Quantization of Images and Lloyd's Algorithm

389.130 Bachelor Thesis
Institute of Communications and Radio-Frequency Engineering

Author: Martin Mayer | 0725729 | e0725729@student.tuwien.ac.at
Supervisors: Univ.Prof. Dr.techn. Markus Rupp, Dipl.-Ing. Michal Simko

September 26, 2010

Contents

Chapter 1

Introduction

1.1 Abstract

Modern digital communication systems require more sophisticated quantization techniques. The iterative Lloyd algorithm is introduced and compared to the uniform quantization using Gaussian-, Laplace- and Uniform-distributions. The place of the quantizer in source coding is illustrated, and Lloyd‘s algorithm is used on pictures to compare results, visually and numerically. The benefits of discrete transformations are highlighted to summarize the advantages of Lloyd‘s algorithm over uniform quantization. All necessary modules for calculation and simulation were built in Matlab.

1.2 Preface

Whenever data from our environment is captured, it has to be quantized in order to store and process it with digital media. The environment with its infinite level of detail is mapped to a finite number, depending on the desired accuracy of the operation. This kind of quantization is usually done by an analogue to digital converter, the digitalized sensor information is stored in a memory using ones and zeros. In information technology and digital communications, one is always searching for adaptive quantization techniques for compression that deliver optimal results depending on the particular input data. For a digital transmission, we want to lower the necessary bitrate to a minimum (lower entropy of source) without losing recognizable/too much detail of the information. Therefore, the quantization algorithm has to recognize and retain the important features of the input data and to discard the unimportant ones ("no waste of quantization levels"). Information theory has evoked some useful algorithms to accomplish this and get close to an optimum (at the expense of memory and computational power). In combination with several other techniques used in source coding, Lloyd‘s algorithm delivers what’s desired. It’s an iterative algorithm that moves the quantization levels and intervals with respect to the input data until a specified accuracy (break condition) is reached. In this bachelor thesis, Lloyd‘s algorithm is explained and compared to the uniform quantization. Therefore, the necessary modules were created and evaluated using Matlab. Since this kind of quantization is part of source coding, its basics are explained as well. At last, discrete cosine transformation is introduced to show the real benefits of Lloyd‘s algorithm over the uniform quantization.

1.3 Quantization

> Quantization is the procedure of constraining something from a continuous set of values (such as the real numbers) to a discrete set (such as the integers). [1]

In signal processing, a signal is usually digitalized using an analog to digital converter which samples the continuous input signal f(t) at multiples n×Ts of the sampling period and quantizes

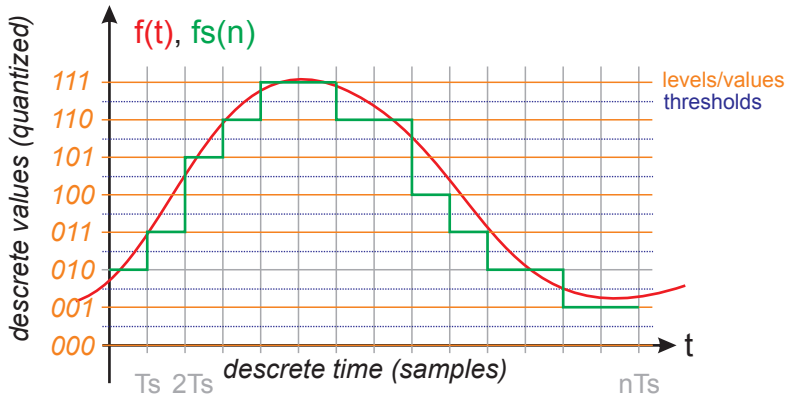

Figure 1.1: sampled and quantized signal

the amplitude with a specified resolution (higher resolution means more memory per sample). Figure 1.1 shows an examle. The vertical grid (on ordinate) are the quantization levels, the possible values of the quantized signal. Between the quantization levels are the thresholds, they mark the interval borders in which a value is mapped. Each interval has got one corresponding value. A sampled signal value is mapped to the corresponding value of the interval.

Example: Continuous Signal A continous signal is sampled. We have n=3 bits to store a sampled value. Therefore we've got M=2^n=8 quantization levels (possible values) to store the sampled data. This gives us M-1=7 thresholds.
The quantization levels and intervals in our example are equidistant, and the quantization levels are centered between two thresholds. This is the "classic" way, let's call it **uniform quantization**.

If we are quantizing a signal, we can only estimate in which area the next sampling value will be, the information comes sequential with time, sample by sample (predictive quantizers used). Now if we take a look at image processing, the quantization procedure is a little bit different. Looking at an image, we now all its values right from the beginning (matrix of values). That allows us to generate a histogram of the occurance of each value, but more on that later. Let's take a look at a basic example:

Example: Grayscale Gradient An 8 bit grayscale gradient with resolution 512×32 pixels is quantized. 8 bit means 2^8=256 possible values, 0 (black) to 255 (white). If we use a M=16 level quantization, the originally 256 values are mapped to only 16 values. The "intital set" of {0,1,2,3,...,255} (256 elements) is mapped to 16 quantization levels {8,24,40,56,...,247} (16 elements) using the uniform quantization.

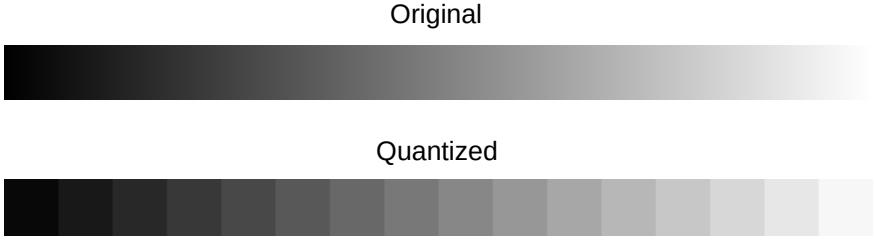

Figure 1.2: grayscale gradient, before and after 16 bit quantization

The quantization levels are calculated as follows: We use M=16 level quantization, this divides our initial set of 256 into 16 intervals, every interval corresponds to one quantization level. The 16 intervals range from [0,16), [16,32), [32,48), ..., [239,255]. Every value in interval [0,16) becomes

8, every value in interval [16,32) becomes 24 and so on. Since we use the uniform quantization, the quantization values are the centroids of each interval (here rounded to integer values) and the intervals are same length.

1.4 Lloyd's Algorithm

> In computer science and electrical engineering, Lloyd's algorithm, also known as *Voronoi iteration* or relaxation, is an algorithm for grouping data points into a given number of categories, used for *k-means clustering*.
> Lloyd's algorithm is usually used in a Euclidean space, so the distance function serves as a measure of similarity between points, and averaging of each dimension for the averaging, but this need not be the case.
> Lloyd's algorithm starts by partitioning the input points into k initial sets, either at random or using some heuristic. It then calculates the average point, or centroid, of each set via some metric (usually averaging dimensions in Euclidean space). It constructs a new partition by associating each point with the closest centroid, usually using the Euclidean distance function. Then the centroids are recalculated for the new clusters, and algorithm repeated by alternate application of these two steps until convergence, which is obtained when the points no longer switch clusters (or alternatively centroids are no longer changed). [1]

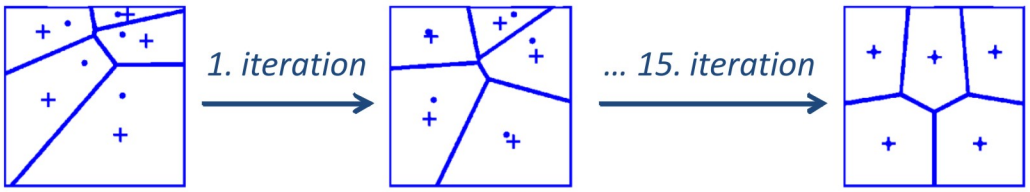

Figure 1.3: Voroni iteration

Summarized, the *important features of Lloyd's algorithm* are:

- changes partitions iterative until break condition reached or no further changes
- uses distance measure (usually Euclidean norm)
- uses only necessary but no definite conditions
- result varies with initial set and used conditions (no definite result)

To get a better understanding of image quantization, Lloyd is explained using an example image:

Lloyd on Images: Let's use the 8 bit Lena grayscale picture (figure 1.4), dimensions 256×256 pixels. This gives us a total pixel number of 65536. Each of these pixels is in the 8 bit value range from 0 (black) to 255 (white). We can count how many pixels of each value there are, creating a histogram (probabilities over pixels). This is an important procedure that gives us information about the entropy (see 1.5 Source Coding) and about the occurance probability of each value. Lloyd's algorithm will be adaptive to these characteristics. To generalize the procedure, the values can also be called symbols (each symbol represents a value). The histogram can be interpreted as a probability density function (pdf) which in our case is descrete.
If we want to quantize an image using Lloyd and an M-level quantization, we have to divide the symbols (possible values) into M sets. The resulting partition is called initial set. The main objective of Lloyd's algorithm is to minimize a distance metric within each set. Applied on images,

Figure 1.4: Lena grayscale 256x256x8bit

Lloyd minimizes the "distance" between the corresponding value of an interval to its borders (the thresholds) using a certain condition, e.g. minimize mean squared error if MSE is used as distance metric. With every iteration, the thresholds are moved as well (threshold condition) so that the partition changes from iteration to iteration, until a certain break condition is reached or there are no further changes. It's also possible to apply specific constraints with additional conditions, an important example would be the *entropy constrained scalar quantizer* which changes the partition in order to reach a predefined entropy after quantization.
Note: the corresponding value of an interval will now be called its representative level.
In this theses, we use "basic" Lloyd algorithm which just minimizes the mean squared error in each interval. How does this work mathematically?

- Mean Squared Error of interval m, using continuous pdf f_X:

$$MSE_m = \int_{t_{q,m}}^{t_{q,m+1}} (x - x_{q,m})^2 f_X dx \tag{1.1}$$

  $t_{q,m}$, $t_{q,m+1}$ ... lower and upper threshold of interval m
  x ... current value in interval m
  $x_{q,m}$ ... representative level of interval m (constant in interval)

- Mean Squared Error of interval m, using discrete pdf $p(i)$:

$$MSE_m = \sum_{i=u}^{v} (x(i) - x_{q,m})^2 p(i) \tag{1.2}$$

  i ranges over all indices (u to v) in actual interval m
  $x(u) = t_{q,m}$, $x(v) = t_{q,m+1}$ ... lower and upper threshold of interval m
  $x(i)$ ... current value i in interval m
  $x_{q,m}$ ... representative level of interval m (constant in interval)
  $p(i) = P(x(i))$ ... probability of symbol $x(i)$

- We use discrete version since we are dealing with images.

- With m ... number of current interval ($m = 1...M$),
  i ... index of symbol in interval, i goes over all symbols in current interval,
  $x_{q,m}$ ... representative level of interval m, $x(i)$ ... symbol i in interval m.

  Note: There are $m = 1...M$ intervals. Each interval contains symbols (values), all intervals together are building a partition, let's call it the initial set.

To minimize the MSE in each interval, we have to build the derivative of MSE_m with respect to $x_{q,m}$ and set it to 0:

$$\frac{dMSE_m}{dx_{q,m}} = 0 \tag{1.3}$$

After some rearrangements, this delivers a formula for $x_{q,m}$:

- Minimize MSE_m condition, using continuous pdf:

$$x_{q,m} = \frac{\int_{t_{q,m}}^{t_{q,m+1}} x f_X dx}{\int_{t_{q,m}}^{t_{q,m+1}} f_X dx} \tag{1.4}$$

- Minimize MSE_m condition, using discrete pdf:

$$x_{q,m} = \frac{\sum_{i=u}^{v} x(i)p(i)}{\sum_{i=u}^{v} p(i)} \tag{1.5}$$

We use the equation for discrete distributions since we are dealing with images. Equation 1.5 is the necessary condition to minimize the MSE in each interval. To get Lloyd working, another condition, the centroid or threshold condition (equation 1.6) is necessary:

$$t_{q,m} = \frac{x_{q,m+1} - x_{q,m}}{2} \tag{1.6}$$

This condition basically centers each threshold between two representative levels.

