МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДНІПРОВСЬКИЙ ДЕРЖАВНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

О.О. Шумейко

КОНСПЕКТ ЛЕКЦІЙ

з дисципліни
«Технології створення мультимедіа застосувань»
за освітньо-професійною програмою
«Інженерія програмного забезпечення»
для здобувачів вищої освіти першого (бакалаврського) рівня
зі спеціальності 121 – «Інженерія програмного забезпечення»

Затверджено редакційно-видавничою
секцією науково методичної ради ДДТУ
протокол № 9 від 11.11.2019 р.

Кам’янське
2019

Конспект лекцій з дисципліни «Технології створення мультимедіа застосувань» для здобувачів вищої освіти першого (бакалаврського) рівня за освітньо-професійною програмою «Інженерія програмного забезпечення» зі спеціальності 121 – «Інженерія програмного забезпечення»/Укладач О.О.Шумейко.– Кам'янське: ДДТУ, 2019– 169 с.

Укладачі: д.т.н., проф. Шумейко О.О.

Рецензент: д.т.н., проф. Самохвалов С.Є.
(Дніпровський державний технічний університет)

Затверджено на засіданні кафедри ПЗС
(протокол № 11 від 11.11.2019 р.)

Коротка анотація видання. Конспект лекцій складено відповідно до освітньо-професійною програмою «Інженерія програмного забезпечення» зі спеціальності 121 – «Інженерія програмного забезпечення»ої для здобувачів вищої освіти першого (бакалаврського) рівня та робочої програми з дисципліни «Технології створення мультимедіа застосувань» і відповідає вимогам модульного засвоєння курсу. Викладено основні теми дисципліни, що присвячені основам розробки мультимедіа додатків, окремі теми супроводжуються прикладами розв'язання типових задач.

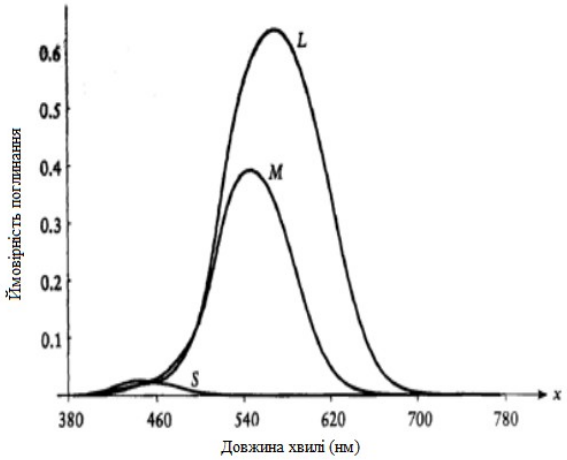
ЗМІСТ

ТЕМА 1. СВІТЛО І КОЛІР

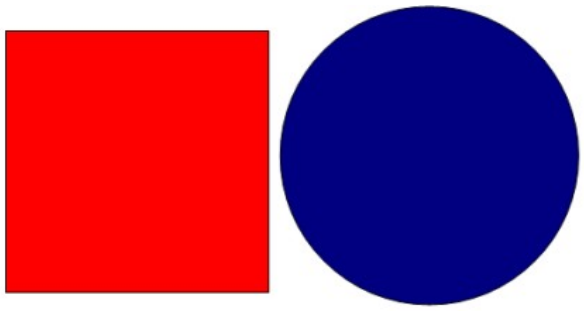
Використання кольорових схем є одним із найбільш важливих елементів комп'ютерної графіки. Цій темі присвячена чудова книга James Gurney "Color and Light. A Guide for the Realist Painter", яка вийшла у видавництві Adrews McMell Publishing, в якій читачі можуть знайти багато цікавого матеріалу на дану тему.

Колір - це сенсорне враження, яке виникає, коли хвильові світлові подразники певної довжини (електромагнітне випромінювання в діапазоні приблизно від 180 до 780 нм, так званий світловий спектр) потрапляють на рецептори зіниці. Звідти по нейронних мережах цей імпульс передається в мозок і починає сприйматися як колір. Сприйняття кольору навколишніх об'єктів завжди суб'єктивно, так як воно виникає тільки в головному мозку. У фізичному сенсі об'єкти не мають кольору, ми всього лише сприймаємо їх такими.

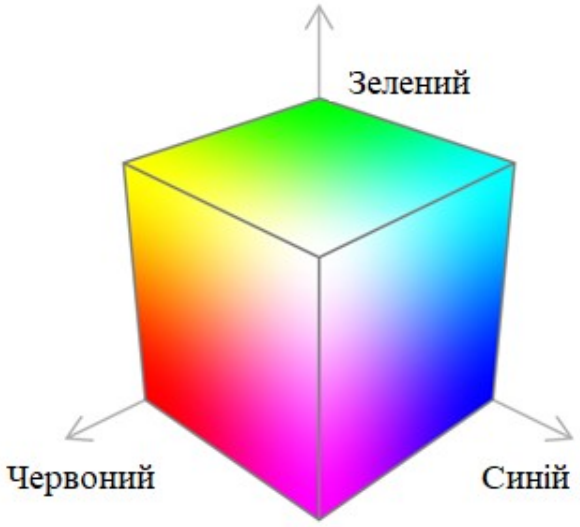
Фізіологічний аспект сприйняття кольору полягає в тому, що в процесі еволюції на задній частині людського ока з'явилися фоторецептори. При впливі електромагнітних хвиль довжиною від 380 до780 нм вони здатні генерувати фізіологічний (нервовий) сигнал. Існує два види фоторецепторів: палички і колбочки. Палички розпізнають тільки контраст чорного і білого, дуже світлочутливі і роблять можливим зір в сутінках та в темноті (зазначимо, що та освітленість, при якій починають функціонувати палички, співпадає зі світловим потоком віддзеркаленого світла Місяця, окрім того, палички представляють собою ідеальний детектор – людина сприймає у повній темряві одиничний фотон!). Колбочки відповідальні за сприйняття кольору. Існує три типи колбочок, які сприймають короткі (синій колір), середні (зелений) і довгі хвилі (червоний колір) колірного випромінювання. Кожне поєднання світлових променів, що падають на сітківку, певним чином збуджує ці три види колбочок і дає відповідне враження про колір. Колбочки кожного виду містять свій особливий пігмент. Три типи колбочок називають або як B (Blue), G (Green) і R (Red), або як S, M і L (Small, Medium, Long). Кількість та розташування різного типу колбочок призводить до того, що кожний колір ми сприймаємо по різному, що відображено на наступному графіку. Як видно, найгірше ми сприймаємо синій колір, зауважте, що здалеку ми синій колір сприймаємо як чорний і взагалі досить погано сприймаємо відтінки синього.



Ця особливість призводить до оптичних ілюзій, так червоні об'єкти сприймаються нами ближче, сині – далі.



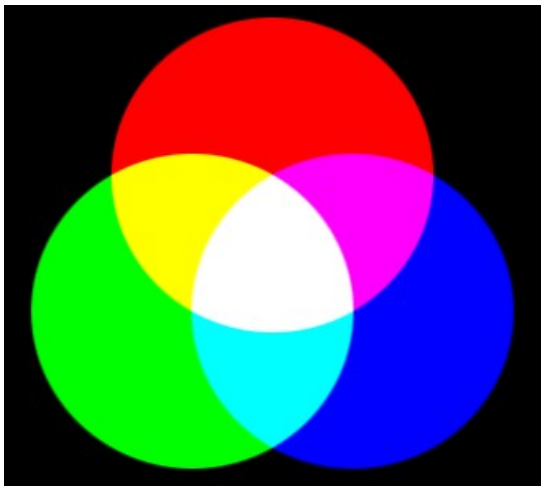
Чисто фізіологічно, так як кожен із базових кольорів (червоний, зелений та синій) сприймається своєю групою світлочутливих клітоk, то ці кольори можна вважати незалежними (ортогональними) і, відповідно, відобразити всю палітру кольорів, яку сприймає людина, у вигляді кубу кольорів. Кожен із цих кольорів є результат змішування базових кольорів. З точки зору комп'ютерної графіки, дана модель реалізована у дисплеї.



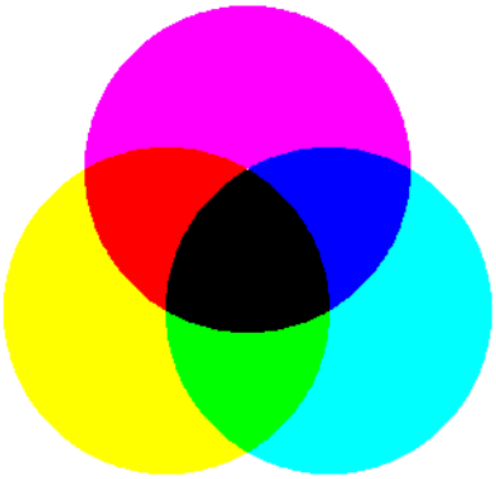
Для принтерів використовується інверсна модель. Так як білий колір представляє собою змішування базових кольорів, і можна вважати лист паперу білим, то в нього вже ніякого кольору додати не можна – там є все! Але можна відняти. Така модель називається CMYK (Cyan, Magenta, Yellow, blacK). У разі, коли базові кольори змінюються в діапазоні від 0 до 1, маємо

$$Cyan = 1 - Red,$$
$$Magenta = 1 - Green,$$
$$Yellow = 1 - Blue.$$

Кольори, які зв'язані відповідним співвідношення, називаються доповнюючими. Зазначимо, що додавання у дану модель чорного пов'язане з тим, що, не зважаючи на те, що чорний колір представляє собою відсутність базових кольорів, папір не є ідеально білим і віднімання R,G.B призводить не до чорного, а до брудного.



Модель RGB



Модель CMYK

CIE XYZ –лінійна три-компонентна модель кольорів, заснована на результатах виміру середньо статистичних характеристик людського зору. Модель була запропонована Міжнародною комісією з освітлення CIE (Commission Internationale de l`Eclairage).

Модель CIE XYZ є базовою моделлю практично всіх інших колірних моделей, що використовуються в різних технічних областях. Колір XYZ задається наступним чином:

$$X = \int_{380}^{780} I(\lambda)\bar{x}(\lambda)d\lambda,\ Y = \int_{380}^{780} I(\lambda)\bar{y}(\lambda)d\lambda,\ Z = \int_{380}^{780} I(\lambda)\bar{z}(\lambda)d\lambda,$$

де $I(\lambda)$ - спектральна щільність будь-якої енергетичної фотометричної величини (наприклад потоку випромінювання, енергетичної яскравості і т.п. в абсолютному або відносному вираженні). Для моделі бралися умови, щоб компонента Y відповідала візуальної яскравості сигналу ($\bar{y}(\lambda)$ - це та сама відносна спектральна світлова ефективність монохроматичного випромінювання для денного зору, яка використовується в усіх світових фотометричних величинах), координата Z відповідала відгуку S («short», короткохвильових, «синіх») колбочок, а координата X була завжди невід'ємною. Криві відгуку нормуються таким чином, щоб площа під усіма трьома кривими була однаковою.

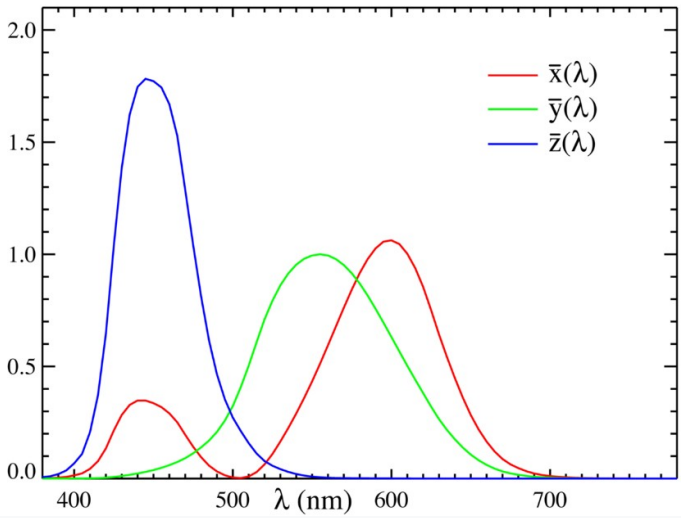
Перетворення RGB у XYZ проводиться наступним чином

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124564 & 0.3575761 & 0.1804375 \\ 0.2126729 & 0.7151522 & 0.0721750 \\ 0.0193339 & 0.11911920 & 0.9503041 \end{bmatrix} \begin{bmatrix} \bar{R} \\ \bar{G} \\ \bar{B} \end{bmatrix},$$

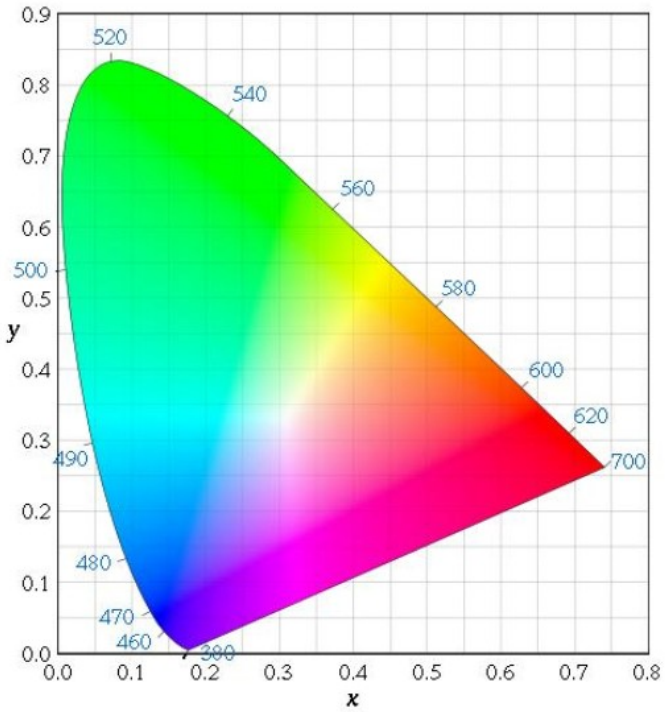
де $\bar{R}$, $\bar{G}$, $\bar{B}$ - нормалізовані значення кольорів, тобто із проміжку [0,1].

Зворотне перетворення

$$\begin{bmatrix} \bar{R} \\ \bar{G} \\ \bar{B} \end{bmatrix} = \begin{bmatrix} 3.2404542 & -1.5371385 & -0.4985314 \\ -0.969266 & 1.8760108 & 0.0415560 \\ 0.0556434 & -0.2040259 & 1.0572252 \end{bmatrix}, \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}.$$



Функції колірної відповідності стандартного колориметричного спостерігача, обчислені комітетом CIE у 1931 році на діапазоні довжин хвиль від 380 до 780 нм (з 5 нм інтервалом).



Дана хроматична діаграма моделі XYZ з довжинами хвиль, де

$$x = \frac{X}{X+Y+Z}, y = \frac{Y}{X+Y+Z}.$$

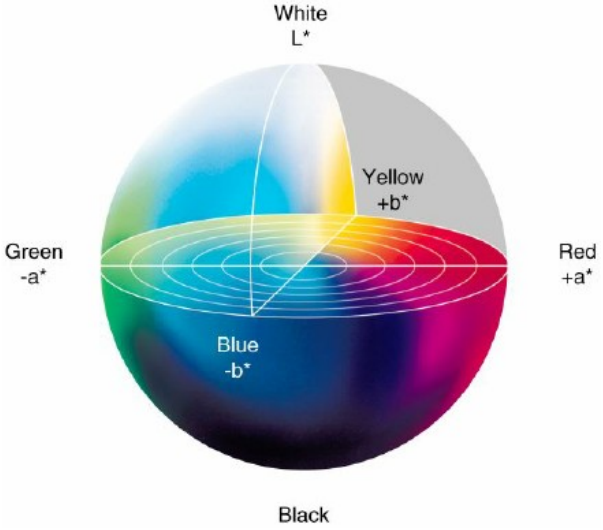
Дана діаграма має наступні властивості:

- на ній наявні всі кольори, які сприймає середньостатистична людина;
- усі кольори, які можуть бути отримані змішуванням двох, лежать на прямій між ними;
- усі кольори, які можуть бути отримані змішуванням трьох кольорів, лежать у трикутнику, з вершинами у точках, що відповідають даним кольорам.

Колірний простір CIELAB

З теоретичної точки зору, проблема з XYZ полягає в тому, що дана модель не є принципово однорідною: зміна одиниці кольору однієї складової, за якою послідовно іде зміна того ж розміру іншої складової, не сприймається людським оком як зміна

рівної кількості кольору. Таким чином, виникла необхідність отримання простору, який є більш сприйнятливо однорідним. І таким простором є CIELAB.
Подібно географічним координатам - довготі, широті і висоті - значення кольору L *, a * та b * дають можливість визначати місцезнаходження кольору і передавати інформацію про нього.



У 1940-х роках Річард Хантер (Richard Hunter) представив модель Lab зі шкалою, організованою таким чином, щоб отримати практично однакову відстань між колірними відмінностями, що приймаються оком. Хоча модель Lab Хантера була прийнята в якості де-факто моделі для знаходження абсолютних колірних координат і відмінностей між кольорами, вона ніколи не була офіційно прийнята в якості міжнародного стандарту. Тридцять один рік по тому CIE опублікувала оновлену версію Lab Хантера: CIELab.
Що означає L * a * b *?
L *: світлота;
a *: червоне / зелене значення;
b *: синє / жовте значення.
Вісь a * йде зліва направо. Рух кольору в напрямку + показує зрушення в бік червоного.
Уздовж вісі b * рух b в сторону + показує зрушення в бік жовтого кольору.
Центральна вісь L * показує L = 0 (чорний колір або повне поглинання) внизу.
У центрі знаходиться нейтральний або сірий колір.
Координати CIELAB знаходяться наступним чином

$$L^* = 116\, f\left(\frac{Y}{Y_n}\right) - 16,\ a^* = 500\left[f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right],\ b^* = 200 - \left[f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right],$$

де

$$f(t) = \begin{cases} t^{1/3}, & t > (6/29)^3, \\ \frac{1}{3}\left(\frac{29}{6}\right)^2 t + \frac{4}{29}, & \text{інакше.} \end{cases}$$

Поділ функції f(t) на два проміжки було зроблено, щоб уникнути точки нескінченної сингулярності при t=0.
Значення X_n, Y_n, Z_n є координатами білого. Для стандартного освітлення D65 з нормалізацією Y = 100 ці нормалізовані значення дорівнюють

$$X_n = 95.0489,\ Y_n = 100,\ Z_n = 108.8840.$$

Значення для підсвічування D65 приблизно відповідає середньому полуденному світлу в Західній Європі (включаючи як пряме сонячне світло, так і світло, яке розсіюється при ясному небі).

Зворотне перетворення має вигляд

$$X = X_n f^{-1}\left(\frac{L^* + 16}{116} + \frac{a^*}{500}\right),\ Y = Y_n f^{-1}\left(\frac{L^* + 16}{116}\right),\ Z = Z_n f^{-1}\left(\frac{L^* + 16}{116} - \frac{b^*}{200}\right),$$

де

$$f^{-1}(t) = \begin{cases} t^3, & t > (6/29)^3, \\ 3\left(\frac{6}{29}\right)^2\left(t - \frac{4}{29}\right), & \text{інакше.} \end{cases}$$

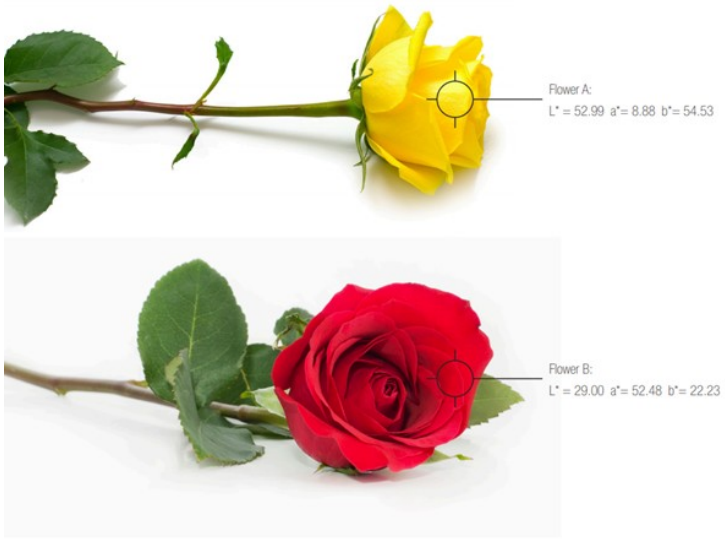
Колірність визначається наступним чином $C^* = \sqrt{(a^*)^2 + (b^*)^2}$, а тональність відповідає значенню

$$H^* = tg^{-1}\frac{b^*}{a^*}.$$

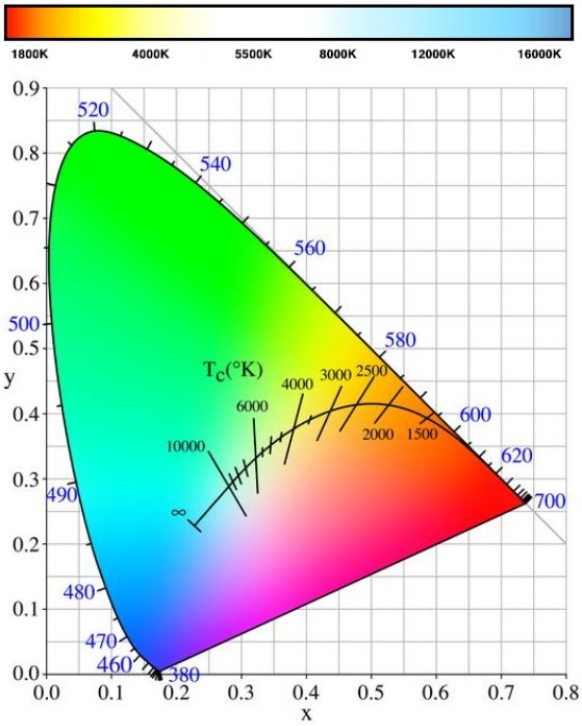
І, відповідно, різниця між кольорами дорівнює

$$\Delta E^* = \sqrt{(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2}\ .$$

Особливості CIELAB проілюстровані на наступному зображенні.



Колірна температура — характеристика розподілу інтенсивності випромінення джерела світла як функція довжини хвилі в оптичному діапазоні, температура абсолютно чорного тіла, при якій воно випускає випромінювання з тією ж хроматичністю, що і дане випромінення. Колірна температура характеризує спектральний склад випромінювання джерела світла. Вимірюється в міредах (M) і Кельвінах (K), «Міред» обернено пропорційний до «Кельвін»: $M = 10^6 K^{-1}$.



HSV (Hue, Saturation, Value-тональність, насиченість, величина). Дана модель була запропонована Alvy Ray Smith, одним із засновників Pixar. Для даної моделі кольорів маємо:

- Hue – тональність кольорів (червоний, зелений, синій і всі проміжні між ними). Змінюється у проміжку 0-360°.
- Saturation – насиченість. Змінюється в межах від 0 до 1. Є характеристикою «чистоти» кольору, чим ближче це значення до 0, тим колір ближче до нейтрального сірого.
- Value –характеризує наповнюваність даного кольору білим. Змінюється від 0 (чорне) до 1 (біле) через всі відтінки сірого.

Якщо $H \in [0,360], S, V, R, G.B \in [0,1], max = max(R, G, B), min = min(R, G, B)$, то

$$H = \begin{cases} 0, & якщо \quad max = min, \\ 60\dfrac{G - B}{max - min}, & якщо \quad max = R, G \geq B, \\ 60\dfrac{G - B}{max - min} + 360 & якщо \quad max = R, G < B, \\ 60\dfrac{B - R}{max - min} + 120, & якщо \quad max = G, \\ 60\dfrac{R - G}{max - min} + 240, & якщо \quad max = B, \end{cases}$$

$$S = \begin{cases} 0, & якщо \quad max = 0, \\ 1 - \dfrac{min}{max}, & інакше, \end{cases}$$

$$V = max.$$

Для зворотного перетворення маємо

$$C = V \times S,$$

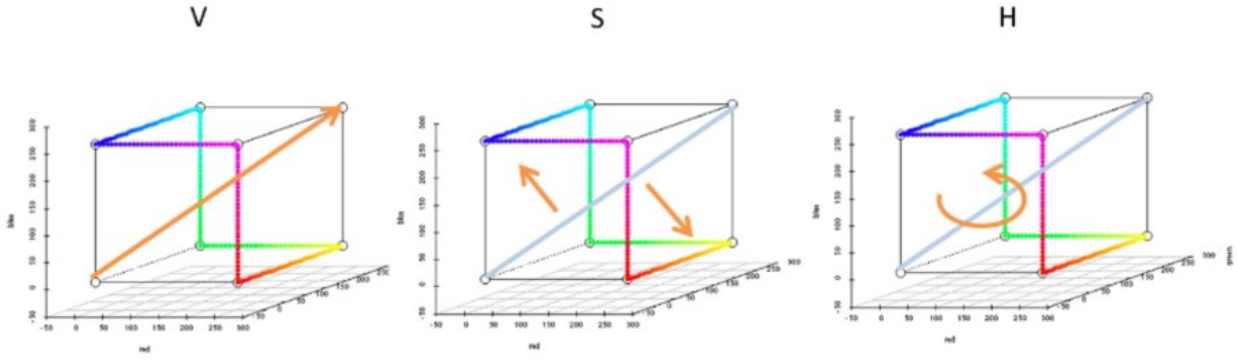
$$X = C \times \left(1 - \left|\left(\frac{H}{60}\right) \bmod 2 - 1\right|\right),$$

$$m = V - C,$$

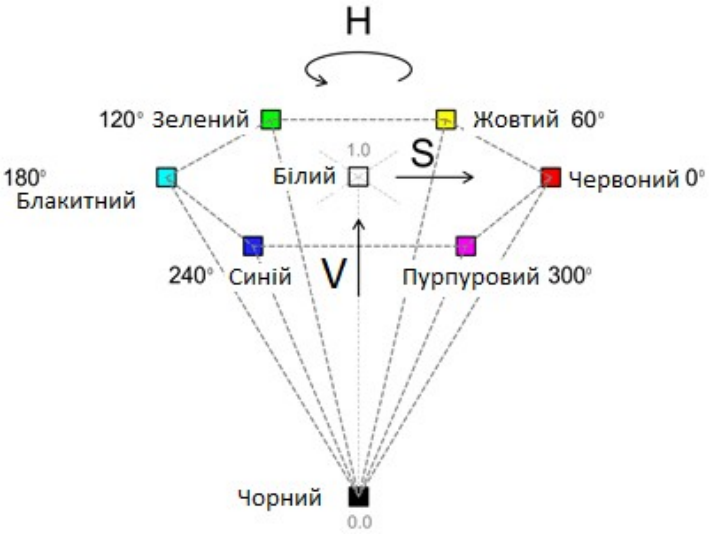
$$(R', G', B') = \begin{cases} (C, X, 0), & 0 \le H \le 60, \\ (X, C, 0), & 60 \le H \le 120, \\ (0, C, X), & 120 \le H \le 180, \\ (0, X, C), & 180 \le H \le 240, \\ (X, 0, C), & 240 \le H \le 300, \\ (C, 0, X), & 300 \le H \le 360, \end{cases}$$

$$(R, G, B) = (R'+m, G'+m, B'+m).$$

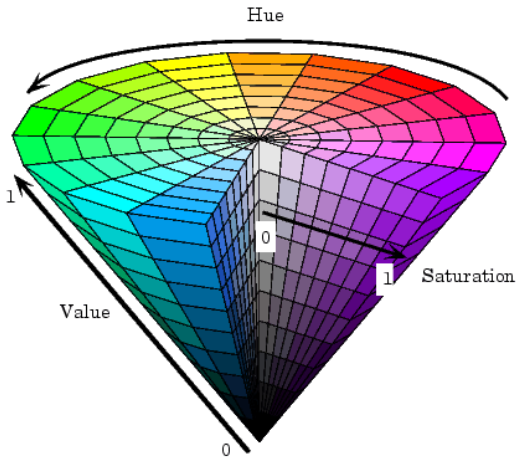
Схематично дане перетворення може бути записане наступним чином – треба поставити куб кольорів RGB так, щоб діагональ із сірих кольорів стояла вертикально і зробити проекцію на верхню опірну площину:



Візуалізація простору кольорів HSV буде виглядати наступним чином



або детальніше



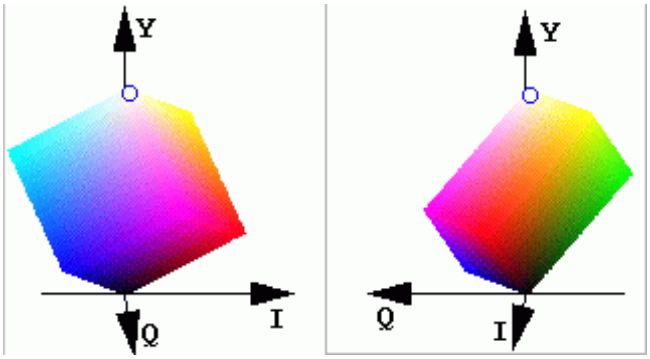
Той факт, що сприймання базових кольорів людиною не однакове, призводить до оцінки важливості їх сприйняття. Для людини більш важливі «теплі» кольори і менш – «холодні». Дане ствердження активно використовується у задачах стиску зображень, наприклад, у методах стиску, таких як jpeg, jpeg2000, djvu, і, особливо для стиску відеопотоку.

Системи передачі кольорів YUV, YIQ, YCrCb

Популярність RGB моделі пояснюється, перш за все, тим, що відображення кольору на екрані монітора реалізується саме змішуванням базових кольорів. Однак великим недоліком цієї системи є рівноправність всіх колірних компонентів.

Цей фактор стимулював появу інших систем передачі кольору, в яких основну інформаційне навантаження несе одна колірна компонента - люмінесцентна складова (яскравість зображення). Для доповнення люмінесцентної складової до оригінального сигналу існують дві колірні компоненти. Структура цих компонентів визначає ту чи іншу систему передачі кольору - YUV, YIQ, YCrCb.

У 1953 р Національний комітет з телевізійних систем (NTSC - National Television System Committee) прийняв в якості стандарту колірну систему YIQ, засновану на моделі МКО XYZ. Через обмеження на ширину смуги пропускання, яскравість визначається однією координатою Y. Сигнал Y займає основну частину смуги частот (0-4 МГц), причому в ньому пропорції червоного, зеленого і синього основних кольорів NTSC обрані так, що він відповідає кривій спектральної чутливості ока. У сигналі Y міститься інформація про яскравість, тому в чорно-білому телебаченні використовується тільки ця координата. Для того щоб передавати колір, використовуючи тональність і насиченість, за допомогою більш вузької смуги частот, враховуються деякі особливості сприйняття кольорів оком. Зокрема, чим менше предмет, тим гірше розрізняються його колір, а об'єкти, менші певного розміру, здаються чорно-білими. Якщо ж об'єкт менше деякої мінімальної межі, то його колір взагалі не сприймається. В системі YIQ інформація про тональність і насиченість кольору представляється за допомогою лінійних комбінацій різниць червоного, зеленого і синього кольорів і значення Y. Координата кольору I (синфазний сигнал) відповідає кольорам від помаранчевого до блакитного, так званим "теплим" тонам, Q (інтегрований сигнал) - від зеленого до пурпурового, тобто, всім іншим. Координата I займає смугу частот приблизно 1.5 МГц, а Q - тільки 0.6МГц.



Інтерпретація простору YIQ

Для зв'язку між RGB та YIQ використовуються співвідношення

Y=0.299Red+0.587Green+0.114Blue,
I=Red-Cyan=0.596Red-0.275Green-0.3216Blue,
Q=Magenta-Green=0.212Red-0.523Green+0.311Blue,

і, відповідно,

Red=Y+0.956 I+0.621 Q,
Green=Y-0.272 I-0.647 Q,
Blue=Y-1.107 I+1.704 Q.

Зазначимо, що для традиційного однобайтового діапазону вимірювання базових кольорів RGB область значень Y від 0 до 255, а I вимірюється у проміжку від 0 до ±152, а V у проміжку від 0 до ±134.

Колірний простір YUV використовується для передачі кольорового зображення у телевізійних системах PAL (Phase Alternation Line) та SECAM (Sequentiel Couleur Avec M'emoire or Sequential Color with Memory).

Співвідношення, яке дозволяє зв'язати компоненти YUV з базовими кольорами RGB виглядає наступним чином

Y=0.299Red+0.587Green+0.114Blue,
U=-0.147Red-0.289Green+0.436Blue =0.492(Blue-Y),
V=0.615Red-0.515Green-0.100Blue =0.877(Red-Y)

і, відповідно,

Red=Y+1.140V,
Green=Y-0.395U-0.581V,
Blue=Y+2.032U.

Компоненти I,Q пов'язані з U,V наступним чином

$$\begin{pmatrix} I \\ Q \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \cos 33^\circ & \sin 33^\circ \\ -\sin 33^\circ & \cos 33^\circ \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix}.$$

Зазначимо, що якщо базові кольори RGB змінюються у діапазоні від 0 до 255, то область значень Y також від 0 до 255, а U лежить у діапазоні від 0 до ±112, а V від 0 до ±157.

Колірний простір YCrCb (люмінесцентна складова, хроматичний червоний, хроматичний синій) найрозповсюджений серед комп'ютерних відео- стандартів. Як раз цей простір кольорів використовується у таких популярних форматах як JPEG, MPEG, Kodak's Photo YCC. YCrCb представляє собою масштабований в один байт (по кожній компоненті) простір кольорів YUV.

Існує декілька модифікацій цього простору кольорів. У YCrCb-SDTV компонента Y займає 8 біт і змінюється у діапазоні від 16 до 235, а Cr та Cb від 16 до 240.

Y=0.299Red+0.587Green+0.114Blue,

Cb=-0.172Red-0.339Green+0.511Blue+128,
Cr=0.511Red-0.428Green-0.083Blue+128,

і, відповідно,

Red=Y+1.371(Cr-128),
Green=Y-0.698(Cr-128)=-0.336(Cb-128),
Blue=Y+1.732(Cb-128).

А так як, під кожне значення кольорової компоненти відводиться по вісім біт, то дана система не повністю використовує надані ресурси. Для того, щоб мати можливість повністю використовувати байт, який виділений під кожну кольорову компоненту, можна провести модифікацію

Y=0.257Red+0.504Green+0.098Blue+16,
Cb=-0.148Red-0.291Green+0.439Blue+128,
Cr=0.439Red-0.368Green-0.071Blue+128,

і, відповідно,

Red=1.164(Y-16)+1.596(Cr-128),
Green=1.164(Y-16)-0.813(Cr-128)=-0.391(Cb-128),
Blue=1.164(Y-16)+2.018(Cb-128).

У YCrCb-HDTV кольорова компонента Y займає 8 біт і змінюється у діапазоні від 16 до 235, а Cr та Cb від 16 до 240.

Y=0.213Red+0.715Green+0.072Blue,
Cb=-0.117Red-0.394Green+0.511Blue+128,
Cr=0.511Red-0.426Green-0.047Blue+128,

і, відповідно,

Red=Y+1.540(Cr-128),
Green=Y-0.459(Cr-128)=-0.183(Cb-128),
Blue=Y+1.816(Cb-128).

Комп’ютерна реалізація цього простору кольорів наступна

Y=0.183Red+0.614Green+0.062Blue+16,
Cb=-0.101Red-0.338Green+0.439Blue+128,
Cr=0.439Red-0.399Green-0.040Blue+128,

та

Red=1.164(Y-16)+1.793(Cr-128),
Green=1.164(Y-16)-0.534(Cr-128)= -0.213(Cb-128),
Blue=1.164(Y-16)+2.115(Cb-128).

У стандарті JPEG2000 використовується наступна схема

Y=0.299Red+0.587Green+0.144Blue,
Cb=-0.16875Red-0.33126Green+0.5Blue,
Cr=0.5Red-0.41869Green-0.08131Blue,
Red=Y+1.402Cr,
Green=Y-0.34413Cr-0.71414Cb,
Blue=Y+1.772Cb.

Ще однією модифікацією є розробка Eeastman Kodak Company – простір кольорів PhotoYCC. Для даного простору кольорів люмінесцентна складова і хроматичні сигнали отримуються наступним чином

Y=0.213Red+0.419Green+0.081Blue,
C1=-0.131Red-0.256Green+0.387Blue+156,
C2=0.373Red-0.312Green-0.061Blue+137,
Red=0.981Y+1.315(C1-137),

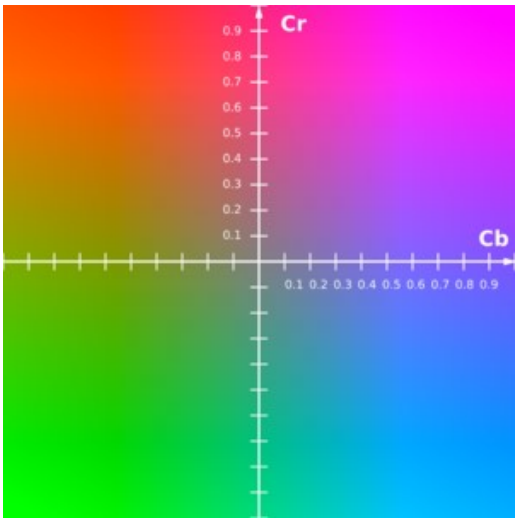
$$Green=0.981Y-0.311(C1-156)=-0.699(C2-137),$$
$$Blue=0.981Y+1.601(C2-156).$$

Для того, щоб реалізувати швидке відновлення базових кольорів, використовуються формули:

$$Y=\frac{1}{23}(7Red+14Green+2Blue),$$
$$Cb=-\frac{4}{23}(Red+2Green-3Blue),$$
$$Cr=\frac{4}{69}(8Red-7Green-Blue).$$

і зворотні

$$Red=Y+\frac{3}{2}Cr,$$
$$Green=Y-\frac{1}{4}(3Cr+Cb),$$
$$Blue=Y+\frac{7}{4}Cb.$$



Площина CrCb при нормалізованому значенні Y=0.5

Психологічне сприйняття кольорів

Численні психологічні і соціологічні дослідження на цю тему дають в цілому однакову картину. З одного боку, емоційний вплив кольорів обумовлений властивостями універсальних об'єктів (блакитне небо, червона кров, жовте сонце, зелена трава і ін.), З іншого боку - впливом культури, і неважливо, усвідомлюємо ми її вплив чи ні. У повсякденному житті з символікою кольорів мають справу не тільки живописці, дизайнери, графіки та інші представники мистецьких професій. Емоційний вплив кольорів відбивається, наприклад, у мові, наприклад, вирази «немов червона ганчірка для бика», «люди в білих халатах», «почорніти від злості» і багато інших. У таблиці дані варіанти символіки кольорів і характер їх впливу на людину.

Колір / тон	**Асоціації та вплив**
яскраві тони	Жвавість, яскравість
Неяскраві	Спокій, м'якість

(пастельні) тони	
жовтий	Привітність, веселість, енергійність, жвавість, «сонячна» атмосфера.
помаранчевий	Тепло, впевненість, світло.
синій	Розслаблює, заспокійливо. Сам по собі або в поєднанні з білим блакитний колір вселяє відчуття холоду, стерильності, некомунікабельності.
зелений	Стабільність, надійність. Впливає освіжаюче, заспокійливо або нейтрально. Асоціюється з природою навесні. колір життя.
коричневий	Створює атмосферу затишку і безпеки. Відтінки коричневого сприймаються як приємні і які спонукають до комунікації. Бежевий колір сприймається як претензійний.
червоний	Активізує і збуджує. Теплий колір, який здавна вважається кольором пристрасті, провокації, небезпеки. Це колір екстремізму, крайнощі
рожевий	Цей колір набагато спокійніше, ніж червоний. З рожевим асоціюються інтимність і щастя
сірий	Гідність, впевненість. Однак цей колір може впливати депресивно, вселяти думки про старість. Емоційно - нейтральний
чорний	Передає глибину, однак може впливати депресивно, наводити на думки про хвороби і смерті (траур).Крім того, асоціюється з владою
білий	Символізує невинність, божественність, нейтральність, при використанні у якості фону створює відчуття стерильності.

Вивчення психології кольору (Psychology of Color) може істотно допомогти дизайнеру в роботі над макетом сайту. Використання правильних відтінків, гармонійно поєднаних між собою, дозволяє в якійсь мірі впливати на користувачів і налаштовувати їх на певну модель поведінки. Давно відомо, що різні кольори викликають різні емоційні реакції, а це означає, що грамотне використання кольору може допомогти в підвищенні конверсії сайту, так як відвідувачі не будуть відчувати негативних емоцій.

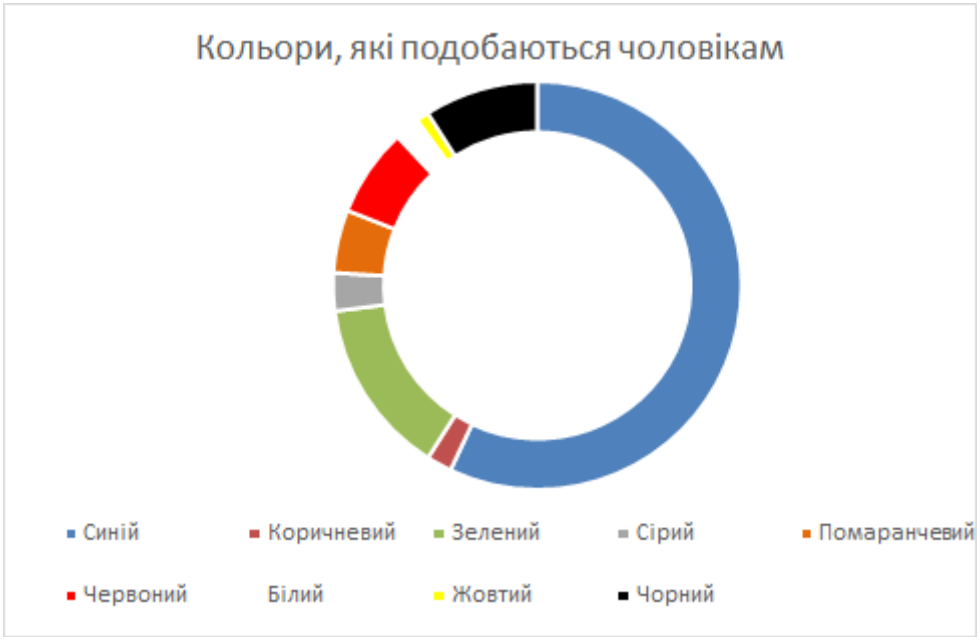
Основний колір, який використовується на сайті, в більшості випадків відповідає заявленій тематиці. Однак є ряд випадків, коли потрібно піти від обраних відтінків і додати інші кольори. Але потрібно бути обережним, так як неправильний вибір кольору може відштовхнути користувача і всі зусилля дизайнера не принесуть результату.

Згідно з даними, опублікованими на сайті moz.com , дизайнери, що працюють над сайтом онлайн-казино, просто змінили колір кнопки із закликом до дії з зеленого на жовтий. Результат - колосальне зростання конверсії на 187,4%. При цьому вибір був не випадковий - до таких результатів вдалося прийти, перепробувавши безліч відтінків, і тільки один з них дійсно показав зростання конверсії. Навіть інший відтінок жовтого збільшував конверсію максимум на 15%. Є над чим замислитися.

У психології кольору кожен відтінок має емоційне забарвлення. Правильне використання колірної гами дійсно може допомогти в підвищенні конверсії. Але який колір надає найбільшу увагу на користувача сайту? Це залежить не тільки від того, що зазначено вище, але і від того, яка цільова аудиторія сайту.

Рожевий . Якщо цільова аудиторія сайту в основному жіноча, то відтінки рожевого будуть хорошим вибором. Рожевий - це колір радості і романтики. Все що залишається - вибрати потрібний відтінок, максимально відповідний образу бренду.
Синій . Синій колір асоціюється з надійністю і спокоєм. Для серйозних комерційних сайтів, соціальних мереж та інших веб-ресурсів, які позиціонують себе як надійні, вибір синього кольору представляється найбільш вірним рішенням.
Червоний . Червоний колір краще за інших привертає увагу. Якщо потрібно виділити який-небудь елемент, наприклад, кнопку дії, то є сенс використовувати відтінки червоного. Однак при цьому червоний не повинен вибиватися з основної палітри сайту.
Зелений . Зелений - це колір життя, миру і спокою. Якщо потрібно створити сайт, присвячений проблемам екології або інтернет-магазин з продажу екологічних продуктів, то вибір очевидний.
Жовтий . Маркетологи знають, що жовтий колір асоціюється з позитивом і відсутністю турбот. Жовтий найкраще підходить для інтернет-магазинів, які торгують іграшками та дитячим одягом.
Фіолетовий . Це вишуканість і елегантність. У поєднанні з відтінками жовтого і золотом фіолетовий можна використовувати на сайтах, тематика яких - комфорт і розкіш.
Помаранчевий . Дуже часто відтінки оранжевого в поєднанні з контрастними кольорами використовують бренди, пов'язані з високими технологіями або сайти, що представляють різні електронні гаджети.
Золотий . Як відомо, золото асоціюється з впливом і престижем. Сьогодні «металеві» відтінки не так популярні, але вони можуть бути корисні, якщо потрібно продемонструвати елегантність або стабільність.
Чорний . Універсальний колір, який чудово поєднується з іншими відтінками. Його краще використовувати тоді, коли потрібно добитися контрасту на веб-сторінці.
Коричневий . Даний колір, не відрізняється яскраво вираженим позитивом чи емоційним забарвленням. Проте, даний колір прекрасно підходить для безлічі сайтів. Меблі, будівництво, екологія - це теми, де відтінки коричневого будуть цілком доречні.

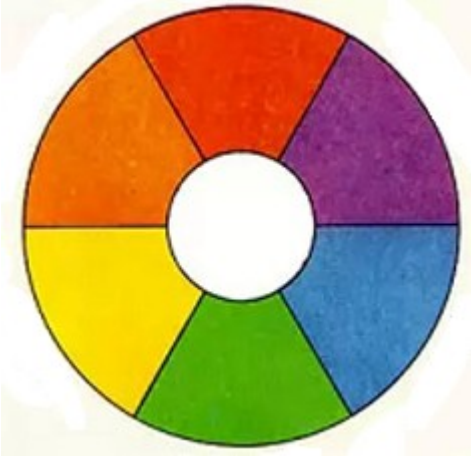
Орієнтація за статтю також має велике значення.



При цьому потрібно пам'ятати, що, незважаючи на глобалізацію, значення кольорів можуть істотно відрізнятися в різних культурах. У Китаї білий колір асоціюється зі смертю, а в європейській культурі це колір чистоти і невинності. Фіолетовий і бузковий вважаються жіночними відтінками в більшості країн, крім Бразилії, де він так само, як і у Китаї, нагадує про смерть. Так що, приступаючи в підбору кольорової гами, не завадить замислитися про цільову аудиторію і підібрати такі відтінки, які не викликають негативних емоцій у відвідувачів сайту.

Але важливо не тільки вибрати відповідний колір, а й колірну гамму, яка буде гармонійно взаємодіяти, тобто треба знати як психологічно взаємодіють кольори між собою, наприклад, жовтий, червоний і блакитний виглядають яскравіше на сірому фоні.
Сер Ісаак Ньютон (1642-1727), проводячи експерименти зі світлом, з'єднав у коло обидва края видимого спектру - червоний і фіолетовий, виділивши сім базових

кольорів. В англійській мові їм відповідає абревіатура ROYGBIV (червоний, оранжевий, жовтий, зелений, блакитний, синій і фіолетовий).
За традицією часто опускають синій колір і беруть до уваги шість основних, або первинних квітів.



Колірне коло

Будь-який колір, розташований на колірному колі безпосередньо навпроти іншого кольору, називається додатковим. Якщо змішати разом доповнюючі кольори, вийде нейтральний сірий колір.
Хроматична складова, або кольоровість - це інтенсивність колірної поверхні, що сприймається оком, в порівнянні з білим кольором (насиченість - близький за змістом термін, який має відношення, скоріше, до чистоти світла). У міру просування колірних відтінків від серцевини по краю колірного кола хроматична складова посилюється. Центр кола забарвлений в нейтральний сірий колір.

Насичення хроматичних кольорів ахроматичними

Досить спірне питання полягає в тому, що розподіл кольорів за класичним колом не пропорційне: жовто-помаранчеве-червоний сектор спектра надмірно розширено. Такий нерівномірний розподіл пояснюється тим, що наші очі набагато чутливіші до невеликих варіацій між жовтим / помаранчевим / червоним відтінками, тому, що у людському оці пігментів теплих тонів значно більше, ніж холодних.
Гармонійні поєднання кольорів

Тепер ми впритул підійшли до найважливішого моменту, щоб дізнатися, як же ці хроматичні кольори можна поєднувати один з одним. Перше гармонійне поєднання кольорів **одноколірні (монохромне)**. Виконується воно на основі відтінків в межах одного сектора колірного кола.

Друге гармонійне поєднання це поєднання сусідніх кольорів воно називається **аналогічне або аналогічна тріада** .

Третє гармонійне поєднання називається **додаткове** поєднання, тобто два кольори протилежні одна одній на колірному колі.

Якщо до двох гармонійних додаткових кольорів колірного кола додати сусідні, то отримаємо поєднання, зване **розбите доповнення** .

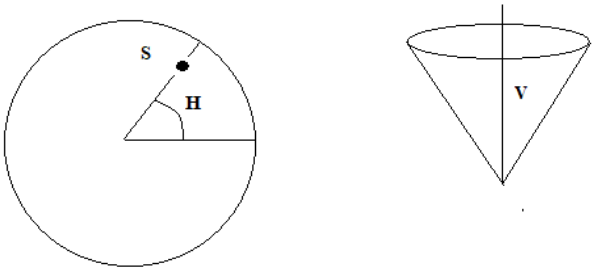
І останнє гармонійне поєднання ґрунтується на **трьох рівновіддалених** один від одного кольорах колірного кола.

Також існують і різні гармонійні поєднання кольорів на основі квадрата і прямокутника (чотирьох гармонійних кольорів) і п'ятикутника (п'ять гармонійних кольорів)

Як видно, ні білий, ні чорний, ні сірий колір не входять до складу колірного кола, а все тому, що вони є, по-перше не спектральними, а по-друге ахроматичними (тобто не кольоровими) . Так ось, ахроматичні кольори: чорний, білий і їх відтінок - сірий, дуже добре поєднуються з усіма спектральними кольорами колірного кола, так як є нейтральними до хроматичних, і мають всього одну якісну характеристику - світлоту. Вони відмінно доповнюють хроматичні кольори, підкреслюють їх, і вносять додаткову гармонію в колірну гамму. На практиці можна використовувати і поєднувати ахроматичні кольори разом з хроматичними в будь-якій кількості.

Що ж стосується **коричневого кольору** , то тут все дещо складніше. Оскільки він, є третинним кольором, тобто змішаним, то велике значення має те, від якого саме кольору він утворився. Якщо коричневий утворений на основі підмішування чорного в червоний або помаранчевий, то поєднання цих кольорів створюється за загальними правилами колірного кола.

Наведемо інший підхід до вибору гармонійної палітри. Часто дизайнер робить головну сторінку сайту у відповідних кольорах і має бажання, щоб інші сторінки, відповідали іншому базовому кольору, але зберегти співвідношення між кольорами кожної сторінки такими, які були на головній сторінці.

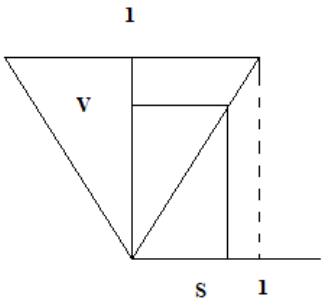


Нехай P=(H,S,V) опірна і $P_1 = (H_1, S_1, V_1)$ довільна точки із конусу HSV. Виберемо довільну фіксовану точку $\overline{P} = (\overline{H}, \overline{S}, \overline{V})$ і знайдемо точку $\overline{P}_1 = (\overline{H}_1, \overline{S}_1, \overline{V}_1)$, так, щоб між цією парою точок зберегти ті ж співвідношення, що і у першій парі.

А тепер, що будемо розуміти під цим співвідношенням: якщо $h = H - H_1$, то $H_1 = H - h$ і

$$\overline{H}_1 = \overline{H} - (H - H_1) \qquad (1.1)$$

Так як на ребрі конусу S=V, то проведемо нормування



Тоді

$$\frac{S}{V}\frac{V_1}{S_1}=\frac{\overline{S}}{\overline{V}}\frac{\overline{V}_1}{\overline{S}_1}$$

Окрім того,

$$\frac{VS}{V_1S_1}=\frac{\overline{VS}}{\overline{V}_1\overline{S}_1}.$$

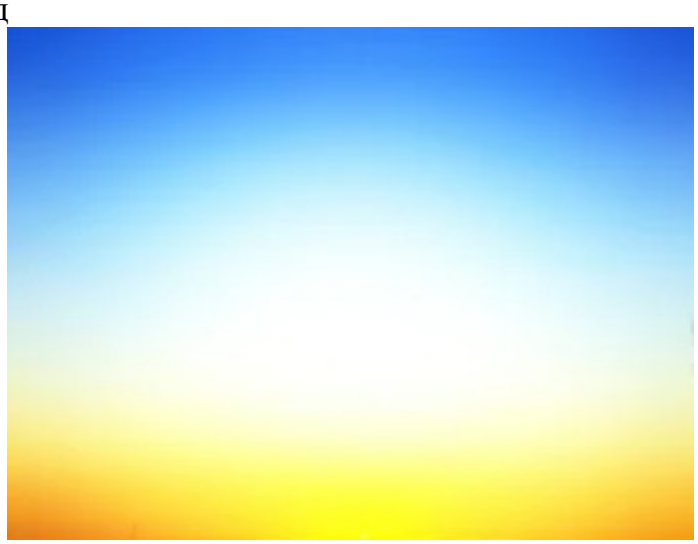
Звідси

$$\overline{V}_1=\sqrt{ab}\ \text{і}\ \overline{S}_1=\sqrt{\frac{a}{b}}, \qquad (2)$$

де

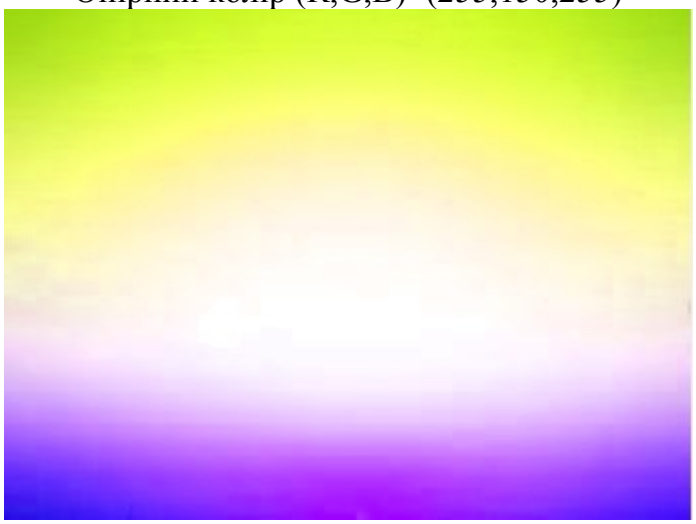
$$a=\frac{\overline{VS}}{VS}V_1S_1\ \text{і}\quad b=\frac{S}{V}\frac{V_1}{S_1}\frac{\overline{V}}{\overline{S}}.$$

Розглянемо приклад



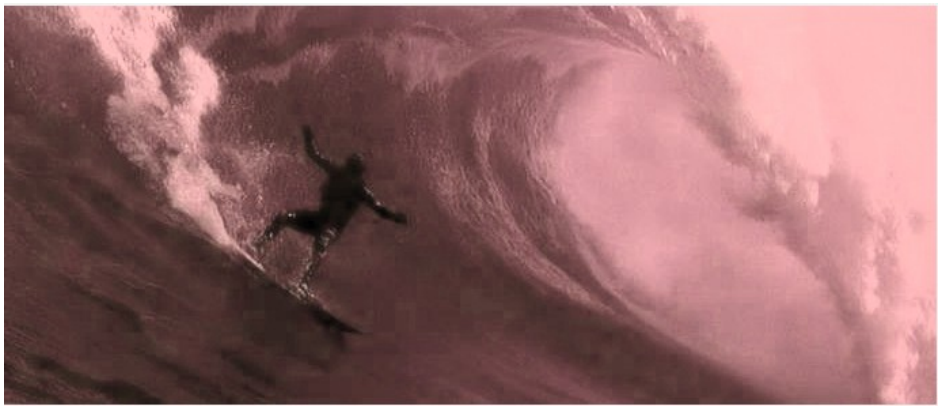
Базовий малюнок

Опірний колір (R,G,B)=(255,150,255)

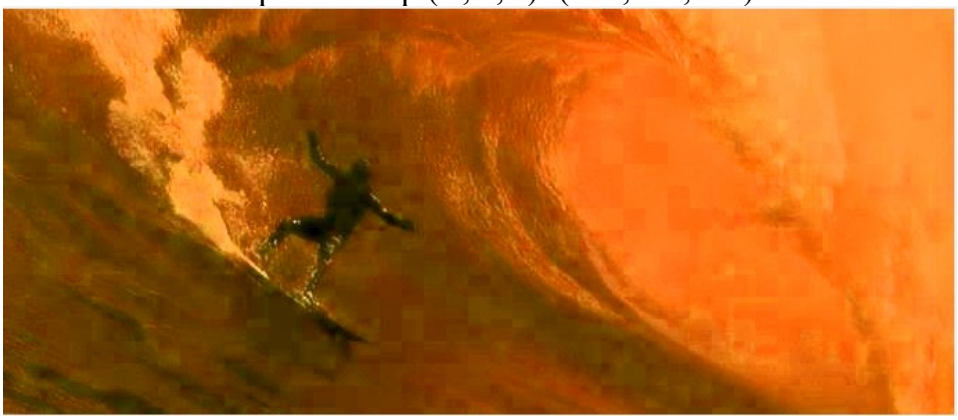


Опірний колір (R,G,B)=(255,150,150)

Базове зображення



Опірний колір (R,G,B)=(100,150,100)



Опірний колір (R,G,B)=(0,200,100)

Проблеми зеленого кольору

Зелений колір - один з найпоширеніших у природі, але для мистців він перетворився у вічну проблему, і багато хто вигнав його зі своєї палітри. Так чому ж зелений колір викликає такі труднощі, і як можна з ними впоратися?

Зелений колір у природі, безсумнівно. є важливим кольором. У сучасній англійській мові слово green (зелений) зустрічається в два рази частіше, ніж yellow (жовтий). Людське око більш чутливе до довжини хвиль жовто-зеленого кольору, ніж до хвиль будь-якого іншого кольору, ось чому колірний спектр або веселка в цьому сегменті здаються світліше.

Однак серед оформлювачів книжкових обкладинок існує усталена думка, що «зелені обкладинки не продаються». На державних прапорах Європи на 79% присутній червоний колір, але зелений тільки на 16%. Дизайнери одягу кажуть, що зелений колір на подіумі часто виглядає жахливо. Галеристи відзначають, що клієнтів не приваблюють картини в яскраво-зеленій тональності, за винятком тих випадків, коли автор знає, як з нею поводитися.

Поради по оволодінню зеленим кольором

1. Можна відмовитися від зеленого кольору на своїй палітрі і змішати його з різних відтінків блакитного і жовтого. Одержаний колір буде менш інтенсивним і більш різноманітним,
2. Уникайте монотонності. Варіюйте відтінки зеленого кольору як на дрібних деталях. так і на великих фрагментах.
3. Використовуйте допоміжні рожевий або червонувато-сірий кольори і позбавте ними зелені плями і області поруч з ними (метод контрабандного червоного кольору).

Розмитість у русі

Існують два роду розмитості зображення, що створюють враження руху або дії: розмитість зображення об'єктів, що рухаються - коли об’єкт швидко переміщається перед спостерігачем, що нерухомо стоїть, або зафіксованою фотокамерою, і розмитість на швидкості, коли камера переміщається разом зі швидко рухаючим об'єктом.

Якщо переглянути покадрово звичайну кінозйомку, край всіх об'єктів, що рухаються, виявляться трохи змазаними.

Візуалізація руху (http://www.gunook.com/bewegungsunscharfe-fotografie/).

Зворотна повітряна перспектива

Згідно із загальним законом повітряної перспективи, «теплі кольори просуваються вперед, а холодні відступають назад». Але в рідкісні, дивовижні миті це правило встає з ніг на голову, і вся сцена теплішає в міру віддалення від глядача.

Running through a fire background (2012)- *James Bond SkyFall*

Зверніть увагу, що кольори на передньому плані холодніше, ніж на видаленні.

Через те що явище зворотної повітряної перспективи в природі зустрічається досить рідко, воно породжує відчуття чогось дивного і хвилюючого.

Колірний акцент

Легкий кольоровий штрих може оживити чорно-білий ескіз або картину в сірих тонах. Колірний акцент притягує очей до ключового моменту композиції.

Колірний акцент - це будь-який невеликий фрагмент кольору, що помітно відрізняється від решти колірної гами композиції. Колірний акцент зазвичай роблять доповнюючим або близьким до нього кольором і, як правило, для цього вибирають більш насичений в порівнянні із загальною тональністю композиції, колір. Якщо ви обмежите колірну гамму, наблизивши зображення до монохромного композиції, ніщо не буде кидатися в очі, привертаючи увагу глядача.

«Ladies & umbrellas» Helen Cottle 1962

Контрольні питання

1. Чому в якості базових кольорів використовуються три кольори: червоний,синій та зелений?
2. Чим відрізняються простори кольорів YUV та HSV?
3. Як співвідносяться кольори та їх психологічне сприймання у людей?
4. Чи відрізняються сприймання кольорів у людей за статтю?
5. В чому потреба в різних просторах кольорів?

Рекомендована література

1. Gurney J. Color and Light. A Guide for the Realist Painter / J.Gurney .— Kansas Sydney London: Adrews McMell Publishing, 2010 .— 221 p.
2. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс . – М.: Техносфера, 2005 .– 1070 с.

3. Лигун А.О. Комп'ютерна графіка (Обробка та стиск зображень) / А.О.Лигун, О.О.Шумейко .— Дніпропетровськ: Біла К.О., 2010 .— 114 с.
4. Психологические аспекты web-дизайна [Електронний ресурс] .— Режим доступу: https://studbooks.net/2141278/informatika/psihologicheskie_aspekty_web-dizayna
5. Как использовать психологию цвета в веб-дизайне? [Електронний ресурс] // https://rusability.ru/usability/ .— Дата звернення: 06.04.2019
6. Цвет влияет на принятие решений [Електронний ресурс] // http://itnews.com.ua/analitics/ .— Дата звернення: 06.04.2019

ТЕМА 2. ОБРОБКА РАСТРОВИХ ЗОБРАЖЕНЬ

Шуми, їх властивості, методи зменшення впливу

Так як даний конспект лекцій присвячений комп'ютерній графіці, то у подальшому, говорячи про сигнал, ми будемо розуміти або зображення в градація сірого як однокомпонентний двовимірний сигнал, або, в разі повнокольорового зображення – трикомпонентний двовимірний сигнал. У останньому випадку компонентами сигналу можуть бути кольори – червоний (R), зелений (G), синій (B), або хроматичні складовіY - освітленість, Cr- теплі відтінки, Cb-холодні відтінки, або елементи простору HSV, головні компоненти PCA простору кольорів, або компоненти іншого простору кольорів.

