Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпеченя

Мінаєв Сергій Дмитрович,
студент групи РФ-181

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка алгоритму ідентифікації цифрових зображень

Спеціальність:
122 Комп’ютерні науки

Спеціалізація, освітня програма:
Програмне забезпечення систем захисту інформації

Керівник:
Трифонова Катерина Олексіївна,
ст. викладач

Одеса – 2022

Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпеченя
Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 – Комп'ютерні науки
Освітня програма – Програмне забезпечення систем захисту інформації

ЗАТВЕРДЖУЮ
Завідувач кафедри КБПЗ

_____
д.т.н., проф. А.А.Кобозєва
____ _____2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Мінаєву Сергію Дмитровичу

1.Тема роботи: *Розробка Алгоритму ідентифікації цифрових зображень* ,
керівник роботи *Трифонова Катерина Олексіївна, ст. викладач,*
затверджені наказом ректора № 168-в від „17" 05.2022р.
2.Зміст роботи*: формування зображення, ідентифікфація цифрових зображень, практична реалізація алгоритму ідентифікації цифрових зображень, охорона праці.*
3. Перелік ілюстративного матеріалу: *Схема відображення видимого предмета на сітківці ока, конвеєр зображення для цифрової камери, мозаїка Байера, структура процеса формування зображення обличчя, блок схема керованих генерацій зображень, Модель TL-GAN прихованого простору для керування процесом генерації, схема знаходження еталонного шуму камери, зображення шумов, графіки тестування на шуми, знімки екрану інтерфейсу програми.*

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
Охорона праці	к.т.н, доцент Ярова І.А.	Завдання видав	Завдання прийняв

5. Дата видачі завдання "_____"_____2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз джерел з теми випускної кваліфікаційної роботи*	*15.11.2021*	*виконано*
2	*Обґрунтування вибору рішення. Збір даних*	*15-12-2021*	*виконано*
3	*Аналіз основних аспектів дослідження зображень*	*11-01-2022*	*виконано*
4	*Аналіз умов використання методів дослідження*	*20-02-2022*	*виконано*
5	*Розроблення власного методу розрахунку шуму зображення*	*30-03-2022*	*виконано*
6	*Підготовка тексту роботи*	*11-05-2022*	*виконано*
7	*Підготовка презентації та доповіді*	*15-05-2022*	*виконано*
8	*Попередній захист*	*20-05-2022*	*виконано*
9	*Нормоконтроль, рецензування*	*20-05-2022*	*виконано*
10	*Занесення роботи в електронний архів*	*28-05-2022*	*виконано*
11	*Допуск до захисту у завідувача кафедри*	*01-06-2022*	*виконано*

Здобувач вищої освіти _____ Мінаєв С. Д.

Керівник роботи _____ Трифонова К. О.

ЗАВДАННЯ

на розробку розділу “Охорона праці”

Мінаєву Сергію Дмитровичу,

група РФ-181

Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій

Кафедра кібербезпеки та програмного забезпечення

Тема роботи *Робоче місце інженера-програміста IT-Matlab*

Зміст розділу:

1 Аналіз умов праці і вибір основних заходів виробничої безпеки.

2 Аналіз пожежної безпеки. Вибір заходів та засобів пожежної безпеки.

Керівник роботи

_____ (Трифонова К. О.)

«____» _____ 2022 р.

Консультант з охорони праці

_____(Ярова І.А)

«____» _____ 2022 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему “Розробка алгоритму ідентифікації цифрових зображень” на здобуття першого (бакалаврського) рівня вищої освіти за спеціальністю 122 - Комп’ютерні науки, спеціалізація, освітня програма: Програмне забезпечення систем захисту інформації, містить 17 рисунків, 1 таблицю, 1 додаток, 11 літературних джерел за переліком посилань. Робота виконана на 72 сторінках загального тексту і 44 сторінках основного тексту.

Метою роботи було розпізнавання штучно згенерованих зображень за допомогою глибоких нейронних мереж стосовно реальних фотографій.

Результати даної роботи можуть бути використані для індивідуального використовування особою для ідентифікації зображень.

КОМП’ЮТЕРНІ НАУКИ, ЗОБРАЖЕНЯ, ІДЕНТИФІКАЦІЯ, MATLAB

ANNOTATION

Qualification work on " Development of Digital Image Identification Algorithm " for the first (bachelor's) level of higher education in the specialty 122 - Computer Science, specialization, educational program: Information security systems software, contains 17 figures, 1 table, 1 appendix, 11 references according to the list of references. The work is performed on 72 pages of general text and 44 pages of main text.

The aim of the work was to recognize artificially generated images using deep neural networks in relation to real photographs.

The results of this work can be used for individual use by a person to identify images.

COMPUTER SCIENCES, IMAGES, IDENTIFICATION, MATLAB

ЗМІСТ

ВСТУП

Людина сприймає інформацію навколишнього світу декількома органами почуттів, але головна та основна інформація приходить до нього через світловий потік, що формує у нього зображення зовнішнього світу. Також людина реєструє світлову інформацію на цифрові камери та смартфони. Обробка інформації, що надійшла у вигляді зображення, стає головним завданням на сьогоднішній день.

В сучасний час компьютерних та телекомуніціних технологій набуває розвиток та зумовлює використання цифрових методів обробки сигналів особливого розвитку, зараз набувають методи цифрової обробки зображень, тому що вони, складають основну частину трафіку мультимедійних мереж, методи цифрової обробки зображень проникли майже у всі області інформаційного кола людини. Серед них можливо виділити два методи котрі набули широкого втілення, такі як компрессія цифрового зображення та фільтрація шумів зображення.

В наш час метод фільтрації шумів зображення набув дуже великого значення, тому що це пов’язана в чіткості зображення. Фільтрація шумів сьогодні використовується у військовій справі, у кримінальних справах, у цифровому телебаченні для того щоб підвищити чіткість зображення.

Пісня аналізу матеріалів які висвітлюють появу шумів у цифрових камерах та появу несправжних зображень облич людей за домомогою нейроних мереж, це лягло підставою та вихідними даними для розробки теми ідентифікації зображення. Метою ідентифікації зображення буде ідентифікація спражнього зображення, яке зроблено цифровою камерою або ідентифікація несправжного зображення сдобутого за допомогою нейроних мереж. Актуальність цієї теми дуже важлива, тому що в світі поширюється втілення роботів які віконують дію замість людини.

Ця робота може бути допомогою для розпізнавання штучно створених

фотографій та фільтрації шумів на основі сингулярного розкладання, може застосовуватися як для військової сфери так і у кримінальних справах.

Фільтрацію зображення можливо виконати за допомогою медіаного фільтру і це зараз дуже поширено в різних газузях.

Відштовхуячись від метода фільтрації зображення і бравши його першим в розробці та реалізації данної роботи.

Тому, метою даної роботи є розробка алгоритму ідентифікації цифрових зображень на основі сінгулярного роскладання зображень.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

а) Пошук 3 різних видів генерації штучних зображень у кількості 50 шт.

б) Зробити обробку усередненого шума зображення з двома методами.

в) Зробити обробку шуму окремого зображення.

г) Розробити алгоритм кореляції шумів зображення.

д) З'ясувати який із методів ефективний

Процес дослідження– процес сингулярного розкладання зображення.

Обьектом дослыдження э цифрове зображення.

Робота складається із вступу, чотирьох розділів, висновку, переліку посилань та додатків.

У вступі розглядаеться акутальныcть питання та застосування теми роботи.

У першому розділі розглядаеться формування зображення за допомогою ока та цифрової камери.

У другому розділі розглядаеться ідентифікфація цифрових зображень.

У третьому розділі розглядаеться практична реалізація алгоритму ідентифікації цифрових зображень.

В четвертому розділі розглядаеться охорона праці.

Була публікація на тему: Обгляд сучасних механізмів формування зображення в журналі INTERNATIONAL SCIENCE JOURNAL, May 2022 ,Part 2.

1 ФОРМУВАННЯ ЗОБРАЖЕННЯ

1.1 Формування зображення зоровим сприйняттям людини

Більшість інформації людина приймає на власні очі. Людина має два ока, які окремо один від одного отримують інформацію від зовнішнього світу. У око входить хрусталь, райдужна оболочка, склоподібне тіло, рогівка, зіниця, сітківка. Око приймає світловий сигнал за допомогою хрусталика і як лінза формує на сітківці зменшене та перевернуте зображення яке ми бачимо на рисунку 1.1 [1].

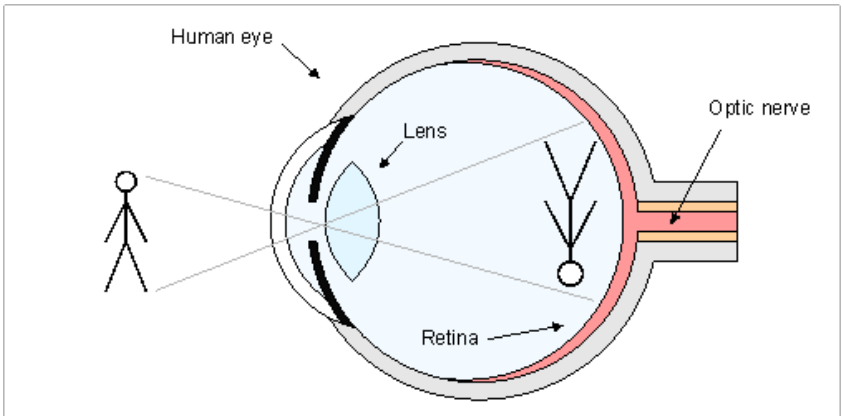


Рисунок 1.1 – Схема відображення видимого предмета на сітківці ока

Структура ока нагадує складний механізм обробки інформації. Усередині людське око організоване таким чином, що відстань між кришталиком-лінзою і сітківкою фіксована, тому природа створила інший механізм створення чіткої картини за рахунок зміни форми кришталика, що змінює фокусні характеристики кришталика. Цей механізм створень на основі можливості змінюватися волокнам віїного пояска, що надають волокнам як результат зміни створення форми для різних ситуацій та стиснення або розтягування кришталика. Сплощена форма зору відповідає на віддаленому предметі з фокусною відстанню кришталика приблизно 14 мм і

це значення при розслабленому стані вії м'яза, в цей момент око розглядає предмети розташовані на відстані більше 3 м.

Округлена форма зору відповідає на близько розташовані предмети з фокусною відстанню кришталика – приблизно 17 мм і це значення при напруженому стані віїного м'яза, в цей момент око розглядає предмети розташовані дуже близько.

Зображення, що потрапляє на сітківку ока, сприймається рецепторами розташованими поблизу області жовтої плями. Збудження цих рецепторів пов'язане і залежить від інтенсивності світла падаючого на око. Світлове збудження рецепторів створює електричні нервові імпульси, що потрапляють по ритнильних каналах в мозок людини. Проходячи по нейроних вузлах мозку людини відбувається розпізнавання та аналіз зображення, що потрапило в око людини.

Зорова система по-різному реагує на випромінювання, які рівні за потужністю, але мають різні довжини хвилі. Людське око сприймає хвилі електромагнітного діапазону в світовому діапазоні. Крім того, зорова система характеризується чутливістю до насиченості кольору. Оскільки цифрові зображення відтворюються як дискретна безліч елементів з різною яскравістю, тому зорова система людини здатна адаптуватися до величезного, порядку 1010, діапазону яскравостей від порога чутливості скотопічного зору, до краю сліпучого блиску. Людина сприймає світловий потік оком діапазоном яскравістю від 0,003 до 0,3 кд/м2. Здатність ока розрізняти мінімальні відмінності яскравості суміжних областей зображення, характеризується контрастною чутливістю. Чутливість зорової системи наведених типів не є постійною, а залежить від багатьох факторів, зокрема умов освітлення. Цей процес називається яскравою адаптацією ока.

Відчуття кольору описують три основні характеристики: світло, кольоровий тон та насиченість. В оці людини містяться два типи світлочутливих клітин: палички та колбочки. Завдяки ним можливо

розрізняти коли темно, а коли світло. Палички задіяні коли сутінки, вони отримають безбарвне зображення. На відміну від паличок колбочки задіяні лише вдень і при яскравому освітленні. Їхня функція – визначення відчуття кольоровості.

Палички – на вигляд периферичні відростки світлочутливих клітин сітківки ока, та вони мають витягнуту циліндричну форму. Сітківка ока містить їх величезне число, 120 млн паличок розмір яких завдовжки 0,06 мкм та діаметром 0,002 мкм. Чутливість палички неймовірно висока і дозволяє відчути людині попадання на неї навіть двох трьох фотонів [2].

Інший тип фоторецепторів – колбочки і на сітківці ока людини налічується близько 6–7 млн. колб з довжиною приблизно 50 мкм і діаметром від 1 до 4 мкм. Колбочки майже в 100 разів менш чутливі до світла, ніж палички сітківки, але вони виграють паличок швидкістю реакцією зміни світла.

В області виходу зорового нерва розташована сліпа пляма, в якій рецептори повністю відсутні. Щодо жовтої плями рецептори по всій області сітківки розташовані симетрично, але щільність розподілу колб та паличок не однакова. Щільність паличок має три максимуми щодо центру сітківки. Перший максимум щільності паличок розташований нижній частині ока поблизу сліпої плями та поступово зменшується на периферію сітківки. Другий сплеск максимуму щільності паличок спостерігається верхній частині ока поблизу сліпої плями і зменшується до центра сітківки. Третій сплеск максимуму щільності спостерігається верхній частині ока поблизу центру сітківки і зменшується на периферію сітківки. Щільність колб максимальна в центрі сітківки і різко спадає на периферію сітківки.

Наявність трьох видів колб і паличок дозволяє людині розрізняти кольори і відповідають трьом «основним» кольорам. Вони дозволяють розпізнавати людині тисячі відтінків кольору.

Вночі працюють тільки палички і людина не розрізняє кольори. Розподіл кольорових колб у сітківці нерівномірний. Сині колбочки за щільністю розподілені переважно ближче до периферії сітківки. Червоні колби розподілені хаотично. Аналогічно розподілені випадково зелені колбочки. Спектральна чутливість червоних і зелених колб дуже близька і ці обидві чутливості кольору частково перекриваються зі спектральною чутливістю синіх колб. Рівномірний відразу трьох елементів, що відповідає звичайному денному світлу і дає відчуття білого кольору.

1.2 Формування зображення цифровими пристроями

Щодня людина тримає у руці смартфон. Детально розглянемо основний тип пристроїв повсякденності людини, який реєструє та перетворює.

Дуже коротко описати послідовний конвеєр формування зображення будь-якого цифрового пристрою можна представити так: по-перше, світло від об'єкта зйомки потрапляє в об'єктив камери і далі світлові промені проходять через набір лінз об'єктиву камери, масштабуючись точно під розміри фільтра мозаїки Баєра
(рис. 1.3).

Рисунок 1.3- Конвеєр зображення для цифрової камери

Потрапляючи на фільтр у цьому етапі промені світла проходять незалежно через набір червоних, синіх та зелених фільтрів мозаїки. Після

чого для корекції світлові промені ще проходять згладжуючий фільтр. Сформоване таким чином світло «захоплюється» датчиками, пікселями матриці цифрової камери. Кожен піксель фіксує тільки інтенсивність світла, що падає на нього. Для остаточної обробки блоку пікселів 4×4 використовуючи різні алгоритми інтерполяції та шаблони колірних просторів RGB та YMCG заповнюють колірні площини. Механізм цих обробок залежить від виробника. Передостанній етап перед записом і зображення-коригування точки білого. І останній етап гамма-корекція зображення.

