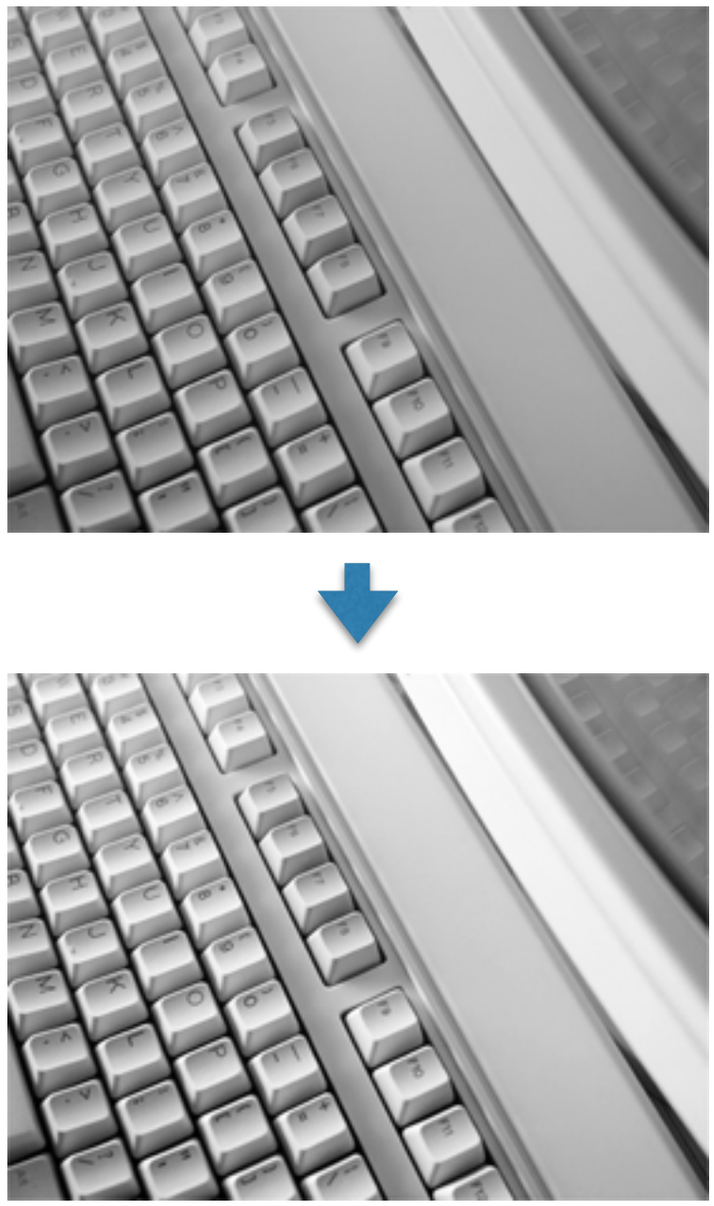
# Level (Point) Operations

CS 355: Interactive Graphics and Image Processing

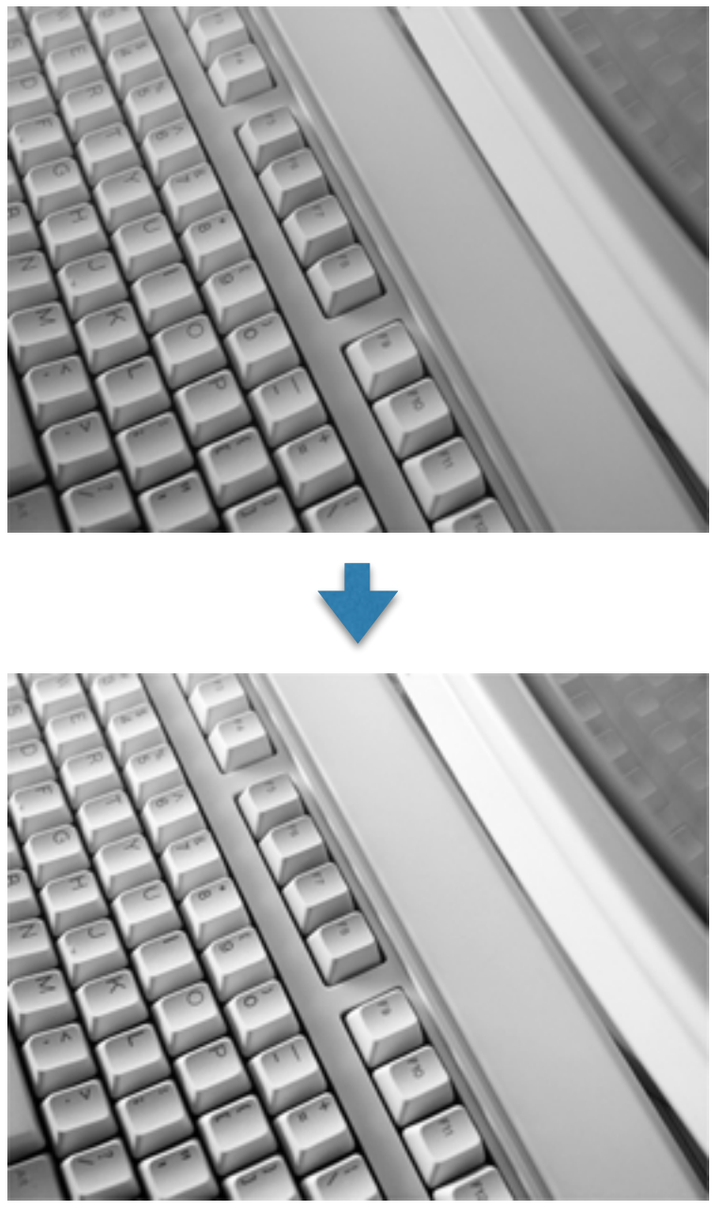
# Level Operations

- Simplest enhancement: *process each point independent of others*
- Output value is a function of the input value *only*
- “Point operations” or “level operations”



