Real-Time Motion Estimation: Optical Flow - Feature Point Tracking

Jenn-Jier James Lien (連 震 杰)

Professor

Computer Science and Information Engineering

National Cheng Kung University

(O) (06) 2757575 ext. 62540

jjlien@csie.ncku.edu.tw

http://robotics.csie.ncku.edu.tw

Major Issues

1. **Template matching**

   1) 0 order/moment = texture (grayvalue or color)

   2) 1^{st} order/moment/derivation = gradient component

   > Optical flow for sub-pixel matching

2. **Optimization Using Sum of Squared Difference (SSD)**

   $\min \; E = \sum [Ax - b]^2$ **using 1st order Taylor series expansion**

   $Ax = b'$: estimation value. b: ground truth,

   $\min E = \min \sum [b - b']^2$

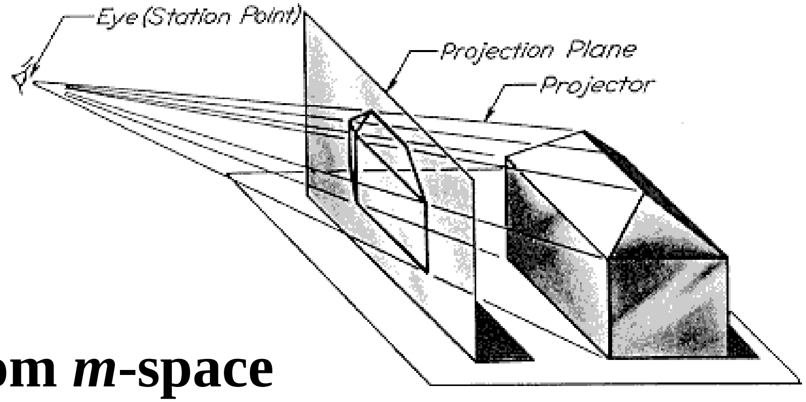
3. **Hessian matrix**

   1) Aperture problem

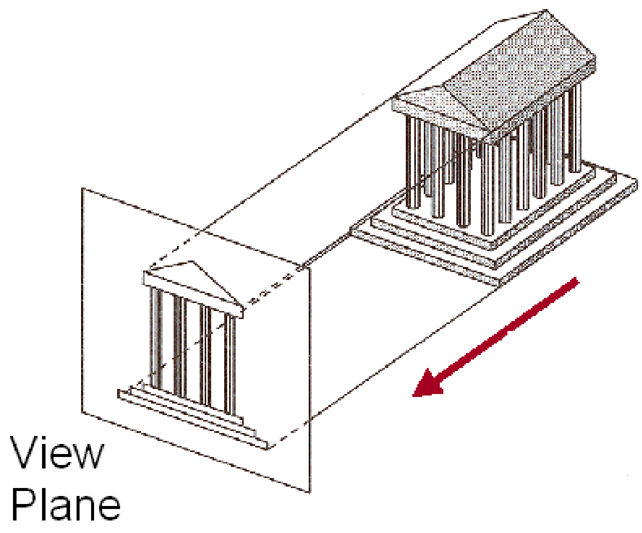
   2) Texture or textureless judgment

   3) Uncertainty

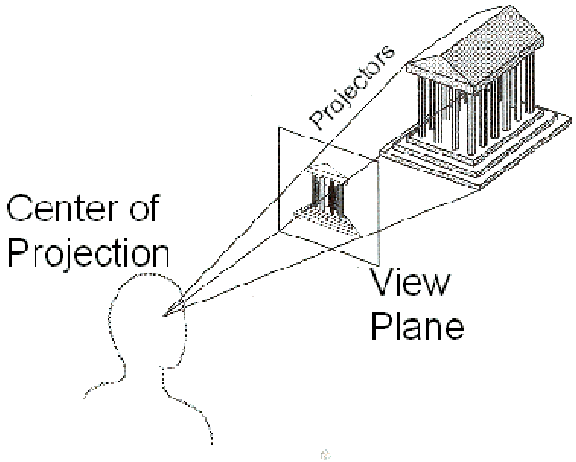
Camera Project



- **Projection – a transformation from *m*-space to *n*-space (*m*>*n*)**
  - For vision and graphics, it's 3D to 2D

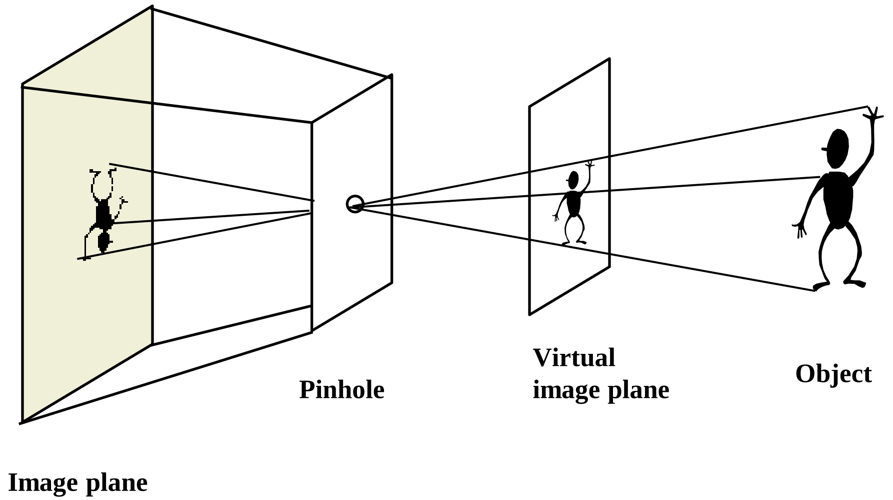


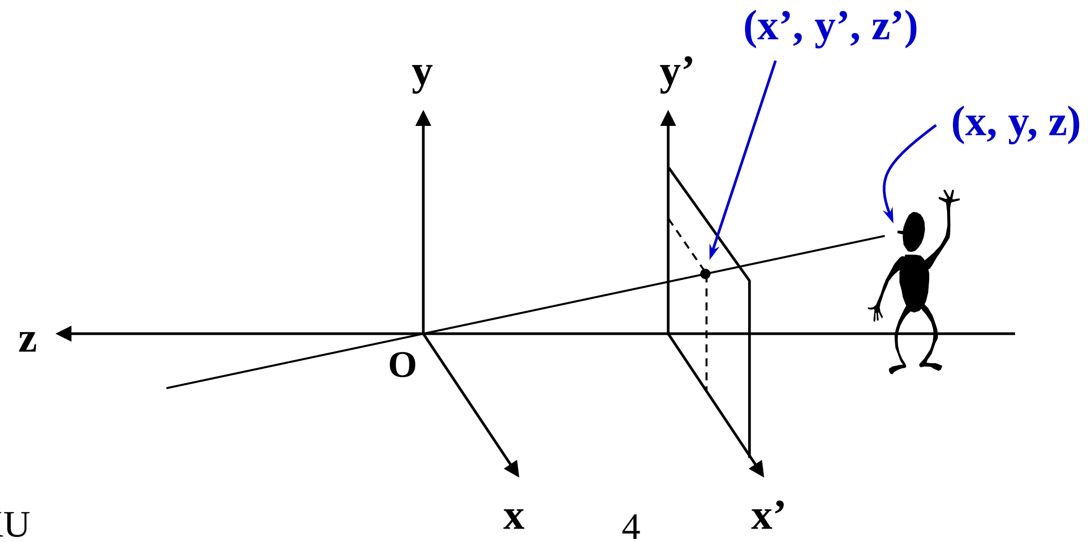
Orthogonal planar



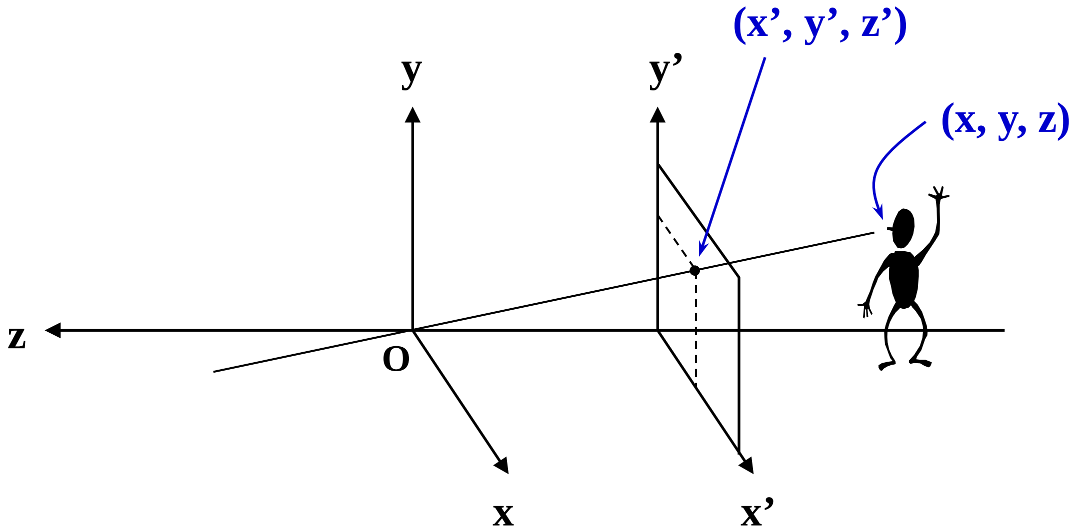
Perspective planar

Pinhole Model for Camera System: Perspective Projection





Perspective Projection Model

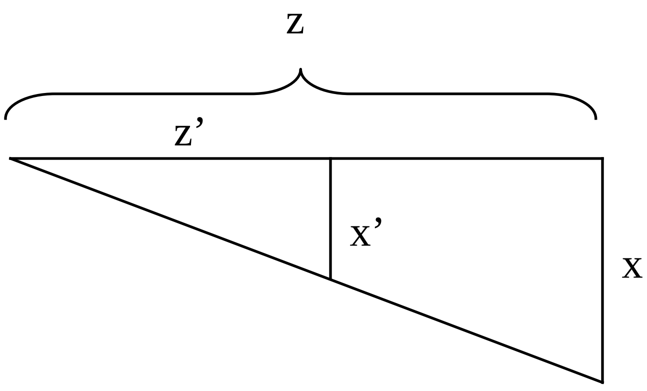


Projection from (x,y,z) to (x',y',z'):

$$x' = z' \frac{x}{z}$$

$$y' = z' \frac{y}{z}$$

$$z' = z' = \mathbf{f}$$



$$x' = z'\frac{x}{z}, \quad y' = z'\frac{y}{z}, \quad z' = z'$$

- **Can we compute this in a 3x3 matrix?**
  - No, it’s not linear
- **Instead, use *homogeneous coordinates***

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/z' & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

and

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x'/w' \\ y'/w' \\ z'/w' \end{bmatrix}$$

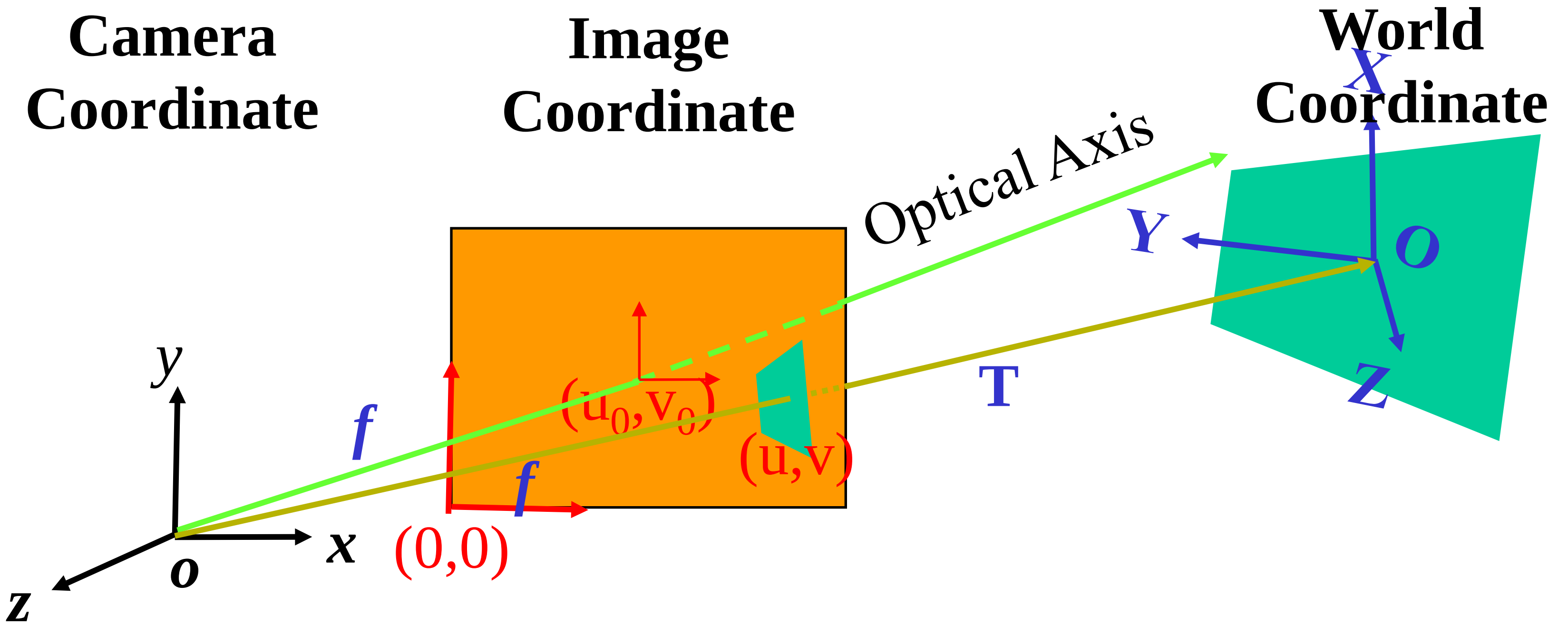
Homogenous Coordinates

- **A simplistic view**
  - Homogenous coordinates are a mechanism that allows us to associate points and vectors in space with vectors in R^4
- **Consider the four vector $(p, q, r, s)^t$**
  - if s is not equal to zero then this vector denotes the point with coordinates $(p/s, q/s, r/s)^t$
  - if s is equal to zero then this vector denotes the vector in the direction $(p, q, r)^t$
  - N.B. the vector $(0, 0, 0, 0)^t$ is explicitly disallowed
- **Note that under this association $(p, q, r, s)^t$ and $a*(p, q, r, s)^t$ where a is an arbitrary non-zero scalar denote the** same **entity**
- **This means that we cannot make distinctions between vectors that point along the same ray but have different magnitudes or signs**

Advantage of Homogenous Matrix

- **Homogenous coordinates allow us to**
  - Express rigid transformations in terms of matrix multiplication
  - Treat points and vectors in a unified framework

Pinhole Model for Camera System: Perspective Projection



$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{pmatrix} R_{11} & R_{21} & R_{31} \\ R_{12} & R_{22} & R_{32} \\ R_{13} & R_{23} & R_{33} \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$

Extrinsic Parameters

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Intrinsic Parameters

Camera Parameters

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R_{11} & R_{21} & R_{31} & T_1 \\ R_{12} & R_{22} & R_{32} & T_2 \\ R_{13} & R_{23} & R_{33} & T_3 \end{pmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Scale Factor: λ

- **Intrinsic Parameters:**
  - Scale Factor: $\alpha = -f\, k_u$
  - Scale Factor: $\beta = -f\, k_v$
  - **Skew Factor: □γ**
  - **Principal Point: (u_0, v_0)**
- **Extrinsic Parameters:**
  - Rotation: R
  - **Translation: T**

Local optimization (0 moment): 1 and 2
Global optimization: 3

Solution/Optimization of Homogenous Matrix

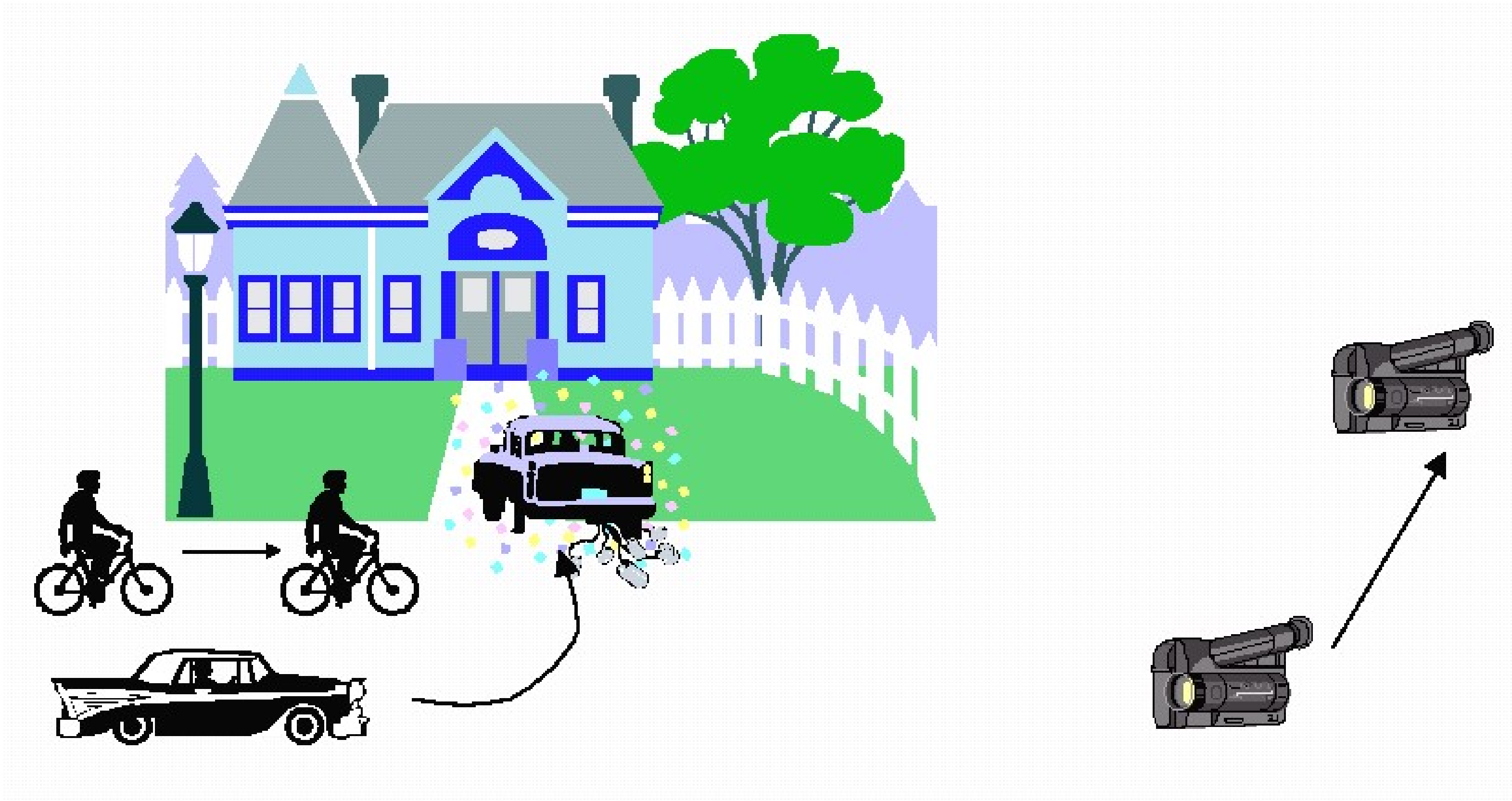
1. **Close Form Solution:**
   $Ax = 0$ => A^tA = Covariance Matrix => PCA => Smallest not-zero eigenvalue => eigenvector
2. **Pseudo Inverse:**
   $Ax = b$, $x=?$
3. **Sum of Squared Difference: (max liklihood – exponential term)**
   min $E = \sum [Ax - b]^2$

