ECSE 4540: Introduction to Image Processing	**Fall 2019**

Lecture 5: Geometric Operations

Lecturer: Rich Radke *Scribes: Yao Zhang*

It covers: geometric operations, translation, scaling, flipping, linear transformations, rotation, similarity transformations, shears, affine transformations, Matlab examples, projective transformations, example estimating a projective transformation, creating the output image, bilinear interpolation, extensions.

Today we stay in Euclidean coordinates:

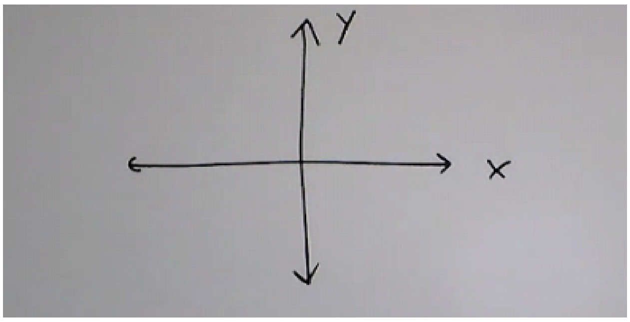

Figure 5.1: Euclidean coordinates

$$\begin{aligned} J(x,y) &= I(T(x,y)) && \textit{geometric} \\ J(x,y) &= T(I(x,y)) && \textit{point operartor} \end{aligned} \tag{5.1}$$

Example 1

1.
$$J(x,y) = I(x+2,y) \tag{5.2}$$
then we can get $J(0,0) = I(2,0)$ *and* $J(-2,0) = I(0,0)$.

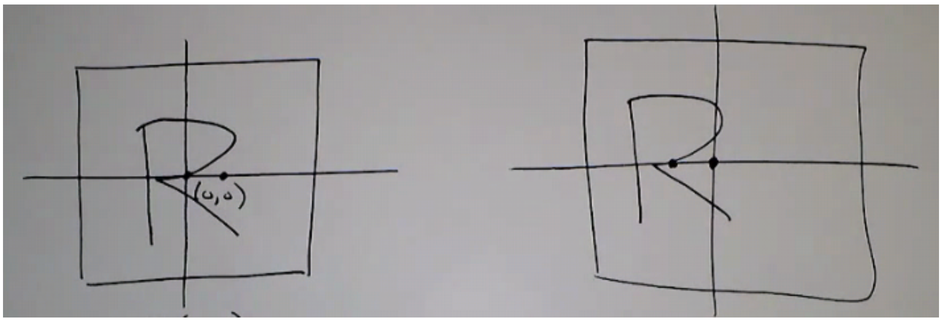

Figure 5.2

2.
$$J(x,y) = I(x,y-10) \tag{5.3}$$
then we can get $J(0,0) = I(0,-10)$ *and* $J(0,10) = I(0,0)$.

3. *scaling*

$$J(x,y) = I(2x,2y) \tag{5.4}$$

then we can get $J(0,0) = I(0,0)$ *and* $J(1,1) = I(2,2)$.

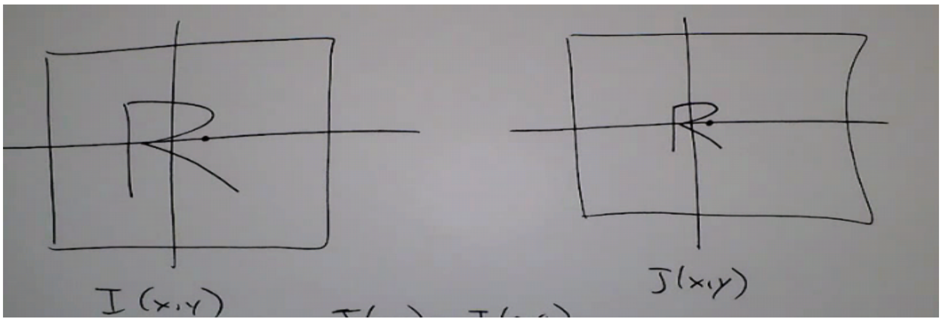

Figure 5.3

4.