Переходячи до розгляду теми розділу, треба зазначити, що в реальних сигналах шум так чи інакше присутній. Цей факт обумовлений як апаратною складовою, за допомогою якої отримане зображення, так і впливом стану атмосфери, рухом відносно об'єкта з'йомки, та ін. Компоненти шуму зображень n [k1, k2] можуть об'єднуватися з корисним сигналом s [k1, k2] адитивно:

$$f [k1, k2] = s [k1, k2] + n [k1, k2],$$

або мультиплікативно:

$$f [k1, k2] = s [k1, k2] * n [k1, k2].$$

Типовий приклад мультиплікативної взаємодії сигналу з шумом - зв'язок між освітленістю відеооб'єктиву (корисний сигнал) і світловим потоком, відбитим від місцевих об'єктів (шум).

До аддитивних відносяться, в основному, шуми, обумовлені властивостями чутливих елементів відео або фотокамер. Ці шуми виникають з наступних причин:

1. Дефекти (домішки та ін.) потенційного бар'єру, які викликають витікзаряду, згенерованого за час експозиції - т. зв. чорний дефект. Такідефекти видно на світлому фоні у вигляді темних крапок.

2. Темновий струм (Dark current) - є шкідливим наслідком термоелектронної емісії і виникає в сенсорі при подачі потенціалу на електрод. Такі дефекти видно на темному тлі у вигляді світлих точок, це т. зв. білий дефект. Білі дефекти особливо проявляються при великих експозиціях.

Основна причина виникнення темнового струму - це домішки в кремнієвій пластині або пошкодження кристалічної решітки. Чим чистіше кремній, тим менше темновий струм. На темновий струм впливає температура елементів камери і електромагнітні наводки. При збільшенні температури на 6-8 градусів, значення темнового струму подвоюється.

3. Шум, що виникає внаслідок стохастичної природи взаємодії фотонів світла з атомами матеріалу фотодіодів сенсора. Під час руху фотону усередині кристалічної решітки може виникнути ситуація, що фотон, «потрапивши» в атом кремнію, виб'є з нього електрон, народивши пару електрон-дірка. Електричний сигнал, що знімається з сенсора, буде відповідати кількості народжених пар.

4. Наявність дефектних (які не працюють) пікселів, які виникаютьпри виробництві фотосенсорів. Для усунення їх негативного впливу використовуються математичні методи інтерполяції, коли замість дефектного «підставляється» або просто сусідній елемент, або середнє по прилеглим елементам, або значення, обчислене більш складним чином. Природно, що обчислене значення відрізняється від фактичного і погіршує якість отриманого зображення.

Далі розглянемо різні методи боротьби з шумом, тобто, очищення зображень.

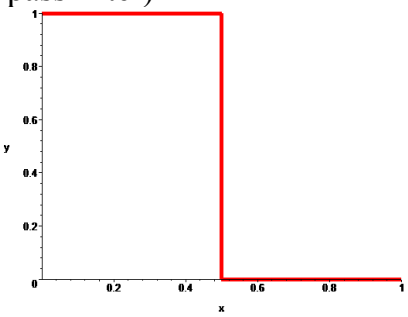
Трохи про фільтри

Імпульсна перехідна функція (impulse response) або імпульсна характеристика - це відгук системи на дельта-функцію Дірака. Якщо розглядати лінійну систему в просторі часу, то сигнал на виході лінійної системи y (t) можна розрахувати як згортку вхідного сигналу x (t) з імпульсною характеристикою h (t):

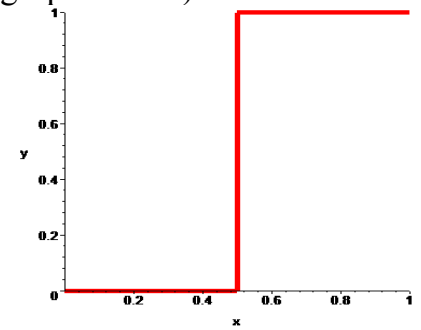
$$y(t) = h(t) \otimes x(t) = \int_{-\infty}^{\infty} h(\tau) x(t - \tau) d\tau.$$

Для перетворення сигналів часто використовуються лінійні системи, звані фільтрами, передавальна функція яких (частотна характеристика) має певну форму. Одні з найбільш використовуваних фільтрів - це порогові фільтри, наприклад,

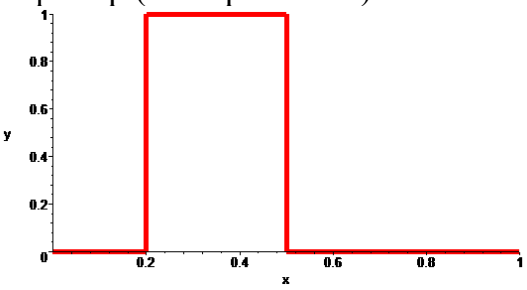
- фільтр низких частот (low-pass filter)



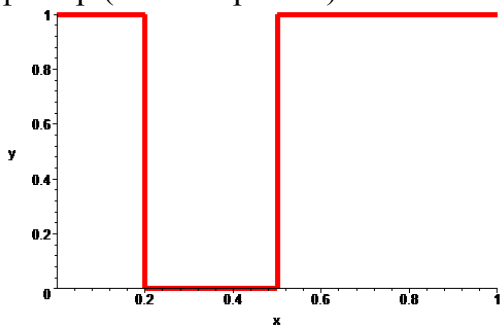
- фільтр високих частот (high-pass filter)



- полосно-пропускаючий фільтр (band-pass filter)



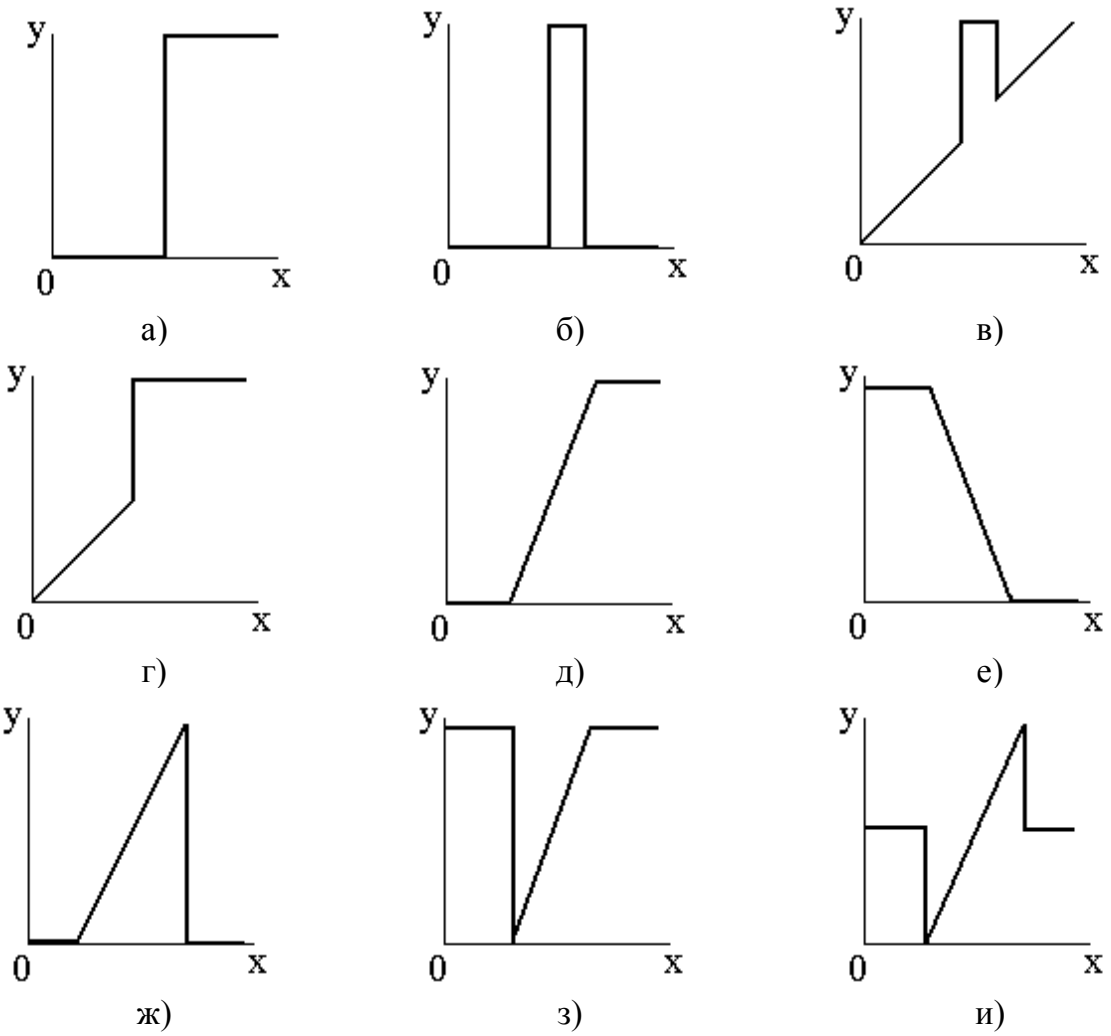
- полосно-обмежуючий фільтр (band-stop filter)



Розглянемо деяке узагальненя розглянутих фільтрів.

Препарування зображення

Препарування представляє собою клас поелементного перетвореня (як правило, освітлення) зображень. Характеристики застосовуваних на практиці процедур препарування наведені на рисунку нижче.



Перетворення з пороговою характеристикою (а) називається бінарізацією, тобто, зображення перетворюються в бінарне – якщо яскравість (освітленність) менше порогового значення, то компонента отримує значення 0, інакше – 1. Операція бінаризації або бінарного квантування, може бути корисною, у разі, коли спостерігачу важливі контури об'єктів, присутніх на зображенні, а деталі, що містяться всередині об'єктів або всередині фону, не представляють інтересу. Основною проблемою при проведенні такої обробки є визначення порогу x_0, порівняння з яким яскравості вхідного зображення дозволяє визначити значення вихідного зображення в кожній його точці. Заміна вхідного напівтонового зображення бінарним дозволяє досягти більшої наочності при візуальному сприйнятті, ніж у вхідного зображення.

Безумовно, ключовим у бінарізації є вибір порогу. Існує багато різних підходів до вибору порогу бінарізації, але найбільш популярним є метод Оцу (大津展之 *ŌtsuNobuyuki*).

Ідея методу Оцу полягає в тому, щоб виставити поріг між класами таким чином, щоб кожен з них був якомога більш «щільним». Якщо виражатися математичною мовою, то це зводиться до мінімізації внутрішньокласової дисперсії, яка визначається як зважена сума дисперсій двох класів:

$\sigma_W^2 = w_1\sigma_1^2 + w_2\sigma_2^2,$

де для будь якого фіксованого k=0,1,…,255

$$w_1 = \frac{\sum_{i=0}^{k} h_i}{H \times W}, \quad \mu_1 = \frac{\sum_{i=0}^{k} ih_i}{\sum_{i=0}^{k} h_i}, \sigma_1^2 = \frac{\sum_{i=0}^{k} (i-\mu_1)h_i}{\sum_{i=0}^{k} h_i},$$

$$w_2 = \frac{\sum_{i=k+1}^{255} h_i}{H \times W}, \quad \mu_2 = \frac{\sum_{i=k+1}^{255} ih_i}{\sum_{i=k+1}^{255} h_i}, \quad \sigma_2^2 = \frac{\sum_{i=k+1}^{255} (i-\mu_1)h_i}{\sum_{i=k+1}^{255} h_i},$$

і $\{h_i\}_{i=0}^{255}$ –гістограма інтерсивності освітлення (компоненти Y).

Перебираємо по черзі всі k=0,1,…,255 і те значення, при якому досягається мінімум величини σ_W^2 буде відповідати порогу бінарізації.

Оригінальне зображення Lena

Бінарізація зображення Lena методом Оцу.

Сенс інших перетворень неважко зрозуміти, якщо розглянути їх характеристики. Наприклад, перетворення (б) виконує зріз яскравості зображення, виділяючи ті його ділянки, де яскравість відповідає виділеному інтервалу. При цьому інші ділянки виявляються повністю "погашеними" (мають яскравість, що відповідає рівню чорного). Переміщуючи виділений інтервал по шкалі яскравості і змінюючи його ширину, можна детально дослідити зміст зображення.

Просторові методи обробки зображень

Просторові методи обробки зображень об'єднуєть підходи, засновані на прямому маніпулюванні пікселями зображення. Деякі локальні перетворення оперують одночасно як зі значеннями пікселів в околі, так і з відповідними їм значеннями деякої матриці, що має ті ж розміри, що і окіл. Таку матрицю називають фільтром, маскою, ядром, шаблоном або вікном. Значення елементів матриці прийнято називати коефіцієнтами.

У загальному випадку просторова обробка зображення описується рівнянням:

$$G(x,y) = T(I(x,y))$$

де G(x.y)- зображення на виході процедури обробки;

I(x,y) - вхідне зображення для обробки;

T - оператор системи обробки.

Найпростіша форма оператора T досягається, коли окіл має розміри 1×1 (один піксель). В цьому випадку G залежить тільки від значення Iв точці (x,y), і T

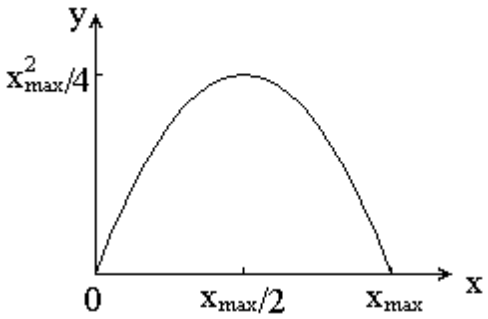
називається функцією градаційного перетворення (функцією перетворення інтенсивностей або функцією відображення).

Градаційні методи покращення зображень

Розглянемо деякі найбільш вживані градаційні методи.

Соляризація полягає в тому, що ділянки вхідного зображення, що мають рівень білого або близького до нього рівень яскравості, після обробки мають відповідний рівень чорного. При цьому зберігається рівень чорного і ділянки, що мають його на оригінальному зображенні.

Функція перетворення має вигляд: $y = k \cdot x \cdot (x_{max} - x)$, де x_{max} - максимальне значення вихідного сигналу, а k- константа, що дозволяє управляти динамічним діапазоном перетвореного зображення. Функція, що описує дане перетворення, є квадратичною параболою. При $y_{max} = x_{max}$ динамічні діапазони зображень збігаються, що може бути досягнуто при $k = 4/x_{max}$



Функція, яка описує солярізацію

Солярізація зображення Lena

Дана обробка призводить до підвищення чіткості деталей зображення: поліпшені зображення очей, підвищений контраст на переході "обличчя - волосся" і так інше.

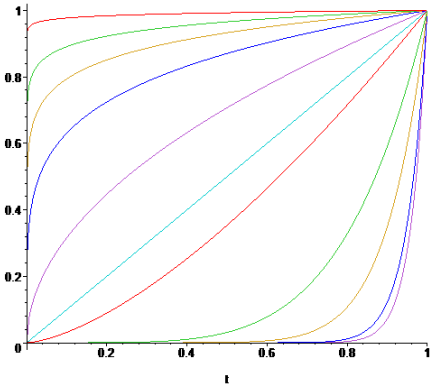
Гамма-корекція яскравості

Наші очі сприймають зображення не так, як цифрові пристрої. Гамма-корекція зображень - це процес, за допомогою якого цифрове кодування зображення приводиться у відповідність з нашим сприйняттям зображення.

Цей метод зміни яскравості зображення відноситься до статичних перетворень і описується виразом:

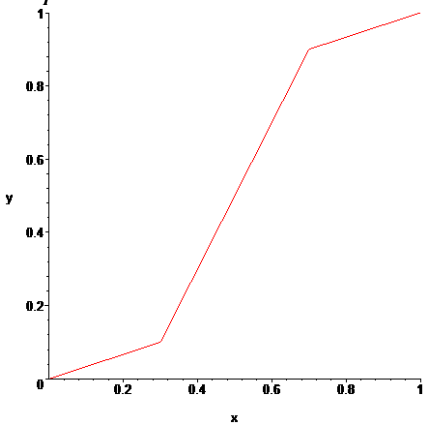
$$s_{вих} = c * s_{вх}^{\gamma}$$

де c та γ – додатні константи, а $s_{вх}$ і $s_{вих}$ - відповідно значення яскравості на вході і виході процедури її зміни. Вигляд відповідних співвідношень наведений на зображені



По горизонталі - яскравість на вході, під вертикалі - на виході.

Окремим випадком гамма-корекції можна вважати перетворення за допомогою кусково - лінійних функцій, форма яких може бути досить складною. Недолік цього підходу - необхідність введення та обчислення параметрів для кожного з ділянок опису характеристики перетворення.

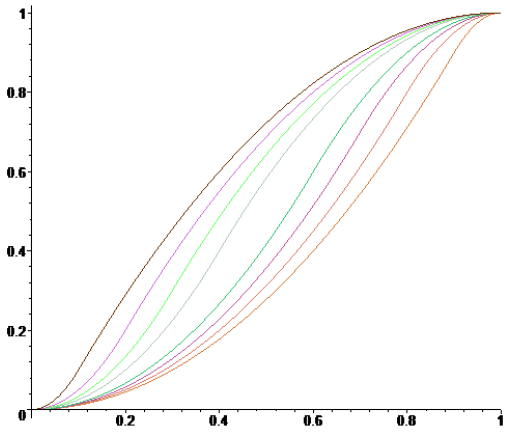


По горизонталі - яскравість на вході, під вертикалі - на виході.

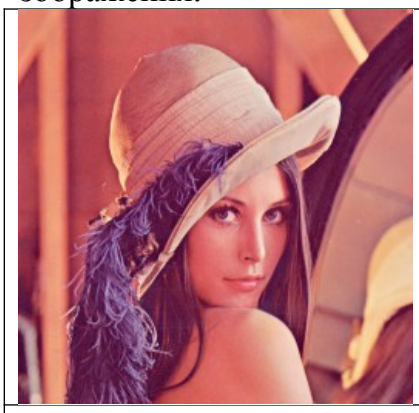
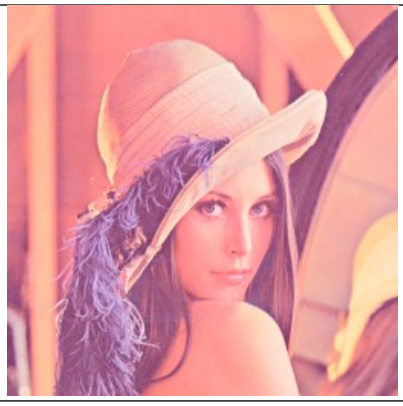
Цей підхід можна узагальнити, використовуючи функцію коригування яскравості у вигляді сплайну

$$S_{x0}(t) = \begin{cases} \dfrac{1}{x_0} t^2 & \text{если} \quad t \in [0, x_0], \\ 1 - \dfrac{(1-t)^2}{1-x_0} & \text{если} \quad t \in [x_0, 1] \end{cases}$$

при $x_0 \in (0,1)$.



За горизонталлю – освітленість на вході, за вертикаллю – на виході для функції корекції яскравості в вигляді сплайну при $x_0 \in \{0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9\}$. Змінюючи значення $x_0 \in (0,1)$, отримаємо цілий спектр функцій корекції яскравості, з яких можна вибрати як раз той сплайн, який найкращим чином перетворює зображення.

		
Оригінальне зображення Lena	*Зображення після γ-корекції для γ=0.5*	*Зображення після сплайн-корекції*

Вирівнювання гістограм

Гістограми є вихідним матеріалом для багатьох методів обробки зображень в просторовій області. Гістограма є характеристикою кількості пікселів в заданому діапазоні освітленості. Оскільки цю кількість ділять на загальну кількість пікселів, то таким чином, значення нормалізованої гістограми показують ймовірності появи в зображенні пікселів у цьому діапазоні освітленості. Очевидно, що сума ймовірностей повинна дорівнювати одиниці.

З вигляду гістограми можна зробити деякі висновки і про саме оригінальне зображення. Наприклад, гістограми дуже темних зображень характеризуються тим, що ненульові значення гістограми сконцентровано близько нульових рівнів яскравості, а для дуже світлих зображень навпаки - все ненульові значення сконцентровані в правій частині гістограми.

Розглянемо методи покращення зображень шляхом корекції їх гістограм.

1. Лінійна корекція гістограм (лінійне контрастування) зводиться до зміни діапазону інтенсивності вхідного зображення, так, у разі наявності області освітлення [a,b], треба розтягнути його до [0,255], тобто

$$\tilde{Y}_{i,j} = 255 \times \frac{Y_{i,j} - a}{b - a}$$

2. Нормалізація гістограми передбачає не тільки зміну діапазону інтенсивності освітлення, але і їх найінформативніші значення. Ясно, що найінформативніші значення відповідають пікам гістограм. Значення гістограми, які відповідають значенням освітлення, що зустрічаються рідко, в процесі нормацізації викидаються, після чого проводиться лінійна корекція.
3. Інтуїтивно можна зробити висновок, що найбільш зручним для сприйняття людиною буде зображення, для якого гістограма близька до рівномірного розподілу. Тобто для поліпшення візуальної якості, до зображення треба застосувати таке перетворення, щоб гістограма результату містила всі можливі значення яскравості і при цьому в приблизно однаковій кількості. Таке перетворення називається еквалізацією гістограми.

Нехай є зображення в градаціях сірого (компонента Y, освітленність, яскравість) I розміром H ⩳ W. Кількість рівнів квантування яскравості пікселів (число бінов) становитьN. Таким чином на кожен рівень припадає $n = \frac{H \times W}{N}$ пікселів. Нехай p(x) функція щільності розподілу інтерсивності освітлення на вхідному зображенні і p(y)-бажана щільність розподілу, така, що

$$p(y) = \begin{cases} \dfrac{1}{y_{max} - y_{min}}, & y_{min} \le y \le y_{max} \\ 0, & \text{інакше.} \end{cases}$$

Позначимо через F(x) та F(y) інтегральні закони розподілу випадкових величин x та y. З умови імовірностної еквівалентності маємо F(x)= F(y), таким чином

$$F(x) = F(y) = \int_{y_{min}}^{y} p(y)dy = \frac{y - y_{min}}{y_{max} - y_{min}}.$$

Звідси маємо

$$y = (y_{max} - y_{min})F(x) + y_{min}$$

Залишилося оцінити інтегральний закон розподілу F(x). Але це вже зовсім не складно – знайдемо гістограму зображенні і проведемо її нормапізацію, поділивши величину кожного біну на загальну кількість пікселів, тоді значення бінів можна розглядати як наближене значення функції щільності розподілу $p_i(i = 0,1, \dots ,255).$ Наближене значення функції розподілу можна записати наступним чином

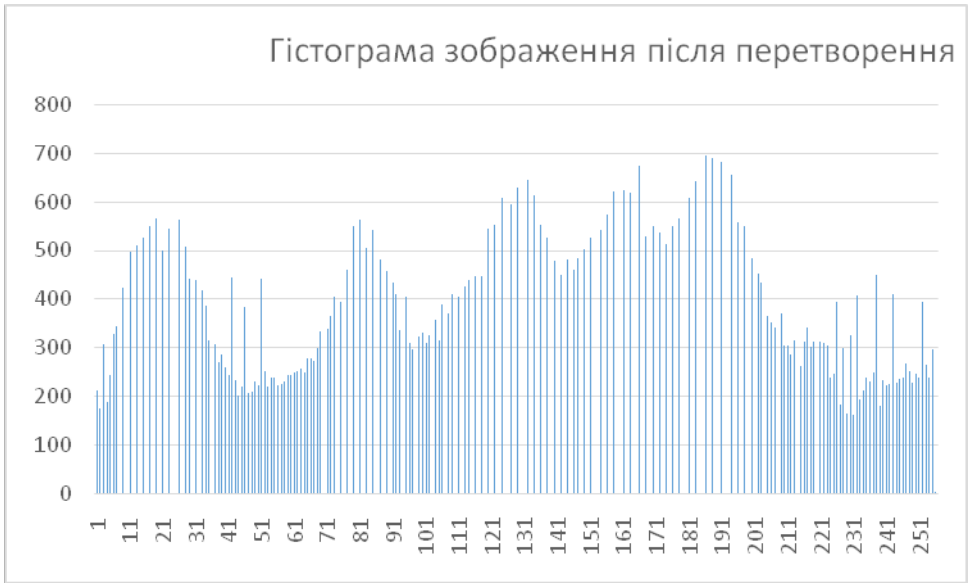
$$\tilde{F}(x) = \sum_{i=1}^{x} p_i.$$

Наведемо формалізацію еквалізації гістограми. На вході зображення в градаціях сірого (яскравість, освітлення, компонента Y простору кольорів YCrCb)

1. Знаходимо гістограму зображення h.
2. Будуємо функцію розподілу $F(x) = \sum_{i=1}^{x} h_i$ за гістограмою.
3. Обновляємо пікселі зображення за правилом

$$f(x) = round\left(255 \times \frac{F(x) - F_{min}}{W \times H - 1}\right)$$

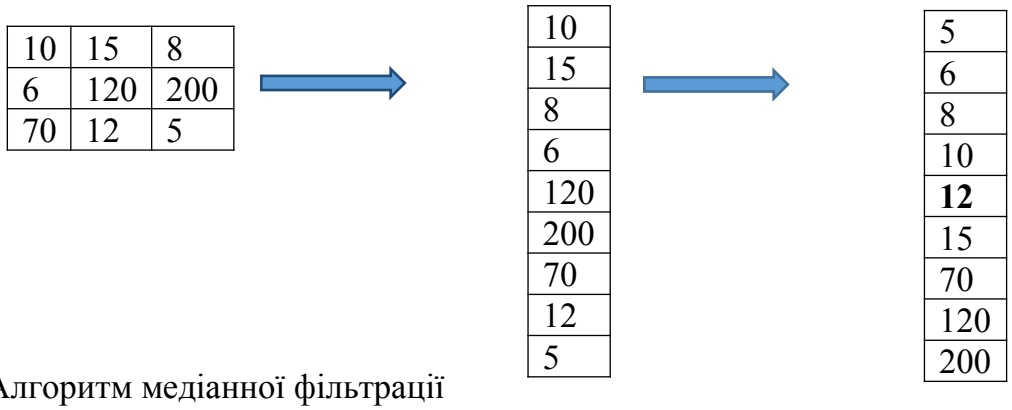
У разі повнокольорового зображення проводимо еквалізацію гістограми компоненти Y, а по отриманим значенням компоненти освітлення і оригінальним хроматичним складовим збираємо нове зображення.
Результат такої процедури для зображення разом з гістограмами вхідного і перетвореного зображень наведені.



Імпульсні шуми і заходи зменшення їх впливу

Медіанна фільтрація є ефективним способом придушення імпульсних шумів, які, зокрема, неминуче з'являються в цифрових камерах в умовах малого освітлення сцени. Алгоритм медіанної фільтрації є масковим:

- Для кожної точки вхідного зображення береться деяка область (наприклад, 3 ≗ 3).
- Точки даної області сортуються за зростанням яскравості.
- Серединна (медіанна) точка (5-я для фільтра 3 ≗ 3) відсортованої множини записується в підсумкове зображення.
- На наступному кроці вікно пересувається на один відлік і обчислення повторюються. Крайні значення масиву дублюються стільки раз, щоб можна було застосувати вікно до першого і до останнього значенням.



Алгоритм медіанної фільтрації

Алгоритм досить ресурсоємкий: так, наприклад, при обробці зображення в градаціях сірого медіанним фільтром 3 ≗ 3 потрібно близько 50 операцій на одну точку зображення. Але в той же час він оперує тільки з 8-бітними числами і йому для роботи потрібно порівняно небагато вхідних даних, що робить алгоритм досить простим. Зауважимо, що, медіанна фільтрація згладжує зображення.

Просторові методи фільтрації з використанням згортки

Для випадку двох змінних згортка неперервного сигналу $x(t,\tau)$ з імпульсною характеристикою (ядром перетворення) $h(t,\tau)$ має вигляд

$$y(t,\tau) = h(t,\tau) \otimes x(t,\tau) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\xi,\vartheta)x(\xi - t,\vartheta - \tau)d\xi d\vartheta.$$

Розглянемо поняття згортки для випадку дискретних сигналів.

Нехай маємо сигнал $X = \{x_{i,j}\}$ і ядро (маску) перетворення (імпульсна характеристика) представляє собою матрицю розміром $(N+1)\times(N+1)$, тобто $H = \{h_{i,j}\}_{i,j=-N}^{N}$, тодізгорткою $X \otimes H$ будемо називати матрицю $Y = \{y_{i,j}\}$, таку, що

$$y_{i,j} = \sum_{\nu,\mu=-N}^{N} h_{\nu,\mu} x_{i+\nu,j+\mu}.$$

Просторова фільтрація зображень складається з наступних дій:

1. Визначення центральної точки (x, y) околу;

2. Здійснення операції, яка використовує лише значення пікселів з заздалегідь обумовленої області навколо центральної точки;
3. Призначення результату цієї операції центральній точці;
4. Повторення всього процесу для кожної точки зображення. В результаті переміщення позиції центральної точки утворюються нові області, що відповідають кожному пікселю зображення.

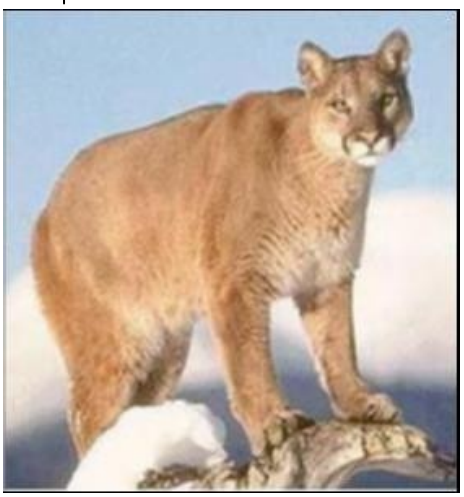
Іншими словами, просторовою фільтрацією називається процес отримання згортки зображення з ядром фільтру.

Лінійна фільтрація зображень в просторовій області полягає в обчисленні лінійної комбінації значень яскравості пікселів у вікні фільтрації з коефіцієнтами матриці ваг, званої маскою або ядром лінійного фільтру.

Найпростішим видом лінійної віконної фільтрації в просторовій області є ковзаюче вікно середніх значень. Результатом такої фільтрації є значення математичного очікування, обчислене по всім пікселям вікна. Математично це еквівалентно згортці з ядром, всі елементи якої дорівнюють 1/n, де n - число елементів ядра, тобто для випадку ядра розміром 3×3 має вигляд:

$$\frac{1}{9}\begin{vmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{vmatrix}$$

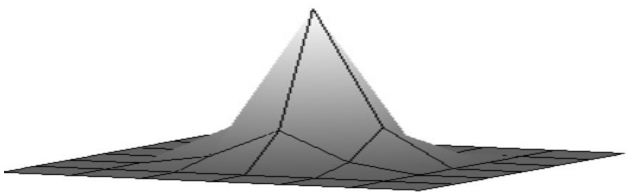
Оригінальне зображення Cougar



Згладжене зображення Cougar

Цей варіант можна розглядати як «вироджений» випадок лінійної фільтрації з однорідним ядром. Оскільки результат усереднення привласнюється центральному пікселу, доцільно надати більш близьким точкам його околу більший вплив на остаточний результат, ніж більш далеким. Прикладом реалізації цієї ідеї для вікна розміром 3×3 є наступний фільтр:

$$\frac{1}{16}\begin{vmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{vmatrix}$$



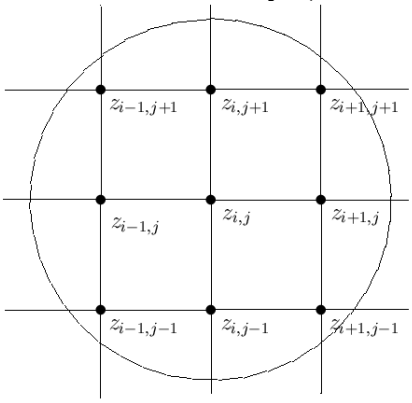
Фільтри такого роду називаються Гаусовими, їх коефіцієнти є дискретизацією функції Гауса з різними параметрами. Прикладом фільтру Гауса розміром 5×5 є матриця

$$\frac{1}{571}\begin{vmatrix} 2 & 7 & 12 & 7 & 2 \\ 7 & 31 & 52 & 31 & 7 \\ 12 & 52 & 127 & 52 & 12 \\ 7 & 31 & 52 & 31 & 7 \\ 2 & 7 & 12 & 7 & 2 \end{vmatrix}$$

Важливо щоб згладжуючі або фільтруючі лінійні фільтри мали суму всіх елементів рівну 1, для чого кожне з вагових значень потрібно розділитина суму всіх вагових коефіцієнтів. Дана умова нормування гарантує адекватний відгук фільтра на постійний сигнал (постійне зображення), тобто збереження середньої яскравості зображення.

Розглянемо одну просту конструкцію згладжування. Вважатимемо, що дані про поверхню задані значеннями $z_{i,j}$ у вузлах решітки *(ih,jh)* однозв'язної області *D*.

Для побудови алгоритмів згладжування, разом з кожним значенням $z_{i,j}$, що лежить в середині області *D*, потрібні сусідні значення $z_{i-1,j-1}$, $z_{i-1,j}$, $z_{i,j-1}$, $z_{i+1,j-1}$, $z_{i-1,j+1}$, $z_{i+1,j+1}$, $z_{i+1,j}$, $z_{i,j+1}$, тобто значення поверхні, відповідні *r*-околу ($\sqrt{2}h \le r \le 2h$) точки *(ih,jh)*.



У випадку, якщо якесь з цих значень не задане, слід провести поповнення даних.

Нехай $P(x,y) = ax^2 + by^2 + cxy + dx + ey + f$ квадратична функція двох змінних. Виберемо коефіцієнти *a, b, c, d, e* і *f* квадратичної функції з умови мінімізації суми квадратів похибок, тобто з умови мінімуму величини

$$\sum_{\nu=i-1}^{i+1}\sum_{\mu=j-1}^{j+1}\left(z_{\nu,\mu} - P(\nu h, \mu h)\right)^2 .$$

В якості згладженого значення, будемо брати значення екстремальної квадратичної функції в точці *(ih,jh)*. Для квадратичної функції необхідні умови мінімуму співпадають з достатніми. Для визначення коефіцієнтів екстремальної функції необхідно і достатньо узяти часткові похідні від *P(x,y)* і прирівняти їх донуля. При цьому одержимо систему шести лінійних рівнянь з шістьма невідомими. Вирішуючи її, одержуємо квадратичну функцію

$$P^*(x,y) = a^*(x - ih)^2 + b^*(y - jh)^2 + c^*(x - ih)(y - jh) + d^*(x - ih) + e^*(y - jh) + f^*,$$

де

$$a^* = \frac{1}{6h^2}\left(z_{i-1,j-1} + z_{i-1,j} + z_{i-1,j+1} + z_{i+1,j-1} + z_{i+1,j} + z_{i+1,j+1} - 2(z_{i,j-1} + z_{i,j} + z_{i,j+1})\right),$$

$$b^* = \frac{1}{6h^2}\left(z_{i-1,j-1} + z_{i,j+1} + z_{i-1,j+1} + z_{i+1,j-1} + z_{i,j-1} + z_{i+1,j+1} - 2(z_{i-1,j} + z_{i,j} + z_{i+1,j})\right),$$

$$c^* = \frac{1}{6h^2}\left(z_{i-1,j-1} + z_{i+1,j+1} - z_{i-1,j+1} - z_{i+1,j-1}\right),$$

$$d^* = \frac{1}{6h}\left(z_{i+1,j+1} + z_{i+1,j} + z_{i+1,j-1} - z_{i-1,j+1} - z_{i-1,j} - z_{i-1,j-1}\right),$$

$$e^* = \frac{1}{6h}\left(z_{i+1,j+1} + z_{i-1,j+1} + z_{i,j+1} - z_{i-1,j-1} - z_{i,j-1} - z_{i-1,j-1}\right),$$

$$f^* = z_{i,j} + \frac{2}{9}\left(z_{i-1,j} + z_{i,j+1} + z_{i+1,j} + z_{i,j-1} - 4z_{i,j}\right) - \frac{1}{9}\left(z_{i-1,j+1} + z_{i+1,j+1} + z_{i+1,j-1} + z_{i-1,j-1} - 4z_{i,j}\right) =$$

$$z_{i,j} + \frac{1}{9}\left(2\Delta^2 z_{i,j} - \tilde{\Delta}^2 z_{i,j}\right).$$

Таким чином, згладженим значенням можна рахувати значення $\tilde{z}_{i,j}$, що визначається рівністю

$$\tilde{z}_{i,j} = z_{i,j} + \frac{1}{9}\left(2\Delta^2 z_{i,j} - \tilde{\Delta}^2 z_{i,j}\right),$$

де $\Delta^2 z_{i,j} = z_{i-1,j} + z_{i,j+1} + z_{i+1,j} + z_{i,j-1} - 4z_{i,j}$ і $\tilde{\Delta}^2 z_{i,j} = z_{i-1,j+1} + z_{i+1,j+1} + z_{i+1,j-1} + z_{i-1,j-1} - 4z_{i,j}$

.

В цьому випадку оператор згладжування породжений фільтром

$$S = \left[s_{i,j}\right]_{i,j=-1}^{1} = \frac{1}{9}\begin{bmatrix} -1 & 2 & -1 \\ 2 & 5 & 2 \\ -1 & 2 & -1 \end{bmatrix}.$$

Методи зміни різкості зображень

Раніше були розглянуті методи просторової фільтрації, які призводили до зглажування зображень. Тепер розглянемо зворотну задачу – підвищити контрастність, різкість зображення.

Методи, які використовують другі дискретні різниці

Різкість зображення (його контраст) характеризується ступенем помітності (розрізнення) перепадів яскравості сусідніх ділянок зображення і якщо зменшення різкості (згладжування) досягається шляхом застосування низькочастотних фільтрів, то зворотна процедура - підвищення різкості пов'язана з використанням високочастотних фільтрів.

Візуально підвищення різкості призводить до підкреслення дрібних деталей і контурів зображення або поліпшення розрізнення тих його деталей, які опинилися розфокусованими з якоїсь причини.

У подальшому нам знадобиться перша дискретна різниця - аналог першої похідної:

$$\frac{\partial s}{\partial x} \to \Delta s[x+1] = \mathrm{s[x+1]} - \mathrm{s[x]},$$

та друга дискретна різниця – аналог другої похідної:

$$\frac{\partial^2 s}{\partial x^2} \to \Delta^2 s[x+1] = \Delta \mathrm{s[x+1]} - \Delta\,\mathrm{s[x]} = \mathrm{s[x+1]} - 2\mathrm{s[x]} + \mathrm{s[x-1]},$$

які широко застосовуються в методах посилення різкості зображень. Зокрема, на їх використанні заснований оператор Лапласа (лапласіан), який для функції двох змінних для аналогових сигналів визначається як

$$\nabla^2 s = \frac{\partial^2 s}{\partial x^2} + \frac{\partial^2 s}{\partial y^2}.$$

У простійшому випадку лапласіан дає ядро згортки

$$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{vmatrix}$$

але для підвищення різкості краще використовувати ядро

$$\begin{vmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{vmatrix}$$

або ядро 5 ≅ 5

$$\begin{vmatrix} -1 & -3 & -4 & -3 & -1 \\ -3 & 0 & 6 & 0 & -3 \\ -4 & 6 & 20 & 6 & -4 \\ -3 & 0 & 6 & 0 & -3 \\ -1 & -3 & -4 & -3 & -1 \end{vmatrix}$$

В цілому алгоритм використання лапласіану для підвищення різкости зображення має вигляд:

$$\bar{s}[x,y] = \begin{cases} s[x,y] - \nabla^2 s[x,y]/k, & \text{если} \quad w[0,0] \le 0, \\ s[x,y] + \nabla^2 s[x,y]/k, & \text{если} \quad w[0,0] > 0, \end{cases}$$

Результат обробки зображення із застосуванням оператора Лапласу, тут w[0,0] - значення центрального коефіцієнта маски Лапласіану.

	
Оригінальне зображення Cougar	Результат застосування Лапласіана

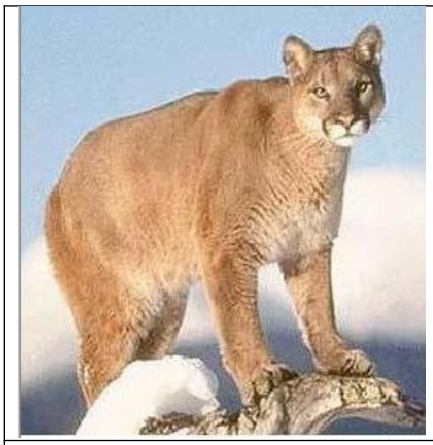
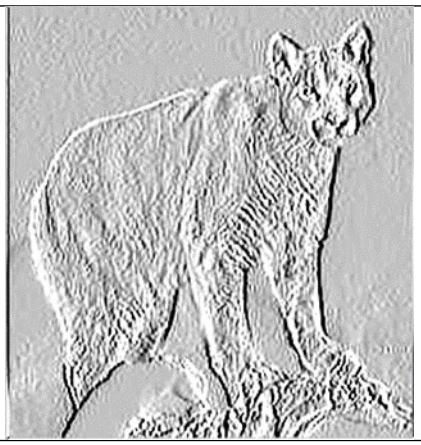
Градієнтні методи

Крім аналогів других похідних, яким є лапласіан(оператор Лапласа), для підвищення різкості зображення можуть бути використані і оператори першого порядку або градієнтні оператори. До них відносяться оператори Робертса, Канні, Собеля і Превіта. Вони призначені, в основному, для виділення контурів. Ця процедура є попередньою, крім підвищення різкості зображення, і в інших завданнях обробки, зокрема, в задачі розпізнавання об'єктів.

Принцип роботи градієнтних операторів зручно розглянути на прикладі алгоритму Собеля. Оператор обчислює градієнт яскравості зображення в кожній точці. Так знаходиться напрямок найбільшого збільшення яскравості і величина її зміни в цьому напрямку. Результат показує, наскільки різко або плавно змінюється яскравість зображення в кожній точці, а значить, ймовірність знаходження точки на границі і орієнтацію границі.

Градієнт функції двох змінних для кожної точки зображення (зокрема і функції яскравості) - двовимірний вектор, компонентами якого є похідні яскравості зображення по горизонталі і вертикалі. У кожній точці зображення вектор орієнтований в напрямку найбільшої зміни яскравості, а його довжина відповідає величині зміни яскравості. Оператор використовує ядро $3 \triangleq 3$ і з ним згортається вихідне зображення. Нехай A- вхідне зображення, а Gx і Gy - дві матриці такого ж розміру як і зображення, кожна точка яких є похідні по x і y

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \otimes A, G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \otimes A,$$

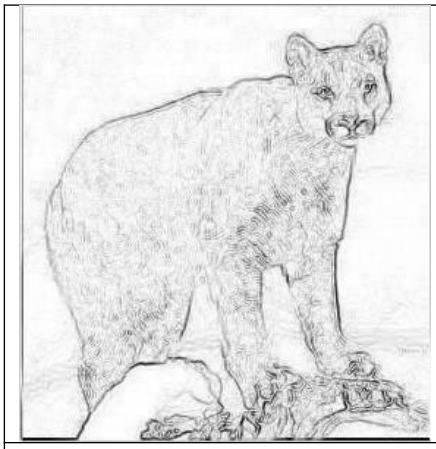
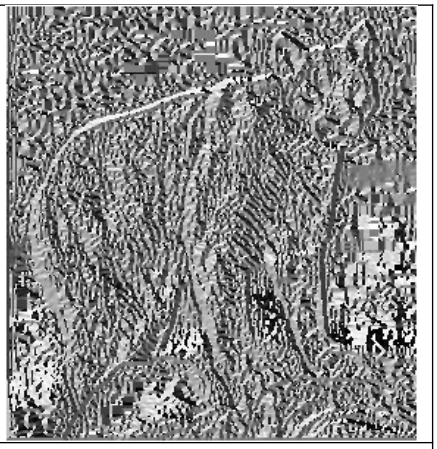
		
Оригінальне зображення Cougar	Мапа змінності освітлення по горизонталі	Мапа змінності освітлення по вертикалі

де $\sqsubset$ оператор згортки. У кожній точцізображення можназнайти величину градієнту

$$G = \sqrt{G_x^2 + G_y^2}$$

та його напряму

$$\theta = arctg\left(\frac{G_y}{G_x}\right).$$

	
Інверсне зображення величини градієнта Cougar	Мапа нормованного направлення градієнта Cougar

Нерізке маскування

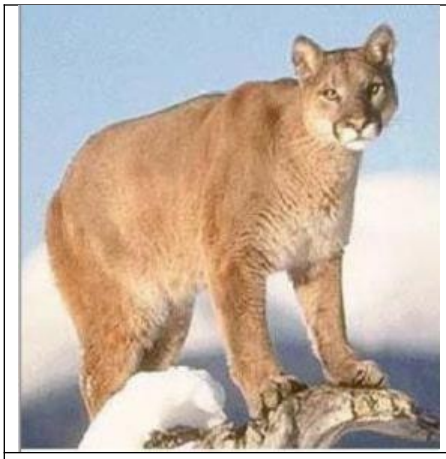
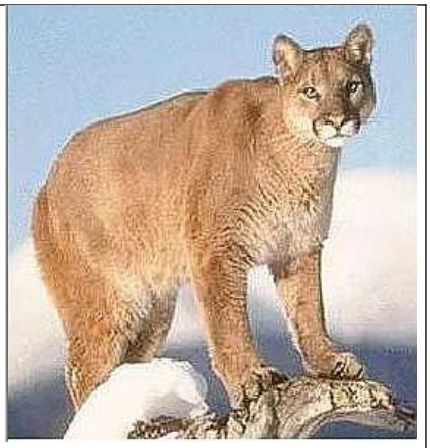
Нерізке маскування - технологічний прийом обробки зображення, який дозволяє домогтися ефекту збільшення різкості за рахунок посилення контрасту тональних переходів. У ході процедури відбувається об'єднання трохи розпливчатою (нерізкої) версії зображення з оригіналом. В результаті виходять різкі деталі в високо контрастних областях без посилення тонових стрибків в областях з низькою контрастністю там, де різкі тонові скачки можуть порушити плавність переходів.

Фільтрація з підйомом високих частот є узагальненням процедури нерізкого маскування. Ця операція зводиться до застосування правила контрастування виду

$$h(x, y)= kf(x, y) - \nabla^2 f(x, y),$$

за умови, що ваговий коефіцієнт у центрі фільтру менше нуля, інакше знак перед лапласіаном змінюється на протилежний. У такій ситуації підвищення різкості зображення може проводитися із застосуванням фільтрів

$$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 4+k & -1 \\ 0 & -1 & 0 \end{vmatrix} \quad \textit{або} \quad \begin{vmatrix} -1 & -1 & -1 \\ -1 & 8+k & -1 \\ -1 & -1 & -1 \end{vmatrix}$$

		
Оригінальне зображення Cougar	Застосування нерізкого маскування з маскою M_1	Застосування нерізкого маскування з маскою M_2

Зауважимо, що з ростом коефіцієнта k, ефект підвищення різкості зменшується. Виконання нерізкого маскування і фільтрації з підйомом високих частот вимагає застосування умови нормування, або застосування градаційній корекції одержуваних значень яскравості відклику h(x,y), оскільки ці значення можуть виходити за межі допустимих значень. Застосування фільтрів нерізкого маскування також призводить до посилення шумів зображення, до формування помітних ореолів уздовж границь, а також до нерівномірного збільшення локального контрасту вздовж границі. Алеж, цими недоліками можна охарактеризувати всі фільтри різкості, засновані на використанні похідних.

Побудова методів відновлення зображення

Ясна річ, використання будь-якого методу фільтрації тягне за собою і зворотну операцію – відновлення зображення до оригінального вигляду, так операція згладжування породжує операцію контрастування і навпаки. Далі нам знадобиться одне твердження.

Через ℓ_2^2 позначимо лінійний простір всіх обмежених двовимірних послідовностей (масивів)

$A=\{a_{i,j}\}_{(i,j)\in Z^2}$ з нормою $\|A\|=\left(\sum_{i,j\in Z}a_{i,j}^2\right)^{1/2}$ і покладемо $|A|=\left|\sum_{i,j\in Z}a_{i,j}\right|$.

Хай L лінійний оператор, який відображає простір X у простір Y. Як завжди, нормою оператора L називатимемовеличину $\|L\|_{X\to Y}=\sup\{\|L(F)\|_Y \mid \|F\|_X\le 1\}$.

Позначимо через C=A⊏ B згортку масивів (матриць) A і B, тобто масив C такий, що

$$c_{i,j}=\sum_{\nu,\mu}a_{\nu,\mu}b_{\nu-i,\mu-j}.$$

Розглянемо рівняння E⊏ X=F, де |E|≠ 0, тоді $\frac{E}{|E|}\otimes X=\frac{F}{|E|}$, або, що те ж, $\widetilde{E}\otimes X=\Phi$, де

$$\widetilde{E}=\frac{E}{|E|}\ \text{і}\ \Phi=\frac{F}{|E|}$$

Основою для побудови відновлюючих фільтрів є наступне твердження (див.[1]).

Теорема 1. *Якщо для будь-якого натурального n виконується нерівність*

$$\left\|I-\widetilde{E}\right\|_{\ell_2^2\to\ell_2^2}<1, \tag{2.1}$$

то має місце співвідношення

$$X_n=C_n^1\Phi-C_n^2\widetilde{E}\otimes\Phi+C_n^3\widetilde{E}^2\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^n\otimes\Phi+\varepsilon_n, \tag{2.2}$$

де ε_n таке, що

$$\widetilde{E}\otimes\varepsilon_n=(I-\widetilde{E})^{n+1}\otimes\Phi. \tag{2.3}$$

Дійсно, якщо

$$X_n=C_n^1\Phi-C_n^2\widetilde{E}\otimes\Phi+C_n^3\widetilde{E}^2\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^n\otimes\Phi+\varepsilon_n,$$

то

$$\widetilde{E}\otimes X_n=\widetilde{E}\otimes\left(C_n^1\Phi-C_n^2\widetilde{E}\otimes\Phi+C_n^3\widetilde{E}^2\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^n\otimes\Phi+\varepsilon_n\right)=$$
$$C_n^1\widetilde{E}\otimes\Phi-C_n^2\widetilde{E}^2\otimes\Phi+C_n^3\widetilde{E}^3\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^{n+1}\otimes\Phi+\widetilde{E}\otimes\varepsilon_n.$$

Звідси, і з (2.3) відразу одержуємо

$$\widetilde{E}\otimes X_n=C_n^1\widetilde{E}\otimes\Phi-C_n^2\widetilde{E}^2\otimes\Phi+C_n^3\widetilde{E}^3\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^{n+1}\otimes\Phi+(I-\widetilde{E})^{n+1}\otimes\Phi=$$
$$C_n^1\widetilde{E}\otimes\Phi-C_n^2\widetilde{E}^2\otimes\Phi+C_n^3\widetilde{E}^3\otimes\Phi+\ldots+(-1)^nC_n^n\widetilde{E}^{n+1}\otimes\Phi+$$
$$\Phi-C_n^1\widetilde{E}\otimes\Phi+C_n^2\widetilde{E}^2\otimes\Phi-C_n^3\widetilde{E}^3\otimes\Phi+\ldots+(-1)^nC_n^{n+1}\widetilde{E}^{n+1}\otimes\Phi=\Phi.$$

Зазначимо, що, якщо виконується умова (1), то

$$\|\varepsilon_m\|\le\left|\widetilde{E}^{-1}\right|\left\|I-\widetilde{E}\right\|^{n+1}|\Phi|,$$

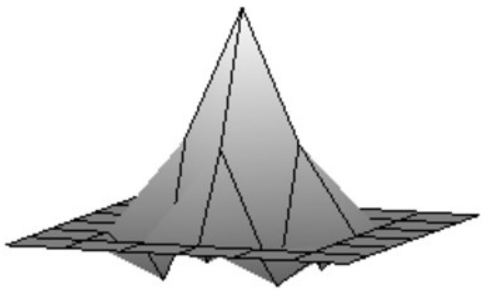
і, в цьому випадку, рішення можна записати в операторному вигляді

$$X=\lim_{n\to\infty}\left(I-(I-\widetilde{E})^n\right)\Phi.$$

Використовуємо цей результат для побудови оператора контрастування, що повертає зображення, змінене в результатізастосування оператора згладжування.

Як приклад розглянемо побудову контрастуючого фільтру псевдозворотного до згладжуючого фільтру S, побудованого раніше:

$$S=\left[s_{i,j}\right]_{i,j=1}^{1}=\frac{1}{9}\begin{bmatrix}-1&2&-1\\2&5&2\\-1&2&-1\end{bmatrix}$$

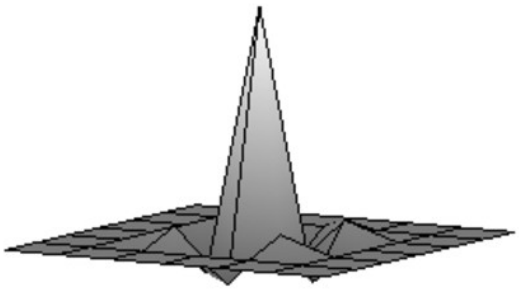


Відповідний контрастуючий фільтр S^{-1} повинен задовольняти умові $S \otimes S^{-1} = I$, де

$$I = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

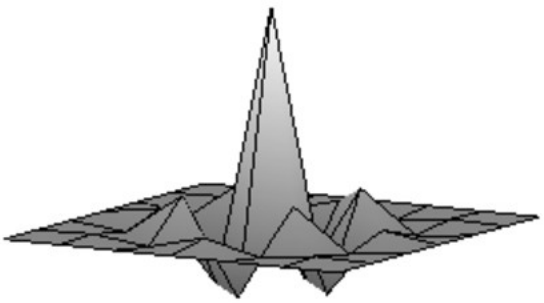
Застосовуючи формулу (2.2), отримуємо наближене значення S^{-1}. Перша ітерація дає контрастуючий фільтр

$$S^{-1,1} = 2I - S \otimes I = \frac{1}{9} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 13 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$



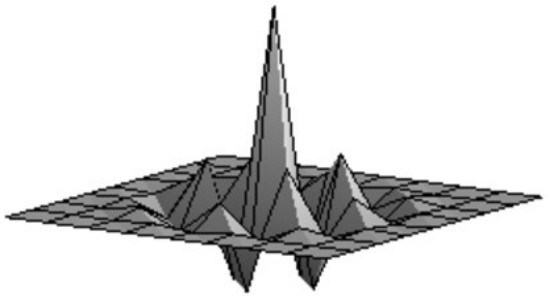
Застосування другої ітерації дозволяє побудувати контрастуючий фільтр, що повертає згладжені дані з вищою точністю

$$S^{-1,2} = 3I - 3S \otimes I + S^2 \otimes I = \frac{1}{81} \begin{bmatrix} 1 & -4 & 6 & -4 & 1 \\ -4 & 25 & -42 & 25 & -4 \\ 6 & -42 & 153 & -42 & 6 \\ -4 & 25 & -42 & 25 & -4 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}$$



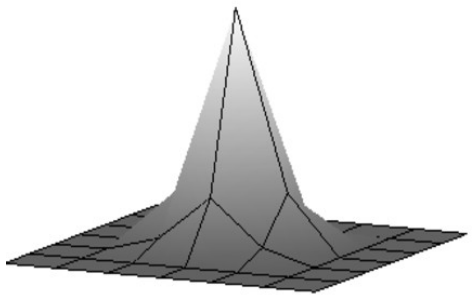
Результат третьої ітерації дає фільтр

$$S^{-1,3} = 4I - 6S \otimes I + 6S^2 \otimes I - S^3 \otimes I = \frac{1}{729}\begin{bmatrix} 1 & -6 & 15 & -20 & 15 & -6 & 1 \\ -6 & 45 & -126 & 174 & -126 & 45 & -6 \\ 15 & -126 & 450 & -696 & 450 & -126 & 15 \\ -20 & 174 & -696 & 1777 & -696 & 174 & -20 \\ 15 & -126 & 450 & -696 & 450 & -126 & 15 \\ -6 & 45 & -126 & 174 & -126 & 45 & -6 \\ 1 & -6 & 15 & -20 & 15 & -6 & 1 \end{bmatrix}$$

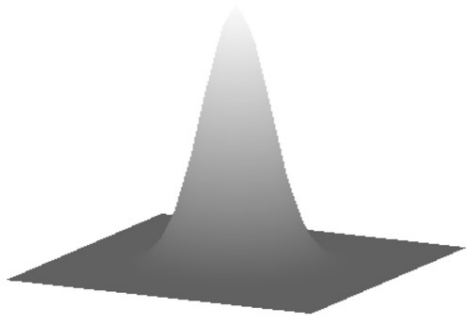


Приведемо контрастуючі фільтри для згладжуючого фільтру

$$G = \frac{1}{36}\begin{bmatrix} 1 & 4 & 1 \\ 4 & 16 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

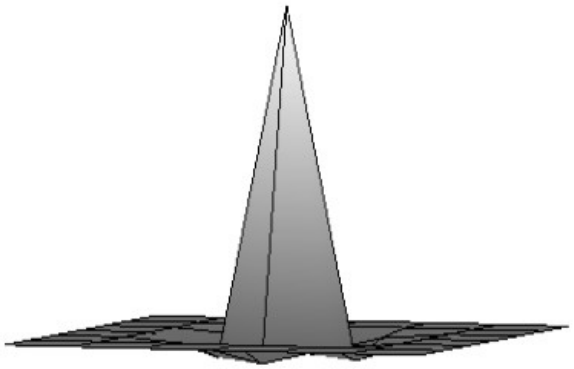


породженого функцією Гауса $z = 2^{-2(x^1+y^2)}$

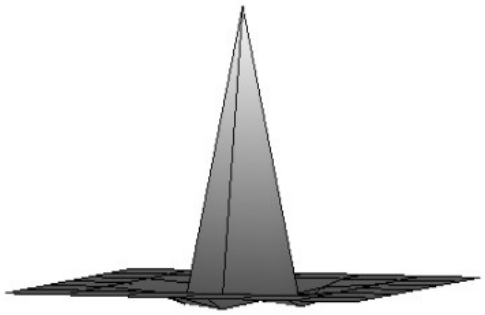


Маємо

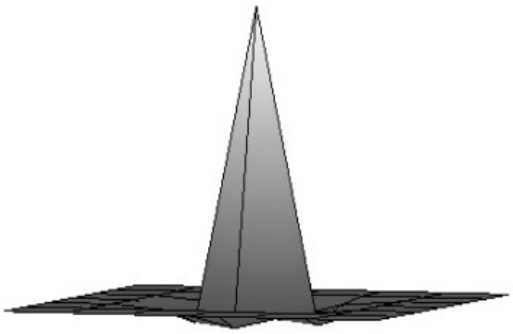
$$G^{-1,1} = \frac{1}{36}\begin{bmatrix} -1 & -4 & -1 \\ -4 & 56 & -4 \\ -1 & -4 & -1 \end{bmatrix}$$



$$G^{-1,2} = 3I - 3G \otimes I + G^2 \otimes I = \frac{1}{1296}\begin{bmatrix} 1 & 8 & 18 & 8 & 1 \\ 8 & -44 & -288 & -44 & 8 \\ 18 & -288 & 2484 & -288 & 18 \\ 8 & -44 & -288 & -44 & 8 \\ 1 & 8 & 18 & 8 & 1 \end{bmatrix}$$



$$G^{-1,3} = 4I - 6G \otimes I + 6G^2 \otimes I - G^3 \otimes I = \frac{1}{46656}\begin{bmatrix} -1 & -12 & -51 & -88 & -51 & -12 & -1 \\ -12 & 0 & 540 & 1536 & 540 & 0 & -12 \\ -51 & 540 & -1161 & -14856 & -1161 & 540 & -51 \\ -88 & 1536 & -14856 & 101120 & -14856 & 1536 & -88 \\ -51 & 540 & -1161 & -14856 & -1161 & 540 & -51 \\ -12 & 0 & 540 & 1536 & 540 & 0 & -12 \\ -1 & -12 & -51 & -88 & -51 & -12 & -1 \end{bmatrix}$$



Приведемо приклад використання контрастуючого фільтрудля зображення superfood

			
PSNR=27.26	PSNR=32.08	PSNR=35.53	PSNR=37.96

Пікове співвідношення сигналу до шуму (англ. peak signal-to-noise ratio) позначається абревіатурою PSNR і є терміном, що означає співвідношення між максимумом можливого значення сигналу (для байтових зображень це 255) та потужністю шуму, що спотворює значення сигналу. PSNR зазвичай вимірюється логарифмічною шкалою в децибелах.

PSNR найчастіше використовується для вимірювання якості реконструкції зображень. Сигнал у цьому випадку є вихідними даними, а шум - це помилка.

Типові значення PSNR для порівняння зображень лежать в межах від 30 до 40 dB. Чим вище значення PSNR, тим менша різниця між зображеннями, що порівнюються.

Для монохромних зображень *I* та *K* розміру H×W, одне з яких вважається зашумленими наближенням іншого, обчислюється наступним чином:

$$PSNR = 20 \times \lg \frac{MAX(I)}{\sqrt{MSE}},$$

де

$$MSE = \frac{1}{H \times W} \sum_{i=1}^{W} \sum_{j=1}^{H} |I_{i,j} - K_{i,j}|^2$$

У разі порівняння повнокольорових зображень, використовуємо сумму MSE по кожній кольоровій компоненті і поділемо на 3.

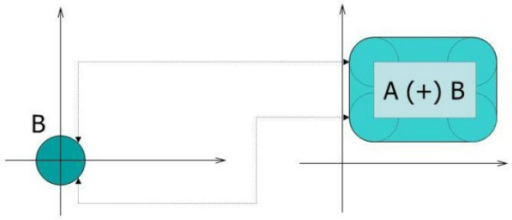
На завершення параграфа, присвяченого створенню просторових фільтрів, розглянемо реалізацію ефекту акварелізації.

Акварельний фільтр перетворить зображення, і, після обробки воно виглядає так, як ніби написано аквареллю. Дляотримання такого ефекту використовується метод медіанного усереднення кольору в кожній точці. Значення кольорукожного піксела і його 24 сусідів поміщаються в список і сортуються від меншого до більшого. Медіанне (тринадцяте)значення кольору в списку привласнюється центральному пікселу. Після цього, для виділення межі переходів кольорів, доодержаного зображення застосовується контрастуючий фільтр. Результуюче зображення нагадує акварельний живопис.

