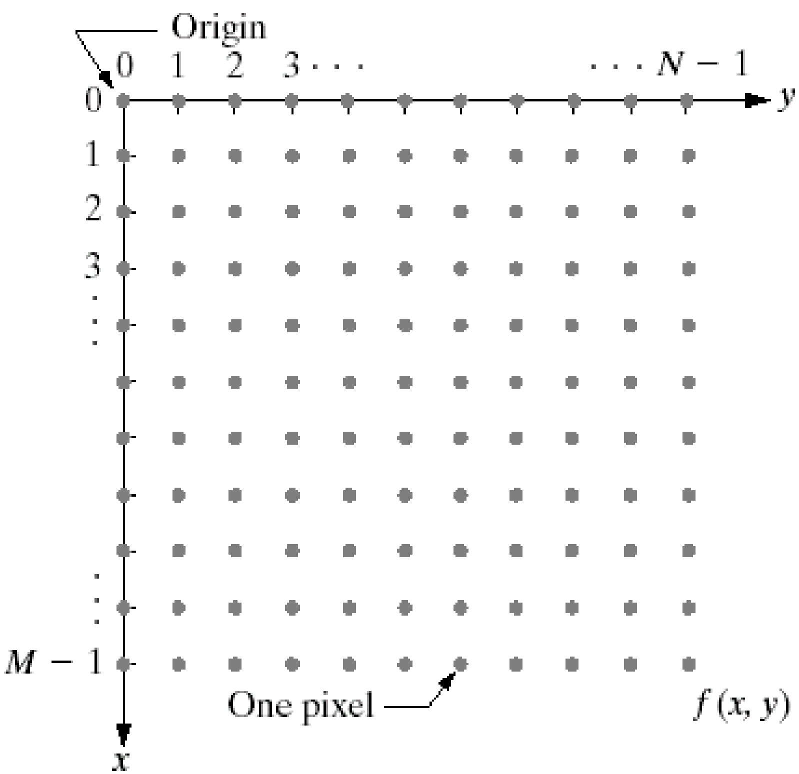
Basics of Digital Imaging

Digital Image Representation

The most standard representation of a digital image, as a grid of **pixels** (picture elements):



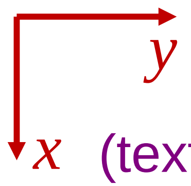
$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}$$

$$\boldsymbol{A}=\begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{0,0} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix}$$

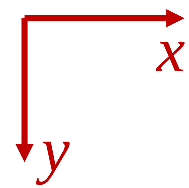
Digital Image Representation

A few things to note about the image coordinate systems:

- Where the origin is.
- Axis designations (be careful when doing implementation):



(textbook)



Used practically
(e.g., in Matlab)

