Video Signals

IMAGE DEFINITION AND POINT OPERATION

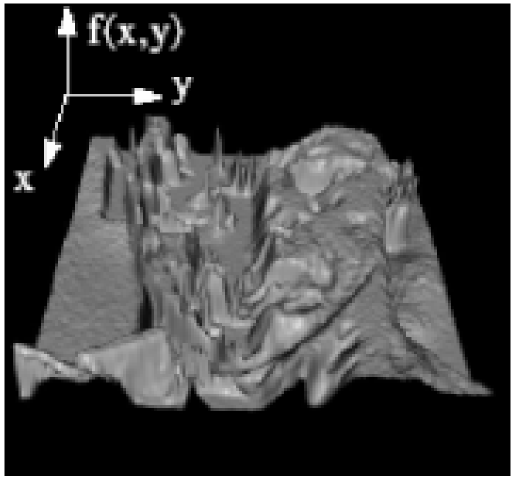
What is an image?

Ideally, we think of an **image** as a **2-dimensional light intensity function**, *f(x,y)*, where x and y are spatial coordinates, and *f* at *(x,y)* is related to the brightness or color of the image at that point.

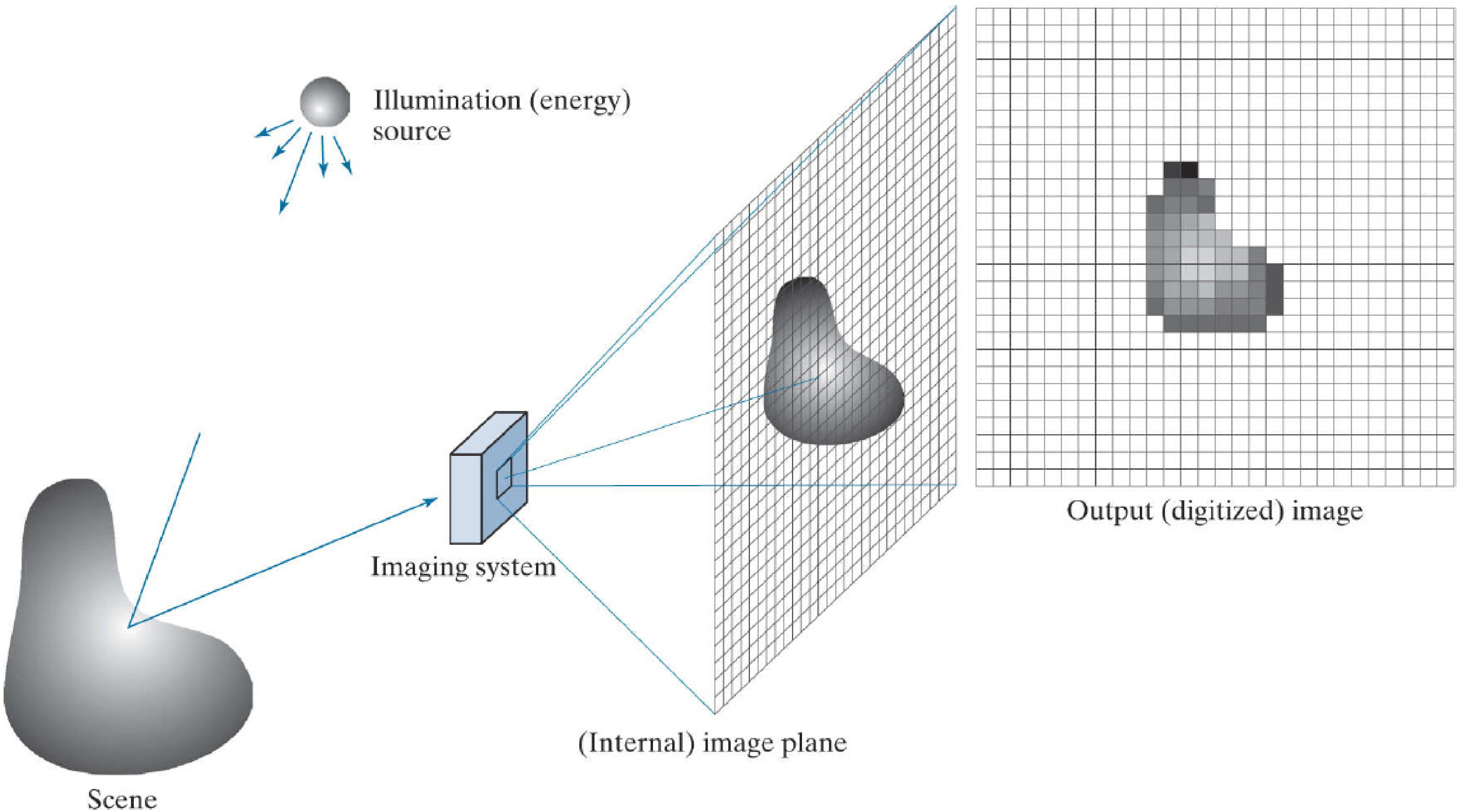
In practice, most images are defined over a rectangle.

Continuous in amplitude („continuous-tone“)

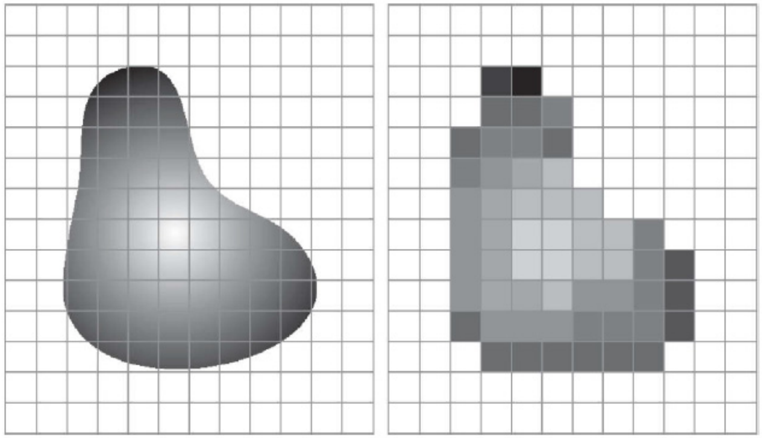
Continuous in space: no pixels!

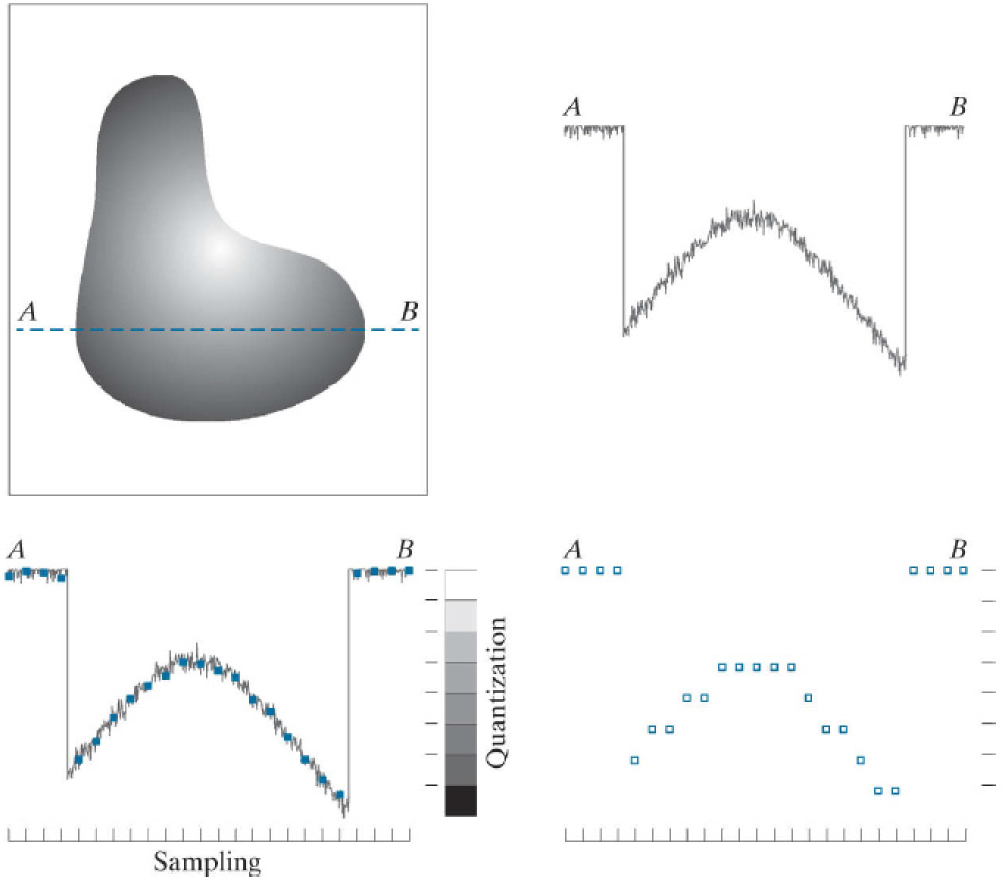


SAMPLING AND QUANTIZATION

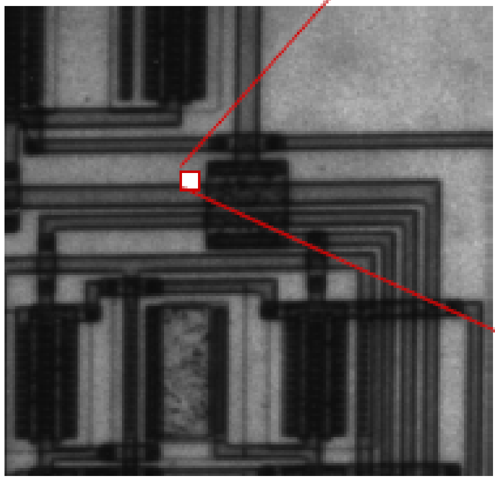


SAMPLING AND QUANTIZATION





A Digital Image is Represented by Numbers



128	125	107	105	110	118	116	114	110
121	122	115	108	106	107	116	116	107
110	114	112	107	105	103	106	106	100
100	96	100	99	94	94	101	101	89
85	82	81	80	76	75	80	82	72
58	58	56	54	53	52	51	49	45
41	41	41	39	39	38	36	35	33
43	43	42	43	41	41	41	43	40
60	60	59	59	60	59	59	58	56

- Pixel = “picture element”
- Represents brightness at one point

An image can be represented as a matrix

The pixel values *f(x,y) are sorted into the matrix in „natural“ order, with x corresponding to the column and y to the row index.*

Matlab, instead, uses matrix convention. This results in $f(x,y) = f_{yx}$, *where* f_{yx} *denotes an individual* element in common matrix notation.

