Image Processing

Geometric Transform

Yeong Jun Koh
Department of Computer Science & Engineering
Chungnam National University

Objectives

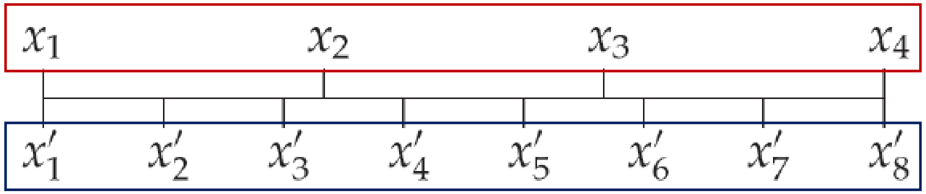
- Interpolation
  - 1D signal interpolation
  - 2D image interpolation

- Relation between interpolation and spatial filtering

- Parametric transform
  - Scaling
  - Rotation
  - Translation
  - Affine

Interpolation

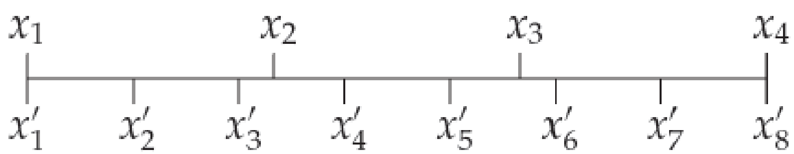
- A set of four values that we wish to enlarge to eight
  - In discrete signal, output x_i' points does not coincide with input x_i points, except for the first and last points
  - Input signal: 4 values
  - Output signal: 8 values

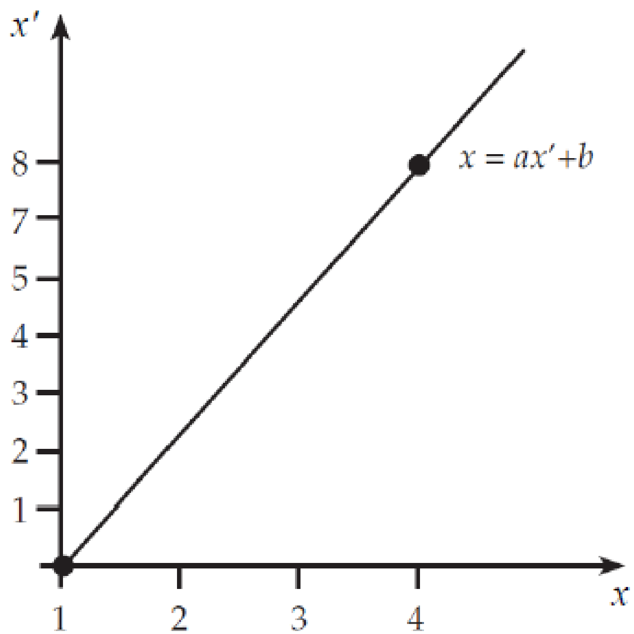


- Interpolation
  - Estimate $f(x_i')$ based on the known values of nearby $f(x_i)$, i.e., neighboring pixels

1. Compute the location x_i'
2. Compute the function value (intensity) $f(x_i')$

Compute the location x_i'-1D

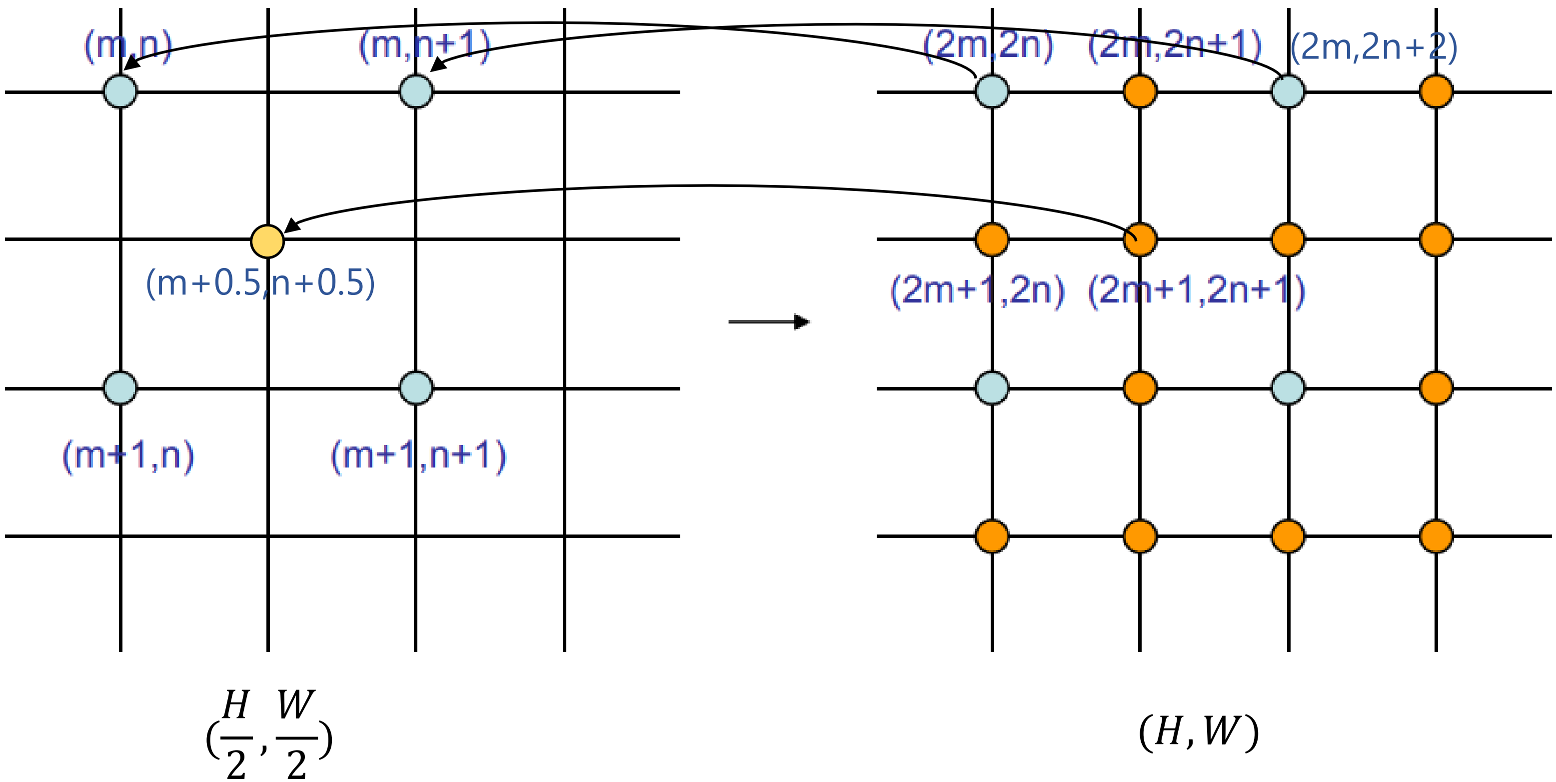




$$\begin{cases} 1 = a + b \\ 4 = 8a + b \end{cases}$$

$$\boxed{x = \frac{3}{7}x' + \frac{4}{7}}$$

Compute the location x_i'-2D

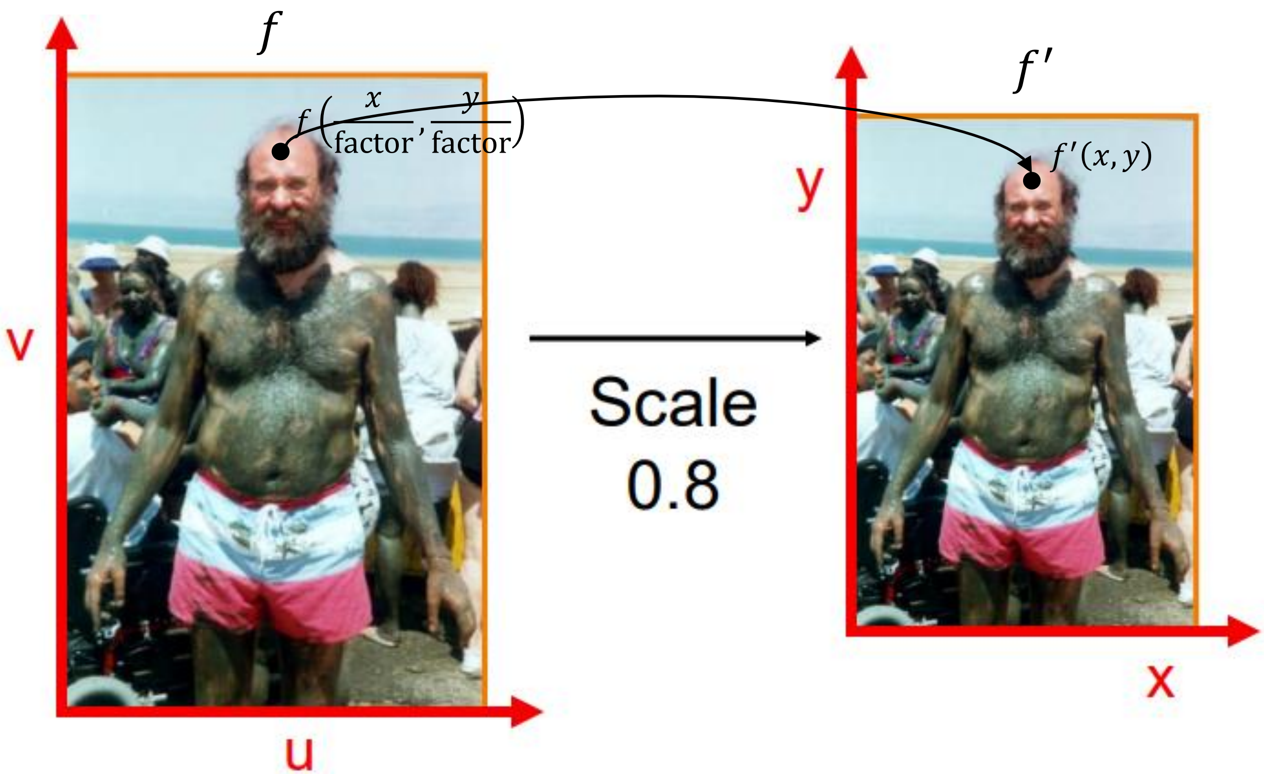


Factor of 2 Up-Sampling

Compute the location x_i'-2D

- Scale factor
  - $x = \text{factor} \times u$
  - $y = \text{factor} \times v$

$$f'(x, y) = f\left(\frac{x}{\text{factor}}, \frac{y}{\text{factor}}\right)$$

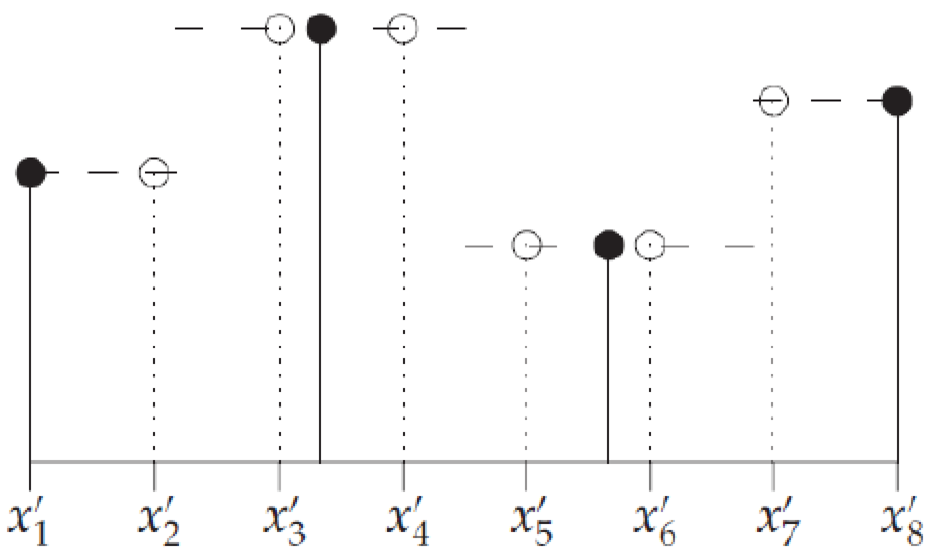


Compute the function value $f(x_i')$

- Several interpolation methods in MATLAB
  - function imresize()
  - 'nearest': nearest-neighbor interpolation
  - 'bilinear': bilinear interpolation
  - 'bicubic': cubic interpolation (default in MATLAB)
  - 'lanczos2' and 'lanczos3': the best performance