Це лише один приклад, який показує, як можна об'єднувати різні методи обробки зображень і отримувати незвичайнівізуальні ефекти, наприклад, такий

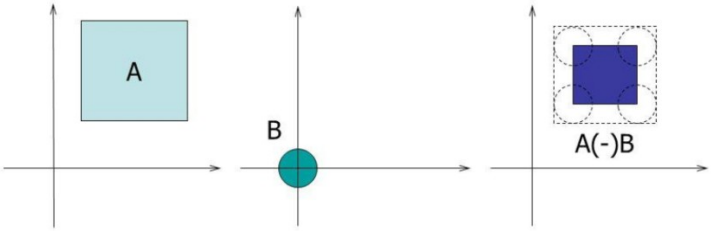
Морфологічна обробка бінарних зображень
Расширення (Dilation)

$$A \oplus B = \{t \in R^2 : t = a + b, a \in A, b \in B\}$$

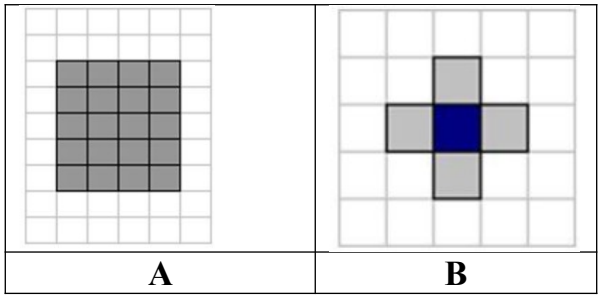


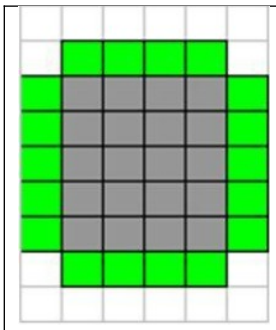
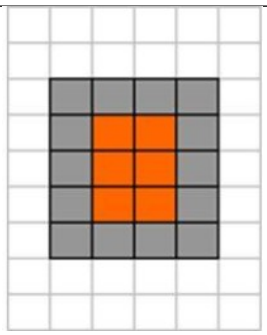
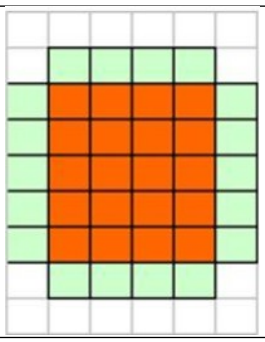
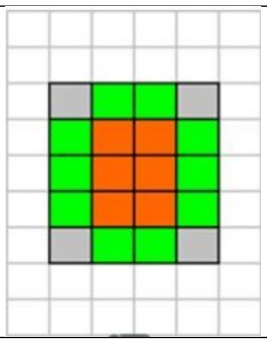
Звуження(Erosion)

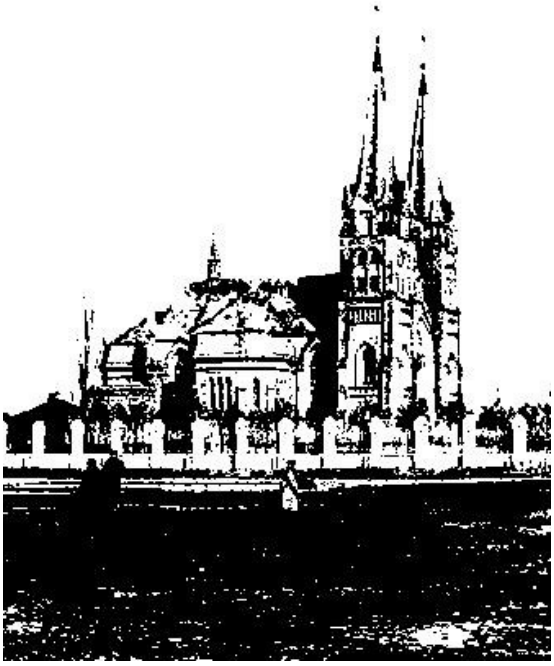
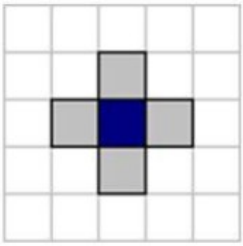
$A \ominus B = (A^c \oplus B^c)^c$, де A^c доповнення A

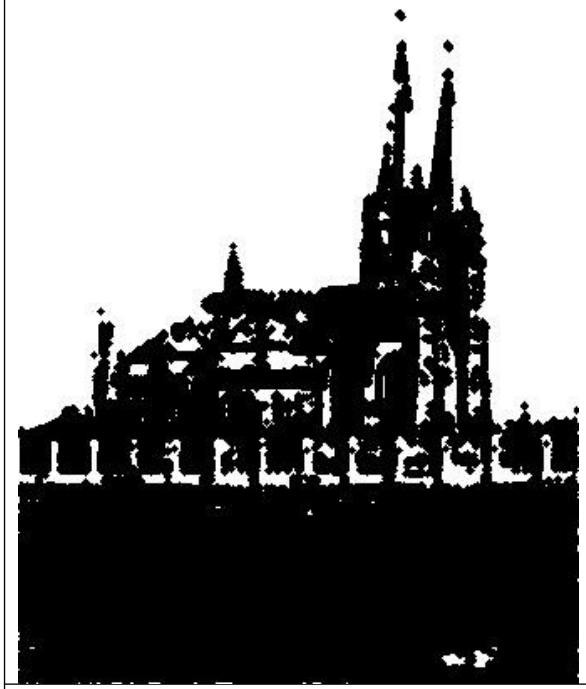
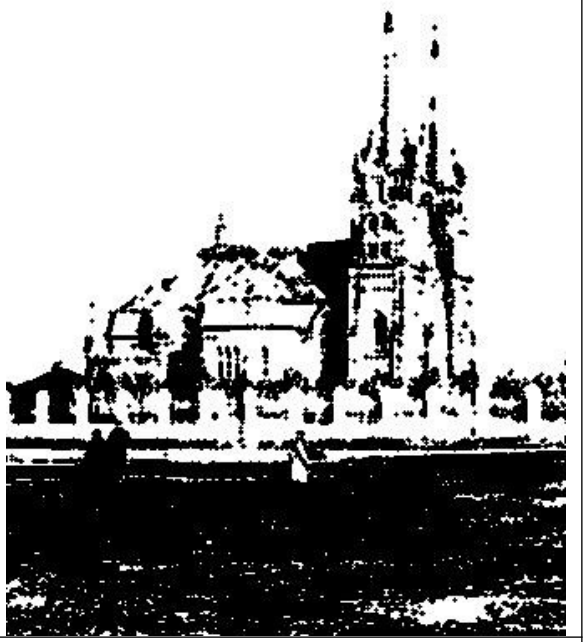


Приклад застосування математичної морфології

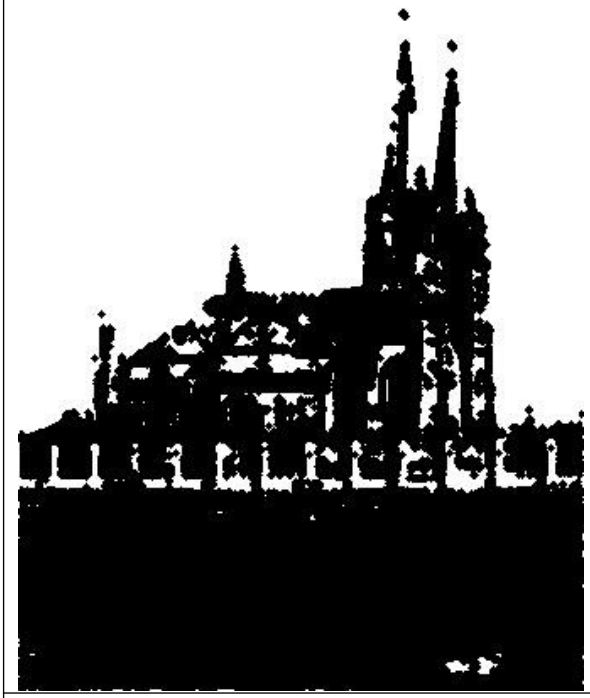
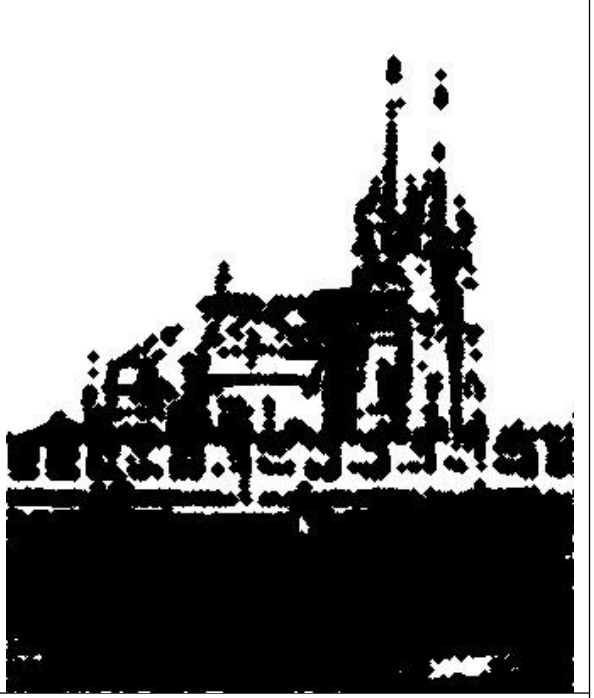


			
A ▫ B Розширення (dilation)	**AϴB Звуження (erosion)**	**(A ▫ B)ϴB Відкриття (open)**	**(AϴB) ▫ B Закриття (close)**

	
A	**B**

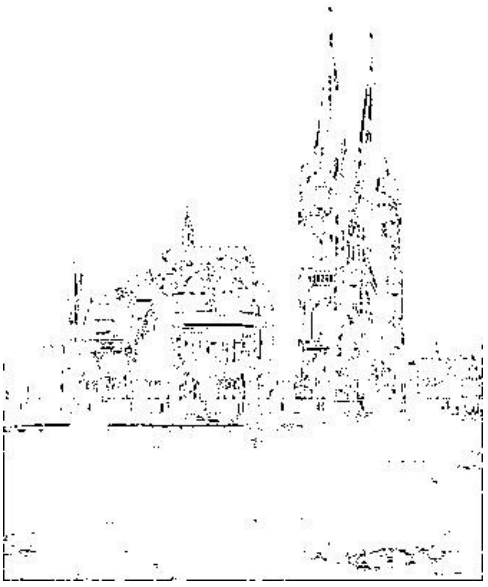
	
A ▫ B розширення	**AϴB звуження**

Використання морфологічної обробки для знаходження контуру

	
(A ⊓ B)⊖B **відкриття**	**(A⊖B) ⊓ B** **закриття**

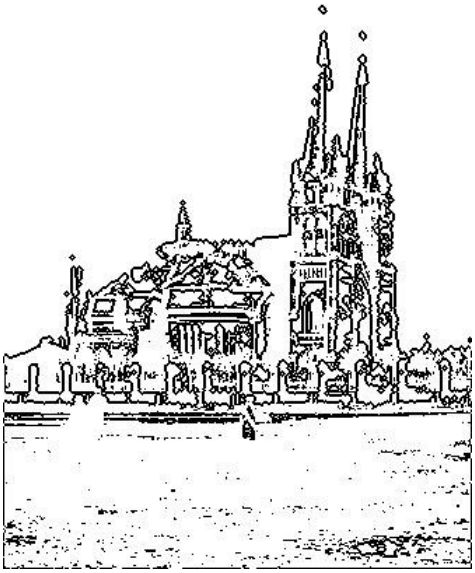
Формування внутрішнього контуру

$$C^- = A - (A \ominus B)$$



Формування зовнішнього контуру

$$C^+ = (A \oplus B) - A$$



Для зображень у градаціях сірого морфологічні операції виглядаютьнаступним чином

$$A(x,y) \oplus B(u,v) = \max_{(u,v)}\{A(x-u.y-v) + B(u,v)\}$$

та

$$A(x,y) \ominus B(u,v) = \min_{(u,v)}\{A(x+u.y+v) - B(u,v)\}$$

Метаморфологічної обробки:

• Зниження шумів.
• Виділення меж об'єкта.
• Виділення кістяка об'єкта.
• Модифікація контурного представлення.

Сегментація зображень

Сегментація поділяє зображення на окремі регіони, що містять пікселі із подібними значеннями. Для того, щоб бути корисними для аналізу та інтерпретації зображень, регіони повинні сильно пов'язуватись із зображеними об'єктами або цікавими (для користувача) ознаками. Технології сегментації можуть бути або контекстними, або неконтекстними. Останні не враховують просторових зв'язків між функціями зображення та групування пікселів разом на основі деякого глобального атрибуту, наприклад, одного рівня або кольору. Контекстні методи додатково використовують ці зв'язки, наприклад, об'єднують пікселі з подібними рівнями та близьким просторовим розташуваннями.

k-means

Даний метод сегментації є неконтекстним і зображення сегментується по близкості кольорів RGB. Мірою близкості є евклідова міра. В основі даного підходу лежить метод кластеризації k-середніми (k-means) (див., наприклад, [2]).

Ідея методук-середніх полягає в наступному - спочатку вибирається k довільних початкових центрів з множини $\Im$. Далі всі об'єкти розбиваються на k груп, найбільш близьких до відповідного центру. На наступному кроці обчислюються центри

знайдених кластерів. Процедура повторюється ітераційно до тих пір, поки центри кластерів не стабілізуються.

Алгоритм розбиття об'єктів $x_i(i = 0,1, \dots, n)$ заснований на мінімізації межкластерної відстані, у разі, якщо в якості відстані використовується середньоквадратична норма ℓ_2, тобто цільовою функцією є

$$J = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2$$

де x_i-і -й об'єкт, а C_j являє собою j -й кластер із центром μ_j .

Структура алгоритму полягає в наступному:
Для ініціалізації алгоритму вибираємо k центрів кластерів.
Кожному з n об'єктів ставимо у відповідність кластер, виходячи з мінімізації ℓ_2 норми між об'єктом і центром відповідного кластера.
Перераховуємо центри знову отриманих кластерів.
Для вирішення цього завдання серед усіх елементів кластера $x \in C_j$ знайдемо елемент μ_j, який мінімізує ухилення $\sum_{x_i \in C_j} \|x_i - \mu_j\|^2$, для чого знайдемо розв'язок задачі

$$\frac{\partial}{\partial \mu_j} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2 = \frac{\partial}{\partial \mu_j} \sum_{x_i \in C_j} \left(\|x_i - \mu_j\|^2 - 2x^T \mu_j + \|\mu_j\|^2 \right) = \sum_{x_i \in C_j} (x_i - \mu_j) = 0,$$

тобто

$$\mu_j = \frac{1}{n_j} \sum_{x_i \in C_j} x_i$$

Для кожного i , такого, що $x_i \in C_j$ обчислимо

$$h = argmin \left\{ \frac{n_j \|x_i - \mu_j\|}{n_j - 1} \right\}$$

де n_j число об'єктів кластера C_j.
Якщо виконується умова

$$\frac{n_h \|x_i - \mu_h\|}{n_h - 1} < \frac{n_j \|x_i - \mu_j\|}{n_j - 1},$$

то слід перемістити об'єкт xi з кластера C_j в кластер C_h , після чого перерахувати значення центрів кластерів.
Якщо i<n , то переходимо до кроку 4, інакше до кроку 3.
Критерієм зупинки алгоритму може служити або досягнення заданого числа ітерацій алгоритму, або досягнення функцією цілі заданого значення порогу.
Унашому випадку для двох пік селів

$$X(X_{red}, X_{green}, X_{blue})$$

Та

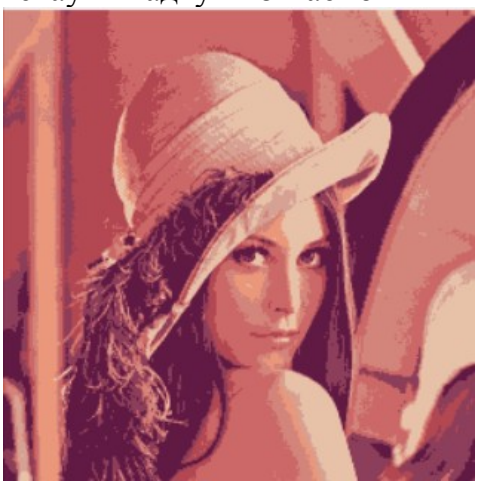
$$Y(Y_{red}, Y_{green}, Y_{blue})$$

відстань між ними дорівнює

$$\|X - Y\|^2 = (X_{red} - Y_{red})^2 + (X_{green} - Y_{green})^2 + (X_{blue} - Y_{blue})^2$$

і колір пікселів всього кластеру C_j співпадає з кольором центру кластера μ_j.

Для тестового зображення Lenay випадку k=8 маємо



ЕМ-алгоритм

В основі даного неконтекстного методу сегментації лежить аналіз суміші розподілів

$$p(x) = \sum_{i=1}^{k} w_i p_i(x)$$

у припущенні, що відомий тільки загальний вигляд розподілу ймовірності і потрібно оцінити його параметри. Для вирішення цієї задачі використовується ЕМ-алгоритм (див., наприклад, [2]).ЕМ-алгоритм для відомого фіксованого числа компонентів суміші можна записати в наступному вигляді.

Нехай $X = \{x_1, \dots, x_n\}$ вибірка спостережень, k число компонентів суміші, $\Theta = \{(w_i, \theta_i)\}_{i=1}^{k}$- початкове наближення параметрів суміші, і ε число, що визначає зупинку алгоритму. ЕМ-алгоритм складається з послідовного застосування двох кроків.

Е-крок (expectation).

$$g_{i,j}^0 = g_{i,j}$$

$$g_{i,j} = \frac{w_i p_i(x_j)}{\sum_{\theta=1}^{k} w_\theta p_\theta(x_j)}, i = 1, \dots, k, j = 1, \dots, n.$$

$$\delta = max\{|g_{i,j}^0 - g_{i,j}|\}$$

М-крок (maximization).

$$\sum_{j=1}^{n} g_{i,j} \ln p_i(x_j) \to \max_{\Theta}, i = 1, \dots, k,$$

$$w_i = \frac{1}{n} \sum_{j=1}^{n} g_{i,j}, i = 1, \dots, k.$$

якщо δ>ε, то переходимо до Е-кроку, якщо δ≤ ε, то повертаємо знайдені параметри суміші , $\Theta = \{(w_i, \theta_i)\}_{i=1}^{k}$.

У разі суміші нормальних розподілів $N(\mu_i, \sigma_i^2)$ Е-крок буде виглядати наступним чином

$$g_{i,j} = \frac{w_i p_i(x_j)}{\sum_{\vartheta=1}^{k} w_\vartheta p_\vartheta(x_j)}, i = 1, \dots, k, j = 1, \dots, n$$

де

$$p(x|\mu_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right).$$

Для М-кроку задача

$$\sum_{j=1}^{n} g_{i,j} \ln p_i(x_j) \to \max_{\Theta}, i = 1, \dots, k,$$

запишеться у наступному вигляді

$$\sum_{j=1}^{n} g_{i,j} \ln\left(\frac{1}{\sigma_i\sqrt{2\pi}} exp\left(-\frac{(x_j-\mu_i)^2}{2\sigma_i^2}\right)\right) \to \max_{\Theta}, i = 1, \dots, k,$$

або,що теж саме

$$G(\{\mu_i, \sigma_i\}_{i=1}^{k}) = \sum_{j=1}^{n} g_{i,j}\left(-\ln(\sigma_i\sqrt{2\pi}) - \frac{(x_j-\mu_i)^2}{2\sigma_i^2}\right) \to \max_{\Theta}, i = 1, \dots, k,$$

Знайдемо похідні та прирівняємо їх до нуля

$$\frac{\partial}{\partial\mu_i} G(\{\mu_i, \sigma_i\}_{i=1}^{k}) = -\sum_{j=1}^{n} g_{i,j} \frac{x_j-\mu_i}{\sigma_i^2} = 0,$$

звідси

$$\mu_i = \frac{\sum_{j=1}^{n} g_{i,j} x_j}{\sum_{j=1}^{n} g_{i,j}}.$$

Аналогічно,

$$\frac{\partial}{\partial\sigma_i} G(\{\mu_i, \sigma_i\}_{i=1}^{k}) = -\sum_{j=1}^{n} g_{i,j}\left(\frac{1}{\sigma_i} - \frac{(x_j-\mu_i)^2}{\sigma_i^3}\right) = -\sum_{j=1}^{n} g_{i,j}\left(\frac{\sigma_i^2 - (x_j-\mu_i)^2}{\sigma_i^3}\right) = 0,$$

таким чином,

$$\sigma_i^2 = \frac{\sum_{j=1}^{n} g_{i,j}(x_j-\mu_i)^2}{\sum_{j=1}^{n} g_{i,j}}$$

що дозволяє отримати параметри уявному вигляді.

Застосування ЕМ-алгоритму до сегментації зображень

Для цієї мети побудуємо гістограму кольору для кожної компоненти - червоного, зеленого і синього. По осі х змінюється інтенсивність кольору від 0 до 255, а по осі у-кількість пікселів з даної інтенсивністю.

Вважаючи, що кожна гістограма являє собою суміш функцій Гаусса

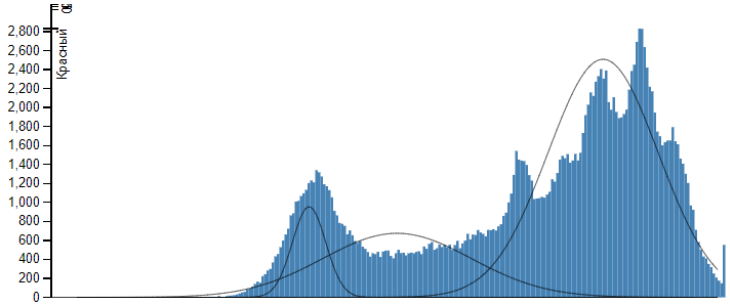
$$\sum_{i=1}^{n} \frac{w_i}{\sigma_i\sqrt{2\pi}} exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)$$

застосуємо ЕМ-алгоритм із заданою кількістю елементів суміші n (це ж кількість сегментів, на яке ми розбиваємо кожну колірну площину зображення). І як результат, нехай множина значень інтенсивності $X_j = \left(x_j, x_{j+v_j}\right)$ така, що

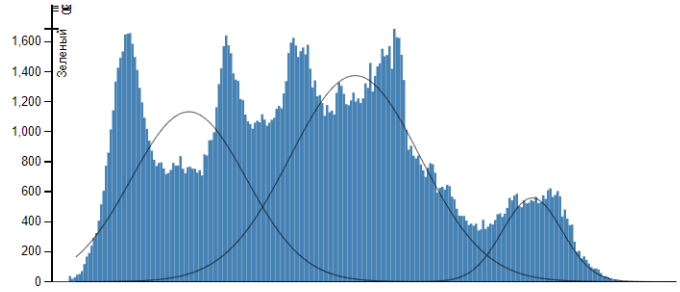
$$\forall x \in X_j : \frac{w_j}{\sigma_j\sqrt{2\pi}} exp\left(-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right) \geq \frac{w_i}{\sigma_i\sqrt{2\pi}} exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right) (i \neq j)$$

Тоді всім точкам, інтенсивність яких лежить в проміжку X_j, Поставимо у відповідність значення μ_j (при практичній реалізації - її цілу частину). Результатом роботи алгоритму буде сегментація зображення на nсегментів по кожній колірної компоненті.

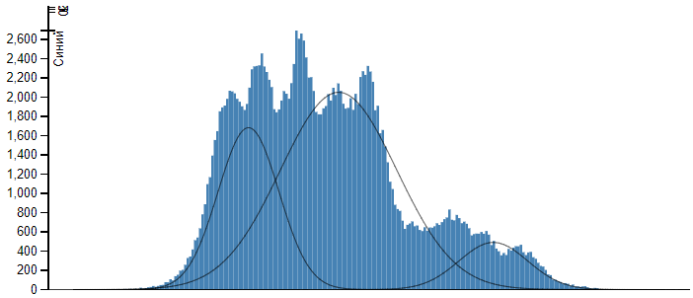
Для зображення Lena маємо



Опис гістограми Red лінійною комбінацією функцій Гауса.



Опис гістограми Green лінійною комбінацією функцій Гауса.



Опис гістограми Blue лінійною комбінацією функцій Гауса.

ЕМ-алгоритм для випадку двох змінних

Даний метод можна узагальнити на випадок двох змінних, тобто розглянути залежність пікселів не лише за кольором, а й за їх положенням – за координатами (x,y).

Як видно з розглянутого ЕМ-алгоритму, у разі використання даних двох змінних $(i,j,r_{i,j})$, $(i,j,g_{i,j})$, $(i,j,b_{i,j})$ кожну з множин можна описати лінійною комбінацією двовимірних функцій Гауса

$$\frac{1}{(2\pi)^{n/2}\left|\Sigma_i^{-1}\right|^{1/2}} exp\left(-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)\right)$$

за допомогою ЕМ-алгоритму аналогічно тому, як було зроблено у одновимірному випадку.

Зазначимо, що для багатовимірного випадку нормальний розподіл визначається наступним чином

$$p(x|\mu_i,\Sigma_i)=\frac{1}{(2\pi)^{n/2}\left|\Sigma_i^{-1}\right|^{1/2}} exp\left(-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)\right).$$

де $\neg$*-кореляційна матриця і параметри дорівнюють*

$$\mu_i=\frac{\sum_{j=1}^{n} g_{i,j}x_j}{\sum_{j=1}^{n} g_{i,j}}$$

та

$$\Sigma_i=\frac{\sum_{j=1}^{n} g_{i,j}(x_j-\mu_i)(x_j-\mu_i)^T}{\sum_{j=1}^{n} g_{i,j}}.$$

Порогова і мультіпороговая сегментація

Порогова сегментація зображення за рівнями яскравості - найпростіший вид сегментації зображення. Цей метод заснований на тому, що багато об'єктів або області зображення характеризуються постійної відбивною здатністю або поглинанням світла на їх поверхні. Відмінною рисою порогової сегментації є обчислювальна ефективність і можливість використання в системах реального масштабу часу.

Порогова сегментація виконується наступним чином:

$$Bin_{i,j} = 1, \text{якщо } I_{i,j} \geq T,$$
$$Bin_{i,j} = 0, \text{якщо } I_{i,j} < T,$$

де $Bin_{i,j}$ - елемент результуючого бінарного зображення, $I_{i,j}$ - елемент вихідного зображення.

Успіх порогової сегментації залежить від способу вибору порога. У разі використання методу Оцу, зображення розбивається на передній і на задній план, тобто, проводиться бінарізація зображення.

Мультіпороговая сегментація **.**

Використовується в тому випадку, якщо вихідне зображення характеризується не бімодальною, а мультимодальною гістограмою. В цьому випадку результуюче зображення розбивається на кілька рівнів:

$$G_{i,j} = 1, \text{якщо } I_{i,j} \in D_1, \quad G_{i,j} = 2, \text{якщо } I_{i,j} \in D_2, \ldots$$
$$G_{i,j} = n, \text{якщо } I_{i,j} \in D_n, G_{i,j} = 0, \text{інакше.}$$

Розглянемо задачу вибора порогів у разі мультіпорогової сегментації. По суті, вирішення цієї задачі зводиться до побудови гістограми з заданим числом вузлів n<255, яка найкращим чином апроксимує гістограму зображення. Розподіл вузлів цієї гістограми з малим числом вузлів і є розподілом порогів.

Про побудову асимптотически оптимальних гістограм

Термін гістограма ввів Карл Пирсон ще в 1895 році. Гістограми дозволяють побачити, як розподілені значення змінних по інтервалах угруповання, наскільки сильно вони різняться між собою, як сконцентрована більшість спостережень навколо середнього, є розподіл симетричним чи ні, чи має воно одну моду та інше.

Нехай $X_1, X_2, \ldots, X_N$ - вибірка заданого розподілу, розбиття

$\Delta_n = \{-\infty < t_0 < t_1 < \ldots < t_n < \infty\}$ із кроком $h_i = t_i - t_{i-1}$ і $m_i = \sum_{k=1}^{N} \{1 | X_k \in (t_{i-1}, t_i]\}, i = 1, \ldots, n$

число елементів вибірки з і-го інтервалу. Тоді кусково-постійна функція $H : R \to R$

$$H(t) = H(\Delta_n, t) = m_i, t \in (t_{i-1}, t_i], i = 1, 2, \ldots, n$$

називається гістограмою вибірки $X_1, X_2, \ldots, X_N$, якщо ж кусочно-постійна функція $\hat{H} : R \to R$ така, що

$$\hat{H}(t) = \hat{H}(\Delta_n, t) = \frac{m_i}{N h_i}, t \in (t_{i-1}, t_i], i = 1, 2, \ldots, n,$$

то вона називається нормалізованою гістограмою. Помітимо, що нормалізована гістограма є функцією щільності ймовірності, тому що

$\hat{H}(t) \geq 0$ при $t \in R$ і, крім того, $\int_{-\infty}^{\infty} \hat{H}(t) dt = 1$.

Для абсолютно неперервних розподілів випадкових величин із щільністю ймовірності $f(x)$ для нормалізованих гістограм при $N \to \infty$

$$\forall t \in (t_{i-1}, t_i], \hat{H}(t) = \frac{m_i}{Nh_i} \to P(X \in (t_{i-1}, t_i]) \equiv \int_{t_{i-1}}^{t_i} f(t)dt, i = 1,2,...,n.$$

Таким чином, для неперервної випадкової величини, гістограма на кожній ділянці свого постійного значення повинна зберігати середнє значення. Традиційно розглядаються гістограми з рівновіддаленими вузлами, але вибір вузлів гістограмы істотно впливає на якість опису випадкового процесу. Розглянемо задачу побудови гістограми з асимптотично оптимальними вузлами.

Позначимо через $\Delta_n = \{x_i\}_{i=0}^n$ - довільне розбиття $0 = x_0 < x_1 ... < x_n = T$, і покладемо $h_{i+\frac{1}{2}} = x_{i+1} - x_i$, $h = \max_i h_{i+\frac{1}{2}}$ і $x_{i+\frac{1}{2}} = \frac{x_{i+1} + x_i}{2}$. Через $S\left(\{a_{i+1/2}\}_{i=1}^{n-1}, \Delta_n, x\right)$ позначимо кусково-постійну функцію зі значеннями $a_{i+\frac{1}{2}}$ для $x \in [x_i, x_{i+1})$.

Для функції $y(x)$, неперервної на [0,T], розглянемо задачу

$$\varepsilon = \min_{x_i} \min_{a_{i+\frac{1}{2}}} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(a_{i+\frac{1}{2}} - y(x) \right)^2 dx. \tag{2.4}$$

Теорема 1. Нехай $y(x)$ - неперервно диференційована функція $y \in C^2_{[0,T]}$ така, що $y'(x)$ на проміжку $[0, T]$ має не більше скінченного числа нулів, тоді

$$\varepsilon = \frac{1}{12n^2} \left(\int_0^T (y'(x))^{\frac{2}{3}} dx \right)^3 + O\left(\frac{1}{n^3}\right) \tag{2.5}$$

і при цьому мінімум досягається для розбиття $\Delta_n^* = \{x_i^*\}_{i=0}^m$ такого, що

$$\int_{x_i^*}^{x_{i+1}^*} (y'(x))^{\frac{2}{3}} dx = \frac{1}{n} \int_0^T (y'(x))^{\frac{2}{3}} dx \qquad \text{и} \qquad a^*_{i+\frac{1}{2}} = \frac{1}{h^*_{i+\frac{1}{2}}} \int_{x_i^*}^{x_{i+1}^*} y(x)dx. \tag{2.6}$$

Доведемо це твердження. На початку розглянемо наступну задачу

$$\Phi\left(a_{i+\frac{1}{2}} \right) = \int_{x_i}^{x_{i+1}} \left(a_{i+\frac{1}{2}} - y(x) \right)^2 dx \to \min_{a_{i+\frac{1}{2}}} \tag{2.7}$$

Тоді зважаючи на опуклість задачі (2.4), необхідна умова мінімума є і достатньою, тобто, розв'язок цієї задачі знаходиться з рівняння

$$\frac{d\Phi\left(a_{i+\frac{1}{2}} \right)}{da_{i+\frac{1}{2}}} = 2 \int_{x_i}^{x_{i+1}} \left(a_{i+\frac{1}{2}} - y(x) \right) dx = 0,$$

і тоді,

$$a_{i+\frac{1}{2}} = \frac{1}{x_{i+1} - x_i} \int_{x_i}^{x_{i+1}} y(x)dx$$

Таким чином, задачу (2.4) можна переписати у вигляді

$$\varepsilon = \min_{x_i} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(y(x) - \frac{1}{h_{i+\frac{1}{2}}} \int_{x_i}^{x_{i+1}} y(x)dx \right)^2 dx.$$

Використовуючи формулу Тейлора

$$y(x) = y_{i+\frac{1}{1}} + y'_{i+\frac{1}{2}}\left(x - x_{i+\frac{1}{2}}\right) + O(h^2)$$

де $h = \max_i h_{i+\frac{1}{2}}$, маємо

$$\varepsilon = \min_{x_i} \min_{a_{i+\frac{1}{2}}} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(a_{i+\frac{1}{2}} - y(x)\right)^2 dx =$$

$$= \min_{x_i} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(y_{i+\frac{1}{1}} + y'_{i+\frac{1}{2}}\left(x - x_{i+\frac{1}{2}}\right) + O(h^2) - \frac{1}{h_{i+\frac{1}{2}}} \int_{x_i}^{x_{i+1}} \left(y_{i+\frac{1}{1}} + y'_{i+\frac{1}{2}}\left(x - x_{i+\frac{1}{2}}\right) + O(h^2)\right)dx\right)^2 dx =$$

$$= \min_{x_i} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(y_{i+\frac{1}{1}} + y'_{i+\frac{1}{2}}\left(x - x_{i+\frac{1}{2}}\right) - y_{i+\frac{1}{1}} + O(h^2)\right)^2 dx = \min_{x_i} \frac{1}{12} \sum_{i=0}^{n-1} \left(\left(y'_{i+\frac{1}{2}}\right)^2 h^3_{i+\frac{1}{2}} + O(h^4)\right).$$

Із теореми про середнє для інтегралів маємо, що знайдеться точка $\xi_{i+\frac{1}{2}} \in \left[x_i, x_{i+1}\right]$ така, що $\left(y'\left(\xi_{i+\frac{1}{2}}\right)\right)^2 h^3_{i+\frac{1}{2}} = \left(\left(y'\left(\xi_{i+\frac{1}{2}}\right)\right)^{\frac{2}{3}} h_{i+\frac{1}{2}}\right)^3 = \left(\int_{x_i}^{x_{i+1}} (y'(x))^{\frac{2}{3}} dx\right)^3$,

Звідси і із попереднього одразу маємо

$$\varepsilon = \min_{x_i} \frac{1}{12} \sum_{i=0}^{n-1} \left(\left(y'_{i+\frac{1}{2}}\right)^2 h^3_{i+\frac{1}{2}} + \left(y'\left(\xi_{i+\frac{1}{2}}\right)\right)^2 h^3_{i+\frac{1}{2}} - \left(y'\left(\xi_{i+\frac{1}{2}}\right)\right)^2 h^3_{i+\frac{1}{2}} + O(h^4)\right) =$$

$$= \min_{x_i} \frac{1}{12} \sum_{i=0}^{n-1} \left(\left(\int_{x_i}^{x_{i+1}} (y'(x))^{\frac{2}{3}} dx\right)^3 + O(h^4)\right). \qquad (2.8)$$

Лема. Нехай $\alpha > 0$ і $C > 0$, тоді

$$\min\left\{\sum_{i=0}^{n-1} C_i^{\alpha} \middle| C_i \geq 0, \sum_{i=0}^{n-1} C_i = C\right\} = n\left(\frac{C}{n}\right)^{\alpha},$$

і при цьому мінімум досягається тоді, коли всі C_i рівні між собою, тобто,

$$C_i^* = \frac{C}{n} (i = 0, \ldots, n-1).$$

Для доведення цього твердження використаємо метод невизначених множників Лагранжу. Випишемо функцію мети

$$L = \lambda_0 \sum_{i=0}^{n-1} C_i^{\alpha} + \lambda_1 \left(\sum_{i=0}^{n-1} C_i - C\right),$$

тоді

$$\frac{\partial L}{\partial C_i} = \lambda_0 \alpha C_i^{\alpha-1} + \lambda_1 = 0,$$

і

$$\lambda_0 \alpha \left(C_i^{\alpha-1} + \lambda_2\right) = 0, \text{ де } \lambda_2 = \frac{\lambda_1}{\lambda_0 \alpha}.$$

Таким чином, маємо $C_i = -\lambda_2^{\frac{1}{\alpha-1}}$ і

$$\sum_{i=0}^{n-1} C_i = -\sum_{i=0}^{n-1} \lambda_2^{\frac{1}{\alpha-1}} = -n\lambda_2^{\frac{1}{\alpha-1}} = C.$$

Звідси $\lambda_2^{\frac{1}{\alpha-1}} = -\frac{C}{n}$, або, що те ж саме, $C_i = \frac{C}{n} (i = 0, \ldots, n-1)$.

Лема доведена.

Таким чином, з (2.7) і доведеної леми маємо

$$\varepsilon = \min_{x_i} \frac{1}{12} \sum_{i=0}^{n-1} \left(\left(\int_{x_i}^{x_{i+1}} (y'(x))^{\frac{2}{3}} dx \right)^3 + O(h^4) \right) = \frac{1}{12} \sum_{i=0}^{n-1} \left(\left(\int_{x_i^*}^{x_{i+1}^*} (y'(x))^{\frac{2}{3}} dx \right)^3 + O(h^4) \right) =$$

$$= \frac{1}{12n^2} \left(\int_0^T (y'(x))^{\frac{2}{3}} dx \right)^3 + O\left(\frac{1}{n^3} \right)$$

і при цьому мінімум досягається для разбиття $\Delta_n^* = \{x_i^*\}_{i=0}^m$

такого, що

$$\int_{x_i^*}^{x_{i+1}^*} (y'(x))^{\frac{2}{3}} dx = \frac{1}{n} \int_0^T (y'(x))^{\frac{2}{3}} dx.$$

Крім того, зазначимо, що для заданої похибки ε можна знайти кількість вузлів n кусково-постійної функції $S(\{a_{i+1/2}\}_{i=1}^{n-1}, \Delta_n, x)$ з асимптотично оптимальними вузлами, які вибирається із умови (1.6)

$$n = \left[\sqrt{\frac{1}{12\varepsilon} \left(\int_0^T (y'(x))^{\frac{2}{3}} dx \right)^3} \right] + 1,$$

де $\lfloor m \rfloor$ - ціла частина числа m.

Ідея отриманого результату складається в тому, що вузли розподіляються таким чином, щоб похибки наближення на кожному відрізку функції x(t) кусково-постійною функцією $S(\{a_{i+1/2}\}_{i=1}^{n-1}, \Delta_n, x)$ будуть рівні між собою.

Розподіл вузлів $S(\{a_{i+1/2}\}_{i=1}^{n-1}, \Delta_n, x)$ у разі наближення гістограми зображення може служити множиною порогів для сегментації даного зображення.

Алгоритм вибору асимптотично оптимальних порогів сегментації

Спираючись на отримані результати, формалізуємо алгоритм вибору порогів сегментації. Нехай $\{h_i\}_{i=0}^{255}$ –гістограма інтерсивності освітлення (компоненти Y).

Знайдемо

$$d_i = |h_{i+1} - h_i|^{2/3}, i = 1, \dots, 255.$$

Далі

$$\varphi(0) = 0, \qquad \varphi(k) = \sum_{i=1}^{k} d_i, \quad k = 1, \dots, 255.$$

У разі відомого числа сегментів nпороги $T_j, \ j = 0, 1, \dots, n$ знаходяться наступним чином

$$T_0 = 0, \frac{(j-1)\varphi(255)}{n} \le \varphi(T_j) \le \frac{j\varphi(255)}{n}, j = 1, \dots, n.$$

Таким чином, можна розбити зображення на сегменти по обчисленим порогам.

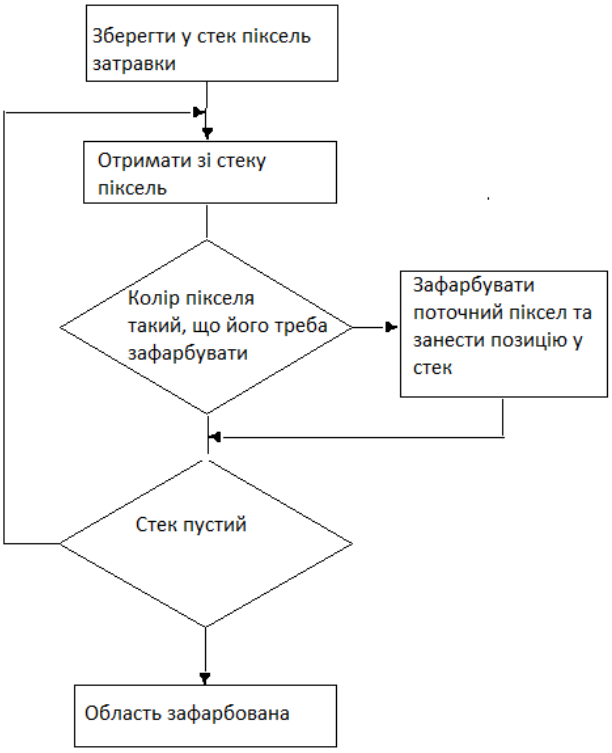
Ясно, что кожен сегмент буде складатися з декількох не зв'язаних з собою областей. Наступним кроком буде злиття областей, тобто, якщо кількість точок, з яких

складається область, менше заданого числа К, то дану область будемо об'єднувати з тією сусідньою областю, рівень сегментації якої найбільш схожий з рівнем поточної області.
Для визначення точок, які належать області, в тому числі, а також знайти кількість точок області, треба використати той чи інший алгоритм заповнення однозв'язаної області. Розглянемо один з них.

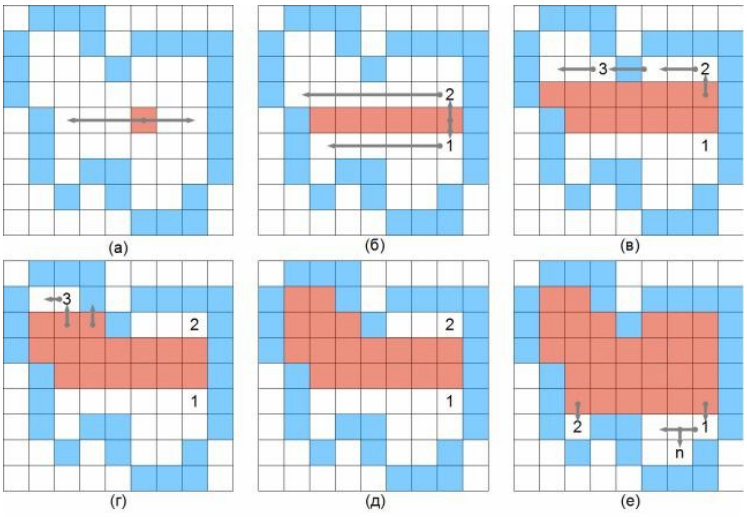
Простий алгоритм заповнення із затравкою
Використовуючи стек, можна розробити простий алгоритм заповнення зв'язної області. Стек - це просто масив або інша структура даних, в яку можна послідовно поміщати значення і з якої їх можна послідовно вибирати. Коли нові значення додаються або поміщаються в стек, всі інші значення опускаються вниз на один рівень. Коли значення видаляються або витягуються зі стека, інші значення спливають або піднімаються вгору на один рівень. Такий стек називается стеком прямої дії або стеком з дисципліною обслуговування "першим прийшов, останнім вийшов" (LIFO). Простий алгоритм заповнення із затравкою можна представити в наступному вигляді:

Простий алгоритм заповненя
Затравка(x, y) *видає затравочный піксел.*
Push - *процедура, якавкладає піксел в стек.*
Pop - *процедура, якавиймає піксел зі стека.*

```
Піксел(x, y) = Затравка(x, y)
PushПіксел(x, y)  ;ініціалізація стеку
While (стек не пустий)
   PopПиксел(x, y)  ;виймаємо піксел зі стека
   IfПиксел(x, y) <> Нов_значенняthen
   Пиксел(x, y) = Нов_значення
   Endif
   Перевірка, чи треба вкладати сусідні пікселі у стек
   If (Піксел(x + 1, y) <> Нов_значенняand
   Піксел(x + 1, y) <> Гран-значення)then
   PushПіксел (x + 1, y)
   If (Піксел(x, y + 1) <> Нов_значенняand
   Піксел(x, y + 1) <> Гран_значення)then
   PushПиксел (x, y + 1)
   If (Піксел(x - 1, y) <> Нов_значенняand
   Піксел(x - 1, y) <> Гран_значення)then
   PushПіксел (x - 1, y)
   If (Піксел(x, y - 1) <> Нов_значенняand
   Піксел(x, y - 1) <> Гран_значення)then
   PushПіксел (x, y - 1)
   End If
EndWhile
```



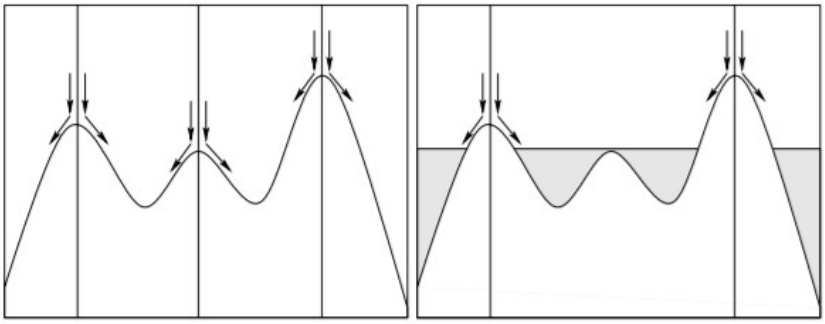
В алгоритмі перевіряються і поміщаються в стек 4-зв'язні пікселі, починаючи з правого від поточного пікселя. Напрямок обходу пікселов- проти годинникової стрілки.



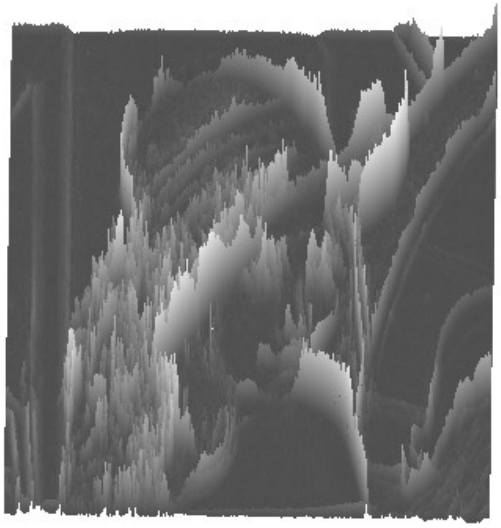
Для тестового зображення Lena розбиття на k=5 сегментів маємо

Алгоритм водорозділу

Сегментація водотоків - алгоритм, натхненний природою, що імітує затоплення водою топографічного рельєфу.



У сегментації водорозділу зображення вважається топографічним рельєфом, де градієнтна величина трактується як інформація про висоту.



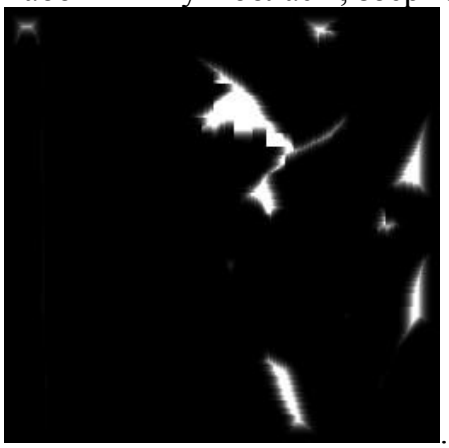
Алгоритм включає в себе декілька кроків, а саме: бінарізація по Оцу, морфологічне відкриття, трансформація відстані і, нарешті, алгоритм вододілу, після чого злиття малих областей з сусідами.

Проілюструємо крок за кроком алгоритм водорозділу на зображення Lena.

Крок перший – бінарізація.

Крок другий – морфологічне відкриття.

Крок третій -дистанційна трансформація.
Даний процес є представлення собою співставлення переднього плану бінарного зображення з відстанню від найближчої перешкоди або фонового пікселя. Цей процес дозволяє прибрати невеличкі або витягнуті області, зберігаючи регіональні центри.



Ну і, нарешті, використовуючи градієнт зображення в якості топографічної мапи, інтенсивність освітлення асоціюється з висотою, а світлі області попереднього зображення використовуємо в якости шлюзів, з яких проводиться заповнення, проведемо сегментацію методом заводнення.

Після злиття областей, отримуємо сегментоване зображення.

Суперпіксельна сегментація

Суперпіксельна сегментація реалізує розбиття зображення на множину дрібних фрагментів (суперпікселей), що представляють собою відносно однорідні групи розташованихпоруч пікселів. Кожен суперпіксель потенційно є атомарним регіоном (фрагментом) зображення, тобто, всі вхідні в нього пікселі розглядаються при подальшій обробці як єдинеціле. При цьому суперпікселі не обов'язково повинні мати правильну форму і, природно, завжди є певне число помилок, що допускаються при прагненні розбити зображення на однорідні фрагменти. Основна вимога до суперпіксельної сегментації полягає в наступному: пікселі всередині кожного суперпікселя повинні бути максимально схожі, а пікселі, що знаходяться в різних суперпікселях, повинні до певної міри відрізнятися. Дане завдання може вирішуватися принципово різними способами. Розглянемо один із найкращих методів суперпіксельної сегментації- SLOC (Simple Linear Iterative Clustering).

Проста лінійна ітераційна кластеризація (SLIC) є адаптацією k-середніх для генерації суперпікселів з двома важливими відмінностями:

1) Оптимізація кількісті розрахунків відстані значно скорочується шляхом обмеження простору пошуку на область, пропорційну розміру суперпікселя. Це робить складність лінійною за кількістю пікселів N і незалежним від числа суперпікселів k.

2) Міра зваженої відстані поєднує в собі колірну і просторову близькість, одночасно забезпечуючи контроль за розміром і компактністю суперпікселів.

Алгоритм

SLIC простий у використанні та розумінні. За замовчуванням єдиним параметром алгоритму є параметр k - бажана кількість суперпікселів приблизно рівного розміру. Для кольорових зображень у колірному просторі CIELAB процедура кластеризації починається з етапу ініціалізації, де k стартових центрів кластерів $C_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$ вибираються на регулярній сітці, з кроком на S пікселів. Для отримання приблизно однакових розмірів суперпікселів, інтервал сітки $S = \sqrt{N/k}$.На кожній ітерації центри переміщуються на місце, що відповідає найменшій позиції градієнта в околі 3 × 3 пікселя

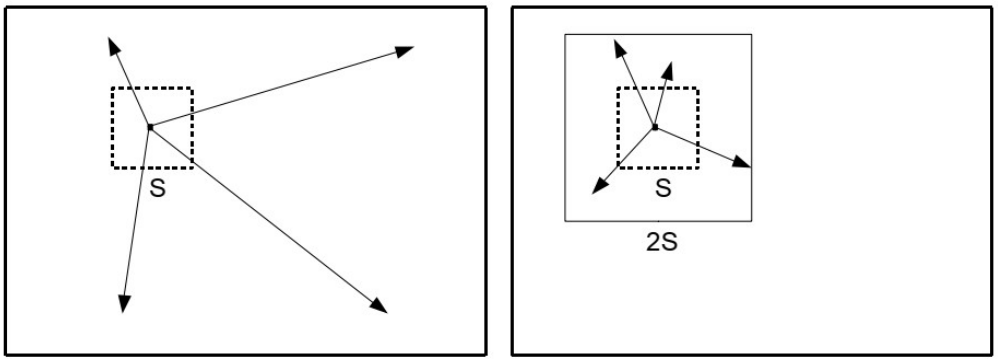
$$G(x,y) = \|I(x+1,y) - I(x-1,y)\|^2 + \|I(x,y+1) - I(x,y-1)\|^2,$$

де *I(x,y) lab* вектор, що відповідає пікселю з позицією (x,y) та $\|\cdot\|$норма L_2.

Це робиться для того, щоб уникнути центрування суперпікселя по краю, і щоб зменшити ймовірність співпадання суперпікселів з зашумленим пікселем. Оскільки очікувана просторова протяжність суперпікселя є областю приблизно розміру $S \times S$, пошук подібних пікселів здійснюється в області $2S \times 2S$ навколо центру суперпікселів.

Як тільки кожен піксель асоціюється з найближчим центром кластера, крок оновлення регулює центри кластера як середнього вектору $[l\ a\ b\ x\ y]^T$всіх пікселів, що належать кластеру.

Норма L_2 використовується для обчислення залишкової помилки E між новими місцями центру кластера і попередніми центрами кластера. Кроки призначення та оновлення можна повторювати ітераційно, поки помилка не збігається, але для більшості зображень достатньо 10 ітерацій.



Алгоритм SLIC суперпіксельної сегментації

/ ∗ Ініціалізація /

Ініціалізація центрів кластерів $C_k = [l_k;\ a_k;\ b_k;\ x_k;\ y_k]^T$ - вибірка пікселів на регулярній сітціз кроком S.

Перемістити центри кластера в найнижчу позицію градієнта в околі 3 × 3.

Встановити мітку l (i) = −1 для кожного пікселя i.

Встановити відстань d (i) = ∝ для кожного пікселя i.

repeat

/ ∗ Призначення /

forкожного кластерного центру C_k***do***

for кожного пікселя i в області 2S × 2S навколо C_k***do***

Обчислити відстань D між C_kта i.

if*D <d (i)****then***
множина *d (i) = D*
множина *l (i) = k*
end if
end for
end for
/ ∗ Оновлення /
Обчислити нові центри кластерів.
Обчислення залишкової помилки E.
until E ≤ порогу

Міра відстані

Суперпікселі SLIC відповідають кластерам в просторі площини кольору labxy. Це створює проблему при визначенні міри D, яка може бути не очевидною. D - відстань між пікселем і та центром кластера C_k в алгоритмі. Колір пікселя представлений у колірному просторі CIELAB $[l\ a\ b]^T$. Положення позиції пікселя $[x\ y]^T$, з іншого боку, може приймати діапазон значень, що змінюється залежно від розміру зображення.

Просто визначення D як п'ятивимірної евклідової відстані у просторі labxy призведе до невідповідності у кластеризації. Для великих суперпікселів просторові відстані перевершують колірну близькість, надаючи більш відносне значення просторовій близькості, ніж колір. Це продукує компактні суперпікселі, які не добре дотримуються меж зображення. Для дрібних суперпікселів є вірним зворотне.

Щоб об'єднати дві відстані в єдину міру, необхідно нормалізувати близькість кольору і просторову близькість за їх відповідними максимальними відстанями в межах кластера. Нехай

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2},$$

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2},$$

$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2}.$$

Максимальна просторова відстань, що очікується в межах даного кластера повинна відповідати інтервалу вибірки, $N_s = S = \sqrt{N/K}$. Визначення максимальної відстані кольору Nc не так просте, оскільки кольорові відстані можуть істотно відрізнятися від кластера до кластера і від зображення до зображення. Цієї проблеми можна уникнути, замінивши Nc на константу m, тоді

$$D' = \sqrt{\left(\frac{d_c}{m}\right)^2 + \left(\frac{d_s}{S}\right)^2},$$

що спрощує вимірювання відстані. На практиці використовується значення відстані

$$D = \sqrt{(d_c)^2 + m^2 \left(\frac{d_s}{S}\right)^2}.$$

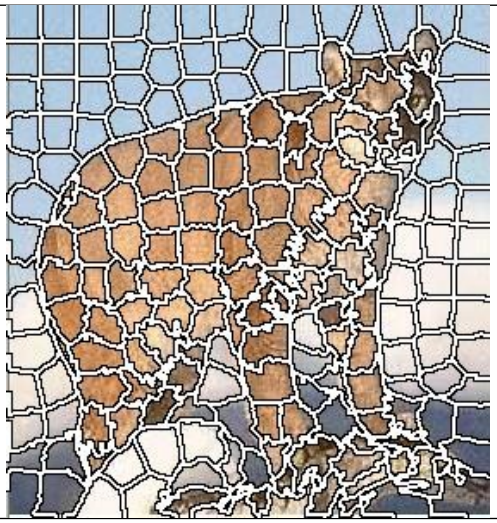
Визначаючи D таким чином, m також дозволяє зважувати відносну важливість між колірною подібністю і просторовою близькістю. Коли m є великим, просторова близькість є більш важливою і результуючі суперпікселі є більш компактними (тобто вони мають більш низьке співвідношення області до периметра). Коли m є малим, отримані суперпікселі щільніше прилипають до меж зображення, але мають менший розмір і форму. При використанні колірного простору CIELAB параметр m змінюється у діапазоні $m \in [1; 40]$, але, як правило, вибирається значення m=20.
В деяких випадках в якості характеристики відстані використовується значення

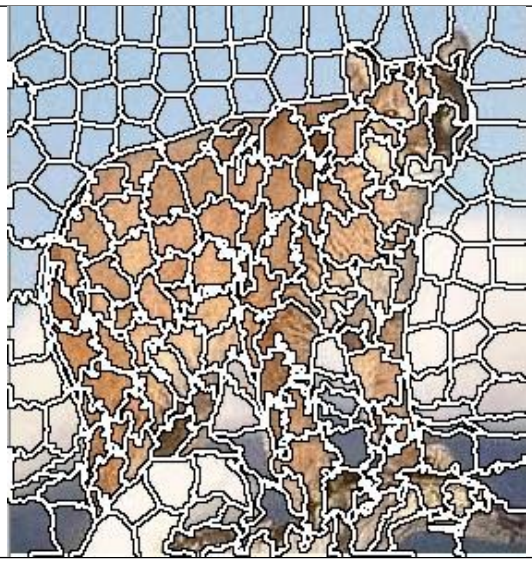
$$D = d_c + \frac{d_s}{S} m, m \in [1,20].$$

Рівняння може бути пристосовано для зображень у градаціях сірого шляхом налаштування

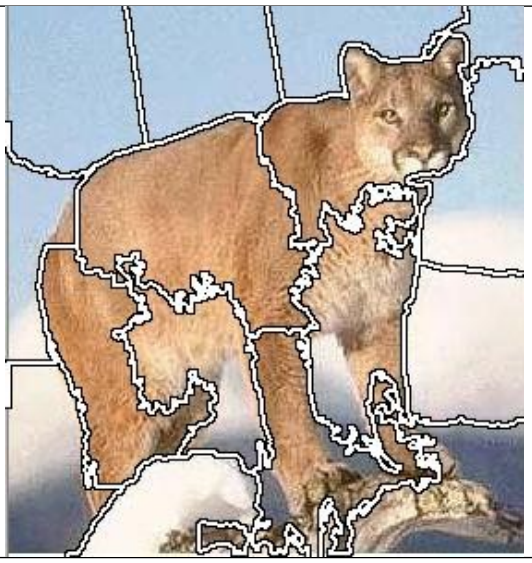
$$d_c = \sqrt{(l_j - l_i)^2}.$$

Наведемо приклади суперпіксельної сегментації для різних параметрів.

	
Кількість сегментів 200, m=10	Кількість сегментів 200, m=40



Кількість сегментів 200, m=20



Кількість сегментів 20, m=20

Текстурна сегментація за фрактальними характеристиками.
Важливою задачею аналізу зображень є формування сегментів за текстурою областей. Проблема складна і погано формалізується. Один з підходів для вирішення такого роду задачіскладається в тому, що фрагменти зображення аналізуються з точки зору самоподобія, тобто з точки зору фрактальності.

Фрактали іфрактальная розмірність

Термін фрактал був запропонований Бенуа Мандельбротом (B. Mandelbrot) у 1975 році для позначення нерегулярних самоподібних математичних структур. Основне визначення фрактала, дане Мандельбротом, звучало так: "Фракталом називається структура, що складається із частин, які в якомусь сенсі подібні до цілого". Варто визнати, що це визначення, через свою нестрогість, не завжди вірно. Можна привести багато прикладів самоподібних об'єктів, що не є фракталами, наприклад, залізничні колії що сходяться до обрію.

У найпростішому випадку невелика частина фрак талу містить інформацію про весь фрактал. Строге визначення самоподібних множин було дано Дж. Хатчинсоном (J. Hutchinson) в 1981 році. Він назвав множину самоподібною, якщо воно складається з декількох компонентів, подібних всій цій множині, тобто компонент одержуваних афінними перетвореннями.

Однак самоподобність – це хоча й необхідне, але далеко не достатня властивість фракталів. Адже не можна ж, справді, уважати фрак талом точку, або площину, розкреслену клітками. Головна особливість фрактальных об'єктів полягає в тому, що для їхнього опису недостатньо «стандартної» топологічної розмірності, що, як відомо, для лінії дорівнює 1, для поверхні 2, і т.д. Фракталам характерна геометрична «ізрізаність». Тому використовується спеціальне поняття фрактальної розмірності, що введене Ф. Хаусдорфом (F. Hausdorf) і А.С. Безиковичем. Стосовно до ідеальних об'єктів класичної евклідової геометрії вона давала ті ж чисельні значення, що й топологічна розмірність, однак нова розмірність мала більш тонку чутливість до всякого роду недосконалостям реальних об'єктів, дозволяючи розрізняти й індивідуалізувати те, що колись було безлике й нерозрізнене. Розмірність Хаусдорфа - Безиковича саме й дозволяє вимірювати ступінь «ізрізаності». Розмірність фрактальных об'єктів не є цілим числом, характерним для звичних геометричних

об’єктів. Разом з тим, у більшості випадків фрактали нагадують об'єкти, що щільно займають реальний простір, але не використовують його повністю.

Нехай є множина G в евклідовому просторі розмірності r. Ця множина покривається кубиками тієї ж розмірності, при цьому довжина ребра будь-якого кубика не перевищує деякого значення δ , тобто $\delta_i<\delta$.

Уводиться залежна від деякого параметра d і δ сума по всіх елементах покриття:

$$\ell_d(\delta) = \sum_i \delta_i^d$$

Визначимо нижню грань даної суми:

$$L_d(\delta) = \inf \sum_i \left| \delta_i^d \middle| \delta_i < \delta \right|$$

При зменшенні максимальної довжини δ, якщо параметр d буде досить великий, мабуть, буде виконуватися $\lim_{\delta \to 0} L_d(\delta) \to 0$.

При деякому досить малому значенні параметра d буде виконуватися:

$$\lim_{\delta \to 0} L_d(\delta) \to \infty.$$

Проміжне, критичне значення σ , для якого виконується:

$$\lim_{\delta \to 0} L_d(\delta) = \begin{cases} 0, & d > \sigma, \\ \infty, & d < \sigma, \end{cases}$$

і називається розмірністю Хаусдорфа-Безиковича (або фрактальною розмірністю). Для простих геометричних об'єктів розмірність Хаусдорфа-Безиковича збігається зтопологічної (для відрізка 1, для квадрата2, для куба 3 і т.д.)