$$J(x,y) = I(\frac{1}{3}x, \frac{1}{3}y) \tag{5.5}$$

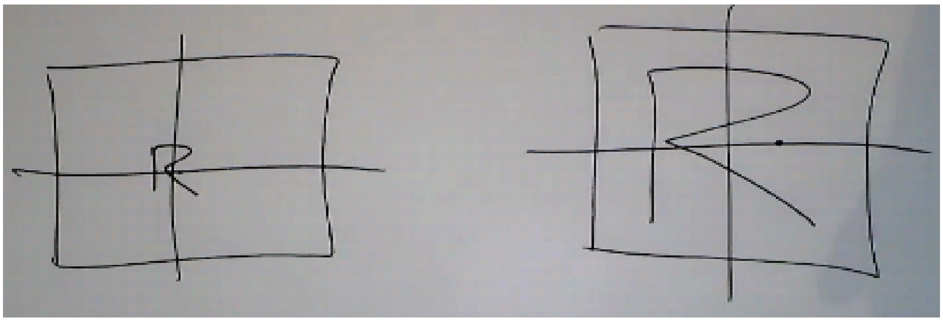

Figure 5.4

5. *flip*

$$J(x,y) = I(-x,y) \tag{5.6}$$

6. *reflect across y axis*

$$J(x,y) = I(x,-y) \tag{5.7}$$

7.

$$J(x,y) = I\left(2x, \frac{1}{2}y\right) \tag{5.8}$$

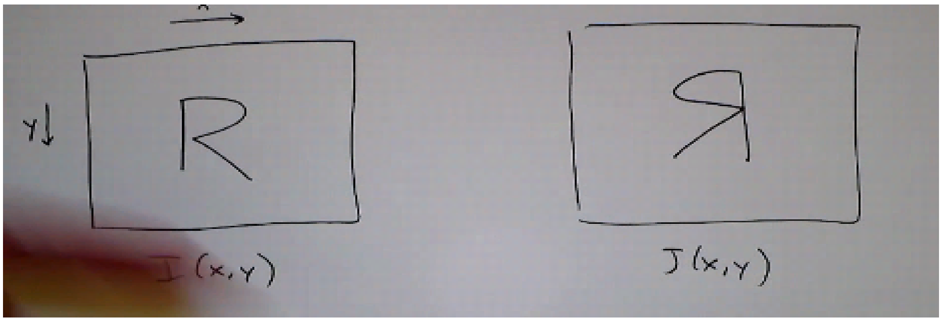

Figure 5.5: 5

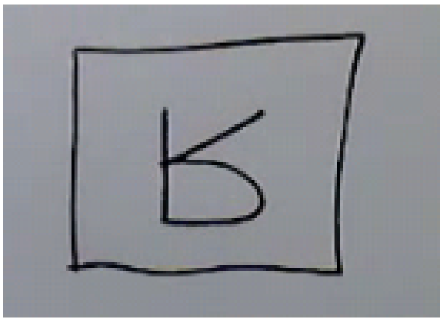

Figure 5.6: 6

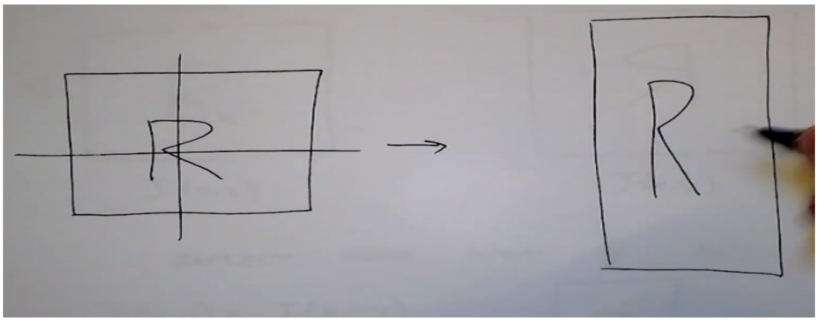

Figure 5.7: 7

It is common for scale + shift + flip to be combined into a 2D linear transformation.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} * & * \\ * & * \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} * \\ * \end{bmatrix} \tag{5.9}$$

where $\begin{bmatrix} x' \\ y' \end{bmatrix}$ is the coordinates of translations image $J(x, y)$, and $\begin{bmatrix} x \\ y \end{bmatrix}$ is the coordinates of original image $I(x, y)$. Where does (x, y) in the old image go to? (forward mapping).

We can do things to images that we cannot do to 1D signals, e.g. rotation rotation by θ° counter clocking.