> Note: within one iteration step, the new thresholds and the new representative levels *of all intervals* are computed parallely.

Now let´s get this knowledge into the right order and summarize it up: Figures 1.5 to 1.8 are showing the pdf of the Lena image (figure 1.4) at different stages of Lloyd‘s algorithm. Note that the symbols (values on abscissa) are discrete {0,1,2,...,255}, but were **interpolated for graphical representation** (it now looks like a continuous density function).

Lloyd‘s algorithm:

1. Get *pdf of image* (symbols a and corresponding probabilities p)
2. "Guess" (random or heuristic) *initial set* of M representative levels $x_{q,m}$ $(m = 1...M)$
   - pdf is now divided into M intervals
3. Apply necessary centroid/threshold condition (equation 1.6)
   - this calculates (new) thresholds for each interval (centroid of $x_{q,m}$ and $x_{q,m+1}$)
   - $x_{q,m}$ is representative level of interval m
4. Apply necessary minimize MSE_m condition (equation 1.5)
   - $x_{q,m}$ of each interval $m = 1...M$ is calculated to minimize MSE_m of intervals
   - MSE_m of each interval is calculated (equation 1.2)
5. Iteration process:
   - Steps 3. and 4. are repeated until no further decrease in total $MSE = \sum_m MSE_m$ (if all MSE_m are minimized, their sum is also minimized)
6. Apply quantization on image!

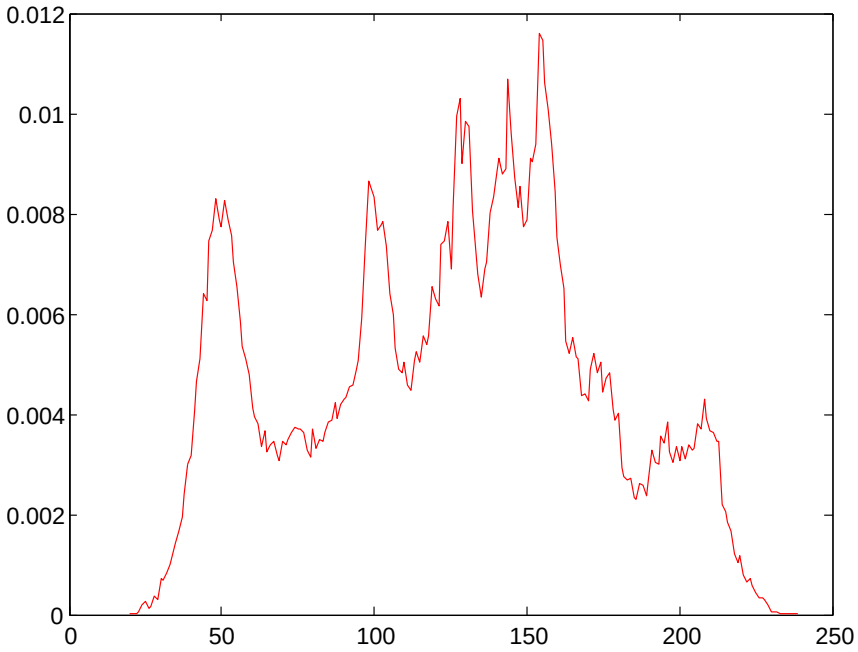

Figure 1.5: pdf of Lena (interpolated)

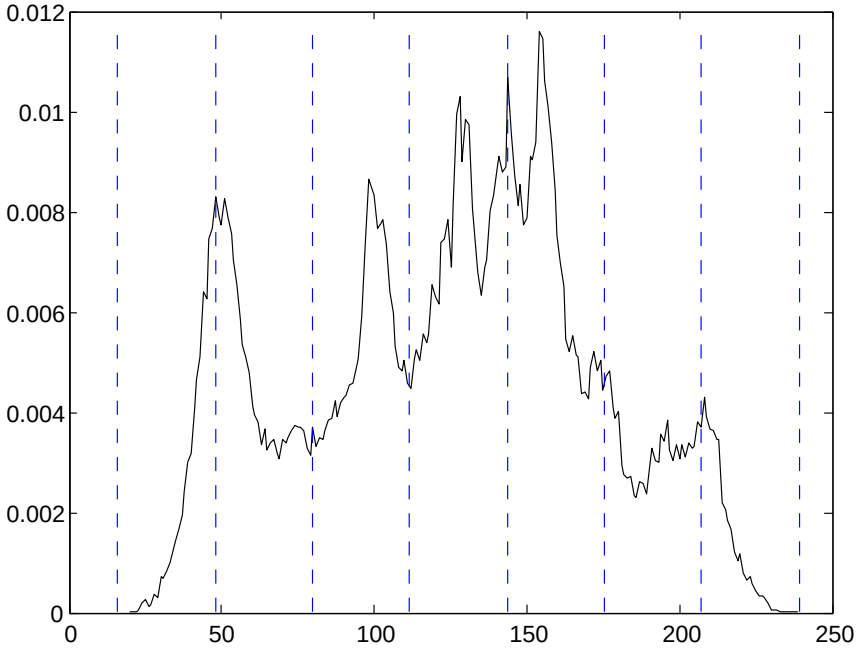

Figure 1.6: initial representative levels (blue)

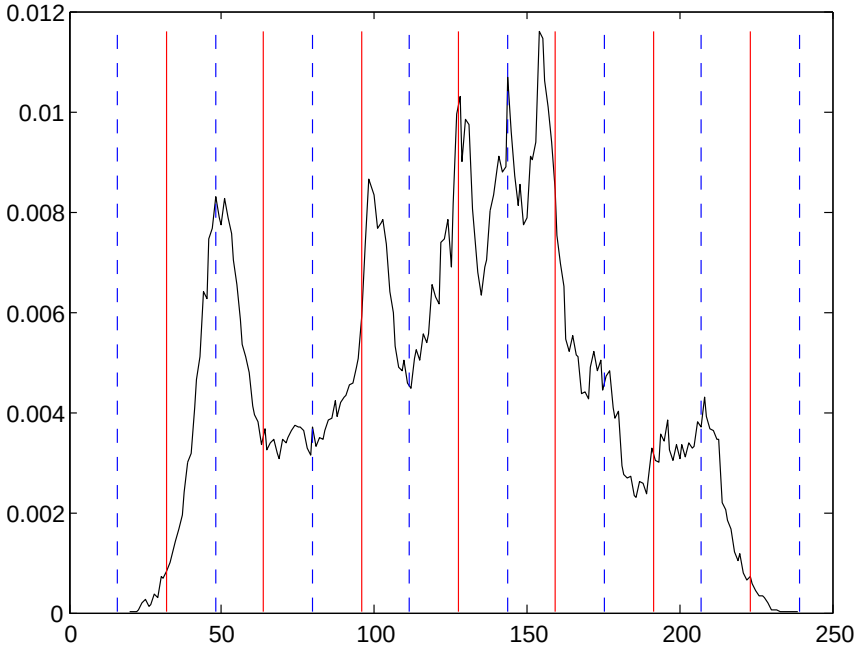

Figure 1.7: initial thresholds (red)

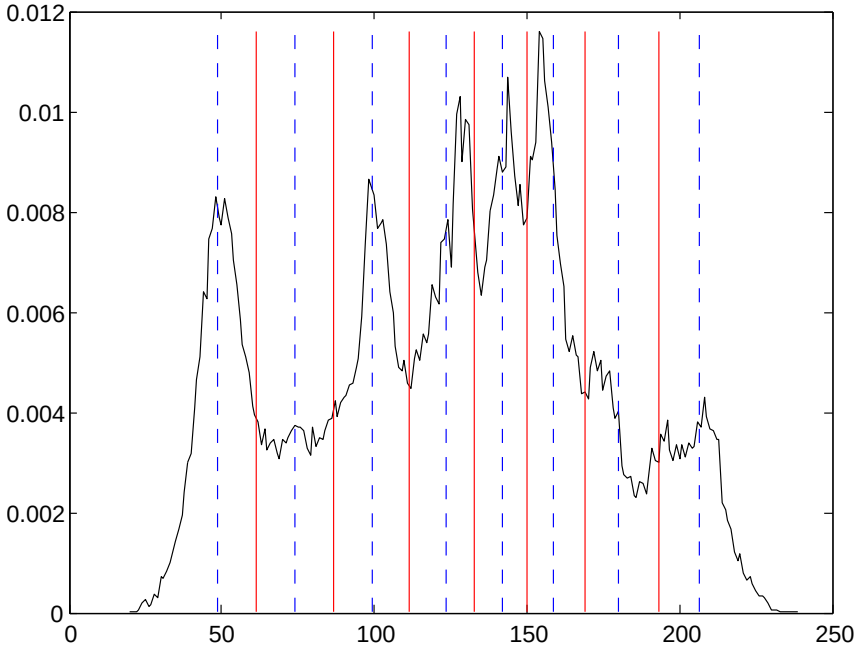

Figure 1.8: representative levels and thresholds after Lloyd (25 iterations)

Lena Quantization Results Now the procedure was applied on the Lena image (figure 1.4) using M=8 level quantization. As initial set, the picture was quantized with uniform quantization (equidistand thresholds and representative levels over the whole 8 bit area).

- Initial set of representative levels $x_{q,m}$ (rounded to integer):
  {16, 48, 80, 112, 143, 175, 207, 239}
- Initial thresholds $t_{q,m}$ after threshold condition (rounded to integer):
  {32, 64, 96, 128, 159, 191, 223}
- Initial intervals resulting out of thresholds:
  {[0,32), [32,64), [64,96), [96,128), [128,159), [159,191), [191,223), [223,255]}

After applying Lloyd's Algorithm on Lena (figure 1.4), the following results were obtained after 25 iterations:

- Final representative levels $x_{q,m}$ (rounded to integer):
  {49, 74, 100, 123, 142, 158, 180, 206}
- Final thresholds $t_{q,m}$ (rounded to integer):
  {61, 87, 112, 133, 150, 169, 193}
- Final intervals resulting out of thresholds:
  {[0,61), [61,87), [87,112), [112,133), [133,150), [150,169), [169,193), [193,255]}

Figure 1.9 shows the graphic representation of the representative levels $x_{q,m}$, figure 1.10 shows the resulting pictures after quantization (**the uniform quantization is the starting point/initial set for Lloyd**).
The obvious result: The values with high occurance probability have small intervals (fine quantization), the values with low occurance probability have big intervals (coarse quantization). Lloyd moves the representative levels to the areas of high occurance in order to quantize only the "important" features of the image. The uniform quantizer will also quantize areas with no occurances at all, "wasting" quantization intervals which throws away information. This is visible in figure 1.10, the Lloyd quantized image has more detail (finer gradients).

