Міністерство освіти і науки України

Рівненський державний гуманітарний університет

Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота

за освітнім ступенем «магістр»

на тему: Інтелектуальна задача розпізнавання та пошуку складного графічного образу при невизначених умовах фіксації зображення

Виконала: магістрантка 2 курсу

групи М-КН-21

спеціальності 122 Комп’ютерні науки

Шевцова Наталія Вікторівна

Науковий керівник: к.т.н, доц., завідувач кафедри

інформаційних технологій та моделювання,

Сяський Володимир Андрійович

Рівне-2022

РЕФЕРАТ

У кваліфікаційній роботі запропоновано оригінальний підхід до вирішення інтелектуальної задачі розпізнавання складного графічного образу за умов нечітко визначених даних. В якості модельного прикладу розглянуто задачу ідентифікації та пошуку сузір'я на карті зоряного неба, що було сфотографоване при невідомих умовах фіксації зображення.

Об'єкт дослідження – методи, технології ідентифікації та пошуку образів.

Предмет дослідження – інтелектуальна система ідентифікації та пошуку графічних образів за умов нечітко визначених даних.

Розроблено процедуру ідентифікації об'єктів складного графічного образу, що дозволяє моделювати зображення окремих зірок наборами числових даних, які однозначно характеризують їх кількісний склад у образі, положення, розміри та взаємне розміщення. Побудовано ефективний алгоритм пошуку графічних образів меншого зображення у базовому образі з врахуванням можливих геометричних перетворень та наявності шумів.

Створено програмний застосунок для пошуку образу у великому зображенні при невизначених умовах його фіксації. Проведено тестування алгоритму на наборі модельних прикладів, яке показало його ефективність при застосуванні до зображення образу повороту, переміщення, масштабування або їх комбінацій.

Кваліфікаційна робота складається зі змісту, вступу, трьох розділів, висновків та списку використаних літературних джерел із 23 найменувань. До роботи додається реферат українською та англійською мовами. Загальний обсяг роботи складає 65 сторінок.

Ключові слова: розпізнавання образів, інтелектуальний аналіз, нечіткі дані, складний графічний образ.

ABSTRACT

At the qualification work an original approach to solving the intellectual problem of recognizing and searching for a complex graphic image under conditions of fuzzy data is proposed. The task of identifying and searching for a constellation on the map of the starry sky, which was photographed by uncertain conditions fixation image, as a model example is considered.

The object of research – methods, technologies for identification and search of images.

The subject of the research – is the intellectual system of identification and the search for graphic images under conditions of fuzzy data.

A procedure for identifying the objects of a complex graphic image is developed, which allows modeling the image of individual stars with sets of numerical data that uniquely characterize their quantitative composition in the image, position, size, and mutual placement. An effective algorithm for searching for graphic images of a smaller image in the base image is built, taking into account possible geometric transformations and the presence of noises.

A software application is created for searching for an object in a large image under uncertain conditions of its fixation. The algorithm is tested on a set of model examples, which showed its effectiveness when rotation, displacement, scaling, or their combinations applied to image.

The qualification work consists of a table of contents, an introduction, three parts, conclusions and a list of used literary sources from 23 titles. An abstract in Ukrainian and English is attached to the work. The total contents of work are 65 pages.

Key words: image recognition, intellectual analysis, fuzzy data, complex graphic image.

ЗМІСТ

ВСТУП

Актуальність теми. Розпізнавання образів широко використовується при розробці інформаційних систем, на які покладаються інтелектуальні функції прийняття рішень замість людини: медична діагностика, криміналістична експертиза, пошук інформації, інтелектуальний аналіз даних тощо. Розробка методів машинного розпізнавання дозволяє розширити коло завдань, що виконуються комп'ютерами, і зробити обробку інформації більш автоматизованою.

Прикладами застосунків для такого типу завдань можуть бути системи розпізнавання тексту, комп’ютерного стеження, голосового перекладу, пошуку по зображенню тощо. Незважаючи на те, що більшість з цих завдань легко вирішуються людиною, досі не створено універсальних комп'ютерних програм для реалізації задачі ідентифікації у загальному вигляді. Існуючі системи розпізнавання образів призначені для роботи лише у окремих випадках із вузькою областю застосування.

Мета: розробка алгоритму розв’язування інтелектуальної задачі розпізнавання та пошуку складного графічного образу при невизначених умовах фіксації зображення і його програмна реалізація.

Для досягнення поставленої мети потрібно виконати наступні **завдання**:

– дослідити предметну область інтелектуального аналізу та розпізнавання образів;

– конкретизувати задачу розпізнавання образів на модельному прикладі пошуку сузір’я на мапі зоряного небу при невизначених умовах зйомки (географічне положення, дата і час, кут повороту об’єктиву, масштаб);

– розробити процедуру ідентифікації об’єктів складного графічного образу, що дозволяє моделювати зображення окремих зірок наборами числових даних;

– побудувати ефективний алгоритм пошуку графічних образів меншого

зображення у базовому образі з врахуванням можливих геометричних перетворень та наявності шумів;

– створити програмний застосунок для пошуку образу при невизначених умовах фіксації зображення;

– дослідити ефективність алгоритму пошуку складного графічного образу на множині тестових прикладів.

Об'єкт дослідження – методи, технології ідентифікації та пошуку образів.

Предмет дослідження – інтелектуальна система ідентифікації та пошуку графічних образів за умов нечітко визначених даних.

Методи досліджень передбачають використання методів аналітичної геометрії, обробки графічних зображень, теорії розпізнавання образів.

Наукова новизна отриманих результатів. Побудовано ефективний алгоритм ідентифікації графічних образів, що базується формалізації їх геометричних параметрів у вигляді наборів числових даних, та пошуку таких об'єктів в складних графічних зображеннях при нечітко визначених умовах. Запропоновані шляхи модифікації та вдосконалення алгоритму пошуку.

Апробація результатів дослідження. Результати дослідження доповідалися на XV Всеукраїнській науково-практичній конференції «Інформаційні технології в професійній діяльності», яка проводилася в онлайн форматі 1 листопада 2022 року на базі Рівненського державного гуманітарного університету. Тези доповіді «Вирішення інтелектуальної задачі розпізнавання та пошуку складного графічного образу» опубліковані у збірнику матеріалів конференції [23].

Структура роботи. Кваліфікаційна робота складається зі змісту, вступу, трьох розділів, висновків та списку використаних літературних джерел. До роботи додається реферат українською та англійською мовами.

Загальний обсяг роботи складає 65 сторінок. Робота містить 13 рисунків, 14 таблиць та 16 лістингів з фрагментами коду програми. Список використаних літературних джерел складає 23 найменування.

