BLG 453E

Week 3
Geometric/Coordinate Transforms I

Dr. Yusuf H. Sahin

An example for Pointwise Image Processing

*«Nobody does whatever they like,
Umberto, they do what you let
them do»*

Tommy Vercetti

*«On the floors of Tokyo
A-down in London town's a go go
A-with the record selection,
And the mirror's reflection,
I'm a dancin' with myself»*

Bily Idol

Part 1: Dancing alone

- Read the background image and some cat images from the specied folders.
- Place the cat images onto the background image.
- Write the obtained frames as a video.

Part 2: Dancing with myself

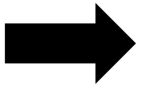
- As mentioned in the song, make the cat dance with his reection on the mirror. An example frame is given below.

Part 3: Dancing with my shadow

- Use a previously defined transform mapping to make the cat on the right darker.

Part 4: Dancing with my friend

- Use histogram matching between the video frames and the target image to create a different cat!



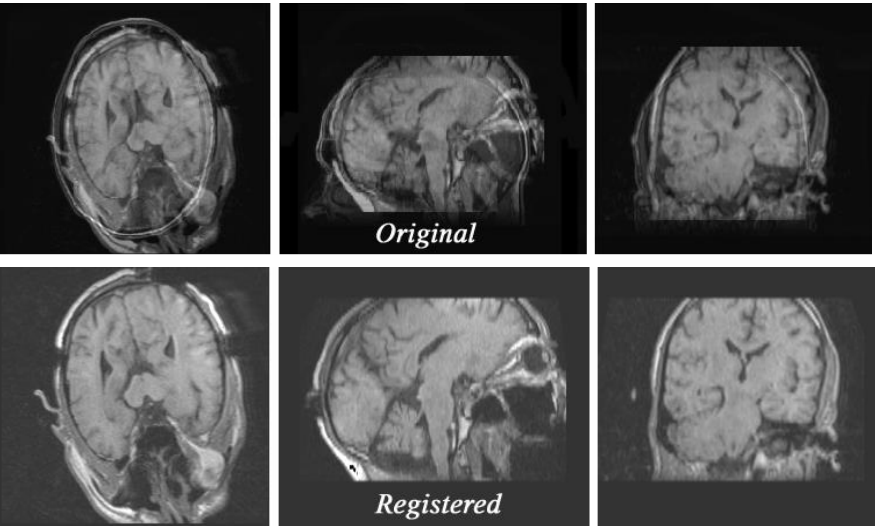
Part 5: Disco Dancing

- Obtain the flashing light effect by:
  - Randomly perturbing the cat’s histogram for the cat on the left.
  - Randomly perturbing the target histogram for the cat on the right.

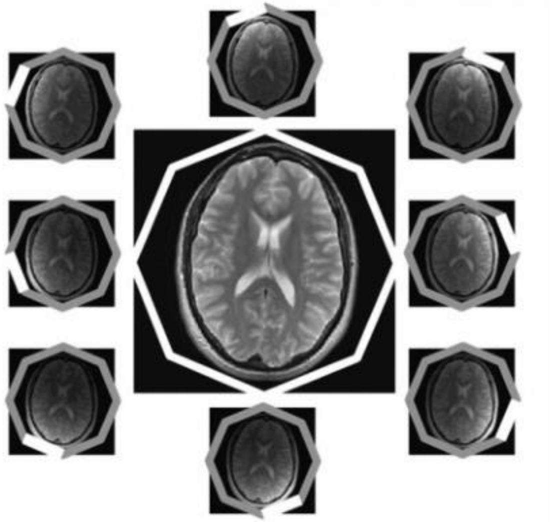
Coordinate transforms

Translating & rotating every pixel/voxel of the image.

