

Digital Image

(B4M33DZO, Summer 2024)

Lecture 9:

Image Registration 2

<https://cw.fel.cvut.cz/wiki/courses/b4m33dzo/start>

Daniel Sýkora & Ondřej Drbohlav

Department of Cybernetics

Faculty of Electrical Engineering

Czech Technical University in Prague

© Daniel Sýkora & Ondřej Drbohlav, 2024



Polar transformation:

rotation \implies **translation on y-axis**

Polar transformation:

rotation \implies **translation on y-axis**

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$$

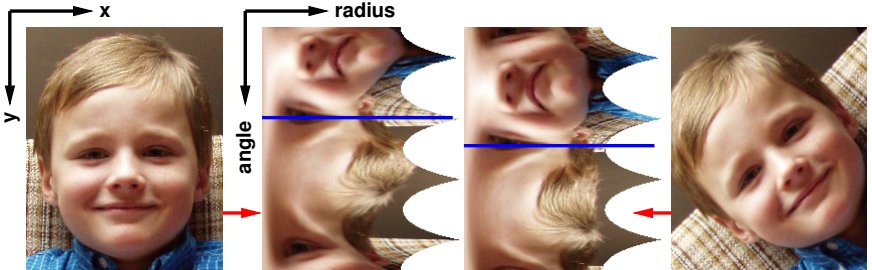
$$\alpha = \arctan \frac{y - y_c}{x - x_c}$$

Polar transformation:

rotation \implies translation on y-axis

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$$

$$\alpha = \arctan \frac{y - y_c}{x - x_c}$$



Log-polar transformation:

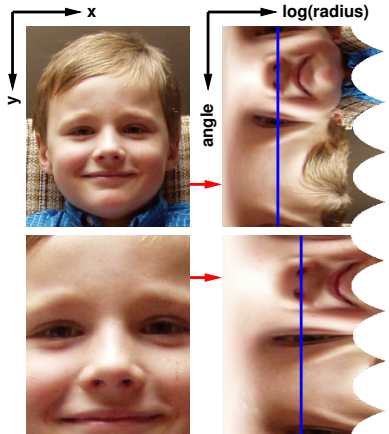
scale \implies **translation on x-axis**
rotation \implies **translation on y-axis**

Log-polar transformation:

scale \implies translation on x-axis
 rotation \implies translation on y-axis

$$r = \log \left(1 + \sqrt{(x - x_c)^2 + (y - y_c)^2} \right)$$

$$\alpha = \arctan \frac{y - y_c}{x - x_c}$$

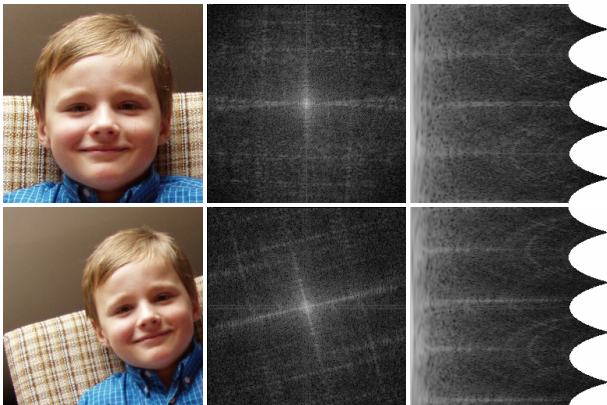


Fourier-Mellin transformation:

1. Estimate rotation and scale using frequency domain.
2. Resample image to have the same rotation and scale.
3. Use standard phase correlation to estimate translation.

Fourier-Mellin transformation:

1. Estimate rotation and scale using frequency domain.
2. Resample image to have the same rotation and scale.
3. Use standard phase correlation to estimate translation.



Generalized gradient descent:

$$E = \sum_i (\mathbf{A}[\mathbf{W}(\mathbf{x}_i, \mathbf{p})] - \mathbf{B}[\mathbf{x}_i])^2 \approx \sum_i (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i)^2$$

Generalized gradient descent:

$$E = \sum_i (\mathbf{A}[\mathbf{W}(\mathbf{x}_i, \mathbf{p})] - \mathbf{B}[\mathbf{x}_i])^2 \approx \sum_i (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i)^2$$
$$\mathbf{A}_i \rightarrow \mathbf{A}[\mathbf{x}_i] \quad \mathbf{B}_i \rightarrow \mathbf{B}[\mathbf{x}_i] \quad \mathbf{A}'_i \rightarrow \frac{\partial}{\partial \mathbf{x}} \mathbf{A}_i$$