Всі корекції зображення необхідні, щоб прибрати шум цифрової камери. Цифровий шум виникає через дефект матриці або пошкодження кристалічної решітки кремнію підкладки матриці і звичайно внаслідок стохастичної природи взаємодії фотонів світла і атомів матеріалу фото сенсора. Шум збільшує брак- наявність дефектних і не працюючих пікселів, що виникають при виробництві фотосенсорів через недосконаліть технології. Дефект зображення вносить інтерполяція, що коригує кінцеве зображення при використанні фільтра Байєра. Людське око має логарифмічну чутливість до світла, а фотосенсори - лінійну, тому слабкі сигнали посилюються більше, ніж сильні, щоб зображення мало звичний для людини вигляд, потрібна корекція.

Підсумкове зображення записується у пам'ять камери у форматі JPEG або іншого формату.

Найпоширенішим фільтром є мозаїка кольорових фільтрів, та найвідомішим є мозаїка Байєра (рис. 1.4). Цікаво, що з чотирьох сусідніх сенсорів два реєструють зелені кольори, інші два – червоні та сині (блоки RGGB) [3].

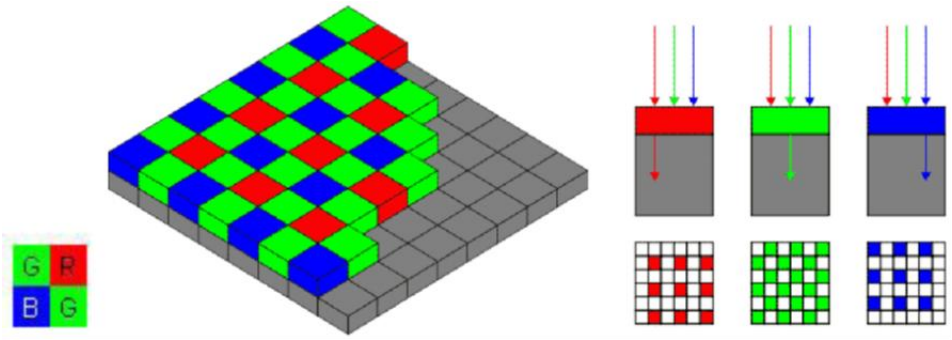


Рисунок 1.4- Мозаїка Байера

Чому в матриці Байєра міститься вдвічі більше зелених сенсорів, ніж червоних та синіх. Відхилення трійки головних кольорів викликано тим, що око людини є найбільш чутливим до зеленого кольору, менш до червоного та до синього кольорів разом узятих. Технічно слід зауважити, що не вся поверхня пікселю є однаково світлочутливою, і тому для збільшення кількості фотонів котрі потрапляють на піксель над ним встановлюють мікролінзу [4].

У багатошарових матрицях поділ світлового потоку на складові кольору відбувається за природою того, що хвилі синього кольору проникає в шар кремнію на найменшу глибину, а червоного на найбільшу глибину. Фотоелемент кожного пікселя багатошарових матриць складається з трьох шарів основних кольорів. Електричні заряди, що утворюються, накопичуються в трьох областях на кожному шарі. Так для будь-якого пікселя матриці формується значення відповідного кольору. Це дозволяє відмовитися від мозаїчних схем Баєра. Проте недолік є тому, що при переході з одного шару в іншу частину фотонів неминуче поглинається і це позначається на чутливості матриці, вона послаблюється.

Мікролінзи цифрових камер конструюють так, щоб поліпшити збір фотонів, кожним осередком і створити хороше зображення за однакового часу експозиції (витримки). При спуску затвора фотоапарата на мільйони крихітних піксель потрапляє світло, на них накопичується заряд, який звичайно передається на електричну схему. Посилення сигналу виконується

відповідно до налаштувань чутливості ISO, вибраних камерою. У всіх на слуху дуже відомі японські камери фірми Nikon. Наприклад матриця фотокамери Nikon D40[5].

В даний час при створенні світлочутливих матриць для цифрових пристроїв, використовуються головним чином дві технології – CMOS (Complementary Metal Oxide Semiconductor) та CCD (Charge Coupled Device). КМОП-матриця може перетворювати заряд на напругу, перетворимо прямо на кожному пікселі. Ця технологія дозволяє багато разів підвищити швидкість пристрою при конвертації інформації по матриці. Збільшення фізичних розмірів матриці в будь-якому разі спричиняє зміни діаметра лінз [6].

Матриця зберігає інформацію про кількість фотонів у аналоговому електронному вигляді, а ми його повинні конвертувати в цифровий формат. Процес конвертації відбувається у пристрої конверторі, котрий є у кожному смартфоні або цифрової фотокамери -Analog to Digital Converter (ADC).

Наприклад, візьмемо 16-бітний ADC. Він дозволяє розділити яскравість для кожного пікселя від 0 до 32667 кожного кольору і тепер кожна точка зображення представлена цифрою. Інформація у вигляді багатовимірного масиву потрапляє до буфера, а від туди смартфон записує зображення на карту пам'яті, а потім воно потрапляє до комп'ютера. Наскільки швидко камера записує зображення з буфера в карту пам'яті і показує швидкість зйомки зображення у секунду.

1.3 Формування зображення математичними методами

Можна просто створити зображення обличчя випадковим і неконтрольованим способом, але це погано, тому що для генерації зображення обличчя по завданню згенерувати те, що хочемо задати, наприклад, певний вид обличчя, стать людини, зачіску, усміхнене або сумне

обличчя і т.д. По завданню потрібно представити алгоритм різноманітного та контрольованого процесу генерації особи. Схематично представлено на малюнку рис.(1.7) [7].

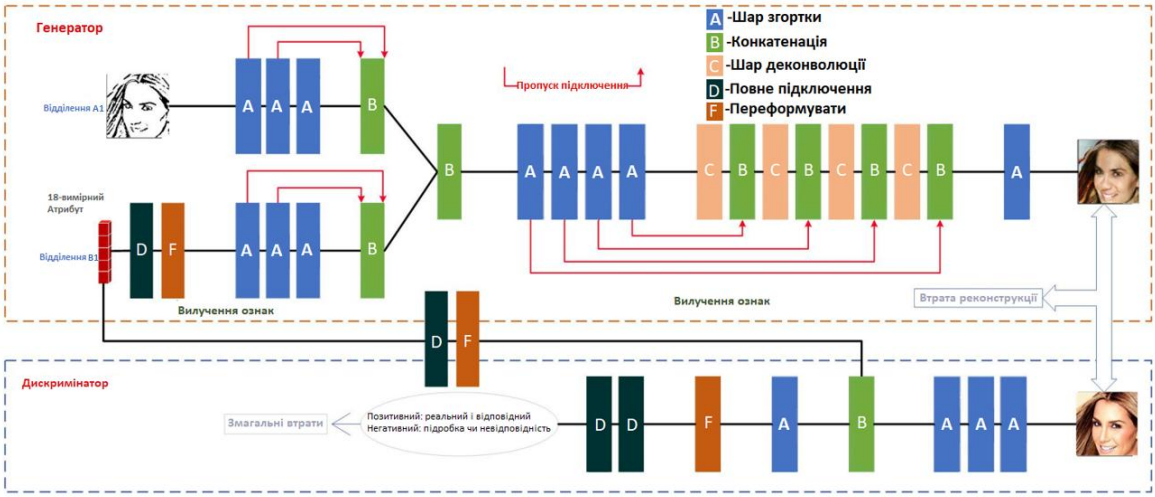


Рисунок 1.7 – Структура процеса формування зображення обличчя

На ньому структура процесу алгоритму формування зображення представлений сам процес генерації особи у вигляді нейронної мережі на базі ескізних еталонних зображень, яка розділена на дві гілки: генератор і дискримінатор. Вхідними даними для генератора є зображення ескізу, які є вхідними даними гілки A1, та вектори атрибутів ознак, які є вхідними даними гілки B1. Зображення особи генерується після проходження через генераторну мережу, згенерована особа створюється шляхом додавання до зображення ескізу додаткової функції атрибута особи. Генератор мережі складається з шарів мережі, вилучення ознак особи, мережі підвищуючої та знижувальної дискретизації. Дискримінатор мережі служить для перевірки того, чи містять згенеровані особи бажані атрибути чи ні. Це етап отримання основних ознак небажаних атрибутів. По суті перцептивно з'єднуємо ескізні зображення та атрибути. Відмінність згенерованого від реального зображення особи становить змагальні втрати реконструкції, що виникають після роботи дискримінатора. Функція змагальних втрат використовується для оновлення як параметрів мережі генератора так і параметрів мережі

дискримінатора. Які будуть використані для синтеза майбутнього привабливого образу.

Програми глибокого навчання вдосконалюються з кожним роком і з'являються нові й нові генеративні моделі. Серед них можливо представити три типи: авторегресійні моделі, варіаційні автоенкодери (VAE) і генеративні змагальні мережі (GAN)[8].

В даний час найкращі та якісні зображення генерують мережі GAN (вони виглядають реалістично та різноманітно, з дуже переконливими деталями обличчя та у гарному дозволі).

Програма версії GAN базується на версії моделі навчання без вчителя. На етапі після навчання нейромережі на вхід приймає випадковий шум і по ньому створює зображення людини, яка майже не відрізняється від еталоного зображення. У різних сферах діяльності людини замовники подібних програм хочуть бачити в програмі штучного інтелекту зразки з довільними ознаками такими як вік, колір волосся, обличчя, губи, ніс, брови.

На малюнку рис.(1.8) представлена блок схема створення та генерація штучним інтелектом привабливого образу.

Рисунок 1.8 – Блок схема керованих генерацій зображень

Для численних генерацій образів людини створено велику кількість варіантів GAN. Умовно прийнято два типи: нейронні мережі перенесення стилю та генератори образів за умовою.

На даний момент відомі умовні генератори - GAN, AC-GAN і Stack-GAN - під час навчання одночасно є можливість налаштовувати генеровані

зображення за шкалою ознак. Однак і тут суттєвий недолік такої генерації образу людини який полягає в тому, що необхідно перенавчити всю модель GAN для додавання нових ознак має бути тривалий процес. При цьому навчанні можливе лише на одному наборі даних банку осіб зображень. Також можемо уявити генеративно-змагальну мережу з прозорим прихованим простором (Transparent Latent-space GAN, TL-GAN) , яка використовує і пропонує можливість плавно налаштовувати кілька ознак генерації образу обличчя на одній мережі.

Якщо подивитися на характеристики моделі генерації зображення обличчя pg-GAN від Nvidia, слід відзначити дуже реалістичне зображення з високою роздільною здатністю 1024×1024px. Повний контроль генерації зображення особи надає приховане пространство.

Щоб зрозуміти прихований простір для створення образу особи людини в процесі генерації необхідно відзначити, прихований простір добре заповнено і всі точки простору створюють привабливі та розумні зображення.

Приховане пространство містить вектора ознак, які необхідні для генерації особи і орти напрямків цих векторів ознак служать для управління процесом генерації. Це можна уявити як систему координат з осями ознак генерації особи і всі дані цієї системи координат містять прихований простір.

На сьогоднішній день TL-GAN рис.(1.9) є найуспішнішою моделлю для генерації обличчя людини. Використовуючи класифікатор дискретних методів, можливо легко уявити ключові моменти ознак підсумкових зображень завдяки навченій мережі.

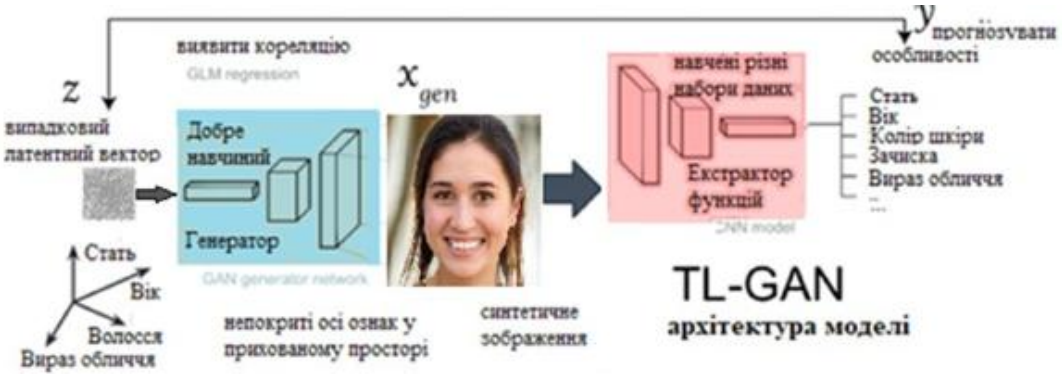


Рисунок 1.9 – Модель TL-GAN прихованого простору для керування процесом генерації

На рисунку 1.9 архітектура моделі TL-GAN.

1. Підбір добре навченої моделі GAN, яка дає найкращу якість генерації осіб.
2. Вибираємо набір ознак для класифікації використовуючи просту згорткову нейронну мережу, що містить більше тисяч осіб і має кілька десятків міток ознак.
3. У процесі генерації послідовно задаємо невелику кількість довільних прихованих векторів і генеруємо привабливі зображення після чого застосовуємо навчений коректор ознак для зміни на кожному викривленні особи.
4. Здійснюємо регресію між прихованими векторами та ознаками застосовуючи лінійну модель. Лінія проходження регресії співвідноситься з осями ознак особи.
5. Стартуємо генерацію з першого розкритого вектора і послідовно переміщуємо в долю вибраних осей ознак і досліджуємо його вплив на результат обличчя.

Результат роботи програми за кількома осями ознак представлена на малюнку рис.(1.10) і на подив працює чудово.

Рисунок 1.10– результати переміщення прихованого вектора вздовж заплутаних осей ознак

Ми розглянули формування зображення математичними методами. Було представлено огляд сучасних методів формування винаходи обличчя людини за допомогою штучного інтелекту на основі глибоких нейронних мереж. У ході роботи розглянуто методи створення зображення на основі моделей навчання без вчителя GAN. Докладно вивчений метод формування винаходів на основі прихованого простору. Були виявлені у даного методу формування зображення особи позитивні та негативні сторонні методи. При русі вектора ознак по осі прихованого простору зображення перетворюється відповідно до обраної ознаки осі та забезпечує правильне зображення особи згідно з вимогами. Метод GAN показав хороші результати та виявив ефективні способи генерації винаходи обличчя. Ефективність додавання нових ознак займає мало часу та не потребує перенавчання мережі моделі GAN.

2 ІДЕНТИФІКФАЦІЯ ЦИФРОВИХ ЗОБРАЖЕНЬ

2.1 Ідентифікація цифрової камери за цифровим зображенням

В оперативно-розшуковій та експертній практиці зараз переважно використовують цифрові зображення, зафіксовані камерами відеоспостереження або задокументовані криміналістами знімки, зняті цифровою камерою на місці злочину. Встановлення справжності зображення із конкретної цифрової камери, відсутність фотомонтажу у зображеннях. У більшості випадків експертиза утруднена, оскільки потрібно встановити належність зображення конкретної фотокамери і при цьому додається зображення, представлене у форматі компресії з втратою якості. Набагато легше ідентифікувати знімок, отриманий гарною цифровою камерою, використовуючи RAW-зображення, яке піддане тільки корекції піксельних дефектів.

