МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего профессионального образования
Санкт-Петербургский государственный университет аэрокосмического
приборостроения

М. Р. Гильмутдинов, А. М. Тюрликов, Е. М. Линский

ЦИФРОВАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

СТАТИСТИЧЕСКИЙ АНАЛИЗ И КВАНТОВАНИЕ ВИЗУАЛЬНЫХ ДАННЫХ

Учебное пособие

Санкт-Петербург
2013

УДК 621.397
ББК 30
Г47

Рецензенты:
доктор технических наук, профессор кафедры вычислительной техники СПб НИУ ИТМО *А. Ю. Тропченко*;
доктор технических наук, профессор кафедры информатики и информационной безопасности ПГУПС *П. В. Мирончиков*

Утверждено
редакционно-издательским советом университета
в качестве учебного пособия

Гильмутдинов, М. Р.

Г47 Цифровая обработка изображений. Статистический анализ и квантование визуальных данных: учебное пособие/ М. Р. Гильмутдинов, А. М. Тюрликов, Е. М. Линский; СПб.:ГУАП, 2013. 41 с.
ISBN 978-5-8088-0864-5

Учебное пособие подготовлено кафедрой инфокоммуникационных систем Санкт-Петербургского государственного университета аэрокосмического приборостроения.

В учебном пособии рассматриваются методы статистической обработки и квантования визуальных данных. Рассмотрение теоретических вопросов сопровождается описанием исследований, в ходе которых обучающиеся на практике осваивают методы цифровой обработки визуальных данных. Учебное пособие может быть использовано студентами, обучающимися по направлениям 090900 “Информационная безопасность” и 210700 “Инфокоммуникационные технологии и системы связи”, а также для самостоятельной работы аспирантов.

УДК 621.397
ББК 30

ISBN 978-5-8088-0864-5

Введение

Различные методы цифровой обработки изображений используются в современных стандартах сжатия как неподвижных изображений, так и видеопоследовательностей. В настоящем учебном пособии рассматриваются методы статистической обработки и квантования визуальных данных. Для того чтобы обучающиеся могли применять полученные знания в практической работе, учебное пособие построено как описание двух логически связанных исследований. Каждому исследованию посвящен отдельный раздел учебного пособия. В начале каждого раздела формулируется цель исследования, затем приводятся теоретические сведения, описывается методика проведения исследования. В конце каждого исследования приведены списки обязательных и индивидуальных практических заданий. В качестве основного инструмента, который используется при выполнении практической части исследований, выступает язык программирования C.

Исследование №1
Статистический анализ цифровых изображений

Цель исследования: изучение способов представления изображений, ознакомление со структурой формата BMP, анализ статистических свойств изображений, а также получение практических навыков обработки изображений.

1. Теоретические основы исследования

1.1. Основные определения

Изображение в цифровом виде представляется в виде прямоугольной матрицы, элементы которой называют *пикселями* (от английского *pix-el* — *pic*ture *el*ement). Пиксель может быть представлен одним или несколькими числовыми значениями, каждое из которых принадлежит логической структуре, называемой *компонентой*. В качестве примера можно рассматривать красную (*red*), зеленую (*green*) и синюю (*blue*) компоненты. Будем называть значение пикселя в компоненте A *интенсивностью* и обозначать $I_{y,x}^{(A)}$, где пара (y, x) указывает положение пикселя в изображении (y — номер строки, x — номер столбца).

В нескольких приведенных ниже заданиях требуется построить гистограммы частот для выборок, формируемых на основе данных компонент. Гистограмму следует формировать в виде двумерного графика, где на оси абсцисс отложены все возможные значения в компоненте от 0 до $2^L - 1$, где L определяет размер разрядной сетки. Над каждым значением на оси абсцисс изображается прямоугольник, высота которого пропорциональна частоте появления данного значения в компоненте. Все прямоугольники имеют одинаковую ширину.

1.2. Структура BMP-файла

В данном исследовании рассматриваются изображения, хранимые в наиболее распространенном формате RGB24 без сжатия. Файл в формате BMP имеет следующую структуру:

1. Заголовок.

2. Палитра — таблица цветов (отсутствует для RGB24).

3. Данные по пикселям.

Заголовок состоит из двух последовательно расположенных частей. В файле **Windows.h** содержится описание двух структур, относящихся к этим частям (здесь и далее примеры структур будут приведены на языке Си с ориентацией на MS Visual C).

```
typedef struct tagBITMAPFILEHEADER {
  WORD    bfType;
  DWORD   bfSize;
  WORD    bfReserved1;
  WORD    bfReserved2;
  DWORD   bfOffBits;
} BITMAPFILEHEADER, *PBITMAPFILEHEADER;
```

Поля структуры:

- `bfType` — описывает тип файла, всегда ASCII-код двух символов 'B' и 'M' (заглавные латинские буквы);
- `bfSize` – размер файла в байтах;
- `bfReserved1` — зарезервировано, должно быть 0;
- `bfReserved2` — зарезервировано, должно быть 0;
- `bfOffBits` — смещение в байтах от начала структуры `BITMAPFILEHEADER` до массива пикселей.

```
typedef struct tagBITMAPINFOHEADER{
  DWORD  biSize;
  LONG   biWidth;
  LONG   biHeight;
  WORD   biPlanes;
  WORD   biBitCount;
  DWORD  biCompression;
  DWORD  biSizeImage;
  LONG   biXPelsPerMeter;
  LONG   biYPelsPerMeter;
```

```
  DWORD  biClrUsed;
  DWORD  biClrImportant;
} BITMAPINFOHEADER, *PBITMAPINFOHEADER;
```

Поля структуры:

- `biSize` — размер структуры в байтах;
- `biWidth` — ширина изображения в пикселях;
- `biHeight` — ширина изображения в пикселях. Если эта величина положительная, то строки изображения хранятся снизу вверх. Если высота отрицательна, то строки изображения идут сверху вниз;
- `biPlanes` — количество плоскостей изображения всегда должно быть 1;
- `biBitCount` — количество бит на пиксель. Для RGB24 это поле равно 24, таким образом, каждый пиксель описывается 8-ми битной интенсивностью красного, зеленого и синего;
- `biCompression` — тип сжатия, для RGB24 равно `BI_RGB` (нулевое значение);
- `biSizeImage` — размер изображения в байтах. Необязательное поле может быть равно нулю;
- `biXPelsPerMeter` — горизонтальное разрешение в пикселях на метр. Используется для указания физического размера изображения. Приложение, работающее с этим изображением, может использовать эту информацию при выводе на специфическое устройство отображения. Например, значение 2834 соответствует 72dpi — точкам на дюйм (Dots per Inch), что являлось стандартным разрешением экранов мониторов Apple Macintosh. Поле носит информативный характер и не является обязательным (значение этого поля, как правило, нулевое);
- `biYPelsPerMeter` — вертикальное разрешение в пикселях на метр (назначение этого поля аналогично `biXPelsPerMeter`);
- `biClrUsed` — размер палитры (равно 0 для RGB24);
- `biClrImportant` — число значимых элементов палитры. Равно 0 для RGB24 . Вопросы работы с палитрой будут подробно рассмотрены в следующем исследовании.

За структурами по смещению, указанному в `bfOffBits`, располагаются данные, относящиеся к пикселям изображения.

1.3. Особенности представления данных в формате Windows BMP

Здесь перечислено несколько важных особенностей представления данных в формате RGB24, характерных для операционной системы Windows:

• Данные представляются в виде одномерного массива, в котором значения, относящиеся к отдельным пикселям, записаны строка за строкой.

• Строки изображения могут идти как снизу вверх (наиболее частый случай), так и сверху вниз.

• Ширина строки *в байтах* должна быть выравнена по границе двойного слова (32 бита), т.е. должна быть кратна числу 4. При необходимости для выполнения этого условия в конце *каждой* строки добавляются дополнительные байты (от одного до трех), значения которых несущественны. В англоязычной литературе добавляемые байты называют *dummy bytes*.

• Каждый пиксел в формате RGB24 представляется тремя байтами. Первый содержит значение компоненты B, второй — G и третий — R.

1.4. Критерии эффективности обработки изображений

Под *обработкой* будем понимать применение к изображению некоторой процедуры *преобразования* пикселей. Примерами преобразования изображения могут служить методы фильтрации, масштабирования или сжатия. Иногда, например в задаче сжатия, для восстановления исходного изображения требуется дополнительное *обратное преобразование*. Преобразования могут быть *обратимыми* или *с потерями*. Очевидно, что при использовании обратимого преобразования по данным, полученным в результате преобразования, можно в точности восстановить исходное изображение. Для оценки искажений, вносимых при обработке с потерями, часто используют понятие *шума*, который как-бы был внесен в процессе применения цепочки «преобразование — обратное преобразование». Для определения количественных характеристик шума интенсивность пикселя исходного изображения на позиции (y, x) обозначим за $I^{(A)}_{y,x}$ (в строке y и в столбце x), а интенсивность пикселя восстановленного изображения на той же позиции обозначим за $\hat{I}^{(A)}_{y,x}$. Размеры изображения будем полагать равными W по ширине и H по

высоте. Тогда шумом на позиции (y, x) в компоненте A можно считать разность между исходным и восстановленным пикселем:

$$N_{y,x}^{(A)} = I_{y,x}^{(A)} - \hat{I}_{y,x}^{(A)}.$$

Наиболее простыми с вычислительной точки зрения критериями оценки искажений при обработке изображений с потерями являются сумма абсолютных значений разностей между исходными и восстановленными после обработки пикселями (*Sum of Absolute Differences* — SAD) и сумма квадратов разностей между исходными и восстановленными пикселями (*Sum of Squared Differences* — SSD):

$$SAD = \sum_{i=1}^{H} \sum_{j=1}^{W} |I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)}|; \tag{1.1}$$

$$SSD = \sum_{i=1}^{H} \sum_{j=1}^{W} (I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)})^2. \tag{1.2}$$

Вместо слова *разность* (*Difference*) иногда используют слово *ошибка* (*Error*).

