Pre-processing

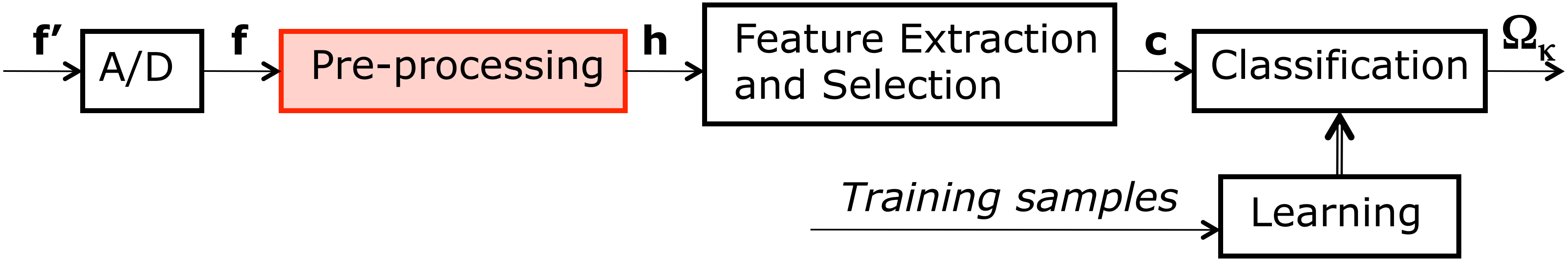
Histogram Equalization and Thresholding

Dr. Elli Angelopoulou

Lehrstuhl für Mustererkennung (Informatik 5)

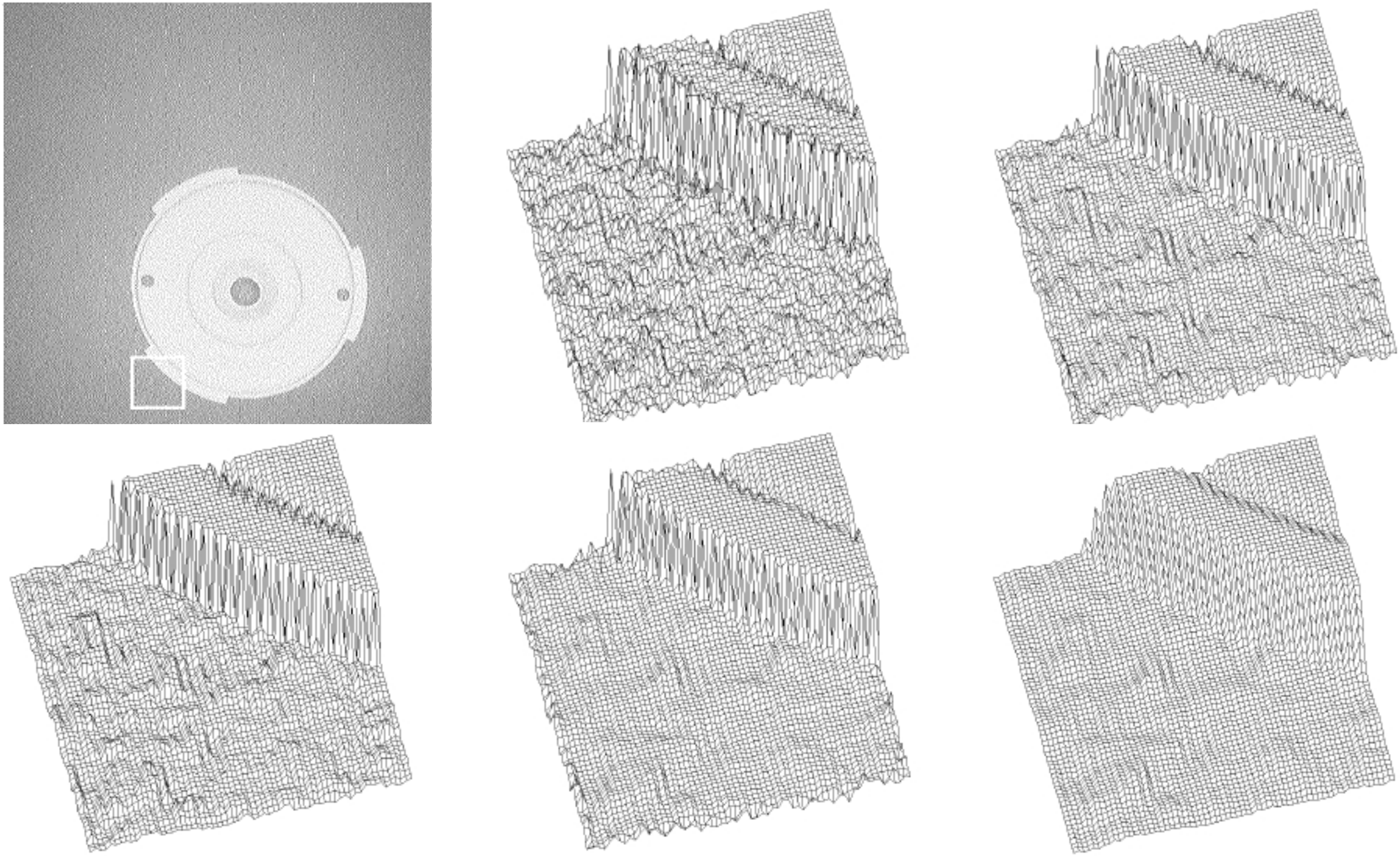
Friedrich-Alexander-Universität Erlangen-Nürnberg

Pattern Recognition Pipeline



- The goal of pre-processing is to transform a signal f to another signal h so that the resulting signal h
  - makes subsequent processing easier
  - makes subsequent processing better (more accurate)
  - makes subsequent processing faster

Pre-processing Example



Low Contrast Signal

- Quite often the captured image has a small dynamic range, i.e. all the pixels in an image have values that are within a small range of values, typically much smaller than the capability of the sensor.

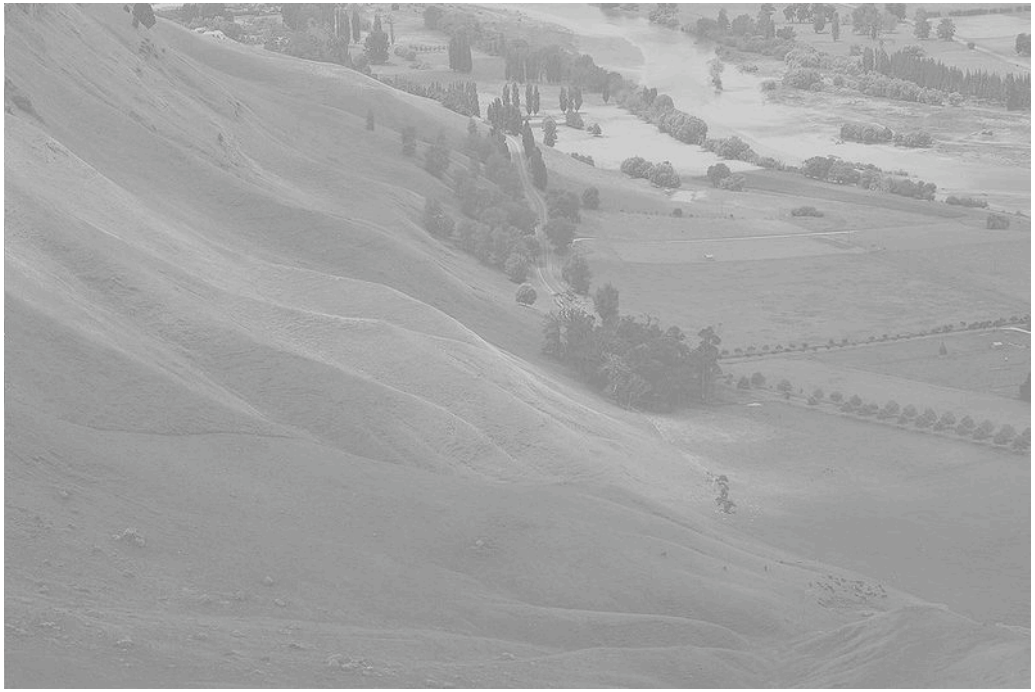
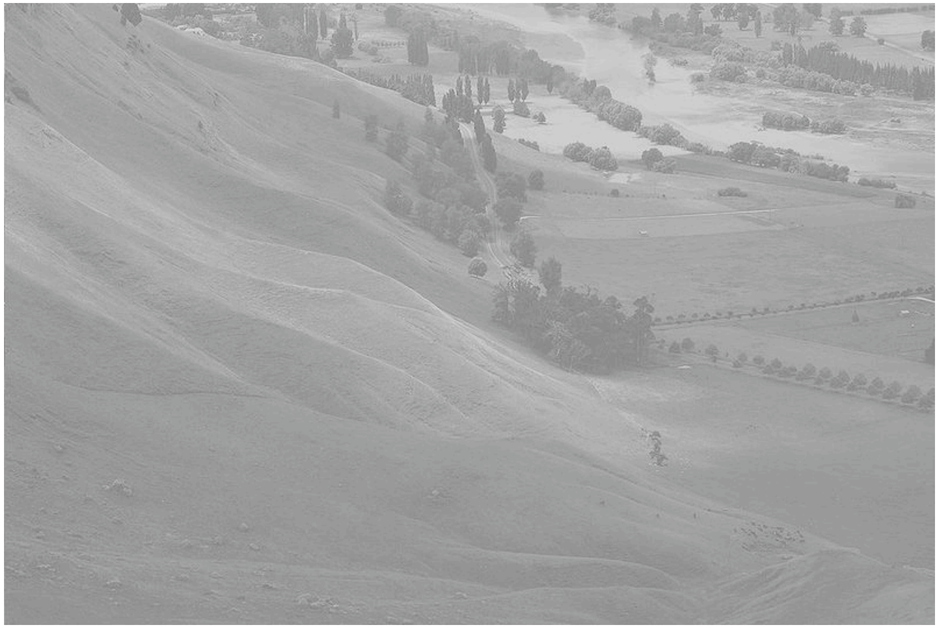
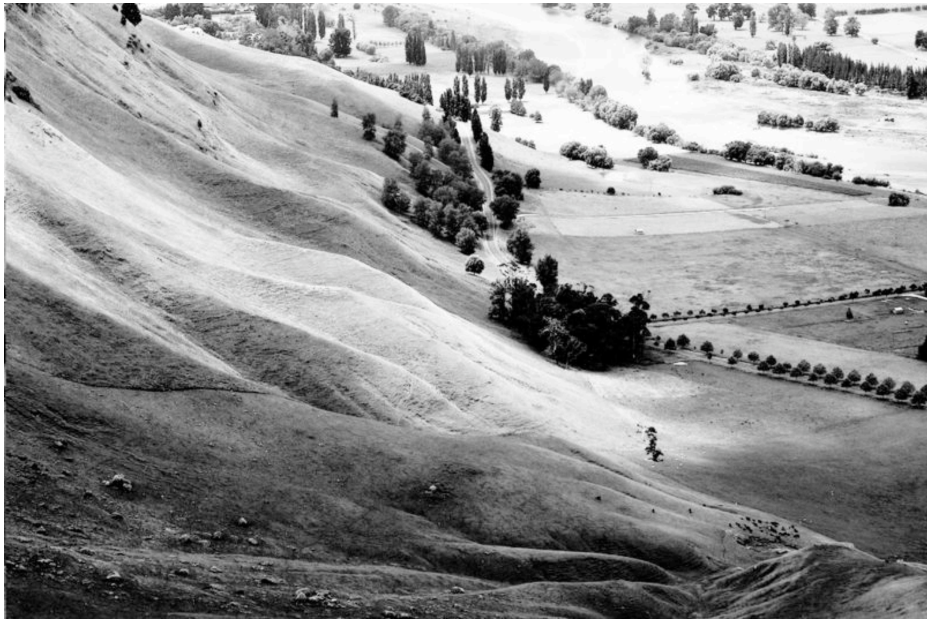


Image courtesy of Phillip Capper, http://en.wikipedia.org/wiki/File:Unequalized_Hawkes_Bay_NZ.jpg

Enhancement of Low Contrast Signal

- One goal of pre-processing is to improve the contrast by changing the intensity values in the image so that they now cover a wider range of values.