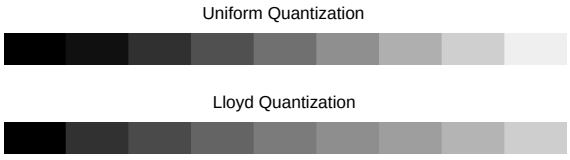

Figure 1.9: representative levels $x_{q,m}$ before (uniformly distributed) and after Lloyd

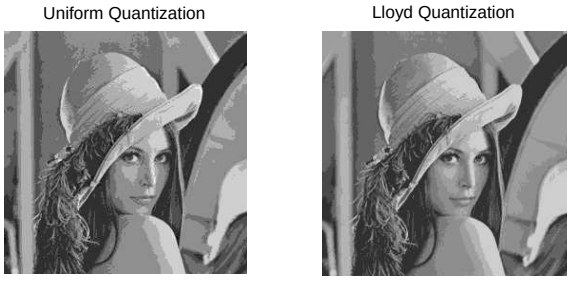

Figure 1.10: Lena uniform quantized (initial set) and Lloyd quantized

1.5 Source Coding

To understand source coding, we introduce a basic model (figure 1.11) for data transmission in a digital communication system. The information source delivers us the digitalized raw data of the information we want to transmit. The aim of source encoding now is to reduce the necessary bitrate for transmission by removing redundancy and irrelevancy from information. Channel encoding adds redundancy to make sure that the receiver can detect and ideally correct the errors occured during transmission (channel adds noise). We concentrate on source encoding and divide it into three parts:

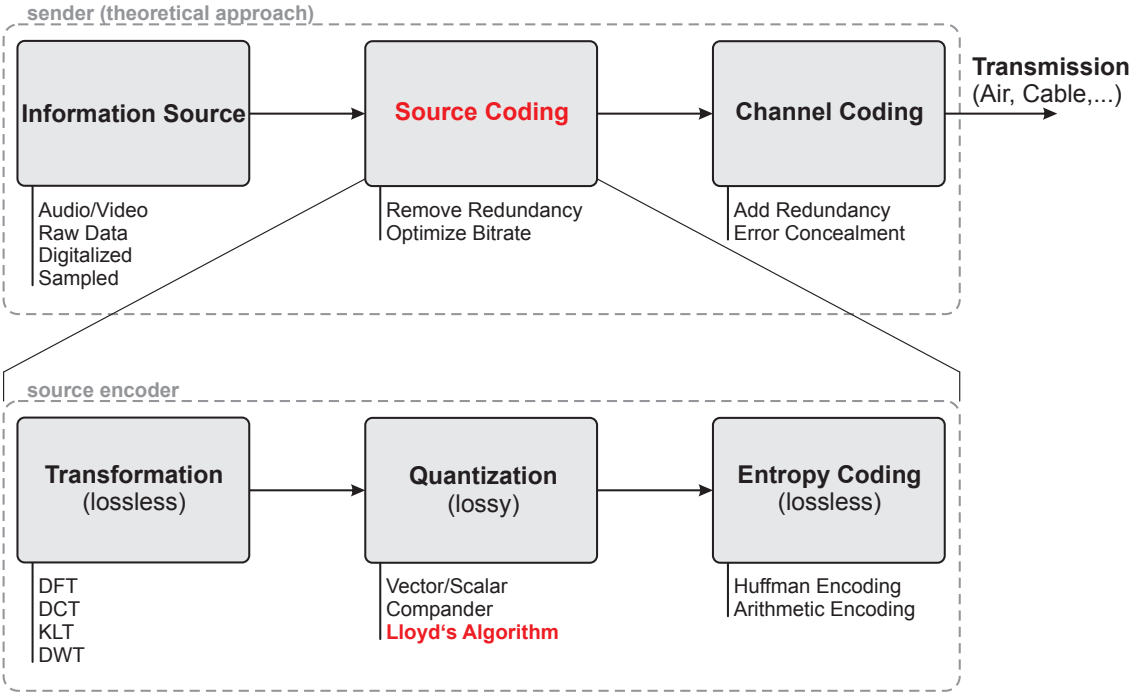

Figure 1.11: basic model of sender

- *Discrete Transformation* decorrelates the image data and does energy compression, this process is (usually) lossless since the transformations are orthonormal. This first step can be very effective. Important transformations are the *discrete cosine transformation* (DCT) and the *discrete wavelet transformation* (DWT).
- *Quantization* is a lossy compression technique. Some parts of the information can be discarded without visible difference but with effective reduction of the entropy (necessary bitrate). There are some good techniques, Lloyd's algorithm is one since it's adaptive to the important areas of the information.
- *Entropy encoding* is a lossless technique which lowers the necessary bitrate ideally to the entropy of the data. The basic idea is to use long codewords for unimportant symbols (symbols with low probability, rarely sent) and short codewords for important ones (symbols with high probability, often sent). The Morse code works on the same principle.

Important definitions in Information Theory:

- *Information of a symbol:*

$$I(a_i) = log(\frac{1}{p_i}) = -log(p_i) \tag{1.7}$$

  $p_i = P(a_i)$ ... probability of symbol a_i
  The information of symbol a_i is big if it's probability p_i is small and vice versa. If a symbol is sent all the time ($p_i = 1$), it's information is $-log(1) = 0$ because there is no new information. If a symbol is never sent ($p_i = 0$), it's information would theoretically be $-log(0) = \infty$.

- *Entropy of binary source:*

$$H(S) = \sum_i p_i I(a_i) = -\sum_i p_i log_2(p_i) \tag{1.8}$$

  Because we are dealing with a binary source, we use binary logarithm log_2, the result's unit will be $[H(S)] = \frac{bit}{symbol}$. Entropy of a binary source can be interpreted as the minimal average number of bits per symbol, a lower bound for the necessary bitrate. It is a measure for the minimum average codeword length of the transmission.
  Example: Binary source with N=4 symbols.

  1. Symbols equally (uniform) distributed: $p_i = \frac{1}{N} = \frac{1}{4} = 0.25$, $I(a_i) = log_2(4) = 2bit$, meaning that one symbol needs at least a 2 bit codeword ($2^2 = 4$).
     $H(S) = \sum_{i=1}^{4} \frac{1}{4} \cdot 2 = 2\frac{bit}{symbol}$, so the minimal average word length is 2 (trivial since every word is 2 bit long).
  2. Symbols not equally distributed: $p_1 = \frac{1}{8}, p_2 = \frac{1}{8}, p_3 = \frac{1}{4}, p_4 = \frac{1}{2}$. The resulting informations are: $I(a_1) = 3bit, I(a_2) = 3bit, I(a_3) = 2bit, I(a_4) = 1bit$.
     $H(S) = \frac{1}{8} \cdot 3 + \frac{1}{8} \cdot 3 + \frac{1}{4} \cdot 2 + \frac{1}{2} \cdot 1 = \frac{7}{4} = 1.75\frac{bit}{symbol}$. This means that with the right entropy encoding, the necessary bitrate can be lowered to $1.75\frac{bit}{symbol}$ instead of $2\frac{bit}{symbol}$!

Quantization lowers the entropy since it takes away information (irreversible, lossy compression)! Using good entropy encoding techniques (depending on input data, e.g. Huffman, arithmetic, ...), the necessary bitrate can be lowered down to the entropy. The influence of discrete transformations will be discussed in chapter *5. Benefits of Discrete Transformations.*

Chapter 2

Quantization of Distributions

In this chapter, common probability distributions are quantized. The reason is that picture symbols are usually somehow Laplace distributed (especially after discrete transformations, see chapter 5). The results are compared using diagrams and an error measures. The numerical results of Lloyd are also compared to their asymptotical solutions ("On Asymptotic Solutions of the Lloyd-Max Scalar Quantization" [2]).
As error measure we use the mean of the mean squared errors of all intervals (equation 1.2).

$$MSE_{mean} = \frac{1}{M} \sum_{m=1}^{M} MSE_m \tag{2.1}$$

As an additional error measure, the "normalized sum of quadratic deviations" is introduced:

$$\epsilon = \sum_{i=1}^{n} \frac{(x_i - \hat{x_i})^2}{n} \tag{2.2}$$

with x_i ... value of pixel i, $\hat{x_i}$ ... quantized value of x_i, n ... number of pixels.

2.1 Gaussian Distribution

The Gaussian or normal distribution is an important and common distribution used in a lot of areas. It has a typical bell shaped graph.

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/2\sigma^2} \tag{2.3}$$

We use the very basic *standard normal distribution* with $\sigma^2 = \mu = 0$ (variance and mean). The values ("symbols") for our numerical calculations in Matlab are ranged from $x = -6...6$ with an increment of 0.0001 (=120001 values).

M = 16 Quantization Intervals Now the standard normal distribution (equation 2.3) is quantized using M=16 quantization intervals. To visualize the process, figure 2.1 shows the MSE of every interval (black) and the mean MSE (red) which is reduced by every iteration. All iterations are printed in one diagramm. At the borders, the interval-size (and therefore MSE_m) is increasing with every iteration because the probability is low (coarse quantization). In the center, interval-size (and therefore MSE_m) is decreasing with every iteration because probability is high (fine quantization). The light green curve indicates the start set (interpolated). MSE values (ordinate) are logarithmic.
Figure 2.2 shows MSE_{mean} (equation 2.1) over the iterations. At the beginning, Lloyd works fast in shifting intervals and borders, but at the end, the decrement of MSE_{mean} is pretty slow until no further changes are noticeable (calculation precission of computer).