У першому розділі «Загальні відомості з теорії розпізнавання образів» викладено понятійний апарат предметної галузі, сформульовано класичні постановки задачі розпізнавання образів, розглянуто основні підходи до її розв’язання. Другий розділ «Алгоритм ідентифікації та пошуку складного графічного образу та особливості його програмної реалізації» містить формалізацію постановки задачі на прикладі ідентифікації та пошуку сузір’я на карті зоряного неба, що було сфотографоване при невизначених умовах фіксації зображення. У цьому ж розділі наведено опис алгоритму ідентифікації складного графічного образу та його пошуку у великому зображенні. Також висвітлено особливості його програмної реалізації. У третьому розділі «Модельні експерименти для аналізу ефективності алгоритму пошуку складного графічного образу» проведено дослідження ефективності застосунку на множині тестових зразків даних.

РОЗДІЛ 1
ЗАГАЛЬНІ ВІДОМОСТІ З ТЕОРІЇ РОЗПІЗНАВАННЯ ОБРАЗІВ

1.1. Основні поняття теорії розпізнавання образів та актуальність її задач

Теорія розпізнавання образів – розділ кібернетики та суміжних дисциплін, що розвиває основи та методи класифікації та ідентифікації предметів, явищ, процесів, сигналів, ситуацій тощо об'єктів, які характеризуються кінцевим набором деяких властивостей і ознак [15].

Задачі розпізнавання зображень зустрічаються в різних сферах, зокрема охорона здоров’я, маркетинг, транспорт, геоінформаційні системи та соціальні мережі. Методи теорії розпізнавання образів можна використовувати для ідентифікації об’єктів на зображеннях, щоб класифікувати їх для майбутнього використання. Наприклад, виявлення обличь на зображеннях або розпізнавання тексту на сторінці. Розпізнавання образів є однією з найважливіших технологій, які розвиваються сьогодні, оскільки завдяки їм можна розв’язати низку проблем: проводити діагностику раку з більшою точністю або виявляти шахрайство за допомогою аналізу зображень банкнот.

Розпізнавання образів включається в ширшу наукову дисципліну – теорію машинного навчання, метою якої є розробка методів побудови алгоритмів, що здатні навчатися.

Предметом дослідження теорії розпізнавання образів є математичні методи, алгоритми, моделі, технології та засоби класифікації об’єктів шляхом визначення, обробки, аналізу, зберігання та використання їх ознак [4].

Методи розпізнавання образів базуються на багатьох математичних дисциплінах: статистиці, матричній алгебрі, аналітичній геометрії, диференціальному та інтегральному численні, аналізу даних, теорії нечітких множин, дослідженні операцій, теорії графів, математичному моделюванні.

Реалізація методів розпізнавання в автоматизованих системах, що користуються можливостями штучного інтелекту, призначених для вирішення

завдань діагностики, моніторингу, прогнозування, управління поведінкою складних систем у відповідності із заданими специфікаціями, є розділом теорії штучного інтелекту. Такі методи теорії розпізнавання, як кластерний аналіз, виявлення закономірностей в експериментальних даних, прогнозування різних процесів або явищ, широко використовуються в наукових дослідженнях.

Система розпізнавання образів – електронно-обчислювальний комплекс, здатний моделювати розумові процеси, властиві людині під час прийняття рішень із метою виявлення аналогій серед досліджуваних об’єктів [3].

Сучасні інформаційні системи розпізнавання образів є складовою частиною штучних інтелектуальних систем, використовуються в експертних системах, базах даних та базах знань, в інформаційних, прогнозних та інших системах. Зокрема комплексні інтелектуальні системи DataMining для прийняття рішень використовують ряд методів розпізнавання образів [4].

Можливість розпізнавання базується на подібності однотипних об'єктів. Незважаючи на те, що всі предмети та ситуації унікальні у строгому сенсі, між деякими з них можна знайти схожість за тією чи іншою ознакою. Звідси випливає поняття класифікації – розбиття всієї множини об'єктів на непересічні підмножини – класи, елементи яких мають деякі схожі властивості, що відрізняють їх від елементів інших класів.

Образ – це об’єкт, процес або явище реального та абстрактного світу, який розпізнається за даними (ознаками), що збираються та оброблюються індивідуально і у сукупності. Розпізнавання образу завжди супроводжується дією. В штучних інтелектуальних системах дія має вигляд порівняння з еталоном, увімкнення в роботу виконавчого пристрою, вилучення з розгляду образу, запису або видачі інформації тощо [4].

1.2. Обробка вхідних даних для розпізнавання образів

Наявність шумів та нечітких даних під час вимірювання значень ознак розпізнавання обумовлює необхідність їх попередньої обробки. При цьому

можуть виконуватися наступні операції: нормалізація, фільтрація, дискретизація та квантування, фрагментизація, диференціювання, об’єднання та згладжування ознак.

Нормалізацією називають метод попередньої обробки числових ознак у вхідних даних з метою зведення їх до деякої спільної шкали без втрати інформації про відмінність діапазонів. Необхідність нормалізації викликана тим, що різні ознаки навчального набору даних можуть бути представлені в різних масштабах та змінюватись у різних діапазонах, що унеможливлює їх порівняння з еталоном.

Фільтрацію даних виконують шляхом вилучення шумів з метою отримання достовірного зображення.

Дискретизація – процес перетворення безперервного сигналу на цифровий, шляхом вимірювання числових значень амплітуди сигналу через рівні проміжки часу. Кроком дискретизації називається інтервал, через який фіксується значення безперервного сигналу. Найбільш зручним з погляду організації обробки вхідних даних та природним способом дискретизації є представлення сигналів у вигляді вибірки їх значень в окремих, регулярно розташованих точках. Такий спосіб називають раструванням, а сукупність вузлів, у яких проводиться вимірювання – растром. Відновлення безперервної функції по дискретним значенням ознак виконується інтерполяційними методами.

Під квантуванням неперервної або дискретної величини розуміють розбивку діапазону її значень на скінченну кількість інтервалів. Існує також векторне квантування – розбиття простору можливих значень векторної величини на скінченну кількість областей. Квантування часто використовується при обробці цифрових сигналів, у тому числі при стисканні звуку й зображень. Оптимальне значення рівня квантування відповідає середині інтервалу квантування. У цьому випадку максимальна похибка в центрі проміжку не перевищує половини інтервалу квантування.

Фрагментизація має на меті розклад зображення на окремі фрагменти (сегменти та дискретні елементи) з наступним об’єднанням їх у більш складні форми. Універсального методу фрагментизації не існує. Найчастіше аналізується інформація про ділянку (границю) області: яскравість; товщина та/або напрям лінії; розмір і т. д.

Диференціювання ознак виконують для «загострення» величини ознаки (це відноситься до похідних по отриманих кривих яскравості, зміни контрастності зображення, збільшення/зменшення числових ознак і т. д.) [4].