# Level Operations

- Simple idea with lots of applications:
  - Brightness
  - Contrast
  - Scaling
  - Clipping
  - Negatives
  - Thresholding
  - Quantization
  - Logarithmic encoding
  - Gamma correction
  - Windowing
  - Equalization
  - and many more…



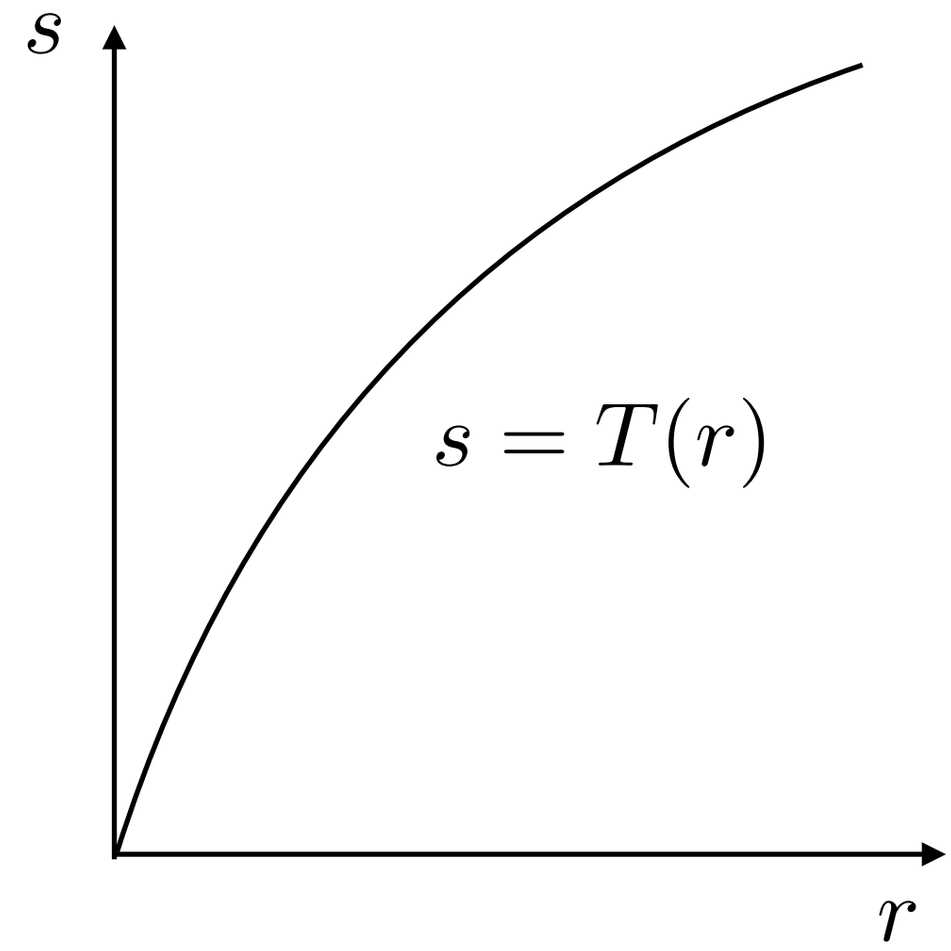
# Level Operations

- Output value is a function of the input value *only*

$$r = \text{input value}$$

$$s = \text{output value}$$

$T =$ a greylevel transformation



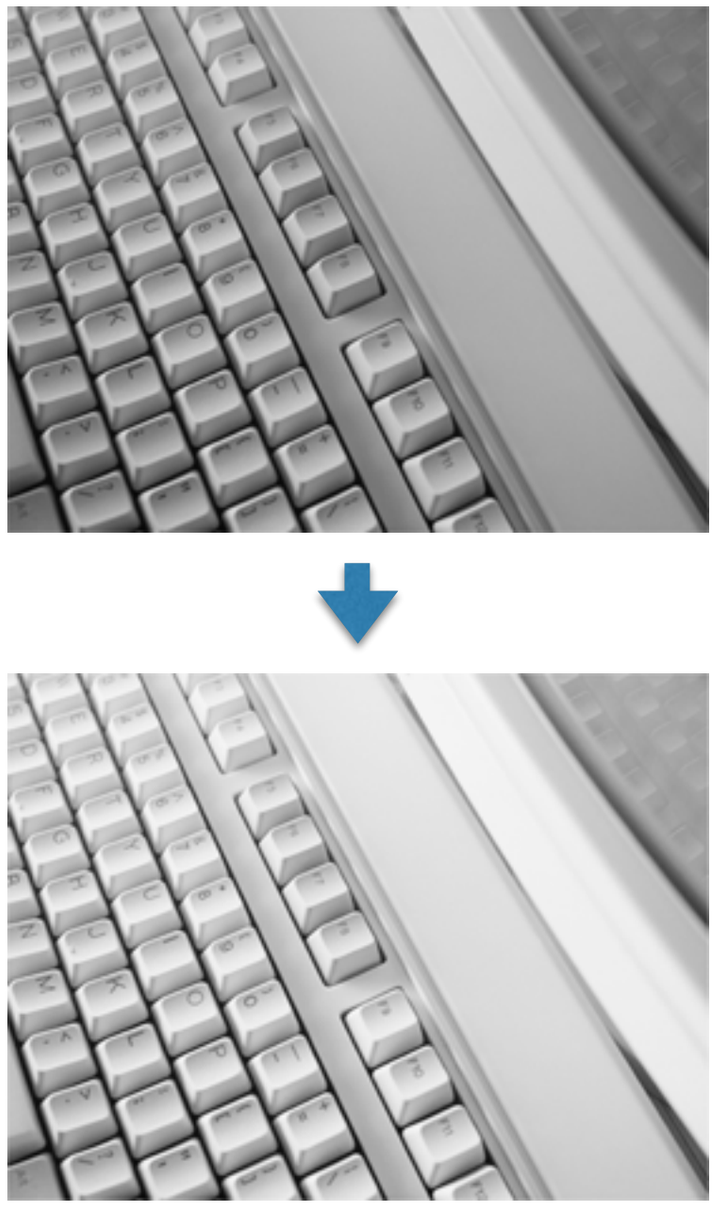

```
for all pixel positions x, y:
    out[x,y] = func(in[x,y])
```

