МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з дисципліни

“МЕТОДИ ОБРОБКИ БІОМЕДИЧНИХ ЗОБРАЖЕНЬ”

для студентів усіх форм навчання
напряму 6.051402 «Біомедична інженерія»

Електронне видання

ЗАТВЕРДЖЕНО
кафедрою «Біомедична інженерія».
Протокол № 6 від 18.01.2014 р.

ХАРКІВ 2014

Методичні вказівки до лабораторних робіт з дисциплін «Методи обробки біомедичних зображень» для студентів усіх форм навчання напряму 6.051402 «Біомедична інженерія» [Електронне видання] / Упоряд. О.Г. Аврунін, О.І. Скляр – Харків: ХНУРЕ, 2014. – 68 с.

Упорядники: О.Г. Аврунін
О.І. Скляр

Рецензент: Л.О. Авер’янова, канд. тех. наук, доцент каф. БМІ

ЗМІСТ

.

ВСТУП

Медичні зображення, які отримують при обстеженні пацієнта (рентгенологічному, ультразвуковому, магніто-резонансному тощо), є найбільш інформативним носієм діагностичної інформації. На даному етапі автоматизована програмна обробка таких зображень проводиться стандартними засобами, можливості яких є досить обмеженими, в той же час потреби в оригінальних засобах обробки біомедичних зображень є невичерпними. Необхідність підготовки спеціалістів, здатних розробляти спеціалізоване програмне забезпечення для обробки діагностичних медичних зображень, є актуальним завданням.

Суть та задачі дисципліни. Дисципліна Методи обробки біомедичних зображень» є однією з обов'язкових дисциплін для студентів спеціальностей медико-технічного спрямування. Її основою є дисципліни фізика, математика, основи програмування, а також математичні методи обробки інформації. Дисципліна «Методи обробки біомедичних зображень» надає методи обробки та способи їх програмної реалізації у об'єктно-орієнтованому середовищі Delphi.

Метою вивчення дисципліни «Методи обробки біомедичних зображень» є засвоєння принципів створення програмного забезпечення для обробки растрових медичних зображень.

Вивчення дисципліни «Методи обробки біомедичних зображень» потребує знання:

– фізичних явищ, покладених в основу побудови растрових медичних зображень;

– основ принципів об'єктно-орієнтованого програмування;

– систем відображення інформації;

– математичних методів обробки інформації та ін.

Окрім лекцій, при вивченні цієї дисципліни, передбачено виконання лабораторних робіт з використанням комп'ютерів, на яких встановлене програмне забезпечення Delphi та є бібліотеки традиційних медичних зображень у форматі .bmp. Це дозволяє студентам набувати навичок створення програмного забезпечення для обробки реальних біомедичних зображень.

Опис лабораторного робочого місця. На лабораторному місці встановлено ПК типу IBM PC/AT з встановленим програмним забезпеченням: Borland Delphi 6.

Порядок виконання лабораторних робіт. На початку першого заняття всі студенти повинні ознайомитися з правилами техніки безпеки і розписатися про це в журналі обліку виконання лабораторних робіт. Студенти, які не ознайомилися з правилами техніки безпеки, до виконання лабораторних робіт не допускаються.

Кожній лабораторній роботі (ЛР) має передувати самостійна підготовка студентів, у процесі якої потрібно вивчити методичні вказівки до лабораторної роботи, конспект лекцій та рекомендовані літературні джерела. Перед початком ЛР викладач перевіряє підготовленість студентів до виконання конкретної лабораторної роботи, де студенти мають знати мету і порядок виконання роботи.

Результати виконання ЛР відображаються у звіті, який має вміщувати: назву лабораторної роботи, мету роботи, опис використаних візуальних об’єктів, зображення створеного інтерфейсу, лістинг розробленого програмного засобу, результати виконаного завдання та висновки.

До початку наступної ЛР студент має подати викладачеві повністю оформлений звіт про попередню роботу та захистити її. Залік з ЛР студент отримує після співбесіди з викладачем за темою виконання робіт.

ТЕМА 1 ВИВЧЕННЯ БІБЛІОТЕКИ ВІЗУАЛЬНИХ КОМПОНЕНТІВ DELPHI, НАПИСАННЯ ПРОГРАМ ОБРОБКИ ПОДІЙ ТА СТВОРЕННЯ ІНТЕРФЕЙСА З МОЖЛИВІСТЮ ОБРОБКИ БІОМЕДИЧНИХ ЗОБРАЖЕНЬ

При створенні програмного забезпечення для обробки біомедичних зображень роль середовища, в якому створено програмне забезпечення, дуже важлива, адже користуватись цим програмним продуктом будуть люди досить далекі від програмування та понять, якими користуються програмісти. Якщо це програмне середовище буде візуальним, то користувачу буде легше його освоїти та використовувати всі його можливості. Саме таким програмним середовищем є система Delphi, яка постійно розвивається та удосконалюється.

Система Delphi є інтуїтивно зрозуміла. Бібліотека візуальних компонентів цієї системи досить розвинута. Візуальні компоненти з бібліотеки розміщуються на формі, саме після чого й починається програмування — це написання процедур обробки подій. При розміщенні компонента на формі слід уважно вивчити в інспекторі об’єктів (Object Inspector) властивості (Properties) кожного компонента та події (Events), на які може реагувати компонент.

Під час розробки будь-якого проекту всі файли одного проекту потрібно зберігати в окремій папці.

ЛАБОРАТОРНА РОБОТА № 1 ВИВЧЕННЯ БІБЛІОТЕКИ ВІЗУАЛЬНИХ КОМПОНЕНТІВ DELPHI З ПАНЕЛЕЙ STANDART, ADDITIONAL, УПРАВЛІННЯ НИМИ ЗА ДОПОМОГОЮ OBJECT INSPECTOR ТА СТВОРЕННЯ ІНТЕРФЕЙСА З МОЖЛИВІСТЮ ОБРОБКИ БІОМЕДИЧНИХ ЗОБРАЖЕНЬ

1.1 Мета роботи

Ознайомлення з системою візуального програмування Delphi, бібліотекою візуальних компонентів та розробка інтерфейсу програмного засобу, що призначений для обробки біомедичних зображень.

1.2 Підготовка до виконання роботи

Під час підготовки до виконання роботи слід повторити матеріал, який стосується використання понять „класи”, „об’єкти ” і згадати особливості мови програмування Object Pascal – базової мови програмування Delphi [1].

1.3 Порядок виконання роботи

1.3.1 Запустити програму Delphi 6, при цьому відкриються декілька різних вікон (рис.1.1). Вивчити усі відкриті вікна.

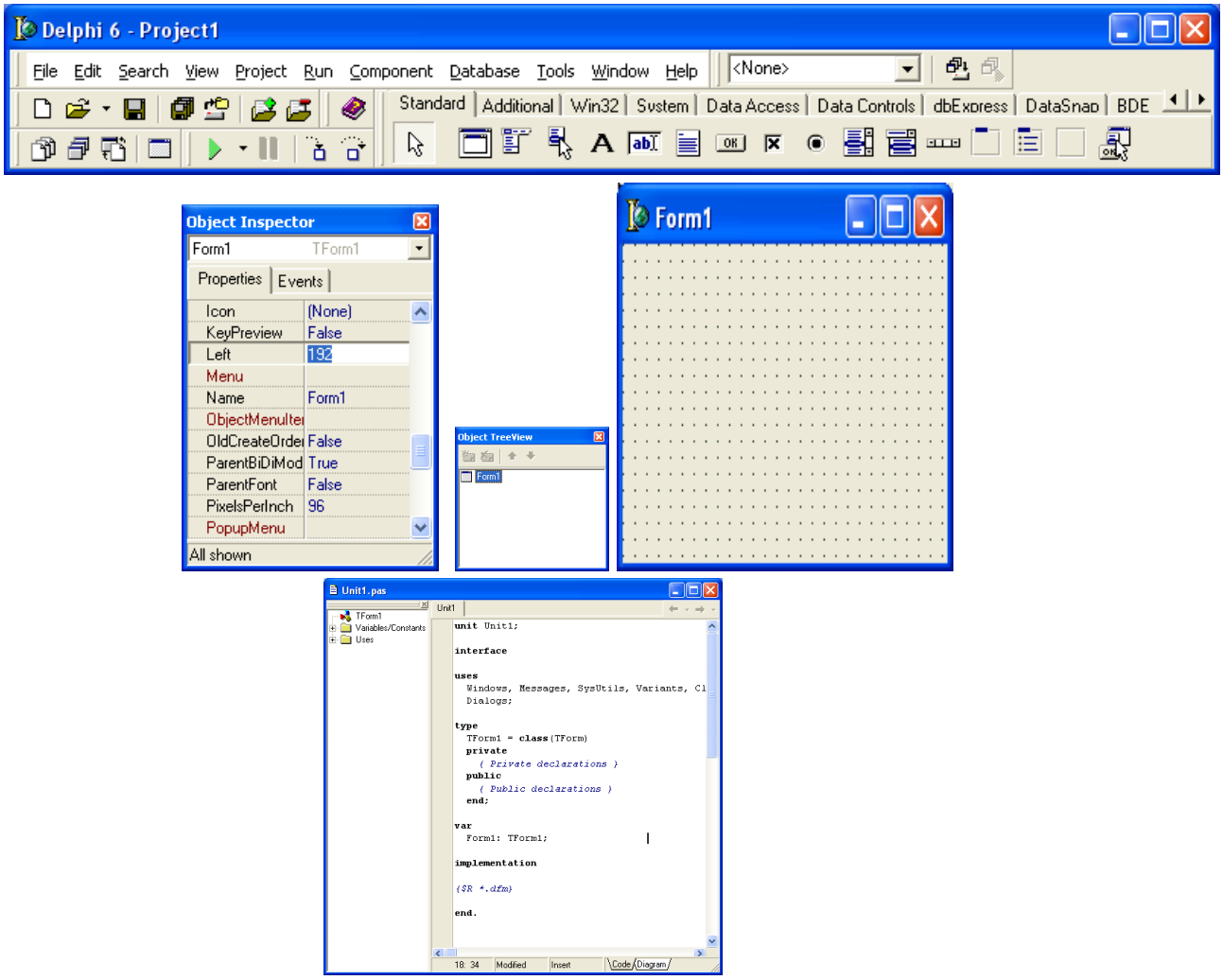


Рисунок 1.1 — Зовнішній вигляд (основні вікна) програмного засобу Delphi

1.3.2 Створити на диску D папку зі своїм прізвищем та зберегти створюваний проект у цій папці під назвою Project1 .

1.3.3 Створити одновіконний (single window) інтерфейс програмного засобу з можливістю обробки біомедичних зображень.

Інтерфейс створюється на базі вікна форми програми — TForm1. При створенні інтерфейсу найчастіше використовуються компоненти типу: TMainMenu, TMemo, TEdit, TButton, TLabel, TImage. Вивчити основні властивості цих компонентів.

1.3.3.1 Вивчити властивості візуального вікна програми — TForm1. Наприклад, вибрати у інспекторі об’єктів властивість „колір” та змінюючи її значення спостерігати зміни кольору Form1. Змінити назву Form1, вибравши властивість „назва” (Caption), та записати проект з новою назвою. Вибрати властивості „ширина” і „довжина” та спостерігати за зміною виду Form1.

1.3.3.2 Вивчити методику створення головного меню програми.

Будь-який розроблений програмний засіб має своє меню. З палітри компонентів Standart вибрати компонент типу TMainMenu. Після цього вказати за допомогою „мишки” місце на Form1 (саме в цьому місці буде розміщено цей компонент, але при запуску програми він не буде візуалізований). Розмістити курсор „мишки” на цьому встановленому елементі та двічі натиснути ліву клавішу (підключиться до роботи дизайнер меню). На формі з’явиться нове вікно Form1.MainMenu1 (рис.1.2), у якому видно зафарбований прямокутник (місце під назву першого компонента меню). В інспекторі об’єктів потрібно вибрати властивість Caption і записати слово, яке матиме назву елемента меню (обраною мовою), що зразу ж відобразиться на Form1 та у робочому вікні, а дизайнер меню запропонує новий елемент. Якщо при записі властивості Caption перед назвою вставляти знак & (він не буде відтворюватись у назві), то даному об’єкту буде присвоєне ім’я назви пункту меню, а якщо знак & не вказати, то кожному пункту меню автоматично буде присвоєна назва „N” з відповідним номером, саме під цим номером для вказаного пункту меню створюватиметься оброблювач подій у вікні Unit1. Коли меню створено, то потрібно закрити дизайнер меню.

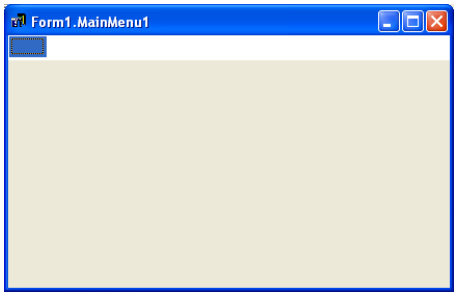


Рисунок 1.2 — Вид форми для проектування головного меню

1.3.3.3 Вивчити методику створення вікон для внесення необхідної інформації при роботі програми.

Під час обробки біомедичної інформації лікарю часто буває необхідно записувати діагноз, або деякі коментарі. З цією метою може використовуватись

багаторядковий текстовий редактор типу TMemo (рис.1.3). Вивчити властивості компонента типу TMemo. У головній формі Delphi вибрати закладку Standart, а на цій закладці вибрати „мишкою” компонент типу TMemo. Перевести курсор на Form1 і вказати „мишкою” місце, де буде розташовано новий компонент. З’явиться вікно з назвою Memo1 та „габаритними” чорними квадратами. За допомогою цих квадратів можна змінити розміри встановленого компонента, також розміри можуть бути змінені за допомогою інспектора об’єктів. Вивчити можливості інших редагувань за допомогою інспектора об’єктів (очистка назви, шрифти, колір, встановлення скролінгів тощо).

Інформація, яка може бути введена користувачем у поле компонента типу TMemo, матиме тип TStrings, тобто вона оброблюватиметься усіма методами, які доступні у класі TStrings.

1.3.3.4 Аналогічно п.1.3.3.3 вивчити властивості компоненту типу TEdit (рис.1.3). На відміну від компонента типу TMemo компонент типу TEdit дозволяє заносити інформацію довжиною лише в один рядок. Занесена інформація відноситиметься до класу TStrings.