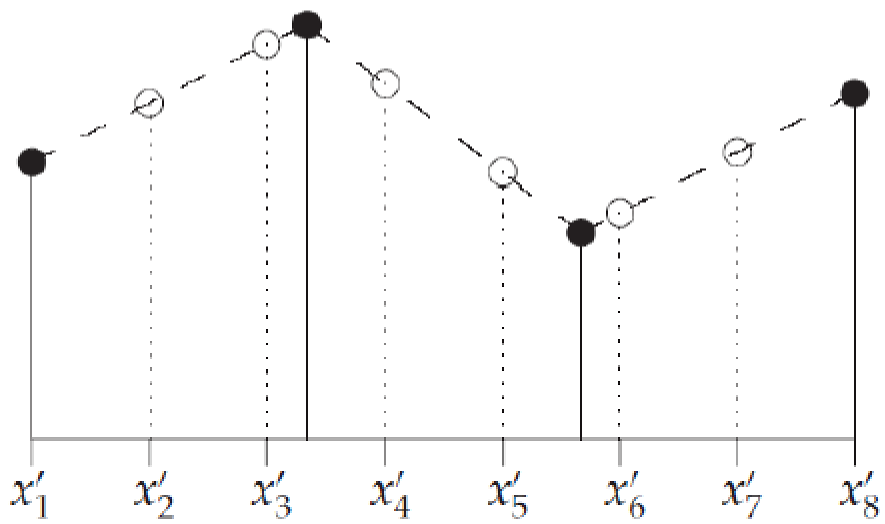
Compute the function value $f(x_i')$

- Nearest-neighbor interpolation
  - Simply copy-and-paste the neighbor intensity value



Compute the function value $f(x_i')$

- Linear interpolation
  - Using linear equation $f(m) = am + b$



Compute the function value $f(x_i')$

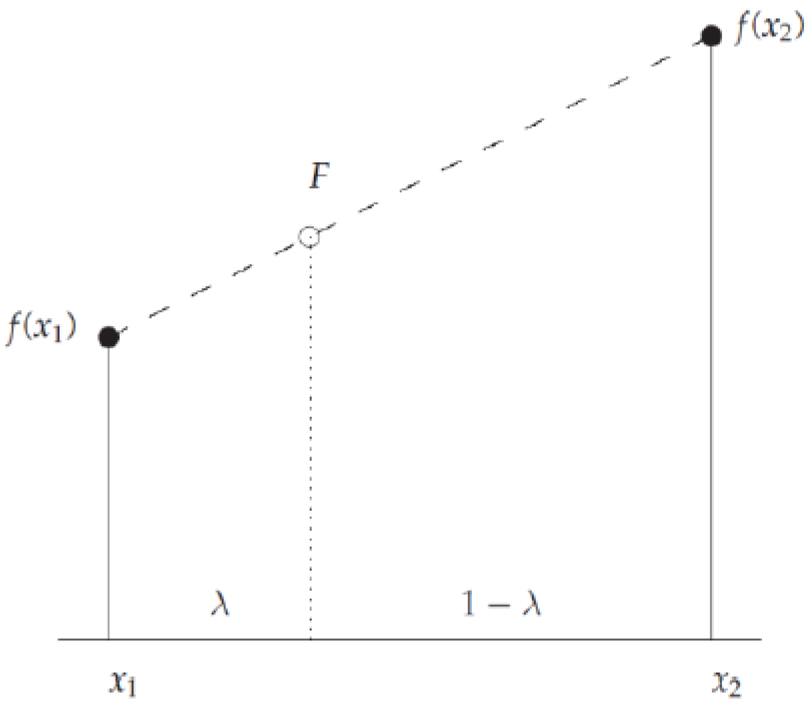
- Linear interpolation
  - Linear equation: using proportional formula

$$\frac{F - f(x_1)}{\lambda} = \frac{f(x_2) - f(x_1)}{1}$$

$$\Rightarrow \quad F = \lambda f(x_2) + (1 - \lambda) f(x_1)$$

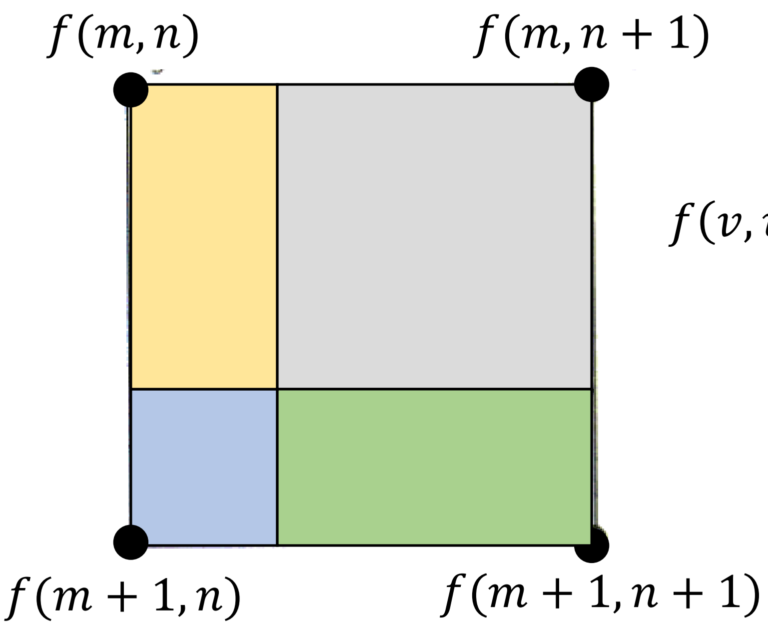
It is assumed that the distance between x_1 and x_2 is normalized $(= 1)$.

Why?



Bilinear Interpolation (in 2D)

- Applying the linear interpolation to 2D data (image)

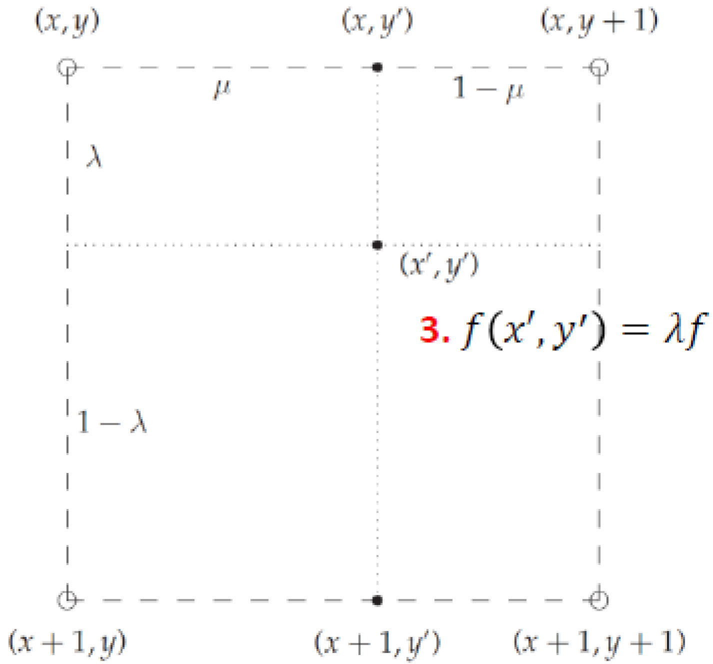


$$
\begin{aligned}
f(v, u) = &\ (1-s)(1-t) \cdot f(m, n) \\
&+ s(1-t) \cdot f(m, n+1) \\
&+ (1-s)t \cdot f(m+1, n) \\
&+ st \cdot f(m+1, n+1)
\end{aligned}
$$

Bilinear Interpolation (in 2D)

- Applying the linear interpolation to 2D data (image)

1. $f(x, y') = \mu f(x, y+1) + (1-\mu) f(x, y)$



3. $f(x', y') = \lambda f(x+1, y') + (1-\lambda) f(x, y')$

FIGURE 6.8 *Interpolation between four image points.*

2. $f(x+1, y') = \mu f(x+1, y+1) + (1-\mu) f(x+1, y)$

Bilinear Interpolation (in 2D)

- Interpolate $f\left(\frac{1}{2},0\right)$ using $f(0,0)$ and $f(1,0)$
- Interpolate $f\left(\frac{1}{2},1\right)$ using $f(0,1)$ and $f(1,1)$
- Interpolate $f\left(\frac{1}{2},\frac{1}{2}\right)$ using $f\left(\frac{1}{2},0\right)$ and $f\left(\frac{1}{2},1\right)$

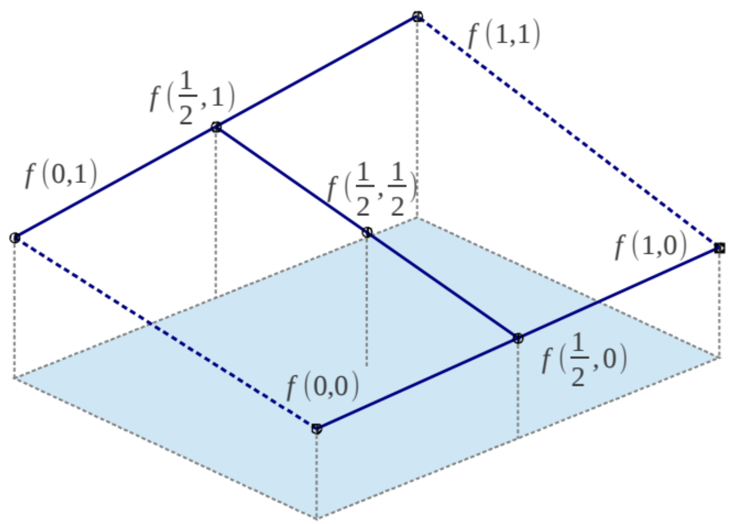


Image Interpolation

- Linear interpolation

Nearest neighbor vs. Bilinear interpolation

FIGURE 6.9 *The head.*

Mathworks

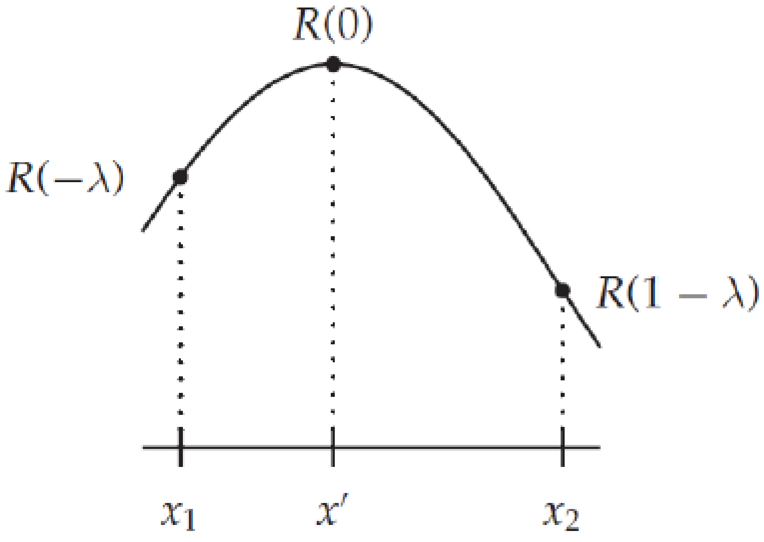
(a)

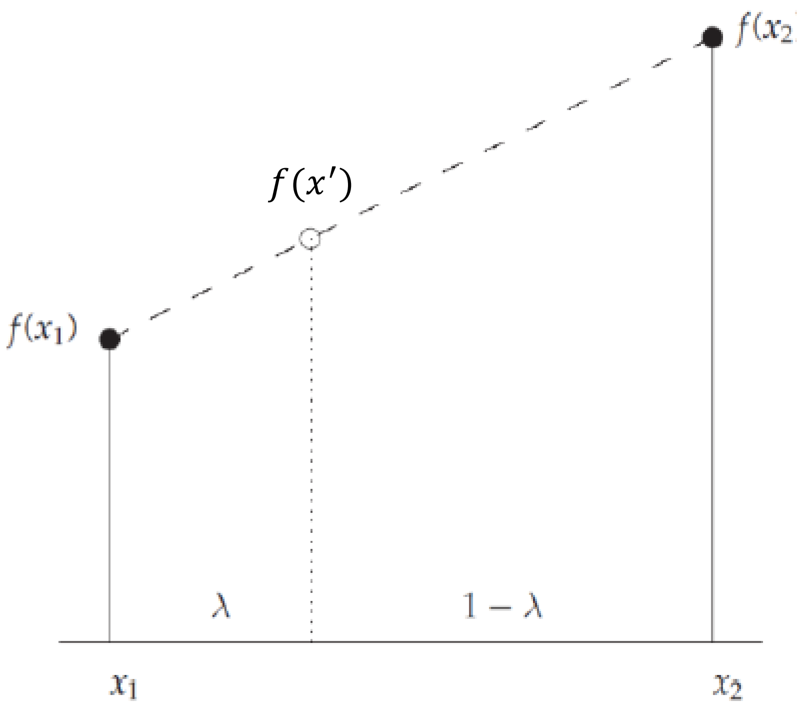
(b)