For a color image, **f** might be one of the components.

$$\xrightarrow{\quad x \quad}$$

$$\mathbf{f} = \begin{bmatrix} f(0,0) & f(1,0) & \cdots & f(N-1,0) \\ f(0,1) & f(1,1) & \cdots & f(N-1,1) \\ \vdots & \vdots & & \vdots \\ f(0,L-1) & f(1,L-1) & \cdots & f(N-1,L-1) \end{bmatrix} \Bigg\downarrow y$$

Image Size and Resolution

200x200 100x100 50x50 25x25

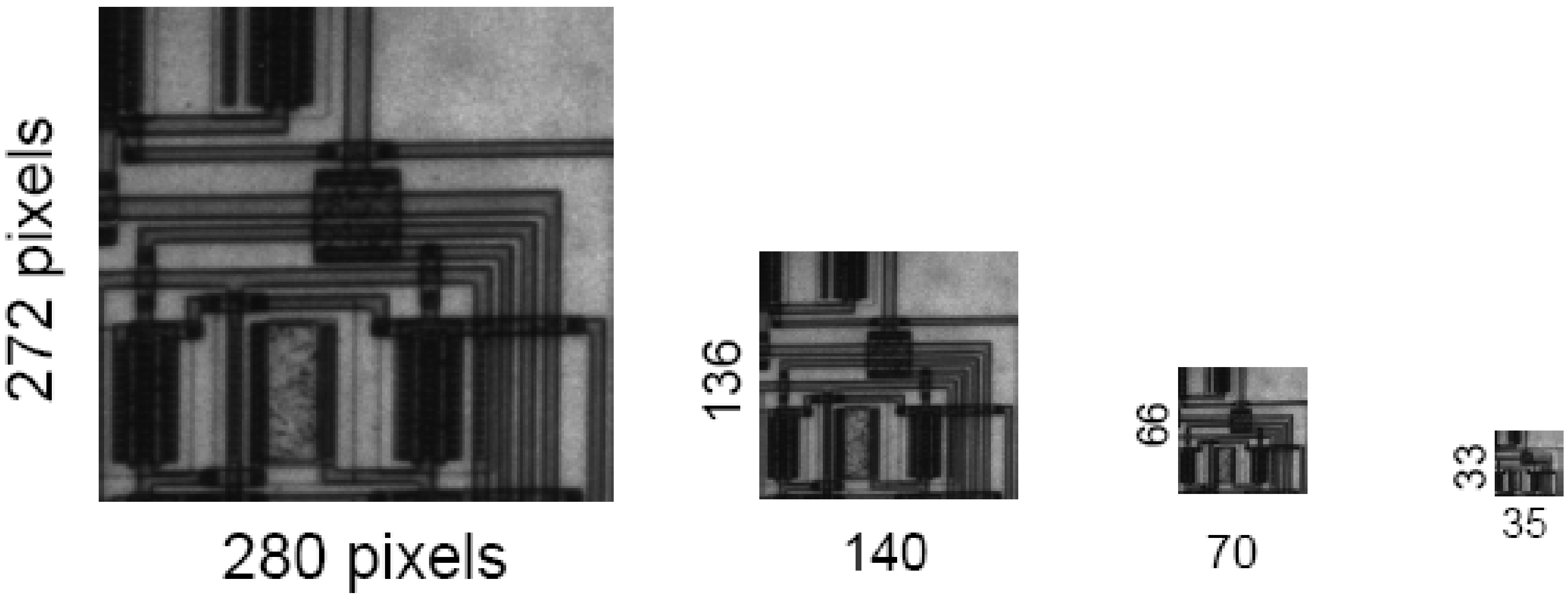
These images were produced by simply picking every *n-th* sample horizontally and vertically and replicating that value *nXn* times.

We can do better

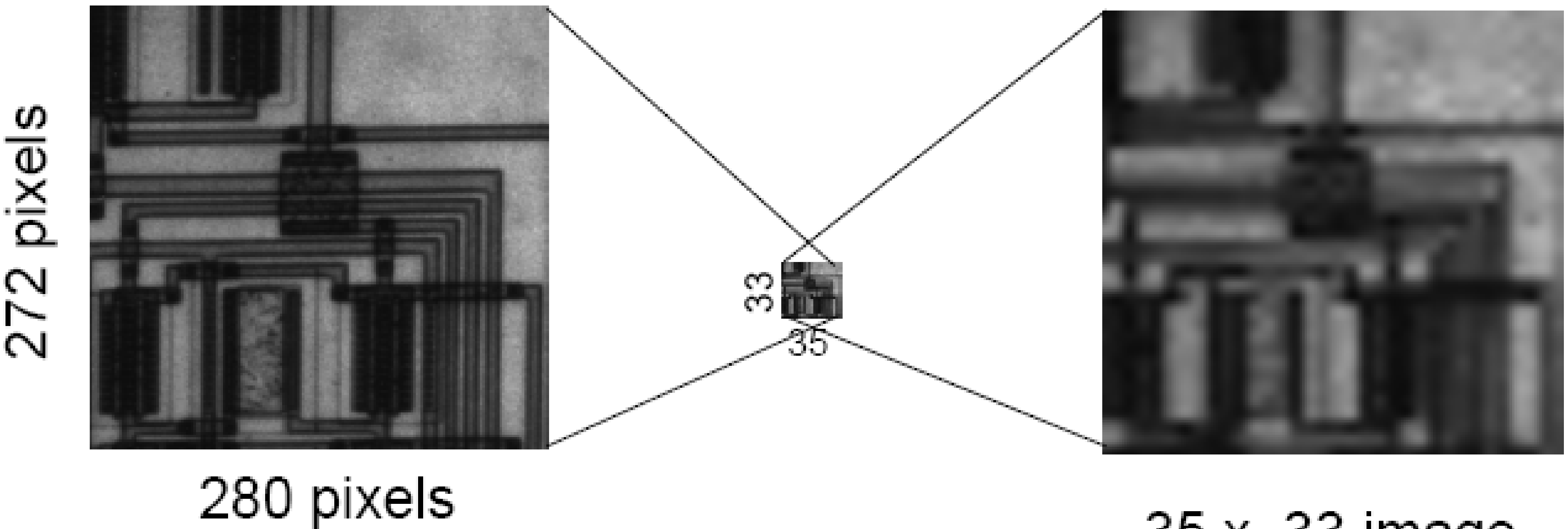
- *prefiltering before subsampling to avoid aliasing*
- *Smooth interpolation*