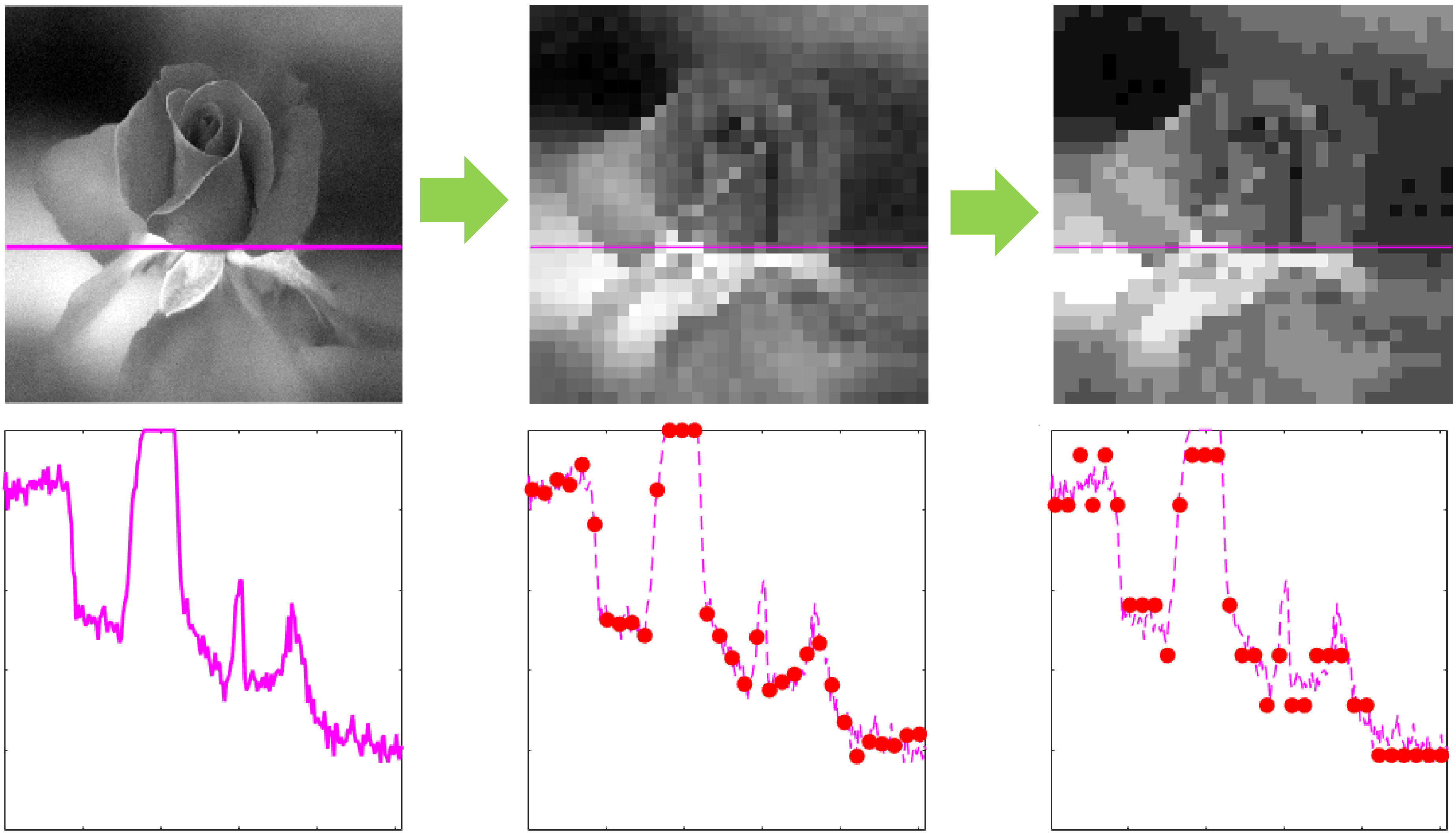
- Representation of multi-valued pixels: The **channel** dimension; $f(x, y, c)$
  - Color images
  - Hyperspectral images
- Third spatial dimension; **voxels** (volume elements); $f(x, y, z)$
- Temporal dimension; videos; $f(x, y, t)$