Данные критерии часто используются для поиска и сравнения отдельных фрагментов изображения. Для сравнения изображений разной размерности или фрагментов изображений используют усредненные значения SAD (*Mean of Absolute Errors* — MAE) и SSD (*Mean of Squared Errors* — MSE):

$$MAE = \frac{1}{WH} \sum_{i=1}^{H} \sum_{j=1}^{W} |I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)}|; \tag{1.3}$$

$$MSE = \frac{1}{WH} \sum_{i=1}^{H} \sum_{j=1}^{W} (I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)})^2, \tag{1.4}$$

где W является шириной, а H — высотой фрагмента или изображения.

Более распространенным критерием оценки искажений, вносимых в изображение, является *пиковое отношение сигнал/шум* (*PSNR* — *peak signal-to-noise ratio*). Эта мера аналогична используемому при обработ-

ке звука *отношению сигнал/шум*, (*SNR — signal-to-noise ratio*):

$$SNR = 10 \lg \frac{E_S}{E_N} = 10 \lg \frac{\sum\limits_{i=1}^{H} \sum\limits_{j=1}^{W} \left(I_{i,j}^{(A)}\right)^2}{\sum\limits_{i=1}^{H} \sum\limits_{j=1}^{W} \left(I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)}\right)^2}, \tag{1.5}$$

где E_S определяет энергию исходного сигнала; E_N — энергию шума. Энергия дискретного по времени сигнала определяется как сумма квадратов всех его значений.

И для аудиоданных, и для изображений шум интерпретируется как несоответствие исходного и обработанного (восстановленного) сигнала, а энергия шума — как сумма квадратов разностей отсчетов исходного и обработанного сигналов, аналогично формуле (1.2). Отличие $PSNR$ от SNR заключается в том, что все значения исходного сигнала в числителе дроби заменяются на максимальное значение интенсивности пикселя в данной компоненте. Однако для упрощения вычислений максимальное значение сигнала, как правило, заменяют на максимально возможное значение, которое может принимать интенсивность пикселя. Таким образом, в используемой разрядной сетке L бит значение $PSNR$ вычисляется по следующей формуле:

$$PSNR = 10 \lg \frac{WH(2^L - 1)^2}{\sum\limits_{i=1}^{H} \sum\limits_{j=1}^{W} \left(I_{i,j}^{(A)} - \hat{I}_{i,j}^{(A)}\right)^2}. \tag{1.6}$$

Более подробно о критериях оценки искажений можно узнать в работах [1], [2].

Эффективность сжатия для изображений обычно оценивается как среднее число бит, затрачиваемых на представление одного пикселя (Bits Per Pixel — BPP) изображения или одной из его компонент (когда учитываются интенсивности всех пикселей только в данной компоненте). Наиболее простым способом анализа эффективности сжатия является оценка энтропии при поэлементном независимом сжатии. В этом случае значения интенсивностей обрабатываемой компоненты интерпретируются как сообщения некоторого постоянного источника X (стационарного источника, на выходе которого сообщения независимы). Алфавит источника X формируется по всем возможным значениям

пикселей оцениваемой компоненты. Величина $\hat{p}(x)$ является оценкой вероятности сообщения x, которая вычисляется как

$$\hat{p}(x) = \frac{n_x}{n},$$

где n_x — количество пикселей в данной компоненте, которые имеют значение x, а n — общее число пикселей в компоненте. Тогда оценка энтропии при поэлементном независимом сжатии вычисляется по следующей формуле [3]:

$$\hat{H}(X) = -\sum_x \hat{p}(x) \log_2 \hat{p}(x). \tag{1.7}$$

1.5. Преобразование цветового пространства

В цветных фотореалистичных изображениях, представленных в формате RGB24 , данные цветовых компонент на соответствующих позициях часто близки по значению (в этом случае говорят, что компоненты сильно коррелируют), поэтому с точки зрения сжатия формат RGB24 является заведомо избыточным. В стандартах телевизионного вещания, а также в серии стандартов сжатия JPEG используется другой формат представления изображений — YC_bC_r. В соответствии с форматом YC_bC_r изображение также представляется с помощью трех компонент, но эти компоненты меньше коррелируют друг с другом. Используя следующие формулы, компоненты R, G и B преобразуются в *яркостную* (Y) и две *цветоразностные* (C_b и C_r) компоненты [4], [1], [2], :

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B; \\ C_b &= 0.5643(B - Y) + 128; \\ C_r &= 0.7132(R - Y) + 128. \end{aligned} \tag{1.8}$$

Обратное преобразование выполняется по формулам:

$$\begin{aligned} G &= Y - 0.714(C_r - 128) - 0.334(C_b - 128); \\ R &= Y + 1.402(C_r - 128); \\ B &= Y + 1.772(C_b - 128). \end{aligned} \tag{1.9}$$

Стоит отметить, что реализация прямого и обратного преобразования выполняется с использованием арифметического округления, т.е.

преобразование форматов в общем случае является необратимым. Более того, в результате обратного преобразования возможен выход восстановленных значений R, G, и B за границы исходного диапазона. В этом случае после восстановления необходимо выполнить операцию *клиппирования* (clipping) или *насыщения* (*saturation*):

$$\mathrm{Sat}(x, x_{min}, x_{max}) = \begin{cases} x_{min} & \text{, если } x < x_{min}; \\ x_{max} & \text{, если } x > x_{max}; \\ x & \text{, во всех иных случаях.} \end{cases} \tag{1.10}$$

Большая часть визуальной информации для формата YC_bC_r сосредоточена в яркостной компоненте Y. На практике очень часто к компонентам C_b и C_r применяется процедура *децимации* или *прореживания*, изображенная на рис. 1.1, в процессе которой часть отсчетов исключается из рассмотрения. Искажения в восстановленных после прореживания компонентах C_b и C_r, как правило, несущественны, так как они содержат небольшое количество визуальной информации (см. п.8 из подраздела 2.).

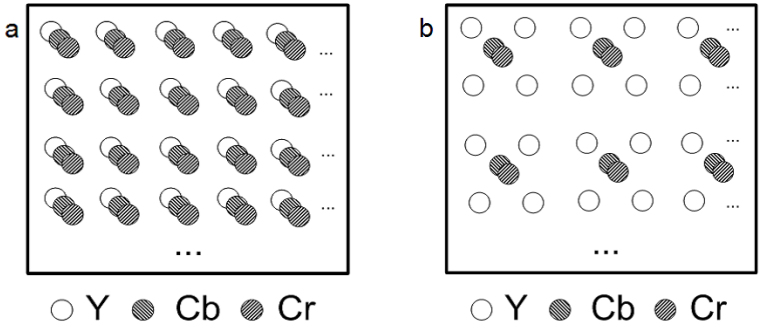

Рис. 1.1. Децимация компонент C_b и C_r, в соответствии с форматом 4:2:0 : a — Исходные данные; b — после децимации

Под децимацией в данном исследовании будет упоминаться прореживание в N раз по ширине и в N раз по высоте.

2. Порядок выполнения исследования

1. Согласовать с преподавателем BMP-файл в формате RGB24 .

2. Выполнить загрузку в память указанного BMP-файла.

3. Выделить содержимое компонент R, G и B и сохранить их в отдельных файлах в формате RGB24 по следующему правилу: байты, не относящиеся к сохраняемой компоненте должны быть обнулены. В этом случае мы получим три файла с соответствующими цветами.