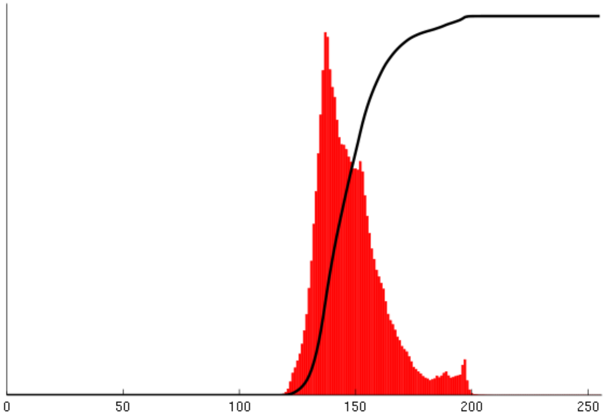


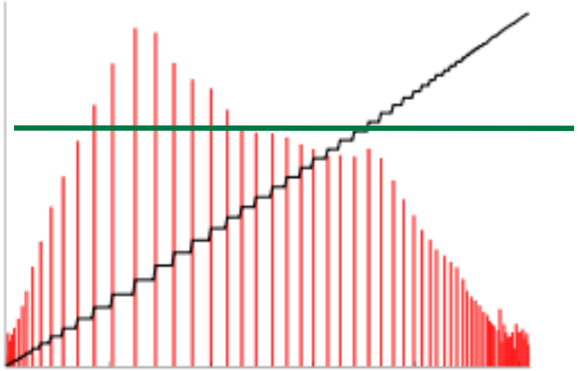
Images courtesy of Phillip Capper, http://en.wikipedia.org/wiki/File:Unequalized_Hawkes_Bay_NZ.jpg, http://en.wikipedia.org/wiki/File:Equalized_Hawkes_Bay_NZ.jpg

Histogram Equalization

- Histogram equalization is a method for improving low images with a limited range of values.
- It achieves that by redistributing the intensities in the image.
- It is based on the frequency with which different intensity values appear in the image (*histogram*).
- It effectively spreads out the most frequently used intensities.
- The goal of histogram equalization is to produce an image where the various intensity values appear almost equally often in the image (*uniformly distributed*)

Image Histogram





Images courtesy of Phillip Capper, http://en.wikipedia.org/

- A histogram plots for each gray level value the frequency with which that value occurs (shown in red)
- The goal of histogram equalization is to have an almost horizontal distribution of values.

Formal Definition of Histogram Equalization

- Histogram equalization is a transformation $g()$ of sample values $f_1, f_2, f_3, \ldots$ so that $g(f_i)$ is [0,1]-uniformly distributed.
- In other words we are looking for a function $f' = g()$ such that $p(f') \in [0,1]$ and each value of f' has equal probability of occurring.
- Does such a function $g()$ exist?
- Yes, it is based on the cumulative distribution function:

$$g(f) = D_X(f)$$

Cumulative Distribution

- Let $p_X(f)$ be the probability distribution function of f.
- The cumulative distribution function (cdf) of f is

$$D_X(f) = \int_{-\infty}^{f} p_X(h)dh$$

- cdf is always a monotonic function that at infinity approaches 1, $\lim_{f \to \infty} D_X(f) = 1$ and $\lim_{f \to -\infty} D_X(f) = 0$.

PDF and CDF plots

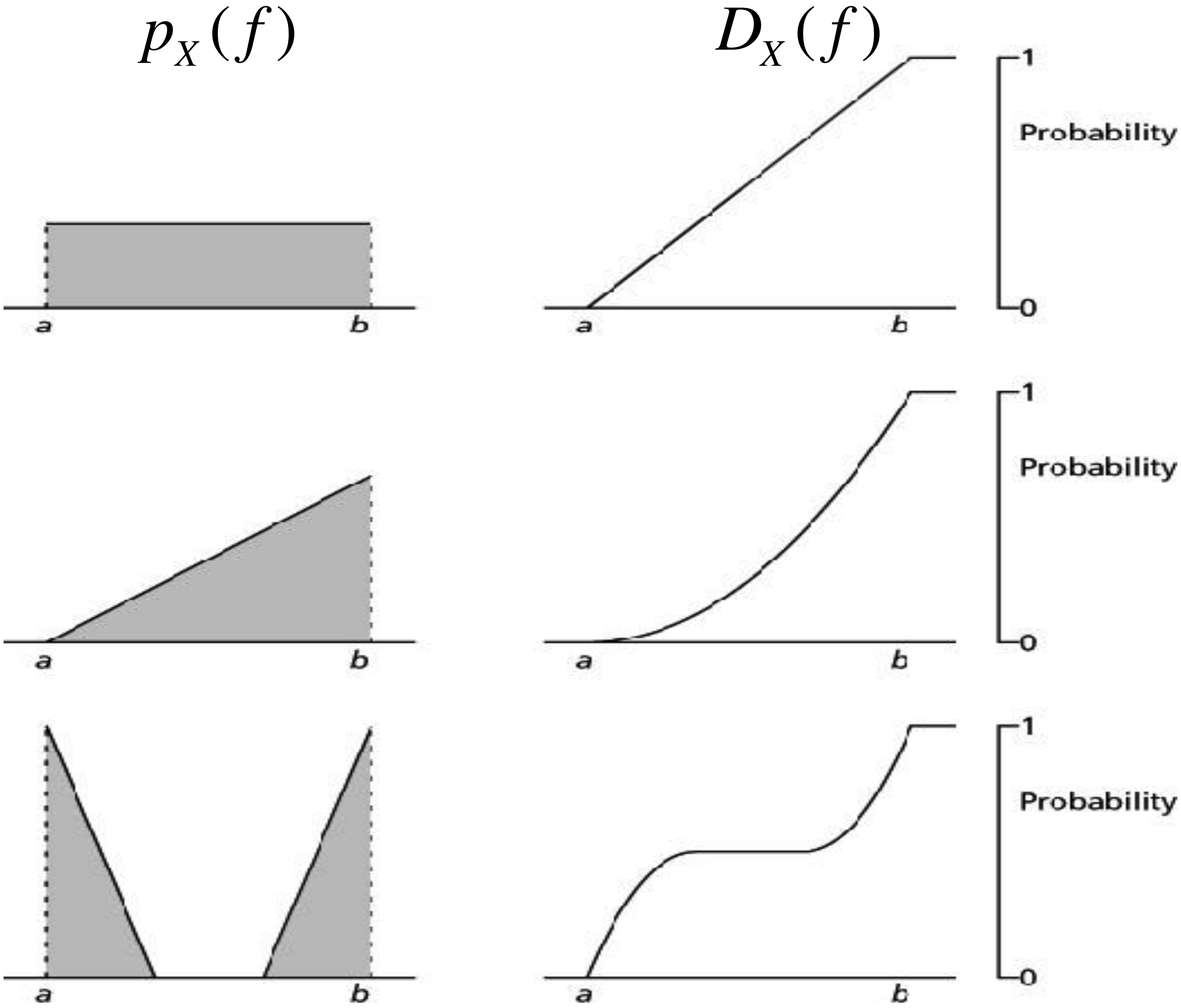
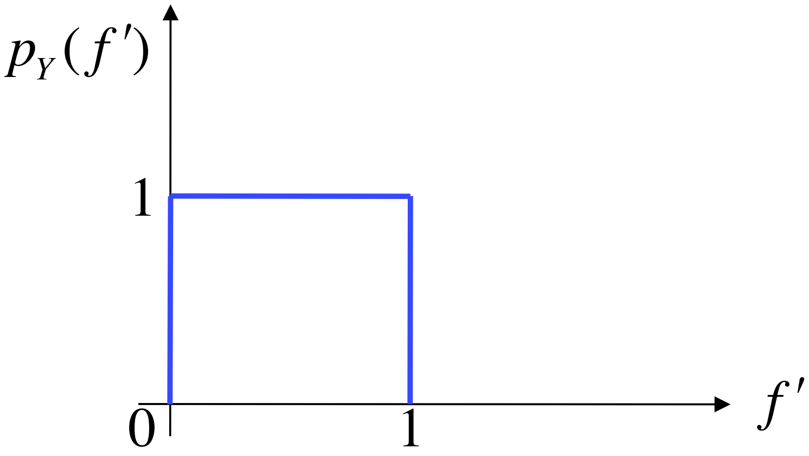
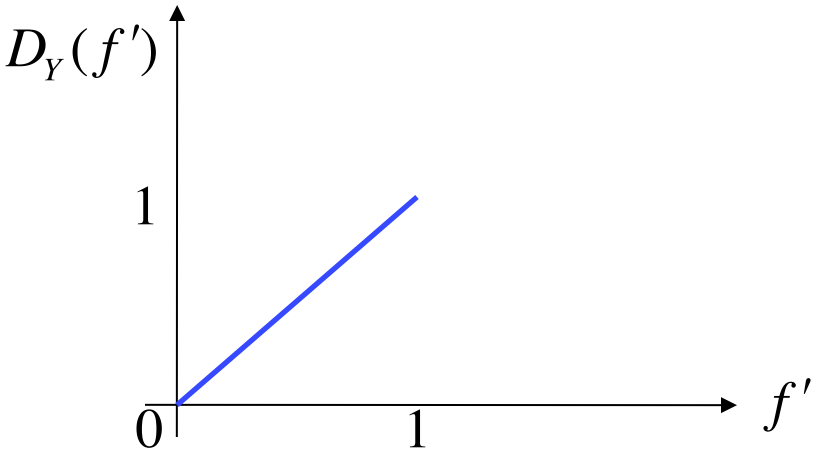


Image courtesy of http://www.answers.com/topic/cumulative-distribution-function

CDF of a CDF

- Let $f' = D_X(f)$.
- Claim: f' is [0,1]-uniformly distributed.
- f' is in [0,1] by definition
- Still need to show that f' is uniformly distributed.
- How does the cdf look like for a uniform distribution?
- Claim: If $p_Y(f')$is a [0,1]-uniform distribution, then its cdf is the variable itself, $D_Y(f') = f'$ and vice-versa.





CDF of a CDF - continued

- Recall that $f' = D_X(f)$ is itself a cdf function and as such it is a monotonic function. This means that:

$$f_1 \leq f_2 \Rightarrow D_X(f_1) \leq D_X(f_2) \Rightarrow f_1' \leq f_2'$$

- How does the cdf of a cdf look like? How does $D_Y(f')$ look like?

$$D_Y(f_1') = p_Y(f' \leq f_1') = p_Y(D_X(f) \leq D_X(f_1)) =$$

$$= p_X(f \leq f_1) = D_X(f_1) = f_1'$$

- But if $D_Y(f') = f'$ then f' has a [0,1]-uniform distr.
- So when we use the mapping $f' = D_X(f)$ the values end-up being uniformly distributed.