   3.1 $Ax = b'$: estimation value. b: ground truth, $E = \sum [b - b']^2$
   - a. Initial value estimation => Pseudo Inverse
   - b. L-M (non-linear approach)
     - b.1 First order Taylor series expansion
     - **b.2** 2^{nd} order Taylor series expansion (sensitive to noise)

   3.2 $Ax = b'$: estimation value. b'': estimation value, $E = \sum [b'' - b']^2$
   - **a. EM (Expected-Maximization), initial b = average value**

4. **Lagrange Approach (outlier)**
   min $E = \sum [Ax - b]^2 + \lambda (x^2+y^2)^2$

Movement between Camera and Object



Definition of Optical Flow

- Optical flow is the 2-D velocity field (u, v) induced in an image due to the projection of 3-D moving objects onto the image plane

Applications

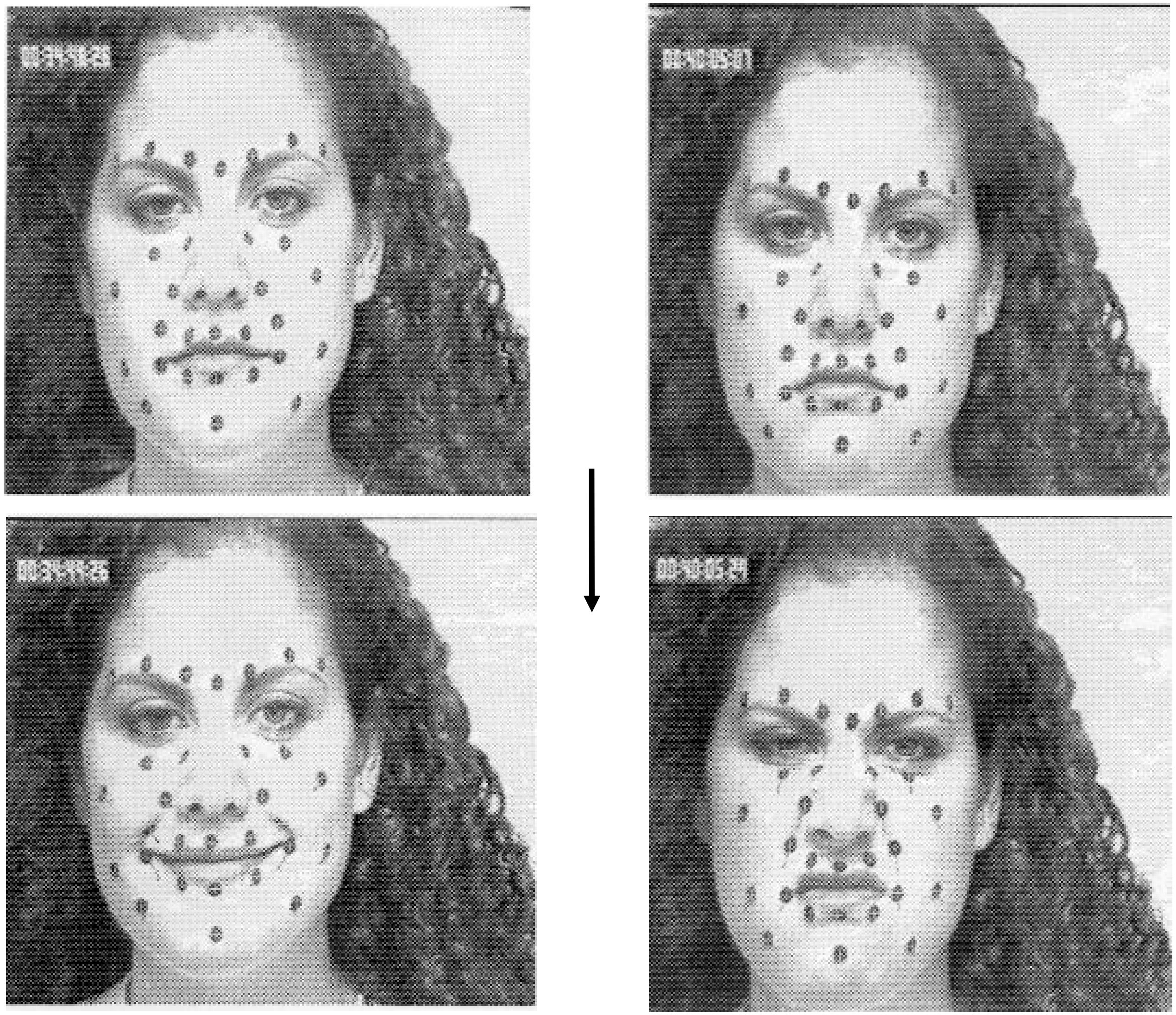
- **Corresponding points Vs. Optical flow**
  - 2D -> 3D reconstruction
  - Mosaic
  - Stabilization
  - …

  - SIFT
  - SURF
  - HOG

Methods of Computing Optical Flow

- **Three prevalent approaches to compute optical flow:**
  1) Token matching or correlation
     - » Extract features from each frame (gray level windows, edge detection)
     - » Match them from frame to frame
  2) Gradient techniques
     - » Relate optical flow to spatial and temporal image derivatives
  3) Velocity sensitive
     - » Frequency domain models of motion estimation

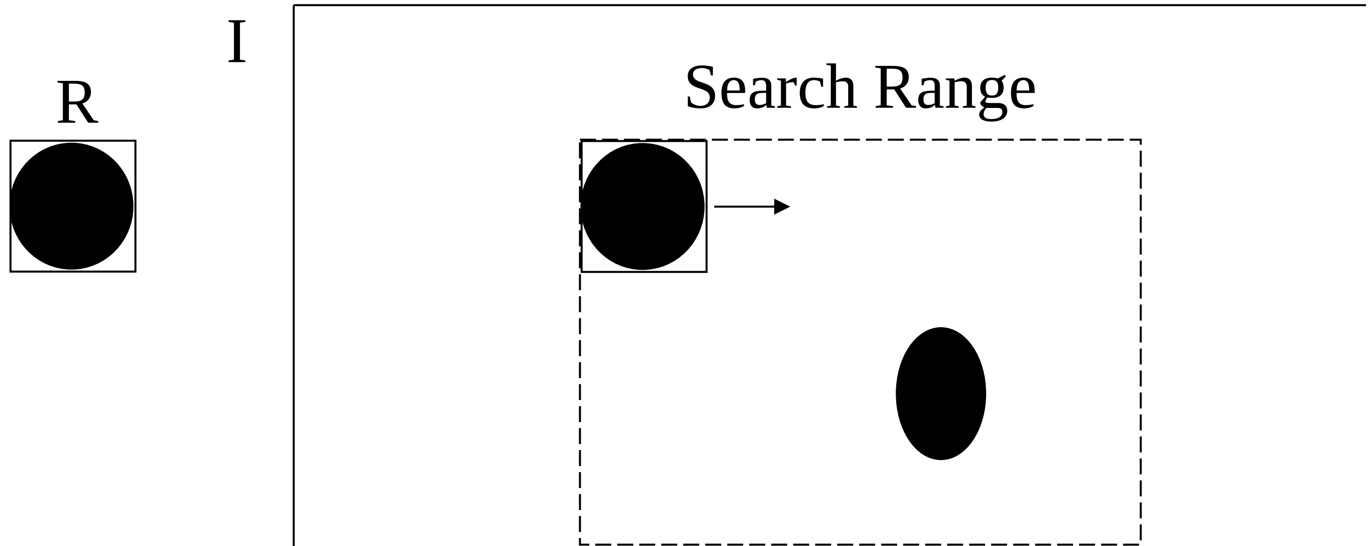
Example: Feature/Dot Tracking: Plastic Surgery



Template Matching

- **Matching (similarity measure) => Likelihood Probability**
  1. Texture
     - 0 order
     - Pro: Stable
     - Con: Sensitive to lighting
  2. Shape
     - 1st order
     - Pro: Not sensitive to lighting
     - Con: Sensitive to noise
  3. Color
     - RGB => HSV => Histogram domain
     - Pro: ROI (Region of interest)
     - Con: Confuse by similar background

Template Matching: Correlation Coefficient



$$\rho(x) = \left| \frac{\sum_{r \in R} [F(r) - \bar{F}][I(x+r) - \bar{I}(x)]}{\sqrt{\sum_{r \in R} [F(r) - \bar{F}]^2} \sqrt{\sum_{r \in R} [I(x+r) - \bar{I}(x)]^2}} \right| \qquad 0.0 \le \rho \le 1.0$$

Why minus $\bar{I}(x)$ and minus $\bar{F}$?

(Ex: Gaussian distribution Normalization)

Why divided standard deviation ?

Template Matching: Sub-Pixel Accuracy and One Pixel Match

- **Weakness: Fix, not flexible**
- **Sum-Square-Difference**
- **X1*X2: Correlation Matrix**
- **X1*X2 / (Lo_X1*Lo_X2): Correlation Matrix**
- **(X1-mX1)*(X2-mX2): Covariance Matrix**
- **(X1-mX1)*(X2-mX2)/(Lo_X1*Lo_X2): Covariance Matrix**
- **Optical Flow**

- **Strong correlation/covariance (matrix):**
  - Diagonal components of correlation/covariance matrix has brightest grayvalues

- **Search range**
- **Window size**

- **SIFT**
- **HOG**

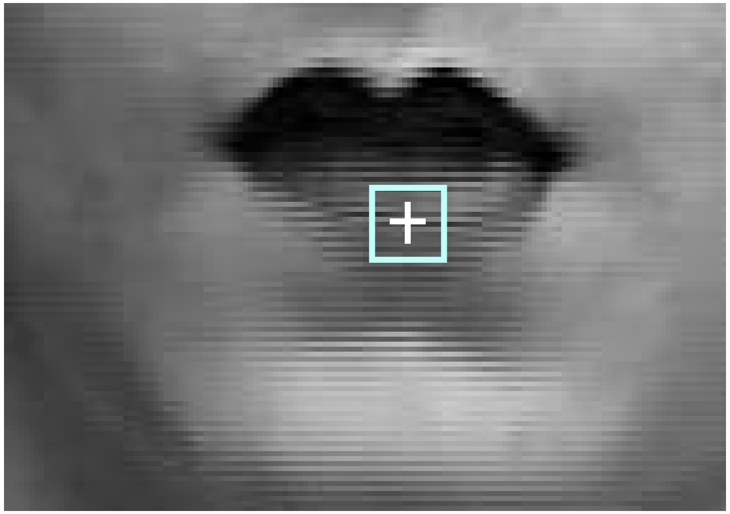
Disadvantage/Inaccurate of Feature/Dot Tracking

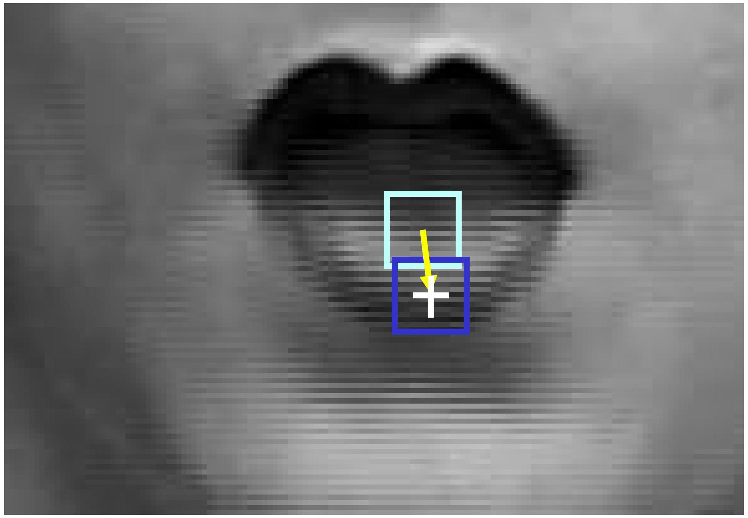
- **Feature/Dot is easily deformed.**
- **Feature/Dot is affected by the reflections/specular due to lighting.**
- **Computational time is slow when the number of feature/dots and search regions are increased.**
- **Mismatching when features/dots are closer to each other.**

Optical Flow: Difference Minimization

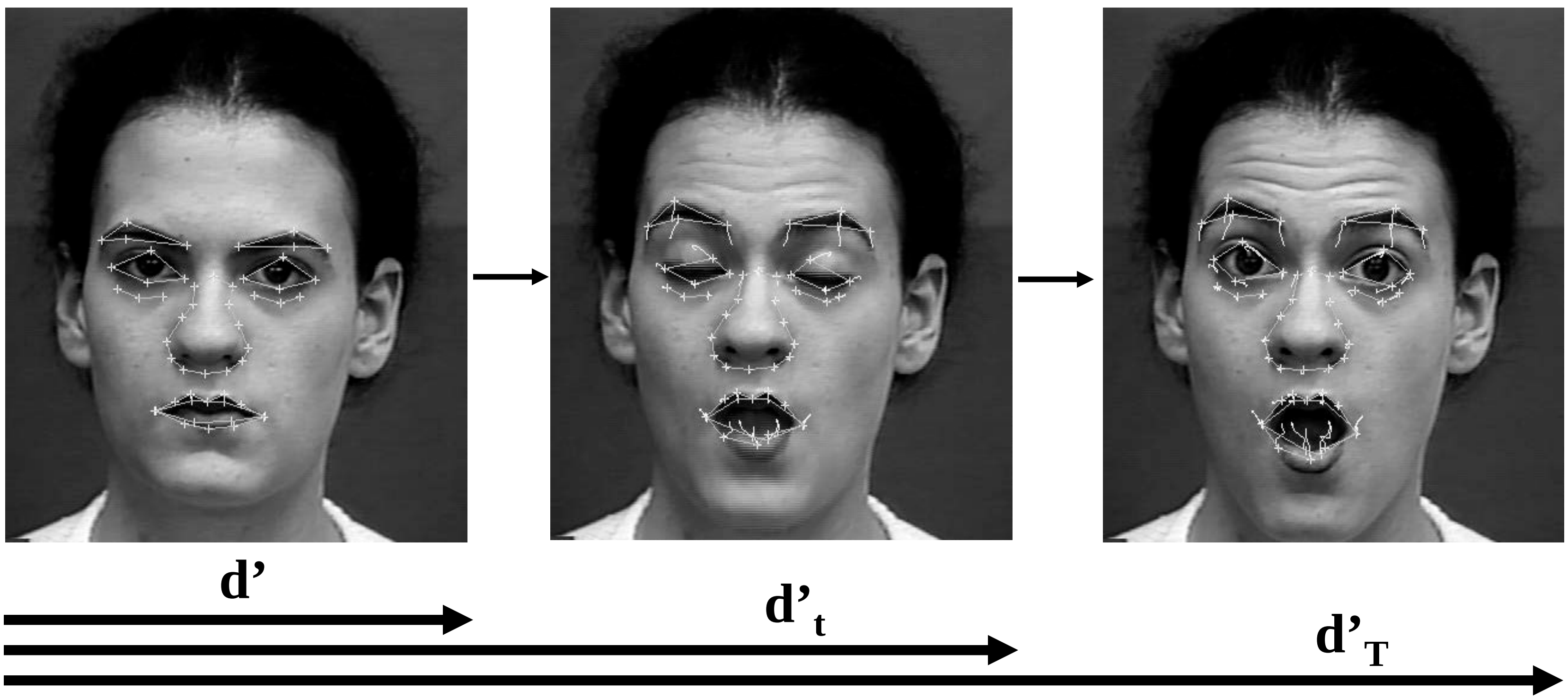
$$\min \; E = \sum_{x,y \in R} [\, I_t(x-u(x,y),\, y-v(x,y)) - I_{t+1}(x,y) \,]^2$$