Sampling and Quantization

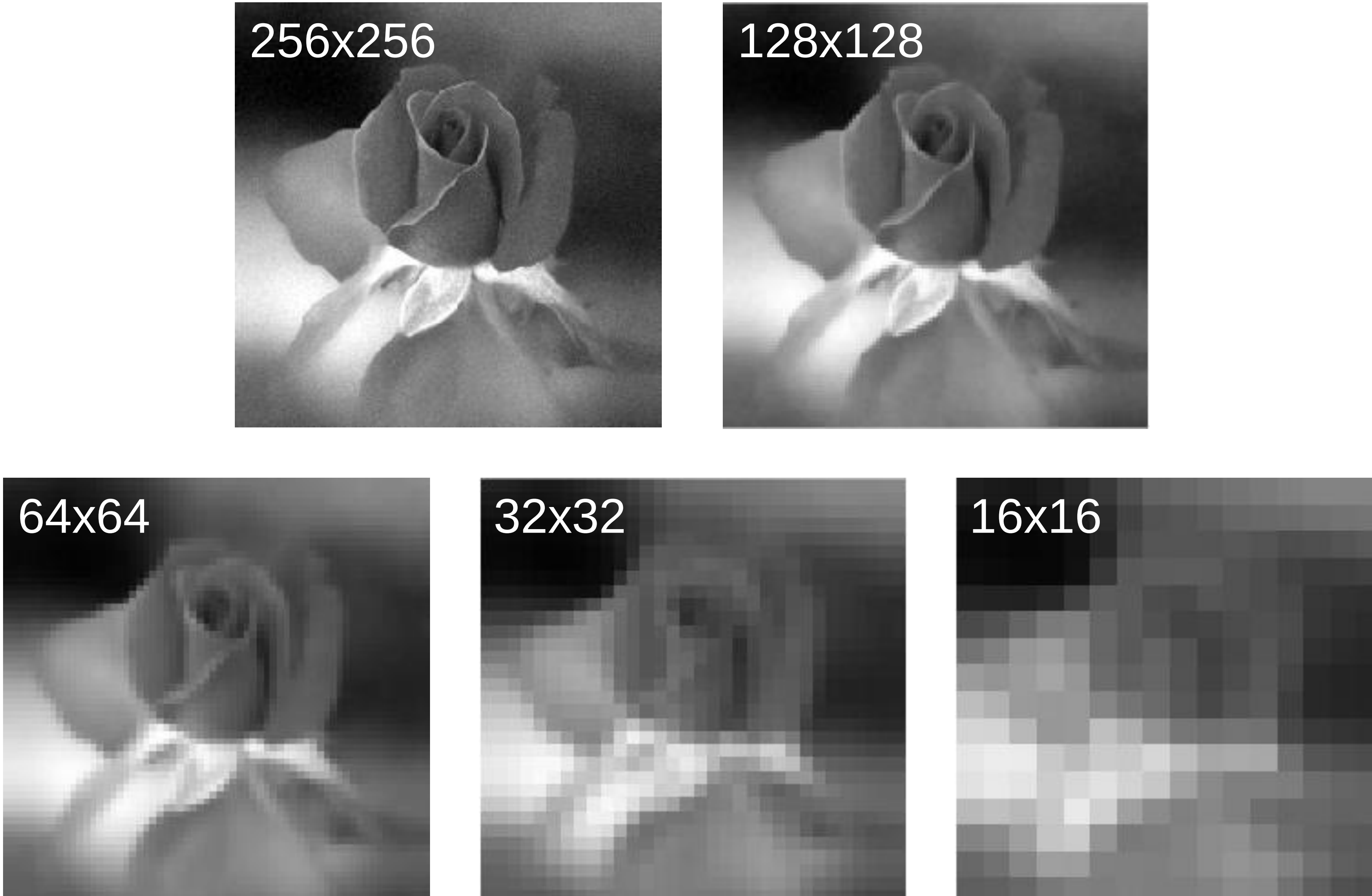
- The process of going from the ***continuous*** real world to the ***discrete*** digital world.
- **Sampling**: Obtaining values at discrete points in the spatial / temporal domains.
- **Quantization**: The representation of the sampled values with a set of discrete values (**quantization levels**).
  - Typically there are 2^k levels (k = bit per pixel).
  - The quantization levels are commonly evenly spaced, but this is not required.