Reducing spatial resolution

Images of different size

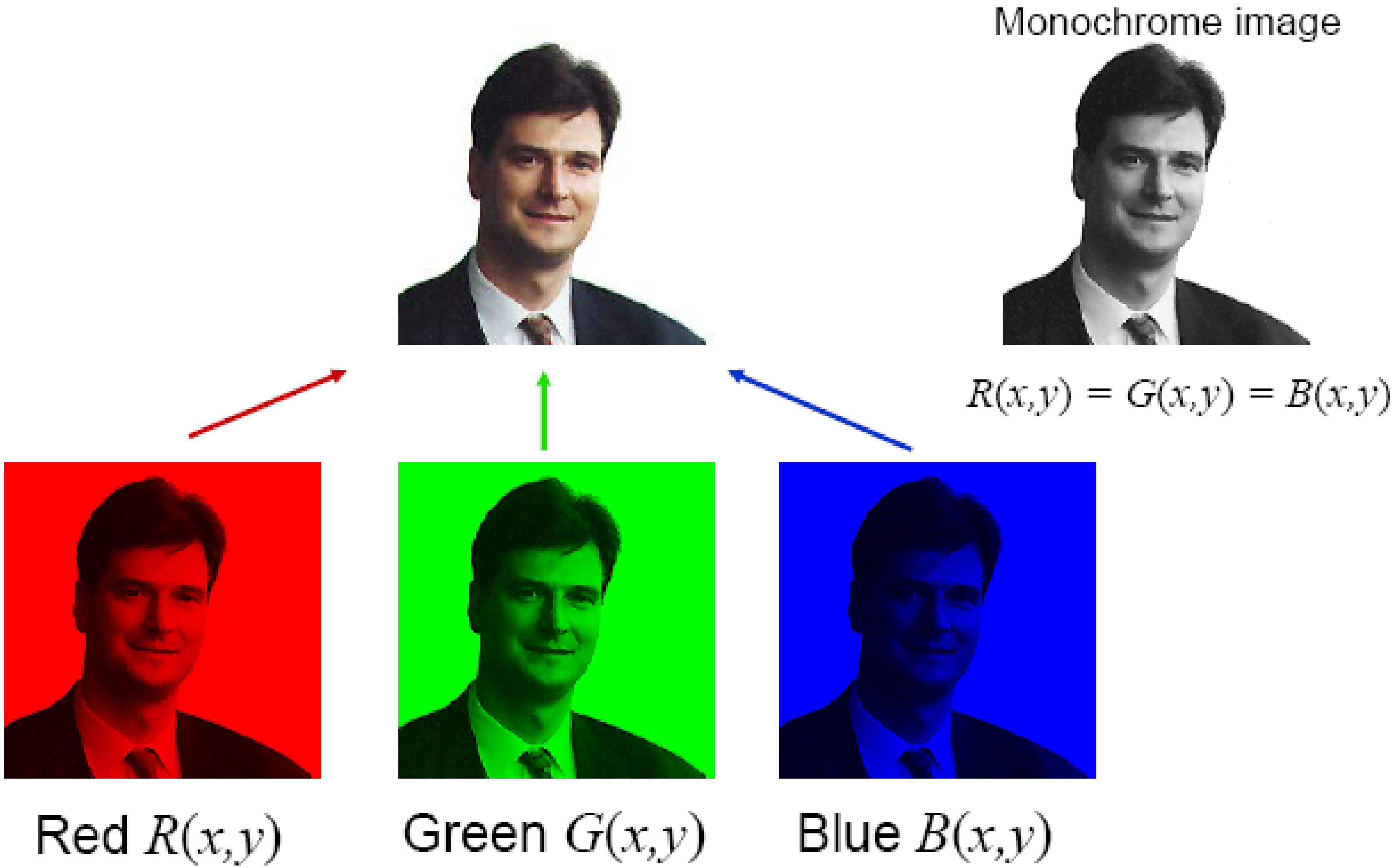


Fewer Pixels Mean Lower Spatial Resolution

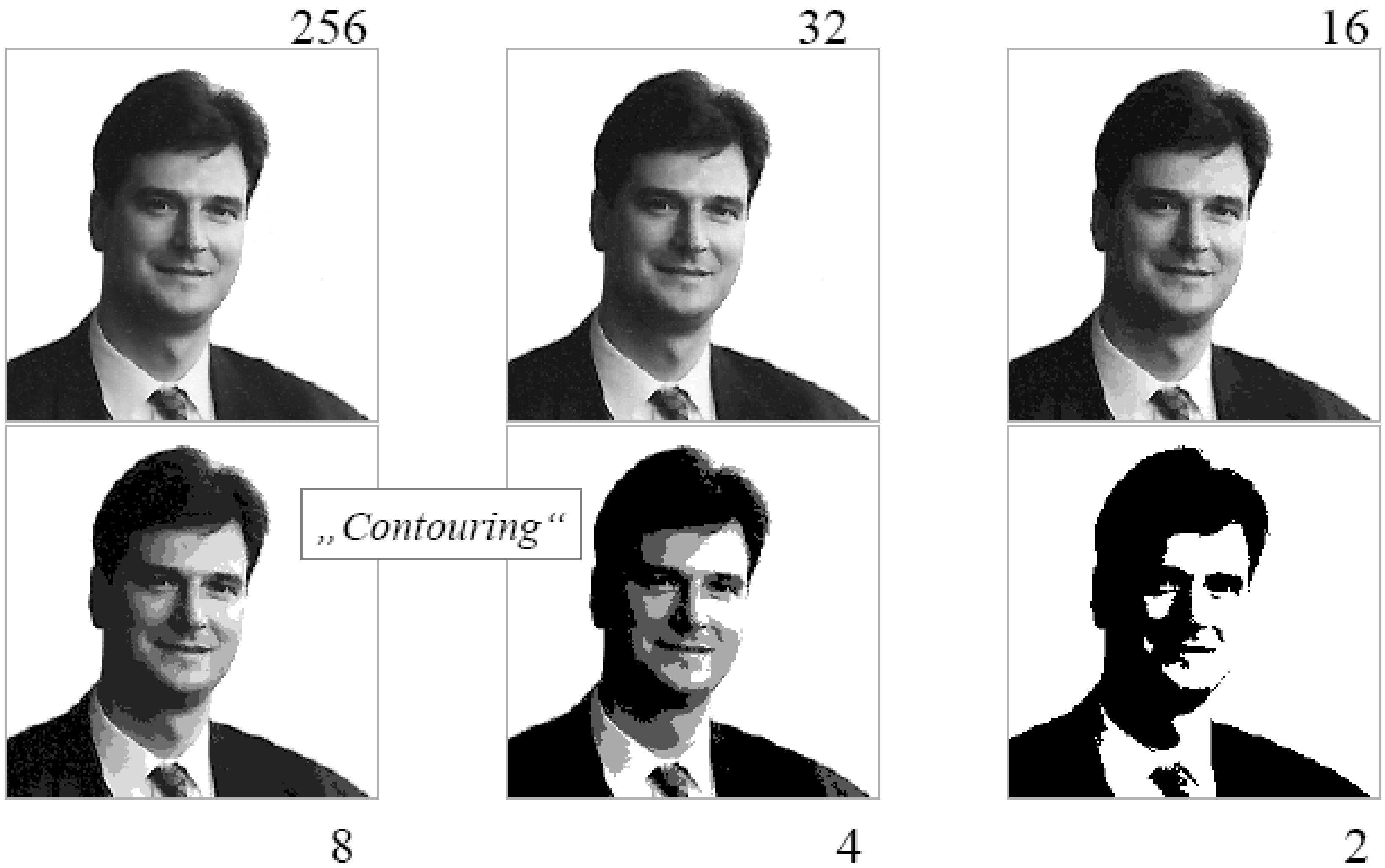


35 x 33 image interpolated to 280 x 272 pixels

Color Components

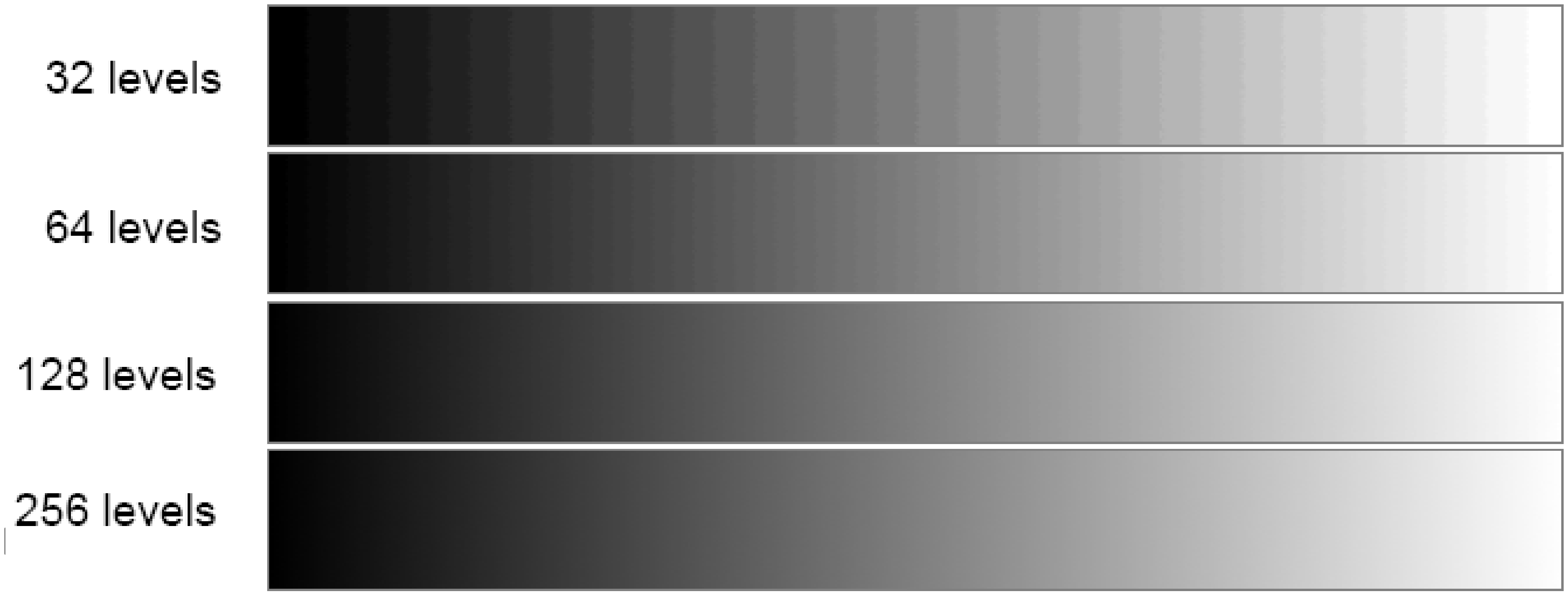


Different numbers of gray levels



How many gray levels are required?

How many gray levels are required?