# Brightness

$$s = r + c$$

$c > 0$ brighter

$c < 0$ darker



# Contrast

$$s = a\ r$$

$a > 1$ more contrast

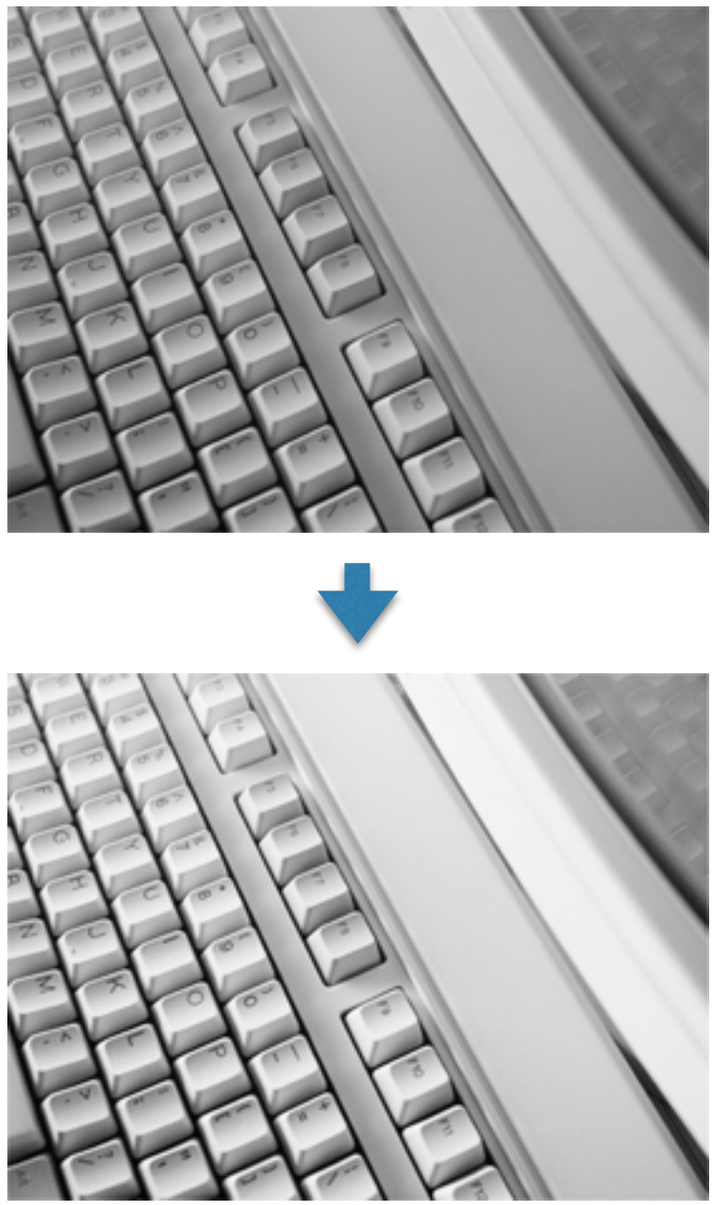
$a < 1$ less contrast

# Linear Operations

$$s = a\ r + c$$

$a$ gain

$c$ bias / offset



# Scaling

Scaling linearly from one range to another:

$$s = (r - r_{\min}) \frac{s_{\max} - s_{\min}}{r_{\max} - r_{\min}} + s_{\min}$$

# Clipping

Clipping to a limited range:

$$s = \begin{cases} s_{\min} & \text{if } r < s_{\min} \\ s_{\max} & \text{if } r > s_{\max} \\ r & \text{otherwise} \end{cases}$$