Sampling and Quantization

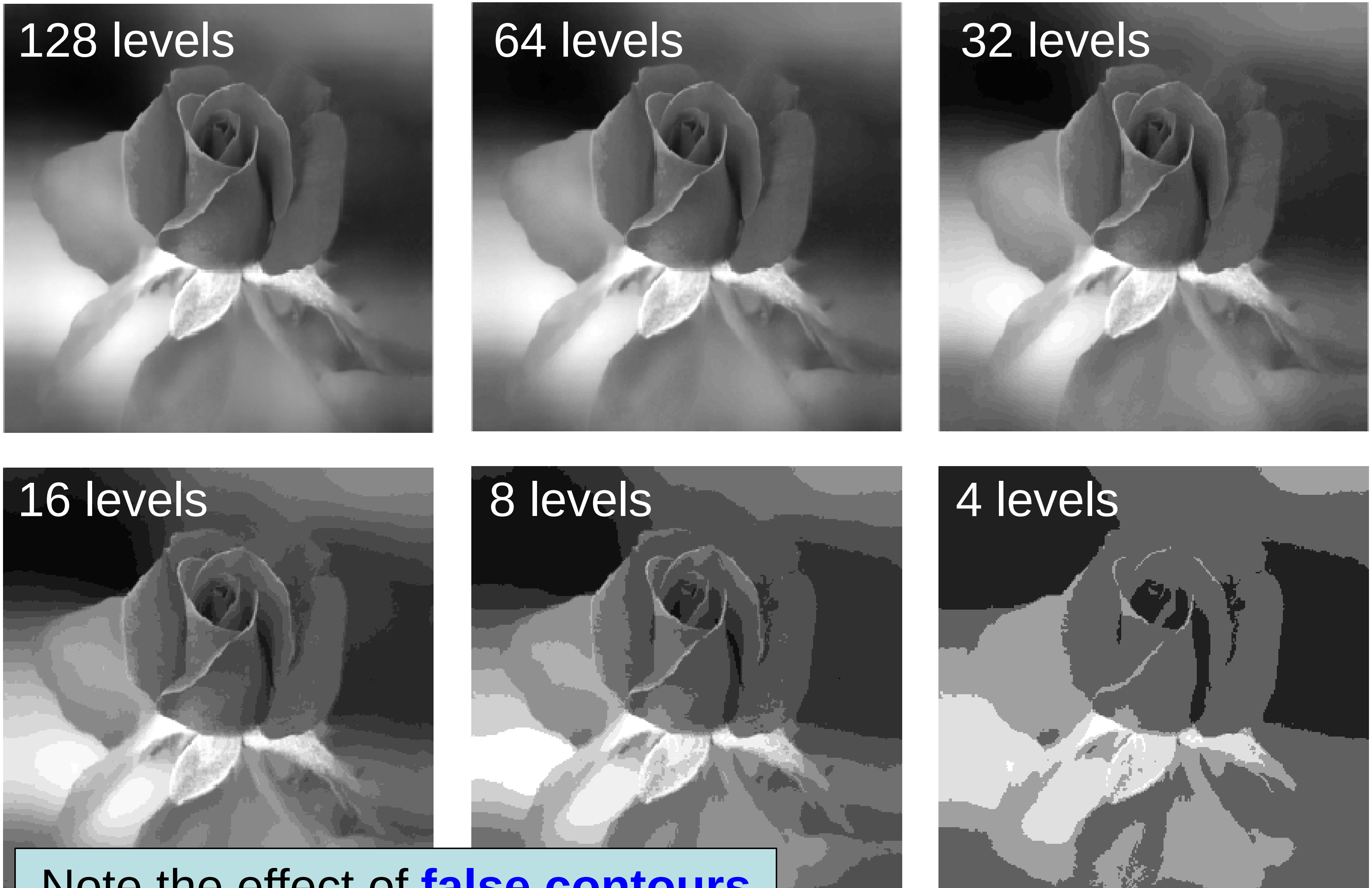
Sampling Quantization



Spatial Resolution



Intensity Resolution



Note the effect of **false contours**

Resolution and Visual Quality

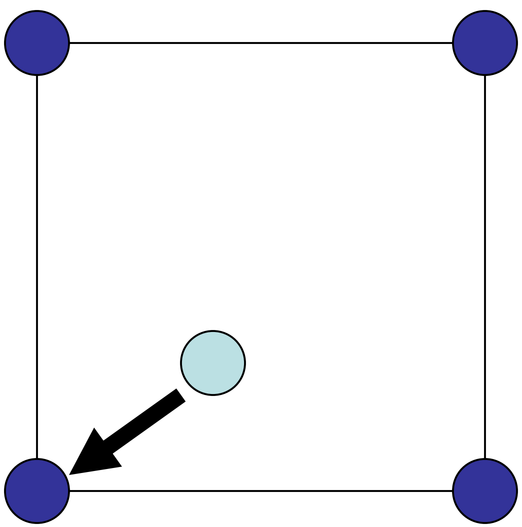
Compare these two images and assess the visual qualities.
How do the two regions differ?

Image Zooming

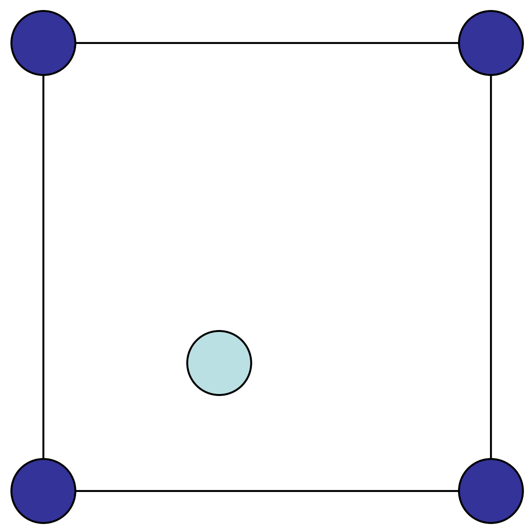
Goal: To show the same image at a different size.