E: Energy or Cost Function





Feature Point Tracking

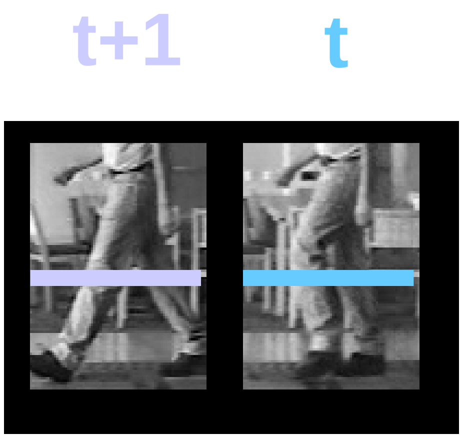


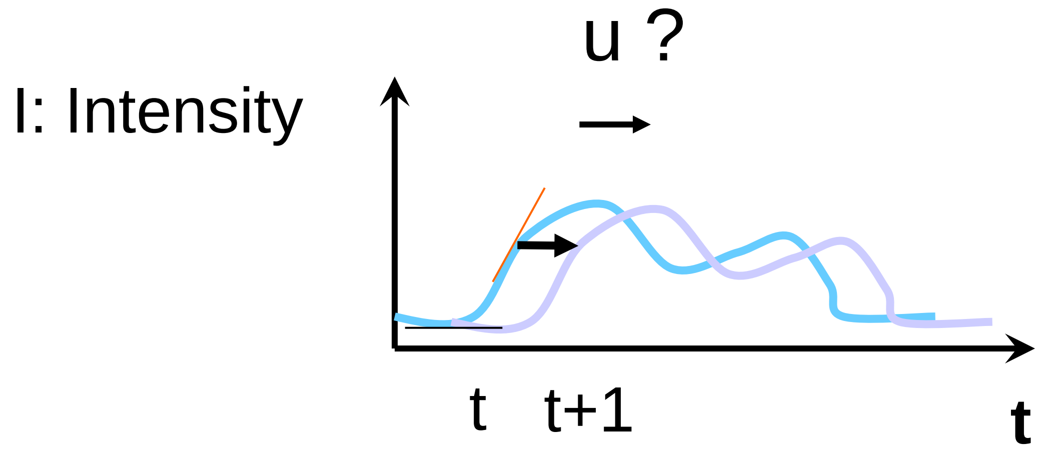
A 1-D Gradient Technique

- **Suppose we have 1-D image that changes over time due to a translation of the image**
- **Suppose we also assume that the image function is, at least over small neighborhoods, well approximated by a linear function (continue, Taylor series expansion)**
  - Completely characterized by its value and slope
- **Can we estimate the motion of the image by comparing its spatial derivative (texture/edge) at a point to its temporal derivative ?**
  - Example: Spatial derivative (gradient) is 10 units/pixel and temporal derivative is 20 units/frame

    Then motion is (20 units/frame)/(10 units/pixel)=2 pixels/frame

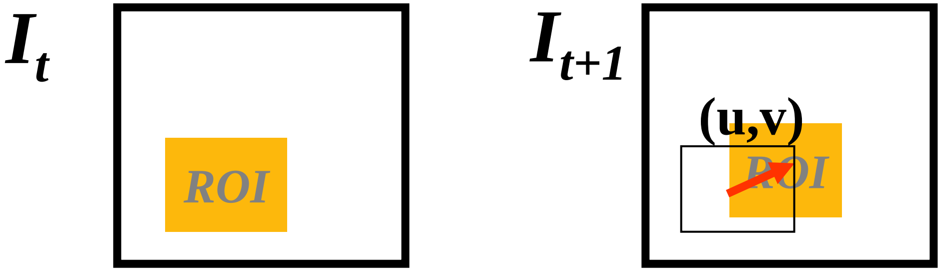
1-D Lucas-Kanade Flow





$$\min \; E = \sum_{x \in R} [\, I_t(x-u) - I_{t+1}(x) \,]^2$$

2-D Lucas-Kanade Flow



$$\min \; E = \sum_{x,y \in R} \left[I_t(x-u(x,y),\, y-v(x,y)) - I_{t+1}(x,y) \right]^2$$

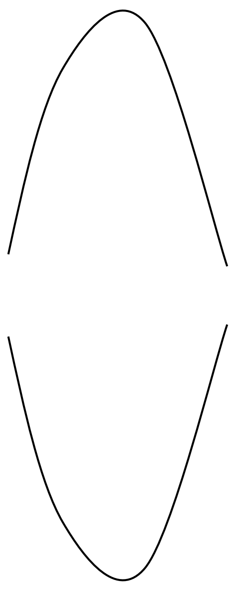
Flow Computation: Physical Interpretation

- Q: what is $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$?
- A: image brightness gradient direction
  - known (spatial derivatives) (texture, edge)
- Q: what is (u, v)?
- A: local motion vector
  - unknown
- Q: what is $\frac{\partial I}{\partial t}$?
- A: change of brightness at a location w.r.t time
  - known (temporal derivative)

Lucas-Kanade Flow: SSD Minimization

$$\boxed{\min \; E = \sum_{x \in R} \left[I(x+h) - F(x) \right]^2}$$

template



$$\frac{\partial E}{\partial h} = \sum_{x \in R} 2\left[I(x+h) - F(x) \right] * \frac{\partial}{\partial h} I(x+h) = 0$$

By first order Taylor series expansion

$$I(x+h) \approx I(x) + h \frac{\partial I(x)}{\partial x}$$

then

$$\sum_{x \in R} 2\left[I(x) + h \frac{\partial I(x)}{\partial x} - F(x) \right] * \frac{\partial}{\partial h} \left[I(x) + h \frac{\partial I(x)}{\partial x} \right] = 0$$

0

$$\sum_{x\in R}\left[I(x)-F(x)\right]\left(\frac{\partial h}{\partial h}\frac{\partial I(x)}{\partial x}\right)+\sum_{x\in R}h\frac{\partial I(x)}{\partial x}(\frac{\partial h}{\partial h}\frac{\partial I(x)}{\partial x})=0$$

$$h=\left[\sum_{x\in R}(\frac{\partial I(x)}{\partial x})\left(F(x)-I(x)\right)\right]\left[\sum_{x\in R}\left(\frac{\partial I(x)}{\partial x}\right)\left(\frac{\partial I(x)}{\partial x}\right)\right]^{-1}$$

Iteration

$$h_0=0$$

$$h_{n+1}=h_n+\left[\sum_{x\in R}\left(\frac{\partial I(x)}{\partial x}\right)\Big|_{x+h_n}\left[F(x)-I(x+h_n)\right]\right]\left[\sum_{x\in R}\left(\frac{\partial I(x)}{\partial x}\right)\left(\frac{\partial I(x)}{\partial x}\right)\Big|_{x+h_n}\right]^{-1}$$

$$h_{n+1}=h_n+e*G^{-1}\quad \text{until}\quad \left|h_{n+1}-h_n\right|<\varepsilon$$

Lucas-Kanade Flow Equation

$$h_{n+1} = h_n + e * G^{-1} \quad \text{until} \quad \left| h_{n+1} - h_n \right| < \varepsilon$$

where

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \frac{\partial I}{\partial x_1}\Big|_{x+h_n} \left[F(x_1) - I(x_1 + h_n) \right] \\ \sum_{x \in R} \frac{\partial I}{\partial x_2}\Big|_{x+h_n} \left[F(x_2) - I(x_2 + h_n) \right] \end{bmatrix} \qquad G^{-1} = \frac{\begin{bmatrix} G_{22} & -G_{12} \\ -G_{12} & G_{11} \end{bmatrix}}{G_{11}G_{22} - G_{12}{}^2}$$

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right)^2 \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} \\ \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_2} \right)^2 \Big|_{x+h_n} \end{bmatrix}$$

Lucas-Kanade Flow: General Case

Minimize a cost function *E* of the sum of squared differences (SSD)

min $$E(d(x)) = \sum_{x \in R} [I_t(x - d(x)) - I_{t+1}(x)]^2 w(x)$$

$$d(x) = d^i(x) + \Delta d(x)$$

$$I_t(x - d) = I_t(x - (d^i + \Delta d))$$

The first order Taylor's expansion:

$$I_t(x \underline{- d^i} - \Delta d) \approx I_t(x - d^i) - I_t^{'}(x - d^i)^T \Delta d = 0$$

The incremental change Δd in the SSD cost function:

$$E(\Delta d) = E(d^i + \Delta d) - E(d^i)$$

$$\approx \sum_{x \in R} [I_t(x - d^i) - I_t^{'}(x - d^i)^T \Delta d - I_{t+1}(x)]^2 w(x) - \sum_{x \in R} [I_t(x - d^i) - I_{t+1}(x)]^2 w(x)$$

$$= \sum_{x \in R} [I_t^{'}(x - d^i)^T \Delta d]^2 w(x) - 2 \sum_{x \in R} [I_t(x - d^i) - I_{t+1}(x)] I_t^{'}(x - d^i)^T \Delta d w(x)$$

$$= \Delta d^T G \Delta d - 2 e^T \Delta d$$

where

$$G = \sum_x I_t^{'}(x - d^i) I_t^{'}(x - d^i)^T w(x)$$

$$e^T = \sum_x [I_t(x - d^i) - I_{t+1}(x)] I_t^{'}(x - d^i)^T w(x)$$

$$G = \sum_{x} I_t^{'}(x - d^{i}) I_t^{'}(x - d^{i})^{T} w(x)$$

The Hessian matrix of the gradients of I_t with a window function $w(x)$

$$e^{T} = \sum_{x} [I_t(x - d^{i}) - I_{t+1}(x)] I_t^{'}(x - d^{i})^{T} w(x)$$

The difference-gradient row vector which is the product of the difference (or error) between the regions in the two consecutive images and the gradient of the gray-value I_t together with a window function $w(x)$

The maximum decrement E(Δ*d*) occurs when its gradient with respect to Δ*d* is zero

$$\frac{\partial E(\Delta d)}{\partial(\Delta d)} = G\Delta d + (\Delta d^T G)^T - 2e = 2(G\Delta d - e) = 0$$

Hence,

$$\Delta d(x) = G^{-1}e$$

Initializing $d^{(0)}(x) = [0,0]^T$ and following above equations, the optical flow $d(x)$ can be robustly estimated through iterations yielding the sub-pixel accuracy.

The motion estimate $d(x)$ is more accurate when the gradients of both I_t(x) and I_{t+1}(x) are large and nearly equal

texture

not deform

LM Optimization

- **1. Initialization by 0 order approach:**
  - Correlation Coefficient Match:
  - Lx = 0 or Ax =b

- **2. LM by 1st and 2nd order deviations (iteration): Taylor Extension**

$$h_{n+1} = h_n + e * G^{-1} \quad \text{until} \quad \left| h_{n+1} - h_n \right| < \varepsilon$$

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \frac{\partial I}{\partial x_1} \Big|_{x+h_n} \left[F(x) - I(x + h_n) \right] \\ \sum_{x \in R} \frac{\partial I}{\partial x_2} \Big|_{x+h_n} \left[F(x) - I(x + h_n) \right] \end{bmatrix}$$

$$G^{-1} = \frac{\begin{bmatrix} G_{22} & -G_{12} \\ -G_{12} & G_{11} \end{bmatrix}}{G_{11} G_{22} - G_{12}{}^2}$$

where

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right)^2 \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} \\ \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_2} \right)^2 \Big|_{x+h_n} \end{bmatrix}$$

Template Matching: Search Range, Window Size and Pyramid

- **Search range**
- **Window size**
  - Large window size
    - Contain more texture + structure information
    - Sensitive to distorted information
    - Slow
  - Small window size
    - Contain less texture + structure information
    - Not sensitive to distorted information
    - Fast
- **Pyramid**
  - Speed up from N^2 to N log N
  - Large tracking distance by extending search range

Hessian Matrix

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right)^2 \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} \\ \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_2} \right)^2 \Big|_{x+h_n} \end{bmatrix}$$

Covariance Matrix

1. **Covariance Matrix A Vs. Correlation Matrix A'**

2. **$A=(A'-m)=UWV^T$**
   - **Gaussian distribution N(m, Lamda^2),**
   - **as affine transform**

3. **Affine transform –**

   m: as the translation terms

   U: Eigenvector matrix as the rotation matrix

   W: Eigenvalues (Lamda^2) as the variance/scaling terms at denomination for normalization.

Demo: Feature Point Tracking

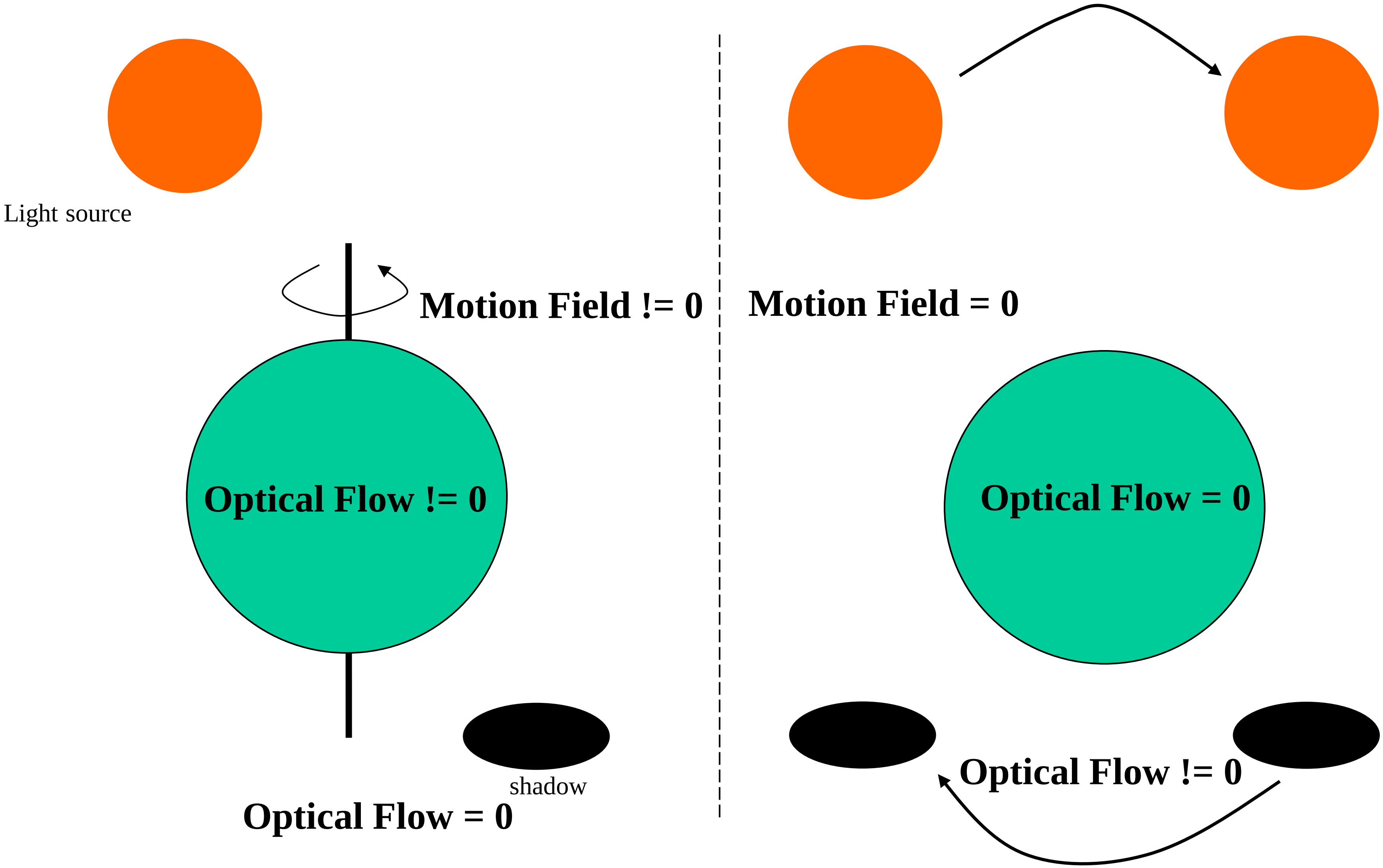
Good Feature to Track

1-Pixel Tracking

Affine Flow

$$\min \ E = \sum_{x \in R} \left[I(x + (ax + h)) - F(x) \right]^2$$

Motion Field and Optical Flow



Optical Flow Constraint Equation

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t)$$

1. Optical flow (velocity) vector (u(x,y),v(x,y))

$$\delta x = u\delta t,\ \ \delta y = v\delta t$$

$$I(x+u\delta t, y+v\delta t, t+\delta t) = I(x, y, t)$$

For small time interval $\delta t \to 0$

and the motion field is continuous almost everywhere

2. And by the first order Taylor's expansion:

$$I(x, y, t) + \delta x \frac{\partial I}{\partial x} + \delta y \frac{\partial I}{\partial v} + \delta t \frac{\partial I}{\partial t} = I(x, y, t)$$

$$\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t}\frac{\partial t}{\partial t} = 0$$

$$\frac{\partial I}{\partial x} u \quad + \frac{\partial I}{\partial y} v \quad + \frac{\partial I}{\partial t}\ 1 = 0$$

Optical flow constraint equation:

(express a constraint on the components u and v of the optical flow)