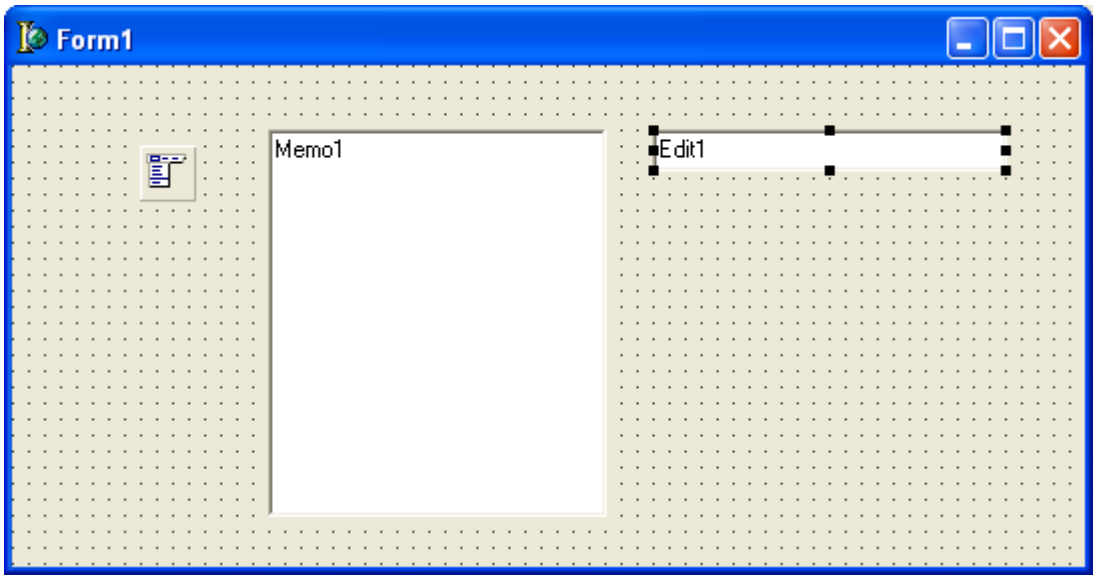


Рисунок 1.3 — Вид головної форми після встановлення компонентів MainMenu1, Memo1 та Edit1

1.3.3.5 Вивчити методику створення різних написів та назв.

Під час створення програм будь-якого призначення часто потрібно створювати різні написи. З цією метою може використовуватись компонент типу TLabel із закладки Standart. Вибрати компонент типу TLabel та встановити

його на головній Form1 і, використовуючи інспектор об’єктів, вивчити властивості цього компонента (зміну назви, розмірів, шрифту, кольору шрифту). Занесена інформація відноситиметься до класу TStrings.

1.3.3.6 Вивчити методику створення кнопок.

Під час виконання будь-яких програм при переході від однієї дії до іншої часто використовується компонент типу TButton (кнопка). Вибрати компонент типу TButton та встановити його на головній Form1, вивчити його властивості.

1.3.3.7 Вивчити методику відкриття зображень.

Під час роботи з будь-якими зображеннями має бути передбачено можливість їх відкриття у створюваній програмі. З цією метою використовується компонент типу TImage із закладки Additional. Найважливішою властивістю цього компонента є властивість Picture, яка саме і дозволяє відкрити потрібне зображення.

Вибрати компонент типу TImage та встановити його на головній формі. В інспекторі об’єктів вибрати властивість Picture, натиснути у правій колонці на зображення трьох крапок, після чого на головні формі з’явиться додаткове вікно, в якому потрібно натиснути на кнопку Load, після чого випадає діалогове вікно для пошуку та відкриття потрібного файлу зображення. Коли потрібне зображення знайдено, то натискають кнопку Ok. Крім того, у інспекторі об’єктів потрібно знайти властивість Proportional і у правій частині цієї властивості вибрати True, тоді зображення, яке відкриватиметься, буде пропорційно до свого розміру вставлене у вікно на формі, якщо цього не зробити, то відкрите зображення може становити лише частку початкового зображення. Відповідно до розмірів відкритого зображення потрібно відкоригувати розміри компонента типу TImage (рис.1.4).

Вивчити інші властивості компонента типу TImage.

Рисунок 1.4 — Вид форми з відкритим біомедичним зображенням

1.3.4 Створити проект інтерфейса програмного засобу для обробки біомедичних зображень, який має містити: компонент типу TMainMenu (з трьох закладок — «Збереження проекту», «Обробка зображення», «Вихід», а пункт меню «Обробка зображення» має мати чотири підпункти — «Гістограма», «Денситограма», «Фільтрація», «Гістограмна корекція»); два компоненти типу TImage, при чому перший компонент має мати назву «Початкове зображення», а другий — «Результати обробки зображення»; чотири компоненти типу TEdit з такими назвами — «Прізвище, ім’я, по батькові», «Вік пацієнта», «Адреса», «Дата проведення обстеження», ці чотири компонента мають мати єдину назву «Дані про пацієнта»; один компонент типу TMemo з назвою «Діагноз»; один компонент типу TButton з назвою «Відкрити зображення». Приклад інтерфейсу показано на рис.1.5. Використовуючи функцію Run, запустити програму на компіляцію і спостерігати створений інтерфейс програмного засобу. В усі поля типу TEdit та TMemo занести відповідну інформацію.

За допомогою функції PrintScreen перенести зображення створеного інтерфейсу до файлу типу .doc.

У лівому верхньому куті інтерфейсу натиснути на логотип Delphi лівою кнопкою «миші», випаде додаткове меню, де потрібно вибрати пункт Close (Закрити), вікно інтерфейсу закриється і повернуться вікна редагування створюваного програмного засобу.

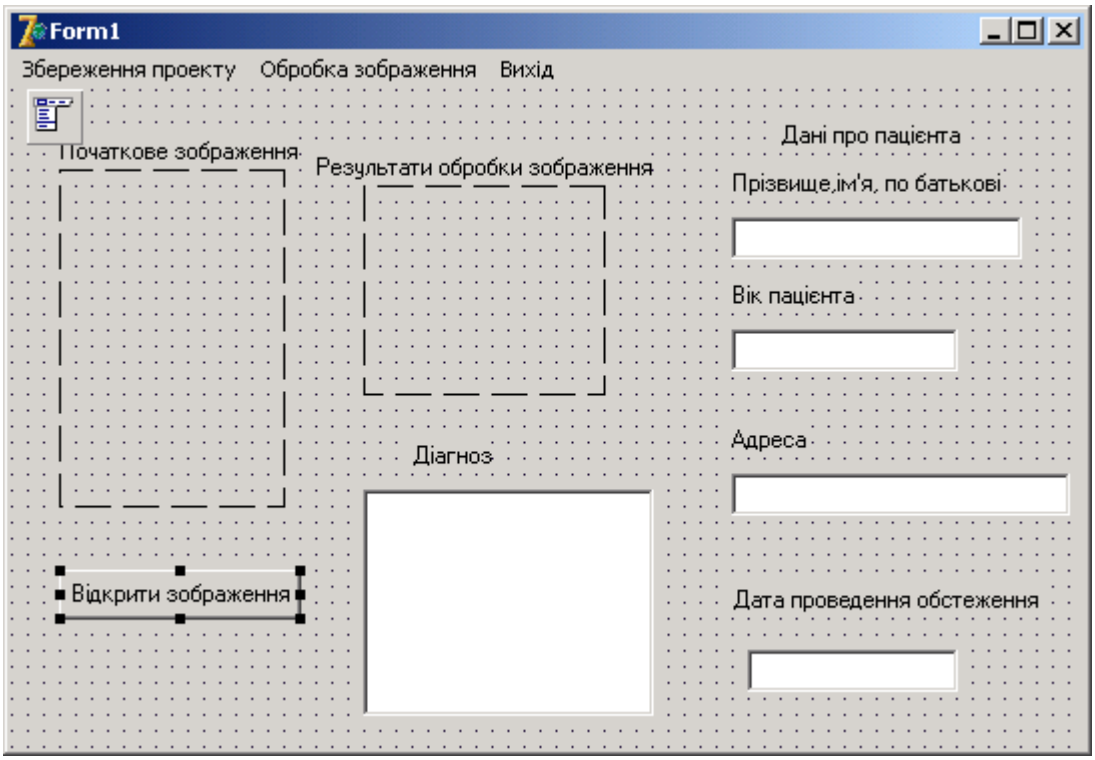


Рисунок 1.5 — Варіант вікна при проектуванні інтерфейсу для обробки біомедичного зображення

1.3.5 Вивчити способи обробки подій.

Все що було викладене вище стосувалось компонентів, які мають бути у розроблюваному програмному засобі, а от способи взаємодії цих компонентів не описувались. Система Delphi дозволяє обробляти події, які можуть виникнути під час роботи програми, наприклад, зображення може бути відкрите лише після натискання на кнопку, чи після вибору відповідного пункту меню. Програміст має сам вирішувати, які події в програмі потребують обробки.

1.3.5.1 Вивчити спосіб відкриття зображення при натисканні на кнопку, яке до цього часу не було видно. Обробка цієї події виконується так. На Form1 двічі натиснути на об’єкт типу TButton, при цьому редактор переключиться ще у одне вікно — Unit1 (рис.1.6), саме у якому і потрібно створити програму обробки події натискання на кнопку. Система Delphi сама створила всі необхідні форми записів, але між операторними дужками **begin** та **end** потрібно записати код виконання дій. Він має бути таким:

Form1.Image1.Picture.LoadFromFile('Hand.bmp'); .

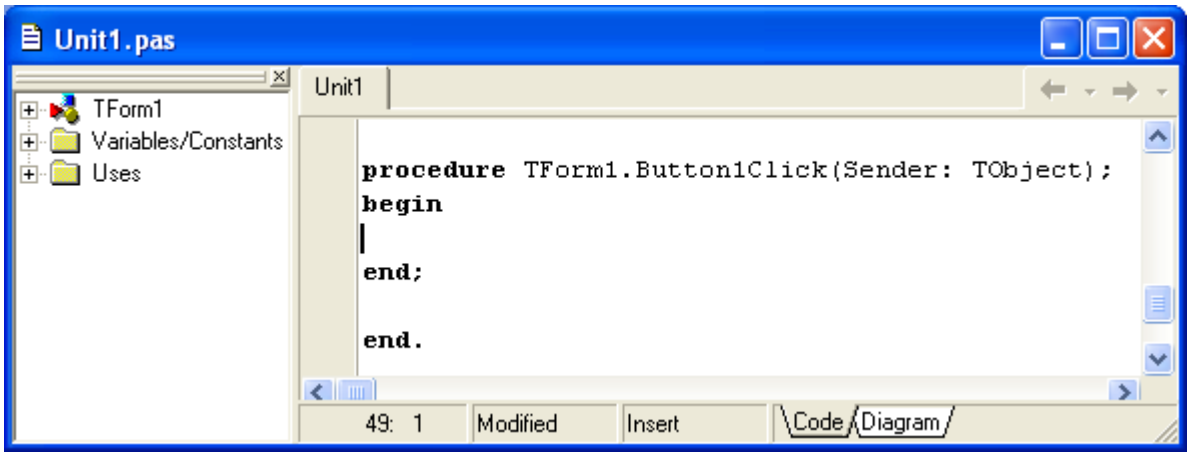


Рисунок 1.6 — Вид вікна для написання програми обробки подій при виборі кнопки

Прокоментуємо всі складові цього запису. Запис Form1 означає, що вибраний об’єкт знаходиться на цій формі. Слід поставити крапку після цього запису і трохи зачекати, тоді випадає підказка, в якій можна вибрати наступний об’єкт або властивість попередньо записаного об’єкту, в даному випадку потрібно поставити курсор на Image1 (якщо ми запланували, що зображення має з’явитися у першому вікні Image1) і натиснути Enter, після цього напис Image1 з’явиться у рядку запису. Таким чином ми вказали на об’єкт, роботу з яким хочемо запрограмувати. Поставимо крапку і знову трохи зачекаємо, з’явиться підказка, де ми виберемо (як і в попередньому випадку за допомогою клавіші Enter) уже властивість цього об’єкту — Picture, це означатиме, що у вікно Image1 ми хочемо вставити зображення. Коли поставимо наступну крапку, то підказка запропонує нам вибрати звідки ми маємо взяти це зображення. Виберемо, як і вище, напис LoadFromFile, після якого потрібно вставити круглі дужки, в яких потрібно вказати шлях до потрібного файла та його назву, причому шлях до файла та його назва мають бути в одинарних кавичках; у даному випадку зображення Hand.bmp знаходилось у папці, де зберігається створюваний проект, тому шлях до файла не вказано, а вказана лише назва файла з розширенням. Отже створено перший оброблювач подій. Вибрати функцію Run, запустити програму, натиснути на кнопку «Відкрити зображення» і спостерігати його, потім (див.п.1.3.4) повернутися до вікна проектування програми.

1.3.5.2 Вивчити спосіб відкриття другого вікна під час вибору пункту меню, наприклад, «Гістограма». Обробка цієї події виконується так. У Form1 вибрати у меню пункт “Гістограма” і двічі натиснути ліву кнопку „миші”,

з’явиться форма оброблювача подій, де між операторними дужками слід вставити такий текст:

```
image2.Canvas.Pen.Color:=clgreen;
image2.Canvas.MoveTo(10,10);
image2.Canvas.LineTo(10,90);
image2.Canvas.LineTo(100,90);
```

після цього вибрати функцію Run і запустити програму на виконання. У вікні програми у головному меню вибрати “Обробка зображення” → “Гістограма”, при цьому на інтерфейсі з’явиться вікно з побудованими координатними осями зеленого кольору, спостерігати його, потім (див.п.1.3.4) слід повернутися до вікна проектування програми.

Тепер прокоментуємо оператори, які були використані в оброблювачі подій. Image2 – це друге вікно, в якому можна виконувати деякі дії з зображенням. Canvas – це властивість зображення, яка подає зображення як канву та дозволяє рисувати у полі цієї канви. Pen – це інструмент, яким виконуватиметься рисунок; якщо вибрано його властивість Color, то слід вибрати колір, яким буде виконуватись рисунок (у даному випадку вибрано зелений колір). У другому рядку слід повторити вибір операторів Image2.Canvas, а потім у круглих дужках використовуючи оператор MoveTo слід вказати координати верхньої точки вісі ординат; у третьому рядку в круглих дужках використовуючи оператор LineTo вказати координати нижньої точки вісі ординат, при цьому у програмі від верхньої до нижньої точки буде прорисована лінія; у четвертому рядку у круглих дужках виуористовуючи оператор LineTo вказати координати правої точки вісі абсцис, при цьому у програмі від лівої нижньої до правої нижньої точки на вісі абсцис буде прорисована лінія.

1.3.5.3. Вивчити можливості рисування на зображенні у відкритому вікні програми.

Рисування ліній. З цією метою потрібно на Form1 вибрати елемент типу TImage, наприклад, Image2, де уже прорисовані вісі координат. Коли цей об’єкт вибрано, то в Object Inspector потрібно вибрати закладку Events, знайти дію OnMouseDown, перейти у праве поле закладки Events та двічі натиснути ліву кнопку миші, при цьому з’явиться оброблювач подій, де між операторними дужками слід вставити такий текст:

```
image2.Canvas.MoveTo(x,y);
```

(при виконанні програми, точка, від якої буде проводитись лінія, матиме координати, що мав курсор при натисканні лівої кнопки миші). Після цього знову слід вернутись до вікна Object Inspector та у закладці Events вибрати дію OnMouseUp, перейти у праве поле і двічі натиснути ліву кнопку миші, при цьому знову з'явиться оброблювач подій, де між операторними дужками потрібно вставити такий текст:

```
image2.Canvas.LineTo(x,y);
```

(при виконанні програми, точка, до якої проводиться лінія, матиме координати, які мав курсор при відпусканні лівої кнопки миші).

Вибрати функцію Run і запустити програму на виконання. У вікні програми у головному меню вибрати "Обробка зображення" → "Гістограма", при цьому на інтерфейсі з'явиться вікно з побудованими координатними осями. Встановити курсор на поле зображення Image2, натиснути ліву кнопку миші і, утримуючи її натиснутою, змістити мишу, після чого кнопку відпустити. Спостерігати побудову лінії. Повторити ці дії. За допомогою функції PrintScreen скопіювати отриману екранну форму та зберегти її в текстовому файлі. Повернутися до вікна проектування програми (див.п.1.3.4).