Незважаючи на те, що розмірність Хаусдорфа-Безиковичаз теоретичної точки зору визначена бездоганно, для реальних фрактальних об'єктів розрахунок цієї розмірності єдосить скрутним. Тому вводиться трохи спрощений показник - ємнісна розмірність d_c . При визначенні цієї розмірності використовуються кубики із гранями однакового розміру. У цьому випадку, природно, справедливо: $L_d(\delta) = N(\delta)\delta^{dc}$,

де N(δ) - кількість кубиків, що покриває область G. Шляхом логарифмування й переходу до межі при зменшенні грані кубика одержуємо:

$$d_c = - \lim_{\delta \to 0} \frac{\log N(\delta)}{\log \delta},$$

якщо ця межа існує. Слід зазначити, що в більшості чисельних методів визначення фрактальной розмірності використовується саме d_c, при цьому необхідно враховувати, що завжди справедливо умову: $\sigma \le d_c$. Для регулярних само подібних фракталов ємнісна розмірність і розмірність Хаусдорфа-Безиковича збігаються, тому термінологічно їх часто не розрізняють і говорять просто про фрактальной розмірності об'єкта. На жаль класичний підхід дозволяє оцінити фрактальний розмір об’єктів на площині, наприклад, берегову лінію, а з тим, щоб оцінити двовимірний об’єкт, такий я зображення, є великі проблеми. Але все не так погано, існують деякі підходи, які ми і розглянемо.

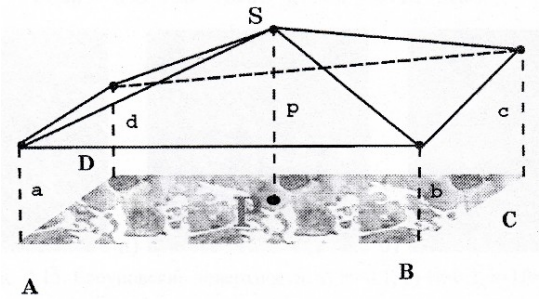
Методи оцінювання фрактальності кольорових/напівтонових зображень

В основі оцінювання кольорових/напівтонових зображень лежить той факт, що будь-яке сіре (напівтонове, grayscale) або кольорове зображення можна представити, як деяку поверхню, де низинам відповідають "темні" пікселі, а пікам - горам відповідають "світлі" пікселі.

Розглянемо декілька методів.

Метод Triangular Prism Surface Area

Метод TPSA [3] призначений для обробки напівтонових (grayscale) зображень.



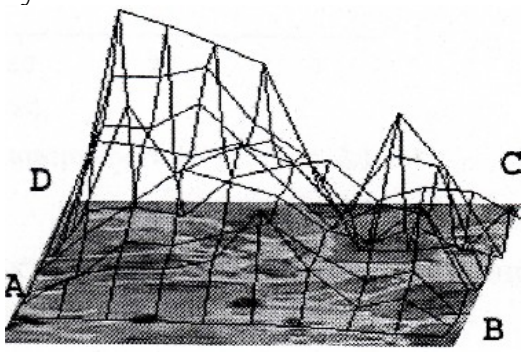
Призма ABCDSabcd. A, B, C, D, P- точки зображення,
a, b, c, d, p - значення яскравості в пікселях A, B, C, D, P

Алгоритм полягає в наступному.
Розглядається центральний пиксель P и його оточення - квадрат ABCD, лінії між величинами a, b, c, d, p - утворять 4 трикутникивершиною S, з площею:

$$S_P = \sum_{i=1}^{4} Str_i,\ Str_i = F(x_i, y_i, r),$$

де S_p - площа фігури, Str_i- площа i-го трикутника $i = 1..4$, x_i, y_i - координати пікселів (вершини i-го трикутника), r - сторона квадрата ABCD.
Подібна процедура повторюється для всіх пікселів зображення. Для фіксованого rплощі всіх фігур формують одну загальну площу A.
Потім r збільшується, і процедура повторюється. Нахил s графіка A(r) у логарифмічному масштабі використовується для визначення фрактальной розмірності: $D_{tP} = 2 - s$.
Даний метод, при дослідженні кольорових зображень дозволяє оцінювати розмірність кожного колірного каналу RGB окремо розмірність його “яркісного” зображення.

Метод 2D Variation Procedure
Метод 2D Variation [3] призначений для обробки бінарних (binary) і напівтонових (grayscale) зображень.
Знаходимо мінімальний і максимальний рівні сірого в межах кожної квадратної області зі стороною r. Таким чином, визначаються двомірна мінімальна й максимальна функції для кожного кроку решітки.Потім для всього зображення визначається різниця об'єму між мінімальними й максимальними функціями.



Метод 2D Variation. P- центральна точка квадратного регіону $r \times r$,

Тоді $V(r) = r^S const$, де V - «різницевий» об'єм фігури r - крок сітки, S- фрактальная розмірність.Подібна процедура повторюється для всіх пикселей зображення, потім r

збільшується, і процедура повторюється. Нахил s графіка V(r) у логарифмічному масштабі використовується для визначення фрактальной розмірності:

$$D_{2D} = 3 - \frac{s}{2}.$$

Даний метод при роботі з кольоровими зображеннями також як і TPSA дозволяє оцінювати як розмірність кожного колірного каналу RGB окремо, так і розмірність "яркостного" зображення.

Броуновска розмірність

Ще одним методом оцінювання фрактальной розмірності напівтонових зображень є броуновская розмірність, що пропорційна експоненті Херста (Hurst`s exponent) $D=3-H$.

Параметр H можна обчислити статистично із закону дисперсії, використовуючи властивості броуновского руху:

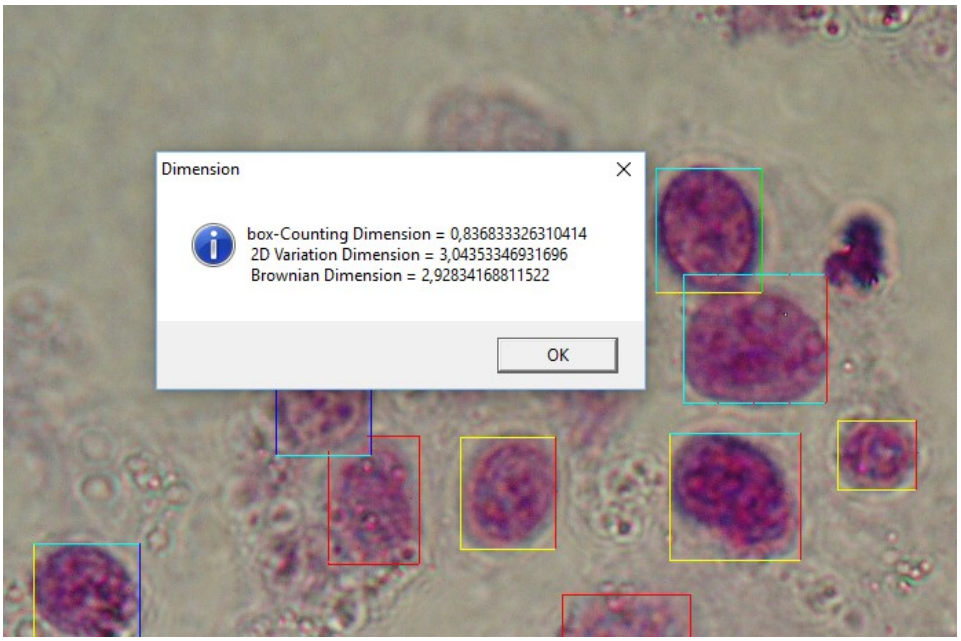
$$H = - \frac{\log(\sigma_{\Delta R}(\Delta I))}{\log(\Delta R)}$$

$\sigma_{\Delta R}(\Delta I)$ - середньоквадратичне відхилення приростів, тобто різниця яскравості в точках $R + \Delta R$ і R, $\Delta I = I(R + \Delta R) - I(R)$), R - координати точки [4].

Алгоритм обчислень виглядає в такий спосіб: для всіх точок обчислюються прирости яскравості ΔI в ΔR-околі, для отриманого масиву приростів обчислюється середньоквадратичне відхилення $\sigma_{\Delta R}(\Delta I)$, потім змінюємо ΔR і повторюємо попередні кроки. Нахил графіка залежності $\sigma_{\Delta R}(\Delta I)$ від ΔR у логарифмічних координатах дає нам величину H.

Даний метод при роботі з кольоровими зображеннями також як і TPSA і 2D Variation Procedure дозволяє оцінювати як розмірність кожного кольору RGB окремо, так і розмірність "яркісного" зображення.

Таким чином, після того, як кожному пікселю поставимо у відповідність фрактальну розмірність, після фрагментації шкали фрактальної розмірності можна провести відповідну сегментацію зображення.



Фрактальні характеристики виділеної клітини аналізу букального епітелію. (Архів цитопрепаратів крові та кісткового мозку, відділ онкогематології ІЕПОР НАНУ)

Використані зображення

Найпопулярніша жінка у фахівців з комп'ютерної графіки – LenaSoderberg (з обкладинки журналу Playboy за листопад 1972 р.)

Костел святого Миколая (Кам'янське) кінець XIX ст.

Cougar. Кугуар або пума. Кішка, красива до смерті.

Зображення Superfood. Мрія вегана.

Контрольні питання

1. Чим відрізняються растрові зображення від векторних?
2. Що є просторовим фільтром?
3. Чи можна відновити згладжені зображення?
4. Які існують методи сегментації зображень?
5. Що визначає фрактальна розмірність зображення?

Рекомендована література

1. Лигун А.О. Комп'ютерна графіка (Обробка та стиск зображень): навч.посіб./ А.О.Лигун, О.О.Шумейко.-Д.:Біла К.Щ., 2010.-144 с.
2. Шумейко А.А. Интеллектуальный анализ данных (Введение в Data Mining) / А.А.Шумейко, С.Л.Сотник .— Днепропетровск: Белая Е.А., 2012 .— 212 с.
3. Flook A.G., The Use of Dilation Logic on the Quantimet to Achieve Fractal Dimension Characterization of Textured and Structured Profiles // Powder Technology. - 1978.
4. Toshitaka Ikeshoji, Tadashi Shioya Brittle-ductile transition and scale dependence: fractal dimension of fracture surface of materials // Fractals - 1999. - vol. 7, no. 2, - pp. 159-168.
5. Лигун А.А. Асимптотические методы восстановления кривых / А.А.Лигун, А.А.Шумейко .— К.: Изд. Института математики НАН Украины, 1997 .— 358 с.
6. Методы компьютерной обработки изображений/ Под ред. ВА.Сойфера.- М.:Физматлит, 2003.-784 с.
7. Гонсалес Р., Вудс Р. Цифровая обработка изображений. — М.: Техносфера, 2006. — 1072 с.
8. Грузман И.С., Киричук В.С., Косых В.П., Перетягин Г.И., Спектор А.А. Цифровая обработка изображений в информационных системах: Учеб. пособие. — Новосибирск.: Изд-во НГТУ, 2003. — 352 с.
9. Сато Ю. Обработка сигналов. Первое знакомство. 2-е издание. — М.: Додэка XXI, 2009. — 176 с.
10. Оппенгейм А. Шафер Р. Цифровая обработка сигналов. 2-е издание. — М.: Техносфера, 2007. — 856 с.
11. Лайонс Ричард. Цифровая обработка сигналов: 2 изд. — М.: ООО Бином-Пресс, 2006. — 656 с.
12. Сергиенко А.Б. Цифровая обработка сигналов. — СПб.: Питер, 2007. — 752 с.
13. Фисенко В.Т., Фисенко Т.Ю., Компьютерная обработка и распознавание изображений: учеб. пособие. — СПб: СПбГУ ИТМО, 2008. — 192 с.
14. Яне Б. Цифровая обработка изображений. — М.: Техносфера, 2007. — 584 с.
15. https://www.codeproject.com/Articles/751744/Image-Segmentation-using-Unsupervised-Watershed-Al.
16. SLIC Superpixels Compared to State-of-the-art Superpixel Methods / [R.Achanta, A.Shaji, K.Smith та ін.] // Journal of latex class files .— 2011 .— №1(6) .— С.1-8.

ТЕМА 3. СПЕКТРАЛЬНІ МЕТОДИ ОБРОБКИ ЗОБРАЖЕНЬ

Цифрова обробка зображень не обмежується задачами трансформації, які були розглянуті у попередньому розділі. Важливою складовою є кодування інформації, тобто представлення зображення у вигляді, який дозволяє ефективне зберігання та передачу графічної інформації, виділити ті чи інші характеристики, які потрібні під час аналізу об'єктів на даному зображенні, виділити найбільш інформативні характеристика та інше. Для цієї мети використовуються частотні методи аналізу, поперед всього це аналіз Фур'є і аналіз головних компонент. Аналіз Фур'є може використовувати тригонометричний базис, або базис з компактним носієм, що призведе до вейвлет-методів.

Пряме дискретне перетворення Фур'є (ДПФ) і зворотне йому перетворення мають стандартну форму

$$F(\nu) = N^{-1}\sum_{\tau=0}^{N-1} f(\tau)\exp[-i2\pi\nu\tau/N],\quad f(\tau) = \sum_{\nu=0}^{N-1} F(\nu)\exp[i2\pi\nu\tau/N].$$

Часто треба виконати перетворення тільки в дійсній області, без використання комплексних змінних. У цьому випадку традиційним методом, який розділяє частотні характеристики сигналу є дискретне косинус перетворення (DCT), з якого ми і почнемо.

Розглянемо одновимірне ДКП для N=8. Вибір такого значення N не випадковий, у форматі Jpeg зображення розбивається на квадрати 8×8, що і призвело до того, що говорячи про використання частотних методів у обробці зображень використовують як раз таке число складових дискретного перетворення.

Пряме дискретне косинус перетворення

$$F_v = \frac{1}{\sqrt{2}} C_v \sum_{i=0}^{7} p_i \cos\frac{(2i+1)\nu\pi}{16}$$

$$C_i = \begin{cases} \frac{1}{\sqrt{2}}, & i = 0. \\ 1, & i > 0. \end{cases}$$

Зворотне дискретне косинус перетворення

$$p_i = \frac{1}{\sqrt{2}} C_i \sum_{v=0}^{7} F_v \cos\frac{(2i+1)\nu\pi}{16}$$

Ясна річ, що окрім ДКТ існують інші тригонометричні перетворення, наприклад ДСТ – дискретне синус перетворення або перетворення Хартлі.

Цей вид перетворення названий на честь Р. Хартлі, що опублікував в 1942 р. статтю про пару інтегральних перетворень - пряму і зворотну, що використовують введену ним функцію $cas\theta = \sin\theta + \cos\theta$. До початку 1980-х років ці результати залишалися в забутті, поки до них не притягнув увагу дослідників Р. Брейсуэлл, що розробив основи теорії як безперервного, так і дискретного перетворення Хартлі, а також один з варіантів його

Дискретне перетворення Хартлі (ДПХ) дійсної функції f(τ) і відповідне зворотне перетворення визначаються співвідношеннями

$$H(\nu) = N^{-1}\sum_{\tau=0}^{N-1} f(\tau)cas(2\pi\nu\tau/N),$$

$$f(\tau) = \sum_{\nu=0}^{N-1} H(\nu) cas(2\pi\nu\tau / N),$$

де використовується позначення cas θ= cos θ+sin θ, введене Хартлі.
Для отримання зворотного ДПХ скористаємося властивістю ортогональності

$$\sum_{\nu=0}^{N-1} cas(2\pi\nu\tau / N) cas(2\pi\nu\tau' / N) = \begin{cases} N, \tau = \tau' \\ 0, \tau \neq \tau' \end{cases}$$

Підставляючи величину

$$N^{-1} \sum_{\tau=0}^{N-1} f(\tau) cas(2\pi\nu\tau / N),$$

яка визначає перетворення H(v),у вираз

$$\sum_{\nu=0}^{N-1} H(\nu) cas(2\pi\nu\tau / N),$$

отримаємо

$$\sum_{\nu=0}^{N-1} H(\nu) cas(2\pi\nu\tau / N) = \sum_{\nu=0}^{N-1} N^{-1} \sum_{\tau'=0}^{N-1} f(\tau') cas(2\pi\nu\tau' / N) cas(2\pi\nu\tau / N) =$$

$$N^{-1} \sum_{\tau'=0}^{N-1} f(\tau') \sum_{\nu=0}^{N-1} cas(2\pi\nu\tau' / N) cas(2\pi\nu\tau / N) =$$

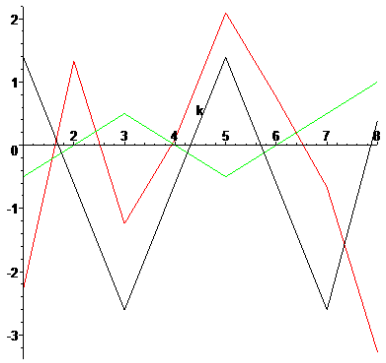
$$N^{-1} \sum_{\tau'=0}^{N-1} f(\tau') \times \begin{cases} N, \tau = \tau' \\ 0, \tau \neq \tau' \end{cases} = f(\tau),$$

що підтверджує справедливість зворотного перетворення.
Коефіцієнт $\mathbf{N}^{-1}$ в ДПХ запозичується з практики використання ДПФ, для якого величина F(0) дорівнює постійній складовій функції $\mathbf{f}(\tau)$; іншими словами, ДПХ є симетричною процедурою. Окрім цього, ДПХ є дійсним перетворенням, оскільки дійсною є функція $\mathbf{f}(\tau)$.

Проведемо порівняння приведених дискретних перетворень на прикладі множини точок***p={12,10,8,10,12,10,8,11}.***Знайдемо ДКТ, ДСТ та ДПХ цієї множини. Проведемо наступний експеримент - отквантуємо отримані результати з точністю до десятків, а для синус перетворення, навіть удвічі точніше.

dct={30,0,0,0,0,0,0,0}- **чорний колір,**
dst={0,25,0,10,0,10,0,10}- **червоний колір,**
dht={80,0,10,0,0,0,10,0}- **зелений колір**

Застосовуючи зворотні перетворення, відновимо дані і знайдемо помилку, яка візуалізована нижче



Помилка відновлення даних по відквантованим частотним коефіцієнтам.

Як видно навіть з наведеного простого прикладу, найгіршу якість показує синус перетворення, а найкращу – перетворення Хартлі. Але не все так просто, тому що кількість ненульових коефіцієнтів ДКТ менша за кількість перетворення Хартлі.

Отримані результати говорять про перспективність перетворення Хартлі. Для того, щоб перевірити і переконатися в ефективності дискретного перетворення Хартлі перед косинус - і синус-перетворенням,побудуємо норми частотних доменів, отриманих з використанням відповідним перетворень.

Використовуючи ідеологію JPEG, розіб'ємо зображення на квадрати із стороною у вісім пікселів і на кожному квадраті вичислимо відповідні частотні . Для DCT це

$$c_{i,j}^{k} = \frac{1}{4} C_i C_j \sum_{x=0}^{7} \sum_{y=0}^{7} p_{x,y} \cos\left(\frac{(2x+1)i\pi}{16}\right) \cos\left(\frac{(2y+1)j\pi}{16}\right),$$

$$\text{де } C_i = \begin{cases} \frac{1}{\sqrt{2}}, & i = 0, \\ 1. & i > 0. \end{cases}$$

для DST

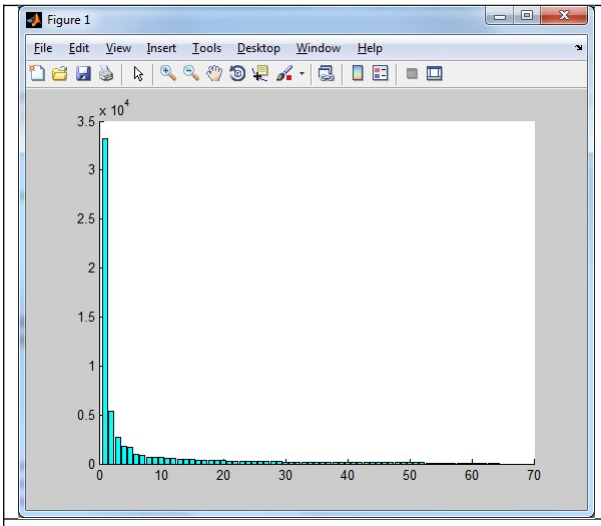
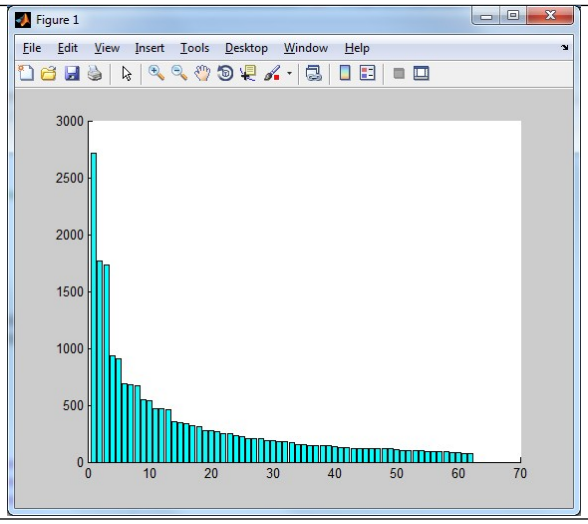
$$s_{i,j}^{k} = \frac{1}{4} C_i C_j \sum_{x=0}^{7} \sum_{y=0}^{7} p_{x,y} \sin\left(\frac{(2x+1)i\pi}{16}\right) \sin\left(\frac{(2y+1)j\pi}{16}\right),$$

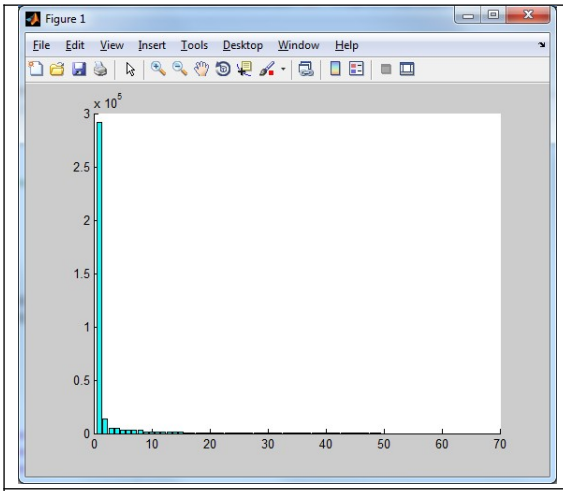
і для DHT

$$h_{i,j}^{k} = \sum_{x=0}^{7} \sum_{y=0}^{7} p_{x,y} cas\left(\frac{2ix\pi}{16}\right) cas\left(\frac{2jy\pi}{16}\right),$$

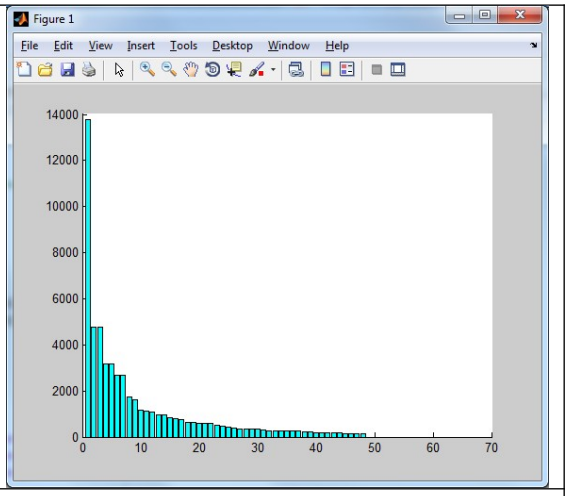
де k - номер квадрата.

Якщо $D(q_{i,j}) = \{ q_{i,j}^{k} \mid k = 0,1,... \}$, то норма цього частотного домену обчислюється за правилом $\|D(q_{i,j})\| = \sqrt{\sum_{k} (q_{i,j}^{k})^2}$. По суті, норма, це енергія, що відповідає коефіцієнтам домену. Чим більше норми, тим більше інформації акумулює цей домен. Тому, про ефективність того або іншого перетворення можна судити по тому, наскільки швидко убувають норми частотних доменів. Для тестового зображення Lena

	
Дискретне косинус-перетворення дає наступний розподіл	Цей же розподіл, але без першого(низькочастотного) домену.

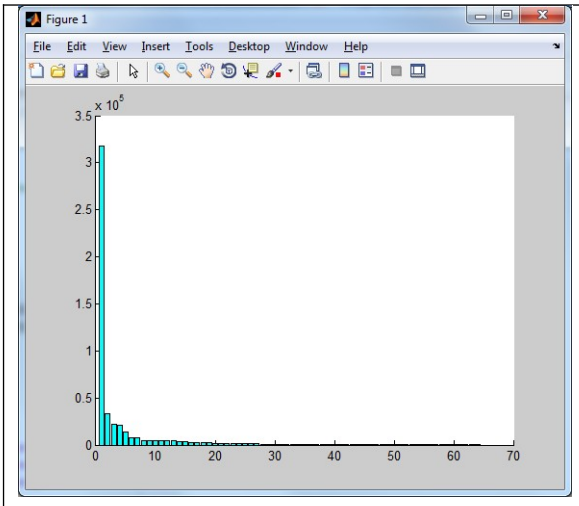


Синус-перетворення дає наступний розподіл

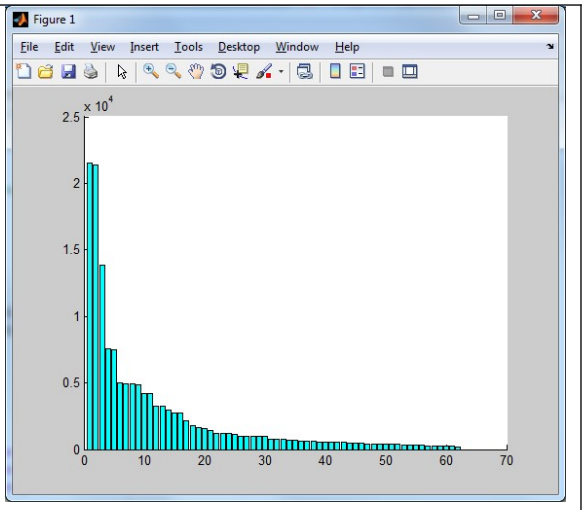


І без низькочастотного домену

Нарешті, перетворення Хартлі ілюструють діаграми



Норми доменів перетворення Хартлі



Ці ж домени без низькочастотного

Щоб порівняти ефективність кодування ДКТ та ДПХ (синус перетворення зразу відкинемо як неефективне, цей факт зразу витікає з аналізу гістограм енергії доменів), візьмемо Jpeg і замінимо ДКТ на ДПХ.

Коротко зупинимося на методі стиску зображень JPEG. Схематично метод JPEG полягає в наступному:

1. Кольорове зображення перетворюється з простору змішування кольорів RGB в простір YCrCb, де Y- люмінесцентна складова (освітленість), Cr- компонента теплих відтінків і Cb- компонента холодних відтінків.

2. На другому етапі відбувається проріджування пікселів (при високих ступенях стиснення з чотирьох значень Cr і Cb беруться не всі, або два або одне значення).

3. Групування значень Y, Cr і Cb в блоки 8×8 з подальшою фільтрацією, шляхом застосування дискретного косинус-перетворення (DCT).

4. Квантування отриманих коефіцієнтів, шляхом ділення кожного коефіцієнта косинус-перетворення Фур'є на спеціальне число (QC-коефіцієнт квантування), з

наступним округленням до цілого. Саме на цьому етапі йде непоправна втрата інформації, за рахунок чого, власне, і станься істотне стиснення зображення.

5. Далі відквантовані значення упорядковуються (по змійці JPEG) і стискаються методом RLE (Run-lengthEncoding) - стиснення з використанням лічильників нулів.

6. І на останньому етапі застосовується префіксний або ентропійний метод стиску даних без втрат.

У якості критерію оцінки відновлення оригінального зображення $I(i,j)$ з зображенням $K(i,j)$, $(i=0,1,\ldots,m-1,\ j=0,1,\ldots,n-1)$, використаємо відношення сигнал/шум

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20\log_{10}\frac{MAX_I}{\sqrt{MSE}},$$

де

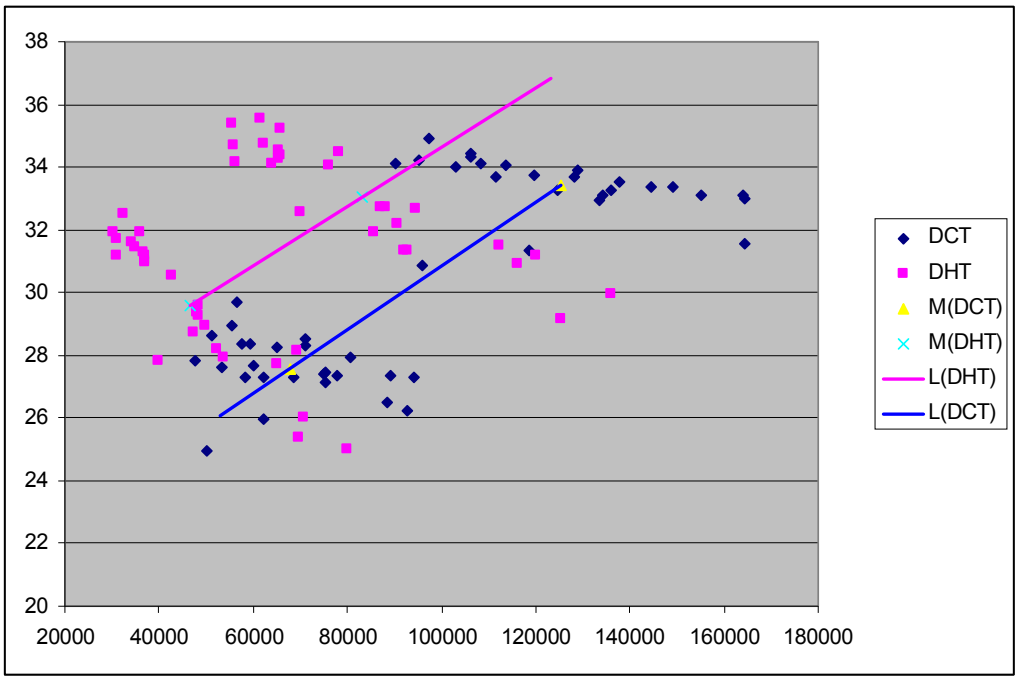
$$MSE = \frac{1}{nm}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left|I(i,j)-K(i,j)\right|^2 .$$

Для тестування використаємо зображення тестовоїбази TID2008 фірми Kodak.

INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" * MERGEFORMATINET INCLUDEPICTURE "http://www.computervisiononline.com/sites/default/files/TID2008.png" *

MERGEFORMATINET

Результати експериментів об’єднаємо на одному графіку.
Вісь абсцис відповідає розміру файлу, вісь ординат - значення PSNR. Точки синього кольору відповідають DCT, пурпурного - DHT. Видно, що вони утворюють по два кластери, що відповідають режимам стиску. Крім того, наведені дві лінійні апроксимації, що з'єднують середні значення кожного кластера. Дана ілюстрація показує істотне поліпшення якості стиснення зображення при використанні DHT замість DCT.



На підставі викладених результатів випливає, що використання дискретного перетворення Хартлі дозволяє отримати метод стиснення зображень перевищує JPEG як за ступенем стиснення, так і за якістю відновлених зображень.
Очевидно, що

$$\sum_{i=0}^{N} p_i \operatorname{cas}\frac{2i\nu\pi}{N} = \sqrt{2}\sum_{i=0}^{7} p_i \cos\left(\frac{2i\nu\pi}{N} - \frac{\pi}{4}\right)$$

тобто, перетворення Хартлі представляє собою перетворення Фур'є по косинусу з фазовим зрушенням $\frac{\pi}{4}$ і це дає непогані результати. Таким чином виникає питання, а чи існує перетворення по косинусу з будь-яким кутом фазового зрушення.

Одновимірне дискретне тригонометричне перетворення з вільним фазовим зрушенням

Теорема. Нехай $\varphi \in \left(0, \frac{\pi}{2}\right)$, тоді для всіх $\{h_m\}_{m=0}^{N-1}$ таких, що $-\infty < h_m < \infty, m = 0,1,\ldots,N-1$ та

$$H_k = \sum_{m=0}^{N-1} h_m \cos\left(\frac{2\pi mk}{N} - \varphi\right);$$

Має місце рівність

$$h_n = \frac{2}{N\sin(2\varphi)}\sum_{k=0}^{N-1} H_k \sin\left(\frac{2\pi nk}{N} + \varphi\right).$$

Доведення. Припустимо, що це насправді так. Підставимо в друге рівняння значення H_k з першої рівності, тоді отримаємо

$$\frac{2}{N\sin(2\varphi)}\sum_{k=0}^{N-1}\left(\sum_{m=0}^{N-1} h_m \cos\left(\frac{2\pi mk}{N} - \varphi\right)\right)\sin\left(\frac{2\pi nk}{N} + \varphi\right) =$$

$$= \frac{2}{N\sin(2\varphi)}\sum_{m=0}^{N-1} h_m \sum_{k=0}^{N-1}\cos\left(\frac{2\pi mk}{N} - \varphi\right)\sin\left(\frac{2\pi nk}{N} + \varphi\right) =$$

$$= \frac{1}{N\sin(2\varphi)}\sum_{m=0}^{N-1} h_m \sum_{k=0}^{N-1}\left(\sin\left(\frac{2\pi k(n+m)}{N}\right) + \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right)\right).$$

Примітимо, що

$$\sum_{\nu=0}^{N-1}\sin(\alpha + \nu\beta) = \frac{1}{\sin\frac{\beta}{2}}\sin\left(\alpha + \frac{N-1}{2}\beta\right)\sin\frac{N\beta}{2}, \qquad (3.1)$$

Нехай, спочатку, $n + m \neq N$, тоді з (3.1) отримаємо,

$$\sum_{k=0}^{N-1}\sin\left(\frac{2\pi k(n+m)}{N}\right) = \frac{1}{\sin\left(\frac{2\pi(n+m)}{2N}\right)}\sin\left(\frac{2\pi(n+m)(N-1)}{2N}\right)\sin\left(\frac{2\pi(n+m)N}{2N}\right).$$

З умови $n + m \neq N$ отримуємо

$$\sin\left(\frac{\pi(n+m)}{N}\right) \neq 0,$$

а, з іншого боку,

$$\sin\left(\frac{2\pi(n+m)N}{2N}\right) = \sin(\pi(n+m)) = 0,$$

тобто, для $n+m \neq N$

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n+m)}{N}\right) = 0 .$$

Якщо ж $n+m = N$, тоді

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n+m)}{N}\right) = \sum_{k=0}^{N-1} \sin\left(\frac{2\pi kN}{N}\right) = \sum_{k=0}^{N-1} \sin(2\pi k) = 0 ,$$

Таким чином для всіх $n, m = 0,..., N-1$

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n+m)}{N}\right) = 0 .$$

Роздивимось зараз,

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right) .$$

Використовуючи рівняння (1), маємо

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right) = \frac{1}{\sin\left(\frac{2\pi(n-m)}{2N}\right)} \sin\left(\frac{2\pi(n-m)(N-1)}{2N} + 2\varphi\right) \sin\left(\frac{2\pi(n-m)N}{2N}\right)$$

Далі, примітимо, що так, як $n, m = 0,..., N-1$, то $n-m \neq N$. Нехай, зараз $n \neq m$, тоді отримаємо, що

$$\sin\left(\frac{\pi(n-m)}{N}\right) \neq 0,$$

але, при цьому,

$$\sin\left(\frac{2\pi(n-m)N}{2N}\right) = \sin(\pi(n-m)) = 0$$

для всіх $n, m = 0,..., N-1$ та для $n \neq m$.

І, під кінець, нехай $n = m$, тоді

$$\sum_{k=0}^{N-1} \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right) = \sum_{k=0}^{N-1} \sin(0 + 2\varphi) = N\sin(2\varphi) .$$

Таким чином, отримаємо

$$\sum_{k=0}^{N-1} \left(\sin\left(\frac{2\pi k(n+m)}{N}\right) + \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right)\right) = \begin{cases} N\sin(2\varphi), & n = m, \\ 0, & n, m = 0,1,..., N-1, n \neq m. \end{cases}$$

Звідси виходить рівняння

$$\frac{1}{N\sin(2\varphi)} \sum_{m=0}^{N-1} h_m \sum_{k=0}^{N-1} \left(\sin\left(\frac{2\pi k(n+m)}{N}\right) + \sin\left(\frac{2\pi k(n-m)}{N} + 2\varphi\right)\right) = \begin{cases} h_n, & m = n, \\ 0, & n, m = 0,1,...,N-1, n \neq m, \end{cases}$$

Що і є завершенням доказу теореми.

Зазначимо, що, якщо $\varphi = \frac{\pi}{4}$ то ми отримаємо дискретне перетворення Хартлі (ДПХ), тобто ДПХ є окремим випадком ДТП

$$P_i = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} p_k cas\left(\frac{2ik\pi}{N}\right), \text{ де } cas\varphi = \cos\varphi + \sin\varphi .$$

Використання можливості використання фазового зсуву для поліпшення відновлення даних дозволяє, на підставі отриманого дискретного перетворення, будувати адаптивні фільтри, підлаштовуючи фільтр не тільки для вхідних даних, але і, наприклад, для використовуваного методу квантування або природи шуму, що вносить спотворення в сигнал.

Двовимірне дискретне тригонометричне перетворення з вільним фазовим зрушенням

Так як одною з популярних сфер використання дискретного косинус-перетворення є цифрова обробка двомірних сигналів, тобто зображень, то для використання отриманого дискретного тригонометричного перетворення в обробці зображень існує двомірне перетворення:

прямий хід

$$h_{i,j} = \frac{2}{N\sin(2\varphi)} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} p_{n,m} \cos\left(\frac{2\pi in}{N} - \varphi\right) \cos\left(\frac{2\pi jm}{N} - \psi\right),$$

та зворотний хід

$$p_{n,m} = \frac{2}{N\sin(2\psi)} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} h_{i,j} \sin\left(\frac{2\pi in}{N} + \psi\right) \sin\left(\frac{2\pi jm}{N} + \varphi\right).$$

Використовуючи це перетворення для блоку 8×8 пікселів, можна отримати кращий результат ніж дискретне косинус-перетворення, але для отримання такого результату необхідно вибрати правильний фазове зрушення. Цим дискретним тригонометричним перетворенням можна замінити дискретне косинус-перетворення в стандарті JPEG без зміни логіки кодування та квантування, лише замінивши перетворення.

Визначення найкращого фазового зрушення

Для визначення найкращого фазового зрушення, який для двомірного сигналу складається з двох параметрів φ та ψ, запропоновано наступний метод:

1. Треба вирахувати значення Peak Signal to Noise Ration(PSNR) для відновленого зображення, при майже крайніх значеннях зрушення, наприклад $\varphi_1 = \pi/100$, $\varphi_2 = \pi/6$ та $\varphi_3 = \pi/3$, значення ψ краще взяти середнє, тобто $\pi/4$.

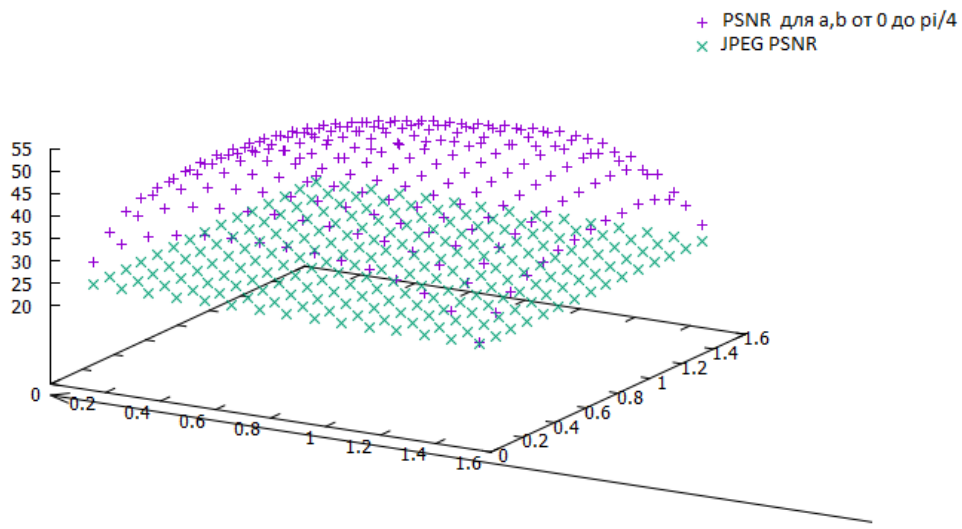
2. Знайти точку максимуму параболи проведеної через точки A($PSNR_1$; φ_1), B($PSNR_2$; φ_2) та C($PSNR_3$; φ_3).

3. Координата X знайденої точки буде апроксимацією шуканого значення φ.

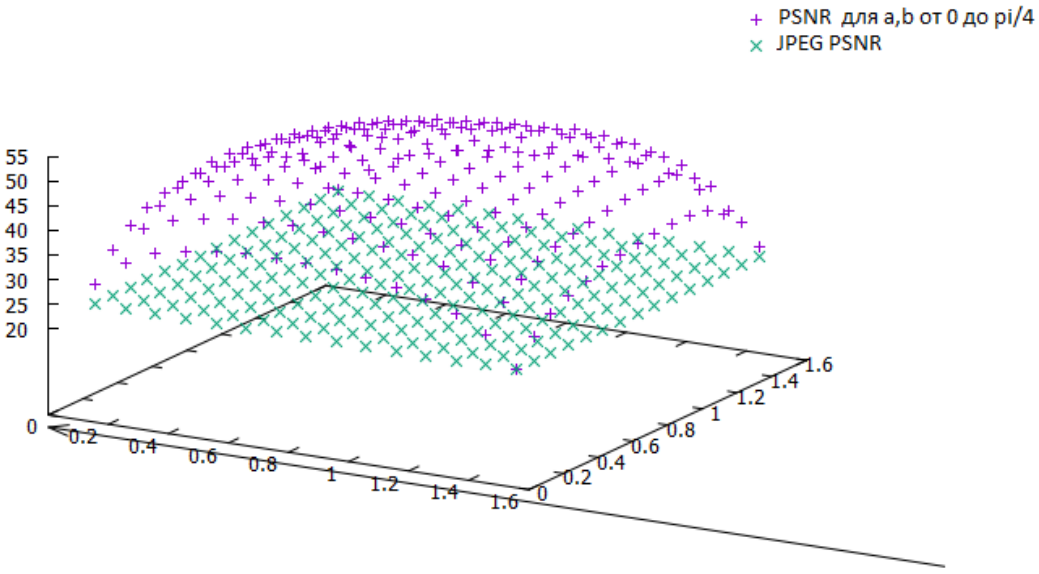
4. Перейти до кроку 1 та провести цю операцію для ψ використовуючи знайдене значення φ, дану операцію можна повторювати безліч разів використовуючи по черзі φ та ψ, але практика показала, що і одного разу достатньо.

Цей метод також підходить для одновимірного сигналу. Дана операція займає багато часу, так як зображення повністю кодується та декодується мінімум 6 разів, але вона дозволяє отримати найкращі значення φ та ψ. Також можна розрахувати значення PSNR для усіх значень φ та ψ з певним кроком та вибрати найкращій результат, але наведений метод дозволяє значно скоротити кількість обчислень перетворення та надає більш точний результат.

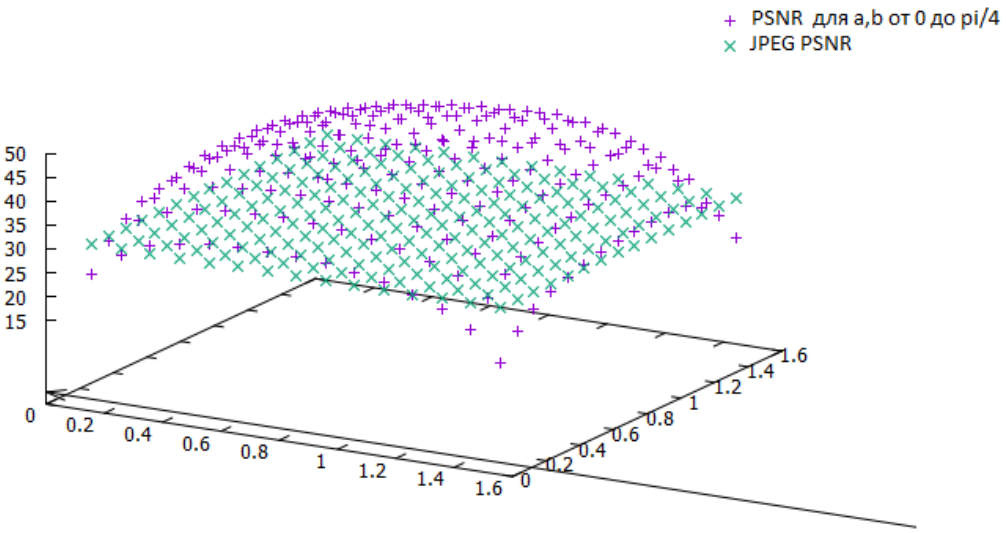
Для тестування було взято тестові зображення з бази даних зображень TID2008 фірми Kodak. Для кожного зображення було визначено кращі значення зрушень, а також побудовані графіки кожного значення PSNR в залежності від фазових зрушень



Розподілення значень PSNR для різних значень φ та ψ на портретному зображенні



Розподілення значень PSNR для різних значень φ та ψ на пейзажному зображенні



Розподілення значень PSNR для різних значень φ та ψ на синтетичному зображенні

Розглянемо наступну задачу: як вибрати базисні вектори, щоб мінімальним числом базисних векторів можна було найкращим чином наблизити дані з деякого набору.
Вирішенню цієї задачі відповідає метод головних компонент - ортогональне лінійне перетворення базису, при якому перший вектор нового базису відповідає напрямку максимальної дисперсії даних, другий вектор - наступного напрямку максимальної дисперсії і так інше.
У реальних задачах комп'ютерної графіки досить продуктивна інша евристика, яка ґрунтується на гіпотезі про наявність «сигналу» (порівняно мала розмірність, відносно велика амплітуда) і «шуму» (велика розмірність, відносно мала амплітуда). З цієї точки зору метод головних компонент працює як фільтр: сигнал міститься, в основному, в проекції на перші головні компоненти, а в інших компонентах пропорція шуму набагато вище.

Метод головних компонент
Метод головних компонент (МНК) — один із основних способів скорочення і даних із зменшенням мінімальної кількості інформації, розроблений Карлом Пірсоном (Karl Pearson) у 1901 р. Використовується у багатьох областях, таких як розпізнавання образів, комп'ютерний зір, стиск даних та ін. Знаходження головних компонент зводиться до обчислення власних векторів та значень коваріаційної матриці вхідних даних. Перейдемо до розглядання метода МГК (PCA). Спочатку знайдемо константу μ, яка найкращим чином описує вхідні дані

$$\varepsilon(\mu) = \sum_{i=1}^{n} (x_i - \mu)^2 \to \min_{\mu}.$$

Для знаходження мінімуму знайдемо похідну і прирівняємо її до нуля, після чого знайдемо значення μ, яке дає мінімум

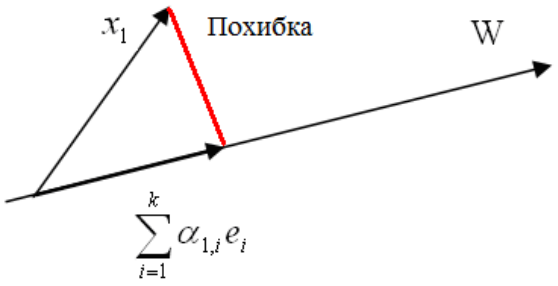
$$\frac{d}{d\mu}\varepsilon(\mu) = -2\sum_{i=1}^{n}(x_i - \mu) = 0 \Rightarrow \sum_{i=1}^{n}\mu = \sum_{i=1}^{n} x_i \Rightarrow \mu n = \sum_{i=1}^{n} x_i \Rightarrow \mu = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Далі проведемо центрування даних, тобто, переозначимо вхідні значення $x_{new} = x_{old} - \mu$, віднімаючи з кожного отримане значення μ. Ясно, що нові дані мають математичне очікування, яке дорівнює нулю

$$E(X - E(X)) = E(X) - E(X) = 0.$$

Таким чином, ми зробили паралельний зсув в існуючий системі координат.
У подальшому будемо вважати, що наші дані центровані и ми маємо бажання знайти найбільш точне представлення даних $D = \{x_1, ..., x_n\}$ у деякому підпросторі W з розмірністю k < n.
Нехай $\{e_1, ..., e_k\}$ ортонормований базис W. Довільний вектор із W може бути записаний у вигляді лінійної комбінації векторів базису, тобто, x_1 можна поставити у відповідність деякий вектор $\sum_{i=1}^{k} \alpha_{1,i} e_i$ із W. Похибка між ними обчислюється наступним чином

$$\varepsilon_1 = \left\| x_1 - \sum_{i=1}^{k} \alpha_{1,i} e_i \right\|_2^2 = \left\langle x_1 - \sum_{i=1}^{k} \alpha_{1,i} e_i, x_1 - \sum_{i=1}^{k} \alpha_{1,i} e_i \right\rangle.$$



Ілюстрація похибки відновлення вектору

Щоб знайти повну похибку, нам треба знайти суму величини похибок по всім x_j, тому повна похибка дорівнює

$$\varepsilon\underbrace{(e_1,...,e_k,\alpha_{1,1},...,\alpha_{n,k})}_{unknowns} = \sum_{j=1}^{n} \varepsilon_j^2 = \sum_{j=1}^{n} \left\| x_j - \sum_{i=1}^{k} \alpha_{j,i} e_i \right\|_2^2. \qquad (3.2)$$

Щоб мінімізувати похибку, треба взяти часткові похідні і урахувати обмеження на ортогональність $\{e_1,...,e_k\}$.

По-перше, спростимо співвідношення (3.2)

$$\varepsilon(e_1,...,e_k,\alpha_{1,1},...,\alpha_{n,k}) = \sum_{j=1}^{n} \left\| x_j - \sum_{i=1}^{k} \alpha_{j,i} e_i \right\|_2^2 = \sum_{j=1}^{n} \|x_j\|_2^2 - 2\sum_{j=1}^{n} x_j^T \sum_{i=1}^{k} \alpha_{j,i} e_i + \sum_{j=1}^{n}\sum_{i=1}^{k} \alpha_{j,i}^2 =$$

$$= \sum_{j=1}^{n} \|x_j\|_2^2 - 2\sum_{j=1}^{n}\sum_{i=1}^{k} \alpha_{j,i} x_j^T e_i + \sum_{j=1}^{n}\sum_{i=1}^{k} \alpha_{j,i}^2.$$

Тоді

$$\frac{\partial}{\partial \alpha_{m,l}} \varepsilon(e_1,...,e_k,\alpha_{1,1},...,\alpha_{n,k}) = -2x_m^T e_l + 2\alpha_{m,l}.$$

Необхідна та достатня умова екстремуму буде мати вигляд

$$-2x_m^T e_l + 2\alpha_{m,l} = 0 \Rightarrow \alpha_{m,l} = x_m^T e_l.$$

Таким чином, похибка (3.2) буде мати вигляд

$$\varepsilon(e_1,...,e_k) = \sum_{j=1}^{n} \|x_j\|_2^2 - 2\sum_{j=1}^{n}\sum_{i=1}^{k} (x_j^T e_i) x_j^T e_i + \sum_{j=1}^{n}\sum_{i=1}^{k} (x_j^T e_i)^2.$$

Проводячи спрощення, маємо

$$\varepsilon(e_1,...,e_k) = \sum_{j=1}^{n} \|x_j\|_2^2 - \sum_{j=1}^{n}\sum_{i=1}^{k} (x_j^T e_i)^2. \qquad (3.3)$$

Зважаючи на те, що $\langle a,b\rangle = a^T b$ и $\langle b,a\rangle = \langle a,b\rangle$, отримуємо

$$(a^T b)^2 = (a^T b)(a^T b) = (b^T a)(a^T b) = b^T (aa^T) b,$$

тобто,

$$\varepsilon(e_1,...,e_k) = \sum_{j=1}^{n} \|x_j\|_2^2 - \sum_{i=1}^{k} e_i^T \left(\sum_{j=1}^{n} (x_j x_j^T) \right) e_i = \sum_{j=1}^{n} \|x_j\|_2^2 - \sum_{i=1}^{k} e_i^T S e_i,$$

де $S = \sum_{j=1}^{n} (x_j x_j^T)$. є коваріаційною матрицею.

Наступним кроком буде мінімізація $\varepsilon(e_1,...,e_k) = \sum_{j=1}^{n} \|x_j\|_2^2 - \sum_{i=1}^{k} e_i^T S e_i$ при умові $e_i^T e_i = 1$ для всіх *i*. Для розв'язку цієї задачі будемо використовувати метод невизначених

множників Лагранжа (Lagrange), введемо множники $\lambda_1,...,\lambda_k$ і, зазначаючи, що $\sum_{j=1}^{n}\|x_j\|_2^2 \equiv Const$, випишемо функцію мети

$$\ell(e_1,...,e_k)=\sum_{i=1}^{k} e_i^T S e_i - \sum_{i=1}^{k}\lambda_i\left(e_i^T e_i - 1\right).$$

Зазначаючи, що $\frac{d}{dX}(X^T X)=\frac{d}{dX}\langle X,X\rangle = 2X$ і якщо A симетрична матриця, то $\frac{d}{dX}(X^T AX)=2AX$, маємо,

$$\frac{\partial}{\partial e_m}\ell(e_1,...,e_k)=2Se_m - 2\lambda_m e_m = 0,$$

тобто, $Se_m=\lambda_m e_m$. Таким чином, необхідно знайти рішення рівняння $(S-\lambda I)e=0$ (де I одинична матриця), що еквівалентно тому, що λ_m та e_m є власні значення і власні вектори коваріаційної матриці S.

При цьому похибка має вигляд

$$\varepsilon(e_1,...,e_k)=\sum_{j=1}^{n}\|x_j\|_2^2 - \sum_{i=1}^{k}\lambda_i\|e_i\|_2^2 = \sum_{j=1}^{n}\|x_j\|_2^2 - \sum_{i=1}^{k}\lambda_i \quad . \tag{3.4}$$

Мінімізація (3.4) складається у виборі базису W із k власних векторів матриці S, які відповідають k найбільшим власним значенням. Більше власне значення S дає більшу варіацію у напрямі відповідного власного вектору. Цей результат можна переформулювати наступним чином – проекція X на підпростір розміру k, яка забезпечує найбільшу варіацію. Таким чином МГК може трактуватися наступним чином - беремо ортогональний базис і обертаємо його поки на одному із напрямів не отримаємо максимальну варіацію. Фіксуємо цей напрям и обертаємо інші, поки не знайдемо другий напрям і так далі.

Нехай $\{e_1,...,e_n\}$ всі власні вектори матриці S, відсортовані у порядку зменшення відповідного власного значення, тоді для довільного

$$x_i=\sum_{j=1}^{n}\alpha_{i,j}e_j = \underbrace{\alpha_{i,1}e_1+...+\alpha_{i,k}e_k}_{approximation} + \overbrace{\alpha_{i,k+1}e_{k+1}+...+\alpha_{i,n}e_n}^{error}$$

коефіцієнти $\alpha_{m,l}=x_m^T e_l$ є координатами головних компонент, і чим більше значення k, тим кращу апроксимацію отримуємо. При цьому головні компоненти стоять у порядку значення, більш важливі мають менший номер.

Приведемо алгоритмізацію МГК .

Нехай $D=\{x_1^0,...,x_n^0\}$ вхідні (оригінальні) дані, кожен із цих векторів x_i^0 має розмір N

1. Знайдемо середнє $\mu=\frac{1}{n}\sum_{i=1}^{n}x_i^0$.
2. Віднімемо середнє із кожного вектору $x_i=x_i^0-\mu$.
3. Знайдемо коваріаційну матрицю $S=\sum_{j=1}^{n}x_j x_j^T$.
4. Обчислимо власні вектори $\{e_1,...,e_k\}$, які відповідають k найбільшим власним значенням S.
5. Нехай $\{e_1,...,e_k\}$ складають матрицю $E=\{e_1...e_k\}$.
6. Тоді найближче до *x* лежить $z=E^T x$.

Ітераційний алгоритм обчислення головних компонент

Описаний метод отримання головних компонент є достатньо ресурсоємним та нестабільним, особливо у випадку, якщо власні значення матриці близькі до нуля.

Тому більш ефективним є використання ітераційного методу знаходження головних компонент. Для цієї мати розглянемо задачу (3.2) з іншою точки зору.

Для випадку *i=1* задача (3.2) зводиться до знаходження однієї компоненти e_1, яка найкращим чином відновлює всі вхідні дані $\{x_1,...,x_n\}$

$$\varepsilon(e_1,\alpha_{1,1},...,\alpha_{n,1}) = \sum_{j=1}^{n} \left\| x_j - \alpha_{j,1} e_1 \right\|_2^2 \to \min \tag{3.5}$$

по всім e_1 та $\{\alpha_{i,1}\}_{i=1}^{n}$ при умові $\sum_{i=1}^{n} \alpha_{i,1}^2 = 1$.

Якщо $\{\widetilde{\alpha}_{i,1}\}_{i=1}^{n}$ та $\widetilde{e}_1$ є розв'язок цієї задачі та $\Delta x_j = x_j - \widetilde{\alpha}_{j,1}\widetilde{e}_1$ - похибка відновлення даних однією першою головною компонентою, то розв'язуючи задачу

$$\sum_{j=1}^{n} \left\| \Delta x_j - \alpha_{j,2} e_2 \right\|_2^2 \to \min$$

по всім e_2 та $\{\alpha_{i,2}\}_{i=1}^{n}$ при умові $\sum_{i=1}^{n} \alpha_{i,1}^2 = 1$, отримуємо другу головну компоненту $\widetilde{e}_2$ і відповідний вектор $\{\widetilde{\alpha}_{i,2}\}_{i=1}^{n}$ і так інше.

При фіксованих $\{\alpha_{i,1}\}_{i=1}^{n}$ задача (3.5) розв'язується методом найменших квадратів. В силу того, що функція мети є квадратичний функціонал, необхідна та достатня умова екстремуму співпадають. Таким чином, рішення задачі зводиться до пошуку розв'язку рівняння

$$\frac{\partial}{\partial e_1}\varepsilon(e_1,\alpha_{1,1},...,\alpha_{n,1}) = -2\sum_{j=1}^{n}(x_j - \alpha_{j,1}e_1)\alpha_{j,1} = -2\left(\sum_{j=1}^{n} x_j\alpha_{j,1} - \sum_{j=1}^{n}\alpha_{j,1}^2 e_1\right).$$

Звідси маємо

$$e_1 = \frac{\sum_{j=1}^{n} x_j\alpha_{j,1}}{\sum_{j=1}^{n}\alpha_{j,1}^2},$$

враховуючи умову нормування одиницею, тобто $\sum_{i=1}^{n} \alpha_{i,1}^2 = 1$, маємо

$$e_1 = \sum_{j=1}^{n} x_j\alpha_{j,1}.$$

Наступний крок будемо робити з умови, що в задачі (3.5) нам відома компонента e_1 і треба знайти екстремум по $\{\alpha_{i,1}\}_{i=1}^{n}$

$$\frac{\partial}{\partial \alpha_{\nu,1}}\varepsilon(e_1,\alpha_{1,1},...,\alpha_{n,1}) = -2(x_\nu - \alpha_{\nu,1}e_1)e_1 = -2(\langle x_\nu, e_1\rangle - \alpha_{\nu,1}\langle e_1, e_1\rangle) = 0,$$

тобто

$$\alpha_{\nu,1} = \frac{\langle x_\nu, e_1\rangle}{\langle e_1, e_1\rangle},$$

де,як звичайно, $\langle x, y\rangle$ - скалярний добуток векторів *x* та *y*.

Далі, вважаючи знайдені $\{\alpha_{i,1}\}_{i=1}^{n}$ вже відомими, повторюємо весь процес, доки не досягнемо стабілізації похибки. Отримані e_1 будемо вважати першою головною

компонентою $\widetilde{e}_1$. Тоді $\Delta x_j = x_j - \widetilde{\alpha}_{j,1}\widetilde{e}_1$ - похибка відновлення даних однією першою головною компонентою.

Застосовуючи цей алгоритм до похибки відновлення Δx_j , знаходимо другу головну компоненту e_2 разом з коефіцієнтами $\alpha_{j,2}$,і так інше.

Наведемо алгоритмізацію цього алгоритму.

По-перше центруємо дані, віднімаючи від вхідних даних середнє значення і в подальшому вважаємо, що дані у середньому дорівнюють нулю.

1. Нехай номер ітерації $\nu = 1$.

2. Вибираємо стартові значення $\left\{\alpha_{i,1}^{\nu}\right\}_{i=1}^{n}$, наприклад, нехай всі вони між собою рівні, тобто $\alpha_{i,1}^{\nu} = \frac{1}{\sqrt{n}}, i = 1,2,...,n.$

3. Обчислимо $e_1^{\nu} = \sum_{j=1}^{n} x_j \alpha_{j,1}^{\nu}$.

4. Далі знайдемо $\beta_i = \frac{\left\langle x_i, e_1^{\nu}\right\rangle}{\left\langle e_1^{\nu}, e_1^{\nu}\right\rangle}$, і, після нормування одиницею,маємо

$$\alpha_{i,1}^{\nu+1} = \frac{\beta_i}{\sqrt{\sum_{j=1}^{n} \beta_j^2}}.$$

5. Нехай $\nu = \nu + 1$.

6. Проведемо перевірку критерію зупинки, в якості якого може бути або стабілізація коефіцієнтів $\left\{\alpha_{i,1}^{\nu}\right\}_{i=1}^{n}$, або стабілізація головної компоненти e_1^{ν}, або перевірка на задане фіксоване число ітерацій. Якщо умова зупинки закінчення ітераційного процесу не виконане, то переходимо до пункту 3.

Перехід від моделі RGB до оптимальної трикомпонентної моделі

Перейдемо до використання метода головних компонент в комп'ютерній графіці. Тут важливу роль грає конвертація зображення із простору рівноправних кольорових характеристик в простір нерівноправних. Довільне зображення візуалізується з використанням змішування рівноправних кольорових компонент - червоної, зеленої та синьої складових - модель RGB. В якості нерівноправних кольорових компонент , як правило, використовують, відповідно, також три компоненти – значення освітленості (люмінесцентну складову), характеристику теплих тонів и характеристику холодних тонів. Використання нерівноправних кольорових компонент використовують для стиску зображень та відео-потоків,застосовуючи до кожної із нерівноправних компонент свій метод стиску.

Можна підходити до проблеми побудови нерівноправного кольорового простору з іншої точки зору, виходячи із максимальної інформативності кожної компоненти. Застосуємо метод головних компонент для отримання нерівноправної трикомпонентної моделі оптимальної с точки зору мінімізації середньоквадратичної похибки відновлення даного зображення. Таким чином, перша із отриманих кольорових компонент буде нести найбільшу інформацію щодо зображення між всіх отриманих кольорових компонент, а друга буде мати найбільшу інформацію серед тих, що осталися.

Таким чином, в нашій термінології, задача (2.1) буде мати вигляд

$$\left\| R - \sum_{i=1}^{3} \alpha_{r,i} e_i \right\|_2^2 + \left\| G - \sum_{i=1}^{3} \alpha_{g,i} e_i \right\|_2^2 + \left\| B - \sum_{i=1}^{3} \alpha_{b,i} e_i \right\|_2^2 \to \min ,$$

де мінімум береться по всім $\alpha_{r,i}, \alpha_{g,i}, \alpha_{b,i}$ и e_i *i=1,2,3.*
В якості даних розглянемо тестове зображення Lena.
Застосовуючи метод головних компонент, маємо

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 0.767785 & 0.45439 & 0.4517034 \\ -0.6164395 & 0.716085 & 0.3274513 \\ -0.174667 & -0.5298553 & 0.8299064 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$$

.