Image Registration

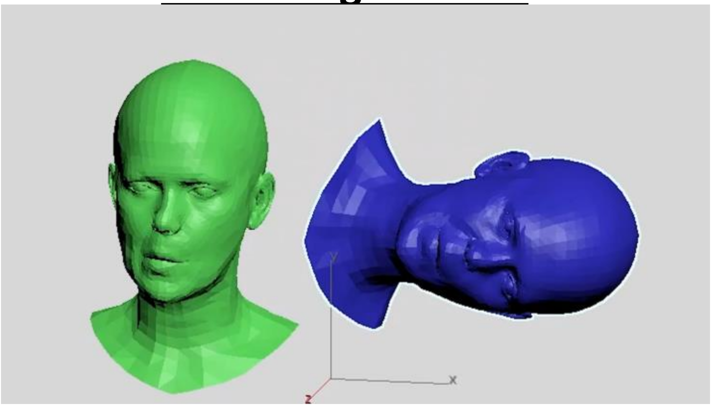


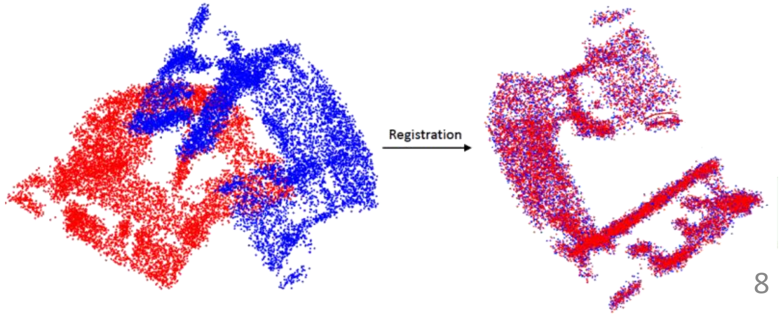
Whole-to-whole registration of the brain.



Coils with different sensitivity maps around the brain could be registred into one.

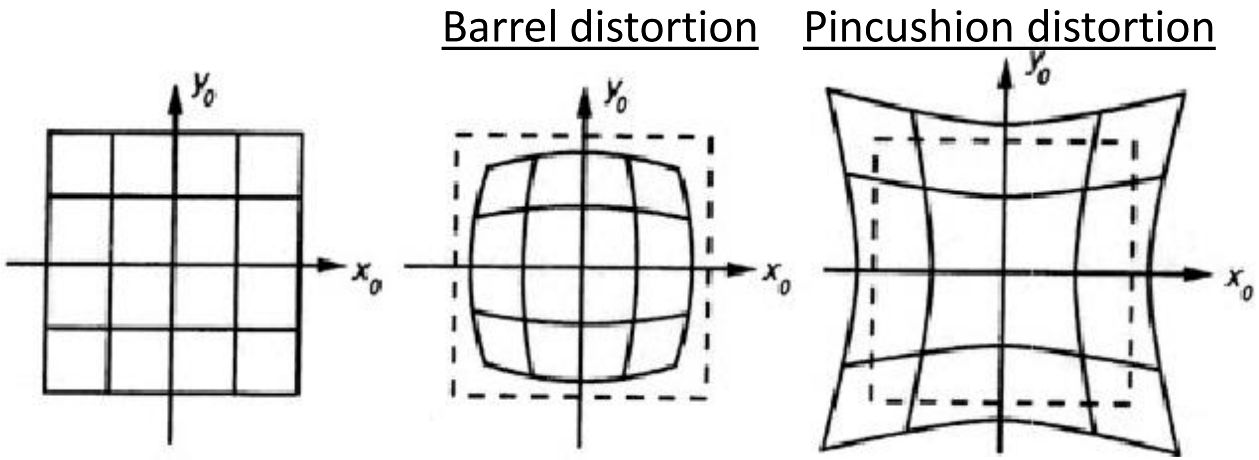
3D PC Registration



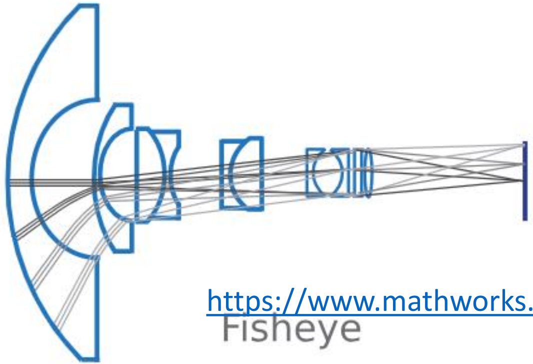


Radial Distortion

- Radial distortion is observed in images captured through wide-angle lenses.



Park, J., Byun, S. C., & Lee, B. U. (2009). Lens distortion correction using ideal image coordinates. *IEEE Transactions on Consumer Electronics*, *55*(3), 987-991.