Рисування фігур. З цією метою потрібно на Form1 вибрати елемент типу TImage, наприклад, Image1. Встановити на Form1 нову кнопку та назвати її "Рисування прямокутника". Двічі натиснути на цю кнопку та в оброблювачі подій між операторними дужками вставити такий рядок:

```
Image1.Canvas.Rectangle(30,70,150,190);
```

вибрати функцію Run і запустити програму на виконання. Натиснути кнопку "Рисування прямокутника" і спостерігати його появу. При задаванні процедури рисування прямокутника у круглих дужках вказуються x1=30, y1=70 — координати лівого верхнього кута прямокутника, x2=150, y2=190 — координати правого нижнього кута прямокутника.

Для того, щоб координати прямокутника можна було задавати довільно, то слід спочатку ввести змінні, наприклад x1, y1, x2, y2, і описати їх тип:

```
var
x1,y1,x2,y2:Integer;
```

потім на формі створити додатково чотири об’єкти типу TEdit та чотири об’єкти типу TLabel та написати такий оброблювач подій при натисканні кнопки “Рисування прямокутника”:

```
x1:=strToInt(Edit5.Text);
y1:=strToInt(Edit6.Text);
x2:=strToInt(Edit7.Text);
y2:=strToInt(Edit8.Text);
Image1.Canvas.Rectangle(x1,y1,x2,y2);
```

Використовуючи функцію PrintScreen,65 скопіювати це зображення та задокументувати його. Закрити програму.

1.3.6 Зберегти розроблені файли на дискету. Закрити програму Delphi.

1.4 Зміст звіту

У звіті наводиться:
— назва роботи;
— мета роботи;
— зображення розробленого інтерфейса;
— лістинг програми;
— висновки.

1.5 Контрольні запитання

1. Як вибираються та встановлюються об’єкти при створенні інтерфейса програмного засобу для обробки зображення у середовищі Delphi?
2. Які найпоширеніші об’єкти має мати інтерфейс для обробки біомедичного зображення?
3. Що таке оброблювач подій та коли він використовується?
4. Як спостерігати роботу створеної програми?
5. Як відкрити медичне зображення у створюваному програмному засобі?
6. Якими засобами можна рисувати у полі зображення?
7. За допомогою яких об’єктів створюються написи на інтерфейсі програмного засобу?

ТЕМА 2 ВИВЧЕННЯ МЕТОДИК СТВОРЕННЯ ПРОГРАМ ДЛЯ АНАЛІЗУ БІОМЕДИЧНОГО ЗОБРАЖЕННЯ БЕЗ ЙОГО ЗМІНИ

Сучасні біомедичні пристрої дозволяють отримувати різноманітні зображення (рентгенограми, томографічні зрізи тощо). Зазвичай лікарі проводять аналіз отриманих зображень візуально, що накладає суб'єктивний відбиток на отриману інформацію. Використання технічних засобів для обробки біомедичних зображень дозволить усунути цей фактор суб'єктивності і в той же час надасть додаткові можливості аналізу. Аналіз зображення — це виділення з зображення потрібної інформації за допомогою автоматичних чи напівавтоматичних систем. Результатом аналізу зображення є не інше зображення, а чисельний опис (у будь-якому вигляді) цього зображення, або його ознак. Ознаками зображення є його найпростіші характеристики, або властивості. До природних ознак зображення відносяться яскравість, текстура різних частин зображення, форма контурів. Розподіл яскравостей, або спектри просторових частот є прикладами штучних ознак.

Біомедичні зображення зазвичай розглядаються та відображуються як двовимірні. Для аналізу технічними засобами зображення має бути виведене на екран монітору ЕОМ (тобто оцифроване), а потім до нього можуть бути застосовані різні методи обробки.

Для можливості аналізу в комп'ютері зображення має подаватися в вигляді матриці елементів розміром m x n. Ці елементи називають пікселами — pixel (picture element), а таке зображення — растровим. Кожен елемент зображення характеризується функцією інтенсивності $I_{i,j} = f\left(x_i, y_j\right)$, де $x_i = x_0 + i \cdot \Delta x$, $y_j = y_0 + j \cdot \Delta y$. Зазвичай при визначенні алгоритмів обробки зображення індекси i, j опускаються, але враховуються при реалізації відповідних процедур. Значення функції I квантовані у межах $I \in \left(0 .. I_{max} - 1\right)$. Ці межі — це діапазон інтенсивностей; часто використовують рівномірне (лінійне) квантування. Значення інтенсивності I задають цілочисельними кодовими комбінаціями, відповідно до кількості відліків рівнів квантування. Зазвичай для опису інтенсивності використовують однобайтові числа, для більшості задач цього достатньо, при цьому числу „0" відповідає мінімальна інтенсивність, а числу 255 — максимальна.

Зображення можуть бути:

1) бінарні (двотонові) – це цифрове зображення, інтенсивність пікселів якого має значення 0 або 1;

2) однотонові – це зображення, де всі пікселі зображення мають тільки одну градацію кольору;

3) напівтонові — це монохромні цифрові зображення, де інтенсивність піксела I задається в градаціях тільки одного кольорового тону і часто називається яскравістю;

4) палітрові – це зображення, які формуються з набору кольорів (відтінків), інтенсивність (колір) яких задається чотирьох або восьмибітним індексом у спеціальній палітрі, яка зберігається разом із зображенням, такі зображення мають назву зображень у індексних кольорах;

5) у натуральних кольорах — це зображення, де кожен піксел характеризується векторним значенням інтенсивності трьох кольорових компонент – червоної, зеленої і синьої (RGB.)

Адитивна колірна модель RGB побудована на основі кубу, заданому у декартових координатах, де значення кольорових складових відкладаються вздовж ребер куба (рис. 2.1). Результуючий колір отримується, якщо деякі кількості кожного з кольорів скласти (змішати). При змішуванні рівної кількості цих (RGB) кольорів отримуються відтінки сірого, координати яких розташовані вздовж головної діагоналі кубу кольорів (пунктирна лінія 0A на рис.2.1).

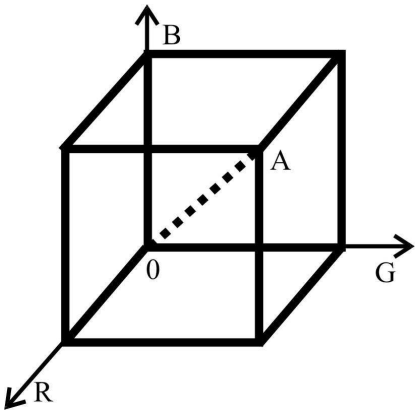


Рисунок 2.1 — Колірна модель RGB

Значення інтенсивності елементу зображення при використанні колірної моделі RGB визначається так:

$$I = B \cdot 0{,}1 + R \cdot 0{,}3 + G \cdot 0{,}6\,. \qquad (2.1)$$

Дана формула ілюструє різний вклад кольорових складових у загальну інтенсивність елементу зображення.

У стандартному 32-бітному форматі дані про кольорові складові RGB та коефіцієнт прозорості α (α-канал використовується у 3-вимірній графіці) записуються чотирма байтами. Значення кожного із кольорів задається байтом. Структура цих даних показана на рис. 2.2.

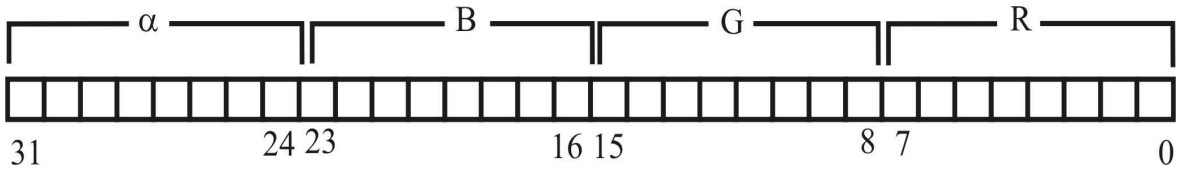


Рисунок 2.2 — Структура даних при використанні колірної моделі RGB

При аналізі зображень використовуються такі поняття як розрізнення, розрізнювальна здатність, контрастність.

Розрізнення — мінімальний розмір елементу зображення, який можна розрізнити. Як правило вимірюється у міліметрах.

Розрізнювальна здатність — кількість елементів на одиницю довжини зображення, які можна розрізнити. Як правило вимірюється у кількості точках на дюйм, або на міліметр.

Контрастність — це різниця між максимальним та мінімальним рівнями інтенсивності зображення.

У комп’ютерах при аналізі зображення використовується система координат показана на рис.2.3.

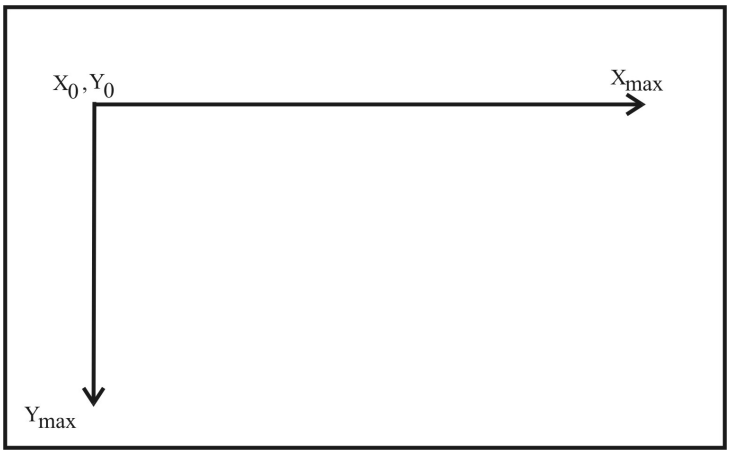


Рисунок 2.3 — Координати при аналізі зображення

ЛАБОРАТОРНА РОБОТА № 2
ВИВЧЕННЯ СПОСОБУ АНАЛІЗА РОЗПОДІЛУ ІНТЕНСИВНОСТІ РАСТРОВОГО БІОМЕДИЧНОГО ЗОБРАЖЕННЯ ТА ПОБУДОВИ ЙОГО ГІСТОГРАМИ

2.1 Мета роботи

Розробка алгоритмів в відповідного програмного забезпечення для аналізу зображення за допомогою гістограми. Ознайомлення з методикою аналізу зображень за допомогою гістограм.

2.2 Підготовка до виконання роботи

Під час підготовки до лабораторної роботи потрібно повторити матеріал, який стосується принципів формування цифрового зображення та його характеристик [2].

Однією з штучних характеристик зображення є гістограма.

Гістограма — це стовпчиковий графік, який надає інформацію про відносну кількість елементів зображення певного кольору (або відтінку) у даному зображенні. На цьому графіку кольори (відтінки) відкладаються вздовж вісі абсцис, а відносна кількість елементів кожного кольору — вздовж вісі ординат.

Відносна частота появи елемента з певним кольором або відтінком визначається так:

$$P(l)=\frac{k_l}{K}, \tag{2.2}$$

де k_l — кількість елементів зображення з l–им кольором або відтінком (для монохроматичного зображення);

l — кількість кольорів або відтінків кольорів у зображенні;

K — загальна кількість пікселів у растровому зображенні, $K = n \cdot i$, де n — кількість пікселів у рядку зображення, i — кількість рядків у зображенні.

При розробці алгоритмів програм для побудови гістограми слід у зображенні проаналізувати колір або відтінок кожної точки. Для цього потрібно задати змінні n та i, де n — кількість пікселів у строчці, а i — кількість рядків. Засобами Delphi це можна зробити використовуючи властивість

Image.Canvas.Pixels[x,y], де x та y відповідно n та i. При цьому буде отримано 32–бітний колірний атрибут піксела, у якому міститься інформація про кольори або відтінки кожного піксела. Значення 32–бітного слова присвоюється змінній c. З цього подвійного слова необхідно вибрати окремі складові значення кожного кольору (див. рис. 2.2) і присвоїти отримані значення змінним r, g, b.

У програмі отримати значення кожного кольору можна використовуючи фрагмент програми мовою асемблера, де це 32–бітне слово буде розділене на окремі складові по байтам.

Для зображення за формулою (2.1) слід визначити інтенсивність кожного елемента, яка зберігається у змінній y.

Після цього можна приступати до формування гістограми. Оскільки весь діапазон інтенсивностей зазвичай записується одним байтом, то це означає що вздовж вісі абсцис може бути 256 стовпчиків. Для побудови гістограми використовується масив Hist[0..255] та змінна y, яка визначає номери елементу масиву (номер стовпчика у гістограмі). Спочатку усім елементам масиву Hist присвоюються нульові стартові значення. Після визначення значення інтенсивності y до елементу масиву Hist з індексом y (Hist[y]) додається одиниця (до його стартового значення 0). Коли при аналізі іншої точки зображення буде знайдено значення інтенсивності, яке також дорівнюватиме y, то до значення у відповідному номері елементу масиву (Hist[y]) знову буде додано одиницю (тобто значення елементу у масиві Hist з номером y уже становитиме двійку). Так будуються всі стовпчики гістограми. Схема алгоритму для формування гістограми яскравості наведена на рис. 2.4.

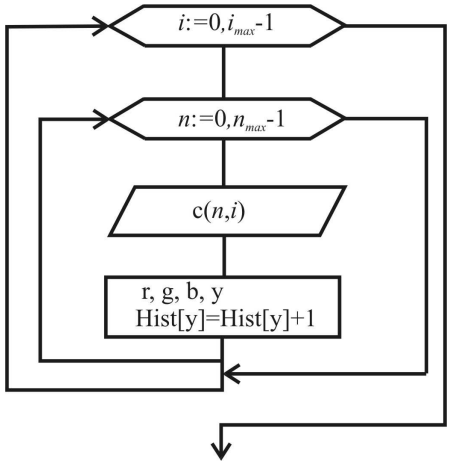


Рисунок 2.4 — Фрагмент алгоритму побудови гістограми зображення

Під час розробки програми для побудови гістограми задається вікно, де вона буде будуватись, з розмірами $p \cdot q$ пікселів. Слід коректно визначити розміри цього вікна.

Для коректного та наглядного відображення висоти стовпчиків гістограми використовуються масштабний коефіцієнт, який враховує максимальну висоту стовпчика (описується змінною $max\,g$) та висоту вікна p, де розміщуватиметься гістограма. Цей масштабний коефіцієнт визначається так:

$$h = \frac{p \cdot Hist[y]}{max\,g}.$$

При побудові стовпчика гістограми слід мати на увазі, що будується він знизу вверх, а відлік координати по осі ординат у комп'ютерах відбувається зверху вниз. Тобто при побудові лінії (стовпчика) у програмі слід вказувати координату по вісі ординат як $p - h$, де p має конкретне значення, а h задається формулою.

Ширина вікна q для побудови гістограми має бути більше 256 пікселів, оскільки саме така кількість стовпчиків може бути на гістограмі.

2.3 Порядок виконання роботи

2.3.1 Запустити програму Delphi 6.

2.3.2 На диску D у папці зі своїм прізвищем відкрити нову папку Lab2, у якій зберегти створюваний проект під назвою Project1. У цю ж папку зберегти шість зображень, на які вкаже викладач, під назвами: 1.bmp, 2.bmp, 3.bmp, 4.bmp, 5.bmp, 6.bmp.

2.3.3 У вікні Form1 розмістити 5 компонентів типу TImage, 11 компонентів типу TButton (рис.2.5). Для першого компоненту Image1 розмір встановити довільний, лише у інспекторі об'єктів властивість Proportional встановити як True. Для компонентів Image2 – Image5 вибрати висоту (height) p однакову та запам'ятати це значення, а ширину (width) q встановити більшою ніж 256 пікселів.

