МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНІЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ „ХПІ”

Кафедра „Обчислювальна техніка та програмування”

**ПРОГРАМА ОБРОБКИ ЗОБРАЖЕНЬ**

**Альбом документів курсового проекту по дисципліні**

**"Спеціалізовані комп’ютерні системи"**

**КІТ32.02198.022 ДКП**

Керівник проекту:

\_\_\_\_\_\_\_\_\_\_\_ /Xxxxxxxxxx./

Виконав:

студент групи XXXX

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Xxxxxxxxxx./

„ \_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2007 р.

Харків 2007

**АНОТАЦІЯ**

В даному курсовому проекті розроблена програма обробки зображень, що дає можливість фільтрувати зображення.

**SUMMARY**

In the given course project is designed program processing images, which can image filtering.

##### МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНІЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ „ХПІ”

Кафедра „Обчислювальна техніка та програмування”

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Домнін Ф.А./

„\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2007

Програма обробки зображень

Специфікація

ЛИСТ ЗАТВЕРДЖЕННЯ

##### КІТ32.02198.022 ЛЗ

Керівник проекту:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Xxxxxxxxxx./

Виконав:

студент групи XXXXа

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Xxxxxxxxxx./

„ \_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2007 р.

Харків 2007

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

“ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Кафедра “ Обчислювальна техніка і програмування”

## ЗАТВЕРДЖУЮ

Завідувач кафедрою ОТП

/проф. Домнін Ф.А./

“ “ 2007р.

**Програма обробки зображень**

#### **Технічне завдання**

**КІТ32.02198.022 ТЗ**

#### Розробники

###### Керівник проекту

/Xxxxxxxxxx./

“ ” 2007р.

Виконавець

/ студ. Xxxxxxxxxx./

“ ” 2007р.

Харків 2007

### ЗМІСТ

Вступ

1. Підстави до розробки
2. Призначення до розробки
3. Вимоги до програмного продукту

3.1 Функціональні характеристики

3.2 Склад та параметри технічних засобів

3.3 Інформаційна та програмна сполучність

1. Стадії розробки
2. Програмна документація
3. Порядок контролю та приймання

**ВСТУП**

Програмний продукт під назвою „Програма обробки зображень” призначено для фільтрації зображення.

При створенні програми використовувалось середовище VISUAL STUDIO 2005.

**1 ПІДСТАВИ ДО РОЗРОБКИ**

Розробка проводиться у відповідності до графіку навчального плану на 2007 р. з курсу «Спеціалізовані комп’ютерні системи».

Тема проекту: «Програма обробки зображень».

**2 ПРИЗНАЧЕННЯ РОЗРОБКИ**

Програмний продукт призначений для фільтрації зображення.

**3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ**

**3.1 Функціональні характеристики**

Програмний продукт має забезпечити виконання наступних дій:

* Відкривати зображення різних форматів (GIF, JPEG, PNG,);
* Можливість перегляду зображення
* Вивід результатів;
* Можливість зберігання результатів конвертації.

**3.2 Склад та параметри технічних засобів**

Функціонування програми забезпечується:

ПК х86, з 128 Мб оперативної пам‘яті, встановленою системою Windows 98/МЕ/2000/ХР, та не менш 2 Мб вільного місця на жорсткому диску.

**3.3 Інформаційна та програмна сполучність**

Програмний продукт повинен коректно функціонувати під керуванням ОС Microsoft Windows 98/NT/2000/XP. Програмний продукт має назву „Програма обробки зображень” і повинен бути написаний мовою С#.

**4. Стадії розробки**

* Технічне завдання;
* Технічний проект.

5. ПРОГРАМНА ДОКУМЕНТАЦІЯ

Для розроблюваного об’єкту повинні бути зроблені програмні документи:

- Специфікація;

- Технічне завдання;

- Пояснювальна записка;

**6. ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ**

Розроблений програмний продукт має виконувати всі вимоги, що складаються з перерахованих у п.3.1 характеристиках.