General Interpolation

- More general form for interpolation

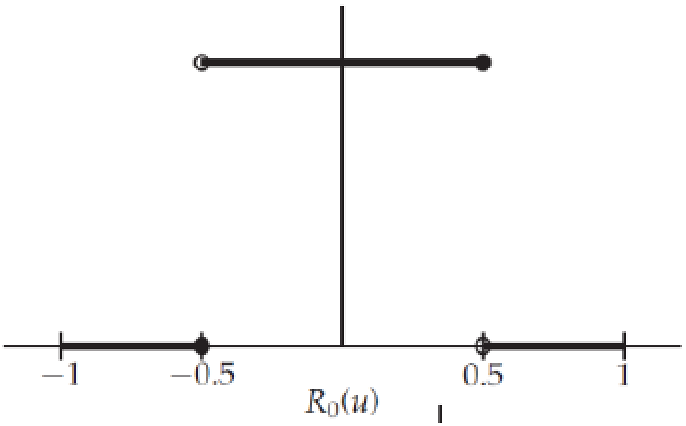
$$f(x') = R(-\lambda)\, f(x_1) + R(1-\lambda) f(x_2)$$

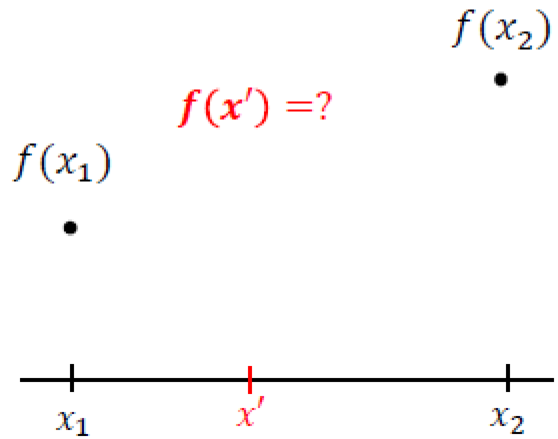




Linear interpolation

Nearest Neighbor Interpolation





$$R_0(u) = \begin{cases} 1 & if \; -0.5 < u \leq 0.5 \\ 0 & otherwise \end{cases}$$

$$x' - x_1 = \lambda$$
$$x_2 - x' = 1 - \lambda$$

$$f(x') = R_0(-\lambda)\, f(x_1) + R_0(1-\lambda) f(x_2)$$

$$\Rightarrow \quad f(x') = \begin{cases} f(x_1) & if \; \lambda < 0.5 \\ f(x_2) & otherwise \end{cases}$$

Linear Interpolation

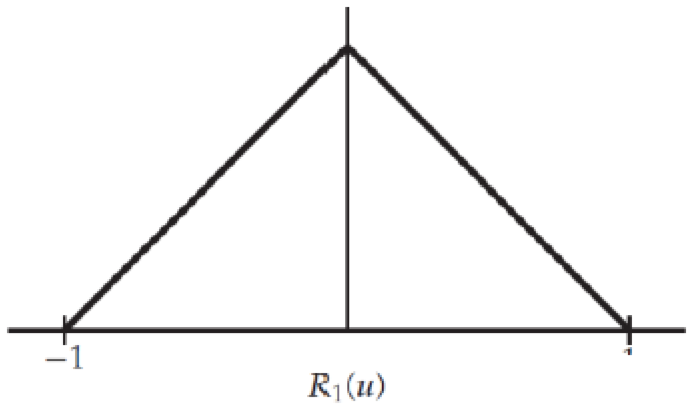
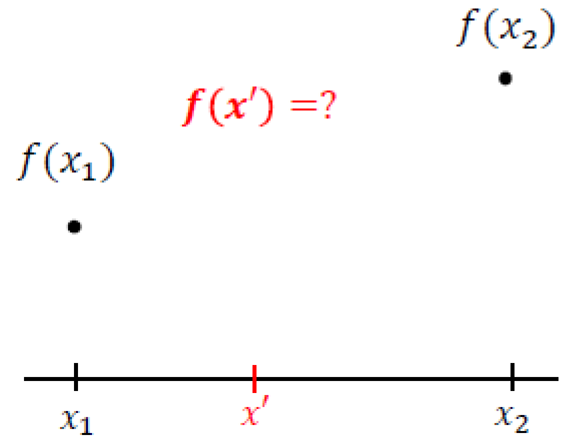


FIGURE 6.12 *Two interpolation functions.*

$$R_1(u) = \begin{cases} 1 + u & if\ u \leq 0 \\ 1 - u & otherwise \end{cases}$$



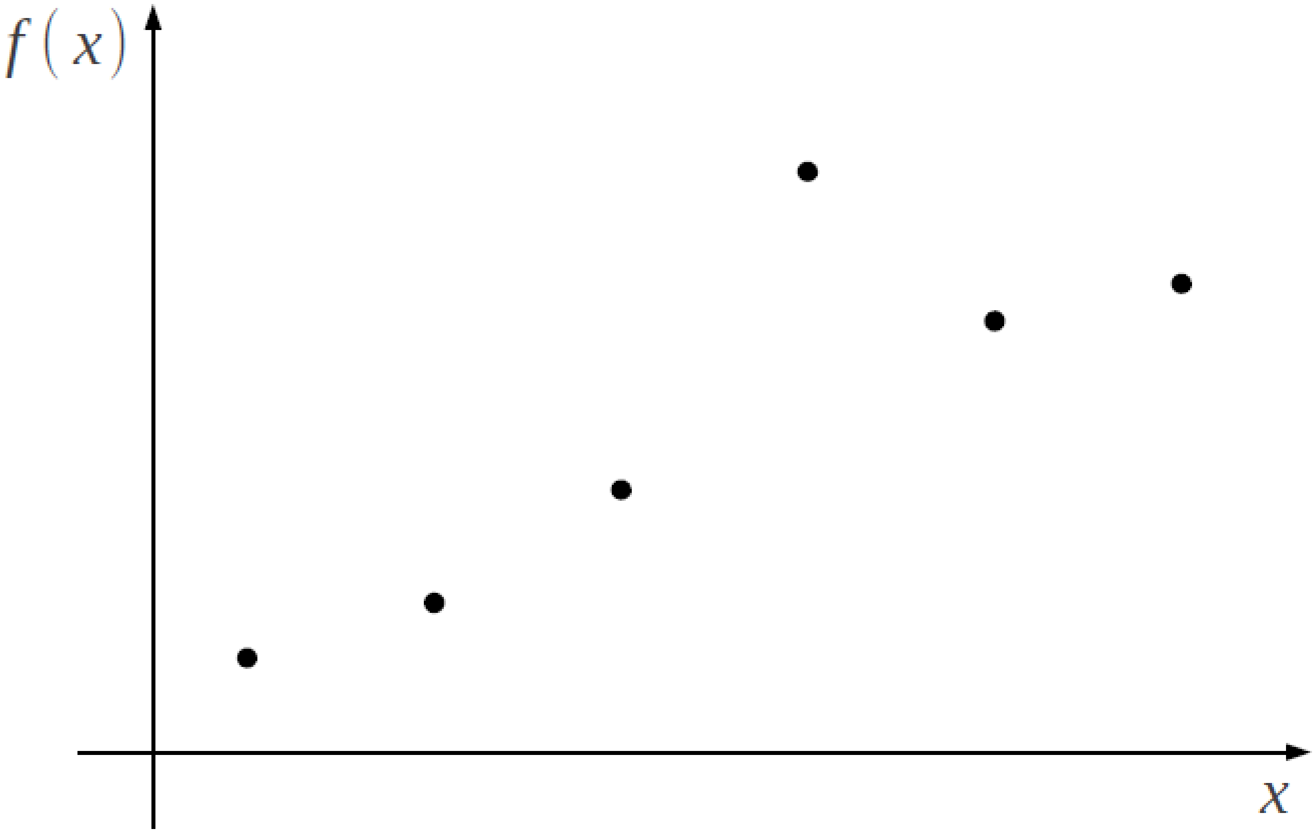
$$x' - x_1 = \lambda$$
$$x_2 - x' = 1 - \lambda$$