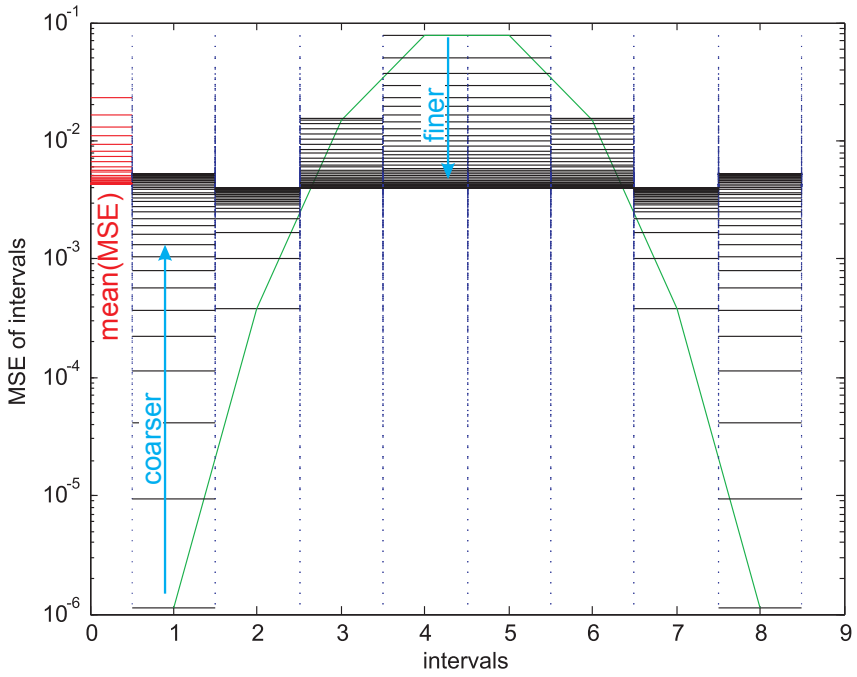

Figure 2.1: 8 level Gaussian, $log(MSE)$ over intervals, all iterations printed

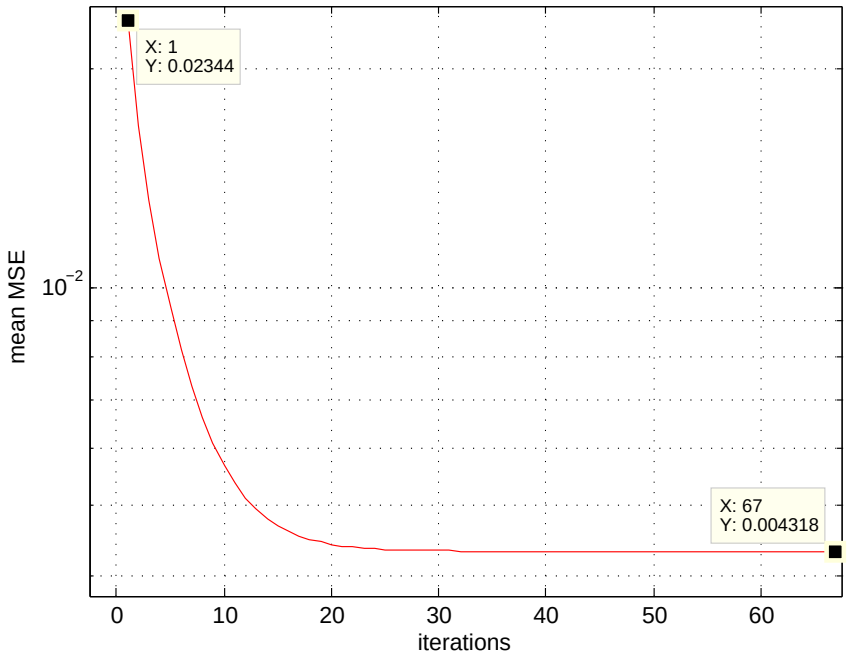

Figure 2.2: 8 level Gaussian, $log(MSE_{mean})$ over iterations

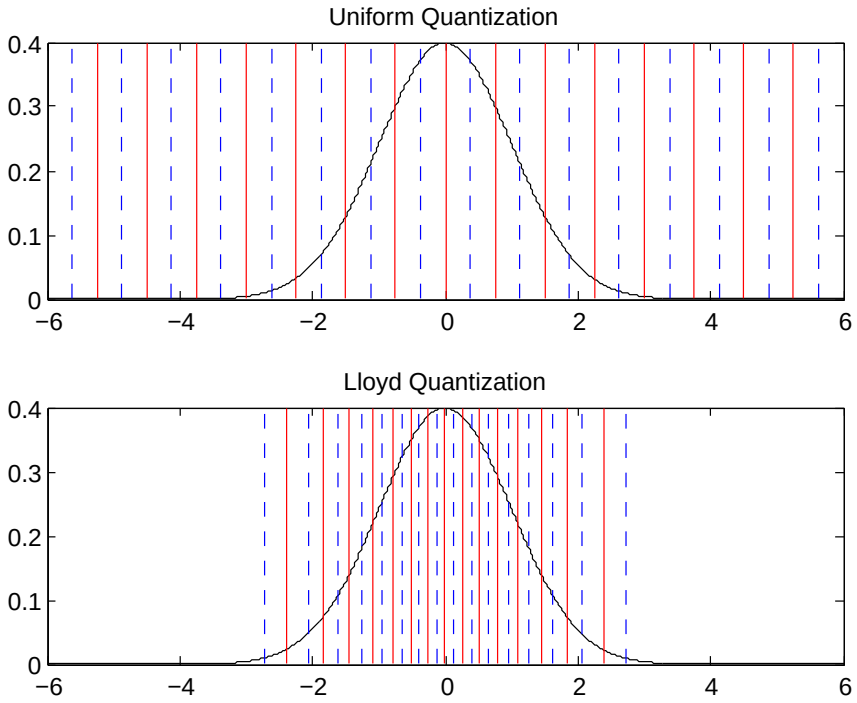

Figure 2.3: 16 level Gaussian ($x_{q,m}$...blue, $t_{q,m}$...red)

Resulting Error:

- MSE_{mean}=0.0029296, $\epsilon_{uniform}$=0.1016 (initial)
- MSE_{mean}=0.0005938, ϵ_{Lloyd}= 0.0114 (after 203 iterations)

M = 8 Quantization Intervals

Resulting Error:

- MSE_{mean}=0.0266145, $\epsilon_{uniform}$=0.4610 (initial)
- MSE_{mean}=0.0043184, ϵ_{Lloyd}= 0.0428 (after 67 iterations)

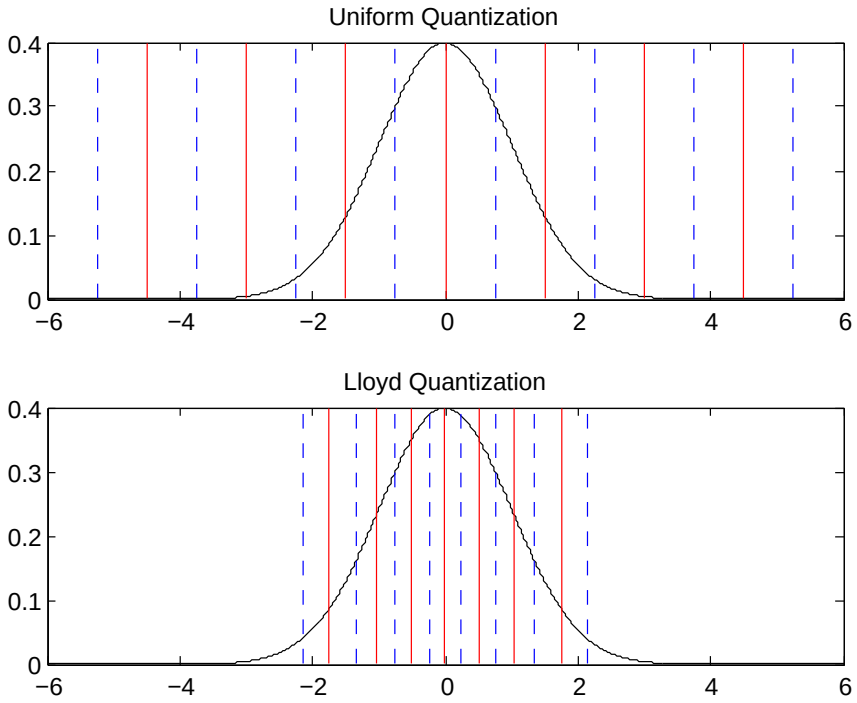

Figure 2.4: 8 level Gaussian ($x_{q,m}$...blue, $t_{q,m}$...red)

2.2 Laplace Distribution

The Laplace distribution has a sharp peak. We use the very basic density function:

$$p(x) = \frac{1}{\sqrt{2}} e^{-|x|\sqrt{2}} \tag{2.4}$$

M = 16 Quantization Intervals

Resulting Error:

- MSE_{mean}=0.0030366, $\epsilon_{uniform}$=0.1076 (initial)
- MSE_{mean}=0.0008786, ϵ_{Lloyd}= 0.0105 (after 197 iterations)

M = 8 Quantization Intervals

Resulting Error:

- MSE_{mean}=0.0266145, $\epsilon_{uniform}$=0.4670 (initial)
- MSE_{mean}=0.0064534, ϵ_{Lloyd}= 0.0409 (after 68 iterations)

Conclusion: The obvious conclusion is that the MSE is smaller after Lloyd. The second error measure (ϵ) tells us that the uniformly quantized picture differs a lot more to the original picture than the Lloyd quantized one, in other words: *Entropy after Lloyd quantization is higher than after uniform quantization!* Lloyd retains more information than other quantizers because it's adaptive.

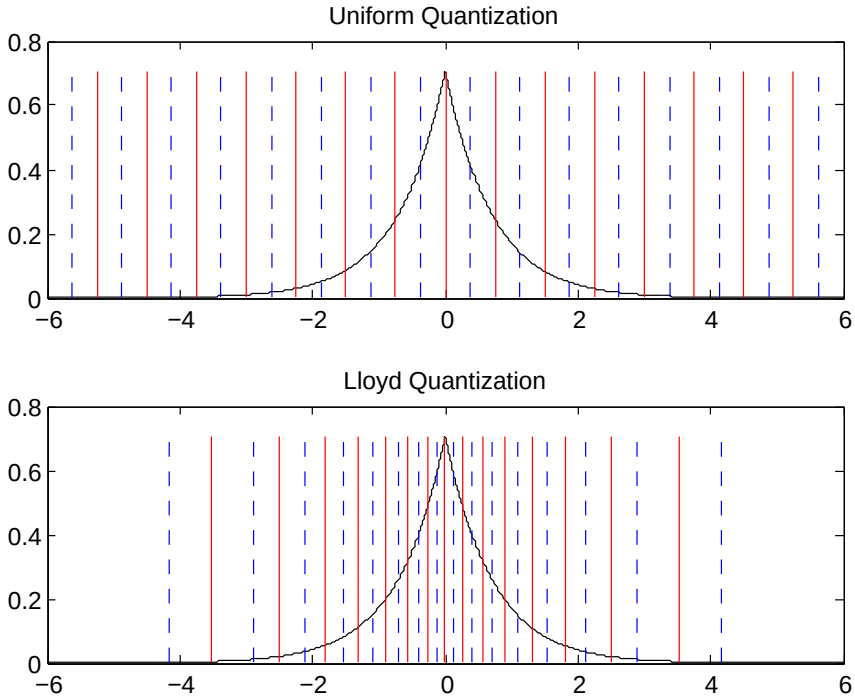

Figure 2.5: 16 level Laplace ($x_{q,m}$...blue, $t_{q,m}$...red)

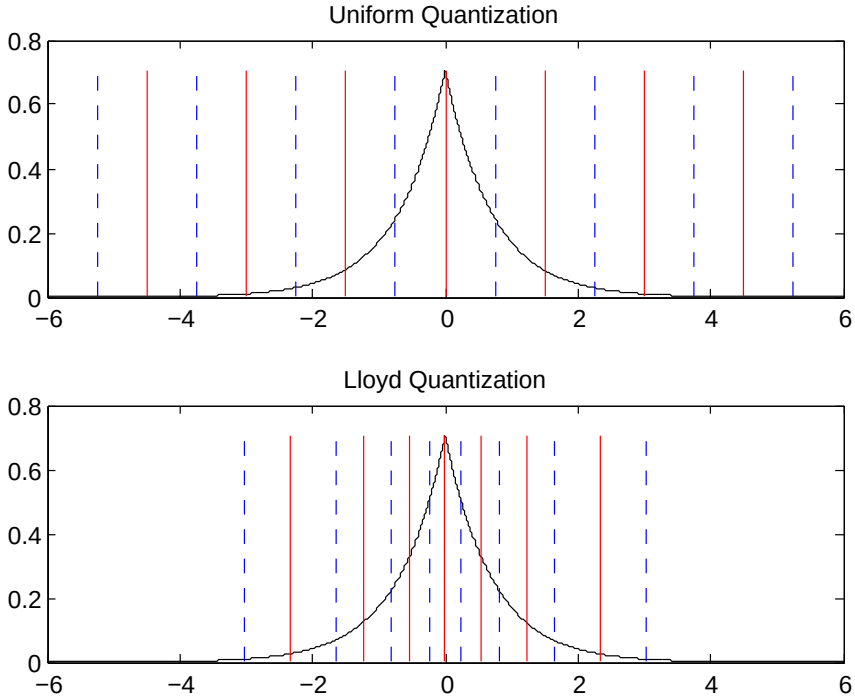

Figure 2.6: 8 level Laplace ($x_{q,m}$...blue, $t_{q,m}$...red)

2.3 Numerical Comparison with Asymptotic Solution of Lloyd

This comparsion should verify that the implemented Lloyd algorithm works like it is supposed to. The values to compare are from the IEEE paper "On Asymptotic Solutions of the Lloyd-Max Scalar Quantization" [2], where an asymptotical solution of the algorithm is computed using Gaussian-/Laplace-distribution and M=32 level quantizer. The represented solution is an ideal one. Lloyd's algorithm has no definite solution and varies with the initial set it starts with. Through heuristic experiments (see chapter 4), uniformly distributed representative levels and thresholds seem to be the best initial set to start with, especially if we use symmetric distributions like Gaussian and Laplace.
For the comparison-computation, the uniformly distributed initial set is used. The values range from $x = -6...6$ with an increment of 0.00001 (=1200001 values and long computation time). Ideally, the values would range from -∞ to +∞ with infinite small increment. The deviations from the optimum solution are resulting from...