Storage requirements for digital images

Image LxN pixels, *2^B gray levels, c color components*

$$Size = L \times N \times B \times c$$

- *Example: L=N=512, B=8, c=1 (i.e., monochrome) Size = 2,097,152 bits (or 256 kByte)*
- *Example: LxN=1024x1280, B=8, c=3 (24 bit RGB image) Size = 31,457,280 bits (or 3.75 MByte)*

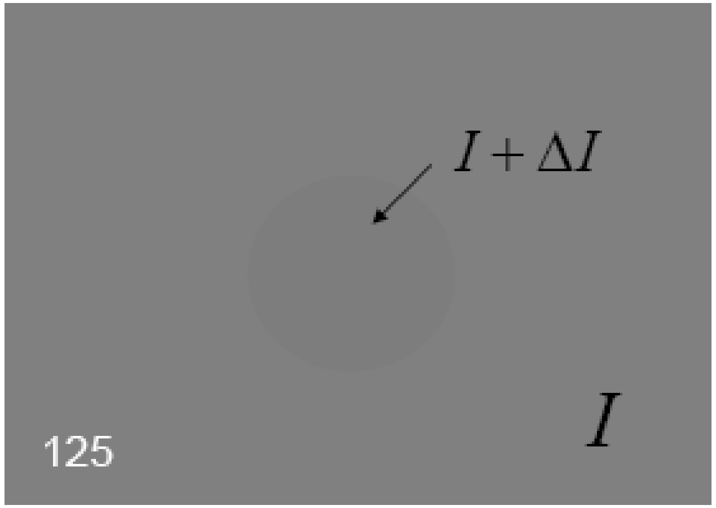
Much less with (lossy) compression!

For a video multiply by the frame rate and by the number of seconds of its length:

a 4K video at 50 fps would be: 2160x3840x8x3x50≈10Gb/s ≈ 1.2GB/s

Brightness discrimination experiment

Can you see the circle?



Note: I is luminance, measured in cd/m^2

Visibility threshold

$$\Delta I / I \approx const. \approx 1 \ldots 2\%$$

„Weber fraction“
„Weber‘s Law“

Contrast with 8 Bits According to Weber's Law

Assume that the luminance difference between two successive representative levels is just at visibility threshold

$$\frac{I_{\max}}{I_{\min}} = (1 + const.)^{255}$$

For

$$const. = 0.01 \cdots 0.02 \qquad \frac{I_{\max}}{I_{\min}} = 13 \cdots 156$$

Typical display contrast

- Cathode ray tube 100:1
- Print on paper 10:1

Suggests uniform quantization in the log(*I*) domain

Histograms

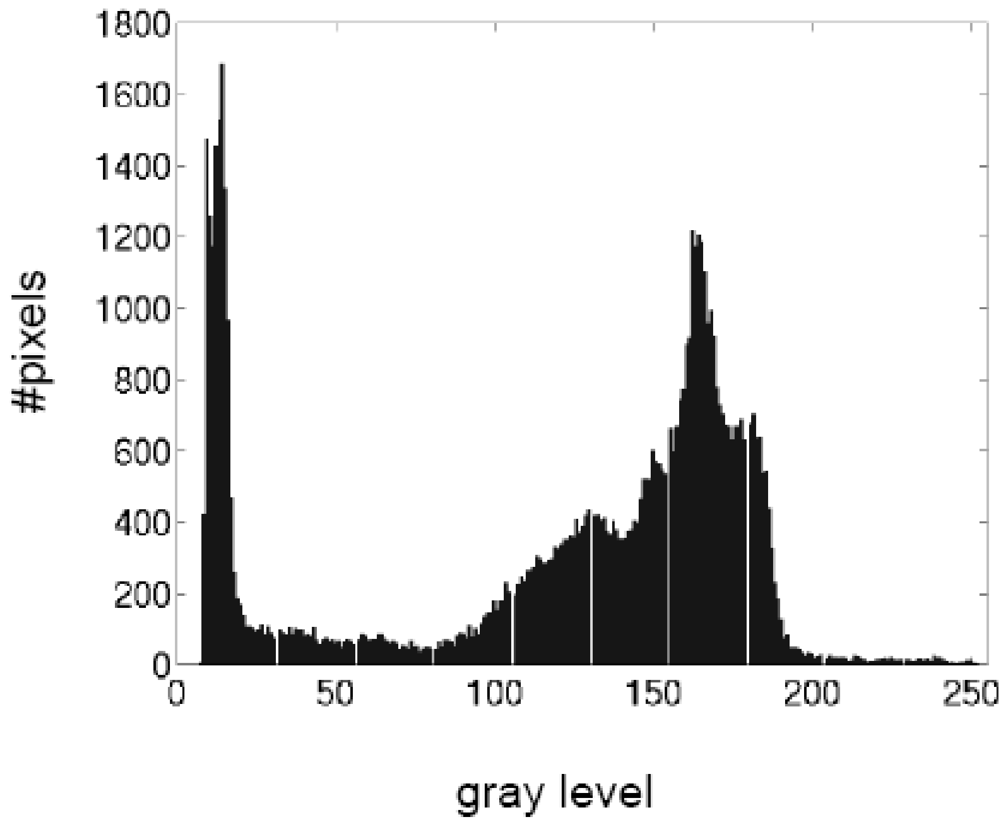
Distribution of gray-levels can be evaluated by measuring a histogram:

- For B-bit image, initialize 2^B counters with 0
- Loop over all pixels *x,y*
- When encountering gray level *f(x,y)=i, increment counter #i*

Histogram can be interpreted as an estimate of the probability density function (*pdf*) of an underlying random process.

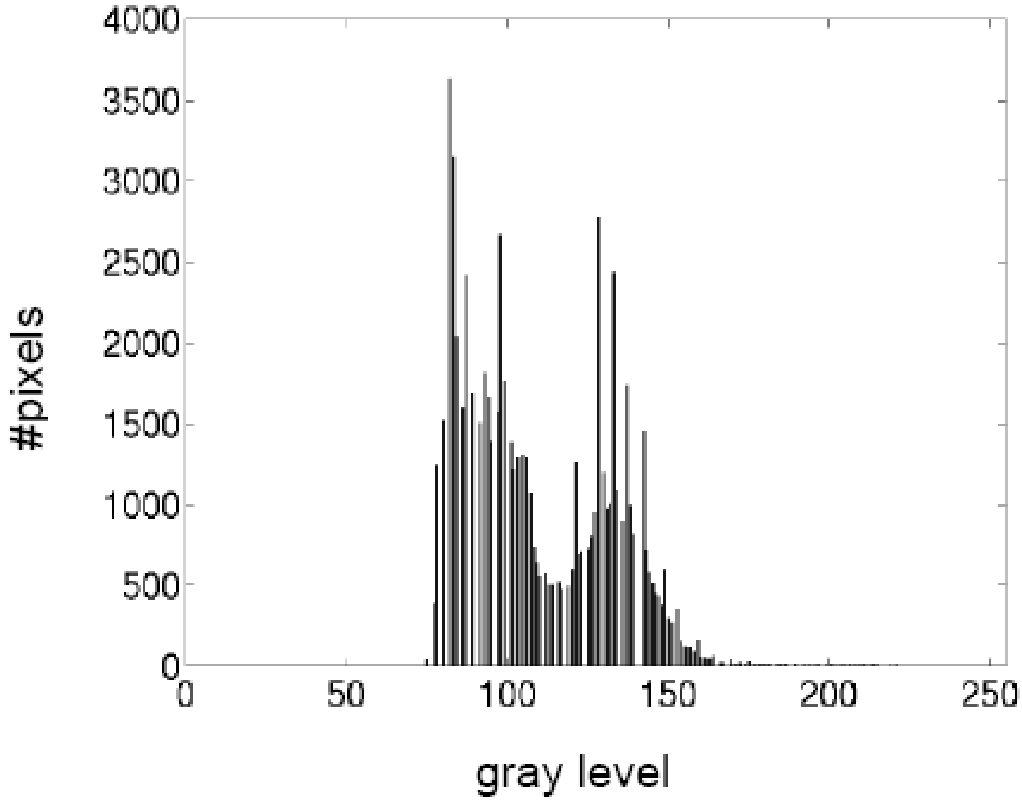
You can also use fewer, larger bins to trade off amplitude resolution against sample size.