Згладжування полягає в обробці сусідніх ознак, які не повинні суттєво відрізнятись. Наявність викидів, тобто значень, які виходять за межі нормальних значень змінної, може суттєво спотворити образ. Їх замінюють апроксимуючими значеннями з урахуванням сусідніх значень.

1.3. Постановка задачі розпізнавання образів та підходи до її вирішення

Класична постановка задачі розпізнавання образів формулюється наступним чином: дано множину об'єктів, кожен з яких потрібно віднести до певного класу, виділивши істотні ознаки із загальної маси несуттєвих даних.

Формальний запис даної задачі має вигляд: нехай довільний елемент ω множини образів M задано значеннями деяких ознак x_i, $i = 1, 2, ..., N$, причому набори характеристик однакові для всіх елементів множини. Сукупність ознак, представлена у вигляді N-вимірного вектора $I(\omega) = \left(x_1, x_2, ..., x_N\right)$, визначає повну характеристику образу ω.

У найбільш поширеному випадку атрибути мають вигляд числових значень, але можуть виражатися в термінах "так/ні", набувати значень із переліку можливих варіантів. Тому ознаки опису образу поділяють на числові, реальні, абстрактні, лінгвістичні, якісні, детерміновані, стохастичні, просторові, часові, причинно-наслідкові характеристики об’єкта та ін.

Якщо розглянути для прикладу опис людини, то отримаємо набір характеристик різного типу: числові ознаки (зріст, вагу), якісні дані

(порядність, ерудицію, психологічні дані), описові дані (лінгвістичні опис рис обличчя та одежі, колір очей, кваліфікацію, хобі) тощо.

В якості ознак також можуть виступати самі об’єкти розпізнавання образів, події, процеси, явища, які, у свою чергу, стають ознаками ієрархічно вищих образів [4].

Під час формування простору ознак розпізнавання виконують інформаційний аналіз характеристик символів. Для кожної з характеристик оцінюють її релевантність – ступінь відповідності до поставленої задачі, та інформативність – вплив майбутньої ознаки на ефективність вирішальних правил. Вирішальним правилом (або класифікатором) називається математичний вираз або алгоритм визначення належності об’єкта, що розпізнається, одному з класів розпізнавання [3].

Наприклад, те, що символи одного класу складаються з голосних літер, а іншого – з приголосних, не має відношення до задачі розпізнавання друкованого тексту, тобто дана ознака не буде релевантною.

Вхідні дані для визначення характеристик образу є сукупністю чітких або нечітких, повних або неповних, постійних або змінних, точних або з масивом неістотних деталей даних.

На множині M існує розбиття на класи об’єктів Ω_k (1.1)

$$M = \bigcup_{k=1}^{K} \Omega_k \,, \tag{1.1}$$

коли кожен елемент $\omega \in M$ належить одній з підмножин $\omega \in \Omega_k$ $\left(k = 1, 2, ..., K\right)$, де K – кількість класів. При цьому класи не можуть мати спільних елементів, що виражається рівністю (1.2)

$$\forall m, l \;\; (m \neq l) \;\; \Omega_m \cap \Omega_l = \varnothing \,. \tag{1.2}$$

Якщо розглянути різниця між об’єктами різних класів досить розмита, то умова (1.2) не виконується і виникає поняття нечітких класів розпізнавання. Такі множини можуть перетинатися в просторі ознак розпізнавання, а їх

реалізації характеризуються ступенем належності до декількох класів розпізнавання.

Універсумом зветься виділена для класифікації сукупність образів з однаковим вмістом подібних суттєвих ознак $(x_1, x_2, ..., x_N)$. Алфавіт образів розділяє універсум на однотипні класи. Часто до існуючих K класів додається ще один $(K+1)$ для нерозпізнаних образів [4].

Розбиття множини образів M на класи Ω_k може бути здійснене наступними способами:

1. Переліком*:* в цьому випадку кожен клас задається шляхом вказання усіх його елементів. Такий підхід використовується тоді, коли доступна повна апріорна інформація про всі об'єкти розпізнавання. Образи, що аналізуються системою порівнюються із обраними представниками класів і встановлюється їх приналежність до того класу, якому належать найбільш подібний до них зразок. Такий підхід називають методом порівняння з еталоном. Його можна застосувати при розпізнаванні друкованих символів певного шрифту. Недоліком цього способу є слабка стійкість до шумів і спотворень.

2. Визначенням загальних властивостей: в цьому випадку клас задається сукупністю ознак, властивих усім його елементам. Розпізнаваний об'єкт не порівнюється безпосередньо із групою еталонних об'єктів, а виділяються значення певного набору ознак у його первинному описі, які потім порівнюються із заданими ознаками класів. Такий підхід називається зіставленням за ознаками. Він більш економний за метод порівняння з еталоном у питанні кількості пам'яті, необхідної для зберігання описів класів. Крім того, він допускає деяку варіативність розпізнаваних образів. Однак, головною складністю є визначення повного набору ознак, що точно відрізняють членів одного класу від решти елементів.

3. Кластеризацією: у цьому випадку розпізнавання здійснюється на основі розрахунку відстані опису об'єкта (вектора) до кожного з наявних кластерів. Кластер складається з множини подібних реалізацій образу, які можна відділити за певними критеріями від інших об’єктів. У процесі

кластеризації виконується не лише пошук елементів класу, але й формування вирішальних правил для кожного кластера. Якщо кластери досить рознесені у просторі, то метод оцінки відстаней від розглядуваного образу до кожного з кластерів працює добре. Складність розпізнавання зростає, якщо кластери перекриваються. Зазвичай це є наслідком недостатності вихідної інформації та може бути вирішено збільшенням кількості ознак, що беруться до уваги.

Якість класифікації об’єктів істотно залежить від вибору міри їх близькості (подібності). Мірою близькості називається величина, що має граничне значення і здатна збільшуватися із зменшенням відстані між реалізаціями класів розпізнавання. Розглянемо основні способи визначення близькості між об’єктами для кількісних шкал вимірювань [3].

Найбільш поширеною мірою близькості в теорії розпізнавання образів є аналог евклідової відстані:

$$d(x_m, x_l) = \sqrt{\sum_{i=1}^{N} \left(x_m^i - x_l^i \right)^2} \ . \qquad (1.3)$$

де x_m – об’єкт класу Ω_m; x_l – об’єкт класу Ω_l; x_m^i – i-та ознака розпізнавання класу Ω_m; x_l^i – i-та ознака розпізнавання класу Ω_l; N – кількість ознак розпізнавання.