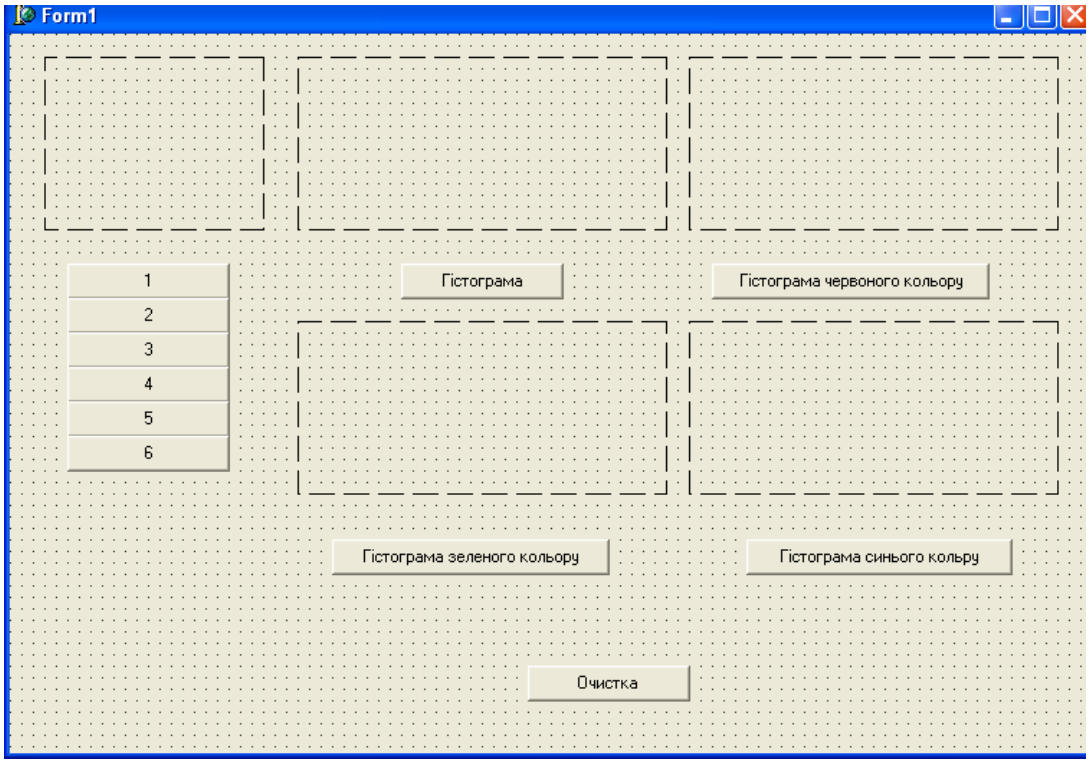


Рисунок 2.5 — Варіант вікна при проектуванні інтерфейса програми для гістограмної обробки зображення

2.3.4 Для перших шести кнопок написати оброблювачі подій, які мають відкривати у Image1 всі шість зображень, які було підготовлено у п. 2.3.2, наприклад: Form1.Image1.Picture.LoadFromFile(‘1.bmp’) тощо.

2.3.5 У вікні Unit1 задати змінні, які використовуватимуться у цій програмі:

```
i, n, p, q:integer;
y, r, g, b: byte;
c:tcolor;
hist:array[0..255] of longint;
maxg:longint;
```

2.3.6 Для кнопки „Гістограма” оброблювач подій написати так:

```
for i:=0 to 255 do
hist[i]:=0;
for i:=0 to Image1.Picture.Height do begin
for n:=0 to Image1.Picture.Width do begin
```

```
c:=Image1.Canvas.Pixels[n,i];
asm
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
y:=round(b*0.1+r*0.3+g*0.6);
hist[y]:=hist[y]+1;
end;
end;
maxg:=hist[0];
for i:=1 to 255 do
if hist[i]>maxg then maxg:=hist[i];
p:=120;
for i:=0 to 255 do begin
image2.Canvas.MoveTo(i,p);
image2.Canvas.LineTo(i,p-round(gist[i]*p/maxg));
end;
```

для змінної p у останніх рядках оброблювача потрібно вказати конкретну величину (див. п. 2.3.3), у цьому прикладі $p = 120$.

2.3.7 Для кнопок „Гістограма червоного кольору”, „Гістограма зеленого кольору”, „Гістограма синього кольору” оброблювачі подій пишуться подібно та відрізняються виразом для змінної y та номером компонента TImage, наприклад, для кнопки „Гістограма червоного кольору” оброблювач подій пишеться так:

```
for i:=0 to 255 do
hist[i]:=0;
for i:=0 to Image1.Picture.Height do begin
for n:=0 to Image1.Picture.Width do begin
c:=Image1.Canvas.Pixels[n,i];
asm
```

```
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
y:=r;
hist[y]:=hist[y]+1;
end;
end;
maxg:=gist[0];
for i:=1 to 255 do
if hist[i]>maxg then maxg:=hist[i];
p:=120;
for i:=0 to 255 do begin
image3.Canvas.MoveTo(i,p);
image3.Canvas.LineTo(i,p-round(hist[i]*p/maxg));
end;
```

2.3.6 Оброблювач подій для кнопки „Очистка” пишеться так:

```
q:=257;
p:=120;
Form1.Image2.Canvas.Rectangle(-1,-1,q+1,p+1);
Form1.Image3.Canvas.Rectangle(-1,-1,q+1,p+1);
Form1.Image4.Canvas.Rectangle(-1,-1,q+1,p+1);
Form1.Image5.Canvas.Rectangle(-1,-1,q+1,p+1);
```

2.3.7 Запустити створену програму. Почергово відкрити всі шість зображень. Для кожного із зображень спостерігати всі чотири гістограми та задокументувати їх використовуючи функцію PrintScreen. Перед побудовою гістограми для кожного нового зображення слід користуватись кнопкою „Очистка”.

2.3.8 Скопіювати лістинг створеної програми та закрити її. Зберегти розроблені файли на дискету.

2.4 Зміст звіту

У звіті наводять:
— назву роботи;
— мету роботи;
— зображення інтерфейса програми з різними гістограмами;
— лістинг програми;
— висновки з порівняннями гістограм досліджуваних зображень.

2.5 Контрольні запитання

1. Які бувають типи цифрових зображень за градаційними параметрами?
2. Як визначається розмір цифрового зображення?
3. Що таке адитивна колірна модель?
4. Як визначається значення інтенсивності елемента зображення за його колірними складовими?
5. Яка структура даних кожного піксела зображення при використанні RGB колірної моделі?
6. Яка система координат використовується при аналізі зображення у комп’ютерах?
7. Що таке гістограма?
8. Який алгоритм програмного формування гістограм?
9. Як вибираються розміри компоненту типу TImage під час побудови гістограми?
10. Чим відрізняються гістограми зображень з різними градаційними параметрами?

ЛАБОРАТОРНА РОБОТА №3
ВИВЧЕННЯ СПОСОБУ АНАЛІЗА РОЗПОДІЛУ ІНТЕНСИВНОСТІ БІОМЕДИЧНОГО ЗОБРАЖЕННЯ ВЗДОВЖ ВИЗНАЧЕНОГО НАПРЯМКУ ТА ПОБУДОВА ДЕНСИТОГРАМИ

3.1 Мета роботи

Розробка алгоритму та відповідного програмного забезпечення для аналізу розподілу інтенсивності зображення вздовж визначеного напрямку та вивчення можливостей роботи створеної програми.

3.2 Підготовка до виконання роботи

Під час підготовці до виконання цієї роботи потрібно повторити теоретичний матеріал з аналізу інтенсивності зображення [2, 3]. Крім того, слід згадати про поняття „метрика зображення”. Це поняття використовується для визначення відстані між двома точками на растровому зображенні: $d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$, де $x1, y1$ та $x2, y2$ координати першої та другої точки на зображенні відповідно. Геометричні фігури на растрі доцільно задавати у параметричній формі, тому з дисципліни вищої математики потрібно пригадати спосіб параметричного завдання прямої.

Аналіз інтенсивності елементів зображення зазвичай проводять для напівтонових зображень, наприклад, серед біомедичних зображень це рентгенівські, ультразвукові, томографічні зображення тощо.

Аналіз інтенсивності зображення проводять вивчаючи значення інтенсивності зображення в кожній точці вздовж визначеної траєкторії, зазвичай вздовж прямої. Значення інтенсивності у відносних одиницях (однобайтне число) відкладають по осі ординат, вздовж вісі абсцис вказують координати x точки, яка аналізується. Такий графік розподілу інтенсивності вздовж певного напрямку називають також профілем яскравості або денситограмою. За денситограмою проводять денситографічний аналіз (аналіз розподілу інтенсивності зображення вздовж певного напрямку), який ефективний при дослідження щільностей анатомічних структур.

Алгоритм побудови денситограми може бути такий.

1. На відкритому компоненті зображення типу TImage, використовуючи властивість Canvas, визначають координати двох точок, на які було вказано під час натискання або відпускання лівої клавіші „миші”.

2. Визначають координати всіх точок, які лежатимуть на прямій (траєкторії), між двома заданими точками.

3. Для кожної точки, координати якої розрахували, проводять вибірку значення інтенсивності.

4. У новому компоненті типу TImage будують графік, де по вісі абсцис відкладають значення розрахованої координати x, а вздовж вісі ординат відповідні значення інтенсивності, які візуально відображають у вигляді графіка.

При визначенні розмірів вікна для побудови денситограми потрібно виходити з того, що значення інтенсивності лежить у межах 0..255, а

координата вздовж вісі абсцис визначається максимальним розміром аналізованого зображення вздовж його діагоналі, або має бути пропорційна їй.

Денситограма, яка побудована вздовж вибраного напрямку може мати шумові завади, тому часто застосовують усереднення отриманої кривої.

Однією з важливих характеристик зображення є різкість між двома точками зображення, яка визначається так:

$$R = \frac{\Delta I}{d} = \frac{I1 - I2}{\sqrt{(x2 - x1)^2 + (y2 - y1)^2}},$$

де $I1, I2$ — інтенсивність зображення у точках з координатами $x1, y1$ та $x2, y2$.

3.3 Порядок виконання роботи

3.3.1 Запустити програму Delphi 6.

3.3.2 На диску D у папці зі своїм прізвищем відкрити нову папку Lab3, у якій зберегти створюваний проект під назвою Project1. У цю ж папку зберегти п’ять зображень (1.bmp, 2.bmp, 8.bmp, 9.bmp, Hand.bmp), на які вкаже викладач.

3.3.3 Розробити програму для побудови денситограми та її найпростішого аналізу.

3.3.3.1 У вікні Form1 розмістити 2 компоненти типу TImage (рис. 2.6). Компоненти типу TImage краще розмістити одне під одним. Розміри цих компонентів вибираються такі, щоб кількість пікселів за координатою абсцис у цих компонент та у зображенні, яке буде аналізуватись, була однаковою, наприклад, 155. Розмір по вісі ординат для компоненти Image2 має бути 256 пікселів, саме стільки градацій сірого має зображення. Розмір компоненти Image1 по вертикалі довільний. Для обох цих компонент у Object Inspector встановити властивості Proportional та Stretch як True.

У вікні Form1 розмістити 8 компонент типу TButton, які назвати так: „10 щаблів”, „Лінійка”, „Трикутник”, „Два трикутники”, „Рука”, „Очистка денситограми”, „Усереднення”, „Різкість”.

У Form1 розмістити 7 компонент типу TLabel, трьом з яких дати такі назви: „Координати курсору на денситограмі”, „Яскравість”, „Параметр фільтру”, а для інших чотирьох у Object Inspector властивість Caption очистити, але розмістити їх як показано на рис. 2.6.

У Form1 встановити компонент типу TSpinEdit (див. рис. 2.6).

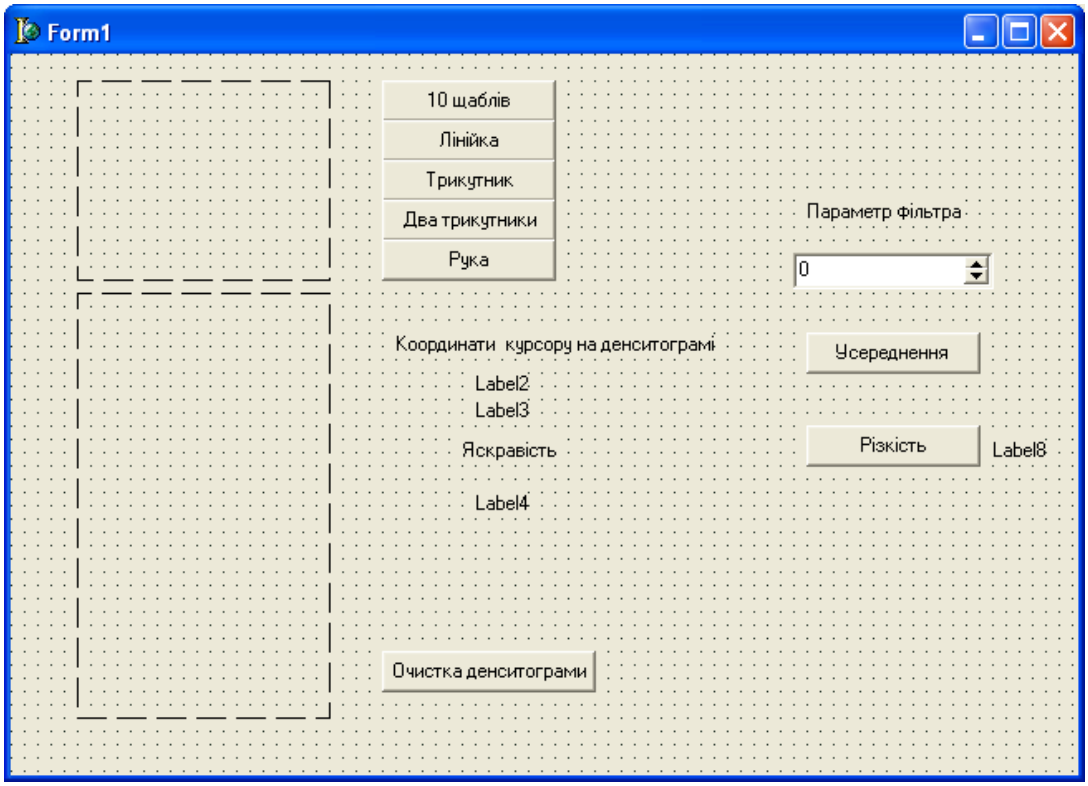


Рисунок 2.6 — Можливий вигляд інтерфейса для програми побудови денситограми

3.3.3.2 Для перших п’яти кнопок написати оброблювачі подій (див. Лабораторну роботу №2, п. 2.3.2), які повинні відкривати у Image1 почергово всі п’ять зображень (`1.bmp`, `2.bmp`, `8.bmp`, `9.bmp`, `Hand.bmp`), які було підготовлено у п. 3.3.2.