Example histogram



Cameraman image

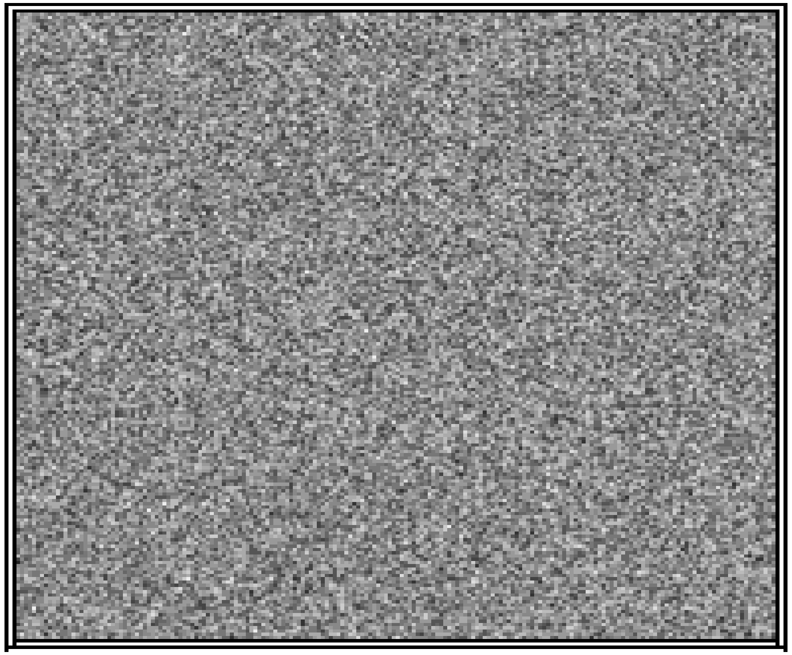
Example histogram

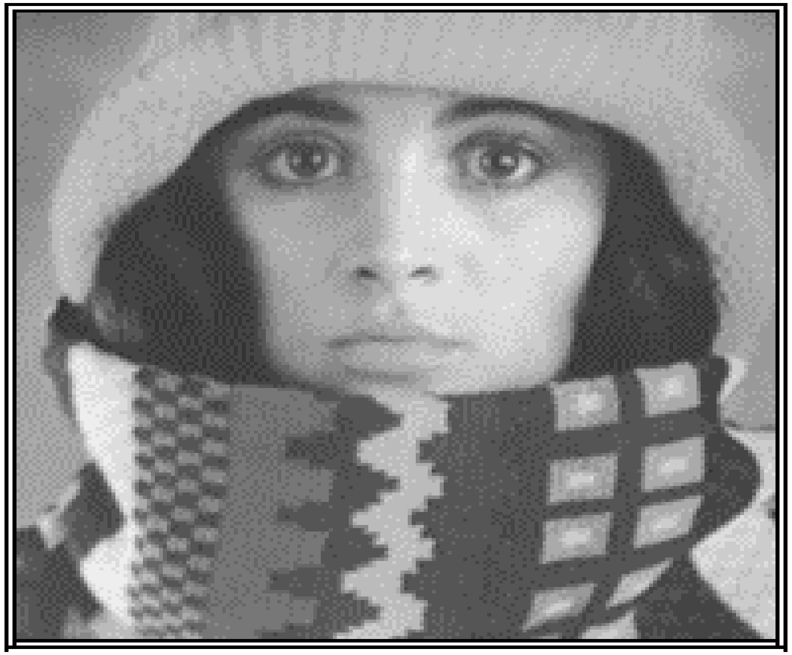


Pout
image

Histogram comparison

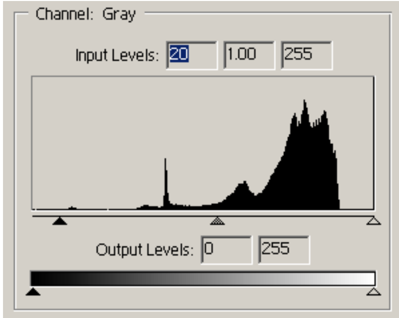
Both these images present the same Histogram

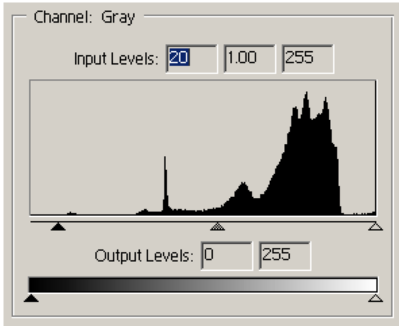


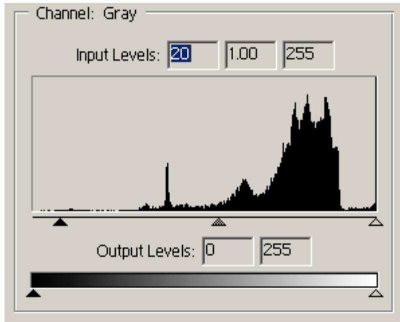


Histogram comparison

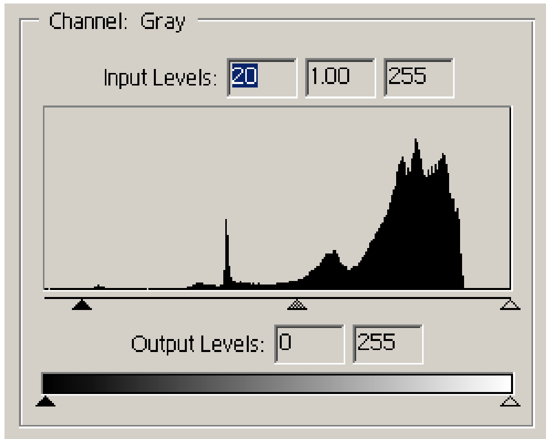
Histogram as an invariant feature

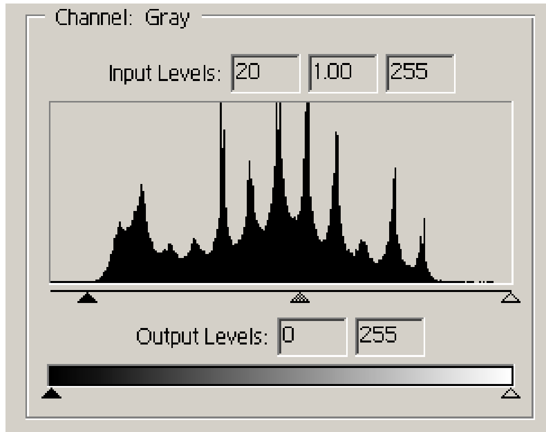






Histogram comparison





Histogram equalization

Idea: find a non-linear transformation

$$g = T(f)$$

to be applied to each pixel of the input image ***f(x,y)****, such that a uniform distribution of gray levels in the entire range results for the output image* ***g(x,y)****.*

Analyze ideal, continuous case first, assuming

$$0 \le f \le 1 \qquad 0 \le g \le 1$$

T(f) is strictly monotonically increasing, hence, there exists

$$f = T^{-1}(g) \qquad 0 \le g \le 1$$

Goal: pdf (probability density function) $p_g(g) = \text{const.}$ *over the whole range.*

Histogram equalization for continuous case

From basic probability theory

$$p_g(g) = \left[p_f(f) \frac{df}{dg} \right]_{f=T^{-1}(g)}$$

Consider the transformation function

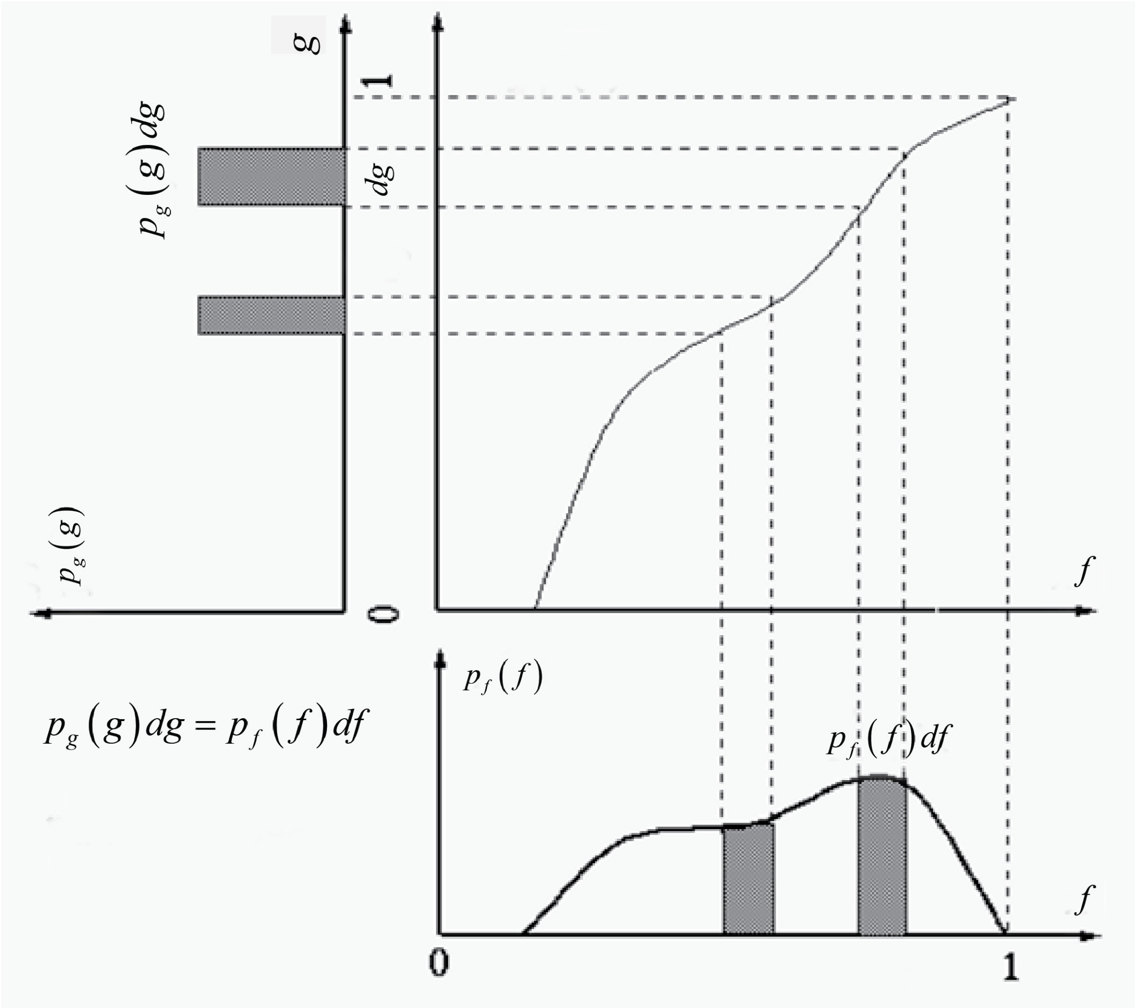
$$g = T(f) = \int_0^f p_f(\alpha) d\alpha \qquad 0 \le f \le 1$$