Приймання проводиться керівником проекту й затверджується викладачем.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

## НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

“ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

## Кафедра “Обчислювальна техніка та програмування “

ЗАТВЕРДЖУЮ

Завідувач кафедрою ОТП

\_\_\_\_\_\_\_\_\_\_\_ /Домнін Ф.А./

**Програма обробки зображень**

**Пояснювальна записка**

**ЛИСТ ЗАТВЕРДЖЕННЯ**

##### **КІТ 32.02198.022 ЛЗ**

Розробники

Керівник проекту

\_\_\_\_\_\_\_\_\_/Xxxxxxxxxx./

“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_2007р.

Виконавець

\_\_\_\_\_\_\_\_/студ. Xxxxxxxxxx./

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_2007р.

Харків 2007

**Зміст**

Вступ

1. Призначення та галузь застосування

2.Технічні характеристики

2.3 Фільтрація зображень

2.4 Опис алгоритму і функціонування програми

2.5 Опис та обґрунтування вибору складу технічних та програмних засобів

3. Опис інтерфейсу програми

4. Перевірка функціонування програми

Висновки

Джерела, що використані при розробці

Додаток 1. Лістинг програми

**ВСТУП**

У останній час актуальною задачею є обробка зображень. Доволі часто необхідно фільтрувати зображення.

**1. ПРИЗНАЧЕННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ**

Даний курсовий проект призначений для розробки програми обробки зображень.

Цей проект може бути використана для наукових досліджень або може виступати як посібник.

Курсовий проект розроблений на підставі технічного завдання та методичних вказівок до виконання та оформлення курсового проекту .

**2. ТЕХНІЧНІ ХАРАКТЕРИСТИКИ**

### Фільтрація зображень

Реальні зображення поряд з корисною інформацією містять різні перешкоди. Джерелами перешкод є власні шуми фотоприйомних пристроїв, зернистість фотоматеріалів, шуми каналів зв'язку. Нарешті, можливі геометричні перекручування, зображення може бути розфукусовано. Нехай f (x,y) – деяке зображення, х, у – координати. Реальне растрове зображення має кінцеві розміри: A ≤ x ≤ B, C ≤ y ≤ D і складається з окремих пікселів, розташованих з деяким кроком у вузлах прямокутної сітки. Лінійне перетворення зображення можна описати вираженням

(2)



Вираження (2), де інтегрування ведеться по всій області визначення x і y, характеризує перетворення всього зображення цілком  глобальну фільтрацію. Ядро перетворення h1 (x,y,х',у')в оптику іменують функцією розсіювання крапки (ФРТ). Це зображення крапкового джерела на виході оптичної системи, що вже є не крапкою, а деякою плямою. Відповідно до (2), усі крапки зображення f (х',у') перетворюються в плями, відбувається підсумовування (інтегрування) усіх плям. Не слід думати, що ця процедура обов'язково приводить до розфукусування зображення, навпаки, можна підібрати таку ФРТ, що дозволить сфокусувати розфукусоване зображення.

На мал. 5 представлена одна з можливих ФРТ. Узагалі говорячи, ФРТ визначена на ( ∞ < x < ∞), ( ∞ < y < ∞). ФРТ не повинна змінюватися при зміні початку відліку по x і y, для цього вона повинна мати вигляд: h1(x,y,х',у')= =h1(xх', yу'). Крім того, ФРТ повинна мати осьову симетрію. У цьому випадку всі крапки зображення "розпливаються" однаковим образом, рівномірно в усі сторони (принцип просторової інваріантості).

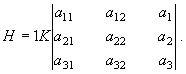


На практиці глобальна фільтрація застосовується рідко. Частіше використовують локальну фільтрацію, коли інтегрування й усереднення проводиться не по всій області визначення x і y, а по порівняно невеликій околиці кожної крапки зображення. Функція розсіювання крапки при цьому має обмежені розміри. Достоїнством такого підходу є гарна швидкодія. Лінійне перетворення приймає вид:

(3)



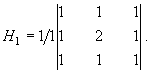
При обробці растрових зображень, що складаються з окремих пікселів, інтегрування замінене підсумовуванням. Найпростіше реалізувати ФРТ кінцевих розмірів у виді прямокутної матриці форматом NN. N може бути рівним 3, 5, 7 і т.д. Наприклад, при N = 3