Шум цифрової камери у зображеннях – це подібність зерна фотоплівки. Цифровий шум - це окремі пікселі один або кілька і відрізняється більш темними або світлими відтінками зображення кольору, як яскравий шум і за кольором, як хроматичний шум. Для різних фотоапаратів шуми різні і це залежить від фізичного розміру матриці та її пікселів, кількості пікселів у матриці та від алгоритму електронного фільтрування шуму. Чим більша світлочутливість камери, довжина експозиції знімка і температура тим більший шум. А також шуми камер поділяються на випадковий, структурний та лінійний.

Випадковий шум дає коливаннями яскравості та кольоровості більше і менше реальних. Випадковий шум присутній за будь-якого часу експозиції і сильно залежить від світлочутливості і змінюється від кадру до кадра.

Структурний шум характеризується тим, що інтенсивність яскравості і кольоровості більш ніж випадковий шум. Він унікальний.

Лінійчастий шум вносить сама камера в процесі зчитування даних із цифрового сенсора. Цифровий шум у датчика виникає з причин: Чорний дефект - такі дефекти на світлому зображенні помітні як чорні точки, Темновий струм - такі дефекти на темному зображенні помітні як білі точки і з'являються при великих експозиціях, наявність дефектних пікселів, що виникають при виробництві. Такий же дефект вносить інтерполяцію, що коректує останні зображення, що використовує фільтр Байєра, дефекти внаслідок гамма-корекції. На величину цифрового шуму впливає: Розмір сенсора датчика та його роздільна здатність. Розмір сенсора залежить від технології CCD чи CMOS.

Типовий шум датчика зображення використовується для ідентифікації цифрової камери. Шум Камери виходить шляхом віднімання очищеної від шуму версії зображення оригінального початкового зображення.

Для очищеного зображення від шуму використовується шумозаглушуючий фільтр на основі вейвлета. Еталонний шаблон камери визначається шляхом усереднення шаблонів шуму з кількох зображень, знятих на камеру. Цей еталонний шаблон шуму є ідентифікацією камери.

Для демонстрації методу ідентифікації цифрової камери вибираємо колекцію зображень, отриманих на п'яти камерах. З кожної камери беремо двадцять різних фотозображень, отриманих у максимальній роздільній здатності без будь-якої обробки. З огляду на різну роздільну здатність фотокамер для побудови карти неоднорідностей використовувалася центральна ділянка кадру розміром 1024x1024 пікселя.

Послідовно перебираючи кожну з двадцяти фотографій однієї з камер і обчислюючи матрицю шуму кожної фотографії як різницю матриці інтенсивності вхідного необробленого зображення (M×N пікселів) і матриці того ж зображення, що пройшло шумоподавляющій фільтр.

Таким чином, отримуємо набір шаблонів шумів камери для кожної фотографії. Знаходячи матрицю середніх значень за набором шаблонів

матриць всіх шумів кожної фотографії камери і тим самим отримуємо еталонний шум камери. (Рис 2.1) Повторюючи процедуру кожної з п'яти камер отримаємо п'ять еталонних шумів.

Рисунок 2.1 – Схема знаходження еталонного шуму камери.

Використовуючи ці п'ять еталонних шумів, ми точно можемо індифікувати тестовий знімок, знятий на вибір будь-якої з п'яти камер.

Для цього беремо тестове зображення зняте на одній з п'яти камер з прилеглим з нею шумом її рідного датчика і віднімаємо з нього чисте зображення тієї ж тестової фотографії обробленим шумопідводним фільтром. В результаті отримуємо шум тестового зображення невідомої камери з множини п'яти камер. Перебираючи набір еталонних шумів всіх п'яти камер щодо кореляції з тестовим шумом невідомої камери. Після обчислення кореляції вибираємо еталонний шум із найвищою кореляцією із шумом тестового зображення. Таким алгоритмом визначається камера, що входить до цього списку з п'яти камер і з якої камери був зроблений певний знімок за списком еталонних шумів використаних камер.

2.2 Опис нового алгоритму, заснованого на сингулярних числах

Сингулярні розкладання широко використовуються для вирішення великого кола завдань. Сингулярне розкладання походить з теорії матриць. З цієї теорії відомо, що будь-яку матрицю розміром M x N можливо подати у вигляді добутку трьох матриць формула 2.1: [9]

$$MM = UM * DM * VM^T \quad (2.1)$$

Де MM вихідна матриця для розкладання розміром M x N.

UM- ортонормована матриця, розміром M x M яка відповідає умові $UM*UM^t=I$, де I - одинична діагональна матриця.

DM- діагональна матриця розміром M x N з невід'ємними елементами, у якої елементи, що лежать на головній діагоналі, є сингулярними числами. Всі інші елементи матриці MD, що не лежать на діагоналі, є нульовими.

VM^t - пов'язана транспонована матриця розміром N x N ортонормована матриця правих. При цьому вона також задовольняє умову $VM*VM^t=I$, де I- одинична діагональна матриця.

Сингулярне розкладання вихідної матриці також можливо у вигляді малоранговой апроксимації. Перший у тому числі U масив однорангових k стовпців, другий у тому числі V масив однорангових k рядків. Матриця S діагональна матриця розміром k служить малоранговой апроксимації k значеннями сингулярних значень матриці. Такою апроксимацією можливо досягти мінімальної відмінності від значень вихідної матриці, але при цьому в розкладанні використовується малорангові вектори, що спрощують і прискорюють розрахунок апроксимації.

Усічена k сингулярна декомпозиція матриці M представляє та відображає малоранову апроксимацію. При заданому рівні зрізання k апроксимації вихідної матриці M розкладання створюється з урахуванням лише перших k стовпців лівої сингулярної векторної матриці U і першими k рядками правої сингулярної векторної матриці V, а також усіченої частини k

× k діагональної матриці S матриці сингулярних значень M. Розглянуті підматриці виділені кольором (Рис 2.2).

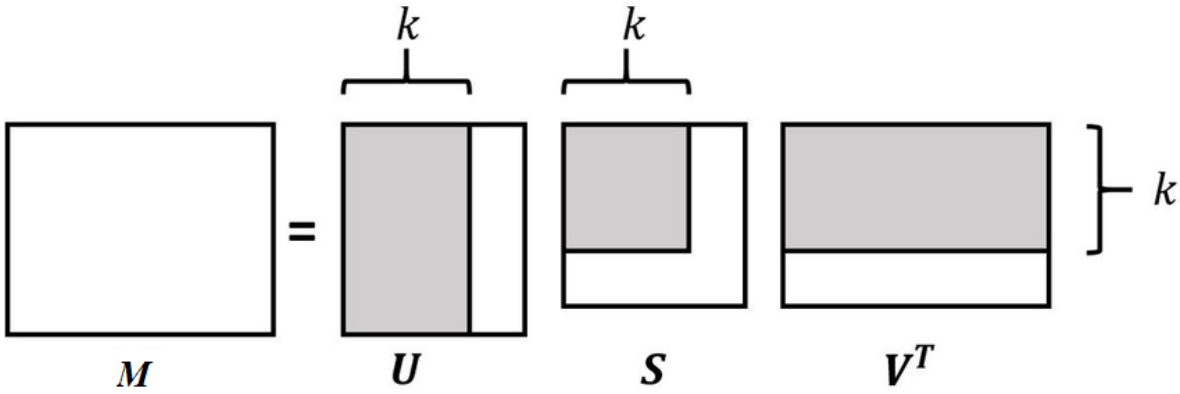


Рисунок 2.2 – Усічена k сингулярна декомпозиція матриці M.

Малорангову апроксимацію можна представити у вигляді суми R, складових вихідної матриці. Формула розкладання матриці F також можливо представити у вигляді суми сингулярних значень і добуток стовпців і рядків власних векторів відповідно до цих значень формула 2.2:[10]

$$F = \sum_{j=1}^{R} \lambda(j) u_j v_j^t \qquad (2.2)$$

при цьому сингулярні значення відсортовані від максимальних до мінімальних, таке розкладання дозволяє представити матрицю з неповним використанням сингулярних значень і в даному випадку використовується тільки R сингулярних значень.

У наведеній вище формулі λ (j) є j-е сингулярне значення U_j=U(j), є j-й стовпець власного вектора, а V_j^t =V^t(j) j-й рядок з транспонованого власного вектора. Реалізація розкладання зображення з урахуванням сингулярних значень виконується серед Matlab. У разі застосування сингулярного розкладання для представлення зображення, необхідно сформувати вихідну матрицю. Зазвичай кольорове зображення представлене у вигляді трьох матриць: червоного M^r, зеленого M^g та синього M^b. Вихідна матриця формується як функція Matlab за формулою 2.3:

$$M = rgb2gray(M) \quad (2.3)$$

таким чином виходить усереднена матриця сірого кольору.

Використовуючи інструментарій Matlab для знаходження сингулярних значень, можна скористатися функцією SVD формула 2.4 [11]

$$[U,S,V] = svd(A) \quad (2.4)$$

U – матриця лівих, S – діагональна матриця сингулярних значень, V – матриця правих. Підставляючи у функцію svd можливо отримати U, S, V розкладання. Розкладаючи матрицю, таким чином можна використовувати для апроксимації не весь набір сингулярних значень для представлення усередненої матриці сірого кольору. Для отримання R наближення матриці зображення сірого кольору використовуємо суму R доданків:

$M\Sigma=S(1,1)*U(1)*V^t(1)+S(2,2)*U(2)*V^t(2)+S(3,3)*U(3)*V^t(3)+\ldots+S(R,R)*U(R)*V^t(R)$.

Це наближення вважається чистим зображенням R розкладання по сингулярним значенням.

Шумом H вважається різниця вихідного зображення матриці сірого кольору та умовного чистого зображення R розкладання за сингулярними значеннями формула 2.5:

$$H=M - M\Sigma \quad (2.5)$$

2.3 Опис ефективності кожного з алгоритмів

Основною метою роботи було розпізнавання штучно згенерованих зображень за допомогою глибоких нейронних мереж стосовно реальних фотографій. При розпізнаванні штучно згенерованого зображення важливим фактором є шум зображення. Випробовано два способи отримання шуму зображення. Шум зображення визначався за різницею вихідного зображення та зображення обробленого фільтром у першому способі та другому.

Як фільтр використовувався Медіанний фільтр середовища Matlab. У ньому для очищення зображення фільтр виконує медіанну фільтрацію зображення у двох вимірах. Кожен вихідний піксель містить середнє значення на околиці 3 на 3 навколо відповідного пікселя присутнім у вхідному зображенні. Даний алгоритм приносить зміни по відношенню до вхідного зображення усереднюэ кожен піксель.

У другому способі отримання шуму зображення використовуються сингулярне розкладання зображення. У ньому для отримання відфільтрованого зображення береться певна кількість складових шарів розкладання по сингулярних числах, кількість сингулярних шарів вірується в діапазоні від 5 до 150. Шум зображення визначався різницею вихідного зображення і відфільтрованого зображення на основі сингулярних чисел.

У другому способі варіюючи кількість сингулярних чисел для формування відфільтрованого зображення можна досягти необхідної величини шуму. У першому способі необхідної велечени шуму досягти неможливо, тому що різні способи формування фільтровоного зображення, в медіаному фильтрі це спосіб обробки за допомогою медіанна. Вихідний піксель медіанного фільтра, є середні значення відсортерованого списку вікна фільтрації.

На рисунку (2.3) представлені усереднені шуми зображень, бази даних не реальних штучно синтезованих фотографій людей, створених нейронною мережею.

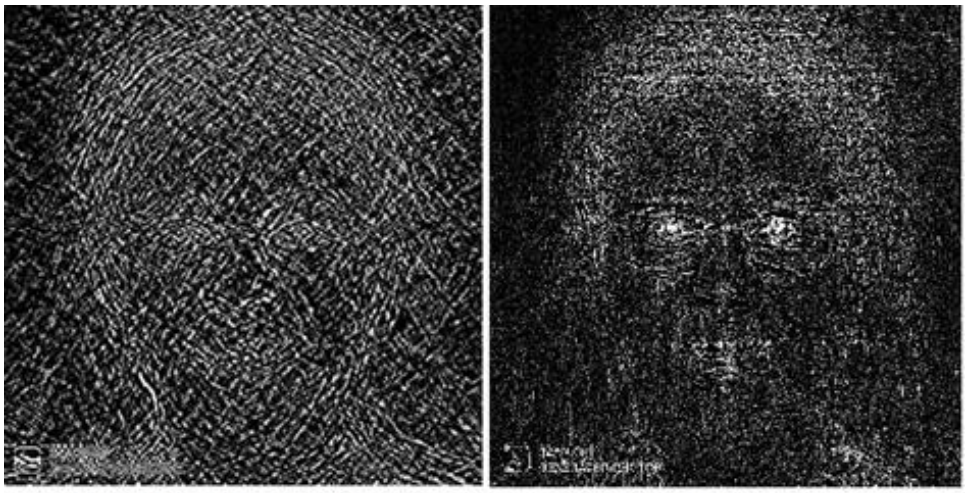


Рисунок 2.3 – Шум Сингулярного розкладання та Медіанного фільтра.

На рисунку (2.4) представлені шуми зображення реальної людини для порівняння.