Відстань (1.3) відповідає уявленню про геометричну близькість двох точок у багатовимірному просторі ознак. Ця міра широко використовується в методах кластер-аналізу, які ґрунтуються на детермінованих дистанційних критеріях близькості [3].

Основними підходами, що базуються на різних галуззях знань розв’язання задач розпізнавання образів, є:

1) алгебраїчний, основною перевагою якого є прості вирішальні правила. Основний недолік цього підходу полягає в невисокій достовірності розпізнавання, оскільки він не враховує неконтрольовані фактори, які впливають на процес розпізнавання;

2) геометричний, який характеризується наочністю та простотою

інтерпретації алгоритмів розпізнавання;

3) статистичний, який використовує статистичні методи аналізу даних;

4) біологічний, до якого відносять також штучні нейронні мережі. Алгоритми побудовані згідно цього підходу моделюють когнітивні процеси, що відбуваються у нервових клітинах мозку людини. Основний недолік біологічного підходу – це висока чутливість до кількості ознак розпізнавання;

5) мережевий, що базується на використанні семантичних мереж, мереж Петрі, дерев рішень, тощо. Перевагами цього підходу є простота моделі, можливість її ускладнення та розширення, а основний недолік – складність побудови вирішальних правил;

6) нечіткий, який дозволяє моделювати процеси розпізнавання образів, що перетинаються в просторі ознак розпізнавання. Основним недоліком цього підходу є те, що він не пристосований до оптимізації параметрів функціонування системи розпізнавання;

7) теоретико-ігровий підхід, вирішальні правила якого характеризуються високою складністю та невисокою достовірністю розпізнавання.

Незважаючи на те що наведені підходи відрізняються один від одного рівнем видом математичної формалізації, між ними не існує чіткої межі, а самі підходи часто доповнюють один одного. Оскільки всі основні підходи перетинаються з геометричним, то саме в рамках геометричного підходу формування загальної теорії прийняття рішень є найбільш виправданим [3].

В межах геометричного підходу можна виділити два основних принципи:

- максимально-дистанційний, за яким вирішальні правила будуються шляхом знаходження максимальної середньої відстані між класами;
- мінімально-дистанційний, за яким вирішальні правила будуються за умови мінімізації середньої відстані від образу до центру свого класу.

Виконання одного з цих принципів є необхідною умовою одержання максимальної достовірності розпізнавання.

У методології прийняття рішень при розпізнаванні образів виділяють три основні напрями [12]:

1) евристичні методи;

2) математичні методи;

3) лінгвістичні (синтаксичні) методи.

Евристичні методи ґрунтуються на досвіді та інтуїції розробника системи розпізнавання. Як правило, ці методи орієнтовані на розв’язання вузького класу задач розпізнавання і безпосередньо прив'язані до способу синтезу образів. Найбільш часто застосовуються при класифікації об’єктів з використанням способів порівняння з еталоном та визначенням загальних властивостей.

Математичні методи опираються на використання класичного математичного апарату: лінійного програмування, кореляційного аналізу, теорії прийняття рішень тощо. Застосовуються у випадках, коли ознаки представлені числовими параметрами, а їх зв'язки можуть бути описані у вигляді аналітичних залежностей. В певній мірі математичні методи використовуються у всіх підходах до розпізнавання [12].

Лінгвістичні (синтаксичні, структурні) методи застосовуються в тих випадках, коли образ є деякою структурою, що складається з так званих непохідних (первинних) елементів та ознак, що описують зв'язки між ними. У цих методах широко використовується апарат алгебри логіки та теорії формальних мов [12].

Сучасний підхід до розробки систем розпізнавання дозволяє комбінувати як різні підходи до опису образів, так і методи розпізнавання.

1.4. Комп'ютерний зір як галузь застосування теорії розпізнавання образів

Комп’ютерний зір – це технологія, що автоматично фіксує та обробляє зображення, як рухомих, так і нерухомих об’єктів. Дані у систему комп’ютерного зору можуть бути передані у різних форматах: відео, двовимірне чи тривимірне зображення [16].

Як наукова дисципліна комп'ютерний зір належить до теорії та технологій створення штучних систем, які отримують інформацію у вигляді зображень та можуть проводити виявляння та відстеження об'єктів.

Найпоширенішими областями застосування даної технології є медицина, робототехніка, системи відеоспостереження тощо [8].

Використання комп’ютерного зору в медицини характеризується аналізом інформації з графічних даних, отриманих за допомогою мікроскопії, рентгенографії, ангіографії, ультразвукових досліджень та томографії, для встановлення медичного діагнозу пацієнту. Таким чином можна виявити пухлину, атеросклероз чи інші злоякісні зміни.

Ще однією прикладною галуззю застосування комп'ютерного зору є промисловість. Тут його використовують, для організації виробничого процесу, наприклад визначення положення та орієнтації деталей роботом-маніпулятором, для контролю якості шляхом виявлення дефектів.

Комп'ютерний зір використовують також у військовій сфері, зокрема для виявлення ворожих солдат, транспортних засобів та для керування ракетами. Найбільш досконалі системи керування ракетами відправляють їх в певну область, замість конкретної цілі, а уточнення цілей базується на відеоданих, що надходять.

Однією з перспективних галузей застосування комп’ютерного зору є автономні транспортні засоби: підводні, наземні (роботи, машини), повітряні. Рівень автономності вимірюється від повністю автономних (безпілотних) до транспортних засобів, де системи штучного інтелекту допомагають водію чи пілоту приймати рішення в різноманітних ситуаціях. Повністю автономні транспортні засоби використовують комп'ютерний зір для навігації, тобто для отримання інформації про свого місце положення та визначення перешкод [16].

Завдання комп'ютерного зору включають методи отримання, обробки, аналізу цифрових зображень, для отримання числової чи символічної інформації для їх розуміння. Під розумінням зображення мається на увазі адекватна реакція (дія) системи, що випливає із проведеного аналізу.

Немає єдиної стандартної методики вирішення проблем комп'ютерного зору, натомість існує багато методів для розв’язання різноманітних, строго визначених задач комп'ютерного зору, де методи часто залежать від завдань і рідко коли можуть бути узагальнені для широкого застосування [16].

Основні підходи до вирішення проблеми комп'ютерного зору відносяться до галузі штучного інтелекту, а точніше до теорії розпізнавання образів та використання штучних нейронних мереж. Хоча є ряд питань, які можна дослідити з суто математичної точки зору. Багато методів базується на статистиці, методах оптимізації або геометрії.

Обробка зображень та їх аналіз в основному зосереджені на роботі з двовимірними зображеннями. Вони включають операції збільшення контрастності, операції з виділення країв, усунення шумів чи геометричні перетворення, такі як гомотетія чи поворот навколо точки.