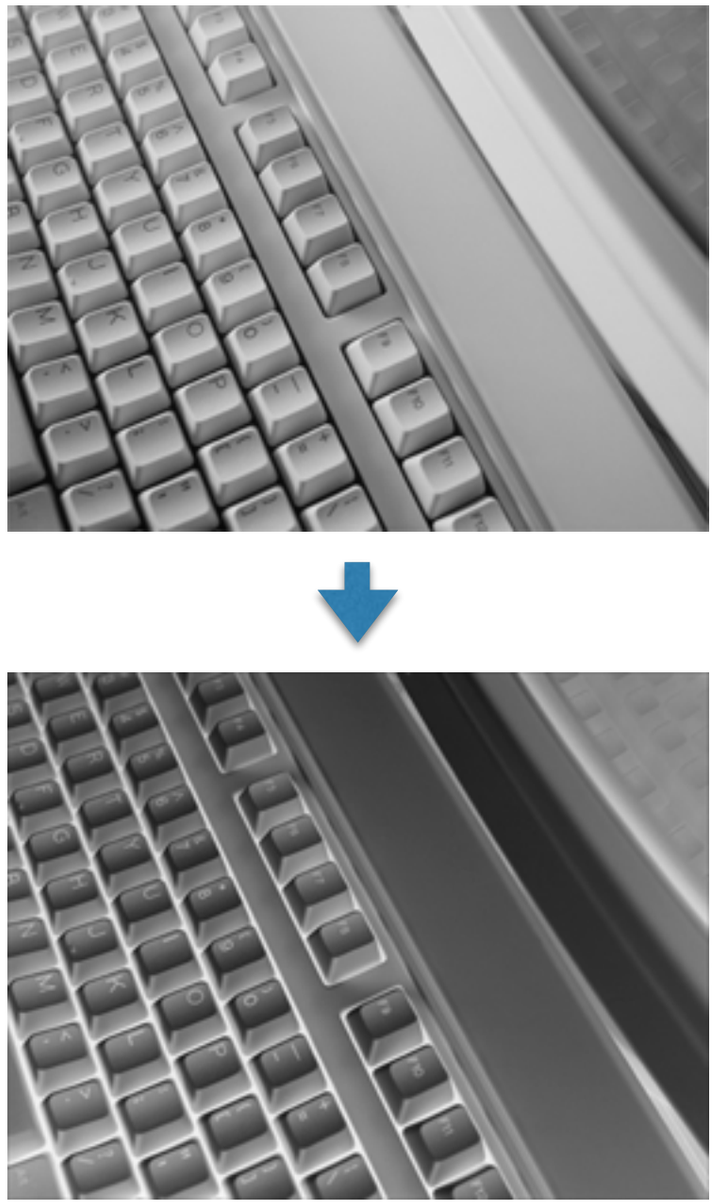
Very common to clip to [min,max] of the range
to avoid unsigned wrap-around