4. Проанализировать корреляционные свойства компонент R, G и B, используя формулу оценки коэффициента корреляции:

$$\hat{r}_{A,B} = \frac{\hat{M}[(A - \hat{M}[A])(B - \hat{M}[B])]}{\hat{\sigma_A}\hat{\sigma_B}}, \tag{1.11}$$

где A и B — компоненты изображения; $\hat{M}[.]$ — оценка математического ожидания в соответствии с формулой

$$\hat{M}[I^{(A)}] = \frac{1}{WH}\sum_{i=1}^{H}\sum_{j=1}^{W} I_{i,j}^{(A)},$$

где $\hat{\sigma_A}$ и $\hat{\sigma_B}$ — оценки среднеквадратичного отклонения компонент A и B, вычисляемые по формуле

$$\hat{\sigma_A} = \sqrt{\frac{1}{WH-1}\sum_{i=1}^{H}\sum_{j=1}^{W}\left(I_{i,j}^{(A)} - \hat{M}[I^{(A)}]\right)^2}.$$

Дополнительную информацию по корреляционному анализу можно найти в главах 12 и 18 книги [5]. При анализе необходимо:

a) вычислить оценки коэффициентов корреляции между каждой парой компонент изображения $\hat{r}_{R,G}$, $\hat{r}_{R,B}$ и $\hat{r}_{B,G}$.

b) построить сечения трехмерного графика оценки нормированной автокорреляционной функции $r_{A,A}(x, y)$. Под автокорреляционной функцией здесь понимается коэффициент корреляции, рассчитанный по двум выборкам. Первая формируется по данным $A_{i,j}$, где $i = 1, ..., H - y$, а $j = 1, ..., W - x$. Вторая выборка формируется по $A_{m,n}$, где $m = y + 1, ..., H$, а $n = x + 1, ..., W$. Значения по оси y необходимо зафиксировать в точках (0, ±5, ±10). Шаг по оси x определить самостоятельно. Максимальное значение x не должно превышать четверти значения ширины изображения W. Таким образом необходимо построить пять графиков сечения автокорреляционной функции для каждой компоненты.

5. Выполнить преобразование данных из формата RGB в YC_bC_r, используя формулы (1.8). Для компонент Y, C_b и C_r вычислить оценки в соответствии с пунктом 4.

6. Сохранить содержимое каждой из компонент Y, C_b и C_r в отдельном файле в формате RGB24 . Значение компоненты для каждого пикселя следует сохранять во всех трех байтах, соответствующих позициям R, G и B этого пикселя. В этом случае будут сформированы три файла с интенсивностями в градациях серого цвета.

7. Выполнить обратное преобразование данных из формата YC_bC_r в RGB, используя формулы (1.9). Для компонент B, G, и R вычислить значение $PSNR$ по исходным и восстановленным данным.

8. Произвести децимацию компонент C_b и C_r в два раза по ширине и в два раза по высоте следующими способами:

a) исключив строки и столбцы с четными номерами;

b) вычислив значения элементов прореженного изображения как среднее арифметическое четырех смежных элементов из исходного изображения.

9. Произвести восстановление исходного размера для компонент C_b и C_r путем копирования недостающих значений слева и/или сверху, после чего выполнить обратное преобразование из YC_bC_r в RGB, используя формулы (1.9).

10. Для компонент C_b, C_r, B, G, и R вычислить значение $PSNR$ по исходным и восстановленным данным.

11. Повторить пункты 8 – 10, уменьшив размеры компонент C_b и C_r в 4 раза по ширине и высоте.

12. Построить гистограммы частот для компонент R, G, B, Y, C_b и C_r. Гистограммы для каждой компоненты необходимо построить отдельном рисунке. Компоненты C_b и C_r брать с исходными размерами.

13. Оценить число бит, затрачиваемых при поэлементном независимом сжатии компонент R, G, B, Y, C_b и C_r, воспользовавшись формулой (1.7).

14. Следующая серия заданий предназначена для количественного анализа эффективности разностного кодирования, которое используется в алгоритмах класса Differential Pulse-Code Modulation ($DPCM$). Требуется сформировать массивы разностей $D_A^{(r)}$, где A указывает компоненту исходного изображения, для которой вычисляется разность, а

r — номер правила вычисления разности. Значение на позиции (i, j) следует вычислять по следующей формуле:

$$d_A^{(r)}(i,j) = a(i,j) - f^{(r)}(i,j),$$

где $a(i,j)$ — значение пикселя в компоненте A на позиции (i,j), а $f^{(r)}(i,j)$ — правило с номером r, по которому вычисляется предсказание пикселя на позиции (i,j). В исследовании следует использовать следующие правила предсказания:

1 — сосед слева (на позиции $(i, j-1)$);

2 — сосед сверху (на позиции $(i-1, j)$);

3 — сосед сверху слева (на позиции $(i-1, j-1)$);

4 — среднее арифметическое трех соседей — сверху, слева и сверху слева.

Пиксели, находящиеся в первой строке и в первом столбце, из рассмотрения исключить, так как они не имеют некоторых соседей для построения предсказания. Массивы следует построить для компонент R, G, B, Y, C_b и C_r.

15. Для массивов $D_A^{(r)}$, сформированных в предыдущем задании, построить гистограммы частот и сравнить их с гистограммами значений соответствующих компонент.

16. Оценить число бит, затрачиваемых при поэлементном независимом сжатии массивов $D_A^{(r)}$, воспользовавшись формулой (1.7). Сравнить полученные значения с оценками энтропии для соответствующих компонент.

17. Согласовать с преподавателем и выполнить индивидуальное задание из перечня индивидуальных заданий.

3. Требования к оформлению отчета

Отчет должен содержать следующие пункты:

1. постановка задачи;

2. описание структуры формата RGB24 . Дать дополнительные пояснения, какие значения полей использовались при выполнении исследования;

3. описание алгоритма работы (чтение/запись) с файлом RGB24 ;

4. описание реализованных алгоритмов;

5. результаты (графики, численные расчеты) выполнения исследования;

6. результаты (графики, численные расчеты) выполнения индивидуального задания;

7. выводы, сформулированные на основе полученных результатов;

8. листинги реализованных программ.

4. Перечень индивидуальных заданий

1. Изменение местоположения пикселей исходного файла для получения следующих результатов:

a) сформировать зеркальное изображение (отразить относительно вертикали);

b) результирующий файл должен содержать исходное изображение, отраженное относительно горизонтали (эффект отражения в воде);

c) осуществить поворот изображения против часовой стрелки на 90^o ;

d) то же на 180^o;

e) то же на 270^o.

2. Изменение яркости и цвета изображения для получения следующих результатов:

a) изменение яркости на N градаций;

b) удаление цвета из исходного изображения (перевод в полутоновый формат представления);

c) формирование битонального изображения по порогу Thr. Если значение яркости пикселя превышает порог, то оно заменяется на максимальное (в нашем случае — 255). В противном случае значение пикселя заменяется на минимальное, т.е. нулевое.

3. Разложение изображения на битовые плоскости.

Для выполнения этой серии заданий необходимо представить матрицу значений яркостной компоненты Y в виде набора двоичных матриц Y_i^B ($i = 0, ..., 7$ для восьмибитных значений), называемых *битовыми плоскостями*. Каждая плоскость формируется по следующим правилам:

- значение бита $Y_i^B(y, x)$, расположенного в строке y и столбце

x, соответствует биту с номером i пикселя, находящегося в той же позиции компоненты Y;

• нумерация выполняется от старших бит к младшим.

Требуется выполнить следующие действия:

a) сформировать восемь битональных изображений на основе значений битовых плоскостей;

b) вычислить коэффициент корреляции между каждой парой битовых плоскостей. Результаты сформировать в виде сводной таблицы;

c) построить гистограммы частот и оценить число бит, затрачиваемых при поэлементном независимом сжатии каждой битовой плоскости, воспользовавшись формулой (1.7). Результаты сформировать в виде сводной таблицы.

4. Разложение изображения на подкадры.

Для выполнения этого задания потребуется представить исходную яркостную компоненту Y в виде четырех *поджаров* $Y_{(i,j)}$ $(i,j=0,1)$, которые формируются следующим образом. $Y_{(i,j)}(y,x) = Y(2y+i, 2x+j)$. Например, кадр $Y(0,0)$ формируется по значениям яркостей пикселей, находящихся в четных строках и четных столбцах исходного изображения. Необходимо выполнить следующие действия:

a) сформировать четыре BMP-файла, содержащих подкадры $Y(i,j)$.

b) построить график автокорреляционной функции для каждого подкадра.

c) вычислить коэффициент корреляции между каждой парой подкадров.

d) построить гистограммы частот и оценить число бит, затрачиваемых при поэлементном независимом сжатии каждого подкадра, воспользовавшись формулой (1.7).

e) модифицировать значения подкадров следующим образом. Подкадр $Y_{(0,0)}$ оставить без изменений $Y'_{(0,0)}(y,x) = Y_{(0,0)}(y,x)$. Пиксели подкадра $Y_{(0,1)}$ представить в виде разности пикселей исходного и соответствующих пикселей подкадра $Y_{(0,0)}$: $Y_{(0,1)}(y,x)' = Y_{(0,1)}(y,x) - Y_{(0,0)}(y,x)$. Пиксели подкадра $Y_{(1,0)}$ представить в виде разности пикселей исходного и полусуммы соответствующих пикселей из $Y_{(0,0)}$ и $Y_{(0,1)}$: $Y_{(1,0)}(y,x)' =$

$Y_{(1,0)}(y,x) - round((Y_{(0,1)}(y,x) + Y_{(0,0)}(y,x))/2)$. Пиксели последнего подкадра формируются как разность его значений и среднего по соответствующим пикселям трех первых подкадров: $Y_{(1,1)}(y,x)' = Y_{(1,1)}(y,x) - round((Y_{(1,0)}(y,x) + Y_{(0,1)}(y,x) + Y_{(0,0)}(y,x))/3)$.