Histogram Equalization Algorithm

1. Compute the histogram of a given image
2. Compute its cumulative distribution function.
3. Break the vertical axis of the cdf plot, into *n* equidistant buckets, where *n* is the number of gray values in the output image.
4. Each bucket in the vertical axis corresponds to a gray value of the output "equalized" image.
5. Let $K = \dfrac{\text{\# of pixels}}{n}$
6. Then assign the first *K* pixel values (in the horizontal x-axis) of the cdf to the 1st vertical bucket. The next *K* pixel values in the 2nd block of the cdf get mapped to the next bucket etc.
7. In the resulting image each of the *n* intensities has the same probability of occurring. The pixels are spread evenly across the entire range of these *n* pixel values. The image has the highest possible contrast.

Histogram Equalization - Clarifications

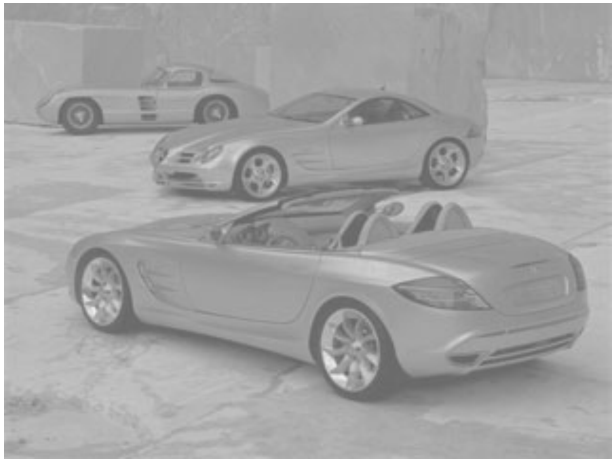
- The redistributed values in the tessellation of the vertical axis correspond to the histogram of the equalized image.

- A grey value f is mapped from the cumulative distribution function $D(f)$ to a new “equalized” grey value as follows:

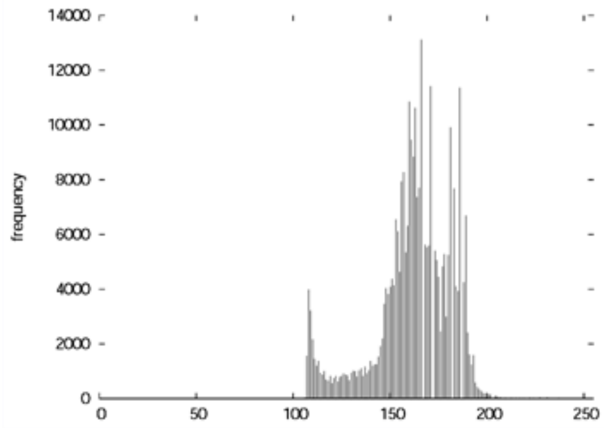
$$f' = \mathrm{g}(f) = \mathrm{round}\left(\frac{\mathrm{D}(f) - D_{\min}}{\#\mathrm{pixels} - D_{\min}}(L-1)\right)$$

where $D_{\min}$ is the smallest non-zero value in the cdf and L is the number of levels in the new image.

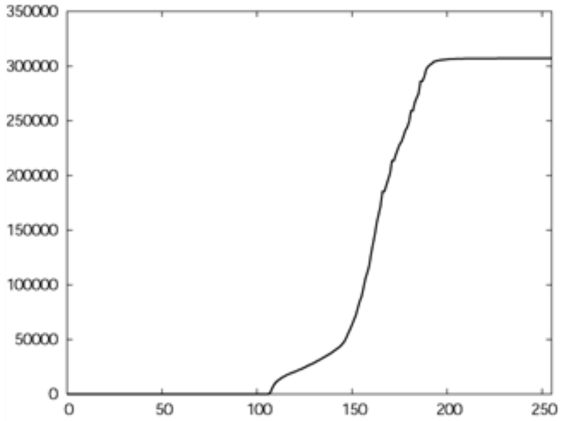
Histogram Equalization Example



Original image



Histogram of original image



cdf of original image

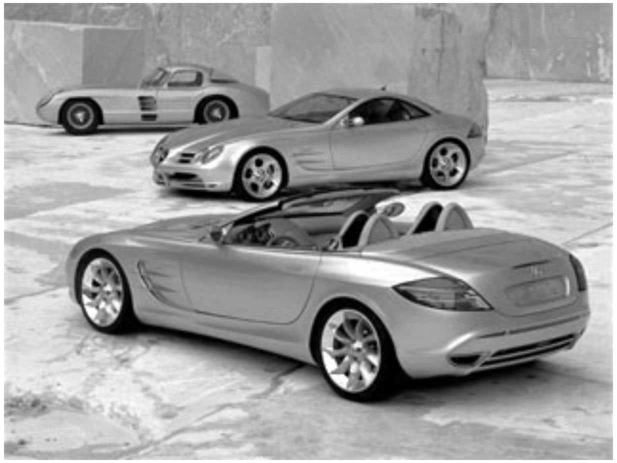
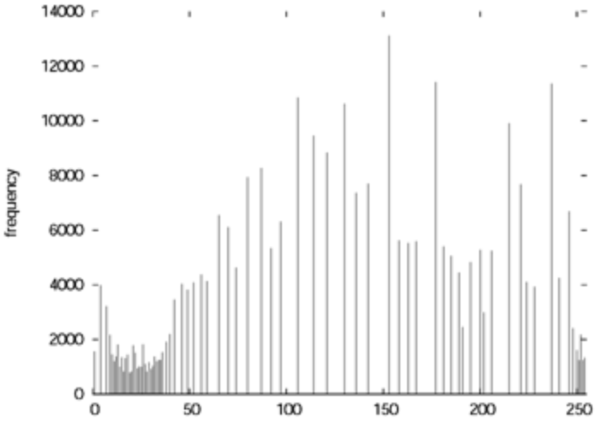
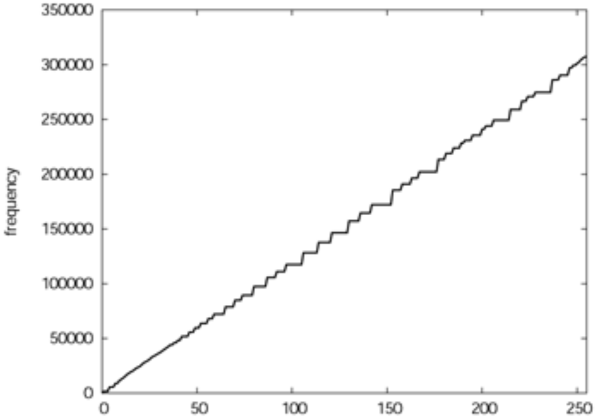


Image after histogram equalization



Histogram of hist. equalized image



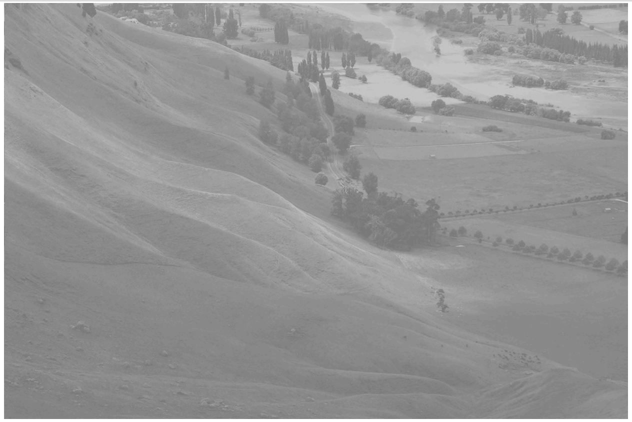
cdf of hist. equalized image

Images courtesy of James Matthews, http://www.generation5.org/content/2004/histogramEqualization.asp

Some Comments on Histogram Equalization

- It is a straightforward, easy to implement technique.
- It is not computationally expensive.
- Works well in images where both the foreground and the background are dark (or both are bright).
- In cases where spatial correlation is more important that the intensity values themselves, a histogram equalized image facilitates further detection and analysis.
- However, histogram equalization often produces unrealistic looking images, or can introduce undesirable effects.

Intensity Stretching vs Contrast Enhancement



Original image

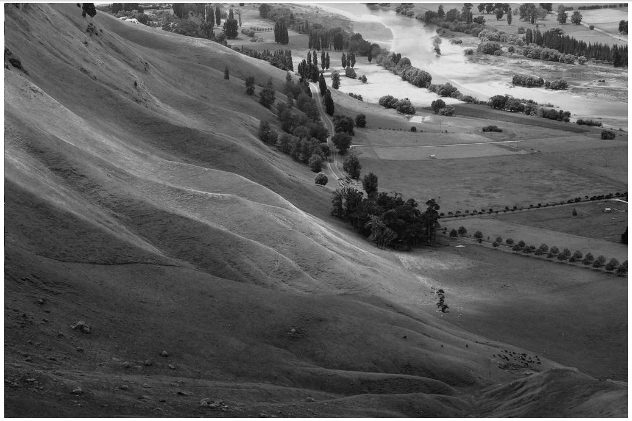


Image after histogram stretching

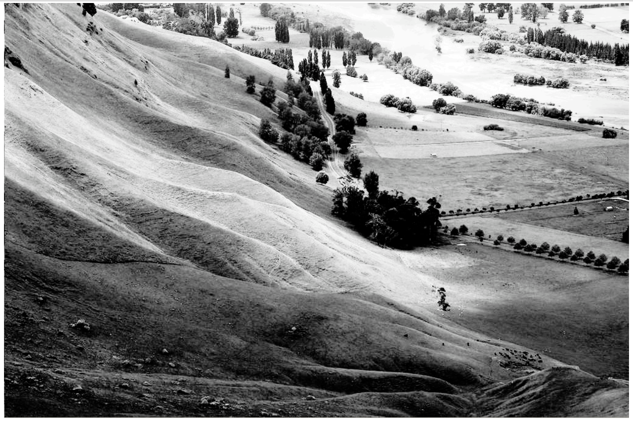
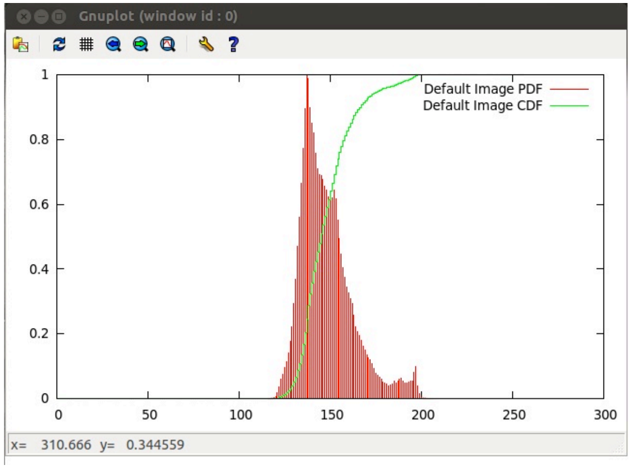
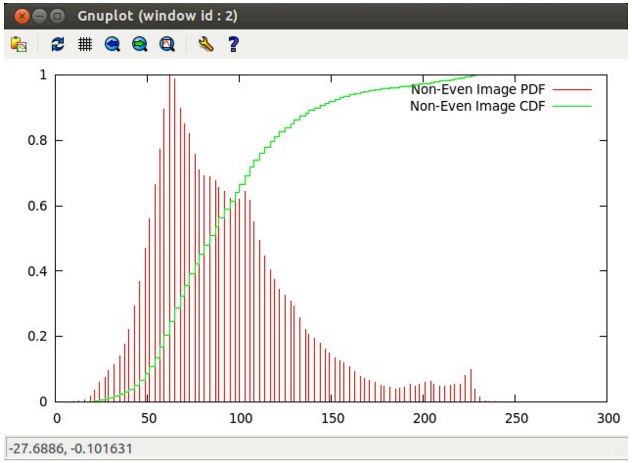