Підсумовування ведеться по околиці D крапки (i, j); akl  значення ФРТ у цій околиці. Яскравості пікселів f у цій крапці й у її околиці збільшуються на коефіцієнти akl, перетворена яскравість (i,j) го пиксела є сума цих добутків. Елементи матриці задовольняють умові просторової інваріантості, тому a11= a13 =a31= a33, a12 = a21= a23 = a32. Тільки три елементи матриці розміром 3х3 незалежні, у цьому випадку матриця інваріантна щодо поворотів, кратних 90˚. Досвід обробки зображень показує, що відсутність більш строгої осьової симетрії ФРТ слабко позначається на результатах. Іноді використовують 8вугільні матриці, інваріантні щодо поворотів на 45˚.

Фільтрація згідно (3) здійснюється переміщенням ліворуч праворуч (або зверху вниз) маски на один пиксел. При кожнім положенні апертури виробляються згадані вище операції, а саме перемножування вагових множників akl з відповідними значеннями яскравості вихідного зображення і підсумовуванням добутків. Отримане значення привласнюється центральному (i,j) му пікселу. Звичайно це значення поділяється на заздалегідь задане число K (нормуючий множник). Маска містить непарне число рядків і стовпців N, щоб центральний елемент визначався однозначно.

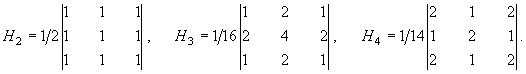
Розглянемо деякі фільтри, що згладжують шум. Нехай маска розміром 3х3 має вигляд:



Тоді яскравість (i,j) -го піксела після фільтрації визначиться як



Хоча коефіцієнти akl можна вибрати зі середньоквадратичного або іншої умови близькості не перекрученого шумом si,j і перетвореного gi,j зображень, звичайно їхній задають евристично. Приведемо ще деякі матриці фільтрів, що придушують шум:



У фільтрів H1  H4 множники, що нормують, K підібрані таким чином, щоб не відбувалося зміни середньої яскравості обробленого зображення. Поряд з масками 3х3 використовуються маски більшої розмірності, наприклад, 5х5, 7х7 і т.п. На відміну від фільтра H2, у фільтрів H1, H3, H4 агарні коефіцієнти на перетинанні головних діагоналей матриці більше, ніж коефіцієнти, що коштують на периферії. Фільтри H1, H3, H4 дають більш плавна зміна яскравості по зображенню, чим H2.

Нехай відліки корисного зображення fk,m мало міняються в межах маски. На зображення накладається аддітівный шум: fk,m + nk,m , відліки шуму nk,m випадкові і незалежні (або слабко залежні) зі статистичної точки зору. У цьому випадку механізм придушення шуму з використанням приведених фільтрів полягає в тому, при підсумовуванні шуми компенсують один одного. Ця компенсація буде відбуватися тим краще, чим більше число членів у сумі, тобто чим більше розмір (апертура) маски. Нехай, наприклад, використовується маска NN, у межах її корисне зображення має постійну яскравість f, шум з незалежними значеннями відліків nk,m, середнім значенням м = 0 і дисперсією у2 у межах маски (такий шум називають білим). Відношення квадрата яскравості (i,j)-го піксела до дисперсії шуму, тобто відношення сигнал/шум, дорівнює f 2/у2.

Розглянемо, наприклад, маску типу H2:



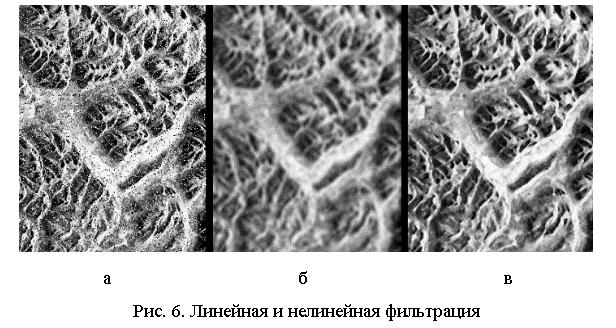
Середній квадрат яскравості дорівнює f 2, середній квадрат інтенсивності шуму