Построить гистограммы частот и оценить число бит, затрачиваемых при поэлементном независимом сжатии каждого модифицированного подкадра.

5. Вопросы для самопроверки

1. Какое число дополнительных (*dummy*) байтов необходимо добавлять в конце каждой строки, если ширина изображения составляет 301 пиксель?

2. Какое значение следует поместить в поле `biSizeImage` структуры `BITMAPINFOHEADER`, если изображение с разрешением 305×77 пикселей представляется в формате RGB24 ?

3. Для каких компонент производится операция *прореживания*? Какое обоснование справедливо для применения этой операции?

4. Для каких целей выполняется операция *насыщения* (*клиппирования*)?

5. Каким образом с помощью формата RGB24 можно сохранить черно-белое изображение (в оттенках серого)?

Исследование №2
Квантование данных

Цель исследования: изучение методов квантования на примере обработки изображений, получение практических навыков работы с форматом BMP, содержащим палитру изображения.

1. Теоретические основы исследования

1.1. Классификация методов квантования

В широком смысле под *квантованием* понимают правило отображения элементов x из некоторого исходного множества X в элементы другого множества Y:

$$y = Q(x), \tag{2.1}$$

причем процедура отображения такова, что несколько элементов из X (для непрерывного множества это подмножество значений) могут отображаться в один и тот же элемент из Y. Подмножество из X, все элементы которого отображаются в один и тот же элемент из Y, образуют *квант*. Множество Y принято называть *аппроксимирующим множеством*. С точки зрения теории множеств, множество X *мощнее* Y. Как правило, множество Y является конечным или счетным.

Конечный результат процедуры квантования зависит от конкретной прикладной задачи, в которой она применяется. Это может быть аппроксимирующий элемент y или номер кванта, в который попадает x. Результат представляется в виде индекса или кода, удобного для дальнейшего хранения или передачи в цифровой форме.

Пример 1. Применение квантования в цифровой технике.

Квантование является одним из методов, применяемых для перевода аналогового сигнала в цифровую форму. Процедура аналого-цифрового преобразования схематично изображена на рис. 2.1. С помощью *дискретизации* аналоговый сигнал преобразуется в дискретный, после чего дискретные *отсчеты* преобразуются в значения цифрового сигнала с помощью процедуры квантования.

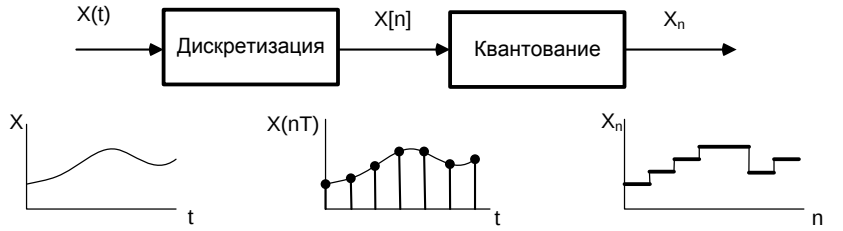

Рис. 2.1. Схема аналого-цифрового преобразования

Квантование также называют *дискретизацией сигнала по уровню*, имея в виду, что по завершению процедуры значения цифрового сигнала выбираются из конечного множества, мощность которого, как правило, равна степени числа 2.

В данном исследовании будут рассмотрены три способа квантования:

1. скалярное равномерное;
2. скалярное неравномерное;
3. векторное.

Квантование называют *скалярным*, если элементами множеств X и Y являются числовые (скалярные) величины. Если элементы обоих множеств являются векторами, то квантование называют *векторным*. В этом случае при отображении по формуле (2.1) длины всех векторов $\bar{x}$ и $\bar{y}$ должны совпадать.

1.2. Критерии оценки эффективности квантования

Поскольку по определению множество X мощнее аппроксимирующего множества Y, квантование в общем случае является преобразованием с потерями. Эффективность квантования, как правило, оценивается по *ошибке*, вносимой при выполнении процедуры. Однако в некоторых прикладных задачах могут также использоваться дополнительные критерии. Например, в задаче сжатия источника с потерями, в которой также применяется квантование, для оценки эффективности совместно с ошибкой используют степень достигаемого сжатия.

Введем в рассмотрение модель дискретного источника, выход которого будет подвергаться квантованию. Данные источника представляют собой *независимые*, *одинаково распределенные* случайные величины X (будем обозначать случайную величину по аналогии с множеством квантуемых значений, поскольку эти понятия взаимосвязаны). Также будем полагать, что случайная величина X является непрерывной и имеет плотность распределения $f(x)$.

Для количественного измерения ошибки квантования будем использовать метрику *эвклидова расстояния*. Если квантованию подвергаются векторы $\bar{x}$ длины n, то в качестве расстояния $\|\bar{x}-\bar{y}\|$ будем использовать евклидово расстояние в n-мерном пространстве:

$$\|\bar{x} - \bar{y}\| \triangleq \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}, \tag{2.2}$$

где x_i и y_i определяют i-ю позицию в соответствующем векторе. Для скалярных величин x и y в качестве расстояния $\|x - y\|$ будем использовать абсолютную разность двух этих величин.

С помощью формулы (2.2) определяется ошибка квантования только для одного элемента исходного множества X. В качестве интегрального критерия эффективности обычно используют взвешенный интеграл квадратов ошибок, который называют *средним квадратом ошибки*:

$$\varepsilon_X^2 = \int_{-\infty}^{\infty} e^2(x) f(x) dx = \int_{-\infty}^{\infty} \|x - Q(x)\|^2 f(x) dx, \tag{2.3}$$

где $f(x)$ задает плотность распределения случайной величины, принимающей значения из X.

Также в качестве критерия оценки ошибки, вносимой при скалярном квантовании, используют отношение сигнал-шум:

$$SNR = 10 \lg \frac{\sigma_X^2}{\varepsilon_X^2}. \tag{2.4}$$

Заметим, что формула (2.4) имеет тот же смысл, что и (1.5), но в качестве энергии сигнала используется σ_X^2 — дисперсия случайной величины, которая подвергается квантованию (обратите внимание, что X должна быть центрированной случайной величиной, т.е. $M[X] = 0$). А в качестве энергии шума используется ε_X^2 — значение среднего квадрата ошибки, определяемого по формуле (2.3).

1.3. Скалярное квантование

1.3.1. Общие сведения

Поскольку элементами множества X и Y для скалярного квантования являются числа, для простоты будем далее полагать, что множество X есть подмножество вещественных чисел. Тогда кванты будут представлять собой интервалы вещественной оси, а определение процедуры квантования можно интерпретировать как задание способа разбиения числовой оси на эти интервалы и выбора в каждом из них единственного аппроксимирующего значения. Таким образом, для того чтобы определить процедуру скалярного квантования, необходимо задать следующие величины:

1. Число квантов N. Кванты при этом нумеруются от 1 до N. Для упрощения кодирования номера кванта с помощью равномерного кода, число квантов выбирают как $N = 2^R$, где R определяет число двоичных разрядов равномерного кода.

2. Границы квантов $t_i, i = 0, ..., N$, j-й квант определяет полуоткрытый интервал $[t_{j-1}; t_j)$. В общем случае неважно, находится закрытая граница '[' справа или слева. В данной работе будем полагать, что такая граница всегда будет располагаться слева. Исключение могут составлять первый и последний кванты. Если множество X не ограничено слева, то первый квант $(-\infty, t_1)$ будет открытым. Если множество X ограничено справа значением t_N, то последний квант $[t_{N-1}, t_N]$ будет закрытым.

3. Аппроксимирующие значения $y_i \in [t_{(i-1)}; t_i), i = 1, ..., N$.

Сама процедура скалярного квантования сводится к замене вещественного числа x на аппроксимирующее значение кванта, в который оно попадает:

$$x \in [t_{(i-1)}, t_i) \longmapsto Q(x) = y_i.$$

Заметьте, что y_i также является вещественным числом.

Формулу (2.3) среднего квадрата ошибки можно переписать следующим образом:

$$\varepsilon_X^2 = \sum_{i=1}^{N} \int_{t_{(i-1)}}^{t_i} (x - y_i)^2 f(x) dx. \tag{2.5}$$

Процедуру отображения при скалярном квантовании часто изображают графически в виде ступенчатой функции в декартовой системе координат, как это показано на рис. 2.2,б.