$$f(x') = R_1(-\lambda)\, f(x_1) + R_1(1-\lambda) f(x_2)$$

$$\Rightarrow \quad f(x') = (1-\lambda)\, f(x_1) + \lambda f(x_2)$$

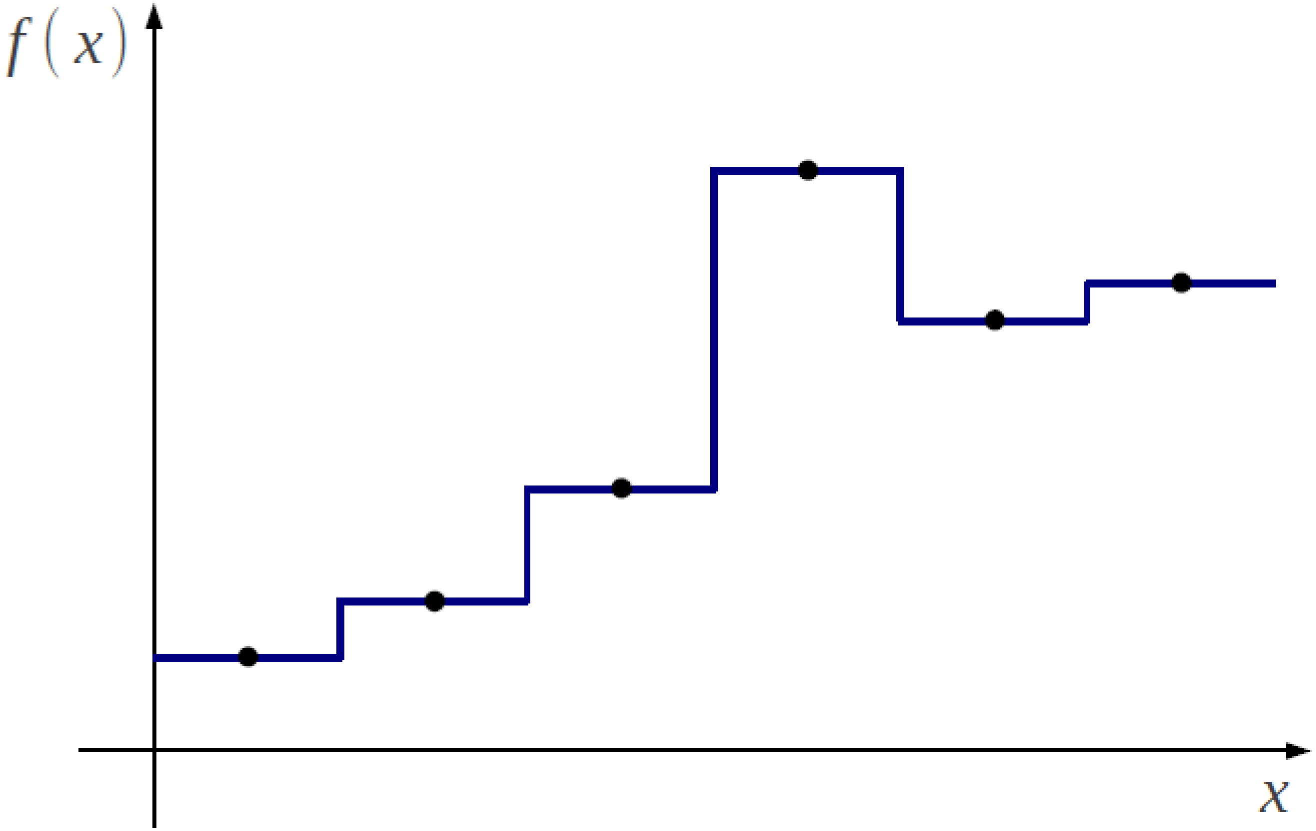
Cubic Interpolation

- Interpolation



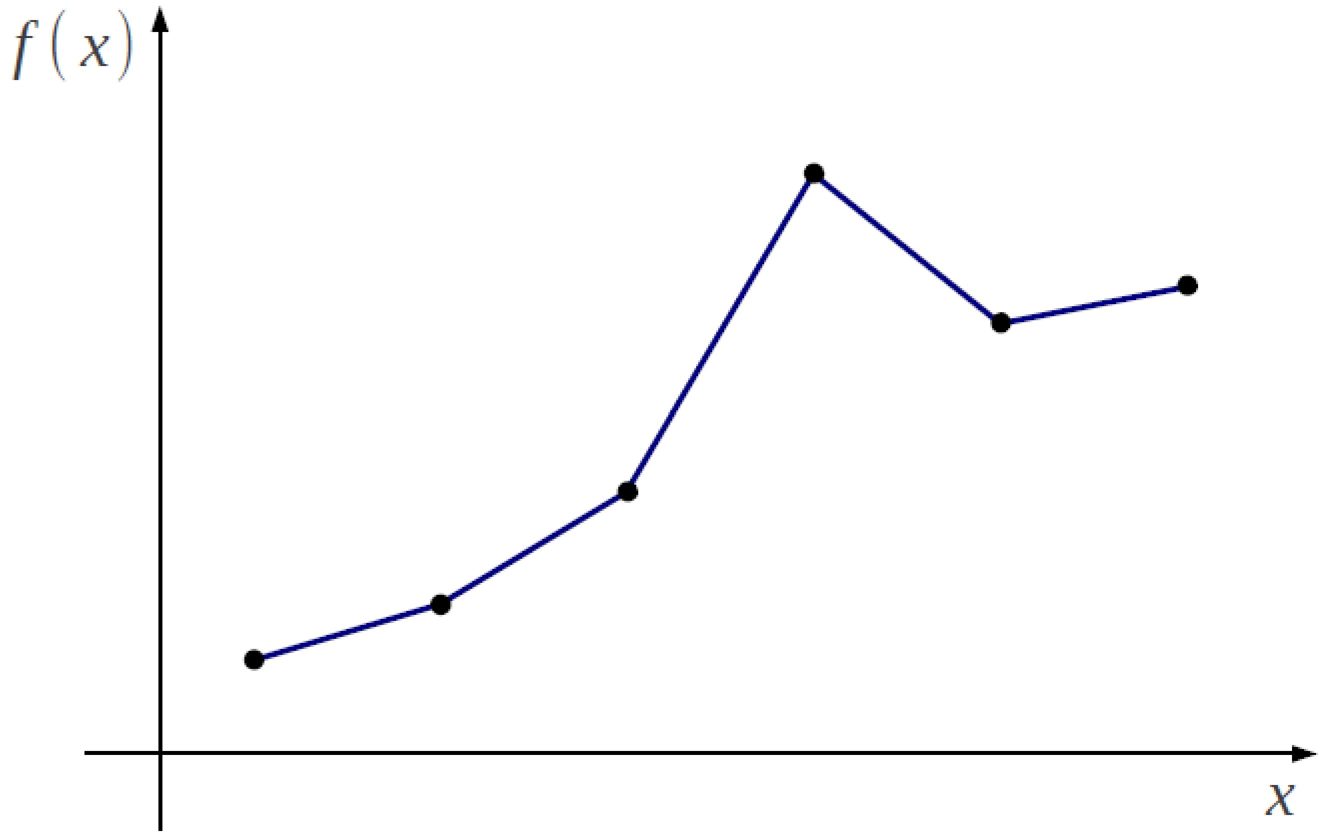
Cubic Interpolation

- Nearest neighbor interpolation



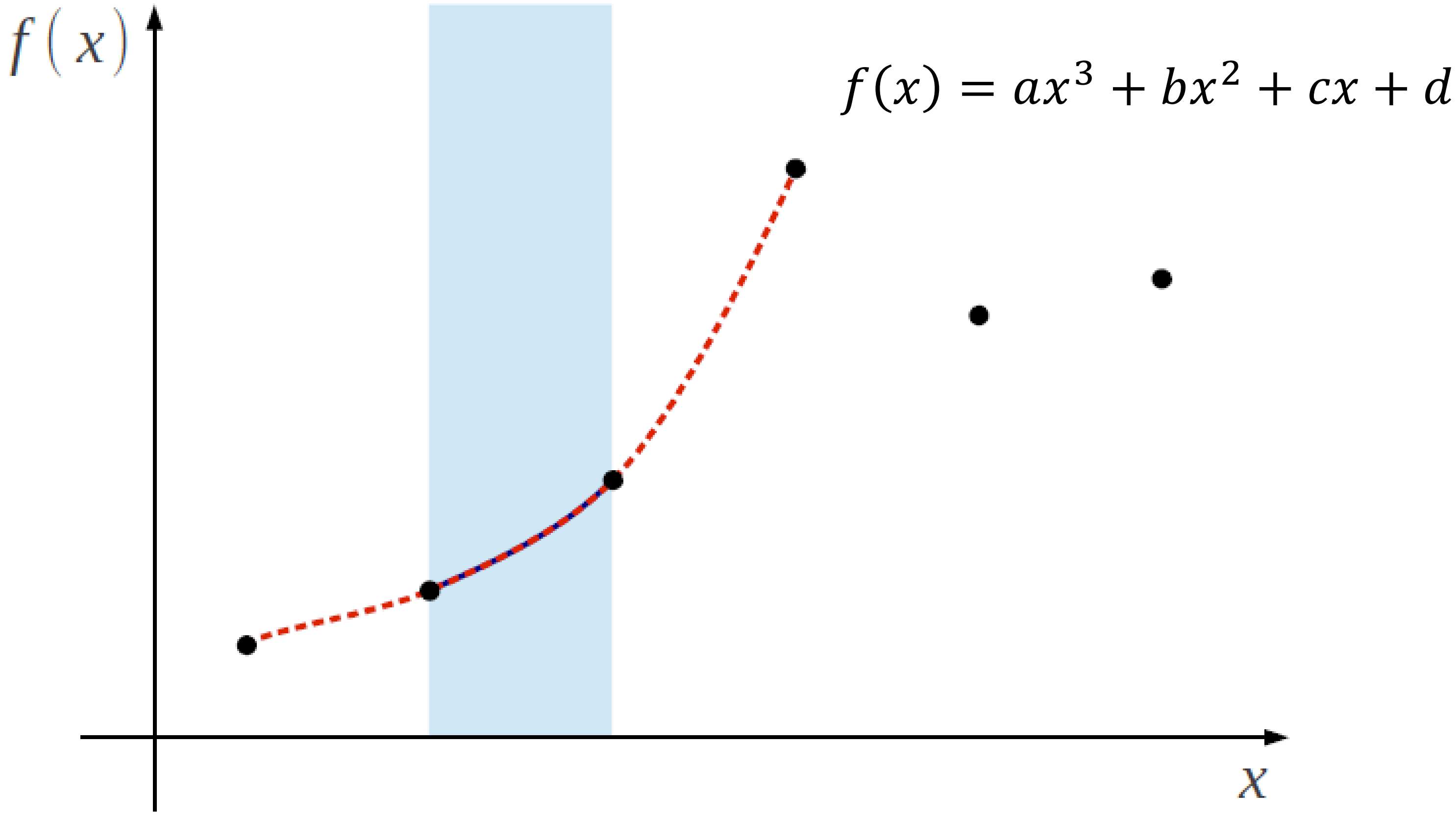
Cubic Interpolation

- Linear interpolation



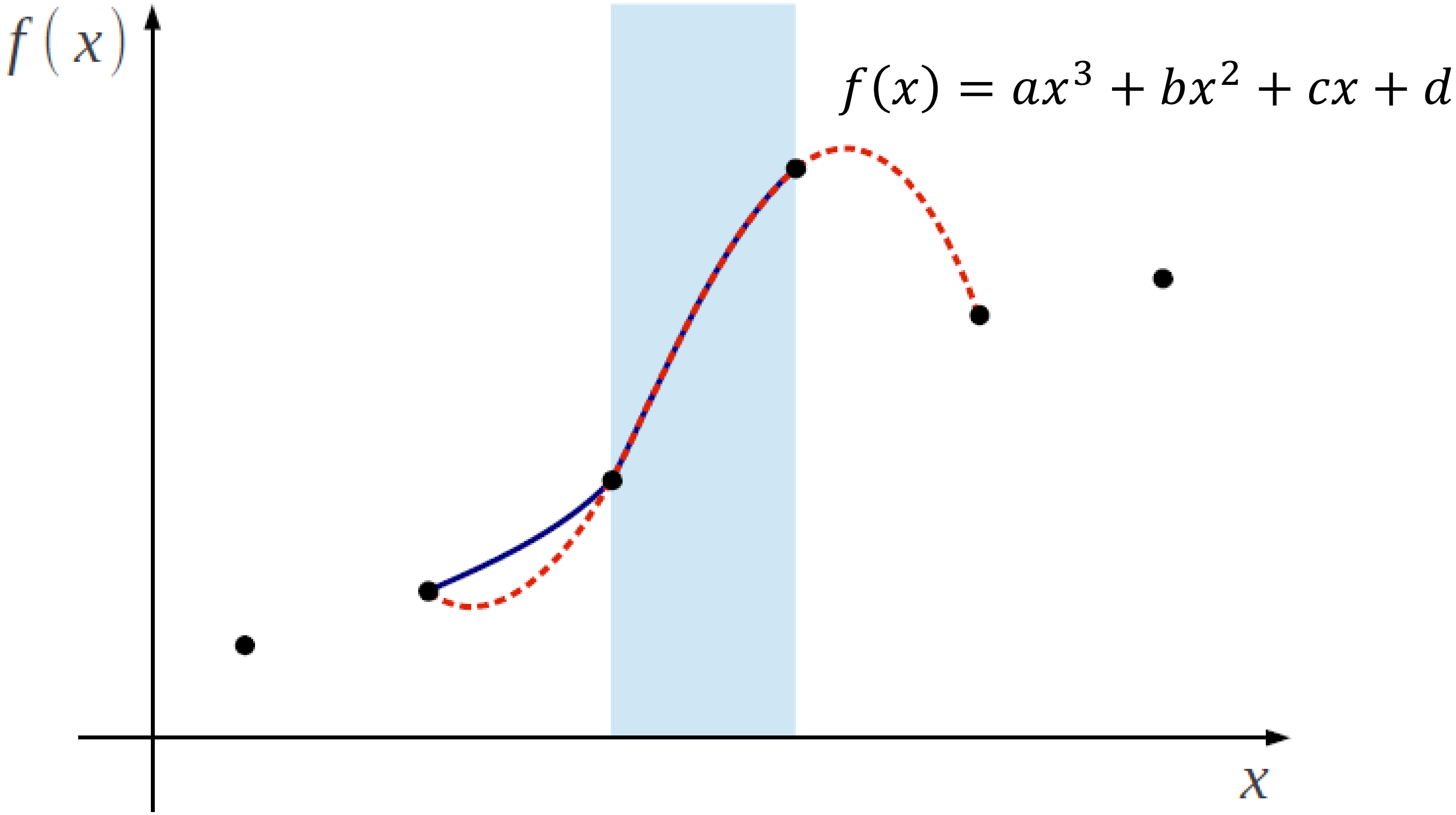
Cubic Interpolation

- Cubic interpolation



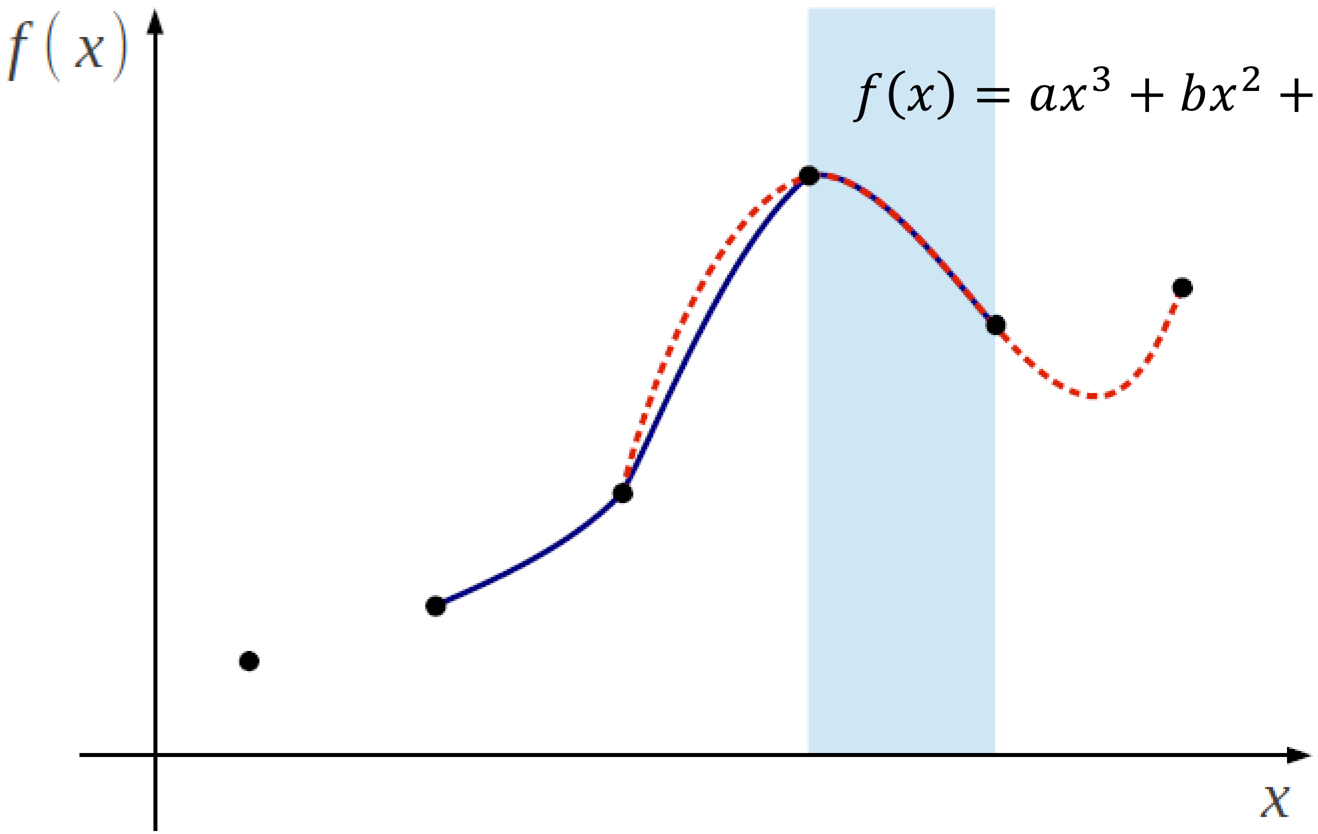
Cubic Interpolation

- Cubic interpolation



Cubic Interpolation

- Cubic interpolation



Cubic Interpolation

$$R_3(u) = \begin{cases} 1.5|u|^3 - 2.5|u|^2 + 1 & if\ |u| \le 1 \\ -0.5|u|^3 + 2.5|u|^2 - 4|u| + 2 & if\ 1 < |u| \le 2 \end{cases}$$

