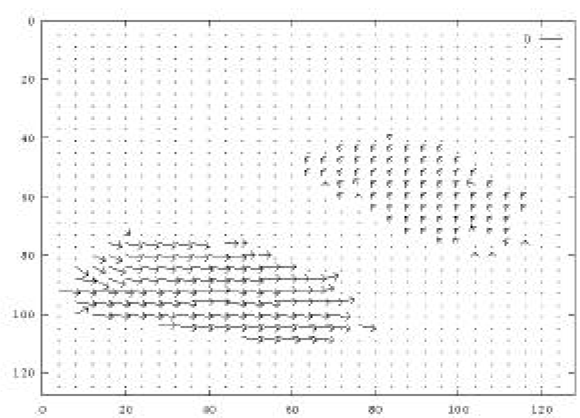
Frame k-1 Frame k

# Object Segmentation

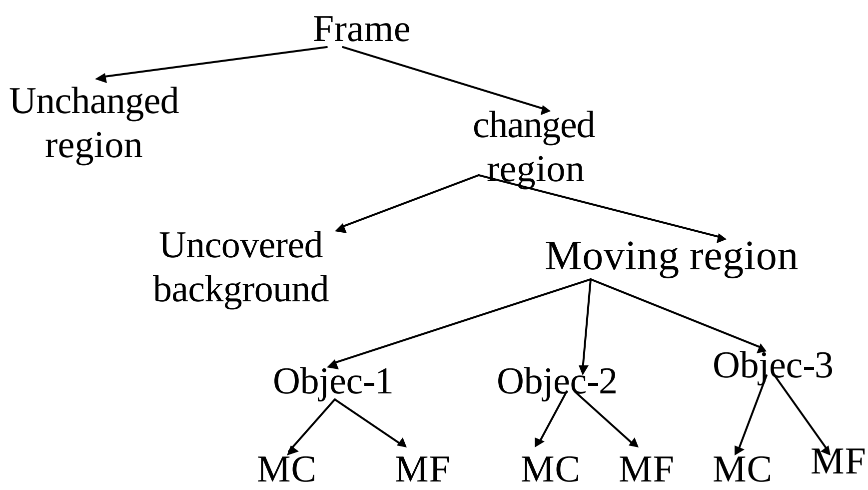
- Segmentation based on single frame (static)
- Motion-based segmentation
  - Optical flow based
    - Compute optical flow
    - Cluster optical flow into regions
  - Change detection
    - Threshold consecutive frame difference
    - Determine connected components
    - Estimate motion for each connected component
    - Determine motion failures
    - Iterate
- Simultaneous motion estimation and segmentation

# Tian & Shah optical flow

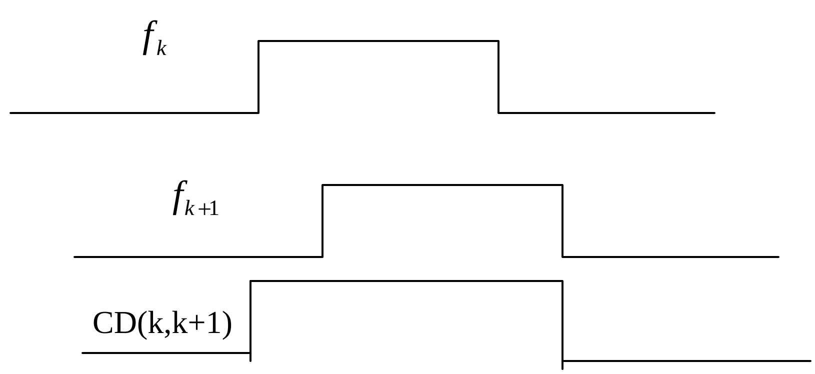
http://www.cs.ucf.edu/~vision/papers/shah/95/TIS95.pdf



# Object-Based Coding

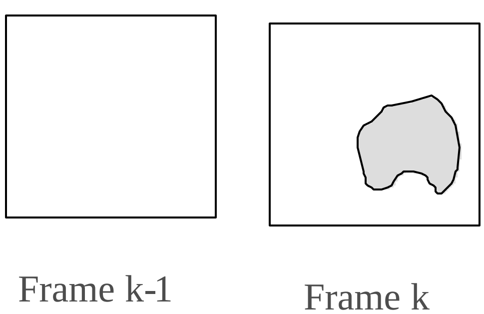



# Detection of Uncovered Background



## Detection of Uncovered Background

- All pixels in frame k+1 that are changed, are traced back to the frame k, using inverse of motion vectors.
- If the inverse of motion vector points to a pixel in frame k, which is within the changed region than it is a moving pixel; otherwise it is an uncovered background pixel.