Подвійна сума відповідає k= p, m = q, ця сума дорівнює у2/N 2.Чотириразова сума дорівнює нулеві, тому що відліки шуму при k ≠ p, m ≠ q незалежні: <n k,mn p,q> = 0. У результаті фільтрації відношення сигнал/шум стає рівним N 2f 2/у2, тобто зростає пропорційно площі маски. Відношення яскравості (i,j)-го піксела корисного зображення до середньоквадратичного відхилення шуму зростає пропорційно N. Застосування маски 3х3, у середньому, підвищує відношення сигналу до шуму в 9 разів.

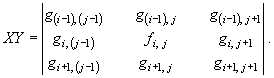
При імпульсній перешкоді механізм придушення полягає в тому, що імпульс "розпливається" і стає мало помітним на загальному тлі.

Однак часто в межах апертури значення корисного зображення все-таки змінюються помітним образом. Це буває, зокрема, коли до маски попадають контури. З фізичної точки зору, усі H1  H4 є фільтрами нижніх частот (зсереднюючими фільтрами), що придушують високочастотні гармоніки і шуму, і корисного зображення. Це приводить не тільки до ослаблення шуму, але і до розмивання контурів на зображенні. На мал. 6а показане вихідне зашумлене зображення, результат застосування фільтра типу Н2 приведений на мал. 6б (маска 5х5).



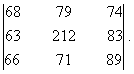
Розглянута вище фільтрація характеризувалася тим, що вихідні значення фільтра g визначалися тільки через вхідні значення фільтра f . Такі фільтри називаються не рекурсивними. Фільтри, у яких вихідні значення g визначаються не тільки через вхідні значення f , але і через відповідні вихідні значення, називаються рекурсивними.

При рекурсивній фільтрації можна зберігати ті ж значення вагових множників, що і приведені вище, існують рекурсивні фільтри зі спеціально підібраними множниками. Елементи вхідного зображення в межах вікна зміняться і приймуть вид:



Вагарні і що нормує множники рекурсивного фільтра залежать від місця розташування маски; рекурсивний локальний фільтр дозволяє враховувати усі вхідні значення фільтруємого зображення, тобто наближається по своїй дії до глобального фільтра.

Для усунення ефекту розмивання контурів при придушенні шуму варто переходити до нелінійної обробки. Прикладом нелінійного фільтра для придушення шуму служить медіанний фільтр. При медіанній фільтрації (i,j)-му пікселу привласнюється медіанне значення яскравості, тобто таке значення, частота якого дорівнює 0,5. Нехай, наприклад, використовується маска 33, у межі якої поряд з більш-менш рівномірним тлом потрапив шумовий викид, цей викид довелося на центральний елемент маски:



Будується варіаційний ряд. Варіаційним рядом V1,..., Vn вибірки f1,..., fn називають упорядковану по не убуванню послідовність елементів вибірки, тобто V1 = min(f1,..., fn), Vn= max (f1,..., fn) і т.д. У нашому випадку варіаційний ряд має вигляд: 63, 66, 68, 71, 74, 79, 83, 89, 212. Тут медіанне значення  п'яте по рахунку (підкреслене), тому що всього чисел у ряді 9. При медіанній фільтрації значення 212, перекручене шумовим викидом, заміняється на 74, викид на зображенні цілком подавлений. Результат застосування медіанного фільтра показаний на мал. 6в.

Функція розсіювання крапки для медіанного фільтра є нуль. При розмірах вікна (2k+1)·(2k+1) відбувається повне придушення перешкод, що складаються не більш, ніж з 2(k2 + k) пікселів, а також тих, котрі перетинаються не більш, з k рядками або k стовпцями, При цьому не змінюється яскравість у крапках тла. Зрозуміло, при медіанній фільтрації може відбуватися перекручування об'єкта на зображенні, але тільки на границі або поблизу неї, якщо розміри об'єкта більше розмірів маски. Фільтр має високу ефективність при придушенні імпульсних перешкод, однак ця якість досягається підбором розмірів маски, коли відомі мінімальні розміри об'єктів і максимальні розміри перекручених перешкодою локальних областей.