Image after histogram equalization



x= 310.666 y= 0.344559

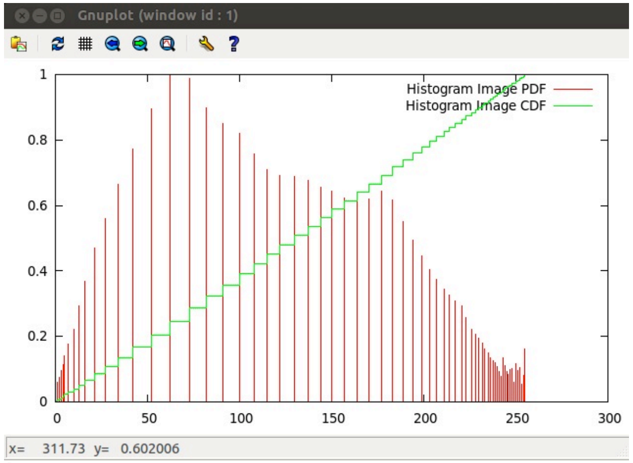
Histogram and cdf of original image



-27.6886, -0.101631

most intensity levels in a small interval
=> most parts of the image have colors of that interval
=> low overall contrast
=> contrast sharpened equally in all regions of the image

Histogram and cdf of stretched image



x= 311.73 y= 0.602006

even distribution of intensity levels
=> additional contrast added to the most present colors
=> high overall contrast
=> low contrast in areas with rare intensities

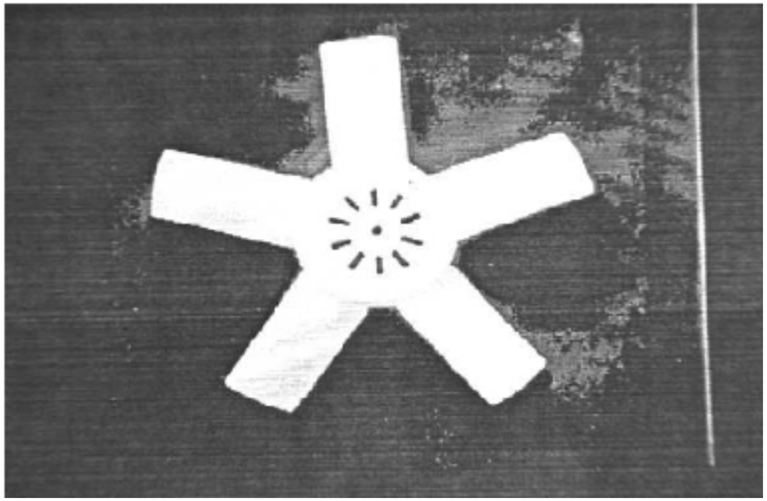
Histogram and cdf of equalized image

Images by Stefan Ploner, http://ziegenpeter.zi.ohost.de/uni/ipr/02-equidistantIntensityLevels_vs_histogram.png

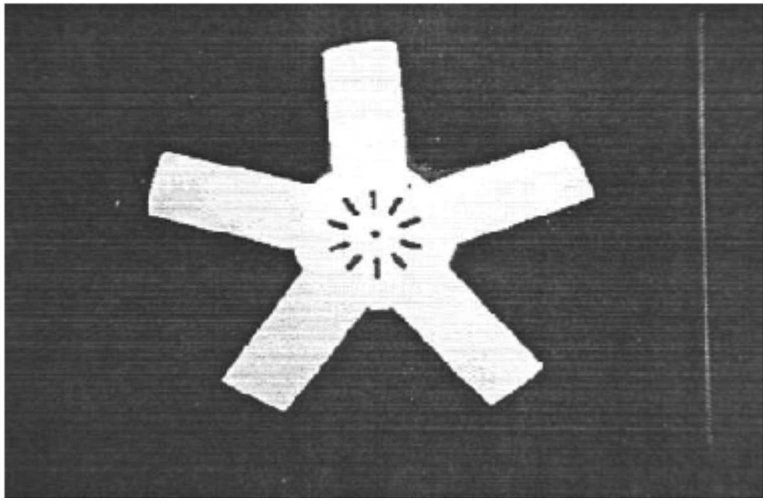
Thresholding

- **Thresholding** is the process of converting a gray-scale image of L_1 gray levels, into one of L_2 gray levels, where $L_2 << L_1$, based on "border" values.
- Typically $L_2 = 2$, in which case the output is a binary image, i.e. an image with only 2 gray-levels, 0 and 1.
- The process of converting an image to a 2 level image is called **binarization**.
- Threshold = brink, border, verge.
- Example applications:
  - Quality control system where the perimeter of the product needs to be evaluated. Light colored products on a dark conveyor belt.
  - Contrast Enhanced CT, where the blood vessels look really dark on a light background. Goal is to measure the vessel geometry.

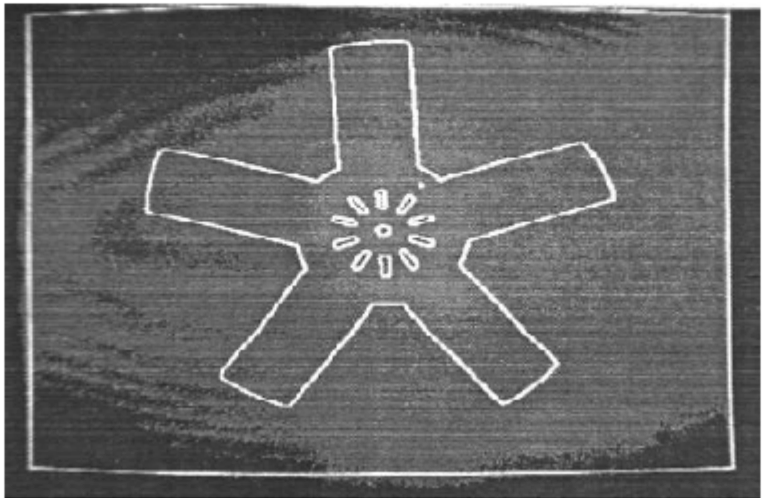
Thresholding Example



Original image

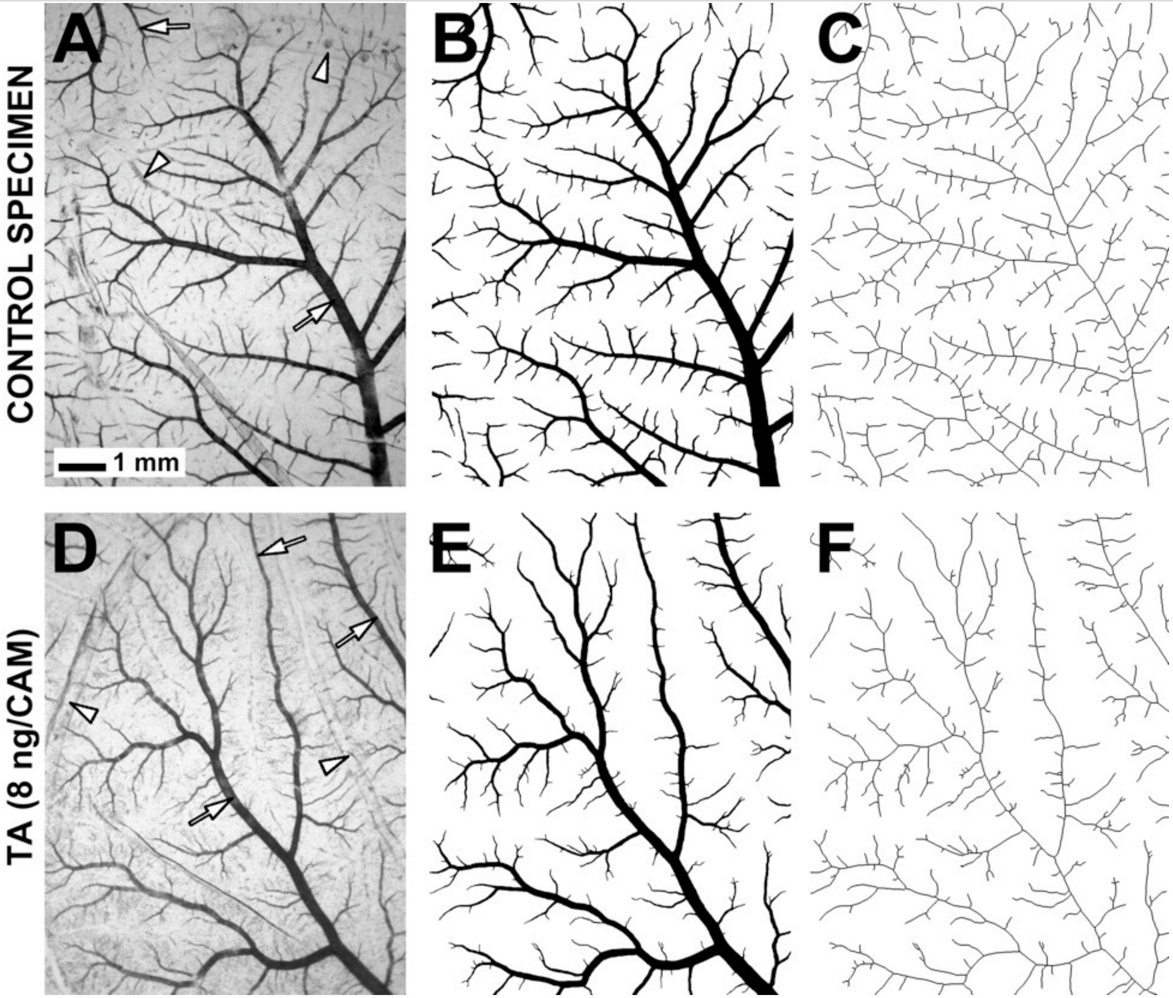


Binary image after thresholding



Perimeter detection on binary image

Another Thresholding Example



Original image

Binary image

Image after medial axis computation

Images by T.L. McKay et al., "Selective Inhibition of Angiogenesis in Small Blood Vessels and Decrease in Vessel Diameter throughout the Vascular Tree by Triamcinolone Acetonide", Invest. Ophthalmol. Vis. Sci., Vol. 49, No. 3, March 2008, pp. 1184-1190, http://www.iovs.org/content/49/3/1184.full

How Does Thresholding Work?

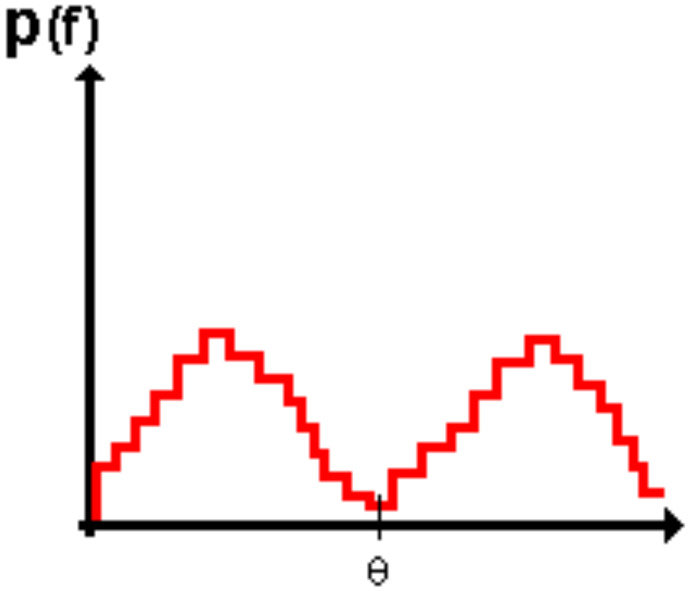
- Thresholding transformation $T(f_{ij})$ for $L_2 = 2$:

$$T(f_{ij}) = \begin{cases} 0 & \text{if } \; f_{ij} \leq \theta \\ 1 & \text{otherwise} \end{cases}$$

- Which value θ should we use?
- How can we choose a value for θ based on what we have done so far in class?
- We know how to compute histograms, cdfs and break the cdf into blocks of equal width.
- Idea: Build the cdf and break into two blocks of equal width.
- May not result in meaningful binary images. The output image will have an equal number of 0s and 1s.