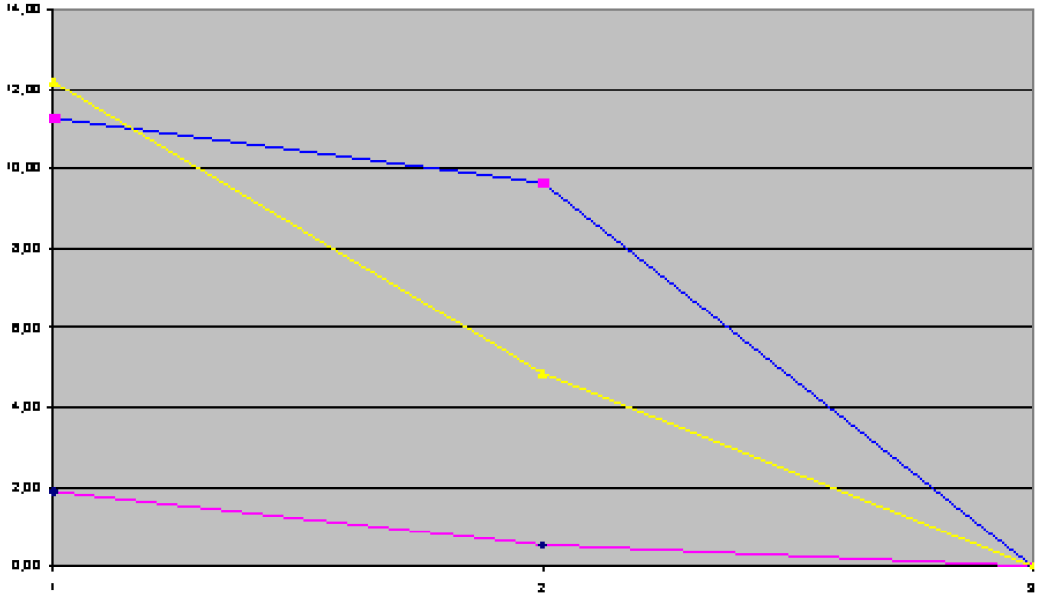
Тут у першому стовбці матриці стоять коефіцієнти $\alpha_{r,1}, \alpha_{r,2}, \alpha_{r,3}$, у другому - $\alpha_{g,1}, \alpha_{g,2}, \alpha_{g,3}$ і в третьому - $\alpha_{b,1}, \alpha_{b,2}, \alpha_{b,3}$. Тоді відновлення тестового зображення по одній компоненті можна записати у вигляді

$$R_{i,j} = 0.767785\ Y_{i,j},$$
$$G_{i,j} = 0.45439\ Y_{i,j},$$
$$B_{i,j} = 0.4517034\ Y_{i,j},$$

де $Y_{i,j}$ – значення першої головної компоненти e_1, які відповідають пікселю з координатами (i,j).

Зображення Lena, відновлене по одній першій головній компоненті

Порівняємо відновлення зображення компонентами різних просторів кольорів



Синім кольором позначена похибка відновлення зображення по одній, за двома або повна реконструкція трьома компонентами простору YUV, жовтим кольором – YCrCb, а пурпуровим – по головним компонентам.

Оптимальний метод кодування зображення

У розглянутому вище прикладі зображення було розкладене на три оптимальних (з точки зору енергії прошарків) інформаційних домени (прошарків). Узагальнимо даний підхід використовуючи для розбиття зображення на $3 \times n^2 (n \geq 1)$ інформаційних доменів з можливістю динамічного підвантаження (за аналогією з прогресивним Jpeg). Тобто, реалізувати оптимальний метод прогресивної розгортки зображення на $3 \times n^2 (n \geq 1)$ прошарків.

Суть методу складається у наступному. Розіб'ємо наше зображення розміром $H \times W$ на квадрати $P_{x,y}$, розміром $n \times n (n \geq 1)$. Для простоти будемо вважати, що $H = N \times n, W = M \times n.$ Кожному пікселю $p_{nx+i,ny+j} (i, j = 0,1,\ldots, n-1)$ даного квадрату відповідають три кольорові компоненти $r_{nx+i,ny+j}, g_{nx+i,ny+j}, b_{nx+i,ny+j}$ (червона, зелена та синя складові). Тобто, загальна інформація на квадраті $P_{x,y}$ описується даними

$r_{nx,ny}, g_{nx,ny}, b_{nx,ny}, r_{nx+1,ny}, g_{nx+1,ny}, b_{nx+1,ny}, \ldots, r_{(n+1)x-1,(n+1)y-1}, g_{(n+1)x-1,(n+1)y-1}, b_{(n+1)x-1,(n+1)y-1}$

.

Сформуємо домени вхідної інформації, наприклад, наступним чином

$$X_0 = \{r_{0,0}, r_{n,0}, \ldots, r_{nx,ny}, \ldots, r_{H-n-1,W-n-1}\},$$
$$X_1 = \{g_{0,0}, g_{n,0}, \ldots, g_{nx,ny}, \ldots, g_{H-n-1,W-n-1}\},$$
$$X_2 = \{b_{0,0}, b_{n,0}, \ldots, b_{nx,ny}, \ldots, b_{H-n-1,W-n-1}\},$$
$$X_3 = \{r_{1,0}, r_{n+1,0}, \ldots, r_{nx+1,ny}, \ldots, r_{H-n,W-n-1}\},$$
$$X_4 = \{g_{1,0}, g_{n+1,0}, \ldots, g_{nx+1,ny}, \ldots, g_{H-n,W-n-1}\},$$
$$X_5 = \{b_{1,0}, b_{n+1,0}, \ldots, b_{nx+1,ny}, \ldots, b_{H-n,W-n-1}\},$$
$$\ldots$$
$$X_{3\times n^2-2} = \{r_{n-1,n-1}, r_{2n-1,2n-1}, \ldots, r_{(n+1)x-1,(n+1)y-1}, \ldots, r_{H,W}\},$$
$$X_{3\times n^2-1} = \{g_{n-1,n-1}, g_{2n-1,2n-1}, \ldots, g_{(n+1)x-1,(n+1)y-1}, \ldots, g_{H,W}\},$$
$$X_{3\times n^2} = \{b_{n-1,n-1}, b_{2n-1,2n-1}, \ldots, b_{(n+1)x-1,(n+1)y-1}, \ldots, b_{H,W}\}.$$

Розглянемо задачу

$$\sum_{j=0}^{3\times n^2}\left\|X_j - \sum_{i=0}^{k}\alpha_{j,i}Y_i\right\|_2^2 \to \min$$

де мінімум береться по всім $\{Y_i\}_{i=0}^{k}, \alpha_{j,i}(i=0,1,...,k,\ j=0,1,...,3\times n^2)$.

Як видно, для розв'язку цієї задачі можна використати ітераційний метод головних компонент, алгоритм якого наведено вище.

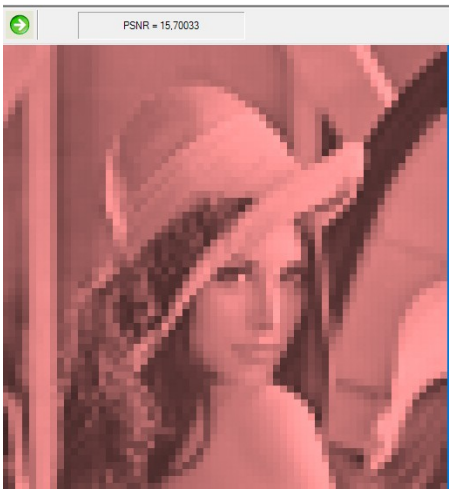
Таким чином, нехай ми отримали всі $\{Y_i\}_{i=0}^{k}, \alpha_{j,i}(i=0,1,...,k,\ j=0,1,...,3\times n^2)$.

Опишемо алгоритм прогресивного відновлення зображення.

Ясна річ, що множина $\{X_j\}_{j=0}^{3\times n^2}$ описує всі пікселі зображення, то, якщо

$$X_j^0 = \alpha_{j,0}Y_0\ ,$$

то $\{X_j^0\}_{j=0}^{3\times n^2}$ є першою ітерацією відновлення нашого зображення.



Зображення Lenaвідновлене за одною ітерацією.

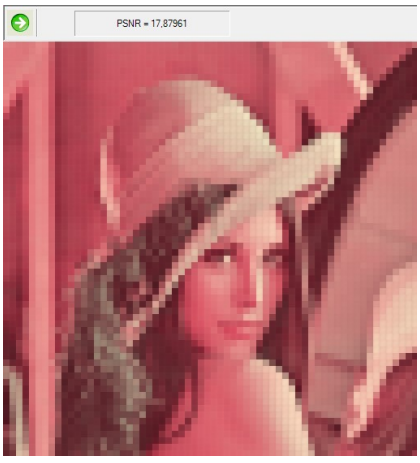
На другому кроці

$$X_j^1 = \alpha_{j,0}Y_0 + \alpha_{j,1}Y_1\ ,$$

тобто

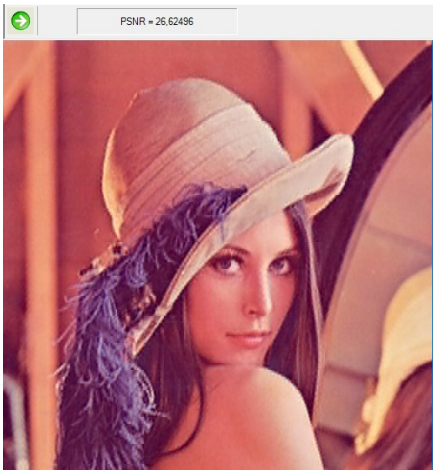
$$X_j^1 = X_j^0 + \alpha_{j,1}Y_1$$

і $\{X_j^1\}_{j=0}^{3\times n^2}$ буде другою ітерацією.

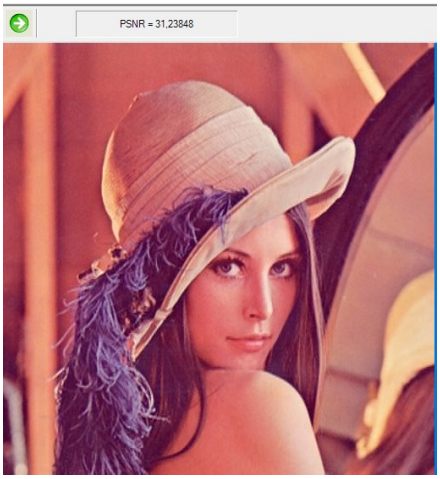


Зображення Lenaвідновлене за другою ітерацією

Таким чином маємо $X_j^k = \sum_{i=0}^{k} \alpha_{j,i} Y_i = X_j^{k-1} + \alpha_{j,k} Y_k$, а $\left\{ X_j^k \right\}_{j=0}^{3 \times n^2}$ буде k-ю ітерацією відновлення нашого зображення.



ЗображенняLena на десятій ітерації



ЗображенняLena на 21 із 192 ітерацій

Зазначимо, що на відміну від фільтрації на основі перетворення Фур'є, форма примітиву, по якому робиться розклад методом головних компонент, може бути будь якої форми за умовою щільної упаковки площини, очевидно, що це може бути прямокутник, шестикутник, який найбільш щільно запаковує площину, або, навіть, по така екзотична фігура, як фрактал-дракон.

Наведемо деякі приклади.

Нехай заданий будь двовимірний паркет, що покриває всю площину. Кожен елемент паркету складається з n пікселів. Перенумеруємо всі пікселі елемента паркету довільним, але фіксованим способом. Вибираючи з кожного елемента паркетупіксель з одним і тим же номером, отримаємо n компонент. Зауважимо, що результат оптимальної фільтрації не залежить відспособу нумерації пікселів на елементі паркету.

Наприклад, для паркету з елементами у вигляді шестикутних сотів з 76 пікселів можна використовувати наступну нумерацію

				1	2				
		3	4	5	6	7	8		
9	10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37	38
39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58
59	60	61	62	63	64	65	66	67	68
		69	70	71	72	73	74		
				75	76				

Застосуємо МГК до зображення Lenay градаціях сірого

Зображення Lena, зібране з одного домену з шестикутним паркетом

Зображення Lena, зібране із сьоми доменів з шестикутним паркетом

Випишемо нумерацію для паркету «Дракон» з 64 пікселів

		1	2			3	4					
			5	6			7	8				
9	10	11	12	13	14	15	16					
	17	18	19	20	21	22	23	24				
25	26			27	28	29	30			31	32	
	33	34			35	36	37	38			39	40
				41	42	43	44	45	46	47	48	
					49	50	51	52	53	54	55	56
				57	58			59	60			
					61	62			63	64		

Зображення Lena, зібране з одного домену з паркетом «Дракон»

Зображення Lena, зібране з чотирьох доменів з паркетом «Дракон»

Зображення Lena, зібране із сьоми доменів з паркетом «Дракон»

Вейвлети у комп’ютерній графіці

Сплески (wavelets) з'явилися на початку 1980-х років на стику теорії функцій, функціонального аналізу, обробки сигналів і зображень, квантової теорії поля. Розвиток теорії сплесків пов'язаний з іменами А. Гросмана, Ж. Морлета, Ж. Стремберга, Я. Мейєра, С. Малла, І. Добеші і багатьох ін.

Подібно аналізу Фур'є теорія сплесків знаходить застосування в дослідженні частотних характеристик функції fза допомогою сплеск-перетворення. При описі функцій зі скінченним носієм, сигналів, зі змінними у часі частотами за допомогою аналізу Фур'є виникають труднощі, які пояснюються властивостями тригонометричних функцій на осі *R*. Наприклад, перетворення Фур'є функції fв разі δ-

функції Дірака, що має в якості носія єдину точку, вироджується в функцію $\hat{\delta}(\omega) = 1$, поширену на всю числову вісь. При обробці сигналів зі змінними частотами аналіз Фур'є не дозволяє виділити частоти, що становлять сигнал в околі довільного моменту часу. Апарат сплесків дозволяє автоматично здійснювати локалізацію як за часом, так і в частотній області.

Наведемо один алгоритм фільтрації векторного поля, тобто набору векторів однакової розмірності.

Ці результати є основоположними для всіх подальших алгоритмів, пов'язаних з оптимальною фільтрацією фільтрами, подібними фільтрам Хаара і Уолша-Адамара однієї і двох змінних.

У чисельному і функціональному аналізі дискретні вейвлет-перетворення (ДВП) відносяться до вейвлет-перетворень, в яких вейвлети представлені дискретними сигналами (вибірками).

Вперше ДВП було придумано угорським математиком Альфредом Хааром. Для вхідного сигналу, представленого масивом 2^nчисел, вейвлет-перетворення Хаара просто групує елементи по 2 і утворює від них суми і різниці. Угруповання сум проводиться рекурсивно для наступного рівня розкладу. У підсумку виходить 2^n-1 різниця і 1 загальна сума.

Це просте ДВП ілюструє загальні корисні властивості вейвлетів. По-перше, перетворення можна виконати за n $\log_2 n$ операцій. По-друге, воно не тільки розкладає сигнал на деяку подобу частотних смуг (шляхом аналізу його в різних масштабах), але і представляє тимчасову область, тобто моменти виникнення тих чи інших частот в сигналі.

Наведемо визначення і позначення, необхідні нам надалі.

Множину векторів $X^k = \{x_i^k\}$ *(k = 0, .., n-1, i = 0,1, .., N-1)* однакової розмірності *N* будемо називатикомпонентами векторного поля або просто компонентами. Під фільтром $H^\nu = \{h_i^\nu\}$ *(ν = 0, .., n-1)* будемо розумітивектор з n координатами.

Систему векторів H^ν (*ν = 0, .., n-1*) будемо називати ортонормованою, якщо

$$H^\nu \perp H^\mu \Leftrightarrow \langle H^\nu, H^\mu \rangle = 0 \; ; \langle H^\nu, H^\nu \rangle = 1.$$

Тут <a, b> традиційний скалярний добуток векторів a і b.

Під фільтрацією компонентів X^k *(k = 0, .., n-1)* системою фільтрів H^k *(k = 0, .., n-1)* будемо розуміти обчисленнямножини (частотного домену) за правилом

$$Y^\nu = \sum_{k=0}^{n-1} h_\nu^k X^k \tag{3.6}$$

У координатної формі це співвідношення буде мати вигляд

$$y_i^\nu = \sum_{k=0}^{n-1} h_\nu^k x_i^k$$

Величину

$$X_m^\nu = \sum_{k=0}^{m-1} h_\nu^k Y^k \tag{3.7}$$

будемо називати відновленням компоненти X^ν по m частотним доменів. Якщо *m = n,* то отримуємо відновленнякомпоненти по повному набору частотних доменів.

Перше співвідношення називається декомпозицією, а останнє - реконструкцією даних.

У разі, якщо для даного масиву $F = \{f_i\}$ і для $n = 4$ компоненти визначені правилом

$$X^0 = \{x_i^0\} = \{f_{4i}\}, X^1 = \{x_i^1\} = \{f_{4i+1}\},$$

$$X^2 = \{x_i^2\} = \{f_{4i+2}\}, X^3 = \{x_i^3\} = \{f_{4i+3}\},$$

для фільтрів Уолша-Адамара

$$H^0 = \frac{1}{2}\{1,1,1,1\}, H^1 = \frac{1}{2}\{1,1,-1,-1\},$$

$$H^2 = \frac{1}{2}\{1,-1,1,-1\}, H^3 = \frac{1}{2}\{1,-1,-1,1\}.$$

процедура декомпозиції (1) буде мати вигляд

$$y_i^0 = \frac{1}{2}(f_{4i} + f_{4i+1} + f_{4i+2} + f_{4i+3}),$$

$$y_i^1 = \frac{1}{2}(f_{4i} + f_{4i+1} - f_{4i+2} - f_{4i+3}),$$

$$y_i^2 = \frac{1}{2}(f_{4i} - f_{4i+1} + f_{4i+2} - f_{4i+3}),$$

$$y_i^3 = \frac{1}{2}(f_{4i} - f_{4i+1} - f_{4i+2} + f_{4i+3}).$$

а формули реконструкції (2), наприклад, для $m = 2$

$$f_{4i} = \frac{1}{2}(y_i^0 + y_i^1), \qquad f_{4i+1} = \frac{1}{2}(y_i^0 + y_i^1),$$

$$f_{4i+2} = \frac{1}{2}(y_i^0 - y_i^1), f_{4i+3} = \frac{1}{2}(y_i^0 - y_i^1).$$

Повна реконструкція (тобто $m = n$) визначається співвідношеннями

$$f_{4i} = \frac{1}{2}(y_i^0 + y_i^1 + y_i^2 + y_i^3),$$

$$f_{4i+1} = \frac{1}{2}(y_i^0 + y_i^1 - y_i^2 - y_i^3),$$

$$f_{4i+2} = \frac{1}{2}(y_i^0 - y_i^1 + y_i^2 - y_i^3),$$

$$f_{4i+3} = \frac{1}{2}(y_i^0 - y_i^1 - y_i^2 + y_i^3).$$

Таким чином, завдання використання непересічних фільтрів розбивається на дві. Перша з яких – зведеннясигналу (в загальному випадку багатокомпонентного і багаторозмірного) до системи компонент однакової розмірності. Зокрема, для фільтрації зображень це 3-компонентний двовимірний сигнал, для відео - трикомпонентний тривимірний ітак інше. Друге завдання полягає в отриманні системи найкращих (в тому чи іншому сенсі) фільтрів для n компонент.

Фундаментальним поняттям теорії вейвлет є поняття кратномасштабного аналізу (КМА).

Кратномасштабний аналіз в $L^2(R^n)$ – це послідовність замкнутих підпросторів

$$\ldots \subset V^{-1} \subset V^0 \subset V^1 \subset$$

для яких виконуються умови

1. $\overline{\bigcup_{j\in Z} V^j} = L^2(R^n);$
2. $\bigcap_{j\in Z} V^j = \{0\};$

3. $f(x) \in V^j \Leftrightarrow f(2^{-j}\circ) \in V^0$;
4. Знайдеться така функція $\varphi \in V^0$ (масштабуюча функція), що множина її зрушень $\varphi(x-n)$створює ортонормований базис простору V^0.

Дана послідовність просторів гарантує, що для довільної інтегрованої функції f і наперед заданого числа ε>0 існує простір V^k і елемент $f_k \in V^k$, що $\|f - f_k\| < \varepsilon$. Зазначимо, що у евклідовому просторі найкращим наближенням функції є ортогональна проекція функції на цей простір, тому у якості f_kбудемо брати ортогональну проекцію f на цей простір. Окрім того, з властивостей КМА витікає, що для будь якого $j \in Z$ множина функцій $\varphi_k^j(x) = 2^{j/2}\varphi(2^j x - k), k = 0, \pm 1, \ldots$ створюють ортонормований базис. Множник $2^{j/2}$ є результатом того, що заміна *x* на *2x* зменшує норму у $2^{1/2}$ раз.

Так як $\mathrm{V}^0 \subset \mathrm{V}^1$, то функція φ є лінійною комбінацією функцій $\{\varphi_n^1\}$. Таким чином, знайдеться множина коефіцієнтів $\{h_n\}$така, що

$$\varphi(x) = \sqrt{2}\sum_{n\in Z} h_n \varphi(2x - n).$$

З рівняння Парсеваля маємо

$$\sum_{n\in Z} h_n^2 = 1.$$

Функція $\varphi(x)$ має назву масштабуючої (скелінг) функції, а функція

$$\psi(x) = \sqrt{2}\sum_{n\in Z} (-1)^{1-n} h_{1-n}\varphi(2x - n)$$

називається вейвлетом або сплеском.

Зважаючи, що $\mathrm{V}^{-1} \subset \mathrm{V}^0$, можна записати $\mathrm{V}^{-1} \oplus W^{-1} = \mathrm{V}^0$. В якості сплеска, який визначає базиси зрушення у просторах W^j возьмемо функцію ψ і покладемо

$$\psi_n^j(x) = 2^{j/2}\psi(2^j x - n).$$

В такому разі

$$\sum_{n\in Z} c_n^{-1}\varphi_n^{-1}(x) + \sum_{n\in Z} d_n^{-1}\psi_n^{-1}(x) = \sum_{n\in Z} c_n^0\varphi_n^0(x).$$

Звідси, застосовуючи операцію скалярного добутку, отримуємо

$$c_k^{-1} = \sum_{n\in Z} c_n^0\langle\varphi_n^0(x), \varphi_n^{-1}(x)\rangle = \sum_{n\in Z} c_n^0 h_{n-2k}$$

і

$$d_k^{-1} = \sum_{n\in Z} c_n^0 g_{n-2k}$$

де $g_n = (-1)^{1-n} h_{1-n}$.

Ясно, що в загальному випадку

$$V^j \oplus W^j = V^{j+1}$$

та

$$\sum_{n\in Z} c_n^j \varphi_n^j(x) + \sum_{n\in Z} d_n^j \psi_n^j(x) = \sum_{n\in Z} c_n^{j+1} \varphi_n^{j+1}(x).$$

і загальні формули декомпозиції

$$c_k^j = \sum_{n\in Z} c_n^{j+1} h_{n-2k}$$

і

$$d_k^j = \sum_{n\in Z} c_n^{j+1} g_{n-2k}.$$

Узагальнюючи вище сказане, маємо розклад V^0 в ортогональну суму

$$V^{-k} \oplus W^{-k} \oplus \ldots \oplus W^{-2} \oplus W^{-1} = V^0$$

і формула реконструкції на кожному кроці буде мати вигляд

$$c_k^{j+1} = \sum_{n\in Z} c_n^j h_{k-2n} + \sum_{n\in Z} d_n^j g_{k-2n}$$

Наведений алгоритм називається каскадним.

Ортогональні вейвлети

Нехай φє ортогональною функцією на решітці з кроком 2, тобто

$$\langle \varphi, \varphi(\cdot - 2k) \rangle = \delta_{0,k}$$

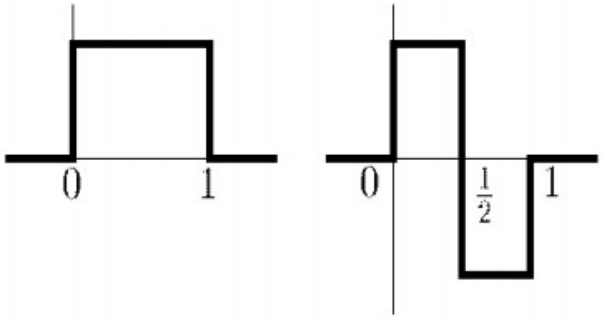
або, що теж саме,

$$\sum_i h_i h_{i-2k} = \delta_{0,k}.$$

Відомий факт, що якщо ортогональна функція має компактний носій, то цей носій парний. Для найпростішого випадку він дорівнює двом

$$h_0 = \frac{1}{\sqrt{2}}, h_1 = \frac{1}{\sqrt{2}}.$$

Відповідна базова функція називається функцією Хаара



Знайдемо тензорний добуток різних варіантів цих функційφ(x)φ(y),φ(x)ψ(y),ψ(x)φ(y)та ψ(x)ψ(y), в результаті отримаємо четвірку фільтрів (фільтрів Хаара)

$$H^{++}=\frac{1}{2}\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}, H^{\pm}=\frac{1}{2}\begin{pmatrix}1 & 1\\ -1 & -1\end{pmatrix}, H^{+-}=\frac{1}{2}\begin{pmatrix}1 & -1\\ 1 & -1\end{pmatrix}, H^{--}=\frac{1}{2}\begin{pmatrix}1 & -1\\ -1 & 1\end{pmatrix}.$$

Результатом фільтрації четвірки пікселів $\{p_{n,m}\}_{n=2i,m=2j}^{2i+1,2j+1}$ буде

$$c_{i,j}^{++}=\frac{1}{2}(p_{2i,2j}+p_{2i+1,2j}+p_{2i,2j+1}+p_{2i+1,2j+1}),$$
$$c_{i,j}^{\pm}=\frac{1}{2}(p_{2i,2j}+p_{2i+1,2j}-p_{2i,2j+1}-p_{2i+1,2j+1}),$$
$$c_{i,j}^{+-}=\frac{1}{2}(p_{2i,2j}-p_{2i+1,2j}+p_{2i,2j+1}-p_{2i+1,2j+1}),$$
$$c_{i,j}^{--}=\frac{1}{2}(p_{2i,2j}-p_{2i+1,2j}-p_{2i,2j+1}+p_{2i+1,2j+1}).$$

Отримані множини називаються частотними доменами, $C^{++}=\{c_{i,j}^{++}\}$ - низько-частотний домен, $C^{+-}=\{c_{i,j}^{+-}\}$ та $C^{\pm}=\{c_{i,j}^{\pm}\}$ середньо-частотні домени та $C^{--}=\{c_{i,j}^{--}\}$ – високо-частотний домен.

З них найбільш інформативним є низько-частотний домен, а найменш – високо-частотний.

Застосуємо дане перетворення (проведення декомпозоції) до тестового зображення Lenai упорядкуємо наступним чином

$$\begin{bmatrix}C^{++} & C^{+-}\\ C^{\pm} & C^{--}\end{bmatrix}$$

Після нормалізації результат фільтрації буде виглядати наступним чином

Зважаючи на той факт, що фільтри Хаара створюють повну систему, можна виписати реконструкцію даних

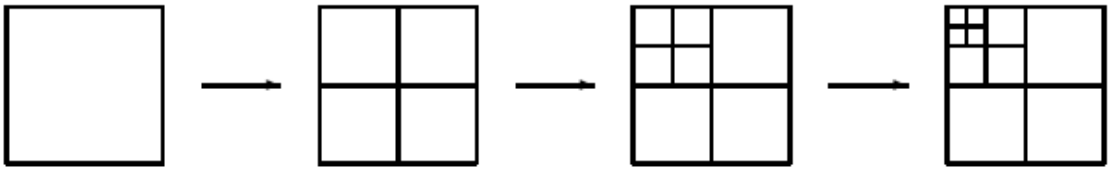
$$p_{2i,2j} = \frac{1}{2}(c_{i,j}^{++} + c_{i,j}^{\pm} + c_{i,j}^{+-} + c_{i,j}^{--}),$$

$$p_{2i,2j+1} = \frac{1}{2}(c_{i,j}^{++} - c_{i,j}^{\pm} + c_{i,j}^{+-} - c_{i,j}^{--}),$$

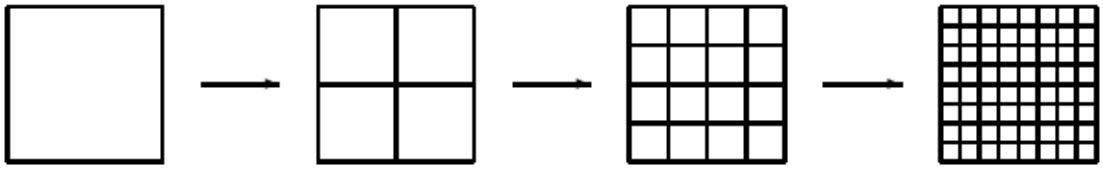
$$p_{2i+1,2j} = \frac{1}{2}(c_{i,j}^{++} + c_{i,j}^{\pm} - c_{i,j}^{+-} - c_{i,j}^{--}),$$

$$p_{2i+1,2j+1} = \frac{1}{2}(c_{i,j}^{++} - c_{i,j}^{\pm} - c_{i,j}^{+-} + c_{i,j}^{--}).$$

Послідовне застосування декомпозиції до низько-частотного домену називається каскадною схемою



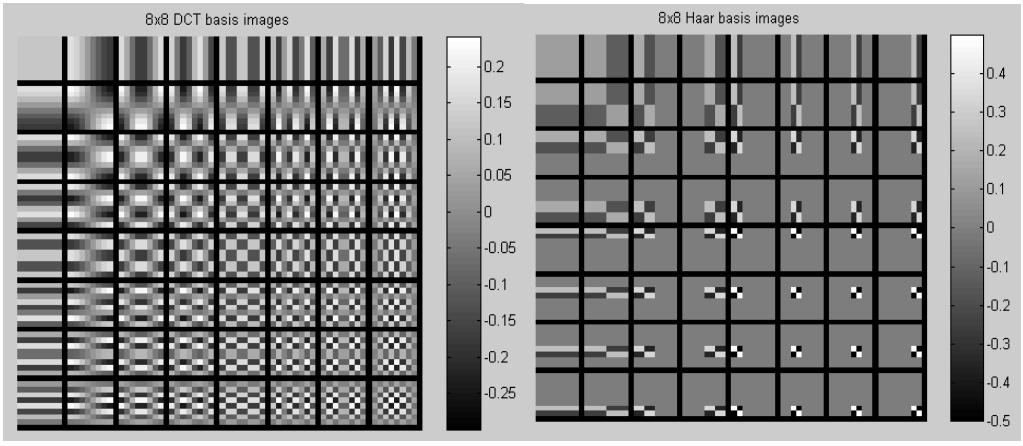
А застосування декомпозиції до усієї множини доменів – схемою повного пакету



Наприклад, схему повного пакету для двох кроків можна проілюструвати наступним чином

Для трьох кроків

Якість фільтрації зображення відзначається, поперед всього, тим, наскільки більш інформативний низько-частотний домен, зрівнюючи його з середньо і високо-частотними доменами.
Серед усіх ортогональних вейвлетів базис Хаара має найгірші показники.
Можна порівняти розклад константи методом DCT (дискретне косинус-перетворення)та за Хааром



Для отримання вейвлетів більш складної конструкції умови ортогональності не досить. Інгрід Добеші запропонувала додати умови дорівнювання нулю моментів до відповідного порядку

$$\int x^k \psi(x) dx = 0, k = 0,1, \dots, n-1,$$

Що еквівалентно умові

$$\sum_i i^k h_i = 0, i = 0,1, \dots, n-1.$$

У разі, коли довжина носія дорівнює 4, умови ортогональності будуть мати вигляд

$$h_0^2 + h_1^2 + h_2^2 + h_3^2 = 1,$$
$$h_0 h_2 + h_1 h_3 = 0.$$

Додаємо дві умови дорівнювання моментів нулю

$$h_0 - h_1 + h_2 - h_3 = 0,$$

$$-h_1 + 2h_2 - 3h_3 = 0.$$

Розв'язуючи отриману систему рівнянь, отримаємо коефіцієнти ортогонального фільтру Добеші порядку 4:

$$h_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, h_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}, h_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, h_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}.$$

У матричному вигляді процес декомпозиції можна записати так

$$\begin{bmatrix} c_0^{-1} \\ c_1^{-1} \\ c_2^{-1} \\ c_3^{-1} \\ d_0^{-1} \\ d_1^{-1} \\ d_2^{-1} \\ d_3^{-1} \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ h_2 & h_3 & 0 & 0 & 0 & 0 & h_0 & h_1 \\ h_3 & -h_2 & h_1 & -h_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_3 & -h_2 & h_1 & -h_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_3 & -h_2 & h_1 & -h_0 \\ h_1 & -h_0 & 0 & 0 & 0 & 0 & h_3 & -h_2 \end{bmatrix} \begin{bmatrix} c_0^0 \\ c_1^0 \\ c_2^0 \\ c_3^0 \\ c_4^0 \\ c_5^0 \\ c_6^0 \\ c_7^0 \end{bmatrix}$$

А реконструкція буде виглядати наступним чином

$$\begin{bmatrix} c_0^0 \\ c_1^0 \\ c_2^0 \\ c_3^0 \\ c_4^0 \\ c_5^0 \\ c_6^0 \\ c_7^0 \end{bmatrix} = \begin{bmatrix} h_0 & 0 & 0 & h_2 & h_3 & 0 & 0 & h_1 \\ h_1 & 0 & 0 & h_3 & -h_2 & 0 & 0 & -h_0 \\ h_2 & h_0 & 0 & 0 & h_1 & h_3 & 0 & 0 \\ h_3 & h_1 & 0 & 0 & -h_0 & -h_2 & 0 & 0 \\ 0 & h_2 & h_0 & 0 & 0 & h_1 & h_3 & 0 \\ 0 & h_3 & h_1 & 0 & 0 & -h_0 & -h_2 & 0 \\ 0 & 0 & h_2 & h_0 & 0 & 0 & h_1 & h_3 \\ 0 & 0 & h_3 & h_1 & 0 & 0 & -h_0 & -h_2 \end{bmatrix} \begin{bmatrix} c_0^{-1} \\ c_1^{-1} \\ c_2^{-1} \\ c_3^{-1} \\ d_0^{-1} \\ d_1^{-1} \\ d_2^{-1} \\ d_3^{-1} \end{bmatrix}$$

Для n=6 відповідний фільтр має вигляд

$$h_0 = \frac{1}{16\sqrt{2}}\left(1+\sqrt{10}+\sqrt{5+2\sqrt{10}}\right),$$

$$h_1 = \frac{1}{16\sqrt{2}}\left(5+\sqrt{10}+3\sqrt{5+2\sqrt{10}}\right),$$

$$h_2 = \frac{1}{16\sqrt{2}}\left(10-2\sqrt{10}+2\sqrt{5+2\sqrt{10}}\right),$$

$$h_3 = \frac{1}{16\sqrt{2}}\left(10-2\sqrt{10}-\sqrt{5+2\sqrt{10}}\right),$$

$$h_4 = \frac{1}{16\sqrt{2}}\left(5+\sqrt{10}-3\sqrt{5+2\sqrt{10}}\right),$$

$$h_5 = \frac{1}{16\sqrt{2}}\left(1+\sqrt{10}-\sqrt{5+2\sqrt{10}}\right).$$

Використання вейвлет Добеші до фільтрації зображення зводиться до послідовного застосування фільтрів спочатку до кожного рядка зображення, тим самим проводячі розподіл на низько та високочастотний домени, а потим до кожного отриманого домену вже по рядка. Така послідовність обробки називається схемою Малла (Mallat).

Ортогональні фільтри Добеші порядку 4 використовуються у методі стиску DJVU.
Але найрозповсюдженими вейвлет фільтрами є біортогональні, які використовуються у широкому спектрі разних задач, так чи інакше пов'язаних з частотним аналізом сигналів, що стосуються нашої теми, то треба зазначити, що метод JPEG2000 заснований на біортогональних вейвлетах.
Біортогональне перетворення є зворотним, але не обов'язково ортогональним, що дає можливість отримати додаткові переваги, як то, зробити вейвлет-фільтри симетричними, на відміну від ортогональних, які зробити симетричними неможливо.
Пара масштабуючих функцій

$$\varphi^{(0)}(x) = \sqrt{2}\sum_{n\in Z} h_n^{(0)}\varphi^{(0)}(2x-n),$$

$$\varphi^{(1)}(x) = \sqrt{2}\sum_{n\in Z} h_n^{(1)}\varphi^{(1)}(2x-n),$$

Разом з відповідною парою вейвлетів

$$\psi^{(0)}(x) = \sqrt{2}\sum_{n\in Z} g_n^{(0)}\varphi^{(0)}(2x-n),$$

$$\psi^{(1)}(x) = \sqrt{2}\sum_{n\in Z} g_n^{(1)}\varphi^{(1)}(2x-n),$$

де $g_n^{(m)} = (-1)^{1-n}h_{1-n}^{(m)}, m = 0,1,$ називаються біортогональними, якщо

$$\langle\varphi^{(m)}, \varphi^{(n)}(\cdot - 2k)\rangle = \delta_{0,k}, (m,n = 0,1; n \neq m; k = 0, \pm 1, \dots),$$

тоді має місце реконструкція сигналу

$$c_k^{j+1} = \sum_{n\in Z} c_n^j h_{k-2n}^{(1)} + \sum_{n\in Z} d_n^j g_{k-2n}^{(0)}$$

де формули декомпозиції такі

$$c_k^j = \sum_{n\in Z} c_n^{j+1} h_{k-2n}^{(0)}, \quad d_k^j = \sum_{n\in Z} c_n^{j+1} g_{k-2n}^{(1)}.$$

Як у ортогональному випадку простішими є фільтри Хаара, так у біортогональному, найпростіший варіант дають фільтри 1-3, які мають наступний вигляд

$$h_0^{(0)} = 1,$$

$$h_0^{(1)} = 1,\ h_{\pm 1}^{(1)} = \frac{1}{2}.$$

У такому разі декомпозиція сигналу виглядає наступним чином

$$c_k^j = c_{2k}^{j+1}, \quad d_k^j = -\frac{1}{2}c_{2k}^{j+1} + c_{2k+1}^{j+1} - \frac{1}{2}c_{2k+2}^{j+1},$$

а реконструкція

$$c_{2k}^{j+1} = c_k^j, \quad c_{2k+1}^{j+1} = d_k^j + \frac{1}{2}c_k^j + \frac{1}{2}c_{k+1}^j.$$

Наведемо розрахунок класичної біортогональної пари 3-5 Коена-Добеші- Фово. Нехай $n=1$ і поставимо умову точності на константі, тоді

$$h_0^{(0)} = \frac{1}{\sqrt{2}}, h_{\pm 1}^{(0)} = \frac{1}{2\sqrt{2}},$$

а для $n=2$

$$h_0^{(1)} + 2h_{\pm 2}^{(1)} = \frac{1}{\sqrt{2}}, h_{\pm 1}^{(1)} = \frac{1}{2\sqrt{2}},$$

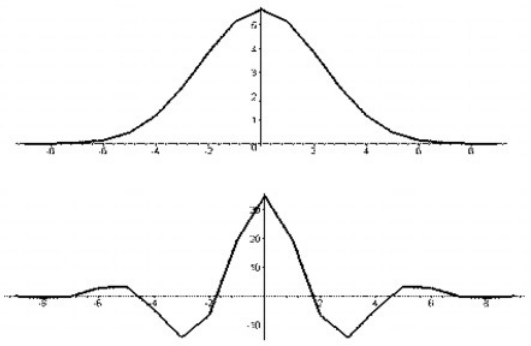
і умова біортогональності має вигляд

$$\frac{1}{8} + h_{\pm 2}^{(1)}\frac{\sqrt{2}}{2} = 0 \quad \text{та} \quad \frac{3}{4} - \sqrt{2}h_{\pm 2}^{(1)} = 1.$$

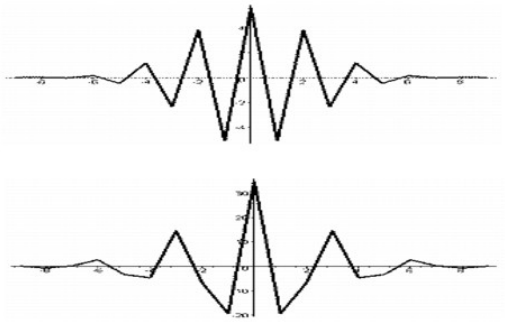
Звідси зразу отримуємо біортогональну пару

$$h_0^{(0)} = \frac{\sqrt{2}}{2}, h_{\pm 1}^{(0)} = \frac{\sqrt{2}}{4},$$

$$h_0^{(1)} = \frac{3\sqrt{2}}{4}, h_{\pm 1}^{(1)} = \frac{\sqrt{2}}{4}, h_{\pm 2}^{(1)} = -\frac{\sqrt{2}}{8}.$$



Базисні функції біортогональної пари 3-5.



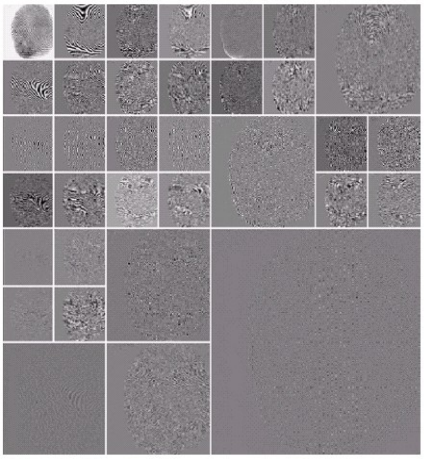
Вейвлет- функції біортогональної пари 3-5.

Наведемо найпопулярніші біортогональні фільтри 5-7

$$h_0^{(0)} = \frac{3\sqrt{2}}{5}, h_{\pm 1}^{(0)} = \frac{\sqrt{2}}{4}, h_{\pm 2}^{(0)} = -\frac{\sqrt{2}}{20}$$

$$h_0^{(1)} = \frac{17\sqrt{2}}{28}, h_{\pm 1}^{(1)} = \frac{73\sqrt{2}}{280}, h_{\pm 2}^{(1)} = -\frac{3\sqrt{2}}{56}, h_{\pm 3}^{(1)} = -\frac{3\sqrt{2}}{280}.$$

Зазначимо, що у разі відомої природи шуму, вейвлет- аналіз є ефективним інструментом його видалення. В цьому разі домени частот, які відповідають шуму, можна просто обернути в ноль, в наслідок чого зображення буде трохи згладжене, але без відповідних завад, для розв'язку цієї задачі зовсім не обов'язково робити розклад у повний пакет.



Зображення відбитку пальця з відповідними частотами

Цілочисельне вейвлет-перетворення

Існує цілий ряд задач, коли частотне перетворення сигналу (у нашому випадку, зображення) не повинне вносити помилку округлення. І це є принциповим. У такому разі, все, що було викладене раніше, не підходить.

Використання дискретного тригонометричного перетворення Фур'є, так само, як і ортогонального (або біортогонального) вейвлет перетворення, призводить до появи помилок округлення. Тому для цілочисельного частотного аналізу зображення було запропоновано використовувати цілочисельне вейвлет-перетворення. Традиційно для цієї мети використовується сепарабельном схема або схема Маллата. Зокрема, в стандарті JPEG2000 застосовується наступна конструкція.

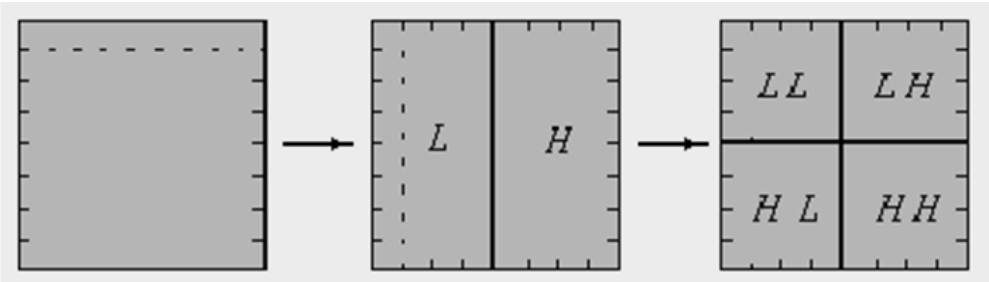
Нехай c_i^0 вихідні дані, тоді прямий хід виглядає наступним чином

$$c_{2i}^1 = c_{2i}^0 - \left\lfloor \frac{c_{2i-1}^0 + c_{2i+1}^0 + 2}{4} \right\rfloor, c_{2i+1}^1 = c_{2i+1}^0 + \left\lfloor \frac{c_{2i}^1 + c_{2i+2}^1}{2} \right\rfloor,$$

а зворотний –

$$c_{2i+1}^0 = c_{2i+1}^1 - \left\lfloor \frac{c_{2i}^1 + c_{2i+2}^1}{2} \right\rfloor, c_{2i}^0 = c_{2i}^1 - \left\lfloor \frac{c_{2i-1}^0 + c_{2i+1}^0 + 2}{4} \right\rfloor.$$

Множина коефіцієнтів з парними індексами утворює низькочастотний домен (L), а з непарними - високочастотний (H). Послідовне застосування того ж алгоритму дозволяє розділити вихідні дані на чотири частотних домену. З яких домен LL найбільш чутливий до помилки, а домен HH найменш чутливий.



Як правило, частотні перетворення використовуються не для компонентів рівноправного простору кольорів (R,G,B), а для нерівноправних, таких як YCrCb, YIQ та інше.

Сам процес перетворення кольорових компонентів з рівноправних (тип даних byte) до нерівноправних (тип даних double або float), вже вносить помилку округлення. У разі, коли точність перетворення є принциповою, треба зробити це у цілих числах. Таке перетворення можна провести наступним чином

$$Y = [(R + 2\,G + B) / 4];\ U = (R - G);\ V = (B - G);$$

і зворотний хід

$$G = (Y - [(U + V) / 4]);\ R = U + G;\ B = V + G;$$

де [.] тут і далі ціла частина числа.

Отримані цілочисельні колірні компоненти YUV не настільки ефективні, як YCrCb або YIQ, зате позбавлені такої проблеми, як помилки округлення.

Зауважимо, що сепарабельна схема з самого початку орієнтована на одномірні дані, тому використання двовимірної вейвлет-схеми для обробки зображень є кращим. Для фільтрації даних $p_{i,j}$ була запропонована несепарабельна (що непредставимо у вигляді тензорного добутку одновимірних) вейвлет-схема з тією ж складністю, тобто з рівним числом арифметичних операцій.

На першому кроці проводиться децимація даних з парними індексами

$$s^0_{i,j} = p_{2i,2j}\ .$$

На другому кроці по децимованим значенням будуємо лінійні відновлення в точках з одним непарних індексом і обчислюємо відповідні частотні коефіцієнти

$$s^1_{i,j} = p_{2i+1,2j} - \left[\frac{p_{2i,2j} + p_{2i+2,2j}}{2}\right], s^2_{i,j} = p_{2i,2j+1} - \left[\frac{p_{2i,2j} + p_{2i,2j+2}}{2}\right],$$

і, нарешті, на третьому кроці отримуємо частотні коефіцієнти з непарними індексами

$$s^3_{i,j} = p_{2i+1,2j+1} - \left[\frac{s^0_{i,j} + s^0_{i,j+1} + s^0_{i+1,j} + s^0_{i+1,j+1}}{4}\right] - \left[\frac{s^1_{i,j} + s^1_{i,j+1} + s^2_{i,j} + s^2_{i+1,j}}{4}\right].$$

За аналогією зі схемою Маллата можна записати

$$HH = \left|s^3_{i,j}\right|, HL = \left|s^2_{i,j}\right|, LH = \left|s^1_{i,j}\right|, LL = \left|s^0_{i,j}\right|.$$

Контрольні питання

1. В чому сенс перетворення Фур'є для неперервних та дискретних сигналів?
2. Дискретне косинус-перетворення, дискретне синус-перетворення, дискретне перетворення Хартлі, їх плюси і мінуси для обробки зображень?
3. Чи існують симетричні ортогональні локальні вейвлети?
4. В чому сенс біортогональних вейвлетів?
5. Методи прогресивної передачі зображень, сфери їх використання?

Рекомендован література

1. *Чуи К.* Введение в вейвлеты. – М.: Мир, 2001.
2. *Babenko V.F., Ligun A., Shumeiko A.* Non-separable wavelets and their application. – Wavelets and Splines. International conference, St,Peterburg, (2003), –p.10–11.
3. *Добеши И.* Десять лекций по вейвлетам, Ижевск, НИЦ "Регулярная и хаотическая динамика",2001.– 464 с.

4. *Петухов А.П.* Введение в теорию базисов всплесков. – СПб, Изд. СПбГТУ, 1999 – 132 с.
5. Новиков И.Я., Протасов В.Ю., Скопина М.А. Теория всплесков.- М., Физматлит, 2005. – 612 с.
6. Бабенко В.Ф. Об одном методе построения несепарабельных всплесков / В.Ф.Бабенко, А.А.Лигун, А.А.Шумейко // Математичне моделювання .— 2007 .— №2(17) .— С.35-40.
7. Бабенко В.Ф. Быстрый алгоритм биортогонального дискретного всплеск-преобразования / В.Ф.Бабенко, А.А.Лигун, А.А.Шумейко // Математичне моделювання .— 2007 .— №1(16) .— С.18-21.
8. Шумейко А. Быстрое дискретное тригонометрическое преобразование со свободным фазовым сдвигом / А.Шумейко, В.Смородский // Системні технології . — 2017 .— №4(111) .— С.46-55.
9. Hartley,R."A more symmetrical Fourier analysis applied to transmission problems," *Proc. IRE* **30**, 144–150 (1942).
10. Брейсуэлл Р. Преобразование Хартли.М.:Мир, 1990.- 175 с.
11. Лигун А.О. Комп'ютерна графіка (обробка та стиск зображень):навч.посіб./ А.О.Лигун, О.О.Шумейко.-Д.:Біла К.О., 2010.- 114 с.
12. Shumeiko A. Discrete trigonometric transform and its usage in digital image processing / A.Shumeiko, V.Smorodskyi // Econtechmod. An Intarnational Quarterly Journal .— 2017 .— №4(6) .— С.21-26.
13. Гонсалес Р., Вудс Р. Цифровая обработка изображений. — М.: Техносфера, 2006. — 1072 с.
14. Методы компьютерной обработки изображений/ Под ред. ВА.Сойфера.- М.:Физматлит, 2003.-784 с.
15. Шумейко А.А. Интеллектуальный анализ данных (Введение в Data Mining) / А.А.Шумейко, С.Л.Сотник .— Днепропетровск: Белая Е.А., 2012 .— 212 с.
16. Mallat S.A. Theory for Multiresolution Signal Decomposition: The Wavelet Representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 11, No. 7, July 1989, p. 674-693.
17. Шумейко А.А. Применение несепарабельной вейвлет-схемы для стеганографической защиты данных / А.А.Шумейко, А.И.Пасько// Вісник східноукраїнського національного університету імені Володимира Даля.— 2008. — №8(126) .— С.50-55.
18. Шумейко А.А. Использование вейвлет для внедрения цифровых водяных знаков / А.А.Шумейко, А.И.Пасько// Інформаційна безпека.— 2011.— №1(5).— С.22-27.

ТЕМА 4. ВЕКТОРИЗАЦІЯ ЕЛЕМЕНТІВ РАСТРОВОГО ЗОБРАЖЕННЯ

У даному розділі розглянемо підходи до вилучення із растрового представлення прошарку ліній, замінюючи їх на векторний опис. Підходи до вирішення цієї проблеми ґрунтуються на методах витягання ліній із зображення і подальшим автоматичним вибором аналітичного опису між прямою, ламаною і сплайном Без'є. Спектр задач, в яких використовується вектризація досить широкий, це аналіз дистанційного зондування поверхні Землі з метою виявлення елементів штучного походження – аеродроми, дороги, аналіз сільськогосподарських угідь та ін., розпізнавання на растрових зображеннях різного роду об'єктів, які представляють інтерес, наприклад, наявність на фотографіях зброї і так інше.

Класично для виявлення контурів використовується згортка матриці яскравості $Y = \left\| y_{i,j} \right\|_{i=1,j=1}^{W,H}$ с оператором Собела:

$$G^S{}_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \otimes Y \ , \ G^S{}_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \otimes Y \qquad (4.1)$$

або з оператором Превіта:

$$G^P{}_x = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \otimes Y \ , \ G^P{}_y = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix} \otimes Y \ . \qquad (4.2)$$

Позначимо через $G = \left\| g_{i,j} \right\|_{i,j}^{W,H}$ ($W, H -$ розміри матриці) матрицю для вибраного оператора, елементи якої мають вид:

$$g_{i,j} = \| grad(i,j) \| = \sqrt{(g_{i,j}^x)^2 + (g_{i,j}^y)^2} \ . \qquad (4.3)$$

Для того, щоб знайти особливо контрастні місця, які належать символам, до матриці G застосовується бінаризація по порогу. Таким чином, знаходимо бінарну матрицю $BIN = \left\| bin_{i,j} \right\|_{i=1,j=1}^{W,H}$, у якої

$$bin_{i,j} = \begin{cases} 1, & якщо \quad g_{i,j} > \varepsilon, \\ 0, & у \ протилежному \ випадку. \end{cases} \qquad (4.4)$$

де ε – поріг бінаризації.

Кожен елемент матриці BIN додатково аналізується на предмет належності до регіону інтересу (ROI-Region Of Interest)– наприклад, видаляються ізольовані точки, які, швидше за все, відносяться до шумів.

Однією з головних проблем даного підходу є вибір порогу бінаризації (один з яких розглянутий у розділі, присвяченому просторовій фільтрації зображень). Особливо ця проблема стає актуальною, якщо на зображенні присутні елементи з різним рівнем освітлення.

Аналіз зв'язних компонент

Даний метод полягає в сегментуванні бінарного зображення на зв'язані регіони і їх подальший аналіз. Зазвичай бінарним зображенням є матриця BIN, отримана внаслідок побудови границь об'єктів.

Будемо називати точки (x_1, y_1) та (x_2, y_2) сусідніми, якщо вони задовольняють умові зв'язності, а саме, для 4-х зв'язної області умова зв'язності має вигляд:

$$x_1 - x_2 = 0, |y_1 - y_2| = 1$$

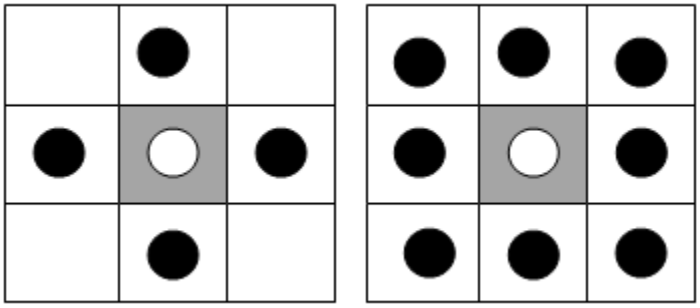
або

$$|x_1 - x_2| = 1, y_1 - y_2 = 0$$

Для 8-ми зв’язної:

$$|x_1 - x_2| \le 1, |y_1 - y_2| \le 1. \quad (4.5)$$

На рис. 4.1 наочно проілюстровано поняття зв'язності.



А Б

Рис. 4.1 А - 4-х зв'язана область; Б - 8-ми зв'язана область

Множину S будемо назвати зв’язною, якщо $\forall (x_1, y_1) \in S$ і

$$(x_n, y_n) \in S \; \exists \{(x_1, y_1), ..., (x_i, y_i), (x_{i+1}, y_{i+1}), ...(x_n, y_n)\}, (x_i, y_i) \in S,$$

де $(x_i, y_i), (x_{i+1}, y_{i+1})$ – сусідні точки.

Існують різні алгоритми побудови зв'язних множин, наприклад, такі як: алгоритм заповнення з «затравкою» (розглянутий у розділі, присвяченому сегментації зображень), порядковий лінійний алгоритм та ін.

Кінцевим кроком роботи алгоритму є аналіз різних характеристик, якими володіють отримані зв'язані множини: відсікання шуму, відсікання таких графічних елементів, як внутрішні зображення і т.д. Очевидно, що в даному підході багато залежить від того бінарного зображення, яке передається для побудови зв'язних компонент.

Вилучення ліній із зображення

Контуром будемо називати області зображення з високою концентрацією інформації, що мало залежить від кольору і яскравості. Виявлення контурів дозволяє перейти з двовимірного простору зображення до простору контурів, що дозволяє знизити обчислювальну та алгоритмічну складність аналізу зображення, підвищити якість відновленого зображення при менших розмірах, а також провести більш детальний аналіз об’єкту, описаного контуром.

Традиційно для виявлення контурів використовуються підходи, які спираються на інформацію про компоненти градієнта та на збіги із шаблоном. Обидві методики оцінюють локальне значення градієнта і його проекції. Відмінність в тому, що для компонента градієнта зазвичай використовуються фільтри, які дозволяють відловлювати перепади в двох ортогональних напрямках, наприклад, фільтри Собела (1.1), Превіта (1.2), а також фільтр Робертса:

$$G^R{}_x = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \otimes Y, \quad G^R{}_y = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes Y,$$

де $Y = \left| y_{i,j} \right|_{i=1,j=1}^{W,H}$ (матриця освітленості).

Для підходу збігу із шаблоном характерне використання такого числа фільтрів, які проявляють потрібні критерії і дають необхідну похибку збігу.

Контрастні перепади на зображенні знаходяться в точках максимального значення градієнта $\|grad(i,j)\|$ (див. (4.1)). Зазвичай для відділення контрастних точок використовується деяке порогове значення ε

$$\|grad(i,j)\| > \varepsilon .$$

Вибір порога – важлива частина визначення перепадів, від якої залежить товщина і неперервність контурів. На даний момент серед подібних методів виділяють фільтр Кенні.

Основні етапи алгоритму Кенні:

1. Згладжування. Фільтр Кенні чутливий до шумів на зображенні, тому первісна матриця згладжується за допомогою фільтра Гаусу:

$$Gauss = \begin{pmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{pmatrix} \otimes Y .$$

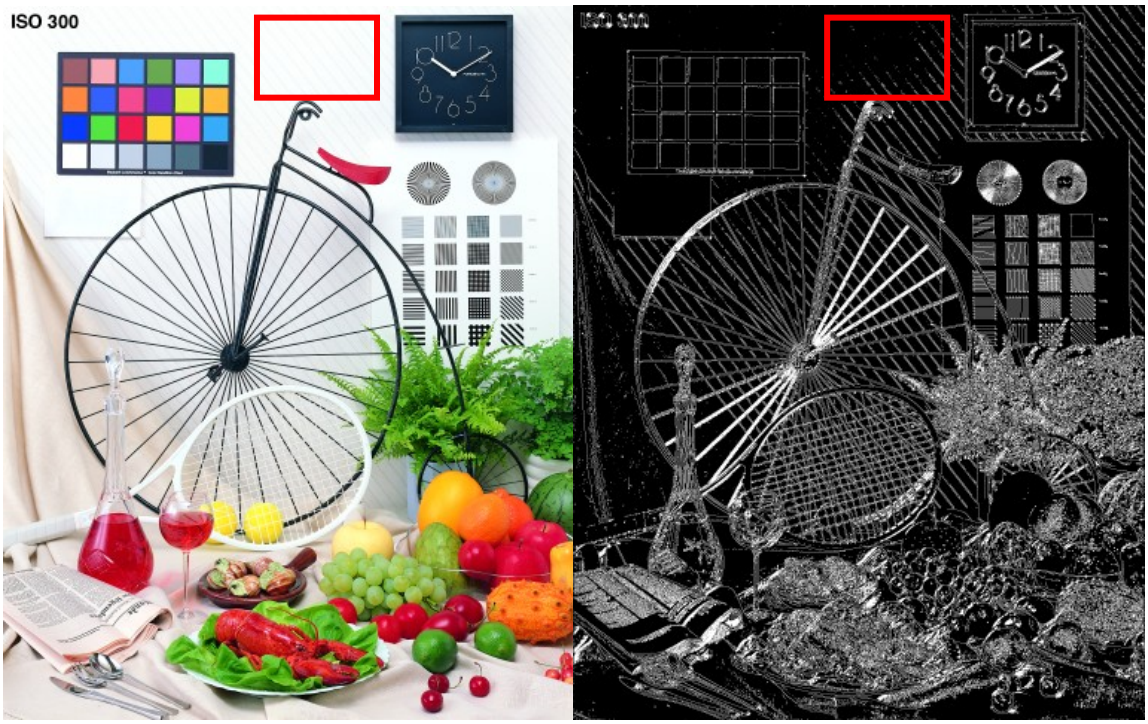
2. Пошук градієнтів. Знаходимо градієнти, згідно вибраним фільтрам (Собела, Первіта, Робертса). Для того щоб відстежити лінії з нахилом, фільтри розширюються такими діагональними операторами, щоб виявити лінії під відповідним кутом: $0^o, 45^o, 90^o, 135^o$.

3. Заглушення немаксимумів. В якості контурів відзначаються тільки ті точки, в яких знаходиться локальний максимум градієнта в напрямку вектору градієнта.

4. Подвійная порогова фільтрація. Високий поріг використовується для виявлення сильних контурів, низький поріг – для приєднання до сильних контурів, менш чітких, але які є їх частиною.

5. Трасування областей неоднозначності. Заглушуються слабкі контури, що не пов'язані з сильними.

Метод Кенні є досить універсальним апаратом вилучення контурів, але, як і більшість універсальних підходів, в спеціалізованих ситуаціях може бути не настільки ефективний. Наприклад, у випадку нечітких ліній, втрата яких буде помітна людському оку. Особливо гостро ця ситуація проявляється, коли нечіткі лінії не пов'язані з чіткими (червоні блоки на рис. 4.2).



А Б

Рис.4.2. Застосування фільтра Кенні до тестового зображенню "Bicycle"
А - оригінальне тестове зображення; Б - зображення після застосування фільтра Кенні

Векторний опис ліній

Для того щоб виділені контури поліпшили свою структуру і краще піддавалися аналітичному опису, класично до отриманого контуру застосовують морфологічні операції виду – нарощування, ерозія, замикання, розмикання (див. розділ, присвячений просторовим фільтрам).

Кінцевою метою обробки контурів є векторизація. Під векторизацією ми будемо розуміти спосіб представлення об'єктів і зображень, заснований на використанні елементарних геометричних об'єктів: точки, лінії, ламані, сплайни, багатокутники.

Перетворення Хафа

Одним з найвідоміших методів виявлення *прямої* є перетворення Хафа. Зазначимо, що перетворення Хафа можна узагальнити на випадок пошуку інших примітивів, таких як дуг кіл, та ін.

У перетворенні Хафа використовується представлення прямої у вигляді: $r = x\cos\theta + y\sin\theta$, де $\theta \in [0,\pi], r \in R$.

Тоді, функція, що задає сімейство прямих: $F(r,\vartheta,x,y) = x\cos\theta + y\sin\theta - r.$

Через кожну точку (x,y) зображення можна провести безліч прямих з різними параметрами r,θ. Кожній точці на зображенні (r_0,θ_0) простору (r,θ) можна поставити у відповідність лічильник, що відповідає кількості точок (x,y), що лежать на прямій $r_0 = x\cos\theta_0 + y\sin\theta_0$. Точка (r_0,θ_0), в якій досягається максимальне значення лічильника, описує параметри шуканої прямої (на рис. 1.3 найбільш темна точка – негатив максимального значення лічильника).