Then

$$\frac{dg}{df} = p_f(f)$$

$$p_g(g) = \left[p_f(f) \frac{df}{dg} \right]_{f=T^{-1}(g)} = \left[p_f(f) \frac{1}{p_f(f)} \right]_{f=T^{-1}(g)} = 1$$

Histogram equalization for continuous case



Histogram equalization for discrete case

Now, *f only assumes discrete amplitude values* f_0, f_1, …, f_{L-1}, with probabilities:

$$P_0 = \frac{n_0}{n} \qquad P_1 = \frac{n_1}{n} \qquad \cdots \qquad P_{L-1} = \frac{n_{L-1}}{n}$$

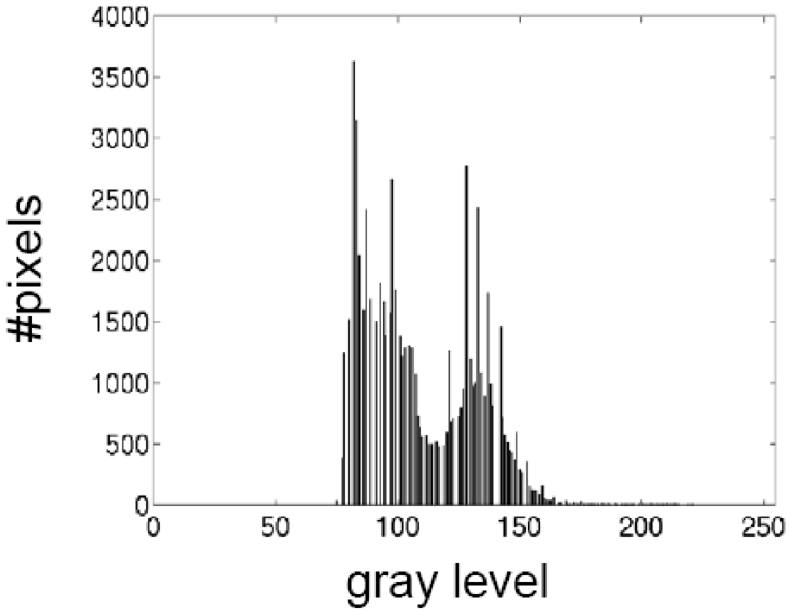
Discrete approximation of $g = T(f) = \int_0^f p_f(\alpha)\,d\alpha$

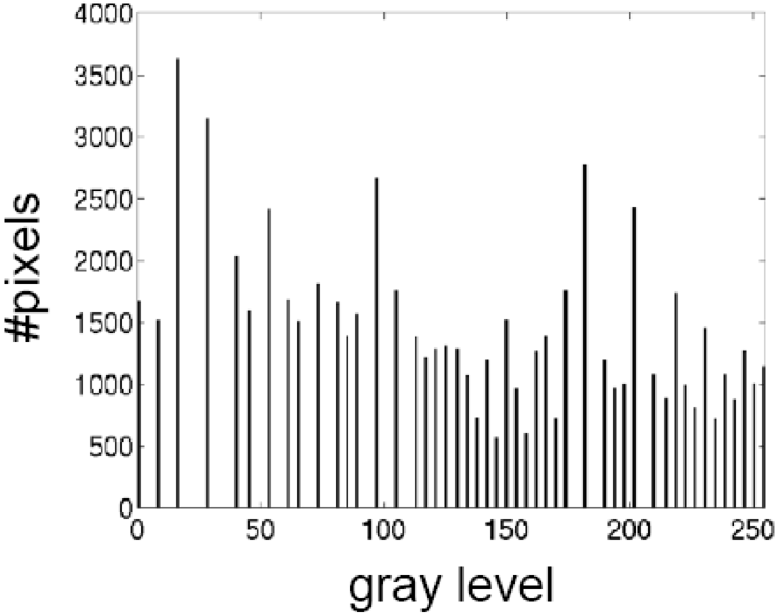
$$g_k = T(f_k) = \sum_{i=0}^{k} P_i$$

The resulting values $\boldsymbol{g_k}$ *are in the range [0,1] and need to* be scaled and rounded appropriately

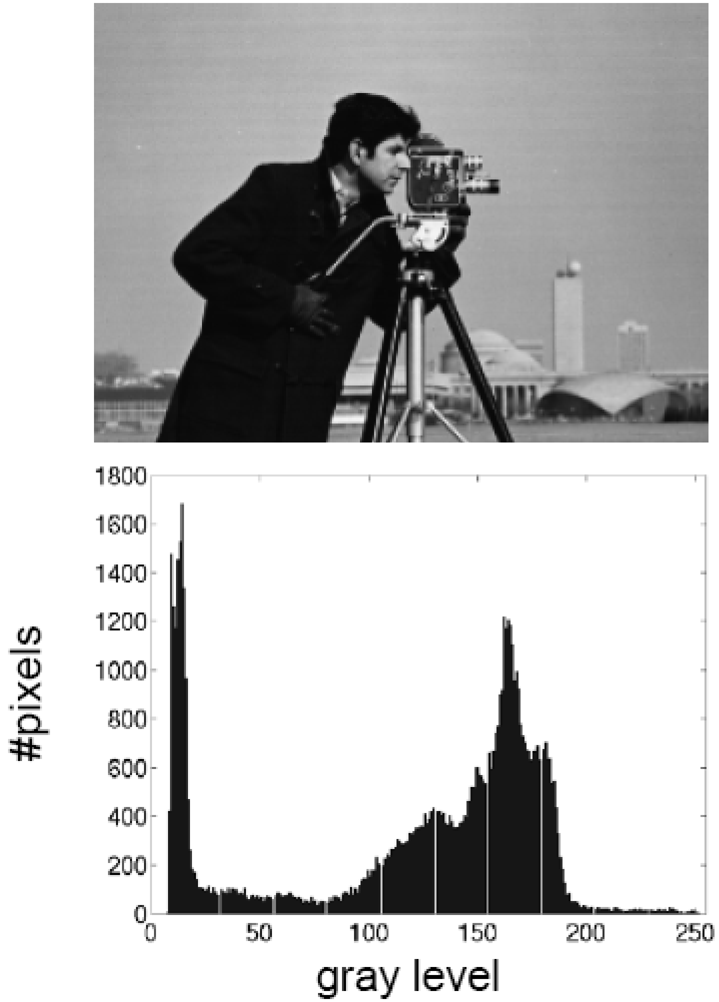
Histogram equalization example

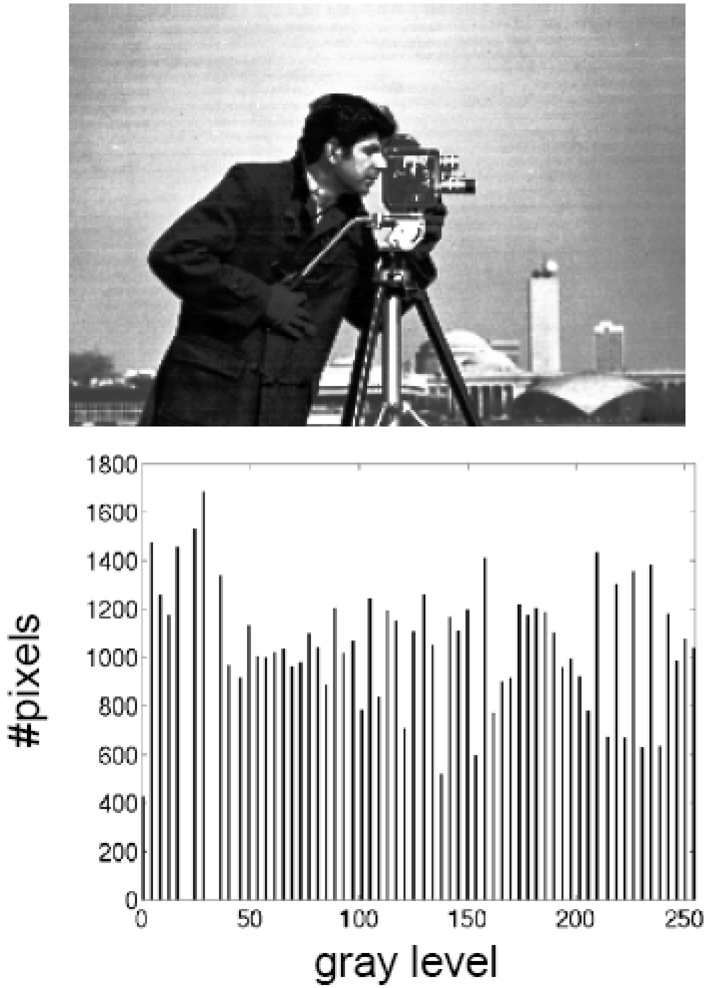
Histogram equalization example





Histogram equalization example



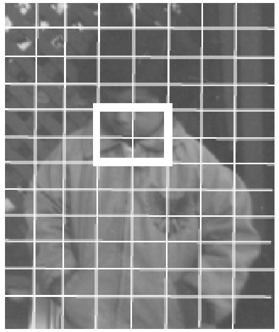


Histogram equalization example

Adaptive Histogram Equalization

Apply histogram equalization based on a histogram obtained from a portion of the image