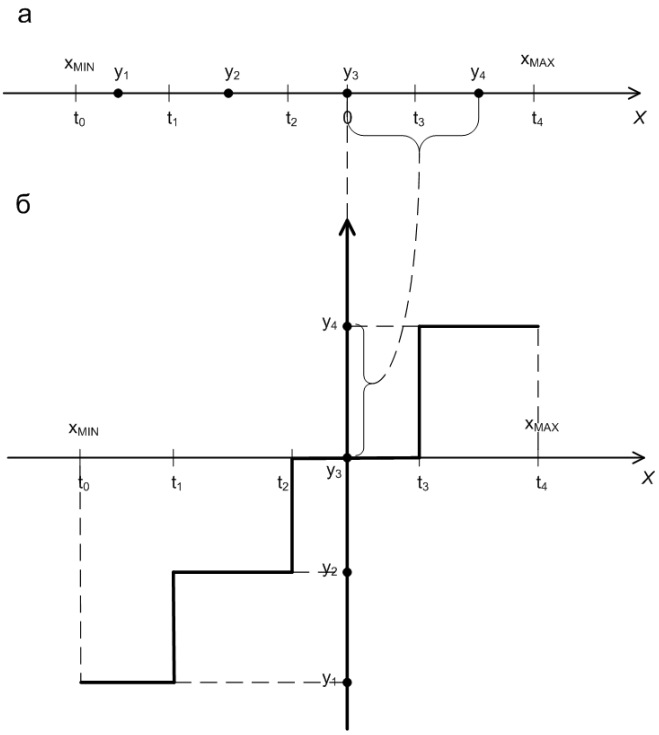

Рис. 2.2. Пример скалярного квантования $N = 4$: а — Пример разбиения числовой оси; б — ступенчатая функция.

Здесь на оси аргумента отложены элементы исходного множества X, а по оси ординат — значения аппроксимирующего множества Y. Если множество X является неограниченным, то $t_0 = -\infty$, а $t_N = \infty$. Для ограниченного множества X границы крайних квантов и определяются предельными элементами множества X.

В зависимости от положения нулевого значения различают два вида скалярного квантования:

1. нулевое значение расположено на границе одного из квантов (рис.

2.3,а). В англоязычной литературе этот вид квантования называют *midrise*. При этом ноль не является аппроксимирующим значением;

2. нулевое значение расположено внутри одного из квантов, часто в его середине (рис. 2.3,б). В англоязычной литературе этот вид квантования называют *midtread*. При этом ноль, как правило, является аппроксимирующим значением. Примером такого вида скалярного квантования является арифметическое округление.

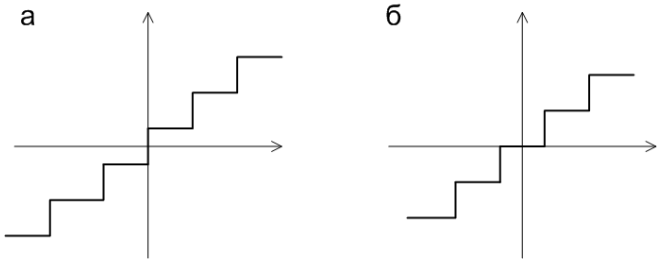

Рис. 2.3. Виды скалярного квантования: а — midrise – ноль на границе кванта; б — midtread – ноль внутри кванта

Если все кванты, кроме крайних для случая, когда множество X неограничено, имеют одинаковый размер, то квантование называют *равномерным* (в англоязычной литературе *uniform scalar quantization*). В другом случае квантование называется *неравномерным*.

Пример 2. Равномерное скалярное квантование случайной величины, распределенной по равномерному закону на интервале $[-3, 3]$**.**
Пусть равномерному квантованию подвергаются отсчеты $x_t \sim U[-3, 3]$, распределенные по равномерному закону на интервале $[-3, 3]$, и число квантов равно 8. В случае равномерного квантования $\Delta = \frac{3}{4}$, $t_i = -3 + i \cdot \Delta$ и $y_i = \frac{t_{(i-1)}+t_i}{2}$. Ниже представлена таблица с полным описанием схемы квантования.

i	$[t_{(i-1)}; t_i)$	y_i	код
1	$[-3; -2.25)$	-2.625	000
2	$[-2.25; -1.5)$	-2	001
3	$[-1.5; -0.75)$	-1.125	010
4	$[-0.75; 0)$	-0.375	011
5	$[0; 0.75)$	0.375	100
6	$[0.75; 1.5)$	1.125	101
7	$[1.5; 2.25)$	2	110
8	$[2.25; 3]$	2.625	111

Рассмотренный пример относится к равномерному скалярному квантованию вида *midrise*.

1.3.2. Анализ эффективности равномерного скалярного квантования

Равномерное скалярное квантование является популярным в силу простоты его реализации. Все кванты за исключением крайних одинаковы и равны некоторой величине Δ. Для того чтобы определить равномерное скалярное квантование, необходимо задать только число квантов N. Границы квантов рассчитываются, исходя из области значений случайной величины X. Аппроксимирующие значения выбираются в середине квантов.

Пример 3. Округление вещественных чисел.
Наиболее известным примером равномерного скалярного квантователя является *округление* вещественных чисел $r \in \Re$ до целых $z \in Z$. При округлении $X \sim \Re, Y \sim Z$ и $\Delta = 1$. Границы квантов совпадают со значениями $[z - 0.5, z + 0.5)$, где $z \in Z$. Особенностью данного примера является то, что при округлении элементы из непрерывного множества преобразуются в элементы из счетного множества. Сама процедура квантования может быть представлена в виде арифметического выражения:

$$y = \lfloor x + 0.5 \rfloor,$$

где $\lfloor . \rfloor$ — операция отбрасывания дробной части, реализуемая на языке C с помощью функции `floor()`.

Размер кванта Δ выбирается в соответствии с количеством квантов N и диапазоном значений, принимаемых случайной величиной X. Наиболее простую зависимость ошибки квантования от количества квантов N можно продемонстрировать для случая (2.4), когда данные распределены по равномерному закону $U[a, b]$. Равномерное распределение задается двумя параметрами — минимальным a и максимальным b значениями, которые может принимать случайная величина X.

В качестве меры вносимых при квантовании искажений будем использовать отношение «сигнал-шум», задаваемое формулой (2.4). Дисперсия равномерно распределенной случайной величины равна:

$$\sigma_X^2 = \frac{(b-a)^2}{12} = \frac{(2^R \Delta)^2}{12}. \tag{2.6}$$

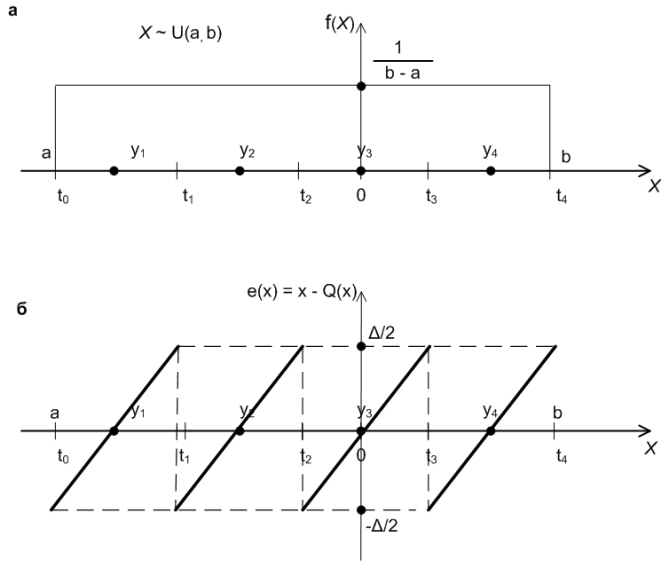

Рис. 2.4. Ошибка квантования случайной величины $X \sim U[a,b]$

Поскольку при квантовании равномерно распределенной случайной величины вероятности ошибок квантования внутри каждого кванта Δ_i будут одинаковые, то, подставляя функцию плотности равномерного распределения в формулу (2.5), получим средний квадрат ошибки вида

$$\varepsilon_X^2 = \sum_{i=1}^{N} \int\limits_{t_{(i-1)}}^{t_i} (x-y_i)^2 \tfrac{1}{N\Delta} dx = \tfrac{1}{N\Delta} \sum_{i=1}^{N} \int\limits_{y_i-\frac{\Delta}{2}}^{y_i+\frac{\Delta}{2}} (x-y_i)^2 dx = \\ = \tfrac{1}{\Delta} \int\limits_{y_i-\frac{\Delta}{2}}^{y_i+\frac{\Delta}{2}} (x-y_i)^2 dx. \tag{2.7}$$

Ошибка внутри кванта с номером i имеет вид функции $e(x) = x - y_i$, т.е. прямой пересекающей ось абсцисс в точке, равной аппроксимирующему значению y_i этого кванта, как показано на рис.2.4,б. Максимальное по модулю значение ошибки не превышает значения $\frac{\Delta}{2}$. Выполним подстановку $x - y_i = z$, в результате которой выражение (2.7) примет вид

$$\varepsilon_X^2 = \frac{1}{\Delta} \int\limits_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} z^2 dz. \tag{2.8}$$

Таким образом, вычислив интеграл (2.7), получим

$$\varepsilon_X^2 = \frac{\Delta^2}{12}. \tag{2.9}$$

Подставляя формулы (2.6) и (2.9) в (2.4), получим окончательную функциональную зависимость отношения «сигнал-шум» от логарифма (по основанию два) числа квантов R, которая имеет линейный характер:

$$SNR(R) = 10 \lg \left((2^R)^2\right) = 20R \lg 2 \simeq 6.02R \text{ dB}. \tag{2.10}$$