# Negative

$$s = r_{\max} - r$$

or

$$s = r_{\max} - r + r_{\min}$$

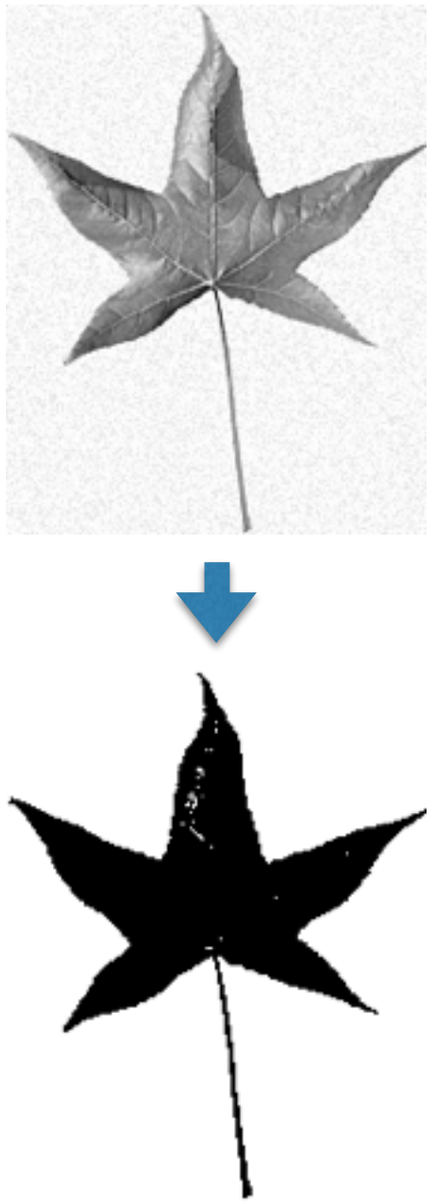


# Thresholding

Binarization based on a threshold

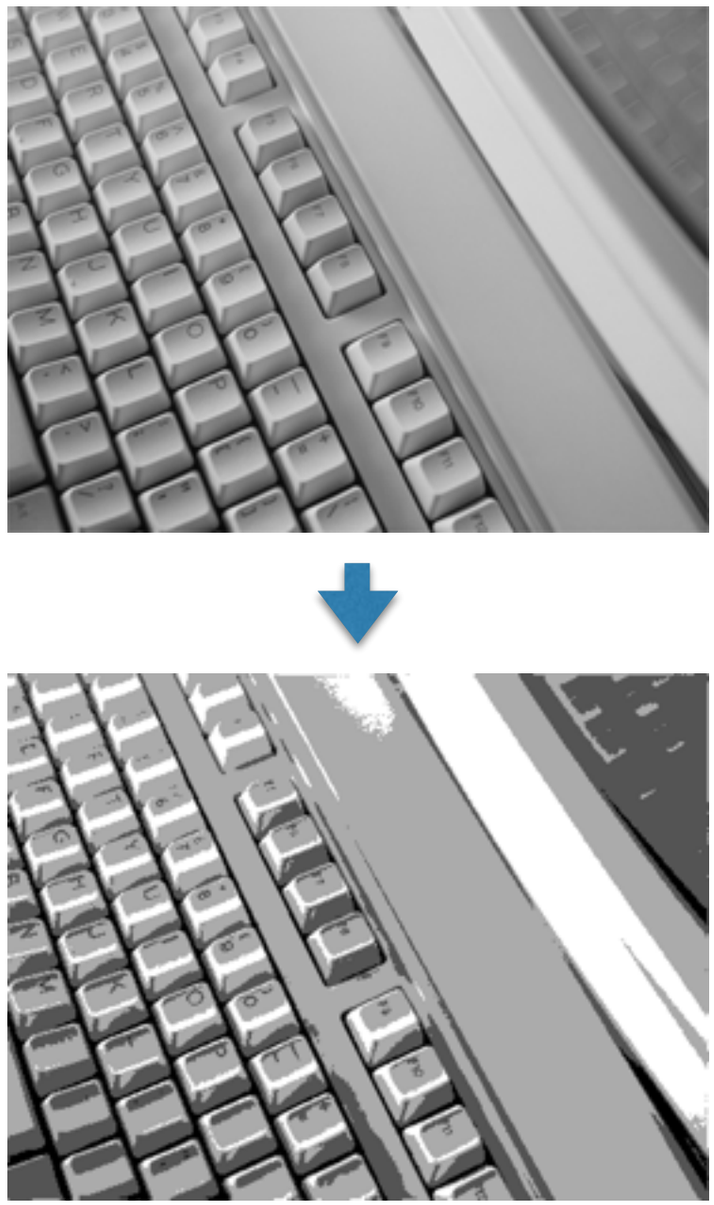
$$s = \begin{cases} 1 & \text{if } r > r_0 \\ 0 & \text{otherwise} \end{cases}$$

$r_0 =$ selected threshold



# Quantization

$$s = \begin{cases} s_0 & \text{if } r_{\min} \leq r_< r_0 \\ s_1 & \text{if } r_1 \leq r_< r_2 \\ s_2 & \text{if } r_2 \leq r < r_3 \\ \vdots & \\ s_n & \text{if } r_n \leq r \leq r_{\max} \end{cases}$$



# Logarithm / Exponent

- Sometimes care more about *relative changes* than absolute ones

- Lots of things use logarithmic scales
  - Decibel (dB) units
  - Apparent brightness
  - Richter scale
  - Human Vision

$$s = \log(r)$$

$$s = e^r$$

- Can “undo” with exponentiation

# Power Functions

Can also raise to a desired power:

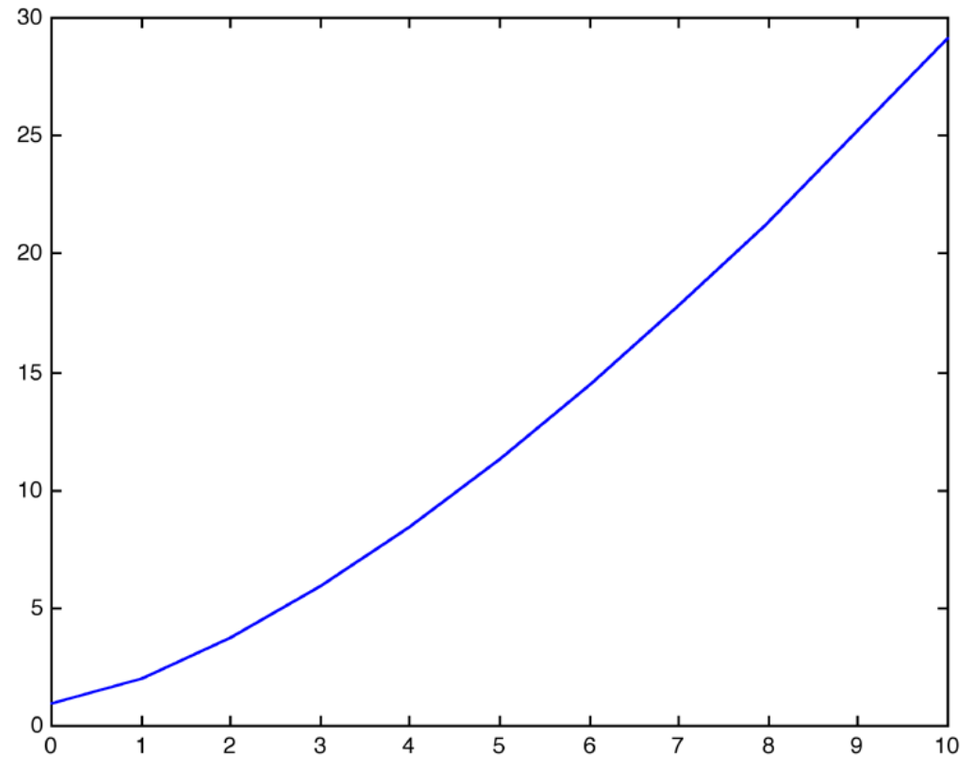
$$s = r^p$$

# Gamma Responses

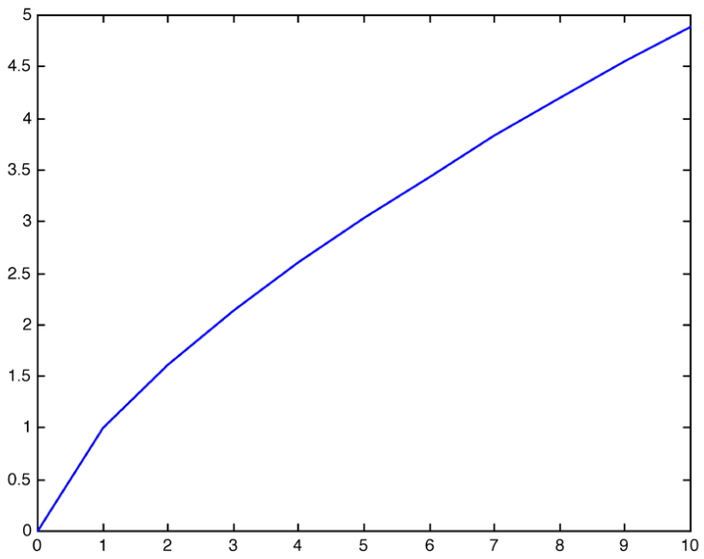
- Many devices have a nonlinear response:
- For a CRT, the intensity is related to the voltage by