Bimodal Histograms

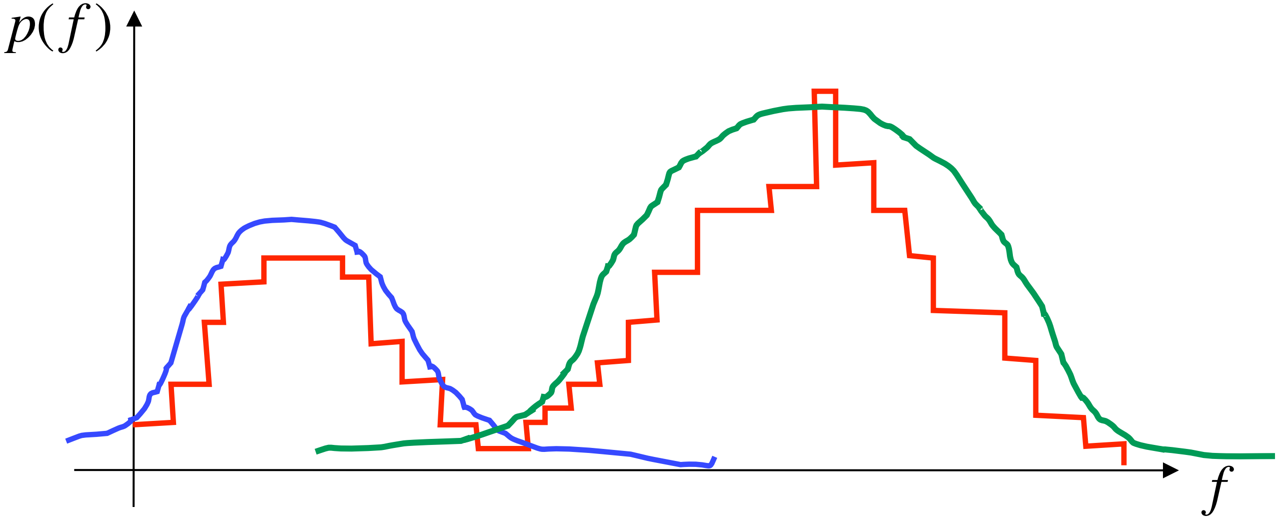
- The simplest binarization algorithms assume that the background has one intensity value and the object(s) of interest have another intensity value. With all the shading variations we end up with a bimodal histogram.



- Bimodal histogram: a histogram with two local maxima (one for foreground and one for background).
- Idea: pick a value between the two maxima and use it as a threshold value θ.
- Everything below the threshold is assigned a 0 and everything above is assigned a 1.

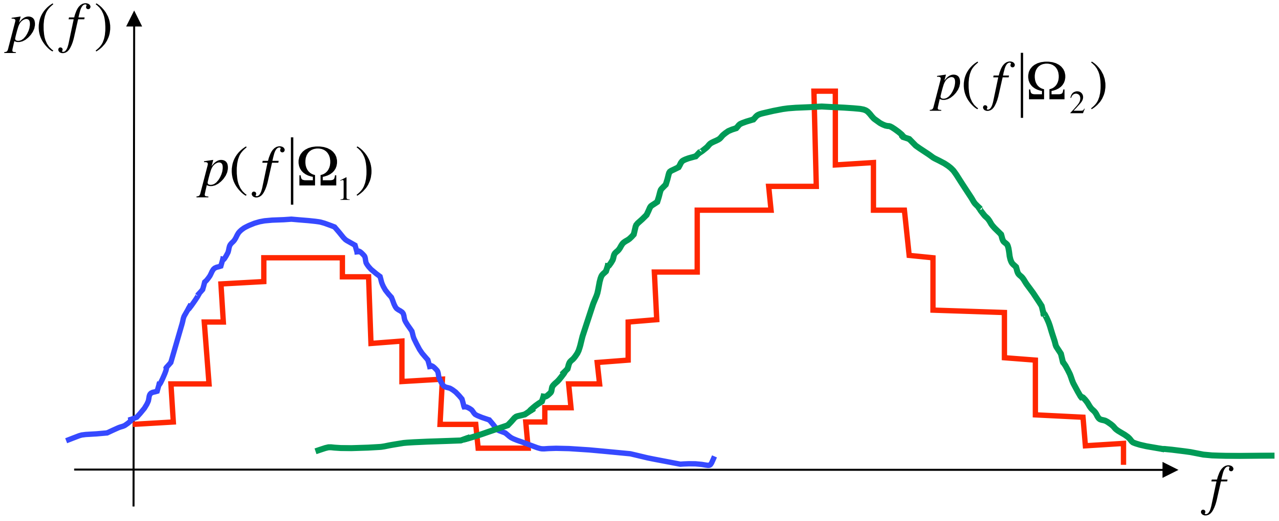
Intersection of Gaussians

- A rigorous method for selecting a threshold value θ that is close to the minimum between the two modes is to *treat each mode as a Gaussian*.
- Methodology: Fit 2 Gaussians and compute their intersection.



Intersection of Gaussians - continued

- One Gaussian describes the probability of a particular pixel value f occurring, given that the pixel belongs to the background, i.e. the pixel is in class Ω_1, $p(f|\Omega_1)$.
- The 2nd Gaussian describes the probability of a particular pixel value f occurring, given that the pixel belongs to the foreground, i.e. the pixel is in class Ω_2, $p(f|\Omega_2)$.



Sum of Gaussians

- If we have these 2 densities, $p(f|\Omega_1)$ and $p(f|\Omega_2)$ we can not just sum them up. Their sum will not be equal to 1, and the integral over all possible values of a pdf of must sum up to 1: $\int_{-\infty}^{\infty} p(f)df = 1$.
- Instead, we have to take under consideration the probability of each class occurring:

$$p(f) = p(\Omega_1)p(f|\Omega_1) + p(\Omega_2)p(f|\Omega_2)$$

where $p(\Omega_1)$ is the probability of having a gray value that belongs to the background class and $p(\Omega_2)$ is the probability of having a gray value that belongs to the foreground class.

Sum of Gaussians - continued

- How do we compute $p(\Omega_1)$? Recall that $p(\Omega_1)$ is the probability of having a gray value that belongs to the background class. Thus,

$$p(\Omega_1) = \frac{\#\text{pixels such that } \{f \leq \theta\}}{\text{total } \#\text{pixels in the image}}$$

and

$$p(\Omega_2) = 1 - p(\Omega_1)$$

- The resulting sum satisfies the proper pdf behavior.

$$\int_{-\infty}^{\infty} p(f)\,df = p(\Omega_1)\underbrace{\int_{-\infty}^{\infty} p(f|\Omega_1)\,df}_{=1} + p(\Omega_2)\underbrace{\int_{-\infty}^{\infty} p(f|\Omega_2)\,df}_{=1}$$

$$\Rightarrow \int_{-\infty}^{\infty} p(f)\,df = p(\Omega_1) + p(\Omega_2) = 1$$

Gaussian Mixture

- If a pdf can be decomposed to a linear combination of Gaussian densities, then it is called a **Gaussian mixture**.
- Gaussian mixtures are widely used in pattern recognition.
- If you have a random variable and you don't know its pdf, use a so-called **convex combination** of Gaussians to approximate it.
- The term **convex** combination denotes that the coefficients (i.e. weights) of the different Gaussians have to sum up to 1.

Computing the Intersection

- Let $\alpha = p(\Omega_1)$.
- Replace $p(f|\Omega_1)$ and $p(f|\Omega_2)$ with Gaussian functions. Then,

$$p(f) \approx \alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(f-\mu_1)^2}{2\sigma_1^2}} + (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(f-\mu_2)^2}{2\sigma_2^2}}$$

  where μ_1,σ_1 are the mean and standard deviation of the Gaussian that approximates $p(f|\Omega_1)$ and μ_2,σ_2 are the mean and standard deviation of the Gaussian that approximates $p(f|\Omega_2)$

- We want to compute the intersection of the 2 Gaussians.
- At the point of their intersection we have

$$\alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(f-\mu_1)^2}{2\sigma_1^2}} = (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(f-\mu_2)^2}{2\sigma_2^2}}$$

- We need to compute the value of f that satisfies this equation.

Solving the Equality

- To simplify the computations, we take the natural logarithm of the equation. The end result is a quadratic equation in f :

$$\ln\alpha - \ln\left(\sqrt{2\pi}\right) - \ln\sigma_1 - \frac{(f-\mu_1)^2}{2\sigma_1^2} = \ln(1-\alpha) - \ln\left(\sqrt{2\pi}\right) - \ln\sigma_2 - \frac{(f-\mu_2)^2}{2\sigma_2^2}$$

- Solve for the roots of this equation and set θ to the real root.
- So far we assumed that we know the parameters of the 2 Gaussians $\mu_1, \sigma_1, \mu_2, \sigma_2$.
- But given an image, we have to compute these parameters before we can compute the intersection.

Procedure for Intersection Computation

- So the whole process involves the following steps:

1. Compute the histogram.
2. Determine the values of $\alpha, \mu_1, \sigma_1, \mu_2, \sigma_2$ that best fit the histogram.
3. Analytically compute θ.

- This is a very computationally expensive process, which may not even fit the data very well.

- More often than not we have a histogram with one large and one considerably smaller hump.

Iterative Solution

- An alternative, though sub-optimal, solution is to compute an estimate of the intersection iteratively:

1. Choose a threshold value θ_0.
2. Estimate μ_1, σ_1 from the histogram values that are smaller than or equal to θ_0 and μ_2, σ_2 from the hist. values that are larger than θ_0. (see next slides)
3. Estimate α from the ratio of the number of pixels that are on each side of θ_0.
4. Compute an approximation error and adjust θ_0 accordingly. (see next slides)
5. Repeat steps 2 - 4 until the approximation error is minimal.