## 2-D objects With Affine Motion

- Analysis
  - Segment image by change detection
  - Compute motion parameters, e.g. affine ($x'=Ax+b$)
  - Synthesize the region in the current frame using previous frame and the motion parameters
  - if the difference between actual and synthesized region is significant, recursively segment the region into small regions

## 2-D objects With Affine Motion

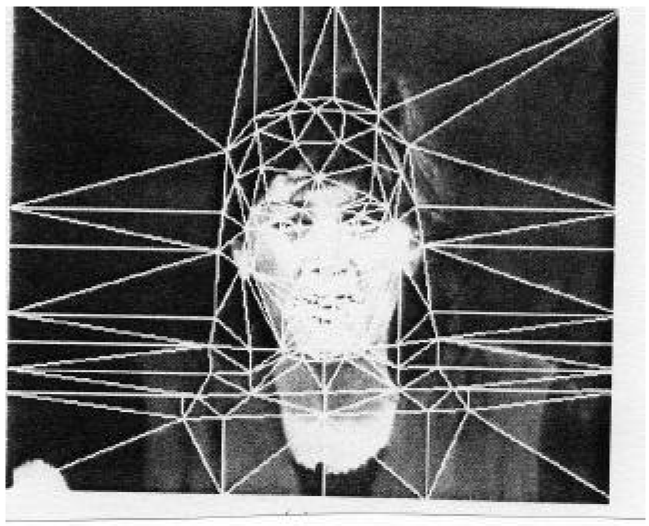
- Synthesis
  - Using final segmented regions and the motion parameters synthesize the frame, and compute the synthesis error.
- Coding
  - Code motion parameters (using 6 to 7 bits each)
  - Code region shapes
  - Code prediction errors

## Affine Transformation With Triangular Meshes

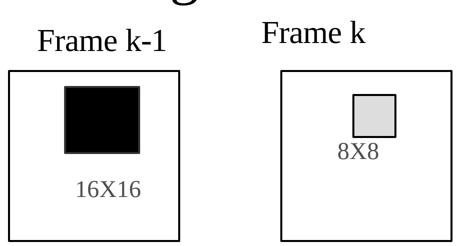
- Partition the current frame into triangular patches.
- Estimate rough motion vectors at vertices of triangles using block matching.
- Fit affine transformation to three vertices of each triangle using the motion vectors from block matching.
- Synthesize the current frame using affine transformation. Compute synthesized error.

## Affine Transformation With Triangular Meshes

- Encode at each grid point
  - motion vectors
  - synthesis error
  - no shape information needs to be coded



# Block Matching



# Block Matching

- For each 8X8 block, centered around pixel (x,y) in frame k, $B_k$
  - Obtain 16X16 block in frame k-1, centered around (x,y), $B_{k-1}$
  - Compute Sum of Squares Differences (SSD) between 8X8 block, $B_{k,}$ and all possible 8X8 blocks in $B_{k-1}$
  - The 8X8 block in $B_{k-1}$ centered around (x',y'), which gives the least SSD is the match
  - The displacement vector (optical flow) is given by u=x-x'; v=y-y'

# Sum of Squares Differences (SSD)

$$(u(x,y),v(x,y)) = \arg\min_{u,v=-3\ldots3} \sum_{i=0}^{-7}\sum_{j=0}^{-7}\left(f_k(x+i,y+j) - f_{k-1}(x+i+u,y+j+v)\right)^2$$

# Minimum Absolute Difference (MAD)

$$(u(x,y), v(x,y)) = \arg\min_{u,v=-3..3} \sum_{i=0}^{-7} \sum_{j=0}^{-7} \left| \left( f_k(x+i, y+j) - f_{k-1}(x+i+u, y+j+v) \right) \right|$$

# Maximum Matching Pixel Count (MPC)

$$T(x,y;u,v) = \begin{cases} 1 & \text{if } |f_k(x,y) - f_{k-1}(x+u, y+v)| \le t \\ 0 & \text{Otherwise} \end{cases}$$

$$(u(x,y), v(x,y)) = \arg\max_{u,v=-3..3} \sum_{i=0}^{-7} \sum_{j=0}^{-7} T(x+i, y+j; u,v)$$

# Cross Correlation

$$(u,v) = \arg\max_{u,v=-3\ldots3} \sum_{i=0}^{-7} \sum_{j=0}^{-7} \left( f_k(x+i, y+j)).( f_{k-1}(x+i+u, y+j+v) \right)$$

# Normalized Correlation

$$(u,v) = \arg\max_{u,v=-3\ldots3} \frac{\sum_{i=0}^{j=-7} \sum_{j=0}^{-7} \left( f_k(x+i, y+j)).( f_{k-1}(x+i+u, y+j+v) \right)}{\sqrt{\sum_{i=0}^{-7} \sum_{j=0}^{-7} f_{k-1}(x+i+u, y+j+v).\, f_{k-1}(x+i+u, y+j+v)}}$$