3.3.3.3 Для кнопки „Очистка денситограми” оброблювач подій написати так:

```
Form1.Image2.Canvas.FillRect(ClientRect);
for i:=0 to 500 do dgr[i]:=0;
```

3.3.3.4 У вікні Unit1 задати змінні, які будуть використовуватись у програмі побудови денситограми, її усереднення та визначення різкості:

```
r,y1,y2:byte;
  p,n,k,i,j,imax:integer;
```

```
c:tcolor;
s:string;
xx,yy:array[1..2] of integer;
d,tx,ty,h,t,dr,rz:extended;
dgr,dgr1:array[0..500] of longint;
```

Змінній *r* присвоюється значення інтенсивності при побудові денситограми. Змінним *y1*, *y2* присвоюється значення яскравості при визначенні різкості. Змінні *i, n, j* використовуються як лічильники циклів. Змінна *k* використовується як лічильник кількості натискань на ліву клавішу миші під час задавання прямої, вздовж якої будується денситограма. Змінна *p* описує параметр усереднення фільтру. Змінній *imax* присвоюється значення кількості точок у заданій траєкторії для побудови денситограми. Змінній *c* присвоюється значення параметрів кольору кожної точки, яка аналізується. Змінній *s* присвоюється значення різкості. Змінним *xx*, *yy* присвоюється значення координат точок, які утворюють лінію вздовж якої буде аналізуватись яскравість зображення. Змінним *d* та *dr* присвоюється значення відстані між двома заданими точками. Змінні *tx*, *ty* визначають поточну координату. Змінній *h* присвоюється значення кроку зміни параметра *t*. Змінна *t* є параметром у параметричному рівнянні прямої і змінюється у межах [0, 1]. Змінним *dgr* та *dgr1* присвоюється значення яскравості при побудові денситограм (неусередненої та усередненої відповідно). Змінній *rz* присвоюється значення різниці інтенсивностей при визначенні різкості зображення між двома точками.

3.3.3.5 Для побудови денситограми вибрати компонент Image1 і у Object Inspector вибрати Events та знайти подію OnMouseUp (або On MouseDown). Двічі натиснути ліву клавішу миші у правому полі цієї події, при цьому відкриється вікно Unit1, де слід написати такий оброблювач події:

```
begin
k:=k+1;
xx[k]:=x;
yy[k]:=y;
if k=2 then begin
for i:=0 to 500 do dgr[i]:=0;
k:=0;
t:=0;
i:=0;
```

```
d:=round(sqrt((xx[2]-xx[1])*(xx[2]-xx[1])+(yy[2]-yy[1])*(yy[2]-yy[1])));
h:=1/d;
Image2.Canvas.MoveTo(xx[1]+i,257);
while t<=1 do begin
tx:=xx[1]+(xx[2]-xx[1])*t;
ty:=yy[1]+(yy[2]-yy[1])*t;
c:=image1.canvas.Pixels[round(tx),round(ty)];
asm
mov eax,c
mov r,al
end;
dgr[i]:=r;
Form1.Image2.Canvas.Pen.Color:=clblack;
Image2.Canvas.LineTo(xx[1]+i,257-dgr[i]);
Image1.Canvas.Pixels[round(tx),round(ty)]:=clred;
t:=t+h;
i:=i+1;
end;
end;
 imax:=i;
end;
```

цей оброблювач дозволяє:

— отримати координати точок при відпусканні лівої клавіші миші;

— знайти відстань між цими точками;

— розбити цю відстань на ділянки для параметричного представлення отриманої прямої;

— отримати координати всіх проміжних точок;

— отримати значення інтенсивності у точках з визначеними координатами;

— за розрахованими координатами та отриманими значеннями інтенсивності побудувати денситограму у другому компоненті Image2;

— побудувати у Image1 лінію червоного кольору, вздовж якої проводиться денситографічний аналіз зображення;

— зафіксувати значення кількості точок, які є у заданій растровій прямій.

3.3.3.6 Після побудови денситограми є можливість проаналізувати значення її яскравості при переміщенні курсору у її полі. Це можна зробити,

якщо для компоненти Image2 у Object Inspector вибрати Events, знайти подію OnMouseMove, двічі натиснути ліву кнопку миші і в вікні Unit1 написати такий оброблювач:

```
c:=Form1.Image2.canvas.Pixels[x,y];
Label2.Caption:=concat('x= ',IntToStr(x));
Label3.Caption:=concat('y= ',IntToStr(257-y));
Label4.Caption:=concat('I= ',IntToStr(dgr[x]));
```

при цьому слід звернути увагу на відповідне використання компонент TLabel, у даному прикладі компонент Label2 дозволяє виводити значення координати x, компонент Label3 — значення координати y, компонент Label4 — значення яскравості у точці з цими координатами.

3.3.3.7 Побудована денситограма може мати досить багато нерівностей, тому можна створити програму для її усереднення, тобто створити деякий усереднювальний фільтр з параметром вікна p, який при роботі програми буде задаватись у компоненті типу TSpinEdit. При цьому будуватиметься нова денситограма зеленого кольору. Створити оброблювач подій при натисканні на компоненту типу TButton, яка має назву „Усереднення”. Цей оброблювач має бути таким:

```
p:=SpinEdit1.Value;
for n:=0 to imax do
dgr1[n]:=0;
Form1.Image2.Canvas.Pen.Color:=clgreen;
Form1.Image2.Canvas.MoveTo(xx[1],257);
for n:=p to imax-p do begin
for j:=-p to p do begin
dgr1[n]:=dgr1[n]+dgr[n+j];
end;
dgr1[n]:=round(dgr1[n]/(p*2+1));
Image2.Canvas.LineTo(xx[1]+n,257-dgr1[n]);
end;
```

3.3.3.8 Ще однією можливістю при обробці зображення є визначення різкості зображення між двома точками. У цьому випадку визначаються

яскравості у двох точках та їх координати при натисканні чи відпусканні лівої клавіші „миші”. Розраховується різкість R, значення якої виводиться у поле мітки при натисканні на відповідну кнопку.

Створити оброблювач подій при натисканні на компоненту типу TButton, яка має назву „Різкість”, цей оброблювач має бути таким:

```
 c:=Form1.Image1.Canvas.Pixels[xx[1]+1,yy[1]+1];
asm
mov eax,c
mov y1,al
end;
c:=Form1.Image1.Canvas.Pixels[xx[2]+1,yy[2]+1];
asm
mov eax,c
mov y2,al
end;
dr:=round(sqrt((xx[2]-xx[1])*(xx[2]-xx[1])+(yy[2]-yy[1])*(yy[2]-yy[1])));
rz:=(abs(y1-y2));
rz:=round(rz/dr);
str(rz:10:6,s);
 Label7.Caption:=s;
```

При нанесенні на зображення лінії, яка задає координати точок для визначення різкості, остання має червоний колір, тому при визначенні координат точок використано штучний прийом зміщення координат точок на 1 (оточення точки), для того щоб аналізувати яскравість саме зображення, а не червоної лінії. Якщо аналізувати яскравість точок заданих саме координатами червоної лінії, то програма потребує додаткового буферу для зберігання початкового зображення.

У цьому оброблювачі компонент Label7 використовується для виведення значення різкості. При виведенні значення різкості потрібно перетворити формат дійсного числа змінної *rz* у формат строкового типу даних для змінної *s*, що зроблено у передостанньому рядку оброблювача.

3.3.4 Запустити створену програму.

Відкрити по черзі всі наявні зображення.

Для кожного відкритого зображення побудувати денситограму, для чого на зображенні вказати „мишкою” дві точки (точки бажано задавати так, щоб

отримана лінія була майже горизонтальною), при цьому на зображенні з'явиться червона лінія, а у другому вікні буде побудована денситограма.

Задати різні значення параметру фільтру і після натискання кнопки „Усереднення" спостерігати нову усереднену денситограму.

Для визначення різкості між двома довільними точками слід „мишкою" задати ці дві точки, а потім натиснути кнопку „Різкість" і спостерігати отримане значення.

При переході від одного зображення до іншого очищати вікно з денситограмою за допомогою кнопки „Очистка денситограми".

3.3.4.1 Для кожного з відкритих зображень зарисувати отриману денситограму.

3.3.4.2 Для зображення „10 щаблів" записати значення яскравості для кожного з щаблів. Для інших зображень спостерігати зміну значення яскравості при зміні положення курсору.

3.3.4.3 Для зображення „10 щаблів" задати різні параметри фільтру та зарисувати отримані усереднені денситограми та знайти оптимальний параметр вікна для фільтру.

3.3.4.4 Оновити зображення (натискаючи на кнопку з відповідним зображенням). На всіх п'яти зображеннях визначити для яких ділянок різкість зображення буде максимальною. Записати отримані значення.

3.3.4.5 Оновити зображення, задати лінію для побудови денситограми не горизонтально, а під кутом $>30^0$, спостерігати побудовану денситограму та порівняти її з денситограмою вздовж горизонтальної лінії. Пояснити відмінності.

3.3.5 Зберегти розроблені файли на дискету, скопіювати лістинг програми та зберегти його як текстовий документ. Закрити програму.

3.4 Зміст звіту

У звіті наводять:

— назву роботи;

— мету роботи;

— зображення інтерфейса програми з будь-якою з денситограм та відповідним зображення;

— отримані денситограми для п'яти досліджуваних зображень з описом зображення або його видом відповідно до п. 3.3.4.1;

— дані відповідно до п. 3.3.4.2;

— дані виконання п.п. 3.3.4.3, 3.3.4.4;
— лістинг програми;
— висновки.

3.5 Контрольні запитання

1. Які основні характеристики зображення?
2. Що таке метрика зображення?
3. Що таке профіль яскравості?
4. Що таке денситографічний аналіз?
5. Який алгоритм побудови денситограми?
6. Що таке різкість зображення між двома точками?
7. Як проводиться усереднення денситографічної кривої?

ТЕМА 3 ВИВЧЕННЯ МЕТОДИК СТВОРЕННЯ ПРОГРАМ ДЛЯ АНАЛІЗУ БІОМЕДИЧНОГО ЗОБРАЖЕННЯ З ЙОГО ЗМІНОЮ

Будь-яке зображення, і біомедичне також, призначене для того, щоб донести до спостерігача (лікаря) деяку сукупність візуальних даних. Однак отримані біомедичні зображення можуть бути різної якості та різного розміру. Такі зображення потребуватимуть деяких операцій над ними. До таких операцій відносяться: зміна яскравості та контрастності, різні види фільтрації, виділення контурів, стиснення або збільшення зображення тощо. Всі зазначені операції призводять до зміни початкового зображення, але такі зміни дозволяють отримати більше інформації з початкового зображення.

У випадках, коли проводиться корекція зображення з його зміною, слід обов'язково зберігати початкове зображення.

ЛАБОРАТОРНА РОБОТА №4
ВИВЧЕННЯ СПОСОБУ ТОНОВОЇ КОРЕКЦІЇ ЗОБРАЖЕННЯ

4.1 Мета роботи

Розробка алгоритму та відповідного програмного забезпечення для тонової корекції зображення.

4.2 Підготовка до виконання роботи

Під час підготовки до роботи необхідно повторити теоретичний матеріал, який стосується покращення якості біомедичного зображення. Особливу увагу слід звернути на покращення зображення, яке є тіньовим.

4.2.1 В попередніх лабораторних роботах вивчався аналіз зображення без зміни його змісту. Якщо зображення, яке потрібно аналізувати, не досить якісне (не контрастне, не різке, тощо), то перед початком його аналізу потрібно виконати його попередню обробку, яка надасть можливість покращити таке зображення.

При аналізі зображення використовують таке поняття як чутливість. Коефіцієнт чутливості C визначається так:

$$C = \frac{I_i - I_{i-1}}{(I_i + I_{i-1})/2} \cdot 100\%,$$

де I_i та I_{i+1} — значення рівнів яскравості двох сусідніх елементів зображення.

Чим менше значення C, тим більша чутливість.

Людина має найбільшу чутливість у області світлих тонів, отже при „покращенні" зображення його необхідно зміщувати в область світлих тонів.

Одним із способів „покращення" зображення є тонова корекція.

Як вказувалось у другій лабораторній роботі гістограма дозволяє визначити, як розподілені рівні яскравості у зображенні. Аналізуючи гістограму, легко визначити, в яких областях яскравості сконцентрована вся інформація про зображення: у тіньових, у світлих чи у області середніх тонів. Щоб змінити тоновий баланс зображення потрібно відкоригувати цей розподіл. Зазвичай це робиться за допомогою градаційних кривих. На графіку такої кривої вздовж вісі абсцис відкладаються рівні сірого для існуючого зображення, а вздовж вісі ординат — рівні сірого для відкоригованого зображення (рис. 3.1). Такий графік є передаточною функцією. Якщо цей графік лінійний, то передаточна функція є лінійною. Якщо функція не лінійна, то корекцію за допомогою такої функції називають гама-корекцією. В випадку лінійної корекції значення вихідної функції визначається так:

$$I_{вих} = \frac{I_{вх} - I_{X1}}{I_{X2} - I_{X1}} \cdot (I_{Y2} - I_{Y1}),$$

де $I_{вх}$— значення яскравості піксела у початковому зображенні, яке лежить у межах I_{X1} та I_{X2};

$I_{X2} - I_{X1}$ — ефективний діапазон вхідної передаточної характеристики, де I_{X2}— максимальне вхідне значення передаточної функції, а I_{X1} — мінімальне вхідне значення передаточної функції;

$I_{Y2} - I_{Y1}$ — ефективний діапазон вихідної передаточної функції, де I_{Y2} – максимальне вихідне значення перехідної характеристики, а I_{Y1} – мінімальне вихідне значення перехідної характеристики.

Ефективні діапазони перехідної характеристики ще мають назву граничних коефіцієнтів контрастності.

Коефіцієнт перетворення (розтягування) діапазону визначається так:

$$K_{П} = \frac{I_{X2} - I_{X1}}{I_{Y2} - I_{Y1}} \ . \qquad (3.1)$$

При $I_{X2} > I_{X1}$ та $I_{Y2} > I_{Y1}$ виконується пряма корекція зображення, якщо $I_{Y2} < I_{Y1}$, то виконується інверсна корекція зображення.

Така корекція дозволяє виконати „розтяжку" певного діапазону яскравостей зображення на весь діапазон яскравостей. Якщо градаційна крива має інший нахил (рис. 3.2), то буде виконано інвертування зображення.

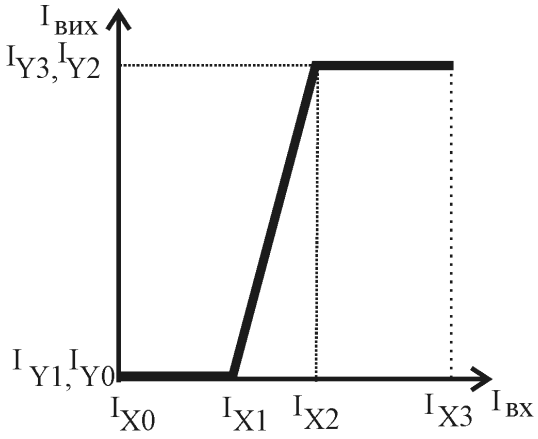


Рисунок 3.1 — Лінійна градаційна крива

На рис. 3.3 представлені дві градаційні криві, які дозволяють змінювати яскравість зображення, зміщення градаційної кривої вверх (рис. 3.3а) робить зображення більш світлим, а зміщення кривої вниз (рис. 3.3.б) — робить зображення більш темним.

Зміна кута нахилу градаційної кривої приводить до зміни контрасту зображення (рис. 3.1). Рівні сірого, які ближчі до білого кольору, стають зовсім білими, а ті, які ближче до чорного, — стають зовсім чорними. Усі проміжні рівні сірого віддаляються один від одного, що збільшує контраст між точками зображення.