Sliding window approach:
different histogram (and mapping) for every pixel



Tiling approach:
subdivide into overlapping regions, mitigate blocking effect by smooth blending between neighboring tiles

Must limit contrast expansion in flat regions of the image, e.g. by clipping individual histogram values to a maximum

Adaptive Histogram Equalization

Original | Global histogram | Tiling 8x8 histograms | Tiling 32x32 histograms

Adaptive Histogram Equalization

Original image *Tire*

Tire after
equalization of
global histogram

Tire after
adaptive histogram equalization
8x8 tiles

Point Operations Between Images

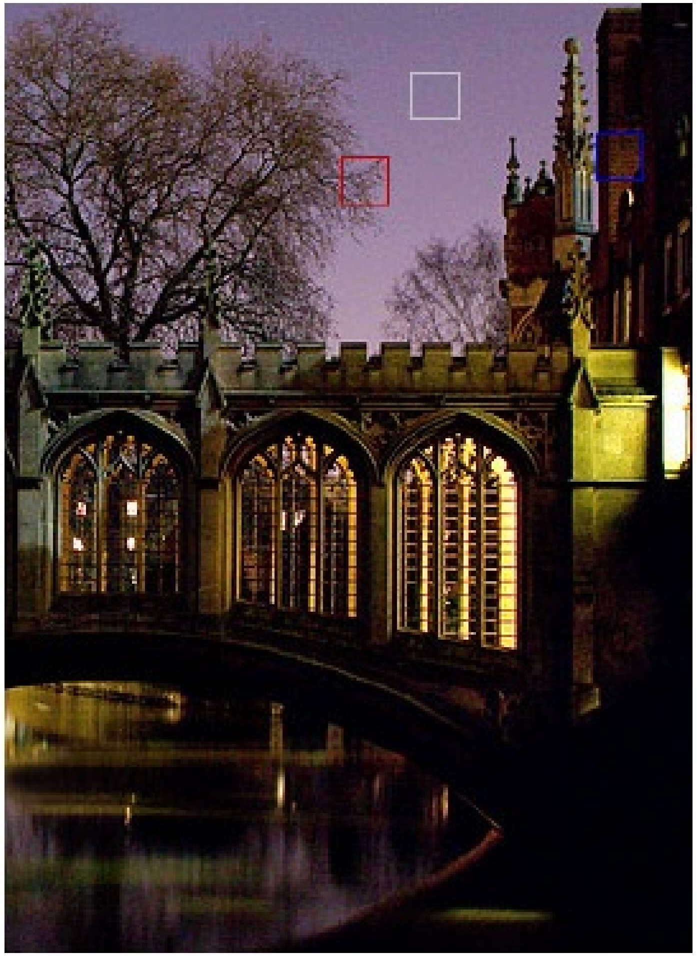
Image averaging for noise reduction

Combination of different exposure for high-dynamic range imaging

Image subtraction for change detection

Accurate alignment is always a requirement

Image averaging for noise reduction



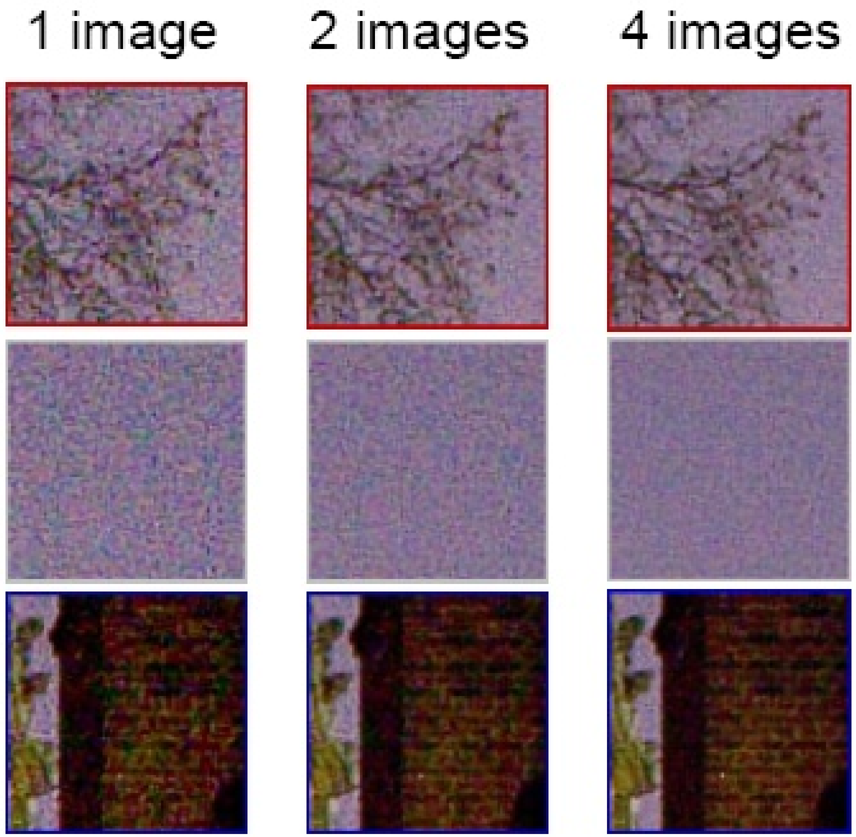


Image averaging for noise reduction

Take N aligned images $f_1(x,y), f_2(x,y), \cdots, f_N(x,y)$

Average Image: $$\overline{f(x,y)} = \frac{1}{N}\sum_{i=1}^{N} f_i(x,y)$$

Mean squared error vs. noise-free image *g*

$$E\left\{\left(\bar{f}-g\right)^2\right\} = E\left\{\left(\left(\frac{1}{N}\sum_i f_i\right)-g\right)^2\right\} = E\left\{\left(\left(\frac{1}{N}\sum_i (g+n_i)\right)-g\right)^2\right\}$$

$$= E\left\{\left(\frac{1}{N}\sum_i n_i\right)^2\right\} = \frac{1}{N^2}\sum_i E\left\{n_i^2\right\} = \frac{1}{N}E\left\{n^2\right\}$$

provided $E\{n_i n_j\} = 0 \; \forall i, j$

$E\{n_i\} = E\{n\} \; \forall i$

High-dynamic range imaging

16 exposures, one f-stop (2X) apart

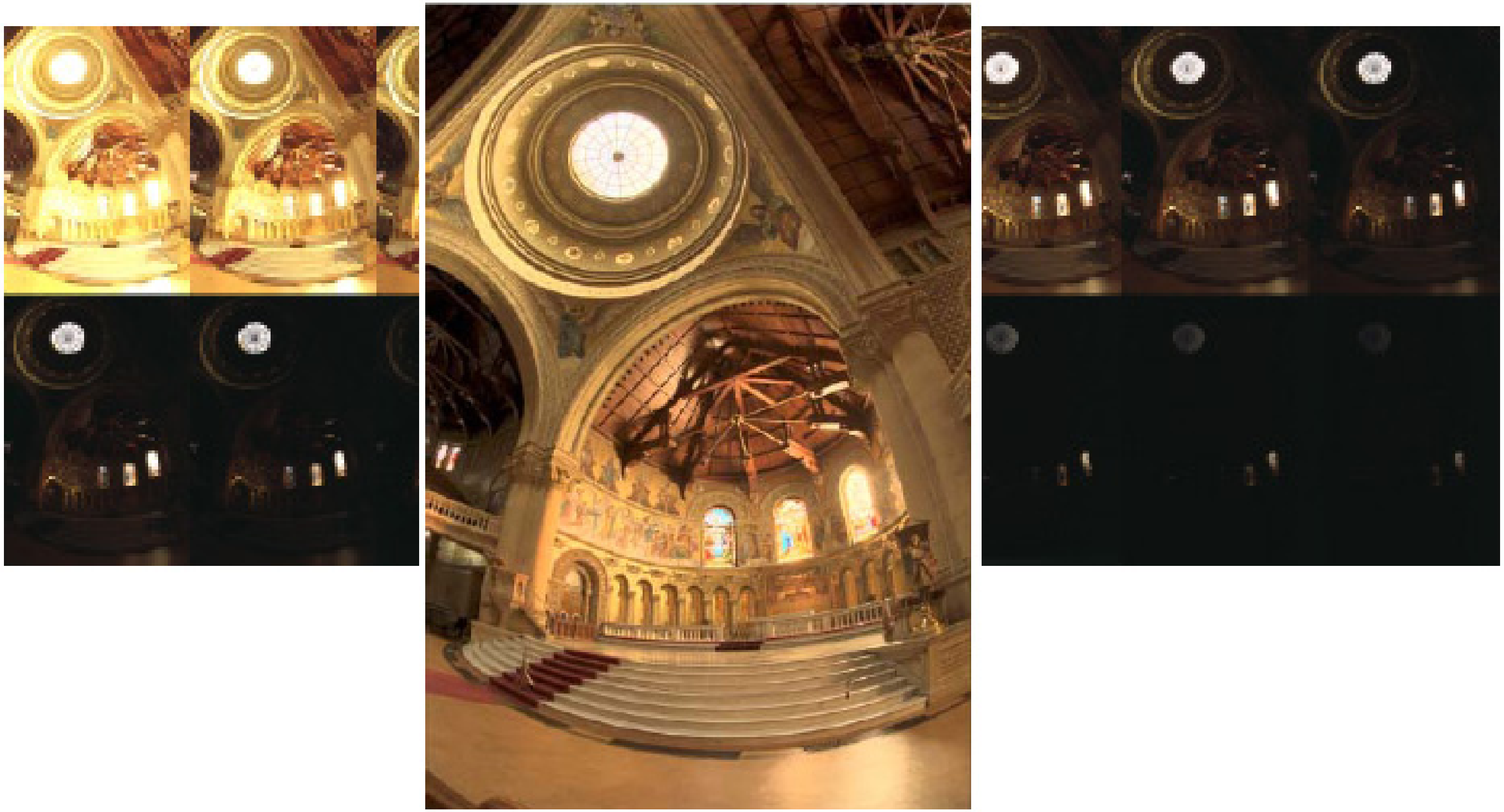
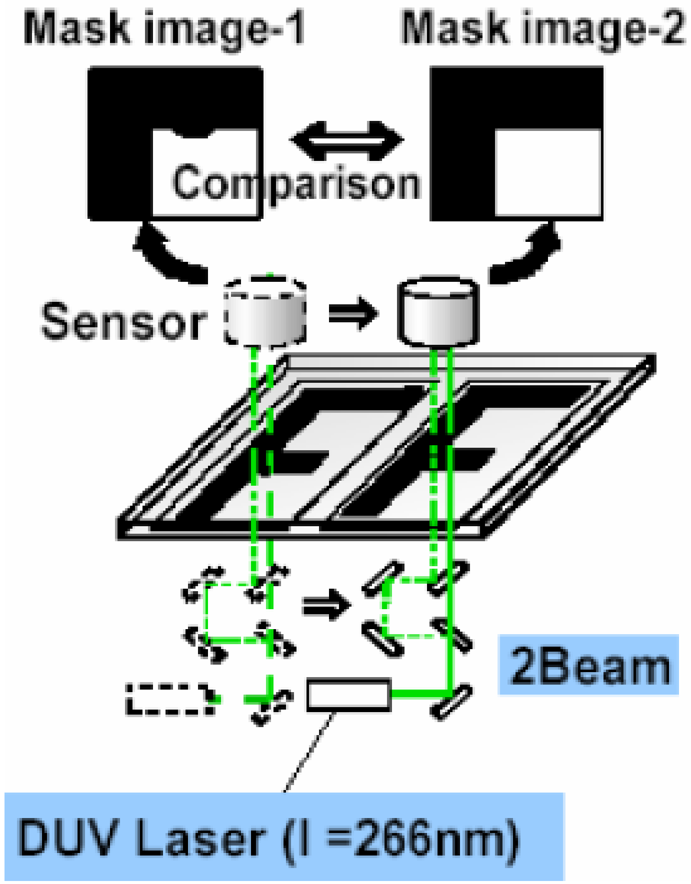