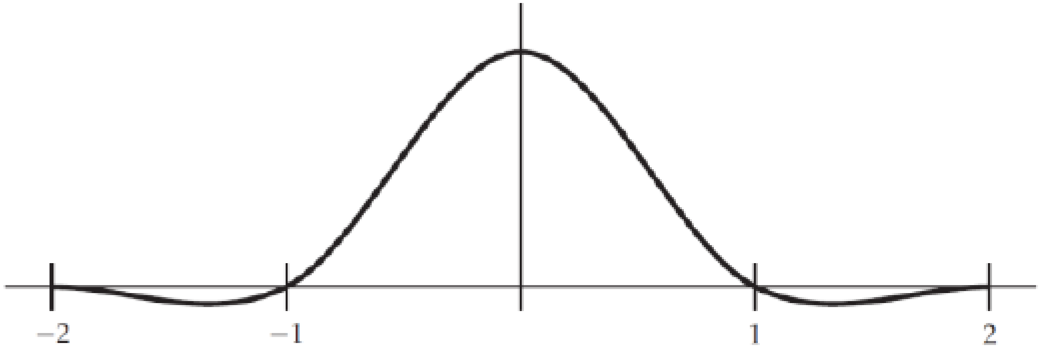


FIGURE 6.13 *The cubic interpolation function $R_3(u)$.*

Q: What is the difference from R_0 and R_1?

Cubic Interpolation

$$f(x') = R_3(-1-\lambda)f(x_1) + R_3(-\lambda)f(x_2) + R_3(1-\lambda)f(x_3) + R_4(2-\lambda)f(x_4),$$

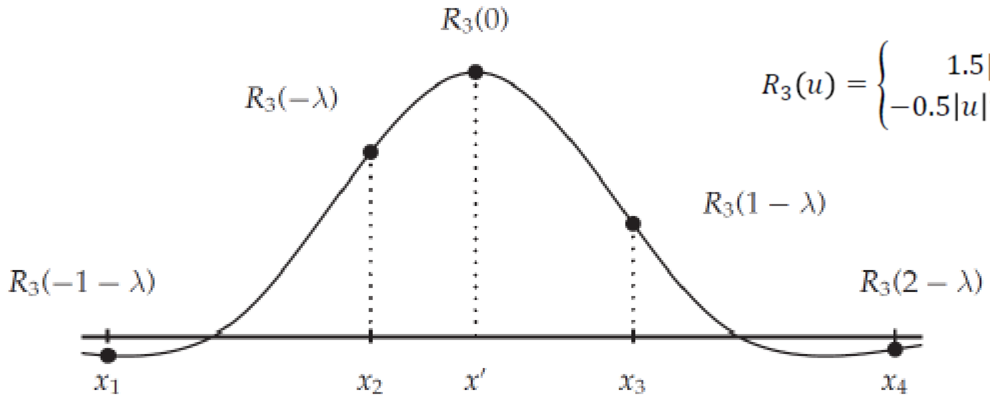
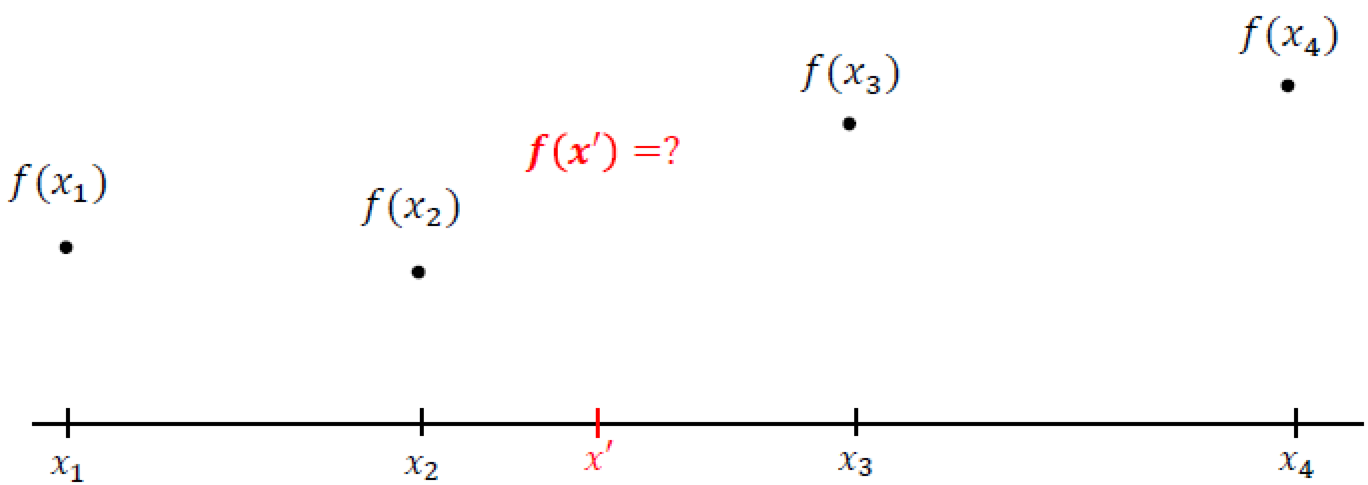


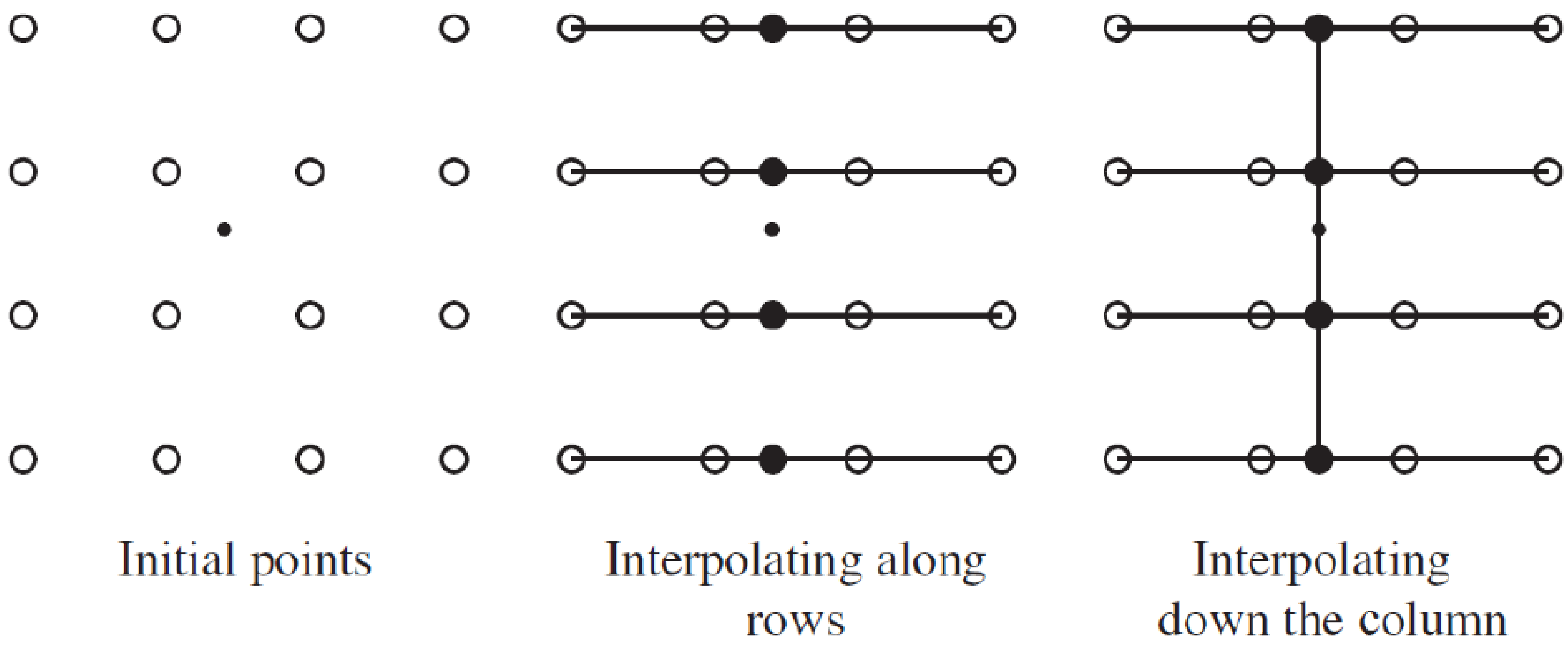
FIGURE 6.14 *Using $R_3(u)$ for interpolation.*

$$R_3(u) = \begin{cases} 1.5|u|^3 - 2.5|u|^2 + 1 & \text{if } |u| \le 1 \\ -0.5|u|^3 + 2.5|u|^2 - 4|u| + 2 & \text{if } 1 < |u| \le 2 \end{cases}$$