Щоб описати комп'ютерну реалізацію, перейдемо до дискретного випадку простору (r,θ). Таким чином, введемо сітку на (r,θ) просторі та поставимо у відповідність кожній клітині $[r_i,\theta_i]\times[r_{i+1},\theta_{i+1}]$ лічильник кількості точок (x,y), що задовольняють рівнянню

$r = x\cos\theta + y\sin\theta$, де $\theta \in [\theta_i, \theta_{i+!}], r \in [r_i, r_{i+1}]$.

А Б

Рис.4.3. Демонстрація перетворення Хафа
А - Оригінальний набір точок; Б - Негатив відображення простору Хафа – чим темніше точка, тим більше значення відповідного лічильника

Важливий аспект алгоритму – вибір розміру клітини. Розмір клітини потрібно вибирати з міркувань, щоб в якості прямої не був сприйнятий розрізнений набір точок і щоб клітини не були настільки малі, що призведе до відсутності різко вираженого максимального значення лічильника. Другий важливий аспект методу – швидкість його роботи.

Якщо на зображенні наявні кілька прямих, їх можна отримати за допомогою перестановки величин лічильників. Однак якщо прямі на зображенні повинні скластися в ламану лінію, після перетворення Хафа можуть з'явитися розриви, так як безперервність ланок ламаної в перетворенні ніяк не враховується. Тому для цього випадку переважний варіант безпосереднього наближення множини точок ламаними. У цьому випадку ми маємо справу з *кусково-лінійною апроксимацією.*

Метод кусково-лінійної апроксимації полягає в заміні заданої множини точок (x, y) ламаною з однією або декількома вузлами.

Опишемо побудову «майже» оптимального відновлення кривої в метриці Хаусдорфа (візуальної метриці) (див.[17]).

Опис асимптотично оптимальної ламаної

Нехай задана впорядкована множина зв’язаних точок (x_i, y_i), тоді для заданого порога $\varepsilon > 0$ число ланок асимптотично оптимальної інтерполяційної ламаної обчислюється за формулою:

$$m = \left[\frac{1}{4\sqrt{2\varepsilon}} \sum_{i=0}^{n-1} \Delta\ell_{i+1/2} (K_i + K_{i+1}) \right] + 1,$$

де $\lfloor \alpha \rfloor$ - ціла частина числа α,

$$K_i = \sqrt{\frac{|x_i'' y_i' - x_i' y_i''|}{\left((x_i')^2 + (y_i')^2\right)^{1/2}}},$$

$$x'_{i+1/2} = \frac{x_{i+1} - x_i}{\Delta\ell_{i+1/2}},\ y'_{i+1/2} = \frac{y_{i+1} - y_i}{\Delta\ell_{i+1/2}},$$

$$x'_{i+1/2} = \frac{x_{i+1} - x_i}{\Delta\ell_{i+1/2}},\ y'_{i+1/2} = \frac{y_{i+1} - y_i}{\Delta\ell_{i+1/2}},$$

$$\Delta\ell_{i+1/2} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2},$$

$$y_i' = \frac{1}{2}(y'_{i+1/2} + y'_{i-1/2}),\ x_i' = \frac{1}{2}(x'_{i+1/2} + x'_{i-1/2})$$

$$x_i'' = 2\frac{x'_{i+1/2} - x'_{i-1/2}}{\Delta\ell_{i+1/2} + \Delta\ell_{i-1/2}},\ y_i'' = 2\frac{y'_{i+1/2} - y'_{i-1/2}}{\Delta\ell_{i+1/2} + \Delta\ell_{i-1/2}}$$

$$x'_{-1/2} = 2x'_{1/2} - x'_{3/2},\ x'_{n+1/2} = 2x'_{n-1/2} - x'_{n-3/2},$$

$$y'_{-1/2} = 2y'_{1/2} - y'_{3/2},\ y'_{n+1/2} = 2y'_{n-1/2} - y'_{n-3/2}.$$

А координати вузлів ламаної $t_j, j = 0,1,...,m$ вибираються з умови:

$$\sum_{i=0}^{t_j} \Delta\ell_{i+1/2}\left(K_i + K_{i+1} + m^{-2/3}\right) \le \frac{j}{m}\sum_{i=0}^{m} \Delta\ell_{i+1/2}\left(K_i + K_{i+1} + m^{-2/3}\right) <$$

$$< \sum_{i=0}^{t_j+1} \Delta\ell_{i+1/2}\left(K_i + K_{i+1} + m^{-2/3}\right).$$

Таким чином, маємо асимптотично оптимальну інтерполяційну ламану з таким числом вузлів та їхнім розташуванням, що забезпечити задану похибку ε наближення.

Очевидно, що не всі криві можуть бути ефективно наближені ламаною с досить малою кількістю вузлів. Для опису складних кривих розроблена методика наближення сплайнами.

Функція $S(t)$, визначена на відрізку $[a,b]$, називається поліноміальних сплайном порядку m с вузлами $x_i \in (a \le x_0 < ... < x_n \le b)]$, якщо на кожному з відрізків $[x_{j-1}, x_j]$, $S(t)$ є алгебраїчним поліномом ступеня, який не перевищує m. Якщо в точці x_j, $(j = 0, n)$ функції $S(t), S^{(i)}(t), ...S^{(m-k)}(t)$ неперервні, а похідна $S^{(m-k)}(t)$ в точці x_j має розрив, число k називають дефектом сплайну. Множина $\{x_0, x_1, ..., x_n\}$ називається сіткою вузлів сплайну, а точки x_j – вузлами або точками склейки сплайну.

Приклад сплайну 2-го порядку (параболічний сплайн) наведений на рисунку.

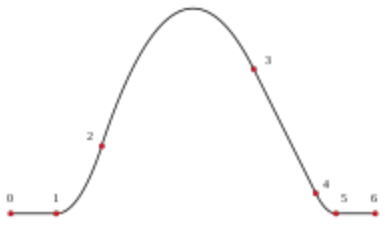


Рис.4.4. Приклад наближення функції сплайном 2-го порядку.

В комп'ютерних програмах обробки обробки графічних об'єктів прийнято використовувати сплайни Без'є. Це зумовлено, поперед всього, простотою їх реалізації.

Для рівномірного розбиття $x \in (a \le h < 2h < ... < (n-1)h < b)$ для точок $M_i, M_{i+1/2}, M_{i+1}$ квадратичний сплайн Без'є буде мати вигляд:

$$sb(\{M_i, M_{i+1/2}, M_{i+1}\}, t) = M_i(1-\tau)^2 + 2M_{i+1/2}\tau(1-\tau) + M_{i+1}\tau^2,$$

де $t \in [ih, (i+1)h]$.

Однак у сплайнів Без'є зворотною стороною їх простоти є достатньо висока помилка наближення. Якість наближення сплайнами Без'є така ж, як у інтерполяційних ламаних.

Локалізація слабоконтрастних ліній

Як вже було зазначено, проблема локалізації ліній, чітко виражених на зображенні, досить добре вивчена і існують алгоритми побудовані, наприклад, на фільтрах Собела, Превітта, Кенні, для виявлення таких ліній. Складність визначення прошарку ліній зростає з ускладненням фону, на яких ці лінії знаходяться. Розглянемо задачу виділення в окремий прошарок слабоконтрастних ліній з нечіткою структурою щодо фону прилеглих точок. Такі лінії, можна побачити на рис. 4.5

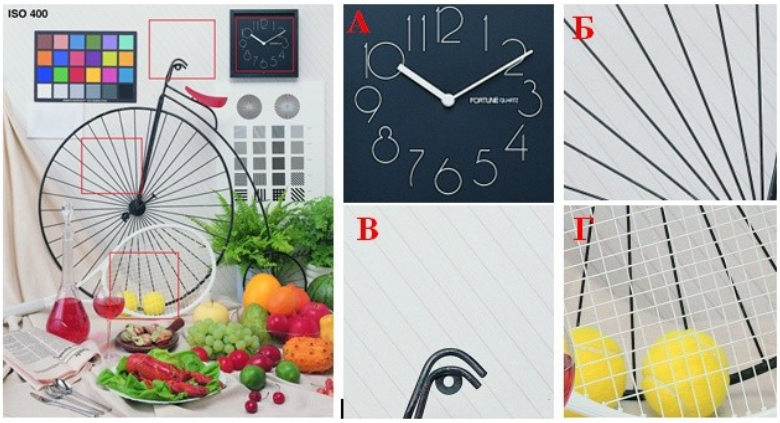


Рис.4.5. Тестовий приклад "Bicycle"
А і.Б - приклади ліній, контрастних на навколишньому фоні,
В і Г - приклади ліній, слабоконтрастних на навколишньому фоні

Виділення слабо контрастних ліній складається з 2-х етапів:
Етап 1. Формування множини точок фону;
Етап 2. Формування множин контрастних щодо фону.
Тепер більш докладно.

Етап 1. Формування точок фону.

Поняття локального фону є ключовим і інтуїтивно під фоном розуміється множина точок зображення, кожна з яких має достатнє число сусідів зі схожою освітленістю. При цьому не дуже помітні лінії на зображенні складаються з точок, які по своїй освітленості дуже близькі до яскравості сусідніх точок фону. Надалі в рамках опису алгоритму, це поняття буде формалізовано.

Будемо працювати з освітленістю зображення, що представляє собою числову матрицю $Y = \left\| y_{i,j} \right\|_{i=\overline{1,W},j=\overline{1,H}}$.

Через $C_{i,j}^R = \left\{ (v,\mu) : (i-v)^2 + (j-\mu)^2 \le R^2 \right\}$ позначимо окіл точки (i,j) радіусом R (у нашому випадку розглядалося $R = 3$) і кожній точці зображення поставимо у відповідність середню освітленість околу

$$b_{i,j}^R = \frac{1}{\sum \left\{ 1 \mid (v,\mu) \in C_{i,j}^R \right\}} \sum \left\{ y_{v,\mu} \mid (v,\mu) \in C_{i,j}^R \right\}. \qquad (4.6)$$

Для заданого $\delta > 0$ визначимо множину

$$\Re(\delta) = \left\{ (i,j) : \left| y_{i,j} - b_{i,j}^R \right| < \delta \right\}.$$

Всі точки цієї множини будемо називати первинними точками фону по порогу δ. При визначенні множини первинних точок фону, використовувалися точки, які в подальшому не увійшли в цю множину, тому треба перевизначити поняття околу, спираючись на інформацію тільки про точки з $\Re(\delta)$

$$\widetilde{C}^R_{i,j} = \{(v,\mu) \mid (v,\mu) \in C^R_{i,j},\ (v,\mu) \in \Re(\delta)\}.$$

Використовуючи уточнене поняття околу первинної точки фону, заново знайдемо значення освітленості фону в точці (i,j)

$$\widetilde{b}_{i,j} = \frac{1}{\sum\left\{1 \mid (v,\mu) \in \widetilde{C}^R_{i,j}\right\}} \sum\left\{y_{v,\mu} \mid (v,\mu) \in \widetilde{C}^R_{i,j}\right\}.$$

Значення освітленості точок, що належать лініям, слабоконтрастним щодо навколишнього фону, більш істотно відрізняються від значень яскравості фону, ніж значення точок, що не лежать на подібних місцях. При виділенні ліній природно вимагати, щоб точки, які можуть їм належати, не брали участь у формуванні освітленості фону. Для цієї мети побудуємо наступну процедуру.

Для кожної точки околу $\widetilde{C}^R_{(i,j)}$ знайдемо помилку опису її освітленості середнім значенням фону

$$e^{(i,j)}_{v,\mu} = \left| y_{v,\mu} - \widetilde{b}_{i,j} \right|,\ \text{де}\ (v,\mu) \in \widetilde{C}^R_{i,j},$$

та побудуємо множину точок $M^R_{(i,j)}$, яка складається з Θ - відсотків точок $(v,\mu) \in \widetilde{C}^R_{i,j}$, для яких значення помилок $e^{(i,j)}_{v,\mu}$ максимальні (наприклад, $\Theta \approx 20\%$). Обчислимо нові середні значення, видаливши точки з максимальним відхиленням:

$$\widetilde{b}^M_{i,j} = \frac{1}{\sum\left\{1 \mid (v,\mu) \in \widetilde{C}^R_{i,j} / M^R_{(i,j)}\right\}} \sum\left\{y_{v,\mu} \mid (v,\mu) \in \widetilde{C}^R_{i,j} / M^R_{(i,j)}\right\}.$$

Аналогічно попередньому твердженню, власне фоновими точками будемо вважати точки, що задовольняють умові:

$$\left| y_{i,j} - \widetilde{b}^{M^R_{(i,j)}}_{i,j} \right| < \varepsilon,\ (i,j) \in \Re(\delta), \tag{4.7}$$

де $\varepsilon > 0$ – параметр, що регулює кількість точок фону.

Всі точки, що задовольняють умові (4.7), утворюють множину точок фону $\widetilde{\Re}(\Theta,\delta)$.

Етап 2. Формування множин точок, контрастних на фоні.

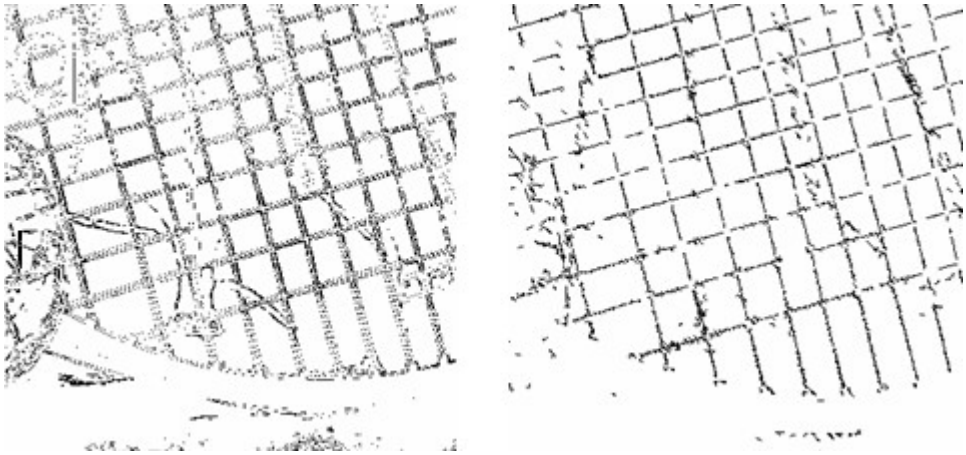
Точки, які належать множині $P^\delta = \Re(\delta) / \widetilde{\Re}(\Theta,\delta)$, будемо називати точками, контрастними на фоні. Всі ці точки "підозрілі" на предмет формування ліній. Через $\{P^\delta_k\}^n_{k=1}$ позначимо множини зв'язаних точок зображення, тобто точок, для яких виконується умова 8-ми зв'язності (1.5) двох точок (i,j) и (v,μ).

Мірою множини A будемо називати величину

$$M(A) = \sum\left\{1 \mid (i,j) \in A\right\}.$$

Зазначимо, що у множину P^δ можуть потрапити точки, що містять шум. Тому перш ніж, приступити до формування ліній на базі множини $\{P^\delta_k\}^n_{k=1}$, необхідно провести очистку зображення від шумів. Не зупиняючись на цьому питанні детально, зауважимо, що фільтрація даних від шуму є важливою і цікавою задачею. Існують різні алгоритми очищення, в даному випадку можна скористатися методом видалення елементів з малою мірою. Таким чином, в результаті отримуємо множину $\{P^\delta_k\}^m_{k=1}$

зв’язаних точок, які будемо називати слабо контрастними (нечіткими) лініями щодо фону.



А Б

Рис.4.6. А - виділення нечітких ліній за допомогою фільтрів Собела;
Б - виділення нечітких ліній за допомогою описаного підходу

Наступним кроком є формування з множин $\{P_k^\delta\}_{k=\overline{1,m}}$ безпосередньо ліній.

Формування суперліній

Складність топологічної структури множин $\{P_k^\delta\}_{k=\overline{1,m}}$ не дозволяє реалізувати їх ефективну обробку, тому є актуальним опис множин $\{P_k^\delta\}_{k=\overline{1,m}}$ об'єктами більш простої конструкції – суперлініями, чому присвячені подальші побудови.

Так як будь-яка зв'язна множина може бути представлена у вигляді підмножини об'єднання кіл, то $P_k^\delta \subset \bigcup_{(i,j)\in B_k} C_{(i,j)}^{R_{(i,j)}}$. Зв'язну множину $B(P_k^\delta) = \left\{(i_\nu, j_\nu)\right\}_{\nu=\overline{1,N_k}}$ будемо називати каркасом множини P_k^δ, якщо воно дає рішення наступної екстремальної задачі

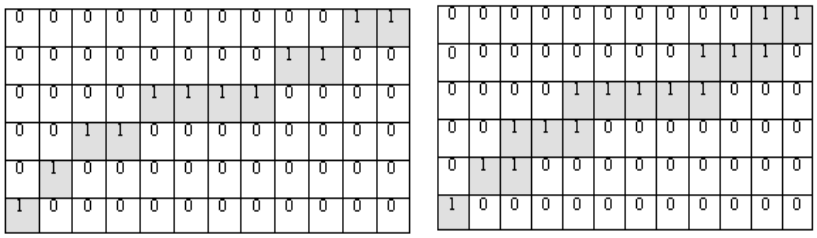
$$\begin{cases} \sum\limits_{(i,j)\in B_k} M\left(C_{(i,j)}^{R_{(i,j)}}\right) \to \min; \\ P_k^\delta \subset \bigcup\limits_{(i,j)\in B_k} C_{(i,j)}^{R_{(i,j)}}. \end{cases} \tag{4.8}$$

Каркас множини описує геометричну поведінку множини в цілому, тому для опису $\{P_k^\delta\}_{k=\overline{1,m}}$ скористаємося їх представленням через каркаси. Зауважимо, що отримання каркаса множини з використанням рішення задачі (4.8) трудомістке і, взагалі кажучи, не потрібне для коректного опису множини. Тому далі розглянемо спосіб побудови множини, близької до каркаса, конструювання якої істотно простіше.

Граничною точкою множини будемо називати точку, з якої граничить, хоча б одна точка, яка не належить цій множині.

Контуром будемо називати впорядковану замкнуту послідовність граничних точок, у якої ν и (ν+1)-я точка зв'язні, але кожна точка межує не більше ніж з двома граничними точками.

Під однопіксельною лінією будемо розуміти зв'язну множину, кожна точка (піксель) якої має не більш ніж 3 сусідів із множини (рис. 4.7Б). Під строго однопіксельною лінією будемо розуміти таку однопіксельну лінію, у якої видалення будь-якої точки, крім кінцевої, призводить до втрати зв'язності (рис. 4.7А).



А Б

Рис. 4.7. А - строго однопіксельна лінія; Б - Однопіксельна лінія

Операцією депіляції над множиною A для заданого околу $D_{i,j}^N$ будемо називати приєднання до множини всіх точок околу $D_{i,j}^N$, побудованої навколо кожної точки (i,j) множини A.

$$Dep(A) = A \cup \{(v,\mu) \mid (v,\mu) \in D_{i,j}^N, \forall (i,j) \in A\}.$$

Операцією епіляції $Epl(A)$ над множиною A будемо називати вилучення із множини A всіх граничних точок, які не порушують зв'язність A.

Застосування операції депіляції дозволяє з'єднати декілька множин в одну зв'язну. Таким чином, послідовне застосування епіляції і депіляції покращує структуру множин, що є підготовкою множини до побудови каркасів

$$A = Epl(Dep(A)).$$

Наведемо алгоритм побудова каркасу для множини точок А.

Етап 1. Побудова маски множини.

Поставимо у відповідність множині точок А матрицю $\{a_{i,j}\}_{i=\overline{1,W^A},j=\overline{1,H^A}}$, де

$$a_{i,j} = \begin{cases} 1, (i,j) \in A; \\ 0, (i,j) \notin A. \end{cases}$$

Етап 2. Знаходження контуру множини.

Задаємо правило обходу граничних точок. Для побудови контуру будемо обходити кожну граничну точку у фіксованому напрямку, наприклад, у додатному, і при цьому кожен елемент контуру p_k будемо характеризувати парою точок $V(p_k)$ і $U(p_k)$, де $V(p_k)$ координати k-ої точки контуру, а $U(p_k)$ – координати точки, з якої при заданому обході була додана k-та точка, тобто $U(p_k) = V(p_{k-1})$.

Побудова контуру починається з довільної граничної точки множини A. Просуваючись по напрямку контуру, приєднуємо черговий елемент. Матриця $\{a_{i,j}\}_{i=\overline{1,W^A},j=\overline{1,H^A}}$ відображає вплив кожної точки на формування контуру. Додаючи k-й елемент в контур, збільшуємо на одиницю значимість точки в матриці $\{a_{i,j}\}_{i=\overline{1,W^A},j=\overline{1,H^A}}$ з координатами $(V(p_k)_x, V(p_k)_y)$. За визначенням контур є замкнутим, тому процес повинен завершитися, коли в контур буде доданий елемент, що співпадає з першим, тобто $V(p_k) = V(p_1)$. Але для коректної обробки ситуації петель цього порівняння не достатньо, тому необхідно порівнювати всі характеристики елементу, тобто і $U(p_k)$. З огляду на те, що перший елемент не володіє інформацією про $U(p_1)$, будемо вважати, що контур побудований, якщо виконано наступні умови:

$$V(p_k) = V(p_2);$$
$$U(p_k) = U(p_2).$$

Останні два доданих елемента необхідні для алгоритмізації зупинки форування контуру, тобто не повинні впливати на формування $\{a_{i,j}\}_{i=\overline{1,W^A},j=\overline{1,H^A}}$.

Етап 3. Вилучення множини контуру з зображення.

Після побудови контуру аналізуємо вплив кожної граничної точки на формування контуру. Зауважимо, що одна і та ж гранична точка може бути присутня в декількох елементах контуру, в цьому випадку вона істотна для зв'язності множини А і не може бути видалена.

Для того щоб ефективно аналізувати входження кожної граничної точки в елементи контуру, не використовуючи додатковий пошук, була введена матриця $\{a_{i,j}\}_{i=1,j=1}^{W^A,H^A}$. Виходячи з її значень, легко проаналізувати вплив граничних точок на формування контуру. Елементи $a_{i,j}$ можуть приймати значення:

1 – якщо точка не входить в контур;

>2 – якщо точка входить у контур не один раз, а, значить, її видалення призводить до порушення зв'язності множини A.

2 – якщо точка, вимагає додаткового аналізу на видалення.

Видалення точки (i,j) супроводжується установкою значення елемента матриці $a_{i,j}=0$. Точки, відповідні значенням 2, які мають серед сусідів одиниці – видаляються. При цьому, можливий розпад множини А на незв'язні підмножини. Підмножини приводяться до строго однопіксельних лінії. Зауважимо, що для строго однопіксельної лінії легко знаходяться кінці – це точки, які мають тільки одного сусіда, що належить лінії. Очевидно, що розрив зв'язності може відбуватися в околі кінцівок строго однопіксельних підмножин. Для того щоб уникнути розриву лінії, кожен кінець отриманих фрагментів поміщається в окіл розміру 5×5. Якщо в цьому околі лежить хоча б одна точка, значення якої дорівнює 1, то повертаємо в окіл ту точку, яка відновить зв'язність. Одночасно відновлюємо точки, які мали в своїх сусідах більше трьох віддалених точок із значенням 2. Контур видаляється до тих пір, поки множина А не буде являти собою строго однопіксельну лінію. Рис. 4.8 ілюструє даний алгоритм.

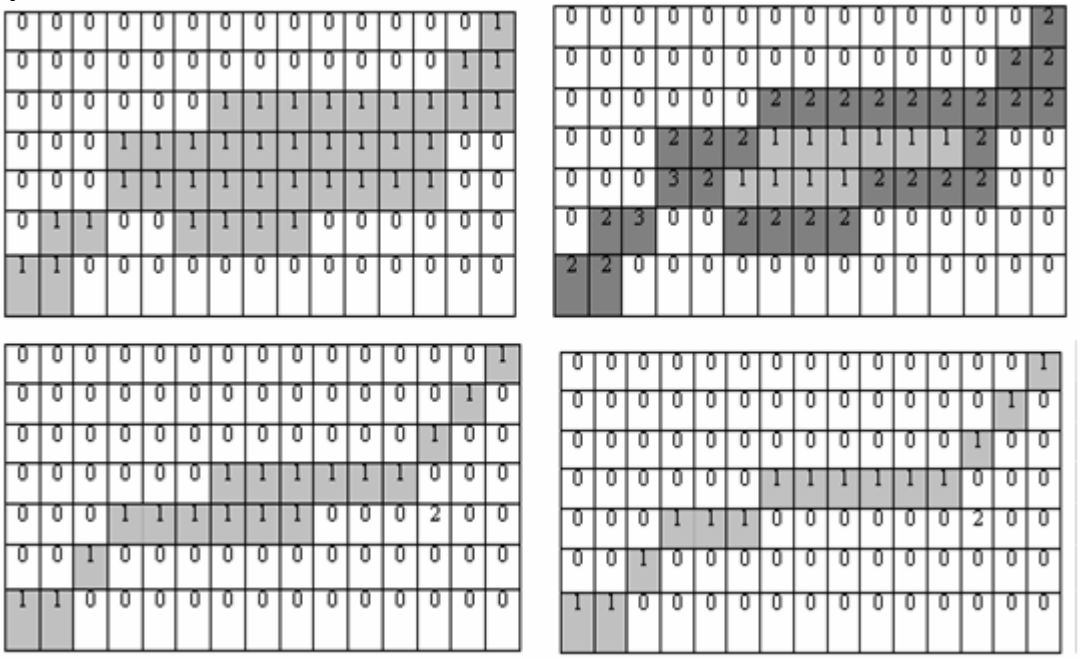


Рис.4.8. Приклад поетапного видалення контуру множини

Таким чином, отримуємо каркас ліній $B_k=\{P_k\}_{k=1}^{n}$. На рис. 4.9Б наведено приклад побудови каркасу для множини точок, зображених на рис. 4.9А.

А

Б

Рис.4.9. А - фрагмент нечітко виражених ліній;
Б - цей же фрагмент після обробки

Кінцевою точкою $\{P_k\}_{k=1}^{n}$ будемо назвати таку точку (x_k, y_k), для якої виконується умова:

$$\sum 1 \mid (x_i, y_i) \in D^1_{x_k, y_k}, (x_i, y_i) \in \{P_k\}_{k=1}^{n} \quad \leq 2 .$$

Суттєвою точкою множини $\{P_k\}_{k=1}^{n}$ будемо називати таку точку (x_k, y_k), яка або є кінцевою або її видалення призводить до порушення зв'язності множини $\{P_k\}_{k=1}^{n}$.

Однак каркас в чистому вигляді ще не є тією лінією, яку зручно наближати аналітично. У каркасі зазвичай присутні перетини, і візуально виділяються довгі лінії, з яких він складається.

Суперлінією будемо називати множину строго однопіксельних ліній, об'єднаних в місцях перетину.

Першим кроком побудови є отримання місць перехрещень.

Будемо називати вузловою точкою строго однопіксельної лінії таку точку (x_k, y_k), для якої виконується умова:

$$\sum \left| 1 \mid (x_{i,j}) \in D^1_{x_k, y_k} (x_{i,j}) \in \{P_k\}_{k=1}^{n} \right| > 3 .$$

Як показала практика для формалізації перетинів каркасу недостатньо спиратися лише на визначення вузлових точок. На рис. 4.10 представлені «гарні» вузлові точки – в цьому випадку перетин явно виражається в одній вузловій точці. На рис. 4.11 ситуація інакша – один перетин характеризується 4-ма вузловими точками.

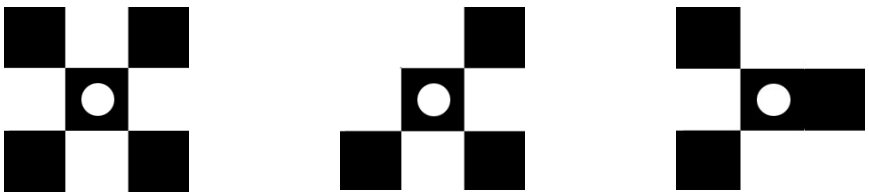


Рис. 4.10. Приклади «гарних» вузлових точок

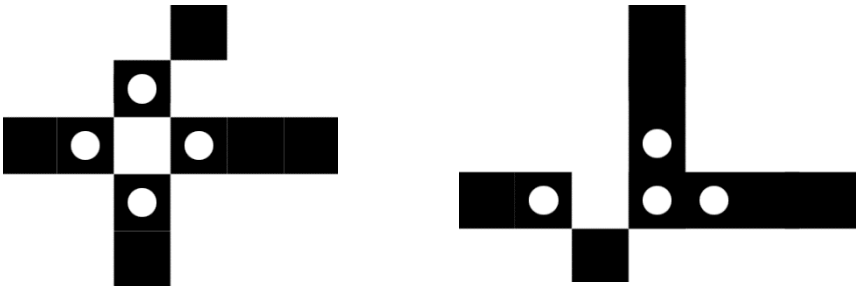


Рис. 4.11. Приклади «поганих» вузлових точок

Щоб описати усі варіанти введемо поняття вузла.

Вузлом будемо називати зв'язану множину точок, що спирається на вузлові точки і містять не більше ніж K не вузлових точок. Параметр K регулює допустиме поширення вузла. Зазвичай $K=3$.

З кожного вузла виходять фрагменти каркаса.

Фрагментом будемо називати строго однопіксельну лінію, яка не є вузлом, і має або дві вузлові точки, або дві кінцеві точки, або один вузол та одну кінцеву точку:

$$F_i = \left\{ p_k^i \right\}_{k=1}^{N_i},$$

де N_i – кількість точок у фрагменті, $p_1^i, p_{N_i}^i$ – вузлові або кінцеві точки.

Таким чином, каркас може бути представлений з вузлів і фрагментів каркаса, що є строго однопіксельними лініями:

$$\bigcup_i^{N_f} F_i \subset \{P_k\}_{k=1}^n,$$

де N_f – кількість фрагментів каркасу.

Будемо вважати, що в одну лінію можуть бути об'єднані два фрагменти, які виходять з одного вузла, якщо вони схожі по градієнту. Формалізуємо це твердження.

Нехай з вузла виходять m фрагментів $\{F_k\}_{k=1}^m$. Вважаємо, що точки кожного фрагмента впорядковані від поточного до наступного вузла або кінцевої точки. Це не представляє складності, враховуючи те, що фрагменти – строго однопіксельні лінії. Перші N (рекомендоване значення $N=5$) точок кожного фрагмента апроксимуємо за допомогою прямої. Таким чином, порівняння градієнта фрагментів зведемо до порівняння параметрів відповідних прямих.

Так як відстань від точки (x_i, y_i) до прямої $x\cos\varphi + y\sin\varphi + c = 0$ дорівнює

$$\varepsilon_i = \left| x_i \cos\varphi + y_i \sin\varphi + c \right|,$$

то пряму, найбільш близьку до точок $\left\{ (x_i, y_i) \right\}_1^N$, будемо шукати з умови:

$$\sum_{i=1}^{N} \left| x_i \cos\varphi + y_i \sin\varphi + c \right|^2 \to \min_{\varphi, c}. \tag{4.9}$$

Не зупиняючись докладно, знайдемо рішення $\tilde{\varphi}$ цієї екстремальної задачі

$$\tilde{\varphi} = \frac{1}{2} arctg \left(2 \frac{\sum_{\nu=1}^{N} x_\nu y_\nu - \frac{1}{N} \sum_{\nu=1}^{N} x_\nu \sum_{\nu=1}^{N} y_\nu}{\sum_{\nu=1}^{N} y_\nu^2 - \frac{1}{N} \left(\sum_{\nu=1}^{N} y_\nu \right)^2 - \sum_{\nu=1}^{N} x_\nu^2 + \frac{1}{N} \left(\sum_{\nu=1}^{N} x_\nu \right)^2} \right) + \frac{\pi k}{4}.$$

Таким чином, кожному фрагменту F_k поставимо у відповідність $\tilde{\varphi}_k$. Для того щоб упорядкувати ступені подібності всіх можливих пар фрагментів у вузлу, введемо

відстань $d(\varphi_1,\varphi_2)$ з врахуванням того, що $\widetilde{\varphi}+\pi$ також є рішенням екстремальної задачі.

Відкладемо кожен кут на одиничному колі.

Нехай функція $Pos(\varphi)$ - повертає координати кута на одиничному колі в декартовій системі координат. Таким чином, відстань між параметрами двох прямих знаходимо за формулою

$$d(\varphi_1,\varphi_2)=\min\left|d_E\left|(Pos(\varphi_1),Pos(\varphi_2)\right|,\ d_E\left|(Pos(\varphi_1),Pos(\varphi_2+\pi)\right|\right|,$$

де функція d_E – евклідова відстань, тобто

$$d_E((x_1,y_1),(x_2,y_2))=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

Побудуємо наступну матрицю відстаней між усіма можливими парами фрагментів

$$W=\begin{Bmatrix}\infty & d(\varphi_1,\varphi_2) & \dots & d(\varphi_1,\varphi_m)\\ d(\varphi_2,\varphi_1) & \infty & \dots & d(\varphi_2,\varphi_m)\\ \dots & \dots & \infty & \dots\\ d(\varphi_m,\varphi_1) & d(\varphi_m,\varphi_2) & \dots & \infty\end{Bmatrix}.$$

Значення елементів по діагоналі встановимо рівними нескінченності, так на цих місцях розташовано значення відстані між однією і тією ж прямою, а нас цікавлять тільки можливі пари.

Пара фрагментів F_i и F_j об'єднується в одну суперлінію, якщо

$$w_{i,j}=\min_{i,j=\overline{1,m}}(W)$$

Вважаючи, що F_i и F_j знайшли пару, встановлюємо у відповідних стовпцях і рядках матриці значення елементів рівними нескінченності. І знову будуємо нову пару. І так до тих пір, поки не залишиться елементів матриці, не рівних нескінченності або такий елемент буде один. Об’єднуємо всі фрагменти в лінію, якщо вони утворюють пари і мають спільний вузол.

Зауважимо, що отримані лінії носять явно дискретний характер, тому наступним етапом буде їх згладжування, що дозволить знаходити їх диференціальні характеристики.

Для простоти подальшого розгляду, наступні викладки будемо розглядати на прикладу однієї суперлінії, яка має собою множину $\{p_i\}_{i=\overline{1,n}}$.

Поповнимо вихідні дані $p_i=(x_i,y_i),\ i=1....n$ точками, щоб алгоритми працювали коректно для всіх точок лінії

$$p_0=\frac{1}{3}(4p_1+p_2-2p_3),\ p_{n+1}=\frac{1}{3}(4p_n+p_{n-1}-2p_{n-2}) \qquad (4.10)$$

і обчислимо згладжені значення

$$p_i^*=p_i+\alpha\Delta^2 p_i=p_i+\alpha(p_{i-1}-2p_i+p_{i+1}), \qquad (4.11)$$

де $\alpha\in\left(0,\frac{1}{3}\right]$ (якщо $\alpha=\frac{1}{3}$, то отримаємо медіанне згладжування).

Після того, як будуть перебрані всі точки, проведемо переприсвоєння $p_i=p_i^*(i=1,2..,n)$ і повторимо процес згладжування до тих пір поки заданого $\varepsilon>0$ и $\forall i=1,2..N$ не виконається умова:

$$(x'_{i+1/2})^2+(y'_{i+1/2})^2=1+\varepsilon,$$

де
$$x'_{i+1/2}=\frac{x_{i+1}-x_i}{\Delta\ell_{i+1/2}},\ y'_{i+1/2}=\frac{y_{i+1}-y_i}{\Delta\ell_{i+1/2}},$$
$$\Delta\ell_{i+1/2}=\sqrt{(x_{i+1}-x_i)^2+(y_{i+1}-y_i)^2},$$
$$x'_{-1/2}=2x'_{1/2}-x'_{3/2},\ x'_{n+1/2}=2x'_{n-1/2}-x'_{n-3/2}, \qquad (4.12)$$
$$y'_{-1/2}=2y'_{1/2}-y'_{3/2},\ y'_{n+1/2}=2y'_{n-1/2}-y'_{n-3/2}.$$

Таким чином, результатом роботи даного етапу є набір згладжених суперліній.

Аналітичний опис суперліній

Суперлінію $\{p_i\}_{i=1}^{n}$ будемо описувати в залежності від її геометричної форми. Будемо використовувати такі типи суперліній, як прямі, ламані, криві. Для визначення приналежності суперлінії до того чи іншого типу будемо застосовувати дискретний аналог кривизни в кожній точці.

Позначимо через Ψ_i дискретне значення кривизни у точці (x_i, y_i), яка обчислюється за формулою:

$$\Psi_i=\frac{\left|x''_i y'_i - x'_i y''_i\right|}{\left((x'_i)^2+(y'_i)^2\right)^{3/2}},\ i=0,1,...n,$$

де

$$y'_i=\frac{1}{2}(y'_{i+1/2}+y'_{i-1/2}),\ x'_i=\frac{1}{2}(x'_{i+1/2}+x'_{i-1/2})$$
$$x''_i=2\frac{x'_{i+1/2}-x'_{i-1/2}}{\Delta\ell_{i+1/2}+\Delta\ell_{i-1/2}},\ y''_i=2\frac{y'_{i+1/2}-y'_{i-1/2}}{\Delta\ell_{i+1/2}+\Delta\ell_{i-1/2}},$$

а відсутні значення похідних визначаються відповідно з формулами (4.12).

Таким чином, для точок $\{p_i\}_{i=1}^{n}$, які утворюють собою суперлінію, отримуємо набір $\{\Psi_i\}_{i=1}^{n}$, який буде критерієм вибору апарату наближення.

Етап 1. Апроксимація прямою.

Будемо вважати, що якщо для заданого порогу δ>0 виконується нерівність $\max\limits_{i=1,...,n}\Psi_i\le\delta$, то лінія достатньо добре описується прямою $x\cos\varphi+y\sin\varphi+c=0$, яка знаходиться з умови регресії.

У цьому випадку:

$$\varphi=\frac{1}{2}arctg\left(2\frac{\sum\limits_{\nu=0}^{n}x_\nu y_\nu-\frac{1}{n+1}\sum\limits_{\nu=0}^{n}x_\nu\sum\limits_{\nu=0}^{n}y_\nu}{\sum\limits_{\nu=0}^{n}y_\nu^2-\frac{1}{n+1}\left(\sum\limits_{\nu=0}^{n}y_\nu\right)^2-\sum\limits_{\nu=0}^{n}x_\nu^2+\frac{1}{n+1}\left(\sum\limits_{\nu=0}^{n}x_\nu\right)^2}\right)$$

та

$$c=-\frac{1}{n+1}\left(\cos\varphi\sum_{\nu=0}^{n}x_\nu+\sin\varphi\sum_{\nu=0}^{n}y_\nu\right).$$

Якщо умова $\max\limits_{i=1,...,n}\Psi_i\le\delta$ не виконується, то на наступному кроці зробимо спробу описати суперлінію ламаною.

Етап 2. Апроксимація ламаною.

При використанні ламаних потрібно, щоб необхідна точність опису лінії давала ламана з малим числом ланок, наприклад, не більше 10 ланок для опису ламаними.

Тобто якщо для множини точок $p_i^*(i=1,2..,n)$ не достатньо ламаної не більше ніж з 10 ланками, значить, для наближення потрібно використовувати інший апарат.

Для побудови такої ламаної використовувався алгоритм асимптотично оптимального відновлення кривої в метриці Хаусдорфа (візуальної метриці).

Наведемо алгоритм побудови оптимальної ламаної (з точки зору мінімізації вузлів при заданій похибці). Цей алгоритм дозволяє гарантовано отримати ламану з найменшим числом вузлів.

Нехай (x_i, y_i) $i=1,2..,n$ впорядковані точки, які відповідають деякому контуру Г, який необхідно описати та ε - допустима помилка. Через (x_1, y_1) позначимо стартову точку, в ролі якої візьмемо будь-яку точку з ε-околу точки (x_1, y_1), наприклад, саме цю точку. Побудуємо наступний ітераційний процес - нехай, спочатку, $i=2$ та $v=1$. Через точки (x^v, y^v) и (x_i, y_i) побудуємо пряму та знайдемо точку (x_j, y_j), віддалену від цієї прямої на відстань, яка дорівнює допустимої похибці, тобто

$$\frac{\left|(y_j - y_i)(x^v - x_i) - (x_j - x_i)(y^v - y_i)\right|}{\sqrt{(x^v - x_i)^2 + (y^v - y_i)^2}} \le \varepsilon \le \frac{\left|(y_{j+1} - y_i)(x^v - x_i) - (x_{j+1} - x_i)(y^v - y_i)\right|}{\sqrt{(x^v - x_i)^2 + (y^v - y_i)^2}}.$$

Проекцію точки (x_j, y_j) на пряму $(y - y_j)(x^v - x_i) = (x - x_i)(y^v - x_i)$ позначимо через (x^{v+1}, y^{v+1}) і довжина ланки ламаної буде дорівнює

$$h^v = \sqrt{(x^{v+1} - x^v)^2 + (y^{v+1} - y^v)^2}$$

Далі вважаємо i = i +1 і повторюємо цей процес, поки не знайдемо максимальне значення h^v. Після цього, в ролі стартової точки візьмемо (x^{v+1}, y^{v+1}) і цей процес будемо повторювати до тих пір, поки $j \le n$. В результаті отримаємо ламану, яка описує дану криву із заданою похибкою ε і з найменшим числом ланок (тобто з найменшим числом інформативних характеристик).

Якщо в результаті отримане число ланок досить мале, наприклад, не більше десяти, то замість шуканої кривої будемо використовувати цю ламану.

Якщо ж точок (x^v, y^v) багато, то для опису кривої використовуємо більш складний апарат.

Етап 3. Апроксимація сплайнами Бєз'є.

Для аналітичного опису суперлінії, яка не може бути наближена прямою чи ламаною з даною точністю, будемо використовувати апарат наближення Без'є - кривими.Розглянемо одну конструкцію сплайнів Без'є, яка, не змінюючи локальних властивостей апроксимації Без'є, для рівномірного розбиття дозволяє одержати інструмент наближення, що асимптотично співпадає з інтерполяційним сплайном мінімального дефекту.

Відомо, що для $t \in [ih, (i+1)h]$ будь-який параболічний сплайн мінімального дефекту по розбиттю ih $(i = 0, \pm 1, \pm 2, \ldots)$ можна записати у вигляді (див.[17])

$$s_2(t) = \sum_{i=-\infty}^{\infty} c_i B_{2,h}(t - (2i+1)h/2) = c_{i-1} B_{2,h}(t - (2i-1)h/2) + c_i B_{2,h}(t - (2i+1)h/2) + c_{i+1} B_{2,h}(t - (2i+3)h/2), \quad (4.13)$$

де $B_{2,h}(t)$ – параболічний В- сплайн по решітці із кроком h.

Покладемо $f_{i+1/2} = f((i+0.5)h)$ й $f_i = f(ih)$. У монографії [17] стор.88 показано, що якщо

$$\widetilde{c}_i = f_{i+1/2} - \frac{1}{8}\Delta^2 f_{i+1/2},$$

то сплайн

$$\widetilde{s}_2(f,t) = \sum_{i=1}^{n} \widetilde{c}_i B_{2,h}(t - (2i+1)h/2) \qquad (4.14)$$

асимптотично співпадає з інтерполяційним сплайном, тобто

$$\widetilde{s}_2(f,(i+0.5)h) = f_{i+1/2} - \frac{1}{64}\Delta^4 f_{i+1/2},$$

і при цьому, якщо $f \in C^3$, то для $t \in [ih,(i+1)h]$

$$f(t) - \widetilde{s}_2(f,t) = \frac{h^3}{3}\tau(1-\tau)(\tau-0.5)f^{(3)}(t) + o(h^3), \qquad (4.15)$$

де $\tau = \frac{t - ih}{h}$.

Сплайни виду (4.14) досить зручні й прості, однак у ряді програмних засобів для графічного відображення використовуваються параболічні сплайни Без'є. Побудуємо таку конструкцію сплайнов Без'є, які будуть співпадать з майже інтерполяційними сплайнами (4.14). Одержання такої конструкції сплайнів Без'є дозволяє підвищити ефективність вбудованих графічних функцій.

Розрахунок опорних точок Без'є кривої

Для $t \in [ih,(i+1)h]$ та точок $M_i, M_{i+1/2}, M_{i+1}$ сплайн Без'є має вигляд

$$sb(\{M_i, M_{i+1/2}, M_{i+1}\}, t) =$$
$$= M_i(1-\tau)^2 + 2M_{i+1/2}\tau(1-\tau) + M_{i+1}\tau^2.$$

Тоді похідна має вигляд

$$sb'(\{M_i, M_{i+1/2}, M_{i+1}\}, t) =$$
$$= \frac{2}{h}(-M_i(1-\tau) + M_{i+1/2}\tau(1-2\tau) + M_{i+1}\tau)$$

та

$$sb'(\{M_i, M_{i+1/2}, M_{i+1}\}, ih) = \frac{2}{h}(-M_i + M_{i+1/2}),$$
$$sb'(\{M_i, M_{i+1/2}, M_{i+1}\}, (i+1)h) = \frac{2}{h}(-M_{i+1/2} + M_{i+1}).$$

Крім того,

$$sb(\{M_i, M_{i+1/2}, M_{i+1}\}, (i+0.5)h) = \frac{1}{4}(M_i + 2M_{i+1/2} + M_{i+1}).$$

Зауважуючи, що

$$\widetilde{s}_2'(f,ih) = \frac{11f_{i+1/2} - f_{i+3/2} - 11f_{i-1/2} + f_{i+3/2}}{8h},$$

$$\widetilde{s}_2'(f,(i+1)h) = \frac{11f_{i+3/2} - f_{i+5/2} - 11f_{i+1/2} + f_{i-1/2}}{8h},$$

та

$$\widetilde{s}_2(f,(i+0.5)h) = \frac{1}{16}f_{i+3/2} - \frac{1}{64}f_{i+5/2} +$$
$$+ \frac{29}{32}f_{i+1/2} + \frac{1}{16}f_{i-1/2} - \frac{1}{64}f_{i-3/2},$$

випишемо умови неперервності сплайну та його похідної, що призводять до системи:

$$\begin{cases} sb(\{M_i, M_{i+1/2}, M_{i+1}\}, ih) = s_2(f, ih); \\ sb(\{M_i, M_{i+1/2}, M_{i+1}\}, (i+1)h) = s_2(f, (i+1)h); \\ sb(\{M_i, M_{i+1/2}, M_{i+1}\}, (i+0.5)h) = s_2(f, (i+0.5)h). \end{cases}$$

Розв'язуючи цю систему, одержуємо

$$\widetilde{M}_i = -\frac{1}{16} f_{i-3/2} + \frac{9}{16} f_{i-1/2} + \frac{9}{16} f_{i+1/2} - \frac{1}{16} f_{i+3/2},$$

$$\widetilde{M}_{i+1/2} = -\frac{1}{8} f_{i+3/2} + \frac{5}{4} f_{i+1/2} - \frac{1}{8} f_{i-1/2},$$

$$\widetilde{M}_{i+1} = -\frac{1}{16} f_{i-1/2} + \frac{9}{16} f_{i+1/2} + \frac{9}{16} f_{i+3/2} - \frac{1}{16} f_{i+5/2}.$$

Сплайн Без'є $sb(\{\widetilde{M}_i, \widetilde{M}_{i+1/2}, \widetilde{M}_{i+1}\}, t)$ буде співпадати з майже інтерполяційним параболічним сплайном мінімального дефекту.

Наведемо один приклад використання векторизації растрових зображень з метою зменшення розміру даних. Нижче наведено фрагмент векторного прошарку, на якому алгоритм автоматично вибрав відповідні варіанти наближення фрагментів ліній.

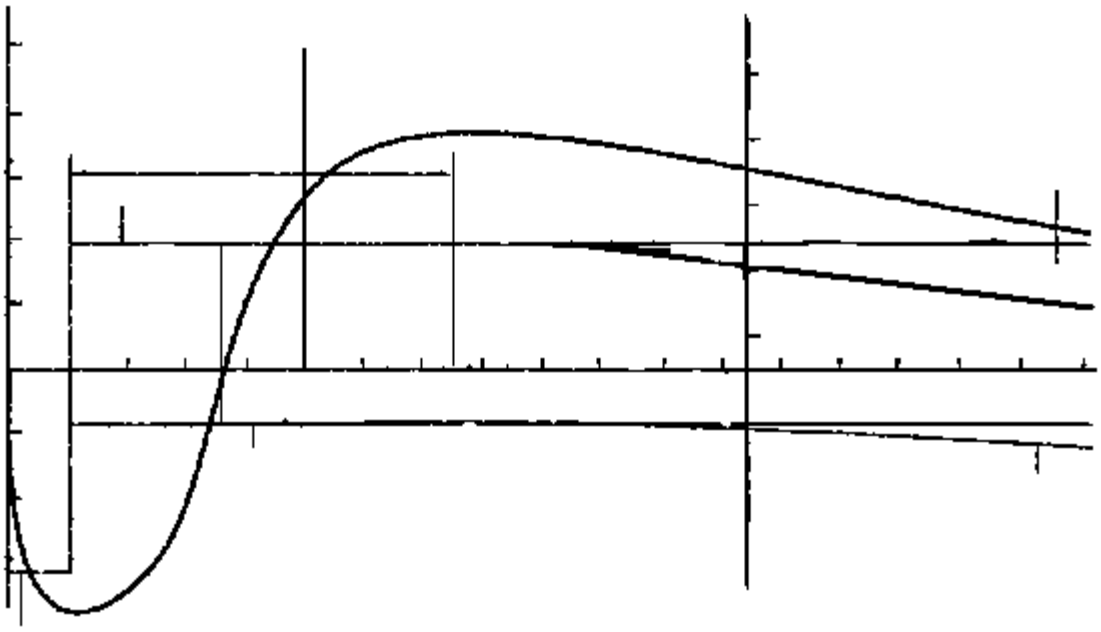


Рис.4.12. Оригінальний фрагмент у форматі TIF (3530 байт)

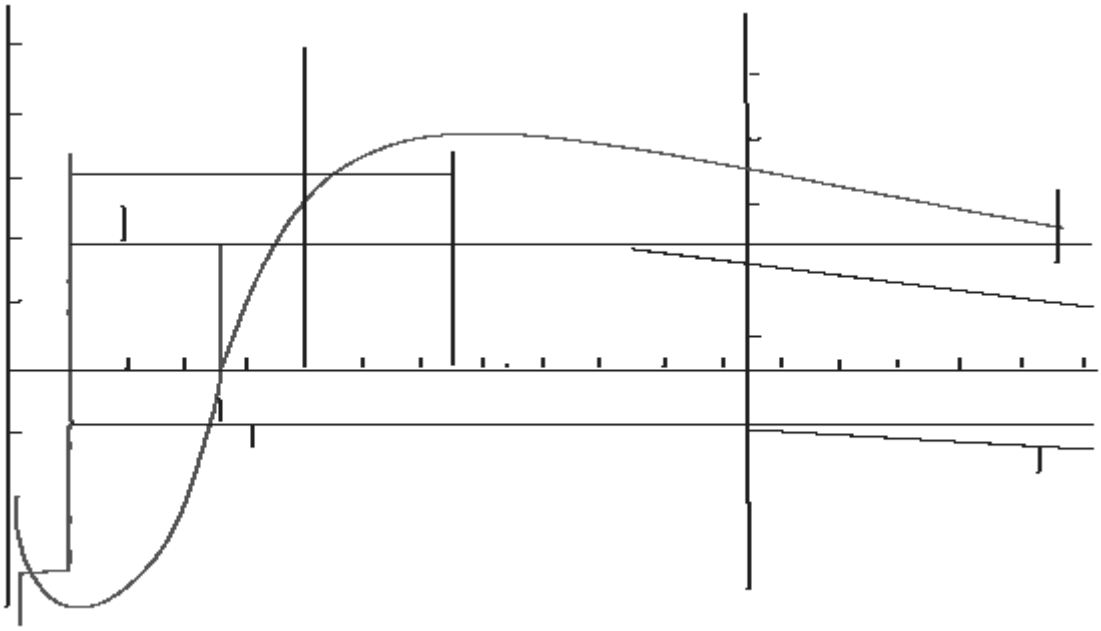


Рис.4.13. Векторизований фрагмент (800 байт)

Контрольні питання

1. Яка мета векторизаціїї растрових зображень?
2. Інструменти аналітичного опису векторних об’єктів?
3. Сплайни якого порядку найбільш вживані?
4. Наведіть плюси і мінуси використання кривих Без’є?
5. Яка ідея перетворення Хаффа?

Рекомендована література

1. Duda R.O. Use Of The Hough Trasformtion To Detect Lines And Curves In Pictures / Richard O. Duda, Peter E. Hart // The Communications of ACM .− 1972 .− Vol 15, No. 1 .− P. 11-15.
2. Денисюк В. С. Применение и оптимизация преобразования Хафа для поиска объектов на изображении / В.С. Денисюк // Международный конгресс по информатике: информационные системы и технологии: материалы международного научного конгресса 31 окт.– 3 нояб. 2011 г. .− Минск: БГУ, 2011.− Ч. 2.– С. 162-165.
3. Guil N. A fast Hough transform for segment detection, Image Processing / N. Guil, J. Villalba, Zapata // IEEE Transactions .− Nov 1995 .− P. 1541–1548.
4. Прэтт У. Цифровая обработка зображений. Кн. 2 / У. Прэтт .− М.: Мир, 1982 .− 480 с.
5. Trajkovich M. Fast corner detection / M. Trajkovich, M. Handley // Image and Vision Computing .−1998 .− 16 .− P. 75-87.
6. Rad Ali A. Fast Circle Detection Using Gradient Pair Vectors / Ali Ajdari Rad, Karim Faez, Navid Qaragozlou // Proc. VIIth Digital Image Computing .− Sydney, Australia, 2003.− P. 879-887.

7. Fisher, Perkins, Walker & Wolfart (2003). "Spatial Filters - Laplacian of Gaussian". Retrieved 2010-09-13 : [Electronic resource] .– http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm.
8. R. Deriche Using Canny's criteria to derive a recursively implemented optimal edge detector / R. Deriche // Int. J. Computer Vision .− 1987 .− Vol. 1 .− P. 167–187.
9. Davis L.S. A survey of edge detection techniques / L.S. Davis // Computer Graphics and Image Processing .− 1975 .− vol 4, no. 3 .− P. 248-260.
10. Роджерс Д. Математические основы машинной графики / Д. Роджерс, Дж. Адамс .– М.: Мир,2001 .– 357 с.
11. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс . – М.: Техносфера, 2005 .– 1070 с.
12. Canny John, A Computational Approach to Edge Detection, Pattern Analysis and Machine Intelligence / John Canny // IEEE Transactions on .– Nov. 1986 .– 679 – 698 p.
13. Prewitt J.M.S. Object Enhancement and Extraction / J.M.S. Prewitt // Picture processing and Psychopictorics .– NY:Academic Press,1970 .– P. 75-149
14. Sobel vs. Prewitt Masks for Edge Detection: [Electronic resource] .– http://the-evan.com/files/eel5820/project%2004/eel5802_pr04-05_turner_evan.pdf.
15. Ту Дж. Принципы распознавания образов / Дж. Ту, Р. Гонсалес .− М., 1978 .− 410 с.
16. Введение в контурный анализ / Я.А. Фурман, А.В. Кревецкий, А.К. Передреев и др.; Под ред. Я.А. Фурмана .− М.: ФИЗМАТЛИТ, 2003 .− 392 с.
17. Лигун А.А. Асимптотические методы восстановления кривих / А.А. Лигун, А.А. Шумейко .− К.: Институт математики НАН Украины, 1997 .− 358 с.
18. Лигун А.О. ALLDocument – технологія нового покоління для збереження, передачі та відображення електронних документів / А.О. Лигун, О.О. Шумейко, Д.В. Тимошенко // Вісник Східноукраїнського національного університету імені Володимира Даля .− 2006 .− №9 (103) Частина 1 .− С. 83-85.
19. Лигун А.А. Локализация и формирование линий на изображении / А.А. Лигун, А.А. Шумейко, Д.В. Тимошенко // Системные технологии. Региональный межвузовский сборник научных трудов .− Днепропетровск, 2007 .− № 3(50) .− С. 5-14.
20. Лигун А.О. Про побудову майже інтерполяційних сплайнів Без'є / А.О. Лигун, О.О. Шумейко, Д.В. Тимошенко // Актуальні проблеми автоматизації та інформаційних технологій. Збірник наукових праць .– Дніпропетровськ, 2008 .– Том 12 .– С. 125-133.
21. Asymptotically Optimum Recovery of Smooth Contours by Bezier Curve / A.A.Ligun, A.A.Shumeiko, S.P.Radzevich, E.D.Goodman // Computer Aided Geometric Design .— 1998 .— №15 .— С.495-506.

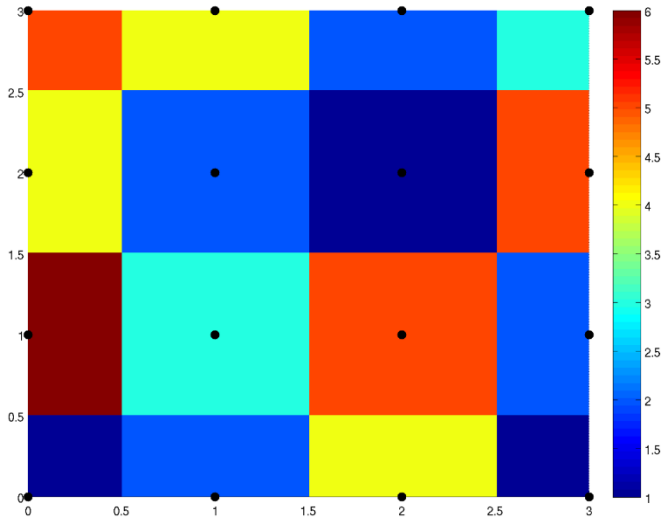
ТЕМА 5. МАСШТАБУВАННЯ ЗОБРАЖЕНЬ

Через дискретність відображення сигналів (образів), вся графічна інформація розглядається як дискретна. Цей факт приводить до певних незручностей при масштабуванні зображень. При зменшенні масштабу (накат на образ, збільшення геометричного розміру зображення, підвищення глибини перенесення кольорів і ін.) кількість інформації, що відображається, збільшується, а при збільшенні масштабу (зменшенні геометричного розміру зображення, зниженні глибини перенесення кольорів і ін.) її кількість зменшується. Процес збільшення точок, які відображуються, називається інтерполяцією, зменшення- децимацією (проріджування)(від латинського decimating - знищення кожного десятого полоненого воїна переможеної римлянами армії).

Методи інтерполяції і децимації одновимірних даних

При реалізації алгоритмів інтерполяції і децимації можливо, що нове значення лежить в точці, в якій раніше було старе значення, в цьому випадку говорять, що фазове зрушення перетворення дорівнює нулю. Якщо координати нової точки лежать у середині між старими, то фазове зрушення дорівнює 1/2, 1/2 і так інше.

Найпростішим алгоритмом масштабування є метод найближчого сусіда. Ідея даного алгоритму дуже проста – при зміні масштабу зображення розраховуємо координати нових пікселів (вузли нової решітки). Для кожного нового вузла знаходимо найближчий вузол старої решітки і колір пікселя, який відповідає знайденому вузлу оригінальної решітки присвоюємо розрахованому пікселю.



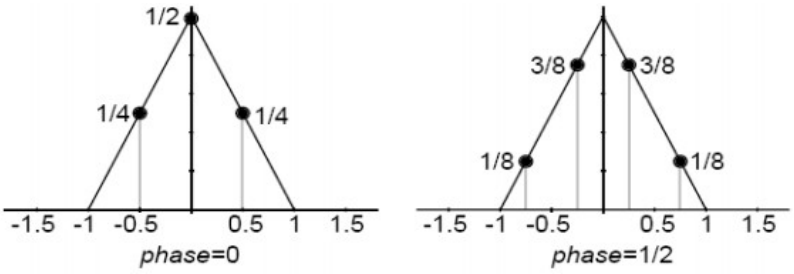
Результат інтерполяції фрагменту зображення заданого на решітці [0,3]×[0,3] методом найближчого сусіда

Очевидно, що результат такого масштабування буде дуже фрагментованим, тому сфера використання даного алгоритму обмежується тими задачами, коли критичним є не якість отриманого результату, а швидкість роботи алгоритму, наприклад, для масштабування відео-потоку на мобільних засобах.

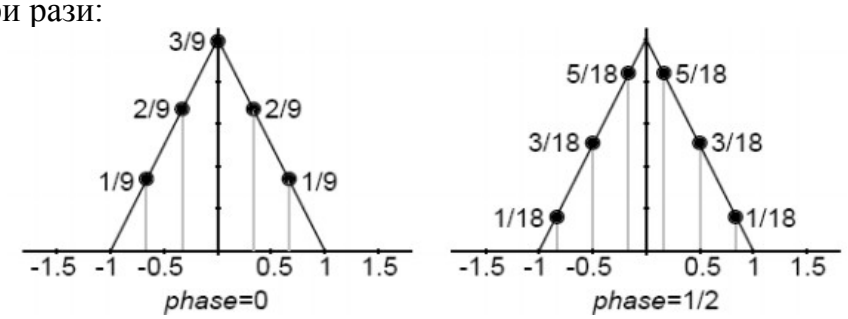
У разі отримання більш пристойних результатів, треба замінити дискретну множину, яку представляє собою растрове зображення, неперервною поверхнею, і вже при масштабуванні зображення, знімати нові дані з поверхні. Таким чином, задача масштабування так чи інакше зводиться до побудови векторного образу растрового зображення.

Розглянемо декілька підходів, які використовуються для вирішення такого роду задач.

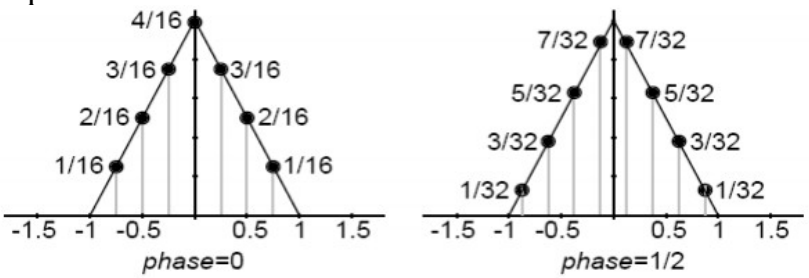
Найчастіше для побудови фільтрів використовуються різні інтерполяційні формули, як правило на основі інтерполяційних многочленів. Ми розглянемо найпростіший випадок алгебраїчної інтерполяції - лінійної. Фільтр для децимації даних в два рази може бути реалізований таким чином:



децимація в три рази:



децимація в чотири рази:



Ясно, що для цього випадку, інтерполяція нових значень виходить зняттям цих значень з інтерполяційної ламаної.