Generalized gradient descent:

$$E = \sum_i (\mathbf{A}[\mathbf{W}(\mathbf{x}_i, \mathbf{p})] - \mathbf{B}[\mathbf{x}_i])^2 \approx \sum_i (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i)^2$$

$$\mathbf{A}_i \rightarrow \mathbf{A}[\mathbf{x}_i] \quad \mathbf{B}_i \rightarrow \mathbf{B}[\mathbf{x}_i] \quad \mathbf{A}'_i \rightarrow \frac{\partial}{\partial \mathbf{x}} \mathbf{A}_i$$

$$E' = 2 \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i) = 0$$

Generalized gradient descent:

$$E = \sum_i (\mathbf{A}[\mathbf{W}(\mathbf{x}_i, \mathbf{p})] - \mathbf{B}[\mathbf{x}_i])^2 \approx \sum_i (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i)^2$$

$$\mathbf{A}_i \rightarrow \mathbf{A}[\mathbf{x}_i] \quad \mathbf{B}_i \rightarrow \mathbf{B}[\mathbf{x}_i] \quad \mathbf{A}'_i \rightarrow \frac{\partial}{\partial \mathbf{x}} \mathbf{A}_i$$

$$E' = 2 \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i) = 0$$

$$\mathbf{p} = \mathbf{H}^{-1} \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{B}_i - \mathbf{A}_i)$$

Generalized gradient descent:

$$E = \sum_i (\mathbf{A}[\mathbf{W}(\mathbf{x}_i, \mathbf{p})] - \mathbf{B}[\mathbf{x}_i])^2 \approx \sum_i (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i)^2$$

$$\mathbf{A}_i \rightarrow \mathbf{A}[\mathbf{x}_i] \quad \mathbf{B}_i \rightarrow \mathbf{B}[\mathbf{x}_i] \quad \mathbf{A}'_i \rightarrow \frac{\partial}{\partial \mathbf{x}} \mathbf{A}_i$$

$$E' = 2 \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{A}_i + \mathbf{A}'_i \mathbf{J} \mathbf{p} - \mathbf{B}_i) = 0$$

$$\mathbf{p} = \mathbf{H}^{-1} \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{B}_i - \mathbf{A}_i)$$

$$\mathbf{J} = \underbrace{\begin{pmatrix} \frac{\partial \mathbf{W}_x}{\partial \mathbf{p}_1} & \frac{\partial \mathbf{W}_x}{\partial \mathbf{p}_2} & \cdots & \frac{\partial \mathbf{W}_x}{\partial \mathbf{p}_n} \\ \frac{\partial \mathbf{W}_y}{\partial \mathbf{p}_1} & \frac{\partial \mathbf{W}_y}{\partial \mathbf{p}_2} & \cdots & \frac{\partial \mathbf{W}_y}{\partial \mathbf{p}_n} \end{pmatrix}}_{\text{Jacobian}}$$

$$\mathbf{H} = \sum_i (\mathbf{A}'_i \mathbf{J})^T (\mathbf{A}'_i \mathbf{J})$$

Jacobian

Jacobian

Affine transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0)$

$$x' = a_{11}x + a_{12}y + x_0 \quad y' = a_{21}x + a_{22}y + y_0$$

Jacobian

Affine transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0)$

$$x' = a_{11}x + a_{12}y + x_0 \quad y' = a_{21}x + a_{22}y + y_0$$

$$\mathbf{J} = \begin{pmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{pmatrix}$$

Jacobian

Affine transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0)$

$$x' = a_{11}x + a_{12}y + x_0 \quad y' = a_{21}x + a_{22}y + y_0$$

$$\mathbf{J} = \begin{pmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{pmatrix}$$

Projective transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0, f_1, f_2)$

$$x' = \frac{a_{11}x + a_{12}y + x_0}{f_1x + f_2y + 1} \quad y' = \frac{a_{21}x + a_{22}y + y_0}{f_1x + f_2y + 1}$$