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \to \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix} \to \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} \tag{5.10}$$

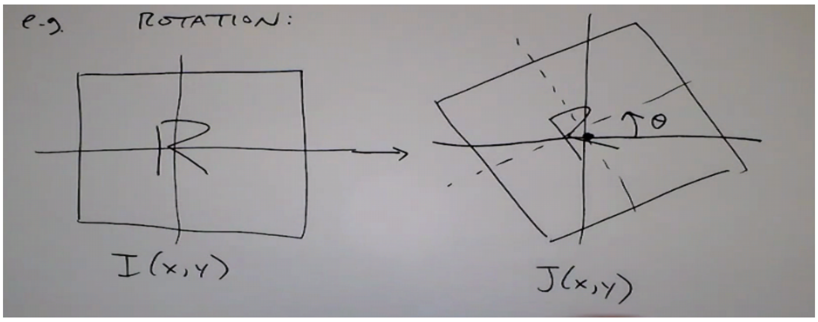

Figure 5.8: rotation

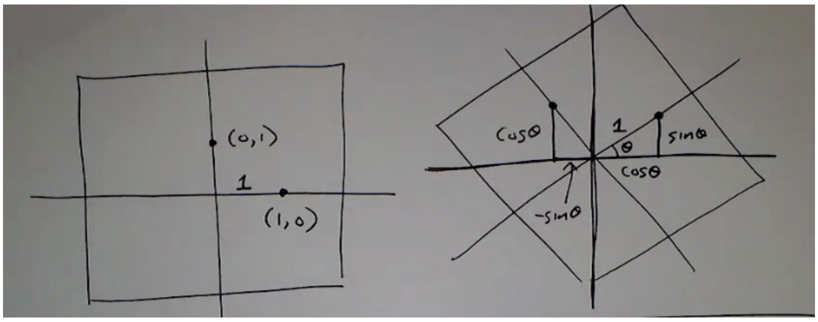

Figure 5.9

and

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.11}$$

Any combination of scale, shift, rotate is called a similarity transformation, preserves parallel lines, if $\alpha, \beta = \pm 1$, isometric transformation (rigid, motion), preserves shapes, angles.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \tag{5.12}$$

what if

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + by \\ y \end{bmatrix} \tag{5.13}$$

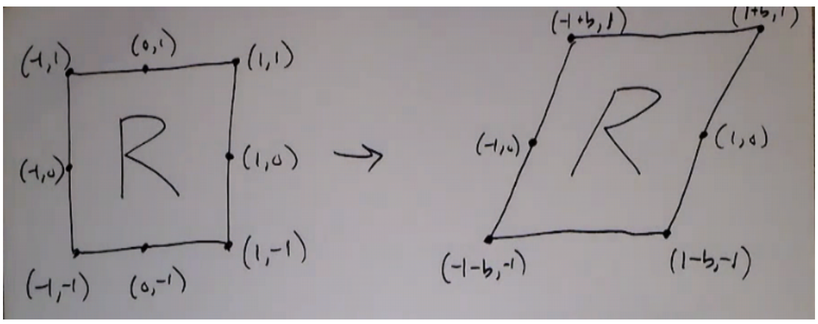

Figure 5.10

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x + by \\ y \end{bmatrix} \quad \textit{shear} \tag{5.14}$$

$$\begin{gathered} (0,0) \to (0,0), (1,0) \to (1,0), (-1,0) \to (-1,0), (-1,-1) \to (-1-b,-1) \\ (0,1) \to (b,1), (1,1) \to (1+b,1), (-1,1) \to (-1+b,1), (0,-1) \to (-b,-1) \end{gathered} \tag{5.15}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.16}$$

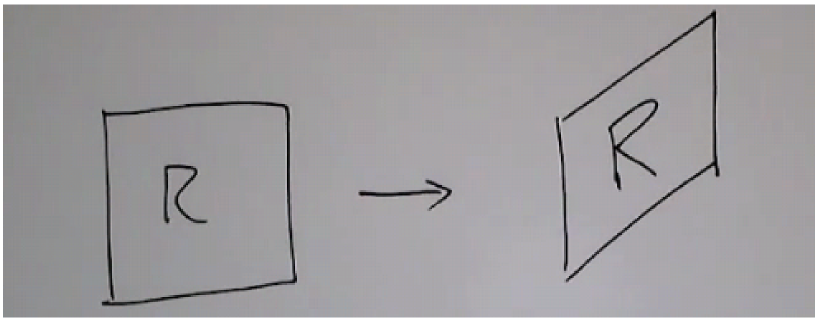

Figure 5.11: vertical shear

$$\begin{bmatrix} 1 & b \\ c & 1 \end{bmatrix} \tag{5.17}$$

A transformation of the form

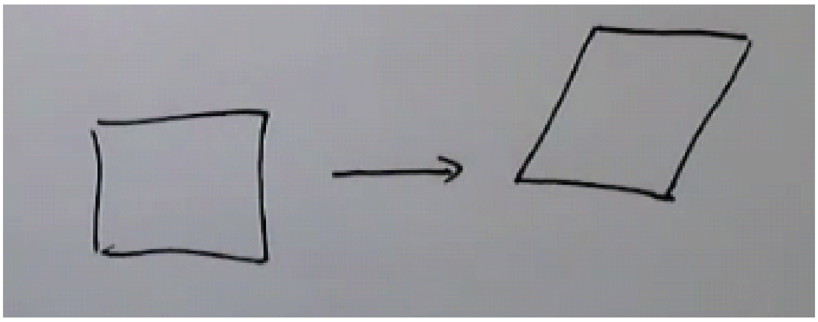

Figure 5.12: rectangle → parallelogram

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{5.18}$$

is called an affine transformation.