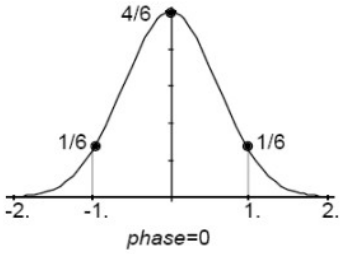
Однією з найчастіше використовуваних функцій для подібного роду задач є функція Гауса

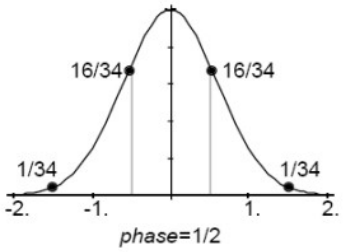
$$y = e^{-a^2x^2}.$$

У якості прикладу, розгляне функцію

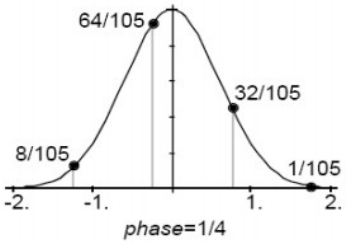
$$y = 2^{-\frac{1}{2}x^2}.$$

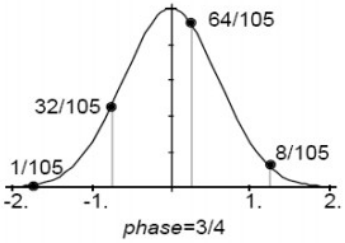
Для інтерполяції в два рази, слід використовувати наступну маску



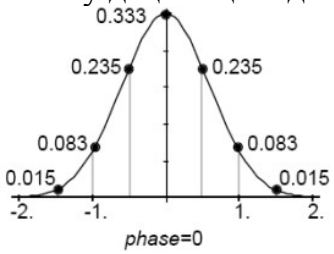


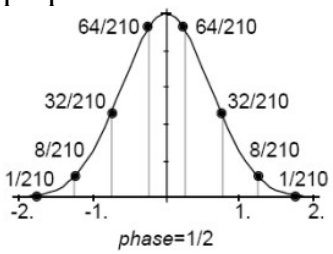
Інтерполяція в три рази виходить таким чином

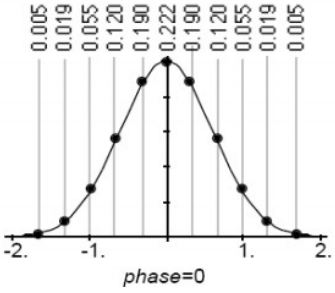


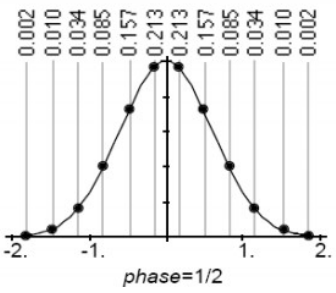


Як приклад приведемо маску децимації в два і три рази:



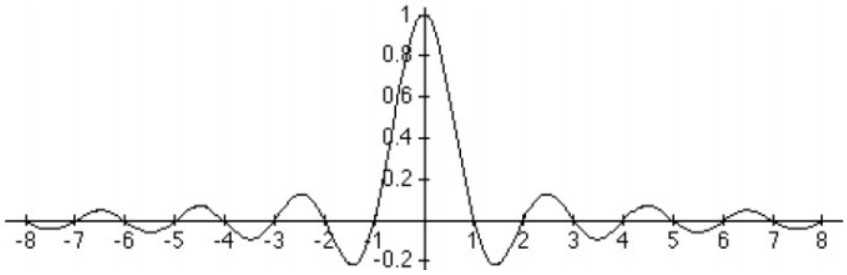






Широко вживаною функцією для побудови низькочастотного фільтру є функція sinc(x) (від лат. sinus cardinalis — кардинальний синус)

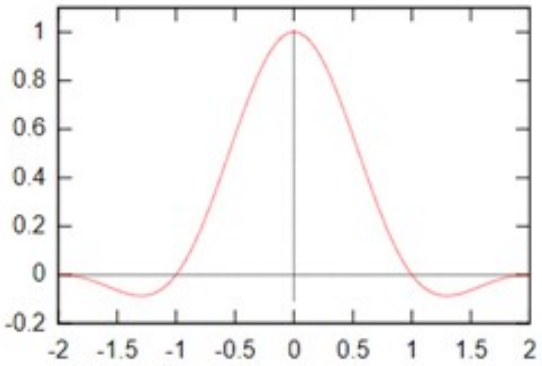
$$\operatorname{sinc}(x) = \begin{cases} \dfrac{\sin(\pi x)}{\pi x}, & x \neq 0, \\ 1, & x = 0. \end{cases}$$



Внаслідок того, що значення цієї функції хоча і швидко затухають, але її носій не обмежений, тому для побудови фільтрів на основі цієї функції беруть добуток її на деяку віконну функцію. Результатом будуть, наприклад, наступні функції

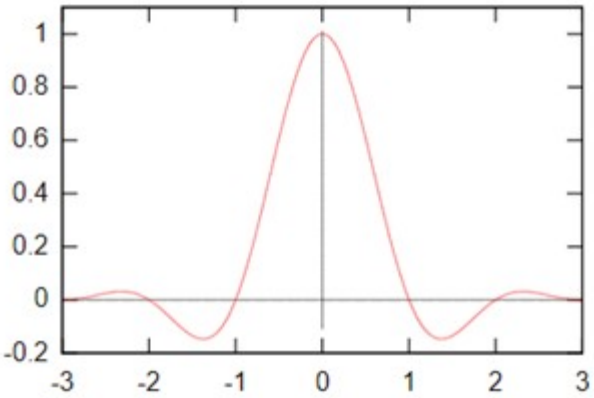
$$\text{Lanczos2}(x) = \begin{cases} \dfrac{\sin(\pi x)}{\pi x}\dfrac{\sin\left(\pi\frac{x}{2}\right)}{\pi\frac{x}{2}}, & |x| < 2, \\ 0, & |x| \geq 2. \end{cases}$$

Яка названа на честь угорського вченого Корнелія Ланцоша (Lánczos Kornél).

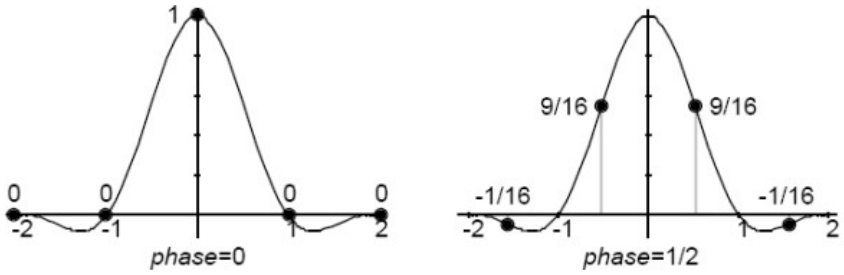


або

$$\text{Lanczos3}(x) = \begin{cases} \dfrac{\sin(\pi x)}{\pi x}\dfrac{\sin\left(\pi\frac{x}{3}\right)}{\pi\frac{x}{3}}, & |x| < 3, \\ 0, & |x| \geq 3. \end{cases}$$



Приведемо фільтри інтерполяції в два рази, одержані на основі функції Lanczos2

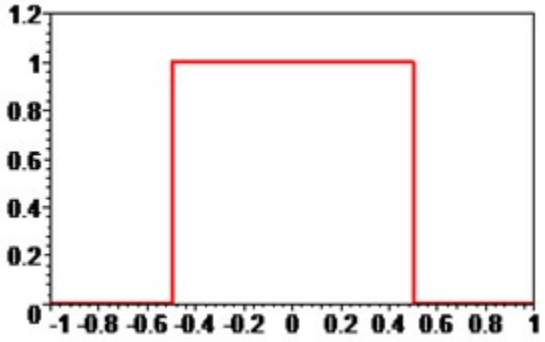


Застосування фільтра Ланцоша дозволяє добитися високої чіткості зображення, але при обробці можлива поява небажаних артефактів типу ореолів навколо контрастних переходів яскравості. Виникнення ореолів обумовлено тим, що ядро Ланцоша приймає від'ємні значення при деяких значеннях аргументу. Тому оброблений сигнал може приймати навіть від'ємні значення при додатних значеннях вибірок.

Популярним інструментом масштабування є використання моделювання поверхні зображення тензорним добутком сплайнів, як правило, це білінійні і бікубічні сплайни.
Для розглядання цього методу, нам необхідні деякі означення.

Базовий сплайн, або B-сплайн, є сплайном з мінімальним носієм для фіксованого ступеня сплайну та порядку гладкості. Їхня безумовна користь полягає в тому, що будь-який сплайн заданого ступеня і гладкості можна записати у вигляді лінійної комбінації відповідних базисних сплайнів. Тобто, їх можна використовувати як "цеглинки", за допомогою яких можна побудувати сплайн із заданими властивостями. Власне чим і пояснюється його визначення – базисний сплайн.
Надалі розглянемо B-сплайни мінімального дефекту порядку r. Найбільш популярними є B-сплайни порядку від 0 до 3, проте у нашому випадку більшу увагу треба приділити сплайнам 1-го та 3-го порядків (лінійний та кубічний відповідно).
B-сплайн нульового порядку $B_0(t)$ має наступний вигляд

$$B_0(t) = \begin{cases} 1, & |t| < \frac{1}{2}, \\ 0, & |t| \geq \frac{1}{2}. \end{cases}$$



Типовим представником сплайнів нульового порядку, визначених на усій координатній осі по одиничній решітці, є B-сплайн нульового порядку, його зрушення та лінійні комбінації:

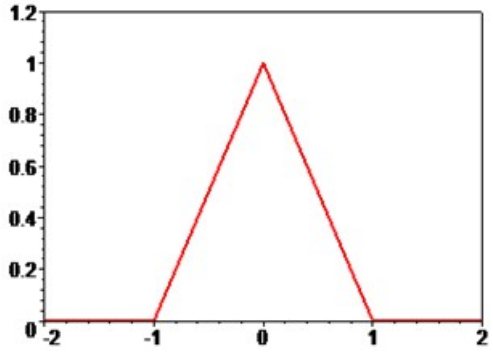
$$s_0(t) = \sum_{i=-\infty}^{i=\infty} c_i B_0\left(t - \frac{2i+1}{2}\right).$$

У загальному вигляді, функцію $B_r(t)$ введемо за допомогою рекурентних співвідношень:

$$B_r(t)=\int_{t-1/2}^{t+1/2} B_{r-1}(x)dx,\ (r=1,2,\dots).$$

B-сплайн першого порядку

$$B_1(t)=\begin{cases} t+1, & t\in[-1,0],\\ 1-t, & t\in[0,1],\\ 0, & |t|\geq 1. \end{cases}$$

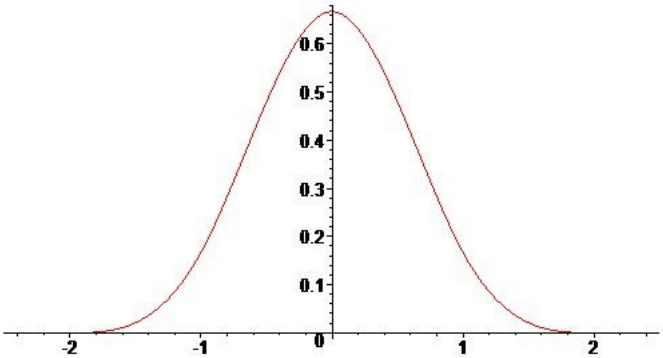


Типовим представником сплайну 1-го порядку є B-сплайн 1-го порядку, його зрушення та лінійні комбінації:

$$s_1(t)=\sum_{i=-\infty}^{i=\infty} c_i B_1(t-i).$$

Нарешті, випишемо кубічний B-сплайн

$$B_3(t)=\begin{cases} (4-2t)^3, & (t\in[1,2]),\\ 3(2t)^3-12(2t)^2+32, & (t\in[0,1]),\\ -3(2t)^3-12(2t)^2+32, & (t\in[-1,0]),\\ (4+t)^3, & (t\in[-2,-1]),\\ 0, & (|t|\geq 2). \end{cases}$$



Типовим представником кубічного сплайну буде

$$s_3(t)=\sum_{i=-\infty}^{i=\infty} c_i B_3(t-i).$$

Нехай маємо одиничну решітку з вузлами $\boldsymbol{i}=(i,j)$, де $\boldsymbol{i}\in Z^2$ і позначимо через

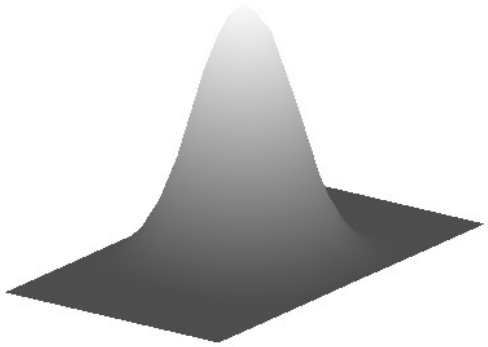
$$s_{r,s}(\boldsymbol{P},\boldsymbol{u})=\sum_{i\in Z^2}\boldsymbol{P}_i N_{r,s}(\boldsymbol{u}-\boldsymbol{i}) \qquad (5.1)$$

де $\boldsymbol{u} = (u, v) \in R^2$ і

$$N_{r,s}(\boldsymbol{u} - \boldsymbol{i}) \equiv B_r(u - i)B_s(u - i)$$

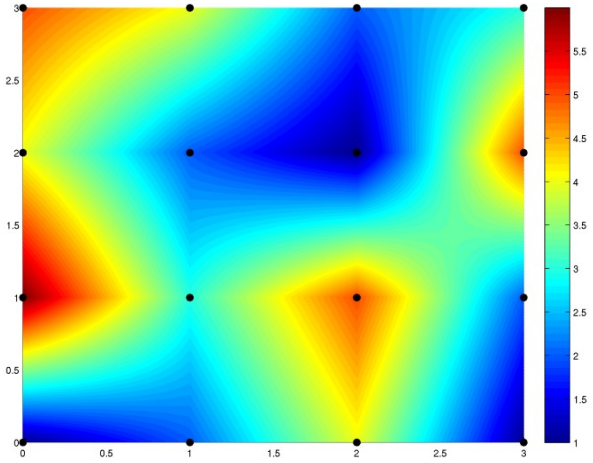
нормалізований тензорний добуток B-сплайнів порядку r та s з вузлами $\boldsymbol{i} = (i, j)$.

У разі $r = s = 1$ такий сплайн називають білінійним, якщо $r = s = 3$, то бікубічним.



Бікубічний сплайн $s_{3,3}(\boldsymbol{P}, \boldsymbol{u})$.

Якщо $\boldsymbol{P}_i$ значення кольору у пікселі $\boldsymbol{i} = (i, j)$, то сплайн (5.1) можна використати у якості аналітичного опису зображення і використовувати для вирішення задачі масштабування.



Результат білінійної інтерполяції фрагменту зображення заданого на решітці [0,3]×[0,3]

Розглянемо нормалізований тензорний добуток кубічних B-сплайнів з вузлами $\boldsymbol{i} = (i, j)$

Для $\boldsymbol{u} \in [i, j] \times [(i + 1), (j + 1)]$ маємо

$$s_{3,3}(\boldsymbol{P},\boldsymbol{u}) = [1 \quad u-i \quad (u-i)^2 \quad (u-i)^3]\mathbf{M}\mathbf{P}\mathbf{M}^{\mathbf{T}}\begin{bmatrix} 1 \\ v-j \\ (v-j)^2 \\ (v-j)^3 \end{bmatrix} \tag{5.2}$$

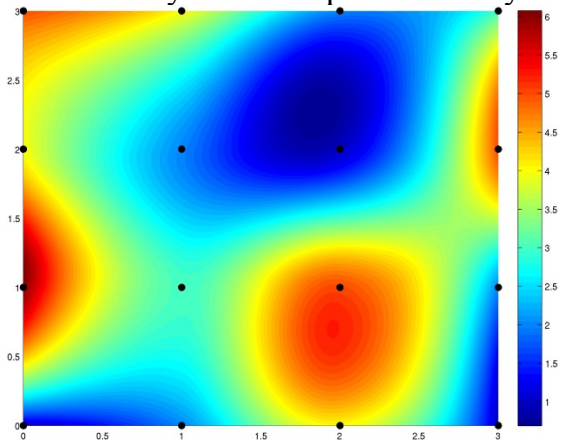
де $\mathbf{M}$ матриця розміру 4×4

$$\mathbf{M} = \frac{1}{6}\begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

та

$$\mathbf{P} = \begin{bmatrix} P_{i-1,j-1} & P_{i,j-1} & P_{i+1,j-1} & P_{i+2,j-1} \\ P_{i-1,j} & P_{i,j} & P_{i+1,j} & P_{i+2,j} \\ P_{i-1,j+1} & P_{i,j+1} & P_{i+1,j+1} & P_{i+2,j+1} \\ P_{i-1,j+2} & P_{i,j+2} & P_{i+1,j+2} & P_{i+2,j+2} \end{bmatrix}$$

Таким чином, підставляючи у (5.2) значення $\boldsymbol{u}$, які відповідають новій решітці, отримуємо рішення задачі масштабування зображення бікубічними сплайнами.



Результат бікубічної інтерполяції фрагменту зображення,
заданого на решітці [0,3]×[0,3].

Порівняємо обробку вхідного зображення (зміна геометричного розміру зображення) за допомогою двох алгоритмів: білінійної та бікубічної інтерполяцій.

Для порівняння візьмемо кілька зображень з TID2008 (Tampered Image Database-2008). Ця база призначена для тестування метрик візуальної якості зображення, що має на увазі наявність еталону. TID2008 є доступною для використання у наукових цілях та вільного завантаження.

Перше тестове зображення з TID2008

Друге тестове зображення з TID2008

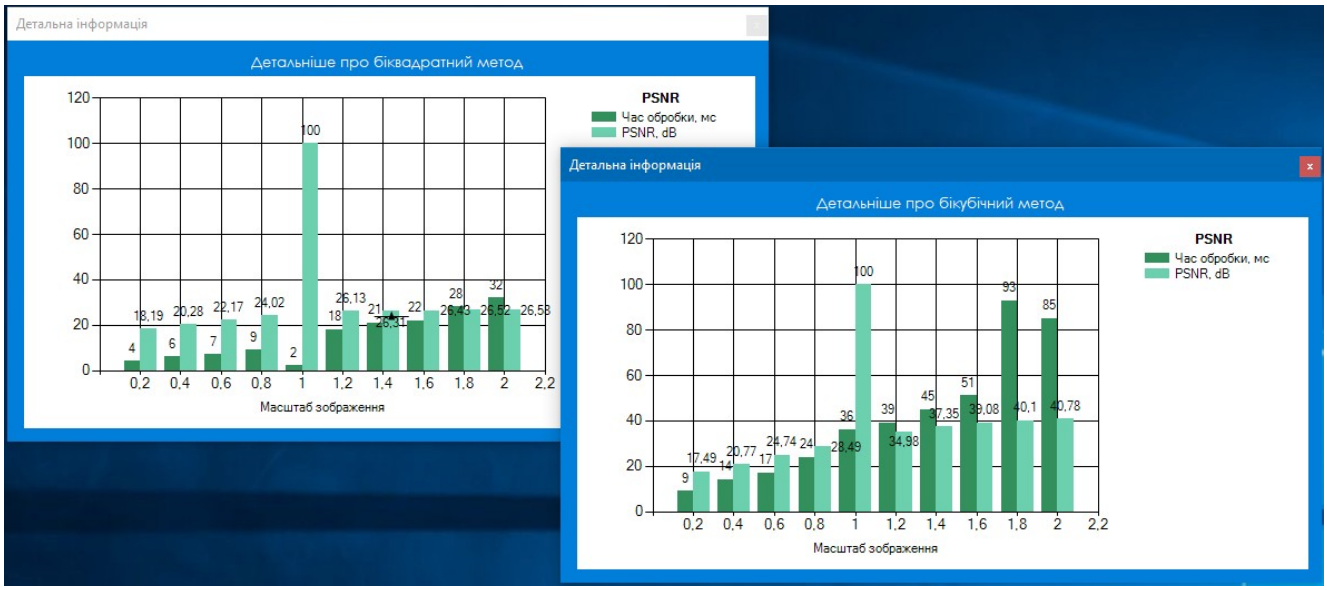
Третє тестове зображення з TID2008

Для порівняння результатів роботи описаних методів будемо використовувати такі показники, як час обробки зображення та PSNR.

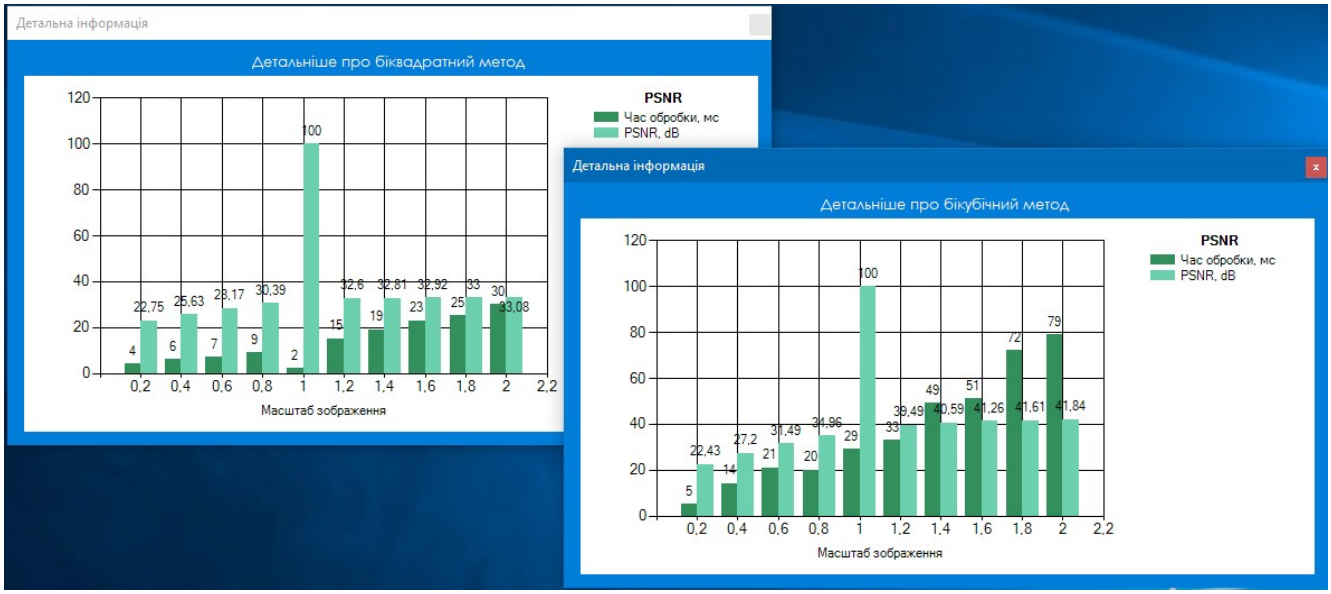
Абревіатура PSNR (pealsignal-to-noiseratio) означає пікове співвідношення сигналу до шуму і є інженерним терміном, що визначає співвідношення між максимумом можливого значення сигналу та потужністю шуму, що спотворює значення сигналу.

Зазвичай, значення PSNR знаходиться у межах від 20 до 40 dB. У разі співпадання зображень, значення MSE стає нульовим, що дає для PSNR значення нескінченності. У такому випадку прийнято вважати, що PSNR = 100.

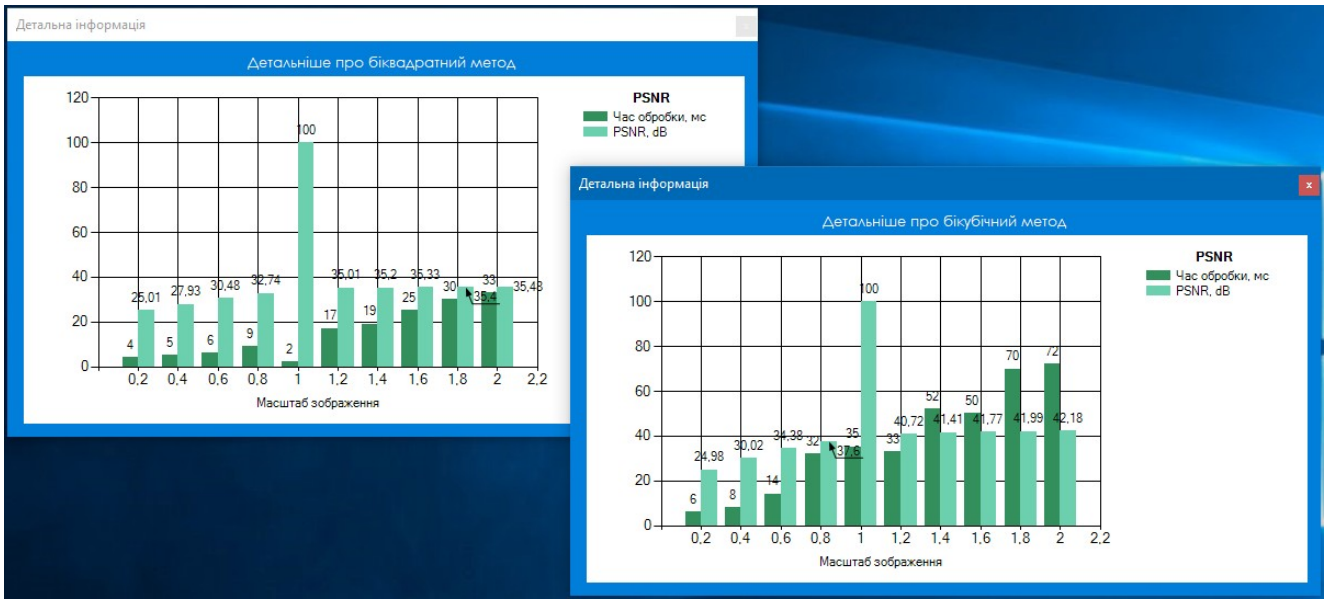
Отримані результати відображають показники часу обробки та PSNR для оброблених зображень до розмірів починаючи з 20% оригінального розміру, закінчуючи 200%.



Результат обробки першого зображення



Результат обробки другого зображення



Результат обробки третього зображення

Проаналізувавши отримані результати можна зробити висновок, що при бікубічній інтерполяції показник PSNR зазвичай вищий, ніж при білінійній. Чим більша схожість між порівнюваними зображеннями, тим менше величина MSE, а отже показник PSNR більший. На основі цього можна зробити висновок про те, що зображення отримане при бікубічній інтерполяції є більш якісним і близьким до оригіналу, проте також видно, що витрати часу на обробку зображення є більшими у порівнянні з білінійною інтерполяцією.

Але, треба зазначити, що апроксимативні характеристики наведених тензорний добутків як білінійного, так і бікубічного, мають один і той же порядок, хоча у другому випадку можна отримати більш якісні результати.

Наведемо один з підходів, який дозволяє це зробити.

Розглянемо опис поверхні тензорним добутком кубічних B-сплайнів на квадратній решітці $\boldsymbol{ih} = (ih, jh)$, де $\boldsymbol{i} \in Z^2$

$$s_{3,3}(\boldsymbol{P},\boldsymbol{u}) = \sum_{i \in Z^2} \boldsymbol{P}_i N_{3,3}(\boldsymbol{u} - \boldsymbol{ih})$$

де $\boldsymbol{u} = (u, v) \in R^2$ і

$$N_{r,s}(\boldsymbol{u} - \boldsymbol{ih}) \equiv B_{r,h}(u - ih) B_{s,h}(u - ih)$$

нормалізований тензорний добуток B-сплайнів порядку r та s з вузлами $\boldsymbol{i} = (i, j)$ по решітці з кроком h.

Для $\boldsymbol{u} \in [ih, jh] \times [(i+1)h, (j+1)h]$ маємо

$$s_{3,3}(\boldsymbol{P},\boldsymbol{u}) = \begin{bmatrix} 1 & \dfrac{u-ih}{h} & \left(\dfrac{u-ih}{h}\right)^2 & \left(\dfrac{u-ih}{h}\right)^3 \end{bmatrix} \mathbf{M}\mathbf{P}\mathbf{M}^{\mathbf{T}} \begin{bmatrix} 1 \\ \dfrac{v-jh}{h} \\ \left(\dfrac{v-jh}{h}\right)^2 \\ \left(\dfrac{v-jh}{h}\right)^3 \end{bmatrix}$$

де $\mathbf{M}$ та $\mathbf{P}$ матриці розміру 4×4 наведені вище.

У точці (ih, jh) значення бікубічного сплайну дорівнює

$$s_{3,3}(\boldsymbol{P},\boldsymbol{ih}) = \frac{4}{9}P_{i,j} + \frac{1}{9}\left(P_{i,j+1} + P_{i,j-1} + P_{i+1,j} + P_{i-1,j}\right) + \frac{1}{36}\left(P_{i-1,j+1} + P_{i-1,j-1} + P_{i+1,j-1} + P_{i+1,j+1}\right) = P_{i,j} - \frac{1}{36}\Delta P_{i,j}$$

де

$$\Delta P_{i,j} = 20P_{i,j} - 4\left(P_{i,j+1} + P_{i,j-1} + P_{i+1,j} + P_{i-1,j}\right) - \left(P_{i-1,j+1} + P_{i-1,j-1} + P_{i+1,j-1} + P_{i+1,j+1}\right)$$

Нехай поверхня $\boldsymbol{f}$ задана точками

$$f_{\boldsymbol{i}} = f(ih, jh), i, j = 0, \pm 1, \pm 2, \ldots$$

і

$$\widehat{\boldsymbol{P}}_{\boldsymbol{i}} = \boldsymbol{P}_{\boldsymbol{i}}(\boldsymbol{f}) = f_i + \frac{1}{36}\Delta f_i$$

Відповідний сплайн буде мати вигляд

$$\hat{s}_{3,3}(\boldsymbol{P},\boldsymbol{u}) = \sum_{i \in Z^2} \widehat{\boldsymbol{P}}_{\boldsymbol{i}} N_{3,3}(\boldsymbol{u} - \boldsymbol{ih})$$

Тоді

$$\hat{s}_{3,3}(\boldsymbol{P},\boldsymbol{ih}) = f_i + \frac{1}{1296}\Delta f_i$$

Якщо функція $f(\boldsymbol{u})$ має всі неперервні похідні до четвертого порядку включно, то існує точка $\boldsymbol{u_0} \in [(i-0.5)h, (j-0.5)h] \times [(i+0.5)h, (j+0.5)h]$ така, що

$$\hat{s}_{3,3}(f,\boldsymbol{ih}) = f_i + \frac{2}{3}h^4\left(\left.\frac{\partial^4 f}{\partial u^4}\right|_{u_0} + 2\left.\frac{\partial^4 f}{\partial u^2 \partial v^2}\right|_{u_0} + \left.\frac{\partial^4 f}{\partial v^4}\right|_{u_0}\right)$$

Таким чином, якщо для бікубічного сплайну (2) замість $\boldsymbol{P_i} = P_{i,j}$ взяти

$$\widehat{P}_{i,j} = P_{i,j} + \frac{1}{36}\Delta P_{i,j}$$

де

$$\Delta P_{i,j} = 20P_{i,j} - 4\left(P_{i,j+1} + P_{i,j-1} + P_{i+1,j} + P_{i-1,j}\right) - \left(P_{i-1,j+1} + P_{i-1,j-1} + P_{i+1,j-1} + P_{i+1,j+1}\right)$$

тоді якість масштабованого зображення буде суттєво краща.

Треба зазначити, що так як задача масштабування зображення реально зводиться до зміни розмірів у раціональне число раз, тобто, треба замінити одну матрицю (растрове зображення) на іншу матрицю (масштабоване растрове зображення), то заміна оригіналу на векторний образ не є обов'язковою. Можна підійти до цієї проблеми, з іншої точки зору. У якості модельної ситуації розглянемо бінарне зменшення зображення з точки зору апроксимаційної точності.

Метод бінарного збільшення для двовимірних даних

Розглянемо конструкцію збільшення розміру, яка є оптимальною для заданого методу відновлення на фіксованій множині фільтрів. На першому кроці розглянемо відновлення функцій, неперервно диференційованих (до шостого порядку включно) на всій дійсній площині. Також будемо вважати, що задана деяка фіксована функція $\varphi(x,y)$ така, що її згортка з константою є константою і центральні моменти якої

$$m_{\vartheta,\mu} = \iint\limits_{R^2} x^{\vartheta} y^{\mu} \varphi(x,y)dxdy = \iint\limits_{D} x^{\vartheta} y^{\mu} \varphi(x,y)dxdy$$

задовольняють умовам

$$m_{\vartheta,\mu} = m_{\mu,\vartheta}, m_{2\vartheta+1,\mu} = m_{\vartheta,2\mu+1},$$

а також, для відновлення константи необхідне виконання умови $m_{0,0} = 1.$

Будемо вважати, що інформація про функцію f задається множиною функціоналів

$$c_{2i,2j} = \iint\limits_{R^2} f(x+ih, y+jh)\varphi(x,y)dxdy,$$

де h - крок решітки та $i,j \in Z$.

Для відновлення функції f у вузлах решітки (ih,jh) побудуємо наступний алгоритм (типу метода Рунге-Кутта).

Напершому кроці побудуємо реконструкцію функції f у вузлах $(2ih,2jh)$

$$\hat{f}_{2i,2j} = \alpha_0 c_{2i,2j} + \frac{\alpha_1}{4}\left(c_{2i-2,2j} + c_{2i+2,2j} + c_{2i,2j-2} + c_{2i,2j+2}\right) + \frac{\alpha_2}{4}\left(c_{2i-2,2j-2} + c_{2i+2,2j-2} + c_{2i-2,2j-2} + c_{2i+2,2j+2}\right) + \frac{\alpha_3}{4}\left(c_{2i-4,2j} + c_{2i+4,2j} + c_{2i,2j-4} + c_{2i,2j+4}\right).$$

На наступному кроці обчислимо реконструкцію функції fу вузлах $((2i+1)h,(2j+1)h)$ з використанням отриманих значень $\hat{f}_{2i,2j}$ за правилом

$$\hat{f}_{2i+1,2j+1} = \frac{\beta_0}{4}\left(c_{2i,2j} + c_{2i+2,2j} + c_{2i,2j+2} + c_{2i+2,2j+2}\right) + \frac{\beta_1}{4}\left(\hat{f}_{2i,2j} + \hat{f}_{2i+2,2j} + \hat{f}_{2i,2j+2} + \hat{f}_{2i+2,2j+2}\right).$$

Нарешті, з урахуванням всіх отриманих значень, знайдемо

$$\hat{f}_{2i+1,2j} = \frac{\gamma_0}{2}\left(c_{2i,2j} + c_{2i+2,2j}\right) + \frac{\gamma_1}{2}\left(\hat{f}_{2i,2j} + \hat{f}_{2i+2,2j}\right) + \frac{\gamma_2}{2}\left(\hat{f}_{2i+1,2j-1} + \hat{f}_{2i+2,2j+1}\right).$$

Аналогічно обчислимо значення $\hat{f}_{2i,2j+1}$.

Відповідний метод позначимо через *F(C,f, φ)*.

Теорема 1. Для того, щоб метод *F(C,f, φ)* був такий, що

$$f_{i,j} - F(C, f, \varphi, ih, jh) = f_{i,j} - \hat{f}_{i,j} = O(h^6), \tag{5.3}$$

необхідно і достатньо, щоб центральні моменти функції φ задовольняли умовам

$$m_{0,4} = -5m_{0,2},\ m_{2,2} = -m_{0,2}, \tag{5.4}$$

і при цьому значення коефіцієнтів будуть визначатися наступним чином

$$\alpha_0 = 1 + \frac{5}{8}m_{0,2} - \frac{1}{16}m_{2,2} + \frac{5}{16}m_{0,2}^2 - \frac{1}{32}m_{0,4},$$

$$\alpha_1 = -\frac{2}{3}m_{0,2} + \frac{1}{8}m_{2,2} - \frac{1}{2}m_{0,2}^2 + \frac{1}{24}m_{0,4},$$

$$\alpha_2 = -\frac{1}{16}m_{2,2} + \frac{1}{8}m_{0,2}^2,$$

$$\alpha_3 = \frac{1}{24}m_{0,2} + \frac{1}{16}m_{0,2}^2 - \frac{1}{96}m_{0,4},$$

$$\beta_0 = -\frac{1}{m_{0,2}}, \beta_1 = 1 + \frac{1}{m_{0,2}},$$

$$\gamma_0 = -\frac{1}{2m_{0,2}}, \qquad \gamma_1 = \frac{1}{2} + \frac{1}{2m_{0,2}}, \qquad \gamma_0 = \frac{1}{2}.$$

Дійсно, використовуючи формулу Тейлору в околі точки *(2ih, 2jh)*, одержуємо

$$c_{2i,2j} = \iint_{R^2} f(x + ih, y + jh)\varphi(x, y)dxdy = f_{2i,2j}m_{0,0} + m_{2,0}\frac{h^2}{2}\frac{\partial^2 f}{\partial x^2}\bigg|_{(2i,2j)} + m_{0,2}\frac{h^2}{2}\frac{\partial^2 f}{\partial y^2}\bigg|_{(2i,2j)} + m_{4,0}\frac{h^4}{24}\frac{\partial^4 f}{\partial x^4}\bigg|_{(2i,2j)} + m_{2,2}\frac{h^4}{4}\frac{\partial^4 f}{\partial x^2 \partial y^2}\bigg|_{(2i,2j)} + m_{0,4}\frac{h^4}{24}\frac{\partial^4 f}{\partial y^4}\bigg|_{(2i,2j)} + O(h^6).$$

Використовуючи формулу реконструкції в точці *(2ih, 2jh),* отримуємо

$$\hat{f}_{2i,2j} = (\alpha_0 + \alpha_1 + \alpha_2 + \alpha_3) f_{2i,2j}$$
$$+ \frac{h^2}{4}\Big(2\alpha_0 m_{0,2} + \alpha_1(4 + 2m_{0,2}) + \alpha_2(8 + 2m_{0,2})$$
$$+ \alpha_3(16 + 2m_{0,2})\Big)\left(\left.\frac{\partial^2 f}{\partial x^2}\right|_{(2i,2j)} + \left.\frac{\partial^2 f}{\partial y^2}\right|_{(2i,2j)}\right)$$
$$+ \frac{h^4}{4}\Big(2\alpha_0 m_{2,2} + \alpha_1(4m_{0,2} + m_{2,2}) + \alpha_2(16 + 8m_{0,2} + m_{2,2})$$
$$+ \alpha_3(16m_{0,2} + m_{2,2})\Big)\left.\frac{\partial^4 f}{\partial x^2 \partial y^2}\right|_{(2i,2j)}$$
$$+ \frac{h^4}{24}\Big(\alpha_0 m_{0,4} + \alpha_1(8 + 12m_{0,2} + m_{0,4}) + \alpha_2(16 + 24m_{0,2} + m_{0,4})$$
$$+ \alpha_3(128 + 48m_{0,2} + m_{0,4})\Big)\left(\left.\frac{\partial^4 f}{\partial x^4}\right|_{(2i,2j)} + \left.\frac{\partial^4 f}{\partial y^4}\right|_{(2i,2j)}\right) + O(h^6).$$

Таким чином, для того, щоб умова (5.3) виконувалась, треба розв'язати систему рівнянь

$$\begin{cases} \alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 = 1, \\ 2\alpha_0 m_{0,2} + \alpha_1(4 + 2m_{0,2}) + \alpha_2(8 + 2m_{0,2}) + \alpha_3(16 + 2m_{0,2}) = 0, \\ 2\alpha_0 m_{2,2} + \alpha_1(4m_{0,2} + m_{2,2}) + \alpha_2(16 + 8m_{0,2} + m_{2,2}) + \alpha_3(16m_{0,2} + m_{2,2}) = 0, \\ \alpha_0 m_{0,4} + \alpha_1(8 + 12m_{0,2} + m_{0,4}) + \alpha_2(16 + 24m_{0,2} + m_{0,4}) + \alpha_3(128 + 48m_{0,2} + m_{0,4}) = 0. \end{cases}$$

Розв'язуючи дану систему, отримаємо коефіцієнти $\alpha_0, \alpha_1, \alpha_2, \alpha_3$, наведені в теоремі.

Аналогічним чином, якщо прирівняємо до нуля множники перших похідних різниці $f_{2i+1,2j+1} - \hat{f}_{2i+1,2j+1}$, тоотримаємо систему рівнянь, розв'язком якої будуть значення

$$\beta_0 = -\frac{1}{m_{0,2}}, \beta_1 = 1 + \frac{1}{m_{0,2}},$$

При цьому похибка відновлення в точці *((2i+1)h,(2j+1)h)* буде дорівнювати

$$f_{2i+1,2j+1} - \hat{f}_{2i+1,2j+1}$$
$$= \frac{h^4}{24}\left(5 + \frac{m_{0,4}}{m_{0,2}}\right)\left(\left.\frac{\partial^4 f}{\partial x^4}\right|_{(2i,2j)} + \left.\frac{\partial^4 f}{\partial y^4}\right|_{(2i,2j)}\right)$$
$$+ \frac{h^4}{4}\left(1 + \frac{m_{2,2}}{m_{0,2}}\right)\left.\frac{\partial^4 f}{\partial x^2 \partial y^2}\right|_{(2i,2j)} + O(h^6)$$

Нарешті, мінімізуючи похибку реконструкції функції в точці *((2i+1)h,2jh),* отримаємо

$$\gamma_0 = -\frac{1}{2m_{0,2}}, \qquad \gamma_1 = \frac{1}{2} + \frac{1}{2m_{0,2}}, \qquad \gamma_0 = \frac{1}{2}$$

і, в такому разі, маємо

$$f_{2i+1,2j} - \hat{f}_{2i+1,2j} = \frac{h^4}{24}\left(5 + \frac{m_{0,4}}{m_{0,2}}\right)\left(\left.\frac{\partial^4 f}{\partial x^4}\right|_{(2i,2j)} + \left.\frac{\partial^4 f}{\partial y^4}\right|_{(2i,2j)}\right) + \frac{h^4}{4}\left(1 + \frac{m_{2,2}}{m_{0,2}}\right)\left.\frac{\partial^4 f}{\partial x^2 \partial y^2}\right|_{(2i,2j)} + O(h^6)$$

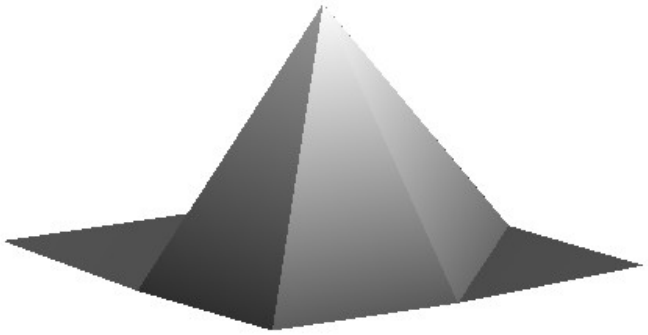
Таким чином, вибір моментів згідно (5.4), забезпечує виконання умови (5.3).

У дискретному випадку, коли значення $\varphi(x,y)$ на кожному елементарному квадраті решітки є константою, складність φ визначимо наступним чином

$$\begin{matrix} a_{-1,1} & a_{0,1} & a_{1,1} \\ a_{-1,0} & a_{0,0} & a_{1,0} \\ a_{-1,-1} & a_{0,-1} & a_{1,-1} \end{matrix}$$

Така функція, що задовольняє умовам теореми, існує і єдина, при цьому ненульові значення декомпозиційного фільтрувизначаються матрицею

$$\begin{matrix} -\frac{31}{9360} & -\frac{19}{720} & -\frac{31}{9360} \\ -\frac{19}{720} & \frac{1309}{1170} & -\frac{19}{720} \\ -\frac{31}{9360} & -\frac{19}{720} & -\frac{31}{9360} \end{matrix}$$



В цьому випадку центральні моменти функції φбудуть такі

$$m_{0,0} = 1, m_{0,2} = \frac{9}{520}, m_{2,2} = -\frac{9}{520}, m_{0,4} = -\frac{9}{104},$$

$$\alpha_0 = \frac{877997}{865280}, \alpha_1 = -\frac{9441}{540800}, \alpha_2 = \frac{2421}{2163200}, \alpha_3 = \frac{7101}{4326400},$$

$$\beta_0 = -\frac{520}{9}, \beta_1 = \frac{520}{9},$$

$$\gamma_0 = -\frac{260}{9}, \quad \gamma_1 = \frac{529}{18}, \quad \gamma_0 = \frac{1}{2}.$$

Виникає питання, чи можливо при тій же асимптотичній точності спростити формули відновлення за рахунок скорочення числаскладових в них? Це можливо, але при

цьому носій декомпозиційного фільтру φ необхідно збільшити. Позначимо через $G(C,f,\varphi)$ метод відновлення, який визначається правилом реконструкції

$$\hat{f}_{2i,2j} = \alpha_0 c_{2i,2j} + \frac{\alpha_1}{4}\left(c_{2i-2,2j} + c_{2i+2,2j} + c_{2i,2j-2} + c_{2i,2j+2}\right) + \frac{\alpha_2}{4}\left(c_{2i-2,2j-2} + c_{2i+2,2j-2} + c_{2i-2,2j-2} + c_{2i+2,2j+2}\right),$$

а відновлення решти значень проводиться по тим же формулам, що й раніше.

Теорема 2.

Для того, щоб метод $G(C,f,\varphi)$ був такий, що

$$f_{i,j} - G(C,f,\varphi,ih,jh) = f_{i,j} - \hat{f}_{i,j} = O(h^6), \tag{5.5}$$

необхідно і достатньо, щоб центральні моменти функції φ були рівні

$$m_{0,0} = 1, m_{0,2} = -\frac{3}{2}, m_{2,2} = \frac{3}{2}, \qquad m_{0,4} = \frac{15}{2},$$

і, при цьому, коефіцієнти формул реконструкції визначаються наступним чином

$$\alpha_0 = \frac{7}{16}, \alpha_1 = \frac{3}{32}, \alpha_2 = \frac{3}{64},$$

$$\beta_0 = -\frac{1}{6}, \beta_1 = \frac{1}{12},$$

$$\gamma_0 = -\frac{1}{6}, \qquad \gamma_1 = \frac{1}{12}, \qquad \gamma_0 = \frac{1}{4}.$$

Без доведення цієї теоремі зазначимо, що у дискретному випадку функція φіснує і єдина. Значення цієї функціївизначаються матрицею

$$\begin{matrix} & & 0 & 0 & \frac{263}{640} & 0 & 0 \\ & 0 & \frac{253}{576} & -\frac{1193}{360} & \frac{253}{576} & 0 & \\ \frac{263}{640} & -\frac{1193}{360} & \frac{15631}{1440} & -\frac{1193}{360} & \frac{263}{640} & & \\ & 0 & \frac{253}{576} & -\frac{1193}{360} & \frac{253}{576} & 0 & \\ & & 0 & 0 & \frac{263}{640} & 0 & 0 \end{matrix}$$

а графік цієї функції має наступний вигляд:

Серед двох наведених методів, для практичного використання прийнятнішим є перший, через той факт, що другийдекомпозиційний фільтр φ надмірно контрастуючий.

Контрольні питання

1. Проблеми зміни геометричних розмірів зображень?
2. Які характеристики використовуються в якості визначення помилки при порівнянні двох зображень?
3. Що є тензорним добутком сплайнів?
4. Особливості перетворення Ланцоша для задачі масштабування зображень?
5. Який алгоритм масштабування є найбільш швидким?

Рекомендована література

1. http://r0k.us/graphics/kodak/TID2008 (Tampered Image Database- 2008).
2. Лигун А.А. Асимптотические методы восстановления кривых / А.А.Лигун, А.А.Шумейко .— К.: Изд. Института математики НАН Украины, 1997 .— 358 с.
3. Де Бор К. Практическое руководство по сплайнам / К.Де Бор .— М: Радио и связь, 1985 .— 303 с.
4. Завьялов Ю.С. Методы сплайн - функций / Ю.С.Завьялов, Б.И.Квасов, В.Л.Мирошниченко .— М.: Наука, 1980 .— 350 с.
5. Farin, G.E. Curves and Surfaces for Computer-Aided Geometric Design. A Practical Guide. Academic Press, 1990, 444 pp.
6. Lancaster P. Curve and Surface Fitting. An Introdution / P.Lancaster, K.Salkauskas .— Calgary: Acadamic Press, 1990 .— 280 p.
7. Гонсалес Р., Вудс Р. Цифровая обработка изображений. — М.: Техносфера, 2006. — 1072 с.
8. Лигун А.О. Комп’ютерна графіка (обробка та стиск зображень):навч.посіб./ А.О.Лигун, О.О.Шумейко.-Д.:Біла К.О., 2010.- 114 с.

ТЕМА 6. МЕТОДИ СТИСКУ ДАНИХ

Мультімедійні дані, що зберігаються на різних носіях і передаються по каналах зв'язку, як правило, мають значну надмірність. Використання алгоритмів стиску інформації дозволяє зменшити розмір файлів і, відповідно, підвищити швидкість обміну по каналах зв'язку і в стільки ж разів економити об'єм дискового простору. На сьогоднішній день існує безліч підходів до стиску інформації і алгоритмів, які їх реалізують. Перш за все, це статистичні і словарні методи. Розглянемо декілька з них.

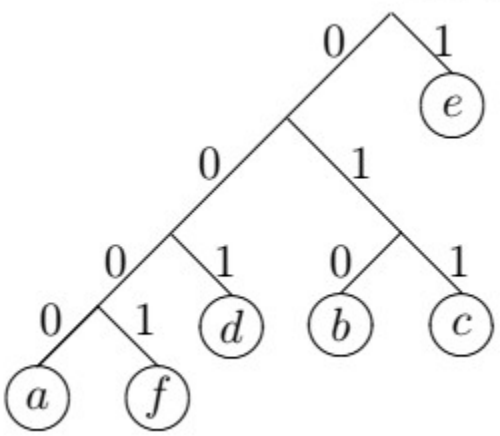
Метод Хаффмана

Метод стиску Хаффмана (Huffman, 1952) - це загальна схема стиску інформації, для дискретних даних різної природи [1]. Ідея методу полягає в заміні даних ефективнішими кодами, і складається з привласнення бінарного коду кожній унікальній величині. Коротші коди використовуються для тих величин, що зустрічаються частіше. Ці привласнення зберігаються в таблиці перекодування, яка завантажується в декодуючу програму перед самими кодами.

Наприклад, в серії ***abccddeeeeeeecbbf*** є шість унікальних величин. Частоти їх запису дорівнюють

a:1, b:3, c:3, d:2, e:7 ,f:1.

Щоб отримати мінімальний код, будемо використовувати бінарне дерево



Основний алгоритм об'єднує разом всі елементи, що з'являються як найрідше, потім пара розглядається як один елемент і їх частоти об'єднуються. Це повторюється до тих пір, поки всі елементи не об'єднаються в пари.

Для даного прикладу найрідше використовуються значення a і f, які складають першу пару - a привласнюється 0-а гілка, а f - відповідно 1 - а. Це значить, що в кодах цих символів молодшими бітами будуть відповідно 0 і 1. Старші біти отримуємо по мірі побудови дерева. Наступна частота - 2. Тому одержана пара об'єднується з символом, який має частоту 2, тобто з d. Початковій парі привласнюється 0 гілка нового дерева, а елементу d гілка 1. Таким чином, на новому етапі елемент a має код 00, а елемент f код 01. Продовжуючи побудову дерева, одержуємо, що чим частіше зустрічається елемент, тим коротше його код.

Таблиця кодів для даного випадку виглядатиме наступним чином:

e:7	0	1		
b:3	0	1	0	
c:3	0	1	1	
d:2	0	0	1	
a:1	0	0	0	0
f:1	0	0	0	1

Хоча даний приклад і ілюструє метод Хаффмана, код, одержуваний в результаті використання такого алгоритму, не є найкоротшим.

Спорідненим методом для кодування Хаффмана є кодування Шеннона-Фано (Shannon-Fano method), яке здійснюється наступним чином:

- Ділимо множину символів на дві підмножини так, щоб сума вірогідності появи символів однієї підмножини була приблизно рівна сумі вірогідності появи символів іншого. Для лівої підмножини кожному символу приписуємо "0", для правого - "1".
- Повторюємо попередній крок до тих пір, поки всі підмножини не будуть складатися з одного елементу.

Розглянемо цей підхід на нашому прикладі. Розташуємо елементи алфавіту відповідно до їх частотних характеристик

e:7, b:3, c:3, d:2, a:1, f:1.

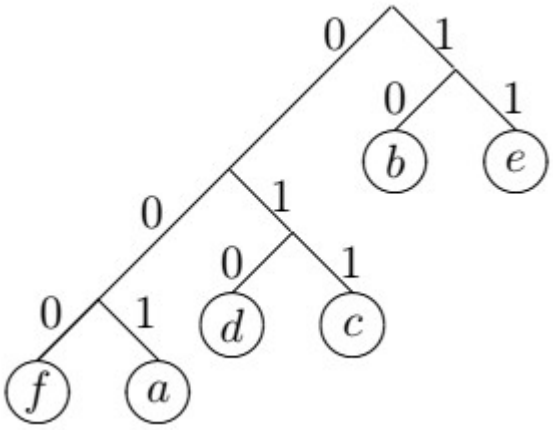
Розіб'ємо одержану таблицю на дві частини так, щоб суми частот в кожній з цих частин були приблизно рівні, і елементам, які стоять в першій таблиці привласнимо значення 1, а в другій - 0.

e:7	1
b:3	1
c:3	0
d:2	0
a:1	0
f:1	0

Далі, з кожною з одержаних таблиць проведемо таке ж перетворення, і так до тих пір, поки не одержимо набір таблиць, що складаються з одного елементу:

e:7	1	1	
b:3	1	0	
c:3	0	1	1
d:2	0	1	0
a:1	0	0	1
f:1	0	0	0

Відповідне бінарне дерево виглядає таким чином:



Код Шеннона-Фано [2] не завжди є оптимальним, кращих результатів дозволяє добитися власне метод Хаффмана. При його використанні кожен етап кодування полягає у відсіканні двох символів, що найбільш часто зустрічаються (що повністю співпадає з приведеним раніше прикладом). Алгоритм створення коду Хаффмана кодує знизу-вгору, а Шеннона-Фано -зверху-вниз.

Схема Хаффмана досягає ступеня стиску 1:8 і потребує точної статистики опису частот використання того або іншого елементу. Без точної статистики кінцевий файл не стає набагато меншим за початковий, тому для забезпечення ефективного ступеня стиску ця схема часто реалізується в два проходи. За перший прохід створюється статистична модель, за другий - кодуються дані. У результаті компресія і декомпресія по Хаффману - порівняльно повільні процеси. Крім того, оскільки біти стискаються без урахування границь байта, єдиний спосіб, за допомогою якого дешифратор може дізнатися про завершення кода - досягти кінця гілки, інакше (тобто за наявності збою) дешифратор починає з середини, і, решта даних стає безглуздою.

Приведемо ще одне зауваження: алгоритм Хаффмана не годиться для адаптивних моделей. На це є наступні причини:

По-перше, кожного разу при зміні моделі необхідно змінювати і весь набір кодів. Хоча ефективні алгоритми роблять це за рахунок невеликих додаткових витрат, їм все одно потрібне місце для розміщення дерева кодів. Якщо його використовувати в адаптованому кодуванні, то, для різної вірогідності розподілу і відповідної множини кодів, будуть потрібні свої класи умов для прогнозу символу. Оскільки моделі можуть бути досить громіздкі, то зберігання всіх дерев кодів стає надмірно дорогим. Добре наближення до кодування Хаффмана може бути досягнуте застосуванням різновиду дерев, що розширяються. При цьому, представлення дерева достатньо компактне, щоб зробити можливим його застосування в моделях, що мають декілька сотень класів умов.

По-друге, метод Хаффмана неприйнятний в адаптивному кодуванні, оскільки виражає значення -log p цілим числом бітів. Це особливо недоречно коли один символ має високу вірогідність (що і є частим випадком в складних адаптованих моделях). Як найменший код, який може бути отриманий методом Хаффмана, має 1 біт у довжину, хоча часто бажано використовувати менший. Наприклад, "o" в контексті "to be or not to be" можна закодувати в 0.014 біти. Код Хаффмана перевищує необхідний вихід в 71 разів, роблячи точний прогноз даремним.

Цю проблему можна подолати блокуванням символів, які зустрічаються досить рідко. Проте, це вносить свої проблеми, пов'язані з розширенням алфавіту (який тепер є множиною всіх можливих блоків).

Словарні методи стиску LZ

Найпоширеніші на сьогоднішній день методи стиску засновані на словарних алгоритмах.

Існують дві гілки словарних алгоритмів. Перша бере початок від алгоритму LZ77 [3] і ґрунтується на заміні рядка символів в потоці на посилання на такий же рядок, що раніше зустрівся, з довжиною замінюваного рядка. Друга походить від алгоритму LZ78 [4]. У алгоритмах цієї гілки по ходу роботи заповнюється словник. Якщо у вхідному потоці зустрівся рядок, вже присутній в словнику, він замінюється на відповідний номер елемента словника. В даний час LZ77-подібні алгоритми практично не використовуються самостійно - створена безліч вдалих комбінованих алгоритмів, де вихідний потік LZ77-подібного алгоритму стискається контекстно-вільним алгоритмом вірогідності. Як правило, це (у порядку спадання швидкодії і підвищення ступеня стиску) алгоритми Шеннона-Фоно, Хаффмана, арифметичного стиску. Саме ці комбіновані алгоритми успішно використовуються в більшості сучасних універсальних архіваторів (ARJ, PKZIP, LHA, RAR). Сімейство LZ78 складається з алгоритму LZW і його незначних модифікацій. LZC - чисто практична реалізація LZW-алгоритму, вживана в програмі COMPRESS, яка використовується в операційній системі UNIX і в графічному форматі GIF.

Своєю появою він зобов'язаний двом дослідникам з Ізраїлю: Якобу Зіву і Абраму Лемпелу (Jacob Ziv & Abraham Lempel). У 1977 році вони оприлюднили роботу [3]. Її поява ознаменувала початок нової ери в стиску інформації. Практично всі сучасні комп'ютерні програми-архіватори використовують ту, або іншу, модифікацію алгоритму LZ.

Основна ідея алгоритму LZ полягає у тому, що друге і подальші входження деякого рядка символів у вхідних даних замінюються посиланням на її першу появу в даних.

LZ77 використовує вже переглянуту частину даних в якості словника. Щоб отримати стиск, він намагається замінити черговий фрагмент даних на покажчик у зміст словника. Як модель даних LZ77 використовує "ковзаюче" по рядку даних вікно, розділене на дві нерівні частини. Перша, велика за розміром, включає вже переглянуту частину вхідних даних. Друга, набагато менша, є буфером, що містить ще не закодовані символи вхідного потоку. Звичайно розмір вікна складає декілька кілобайт. Буфер набагато менший, зазвичай не більше ніж сто байтів. Алгоритм намагається знайти у словнику фрагмент, співпадаючий зі змістом буфера.

Алгоритм LZ77 видає коди, що складаються з трьох елементів:

1. Зсув у словнику щодо його початку підрядка, співпадаючого з вмістом буфера;
2. Довжина підрядка;
3. Перший символ в буфері, наступний за підрядком.

Алгоритми групи LZ мають істотний недолік. Оскільки вони використовують в якості словника тільки невеликий фрагмент вхідних даних, то немає можливості кодувати підрядки, що повторюються, відстань між якими в повідомленні більше, ніж розмір словника. Крім того, алгоритми обмежують розмір підрядка, який можна закодувати.

Чисто механічний підхід до поліпшення характеристик кодеру LZ за рахунок збільшення розмірів словника і буфера звичайно дає результати, зворотні бажаним. Припустімо, що ми збільшимо розмір словника до 64 К і розмір буфера до 1 К. Це приведе до того, що для кодування зсуву в словнику, знадобиться 16 бітів замість 12, а для кодування довжини збігу буде потрібно 10 бітів замість 4. Таким чином кожна фраза буде закодована в 27 бітів замість 17, що, по-перше, приведе до значного

зниження ступеня стиску на коротких рядках даних, а по-друге, зробить неможливим кодувати підрядки, що повторюються та завдовжки менші 4 байтів.

Оскільки перераховані проблеми не могли не турбувати творців алгоритму Якоба Зіва і Абрама Лемпела, в 1978 році вони в роботі [4] запропонували нову версію свого алгоритму, яка далі називатиметься LZ78. LZ78 йде від ідеї ковзаючого по тексту вікна. На відміну від LZ77, в LZ78 словником є потенційно нескінченний список вже проглянутих рядків, а не підрядків, як в LZ77. У LZ78 і кодер, і декодер починають роботу з "майже порожнього" словника, що містить тільки один закодований рядок - НУЛЬ-рядок. Коли кодер прочитує черговий символ даних, символ додається до поточного рядка. Доти, доки поточний рядок відповідає якій-небудь фразі із словника, процес продовжується. Але, рано чи пізно, поточний рядок перестає відповідати якій-небудь фразі словника. У цей момент, коли поточний рядок є "останнім збігом" із словником, плюс тільки що прочитаний символ даних, кодер LZ78 видає код, що складається з індексу збігу і наступного за ним символу, що порушив збіг рядків. Окрім того, нова фраза, що складається з індексу збігу і наступного за ним символу, додається в словник. Наступного разу, коли ця фраза з'явиться в повідомленні, вона може бути використана для побудови довшої фрази, що підвищує ступінь стиску інформації.

У 1984 році Terry A. Welch опублікував роботу [5], в якій приведена модифікація алгоритму LZ78, яка одержала назву LZW (Lempel-Ziv-Welch).

Алгоритм роботи кодеру LZW можна описати таким чином:

- Провести ініціалізацію словника односимвольними фразами, відповідними символам вхідного алфавіту (звично це 256 ASCII символів);
- Прочитати перший символ даних в поточну фразу PHRASE;
- LABEL: Прочитати черговий символ SYMBOL;
- IF КІНЕЦЬ ДАНИХ - Видати код PHRASE;
- EXIT;
- END IF;
- IF фраза PHRASE+SYMBOL вже є в словнику, Замінити PHRASE на код фрази PHRASE+SYMBOL;
- GOTO LABEL;
- ELSE Видати код PHRASE;
- Додати PHRASE+SYMBOL в словник;
- GOTO LABEL;
- END IF;

Алгоритм декодування LZW може бути описаний таким чином:

- CODE = Прочитати перший код даних;
- Попередній CODE = CODE;
- Видати символ SYMBOL, у якого CODE(SYMBOL) = CODE;
- LABEL: Наступний код: CODE = Прочитати черговий код даних;
- Вхідний CODE = CODE;
- IF КІНЕЦЬ ДАНИХ EXIT;
- END IF;
- Наступний символ:
    - Якщо CODE = CODE(PHRASE+SYMBOL) Видати SYMBOL;
- CODE = CODE(PHRASE);
- GOTO LABEL;
- ELSE IF CODE = CODE(SYMBOL) Видати SYMBOL;

- Додати в словник (Попередній CODE, SYMBOL);
- Попередній CODE = Вхідний CODE;
- GOTO LABEL;
- END IF;

При декодуванні ***може виникнути виняткова ситуація***, коли кодер намагається закодувати, наприклад, рядок ABABA, де фраза AB вже присутня в словнику. Кодер виділить AB, видає код(AB) і додає ABA в словник. Потім він виділить ABA і відправить тільки що створений код (ABA). Декодер при отриманні коду (ABA) ще не додав цей код в словник, тому що поки ще не знає символ-розширення попередньої фрази. Проте, коли декодер зустрічає невідомий йому код, він може визначити, який символ видавати першим. Це символ-розширення попередньої фрази, який буде останнім символом поточної фрази, який був останнім символом попередньої фрази, що був останнім розкодованим символом. Для обробки цієї ситуації зручно скористатися стеком. Традиційно функцію поміщення в стек позначимо PUSH, а витягання зі стеку - POP.