- ...finite number of values x
- ...x ranged from -6 to 6 instead of -∞ to +∞
- ...(maybe) not the perfect initial set

Results: Since the optimal solution is symmetric around 0, only the positive half of quantization values $x_{q,m}$ and thresholds $t_{q,m}$ is listed in table 2.1.
Looking at the optimum solution, there is one threshold exactly at 0 and the rest is symmetric around it. However in our own solution, the center is not perfect although the intital set was symmetric around 0. But the results are somehow close to the ideal ones, close enough for applying Lloyd on images in order to highlight the benefits over the uniform quantization.
Figure 2.7 shows the graphical representation of the own calculation. Initital set of Lloyd is the uniform quantization with equidistant intervals symmetric around 0. After 885 iterations, Lloyd´s algorithm is finished (no further change).

m	$x_{q,m}$ (init)	$x_{q,m}$ (fin)	$x_{q,m}$ (ideal)	$t_{q,m}$ (init)	$t_{q,m}$ (fin)	$t_{q,m}$ (ideal)
0	-	-	-	0,00000	-0,0000049927	0,0000000000
1	0,18750	0,0659291004	0,0658896598	3,75000	0,1320512564	0,1319707447
2	0,56250	0,1981734123	0,1980518297	7,50000	0,2648719729	0,2647150677
3	0,93750	0,3315705334	0,3313783058	1,12500	0,3992606811	0,3990389144
4	1,31250	0,4669508287	0,4666995230	1,50000	0,5360913756	0,5358165735
5	1,68750	0,6052319224	0,6049336240	1,87500	0,6763519410	0,6760346638
6	2,06250	0,7474719595	0,7471357037	2,25000	0,8212041214	0,8208504105
7	2,43750	0,8949362834	0,8945651174	2,62500	0,9720619388	0,9716742187
8	2,81250	1,0491875942	1,0487833199	3,00000	1,1307110577	1,1302938503
9	3,18750	1,2122345211	1,2118043806	3,37500	1,2995100014	1,2990723601
10	3,56250	1,3867854816	1,3863403396	3,75000	1,4817322194	1,4812842091
11	3,93750	1,5766789572	1,5762280786	4,12500	1,6821822693	1,6817306482
12	4,31250	1,7876855814	1,7872332177	4,50000	1,9084301047	1,9079808085
13	4,68750	2,0291746281	2,0287283994	4,87500	2,1736708825	2,1732339018
14	5,06250	2,3181671369	2,3177394042	5,25000	2,5048408304	2,5044294908
15	5,43750	2,6915145239	2,6911195774	5,62500	2,9762938546	2,9759260354
16	5,81250	3,2610731853	3,2607324934	-	-	-

Table 2.1: own vs. optimal 32-level quantizer for gaussian distribution

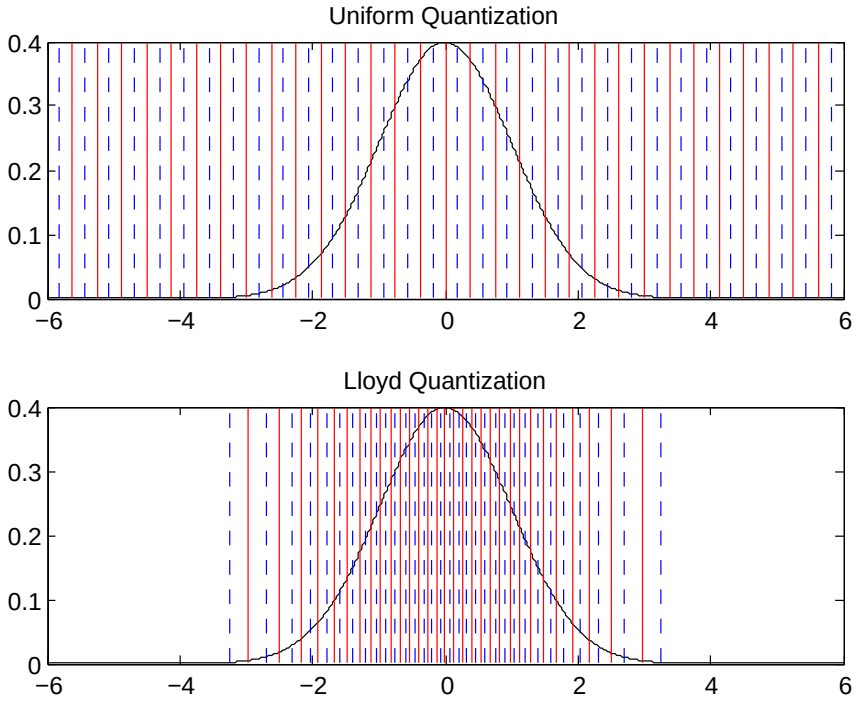

Figure 2.7: 32 level Gaussian ($x_{q,m}$...blue, $t_{q,m}$...red)

Chapter 3

Quantization of Images

The basics of image quantization are explained in section 1.4.

3.1 Lena and various Quantization Levels

Now the Lena image (figure 1.4) is quantized with different quantization levels and the results are compared. As initial set the uniform quantization is used. Entropy of original picture:

- $H_{original} = 7.4475 \frac{bit}{pixel}$

M=16 level Quantization

- Lloyd finished after 16 iterations
- $H_{uniform} = 3.5057 \frac{bit}{pixel}$
- $H_{Lloyd} = 3.8202 \frac{bit}{pixel}$
- $MSE_{mean,uniform} = 1.0171$
- $MSE_{mean,Lloyd} = 0.8570$ (16% improvement)

M=8 level Quantization

- Lloyd finished after 22 iterations
- $H_{uniform} = 2.5271 \frac{bit}{pixel}$
- $H_{Lloyd} = 2.9518 \frac{bit}{pixel}$
- $MSE_{mean,uniform} = 8.6973$
- $MSE_{mean,Lloyd} = 5.1879$ (40% improvement)

M=4 level Quantization

- Lloyd finished after 7 iterations
- $H_{uniform} = 1.7944 \frac{bit}{pixel}$
- $H_{Lloyd} = 1.9434 \frac{bit}{pixel}$
- $MSE_{mean,uniform} = 52.4251$
- $MSE_{mean,Lloyd} = 41.0420$ (22% improvement)

Figure 3.1: M=16 level quantization (Lena image)

Figure 3.2: M=8 level quantization (Lena image)

Figure 3.3: M=4 level quantization (Lena image)

Conclusion: A decrement of quantization levels M (coarser quantization) lowers the entropy (lossy compression, information is lost) and increases the error. If no other procedures are used, one has to decide which quantization is good enough for him. This depends on the application, either high quality or high compression. The number of iterations show us that a result is reached pretty fast using Lena example picture, sometimes Lloyd is "stuck" pretty early and can't decrease MSE anymore.

3.2 Lena vs. Linda

Now two pictures with obviously different densities are compared. Density of Lena (figure 3.6) has several peaks and the values range from 20 to 239, Lloyd can adapt to the peaks.
Density of Linda (figure 3.7) has one significant peak and two small side-peaks in the lower value area, from 100 to 250 it is approximately uniformly distributed. Lloyd goes for the significant peak and leaves the rest pretty untouched (shrinks intervals because the borders are "unimportant").

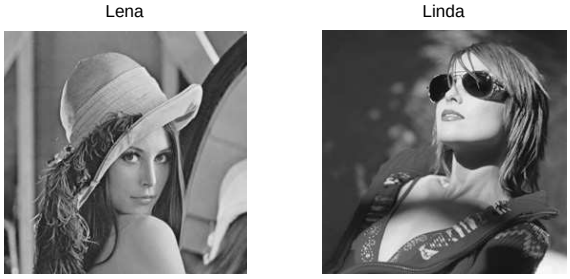

Figure 3.4: Lena and Linda, 256x256 pixels, 8 bit grayscale pictures

Figure 3.5: M=8 level quantization (Linda image)

M=8 level Quantization of Linda

- Lloyd finished after 23 iterations
- $H_{original} = 7.2125 \frac{bit}{pixel}$
- $H_{uniform} = 2.4716 \frac{bit}{pixel}$
- $H_{Lloyd} = 2.6334 \frac{bit}{pixel}$
- $MSE_{mean,uniform} = 8.2231$
- $MSE_{mean,Lloyd} = 6.0060$ (27% improvement)

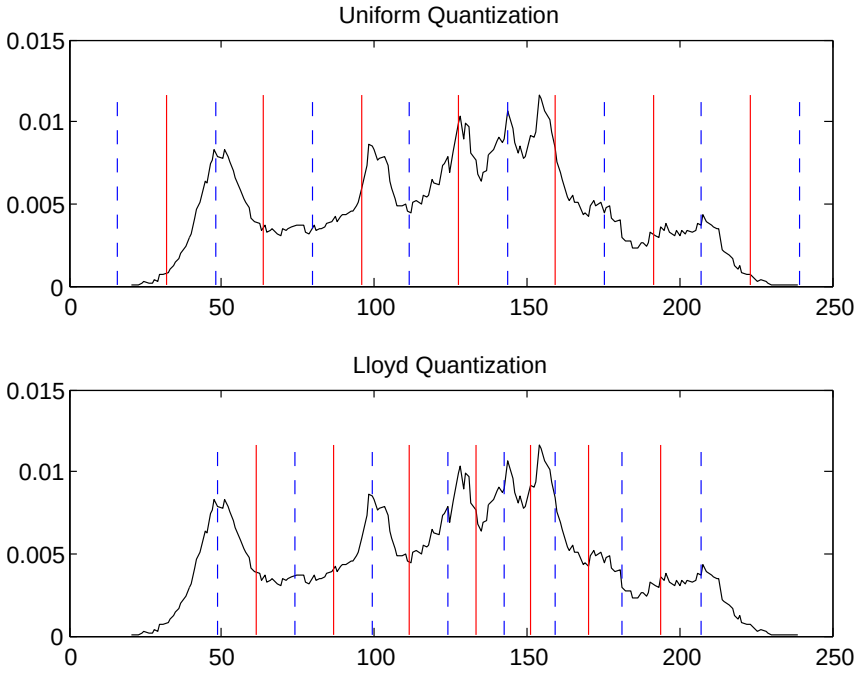

Figure 3.6: density of Lena, M=8 level quantization (22 iterations)

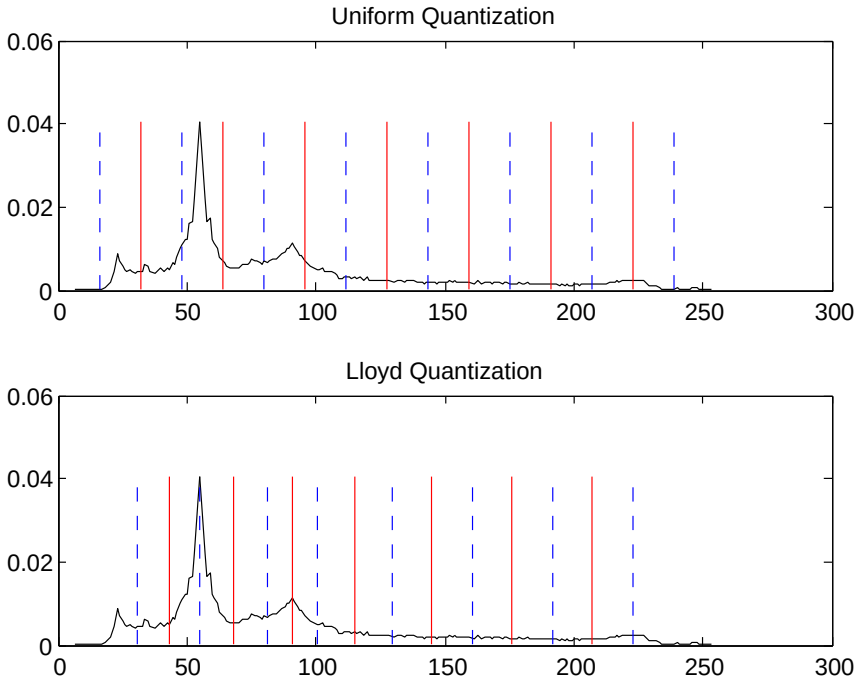

Figure 3.7: density of Linda, M=8 level quantization (23 iterations)

Chapter 4

Different Initial Sets for Lloyd

In this chapter, the effect of different initial sets on Lloyd is investigated. Since Lloyd uses only necessary but no sufficient conditions, there will be different results. The pdf of Lena image (figure 1.4) and the Laplace distribution (see section 2.2) is used to demonstrate the outcome.