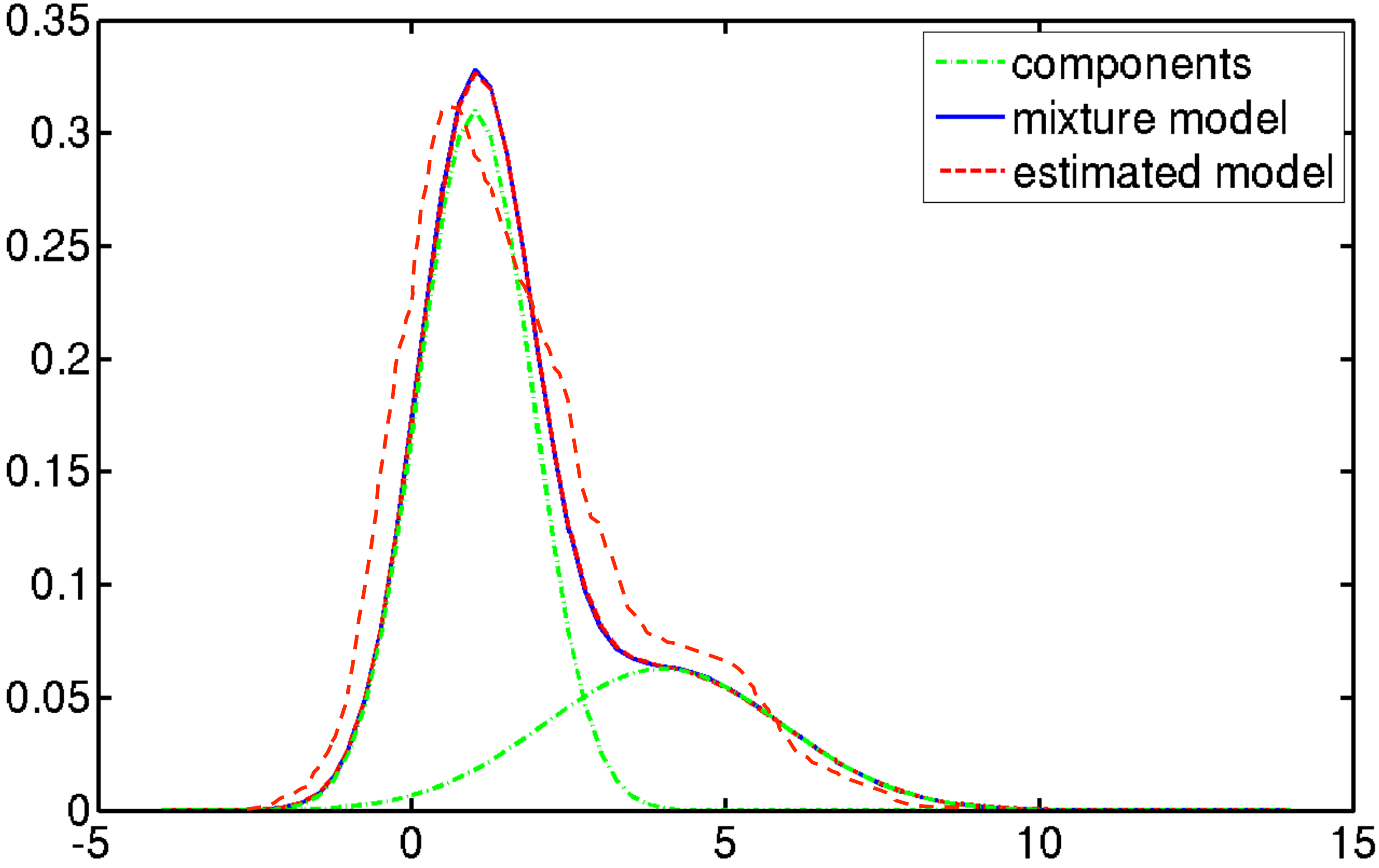
Maximum Likelihood (ML) Estimation

- Maximum Likelihood Estimation is a statistical method for fitting a statistical model to some data and providing estimates for the model’s parameters.
- ML Estimators are particularly well-suited for Gaussian distributions.
- Specifically to compute μ_1:
  - Use the methodology for computing the mean of a set of samples.
  - Sum up all the values smaller than or equal to θ_0 and divide by the number of pixels that have such values.
- To compute σ_1 use the formula for standard deviation:

$$\sigma_1 = \frac{1}{\#\text{pixels s.t. } (f \leq \theta_0)} \sqrt{\sum_{f \leq \theta_0} (f - \mu_1)^2}$$

- Compute μ_2, σ_2 similarly.

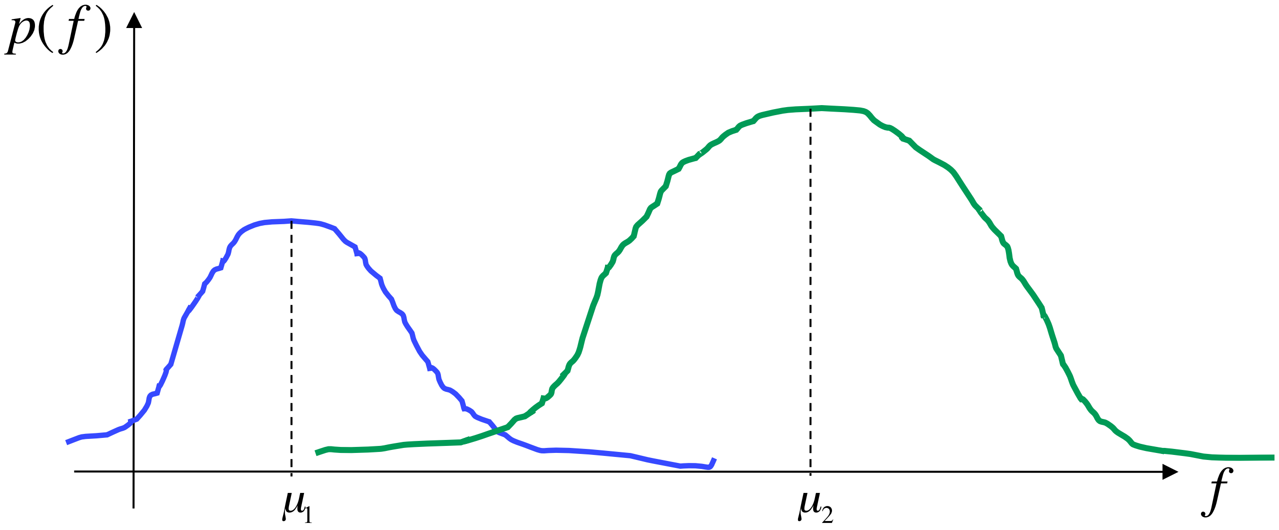
Approximation Error



Images courtesy of http://www.mathworks.de/matlabcentral/fx_files/24867/2/gaussian_mixture_model.png

Optimal Thresholding

- We are still studying images with bimodal histograms, which we approximate by 2 Gaussians.
- Can we express finding the threshold value θ as an optimization problem?
- Devise an objective function, such that when you maximize (minimize) it, it will give you the optimal threshold value for binarization.



Optimality via Error Metric

- After binarization all the values f s.t. $\{f \leq \theta\}$ will be mapped to value f_1' and all the values f s.t. $\{f > \theta\}$ will be mapped to value f_2'.

$$T(f) = \begin{cases} f_1' & \text{if } f_{ij} \leq \theta \\ f_2' & \text{otherwise} \end{cases}$$

- Such a mapping causes some loss of info, some error.
- For f_1' the associated error is: $\varepsilon_1 = \sum_{f=0}^{\theta} (f_1' - f)^2$
- For f_2' the associated error is:

$$\varepsilon_2 = \sum_{f=\theta+1}^{f_{\max}} (f_2' - f)^2$$

Error Metric

- One criterion for selecting a threshold value is then choose a value for θ that minimizes such errors.
- More specifically we want to compute:

$$\hat{\theta} = \arg\min_{\theta} \left(\sum_{f=0}^{\theta} (f_1' - f)^2 + \sum_{f=\theta+1}^{f_{\max}} (f_2' - f)^2 \right)$$

- Open issue: What do we use for f_1' and f_2'?
- It is quite common to set $f_1' = \mu_1$ and $f_2' = \mu_2$.
- The minimization function clearly depends on θ.

$$\hat{\theta} = \arg\min_{\theta} \left(\sum_{f=0}^{\theta} (f_1'(\theta) - f)^2 + \sum_{f=\theta+1}^{f_{\max}} (f_2'(\theta) - f)^2 \right)$$

Optimal Thresholding Algorithm

```
1. Initialize $\theta_{\min}$ and $E_{\min}$.
2. Initialize $\theta = 1$.
Repeat
3. Compute $f_1'(\theta) = \underset{\forall f \le \theta}{\mathrm{mean}}(f)$ and $f_2'(\theta) = \underset{\forall f > \theta}{\mathrm{mean}}(f)$.
4. Compute the total squared error, $E = \varepsilon_1 + \varepsilon_2$.
5.  If $E < E_{\min}$
6.       $E_{\min} = E$
7.       $\theta_{\min} = \theta$
8. Increment $\theta$.
Until you have tested all possible $\theta$ values.
```

Optimality via Probability Distributions

- Another criterion for evaluating whether a given threshold value θ is optimal is how closely the resulting Gaussians satisfy certain probability criteria.
- **Otsu's thresholding criterion**:

  Choose a threshold θ that maximizes the variance between the two classes.

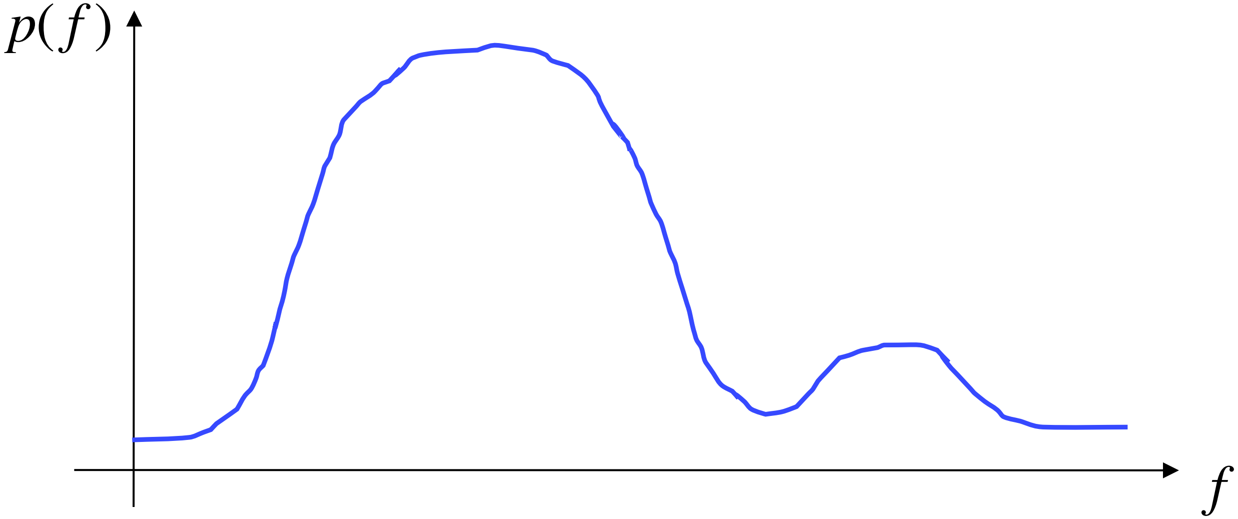
  For Gaussian mixtures this is equivalent to maximizing the distance between the 2 mean values.

$$\hat{\theta} = \arg\max_{\theta} p\big(\Omega_1(\theta)\big) p\big(\Omega_2(\theta)\big)\big(\mu_1(\theta) - \mu_2(\theta)\big)^2$$

- Note that the distance metric is multiplied by the prior class probabilities.
- Implementation as before.