# Mutual Correlation

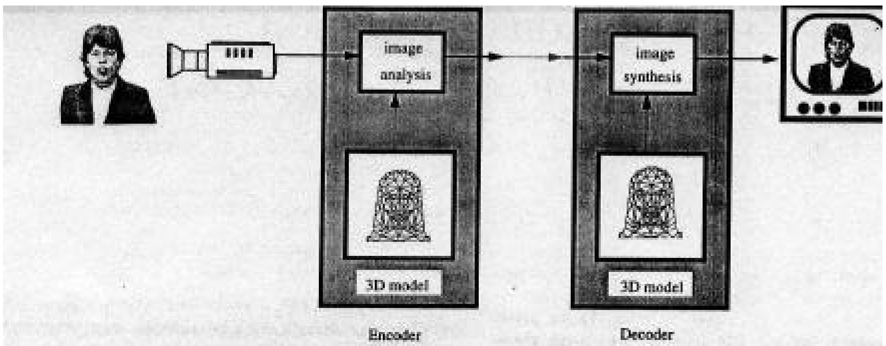
$$(u,v) = \arg\max_{u,v=-3..3} \frac{1}{64\sigma_1\sigma_2} \sum_{i=0}^{-7} \sum_{j=0}^{-7} \left( f_k(x+i, y+j) - \mu_1 \right).\left( f_{k-1}(x+i+u, y+j+v) - \mu_2 \right)$$

Sigma and mu are standard deviation and mean of patch-1 and patch-2 respectively

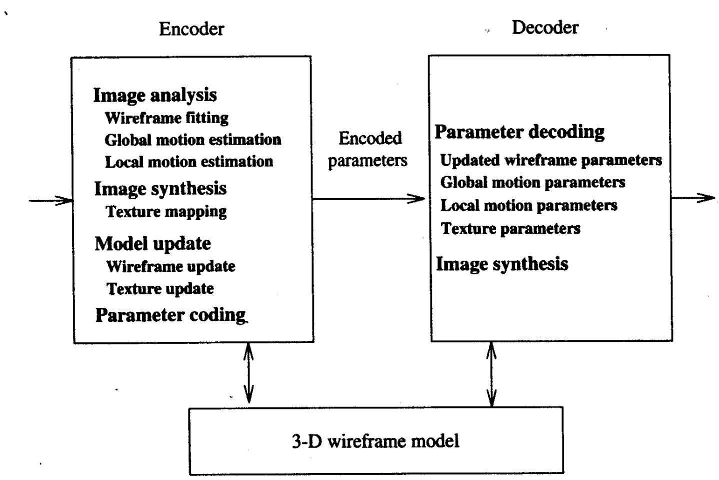
# Contents

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

# Model-Based Image Coding




# Model-Based Image Coding



# 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.

# Candide Model

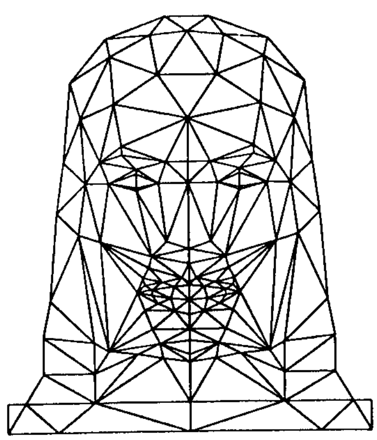


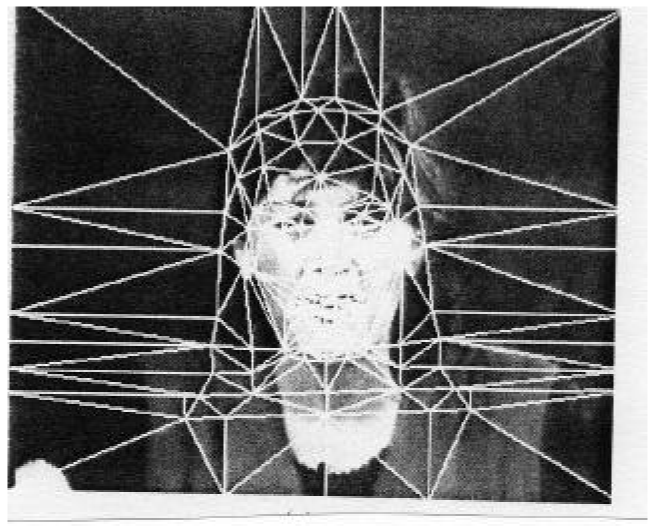
Fig. 2. Wire-frame model of the face.

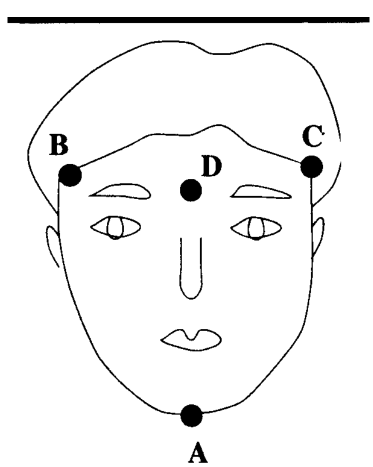
## Face Model

- 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.




B
C
D
A

# 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.

# Texture Mapping

(a) (b)

## 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

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

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

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

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

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

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

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

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

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

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

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

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

# 3-D Rigid+Non-rigid Motion

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

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

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

# 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

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

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