(Or called constraint line L)

$$I_x u + I_y v + I_t = 0$$

$$(I_x, I_y) \cdot (u, v) = -I_t \neq 0$$

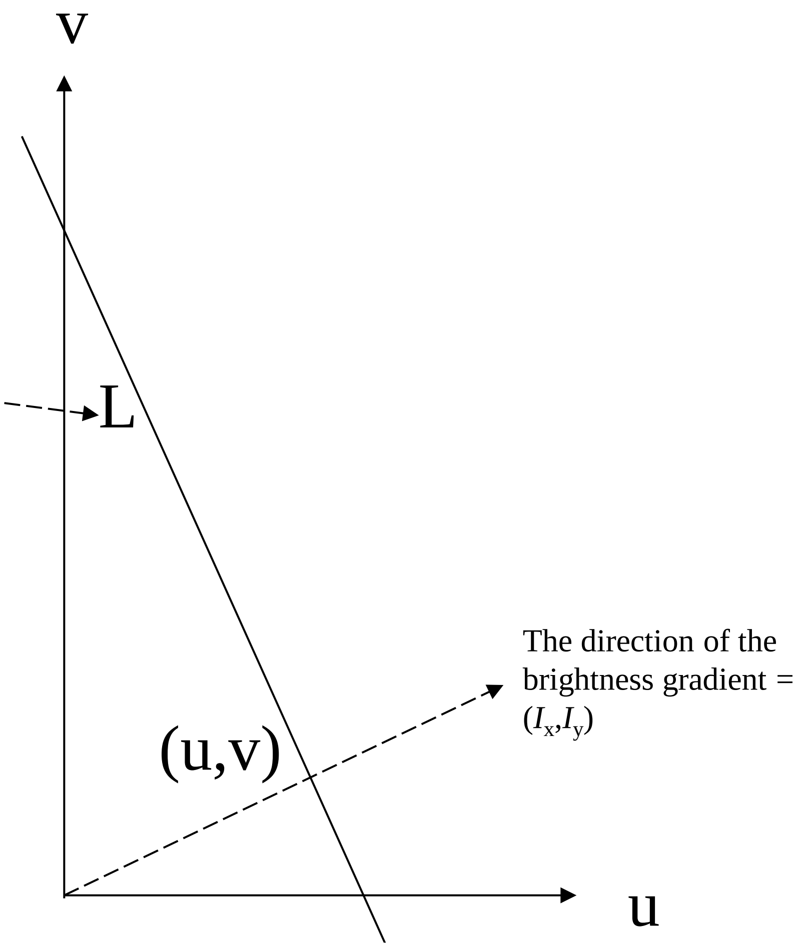
Spatial change =texture => Hessian Matrix

Temporal change =motion flow

Feature/patch moving

The component of optical flow in the direction of the brightness gradient $(I_x, I_y)^T$ is thus

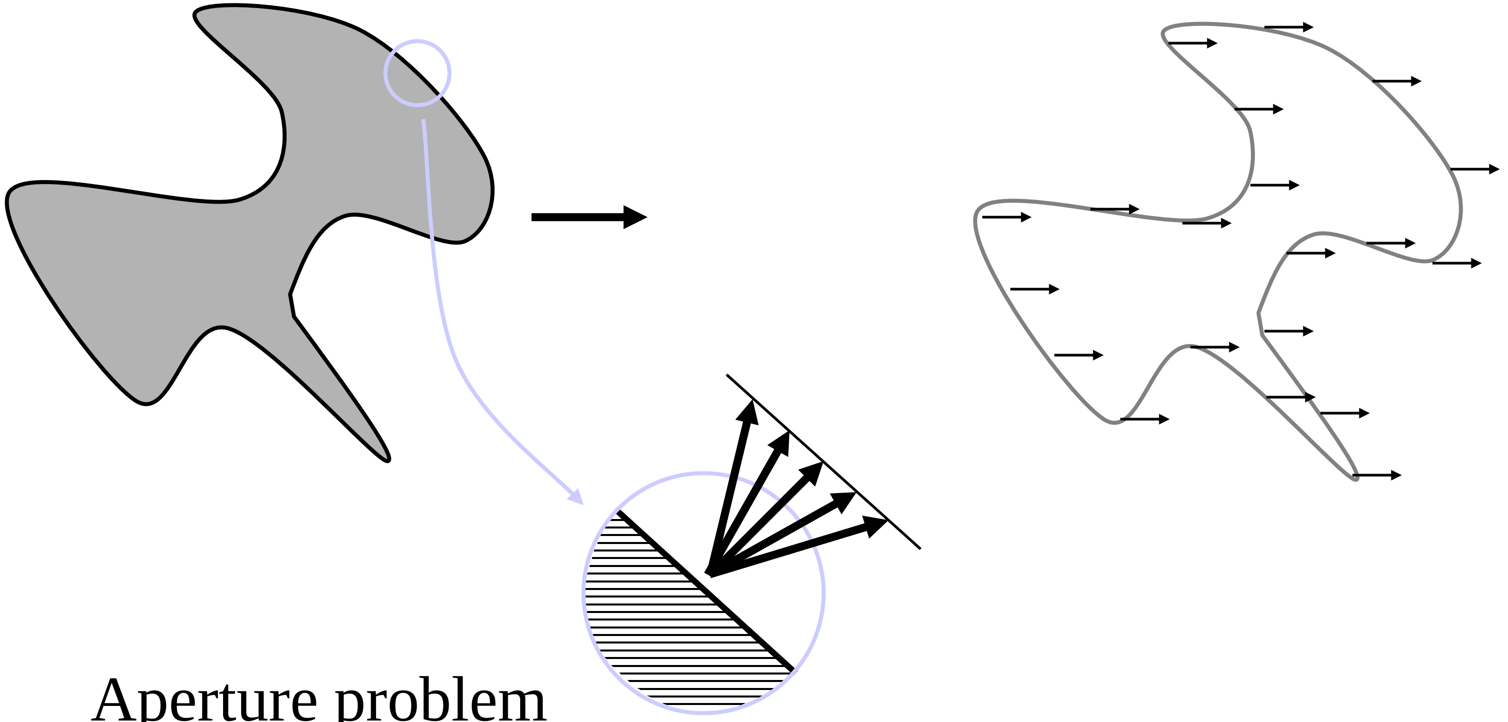
$$d(0, L) = \frac{I_t}{\sqrt{I_x^2 + I_y^2}}$$



We can only determine the component in the direction of the brightness gradient.

We cannot, however, determine the component of the optical flow at right angles to this direction, that is, along the isobrightness contour. This ambiguity is also known as the aperture problem.

Aperture Problem



Aperture problem

$$h_{n+1} = h_n + e * G^{-1} \quad \text{until} \quad \left| h_{n+1} - h_n \right| < \varepsilon$$

where

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \frac{\partial I}{\partial x_1} \Big|_{x+h_n} \left[F(x) - I(x+h_n) \right] \\ \sum_{x \in R} \frac{\partial I}{\partial x_2} \Big|_{x+h_n} \left[F(x) - I(x+h_n) \right] \end{bmatrix} \qquad G^{-1} = \frac{\begin{bmatrix} G_{22} & -G_{12} \\ -G_{12} & G_{22} \end{bmatrix}}{G_{11} G_{22} - G_{12}{}^2}$$

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right)^2 \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} \\ \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_1} \right) \left(\frac{\partial I(x)}{\partial x_2} \right) \Big|_{x+h_n} & \sum_{x \in R} \left(\frac{\partial I(x)}{\partial x_2} \right)^2 \Big|_{x+h_n} \end{bmatrix}$$

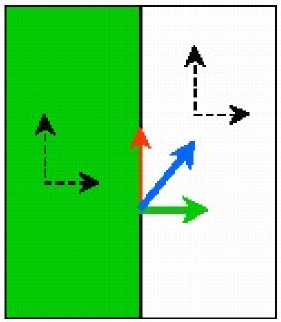
$$I_x u + I_y v + I_t = 0$$

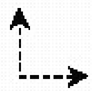
$$(I_x, I_y) \cdot (u, v) = -I_t \neq 0$$

Spatial change =texture => Hessian Matrix

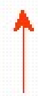
Temporal change =motion flow

Feature/patch moving

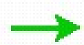




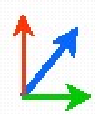
no spatial change in brightness
induce no temporal change in brightness
no discernible motion



motion perpendicular to local gradient
induce no temporal change in brightness
no discernible motion

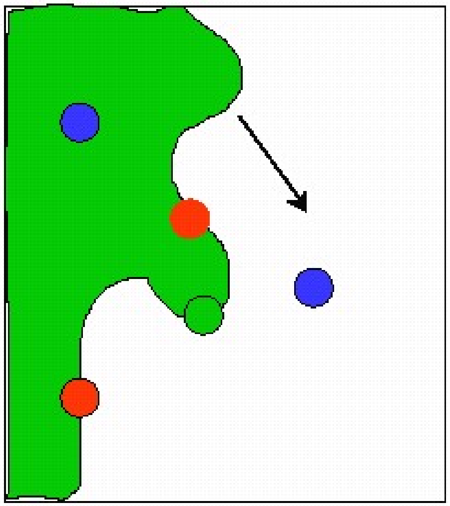


motion in the direction of local gradient
induce temporal change in brightness
discernible motion



only the motion component
in the direction of local gradient
induce temporal change in brightness
discernible motion

Aperture Problem



- intensity gradient is zero
  no constraints on (u,v) $\quad (0,0)\cdot(u,v)=0$
  interpolated from other places

- intensity gradient is nonzero
  but is *constant*
  one constraints on (u,v) $\quad (\frac{\partial I}{\partial x},\frac{\partial I}{\partial y})\cdot(u,v)=-\frac{\partial I}{\partial t}$
  only the component along the gradient
  are recoverable

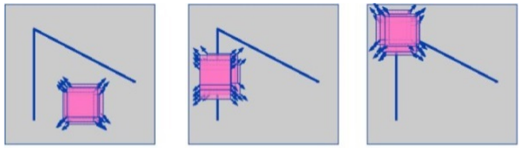
- intensity gradient is nonzero
  and *changing*
  multiple constraints on (u,v)
  motion recoverable

$$(\frac{\partial I}{\partial x_1},\frac{\partial I}{\partial y_1})\cdot(u,v)=-\frac{\partial I}{\partial t}_{(x_1,y_1)}$$

$$(\frac{\partial I}{\partial x_2},\frac{\partial I}{\partial y_2})\cdot(u,v)=-\frac{\partial I}{\partial t}_{(x_2,y_2)}$$

Addition: Harris Corner Detector (1/3)

- Basic Idea



"flat" region: no change in all directions | "edge": no change along the edge direction | "corner": significant change in all directions

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

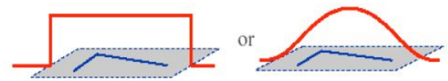
Window function | Shifted intensity | Intensity

textureless

For nearly constant patches, this will be near 0.
For very distinctive patches, this will be larger.
Hence... we want patches where E(u,v) is LARGE.

smoothing

Window function $w(x,y)$ =



1 in window, 0 outside | Gaussian

SSD:

$$\sum [I(x+u,y+v)-I(x,y)]^2 \approx \sum [I(x,y)+uI_x+vI_y-I(x,y)]^2$$ First order approx

$$= \sum u^2I_x^2+2uvI_xI_y+v^2I_y^2$$

$$= \begin{bmatrix} u & v \end{bmatrix} \left(\sum \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}\right) \begin{bmatrix} u \\ v \end{bmatrix}$$ Rewrite as matrix equation

$$= \sum \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Taylor Series for 2D function

$$f(x+u,y+v)=f(x,y)+uf_x(x,y)+vf_y(x,y)+$$

First partial derivatives

$$\frac{1}{2!}\left[u^2f_{xx}(x,y)+uvf_{xy}x,y+v^2f_{yy}(x,y)\right]+$$

Second partial derivatives

$$\frac{1}{3!}\left[u^3f_{xxx}(x,y)+u^2vf_{xxy}(x,y)+uv^2f_{xyy}(x,y)+v^3f_{yyy}(x,y)\right]$$

Third partial derivatives

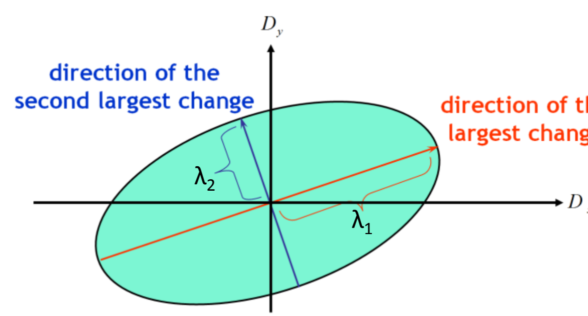
+ . . . (Higher order terms)

- For small shifts [u,v] we have a bilinear approximation:

$$E(u,v)\cong[u,v]\ M\begin{bmatrix} u \\ v \end{bmatrix}$$

Hessian Matrix: M

$$M=\sum_{x,y} w(x,y)\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$$

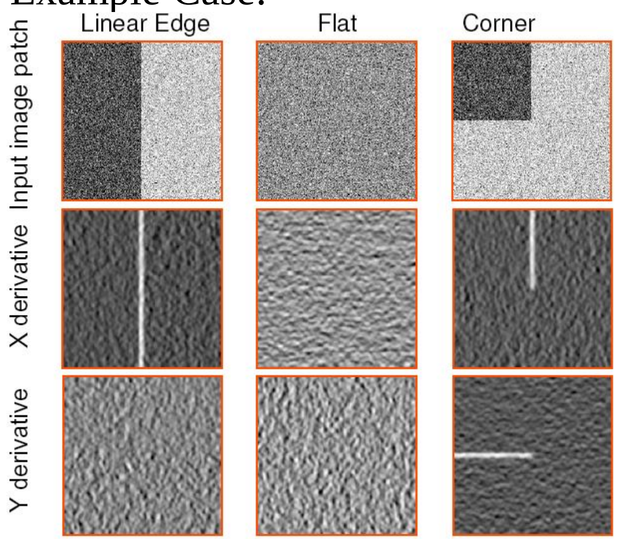


- Compute $\lambda_{1,2}$ (eigenvalues) of M

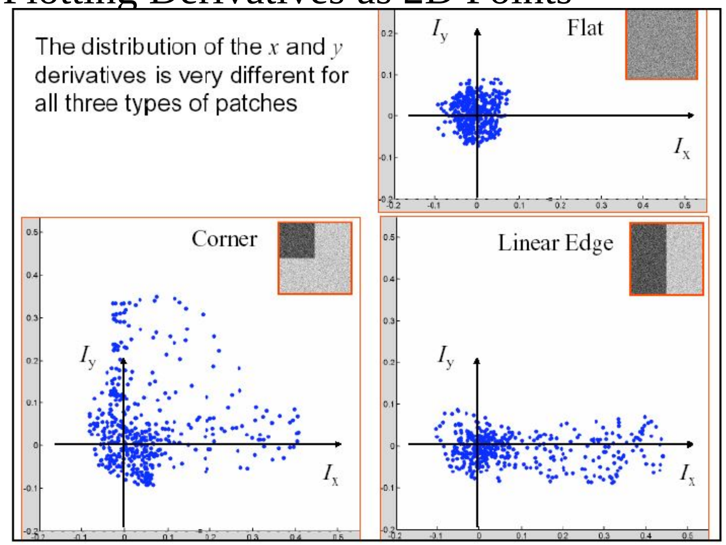
Reference
http://www.cse.psu.edu/~rcollins/CSE486/lecture06.pdf

Addition: Harris Corner Detector (2/3)

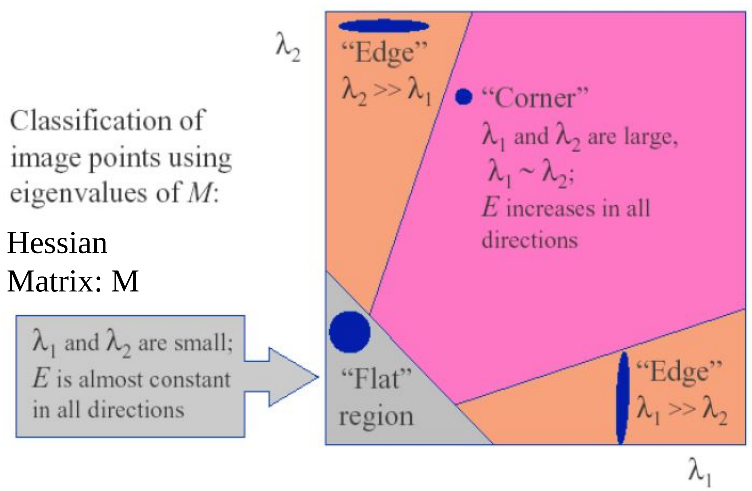
- Example Case:



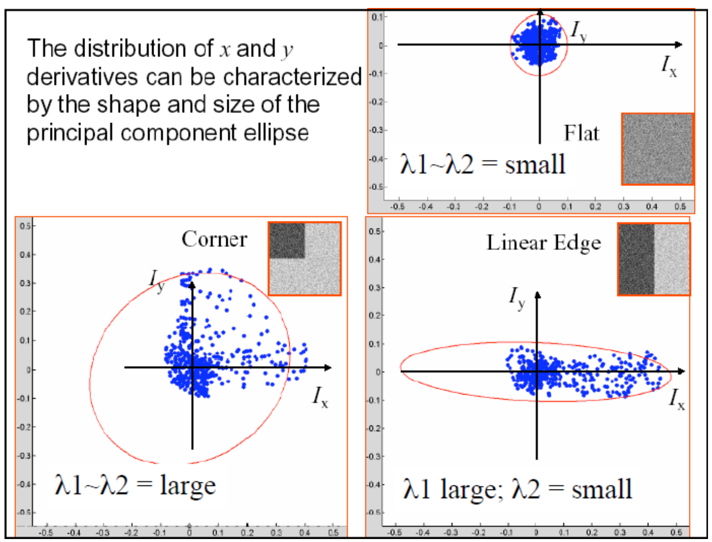
- Plotting Derivatives as 2D Points



- Classification via Eigenvalues



- Fitting Ellipse to each Set of Points



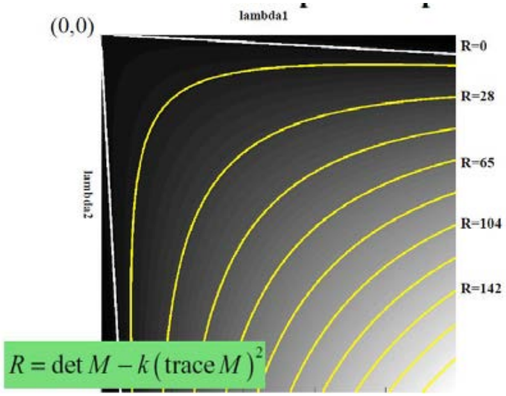
Addition: Harris Corner Detector (3/3)