projective transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \frac{a_{11}x + a_{12}y + b_1}{c_1 x + c_2 y + 1} \\ \frac{a_{21}x + a_{22}y + b_2}{c_1 x + c_2 y + 1} \end{bmatrix} \tag{5.19}$$

$$\begin{bmatrix} \widetilde{x}' \\ \widetilde{y}' \\ \widetilde{z}' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{5.20}$$

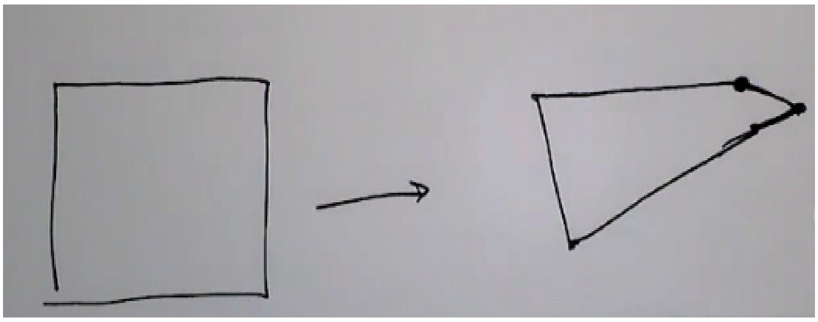

Figure 5.13: rectangle → quadrilateral

$$\begin{aligned} x' &= \frac{\widetilde{x}'}{\widetilde{z}'} \\ y' &= \frac{\widetilde{y}'}{\widetilde{z}'} \end{aligned} \tag{5.21}$$

How to actually create the output image?

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1.2 & 1 \\ 1 & 0.8 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 3.2 \\ -1.6 \end{bmatrix} \tag{5.22}$$

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \to \begin{bmatrix} 5.4 \\ 0.2 \end{bmatrix} \tag{5.23}$$

How to get image colors/intensities on the grid? It's more conventional to use backward mapping?

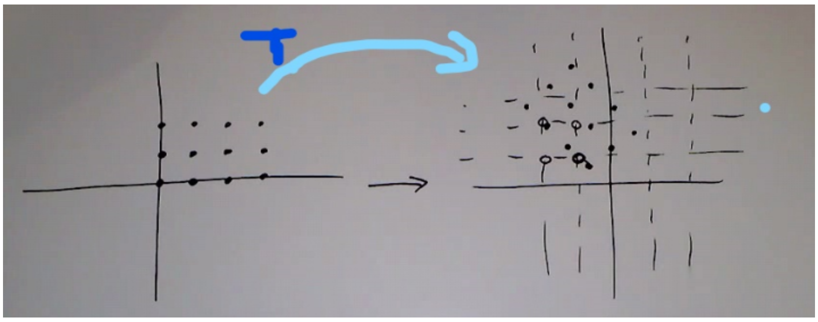

Figure 5.14

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix} + b \Rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = A^{-1} \left(\begin{bmatrix} x' \\ y' \end{bmatrix} - b \right) = A^{-1} \begin{bmatrix} x' \\ y' \end{bmatrix} - A^{-1} b \tag{5.24}$$

$$x = \left[(1-\beta)\, a + \beta c \right] (1-\alpha) + \left[(1-\beta)\, b + \beta d \right] \alpha \tag{5.25}$$

bicubic interpolation uses more points look smoother.