Комп'ютерний зір часто залежить від припущень відносно того, що представлено на зображеннях. Класичне завдання даної галузі – це визначення того, чи містять відеодані деякий характерний об'єкт, особливість чи активність. Це завдання може бути вірогідно та легко вирішено людиною, але досі якісно не вирішено в комп'ютерному зорі в загальному випадку: випадкові об'єкти у випадкових ситуаціях [3].

З розпізнаванням зображень пов’язано ряд задач [16]:

- Розпізнавання об’єктів: один чи декілька попередньо заданих об'єктів або класів об'єктів можуть бути розпізнані, включно з їх положенням на рисунку чи у відеоряді.
- Ідентифікація: розпізнавання індивідуального екземпляра об'єкта, наприклад, ідентифікація конкретного людського обличчя або відбитків пальців.
- Виявлення: послідовність зображень (відеодані) перевіряються на наявність визначеної умови. Прикладом може бути виявлення хворих клітин чи тканин у медичних зображеннях. Виявлення, що ґрунтується на відносно простих і швидких обчисленнях, іноді використовується для знаходження

невеликих ділянок в зображенні, що аналізується, які потім досліджуються за допомогою заходів, які потребують більше ресурсів, для отримання правильної інтерпретації.

Наявні способи розв’язання цих задач придатні тільки для окремих об'єктів, таких як прості геометричні об'єкти (багатокутники), людські обличчя, друковані чи рукописні символи, автомобілі лише у визначених умовах, зазвичай це певне освітлення, тло і положення об'єкта відносно камери [15].

Крім цього є декілька завдань, що пов'язані з оцінкою руху. Для їх розв’язання обробляються відеодані для знаходження швидкості зміни кожної точки зображення чи 3D сцени. Да таких задач належать:

- одометрія – визначення руху камери (переміщення і обертання) в тривимірному просторі на основі низки знімків [15];
- стеження, тобто слідкування за переміщенням об'єкта, наприклад, деякої машини;
- оптичний потік – визначення руху кожного пікселя зображення відносно площини зображення, тобто видимий рух, що є підсумком руху як самої точки, так і камери [15].

Впровадження систем комп'ютерного зору дуже залежить від області їхнього застосування. Деякі системи є автономними і вирішують специфічні проблеми детектування та вимірювання, тоді як інші системи складають підсистеми більших систем, які, наприклад, можуть містити підсистеми контролю за механічними маніпуляторами, планування, інформаційні бази даних, інтерфейси людина-машина тощо [8].

Однак, незалежно від сфери використання, є ряд функцій, типових для багатьох систем комп'ютерного зору:

- Отримання зображень: від одного чи декількох фіксаторів графічної інформації, до яких відносяться різноманітні типи світлочутливих камер, радари, ультразвукові сканери тощо, отримуються, залежно від типу обладнання, цифрові 2D чи 3D зображення або відеоряд. Числові значення

пікселів виражають зазвичай інтенсивності світла в одній чи кількох спектральних смугах (кольорові чи відтінки сірого), але можуть бути пов'язані з різноманітними фізичними величинами, такими як: магнітний резонанс, глибина, поглинання чи відображення електромагнітних або звукових хвиль, тощо.

- Обробка зображень: до того як методи комп'ютерного зору будуть застосовані до відеоданих з метою виділення корисної інформації, необхідно їх попередньо обробити, щоб вони відповідали вимогам методу, який буде використовуватися. Для цього може виконуватися: повторна вибірка, щоб переконатися, що координатна система зображення є правильною; видалення шумів, щоб зменшити спотворення від джерела вхідних даних, підвищення контрастності.
- Сегментація та виокремлення деталей. Сегментація – це процес розділення цифрового зображення на декілька сегментів, однорідних за текстурою, з чіткими межами, сусідні сегменти відрізняються за певними критеріями. Для проведення сегментрації використовують: методи з використанням гістограм, методи перерізу графа, методи розростання областей, водорозділу, методи, що базуються на виділенні країв та інші.

Далі з оброблених графічних чи відеоданих виділяються окремі елементи зображення різного рівня складності. Простими деталями можуть бути: точки, геометричні лінії, кути, границі області. Складніші деталі можуть стосуватися руху чи структурованих об’єктів.

- Високорівнева обробка: на цьому етапі вхідні дані становлять невеликий набір даних, на базі якого приймається рішення про класифікацію виділеного об'єкта за різними категоріями, проводиться перевірка того, чи дані задовольняють визначеним умовам, здійснюється оцінка характерних параметрів, таких як розмір або положення об'єкта.

Для розпізнавання графічних образів можна застосувати метод перебору, порівнюючи вигляд об'єкта з різним масштабом, зсувами та при поворотах навколо точки (осі) під різними кутами, або дослідити контур об'єкта та на зв'язність, рівність кутів, тощо.

РОЗДІЛ 2
АЛГОРИТМ ІДЕНТИФІКАЦІЇ ТА ПОШУКУ СКЛАДНОГО ГРАФІЧНОГО ОБРАЗУ ТА ОСОБЛИВОСТІ ЙОГО ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

2.1. Опис алгоритму ідентифікації та пошуку складного графічного образу

Розглянемо задачу ідентифікації складного графічного образу та наступний пошук цього об’єкта у більшому графічному зображенні, яке в подальшому називатимемо базою. Під складністю образу розуміється об’єднання в одне ціле декількох простих елементів однакової структури.

В якості модельного прикладу розглядається зображення сукупності зірок, що зафіксоване при невизначених умовах зйомки (географічне положення, дата і час, кут повороту об’єктиву, масштаб). Базою пошуку є карта зоряного неба. Основним завданням є встановлення позиції входження образу у базу або фіксація його відсутності.

Для формалізації подання образу введемо наступні припущення та визначення:

- зображення зірок – це круги або еліпси невеликого ексцентриситету;
- зображення не містить «шумів»;
- зображення зірки не може бути фрагментоване (частина зірки не помістилася на рисунку);
- розмір зображення зірки визначається кількістю не чорних пікселів, в подальшому розмір ототожнюється з масою;
- яскравість зображення зірки визначається сумою кольорів усіх його пікселів;
- положення центру зображення зірки визначається координатами центру яскравості.