Модифікований алгоритм декодування виглядає таким чином:

- CODE = Прочитати перший код рядка даних;
- Попередній CODE = CODE;
- Видати символ SYMBOL, у якого CODE(SYMBOL)= CODE;
- Останній SYMBOL = SYMBOL
- Наступний код: CODE = Прочитати черговий код повідомлення;
- Вхідний CODE = CODE;
- IF КІНЕЦЬ ПОВІДОМЛЕННЯ EXIT;
- END IF;
- IF Невідомий(CODE)
  Тоді обробка виняткової ситуації
- Видати(Останній SYMBOL)
  CODE = Попередній CODE
  Вхідний CODE = CODE (Попередній CODE, Останній SYMBOL)
- END IF;
- LABEL: Наступний символ:
  Якщо CODE = CODE(PHRASE+SYMBOL) PUSH(SYMBOL);
  CODE = CODE(PHRASE);
- Повторити Наступний SYMBOL;
- ELSE якщо CODE = CODE(SYMBOL) Видати SYMBOL;
- Останній SYMBOL = SYMBOL;
- IF стек не порожній POP;
- END IF;
- Додати в словник (Попередній CODE, SYMBOL);
- Попередній CODE = Вхідний CODE;
- GOTO LABEL;
- END IF;

Очевидно, що декодер LZW використовує той же словник, що і кодер, будуючи його за аналогічними правилами при відновленні стиснутих даних. Кожен прочитуваний код розбивається за допомогою словника на попередню фразу w і символ K. Потім рекурсія продовжується для попередньої фрази w до тих пір, поки вона не виявиться кодом одного символу, що і завершує декомпресію цього коду. Оновлення словника

відбувається для кожного декодованого коду, окрім першого. Після завершення декодування коду його останній символ, сполучений з попередньою фразою, додається в словник. Нова фраза одержує те ж значення коду (позицію в словнику), що привласнив їй кодер. Так, крок за кроком, декодер відновлює той словник, який побудував кодер.

Розглянемо приклад для трисимвольної абетки: A,B,C. Потік символів виглядає як ABACABABABA.

Кодування. Спочатку проведемо ініціалізацію ланцюжків: 0=A, 1=B, 2=C.

Перший символ є A, який входить в таблицю ланцюжків, отже PHRASE стає рівним A. Далі ми беремо AB, яка не входить в таблицю, отже ми виводимо код 0 (для PHRASE), і додаємо AB в таблицю ланцюжків з кодом 3. PHRASE стає рівним B. Далі ми беремо PHRASE+A = BA, яка не входить в таблицю ланцюжків, отже виводимо код 1, і додаємо BA в таблицю ланцюжків з кодом 4. PHRASE стає рівним A. Далі ми беремо AC, яка не входить в таблицю ланцюжків. Виводимо код 0, і додаємо AC в таблицю ланцюжків з кодом 5. Тепер PHRASE рівно C. Далі ми беремо PHRASE+A = CA, яка не входить в таблицю. Виводимо 2 для C, і додаємо CA до таблиці під кодом 6. Тепер PHRASE=A. Далі ми беремо AB, яка входить в таблицю ланцюжків, отже PHRASE стає рівним AB, і ми шукаємо ABA, якої немає в таблиці ланцюжків, тому ми виводимо код для AB, який дорівнює 3, і додаємо ABA в таблицю ланцюжків під кодом 7. Далі, PHRASE рівно A, ланцюжок AB вже закодований, тому PHRASE+=B, наступний ланцюжок -- ABA також закодована, отже, як PHRASE вже беремо ABA і одержуємо PHRASE+B=ABAB, яке в таблиці кодів ще не зустрічалося, тому додаємо ABAB з кодом 8, а в кодову послідовність записуємо код ABA=7. На наступному кроці PHRASE рівно BA. Ми не можемо більш узяти символів, тому ми виводимо код 4 для BA і закінчуємо процедуру кодування. Отже, потік кодів рівний 0,1,0,2,3,7,4.

Попередній код (префікс)	Наявний символ	Наявна послідовність	Додається у словник	Кодова послідовність
	A	A		
A	B	AB	3	0
B	A	BA	4	1
A	C	AC	5	0
C	A	CA	6	2
A	B	AB		
AB	A	ABA	7	3
A	B	AB		
AB	A	ABA		
ABA	B	ABAB	8	7
B	A	BA		
BA		BA		4

Декодування

Процес декодування практично повністю повторює процедуру кодування.

Також, як і при кодуванні, спочатку ініціалізуємо таблицю ланцюжків: 0=A, 1=B, 2=C.

Тепер перейдемо до процедури декодування початкової послідовності 0,1,0,2,3,7,4.

Перший код 0 відповідає символу A, другий код 1 відповідає символу B. Обидва коди входять в таблицю. Далі ми беремо ланцюжок 01, відповідно AB, яка не входить в таблицю, отже ми додаємо AB в таблицю ланцюжків з кодом 3. PHRASE стає рівним 1=B. Далі ми беремо PHRASE+0 = BA, яка не входить в таблицю ланцюжків, отже виводимо символ A, і додаємо BA в таблицю ланцюжків з кодом 4. PHRASE стає

рівним 2=A. Далі ми беремо AC, яка не входить в таблицю ланцюжків. Виводимо символ C, і додаємо AC в таблицю ланцюжків з кодом 5. Тепер PHRASE рівне 3=C. Далі ми беремо PHRASE+0 = CA, яка не входить в таблицю. Виводимо 2 C, і додаємо CA до таблиці під кодом 6. Наступний код 7.

Важливо!! Оскільки коду 7 в словнику немає, то конструюємо фрагмент для нього з поточного фрагмента+перший символ поточного фрагмента, в результаті отримаємо ABA і додаємо в таблицю ланцюжків під кодом 7.

На наступному кроці PHRASE дорівнює BA. Ми не можемо більш узяти символів, тому ми виводимо код 4 для BA і закінчуємо декодування.

Таким чином початковий потік даних є ABACABABABA.

Вхідні дані	Вихідні дані	Додається в словник
0	A	
1	B	AB(3)
0	A	BA(4)
2	C	AC(5)
3	AB	CA(6)
7	ABA	ABA(7)
4	BA	

Як вже згадувалося, Тері Велч працював над створенням програми compress разом з групою програмістів Unix. При реалізації compress він дещо удосконалив початковий алгоритм LZ78. Найцікавіше удосконалення - адаптивне скидання (обнуління) лічильника (adaptive reset). Замість того, щоб очищати словник після досягнення певного об'єму пам'яті для його зберігання (варіант, розглянутий до цього), програма compress продовжує використовувати словник до тих пір, поки залишається високим рівень стиску. Цей підхід базується на двох спостереженнях, що стосуються процесу стиску LZW. Одне з них полягає у тому, що на перших кроках розмір стиску сильно залежить від розміру словника. Адаптивне скидання намагається експлуатувати великий словник якомога довше. Інше спостереження полягає у тому, що багато файлів (особливо архіви TAR (архіви системи UNIX), що містять різні види файлів усередині архіву) містять області з різними типами даних. В процесі роботи алгоритму LZW формується словник, спеціально пристосований для певного типу даних. При значній зміні типу даних словник вже не даватиме високого ступеня стиску. Контролюючи ефективність стиску, програма compress може скидати словник у момент падіння продуктивності.

При стиску даних (надходженні на вхід програми чергової порції) програма на основі LZW намагається знайти в словнику фрагмент максимальної довжини, співпадаючий з даними, замінює знайдену в словнику порцію даних кодом фрагмента і доповнює словник новим фрагментом. При заповненні всього словника (розмір словника обмежений за визначенням) програма очищає словник і починає процес заповнення словника знову. Реалізації цього методу розрізняються конструкцією словника, алгоритмами його заповнення і очищення при переповненні. Звичайно, при ініціалізації словник заповнюється початковими (елементарними) фрагментами -- всіма можливими значеннями байта від 0 до 255. Це гарантує, що під час попадання на вхід чергової порції даних буде знайдений в словнику хоча б однобайтовий фрагмент. Алгоритм LZW резервує спеціальний код, який вставляється в упаковані дані, відзначаючи момент скидання словника. Значення цього коду звичайно приймають рівним 256. Таким чином при початку кодування мінімальна довжина

коду складає 9 біт. Максимальна довжина коду залежить від об'єму словника -- кількості різних фрагментів, які туди можна помістити. Якщо об'єм словника вимірювати в байтах (символах), то очевидно, що максимальна кількість фрагментів рівна числу символів, а, отже, максимальна довжина коду дорівнює $\log_2 W$ (W -- об'єм словника в байтах).

Арифметичний стиск

Арифметичний стиск, як і алгоритм Хаффмана використовує більш короткі коди для елементів, що часто з'являються, і довші для тих, що рідко з'являються, хоча подібно алгоритму LZW стискає послідовності величин, а не самі величини. Арифметичний стиск достатньо близько лежить біля теоретичної межі стиску. Арифметичний стиск включає відображення кожної відмінної послідовності величин в діапазон цифр між 0 і 1. Потім ця область представляється як двійковий дріб змінної точності, при цьому менш загальні послідовності вимагають вищої точності (більше біт).

Розглянемо приклад. Нехай маємо набір величин з частотою їх появи

a:100, b:300, c:300, d:200, e:900, f:100.

Таким чином вірогідність появи кожної з цих величин з точністю до четвертого знаку

P(a) =0.0526, P(b) =0.1579, P(c) =0.1579,
P(d) =0.1052, P(e) =0.4737, P(f) =0.0526.

Серія елементів ***ab*** матиме вірогідність ***P(a)×P(b),*** проте якщо поставити кожній серії у відповідність її

вірогідність, це не забезпечить її унікальність, в даному випадку таку ж вірогідність мають пари ***bf***.

Представимо величину ***b*** сегментом рядка від 0.0526 до 0.2105. Довжина цього сегменту є вірогідність ***P(b)*** появи елементу ***b***. Величина с має ту ж вірогідність, але розташована в іншому діапазоні: від 0.2105 до 0.3684. Тепер можна однозначно представити елемент числом, що вказує на сегмент цього елементу. Так число 0.25 указує на елемент с. Для представлення послідовностей з двох елементів можна розділити кожний з цих сегментів. Наприклад, послідовність ***ba*** (з вірогідністю 0.0526) буде представлена першими 5.26% цієї області, величина ***bb*** (з вірогідністю 0.1579) наступними 15.79% і так далі. Тоді елемент ***ba*** можна представити будь-яким числом в діапазоні від 0.0526 до 0.0554.

Таким чином, ми одержуємо схему однозначного кодування будь-якої послідовності величин. При цьому потрібна велика точність (більше біт) для маловірогідних величин. Такі послідовності представляють невеликий сегмент і навпаки, потрібно менше біт для послідовностей з високою вірогідністю. Чим більше сегмент, тим більша ймовірність знайти дріб низької точності, наприклад, 1/2, 1/4 або 1/8 для ідентифікації елементів цього сегменту.

Так само як і алгоритм Хаффмана, арифметичне кодування ефективне для тих послідовностей елементів, що часто повторюються. Ефективність алгоритму арифметичного стиску може досягати 1:100.

Найважливішими властивостями арифметичного кодування є наступні:

1. Здатність кодування символу вірогідності p кількістю бітів досить близькою до $\log_2 p$;
2. Вірогідність символів може бути на кожному кроці різною;
3. Дуже незначні обсяги пам'яті незалежно від кількості класів умов в моделі;
4. Велика швидкість роботи.

У арифметичному кодуванні символ може відповідати дробовій кількості вихідних бітів. На практиці результат повинен, звичайно, бути цілим числом бітів, що можливо, якщо декілька послідовних високо вірогідних символів кодувати разом, поки у вихідний потік не можна буде додати 1 біт. Кожен закодований символ вимагає тільки одного цілочисельного множення і декількох додавань, для чого звичайно використовується тільки три 16-бітових внутрішніх регістра. Тому, арифметичне кодування ідеально підходить для адаптивних моделей, і його відкриття породило безліч модифікацій, що набагато перевершує ті, що застосовуються разом з кодуванням Хаффмана.

Складність арифметичного кодування полягає у тому, що воно працює з накопичуваною вірогідністю розподілу, яка вимагає внесення для символів деякої впорядкованості. Відповідна символу накопичувана вірогідність є сумою вірогідності всіх символів, передуючих йому.

Скалярне квантування

Розглядаючи задачі стиску зображень, не можемо не розглянути алгоритми квантування. Як правило, під квантуванням розуміють заміну неперервних даних системою індексів. В разі, коли дані описуються лише своїми значеннями, використовується скалярне квантування. Якщо під час квантування береться до уваги не тільки значення тих чи інших інформаційних характеристик, але і їхнє геометричне положення, то використовується векторне квантування.

Розглянемо деякі питання скалярного квантування.

Основні визначення

Хай $f=\{f_i\}_{i=0}^{N}$ масив вхідних даних. Надалі вважатимемо, що $\varphi=\{\varphi_i\}_{i=0}^{N}$ набір коефіцієнтів початкових даних, розгорнених у одновимірний масив. При цьому виходитимемо з того факту, що в розкладанні використовувалися ортогональні або близькі до ортогональних фільтри. В якості таких фільтрів можуть використовуватися фільтри на основі сплесків, наведених раніше, або перетворення Фур'є.

Для будь-якого масиву $\widetilde{\varphi}=\{\widetilde{\varphi}_i\}_{i=0}^{N}$ через $\widetilde{f}=\{\widetilde{f}_i\}_{i=0}^{N}$ позначимо результат застосування зворотного перетворення до $\widetilde{\varphi}$.

Через ортогональності фільтру має місце рівність Парсеваля

$$\sum_{i=0}^{N} f_i^2=\sum_{i=0}^{N}\varphi_i^2,\ \sum_{i=0}^{N}\widetilde{f}_i^2=\sum_{i=0}^{N}\widetilde{\varphi}_i^2$$

і, через лінійність процесу фільтрації і рівності Парсеваля маємо

$$\sum_{i=0}^{N}(f_i-\widetilde{f}_i)^2=\sum_{i=0}^{N}(\varphi_i-\widetilde{\varphi}_i)^2\ . \qquad (6.1)$$

У подальшому будемо вважати

$$A^-=\min_i\varphi_i,\ A^+=\max_i\varphi_i,$$

і нехай, крім того, $B=B_{n+m+1}$ розбиття відрізка $\lfloor A^-,A^+\rfloor$ на $n+m+1$ проміжків точками

$$A^-=b_{-m-1/2}<b_{-m+1/2}<\ldots<b_{-1/2}<0<b_{1/2}<\ldots<b_{n+1/2}=A^+.$$

Для кожного індексу $i=0,1,2,\ldots,N$ знайдеться індекс $k=k(i)$ $(k=-m,\ldots,n)$ такий, що

$$b_{k-1/2}<\varphi_i\le b_{k+1/2}\quad \text{якщо}\ \ k=1,\ldots,n,$$

або

$$b_{k-1/2}\le\varphi_i<b_{k+1/2}\ \text{якщо}\ \ k=-1,\ldots,-m,$$

або

$$b_{-1/2} \le \varphi_i \le b_{+1/2} \quad \text{якщо } k=0.$$

Легко бачити, що для будь-яких $\{a_i\}_{i=1}^{M}$ задача

$$\sum_{i=1}^{M} (a_i - c)^2 \to min$$

має єдиний розв'язок

$$c = \frac{1}{M} \sum_{i=1}^{M} a_i .$$

Таким чином якщо для фіксованого $k=-m,...,n$ всі значення φ_i, що знаходяться в одному з інтервалів, замінені на одне і те ж число c_k, то найменша похибка в середньоквадратичній метриці буде при $c_k = b_k$, де

$$b_k = \frac{s_k}{n_k}, k = -m,...,n, \; s_k = \sum \{\varphi_i | i : k(i) = k\}, n_k = \sum \{1 | i : k(i) = k\}.$$

Тому послідовність $g_i = b_{(i)} (i = 0,1,2,..., N)$ буде як найкращою (у значенні середньоквадратичного наближення) зі всіх послідовностей постійних на кожному з проміжків.

Для фіксованих чисел $b_k (k = -m,..., n)$ (таблиця квантування) для кожної послідовності $g_i (i = 0,1,2,..., N)$ однозначно ставиться у відповідність послідовність $\psi_i = k(i)$.

Іноді замість цієї послідовності зручніше використовувати наступну послідовність

$$\theta_i = \begin{cases} 2k(i) - 1, & k(i) > 0, \\ -2k(i), & k(i) < 0, \\ 0, & k(i) = 0. \end{cases}$$

Таким чином, при фіксованому векторі B кожне значення φ_i округлено до значення g_i (значення Ψ_i
або θ_i називатимемо кодом масиву φ). Надалі цю процедуру називатимемо квантуванням послідовності φ по вектору B.

З (2.1) витікає, що похибка RMSE (Root Mean Square Error) виникла в результаті квантування

$$RMSE(B, f) = \|f - \tilde{f}\|_{\ell 2}$$

де $\tilde{f}$ результат застосування зворотного ортогонального (чи біортогонального) перетворення до масиву g, дорівнює

$$RMSE(B, f) = \|f - \tilde{f}\|_{\ell 2} = \|\varphi - g\|_{\ell 2} = \sqrt{\frac{1}{N+1} \sum_{i=0}^{N} (\varphi_i - g_i)^2} .$$

Основною характеристикою якості відновлення є величина

$$p = PSNR(B, \varphi) = 12 \log_{10} \frac{\max_i |\varphi_i|}{RMSE(B, \varphi)},$$

PSNR (Peak Signal-to-Noise Ratio) -- відношення сигнал/шум (у dB).

Процес квантування можна переписати в іншій термінології.

Покладемо

$$\Phi^{+}_{k+1/2} = \{\varphi_i : \varphi_i \le b_{k+1/2}\}, \Phi^{-}_{k+1/2} = \{\varphi_i : \varphi_i \ge b_{k+1/2}\},$$

Крім того, нехай для $k=0,1,2,...,n$

$$\Delta\Phi_k = \Phi^{+}_{k+1/2} \setminus \Phi^{+}_{k-1/2},$$

а для $k=-m,..., -1$

$$\Delta\Phi_k = \Phi^{-}_{k-1/2} \setminus \Phi^{-}_{k+1/2}, \quad \text{і} \quad \Delta\Phi_0 = \Phi^{+}_{1/2} \cap \Phi^{-}_{-1/2}.$$

Тут $A\backslash B$ є теоретико-множинна різниця множин A і B.
Далі нехай n_k число елементів множини $\Delta\Phi_k$. Покладемо

$$b_k = \frac{1}{n_k} \sum \left\{ \varphi_i \middle| i : \varphi_i \in \Delta\Phi_k \right\},$$

тоді RMSE можна переписати у вигляді

$$RMSE(B, f) = \sqrt{\frac{1}{N+1} \sum_{k=m}^{n} \sum \left\{ (\varphi_i - b_k)^2 \middle| i : \varphi_i \in \Delta\Phi_k \right\}}.$$

Цей процес легко алгоритмізувати, наприклад, для $k=1,\ldots,n$ псевдокод виглядатиме таким чином

```
i=0;
 for k=1 to n do
  begin
   s[k]=0; n[k]=0;
   if (fi[i]>b1[k])&(fi[i]<=b1[k+1]) then
     begin
      s[k]=s[k]+fi[i];
      n[k]=n[k]+1;
      psi[i]=k;
      i=i+1;
     else
      b[k]=s[k]/n[k];
   end if;
  end do.
```

Далі вважатимемо, що задані метод лінійного кодування послідовностей, що повторюються (RLE) і метод ентропійного кодування A. В якості ентропійного кодування може бути взятий метод Хаффмана, LZW, арифметичного кодування та ін.
Нехай $V_0(\varphi)$ розмір (у байтах) початкового масиву $\{\varphi_i\}$, і $V_1(\varphi)$ розмір вихідного масиву, тобто розмір вихідних даних одержаних після застосування RLE і ентропійного кодування до послідовності θ_i або Ψ_i
Величину

$$c(B) = c(B, \varphi, RLE, A) = \frac{V_0(\varphi)}{V_1(\varphi)}$$

назвемо ступенем стиску даних (compression ratio або CR).
Таким чином, для фіксованого масиву f (а отже, і масиву Φ) кожен вектор B однозначно визначає точку на площині з координатами $(p(B,\varphi), c(B, \varphi))$, де $p(B,\varphi)$-характеристика якості файлу після стиску. Ця точка характеризує якість квантування - чим вона вища, тим квантуватель краще. Точне формулювання цього факту ми наведемо нижче.
Під правилом квантування $\aleph$ будемо розумітимемо процедуру побудови набору векторів {B}.
Умовно розділятимемо цю процедуру на дві частини

- вибір нульового інтервалу квантування $\left[b_{-1/2}, b_{1/2}\right]$,
- по вибраному нульовому інтервалу квантування побудова решти інтервалів $\left[b_{k-1/2}, b_{k+1/2}\right], |k| > 0$.

Процедура квантування полягає в наступному. Виберемо деяке число $\delta>0$ і вважаймо $b_{1/2} = -b_{-1/2} = \delta$.

Кількість інтервалів квантування n і m виберемо таким чином

$$n = \lfloor (A^+ - b_{1/2})\delta \rfloor, m = \lfloor (b_{-1/2} - A^-)\delta \rfloor,$$

де [•] ціла частина числа, і обчислимо межі інтервалів квантування

$$b_{k+1/2} = b_{1/2} + \frac{k}{n}(A^+ - b_{1/2}), k = 1,...,n,\ b_{k-1/2} = b_{-1/2} - \frac{k}{n}(A^- - b_{-1/2}), k = -1,...,-m,$$

А також таблицю квантування $b_k = \frac{1}{n_k}\sum\{\varphi_i \mid i \in I_k\}$, де $I_k = \{i : b_{i-1/2} \le \varphi_i < b_{i+1/2}\}$ і n_k число елементів множини I_k. Ця процедура дає набір B і називається рівномірним квантуванням. Міняючи δ, одержуємо набір векторів B квантування, тобто одержуємо метод квантування.

Надалі вважатимемо, що фільтри, RLE, A і φ фіксовані. Метод квантування (вектор $\hat{B} = \hat{B}(p)$ називатимемо якнайкращим на класі W для заданої якості p_0, якщо $p(\hat{B}, p_0) \ge p_0$, і для будь-якого іншого вектору B∈ W такого, що $p(B, p_0) \ge p_0$, справедлива нерівність $p(\hat{B}, p_0) \ge p(B, p_0)$.

Вибір нульового інтервалу квантування

Розглянемо декілька підходів до рішення першої задачі - побудові нульового інтервалу квантування $\lfloor b_{-1/2}, b_{1/2} \rfloor$.

Позначимо через $\{\phi_i\}_{i=0}^N$ незростаючу перестановку послідовності $\{\varphi_i\}_{i=0}^N$, а через $\{\phi_i^+\}_{i=0}^N$ позначимо незростаючу перестановку множини $\{|\varphi_i|\}_{i=0}^N$. При заданій якості p_0 (PSNR) вважатимемо, що нульовий інтервал квантування дає, наприклад, 80% (це значення параметра підібране статистично) всієї похибки. В цьому випадку число n_0 виберемо з умови

$$0.80(\phi_0^+ 10^{-p0/20})^2 \le \frac{1}{N - n_0}\sum_{i=0}^{N-n0}\phi_i^+ \le 0.81(\phi_0^+ 10^{-p0/20})^2.$$

Розглянемо інший підхід до вирішення цієї задачі.

Нехай i_{-1} і i_1 є розв'язок екстремальної задачі

$$\begin{cases} \sum_{i=i_{-1}}^{i_1}\phi_i \to min, \\ i_1 - i_{-1} = n_0. \end{cases}$$

Тоді вважаємо $b_{1/2} = \phi_{i_1}$ і $b_{-1/2} = \phi_{i_{-1}}$. Для дослідження якнайкращих методів квантування зручно використовувати саме цей підхід. Міняючи n_0 і підбираючи $b_{\pm 1/2}$ можна одержати якнайкращий метод квантування в деякому діапазоні зміни p.

Нульовий інтервал квантування можна також визначити виходячи з вирішення наступної екстремальної задачі.

Нехай числа $\delta_{-1}, \delta_1 > 0, \delta_{-1} + \delta_1 = \delta$. Через I_0 позначимо множину, що складається з n_0 індексів таких, що $-\delta_{-1} \le \varphi_i \le \delta_1$, тобто $I_0 = \{i : -\delta_{-1} \le \varphi_i \le \delta_1\}$ і покладемо

$$\varepsilon(\delta_{-1}, \delta_1) = \sum_{i \in I_0}\left(\varphi_i - \frac{1}{n_0}\sum_{i \in I_0}\varphi_i\right)^2.$$

Для будь-якого фіксованого ε>0 через n_0 позначимо розв'язок екстремальної задачі $n_0 \to max$ при умові $\varepsilon(\delta_{-1}, \delta_1) < \varepsilon$. Кожний з розглянутих підходів має свої плюси і мінуси. Зокрема, перший підхід використовує незростаючу перестановку модулів коефіцієнтів, другий, перестановку самих коефіцієнтів, для третього підходу

перестановка не потрібна, проте обчислювальна складність цього алгоритму не менше. Крім того, в цьому випадку потрібно достатньо точно запам'ятовувати величину

$\frac{1}{n_0}\sum_{i\in I_0}\varphi_i$ до якої квантуються значення в цьому інтервалі.

Перейдемо до розгляду другої задачі. При її вирішенні виходитимемо з того факту, що числа $b_{-1/2}$ і $b_{1/2}$ вже вибрані.

Вибір інтервалів квантування.

Раніше було розглянуте рівномірне квантування. Опишемо ще декілька методів квантування.

Вважатимемо, що задані межі інтервалу $\left[b_{-1/2}, b_{1/2}\right] = \left[\varphi_{i_{-1}}, \varphi_{i_1}\right]$. Обчислимо $i_2 = \left[\frac{i_1}{2}\right]$, де $[\bullet]$ ціла частина числа. Нехай α деякий параметр, на практиці його доцільно брати $\alpha \in \left[\frac{1}{2}, \frac{2}{3}\right]$.

Якщо

$$\frac{1}{i_2^{\alpha}}\sum_{i=0}^{i_2}\left(\varphi_i - \frac{1}{i_2}\sum_{k=0}^{i_2}\varphi_i\right)^2 > \frac{1}{(i_1 - i_2)^{\alpha}}\sum_{i=i_2}^{i_1}\left(\varphi_i - \frac{1}{i_1 - i_2}\sum_{k=i_2}^{i_1}\varphi_i\right)^2$$

то покладемо $i_3 = \left[\frac{i_2}{2}\right]$, інакше $i_3 = i_2 + \left[\frac{i_1 - i_2}{2}\right]$, тобто проміжок з найбільшим значенням RMSE розбиваємо на дві рівні частини.

Далі цей процес продовжуємо аналогічним чином: вибираємо проміжок з найбільшим значенням RMSE і розбиваємо його на дві рівні частини. Цей процес продовжуємо або задане число раз, або, використовуючи деякий критерій, наприклад, поки не одержимо необхідну величину похибки квантування.

Цей метод квантування можна модернізувати, вибираючи на кожному кроці i_k так, щоб сума значень RMSE на даному проміжку була мінімальною.

Далі розглянемо один метод квантування, який спирається на наступну лему.

Лема 1. Нехай ϕ(x) неперервна незростаюча функція, що задана на проміжку x∈ [0,T]. Задача

$$F(B) = \sum_{k=m}^{n}\left(\int_{x_{k-1/2}}^{x_{k+1/2}}\left(\phi(x) - \frac{1}{x_{k+1/2} - x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz\right)^2 dx\right) \to min$$

за умовою

$$0 = x_{1/2} < \ldots < x_{n+1/2} = T \tag{6.2}$$

має єдиний розв'язок $\hat{x}_{k+1/2}$, який визначається співвідношенням

$$\phi(\hat{x}_{k+1/2}) = \frac{1}{2}\left(\frac{1}{x_{k+1/2} - x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz + \frac{1}{x_{k+3/2} - x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right), k = 1,\ldots,n-1.$$

Для вирішення цієї екстремальної задачі використовуємо традиційні методи математичного аналізу.

Дійсно, для *k=1,...,n-1*

$$\frac{\partial F}{\partial x_{k+1/2}} = 2\int_{x_{k-1/2}}^{x_{k+1/2}}\left(\phi(x) - \frac{1}{x_{k+1/2} - x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz\right)\left(\frac{1}{(x_{k+1/2} - x_{k-1/2})^2}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz - \frac{\phi(x_{k+1/2})}{x_{k+1/2} - x_{k-1/2}}\right)dx +$$

$$+\left(\phi(x_{k+1/2})-\frac{1}{x_{k+1/2}-x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz\right)^2+$$

$$+2\int_{x_{k-1/2}}^{x_{k+1/2}}\left(\phi(x)-\frac{1}{x_{k+3/2}-x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right)\left(-\frac{1}{(x_{k+3/2}-x_{k+1/2})^2}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz+\frac{\phi(x_{k+1/2})}{x_{k+3/2}-x_{k+1/2}}\right)dx-$$

$$-\left(\phi(x_{k+1/2})-\frac{1}{x_{k+3/2}-x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right)^2=$$

$$=\left(\frac{1}{x_{k+1/2}-x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz+\frac{1}{x_{k+3/2}-x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right)\times$$

$$\times\left(2\phi(x_{k+1/2})-\frac{1}{x_{k+1/2}-x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz-\frac{1}{x_{k+3/2}-x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right).$$

Звідси прирівнюючи похідну нулю, відразу одержуємо

$$b_{k+1/2}=\phi(x_{k+1/2})=\frac{1}{2}\left(\frac{1}{x_{k+1/2}-x_{k-1/2}}\int_{x_{k-1/2}}^{x_{k+1/2}}\phi(z)dz+\frac{1}{x_{k+3/2}-x_{k+1/2}}\int_{x_{k+1/2}}^{x_{k+3/2}}\phi(z)dz\right),k=1,...,n-1.$$

Наведений доказ не є строгим - необхідно показати, що при цьому досягається мінімум, і що одержана умова є не тільки необхідною, але і достатньою. Для цього потрібно детально розглянути випадок "злиплих" вузлів (коли деякі з нерівностей (6.2) обертаються в рівність). Строгий доказ цього факту є громіздким і ми його наводити не будемо.

У дещо іншому вигляді цей результат приведений в роботах Ллойда і Макса (див. [6-7]).

Переписуючи затвердження леми в термінах приведених вище, приходимо до висновку, що при фіксованих $b_{1/2}$, $b_{-1/2}$ і числах n і m вектор B буде оптимальним по RMSE, тоді і тільки тоді, коли виконується рівність

$$b_{k+1}=2b_{k+1/2}-b_k,k=-m,...,n,k\neq 0,$$

або, що те ж,

$$b_{k+1}=b_{k+1/2}+(b_{k+1/2}-b_k),k=1,...,n,\ \text{і}\ b_{k-1}=b_{k-1/2}+(b_{k-1/2}-b_k),k=-m,...,-1. \tag{6.3}$$

Останні дві рівності можна використовувати як у разі, коли задане число інтервалів квантування (тобто

задані числа n,m) так і вибирати b_k згідно цих рекурентних співвідношень, так і при заданих межах нульового інтервалу квантування $b_{1/2}$, $b_{-1/2}$ і значеннях b_1, b_{-1} В цьому випадку, легко знайти решту інтервалів $[b_{k-1/2},b_{k+1/2}]$, і покласти

$$[b_{n-1/2},b_{n+1/2}]=[b_{n-1/2},A^+],\ [b_{m-1/2},b_{m+1/2}]=[A^-,b_{m+1/2}].$$

При цьому однозначно визначаються безперервні зліва функції $n(\delta)$ і $m(\delta)$. Для лівих меж в точках розривів цих функцій рівності (6.3) виконуються для всіх інтервалів $[b_{k-1/2},b_{k+1/2}]$ $(k=-m,...,n,\ k\neq 0)$.

Модифікуємо цей алгоритм, розглянувши замість співвідношень (6.3) рівності

$$b_{k+1}=b_{k+1/2}+\lambda_{k+1}(b_{k+1/2}-b_k),k=1,...,n,\ \text{і}\ b_{k-1}=b_{k-1/2}+\lambda_{k-1}(b_{k-1/2}-b_k),k=-m,...,-1. \tag{6.4}$$

де числа λ_k вибираються із статистичних даних, які є в наявності, або інших міркувань, про що буде сказано нижче.

Ясно, що при $\lambda_k = 1$ рівності (6.4) переходять в рівності (6.3).

Нехай, як і раніше, $\{\phi_i\}_{i=0}^{N}$ незростаюча перестановка скінченної послідовності $\{\varphi_i\}_{i=0}^{N}$, а також задані межі нульового інтервалу $b_{1/2} = \phi_{i_1}, b_{-1/2} = \phi_{i_{-1}}$ і $b_0 = 0$. Зафіксуємо числа λ_k. Наприклад, $\lambda_{-1} = \lambda_1 = 0.5$ і $\lambda_k = 1$ для |k|>1. Нехай, спочатку, k>0. Із (6.4), при k=1 (при цьому визначено λ_1), знаходимо b_1. Цьому числу відповідає індекс i_1, такий, що $b_1 = \varphi_{i_1}$. Для $i=i_1-1$ виконується умова

$$\sum_{\nu=i}^{i_1-1}(\phi_\nu - b_1) \le 0.$$

Послідовно зменшуючи індекс i на одиницю, однозначно знайдемо такий індекс i_2, що або

$$\sum_{\nu=i_2}^{i_1-1}(\phi_\nu - b_1) \le 0 \text{ і } \sum_{\nu=i_2-1}^{i_1-1}(\phi_\nu - b_1) > 0$$, або досягається значення $i=0$.

У першому випадку покладемо

$$b_{3/2} = \varphi_{i_2} \text{ і } b_2 = b_{3/2} + \lambda_2(b_{3/2} - b_1)$$

у другому покладемо $n=2$ і обчислимо $b_n = \frac{1}{i_n}\sum_{i=0}^{i_n-1}\phi_i, b_{n+1/2} = A^+$.

Нехай вже обчислені $b_{1/2}, b_1, \ldots, b_k$, тоді однозначно знайдеться такий індекс i_{k+1}, що або

$$\sum_{\nu=i_{k+1}}^{i_k-1}(\phi_\nu - b_k) \le 0 \text{ і } \sum_{\nu=i_{k+1}-1}^{i_k-1}(\phi_\nu - b_k) > 0$$, або досягається значення $i=0$.

У першому випадку покладемо

$$b_{k+1/2} = \varphi_{i_{k+1}} \text{ і } b_{k+1} = b_{k+1/2} + \lambda_{k+1}(b_{k+1/2} - b_k)$$

у другому покладемо $n=k+1$ і обчислимо $b_n = \frac{1}{i_n}\sum_{i=0}^{i_n-1}\phi_i, b_{n+1/2} = A^+$.

Обчислення закінчуються, коли буде досягнуто значення i=0.

Псевдокод цієї процедури виглядає наступним чином

```
j=i[1];s=0; k=1; m=0;
 while j>0 do
   begin
     s=s+(fi[j]-b[k]);
     m=m+1;
     if s*(s+(fi[j-1]-b[k]))<=0
       then j=j-1
       else
        begin
          k=k+1;
          i[k]=j;
          b[k]=fi[j]+lambda[k]*(fi[j]-b[k-1]);
          s=0;
          m=0;
        end;
   end if;
   n=k;
   b[n]=s/m;
```

end do.

По аналогії з попередніми викладками, для $k<0$ одержуємо індекс i_{k-1} з умов

$$\sum_{\nu=i_k+1}^{i_{k-1}}(\phi_\nu - b_k) \le 0 \text{ і } \sum_{\nu=i_k+1}^{i_{k-1}+1}(\phi_\nu - b_k) > 0$$, поки $i_{k-1}+1<N$, інакше

$$b_k = \frac{1}{N - i_k - 1}\sum_{i=i_k+1}^{N}\phi_i, b_{k-1/2} = A^-.$$

Псевдокод виглядає аналогічно.

Після того, як будуть визначені всі інтервали квантування $[b_{k-1/2},b_{k+1/2}]$ потрібно перерахувати всі b_k, вважаючи

$$b_k = \frac{1}{i_k - i_{k+1} - 1}\sum_{i=i_{k+1}}^{i_k-1}\phi_i.$$

Таким чином, якщо числа λ_k фіксовані, і вибирати $b_{1/2}$, $b_{-1/2}$, наприклад, по числу n_0 елементів, квантованих у нульовому інтервалі, то однозначно одержуємо набір інтервалів квантування $\left[b_{k-1/2}, b_{k+1/2}\right]$, тобто вектор B.

Підведемо риску під даною темою.

Алгоритм оптимального квантування

Як вже зазначалося раніше, у багатьох задачах цифрової обробки сигналів (і зображень в тому числі) неперервний динамічний діапазон значень ділиться на ряд дискретних рівнів. Ця процедура називається квантуванням. Квантуватель перетворює неперервну змінну x (t) у дискретну змінну $\hat{x}_k$, яка приймає скінченне число значень, і є елементам кодової таблиці квантування. Ці значення називаються рівнями квантування. Значення змінної x (t), що відповідають одному рівню квантування $\hat{x}_k$ описуються індексом k .

Нехай x (t) змінна з областю значень X , і $\{X_k\}_{k=0}^{N-1}$ набір множин таких, що

$$X_k \cap X_\nu = \varnothing, (k \ne \nu), \bigcup_{k=0}^{N-1} X_k = X.$$

Кожному значенню $x(t) \in X_k$ поставимо у відповідність (квантовочне) число $\mu(k)$, яке є унікальним для кожного X_k (наприклад, $\mu(k) = k$). Цю процедуру будемо називати квантуванням змінної $x(t)$ на N рівнів.

Заміна квантовочного числа $\mu(k)$ значенням, що описує елементи класу X_k називається деквантуванням. Цим значенням може бути медіана, середнє арифметичне елементів класу X_k або будь-яке інше значення, яке описує елементи цього класу.

Перепишемо в іншому вигляді результат Ллойда і Макса, наведений раніше.

Нехай набір $\{x_i\}_{i=0}^{n}$ такий, що $\min\limits_{i=0,\dots,n} x_i = 0, \max\limits_{i=0,\dots,n} x_i = M > 0.$

Через $\{\xi_k\}_{k=0}^{N}$ позначимо набір такий, що

$$0 = \xi_0 < \xi_1 < \dots < \xi_{N-1} < \xi_N = M \text{ і } X_{k+1/2} = \{x_i : \xi_k \le x_i \le \xi_{k+1}\}.$$

Через

$$m_{k+1/2} = m(M_{k+1/2}) = \frac{\sum\{x_i | i : x_i \in X_{k+1/2}\}}{\sum\{1 | i : x_i \in X_{k+1/2}\}}$$

позначимо математичне очікування всіх елементів x_i з множини (класу) $X_{k+1/2}$.

Доведено, що існує єдиний набір $\{\hat{\xi}_k\}_{k=0}^{N}$ такий, що

$$\hat{m}_{k+1/2} = \frac{1}{2}(\hat{\xi}_k + \hat{\xi}_{k+1})$$

причому для будь-яких класів $\left\{X_{k+1/2}\right\}_{k=0}^{N-1}$ і будь-яких чисел $\left\{m_{k+1/2}\right\}_{k=0}^{N-1}$ має місце нерівність

$$\sum_{k=0}^{N-1}\sum\left\{(x_i - \hat{m}_{k+1/2})^2 \middle| i : x_i \in \hat{X}_{k+1/2}\right\} \le \sum_{k=0}^{N-1}\sum\left\{(x_i - m_{k+1/2})^2 \middle| i : x_i \in X_{k+1/2}\right\}.$$

Більш того, якщо $X_{k+1/2} \neq \hat{X}_{k+1/2}$ або $m_{k+1/2} \neq \hat{m}_{k+1/2}$, то нерівність строга.

Наведемо ітераційний алгоритм вирішення цієї задачі.

Для будь-якого набору $\left\{\xi_k^0\right\}_{k=0}^{N}$ такого, що $0 = \xi_0^0 < \xi_1^0 < \ldots < \xi_{N-1}^0 < \xi_N^0 = M$ покладемо $v = 0$ і

$$X_{k+1/2}^v = \left\{x_i : \xi_k^v \le x_i \le \xi_{k+1}^v\right\}$$

і

$$m_{k+1/2}^v = m(M_{k+1/2}) = \frac{\sum\left\{x_i \middle| i : x_i \in X_{k+1/2}^v\right\}}{\sum\left\{1 \middle| i : x_i \in X_{k+1/2}^v\right\}}.$$

Знайдемо,

$$\xi_л^{м+1} = \frac{1}{2}\left(m_{k-1/2}^v + m_{k+1/2}^v\right) \text{ і } v := v+1.$$

Далі ітераційно повторюємо цей процес.

Якщо

$$\Delta^м = \sum_{k=0}^{N-1}\sum\left\{(x_i - m_{k+1/2}^v)^2 \middle| i : x_i \in X_{k+1/2}^v\right\}.$$

то доведено, що $\Delta^{v+1} < \Delta^v$.

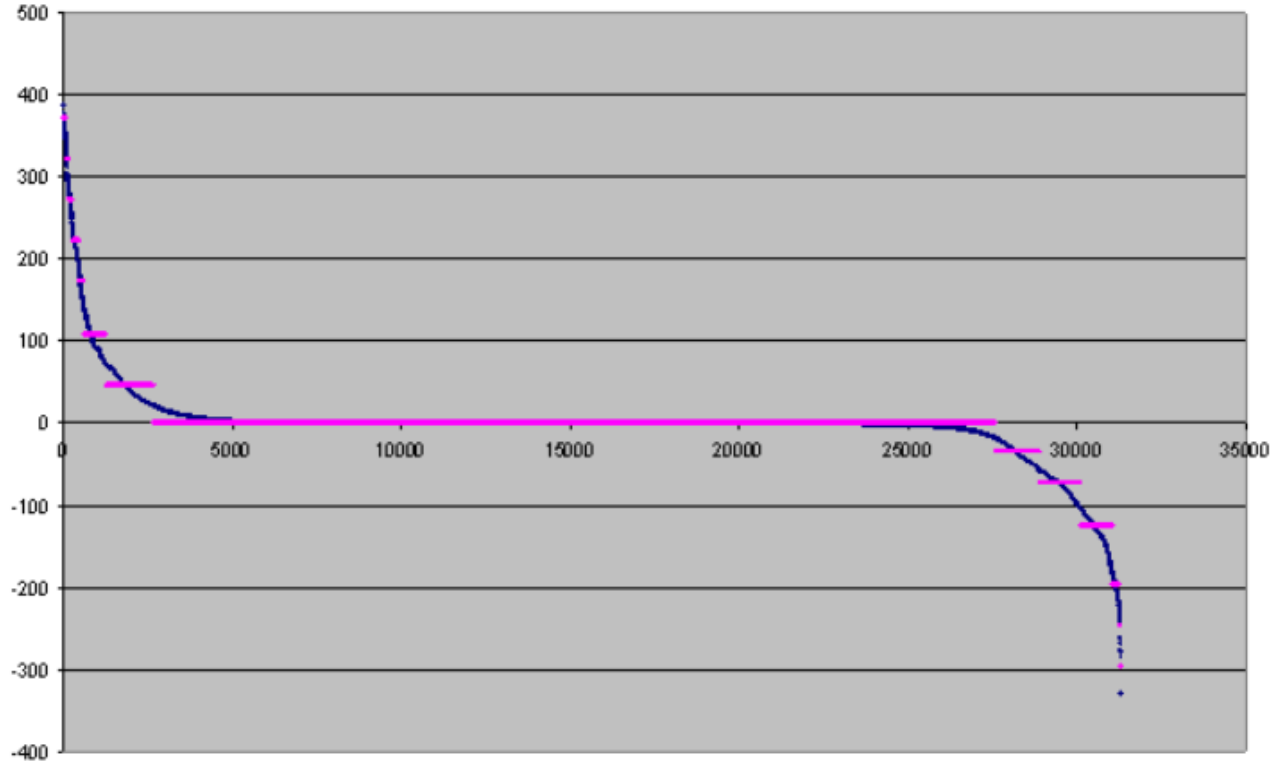
Послідовно використовуючи описану процедуру, отримуємо ітераційний процес, який досить швидко збігається.

Якщо в якості вхідних даних $\left\{\xi_k^0\right\}_{k=0}^{N}$ взяти $\left\{\frac{M}{N}k\right\}_{k=0}^{N}$ і $m_{k+1/2}^0 = \frac{M}{N}\left(k + \frac{1}{2}\right)$, тоді не потрібно зберігати кодову таблицю, але тоді і похибка більше.

У разі якщо для рівномірного наближення замість медіани взяти математичні очікування, тоді похибка буде істотно менше, але при цьому значення цих математичних очікувань потрібно зберігати в кодовій таблиці.

Квантування з найменшою середньоквадратичною помилкою називається квантуванням Ллойда-Макса [8]. Не дивлячись на очевидні переваги, квантування Ллойда-Макса практично не використовується. На це існують вагомі причини. Насамперед, алгоритм Ллойда-Макса досить нестійкий при квантуванні дискретних даних, які використовуються в практичних задачах. По-друге, те, що добре для середньоквадратичної метрики не завжди добре для візуального сприйняття. Мається на увазі той факт, що при оптимальному квантуванні (в сенсі мінімізації середньоквадратичної помилки) інтервали квантування розташовуються таким чином, щоб на кожному проміжку середньоквадратична помилка була однакова. Таким чином, чим більше на проміжку згруповано заквантованих чисел, тим цей проміжок буде вужче. А так як великих (по модулю) чисел істотно більше, то помилка квантування кожного окремого великого числа істотно більше помилки квантування малого числа. Внесок великого коефіцієнта в формування сигналу більше, ніж внесок малого, то як наслідок, відновлюваний сигнал, до якого застосовувалося квантування Ллойда-Макса, створює істотні артефакти. Все це обмежує використання оптимального квантування і призводить до того факту, що найбільш поширеним квантуванням є рівномірний. При цьому квантовочними числами є або значення математичного очікування коефіцієнтів, що потрапили в один інтервал квантування,

або значення медіани - середини проміжку квантування. У першому випадку потрібно зберігати кодову таблицю, яка містить квантовочні числа, а в другому, тільки значення максимального коефіцієнта і число інтервалів квантування.
Наведемо ілюстрацію використання оптимального квантування на шістнадцять інтервалів з десятьма ітераціями для коефіцієнтів, отриманих в результаті фільтрації тестового зображення frymire.bmp

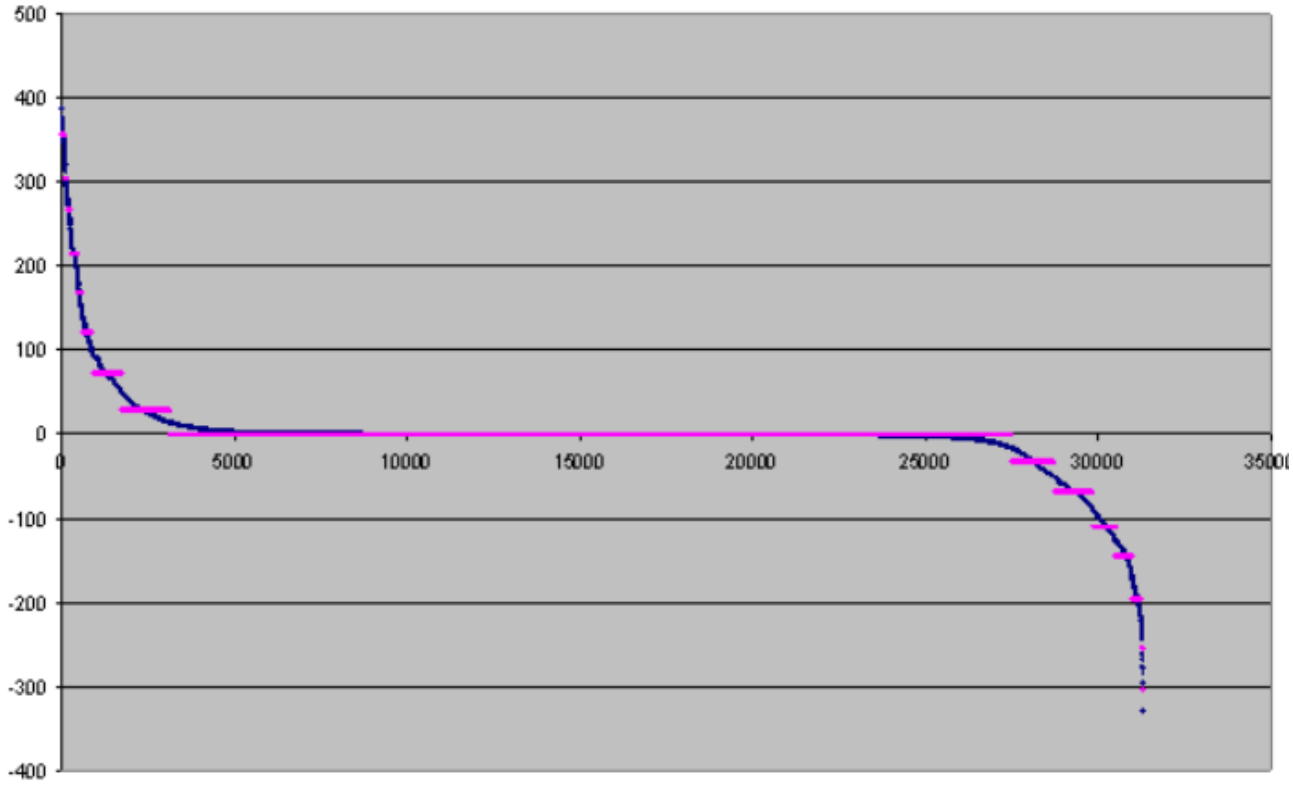


Наведемо значення помилки квантування в залежності від числа ітерацій

Число ітерацій	1	2	3	4	5	32	100
Похибка квантування	6578.5	6443.6	6311.3	6136.1	6014.2	4551.5	4550.4

Зауважимо, що незалежно від числа ітерацій, швидкість декодування одна і та ж. Як видно з наведеного графіка, ширина інтервалу квантування великих чисел істотно відрізняється від ширини інтервалу квантування маленьких чисел. Щоб уникнути артефактів, породжених цим ефектом, нами запропонована наступна модифікація. Спочатку опишемо цю модифікацію для додатних значень квантування. Для заданого числа 2N інтервалів квантування, використовуючи описаний ітераційний процес, одержуємо оптимальний розподіл інтервалів квантування, вибираємо N отриманих інтервалів квантування, відповідних малим значенням квантованим числам і залишаємо їх без зміни. Решту чисел квантуємо рівномірно на N інтервалів.
Якщо ж дані зі знаком, то ця модифікація буде такою: спочатку, для заданого числа 4N інтервалів квантування, використовуючи описаний ітераційний процес, одержуємо оптимальний розподіл інтервалів квантування, вибираємо 2N отриманих інтервалів квантування, відповідних малим значенням квантуємих чисел і залишаємо їх без зміни.
Решта великі за модулем числа квантуємо рівномірно на N інтервалів, відповідно додатні і на N інтервалів від’ємні.

В цьому випадку окремо зберігаються кодові таблиці для малих і великих чисел. Ілюстрація цього методу квантування приведена на наступному малюнку.



Ясно, що можна реалізувати модифікації алгоритму, для будь-якого наперед заданого числа оптимального і рівномірного квантування.
Відзначимо той факт, що отриманий алгоритм оптимального квантування можна використовувати для вирішення інших, супутніх задач - очищення зображенні або для збіднення палітри кольорів. У цьому випадку, на початку проводимо оптимальне квантування колірних компонент, далі вводимо напівінтервали, в яких значення кольорів розподіляємо по ймовірності їх зустрічі. Як результат отримуємо зображення зі збідненою палітрою і без різких границь.

Кодування повторів (Run-Length Encoding)
Кодування повторів або метод групового стиску є одним із старіших і найпростіших алгоритмів. На сучасному етапі він використовується в основному для стиску графічних файлів [9]. Одним з перших поширених форматів, організованих за допомогою цього типу компресії, є формат PCX. "Класичний" варіант цього методу передбачає заміну послідовності символів, що повторюються, на рядок, що містить сам цей символ, і число (лічильник), яке показує, скільки разів цей символ повторюється. Наприклад, рядок 'AAAAAA CCCCCEEEFFFFFF' буде записано у вигляді: '6A5C3E6F'. При цьому можна перші чотири біти відвести під лічильник, другі під значення. Таким чином весь рядок буде записаний в чотирьох байтах.
Існує велика кількість різних модифікацій RLE. Всі вони так чи інакше пов'язані з структурою даних, які підлягають стиску. Перш за все структура методу групового стиску істотно залежить від використовуваного словника і вибору значень, до яких застосовується RLE. Як правило, використовування RLE в класичному значенні не тільки не зменшить загальний об'єм даних, але може навіть його збільшити. Тому, для

ефективного використання методу групового стиску, треба використати ознаку повторів. Як правило, для ознаки повторів використовується прапорець, в якості якого може бути старший біт. Наприклад, якщо словник складається не більше, ніж з 128 символів, то в байті, що визначає символ, сім біт відведені під його значення. Старший біт (прапорець) вказує на наявність чи відсутність повторів. Якщо значення цього біта дорівнює нулю, то повторів немає, якщо одиниці, то наступний байт містить число повторів. В тому разі, якщо ланцюжки символів, що повторюються, істотно довше ніж 256, то старший біт лічильника можна також зарезервувати. Якщо старший біт лічильника дорівнює нулю, то решта семи біт містить значення цього лічильника, якщо одиниці, то під лічильник відводиться не тільки сім біт, що залишилися, але і наступний байт.

При стиску послідовностей елементів, що повторюються, доцільно використовувати ті закономірності, які є у послідовності даних. При використовуванні закономірностей зміни даних можна використовувати різні прийоми для того, щоб перегрупувати дані найвигіднішим для нас чином.

Наприклад, при використанні сплесків і застосування квантування до частотних доменів, відквантовані коефіцієнти не тільки утворюють повторні ланцюжки, але і їх структура така, що довжини цих ланцюжків мають загальну тенденцію до збільшення або зменшення, тобто дані згруповані таким чином, що їх значення по модулю "майже" спадають, тобто даних, які порушують закон спадання на багато менше.

Описаний алгоритм використовує той факт, що якщо з послідовності даних викинути нулі, то з однієї сторони, значення, що залишилися, утворюватимуть "майже" спадаючу послідовність (то є їх розрядність спадатиме), а з другого боку, кількість "пачок" нулів, тобто кількість згрупованих в однозв'язну множину нулів в первинній послідовності, буде такою що їх розрядність буде зростати (але не монотонно).

Приведемо докладний опис алгоритму на прикладі.

Дані	2	3	-5	0	11	31	0	0	-3
Прапорець	1	1	0		1	0			1
Знак	1	1	0		1	1			0
Числа	2	3	5		11	31			3
Нулі				1			2		

-9	0	0	0	0	1	-1	1	0	1
0					1	1	0		1
0					1	0	1		1
					1	1	1		1
	4							1	

Перший рядок таблиці містить вхідні дані.

Другий рядок - послідовність прапорців. Значення прапорця дорівнює одиниці відповідає тому, що за даним числом йде число відмінне від нуля, якщо ж значення прапорця дорівнює нулю, це значить, що за даним числом йде послідовність нулів (може бути ця послідовність складається тільки з одного нуля).

Третій рядок містить прапорці знаків відмінних від нуля. Кожному числу послідовності відмінному від нуля ставиться у відповідність прапорець із значенням рівним 1, якщо це число позитивне і 0, якщо воно негативне.

У четвертому рядку стоять модулі значущих (тобто відмінних від нуля) значень послідовності даних.
Останній рядок містить довжини послідовностей нулів, що повторюються.
На основі цієї таблиці побудуємо наступні бітові послідовності

Число					Прапорець	Знак
0	0	0	1	0	1	1
0	0	0	1	1	1	1
0	0	1	0	1	0	0
	1	0	1	1	1	1
	1	1	1	1	0	1
0	0	0	1	1	1	0
	1	0	0	1	0	0
0	0	0	0	1	1	1
	0	0	0	1	1	0
	0	0	0	1	0	1
0	0	0	0	1	1	1

У зв'язку з тим, що в послідовності значущих чисел даних найбільшу питому вагу мають одиниці, інвертуємо значення останнього біта даних. В результаті одержимо наступну таблицю.

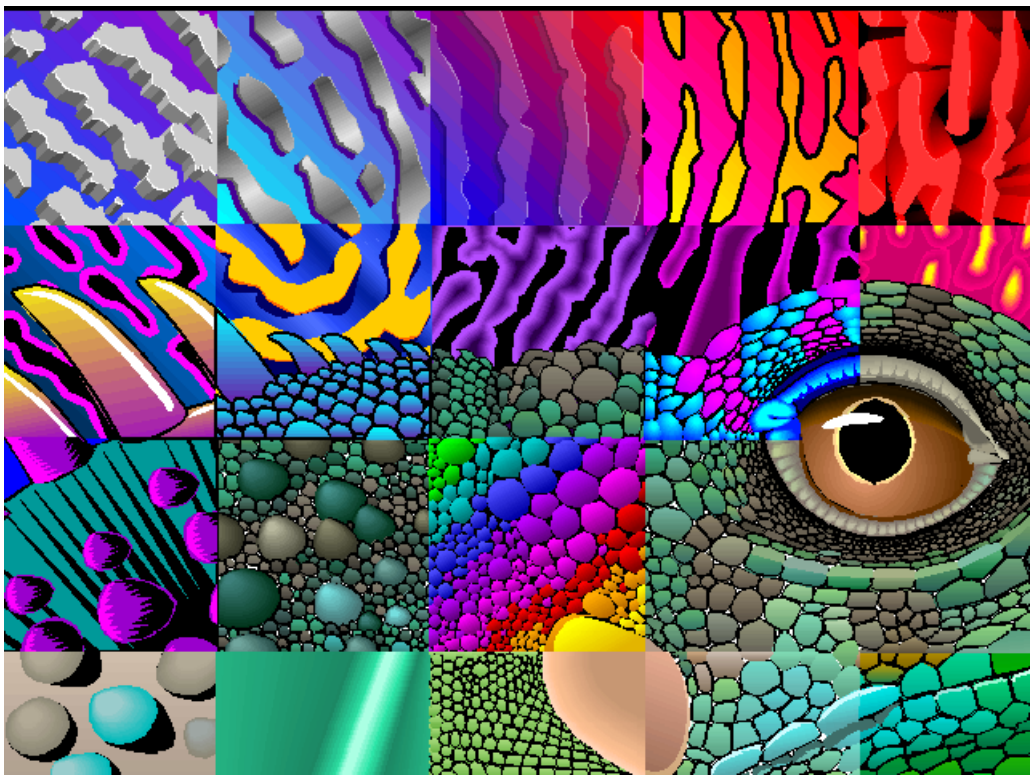
Число					Прапорець	Знак
0	0	0	1	1	1	1
0	0	0	1	0	1	1
0	0	1	0	0	0	0
0	1	0	1	0	1	1
1	1	1	1	0	0	1
0	0	0	1	0	1	0
0	1	0	0	0	0	0
0	0	0	0	0	1	1
0	0	0	0	0	1	0
0	0	0	0	0	0	1
0	0	0	0	0	1	1

Набуті значення упорядкуємо по стовпцях -- на початку перший стовпець (значення старшого біта), потім другий і так далі. Одержану послідовність групуємо відводячи на одне число 4 біти і до одержаної послідовності застосовуємо традиційний метод RLE.
Значення кількості нулів, що повторюються, теж напишемо у таблицю

0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	0	0	1

Також, як і раніше, набуті значення упорядкуємо по стовпцях - на початку перший стовпець (значення старшого біта), потім другий і так далі. Одержану послідовність групуємо відводячи на одне число 4 біти, і до одержаної послідовності застосовуємо традиційний метод RLE.

Тестове зображення frymire



Контрольні питання

1. В чому ідея ентропійних методів стиску?
2. В чому ідея словарних методів стиску?
3. Який метод стиску найщільніше підходе до межі Шенона?
4. Поняття квантування?
5. В чому сенс оптимального квантування?

Рекомендована література

1. Huffman D.A. A method for the construction of minimum redundancy codes / D.A.Huffman // Proc. IRE .— 1952 .— №40 .— C.1098-1101.
2. Shannon C.E. A mathematical theory of communication / C.E.Shannon // Bell System Technical Journal .— 1948 .— №27 .— C.379-423, 623-656.
3. Ziv J. An Universal Algorithm for Sequential Data Compression/J.Ziv, A.Lempel// IEEE Transactions on Information Theory.- 1977.- Vol. 23, N 3.— p. 337-343.
4. Ziv J. Compression of Individual Sequences via Variable Rate Coding/J.Ziv, A.Lempel// IEEE Transactions on Information Theory.- 1978.- Vol. 24., N 5.— p. 530-536.
5. Welch T. A. A Technique for High-Performance Data Compression/T.A.Welch // Computer. --1984.- Vol. 17., N 6.

6. Lloyd S.P. Least squares quantization in PCM / S.P.Lloyd // IEEE Trans. Inform. Theory .— 1982 .— №2(vol. IT-28) .— C.129-136.
7. Max J. Quantization for minimum distortion / J.Max // IRE Trans. Inform. Theory .— 1960 .— №2(vol. IT-6) .— C.7-12.
8. Gray R. Quantization/R.Gray, D.Neuhoff// IEE Transactions on Infirmation Theory.- 1998.- 44(6).- P. 1-63.
9. Golomb S. Run length encoding/S.Golomb// IEE Transactions on Infirmation Theory.- 1966.- 12(7).- P. 399-401.
10. Борн Г. Форматы данных / Г.Борн .— К.: BHV, 1995 .— 668 с.
11. Ватолин Д.С. Алгоритмы сжатия изображений / Д.С.Ватолин .— М: Изд. МГУ, 1999 .— 76 с.
12. Гонсалес Р. Цифровая обработка изображений / Р.Гонсалес, Р.Вудс .— М: Техносфера, 2005 .— 1070 с.
13. Климов А.С. Форматы графических файлов / А.С.Климов .— М: ДиаСофт Лтд, 1995 .— 480 с.
14. Лигун А.О. Комп’ютерна графіка (обробка та стиск зображень)/А.О.Лигун, О.О.Шумейко.- Д.: Біла К.О., 2010.- 114 с.

Конспект лекцій з дисципліни «Технології створення мультимедіа застосувань» за освітньо-професійною програмою «Інженерія програмного забезпечення» для здобувачів першого (бакалаврського) рівня із спеціальності 121 «Інженерія програмного забезпечення» / Укл. Шумейко О.О. // Кам'янське: ДДТУ, 2019. – 169 с.

Укладач: д.т.н., проф. Шумейко Олександр Олексійович

Підписано до друку 21.11.2019 р.
Формат А5 Обсяг 7,68 др. ар.
Тираж 30 екз. Заказ № 247

м. Кам'янське,
вул. Дніпробудівська, 2.