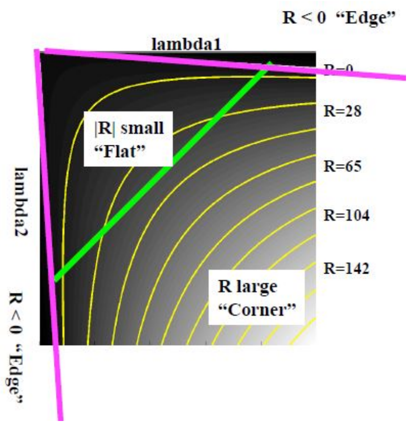
- Measure of corner response:

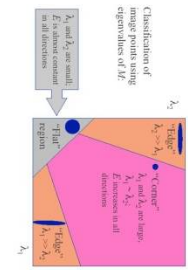
$$R = \det(M) - k\ trace(M)^2 = \lambda_0\lambda_1 - k(\lambda_0 + \lambda_1)^2$$

where k is a determined constant; k = 0.04 - 0.06

- Corner Response Map

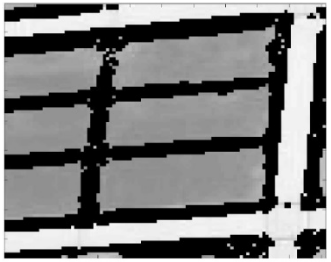




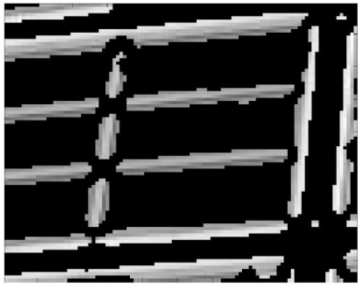


- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region

- Corner Response Example



Threshold: -10000 < R < 10000
(neither edges nor corners)

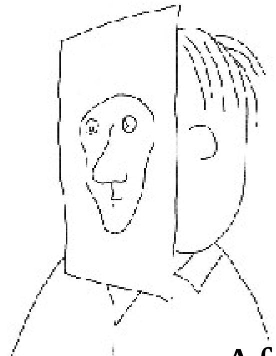


Threshold: R < -10000
(edges)

Threshold: > 10000
(corners)

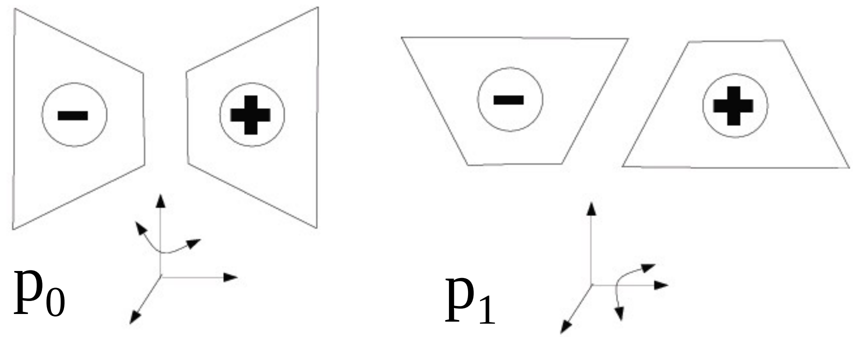
If a Moving Object Is Planar

$$X+Y+Z=k \longrightarrow \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} = \begin{pmatrix} a_0 + a_1 x + a_2 y + p_0 x^2 + p_1 xy \\ a_3 + a_4 x + a_5 y + p_1 xy + p_0 y^2 \end{pmatrix}$$



Perspective Camera $\longrightarrow$

(Pinhole camera)

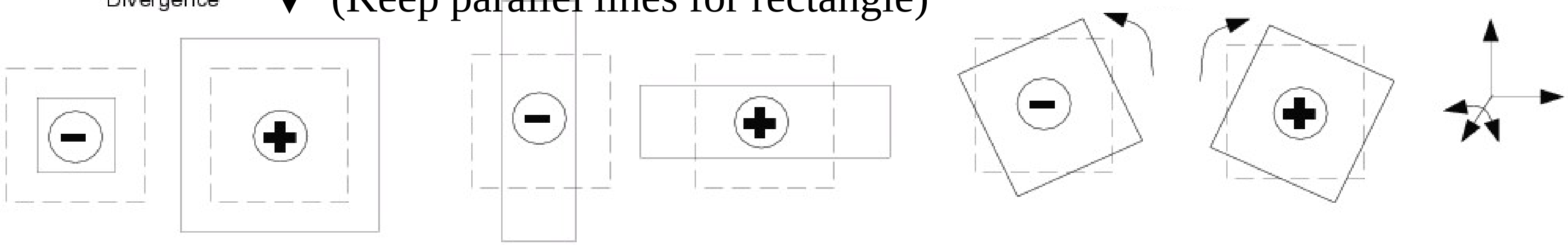


Planar parameters

Affine Camera $\downarrow$

(Keep parallel lines for rectangle)

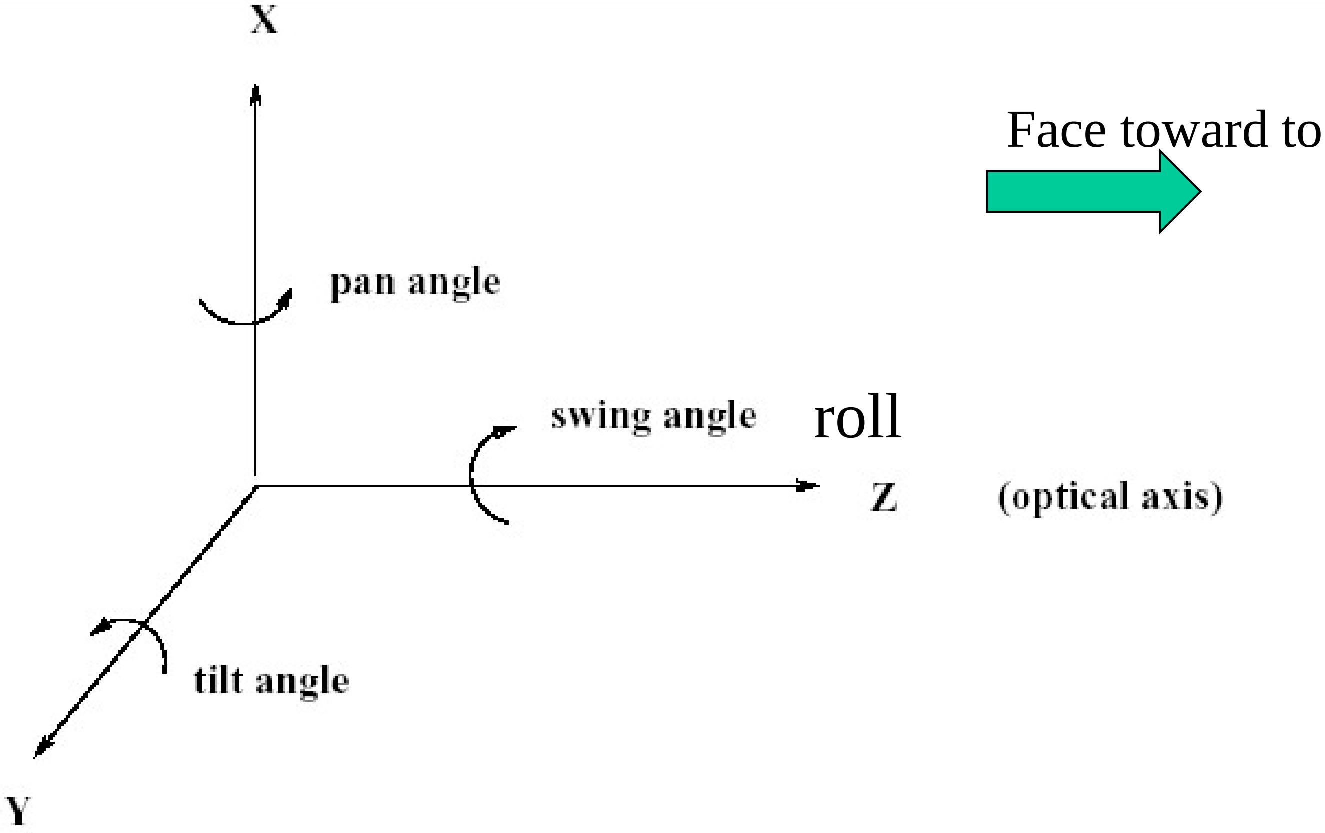
Divergence



$$\text{Divergence}: a_1 + a_5 = u_x + v_v \qquad \text{Curl}: -a_2 - a_4 = -(u_y + v_x)$$

$$\text{Deformation}: a_1 - a_5 = u_x - v_v$$

Rotation Matrix Representation: Euler angles



- **Demonstration: NCTU - car follows people**

Rotation Matrix Representation: Euler angles

Assume rotation matrix R results from successive Euler rotations of the camera frame around its X axis by ω, its once rotated Y axis by ϕ, and its twice rotated Z axis by κ, then

omega, phi, kappa

$$R(\omega, \phi, \ \kappa) = R_X(\omega) R_Y(\phi) R_Z(\kappa)$$

where ω, ϕ, and κ are often referred to as tilt, pan, and swing angles respectively.

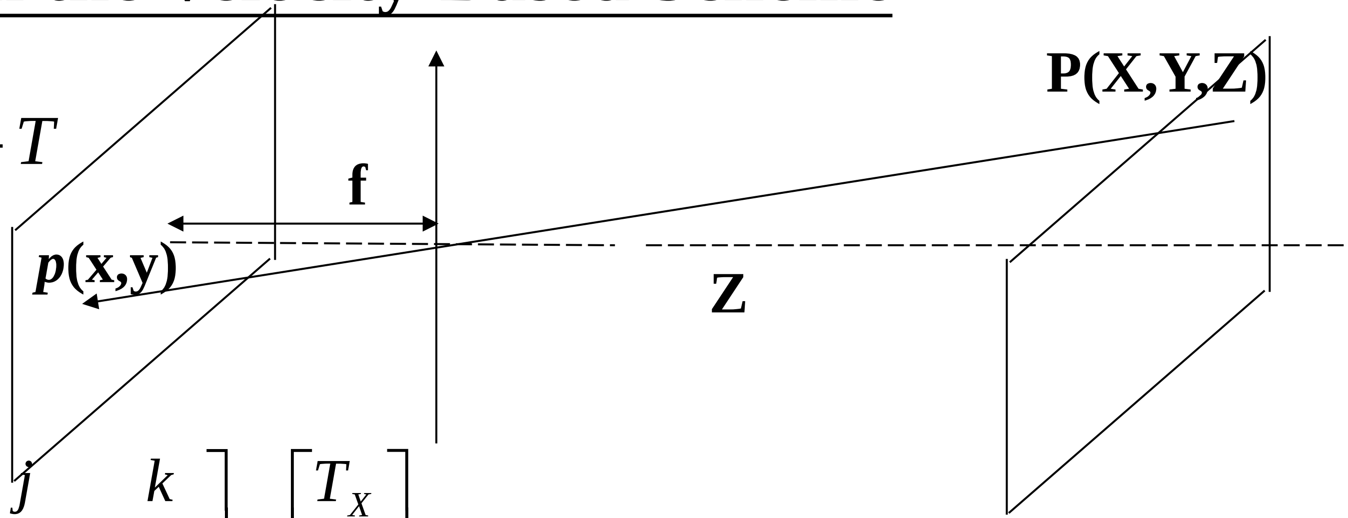
$$R_x(\omega) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & \sin\omega \\ 0 & -\sin\omega & \cos\omega \end{pmatrix}$$

$$R_y(\phi) = \begin{pmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{pmatrix}$$

$$R_z(\kappa) = \begin{pmatrix} \cos\kappa & \sin\kappa & 0 \\ -\sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In the Velocity-Based Scheme

$$\dot{P} = \Omega \times P + T$$



omega

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} i & j & k \\ \Omega_X & \Omega_Y & \Omega_Z \\ X & Y & Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} \Omega_Y Z - \Omega_Z Y + T_X \\ \Omega_Z X - \Omega_X Z + T_Y \\ \Omega_X Y - \Omega_Y X + T_Z \end{bmatrix}$$

$$\left(\begin{array}{c} \dfrac{x}{f=1} = \dfrac{X}{Z} \\ \dfrac{y}{f=1} = \dfrac{Y}{Z} \end{array} \right)$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \dot{x} = \dot{X}/\dot{Z} \\ \dot{y} = \dot{Y}/\dot{Z} \end{pmatrix} = \begin{pmatrix} \frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2} \\ \frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2} \end{pmatrix} = \frac{1}{Z^2}\begin{pmatrix} \dot{X}Z - X\dot{Z} \\ \dot{Y}Z - Y\dot{Z} \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \boxed{-\Omega_X xy + \Omega_Y(1+x^2) - \Omega_Z y} + (T_X - T_Z x)/Z \\ \boxed{-\Omega_X(1+y^2) + \Omega_Y xy + \Omega_Z x} + (T_Y - T_Z y)/Z \end{pmatrix} = \begin{pmatrix} u_R + u_T \\ v_R + v_T \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \left(\Omega_Y + \frac{T_X}{Z}\right) + \left(-\frac{T_Z}{Z}\right)x + (-\Omega_Z)y + \Omega_Y x^2 + (-\Omega_X)xy \\ \left(-\Omega_X + \frac{T_Y}{Z}\right) + \Omega_Z x + \left(-\frac{T_Z}{Z}\right)y + \Omega_Y xy + (-\Omega_X)y^2 \end{pmatrix}$$

X+Y+Z=k

$$\longrightarrow \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} = \begin{pmatrix} a_0 + a_1 x + a_2 y + p_0 x^2 + p_1 xy \\ a_3 + a_4 x + a_5 y + p_1 xy + p_0 y^2 \end{pmatrix}$$

In the Displacement-Based Scheme

- **Convert a rotation vector a=[a_x,a_y,a_z] into a rotation matrix**
- **Skew symmetric matrix**
  - The cross product of two vectors can be written in terms of a skew symmetric matrix

$$a, b \in R^3$$

$$a \times b = \hat{a} b = J(a) b$$

$$\hat{a} = J(a) = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix} \in so(3)$$

Rotation Matrix Representation: Quaternion

$R = (dI + S)(dI - S)^{-1}$ where

$$S = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

$$R = \begin{pmatrix} d^2 + a^2 - b^2 - c^2 & 2(ab - cd) & 2(ac + bd) \\ 2(ab + cd) & d^2 - a^2 + b^2 - c^2 & 2(bc - ad) \\ 2(ac - bd) & 2(bc + ad) & d^2 - a^2 - b^2 + c^2 \end{pmatrix}$$

where $a^2 + b^2 + c^2 + d^2 = 1$ and a, b, c, and d are referred to as the quaternion parameters.

Rotation Matrix Representation: $RR^t = 1$

Prove RR'=1

From the above definition of S, we have

$$
\begin{aligned}
(dI + S)^t &= (dI - S) \\
(dI - S)^t &= (dI + S)
\end{aligned}
$$

As a result, we have

$$
\begin{aligned}
RR^t &= (dI + S)(dI - S)^{-1}[(dI + S)(dI - S)^{-1}]^t \\
&= (dI + S)(dI - S)^{-1}(dI + S)^{-1}(dI - S) \\
&= (dI + S)[(dI + S)(dI - S)]^{-1}(dI - S)
\end{aligned}
$$

$$
\begin{aligned}
&= (dI + S)[(dI + S)(dI - S)]^{-1} \\
&\quad (dI - S)(dI + S)(dI + S)^{-1} \\
&= (dI + S)(dI + S)^{-1} \\
&= I
\end{aligned}
$$

Note $(dI + S)(dI - S) = (dI - S)(dI + S)$

In the Displacement-Based Scheme

In the Velocity-Based Scheme

$$\dot{P} = \Omega \times P + T$$

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} \Omega_Y Z - \Omega_Z Y + T_X \\ \Omega_Z X - \Omega_X Z + T_Y \\ \Omega_X Y - \Omega_Y X + T_Z \end{bmatrix}$$

$$\begin{pmatrix} \frac{x}{f=1} = \frac{X}{Z} \\ \frac{y}{f=1} = \frac{Y}{Z} \end{pmatrix}$$

In the Dispacement-Based Scheme

$$\underbrace{\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}}_{\text{Position at time t}} = R \underbrace{\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{\text{Position at time t-1}} + T = \begin{bmatrix} 1 & -\Omega_Z & \Omega_Y \\ \Omega_Z & 1 & -\Omega_X \\ -\Omega_Y & \Omega_X & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}$$

$$= \begin{bmatrix} \boxed{X} - \Omega_Z Y + \Omega_Y Z + T_X \\ \Omega_Z X + \boxed{Y} - \Omega_X Z + T_Y \\ -\Omega_Y X + \Omega_X Y + \boxed{Z} + T_Z \end{bmatrix} = Z \begin{bmatrix} x - \Omega_Z y + \Omega_Y + \frac{T_X}{Z} \\ \Omega_Z x + y - \Omega_x + \frac{T_Y}{Z} \\ -\Omega_Y x + \Omega_X y + 1 + \frac{T_Z}{Z} \end{bmatrix}$$

C •See Fang and Huang

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \left(\Omega_Y + \frac{T_X}{Z}\right) + \left(-\frac{T_Z}{Z}\right)x + (-\Omega_Z)y + \Omega_Y x^2 + (-\Omega_X)xy \\ \left(-\Omega_X + \frac{T_Y}{Z}\right) + \Omega_Z x + \left(-\frac{T_Z}{Z}\right)y + \Omega_Y xy + (-\Omega_X)y^2 \end{pmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \dfrac{X'}{Z'} \\ \dfrac{Y'}{Z'} \end{bmatrix} = \begin{bmatrix} \dfrac{x - \Omega_Z y + \Omega_Y + T_X / Z}{-\Omega_Y x + \Omega_X y + 1 + T_Z / Z} \\ \dfrac{\Omega_Z x + y - \Omega_X + T_Y / Z}{-\Omega_Y x + \Omega_X y + 1 + T_Z / Z} \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} \dfrac{-\Omega_X xy + \Omega_Y (1 + x^2) - \Omega_Z y + (T_X - T_Z x)/Z}{1 + (\Omega_X y - \Omega_Y x) + T_Z / Z} \\ \dfrac{-\Omega_X (1 + y^2) + \Omega_Y xy + \Omega_Z x + (T_Y - T_Z y)/Z}{1 + (\Omega_X y - \Omega_Y x) + T_Z / Z} \end{bmatrix}$$

Flow from
t-1 to t over
delta_t=1

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \left(\Omega_Y + \frac{T_X}{Z}\right) + \left(-\frac{T_Z}{Z}\right)x + (-\Omega_Z)y + \Omega_Y x^2 + (-\Omega_X)xy \\ \left(-\Omega_X + \frac{T_Y}{Z}\right) + \Omega_Z x + \left(-\frac{T_Z}{Z}\right)y + \Omega_Y xy + (-\Omega_X)y^2 \end{pmatrix}$$

Motion: Displacement Field = Velocity Field ?