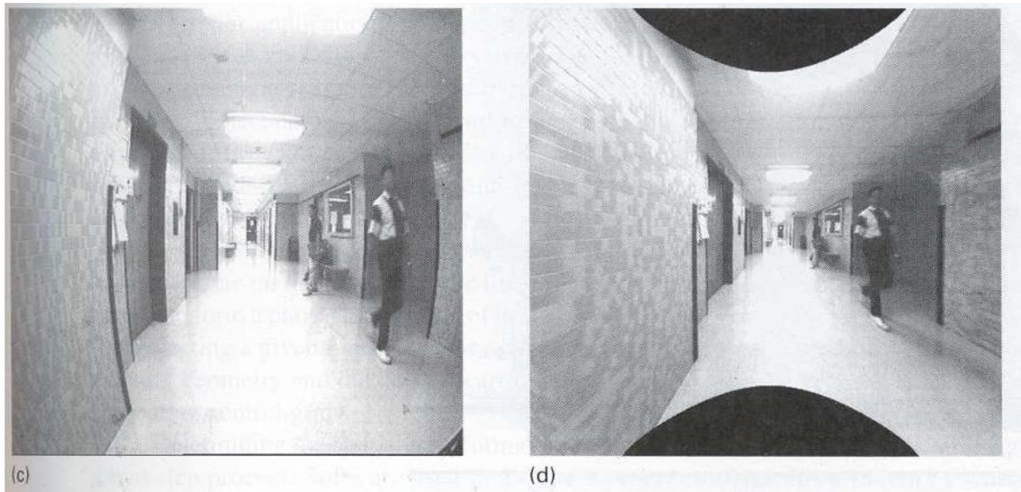
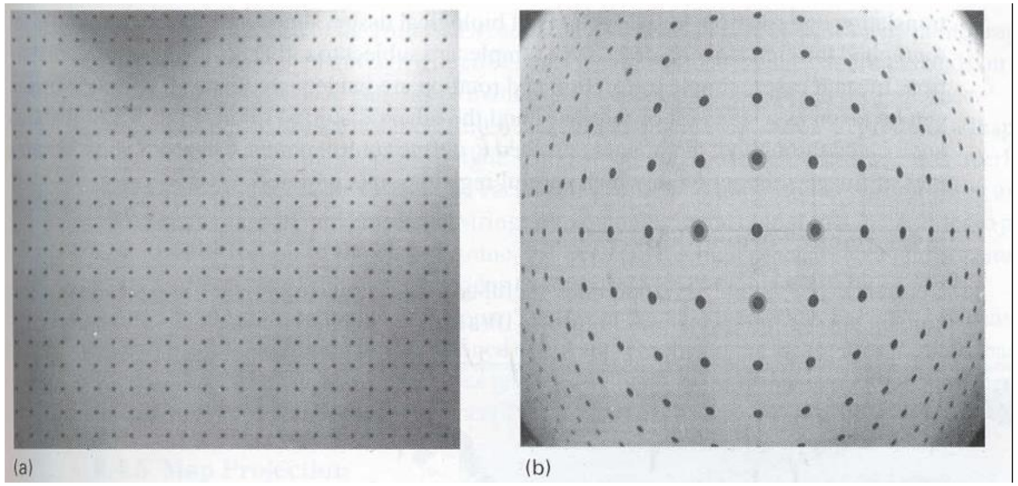


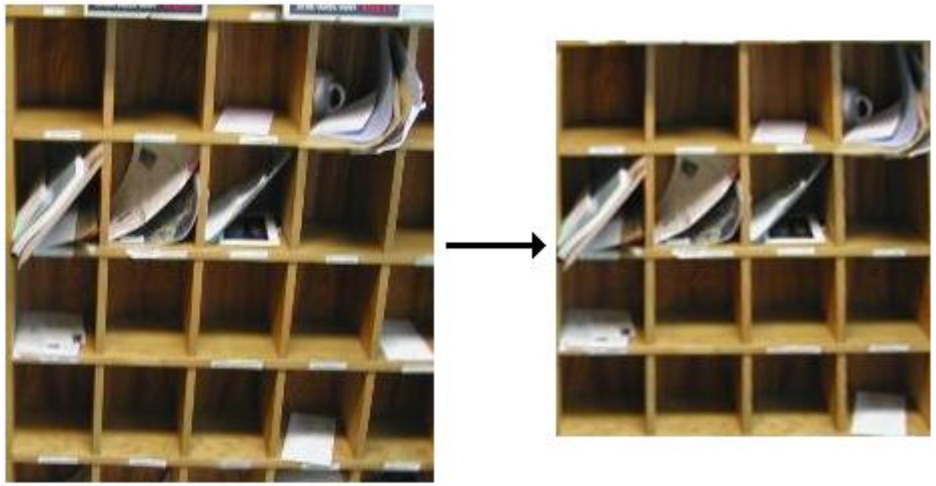
Figure 8–9 Geometric rectification of an image taken with a fish-eye lens: (a) test target, (b) fisheye image; (c) original, (d) rectified hallway image (Courtesy Shishir Shah, The University of Texas at Austin, from [20])