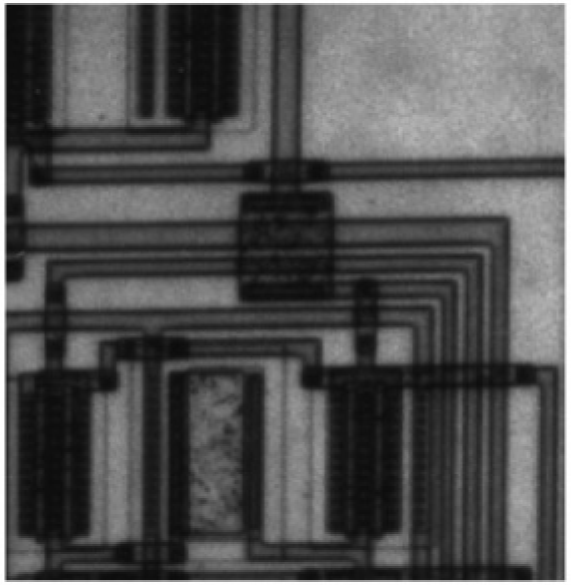
Image subtraction

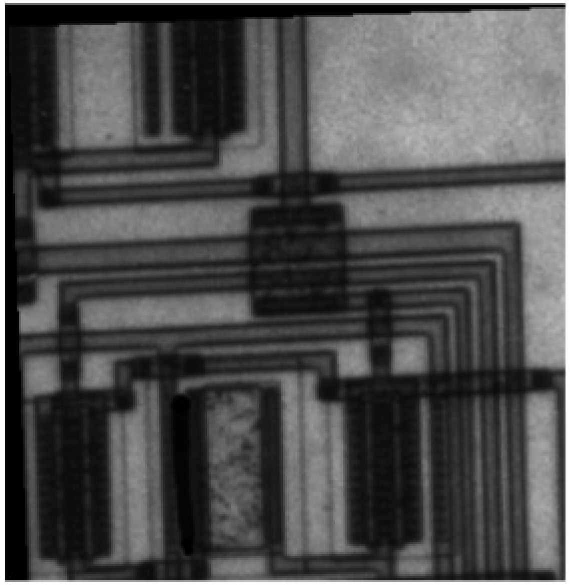
Find differences/changes between 2 mostly identical images

Example from IC manufacturing: defect detection in photomasks by die-to-die comparison

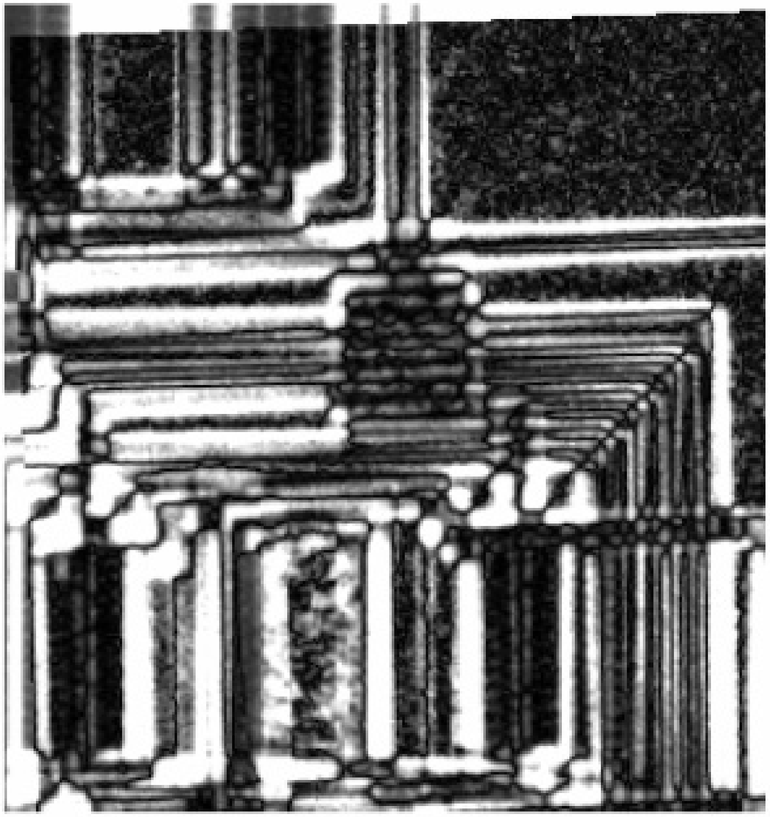


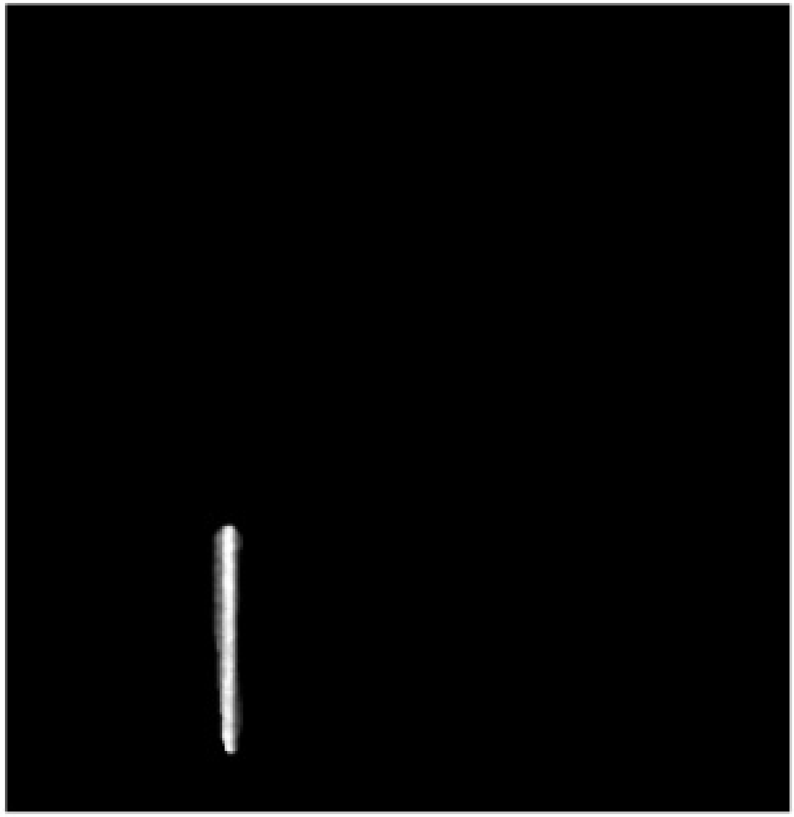
Where is the Defect?





Absolute Difference Between Two Images





Digital subtraction angiography