$$x' - x_2 = \lambda$$
$$x_3 - x' = 1 - \lambda$$



Bicubic Interpolation



Nearest Neighbor

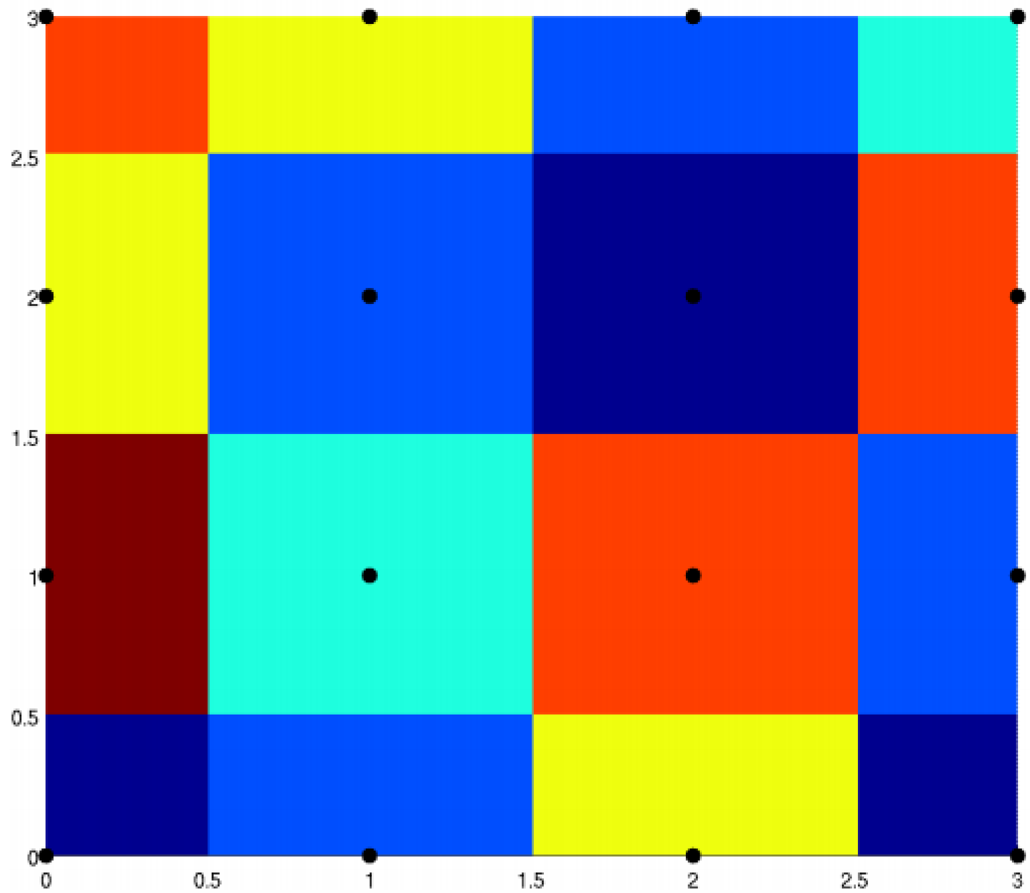


Figure : Nearest Neighbour Interpolation

Bilinear vs. Bicubic

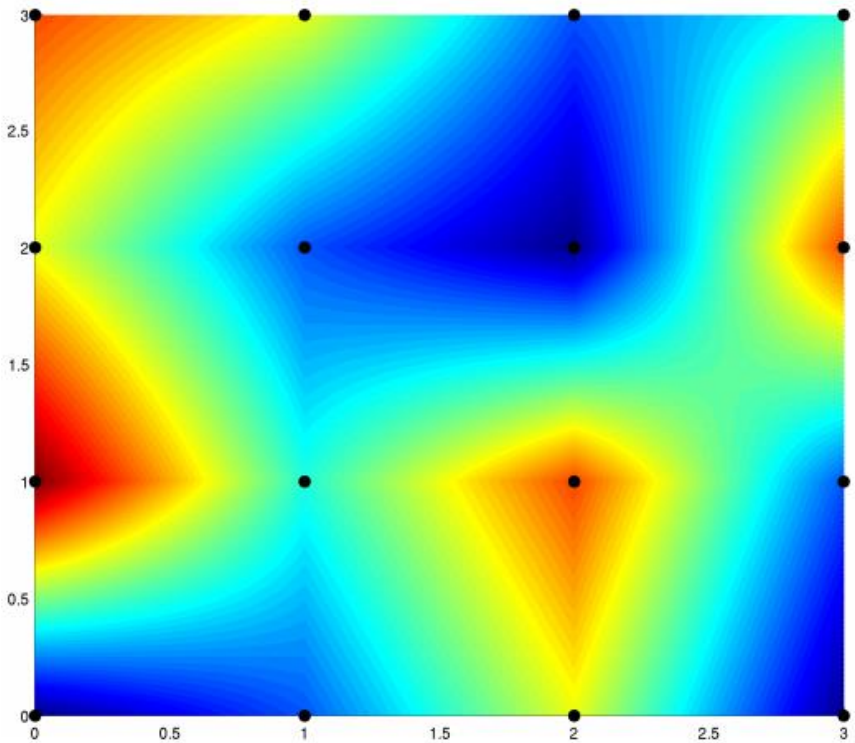


Figure : Bilinear Interpolation

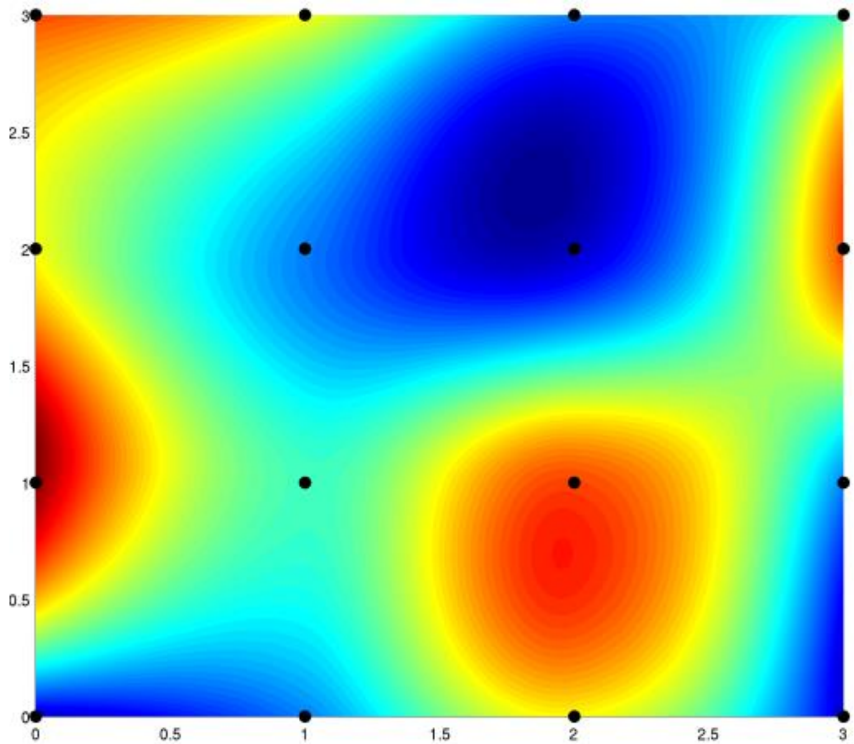


Figure : Bicubic Spline Interpolation

Interpolation

Nearest Neighbor vs. Bilinear vs. Bicubic interpolation

Interpolation with Filtering

- Interpolation ≈ Filtering
- 2 times interpolation
  - $H \times W$ image → $2H \times 2W$ image

1) Generate zero-interleaved image

```
>> m=magic(4)

m =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
>> m2=zeroint(m)

m2 =

    16     0     2     0     3     0    13     0
     0     0     0     0     0     0     0     0
     5     0    11     0    10     0     8     0
     0     0     0     0     0     0     0     0
     9     0     7     0     6     0    12     0
     0     0     0     0     0     0     0     0
     4     0    14     0    15     0     1     0
     0     0     0     0     0     0     0     0
```

2) Replace the zeros by applying a spatial filter

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \frac{1}{64}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Interpolation with Filtering

- Interpolation ≈ Filtering
- 2 times interpolation
  - $H \times W$ image → $2H \times 2W$ image

```
>> m=magic(4)

m =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

1) Generate zero-interleaved image

```
>> m2=zeroint(m)

m2 =

    16     0     2     0     3     0    13     0
     0     0     0     0     0     0     0     0
     5     0    11     0    10     0     8     0
     0     0     0     0     0     0     0     0
     9     0     7     0     6     0    12     0
     0     0     0     0     0     0     0     0
     4     0    14     0    15     0     1     0
     0     0     0     0     0     0     0     0
```

2) Replace the zeros by applying a spatial filter

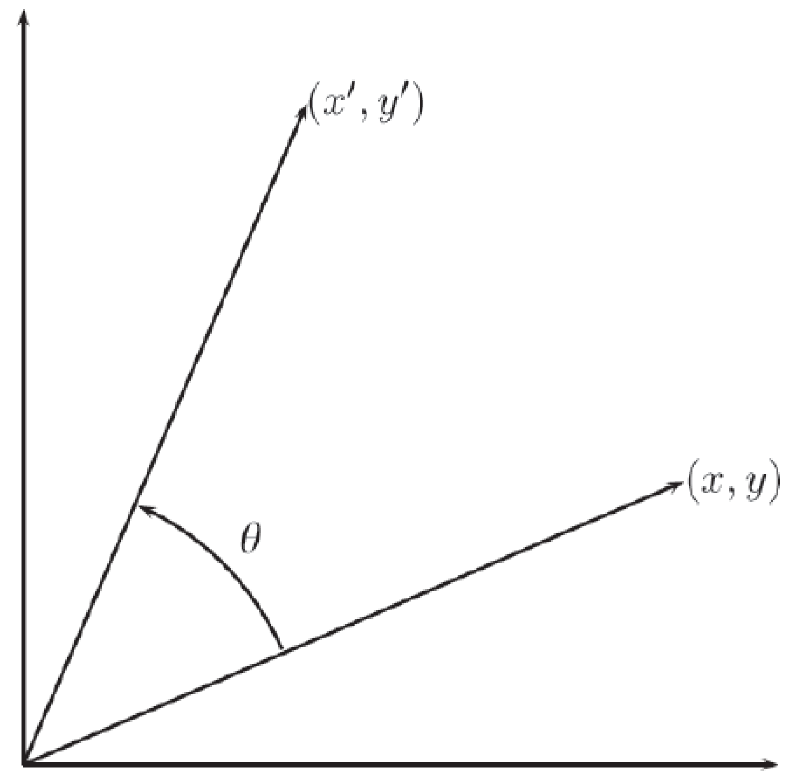
$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \frac{1}{64}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Rotation

- In a continuous signal, the rotation can be done using the following equation

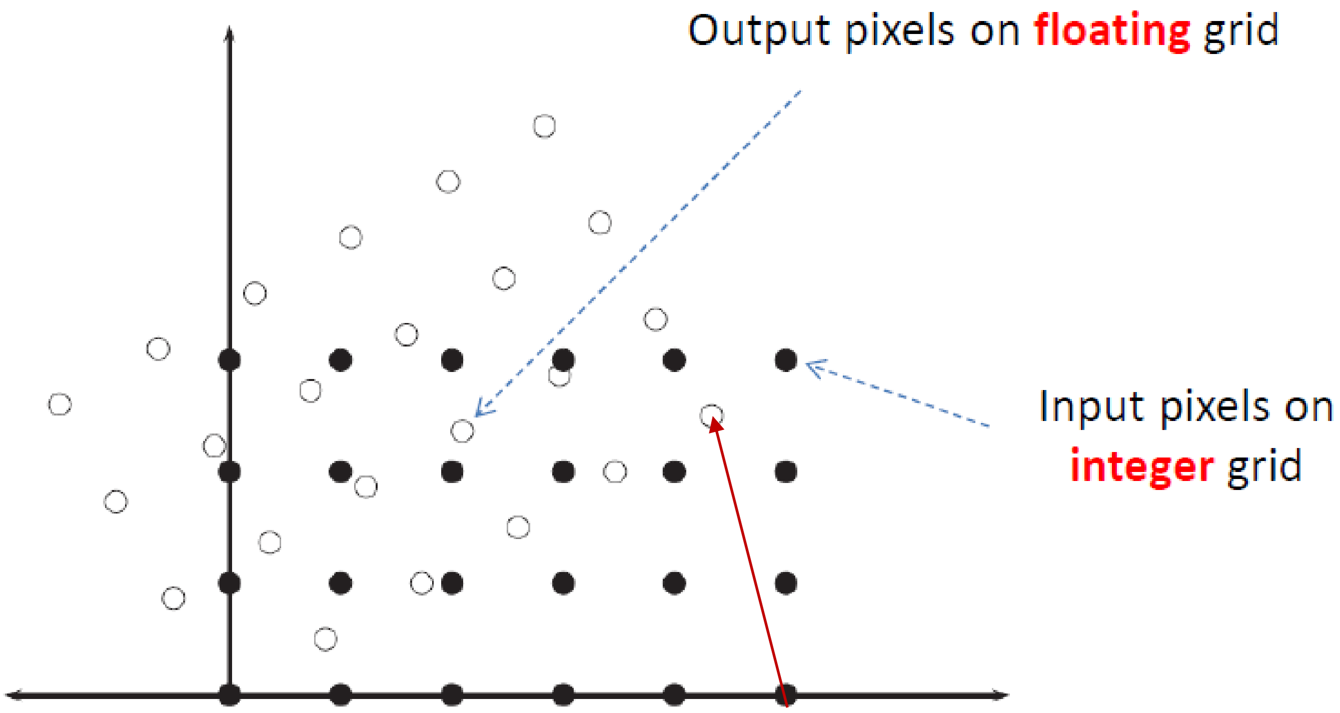
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

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



Rotation

- Discrete signal?
  - Rotation does not always produce an integer pixel location
  - So, the interpolation (or re-sampling) is needed

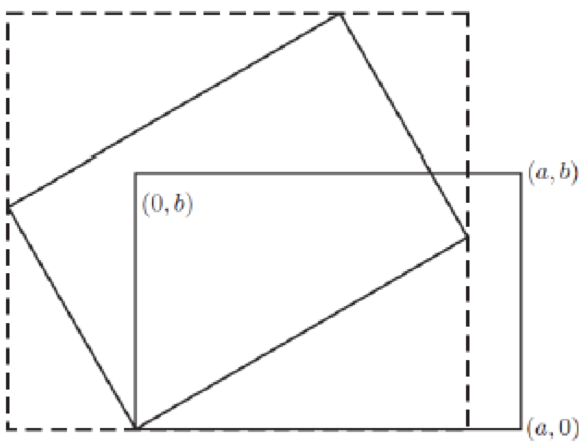


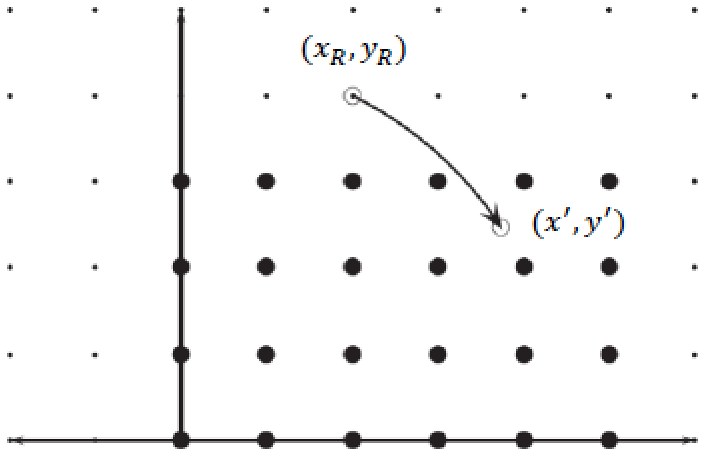
Rotation

- Define an enlarged, rotated image I_R whose size is larger than that of an original image
- Assume $(x_R, y_R) \in I_R$. Then, compute the transform $(x_R, y_R) \rightarrow (x', y')$ Note that (x', y') is a floating pixel

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

- Compute $f(x', y')$ using the interpolation technique
- Then $f(x_R, y_R) = f(x', y')$





Rotation in MATLAB