https://www.mathworks.com/help/vision/ug/camera-calibration.html#bu1ahkv

Geometric transformations

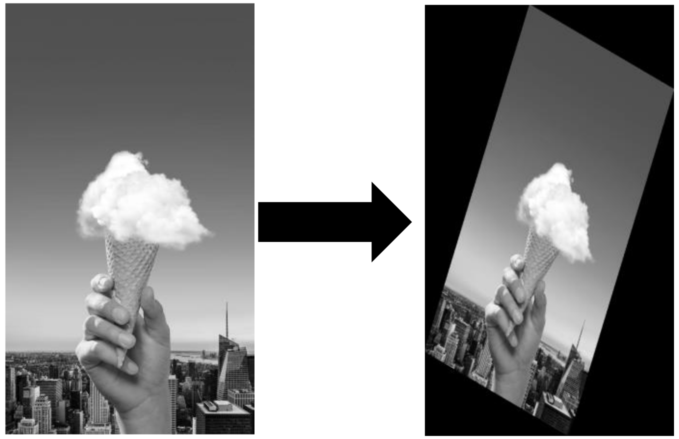
- Geometric transformations change the spatial position of pixels in the image.
- They are also known as *image warps*.
- Practical Uses:
  - Bringing multiple images into the same coordinate system: Registration, Homography Estimation
  - Removing distortion
  - Stereo matching
  - Image morphing



Distorted image Corrected image

Geometric transformations

- In a geometric transformation, the positions of pixels in the image is transformed.
- Mathematically, this is expressed (in a general form) as:



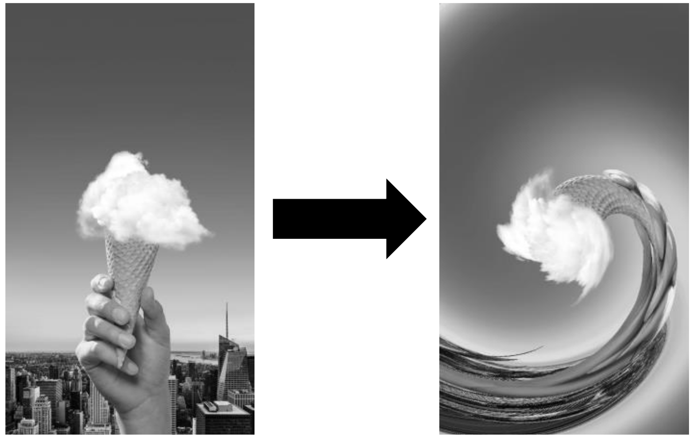
- Map your coordinates first:

$$x' = T(x)$$

Then map the images

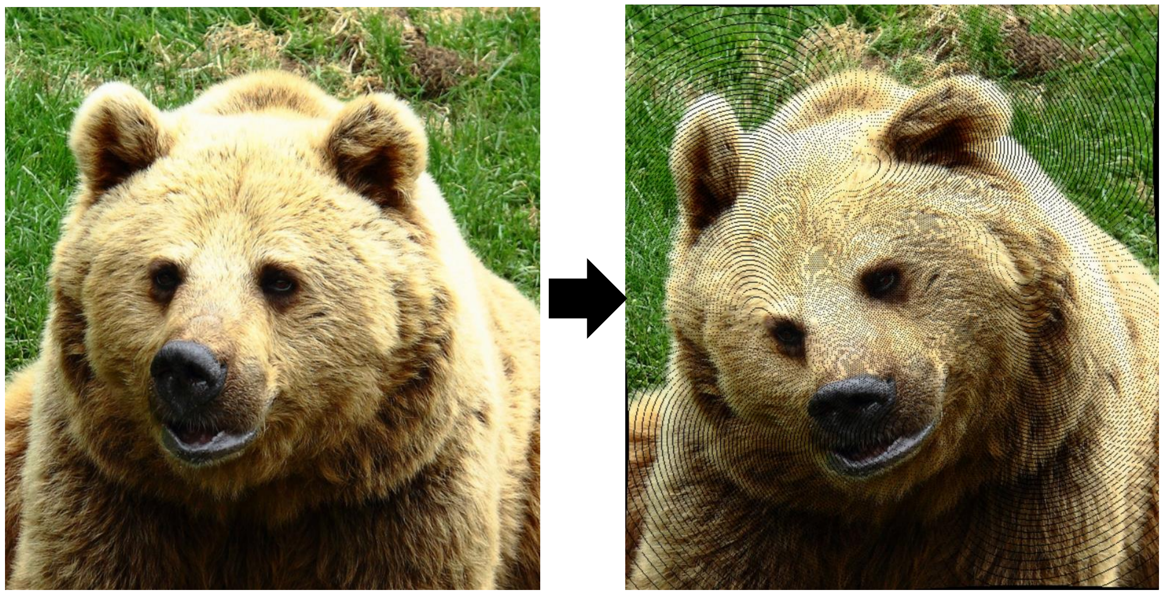
$$J(x') = I(x)$$

Resample if the transformation results in non-integer pixel coordinates or if the pixel grid changes size.