4.1 Lena Image

Initial Set 1 (figure 4.1) is uniformly distributed over the occuring values. This means that the minimum and the maximum value of the picture is searched and between these 2 values, M=8 equal intervals are created. *Results after M=8 level Quantization:*

- Lloyd finished after 25 iterations
- $MSE_{mean,start} = 8.6973$
- $MSE_{mean,end} = 5.1879$

Initial Set 2 (figure 4.2) lies on the left side (low value area) of the pdf. Its intervals are very small compared to initial set 1. *Results after M=8 level Quantization:*

- Lloyd finished after 69 iterations
- $MSE_{mean,start} = 1432.5267$
- $MSE_{mean,end} = 5.6256$

Initial Set 3 (figure 4.3) lies on the right side (high value area) of the pdf. Its intervals are very small compared to initial set 1. *Results after M=8 level Quantization:*

- Lloyd finished after 46 iterations
- $MSE_{mean,start} = 1004.2346$
- $MSE_{mean,end} = 6.8126$

Results are shown in figure 4.4.

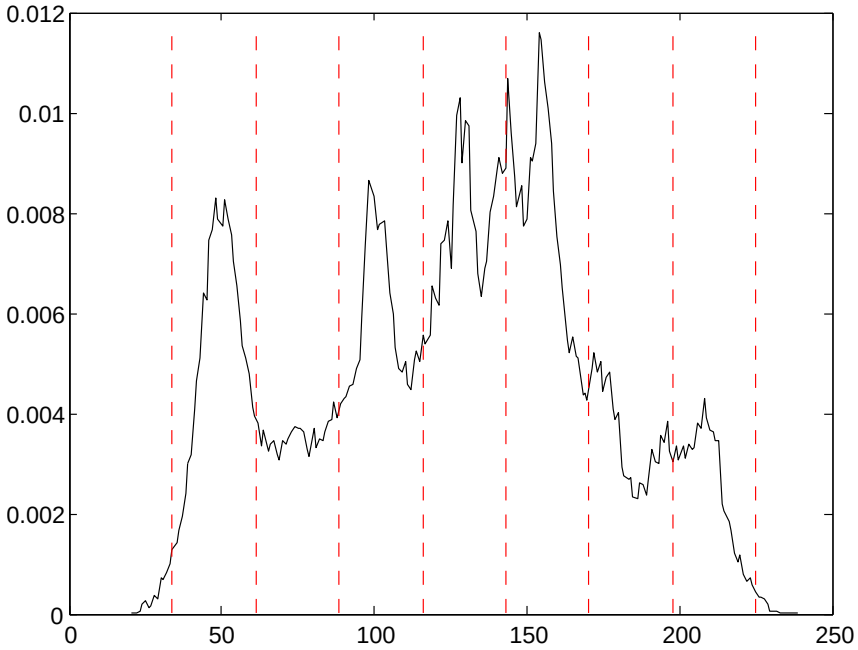

Figure 4.1: $x_{q,m}$ of initial set 1 (red) and Lena density

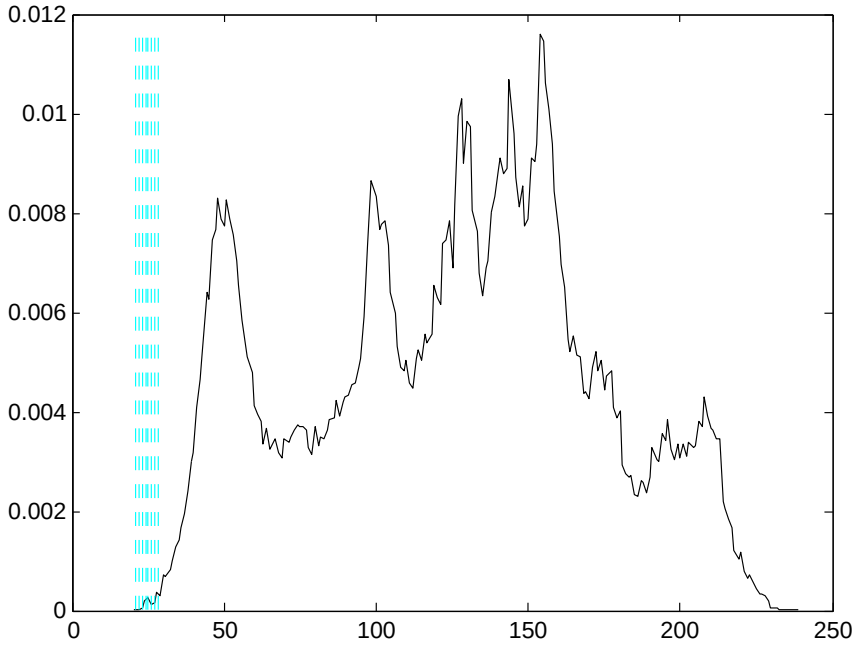

Figure 4.2: $x_{q,m}$ of initial set 2 (cyan) and Lena density

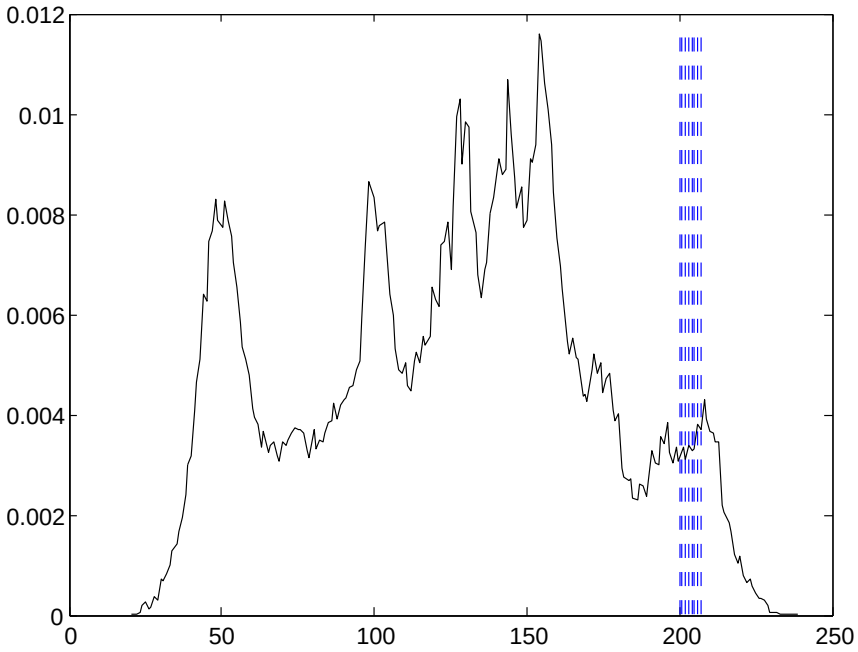

Figure 4.3: $x_{q,m}$ of initial set 3 (blue) and Lena density

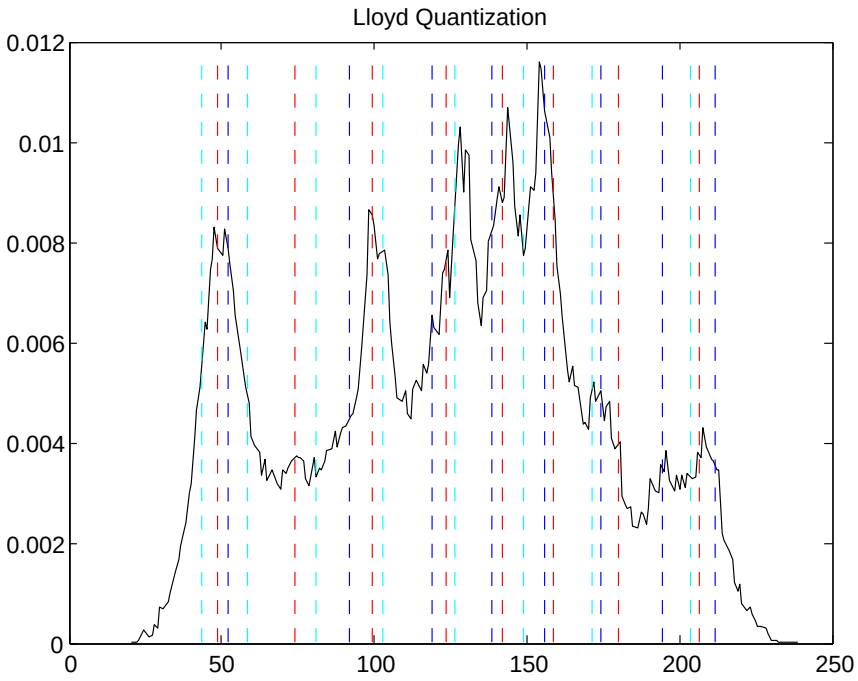

Figure 4.4: $x_{q,m}$ of all initial sets after Lloyd (Lena density)

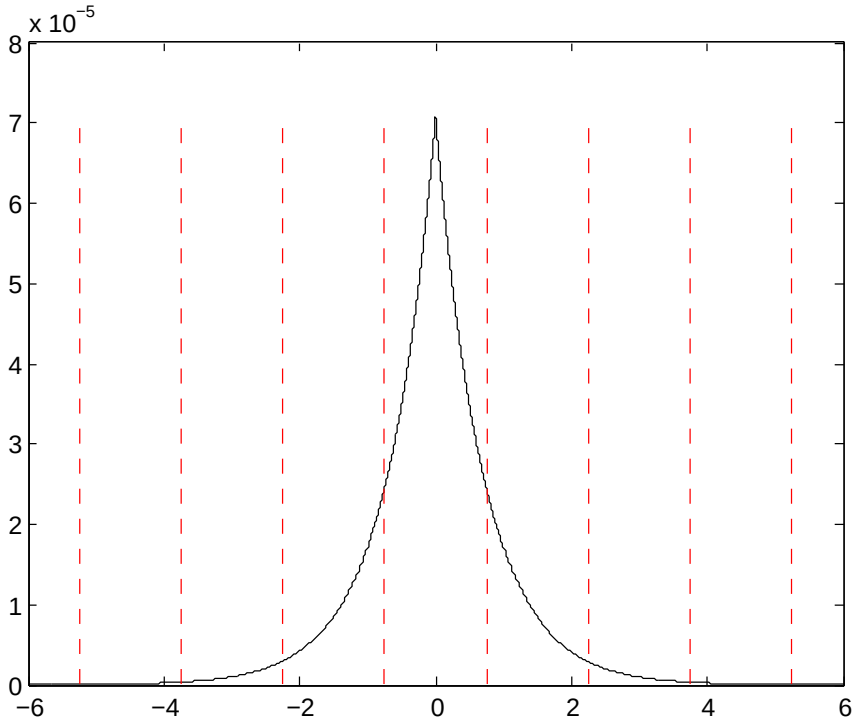

Figure 4.5: $x_{q,m}$ of initial set 1 (red) and Laplace density

4.2 Laplace Distribution

Now the same kind of initial sets are applied on a Laplace distribution.