$$I = V^{\gamma} + c$$

- The exponent is often called the “gamma” of the device

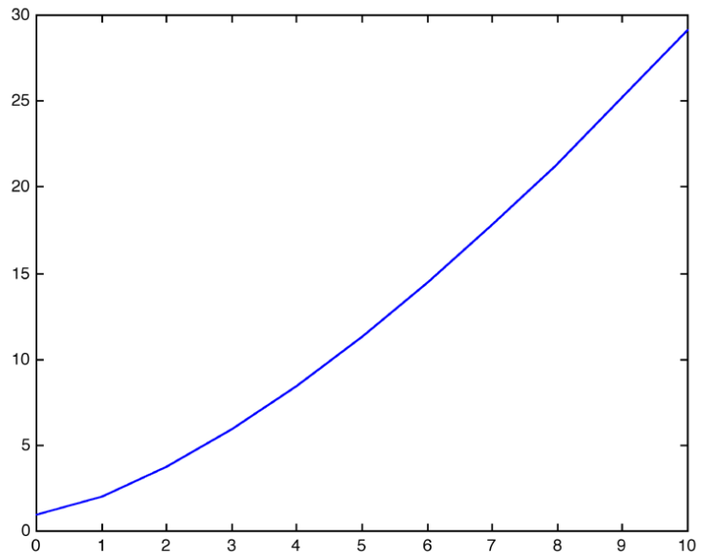


# Gamma Correction



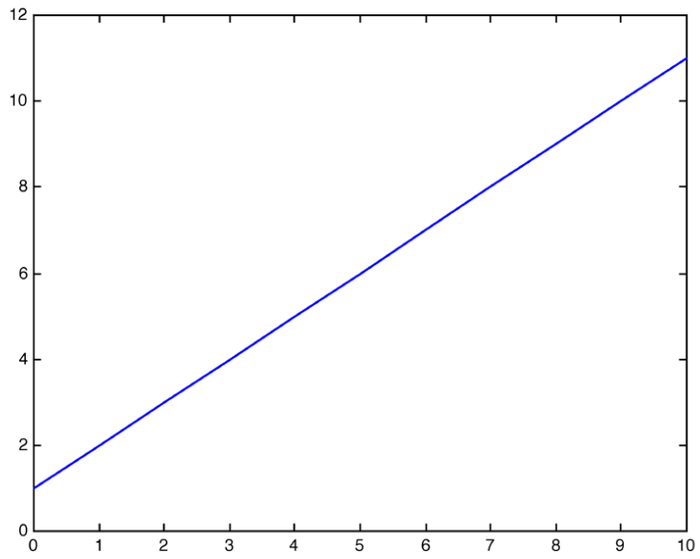

Preprocess

$$s = r^{1/\gamma}$$




Device

$$I = V^{\gamma} + c$$




Result

# Contrast Enhancement

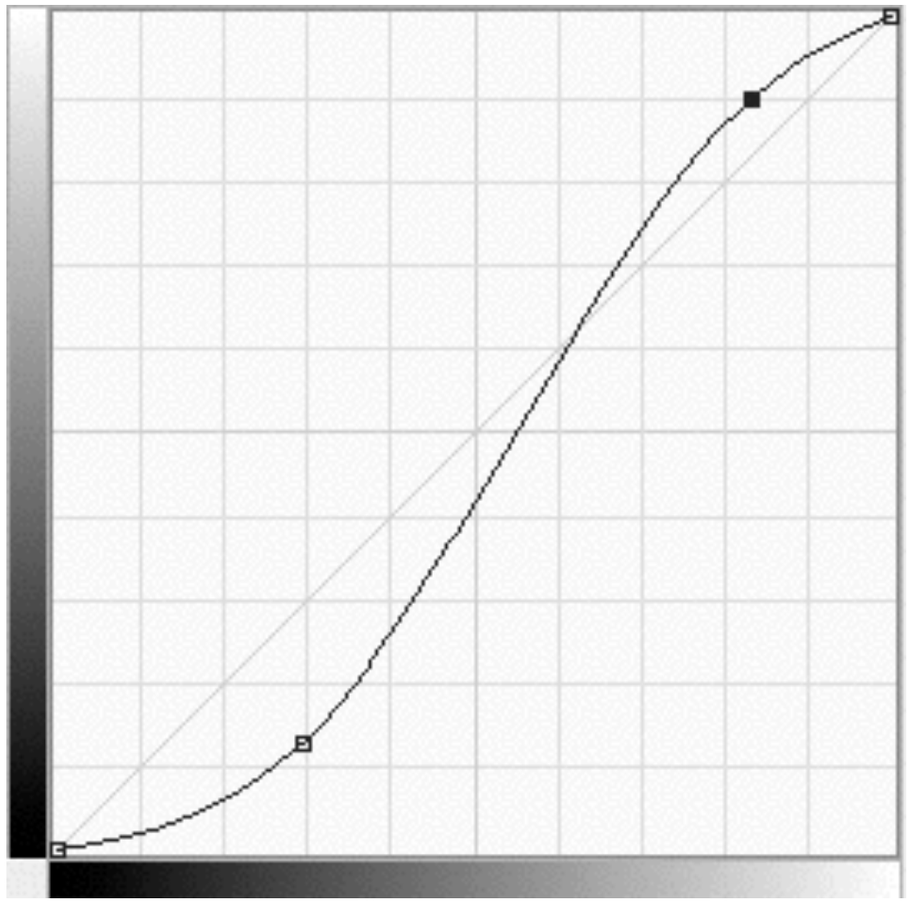
- *Contrast enhancement* makes differences more distinguishable