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -\Omega_X xy + \Omega_Y(1+x^2) - \Omega_Z y + (T_X - T_Z x)/Z \\ -\Omega_X(1+y^2) + \Omega_Y xy + \Omega_Z x + (T_Y - T_Z y)/Z \end{pmatrix} = \begin{pmatrix} u_R + u_T \\ v_R + v_T \end{pmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x' - x \\ y' - y \end{bmatrix} = \begin{bmatrix} \dfrac{-\Omega_X xy + \Omega_Y(1+x^2) - \Omega_Z y + (T_X - T_Z x)/Z}{\boxed{1 + (\Omega_X y - \Omega_Y x) + T_Z/Z}} \\ \dfrac{-\Omega_X(1+y^2) + \Omega_Y xy + \Omega_Z x + (T_Y - T_Z y)/Z}{\boxed{1 + (\Omega_X y - \Omega_Y x) + T_Z/Z}} \end{bmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \left(\Omega_Y + \frac{T_X}{Z}\right) + \left(-\frac{T_Z}{Z}\right)x + (-\Omega_Z)y + \Omega_Y x^2 + (-\Omega_X)xy \\ \left(-\Omega_X + \frac{T_Y}{Z}\right) + \Omega_Z x + \left(-\frac{T_Z}{Z}\right)y + \Omega_Y xy + (-\Omega_X)y^2 \end{pmatrix}$$

$$\begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} = \begin{pmatrix} a_0 + a_1 x + a_2 y + p_0 x^2 + p_1 xy \\ a_3 + a_4 x + a_5 y + p_1 xy + p_0 y^2 \end{pmatrix}$$

- **Optical flow: Displacement field = Velocity field ⇔**
  - If the time interval between two image frames is short enough (high sampling rate) or
  - the motion of the object is slow compared with frame rate
    - » $Z >> T_z$
    - » If small rotation parameters $\Omega_X, \Omega_Y, \Omega_Z$

2D (Planar) Motions (Transformation) (1/11)

6) 8-Parameter planar transformation Vs. perspective projection transformation

Motion flow

$$u(x,y) = a_0 + a_1 x + a_2 y + p_0 x^2 + p_1 xy$$
$$v(x,y) = a_3 + a_4 x + a_5 y + p_0 xy + p_1 y^2$$

Motion flow

$$u(x,y) = a_0 + a_1 x + a_2 y \qquad (8)$$
$$v(x,y) = a_3 + a_4 x + a_5 y + cx^2 \qquad (9)$$

$$\text{divergence} = a_1 + a_5 = (u_x + v_y), \qquad (3)$$
$$\text{curl} = -a_2 + a_4 = -(u_y - v_x), \qquad (4)$$
$$\text{deformation} = a_1 - a_5 = (u_x - v_y) \qquad (5)$$

Yaw = p_0,
Pitch = p_1 .

a_i: Constants.
$u(x) = [u(x,y), v(x,y)]^T$: Horizontal and vertical components of the flow at the image point x = (x,y).

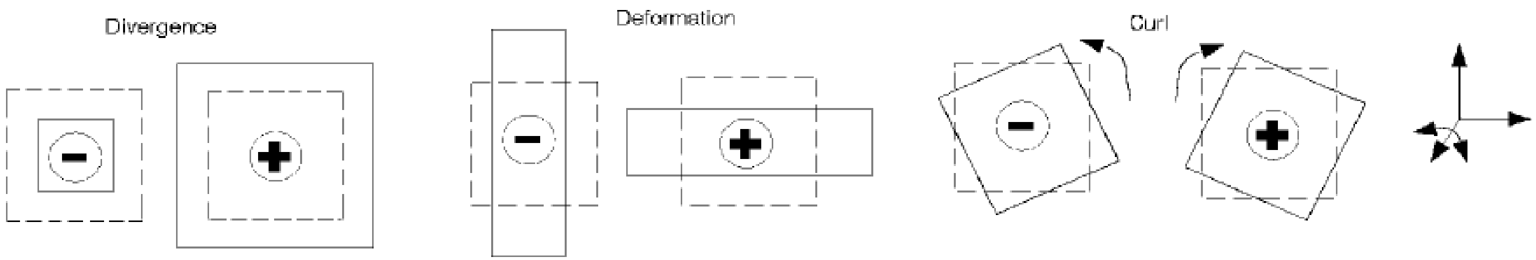


Figure 3. The figure illustrates the motion captured by the various parameters used to represent the motion of the regions. The solid lines indicate the deformed image region and the "−" and "+" indicate the sign of the quantity.

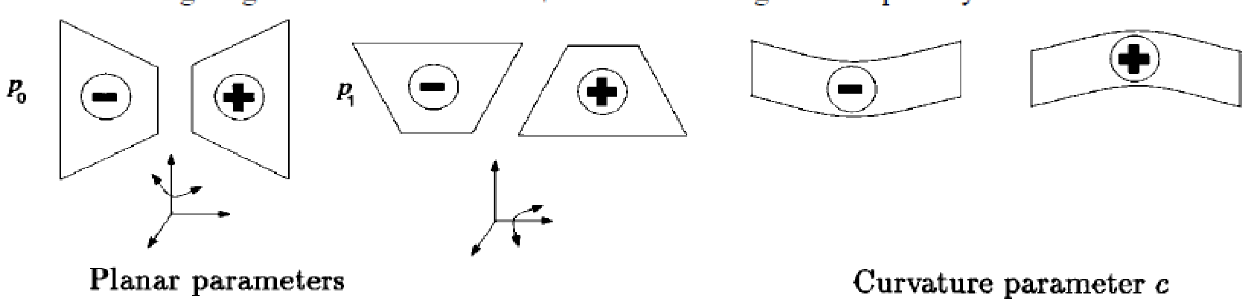


Figure 4. Additional parameters for planar motion and curvature.

M. Black and Y. Yaccob, "Recognizing Facial Expressions in Image Sequences Using Local Parameterized Models of Image Motion", IJCV1997.

2D (Planar) Motions (Transformation) (2/11)

$$\mathbf{X}(\mathbf{x}) = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 & x^2 & xy & 0 \\ 0 & 0 & 0 & 1 & x & y & xy & y^2 & x^2 \end{bmatrix} \tag{10}$$

$$\mathbf{A} = [\, a_0 \;\; a_1 \;\; a_2 \;\; a_3 \;\; a_4 \;\; a_5 \;\; 0 \;\; 0 \;\; 0 \,]^T \tag{11}$$

$$\mathbf{P} = [\, a_0 \;\; a_1 \;\; a_2 \;\; a_3 \;\; a_4 \;\; a_5 \;\; p_0 \;\; p_1 \;\; 0 \,]^T \tag{12}$$

$$\mathbf{C} = [\, a_0 \;\; a_1 \;\; a_2 \;\; a_3 \;\; a_4 \;\; a_5 \;\; 0 \;\; 0 \;\; c \,]^T \tag{13}$$

such that $\mathbf{u}(\mathbf{x};\mathbf{A}) = \mathbf{X}(\mathbf{x})\mathbf{A}$, $\mathbf{u}(\mathbf{x};\mathbf{P}) = \mathbf{X}(\mathbf{x})\mathbf{P}$, and $\mathbf{u}(\mathbf{x};\mathbf{C}) = \mathbf{X}(\mathbf{x})\mathbf{C}$ represent, respectively, the affine, planar, and affine + curvature flow models described above.

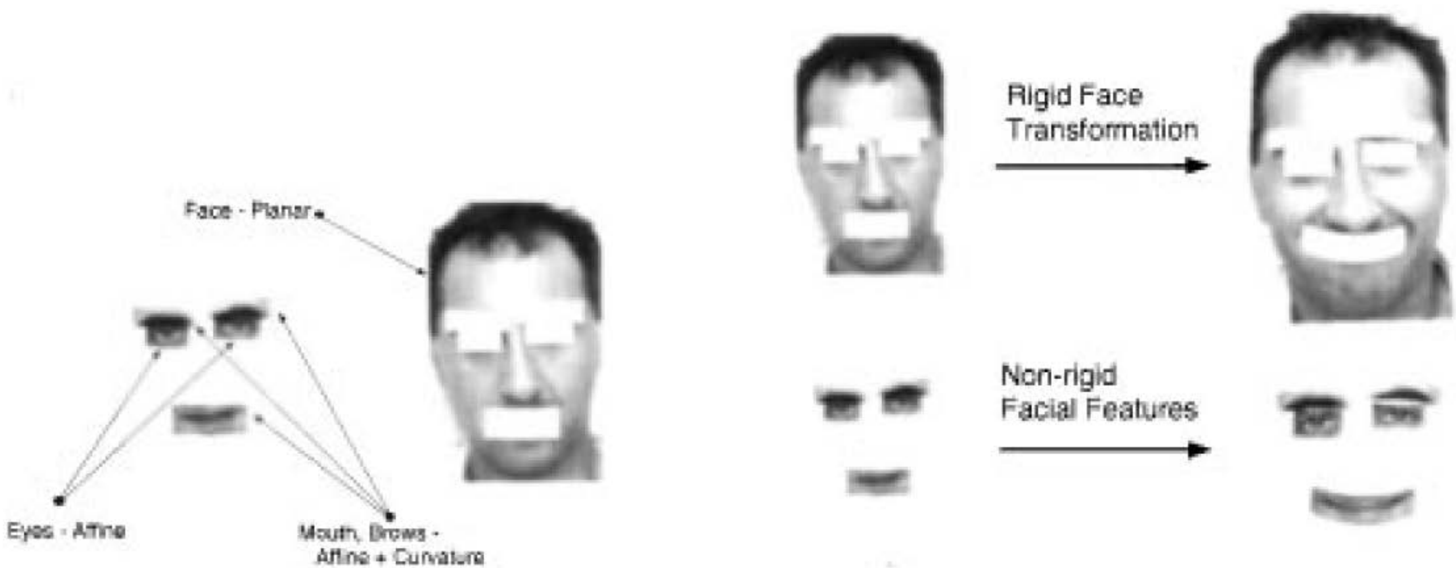


Figure 1. Illustration showing the parametric motion models employed and an example of a face undergoing a looming motion while smiling.

Table 1. The mid-level predicates derived from deformation and motion parameter estimates.

Parameter	Threshold	Derived predicates (mouth)
a_0	>0.25	Rightward
	<-0.25	Leftward
a_3	<-0.1	Upward
	>0.1	Downward
Div	>0.02	Expansion
	<-0.02	Contraction
Def	>0.005	Horizontal deformation
	<-0.005	Vertical deformation
Curl	>0.005	Clockwise rotation
	<-0.005	Counter clockwise rotation
c	<-0.0001	Curving upward ('U' like)
	>0.0001	Curving downward

Table 2. The mid-level predicates derived from deformation and motion parameter estimates as applied to head motion.

Parameter	Threshold	Derived predicates (head)
a_0	>0.5	Rightward
	<-0.5	Leftward
a_3	<-0.5	Upward
	>0.5	Downward
Div	>0.01	Expansion
	<-0.01	Contraction
Def	>0.01	Horizontal deformation
	<-0.01	Vertical deformation
Curl	>0.005	Clockwise rotation
	<-0.005	Counter clockwise rotation
p_0	<-0.00005	Rotate right about neck
	>0.00005	Rotate left about neck
p_1	<-0.00005	Rotate forward
	>0.00005	Rotate backward

Table 3. The rules for classifying facial expressions (B = beginning, E = ending).

Expr.	B/E	Satisfactory actions
Anger	B	Inward lowering of brows and mouth contraction
Anger	E	Outward raising of brows and mouth expansion
Disgust	B	Mouth horizontal expansion and lowering of brows
Disgust	E	Mouth contraction and raising of brows
Happiness	B	Upward curving of mouth and expansion or horizontal deformation
Happiness	E	Downward curving of mouth and contraction or horizontal deformation
Surprise	B	Raising brows and vertical expansion of mouth
Surprise	E	Lowering brows and vertical contraction of mouth
Sadness	B	Downward curving of mouth and upward-inward motion in inner parts of brows
Sadness	E	Upward curving of mouth and downward-outward motion in inner parts of brows
Fear	B	Expansion of mouth and raising-inwards inner parts of brows
Fear	E	Contraction of mouth and lowering inner parts of brows

2D (Planar) Motions (Transformation) (4/11)

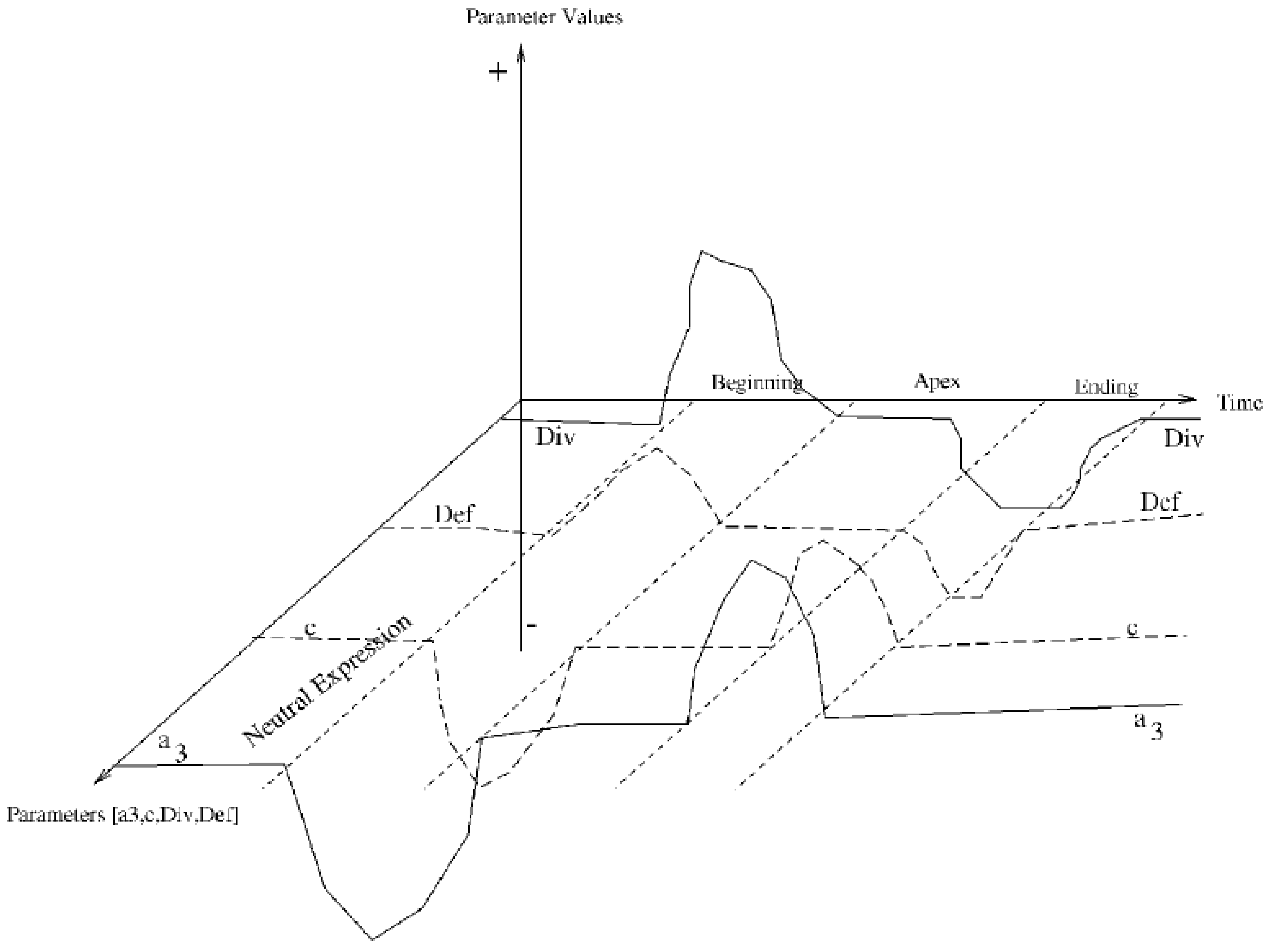


Figure 5. The temporal model of the "smile" expression.