Initial Set 1 (figure 4.5) *Results after M=8 level Quantization:*

- Lloyd finished after 68 iterations
- $MSE_{mean,start} = 0.0266$
- $MSE_{mean,end} = 0.006453419156562$

Initial Set 2 (figure 4.6) *Results after M=8 level Quantization:*

- Lloyd finished after 445 iterations
- $MSE_{mean,start} = 2.6551$
- $MSE_{mean,end} = 0.006453453702243$

Initial Set 3 (figure 4.7) *Results after M=8 level Quantization:*

- Lloyd finished after 445 iterations
- $MSE_{mean,start} = 2.6551$
- $MSE_{mean,end} = 0.006453453702243$

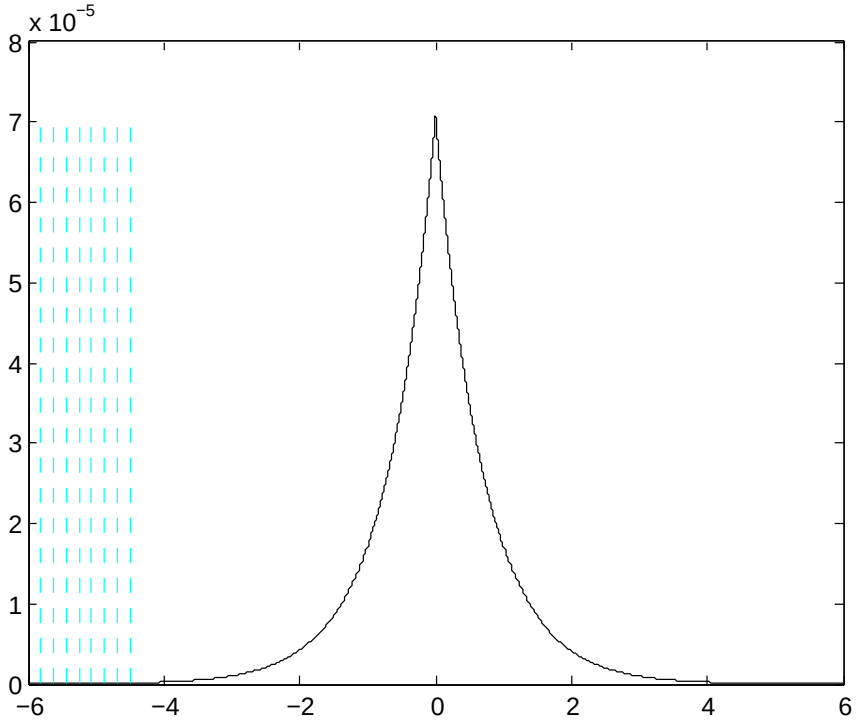

Figure 4.6: $x_{q,m}$ of initial set 2 (cyan) and Laplace density

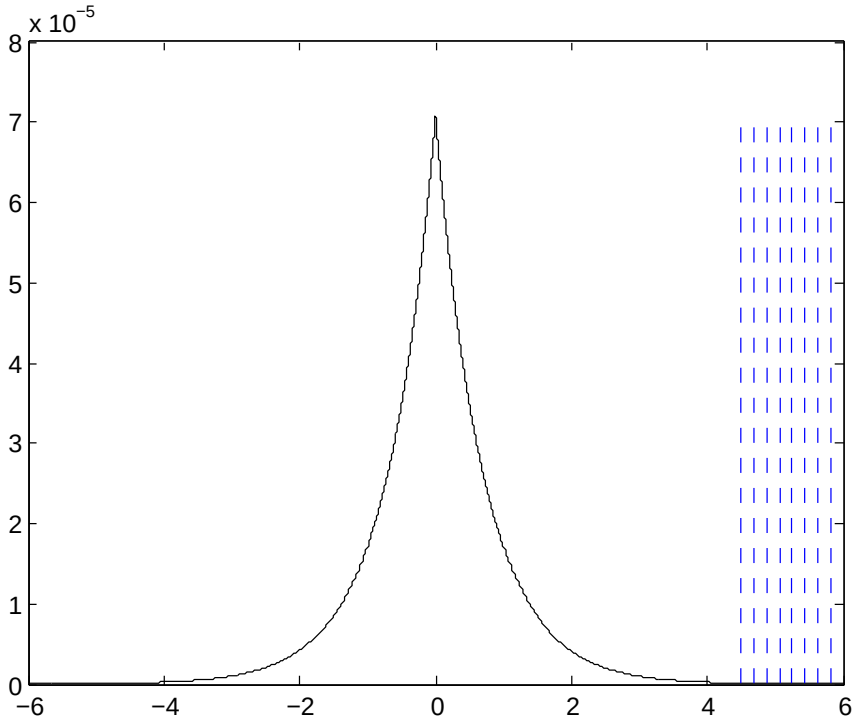

Figure 4.7: $x_{q,m}$ of initial set 3 (blue) and Laplace density

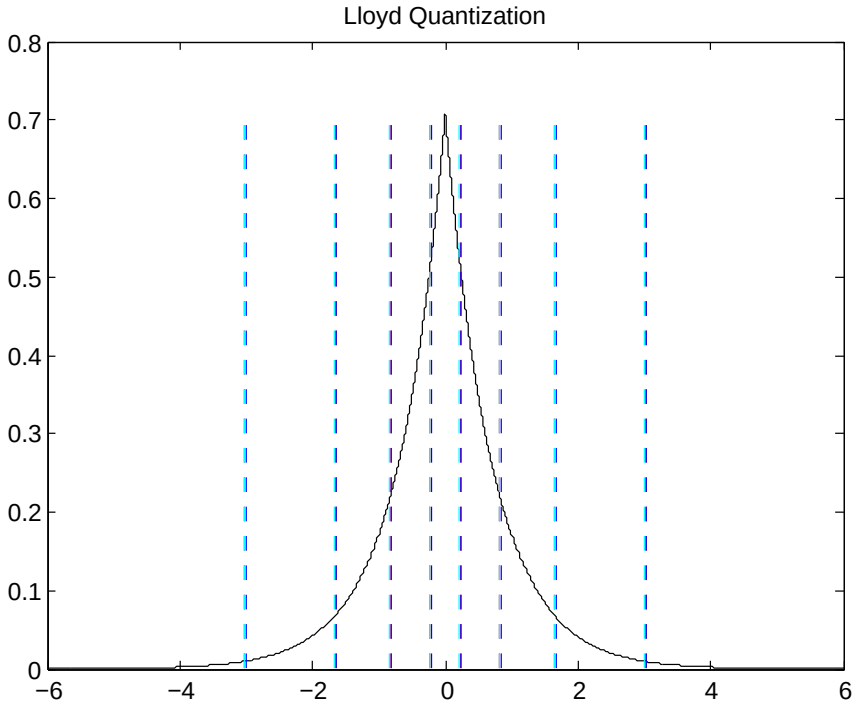

Figure 4.8: $x_{q,m}$ of all initial sets after Lloyd (Laplace density)

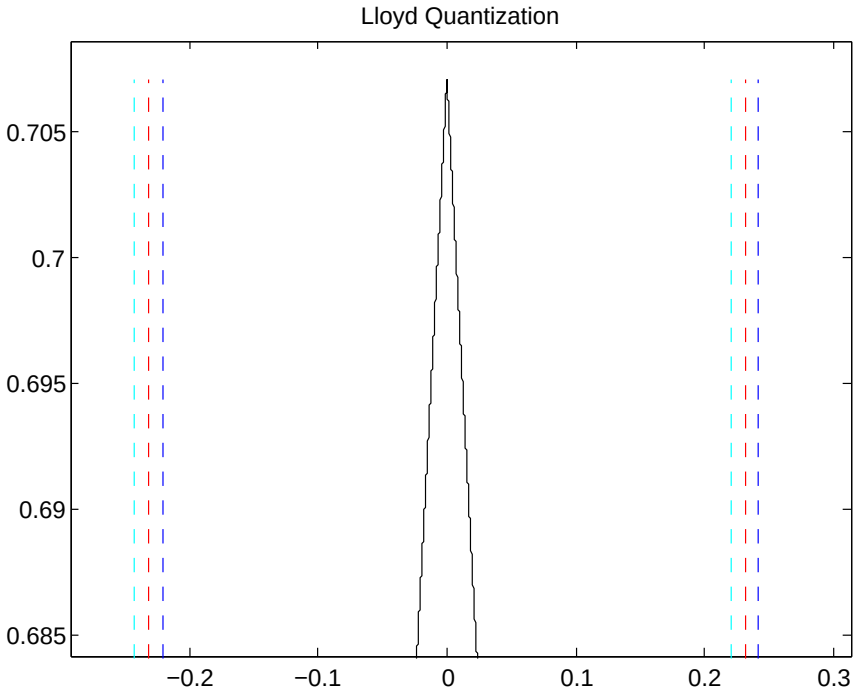

Figure 4.9: $x_{q,m}$ of all initial sets after Lloyd (Laplace density), zoomed in at center

Results are shown in figure 4.8 and 4.9. Since initial set 2 and 3 are symmetric, their MSE_{mean} is the same.

4.3 Conclusion

Figure 4.4 and 4.8 are showing the results of all initial sets after Lloyd. Set 2 (cyan) which started on the left side (low values) is the most left set in the end. Set 3 (blue) which started on the right side is the most right set. Set 1 (red) which was uniformly distributed is between them.
The most important conclusion however is that the MSE_{mean} is smallest using set 1. Therefore, the optimal initial set (heuristic) appears to be equidistant intervals over the occurig values, especially if pdf is symmetric. It always depends on the pdf of course.

Chapter 5

Benefits of Discrete Transformations

In this chapter, the benefits of discrete transformations in combination with Lloyd quantization are discussed. A discrete transformation decorrelates the image data and does energy compression. To demonstrate the procedure, discrete cosine transformation (DCT) is chosen and applied on Lena image (figure 1.4).

Steps in Source Encoding using Discrete Transformation:

1. Digitalized raw image data is obtained (e.g. from sensor)
2. Blockwise 2D DCT is applied
3. Resulting coefficients are reordered to coefficient blocks of same kind
4. Each coefficient block is quantized and values are scaled
5. Run-length-/Entropy-coding is applied
6. Coded data is ready to store or transmit! (A decoder reverses the operations.)

5.1 Discrete Cosine Transformation on Lena

Definitions:

- Energy of an image:

$$E(S) = \sum_{y=0}^{N_y-1} \sum_{x=0}^{N_x-1} (S_{yx})^2 \tag{5.1}$$

- for images, we use 2D-DCT:

$$F_{vu} = \frac{1}{4} C_v C_u \sum_{y=0}^{N_y-1} \sum_{x=0}^{N_x-1} S_{yx} \cos(v\pi \frac{2y+1}{2N_y}) \cos(u\pi \frac{2x+1}{2N_x}) \tag{5.2}$$

  with $C_u = \frac{1}{\sqrt{2}}$ if $u = 0$ else 1, $C_v = \frac{1}{\sqrt{2}}$ if $v = 0$ else 1. S_{yx}... pixel[y,x], discrete input signal. N_x... height of picture, N_y... width of picture.

Beside transformation, we also divide the input picture into blocks/chunks which increases coding efficiency. 8x8 pixel blocks are a good compromise between coding efficiency and complexity. We then call it the 8x8 2D DCT, which is used in the following example.