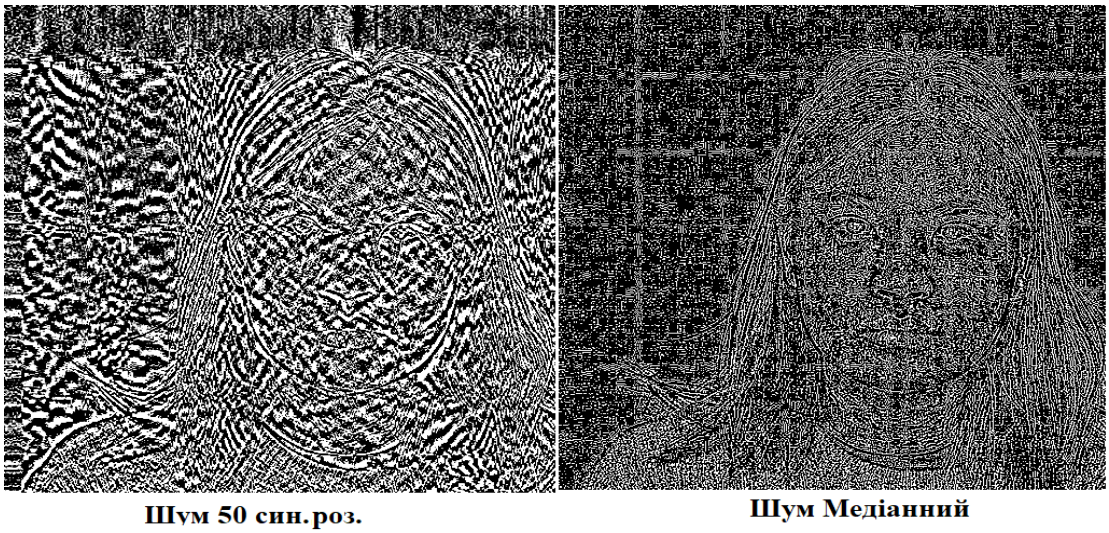
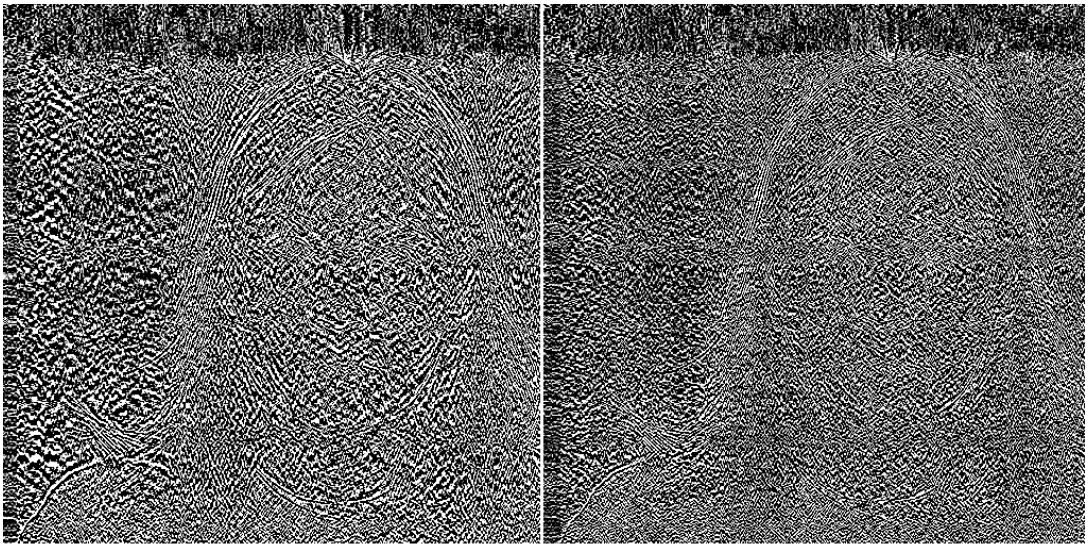


Рисунок 2.4 – Шум Сингулярного розкладання та Медіанного фільтра.

Змінюючи кількість сингулярних чисел для створення чистого зображення, відбувається якісна зміна шуму, це можна спостерігати на малюнку (2.5) для ста і ста п'ятдесяти сингулярних чисел. Три результати зображення шумів отриманих на основі сингулярного розкладання наочно

показують ефективність створення зображення цим методом, що дозволяє плавно змінювати величину шуму.



Шум 100 син.роз **Шум 150 син.роз**

Рисунок 2.5 – Шуми Сингулярного розкладання для 100 та 150.

Тестування методу отримання шуму за допомогою медіанного фільтра, представленого на рисунке (2.6) підтвердило те, що кореляція реальних фотографій від 6 * 10-3 до -7 * 10-3 що значно менше ніж кореляція штучних від 0.3 до 0.8. Кореляція – це статистичний взаємозв'язок двох шумів зображень.

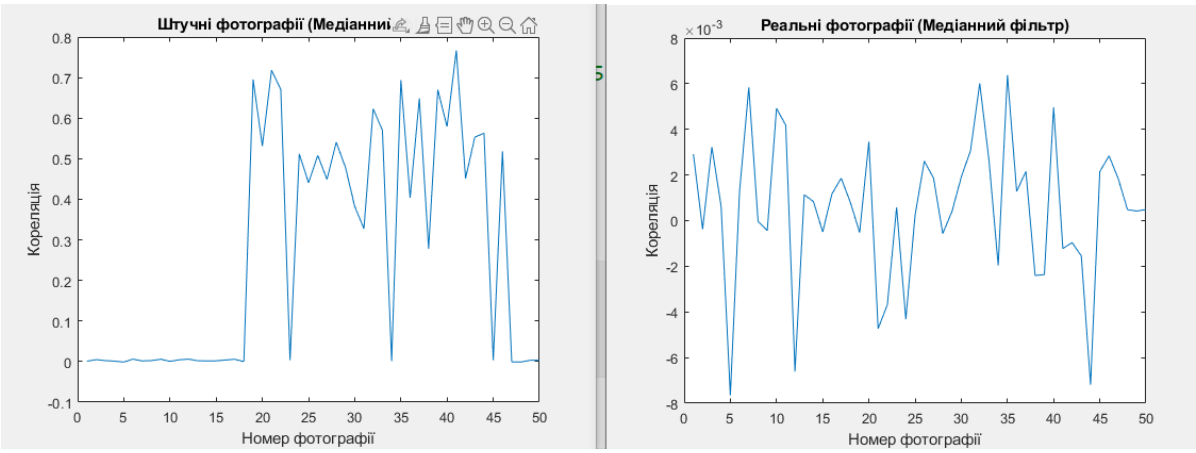


Рисунок 2.6 – Тестування на шуми 50 реальних фотографій та 50 штучних за допомогою медіанного фільтра.

Тестування методу отримання шуму за допомогою сингулярного розкладання, представленого на рисунке (2.7) показало ефективність цього методу. За результатами видно, що кореляція реальних фотографій від 0.01 до -0.01 у прийнятному діапазоні розкладу сингулярних чисел від 50 до 150.

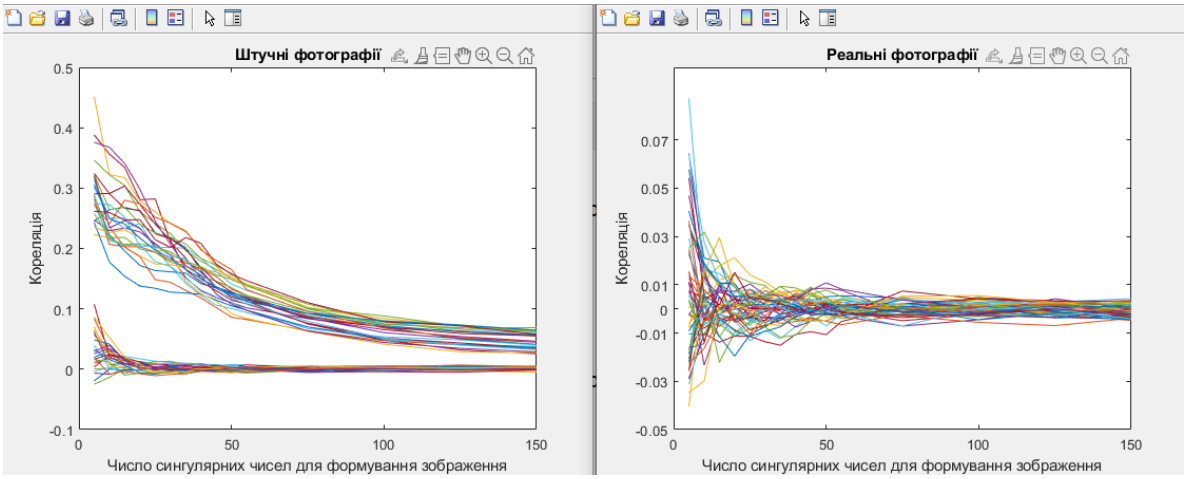


Рисунок 2.7 – Тестування на шуми 50 реальних фотографій та 50 штучних за допомогою сингулярного розкладання.

Порівняння графіків тестування кореляції реальних фотографій і штучних за допомогою двох наведених методів показало, перевагу сингулярного розкладання зображення для отримання шуму.

У розділі досліджено різні варіанти виникнення шумів. Першим досліджено шум цифрової камери, елементи, що становлять шум і причини його виникнення. Досліджено спосіб отримання еталонного шуму камери. Досліджено шуми зображення створених за допомогою нейронних мереж та цифрової камери.

Методика дослідження шумів зазначених джерел, спочатку випробувано вже на відомому методі медіанного фільтра, але зазначений метод недостатньо точно відображав відмінності шумів штучно створених зображень і з цифрових камер. Новий метод відтворення зображення з урахуванням сингулярного розкладання дозволив плавно регулювати шум

зображення. Завдяки отриманню шумів на основі сингулярного розкладання зображення цифрової камери та штучно створеного зображення, метод точно дозволив ідентифікувати зображення створені нейронними мережами.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ІДЕНТИФІКАЦІЇ ЦИФРОВИХ ЗОБРАЖЕНЬ

3.1 Засоби реалізації програмного продукту

Для реалізації програмного забезпечення вибираємо середовище розробки Matlab, тому що він створений мовою високого рівня, в якому легко виробляти технічні та навіть найскладніші наукові обчислення [12]. Matlab - це така платформа, в якій об'єднані обчислення і створення програми у вигляді пакетного файлу або у вигляді виконуваного модуля в зручному для користувача графічному інтерфейсі, в якому можна уявити рішення задачі простими математичними позначеннями. Інструменти Matlab дають професійному програмісту платформу для розробки програм, можливості використовувати добре протестовані і добре документовані засоби і великий спектр об'єктів. З величезного списку можливостей Matlab виділимо стандартний набір, який необхідний для обробки зображень: математичні обчислення, розробка алгоритмів програм у пакетному або в командному вікні, аналіз даних представлених у різних графіках, обробку різних типів зображень та їх візуалізацію, створення графічних програмних додатків у редакторі Matlab із використанням інтерактивних компонентів. Для успішного використання середовища Matlab необхідно добре розуміти ключові елементи системи Matlab, такі як: мова Matlab, середовище Matlab, керована графіка, бібліотека функцій, і графічний інтерфейс розробки програм. Мова Matlab дозволяє представити рішення в командному вікні або у вікні редактора Matlab, що створює графічний додаток. Основним типом даних мови Matlab є багатовимірний масив. Для кольорового зображення використовуються три двомірні масиви. Мова Matlab дозволяє будувати прості та швидкі програми для завантаження, обробки та візуалізації зображень використовуючи матричний формат. Опції мови Matlab

дозволяють прискорити обробку зображення використовувати паралельні обчислення [13].

Середовище Matlab має набір інструментів для створення ефективних програм. Слід зазначити робочий простір змінних, які задіяні в програмі, що розробляється, дозволяє відстежувати значення в будь-який момент програми. Крім цього змінну можна відстежити у спливаючому вікні, навівши курсор на цю змінну в момент зупинки виконання. Налагодження в Matlab дозволяє зайти в функцію, що тестується, або обійти її не заходячи. При наборі в редакторі Matlab середовище пропонує набір функцій, що починаються з префіксу набраного на екрані. Середовище Matlab містить добре розроблений інтерфейс для побудови графіків. Кожен елемент графіка реалізований окремим параметром, що легко настроюється візуально або програмно. Особливо в Matlab слід зазначити наявність спеціальних пакетів функцій бібліотеки Toolbox, які добре розроблені та протестовані і можуть використовуватися з великим спектром параметрів. При цьому ця бібліотека функцій відкрита, оновлюється з кожним роком. Image Processing Toolbox є повноцінним набором еталонних стандартних алгоритмів, а також обробником зображень, аналізу та візуалізації результату. У Matlab для реалізації двомірної та тривимірної графіки існує великий набір функцій, параметри яких дозволяють змінювати зовнішній вигляд графіка і такого інструменту не можна знайти в інших середовищах програмування. Елемент підказки також реалізований у графічному інтерфейсі введення функцій, змінних програм та бібліотек. Для програміста, який має свої напрацювання іншими мовами, Matlab дає можливість викликати функції з бібліотек написаними іншими мовами програмування, що дозволяє в окремих випадках домогтися прискорення виконання програми.

Для розробки та проектування графічної програмної програми в Matlab створено інтерактивне середовище розробки App Designer, який інтегрований

у систему Matlab та оснащений великим набором візуальних компонентів інтерфейсу користувача. Таких як Button, Check Box, Text area та інших [14].

App Designer дозволяє працювати у двох режимах візуальному та програмному і що дозволяє уникнути помилок. У візуальному режимі редактор дозволяє вибрати необхідний візуальний компонент, наприклад: Button і курсором перетнути на форму користувача. Вибравши створений елемент на формі правою кнопкою миші і викликавши Pop меню даного візуального компонента, ми можемо задати callback обробку даного елемента, а також створити контекстне меню для даного елемента. За допомогою правого вікна Component browser в якому міститься список візуальних компонентів поміщених на форму, ми можемо вибрати будь-який з них змінювати його ім'я, яке вимагає завдання, а також налаштувати параметри у вкладці інспектор і пункти обробки у вкладці callback. Кожен візуальний компонент App Designer має власні індивідуальні поля для налаштування, а також спільні поля присутні у кожному візуальному компоненті. Загальні поля візуального компонента зазвичай є найменуванням компонента, позицією компонента, шрифтом.

У програмному режимі редактор App Designer рядки, створені у візуальному режимі, блокують код зміни і програміст не може змінити їх, ці рядки відображені сірим кольором. За допомогою лівого вікна code browser редактора App Designer можна оголосити у вкладці Properties, можна ввести для програми, що розробляється, глобальну змінну видиму для всіх функцій програми. У вкладці Functions можна додати ім'я новоствореної функції та вказати кількість вхідних та вихідних параметрів. Перехід на будь-яку функцію, що розробляється, здійснюється в цьому вікні списку функцій. У вікні code browser у вкладці callback можна перейти на функцію обробки вибраного візуального компонента, аналогічно як і випадки візуального режиму роботи. Для створення діалогових фонів редактор App Designer пропонує створення нових програм у рядку списку програм. У головному

меню App Designer є дві вкладки Designer і Editor. Перша вкладка Designer дозволяє заповнити назву App, автор App, опис. У пункті реалізації, у вкладці Designer можна вказати вид додатків, що створюються: Descktop, web, Matlab. У вкладці Editor App Designer, у пункті run можна запустити виконання програми. При цьому можна вказати вхідні установки програми. Налагодження програми здійснюється установкою точок зупинок у колонці нумерації рядків коду програми. При запуску програми вона зупиниться на вказаних точках.

3.2 Облаштування програми

Програма розроблена у середовищі Matlab у повнофункційному редакторі App Designer.