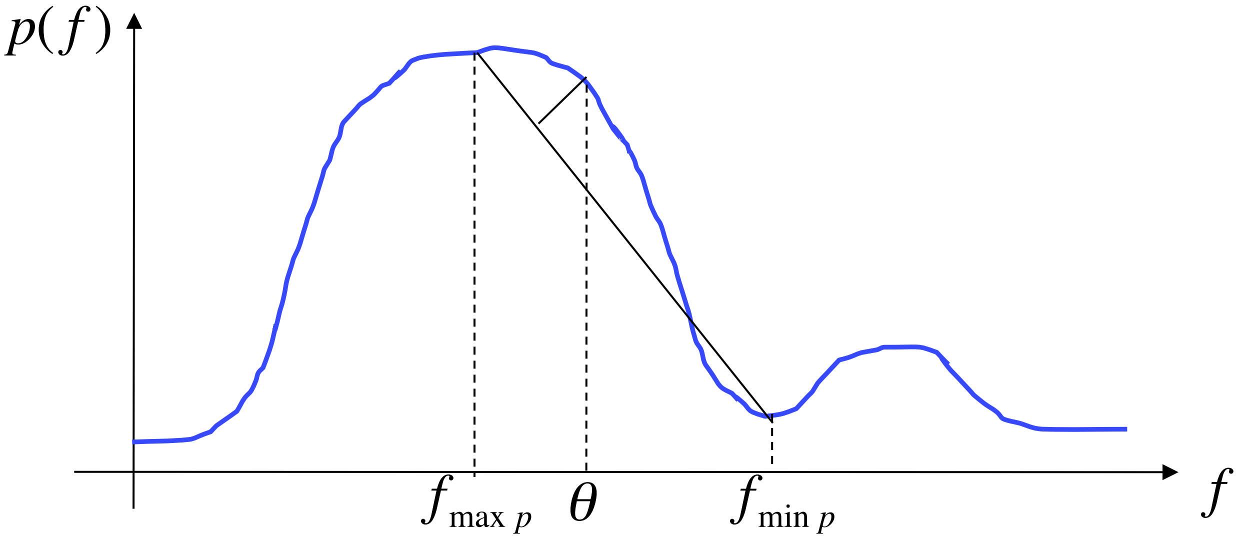
Unimodal Histogram

- What do we do when we don't have bimodal histograms and we still need to do binarization?
- Another common form of gray-value distributions is one with one big, dominant bell-curve (mode) together with a smaller one on one side.



- For all practical purposes this is considered approximately a **unimodal** histogram.

Unimodal Histogram Heuristic



- 1. Find the gray value with the highest number of occurrences, $f_{\max p}$
- 2. Look for the gray value $f_{\min p}$ which:
  - has close to zero occurrences
  - is as close as possible to the gray value with the highest number of occurrences
  - is at the long tail of the distribution.
- 3. Connect the two data points with a straight line.
- 4. Threshold θ is the intensity of that point on the histogram which has the maximum perpendicular distance to the line.

Entropy Based Binarization

- Entropy based binarization makes no assumptions about the shape of the histogram (bimodal, unimodal, etc.)
- It still uses probability information, i.e. how often a particular value occurs in the signal.
- It assumes that there are **two distinct sources of data**, e.g. background (which we will refer to as group/ class Ω_1) and foreground (which we will refer to as group/class Ω_2).
- It makes no assumptions (uses no prior information) about:
  - the range of values associated with each source
  - the frequency with which they occur
  - any separability characteristic of the sources

Brief Introduction to Entropy

- Entropy in information theory is a measure of the uncertainty associated with a random variable.
- It is defined as:

$$H = -\int p(f)\log p(f)df$$

- The logarithm is used in order to capture the additive characteristic for uncertainty.
- Low entropy means low uncertainty. It implies that there is some prior, additional information available.
- Example:
  - Fair dice has high entropy. Each value is equally likely to occur.
  - Loaded dice has low entropy. A particular value is more likely to occur.

Entropy and Thresholding

- Consider the case where we have possibly a multimodal histogram and no information about the two sources: background and foreground.
- Lacking any additional information, we want to be as objective as possible when selecting a threshold θ.
- We want to treat both sources as equally likely to occur.
- We don’t want to introduce any bias.
- In terms of entropy, this means high uncertainty, i.e. maximum entropy.

Two-Source Entropy Computation

- Let $D(\theta) = \sum_{i=0}^{\theta} p(f_i).$
- Assuming that we use θ as the threshold value that separates the two sources, then the amount of uncertainty for one source (e.g. background) is:

$$H_1(\theta) = -\sum_{i=0}^{\theta} \frac{p(f_i)}{D(\theta)} \log \frac{p(f_i)}{D(\theta)}$$

- Similarly the amount of uncertainty for the 2nd source (e.g. foreground) is:

$$H_2(\theta) = -\sum_{i=\theta+1}^{f_{\max}} \frac{p(f_i)}{1-D(\theta)} \log \frac{p(f_i)}{1-D(\theta)}$$

Two-Source Entropy Computation

- We don’t want to introduce any bias for either the background or the foreground.
- We want both of them to be equally likely to occur.
- We want to maximize the entropy over both sources.

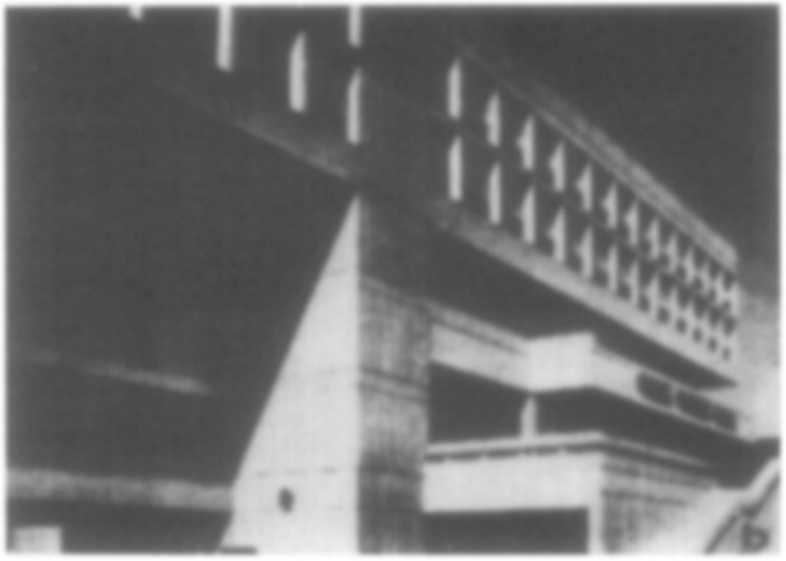
$$\hat{\theta} = \arg\max_{\theta}\left(H_1(\theta) + H_2(\theta)\right)$$

- This method can be easily extended to multi-level thresholding. There are just more classes and thetas. The search space increases, but the method is the same.

Comments on Entropy-Based Approach

- Best suited for:
  - Bimodal histograms with wide valleys with local small modes.
  - Multimodal histograms
- Fitting Gaussians may be misleading.
- There is no clear “bottom of the valley”.
  - It may be an inappropriate threshold value.
  - It can be hard to compute. We can get stuck in other local minima.

Example



Original image

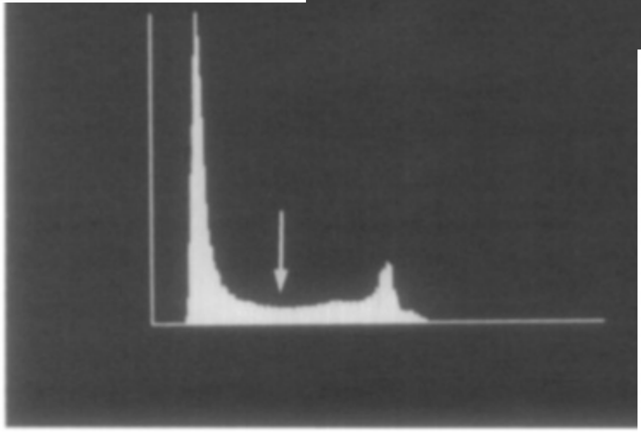
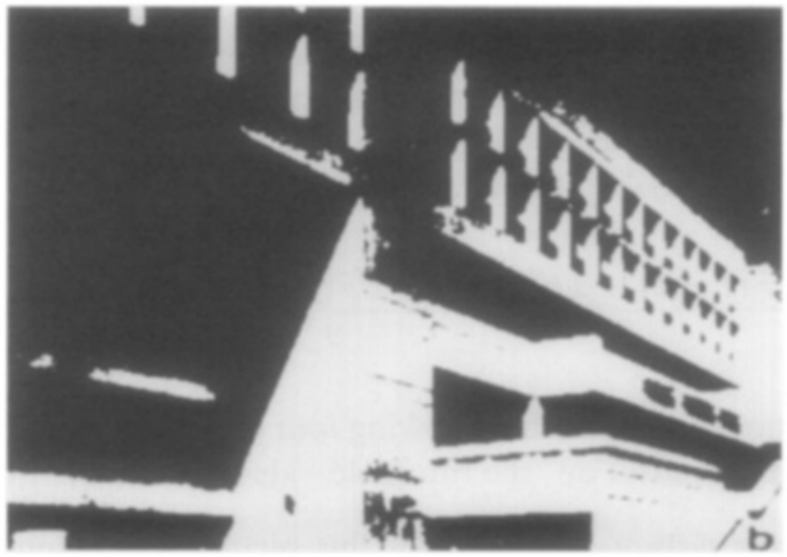


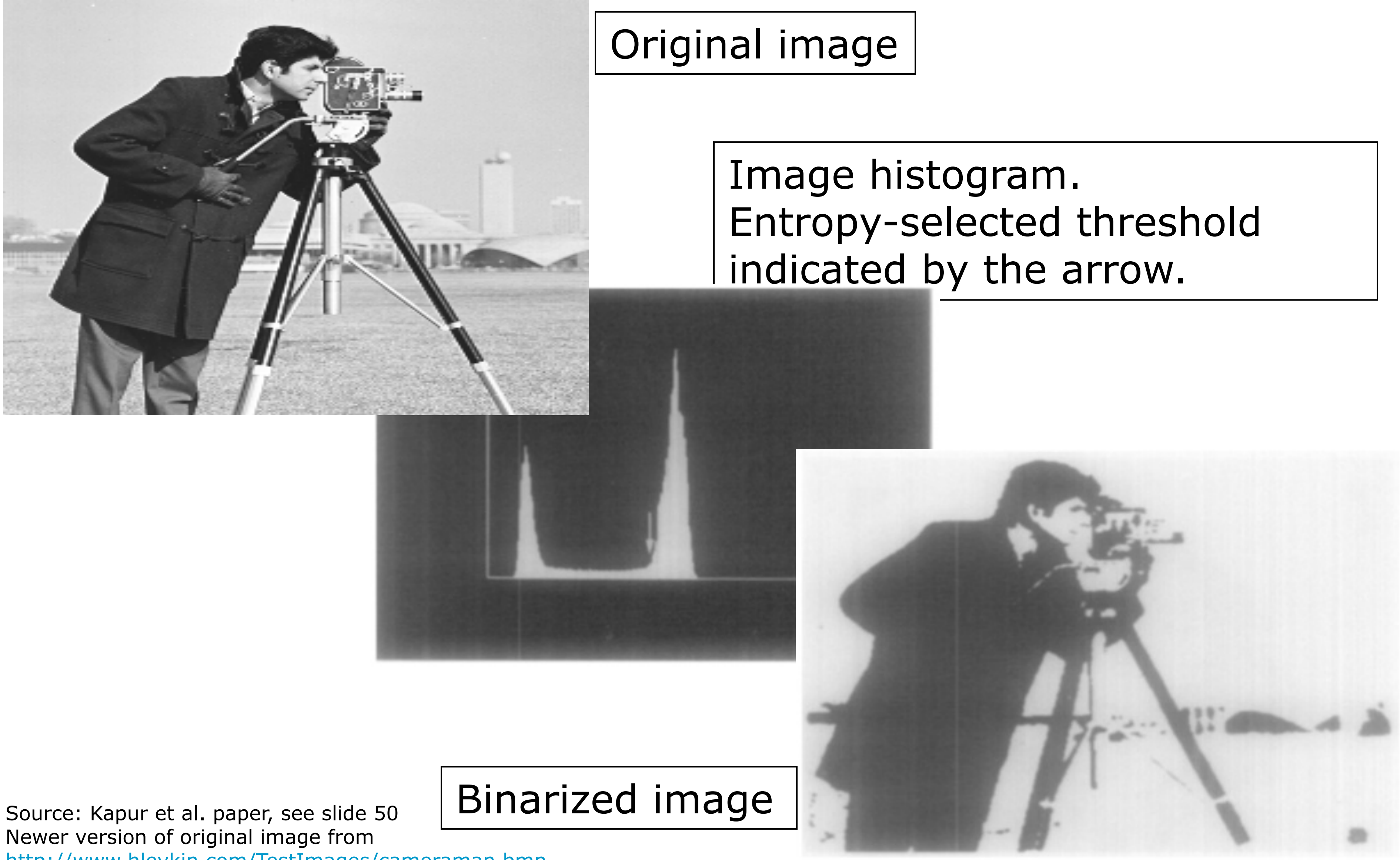
Image histogram. Entropy-selected threshold indicated by the arrow.