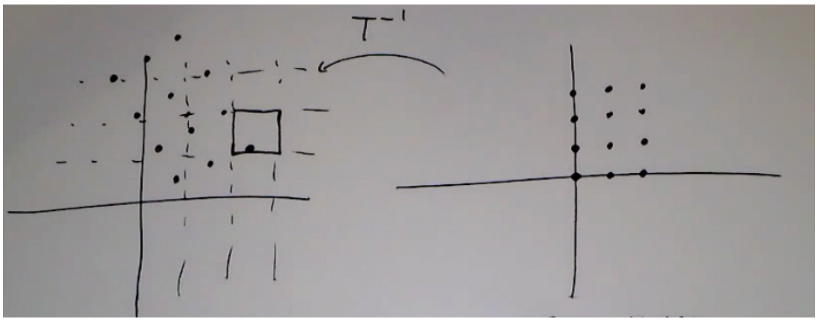

Figure 5.15

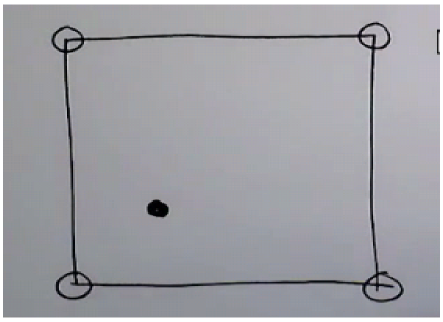

Figure 5.16: bilinear interpolation

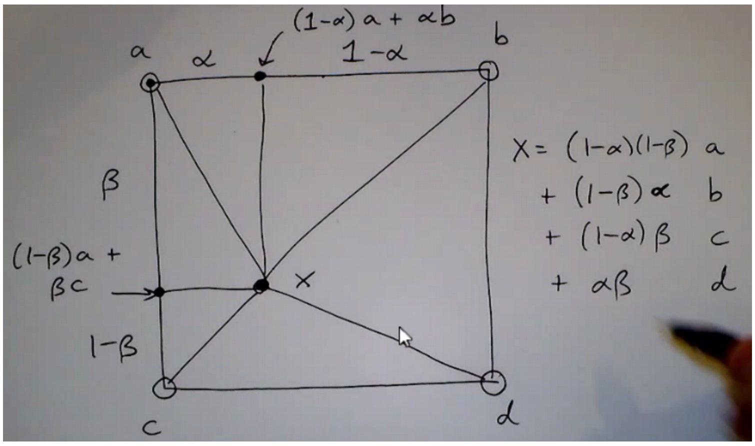

Figure 5.17

Other geometric transformations,

$$
\begin{aligned}
x' &= F\,(x, y) \\
y' &= G\,(x, y)
\end{aligned}
\tag{5.26}
$$

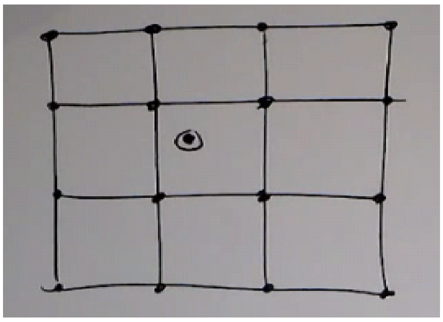

Figure 5.18

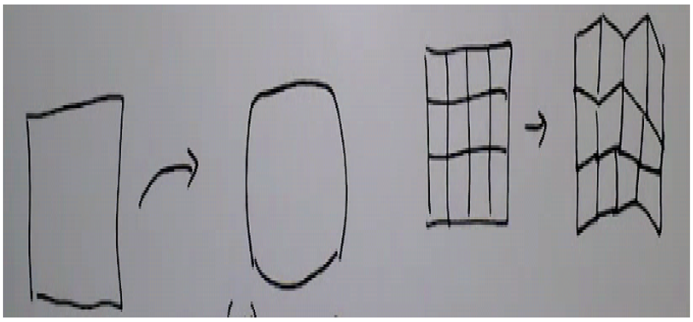

Figure 5.19: lens distortion

```
clc;
clear;
warning off;
im = imread('leftshark.jpg');
imshow(im)
% doc
% doc  imrotate
out = imrotate(im,30);
imshow(out)
% doc imwarp
a = cos(30);
b = sin(30);
c = sqrt(3)/3;
T = [cosd(30) -sind(30) 0; sind(30) cosd(30) 0; 0 0 1];
Tf = affine2d(T);
out = imwarp(im,Tf);
imshow(out)
T = affine2d([1 .1 0; 0 1 0; 0 0 1]);
T.T;
out  = imwarp(im, T);
imshow(out)
% doc affine2d
T = projective2d([1 0.7 .001; 0 1 0; -200 -300 1]);
T.T
out = imwarp(im,T);
```

```
imshow(out)
im1 = imread('img002.png');
im2 = imread('img003.png');
imshow(im1)
imshow(im2)
cpselect(im1,im2);
% doc imrotate
out = imrotate(im, 30);
imshow(out)
out2  = imrotate(im, 30, 'bilinear');
figure
imshow(out2)
```

References

[GW18] Gonzalez and Woods, Digital Image Processing, *Pearson*, 2018.