### Виділення контурів

Лінійні фільтри можуть бути призначені не для придушення шуму, а для підкреслення перепадів яскравості і контурів. Виділення вертикальних перепадів здійснюється диференціюванням по рядках, горизонтальних  по стовпцях. Диференціювання виробляється в цифровій формі:



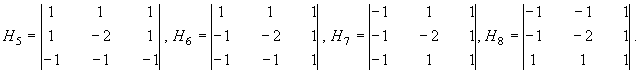
Тут ∆x=1 ? збільшення уздовж рядка, рівне 1 пікселу,

?y=1 ? збільшення уздовж стовпця, також рівне 1 пікселу.

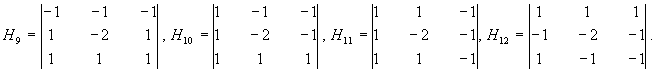
Виділення перепадів по діагоналі можна одержати, обчислюючи різниці рівнів діагональних пар елементів.

Для виділення перепадів використовуються наступні набори вагових множників, що реалізують двовимірне диференціювання:

Північ північний схід схід південний схід



південь південний захід захід північний захід

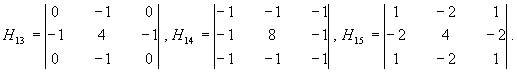


Назва географічних напрямків говорить про напрямок схилу перепаду, що викликає максимальний відгук фільтра. Сума вагових множників масок дорівнює нулеві, тому на ділянках зображення з постійною яскравістю ці фільтри дають нульовий відгук.

Виділення горизонтального перепаду можна виконати також шляхом обчислення збільшення різниці яскравості пікселів уздовж рядка, що рівноцінно обчисленню другої похідної по напрямку (оператор Лапласа):



Це відповідає одномірній масці Н = | 1 2 1|, сума вагових множників дорівнює нулеві. У такий же спосіб можна шукати перепади по вертикалі і по діагоналі. Для виділення перепадів без обліку їхньої орієнтації використовуються двовимірні оператори Лапласа:



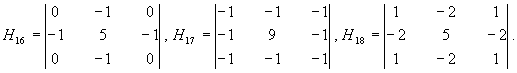
Тут сума вагових множників також дорівнює нулеві. На мал. 7 приведене вихідне зображення і результат застосування оператора Лапласа Н13.

Оператори Лапласа реагують на перепади яскравості у виді східчастого перепаду і на "дахоподібний" перепад. Вони також виділяє ізольовані крапки, тонкі лінії, їхні кінці і гострі кути об'єктів. Лінія підкреслюється в 2 рази яскравіше, ніж східчастий перепад, кінець лінії в 3 рази, а крапка  у 4 рази яскравіше. Оператор Лапласа не інваріантний до орієнтації перепадів: наприклад, відгук оператора на похилий перепад у діагональному напрямку майже вдвічі більше, ніж у горизонтальному і вертикальному.

З фізичної точки зору, фільтри H5  H15 є фільтрами верхніх частот, вони виділяють високочастотні складові корисного зображення, відповідальні за перепади яскравості і контури і придушують "постійну складову". Однак при їхньому використанні рівень шуму на зображенні зростає.

Фільтри для виділення перепадів і границь, як і фільтри H1  H4 для придушення шуму, можуть бути рекурсивними.

Зображення з підкресленими границями (контурами) суб'єктивно сприймається краще, ніж оригінал. Відбувається фокусування частково розфукусованого зображення. При використанні оператора Лапласа для цих цілей застосовують три типових набори вагарень множників:

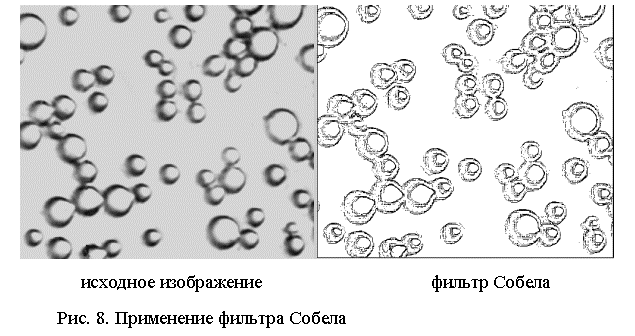


H16  H18 відрізняються від фільтрів H13  H15 тим, що до центрального елемента матриці додана 1, тобто при фільтрації вихідне зображення накладається на контур.