Interpolation is necessary for obtaining the values of pixels on the new grid from the values of the original grid. Common methods are:

Nearest-neighbor



Bilinear



$$v(x', y') = ax' + by' + cx'y' + d$$

Bicubic

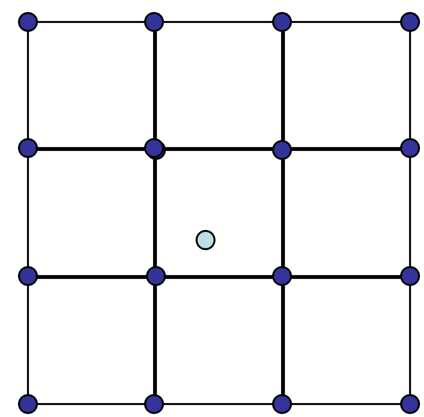
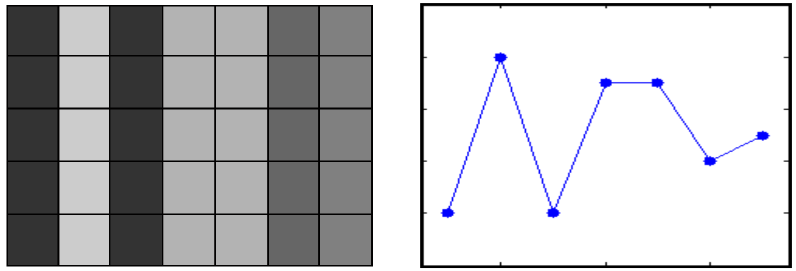
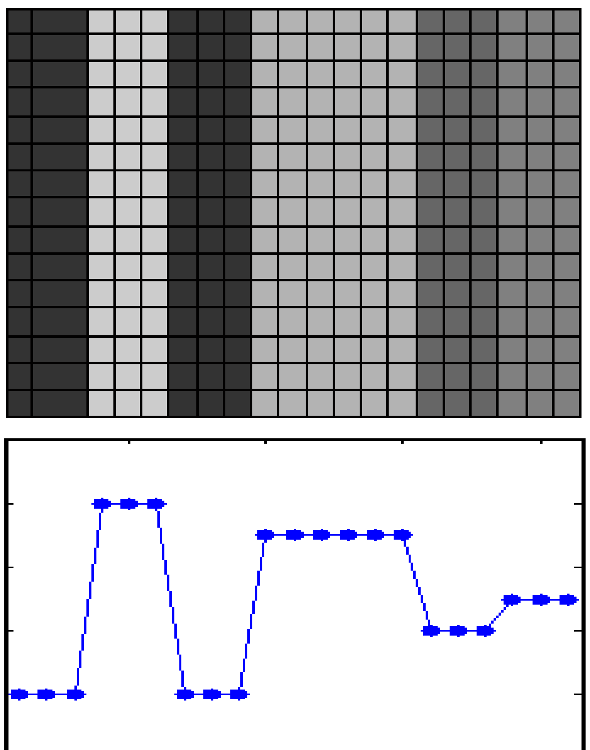