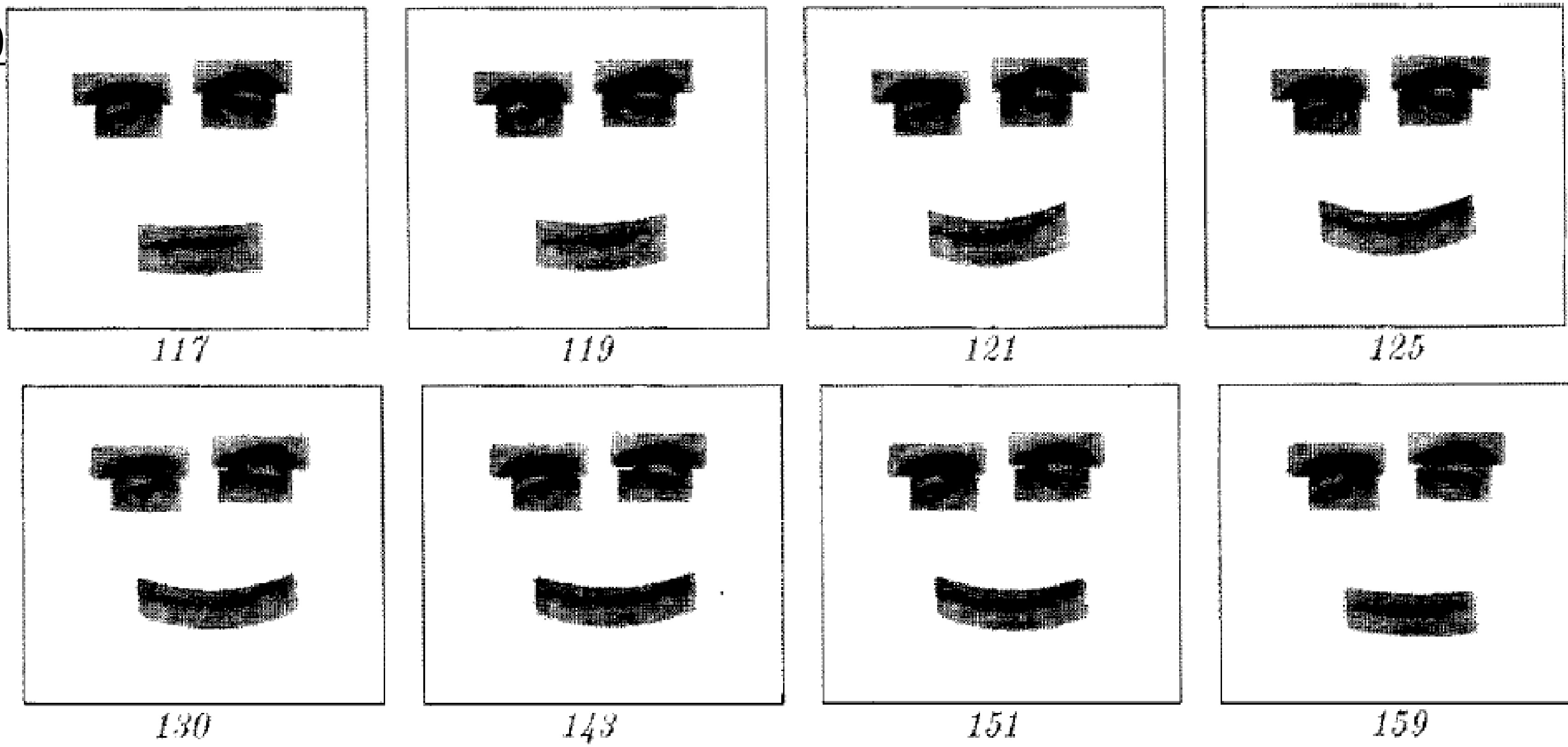


Figure 6. Smile experiment: facial expression tracking.

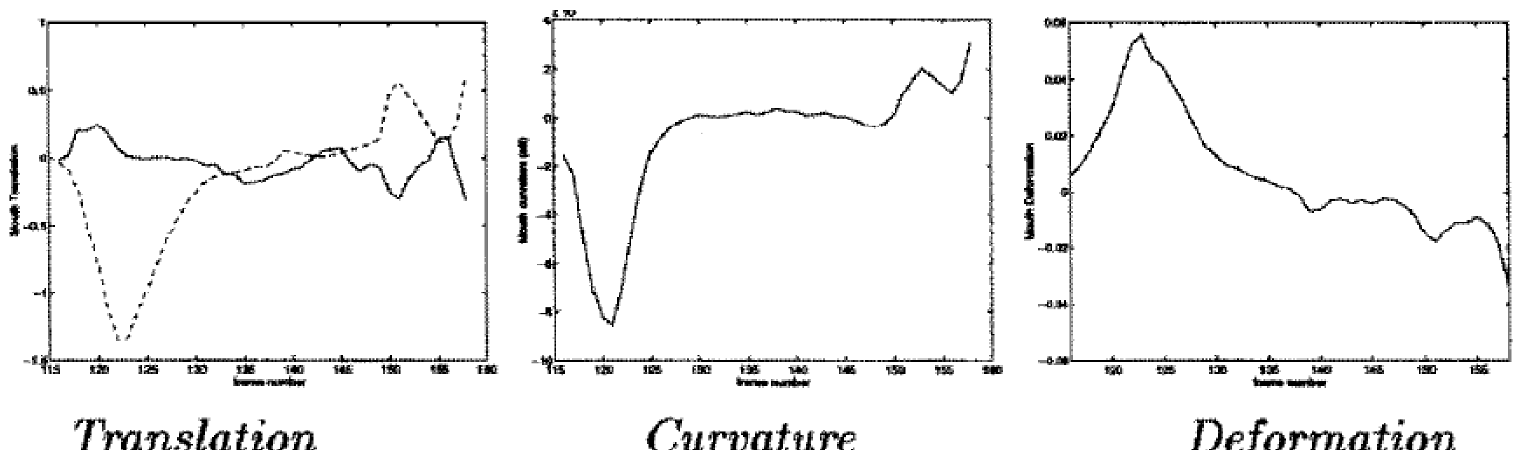


Figure 7. The smile mouth parameters. For translation, solid and dashed lines indicate horizontal and vertical motion respectively.

2D (Planar) Motions (Transformation) (6/11)

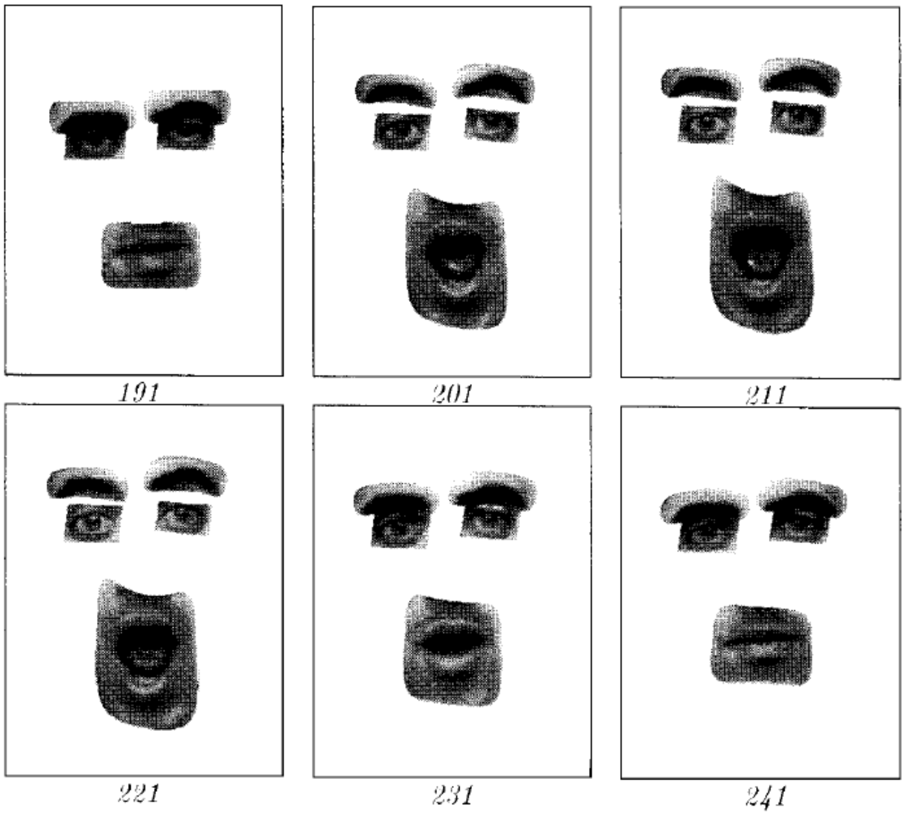


Figure 10. Surprise experiment: facial expression tracking. Features every 10 frames.

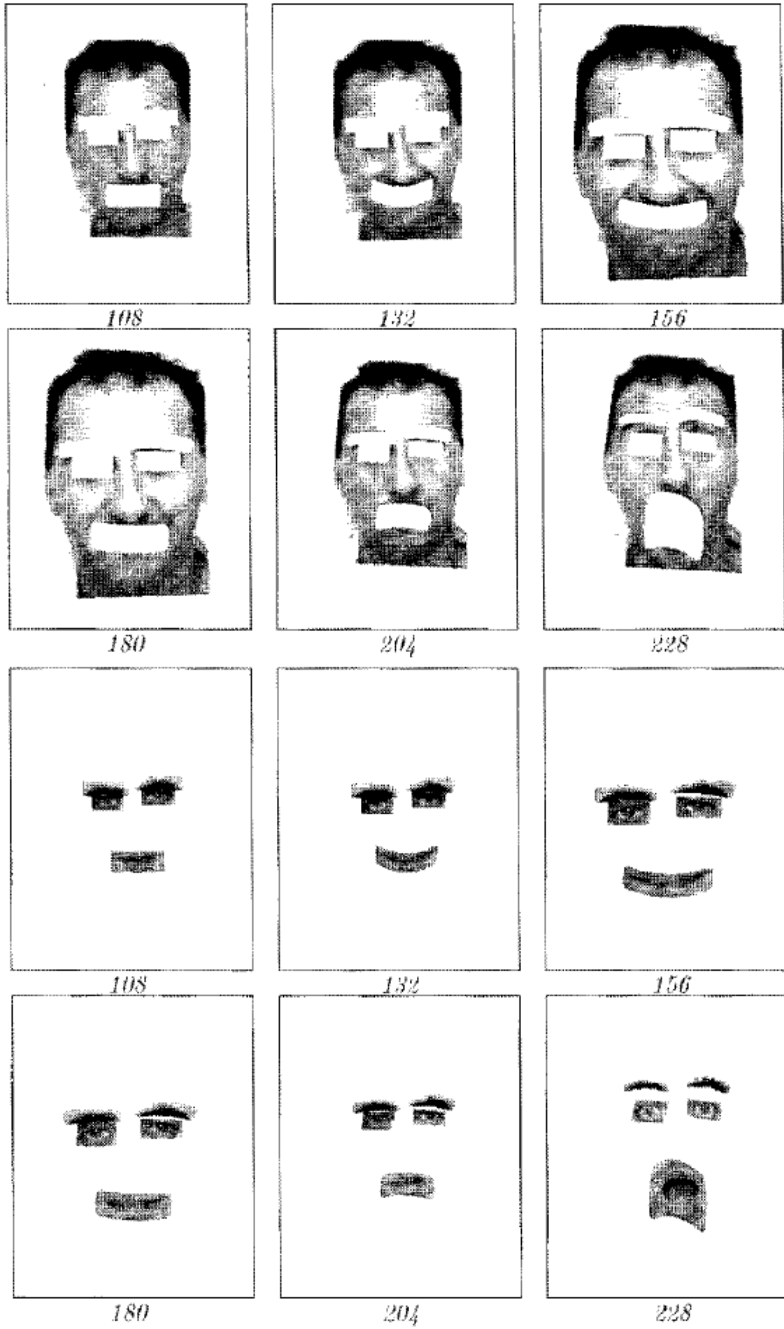


Figure 14. Looming experiment. Facial expression tracking with rigid head motion (every 24 frames).

2D (Planar) Motions (Transformation) (7/11)

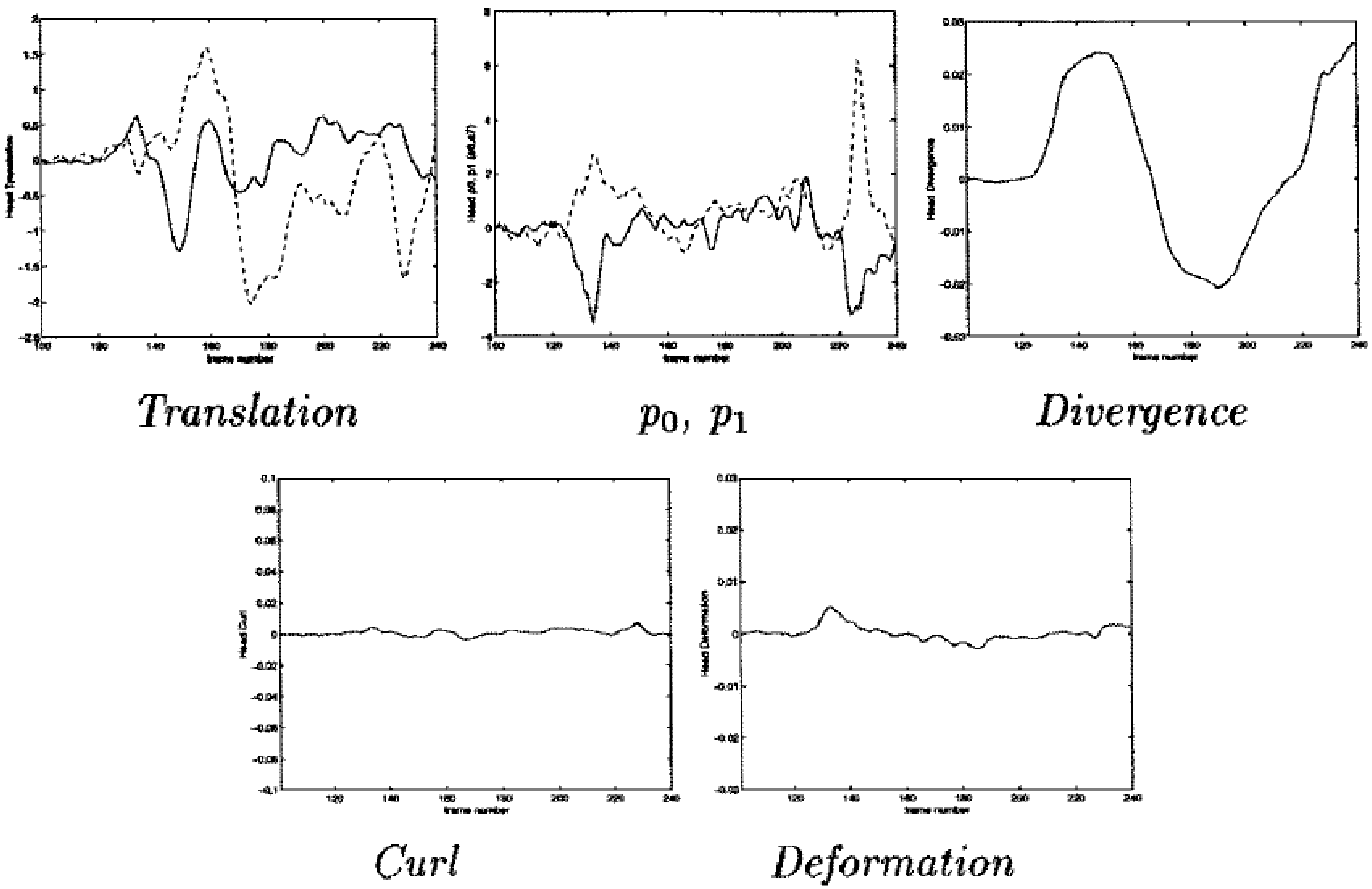


Figure 15. The looming face motion parameters. Translation: solid = horizontal, dashed = vertical. Quadratic terms: solid = p_0, dashed = p_1.

2D (Planar) Motions (Transformation) (8/11)

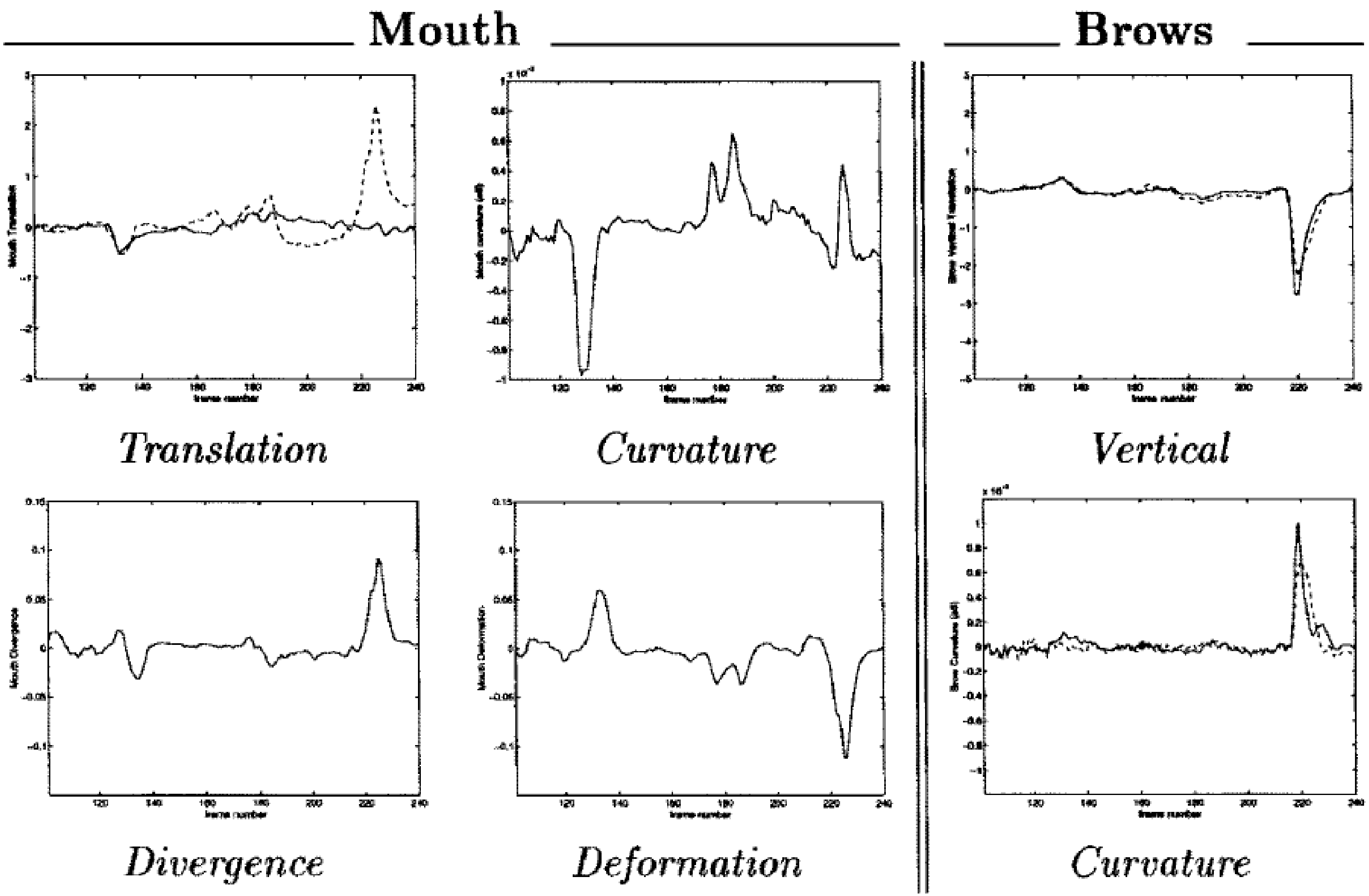


Figure 16. The looming sequence. Mouth translation: solid and dashed lines indicate horizontal and vertical motion respectively. For the brows, the solid and dashed lines indicate left and right brows respectively.