Виходячи із прийнятих домовленостей, сформулюємо алгоритм ідентифікації образу:

1) Перетворюємо чорно-біле растрове зображення у двовимірний масив світлових точок. Якщо розглядати формат файлу bmp, то чорні точки неба мають числове значення 0, а точки зірок – числа $0 < color < 256$.
2) Виділивши в масиві групу сусідніх пікселів, значення кольорів яких відмінні від 0, отримаємо список світлових точок, які відносяться до деякої локалізованої зірки.
3) Визначимо для знайденої зірки наступні атрибути:
   - масу m – кількість сусідніх світлових точок;
   - яскравість C – сума числових значень кольорів списку світлових точок

$$C = \sum_{k=1}^{m} color_k ; \tag{2.1}$$

   - координати (X, Y) центра зірки, який ототожнюються з центром яскравості і визначається аналогічно до центра мас за формулами (2.2)

$$X = \sum_{k=1}^{m} (color_k \cdot x_k) / C; \qquad Y = \sum_{k=1}^{m} (color_k \cdot y_k) / C \tag{2.2}$$

Тут $color_k$ – числове значення кольору та (x_k, y_k) – положення у двовимірному масиві k-тої світлової точки ($k = 1, 2, ..., m$), які відносяться до виділеної зірки.

Впорядкуємо отриманий набір даних за спаданням маси m, в результаті отримаємо множину об’єктів з набором атрибутів: S'_0, S'_1, S'_2, …, S'_{M-1} – елементи образу, де M – кількість зірок в образі.

При описі алгоритмів ідентифікації та пошуку складного графічного образу всі позначення зі штрихом будуть відноситися до образу.

Для зірки з максимальною масою S'_0 обчислюємо кути (в градусах) між напрямками на всі інші зірки в порядку розміщення у структурі даних: φ'_1, φ'_2, φ'_3, … φ'_{M-1}, де

$$\varphi'_j = \angle S'_j S'_0 S'_{j+1}, \quad j = 1, 2, ..., M-2;$$

$$\varphi'_{M-1} = \angle S'_{M-1} S'_0 S'_1; \qquad \varphi'_0 = 0^0. \tag{2.3}$$

Кути за годинниковою стрілкою вважатимемо додатними, а проти годинникової стрілки – від'ємними. Приклад позначення кутів елементів образу з $M = 4$ зірками показано на рисунку 2.1.

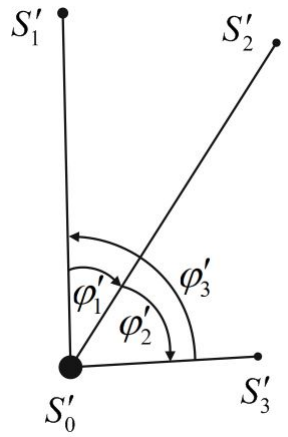


Рис. 2.1. Позначення кутів та відстаней між об'єктами образу

Зауважимо, що $\sum_{j=1}^{M-1} \varphi_j' = 0^0$ або $\sum_{j=1}^{M-1} \varphi_j' = 360^0$.

Окрім кутів знаходимо відстані $l(S_0', S_j')$ $(j = 1, 2, ..., M-1)$ від центральної зірки S_0' до решти.

Для визначення положення об'єктів при обчисленні кутів та відстаней використовуються координати центра, а отже відстань обчислюється за формулою відстані між двома точками на площині:

$$l(S_0', S_j') = \sqrt{\left(X_j' - X_0'\right)^2 + \left(Y_j' - Y_0'\right)^2}, \quad j = 1, 2, ..., M-1. \qquad (2.4)$$

Тут (X_0', Y_0'), (X_j', Y_j') – координати центрів зірок S_0' та S_j' відповідно.

Для знаходження кутів використаємо векторний добуток. Наприклад, для визначення кута $\varphi_1' = \angle S_1' S_0' S_2'$ між зірками S_1' і S_2' обчислимо векторний добуток між векторами $\overrightarrow{S_0'S_1'} = \left(X_1' - X_0'; Y_1' - Y_0'\right)$ та $\overrightarrow{S_0'S_2'} = \left(X_2' - X_0'; Y_2' - Y_0'\right)$ за формулою

$$\left[\overrightarrow{S_0'S_1'} \times \overrightarrow{S_0'S_2'}\right] = \left|\overrightarrow{S_0'S_1'}\right| \cdot \left|\overrightarrow{S_0'S_2'}\right| \sin\varphi_1'. \qquad (2.5)$$

З іншого боку векторний добуток можна визначити як визначник матриці

$$\left[\overrightarrow{S_0'S_1'}\times\overrightarrow{S_0'S_2'}\right]=\begin{vmatrix}\vec{i} & \vec{j} & \vec{k}\\ X_1'-X_0' & Y_1'-Y_0' & 0\\ X_2'-X_0' & Y_2'-Y_0' & 0\end{vmatrix}=$$

$$=\vec{k}\left[\left(X_1'-X_0'\right)\left(Y_2'-Y_0'\right)-\left(X_2'-X_0'\right)\left(Y_1'-Y_0'\right)\right], \qquad (2.6)$$

де $\vec{i}$, $\vec{j}$, $\vec{k}$ – одиничні базисні вектори у тривимірному просторі.

Враховуючи, що $\left|\overrightarrow{S_0'S_1'}\right|=l\left(S_0',S_1'\right)$ та $\left|\overrightarrow{S_0'S_2'}\right|=l\left(S_0',S_2'\right)$, візьмемо по модулю рівність (2.6) та, порівнявши відповідні величини з (2.5), отримаємо формулу для визначення $\sin\varphi_1'$

$$\sin\varphi_1'=\frac{\left(X_1'-X_0'\right)\left(Y_2'-Y_0'\right)-\left(X_2'-X_0'\right)\left(Y_1'-Y_0'\right)}{l\left(S_0',S_1'\right)l\left(S_0',S_2'\right)}. \qquad (2.7)$$

Використовуючи функцію $\arcsin x$, та виконавши перетворення з радіан у градуси знайдемо значення кута φ_1'

$$\varphi_1'=\arcsin(\sin\varphi_1'). \qquad (2.8)$$

Аналогічно визначаємо решту кутів між зірками образу φ_2', φ_3', … φ_{M-1}'.

Застосувавши описаний алгоритм до графічного зображення, що є базою пошуку, отримаємо впорядкована множина елементів $S_0^{(0)}$, $S_1^{(0)}$, $S_2^{(0)}$, …, $S_{N-1}^{(0)}$ – елементи бази на нульовій ітерації, де N – кількість зірок в базі. Верхній індекс позначає номер ітерації зовнішнього циклу алгоритму пошуку, оскільки список елементів бази змінюється після кожного його кроку.

Будемо враховувати, що образ міг піддаватися ряду геометричних перетворень, таких як:

- переміщення (паралельний перенос);
- зміна розмірів (масштабування);
- повороти навколо деякої точки на площині (обертання).

Вони найчастіше застосовуються до графічних зображень.

Алгоритм пошуку складного графічного образу базується на рівності відповідних кутів при перелічених геометричних перетвореннях, оскільки будуть виконуватися властивості подібних фігур.