Інтерфейс виглядає наступним чином на рисунку (3.1)

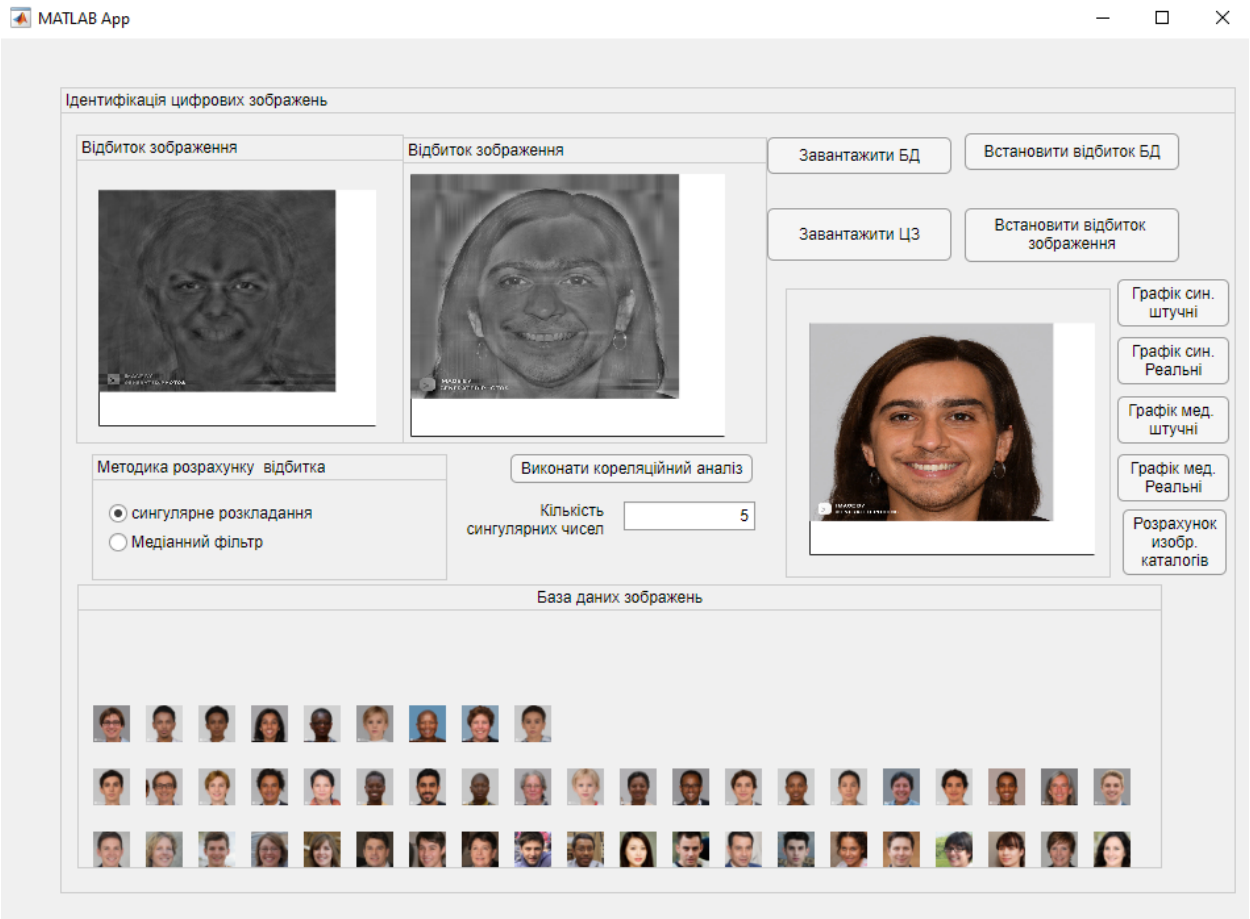


Рисунок 3.1 – Інтерфейс програми.

Програма реалізована у вигляді функціональних клавіш, кожна з яких виконує відповідну дію.

Кнопка “Завантажити БД” служить для завантаження всіх зображень у базі даних. Завантажені зображення відображаються на галереї нижньої панелі.

Функціонально програма підраховує шуми двох режимів. Для цього служить перемикач розрахунку, в якому вибирається алгоритм обчислення шуму зображення.

При виборі розрахунку в режимі медіанного фільтра можливе завантаження на вибір, зображення штучно створеного за допомогою нейронних мереж або реальної фотографії. Після завантаження зображення обчислюється шум зображення за методикою медіа фільтра. Для індикації зображення обчислюється усереднений шум бази даних із 49 штучно створених фотографій. Програма здійснює розрахунок усередненого шуму у вигляді окремої функції. Натискаючи на кнопку «Виконати кореляційний аналіз» виконується розрахунок кореляції зображень, що порівнюються, і після видає результат кореляції.

При виборі розрахунку в режимі сингулярного розкладання проводитися ті ж дії за винятком того, що вказується кількість сингулярних чисел у введеному полі з назвою «Кількість сингулярних чисел» для розрахунку шуму зображення. Програма дозволяє обчислити шуми всіх зображень, що знаходяться у вказаних каталогах: RealPhoto – реальні фотографії, MachinePhoto – зображення сформовані за допомогою нейронної мережі.

Програму реалізовано у вигляді набору функцій Matlab.

function btn_load_dbButtonPushed(app, event) – завантаження бази даних та відображення всіх зображень на окремий візуальний елемент Panel.

function btn_detect_ident_dbButtonPushed(app, event) – обчислення усередненого шуму бази даних та відображення його на окремий візуальний елемент Axes.

function btn_load_imButtonPushed(app, event) – завантаження цифрового зображення та відображення його на окремий візуальний елемент Axes.

function btn_detect_ident_imButtonPushed(app, event) – обчислення шуму зображення та відображення його на окремий візуальний елемент Axes.

function btn_analysisButtonPushed(app, event) - обчислення кореляції.

function UIaxes_dbButtonDown(app, event) – відображення шуму бази даних на окремий графічний редактор.

function UIaxes_imButtonDown(app, event) – відображення шуму цифрового зображення на окремий графічний редактор.

function ButtonPushed(app, event) – відображення графіка кореляції сингулярних чисел для штучних зображень.

function Button_2Pushed(app, event) – відображення графіка кореляції сингулярних чисел для реальних зображень.

function Button_3Pushed(app, event) – відображення графіка кореляції медіанного фільтра для штучних зображень.

function Button_4Pushed(app, event) – відображення графіка кореляції медіанного фільтра для реальних зображень.

function Button_5Pushed(app, event) – розрахунок необхідних даних для відображення графіків за вказаними каталогами зображень.

function results = loadcatal(app,catalog1) – завантаження каталогу зображень, підрахунок шумів, збереження результатів розрахунку в файли.

function results = graficresult(app,catalog1) – відображення графіків за вибраним каталогом результатів.

function results = Calkshum(app,tipalgoritm,countsin) – розрахунок шумів залежно від типу алгоритму.

3.3. Інструкція користувача

Програма розроблена для ідентифікації зображення та його співвідношення до штучно згенерованих зображень за допомогою нейронної мережі або реальних зображень.

Після старту програми необхідно вибрати каталог, натиснувши кнопку «Завантажити БД» на рисунке (3.2)

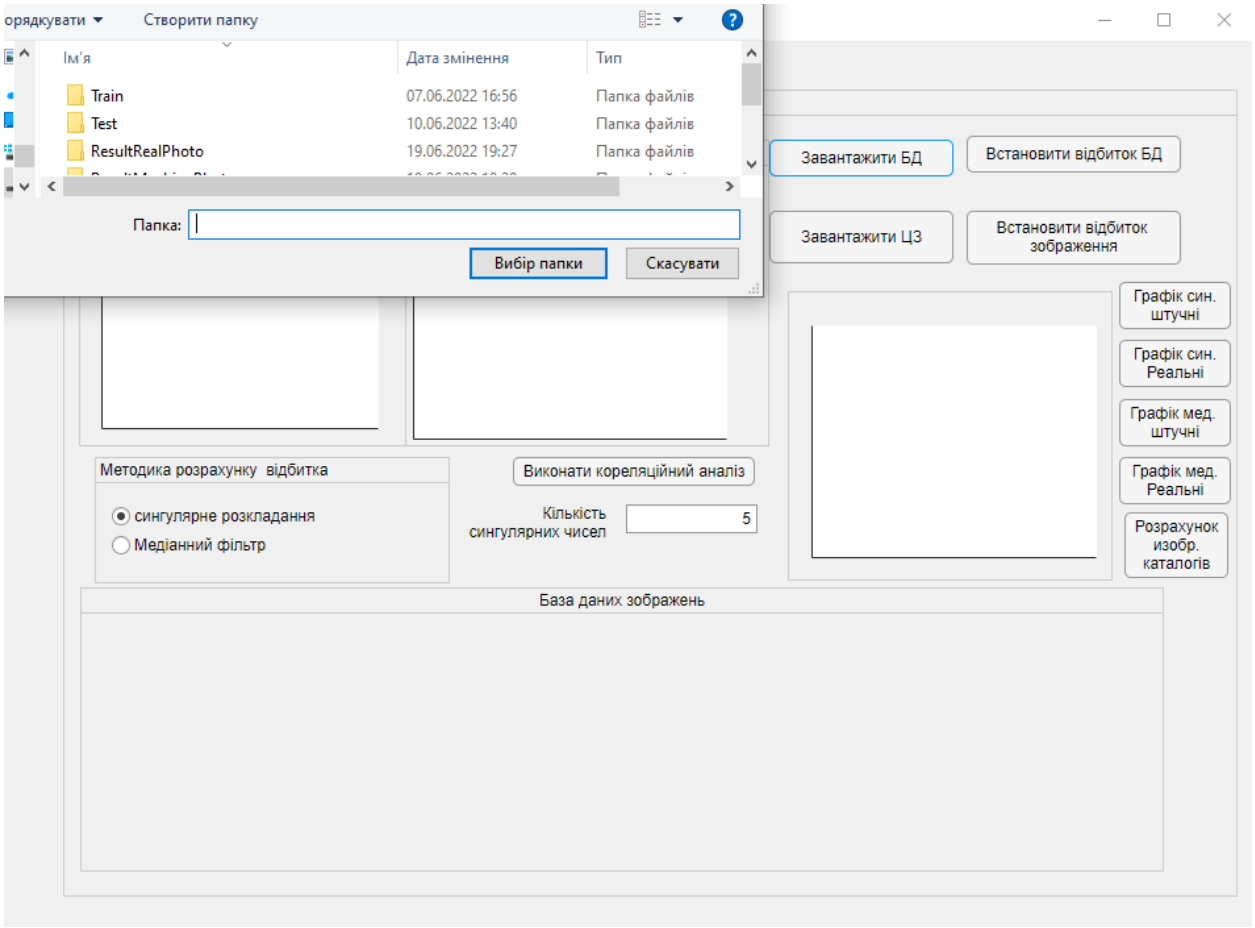


Рисунок 3.2 – Вибір каталогу бази даних зображень.

Після вибору каталогу усі зображення відображаються на нижній панелі. Потім необхідно вибрати метод розрахунку відбитка. Є два варіанти «сингулярне розкладання» та «Медіанний фільтр». Встановивши метод розрахунку, натискаємо на кнопку «Встановити відбиток БД».

У разі вибору «сингулярне розкладання» необхідно встановити у водному полі «Кількість сингулярних чисел» необхідне число. Це важливо для розрахунку усередненого шуму бази даних у зазначеній методиці розрахунку.

За методикою «Медіанний фільтр» необхідно завантажити базу даних зображень, потім натиснути на кнопку «Встановити відбиток БД» і він відображається в крайньому лівому вікні.

Потім вибираємо кнопку «Завантажити ЦЗ» та вибираємо будь-яке зображення, воно відображається у крайньому правому вікні. Щоб отримати шум вибраного зображення необхідно натиснути на кнопку "Встановити відбиток зображення" і після шум відобразиться в середньому вікні.

Для отримання ідентифікації необхідно натиснути кнопку «Виконати кореляційний аналіз» і на екрані з'явиться результат на рисунку (3.3).

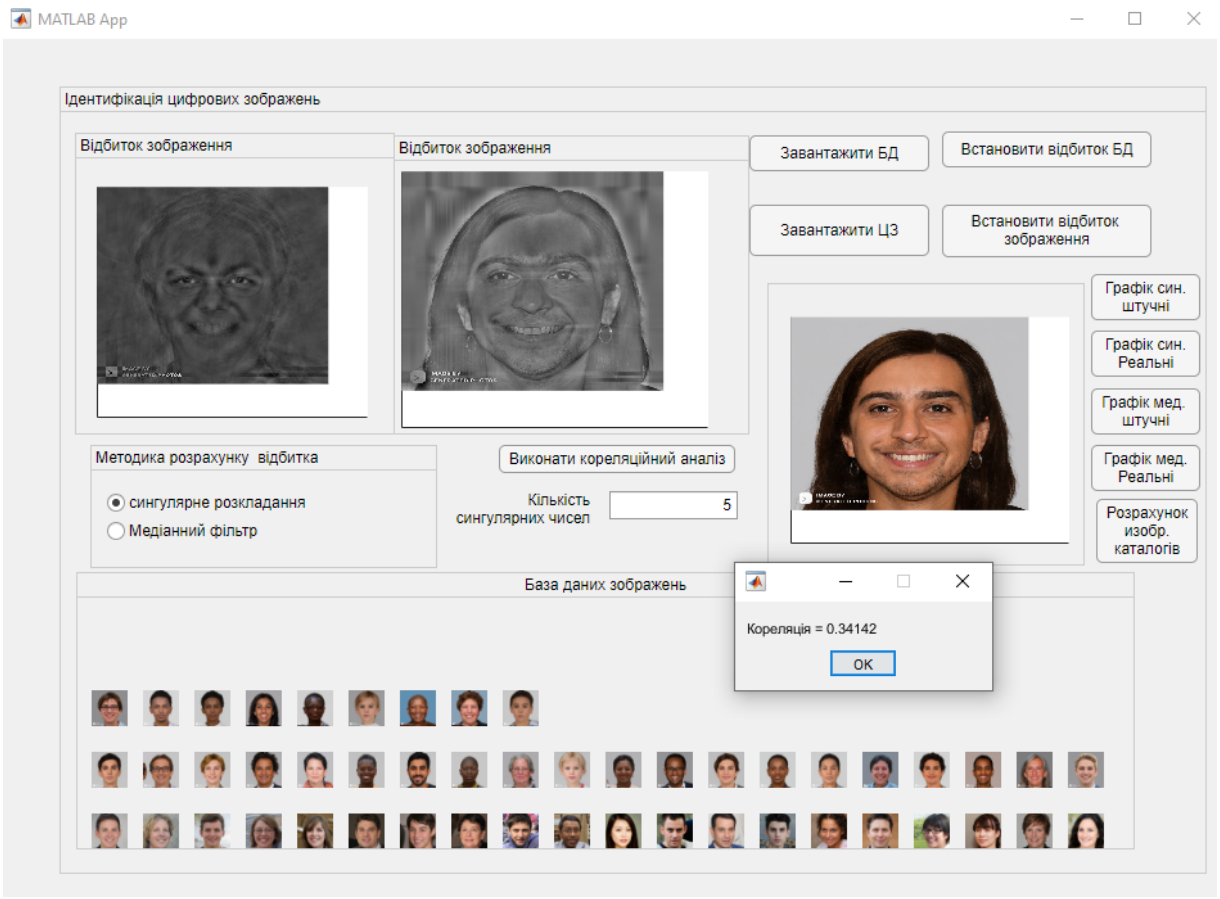


Рисунок 3.3 – Результат кореляції зображень.

Для отримання результатів зображень поміщених у каталог MachinePhoto – зображення сформовані за допомогою нейронної мережі, необхідно натиснути кнопку «Графік син. штучні» за методикою сингулярного розкладання.

Для отримання результатів зображень поміщених у каталог RealPhoto – реальні фотографії, необхідно натиснути кнопку «Графік син. Реальні» за методикою сингулярного розкладання.

Для отримання результатів зображень поміщених у каталог MachinePhoto – зображення, сформовані за допомогою нейронної мережі, необхідно натиснути на кнопку «Графік мед. штучні» за методикою медіанного фільтра.

Для отримання результатів зображень поміщених у каталог RealPhoto – реальні фотографії, необхідно натиснути кнопку «Графік мед. Реальні» за методикою медіанного фільтра.