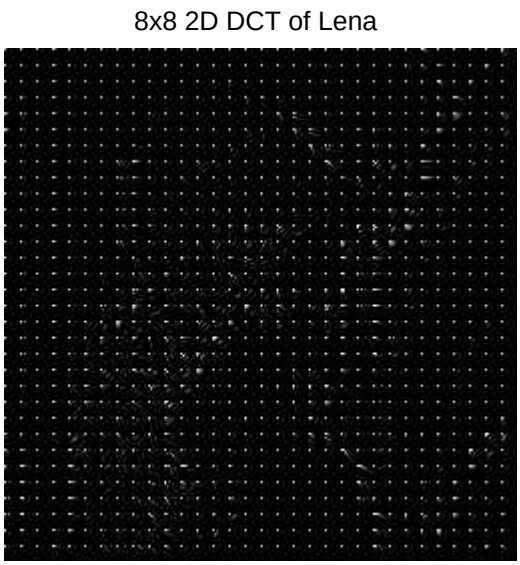

Figure 5.1: Lena after 8x8 2D DCT

2D 8x8 DCT on Lena Image: Now we apply the 8x8 block 2D DCT on Lena image (figure 1.4). The original image was divided into 32x32 8x8 pixel blocks and 2D DCT was applied on each block. The result is shown in figure 5.1. After DCT, the coefficient values are outside the usual 8 bit range, so this is no grayscale image anymore. What can be seen however is that the top left pixel of each block is the brightest meaning highest energy, it is called the *DC coefficient* of the block, because it averages the values of the whole block and therefore behaves like a low pass. The other pixels of the block are called the *AC coefficients*, their values are a lot smaller than the DC coefficient.
Now the coefficients are reordered, all DC coefficients (first pixel of each block) are gathered to one 32x32 pixel block, the same for every AC Coefficient. We obtain 8x8 new coefficient blocks, the first 4 top left blocks are shown in figure 5.2.
Let's take a closer look at those 4 coefficient blocks. Figure 5.3 shows the interpolated (100 points) histogram of the 4 blocks. The DC coefficients look related to the image pdf since they carry the averaged values of every block. The AC coefficients are approximately Laplace distributed.
The most energy is clearly in the DC coefficients. Table 5.1 shows the percentage of the energy of each block compared to the whole energy of the picture. The obvious result: *DC coefficients store almost all the energy!*.
As graphical representation, figure 5.4 shows the logarithm of the Energy of the coefficient blocks.

	1	2	3	4	5	6	7	8
1	97,18504	1,11422	0,26881	0,09023	0,03929	0,01872	0,00857	0,00331
2	0,33749	0,24864	0,10484	0,04600	0,02842	0,01146	0,00663	0,00314
3	0,07048	0,06931	0,05124	0,03466	0,01595	0,01106	0,00585	0,00249
4	0,01873	0,02408	0,02942	0,01731	0,01548	0,00853	0,00401	0,00205
5	0,00817	0,01036	0,01089	0,01114	0,00856	0,00563	0,00311	0,00156
6	0,00392	0,00451	0,00442	0,00475	0,00530	0,00282	0,00168	0,00088
7	0,00148	0,00246	0,00172	0,00214	0,00196	0,00161	0,00070	0,00035
8	0,00054	0,00063	0,00085	0,00055	0,00092	0,00051	0,00027	0,00016

Table 5.1: energy of coefficient blocks comapared to image energy [%] (Lena image)

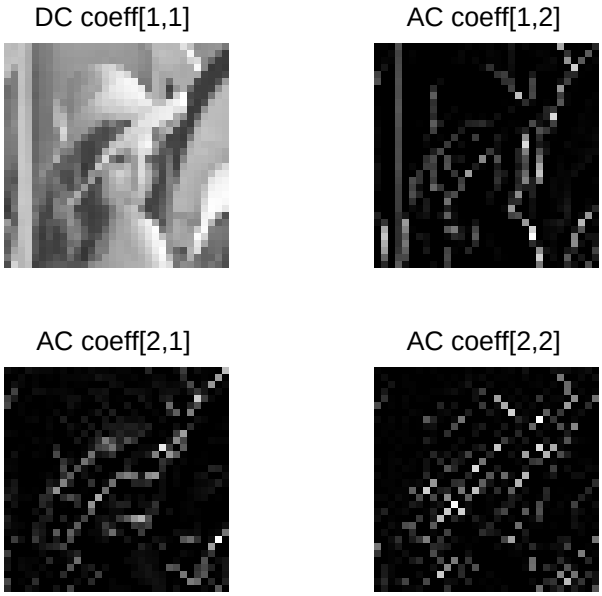

Figure 5.2: Lena 8x8 2D DCT reordered, coefficient blocks (DC and first 3 AC are shown)

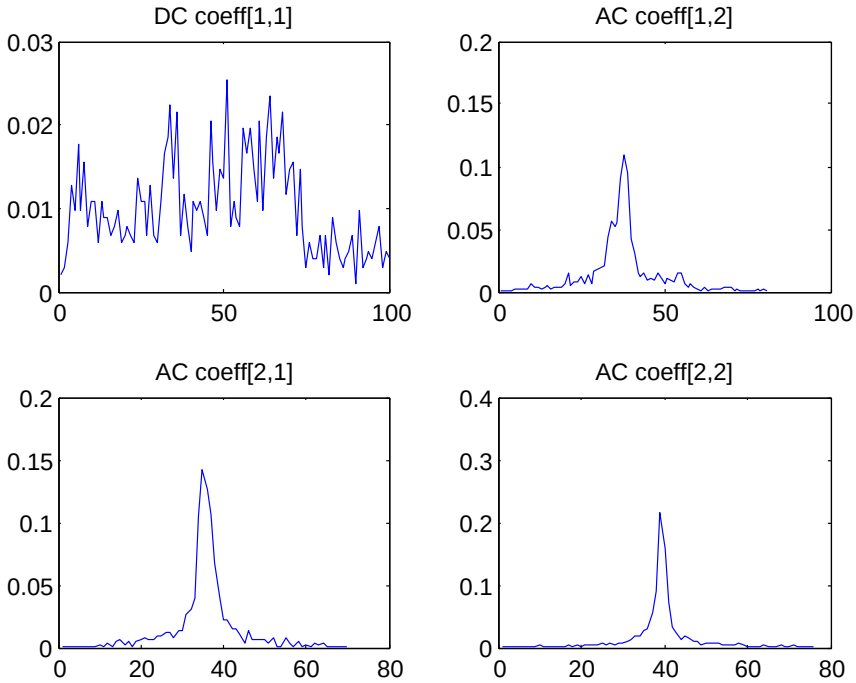

Figure 5.3: Lena after 8x8 2D DCT, densities of coefficient blocks

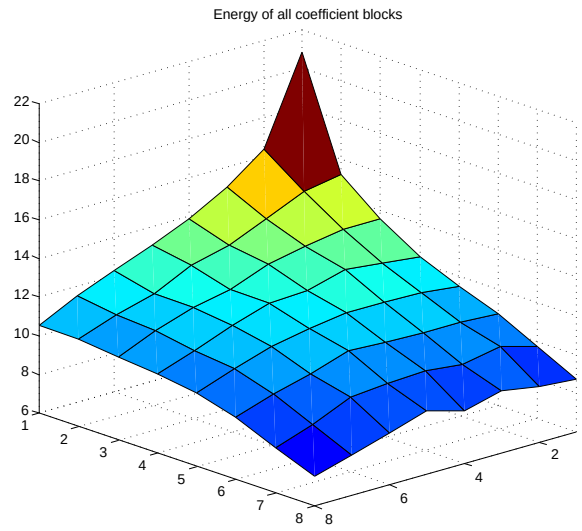

Figure 5.4: Lena after 8x8 2D DCT, energy of all coefficient blocks (logarithmic!)

Now, M=8 level Lloyd and uniform quantization are applied on every coefficient block! After quantization, the coefficient blocks are reordered back to the original form (like figure 5.1), and the inverse 2D DCT is applied. The results are shown in figure 5.5. Figure 5.6 shows the quantized image *without quantizing the DC coefficients*. Quantizing the DC coefficients results in *blocking artefacts* because the average value of each block ("background gray-level") is too coarsely quantized (if M too low), which bothers the viewer pretty much. Without quantizing the DC coefficients, the image looks pretty good, especially if we compare it to the M=8 level quantization without DCT (figure 3.2). The details are well preserved.
The uniform Quantization has got a general graininess, which is not visible using Lloyd. This graininiess is a result of quantizing the AC coefficients. Uniform quantization is not adaptive and "has a hard time" delivering good results, whereas Lloyd's algorithm has no problem in doing so. Therefore, Lloyd and DCT are working well together since the AC coefficients are approx. Laplace distributed and Lloyd has no problem with that.

Summary of 2D block DCT:

- Image is divided into equally sized block, each block is transformed with 2D DCT.
- Resulting coefficients (DC every top left pixel of each block, AC rest) are quantized.
- DC coefficients have "low pass behaviour", meaning they average all the values of each block. They store almost all the energy of the image. They are approximately distributed like the image pdf.
- AC coefficients have "high pass behaviour", they store the detail of the image and are Laplace distributed.
- Coarse quantization of DC coefficients results in blocking artefacts.
- Coarse quantization of AC coefficients results in graininess and loss of detail.
- Lloyd does a better job than uniform quantization because it optimally quantizes Laplace distribution.

Figure 5.5: Lena after $M = 8$ level quantization and DCT

Figure 5.6: Lena after $M = 8$ level quantization and DCT, DC coefficients not quantized

5.2 Conclusion

Table 5.2 and 5.3 show the comparison of uniform and Lloyd quantization, with and without DCT. As error measure, we use formula 2.1. The error of the uniformly quantized picture is twice as big as using Lloyd quantization. Additional 2D DCT reduces the error even more, the result is a picture that looks a lot more like the original one, but with the same quantization level. Furthermore, the entropy H (equation 1.8) is reduced to almost a fourth of the original one. Entropy using DCT is a little higher because it retains more information. This means that the compressed picture can be coded with entropy achieving codes to get a minimal average codeword length of only 2 bit instead of 8 bit per symbol, using M=8 level quantization. This is a huge improvement considered that the image quality is not that bad, especially using Lloyd quantization. In fact, image quality using Lloyd and DCT is a lot better than using uniform quantization and DCT, although the entropy is lower using Lloyd!

	original	quantized	DCT quantized
Error ϵ	0,0000	331,1733	106,2326
Entropy H $[\frac{bit}{pixel}]$	7,4475	1,7955	2,7669

Table 5.2: M=8 level uniform quantization of Lena, with and without DCT

	original	quantized	DCT quantized
Error ϵ	0,0000	164,5855	56,5726
Entropy H $[\frac{bit}{pixel}]$	7,4475	1,9434	1,9863

Table 5.3: M=8 level Lloyd quantization of Lena, with and without DCT

Chapter 6

Epilogue

Now we are at the end of our journey through quantization of images. We remember that every image has got it's distinctive histogram of values, and if we want to quantize it optimally, we have to pay respect to the particular peaks of high value occurances, where we chose numerous small quantization intervals to get as much detail as possible out of it. The areas where very less or no values occure can be discarded, we don't need them. That's what Lloyd's algorithm delivers, it is adaptive to these features. Furthermore, there can be additional conditions, for example making Lloyd shift the intervals until a certain entropy is reached. That makes Lloyd's algorithm a very versatile tool.
We have seen that quantization in general lowers the entropy of the image, which means that information is thrown away. That's obvious since quantization is a lossy compression technique. Discarding information and lowering entropy means that if the source encoded image is stored or transmitted, a lot less bits are needed, which is the basic aim of these techniques.
We introduced discrete cosine transformation (DCT) as a part of source encoding, and it delivers amazing results. Although the entropy after quantization with DCT is pretty similar to the one without DCT, the results look much better. The image has got more detail. This is because the DCT does energy compression and packs together all the important features of the image into one set of coefficients (DC coefficients) that bears about all the energy, the other coefficients are only for the detail and are almost Laplace distributed, which is another plus for the Lloyd quantizer since it has no problems with that (compared to the uniform quantizer).
Quantization is only a small part of source encoding, but it's the sum of the parts which makes the whole process effective. Using transformations and entropy coding techniques along with quantization, todays results of data compression (e.g. JPEG standard) are amazing.
The objective of the whole process is to make these techniques as versatile as the nature itself where the data is derived from, meaning that the most amount of detail can be stored with the least number of bits.

Martin Mayer, September 2010

Bibliography

[1] *http://en.wikipedia.org*, Wikipedia

[2] *"On Asymptotic Solutions of the Lloyd-Max Scalar Quantization"*, by Bormin Huang and Jing Ma, 1-4244-0983-7/07 ©2007 IEEE

List of Figures

List of Tables