Для случайных величин, распределенных не по равномерному закону, значение среднеквадратической ошибки будет отличаться от приведенной в формуле (2.9). Покажем, что при большом числе квантов для произвольного распределения ошибку равномерного квантование также можно оценивать с помощью формулы (2.10). Для этого перепишем формулу (2.5) в форме условного математического ожидания:

$$\varepsilon_X^2 = \sum_{i=1}^{N} M\left[e^2(x) \mid x \in [t_{(i-1)}; t_i)\right] \Pr\left\{x \in [t_{(i-1)}; t_i)\right\}. \tag{2.11}$$

При этом

$$\sum_{i=1}^{N} \Pr\left\{x \in [t_{(i-1)}; t_i)\right\} = 1 \tag{2.12}$$

и условное математическое ожидание квадрата ошибки квантования случайной величины X при условии, что ее значение попадает в i-й квант, определяемый на интервале $[t_{(i-1)}; t_i)$, равно:

$$M\left[e^2(x) \mid x \in [t_{(i-1)}; t_i)\right] = \int_{t_{(i-1)}}^{t_i} (x - y_i)^2 f\left(x \mid \left\{x \in [t_{(i-1)}; t_i)\right\}\right) dx. \tag{2.13}$$

Условная плотность вероятности $f\left(x \mid \left\{x \in [t_{(i-1)}; t_i)\right\}\right)$, представленная в формуле (2.13), определяется отношением плотности вероятности $f(x)$ к вероятности попадания непрерывной случайной величины X в этот интервал. Таким образом

$$M\left[e^2(x) \mid x \in [t_{(i-1)}; t_i)\right] = \int_{t_{(i-1)}}^{t_i} (x - y_i)^2 \frac{f(x)}{\int\limits_{t_{(i-1)}}^{t_i} f(y)dy} dx. \tag{2.14}$$

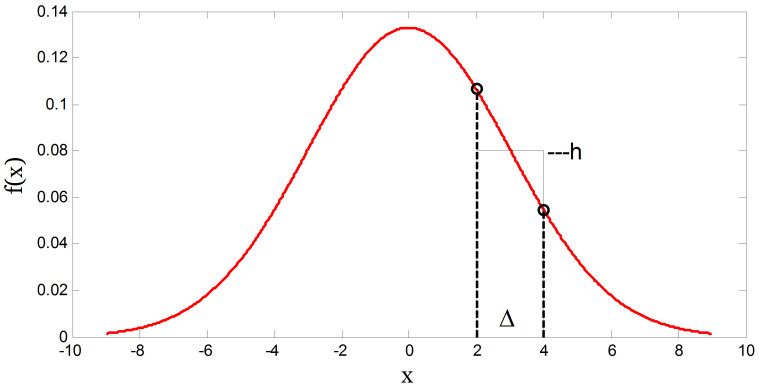

Рис. 2.5. Приближенное вычисление вероятности попадания в квант $[t_{(i-1)}; t_i)$ при равномерном скалярном квантовании

Для равномерного распределения $f(x) = \frac{1}{N\Delta}$ при любом допустимом значении x, а $\int\limits_{t_{(i-1)}}^{t_i} f(y)dy = \frac{1}{N}$. Подставив соответствующие значения в (2.14) и (2.11), придем к результату, полученному в формуле (2.7).

Для распределения, отличного от равномерного, значение $f(x)$ на любом из интервалов не будет постоянным. Для того, чтобы оценить ε_X^2 для произвольного распределения, построим на интервале $[t_{(i-1)}; t_i)$ прямоугольник со сторонами Δ и h (рис. 2.5) таким образом, чтобы выполнялось условие

$$\int\limits_{t_{(i-1)}}^{t_i} f(y)dy = h_i\Delta.$$

При этом на интервале $[t_{(i-1)}; t_i)$ для распределения, отличного от равномерного, $f(x)$ может не совпадать с h_i. Однако при большом числе квантов интервал Δ становится малым и можно считать, что значение $f(x)$ на всем интервале постоянно, т.е. $f(x) \approx h_i$. В итоге выражение

(2.11) сводится к

$$\varepsilon_X^2 \approx \sum_{i=1}^{N} \left[\int_{t_{(i-1)}}^{t_i} (x - y_i)^2 \tfrac{1}{\Delta} dx \right] \Pr\left\{ x \in [t_{(i-1)}; t_i) \right\} = \\ = \tfrac{\Delta^2}{12} \sum_{i=1}^{N} \Pr\left\{ x \in [t_{(i-1)}; t_i) \right\} = \tfrac{\Delta^2}{12}. \tag{2.15}$$

Таким образом, из совпадения (2.9) и (2.15) следует, что формулу (2.10) также можно использовать для приближенной оценки ошибки равномерного квантования значений случайной величины с произвольным распределением. Точность этой оценки необходимо будет проанализировать в задании 1d в ходе выполнения данного исследования.

1.3.3. Неравномерное скалярное квантование. Алгоритм Макса-Ллойда

Равномерное квантование не является эффективным для случая, когда плотность распределения случайной величины отлична от равномерного закона. При минимизации среднеквадратической ошибки ε_X^2 размеры квантов уже не будут одинаковыми, и оптимальные аппроксимирующие значения не обязательно будут располагаться в середине кванта. На рис. 2.6 этот факт поясняется на качественном уровне. Для минимизации ошибок размеры квантов в области наиболее частых значений должны быть меньше, т.е. скалярное квантование становится *неравномерным*.

Для формирования параметров оптимального неравномерного квантования сформулируем следующую оптимизационную задачу. Пусть задана функция плотности вероятности $f(x)$ непрерывной случайной величины X и количество квантов N. Требуется выбрать значения границ квантов $\{t_i\}$ и аппроксимирующих значений $\{y_i\}$, минимизирующие целевую функцию, которой является средний квадрат ошибки квантования (2.5). Результатом решения задачи минимизации целевой функции (2.5) является одновременное выполнение двух следующих условий:

$$t_i = \frac{y_{(i+1)} + y_i}{2}; \tag{2.16}$$

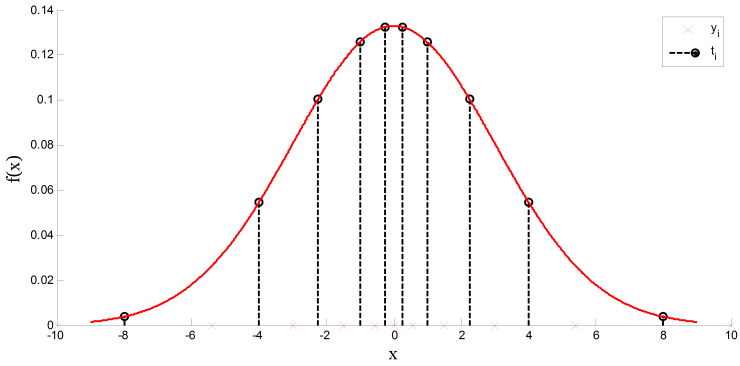

Рис. 2.6. Пример неравномерного скалярного квантования

$$y_i = \frac{\int\limits_{t_{(i-1)}}^{ti} xf(x)\,\mathrm{d}x}{\int\limits_{t_{(i-1)}}^{ti} f(x)\,\mathrm{d}x}. \tag{2.17}$$

На практике функция плотности вероятности $f(x)$ неизвестна, и для формирования параметров оптимального неравномерного квантования используют выборку случайной величины X. Итеративный алгоритм, который на основе выборки находит приближенное значение $\{t_i\}$ и $\{y_i\}$ в соответствии с формулами (2.16) и (2.17) был сформулирован Ллойдом в 1956 г. и опубликован Максом [6] в 1960 г. (работа Ллойда была опубликована в 1982 [7]). Алгоритм состоит из следующих шагов.

Входные данные:

- выборка $\{x_1, x_2, ..., x_M\}$;
- число квантов N, причем $M \gg N$.

Вспомогательные переменные:

- k - номер текущей итерации алгоритма;
- $t_i^{(k)}$ — пороговые значения квантователя на итерации k;
- $y_i^{(k)}$ — аппроксимирующие значения квантователя на итерации k;
- $|Q_i^{(k)}(x)|$ — количество элементов из исходной выборки $\{x\}$, которые попадают в i-й квант на итерации k;
- T_{ε^2} — предустановленный порог для ошибки квантования;

• Δ_{ε^2} — предустановленный порог для разности ошибок квантования на текущей (k) и предыдущей $(k-1)$ итерациях;

• T_k — предустановленный порог для максимального числа итераций.

Инициализация:

• $k = 1$.

• Определение начальных значений $t_i^{(1)}$. Начальные значения можно задать произвольным образом, но так, чтобы выполнялось два условия:

$$t_{(i-1)}^{(1)} < t_i^{(1)}; \tag{2.18}$$

$$|Q_i^{(1)}(x)| \neq 0, \quad \forall i = 1, \cdots, N. \tag{2.19}$$

Последнее условие означает, что в каждый квант должно попасть как минимум одно значение из исходной выборки $\{x_j\}$. Наиболее простой способ — использовать для инициализации $\{t_i^{(1)}\}$ значения границ равномерного квантователя:

$$t_0^{(1)} = \min_j\{x_j\};$$

$$t_N^{(1)} = \max_j\{x_j\};$$

$$t_i^{(1)} = t_0^{(1)} + i\frac{t_N^{(1)} - t_0^{(1)}}{N}.$$

После этого надо убедиться, что условие (2.19) выполняется . В противном случае необходимо изменить ранее сформированное множество $\{t_i^{(1)}\}$.