Для отримання результатів обробки каталогу зображень для подальшого відображення на графіках, необхідно натиснути кнопку "Розрахунок изобр.каталогів"

У розділі розглянуто моменти реалізації програмного продукту із використанням середовища Matlab для реалізації програми ідентифікації зображення. Описано зміст основних функцій програми, що працюють у середовищі Matlab та описано цільове призначення кожної. Подано і розглянуто інтерфейс програми і описані основні функціональні елементи програми і порядок дій для отримання підсумкового результату. Описано можливі режими роботи програми. Наведено особливості роботи з програмою в окремому режимі.

4 ОХОРОНА ПРАЦІ

Кожен громадянин України має право на охорону праці, гарантоване конституцією України. Для успішного виконання завдань програміста зайнятого розробкою матеріалів додатків потрібно забезпечити системи правових соціально-економічних та організаційно-технічних, а також санітарно-гігієнічних заходів та засобів. За трудовим договором або контрактом, власник офісу IT-Matlab має створити безпечні умови праці, забезпечити страхування від нещасних випадків, проводити періодичні медичні огляди працівників, а також оплачувати лікарняні листи у разі захворювання.

Власник офісу IT-Matlab має створити оптимальні умови програмування на всіх етапах виробничого процесу.

Розташування офісу IT-Matlab повинно перебувати в безпечному місці до якого можливо спокійно добиратися, навколо нього не повинно знаходитися нічого що може якось загрожувати життю співробітників. До нормальних умов праці відносяться нова будівля, справний ліфт, добре закріплені поручні сходів, будівля побудована за правилами санітарно-технічних, електричних, газо-технічних, з централізованим опаленням, правильно спроектованою вентиляцією, з встановленими датчиками диму і вогню, охоронною сигналізацією та правильним заземленням будівлі.

На робочому місці в окремій кімнаті використовується комп'ютер, екран, клавіатура, миша, принтер, роутер, кондиціонер. Комп'ютер має 4 ядерні процесори, 8 гб оперативної пам'яті, 1т внутрішньої пам'яті, споживання 850Вт. Екран має дозвіл 1920x1080, що споживає 75Вт. Клавіатура стандартна опукла. Миша steelseries Kinzu v3, споживання 5Вт. Принтер HP Laser 135a з дозволом 1200x1200 dpi, споживання 300Вт. Роутер TP-LINK TL-WR842N, робоча чистота 2.4 ГГц, споживання 22Вт. Кондиціонер COOPER&HUNTER CH-S09FTXQ-NG, все перераховане

обладнання знаходиться в кімнаті площею 25 м/2. Кімната має одне велике вікно розміром 2х4 метри на всю стіну, частина його може відкриватися повністю або на провітрювання. Для захисту працівників від небажаного світла, який може світити в очі в приміщенні з великим вікном встановлені жалюзі або штори, які можна відкривати або закривати коли буде зручно співробітнику. Кімната забезпечена 4 розетками з напругою в 220В і частотою струму 50Гц, трижильний провід з заземленням, підключення до лінії загального розподільного шитка через автомат 16А і потужністю споживання 3,5Квт, а також в кімнаті є кабелі локальної мережі офісу IT-Matlab. Всі кабелі локальної мережі заховані в короби з полімерного матеріалу, що не мають шкідливих речовин. На стелі знаходяться два великі джерела штучного освітлення на лампах Led по 15Вт х 2.

Кімната, де знаходиться персонал що працює на Matlab, підтримує оптимальні умови відповідно до санітарних норм. Протягом дня співробітник за комп'ютером розробляє програми, налагоджує код і пише оновлення для вже створених додатків компанії IT-Matlab. Робочий день має тривалість 8 годин 30 хвилин та обід тривалістю 30 хвилин.

Основними шкідливими та небезпечними факторами при роботі за комп'ютером в офісі IT-Matlab є: Порушення мікроклімату, Ненормативне Освітлення, Шкідливий рівень Шуму, Підвищена напруга в електричній мережі з імовірністю замикання крізь тіло людини.

Порушення температури мікроклімату на робочому місці може викликати захворювання через сильне охолодження або перегрівання.

У теплий період охолодження під кондиціонером, у холодний період охолодження без опалення.

У теплий період перегрів через відсутність охолодження кондиціонером, у холодний період надмірне опалення.

Порушення швидкості руху повітря на робочому місці може викликати захворювання.

У теплий період охолодження під кондиціонером, вентилятором та відкритим вікном внаслідок великої швидкості повітряного потоку. У холодний період охолодження внаслідок поганої теплоізоляції та щілин у вікнах.

У літній період часу в офісі є кондиціонер для урегулювання температури в приміщенні, температура має середній мінімум як за стандартом для роботи в теплі пори року. У холодніші пори року є опалення для утримання допустимої температури робочого процесу. Оптимальні та допустимі норми показників мікроклімату згідно з ДСН 3.3.6.042-99 у кімнаті офісу є такі показники: Температура повітря (оптимальна величина): у холодний період: 19 - 21 С, у теплий період 21–23С;

Допустима величина: у холодний період верхня межа 21-23 С, нижня 12-15 С; У теплий період верхня межа 27-29 С, нижня межа 17-18 С.

Відносна вологість повітря в кімнаті: оптимальна величина у холодний та теплий періоди 60 – 40 %; допустима величина: у холодний період – 75%; У теплий – 65% при температурі 26 С.

Швидкість руху повітря (оптимальна величина): у холодний період 0,2 м/с, у теплий 0,3 м/с; Допустима величина у холодний період не більше 0,3 м/с, у теплий 0,4 –0,2 м/с

Температура повітря в кімнаті 18-25 ° С, повітря обмін не менше 30 м3/год, інтенсивність теплового випромінювання на працюючих не перевищує 35Вт/м2 при опроміненні 50% тіла.

Коли ми заходимо в наш кабінет з права зверху знаходитися кондиціонер біля якого знаходиться одна розетка, далі праворуч знаходиться принтер з ще однією розеткою і біля вікна знаходиться робоче місце біля якого 2 останні розетки, робочий комп'ютер і всі прилеглі речі для його використання. Розрахункова площа, що приходить на одну особу 25 м/2, висота кімнати 3 м та об'єм 75м/3. Відповідно до ДСанПіН 3.3.2.007-98 з розрахунку на одне робоче місце, обладнане ПК, встановлено такі норми:

площа — не менше 6,0 м/2; об'єм - не менше 20,0 м/3. Проектована кімната укладається у всі норми. Кімната дозволяє запросити трьох гостей.

Освітлення кімнати офісу IT-Matlab стандартне як у більшості офісних будівель, добре освічене приміщення для продуктивної роботи. Освітлення в кімнатах офісу в різні часи дня здійснюється природним або штучним світлом. Оскільки природне освітлення непостійне в різні періоди доби та року, то воно доповниться штучним освітленням. Нормовані значення природного та штучного освітлення визначаються «Будівельними нормами та правилами» (СНіП II-4-79). Відповідно до норм програмування відноситься до середньої групи точності. Для цієї групи освітленість при штучному освітленні у разі комбінованого освітлення має бути 300-500 Лк, ми встановимо 500 Лк, загальне світло в кімнатах має бути 400 Лк, ми встановимо 400 Лк. Коефіцієнт природного освітлення в кімнатах при верхньому або комбінованому освітленні має бути 3.5%, ми встановимо 3.5%. Щоб уникнути втоми очей, проблем із зором та погіршення роботи співробітника, ми створюємо так що поверхня столу освітлена рівномірно на підставі встановлених норм. А також ми створюємо освітлення так, щоб не зліпити поверхню столу.

Основними нормативними документами з охорони праці щодо вібрації є ГОСТ 12.1.012-90 та ДСН 3.3.6.039-99. Нормальний рівень шуму в офісах встановлено в діапазоні 50 децибел, максимально значення рівнів звуку та шуму згідно з ДСН.3.3.6.037-99 є 65 децибел. Комфортна робота супроводжується повною тишею для того, щоб зосередитися на виробничих завданнях.

4.1.4 Електронна напруга

Щоб забезпечити захист людей від шкідливого та небезпечного впливу електричного струму, електричної дуги, електромагнітного поля та статичної електрики, існує правила електробезпеки.

Робота з електричними приладами повинна проходити тільки після того, як працівник дізнався, як працює пристрій, для власної безпеки і для того, щоб не зламати його. Вся робоча техніка у приміщенні має бути справною. В офісі використовується напруга змінного струму 220В та 50Гц. У кожній кімнаті є розетки та вимикачі світла, які мають контакт із заземленням. Щоб уникнути псування обладнання та удару струмом співробітників в офісі, по радіусу будівлі розташоване контурне заземлення. Основними причинами електротравм офісу IT-Matlab можуть бути: випадковий дотик до неізольованих проводів живлення від комп'ютера, застосування нестандартних або несправних подовжувачів напругою 220 або 127 В або недотримання правил використання комп'ютерного обладнання.

Будівля офісу IT-Matlab пожежонебезпечна і має категорію В. На дверях кожної кімнати офісу IT-Matlab повішено план евакуації з будівлі у разі пожежі. У кожному кабінеті офісу IT-Matlab перебуває два вогнегасники придатні для використання та які пройшли перевірку Згідно з НАПБ А.01.001-2004 “Правила пожежної безпеки в Україні”

Конструкція будівлі спроектована так що забезпечує швидкий вихід із кожної кімнати офісу у разі пожежі.

У кожній кімнаті офісу має бути встановлена система автоматичної пожежної сигналізації та пожежогасіння.

Для індивідуального захисту в кожній кімнаті працівник офісу забезпечений респіратором або протигазом.

Посилаючись на закон України щодо пожежної безпеки, він встановлює правові, технічні та соціальні аспекти пожежної безпеки. Для забезпечення пожежної безпеки проведемо аналіз стану офісу IT-Matlab щодо виникнення пожежі або вибуху.

Офіс споживає електричну енергію, яка подається по електричній проводці та кабелям. У момент споживання енергії можливий стрибок електрики до 380В, дана ситуація може статися через розбалансування

трифазного підведення до офісу, після цього згоряє все обладнання. Щоб запобігти стрибкам електрики потрібно Реле контролю напруги в розподільчому щитку живлення на кожну лінію офісу. Відповідальність за це лежить на електрику та проектувальнику електричної мережі.

Всі пристрої підключені до мережі в разі несправності можуть спалахнути через легкозаймистий матеріал корпусу. Щоб уникнути цього, потрібно проводити профілактику та діагностику комп'ютера та всіх електронних пристроїв і не мати дотику один до одного. Перевірку та нагляд здійснюють Системний адміністратор, уповноважений керівництвом офісу пожежної безпеки. Щоб уникнути ненавмисної пожежі, що виникає при: Курінні в недозволеному місці, неправильно організованій роботі по зварюванню або пайці при усуненні експлуатаційних неполадок та інших аварійних робіт. Щоб запобігти цим ситуаціям, нам потрібно прийняти "правила пожежної безпеки" для офісу IT-Matlab і довести їх до кожного співробітника під підпис. Для забезпечення контролю, дотримання правил, відповідальними за це є уповноважений керівництвом офісу пожежної безпеки.

Поява пожежі можлива під час навантаження електричної мережі. Це може статися через одночасне використання електричних приладів у приміщенні у великій кількості на одній лінії, на якій у наслідках може статися пожежа. Щоб уникнути такого результату, нам потрібно правильно встановлювати автомати на лінії. Відповідальність за це лежить на електрику та проектувальнику електричної мережі.

Займання також можливе при влучення блискавки в будівлю, якщо вона не обладнана блискавковідводом. Щоб запобігти цій ситуації потрібно, щоб проектувальник будівлі та електричної мережі передбачив блискавковідвід.

Для роботи з використанням відкритого вогню, згідно з правилами пожежної безпеки офісу IT-Matlab, визначаються наказами, розпорядженнями, інструкціями офісу IT-Matlab.

Для початку тимчасових пожежонебезпечних робіт підряднику потрібно отримати допуск від IT-Matlab, який управляв офісом. Після цього повинні бути проведені всі підготовчі роботи, які виключають виникнення пожежі. Після закінчення пожежонебезпечних робот виконавець протягом 2 годин після закінчення робот повинен переконатися, що на місці проведення робіт не залишилося пожежонебезпечних речей.

До проведення пожежонебезпечних робіт, необхідно з місця проведення робіт видалити всі електронні прилади, меблі, що легко горять, легко займисті предмети інтер'єру і не можливо починати роботи з відкритим вогнем поблизу з невисохлими лакофарбовими матеріалами. Під час пожежі ми маємо евакуюватися згідно з планом евакуації, повинні бути світильники евакуаційного освітлення, двері евакуаційних виходів повинні зачинятися лише на внутрішні закрепи. Під час організації евакуаційних шляхів не допускається: Ліфт, ескалатор, розсувні двері, заставлений прохід евакуації меблями та обладнанням.

В організації IT-Matlab призначена особа, яка відповідає за експлуатацію вогнегасників, а також дотримання правил пожежної безпеки. Вибір типу та необхідна кількість вогнегасників здійснюється відповідно до Правил експлуатації та типових норм згідно з наказом Міністерства внутрішніх справ України від 15 січня 2018 р. № 25. У кожному кабінеті повинен бути як мінімум один вогнегасник з масою заряду від 3кг. Один вогнегасник з масою заряду від 3 кг. може охоплювати площу до 20 м/2. Можливо використовувати такі типи вогнегасників: Порошковий, Водопенний, Водяний або Вуглекислотний. В офісах для збереження працездатності електроніки необхідно використовувати Вуглекислотний вогнегасник або Порошковий.

Система протипожежної сигналізації має бути змонтована та введена в експлуатацію. Ця система має працювати без участі людей. До систем протипожежного захисту офісу IT-Matlab належать: автоматичні системи пожежогасіння, системи пожежної сигналізації, системи оповіщування про пожежу та управління евакуюванням людей, системи димо- та тепловидалення та підпору повітря, системи централізованого пожежного спостереження та системи диспетчиризації з центральним пунктом.

У разі пожежі в офісі IT-Matlab порядок дій співробітників повинен бути наступним: повідомити службу 101 по телефону, при цьому потрібно сказати оператору точне розташування офісу IT-Matlab, кількість поверхів у будівлі, точне місце загоряння, чітку обстановку на пожежі, наявність людей, що горять в будівлі та своє повне ім'я та прізвище. По можливості допомогти евакуації людей та за допомогою вогнегасника загасити загоряння. Включити оповіщення про пожежу. Проінформувати керівництво про возгорання. Відповідальна особа за пожежну безпеку офісу IT-Matlab повинна дотримуватися точних інструкцій та зробити наступні дії : Перевірити чи викликали пожежну службу за телефоном 101, у разі знаходження людей у будівлі потрібно здійснити негайну евакуацію їх, прибрати всіх працівників, які не пов'язані з гасінням пожежі, забезпечити евакуацію людей та матеріальних цінностей, забезпечити щоб робітники що брали участь у гасінні пожежі, дотримувалися правил безпеки праці. Під час пожежі в офісі IT-Matlab припиняється вся робота і вимикається вся електрика, крім живлення протипожежної захисту. Коли на місце прибудуть пожежники, потрібно забезпечити їм без перешкодного проходу до місця займання. Після прибуття пожежників потрібно вислухати консультацію головного пожежника та провести перекличку робочого персоналу виведеного з будівлі, щоб переконатися, що всі покинули будівлю.