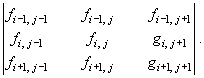
Для виділення контурів і перепадів яскравості можуть застосовуватися нелінійні фільтри. У нелінійних алгоритмах використовують нелінійні оператори дискретного диференціювання. У фільтрі Робертса використовується маска, що переміщається 2:



диференціювання виробляється за допомогою одного з виражень



У фільтрі Собела використовується вікно 33:



Центральному (i,j) -му пікселу замість f i,j привласнюється значення яскравості або , де



На мал. 8 показані результати застосування фільтра Собела.

**2.4. Опис та обґрунтування вибору складу технічних та програмних засобів**

Згідно з поставленою задачею оберемо технічні та програмні засоби реалізації програми. Програма написана на мові програмування високого рівня (С#). Дана програма розрахована на операційні системи Windows 2000, NT, XP з встановленою середою .NET 2.0 чи 3.0.

**3. Опис інтерфейсу програми**

Програма має зручний та доступний інтерфейс для роботи з користувачем. На рис. 1 показано вигляд вікна спілкування з користувачем до початку роботи з програмою.

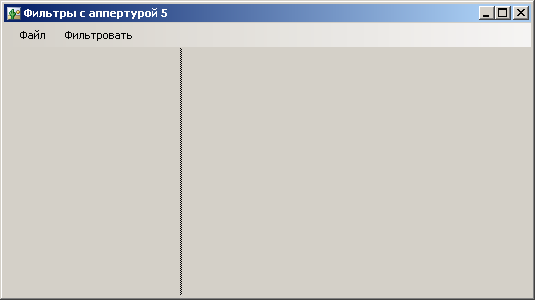


Рис.1. Зображення діалогового вікна програми.

За допомогою меню „Файл” користувач має можливість відкрити файл з зображенням. У першій частині робочого вікна ми маємо змогу бачити не фільтроване зображення. Також меню „Файл” надає можливість зберегти файл під вибраним чи іншим ім’ям.

Меню „Фильтровать” надає змогу відфільтрувати зображення. Він має наступні підменю

1. Лапласіан

2. Гаусовий

3. Smoothing

4. Довільне ядро

Після обрання будь-якого пункту меню на екран виводиться вікно з ядром

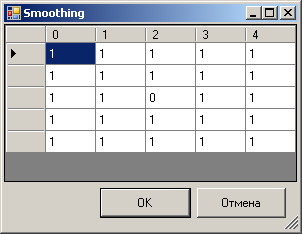


Рис 2. Вікно з ядром

При натисненні на кнопку „ОК” ми бачимо вікно, що виводиться під час обрахунків.

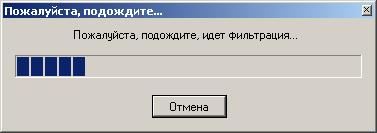


Рис.3 Вікно, що повідомляє про проходження обрахунків

Після виконання обрахунків ми бачимо на головному вікні у другій його частині фільтроване зображення, відповідно до обраної фільтрації.

Також, користувач має можливість власноруч заповнити матрицю з коефіцієнтами.

Далі наведені результати роботи програми з різними фільтрами:

**4. Перевірка функціонування програми**

Програма повністю відповідає наведеному вище алгоритму.

**ВИСНОВКИ**

Розробивши програмну модель обробки файлів зображень, я дійшов висновку, що найбільша складність у подібних задачах є розробка досить універсального алгоритму та оформлення результатів (введення та вивід).