Пошук кута образу φ'_j ($j = 1, 2, ..., M-1$) здійснюємо за умовою:

$$\varphi'_j = \varphi_i^{(t)} + \sum_{k=i+1}^{q} \varphi_k^{(t)}, \tag{2.9}$$

де $i+1 \le q \le N-1, \quad i = 1, 2, ..., N$, $t = 0, 1, 2, ..., N-M$.

У випадку, коли не знаходиться відповідне q, продовжуємо пошук спочатку

$$\varphi'_j = \varphi_i^{(t)} + \sum_{k=i+1}^{N-1} \varphi_k^{(t)} + \sum_{k=1}^{q} \varphi_k^{(t)}, \quad 1 \le q < i. \tag{2.10}$$

Загалом при відсутності співпадіння кутів обчислюються циклічно зі збільшенням кількості доданків суми кутів усіх елементів бази.

Якщо рівність (2.9) або (2.10) з деякою точністю ε виконується, то обчислення сум припиняється. Для знайдених в базі зірки S_i з кутом $\varphi_i^{(t)}$ та в образі зірки S'_j з кутом φ'_j проводимо додаткову перевірку на пропорційність відстаней та розмірів зірок:

$$K = \frac{l'(S'_0, S'_j)}{l(S_0, S_i^{(t)})} = \sqrt{\frac{m(S'_0)}{m(S_0)}} = \sqrt{\frac{m(S'_j)}{m(S_i^{(t)})}}. \tag{2.11}$$

Тут K – коефіцієнт подібності.

Якщо перевірка пройшла успішно, то фіксуємо знайдені об’єкти $S_i^{(t)}$ та S'_j і продовжуємо пошук для наступного елемента у образі. Пошук завершується успіхом навіть при частковому знаходженні зірок образу, тобто при $3 \le p \le M-1$, де p – кількість пар ототожнених елементів образу та бази.

У випадку, коли для зірки $S_0^{(t)}$ з максимальною масою на ітерації t алгоритму не знайдено жодного співпадіння кутів, вилучаємо її зі списку даних бази, кількість елементів множини бази N зменшуємо на одиницю, а номер кроку t збільшуємо на одиницю.

Якщо виявиться, що $N < M$, то пошук завершується невдачею. В іншому випадку проводиться перерахунок кутів та відстаней для нової головної зірки $S_0^{(t+1)} = S_1^{(t)}$ і пошук починається спочатку.

2.2. Огляд інтерфейсу програмного застосунку, що демонструє роботу алгоритму пошуку складного графічного образу

Програмну реалізацію алгоритму пошуку складного графічного образу здійснено на мові програмування C# з використанням IDE (Integrated development environment – інтегроване середовище розробки) Visual Studio 2017 Community. Створено проєкт застосунку типу Windows Presentation Foundation (WPF) Application.

Windows Presentation Foundation – це платформа розробки інтерфейсу користувача. Вона підтримує широкий набір компонентів для розробки програм, включаючи модель програми, ресурси, елементи управління, графіку, макет, прив'язку до даних та безпеку. WPF є частиною платформи .NET, яка використовує XAML (eXtensible Application Markup Language – розширена мова розмітки додатків) для визначення зовнішнього вигляду вікон, а також для наповнення їх елементами управління і графічними елементами [17].

XAML використовується перш за все для створення інтерфейсу користувача декларативним шляхом, схоже на HTML у веб-програмуванні. Кожен елемент у XAML-документі, який по замовчуванню включається з проєкт WPF, відображає екземпляр класу .NET, властивості якого можна встановити через атрибути. Код XAML допускає вкладення одного елемента в інший.

Створення інтерфейсу користувача розпочинається з розробки макету, на якому потрібно скомпонувати елементи управління, налаштовуючи їх розташування і розмір. Платформа WPF, ключовим елементом системи макету якої є відносне позиціонування, спрощує адаптацію змінних характеристик вікна до параметрів екрана [19].

Головне вікно застосунків, створеного при написанні кваліфікаційної роботи, зображено на рисунку 2.2. Для його побудови використано декілька елементів управління макетом: Grid (дочірні елементи керування впорядковуються по рядках та стовпцях як у таблиці); GridSplitter

(перерозподіляє простір між стовпцями або рядками елемента керування Grid); WrapPanel (дочірні елементи керування розташовуються в лінійному порядку і переносяться на наступний рядок, якщо не поміщаються в поточному).

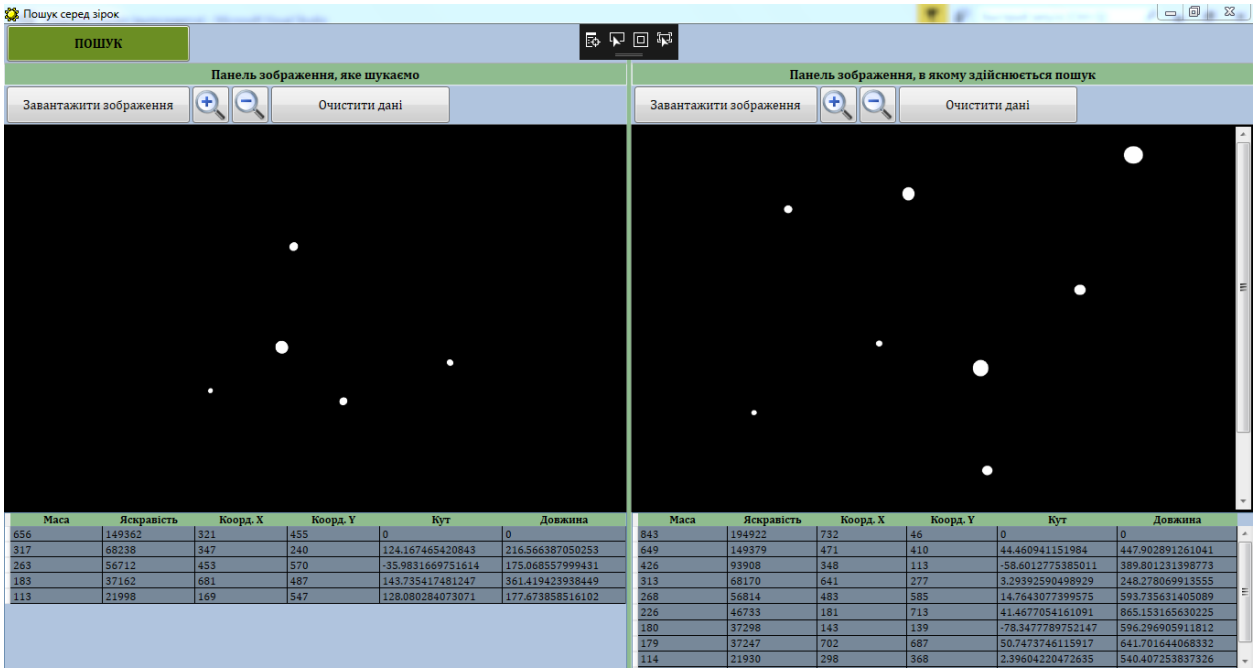


Рис. 2.2. Головне вікно програмного застосунку

Як видно з рисунку 2.2 головне вікно застосунку розділене на дві панелі, що відповідають образу та базі пошуку, кожна з яких поділені в свою чергу на функціональні зони: рядок інструментів (кнопок), область виводу зображення і таблиця для відображення числових характеристик об’єктів пошуку.