У ході виконання були виявлені такі шкідливі фактори, властиві обєкту проектування: Порушення мікроклімату, Ненормативне Освітлення,

Шкідливий рівень Шуму, Підвищена напруга в електричній мережі з імовірністю замикання крізь тіло людини.

Для захисту працівників потрібно регулювати рівень освітлення, відстежувати час безперервної роботи працівника і нормувати регулярними перервами. Основними джерелами виникнення пожежі є обладнання, пошкодження чи некоректна робота якого може спричинити пожежу класу В. Для запобігання виникненню пожеж потрібно передусім підвищувати надійність обладнання за рахунок регулярної перевірки та атестації обладнання. Також необхідно мати порошко

ВИСНОВКИ

На закінчення можна відзначити низку ключових моментів. На початку дослідження було поставлено основну мету роботи – як можна розпізнати штучно згенеровані зображення за допомогою глибоких нейронних мереж та реальних фотографій.

Основним способом вирішення ідентифікації зображення штучне чи реальне, було обрано метод знаходження шуму зображення.

У процесі дослідження над темою проекту було вивчено численні матеріали, що стосуються ідентифікації шумів та причини їх виникнення у цифрових камерах.

Розглянуто перший широко відомий спосіб знаходження шуму зображення, використовуючи медіанний фільтр. На ньому проведено базові дослідження зображень. Досліджено та проаналізовано ефективність даного методу. У ході дослідження створення штучних зображень де результатом є складання елементів обличчя різних людей. Щоб провести аналіз даного додавання було запропоновано використовувати сингулярне розкладання зображення для знаходження шуму зображень.

Проведено дослідження знаходження шумів у достатній кількості на базі пятдесяти реальних та пятдесяти нереальних зображень.

Досліджено створення шуму на основі реальних і не реальних зображень способом сингулярного розкладання по сингулярних числах в діапазоні від 5 до 250. Виявлено ефективність даного методу знаходження шуму внаслідок можливості варіацій кількості сингулярних чисел задіяних у створенні зображення. Метод дозволяє впевнено відокремити та ідентифікувати шум реальної фотографії та шум зображення створеного на основі нейронних мереж кореляції.

Метод знаходження шуму за сингулярним розкладанням можливо надалі запропонувати для ідентифікації зображень.

У ході дослідження даного методу створено програму, що реалізує

зазначений метод і дозволяє ідентифікувати зображення.

Отриманий інтерфейс дозволяє користувачам легко ідентифікувати зображення тестування.

Таким чином, всі поставлені завдання були повністю виконані і мета роботи була досягнута.

ПЕРЕЛІК ПОСИЛАНЬ

1. Старовойтов В.В., Голуб Ю.И. Цифровые изображения: от получения до обработки. Минск: ОИПИ НАН Беларуси, 2014. 202 с.
2. Гонсалес Р., Вудс. Р. Цифровая обработка изображений. М.: Техносфера, 2012. 1104 с.
3. Перелигін Б. В. Цифрова обробка зображень: Одеса: ОДЕУ, 2022. 185 с.
4. Сенсоры цифровых камер. URL: https://www.cambridgeincolour.com.ru/tutorials-ru/camera-sensors.htm
5. Фототехника Фиксация изображения в цифровой фотокамере. URL: https://fotokomok.ru/fiksaciya-izobrazheniya-v-cifrovoj- fotokamere/
6. Как получается цифровое фото – объясним доходчиво. URL: https://pksecurity.ru/blog/kak-poluchaetsya-cifrovoe-foto-obyasnim-dohodchivo
7. Zhao X.X., Lin W., Meng C., Miao Z. Generating Photographic Faces from the SketchGuided by Attribute Using GANJian. *IEEEAccess*. 2019. № 3. P.1–6.
8. Devaki P., Kumar P. Face Generation using Deep Convolutional Generative Adversarial Neural Network. *Bioscience Biotechnology Research Communications*. 2020. № 13. P.20–23.
9. William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery. 2.6 Singular Value Decomposition // Numerical Recipes in C. — 2nd edition. — Cambridge: Cambridge University Press. — ISBN 0-521-43108-5.
10. Tim Roughgarden & Gregory Valiant. The Singular Value Decomposition (SVD)and Low-Rank Matrix Approximations. The Modern Algorithmic Toolbox Lecture №9 P. 6-7.
11. Matlab. URL: https://www.mathworks.com/help/matlab/

12. Гонсалес Р., Вудс Р., Эдинс С. Цифровая обработка изображений в среде MATLAB. М.: Техносфера, 2006. 616 с.
13. Дьяконов В., Авраменкова И. МАТЛАБ. Обработка сигналов и изображений. Специальный справочник. Питер, 2002. 601с.
14. Matlab. URL: https://www.mathworks.com/help/matlab/