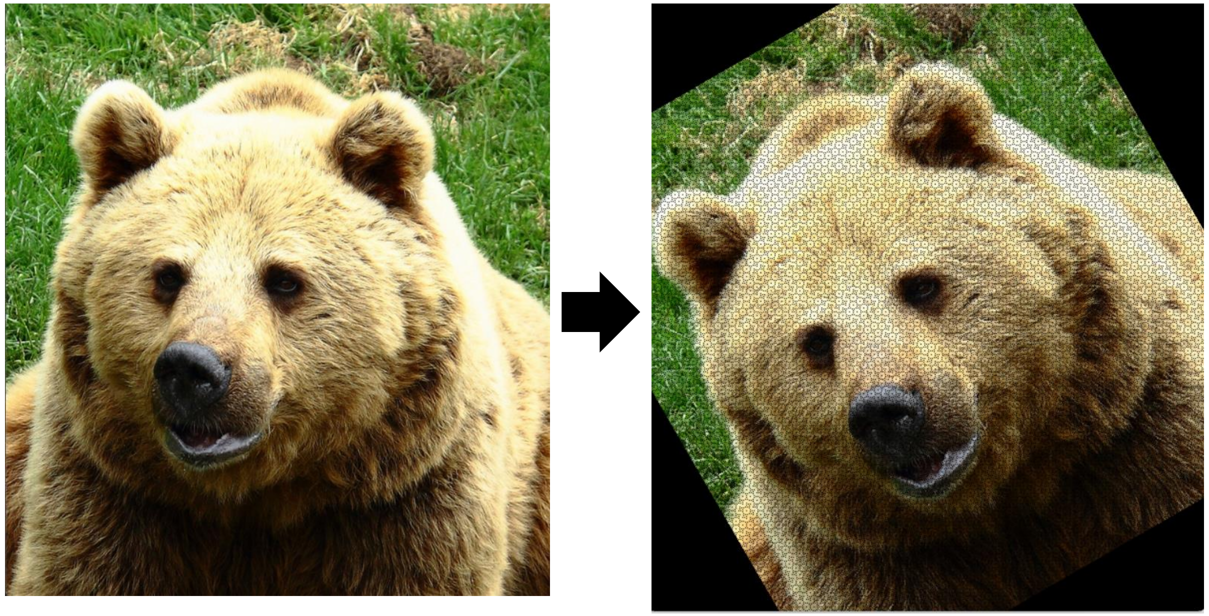


Coordinate Mapping

- Describe the destination (x, y) for every location (u, v) in the source image (or vice-versa, if invertible transformation).