Image Interpolation Methods

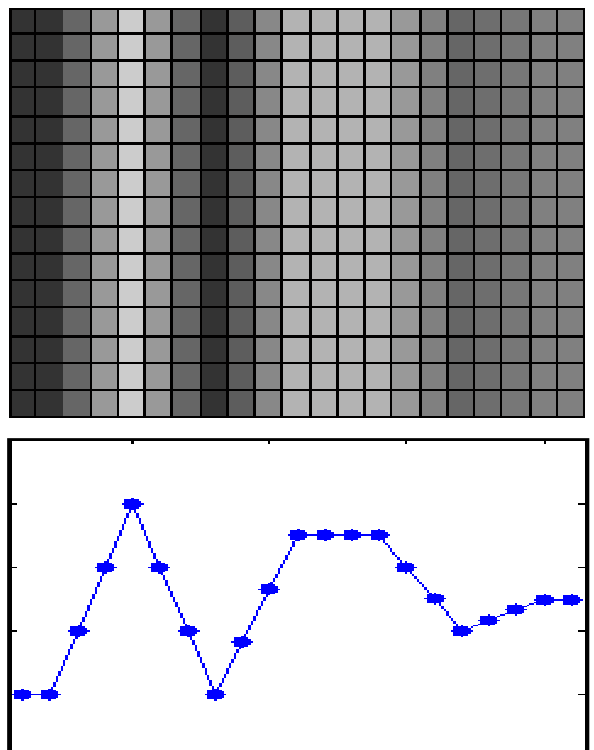
Original



Nearest-Neighbor



Bilinear



Bicubic

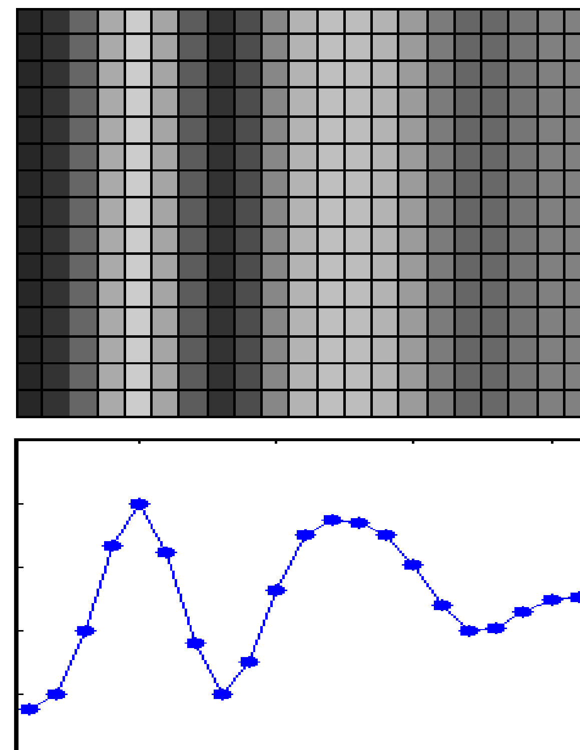
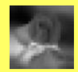