```
>> cr=imrotate(c,60);
>> imshow(cr)
>> crc=imrotate(c,60,'bicubic');
>> imshow(crc)
```

(a)

(b)

Rotation

- In a continuous signal, the rotation can be done using the following equation

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

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

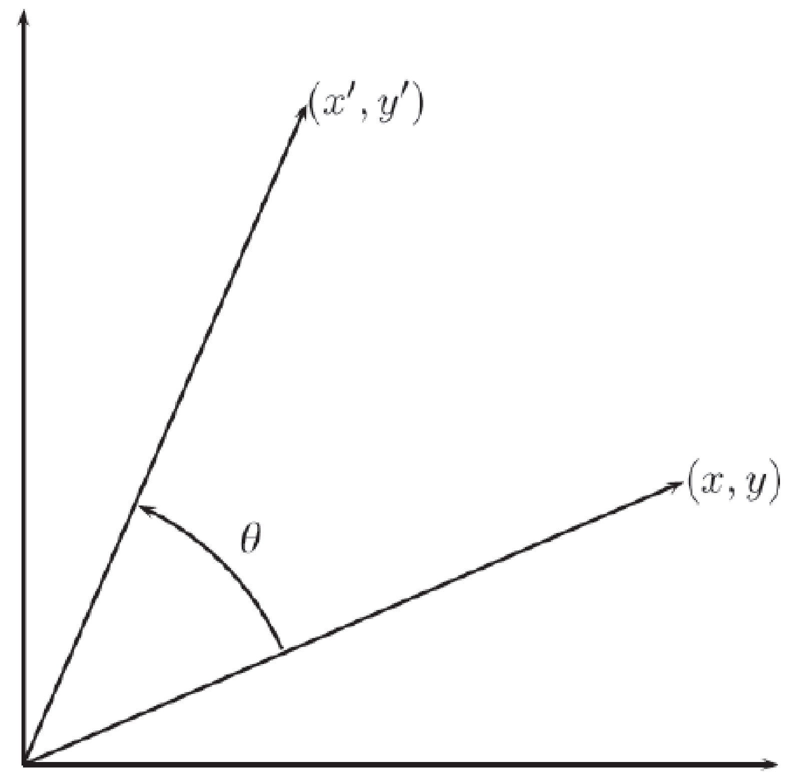


Image Warping

- Parametric (global) warping

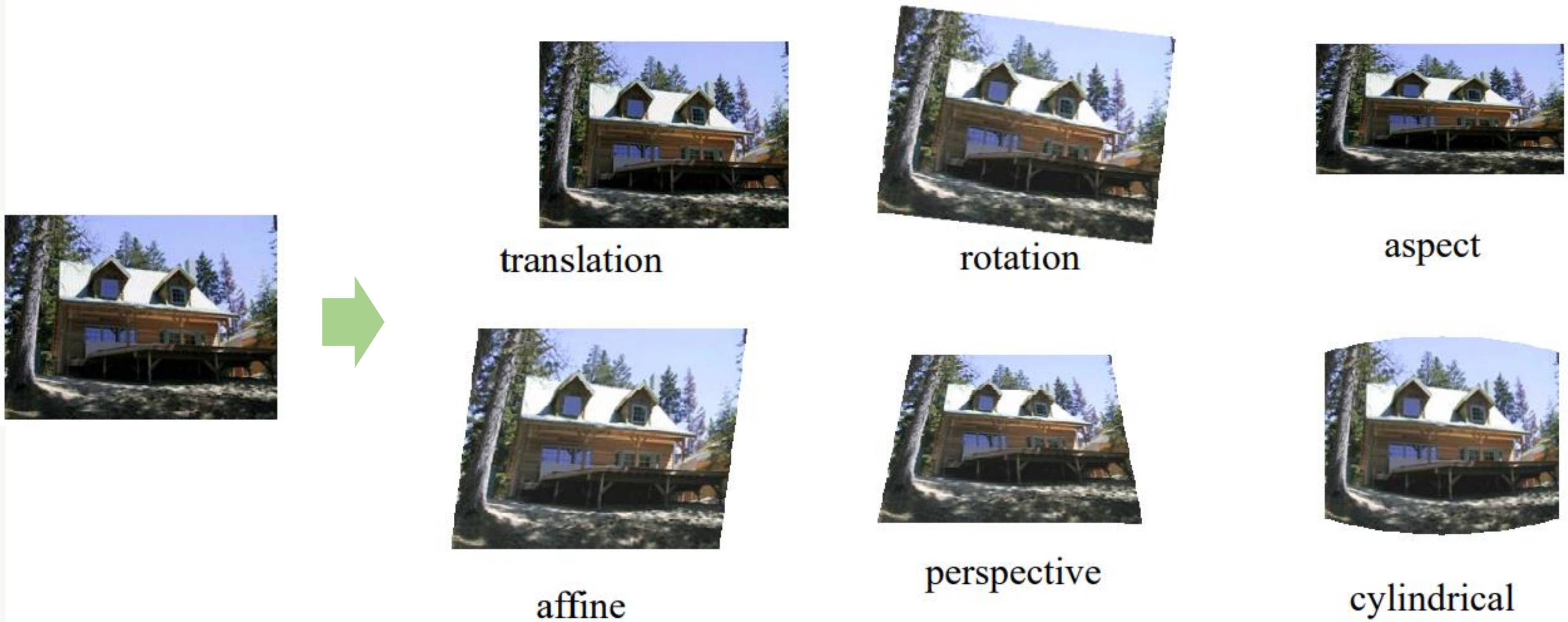
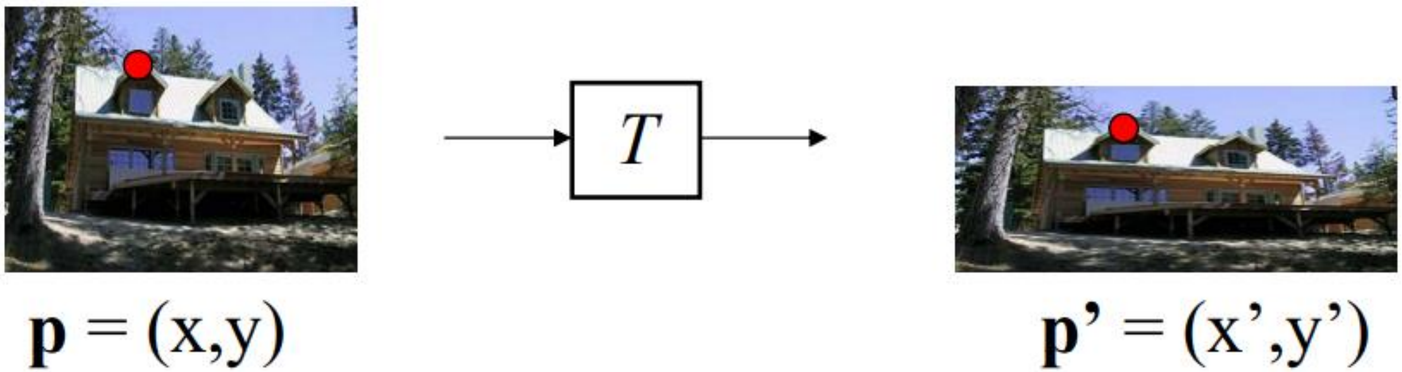


Image Warping

- Parametric (global) warping

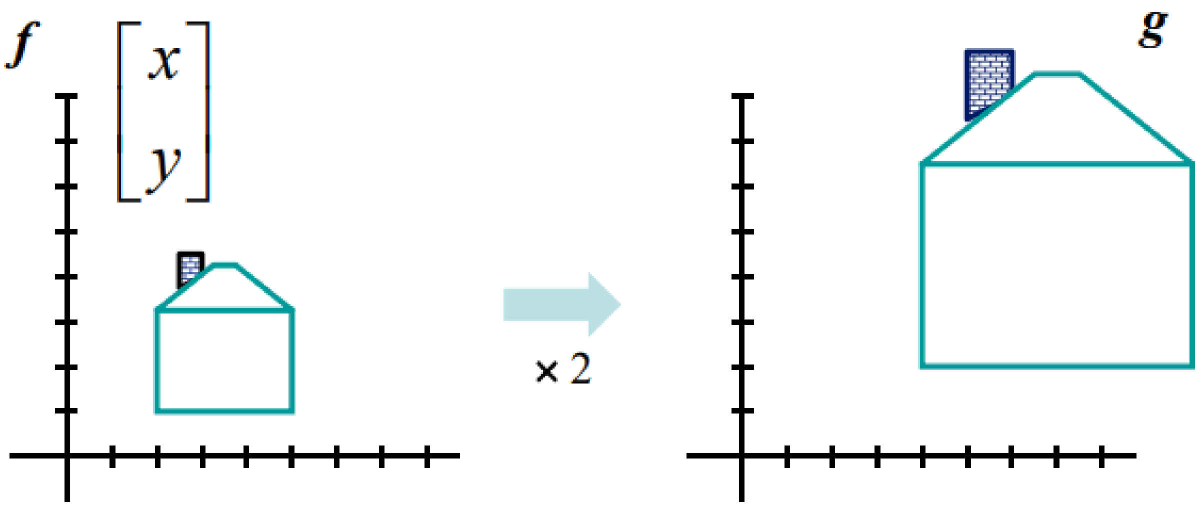


  - Transform T is a coordinate-changing machine: $\mathbf{p}' = T(\mathbf{p})$
  - What does it mean that T is global?
    - Is the same for any point $\mathbf{p}$
  - Represent T as a matrix: $\mathbf{p}' = \mathbf{M}\mathbf{p}$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \end{bmatrix}$$

Parametric Warping

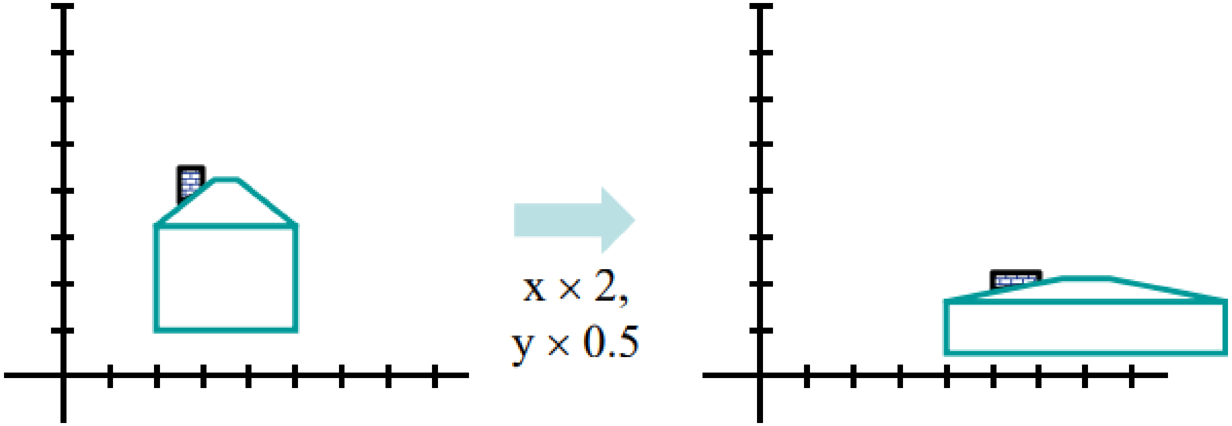
- Scaling
  - Uniform scaling



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

Parametric Warping

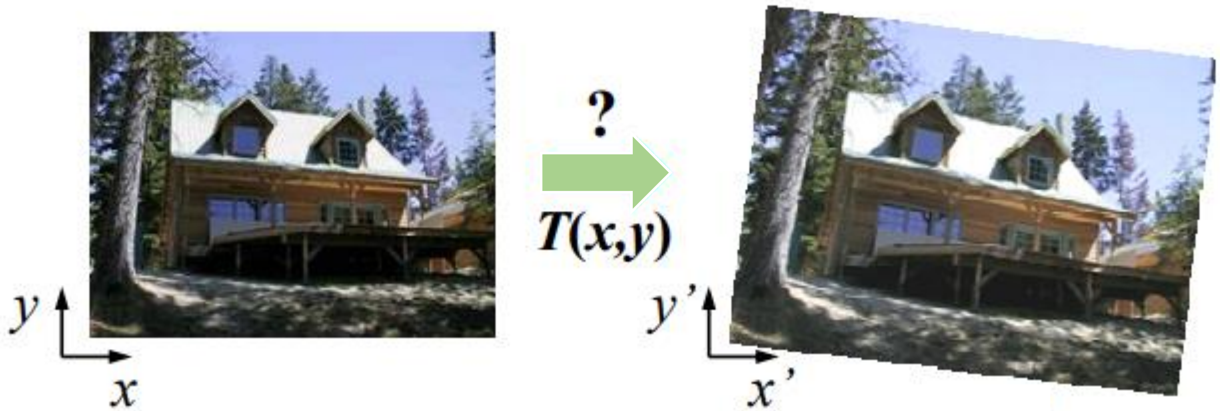
- Scaling
  - Non-uniform scaling: different scalars per component



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x \\ 0.5y \end{bmatrix} \Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

Parametric Warping

- Rotation



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

Parametric Warping

- Shear

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & h_x \\ h_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Parametric Warping

- 2×2 Matrices
  - What types of transformation can be represented with a 2×2 matrix?

Scale matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Sx} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

Rotation matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Rx} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} cos\theta x - sin\theta y \\ cos\theta x + sin\theta y \end{bmatrix}$$

Shear matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Hx} = \begin{bmatrix} 1 & h_x \\ h_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + h_x y \\ x h_y + y \end{bmatrix}$$

$\Rightarrow$ x' is a linear combination of x and y
y' is a linear combination of x and y

Parametric Warping

- 2×2 Matrices
  - What types of transformation can be represented with a 2×2 matrix?

Scale matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Sx} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

Rotation matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Rx} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} cos\theta x - sin\theta y \\ cos\theta x + sin\theta y \end{bmatrix}$$

Shear matrix: $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{Hx} = \begin{bmatrix} 1 & h_x \\ h_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + h_x y \\ x h_y + y \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax + by \\ dx + ey \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{Mx}$$

Parametric Warping

- 2×2 Matrices
  - Scaling + Shear + Rotate

$$\mathbf{x}' = \mathbf{R}\left(\mathbf{H}\left((\mathbf{S}\mathbf{x})\right)\right) = \mathbf{M}\mathbf{x}$$

If we have to transform many points, it is easier to do one matrix multiplication