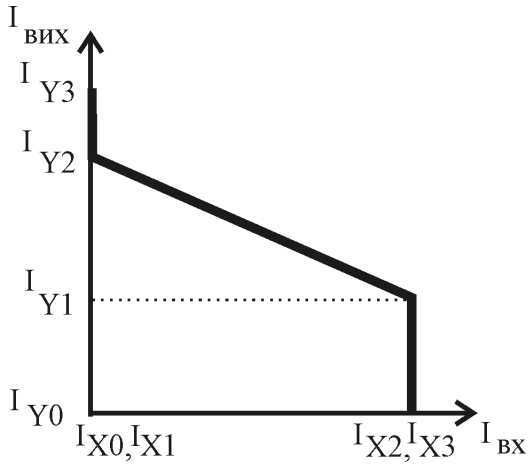


Рисунок 3.2 — Градаційна крива для інвертування зображення і „звуження” градаційного діапазону

Завжди слід мати на увазі, що тонова корекція приводить до втрати інформації у зображенні. При лінійній корекції втрачаються деталі світлих або темних ділянок.

4.2.2 Алгоритм тонової корекції зображення може бути такий.

1. Проводиться аналіз яскравості кожної точки початкового зображення.

2. Задається градаційна крива.

3. Проводиться перерахунок яскравості всіх точок початкового зображення відповідно до заданої градаційної кривої.

4. Формується нове зображення.

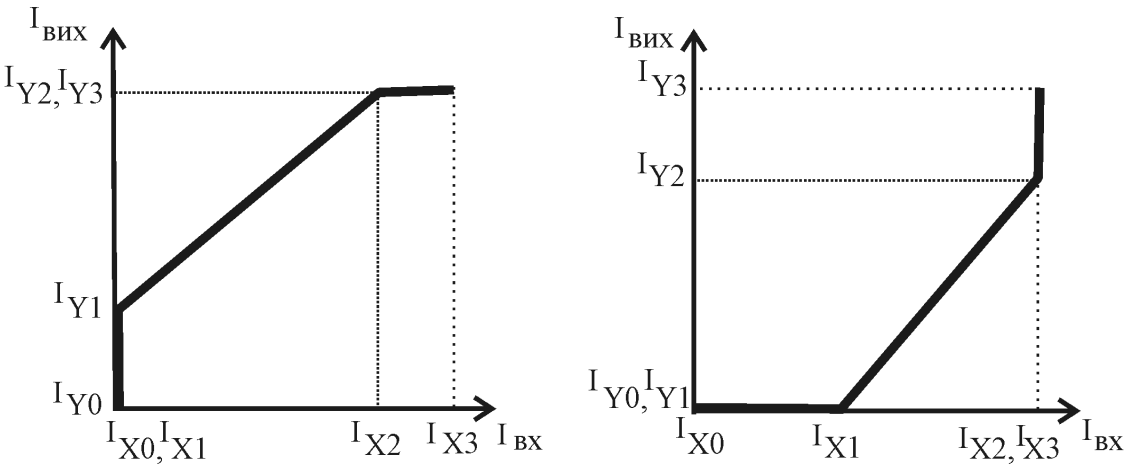


Рисунок 3.3 — Градаційні криві, які дозволяють змінювати яскравість зображення

4.2.3 Алгоритм формування градаційної кривої може бути таким.

1.Задаються значення порогів градаційної кривої I_{X_n}, I_{Y_n} (див. рис. 3.1–3.2).

2. Для кожної області градаційної кривої визначається функція, яка описує зв'язок між початковим та новим значенням яскравості зображення. Так на певних ділянках змінній y присвоюється значення чорного (clBlack), білого (clWhite) кольорів, або певного рівня сірого кольору, у останньому випадку змінній присвоюється значення $y = q + q \cdot 256 + q \cdot 65536$, де q значення сірого (від 0 до 255).

3. Для лінійної функції тонової корекції змінна записується так: $yy = round\left((y - I_{X1}) \cdot (I_{Y2} - I_{Y1}) / (I_{X2} - I_{X1}) + I_{Y1}\right)$, де y — поточна змінна.

4.2.4 При визначенні параметрів тонової корекції слід користуватись гістограмою початкового зображення, яка показує, в якій області яскравостей знаходиться зображення. Для контролю правильності вибраних параметрів слід будувати гістограму відкоригованого зображення.

4.3 Порядок виконання роботи

4.3.1 Запустити програму Delphi 6.

4.3.2 На диску D у папці зі своїм прізвищем відкрити нову папку Lab4, у якій зберегти створюваний проект під назвою Project1.

4.3.3 Розробити програму для тонової корекції зображення.

4.3.3.1 У вікні Form1 розмістити 4 компоненти типу TImage, 4 компоненти типу TButton, 1 компонент типу TOpenPictureDialog, 18 компонентів типу TLabel, 8 компонентів типу TSpinEdit (рис. 3.4).

Компоненти Label1– Label12 мають мати такі назви: „Початкове зображення", „Відкориговане зображення". Компонента Label3 має мати назву „Івх", компонента Label4 — „Івих". Компоненти Label5 – Label8 призначені для назв опорних вхідних величин яскравості, відповідно — „IX0", „IX1", „IX2", „IX3" . Компоненти Label9 – Label12 призначені для назв опорних вихідних величин яскравості, відповідно — „IY0", „IY1", „IY2", „IY3". Компоненти Label13, Label14 мають мати такі назви: „Поточні координати точки". Компоненти Label15, Label16 передбачені для виведення координати x поточної точки на зображенні (див. Лабораторну роботу №3). В Object Inspector для всіх цих компонентів поле Caption потрібно очистити. Розміри компонентів Image1 та Image2 встановити такі: висота – 120, ширина – 161. Розміри

компонентів Image3 та Image4 такі: висота – 128, ширина – 256. Компонент Button1 назвати „Відкрити зображення”, компонент Button2 — „Корекція зображення”, компонент Button3 — „Гістограма 1”, компонент Button4 — „Гістограма 2”.

4.3.3.2 Для першої кнопки оброблювач подій має бути таким:

```
OpenPictureDialog1.Execute;
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
```

При використанні такого оброблювача зображення може бути відкрите з довільної папки.

4.3.3.3 У вікні Unit1 задати змінні, які використовуватимуться у програмі тонової корекції зображення:

```
  c:Tcolor;
 gm, yy:longint;
  i,n,X0,X1,X2,X3,Y0,Y1,Y2,Y3:integer;
g1,g2:array[0..255] of longint;
  y,r,g,b:byte;
```

Змінна c використовується для присвоєння значення параметрів кольору кожної точки, що аналізується. Змінній yy присвоюється значення яскравості у відкоригованому зображенні. Змінні i, n використовуються як лічильники циклів. Змінні *X0, X1, X2, X3, Y0, Y1, Y2, Y3* використовуються для значень яскравостей в опорних точках перехідної характеристики (див. рис. 3.1 – 3.3). Змінній y присвоюється поточне значення яскравості для відкоригованого зображення. Змінним r, g, b присвоюється значення складових кольору для зображення, яке аналізується. Змінні $g1$, $g2$ використовуються для формування масивів під час побудови гістограм початкового та відкоригованого зображень відповідно, змінній gm присвоюється максимальне значення у кожному з масивів.

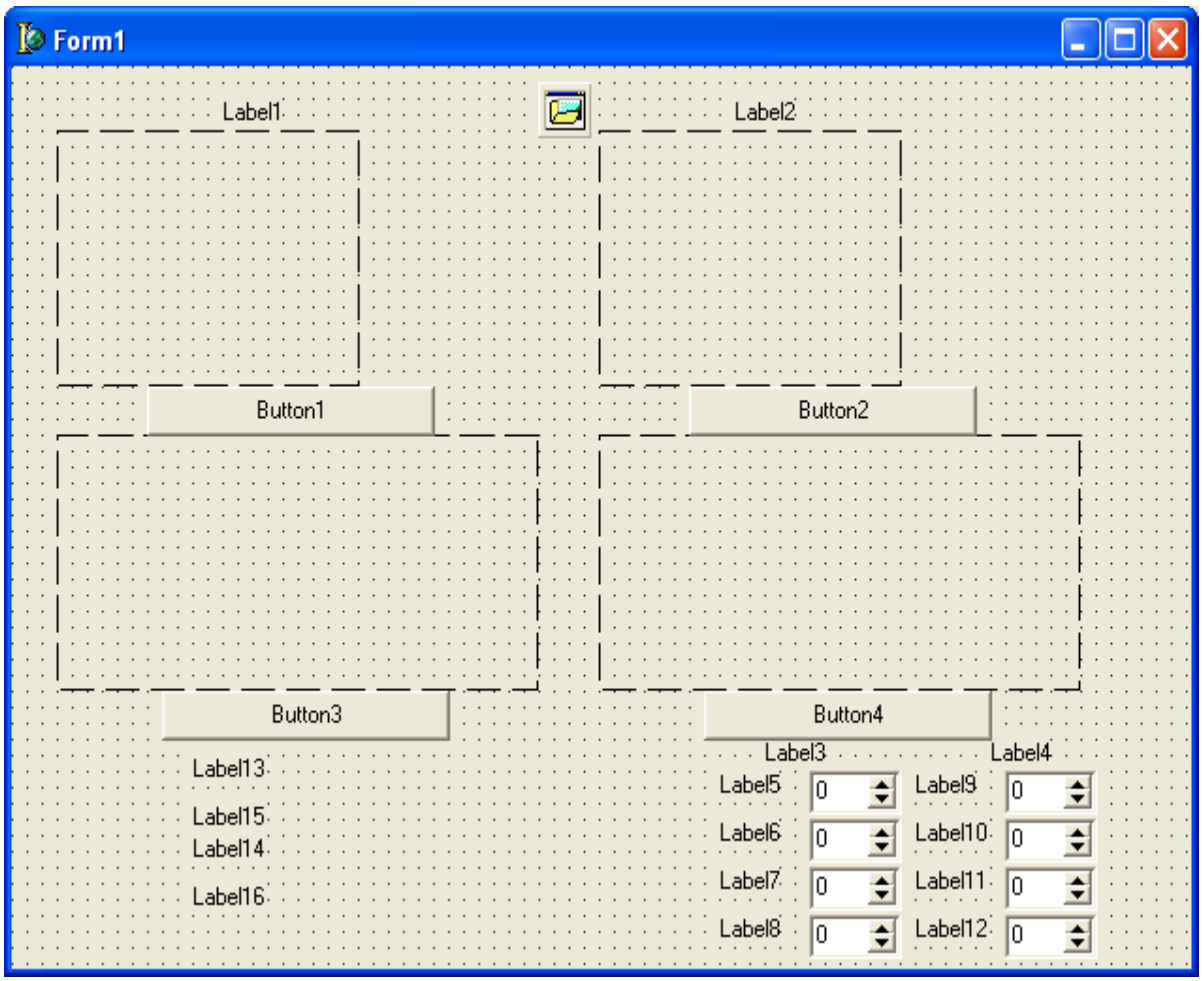


Рисунок 3.4— Можливий вигляд проекту інтерфейсу для тонової корекції зображення

4.3.3.4 Для тонової корекції зображення слід вибрати другу кнопку і написати такий оброблювач подій:

```
for i:=0 to 257 do begin
for n:=0 to 257 do begin
c:=Image1.Canvas.Pixels[n,i];
asm
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
X0:=StrToInt(SpinEdit1.Text);
```

```
X1:=StrToInt(SpinEdit2.Text);
X2:=StrToInt(SpinEdit3.Text);
X3:=StrToInt(SpinEdit4.Text);
Y0:=StrToInt (SpinEdit5.Text);
Y1:=StrToInt(SpinEdit6.Text);
Y2:=StrToInt (SpinEdit7.Text);
Y3:=StrToInt(SpinEdit8.Text);
y:=round(r*0.1+b*0.3+g*0.6);
if y<X1 then Image2.Canvas.Pixels[n,i]:=Y0+Y0*256+Y0*65536;
if y>X2 then  Image2.Canvas.Pixels[n,i]:=Y3+Y3*256+Y3*65536;
if y>=X1 then if y<=X2 then begin yy:=round((y-X1)*((Y2-Y1)/(X2-
X1))+Y1);
Image2.Canvas.Pixels[n,i]:=yy+yy*256+yy*65536;
end;
end;
end;
```

Цей оброблювач подій дозволяє:

— проаналізувати яскравість точок початкового зображення;

— задати параметри перехідної характеристики для тонової корекції зображення;

— розрахувати нові значення для всіх точок відкоригованого зображення відповідно до заданої перехідної характеристики;

— побудувати у другому компоненті Image2 відкориговане зображення.

4.3.3.5 Для аналізу гістограм потрібно створити ще два оброблювачі, які дозволять виводити поточну координату x. З цією метою спочатку для компоненти Image3 у Object Inspector вибрати Events, знайти подію OnMouseMove, двічі натиснути ліву кнопку миші і у вікні Unit1 написати такий оброблювач:

```
c:=Form1.Image3.Canvas.Pixels[x,y];
asm
mov eax,c
mov r,al
end;
Label15.Caption:=concat('x=',IntToStr(x));
```

Для компоненти Image4 аналогічно у Object Inspector вибрати Events, знайти подію OnMouseMove і написати такий оброблювач:

```
c:=Form1.Image4.Canvas.Pixels[x,y];
asm
mov eax,c
mov r,al
end;
Label16.Caption:=concat('x=',IntToStr(x));
```

4.3.3.6 Для компоненти Button3, яка дозволить побудувати гістограму початкового зображення, у відповідності з другою лабораторною роботою потрібно написати такий оброблювач подій:

```
image3.canvas.fillrect(clientrect);
for i:=0 to 255 do
g1[i]:=0;
for i:=0 to image1.picture.height-1 do begin
for n:=0 to image1.picture.Width-1 do begin
c:=image1.Canvas.Pixels[n,i];
asm
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
y:=round(r*0.1+b*0.3+g*0.6);
g1[y]:=g1[y]+1;
end;
end;
gm:=g1[0];
for i:=1 to 255 do
if gm<g1[i] then gm:=g1[i];
if gm=0 then gm:=1;
for i:=0 to 255 do begin
image3.canvas.MoveTo(i,128);
```

```
image3.canvas.lineTo(i,128-round(g1[i]*120/gm));
end;
```

4.3.3.7 Для компоненти Button4, яка дозволить побудувати гістограму відкоригованого зображення, слід написати такий оброблювач подій:

```
image4.canvas.fillrect(clientrect);
for i:=0 to 255 do
g2[i]:=0;
for i:=0 to image2.picture.height-1 do begin
for n:=0 to image2.picture.Width-1 do begin
c:=image2.Canvas.Pixels[n,i];
asm
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
y:=round(r*0.1+b*0.3+g*0.6);
g2[y]:=g2[y]+1;
end;
end;
gm:=0;
for i:=1 to 255 do
if gm<g2[i] then gm:=g2[i];
if gm=0 then gm:=1;
for i:=0 to 255 do begin
image4.canvas.MoveTo(i,128);
image4.canvas.lineTo(i,128-round(g2[i]*120/gm));
end;
```

4.3.4 Запустити створену програму. Відкрити вказане викладачем зображення.