week2_1.py



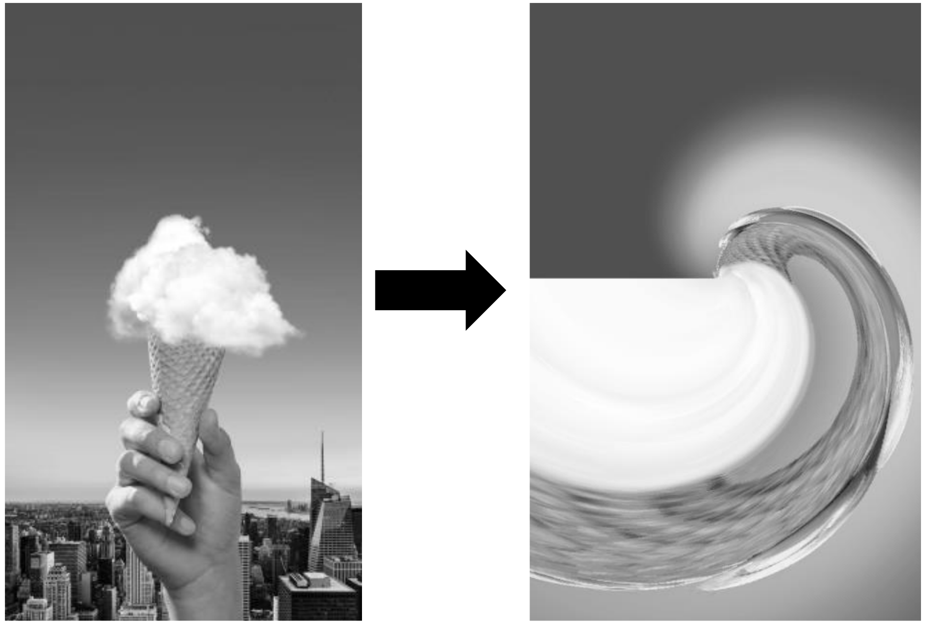
week2_2.py

Coordinate Mapping

- Scale by factor
  - $x' = factor * x$
  - $y' = factor\ * y$

A general coordinate mapping:
- $x' = f_x(x, y)$
- $y' = f_y(x, y)$

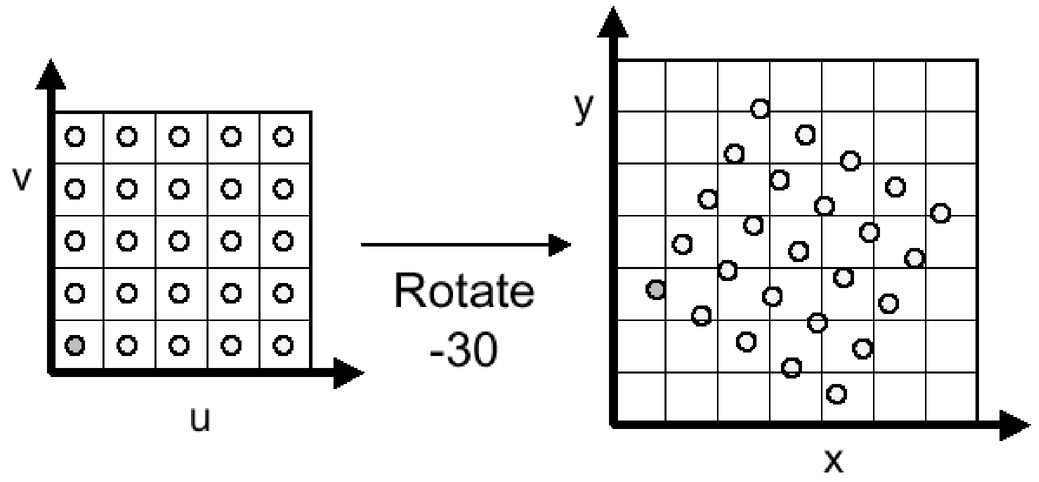
- Rotate by θ degrees
  - $x' = xcos\theta - ysin\theta$
  - $y' = xsin\theta + ycos\theta$



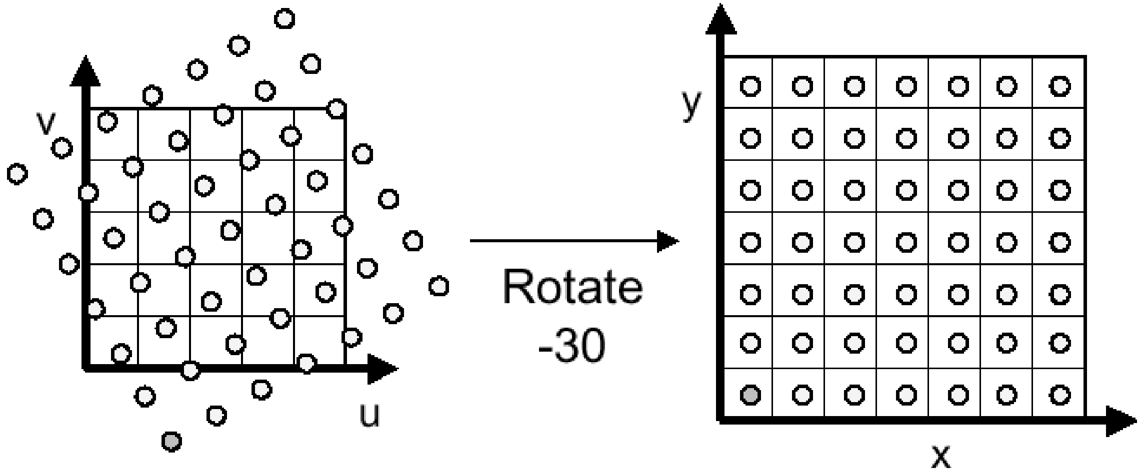
Forward & Backward Mapping

- Iterate over source image
- Some destination pixels may not be covered.
- Many source pixels can map to the same destination pixel.