- Trades off *decreased contrast* in some part(s) of the range for *increased contrast* in the range we're interested in

- If we plot the function,

  - Slope > 1 means enhancement

  - Slope < 1 means reduction



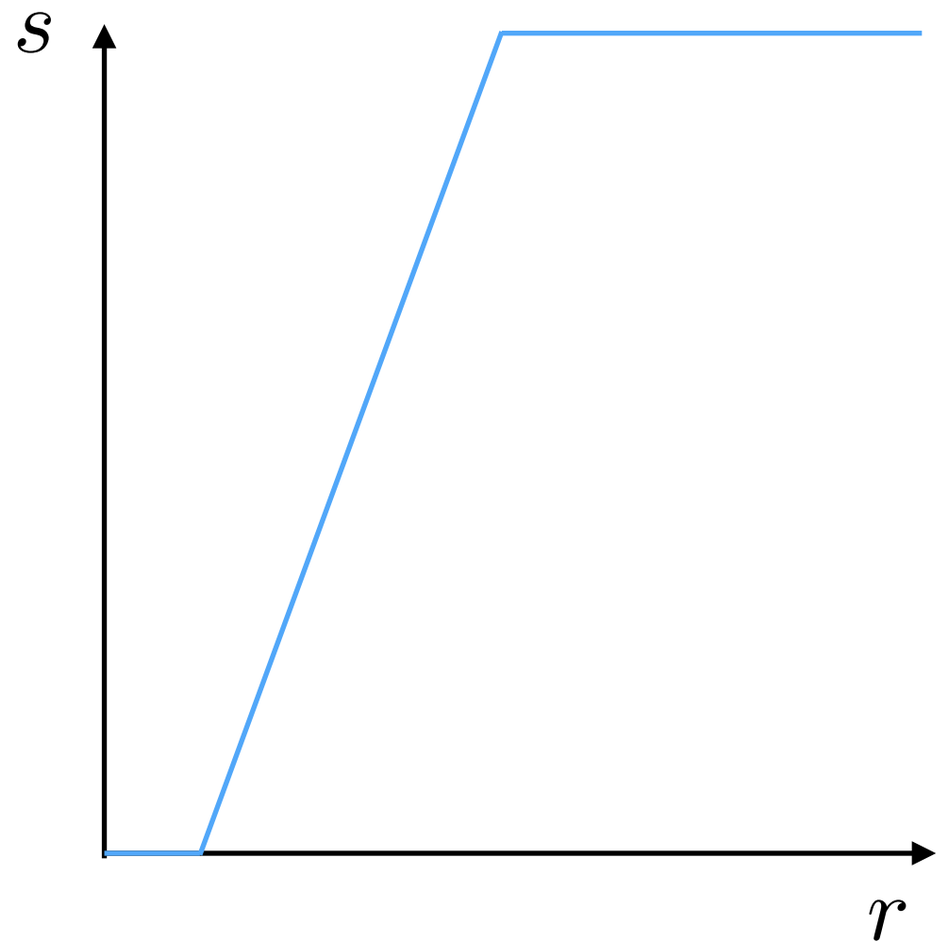
# Windowing

- *Windowing* is enhancement of *one part* of the range

- Example: displaying 12-bit X-rays on an 8-bit screen

- Simple: scale [0,4095] to [0,255] by dividing by 16

- Better: if you know that what you're interested in is in the range [500,2000], *enhance that part of the range*




$$s = \begin{cases} 0 & \text{if } r < 500 \\ 255(r - 500)/(2000 - 500) & \text{if } 500 \leq r < 2000 \\ 255 & \text{if } r \geq 2000 \end{cases}$$