Додаток А. Лістинг програми

```
classdef Prog2a < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure             matlab.ui.Figure
        Panel                matlab.ui.container.Panel
        Panel_2              matlab.ui.container.Panel
        UIaxes_db            matlab.ui.control.UIAxes
        Panel_3              matlab.ui.container.Panel
        UIaxes_im            matlab.ui.control.UIAxes
        btn_load_db          matlab.ui.control.Button
        btn_view_db          matlab.ui.control.Button
        btn_detect_ident_db  matlab.ui.control.Button
        btn_load_im          matlab.ui.control.Button
        btn_view_im          matlab.ui.control.Button
        btn_detect_ident_im  matlab.ui.control.Button
        btn_analysis         matlab.ui.control.Button
        Panel_all_images     matlab.ui.container.Panel
        Panel_4              matlab.ui.container.Panel
        UIAxes               matlab.ui.control.UIAxes
        ButtonGroup          matlab.ui.container.ButtonGroup
        Button_Median        matlab.ui.control.RadioButton
        Button_SVD           matlab.ui.control.RadioButton
        EditFieldLabel       matlab.ui.control.Label
        EditField            matlab.ui.control.NumericEditField
        Button               matlab.ui.control.Button
        Button_2             matlab.ui.control.Button
        Button_3             matlab.ui.control.Button
        Button_4             matlab.ui.control.Button
        Button_5             matlab.ui.control.Button
    end


    properties (Access = private)
        PathTestImg  %  Description    dir  all    images  neural
network
        Images
        ImageTest
        im_filter
        prnu
        FilenameItem
         CatalogRealPhoto="D:\1\RealPhoto"
         CatalogMachinePhoto="D:\1\MachinePhoto"

         CatalogResultRealPhoto="D:\1\ResultRealPhoto"
         CatalogResultMachinePhoto="D:\1\ResultMachinePhoto"
         Images2
         ImageOne
         Arraysingularvalue =[5 10 15 20 25 30 35 40 45 50 55 75
```

```
100 125 150]
         imagesArrbds
         bdsf
    end

    methods (Access = private)

        function results = loadcatal(app,catalog1,tipcataloga)
            files = dir(catalog1);
            ArrayCorelaciaMedian = [];
           ArrayCorelaciaSing = [];
           N=512;
width = N;
height = N;
           imagesArrbds = zeros( height, width,15);
            bdsf= true;
           p=1;
            for k = 1 : length(files);
        if ~files(k).isdir
            %filename =
            [l, name, ext] = fileparts(files(k).name);

            if strcmpi(ext, '.jpg')
                catalog=char(catalog1);
               % ff= catalog1+'\'+name ext;
                app.Images2{p,1}   =(imread([catalog   '\'   name
ext]));
                Filefolder=files(k).folder;
                Filename=files(k).name;
                FilenameItem=                                fullfile(
Filefolder,'\',Filename);


                app.ImageOne=app.Images2{p};
               ShumMedian= Calkshum(app,1,0);
               ShumBazuDanuhMed = CalkshumBazuDanuh(app,1,0);
               Korelacia                                              =
RaschetCorela(app,ShumBazuDanuhMed,ShumMedian);

               ArrayCorelaciaMedian
=[ArrayCorelaciaMedian,[Korelacia]];

               nn=size(app.Arraysingularvalue,2);
               ArrayCorelaciaSing= [];
                for s = 1:nn
                    ks=app.Arraysingularvalue(s);
                    ShumSing = Calkshum(app,2,ks);
                   if(bdsf)

               ShumBazuDanuhSing = CalkshumBazuDanuh(app,2,ks);
               imagesArrbds(:,:,s)= ShumBazuDanuhSing;
```

```
                    else
                        ShumBazuDanuhSing= imagesArrbds(:,:,s);
                end
               Korelacia                                                =
RaschetCorela(app,ShumBazuDanuhSing,ShumSing);

ArrayCorelaciaSing=[ArrayCorelaciaSing,[Korelacia]];
                end
                bdsf=false;
                Filename_RezultatuCorelaciiSing=
app.CatalogResultMachinePhoto   +"\"+   "Sinfile"+   tipcataloga
+name+ ".bin";
           Filename_RezultatuCorelaciiSing                          =
char(Filename_RezultatuCorelaciiSing);

writearraytofile(app,Filename_RezultatuCorelaciiSing,ArrayCorela
ciaSing);

                p=p+1;
            end

        end
            end
            Testr=ArrayCorelaciaMedian;

            Filename_RezultatuCorelaciiMedian=
app.CatalogResultMachinePhoto  +"\"+  "Medianfile"+  tipcataloga  +
".bin";
          Filename_RezultatuCorelaciiMedian                          =
char(Filename_RezultatuCorelaciiMedian);

writearraytofile(app,Filename_RezultatuCorelaciiMedian,ArrayCore
laciaMedian);



        end

        function results = graficresult(app,catalog1)

        end

        function results = Calkshum(app,tipalgoritm,countsin)
        N=512;
        width = N;
        height = N;
        rez = zeros( height, width );

            if (tipalgoritm==1) %Median
               image = app.ImageOne;
              image = image(:,:,3);
```

```
              image = double(image);
              image = image(1:height,1:width);
              im_filter = medfilt2( image, 'symmetric' );
            end

            if tipalgoritm==2 %Singular
                image = app.ImageOne;

       image = rgb2grey(image);
      image = double(image);
      image = image(1:height,1:width);
      [U,S,V]=svd(image);
       V=V';
        sinkat = countsin;

       rezf = zeros( height, width );

       for k2 = 1 : sinkat
           Ui= U(:,k2);
           Vi = V(k2,:);
           rezf=rezf +S(k2,k2)*Ui*Vi;

       end
        minval=min(rezf,[],"all");
       maxval=max(rezf,[],"all");
       im_filter =rezf;

            end

            im_filter = image - im_filter;
            results =im_filter;



i_min=min(min(app.im_filter));
test=app.im_filter;
test=test- i_min;
i_max=max(max(test));
test=test/i_max;
test=test*255;
test=round(test);
test=uint8(test);


        end

        function                    results                    =
CalkshumBazuDanuh(app,tipalgoritm,countsin)
            N=512;
            width = N;
            height = N;
```

```
            rez = zeros( height, width );

            for k = 1 : size(app.Images,1)
            image = app.Images{k};


            if (tipalgoritm==1) %Median
              image = image(:,:,3);
     image = double(image);
     image = image(1:height,1:width);
     im_filter = medfilt2( image, 'symmetric' );
            end

            if tipalgoritm==2 %Singular

                          image = rgb2grey(image);
     image = double(image);
     image = image(1:height,1:width);
     [U,S,V]=svd(image);
      V=V';
    sinkat= countsin;

      rezf = zeros( height, width );

      for k2 = 1 : sinkat
          Ui= U(:,k2);
          Vi = V(k2,:);
          rezf=rezf +S(k2,k2)*Ui*Vi;

      end
%       minval=min(rezf,[],"all");
%       maxval=max(rezf,[],"all");
      im_filter =rezf;

            end

            im_filter = image - im_filter;
%      minval=min(im_filter,[],"all");
%       maxval=max(im_filter,[],"all");
     rez = rez + im_filter;


            end

           results = rez / size(app.Images,1);


 i_min=min(min(app.prnu));
 test=app.prnu;
 test=test- i_min;
 i_max=max(max(test));
```

```
 test=test/i_max;
 test=test*255;
 test=round(test);
 test=uint8(test);


        end

        function results = RaschetCorela(app,shum1,shum2)

             prnu=shum1;
prnu_vector = reshape( prnu, 1, numel( prnu ) );
mean_prnu = mean( prnu_vector );
p = prnu_vector - mean_prnu;

im_filter = shum2;
im_vector = reshape( im_filter, 1, numel( im_filter ) );
mean_image = mean( im_vector );
t = im_vector - mean_image;

correlation = ( t * ( p' ) ) / ( sqrt( t * t' ) * sqrt( p * p' )
);


       results =  correlation;


        end

        function  writearraytofile(app,Filename,ar)

            fileID = fopen(Filename,'w');
            fwrite(fileID,ar,'double');
            fclose(fileID);

        end

        function results = readarrayfromfile(app,Filename,size3)

            fileID = fopen(Filename);

            A = fread(fileID,[size3],'double');
            fclose(fileID);
             results =A;
        end
    end


    % Callbacks that handle component events
    methods (Access = private)
```

```
        % Button pushed function: btn_load_db
        function btn_load_dbButtonPushed(app, event)
           PathTestImg = uigetdir();

if PathTestImg ~=0

    app.PathTestImg=PathTestImg;
    files = dir(PathTestImg);
    p=1;
       f = app.Panel_all_images;
       app.Panel_all_images.BackgroundColor=[0.94,0.94,0.94];

    for k = 1 : length(files)
        if ~files(k).isdir

            [l, name, ext] = fileparts(files(k).name);

            if strcmpi(ext, '.jpg')
                app.Images{p,1}  =(imread([PathTestImg  '\'  name
ext]));
               Filefolder=files(k).folder;
                Filename=files(k).name;
                FilenameItem=                              fullfile(
Filefolder,'\',Filename);



             if p<21
                  im  =  uiimage(f,'position',[(p-1)*28+12*p  0  28
28]);
             elseif p<41
                  im  =  uiimage(f,'position',[(p-21)*28+12*(p-20)
48 28 28] );
             elseif p<61
                  im  =  uiimage(f,'position',[(p-41)*28+12*(p-40)
96 28 28] );
             else
                  im  =  uiimage(f,'position',[(p-61)*28+12*(p-60)
144 28 28] );

             end









             im.ImageSource=FilenameItem;
```

```
            app.Panel_all_images;

                p=p+1;
             end
        end

    end

end

        end

        % Button pushed function: btn_view_db
        function btn_view_dbButtonPushed(app, event)
           n=fix(size(app.Images,1)/4)+1;
m=4;
figure;
for k=1:size(app.Images,1)
    subplot(n,m,k), imshow(app.Images{k});
    end



        end

        % Button pushed function: btn_detect_ident_db
        function btn_detect_ident_dbButtonPushed(app, event)
         N=512;
width = N;
height = N;
rez = zeros( height, width );

 h = waitbar(0,'Please wait...');
 for k = 1 : size(app.Images,1)
     image = app.Images{k};
  %   image = image(:,:,3);
     if  (app.Button_Median.Value ==1)
          image                        =                        (double(
image(:,:,3))+double(image(:,:,1))+double(image(:,:,2)))/3;
          %image=image(:,:,3);
     image = double(image);
     image = image(1:height,1:width);
     [U,S,V]=svd(image);
      V=V';
     sinkat = app.EditField.Value;
      %sinkat = 5;
      rezf = zeros( height, width );

      for k2 = 1 : sinkat
          Ui= U(:,k2);
          Vi = V(k2,:);
```

```
          rezf=rezf +S(k2,k2)*Ui*Vi;

       end
       minval=min(rezf,[],"all");
       maxval=max(rezf,[],"all");
       im_filter =rezf;

      end

      if  (app.Button_SVD.Value==1)
%                                          image      =      (double(
image(:,:,3))+double(image(:,:,1))+double(image(:,:,2)))/3;
%       image = double(image);
%       image = image(1:height,1:width);
     image = image(:,:,3);
      image = double(image);
      image = image(1:height,1:width);
      im_filter = medfilt2( image, 'symmetric' );
      end




      im_filter = image - im_filter;
      minval=min(im_filter,[],"all");
       maxval=max(im_filter,[],"all");
      rez = rez + im_filter;
      waitbar(k/size(app.Images,1),h)
 end
 app.prnu = rez / size(app.Images,1);
 close(h);
 app.UIaxes_db.Colormap=gray;
 imagesc(app.UIaxes_db,app.prnu)


 i_min=min(min(app.prnu));
 test=app.prnu;
 test=test- i_min;
 i_max=max(max(test));
 test=test/i_max;
 test=test*255;
 test=round(test);
 test=uint8(test);

 imagesc(app.UIaxes_db,test)
        end

        % Button pushed function: btn_load_im
        function btn_load_imButtonPushed(app, event)
            [file,path]=uigetfile('*.jpg');
if(~isequal(file,0))
```

```
    path=fullfile(path,file);
    I=imread(path);
    imagesc(app.UIAxes,I);
    app.ImageTest=I;

end

        end

        % Button pushed function: btn_view_im
        function btn_view_imButtonPushed(app, event)
          figure
          imshow(app.ImageTest);

        end

        % Button pushed function: btn_detect_ident_im
        function btn_detect_ident_imButtonPushed(app, event)
            N=512;
width = N;
height = N;
rez = zeros( height, width );

image = app.ImageTest;
%image = image(:,:,3)


     if  (app.Button_Median.Value ==1)

          image                        =                        (double(
image(:,:,3))+double(image(:,:,1))+double(image(:,:,2)))/3;
         % image=image(:,:,3);
     image = double(image);
     image = image(1:height,1:width);
     [U,S,V]=svd(image);
      V=V';
       sinkat = app.EditField.Value;
     % sinkat = 100;
      rezf = zeros( height, width );

      for k2 = 1 : sinkat
          Ui= U(:,k2);
          Vi = V(k2,:);
          rezf=rezf +S(k2,k2)*Ui*Vi;

      end
       minval=min(rezf,[],"all");
      maxval=max(rezf,[],"all");
      im_filter =rezf;

     end
```

```
      if  (app.Button_SVD.Value==1)
      image = image(:,:,3);
      image = double(image);
      image = image(1:height,1:width);
      im_filter = medfilt2( image, 'symmetric' );
      end


im_filter = image - im_filter;
minval=min(im_filter,[],"all");
       maxval=max(im_filter,[],"all");
app.im_filter=im_filter;


i_min=min(min(app.im_filter));
test=app.im_filter;
test=test- i_min;
i_max=max(max(test));
test=test/i_max;
test=test*255;
test=round(test);
test=uint8(test);

app.UIaxes_im.Colormap=gray;

imagesc(app.UIaxes_im,test)

        end

        % Button pushed function: btn_analysis
        function btn_analysisButtonPushed(app, event)
            prnu=app.prnu;
prnu_vector = reshape( prnu, 1, numel( prnu ) );
mean_prnu = mean( prnu_vector );
p = prnu_vector - mean_prnu;

im_filter = app.im_filter;
im_vector = reshape( im_filter, 1, numel( im_filter ) );
mean_image = mean( im_vector );
t = im_vector - mean_image;

correlation = ( t * ( p' ) ) / ( sqrt( t * t' ) * sqrt( p * p' )
);

msgbox(['Кореляція = ' num2str(correlation)]);

        end

        % Button down function: UIaxes_db
        function UIaxes_dbButtonDown(app, event)
```

```
            figure
          imshow(app.prnu);
        end

        % Button down function: UIaxes_im
        function UIaxes_imButtonDown(app, event)
            figure
          imshow(app.im_filter);
        end

        % Button pushed function: Button
        function ButtonPushed(app, event)
            figure
              for (i=1 :50)

            x=[5 10 15 20 25 30 35 40 45 50 55 75 100 125 150];
            Fn=  app.CatalogResultMachinePhoto  +"\"+  "Sinfile"+
'Mach' +num2str(i)+ ".bin";
            Fn=char(Fn);
            y=readarrayfromfile(app,Fn,15);
            y=y';
%

            plot(x,y);
            title("Штучні фотографії");
             xlabel("Число  сингулярних  чисел  для  формування
зображення");
             ylabel("Кореляція");
             hold on;
        end
      hold off;
        end

        % Button pushed function: Button_2
        function Button_2Pushed(app, event)
            figure
            for (i=1 :50)

            x=[5 10 15 20 25 30 35 40 45 50 55 75 100 125 150];
            Fn=   app.CatalogResultRealPhoto   +"\"+   "Sinfile"+
'Real' +num2str(i)+ ".bin";
            Fn=char(Fn);
            y=readarrayfromfile(app,Fn,15);
            y=y';


            plot(x,y);
            yticks ([-0.05   -0.03   -0.01  0.0  0.01   0.03   0.05
0.07 ])
%              YTick([-0.05  -0.04 -0.03 -0.02  - 0.01 0.0  0.01
0.02 0.03  0.04 0.05 0.06 0.07 0.08]);
```

```
            title("Реальні фотографії");
            xlabel("Число сингулярних чисел для формування
зображення");
            ylabel("Кореляція");
            hold on;
           end
           hold off;
        end

        % Button pushed function: Button_3
        function Button_3Pushed(app, event)
          figure
%           x=[1 2 3 4];
%           y=[0.7 0.5 0.06 0.001];
           x=[1:50];
           Fn=app.CatalogResultMachinePhoto  +"\"+  "Medianfile"+
'Mach' + ".bin";
           Fn=char(Fn);
            y= readarrayfromfile(app,Fn,50);
            y=y';
           plot(x,y);
           title("Штучні фотографії (Медіанний фільтр)");
           xlabel("Номер фотографії");
             ylabel("Кореляція");
        end

        % Button pushed function: Button_4
        function Button_4Pushed(app, event)
            figure
%           x=[1 2 3 4];
%           y=[-0.004 0.001 0.0002 0.004];
           x=[1:50];
           Fn=app.CatalogResultRealPhoto   +"\"+   "Medianfile"+
'Real' + ".bin";
           Fn=char(Fn);
            y= readarrayfromfile(app,Fn,50);
            y=y';
           plot(x,y);
           title("Реальні фотографії (Медіанний фільтр)");
           xlabel("Номер фотографії");
             ylabel("Кореляція");
        end

        % Button pushed function: Button_5
        function Button_5Pushed(app, event)
            %proveraem raschetnue failu
files = dir(app.CatalogRealPhoto);
l1=length(files);
if(l1<3)
   loadcatal(app,app.CatalogRealPhoto,'Real');
end
```

```
      files = dir(app.CatalogMachinePhoto);
l1=length(files);
if(l1<3)
   loadcatal(app,app.CatalogMachinePhoto,'Mach');
end

            graficresult(app,app.CatalogResultRealPhoto);
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are
created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 952 675];
            app.UIFigure.Name = 'MATLAB App';

            % Create Panel
            app.Panel = uipanel(app.UIFigure);
            app.Panel.Title     =     'Ідентифікація     цифрових
зображень';
            app.Panel.Position = [47 26 893 613];

            % Create Panel_2
            app.Panel_2 = uipanel(app.Panel);
            app.Panel_2.Title = 'Відбиток зображення';
            app.Panel_2.Position = [12 342 249 235];

            % Create UIaxes_db
            app.UIaxes_db = uiaxes(app.Panel_2);
            app.UIaxes_db.PlotBoxAspectRatio = [1.17365269461078
1 1];
            app.UIaxes_db.XTick = [];
            app.UIaxes_db.YTick = [];
            app.UIaxes_db.ButtonDownFcn = createCallbackFcn(app,
@UIaxes_dbButtonDown, true);
            app.UIaxes_db.Position = [12 0 221 206];

            % Create Panel_3
            app.Panel_3 = uipanel(app.Panel);
            app.Panel_3.Title = 'Відбиток зображення';
            app.Panel_3.Position = [260 342 277 233];

            % Create UIaxes_im
            app.UIaxes_im = uiaxes(app.Panel_3);
```

```
            zlabel(app.UIaxes_im, {''; ''})
            app.UIaxes_im.XTick = [];
            app.UIaxes_im.YTick = [];
            app.UIaxes_im.ButtonDownFcn = createCallbackFcn(app,
@UIaxes_imButtonDown, true);
            app.UIaxes_im.Position = [1 0 249 210];

            % Create btn_load_db
            app.btn_load_db = uibutton(app.Panel, 'push');
            app.btn_load_db.ButtonPushedFcn                    =
createCallbackFcn(app, @btn_load_dbButtonPushed, true);
            app.btn_load_db.Position = [537 547 140 28];
            app.btn_load_db.Text = 'Завантажити БД';

            % Create btn_view_db
            app.btn_view_db = uibutton(app.Panel, 'push');
            app.btn_view_db.ButtonPushedFcn                    =
createCallbackFcn(app, @btn_view_dbButtonPushed, true);
            app.btn_view_db.Enable = 'off';
            app.btn_view_db.Visible = 'off';
            app.btn_view_db.Position = [671 538 106 10];
            app.btn_view_db.Text = {'Переглянути БД'; ''};

            % Create btn_detect_ident_db
            app.btn_detect_ident_db     =     uibutton(app.Panel,
'push');
            app.btn_detect_ident_db.ButtonPushedFcn            =
createCallbackFcn(app, @btn_detect_ident_dbButtonPushed, true);
            app.btn_detect_ident_db.Position = [687 550 163 28];
            app.btn_detect_ident_db.Text = {'Встановити відбиток
БД'; ''};

            % Create btn_load_im
            app.btn_load_im = uibutton(app.Panel, 'push');
            app.btn_load_im.ButtonPushedFcn                    =
createCallbackFcn(app, @btn_load_imButtonPushed, true);
            app.btn_load_im.Position = [537 481 140 40];
            app.btn_load_im.Text = {'Завантажити ЦЗ'; ''};

            % Create btn_view_im
            app.btn_view_im = uibutton(app.Panel, 'push');
            app.btn_view_im.ButtonPushedFcn                    =
createCallbackFcn(app, @btn_view_imButtonPushed, true);
            app.btn_view_im.Enable = 'off';
            app.btn_view_im.Visible = 'off';
            app.btn_view_im.Position = [566 459 106 22];
            app.btn_view_im.Text = 'Переглянути ЦЗ';

            % Create btn_detect_ident_im
            app.btn_detect_ident_im     =     uibutton(app.Panel,
'push');
```

```
            app.btn_detect_ident_im.ButtonPushedFcn             =
createCallbackFcn(app, @btn_detect_ident_imButtonPushed, true);
            app.btn_detect_ident_im.WordWrap = 'on';
            app.btn_detect_ident_im.Position = [687 480 163 41];
            app.btn_detect_ident_im.Text = {'Встановити відбиток
зображення'; ''};

            % Create btn_analysis
            app.btn_analysis = uibutton(app.Panel, 'push');
            app.btn_analysis.ButtonPushedFcn                    =
createCallbackFcn(app, @btn_analysisButtonPushed, true);
            app.btn_analysis.Position = [342 312 184 22];
            app.btn_analysis.Text   =   'Виконати   кореляційний
аналіз';

            % Create Panel_all_images
            app.Panel_all_images = uipanel(app.Panel);
            app.Panel_all_images.TitlePosition = 'centertop';
            app.Panel_all_images.Title = 'База даних зображень';
            app.Panel_all_images.Position = [13 19 824 216];

            % Create Panel_4
            app.Panel_4 = uipanel(app.Panel);
            app.Panel_4.Position = [551 240 247 220];

            % Create UIAxes
            app.UIAxes = uiaxes(app.Panel_4);
            app.UIAxes.PlotBoxAspectRatio  =  [1.23170731707317  1
1];
            app.UIAxes.XTick = [];
            app.UIAxes.YTick = [];
            app.UIAxes.Position = [13 4 227 203];

            % Create ButtonGroup
            app.ButtonGroup = uibuttongroup(app.Panel);
            app.ButtonGroup.Title    =    'Методика    розрахунку
відбитка';
            app.ButtonGroup.Position = [24 238 270 96];

            % Create Button_Median
            app.Button_Median = uiradiobutton(app.ButtonGroup);
            app.Button_Median.Text = 'сингулярне розкладання';
            app.Button_Median.Position = [13 40 159 22];
            app.Button_Median.Value = true;

            % Create Button_SVD
            app.Button_SVD = uiradiobutton(app.ButtonGroup);
            app.Button_SVD.Text = 'Медіанний фільтр';
            app.Button_SVD.Position = [13 18 122 22];

            % Create EditFieldLabel
```

```
            app.EditFieldLabel = uilabel(app.Panel);
            app.EditFieldLabel.HorizontalAlignment = 'right';
            app.EditFieldLabel.Position = [303 270 110 28];
            app.EditFieldLabel.Text = {'Кількість'; 'сингулярних
чисел'; ''};

            % Create EditField
            app.EditField = uieditfield(app.Panel, 'numeric');
            app.EditField.Position = [428 276 100 22];
            app.EditField.Value = 5;

            % Create Button
            app.Button = uibutton(app.Panel, 'push');
            app.Button.ButtonPushedFcn  =  createCallbackFcn(app,
@ButtonPushed, true);
            app.Button.Position = [803 431 85 36];
            app.Button.Text = {'Графік син.'; 'штучні'; ''; ''};

            % Create Button_2
            app.Button_2 = uibutton(app.Panel, 'push');
            app.Button_2.ButtonPushedFcn                          =
createCallbackFcn(app, @Button_2Pushed, true);
            app.Button_2.Position = [803 387 85 36];
            app.Button_2.Text = {'Графік син.'; 'Реальні'};

            % Create Button_3
            app.Button_3 = uibutton(app.Panel, 'push');
            app.Button_3.ButtonPushedFcn                          =
createCallbackFcn(app, @Button_3Pushed, true);
            app.Button_3.Position = [803 342 85 36];
            app.Button_3.Text = {'Графік мед. '; 'штучні'};

            % Create Button_4
            app.Button_4 = uibutton(app.Panel, 'push');
            app.Button_4.ButtonPushedFcn                          =
createCallbackFcn(app, @Button_4Pushed, true);
            app.Button_4.Position = [803 298 85 36];
            app.Button_4.Text = {'Графік мед.'; 'Реальні'};

            % Create Button_5
            app.Button_5 = uibutton(app.Panel, 'push');
            app.Button_5.ButtonPushedFcn                          =
createCallbackFcn(app, @Button_5Pushed, true);
            app.Button_5.BackgroundColor   =   [0.9608   0.9608
0.9608];
            app.Button_5.Position = [807 242 79 50];
            app.Button_5.Text   =   {'Розрахунок';   'изобр.';
'каталогів'};

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
```

```
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = Prog2a

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```