Шаг 1. Для каждого кванта i, определяемого интервалом $[t_{(i-1)}^{(k)}; t_i^{(k)})$ вычисляется среднее значение среди элементов выборки, попавших в этот интервал. Вычисленное среднее присваивается аппроксимирующему значению $y_i^{(k)}$:

$$y_i^{(k)} = \frac{1}{|Q_i^{(k)}(x)|} \sum_{x_j \in \left[t_{(i-1)}^{(k)}; t_i^{(k)}\right)} x_j. \tag{2.20}$$

При работе с выборкой формула (2.20) соответствует формуле (2.17).

Шаг 2. Вычисление ошибки квантования $\varepsilon_X^2(k)$ на текущей итерации k в соответствии с текущими аппроксимирующими значениями $y_i^{(k)}$

и границами $t_i^{(k)}$:

$$\varepsilon_X^2(k) = \frac{1}{M}\sum_{j=1}^{M}\left(x_j - Q(x_j)\right)^2; \tag{2.21}$$

$$Q(x_j) = y_s^{(k)} : x_j \in [t_{s-1}^{(k)}, t_s^{(k)}). \tag{2.22}$$

При работе с выборкой формула (2.21) соответствует (2.3). Напомним, что $Q(x_j)$ определяет правило выбора аппроксимирующего значения по кванту, в который попадает x_j.

Шаг 3. Анализ условий останова алгоритма:

1. $\varepsilon_X^2(k) < T_{\varepsilon^2}$;
2. $\varepsilon_X^2(k) - \varepsilon_X^2(k-1) < \Delta_{\varepsilon^2}$;
3. $k == T_k$.

Если одно из вышеперечисленных условий выполняется, то алгоритм завершает свою работу.

Шаг 4. Установка значений границ $t_i^{(k)}$ в соответствии с формулой (2.16)

Шаг 5. $k = k + 1$. Переход к шагу 1.

Строго говоря, алгоритм никогда не достигает оптимального значения, так как одно из условий (2.16) или (2.17) остается невыполненным после остановки. В оригинальном описании алгоритма [7] проверка на выполнение условия останова (шаги 2 и 3) выполняется после каждой модификации: аппроксимирующих значений $y_i^{(k)}$ (шаг 1) и граничных значений $t_i^{(k)}$ (шаг 4). Изложенный вариант с одной проверкой останова обусловлен практическими соображениями ускорения процедуры. К тому же такой вариант позволяет не хранить границы квантов $\{t_i\}$, так как номер кванта может быть определен по ближайшему к x аппроксимирующему значению y_i.

1.4. Векторное квантование. Алгоритм Линде-Бузо-Грея

Алгоритм построения оптимального векторного квантователя является обобщением скалярного случая. Поэтому его иногда еще называют *обобщенным алгоритмом Макса-Ллойда*. Аппроксимирущие векторы $\bar{y}$ и квантуемые векторы $\bar{x}$ можно интерпретировать, как точки в n-мерном пространстве, где n — длина векторов. Чтобы определить схему векторного квантования, необходимо сформировать множество из

N аппроксимирующих векторов $\{\bar{y}\}$, которое называют *кодовой книгой*. Процедура квантования $\bar{y} = Q(\bar{x})$ заключается в выборе для $\bar{x}$ наиболее близкого по заранее определенной метрике вектора $\bar{y}$ из кодовой книги. Границы между квантами определены условно, исходя из выбранной метрики. В качестве метрики будем вновь использовать евклидово расстояние. Геометрической интерпретацией векторного квантования может служить *диаграмма Вороного*, пример которой приведен на рис. 2.7.

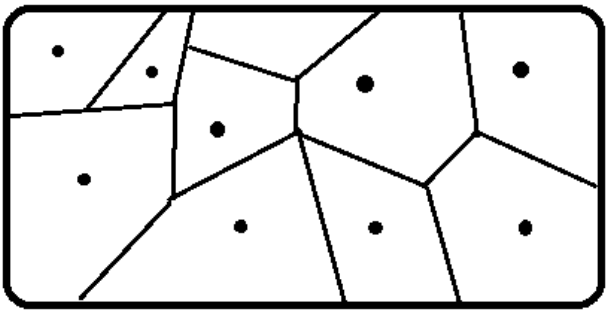

Рис. 2.7. Пример диаграммы Вороного

Пример 4. Палитра BMP-файла как кодовая книга.

Рассмотрим в качестве примера кодовой книги палитру в формате изображений BMP8. Для представления цвета пикселя также используется стандартная тройка базовых цветов (R, G, B), однако тройки выбираются не произвольно, как в формате BMP24, а из набора 256 вариантов, определенных в палитре, которая располагается между заголовком и данными. Поскольку выбор осуществляется из 2^8 вариантов, для представления пикселя необходим один байт. Более подробное описание работы с палитрой в формате BMP8 будет представлено в подразделе 1.5.

Обобщение метода Ллойда-Макса в англоязычной литературе также известно, как алгоритм Линде-Бузо-Грея [8], [3]. Алгоритм состоит из следующих шагов.

Входные данные:

- выборка векторов $\{\bar{x}_1, \bar{x}_2, ..., \bar{x}_M\}$; векторы имеют длину n;
- размер кодовой книги N, причем $M \gg N$.

Вспомогательные переменные:

• k - номер текущей итерации алгоритма;

• $\bar{y}_i^{(k)}, \quad i = 1, \cdots, N$ — аппроксимирующие векторы кодовой книги на итерации k;

• $S_i, \quad i = 1, \cdots, N$ — набор счетчиков, фиксирующих количество векторов исходной выборки, для которых вектор $\bar{y}_i^{(k)}$ будет являться аппроксимирующим на текущей итерации. По смыслу S_i аналогично $|Q_i^{(k)}(x)|$ в алгоритме Ллойда-Макса.

• $\bar{v}_i, \quad i = 1, \cdots, N$ — набор из векторов-счетчиков для накопления суммы квантуемых векторов, отнесенных к i-му кванту;

• T_{ε^2} — предустановленный порог для ошибки квантования;

• Δ_{ε^2} — предустановленный порог для разности ошибок квантования на текущей (k) и предыдущей $(k-1)$ итерациях;

• T_k — предустановленный порог для максимального числа итераций.

Инициализация:

• $k = 1$.

• Формируем кодовую книгу $\bar{y}^{(1)}$ из N векторов исходной выборки произвольным образом. Формирование кодовой книги на основе первых N векторов исходной выборки не рекомендуется в силу высокой корреляции между соседними векторами в выборке. Это особенно характерно для изображений.

Шаг 1. Вычисление ошибки квантования $\varepsilon_X^2(k)$ на текущей итерации k с использованием кодовой книги $\{\bar{y}^{(k)}\}$:

$$\varepsilon_X^2(k) = \frac{1}{M} \sum_{j=1}^{M} \|\bar{x}_j - Q(\bar{x}_j)\|^2, \tag{2.23}$$

$$Q(\bar{x}_j) = \bar{y}_s^{(k)} : s = \underset{i}{argmin}\{\|\bar{x}_j - \bar{y}_i^{(k)}\|^2\}. \tag{2.24}$$

При работе с выборкой формула (2.23) соответствует (2.3).

Шаг 2. Анализ условий останова алгоритма:

1. $\varepsilon_X^2(k) < T_{\varepsilon^2}$;
2. $\varepsilon_X^2(k) - \varepsilon_X^2(k-1) < \Delta_{\varepsilon^2}$;
3. $k == T_k$.

Если одно из вышеперечисленных условий выполняется, то алгоритм завершает свою работу.

Шаг 3. Обнуление всех счетчиков S_i и векторов-счетчиков $\bar{v}_i$:

$$\bar{v}_i = \bar{0}, \quad i = 1, \cdots, N;$$

$$S_i = 0, \quad i = 1, \cdots, N.$$

Шаг 4. Для каждого вектора выборки $\bar{x}_j, j = 1, \cdots, N$ выполняются следующие действия.

• Определяется наиболее близкий по евклидову расстоянию аппроксимирующий вектор $\bar{y}_i^{(k)}$.

$$t = \underset{i}{argmin}\{\|\bar{x_j} - \bar{y}_i^{(k)}\|^2\},$$

где $\|\bar{x_j} - \bar{y}_i^{(k)}\|$ вычисляется по формуле (2.2).

• Модификация вектора $\bar{v}_t$:

$$\bar{v}_t = \bar{v}_t + \bar{x}_j.$$

Сложение векторов выполняется по правилам линейной алгебры (поэлементно).

• Наращивание счетчика S_t:

$$S_t = S_t + 1.$$

Шаг 5. Формирование новой кодовой книги, которая будет использоваться на следующей итерации $k + 1$:

$$\bar{y}_i^{(k+1)} = \frac{\bar{v}_i}{S_i}, \quad i = 1, \cdots, N.$$

Деление вектора $\bar{v}_i$ на скаляр осуществляется по правилам линейной алгебры (поэлементно).

Шаг 6. $k = k + 1$. Переход к шагу 1.