# Windowing

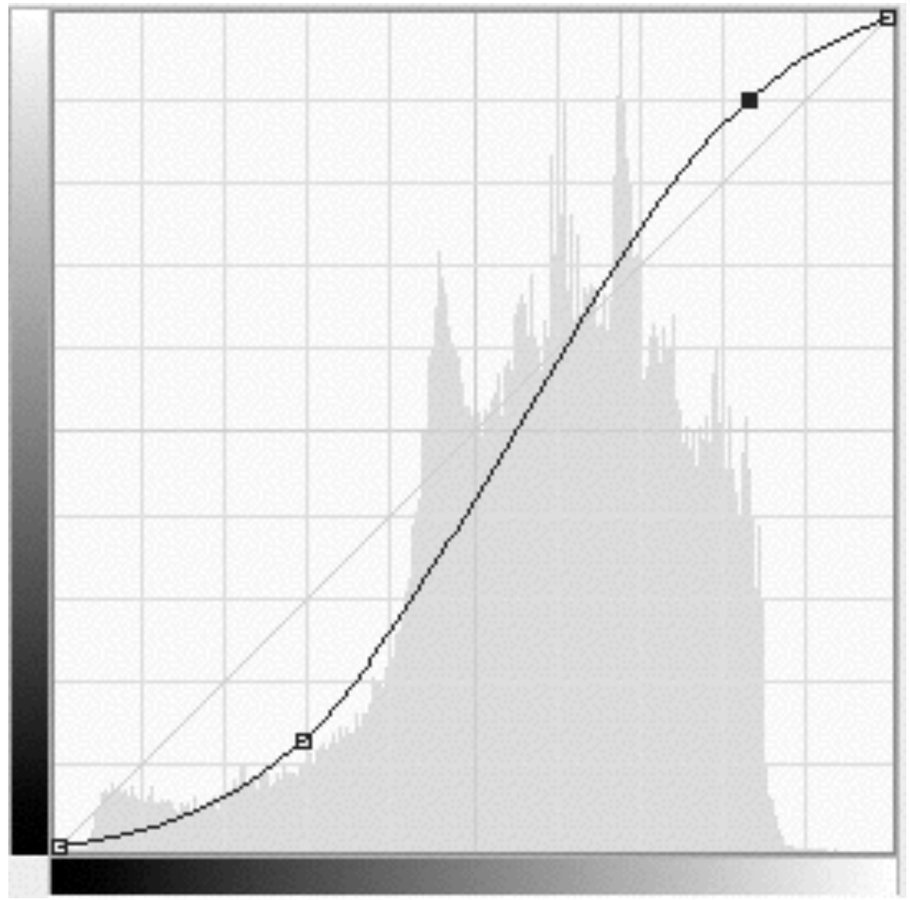
- Another example:
  - Want to better see the reflection of the keys
    - Enhance this range at expense of contrast in brighter or darker areas
- If the full range is already used, contrast enhancement is a zero-sum game

# Histogram Equalization

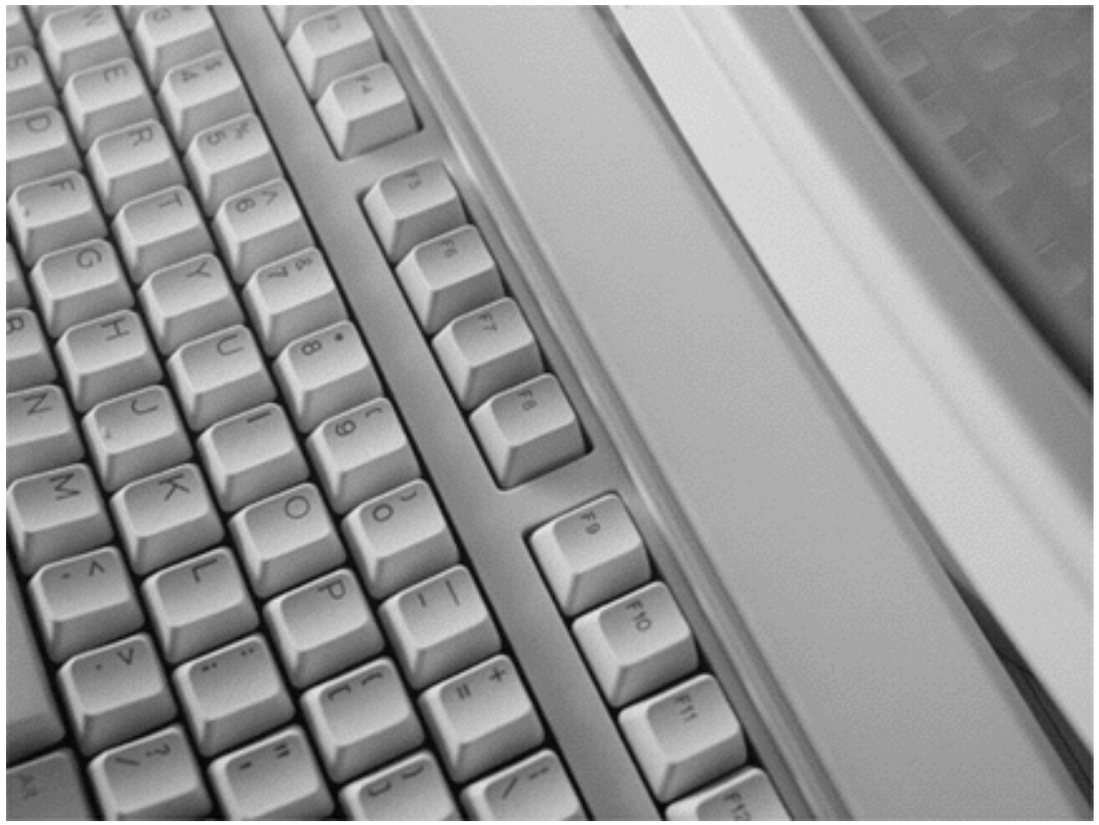
- Can try to automatically optimize contrast by allocating according to brightness distribution (histogram)

- This is called *histogram equalization* because it tries to spread contrast evenly

- Strong discrimination of detail, but not always good “real looking” result



# Histogram Equalization



# Coming up…

- Interimage: blending, masking, differencing, compositing
- Neighborhood operations:
    - noise reduction
    - sharpening
    - edge detection
- Interpolation, curves, surfaces
- Geometric operations: resizing, rotating, warping