Model

2D (Planar) Motions (Transformation) (9/11)

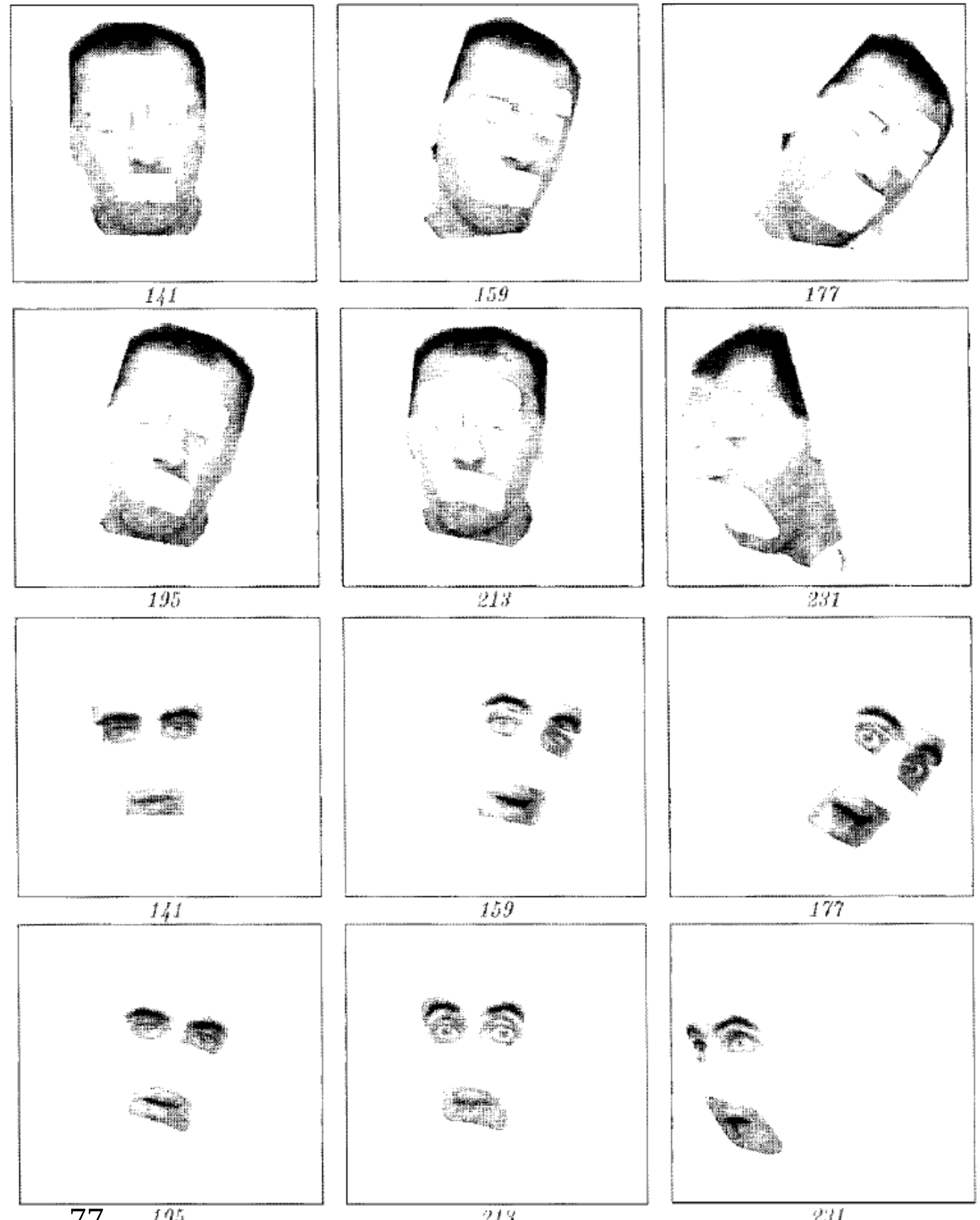


Figure 17. Rotation experiment. Rigid head tracking, every 18th frame.

2D (Planar) Motions (Transformation) (10/11)

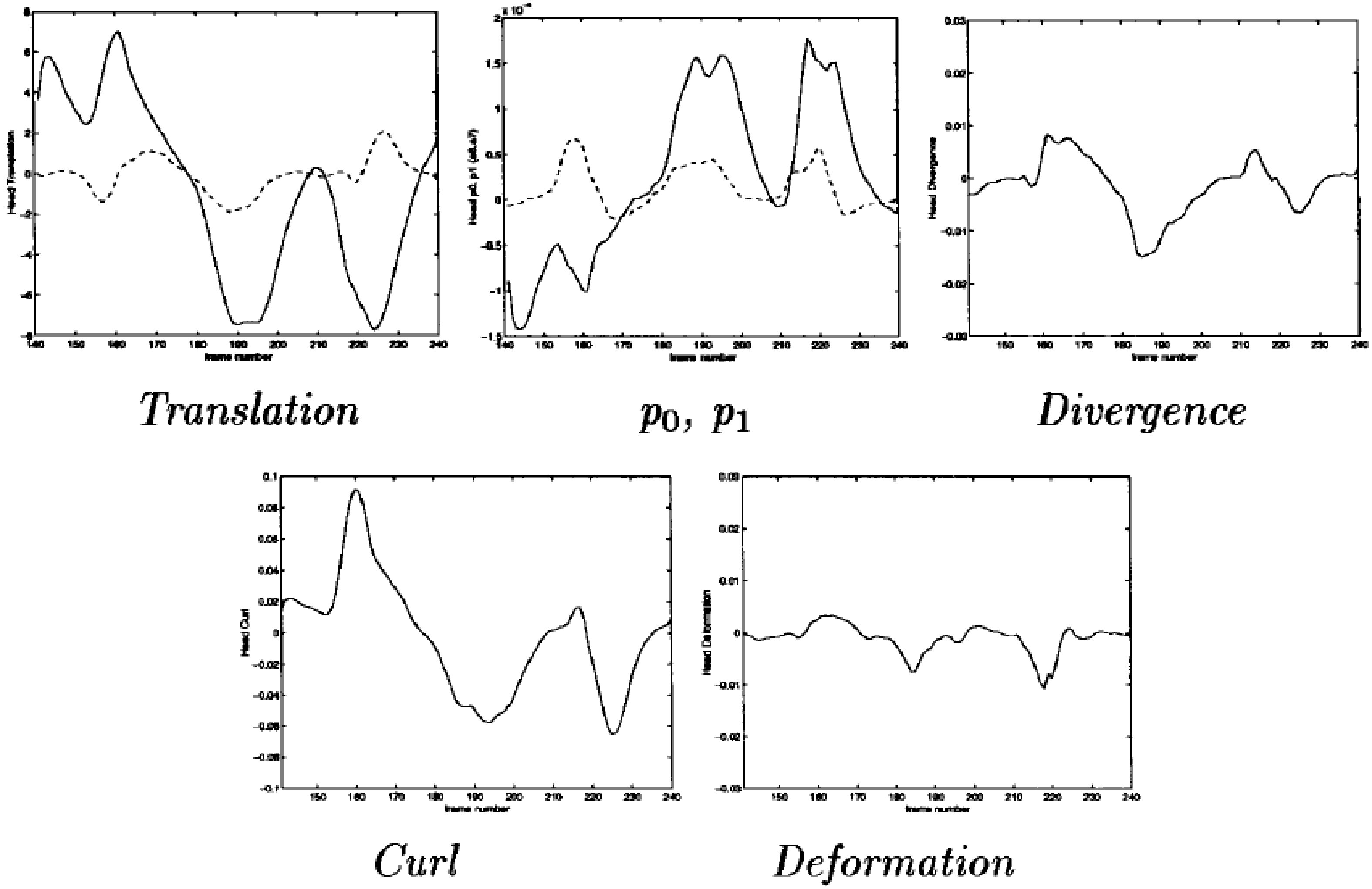


Figure 18. The rotate sequence face motion parameters. Translation: solid = horizontal, dashed = vertical. Quadratic terms: solid = p_0, dashed = p_1.

2D (Planar) Motions (Transformation) (11/11)

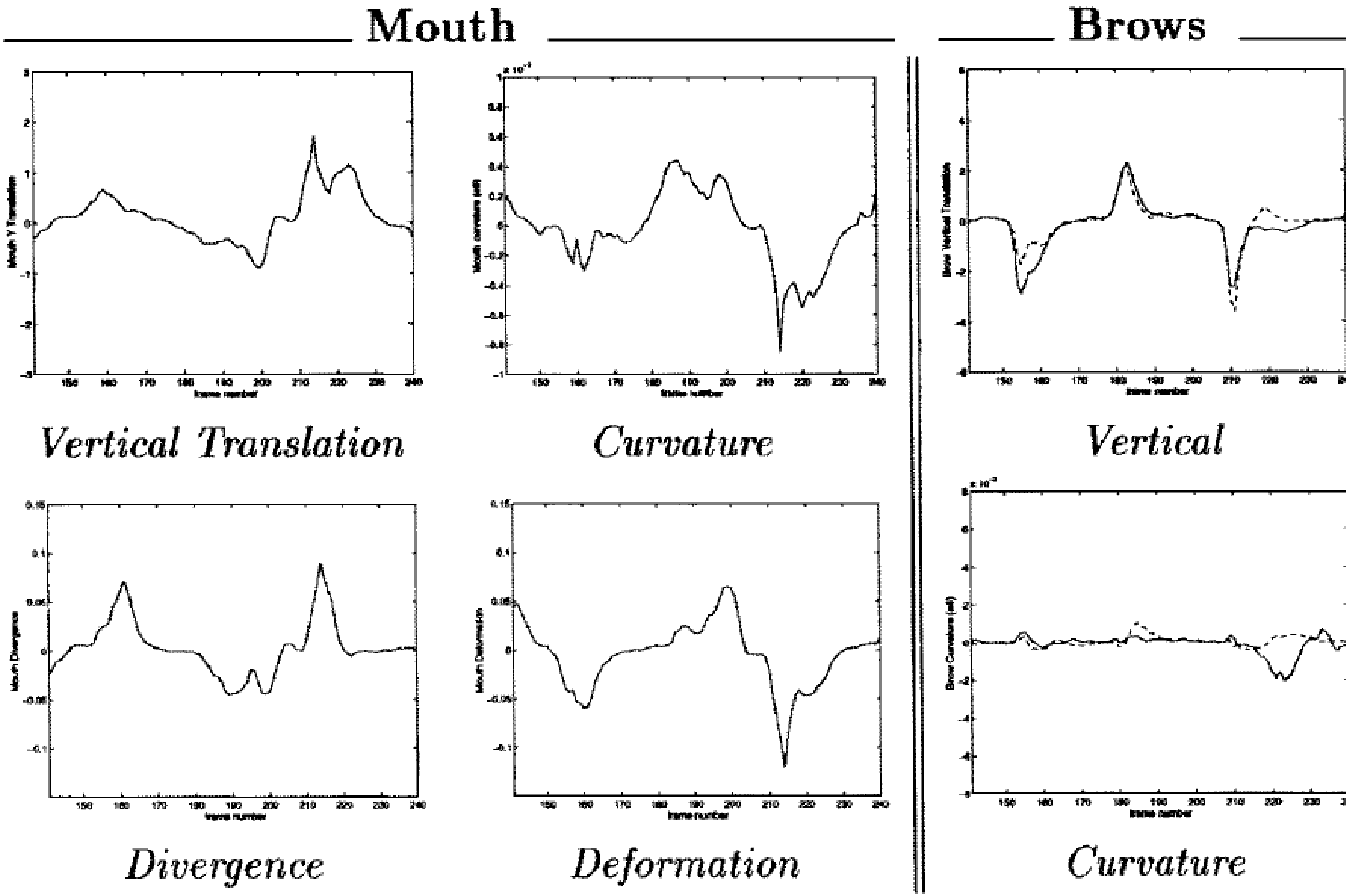


Figure 19. The rotate sequence. For the brows, the solid and dashed lines indicate left and right brows respectively.

2D (Planar) Motions (Transformation) jj

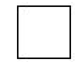
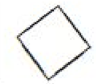
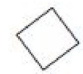
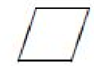
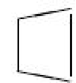
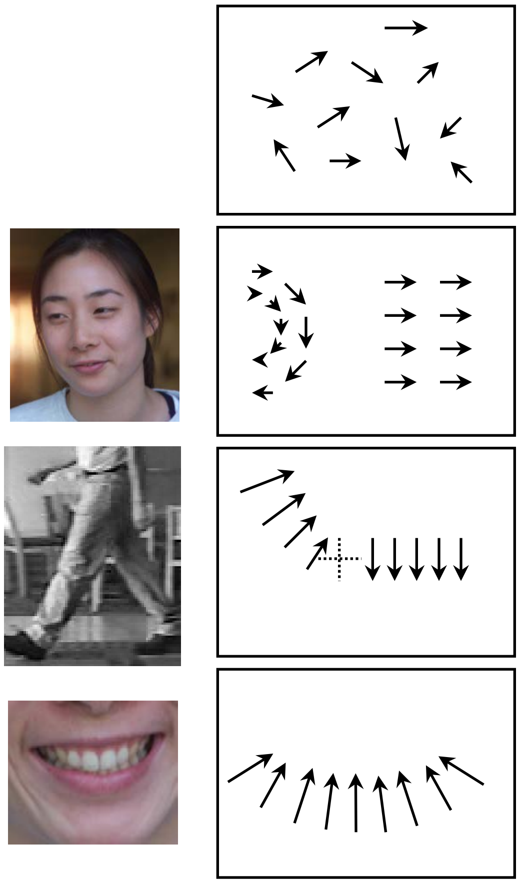
Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} I \mid t \end{bmatrix}_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\begin{bmatrix} R \mid t \end{bmatrix}_{2\times 3}$	3	lengths $+\cdots$	
similarity **S_x=S_y**	$\begin{bmatrix} sR \mid t \end{bmatrix}_{2\times 3}$	4	angles $+\cdots$	
affine **S_x!= S_y**	$\begin{bmatrix} A \end{bmatrix}_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times 3}$	8	straight lines	

Table 1: *Hierarchy of 2D coordinate transformations. The 2×3 matrices are extended with a third $[0^T\ 1]$ row to form a full 3×3 matrix for homogeneous coordinate transformations.*

#D.O.F : Degrees Of Freedom

1) **Translation :** t_x, t_y
2) **Euclidean :** t_x, t_y, θ
3) **Similarity :** t_x, t_y, θ, s
4) **Affine :** $a_{00}, a_{01}, a_{02}, a_{10}, a_{11}, a_{12}$
5) **Projective :** $h'_{00}, h'_{01}, h'_{02}, h'_{10}, h'_{11}, h'_{12}, h'_{20}, h'_{21}$
6) **8-Parameter planar transformation: a0, a1, a2, a3, a4, a5, p0 and p1**

Kinematic Models



- Optical Flow/Feature tracking: no constraints
- Layered Motion: rigid constraints
- Articulated: kinematic chain constraints
- Nonrigid: implicit / learned constraints

References

- **G. Adiv. "Determining Three-Dimensional Motion and Structure from Optical Flow Generated by Several Moving Objects," IEEE Transaction on Pattern Analysis and Machine Intelligence, 7(4):384-401, July 1985.**
- **J.Q Fang and T.S. Huang, "Solving Three Dimensional Small-Rotation Motion Equations," IEEE Conference on Computer Vision and Pattern Recognition, pp. 253-258, 1983.**
- **B.K.P.Horn, Robot Vision, The MIT Press, 1989.**
- **B.D. Lucas, "Generalized Image Matching by the Method of Differences," Carnegie Mellon University, Technical Report CMU-CS-85-160, Ph.D. dissertation, July 1984.**
- **J. Shi and C. Tomasi, "Good Feature to Track," IEEE Conference on Computer Vision and Pattern Recognition, pages 593-600, 1994.**

- **Source Code:**
  - http://vision.stanford.edu/~birch/klt/