Границы квантов можно формально получить по построенной кодовой книге, определив множество векторов $\{\bar{t}_{i,j}\}$, которые равноудалены от аппроксимирующих векторов в $\bar{y}_i$ и $\bar{y}_j$. Для этого воспользуемся следующим равенством:

$$\|\bar{t}_{i,j} - \bar{y}_i\|^2 = \|\bar{t}_{i,j} - \bar{y}_j\|^2.$$

Совокупность векторов $\{\bar{t}_{i,j}\}$ для всех возможных значений пар i и j формирует так называемые *многогранники Вороного*.

1.5. Использование палитры в формате BMP

С помощью формата BMP24 можно представить до $2^{24} \sim 16$ млн. различных цветовых оттенков одного пикселя. Однако далеко не все из них используются в одном изображении. Для сокращения размера файла ранее в формате BMP активно использовался способ *ссылочного* представления значения пикселя вместо *непосредственного*, как для 24-битной версии. Для организации такого представления используется специальная таблица, которую принято называть *палитрой* (*palette*).

В файле BMP палитра хранится после заголовочной структуры `BITMAPINFOHEADER`. Палитра представляет собой логическую таблицу, строки которой последовательно записаны в файле с помощью структур типа `RGBQUAD`. Структура `RGBQUAD` объявляется следующим образом:

```
typedef struct tagRGBQUAD {
    BYTE rgbBlue;
    BYTE rgbGreen;
    BYTE rgbRed;
    BYTE rgbReserved;
} RGBQUAD;
```

В первых трех полях хранится комбинация значений синей (`rgbBlue`), зеленой (`rgbGreen`) и красной (`rgbRed`) компонент. Значение поля `rgbReserved` не используется. Таким образом, строка таблицы описывает цветовой оттенок, используя 24 бита. Значение пикселя в разделе данных интерпретируется как ссылка на строку в таблице палитры.

Количество экземпляров структуры `RGBQUAD`, представленных в разделе палитры, определяется в переменной `biClrUsed` структуры `BITMAPINFOHEADER`. Очевидно, что при затрате k бит на представление одного пикселя, размер палитры следует выбирать 2^k для обеспечения максимально возможного числа используемых цветовых оттенков в этом файле. Чтобы обеспечить такой способ представления данных, следует установить значение поля `biClrUsed` в ноль. В этом случае размер палитры будет определяться полем `biBitCount` структуры `BITMAPINFOHEADER`, которое в случае использования палитры может принимать значения 1, 2, 4 или 8.

2. Порядок выполнения исследования

1. Проверка точности оценки *SNR*, получаемой по формуле (2.10):

a) сформировать выборку объемом $M = 10000$ равномерно распределенной случайной величины $X \sim U[-a, a]$. Значение параметра a определяет преподаватель.

b) применить к выборке $\{x_1, ..., x_M\}$ равномерное квантование с числом квантов 2^R, где $R = 1, ..., 7$.

c) По результатам эксперимента построить график $SNR(R)$ и сравнить его с теоретическим, полученным по формуле (2.10).

d) выполнить аналогичные действия для выборки случайной величины, распределенной по нормальному закону $N(0, \sigma^2)$, где $\sigma = \frac{a}{3}$. Примечания по заданию:

- интервал квантования установить $[-a; a]$ (как и для случайной величины X),
- $t_0 = -\infty$, $t_{N-1} = +\infty$ (N — число квантов).

e) провести сравнение построенных теоретического и экспериментальных графиков, сделать выводы о возможности использования теоретического расчета для экспериментальных выборок данных;

f) выполнить равномерное квантование с числом квантов 2^R, где $R = 1, ..., 7$ для трех выборок, сформированных на основе яркостной Y и цветоразностных компонент C_b, C_r (компоненты вычислить по изображению в формате BMP24, выбранному самостоятельно). Построить график $SNR(R)$ по результатам выполненного квантования.

2. Анализ скалярного квантования включает в себя следующие действия:

a) получить у преподавателя три изображения в формате RGB24, различающиеся по яркости:

- с преобладанием светлых тонов («засвеченное» изображение),
- сбалансированное по яркости изображение,
- затемненное изображение.

b) для каждого получить яркостную составляющую Y, воспользовавшись формулой (1.8). Все дальнейшие действия проводить только с Y компонентой;

c) для каждого тестового изображения построить гистограмму яркостной компоненты Y;

d) выполнить равномерное скалярное квантование исходных данных на 2^R уровней, где $R = 1, ..., 7$;

e) для каждого значения R вычислить $PSNR$, воспользовавшись формулой (1.6) и оценить энтропию H по формуле (1.7);

f) по вычисленным значениям для равномерного квантования построить графики $PSNR(R)$ и $PSNR(H)$;

g) реализовать алгоритм Макса-Ллойда для оптимального неравномерного скалярного квантования на 2^R уровней. Применить алгоритм к каждому изображению;

h) выполнить неравномерное скалярное квантование исходных данных на 2^R уровней, где $R = 1, ..., 7$;

i) для каждого значения R вычислить $PSNR$, воспользовавшись формулой (1.6) и оценить энтропию H по формуле (1.7);

j) по вычисленным значениям для неравномерного квантования построить графики $PSNR(R)$ и $PSNR(H)$.

k) построить график ступенчатой функции $Q(x)$ по параметрам оптимального неравномерного квантования для каждого изображения. Число квантов определяет преподаватель;

l) провести сравнительный анализ графиков, полученных с помощью неравномерного и равномерного скалярного квантования.

3. Создание палитры изображения с помощью методов векторного квантования:

a) используя метод Линде-Бузо-Грея, составить палитру по данным компонент R, G и B исходного изображения в формате BMP24;

b) сформировать выходное изображение в формате BMP8 с палитрой;

c) оценить ошибку восстановления входного изображения. В качестве критерия эффективности необходимо использовать $PSNR$;

d) проанализировать влияние округления на результат формирования палитры (точность аппроксимации, число итераций). На шаге 5 алгоритма Линде-Бузо-Грея значения векторов кодовой книги, соответствующей палитре, перестают быть целочисленными. Округление элементов палитры можно осуществлять в

процессе шага 5 (что приведет к увеличению ошибки квантования) или по окончании работы алгоритма. В задании требуется проанализировать насколько возрастет ошибка квантования для первого варианта.

3. Требования к оформлению отчета

Отчет должен содержать следующие пункты:

1. постановка задачи квантования дискретного сигнала;

2. описание изученных алгоритмов скалярного и векторного квантования;

3. описание реализации алгоритмов;

4. результаты (графики, численные расчеты) в соответствии с выполненными заданиями;

5. выводы по результатам сравнительного анализа различных методов квантования;

6. листинги программ.

4. Вопросы для самопроверки

1. Рассмотрим алгоритм Линде-Бузо-Грея для векторов длины 1. Что общего будет у алгоритма с методом Макса-Ллойда в этом случае и чем они будут различаться?

2. Пусть X распределено по нормальному закону $N(m, \sigma^2)$. Каковы будут границы квантов и аппроксимирующие значения, если при неравномерном квантовании используется два кванта?

3. Пусть двумерная случайная величина равномерно распределена внутри квадрата размером 2×2. Каким образом будут выглядеть кодовые книги, состоящие соответственно из двух и четырех векторов?

4. Рассмотрим равномерное скалярное квантование выборки из равномерного распределения. Как изменится отношение сигнал-шум при увеличении числа квантов в два раза?

Библиографический список

1. *Красильников Н.Н.* Цифровая обработка изображений. М.: Вузовская книга,2001.
2. *Сэломон Д.* Сжатие данных, изображений и звука. М.: Техносфера, 2004.
3. *Кудряшов Б.Д.* Теория информации. СПб.: Питер, 2009.
4. *Шапиро Л., Стокман Дж.* Компьютерное зрение. М.: Бином, 2006.
5. *Гмурман В.Е.* Теория вероятностей и математическая статистика. М.: Высшая школа, 2003.
6. *Max J.* Quantizing for minimum distortion // *IEEE Transactions on Information Theory.* 1960. Vol. 6, № 1.
7. *Lloyd S.P.* Least squares quantization in PCM // *IEEE Transactions on Information Theory.* 1982. Vol. 28, № 2.
8. *Linde Y., Buzo A., Gray R.M.* An algorithm for vector quantizer design // *IEEE Transactions on Communications Technology.* 1980. Vol. 28, № 1.

СОДЕРЖАНИЕ

Учебное издание

Гильмутдинов Марат Равилевич
Тюрликов Андрей Михайлович
Линский Евгений Михайлович

ЦИФРОВАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

СТАТИСТИЧЕСКИЙ АНАЛИЗ И КВАНТОВАНИЕ ВИЗУАЛЬНЫХ ДАННЫХ

Учебное пособие

В авторской редакции

Подписано к печати __.__.__. Формат бумаги 60 × 84 1/16. Бумага офсетная. Печать офсетная. Усл. печ. л. 2,2. Уч.-изд. л. 2,4. Тираж 300 экз.
Заказ №630

Редакционно-издательский центр ГУАП
190000, Санкт-Петербург, Б. Морская ул., 67