Parametric Warping

- What types of transformation can not be represented with a 2×2 matrix?

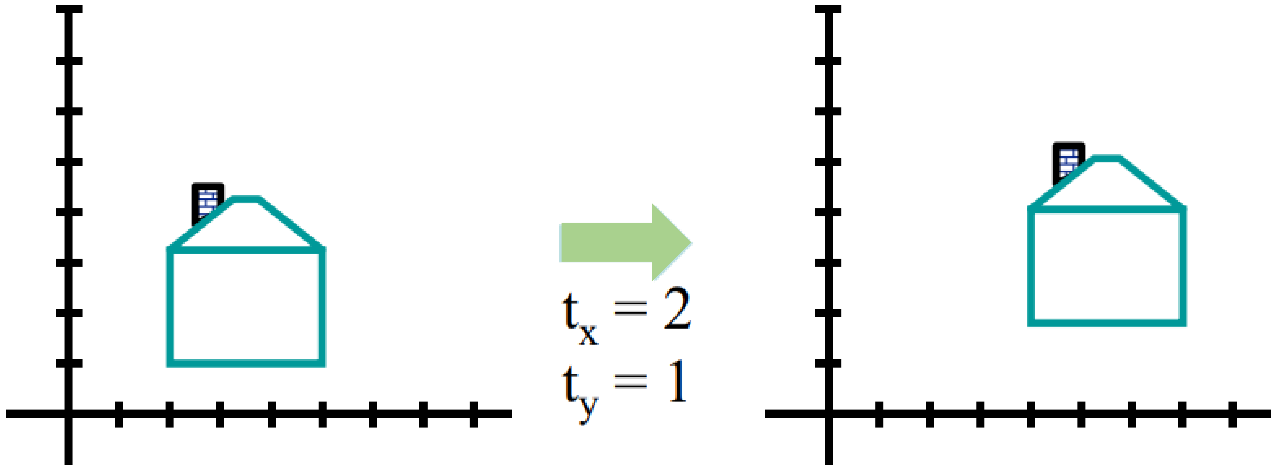
Translation

$$x' = x + t_x$$
$$y' = y + t_y$$

Parametric Warping

- Example of translation

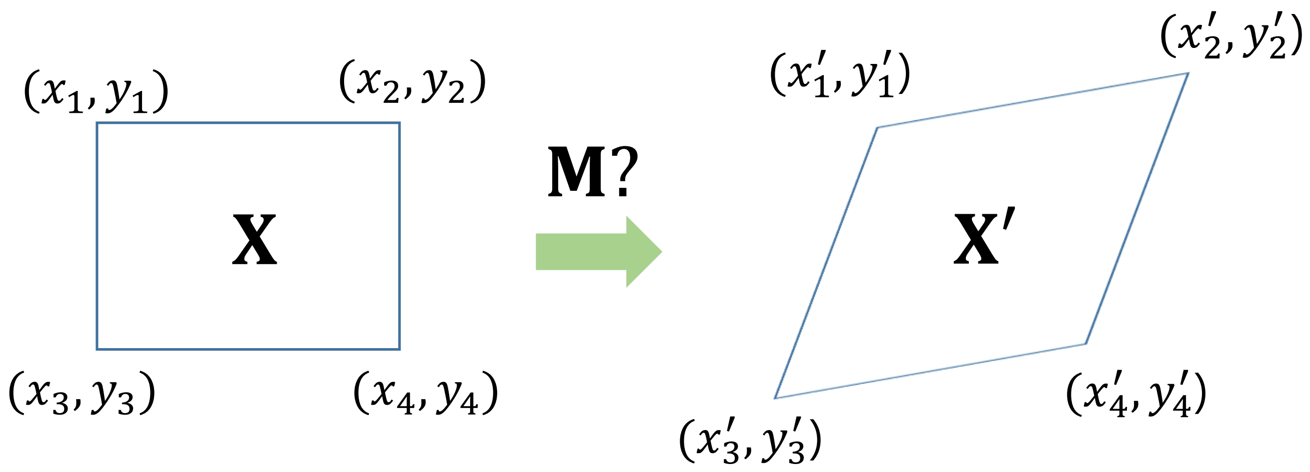
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$



Parametric Warping

- Affine transform
  - Linear transform + translation

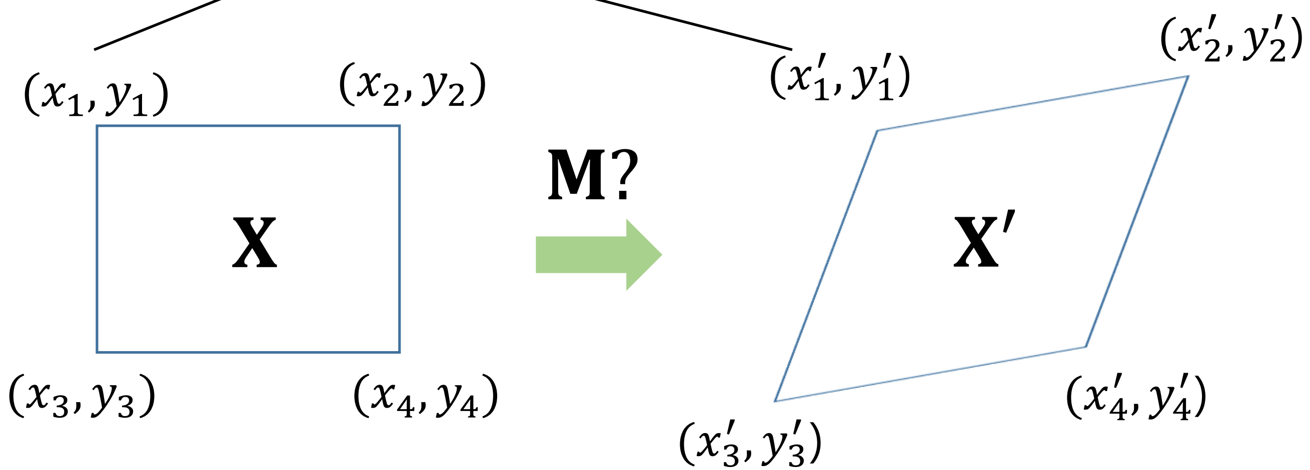
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$



Parametric Warping

- Affine transform
  - Linear transform + translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$



Parametric Warping

- Affine transform

$$x_1' = ax_1 + by_1 + c \qquad x_2' = ax_2 + by_2 + c$$
$$y_1' = dx_1 + ey_1 + f \qquad y_2' = dx_2 + ey_2 + f$$

$$x_3' = ax_3 + by_3 + c \qquad x_4' = ax_4 + by_4 + c$$
$$y_3' = dx_3 + ey_3 + f \qquad y_4' = dx_4 + ey_4 + f$$

Parametric Warping

- Affine transform

$$\boxed{x'_1 = ax_1 + by_1 + c} \qquad x'_2 = ax_2 + by_2 + c$$
$$y'_1 = dx_1 + ey_1 + f \qquad y'_2 = dx_2 + ey_2 + f$$

$$x'_3 = ax_3 + by_3 + c \qquad x'_4 = ax_4 + by_4 + c$$
$$y'_3 = dx_3 + ey_3 + f \qquad y'_4 = dx_4 + ey_4 + f$$

$$\begin{bmatrix} x_1 & y_1 & c & 0 & 0 & 0 \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x'_1 \\ \\ \\ \\ \\ \\ \end{bmatrix}$$

Parametric Warping

- Affine transform

$$x_1' = ax_1 + by_1 + c \qquad x_2' = ax_2 + by_2 + c$$
$$\boxed{y_1' = dx_1 + ey_1 + f} \qquad y_2' = dx_2 + ey_2 + f$$

$$x_3' = ax_3 + by_3 + c \qquad x_4' = ax_4 + by_4 + c$$
$$y_3' = dx_3 + ey_3 + f \qquad y_4' = dx_4 + ey_4 + f$$

$$\begin{bmatrix} x_1 & y_1 & c & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & f \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ \\ \\ \\ \end{bmatrix}$$

Parametric Warping

- Affine transform

$$x_1' = ax_1 + by_1 + c \qquad x_2' = ax_2 + by_2 + c$$
$$y_1' = dx_1 + ey_1 + f \qquad y_2' = dx_2 + ey_2 + f$$

$$x_3' = ax_3 + by_3 + c \qquad x_4' = ax_4 + by_4 + c$$
$$y_3' = dx_3 + ey_3 + f \qquad y_4' = dx_4 + ey_4 + f$$

$$\begin{array}{cc} 8 & \begin{bmatrix} x_1 & y_1 & c & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & f \\ & & & \vdots & & \\ & & & & & \\ & & & & & \end{bmatrix} \end{array} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ \vdots \\ \\ \\ \end{bmatrix} \Rightarrow \mathbf{Ax} = \mathbf{b}$$

Parametric Warping

- Affine transform
  - Least-squares (최소자승법)

$$\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}$$

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$$

Parametric Warping

- Affine transform

$$E(x) = (Ax - b)^T(Ax - b) = \left(x^T A^T - b^T\right)(Ax - b)$$

$$= x^T A^T Ax - x^T A^T b - b^T Ax + b^T b \quad \left(x^T A^T b = b^T Ax\right)$$

$$= x^T A^T Ax - 2b^T Ax + b^T b$$

$$\nabla E(x) = \frac{dE}{dx} = x^T\left(A^T A + A^T A\right) - 2b^T A = 2A^T Ax - 2A^T b$$

$$A^T Ax = A^T b \quad (\textit{normal equations})$$

$$\Rightarrow x^+ = \left(A^T A\right)^+ A^T b$$

$$= \left(A^T A\right)^{-1} A^T b \quad \left(\textit{if the inverse} \left(A^T A\right)^{-1} \textit{exists}\right)$$

$$\boxed{\begin{array}{l} (Ax)^T = x^T A^T \\ \dfrac{d}{dx}(x^T Ax) = x^T(A + A^T) \end{array}}$$

Parametric Warping

- Affine transform