##### **ДЖЕРЕЛА, ЩО ВИКОРИСТАНІ ПРИ РОЗРОБЦІ**

1. В.Б. Жушма, Ф. Колокольцев, Р.Коринь, Ю. Мевшин, Т. Мовчан, Л.Д. Писаренко, Ю. Путилов. Роздільне унітарне перетворення зображень, УЖ МТиТ 1999 № 2-3.
2. В,Б, Жушма, Ф. Колокольцев, Р,Корннь, Ю, Мевшнн, Т.В, Мовчан, Л.Д. Писаренко, Ю. Путилов. Сингулярне перетворення зображень, УЖ МТиТ 1999 №4.
3. http://msdn2.microsoft.com/ru-ru/default.aspx

**ДОДАТОК1**

**ЛІСТИНГ ПРОГРАМИ**

MainForm.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace Kursach

{

public partial class MainForm : Form

{

private string filename = "";

WaitForm win = null;

public MainForm()

{

InitializeComponent();

}

private void ExitToolStripMenuItem\_Click(object sender, EventArgs e)

{

this.Close();

}

private void OpenToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (openFileDialog.ShowDialog() == DialogResult.OK)

picture.Image = new Bitmap(filename = openFileDialog.FileName);

}

private void SaveAsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

picture.Image.Save(filename = saveFileDialog.FileName);

}

}

private void SaveToolStripMenuItem\_Click(object sender, EventArgs e)

{

picture.Image.Save(filename);

}

private void лапласианToolStripMenuItem\_Click(object sender, EventArgs e)

{

Start<Laplacian>(false);

}

public void Start<T>(bool edit) where T : IKernel, new()

{

IKernel kernel = new T();

Bitmap bitmap = new Bitmap(picture.Image);

if (new KernelForm(false, kernel).ShowDialog() == DialogResult.OK)

{

win = new WaitForm(backgroundWorker);

win.Show(this);

backgroundWorker.RunWorkerAsync(new StartParam(bitmap, kernel));

}

}

private void smoothingToolStripMenuItem\_Click(object sender, EventArgs e)

{

Start<Smoothing>(false);

}

private void гаусовыйToolStripMenuItem\_Click(object sender, EventArgs e)

{

Start<Gaussian>(false);

}

private void CustomToolStripMenuItem\_Click(object sender, EventArgs e)

{

Start<Unknown>(true);

}

private void backgroundWorker\_DoWork(object sender, DoWorkEventArgs e)

{

StartParam s = (StartParam)e.Argument;

pictureF.Image = Filter.Filtering(s.b, s.k.Kernel , backgroundWorker);

}

private void backgroundWorker\_ProgressChanged(object sender, ProgressChangedEventArgs e)

{

win.progressBar.Value = e.ProgressPercentage;

}

private void backgroundWorker\_RunWorkerCompleted(object sender, RunWorkerCompletedEventArgs e)

{

win.Close();

}

}

class StartParam

{

public Bitmap b = null;

public IKernel k = null;

public StartParam(Bitmap b, IKernel k)

{

this.b = b;

this.k = k;

}

}

}

WaitForm.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace Kursach

{

public partial class WaitForm : Form

{

BackgroundWorker bw;

public WaitForm(BackgroundWorker bw)

{

InitializeComponent();

this.bw = bw;

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void WaitForm\_FormClosing(object sender, FormClosingEventArgs e)

{

if (bw.IsBusy)

bw.CancelAsync();

}

}

}

KernelForm.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace Kursach

{

public partial class KernelForm : Form

{

private bool allowEdit = false;

public bool AllowEdit

{

get { return allowEdit; }

set

{

allowEdit = value;

kernelgrid.ReadOnly = !value;

}

}

public KernelForm(bool editable, IKernel kernel)

{

InitializeComponent();

for (int i = 0; i < kernel.Kernel.GetUpperBound(0) + 1; i++)

{

kernelgrid.Rows.Add();

for (int j = 0; j < kernel.Kernel.GetUpperBound(1) + 1; j++)

{

kernelgrid.Rows[i].Cells[j].Value = kernel.Kernel.GetValue(i, j);

}

}

this.Text = kernel.Text;

AllowEdit = editable;

}

private void butCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

private void butOK\_Click(object sender, EventArgs e)