Forward Mapping



Backward Mapping

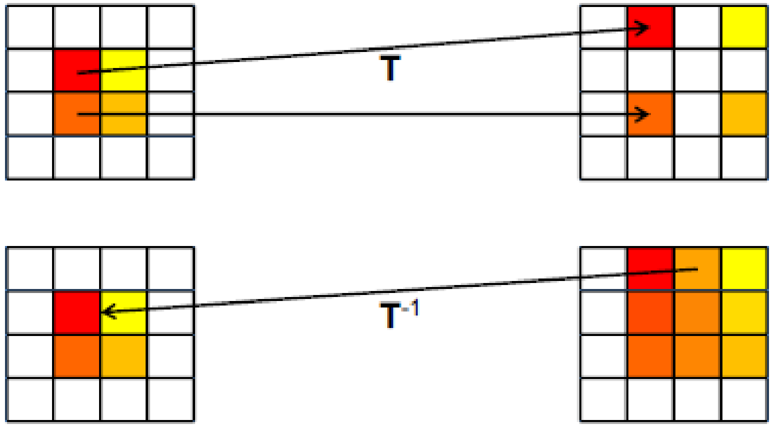


Prof. M. Milanova, University of Arkansas at Little Rock

Geometric transformations

- We will prefer *backwards*, rather than using a (forward) mapping T to transform the pixels from the distorted image to the corrected image, we use an (inverse) transform T^{-1}.

- Using an inverse mapping ensures all the pixels in the corrected image will be filled. However, it's necessary to interpolate pixels from the distorted image.



Forward mapping may result in gaps

Inverse mapping ensures no gaps

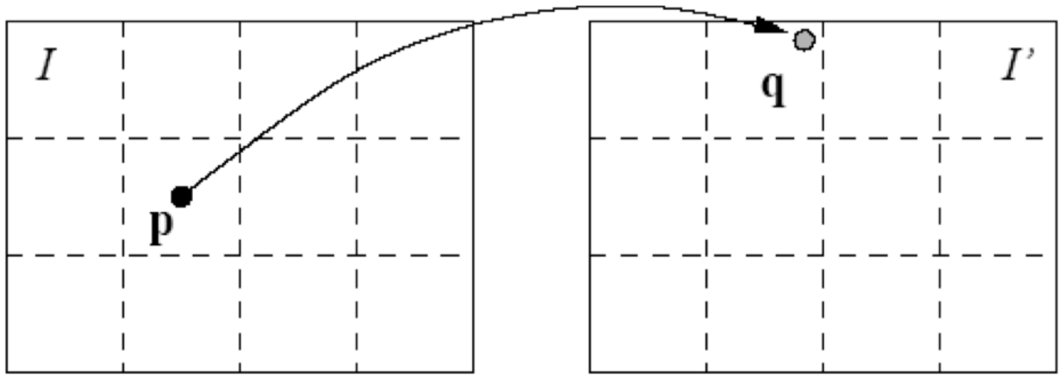
Forward & Backward Mapping

Forward Mapping

procedure *forwardWarp*($f, \boldsymbol{h}$, **out** g):

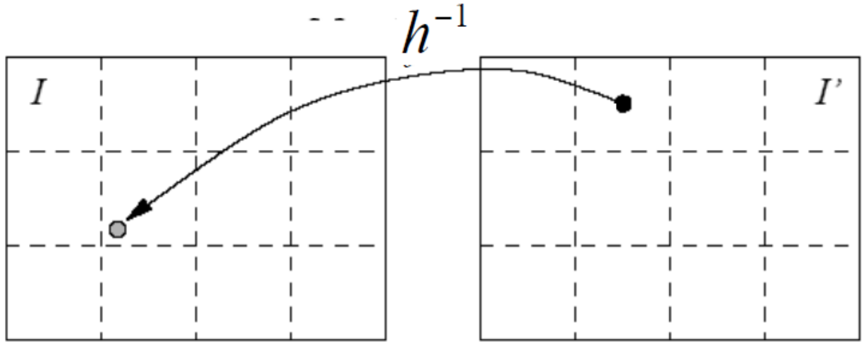
For every pixel $\boldsymbol{x}$ in $f(\boldsymbol{x})$

1. Compute the destination location $\boldsymbol{x}' = \boldsymbol{h}(\boldsymbol{x})$.
2. Copy the pixel $f(\boldsymbol{x})$ to $g(\boldsymbol{x}')$.