Jacobian

Affine transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0)$

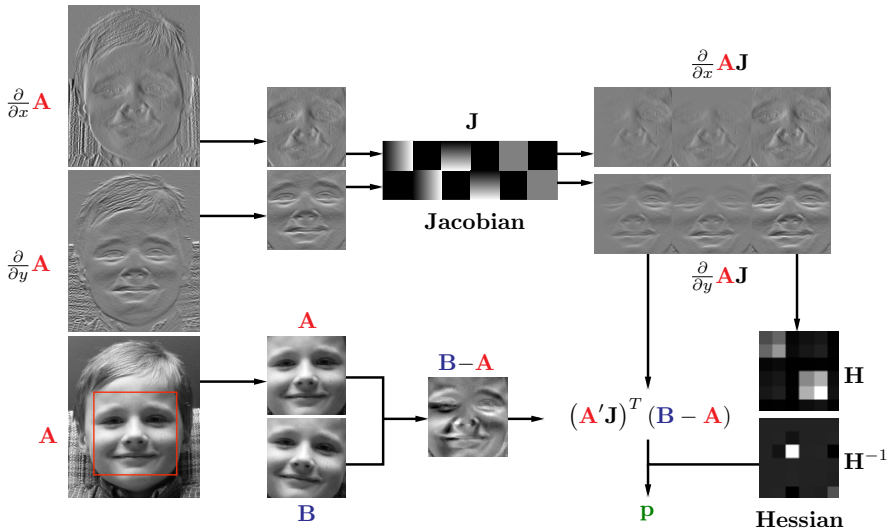
$$x' = a_{11}x + a_{12}y + x_0 \quad y' = a_{21}x + a_{22}y + y_0$$

$$\mathbf{J} = \begin{pmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{pmatrix}$$

Projective transformation: $\mathbf{p} = (a_{11}, a_{12}, a_{21}, a_{22}, x_0, y_0, f_1, f_2)$

$$x' = \frac{a_{11}x + a_{12}y + x_0}{f_1x + f_2y + 1} \quad y' = \frac{a_{21}x + a_{22}y + y_0}{f_1x + f_2y + 1}$$

$$\mathbf{J} = \frac{1}{f_1x + f_2y + 1} \begin{pmatrix} x & y & 0 & 0 & 1 & 0 & -xx' & -yx' \\ 0 & 0 & x & y & 0 & 1 & -xy' & -yy' \end{pmatrix}$$



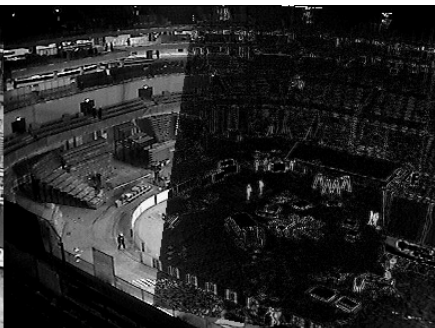
Hierarchical gradient descent (Lucas-Kanade):

1. **Model:** translation \Rightarrow rigid \Rightarrow affine \Rightarrow projective
2. **Scale:** 32x32 \Rightarrow 64x64 \Rightarrow 128x128 \Rightarrow 256x256

Composite



Error

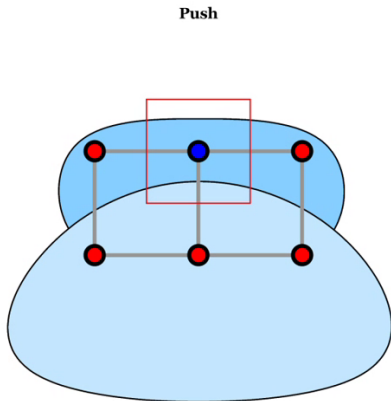


Algorithm:

repeat

1. **Push:**
block matching
2. Regularize:
shape matching

until convergence

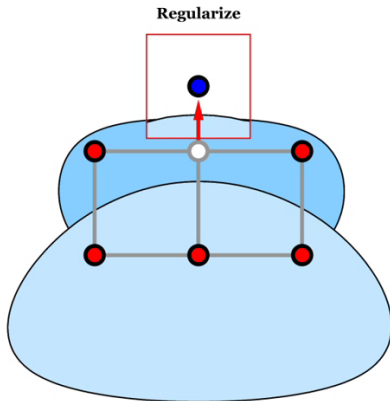


Algorithm:

repeat

1. Push:
block matching
2. **Regularize:**
shape matching

until convergence

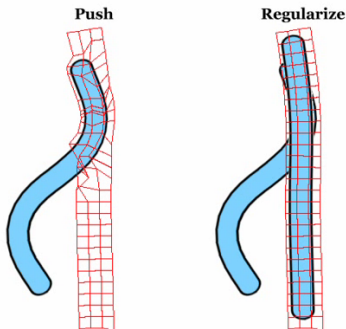


Algorithm:

repeat

1. **Push:**
block matching
2. **Regularize:**
shape matching

until convergence





source frames





source color frame