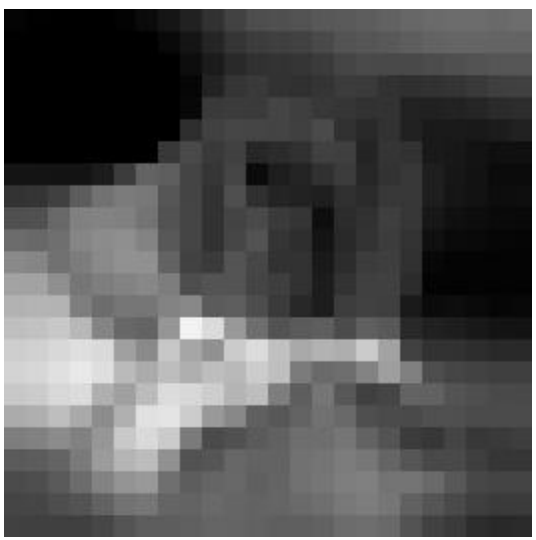
Image Interpolation Methods

256x256
original

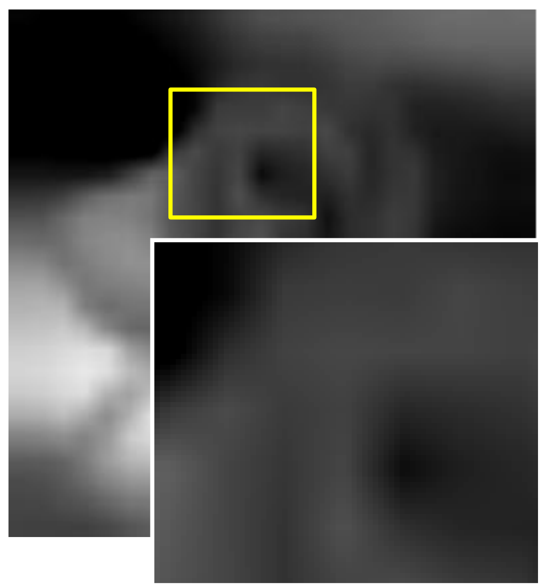


24x24 original

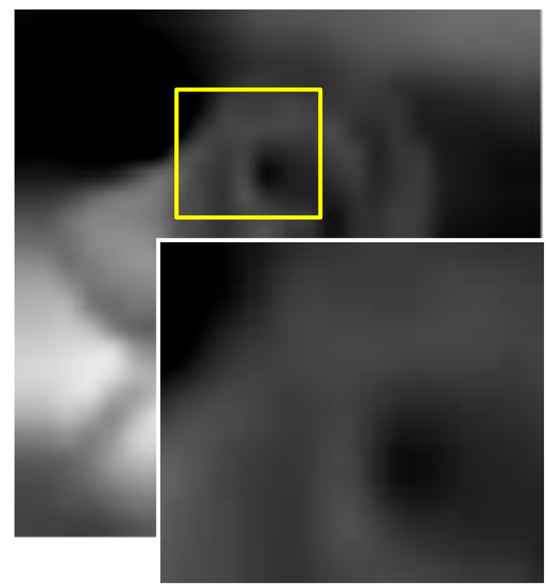
Nearest-Neighbor



Bilinear

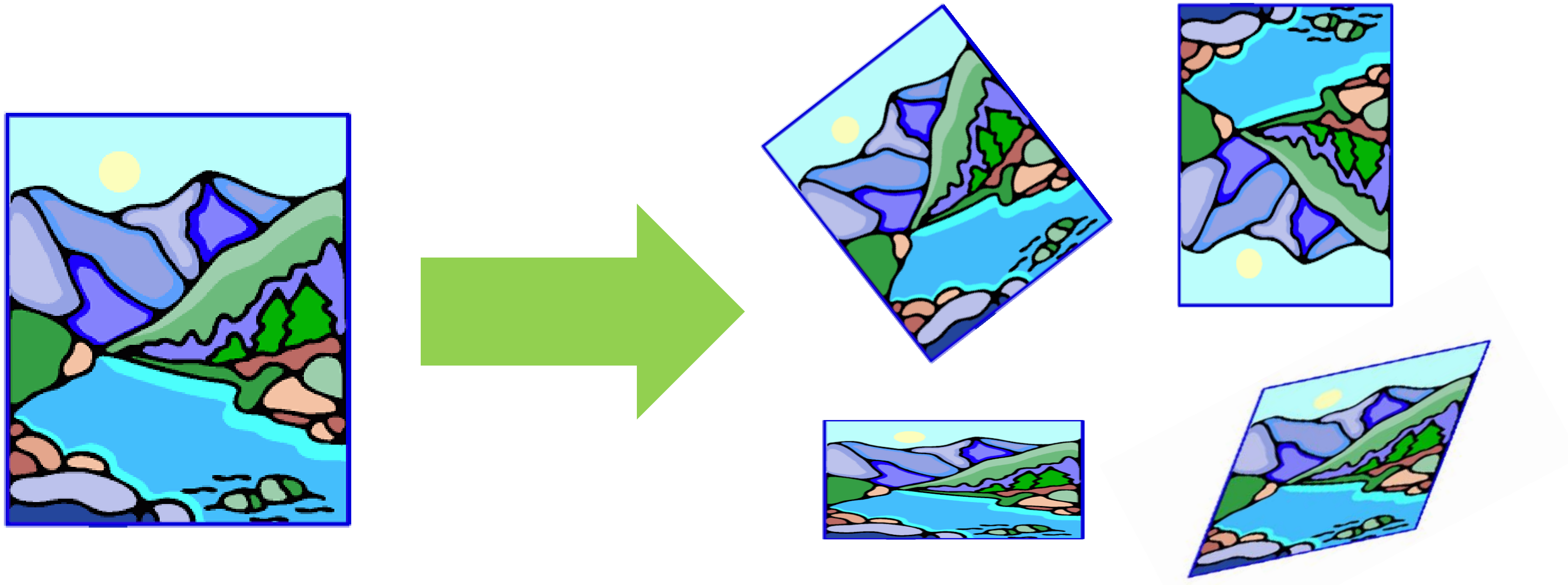


Bicubic



Geometric Transformations

- Generally speaking, geometric transformations (<u>warping</u>) change the spatial arrangement of pixels.
  - Can be linear or nonlinear.
  - Some examples of linear geometric transformations:



Geometric Transformations

- **Affine** transformations (straight lines and parallelism are preserved):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

- A more easy-to-use form of affine transformations:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- See the textbook for the expressions of $\boldsymbol{A}$ for scaling, translation, and rotation.

Geometric Transformations

- However, to construct the "transformed" image from a source image, the standard practice is to consider the transformation in the opposite direction.
  - The inverse transform: $$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \boldsymbol{A}^{-1} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$
  - To get the value at pixel (x',y') of the destination (transformed) image, we first find the corresponding point (x,y) in the source image.
  - When (x,y) contains non-integers, apply interpolation.
  - This is called ***inverse mapping*** in the textbook.