-Round-off errors
-Missing Grid Points

Backward Mapping



$$x = h^{-1}(x')$$

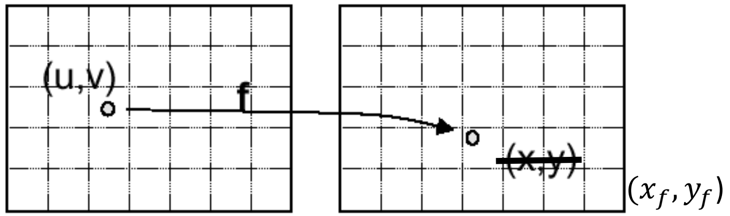
- Use the colors of neighboring integer coordinate points in I to estimate I(p).
  - Bilinear interpolation

$$I'(x', y') = I(h^{-1}(x, y))$$

week2_3.py

Gray Level Interpolation

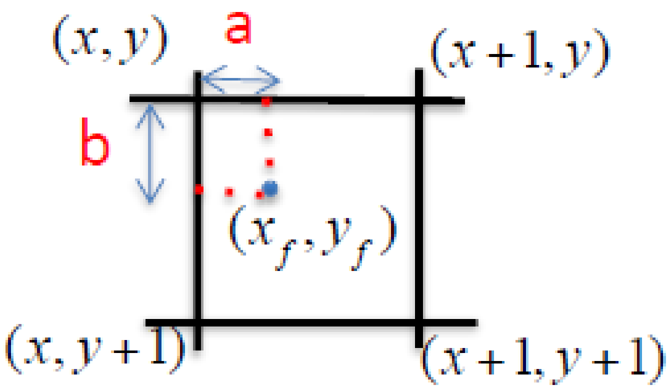
- Through a geometric mapping, pixels in image f can map to positions between pixels in image g.



- Use the colors (or gray values) of neighboring integer-coordinate points in I to estimate $I(x_f, y_f)$.

$$J(u,v) = I(x_f, y_f)$$

Gray Level Interpolation

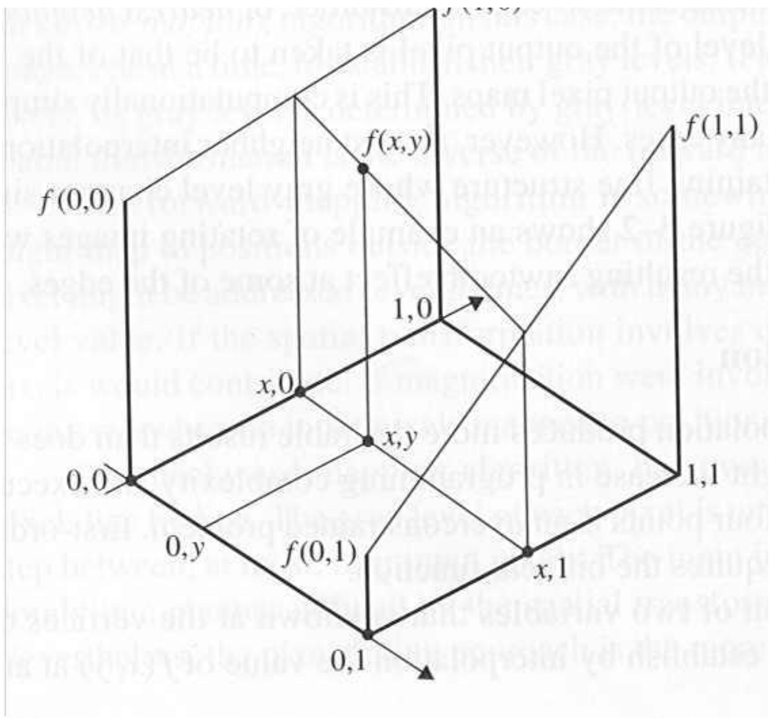


- Nearest Neighbor interpolation:
    - Nearest Integer coordinate values (rounded)

$$J(u,v) = I(x_f, y_f) = I\big(round(x_f), round(y_f)\big)$$

- Bilinear Interpolation
    - Roundoff error is avoided.

$$I(x_f, y_f) = (1-a)(1-b)I(x,y) + a(1-b)I(x+1,y) + abI(x+1,y+1)$$



Fahmy, S. A. (2008, December). Generalised parallel bilinear interpolation architecture for vision systems. In *2008 International Conference on Reconfigurable Computing and FPGAs* (pp. 331-336). IEEE.

Transformations in Homogenous Coordinates

- Using Homogeneous coordinates makes it possible for these geometric transforms.

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

- There are some main operations which could be represented as matrix multiplication.

Translation

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Need to specify the Center of Rotation.
-How?

Scaling

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Inverse transformation

- For the given operations inverse transforms could also be defined: T^{-1}

Translation

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

Scaling

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1/s_x & 0 & 0 \\ 0 & 1/s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

Q:

For the image on the left, X-Y axis and rotation center, what is the correct transformation matrix to obtain the second image?