{

DialogResult = DialogResult.OK;

this.Close();

}

}

}

Filter.cs

using System;

using System.Collections.Generic;

using System.Text;

using System.Drawing;

using System.ComponentModel;

namespace Kursach

{

static class Filter

{

public static Bitmap Filtering(Bitmap img, int[,] kernel, BackgroundWorker bw)

{

int[,] Mk = (int[,])kernel.Clone();

int[] H = new int[kernel.Length];

int counter = 0;

for (int i = 0; i <= kernel.GetUpperBound(0); i++)

for (int j = 0; j <= kernel.GetUpperBound(1); j++)

H[counter++] = kernel[i, j];

Bitmap ret = new Bitmap(img);

int norma = Max(Sum(H), 1);

for (int i = 2; i < img.Width - 2; i++)

{

for (int j = 2; j < img.Height - 2; j++)

{

int[] V = new int[25];

for (int k = 0; k < V.Length; k++)

V[k] = img.GetPixel(i - 1 + k / 5 - 1, j - 1 + k % 5 - 1).ToArgb();

long mul = MulScalar(V, H);

double d =(double)1 / (double)norma;

double px = (d \* mul);

ret.SetPixel(i, j, Color.FromArgb((int)px));

}

bw.ReportProgress(100 \* i / img.Width);

if (bw.CancellationPending)

return ret;

}

return ret;

}

private static int Sum(int[] arr)

{

int res = 0;

foreach (int i in arr)

res += i;

return res;

}

private static int Max(int first, int second)

{

if (first > second)

return first;

else

return second;

}

private static long MulScalar(int[] first, int[] second)

{

long res = 0;

for (int i = 0; i < first.Length; i++)

res += (first[i] \* second[i]);

return res;

}

}

}

Kernel.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Kursach

{

public interface IKernel

{

int[,] Kernel

{

get;

}

string Text

{

get;

}

}

}

Kernels.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Kursach

{

class Gradient : IKernel

{

#region IKernel Members

public int[,] Kernel

{

get

{

return new int[,]

{ { 0, -1, 0, 1, 0},

{-1, -2, 0, 2, 1},

{-1, -2, 0, 2, 1},

{-1, -2, 0, 2, 1},

{0, -1, 0, 1, 0} };

}

}

public string Text

{

get { return "Градиентный"; }

}

#endregion

}

class Laplacian : IKernel

{

#region IKernel Members

public int[,] Kernel

{

get

{

return new int[,]

{ { -1, -1, -1, -1, -1},

{-1, -1, -1, -1, -1},

{-1, -1, 24, -1, -1},

{-1, -1, -1, -1, -1},

{-1, -1, -1, -1, -1}}; ;

}

}

public string Text

{

get { return "Лапласиана"; }

}

#endregion

}

class Smoothing : IKernel

{

#region IKernel Members

public int[,] Kernel

{

get

{

return new int[,]

{ { 1, 1, 1, 1, 1},

{1, 1, 1, 1, 1},

{1, 1, 0, 1, 1},

{1, 1, 1, 1, 1},

{1, 1, 1, 1, 1}};

}

}

public string Text

{

get { return "Smoothing"; }

}

#endregion

}

class Gaussian : IKernel

{

#region IKernel Members

public int[,] Kernel

{

get

{

return new int[,]

{{ 1, 2, 4, 2, 1},

{2, 4, 8, 4, 2},

{4, 8, 16, 8, 16},

{2, 4, 8, 4, 2},

{1, 2, 4, 2, 1}};

}

}

public string Text

{

get { return "Гаусовый"; }

}

#endregion

}

class Unknown : IKernel

{

#region IKernel Members

public int[,] Kernel

{

get

{

return new int[,]

{ { 0, 0, 0, 0, 0},

{0, 0, 0, 0, 0},

{0, 0, 0, 0, 0},

{0, 0, 0, 0, 0},

{0, 0, 0, 0, 0}};

}

}

public string Text

{

get { return "Произвольный"; }

}

#endregion

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Windows.Forms;

namespace Kursach

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainForm());

}

}

}