4.3.4.1 Відкрити зображення ‘Hand.bmp’. Побудувати гістограму для цього зображення. Визначити ефективний діапазон вхідного зображення. За цими значеннями задати вхідні параметри для тонової корекції. За вказівкою

викладача задати ефективний діапазон вихідного зображення. Побудувати гістограму для відкоригованого зображення.

4.3.4.2 За параметрами тонової корекції (табл. 3.1) та за даними варіанта (табл. 3.2) встановити види перехідних характеристик. Отримані види перехідних характеристик з вказаними рівнями опорних яскравостей відповідно до варіанта занести до правої колонки табл. 3.1.

4.3.4.3 Для кожної перехідної характеристики вказати ефективні діапазони (вхідний та вихідний) та коефіцієнт перетворення за (3.1).

4.3.5 Скопіювати вигляд інтерфейсу для звіту. Закрити програму. Лістинг програми скопіювати та зберегти як текстовий документ.

Таблиця 3.1

Параметри тонової корекції								Вид перехідної характеристики з врахуванням варіанта завдання
Опорні значення вхідних яскравостей				Опорні значення вихідних яскравостей				
I_{X0}	I_{X1}	I_{X2}	I_{X3}	I_{Y0}	I_{Y1}	I_{Y2}	I_{Y3}	
0	0	255	255	0	0	255	255	
0	0	255	255	255	255	0	0	
0	1	X2	255	0	0	255	255	
0	1	X2	255	0	Y1	255	255	
0	1	X2	255	0	Y1=Y2		255	
0	1	X2	255	0	255	0	255	
0	1	X2	255	255	Y1=Y2		255	
0	0	255	255	255	Y1	Y2	0	
0	1	X2	255	0	255	255	0	
0	1	255	255	0	0	Y2	Y3	
0	0	X2	255	Y0=Y1		Y2	Y3	

4.4Зміст звіту

У звіті наводять:

- назву роботи;
- мету роботи;

- зображення інтерфейсу програми;
- табл. 3.1 із даними свого варіанта відповідно до табл. 3.2;
- перехідні характеристики (вставити в останню колонку табл. 3.1);
- дані п.4.3.4.2;
- лістинг програми;
- висновки.

Таблиця 3.2

№ варіанту	X1	X2	Y1	Y2	Y3
1	50	100	50	190	220
2	100	150	50	180	200
3	150	200	50	220	240
4	40	120	70	180	210
5	60	140	70	180	215
6	80	160	30	180	220
7	100	200	20	200	230
8	90	190	20	200	230
9	70	150	20	190	220
10	60	130	40	210	230
11	60	140	30	180	220
12	50	110	30	195	220

4.5 Контрольні запитання

1. Що таке тонова корекція зображення?
2. Що таке перехідна характеристика тонової корекції?
3. Які є види тонової корекції?
4. Що таке ефективний діапазон перехідної характеристики?
5. Як визначається коефіцієнт перетворення?
6. Який алгоритм тонової корекції?
7. За допомогою якої перехідної характеристики можна інвертувати зображення?
8. Для чого потрібна гістограма при виборі параметрів тонової корекції?

ЛАБОРАТОРНА РОБОТА №5
ВИВЧЕННЯ СПОСОБУ ЦИФРОВОЇ ФІЛЬТРАЦІЇ ЗОБРАЖЕННЯ

5.1 Мета роботи

Розробка алгоритму та відповідного програмного забезпечення для цифрової фільтрації зображення

5.2 Підготовка до виконання роботи

Під час підготовки до роботи необхідно повторити теоретичний матеріал, який стосується фільтрації зображень [3, 4].

5.2.1 Для підвищення якості цифрових зображень, а також для виділення деяких його характеристик використовуються цифрові фільтри (ЦФ).

Розглядаючи деяку локальну область зображення, її елементи можна позначити так:

i-1, j-1	i, j-1	i+1, j-1
i-1, j	i, j	i+1, j
i-1, j+1	i, j+1	i+1, j+1

Така область називається восьми зв’язаною для центрального елементу цієї області.

Принцип цифрової фільтрації базується на тому, що значення яскравості елемента зображення модифікується з урахуванням значень яскравості сусідніх елементів.

Цифровий фільтр для обробки двовимірних даних являє собою не що інше, як матрицю коефіцієнтів, яка переміщується по зображенню, після чого інтенсивність елементів зображення набуває нових значень. У математиці таку матрицю називають матрицею згортки. Цю матрицю називають також маскою згортки, або просто маскою.

Розрізнюють фільтри нижніх та верхніх частот. Поняття частоти може застосовуватись і до зображення, в цьому випадку мова йде про кількість деталей у зображенні.

При фільтрації за допомогою фільтра нижніх частот „розмиваються” ті елементи зображення, які мають інтенсивні перепади яскравості деталей, і тим

самим зображення згладжується. Наприклад, фільтр, який виконує функцію усереднення – є фільтром нижніх частот. Таким чином, використання фільтра нижніх частот призводить до погіршення різкості зображення.

За допомогою фільтра верхніх частот посилюються області з яскраво вираженими перепадами інтенсивності. Фільтри верхніх частот використовуються, коли потрібно підвищити різкість зображення, або виділити контури.

Для виконання фільтрації зображення створюється вікно фільтру з апертурою (розміром) *M* елементів:

$$M = g \cdot q,$$

де $g = 2\alpha + 1$, де α — горизонтальний параметр вікна фільтру (кількість елементів зображення вправо, або вліво від центрального елемента, які будуть змінюватись під час фільтрації);

$q = 2\beta + 1$, де β — вертикальний параметр вікна фільтру (кількість елементів зображення вверх, або вниз від центрального елемента, які будуть змінюватись під час фільтрації).

Якщо $\alpha = \beta = 1$, то розмір квадратного вікна фільтру (маски) становить 3х3. Фільтр з такою апертурою найчастіше використовується на практиці.

Маска фільтру переміщується по зображенню, при цьому у новому зображення, яке формується, нові значення отримуються як добуток початкових значень інтенсивностей та значень коефіцієнтів маски.

При використанні фільтрів на краю зображення виникають крайові ефекти за рахунок того, що вікно фільтра, коли обробляються крайні точки зображення, виходить за розміри зображення. Для усунення цього фактору слід при фільтрації зображення зменшувати область фільтрації на параметр вікна фільтру.

Приклади низькочастотних масок:

проста усереднювальна:

$$h(i,j) = \frac{1}{9} \cdot \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix},$$

маска Гауссового фільтру:

$$h(i,j)=\frac{1}{16}\cdot\begin{vmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{vmatrix},$$

де 9, 16 — сума елементів маски k.

Високочастотна фільтрація. При високочастотній фільтрації обчислюється друга похідна функції інтенсивності, отримана функція інвертується і додається до початкової функції. Так контур може бути прорисований за допомогою такої „загострюючої” маски:

$$h(i,j)=\frac{1}{k}\cdot\begin{vmatrix}-1 & -1 & -1\\ -1 & X & -1\\ -1 & -1 & -1\end{vmatrix}, \text{ або} \qquad (3.2)$$

$$h(i,j)=\frac{1}{k}\cdot\begin{vmatrix}1 & 1 & 1\\ 1 & -X & 1\\ 1 & 1 & 1\end{vmatrix} \qquad (3.3)$$

де k — сума всіх коефіцієнтів маски;

X — параметр, який характеризує ступінь загострення фільтру.

Параметр загострення фільтру визначає, який процент другої похідної додається до початкового зображення, чим вище його значення, тим слабший ефект загострення. Параметр загострення фільтру обчислюється так:

$$X=\frac{100\%}{K_{заг}\%}-1+8, \qquad (3.4)$$

де $K_{заг}$ — ступінь загострення, яка співпадає зі значенням процентної добавки другої похідної.

Слід мати на увазі, що для розрахунку кожного значення під час фільтрації використовуються лише незмінні початкові значення інтенсивностей. Розраховані нові значення інтенсивностей формують нове зображення.

Приклади наведених фільтрів реалізують лінійну фільтрацію

5.2.2 Алгоритм лінійної цифрової фільтрації може бути такий.

1. Проаналізувати параметри кожної точки початкового зображення.

2. Визначити складові кольору для кожної точки.

3. Визначити заданий коефіцієнт фільтру.

4. Розрахувати масив значень усіх колірних складових як суму інтенсивності в центральній точці маски та у кожній оточуючій точці з урахуванням коефіцієнту у цій точці, відповідно до заданих параметрів фільтру.

5. Розрахувати новий масив значень усіх колірних складових як значення масиву за п.4 помножені на величину $\frac{1}{k}$, де k коефіцієнт фільтру.

6. За отриманими колірними складовими сформувати відфільтроване зображення, яке вивести у новий компонент типу TImage.

5.3 Порядок виконання роботи

5.3.1 Запустити програму Delphi 6.

5.3.2 На диску D у папці зі своїм прізвищем відкрити нову папку Lab5, у якій зберегти новий створюваний проект під назвою Project1.

5.3.3 Розробити програму для цифрової фільтрації зображення.

5.3.3.1 У вікні Form1 розмістити 2 компоненти типу TImage, 2 компоненти типу TButton, 1 компонент типу TOpenPictureDialog, 3 компоненти типу TLabel, 9 компонентів типу TSpinEdit (рис.3.5). Компоненти Label1-Label3 мають мати такі назви: "До фільтрування", "Після фільтрування", "Параметри фільтру". Компоненти Button1 та Button2 мають мати назви "Відкрити зображення" та "Відфільтрувати зображення".

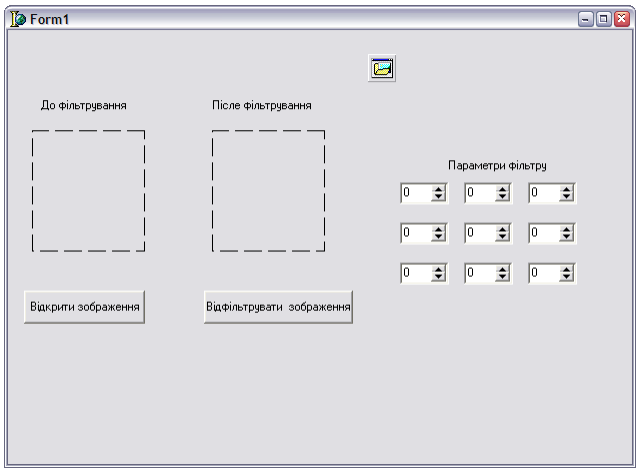


Рисунок 3.5– Можливий вигляд проекту інтерфейсу для фільтрації зображення

5.3.3.2 Для першої кнопки оброблювач подій має бути таким:

```
OpenPictureDialog1.Execute;
Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
```

5.3.3.3 У вікні Unit1 задати змінні, які будуть використовуватись у програмі фільтрації зображення:

```
  c:TColor;
  r,g,b:byte;
  rr2,gg2,bb2:array [0..511,0..511] of byte;
  h:array[1..3,1..3] of integer;
  rr,gg,bb,rr1,gg1,bb1:array[0..511,0..511] of integer;
  i,n,j,k,m:integer;
```

Змінна *c* використовується для присвоєння значення параметрів інтенсивності кольору кожної точки, що аналізується. Змінним r, g, b присвоюється значення складових інтенсивності кольору для зображення, яке аналізується, які утворюють відповідно масиви [*rr*], [*gg*], [*bb*]. Змінні *rr1*, *gg1*, *bb1*, *rr2*, *gg2*, *bb2* утворюють масиви значень складових кольору при фільтрації зображення. Змінна *h* утворює масив значень параметрів фільтру. Змінні *i, n* використовуються як лічильники елементів зображення. Змінні *j, m* використовуються як лічильники параметрів фільтру. Змінній *k* присвоюється значення суми параметрів фільтру.

5.3.3.4 Для написання програми фільтрації зображення слід вибрати другу кнопку і створити такий оброблювач подій:

```
h[1,1]:=SpinEdit1.Value;
h[1,2]:=SpinEdit2.Value;
h[1,3]:=SpinEdit3.Value;
h[2,1]:=SpinEdit4.Value;
h[2,2]:=SpinEdit5.Value;
h[2,3]:=SpinEdit6.Value;
h[3,1]:=SpinEdit7.Value;
h[3,2]:=SpinEdit8.Value;
h[3,3]:=SpinEdit9.Value;
```

```
k:=0;
for j:=1 to 3 do begin
for m:=1 to 3 do begin
k:=k+h[m,j];
end;
end;
if k=0 then k:=1;
for i:=0 to 511 do begin
for n:=0 to 511 do begin
c:=Image1.Canvas.Pixels[i,n];
asm
mov eax,c
mov r,al
mov g,ah
shr eax,8
mov b,ah
end;
rr[n,i]:=r;
gg[n,i]:=g;
bb[n,i]:=b;
end;
end;
for i:=0 to 511 do begin
for n:=0 to 511 do begin
rr1[i,n]:=0;
gg1[i,n]:=0;
bb1[i,n]:=0;
end;
end;
for i:=0 to 511 do begin
for n:=0 to 511 do begin
for j:=1 to 3 do begin
for m:=1 to 3 do begin
rr1[n,i]:=rr1[n,i]+rr[n-1+m,i-1+j]*h[m,j];
gg1[n,i]:=gg1[n,i]+gg[n-1+m,i-1+j]*h[m,j];
bb1[n,i]:=bb1[n,i]+bb[n-1+m,i-1+j]*h[m,j];
end;
```

```
end;
rr1[n,i]:=round(rr1[n,i]/k);
rr2[n,i]:=rr1[n,i];
gg1[n,i]:=round(gg1[n,i]/k);
gg2[n,i]:=gg1[n,i];
bb1[n,i]:=round(bb1[n,i]/k);
bb2[n,i]:=bb1[n,i];
end;
end;
for i:=0 to 511 do begin
for n:=0 to 511 do begin
c:=rr2[n,i]+gg2[n,i]*256+bb2[n,i]*65536;
Image2.Canvas.Pixels[i,n]:=c;
end;
end;
```

Цей оброблювач подій дозволяє:

— отримати яскравість точок початкового зображення;

— задати параметри фільтру;

— розрахувати суму параметрів фільтру;

— розрахувати нові значення для всіх точок нового зображення за параметрами фільтру;

— побудувати у компоненті Image2 відфільтроване зображення.

5.3.4 Запустити створену програму.

5.3.4.1 Відкрити зображення „Hand.bmp”. Задати параметри фільтру відповідно до таблиці 3.3. Спостерігати за отриманим результатом, зробити висновки.

5.3.4.2 Відкрити зображення „11.bmp” (Додаток А). Повторити всі дії відповідно до п.5.3.4.1.

5.3.4.3 Задати параметри фільтру зі ступенем загострення 5%, 10%, 15%, 20%, 30%, 40%, 50% (див. формули (3.2) – (3.4)). Спостерігати отримані зображення, зробити висновки.

5.3.4.4 Скопіювати результати фільтрування зображення для четвертого (див. табл. 3.3) та будь-якого іншого фільтру для звіту.

5.3.5 Скопіювати вигляд інтерфейса для звіту. Закрити програму. Лістинг програми скопіювати та зберегти як текстовий документ.

Таблиця 3.3

Параметри фільтру	Висновки
$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$	
$\begin{vmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{vmatrix}$	
$\begin{vmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{vmatrix}$	
$\begin{vmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{vmatrix}$	
$\begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$	

5.4 Зміст звіту

У звіті наводять:

— назву роботи;

— мету роботи;

— зображення інтерфейсу програми;

— таблицю 5.1 доповнену висновками, які описують характер фільтрації двох зображень;

— лістинг програми;

— висновки.