Binarized image

Source: Kapur et al. paper, see slide 50

Example



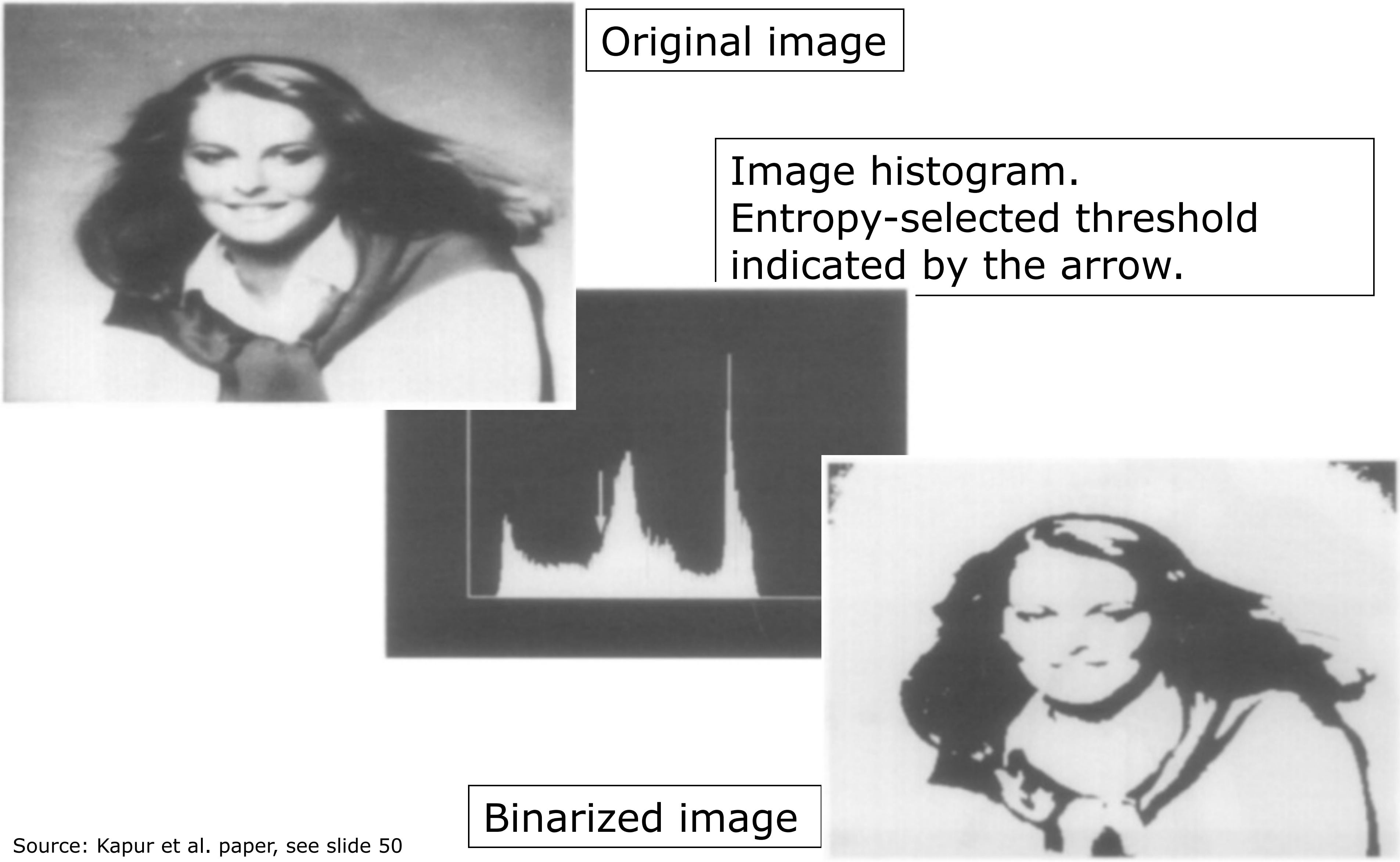
Original image

Image histogram.
Entropy-selected threshold indicated by the arrow.

Binarized image

Source: Kapur et al. paper, see slide 50
Newer version of original image from
http://www.hlevkin.com/TestImages/cameraman.bmp

Example



Original image

Image histogram.
Entropy-selected threshold
indicated by the arrow.

Binarized image

Source: Kapur et al. paper, see slide 50

Entropy Thresholding Bibliography

1. C.-I Chang, Y. Du, J. Wang, S.-M. Guo and P.D. Thouin, “Survey and Comparative analysis of Entropy and Relative Entropy Thresholding Techniques,” *IEE Proc.-Vis. Image Signal Process*., Vol. 153, No. 6, December 2006, pp. 837-850.
2. J.N. Kapur, P.K. Sahoo and A.K.C. Wong, “A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram,” *Comput. Vis. Graph. Image Process*., Vol. 29, 1985, pp. 273–285.

Automatic Thresholding Algorithm

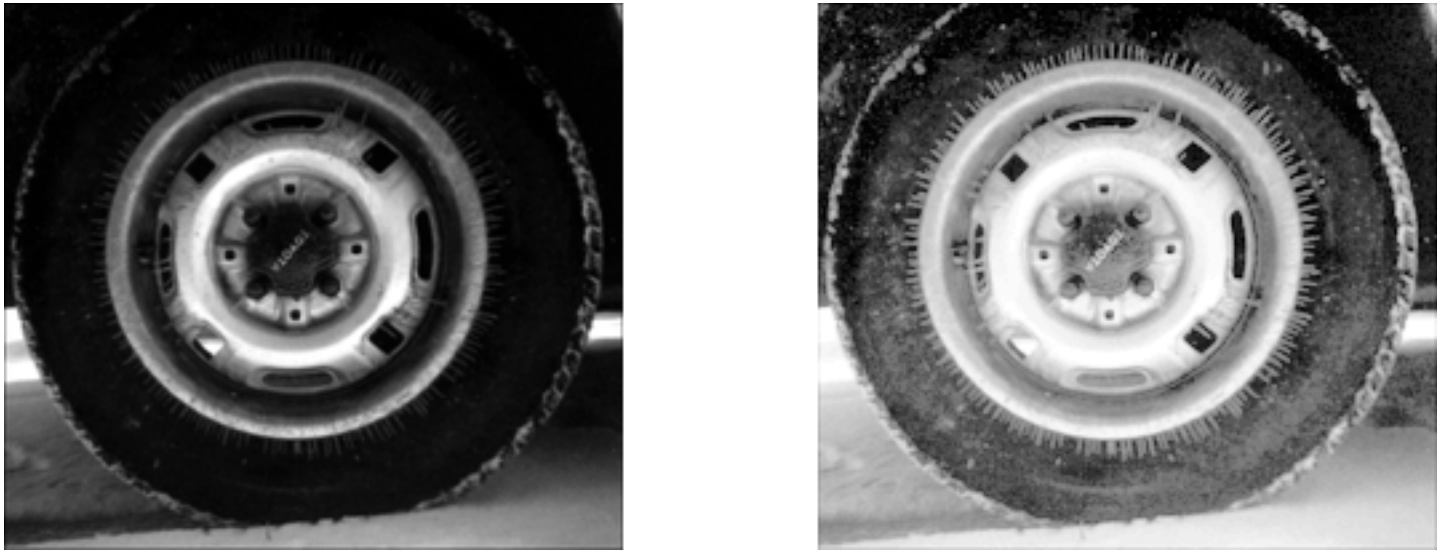
1. Initialize θ to the average gray value: $\theta = \underset{\forall f}{\mathrm{mean}}(f)$
2. Partition the data around θ and compute the mean value μ_1, μ_2 for each partition.
3. Select a new threshold value which is the mean of the two means:

$$\theta = \frac{1}{2}(\mu_1 + \mu_2)$$

4. Repeat steps 2 and 3 until the mean values μ_1 and μ_2 do not change in successive iterations.

Histogram Equalization in Matlab

- Matlab provides 2 versions of histogram equalization:
  - The new image has uniformly distributed gray values. The user specifies the number of gray values.
  - The new image has a distribution of gray values that closely matches a given image histogram.

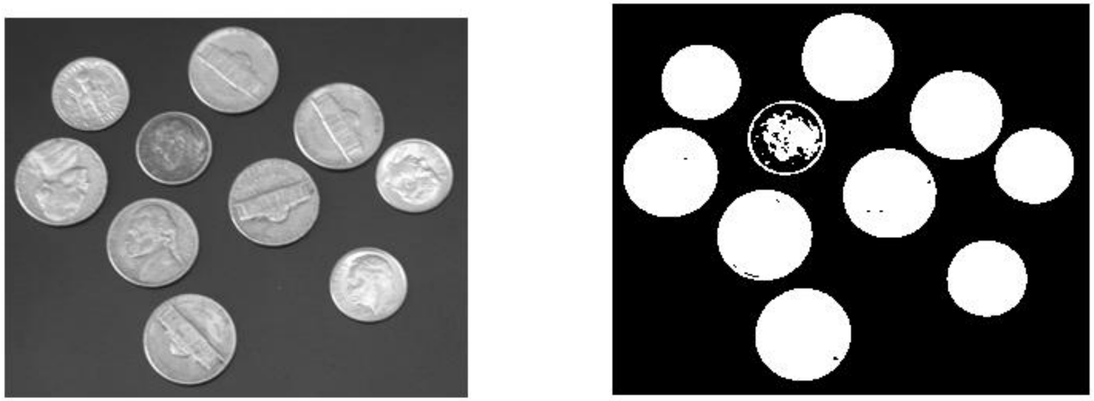


Matlab example where the output image has approximately uniformly histogram distribution

Source: http://www.mathworks.com/help/toolbox/images/ref/histeq.html

Binarization in Matlab

- In Matlab the threshold value is explicitly given as a parameter in the built-in binarization function, `im2bw()`.
- However, Matlab does provide a method for computing a threshold value for a given image.
- Their threshold value computation is based on Otsu’s method.



Matlab example of an Otsu-based threshold value computation.

Sources: http://www.mathworks.com/help/toolbox/images/ref/im2bw.html http://www.mathworks.com/…/ref/graythresh.html http://blogs.mathworks.com/steve/2009/08/31/functional-design-clunkers/