5.5 Контрольні запитання та завдання

1. Що таке цифрова фільтрація зображення?
2. Які бувають фільтри для обробки цифрових зображень?
3. З якою метою використовується загострюючи фільтри?
4. Як визначаються коефіцієнти маски фільтру для різних видів

фільтрації?

5. Поясніть алгоритми та результати різних видів фільтрації?

6. Що таке вікно фільтру і як його апертура впливає на швидкість обробки зображення?

7. Зображення яких типів модна фільтрувати (монохромні, у шкалі сірих тонів, палітрові, чи повноколірні)?

8. Чим відрізняються алгоритми фільтрації від алгоритмів гістограмної корекції?

ЛАБОРАТОРНА РОБОТА №6
ВИВЧЕННЯ СПОСОБУ ЛІНІЙНОЇ ІНТЕРПОЛЯЦІЇ В ПРОСТОРІ КОЛЬОРІВ

6.1 Мета роботи

Розробка алгоритму та відповідного програмного забезпечення для лінійної інтерполяції в просторі кольорів.

6.2 Підготовка до роботи

Під час підготовки до лабораторної роботи слід повторити теоретичний матеріал, який стосується принципів формування кольору у пристроях відображення інформації та сприйняття кольору людиною [2 – 4].

Світло як носій інформації має дві складові — інформацію про яскравість та інформацію про колір. Роздільна обробка інформації біологічними об’єктами про колір та яскравість є досить непростою проблемою. Колірний зір людини базується на наявності на сітківці ока людини трьох різних типів сенсорних клітин, які можуть розрізнювати червоний, зелений та синій колір. При цьому всі ці сенсорні клітини реагують на інтенсивність світла, яке падає.

Аналогічно, більшість пристроїв виводу колірної інформації базуються на роздільному розпізнаванні червоної, зеленої та синьої складових кольорів. Тобто для відтворення будь-якого кольору потрібно змішувати у певній пропорції ці три складові кольору (див. тема №2). При цьому потрібно мати на увазі, що людина сприймає кольори з різною чутливістю. Встановлено, що найкраще сприймається зелений колір, гірше червоний і найгірше — синій. Це призводить до того, що колірні складові вносять різні внески у відчуття яскравості. Було встановлено, що для більшості людей відчуття яскравості при

сприйнятті кольорових зображень визначається на 59% зеленою складовою, на 30 % червоною складовою і на 11 % — синьою. Отже, коли відомі колірні складові, то через різну чутливість до різних кольорів яскравість зображення буде визначатись за формулою (2.1).

Математично найзручніше представляти колірну систему RGB у виді куба (див. рис. 2.1), де значення кожного кольору задається числами від 0 до $(2^8 - 1)$.

Під час обробки зображення, яка пов'язана зі зміною його розмірів (зменшенні або його збільшенні) іноді доводиться розраховувати яскравості проміжних точок зображення, яких раніше не було. Тоді для отримання якісного зображення використовують інтерполяцію, тобто за значеннями яскравості існуючих двох сусідніх точок проводиться за певними формулами розрахунок яскравості проміжних точок, які будуть додані у зображення. Якщо функція, за якою розраховують інтенсивності проміжних точок є лінійною, то таку інтерполяцію називають лінійною. Наприклад:

$$C_n^i = C_1^i - \frac{C_1^i - C_2^i}{N+1} \cdot n, \qquad (3.5)$$

де N — кількість кроків інтерполяції;

n — поточний номер кроку інтерполяції;

C_1^i — значення яскравості i-го колірного компонента попередньої точки;

C_2^i — значення яскравості i-го колірного компонента наступної точки.

При інтерполяції в колірному просторі остання виконується для всіх складових кольорів окремо, тобто значенню C^i присвоюється значення колірних складових.

У даній лабораторній роботі вивчається алгоритм лінійної інтерполяції в колірному просторі, який має в своїй основі колірну RGB-модель.

Алгоритм такої інтерполяції може бути таким:

1. Задаються колірні складові першого зразка.

2. Визначається адитивний колір першого зразка і виводиться на екран.

3. Задаються колірні складові другого зразка.

4. Визначається адитивний колір другого зразка і виводиться на екран.

5. Розраховуються десять проміжних значень між заданими кольорами за формулою (3.5) і виводяться на екран.

Такий алгоритм може застосовуватись при „фарбуванні” чорно-білих біомедичних зображень, як складова, при їх обробці.

6.3 Порядок виконання роботи

6.3.1 Запустити програму Delphi 6.

6.3.2 На диску D у папці зі своїм прізвищем відкрити нову папку Lab6, у якій зберегти створюваний проект під назвою Project1.

6.3.3 Розробити програму для лінійної інтерполяції в просторі кольорів.

6.3.3.1 У вікні Form1 розмістити 12 компонент типу TPanel, 6 компонент типу ESpinEdit, 6 компонент типу TLabel, 1 компонент типу TButton (рис. 3.6).

Компоненти SpinEdit1 – SpinEdit3 мають мати назви „R0”, „G0”, „B0”, а компоненти SpinEdit4 – SpinEdit6 — „R1”, „G1”, „B1” відповідно. Компонент Button1 має мати назву „Інтерполяція”. Компонент Panel1 має розташовуватись поряд з групою компонентів SpinEdit1 – SpinEdit3, а компонент Panel12 — поряд з групою компонентів SpinEdit4 – SpinEdit6. Ці компоненти використовуватимуться для відтворення кольору першого і другого зразків відповідно. Компоненти Panel2 – Panel11 використовуватимуться для відображення результатів інтерполяції між двома заданими зразками кольорів. Для всіх компонентів типу TPanel в Object Inspector властивість поля Caption очистити.

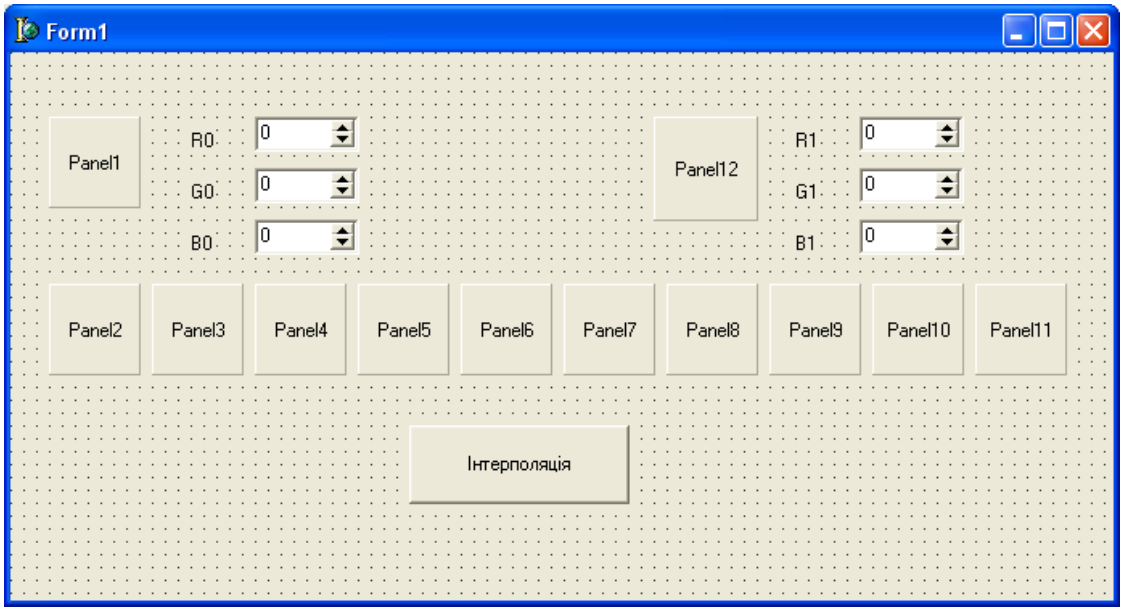


Рисунок 3.6 — Можливий вид інтерфейса при створенні програми лінійної інтерполяції в просторі кольорів

6.3.3.2 У вікні Unit1 задати змінні, які будуть використовуватись у програмі інтерполяції кольорів:

```
w,i:integer;
s:string;
c:tcolor;
r,g,b:array[0..11] of byte;
```

Змінна *w* призначена для формування коду похибки. Змінна *i* використовується як лічильник циклу кількості кроків інтерполяції. Змінній *s* присвоюється символьне значення з компонента типу TSpinEdit. Змінна *c* використовується для присвоєння значення параметрів кольору будь-якій компоненті типу TPanel. Змінним *r*, *g*, *b* присвоюються значення складових кольору у відповідних масивах.

6.3.3.3 Для створення оброблювача при задаванні складових кольору, який має реагувати на ***зміну*** значення у відповідному компоненті типу TSpinEdit (як для першого, так і для другого зразків), потрібно в Object Inspector для кожного компонента вибрати Events і вибрати поле OnChange, двічі натиснути клавішу миші і написати для кожного компонента відповідно такі оброблювачі:

```
s:=SpinEdit1.text;
val(s,r[0],w);
c:=r[0]+256*g[0]+65536*b[0];
panel1.Color:=c;
s:=SpinEdit2.text;
val(s,g[0],w);
c:=r[0]+256*g[0]+65536*b[0];
panel1.Color:=c;
s:=SpinEdit3.text;
val(s,b[0],w);
c:=r[0]+256*g[0]+65536*b[0];
panel1.Color:=c;
s:=SpinEdit4.text;
val(s,r[11],w);
c:=r[11]+256*g[11]+65536*b[11];
panel12.Color:=c;
```

```
s:=SpinEdit5.text;
val(s,g[11],w);
c:=r[11]+256*g[11]+65536*b[11];
panel12.Color:=c;
s:=SpinEdit6.text;
val(s,b[11],w);
c:=r[11]+256*g[11]+65536*b[11];
panel12.Color:=c;
```

Оскільки змінна *s* задається у символьному виді, тому слід використати процедуру перетворення цієї символьної змінної в ціле число, що виконано за допомогою процедури *val*.

6.3.3.4 Для кнопки „Інтерполяція” слід написати такий оброблювач подій:

```
for i:=1 to 10 do begin
r[i]:=round(r[0]-(r[0]-r[11])/11*i);
g[i]:=round(g[0]-(g[0]-g[11])/11*i);
b[i]:=round(b[0]-(b[0]-b[11])/11*i);
c:=r[i]+256*g[i]+65536*b[i];
case i of
1:panel2.Color:=c;
2:panel3.Color:=c;
3:panel4.Color:=c;
4:panel5.Color:=c;
5:panel6.Color:=c;
6:panel7.Color:=c;
7:panel8.Color:=c;
8:panel9.Color:=c;
9:panel10.Color:=c;
10:panel11.Color:=c;
end;
end;
```

6.3.3.5 Запустити створену програму. Створена програма працює так: задається значення складових кольору для першої та для дванадцятої панелі; після натискання кнопки „Інтерполяція” програма виконую інтерполяцію у

просторі кольорів, розраховуючи десять проміжних значень кольору між двома заданими кольорами (перша та дванадцята панелі).

6.3.3.6 Задати значення складових кольору для першого і другого зразків відповідно до табл. 3.4. Виконати інтерполяцію, спостерігати результат і зробити висновки.

Таблиця 3.4

R0	G0	B0	R1	G1	B1	Висновок про отриманий колір першого зразка	Висновок про отриманий колір другого зразка	Висновок про результат інтерполяції
0	0	0	255	255	255			
255	255	255	0	0	0			
90	0	0	230	0	0			
0	85	0	0	240	0			
0	0	95	0	0	250			
255	255	0	0	0	255			
255	255	0	0	0	0			
0	255	255	255	0	0			
255	0	250	0	250	0			

6.3.4 Скопіювати вигляд інтерфейсу для звіту. Закрити програму. Лістинг програми скопіювати та зберегти як текстовий документ.

6.4 Зміст звіту

У звіті наводять:

— назву роботи;

— мету роботи;

— зображення інтерфейса програми;

— таблицю 6.1 доповнену висновками, які описують характер отриманої інтерполяції в колірному просторі;

— лістинг програми;

— висновки.

6.5 Контрольні запитання

1. Що таке інтерполяція зображення?
2. Які є колірні моделі?
3. Що являє собою RGB колірна модель?
4. Яке співвідношення між кольорами при формуванні кольору окремої точки?
5. Як формується колір елемента зображення?
6. У чому суть алгоритму лінійної інтерполяції у колірному просторі?

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

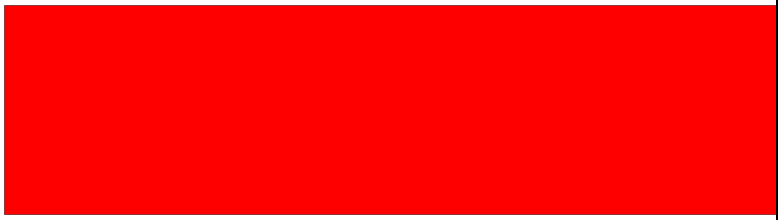
1. С. Бобровский. Delphi 5:Учебный курс.-СПб.: Питер, 2002.-640с.

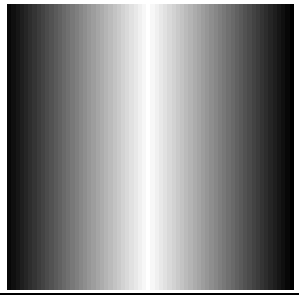
2. Прэтт У. Цифровая обработка изображений. В 2-х т./Пер. С англ. М.Мир, 1982, Т.1 312 с., Т.2 480 с.

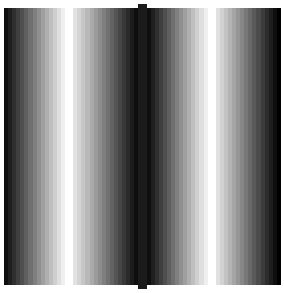
3. Павлидис Т. Алгоритмы машинной графики и обработки зображений.-М.:Радио и связь, 1986.-399 с.

4. Шлихт Г. Цифровая обработка цветных изобюражений. М.:ЭКОМ, 1977.-336 с.

ДОДАТОК А
ВИД ЗОБРАЖЕНЬ, ЯКІ ВИКОРИСТОВУЮТЬСЯ ПРИ ВИКОНАННІ ЛАБОРАТОРНИХ РОБІТ

Назва зображення	Вид зображення
Hand.bmp	Рентгенівське зображення кисті людини 
1.bmp	10 однотонових щаблів 
2.bmp	Плавна зміна кольору у градаціях сірого від чорного до білого 
3.bmp	Червоний колір 

4.bmp	Зелений колір 
5.bmp	Синій колір 
6.bmp	Сірий колір 
7.bmp	Три відтінки сірого 
8.bmp	Зображення для отримання денситограми у вигляді трикутника 

9.bmp	Зображення для отримання денситограми у вигляді двох трикутників 
10.bmp	Кольорове зображення (використовується при вивченні гістограм) 
11.bmp	Зображення геометричних фігур (для вивчення способів фільтрації) 

Електронне навчальне видання

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт

з дисципліни
“МЕТОДИ ОБРОБКИ БІОМЕДИЧНИХ ЗОБРАЖЕНЬ”

для студентів усіх форм навчання
напряму6.051402 «Біомедична інженерія»

Упорядники АВРУНІН Олег Григорович
СКЛЯР Ольга Ігорівна

Відповідальний випусковий А.І. Бих

Авторська редакція