Для розбиття простору вікна на сегменти використано Grid та GridSplitter, оголошення яких наведено в лістингу 2.1.

Лістинг 2.1

Задання характеристик Grid та GridSplitter

```
<Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="0.06*" ></RowDefinition>
            <RowDefinition Height="0.035*" ></RowDefinition>
            <RowDefinition Height="0.06*" ></RowDefinition>
            <RowDefinition Height="0.6*"></RowDefinition>
            <RowDefinition Height="0.245*"></RowDefinition>
```

```
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="0.5*"></ColumnDefinition>
            <ColumnDefinition Width="5"></ColumnDefinition>
            <ColumnDefinition Width="0.5*"></ColumnDefinition>
        </Grid.ColumnDefinitions>
</Grid>
<GridSplitter Grid.Column="1" ShowsPreview="True"
        HorizontalAlignment="Stretch" VerticalAlignment="Stretch"
        AllowDrop="True" Background="DarkSeaGreen" Grid.RowSpan="5"/>
```

Як видно з лістингу 2.1 об’єкт Grid визначає створення таблиці з 5 рядками та 3 колонками із заданими пропорціями ширини Width та висоти Height, які підлаштовуються під фактичний розмір екрану. Центральна колонка із фіксованою шириною Width="5" призначена для розміщення розділювача GridSplitter між двома панелями, який завдяки властивості AllowDrop="True" дозволяє змінювати розмір колонок.

Розглянемо детальніше частину вікна, яка відповідає образу пошуку. На рисунку 2.2 бачимо набір чотирьох кнопок (елемент управління Button), розміщених послідовно у панелі інструментів WrapPanel, опис яких наведено у лістингу 2.2. Дані Button пов’язані з функціями відкриття файлу зображення образу, зміни масштабу зображення та очищення даних, що знаходяться у таблиці, для повторення процесу пошуку з іншим рисунком.

Лістинг 2.2

Характеристики WrapPanel блоку образу пошуку
та частини Button, на ній розміщених

```
<WrapPanel Grid.Column="0" Grid.Row="2" Orientation="Vertical"
           Height="Auto" Width="Auto">
  <Button Content="Завантажити зображення"
          x:Name="ButtonOpenImagePart" Click="ButtonOpenImagePart_Click"
          FontSize="14" FontFamily="Cambria" FontWeight="Normal"
          VerticalAlignment="Center" HorizontalAlignment="Left"
          Height="40"  Width="200"  Margin="3,1,3,1"/>
  <Button x:Name="ButtonZoomUpPart" Click="ButtonZoomUpPart_Click"
          VerticalAlignment="Center" HorizontalAlignment="Left"
          Height="40"  Width="40"  Margin="0,1,3,1">
```

```
                <Grid>
                    <Image Source="zoomUp.png"></Image>
                </Grid>
  </Button>
</WrapPanel>
```

При натисканні на кнопку «Завантажити зображення», з використанням об’єкту OpenFileDialog відкривається обраний графічний файл і рисунок відображається у відповідній зоні. Для організації даної частини вікна у XAML-документі використано елементи керування WPF, включені один в одний як показано у лістингу 2.3: Image – для виводу зображення, InkCanvas – для побудови геометричних фігур по рисунку, ScrollViewer – для використання ліній прокрутки у випадку, коли зображення не вміщується повністю у відведену зону екрану.

Лістинг 2.3

Опис елементів управління для виводу зображення образу

```
<ScrollViewer Grid.Column="0" Grid.Row="3" Margin="3"
VerticalScrollBarVisibility="Auto" HorizontalScrollBarVisibility="Auto">
   <InkCanvas x:Name="ImagePartCanvas" EditingMode="None"
            Height="0" Width="0" Background="Transparent"
            HorizontalAlignment="Center" VerticalAlignment="Center">
       <Image x:Name="ImagePart" Stretch="Uniform" Height="0" Width="0"
        HorizontalAlignment="Center" VerticalAlignment="Center"/>
   </InkCanvas>
</ScrollViewer>
```

Після завантаження зображення починає роботу алгоритм ідентифікації об’єктів складного графічного образу, результатом роботи якого є список елементів з наборами їх характеристик. Для їх виводу передбачена таблиця (рис. 2.2), створена з використанням елемента управління DataGrid, визначення якого наведено у лістингу 2.4.

Лістинг 2.4

Опис DataGrid для виведення числових характеристик об’єктів образу

```
<DataGrid x:Name="ImagePartDataGrid"
          AutoGenerateColumns="False" CanUserSortColumns="False"
```

```
          Grid.Column="0" Grid.Row="4" FontFamily="Cambria"
         RowBackground="LightSlateGray" Background="LightSteelBlue">
   <DataGrid.Columns>
        <DataGridTextColumn Header="Маса"  Width="0.15*"
                            Binding="{Binding Mass}"/>
        <DataGridTextColumn Header="Яскравість"  Width="0.15*"
                            Binding="{Binding Brightness}"/>
        <DataGridTextColumn Header="Коорд. X"  Width="0.15*"
                            Binding="{Binding CenterX}"/>
        <DataGridTextColumn Header="Коорд. Y"  Width="0.15*"
                            Binding="{Binding CenterY}"/>
        <DataGridTextColumn Header="Кут"  Width="0.19"
                            Binding="{Binding Angle}"/>
        <DataGridTextColumn Header="Довжина"  Width="0.19"
                            Binding="{Binding Length}"/>
   </DataGrid.Columns>
</DataGrid>
```

Аналогічні блоки користувацького інтерфейсу створені в другій колонці Grid для роботи з базою пошуку.

Вказівки визначення та форматування елементів користувацького інтерфейсу описані в файлі проєкту MainWindow.xaml, а обробники подій – у файлі MainWindow.xaml.cs.

2.3. Програмна реалізація алгоритму пошуку складного графічного образу

В проєкті програмного застосунку створено користувацьку бібліотеку функцій LibStars.cs, в якій оголошено множину класів для програмної реалізації алгоритму пошуку складного графічного образу.

Спочатку розглянемо процес ідентифікації елементів образу, який запускається під час виконання функції ButtonOpenImagePart_Click (лістинг 2.5), тобто після натисканні на кнопку «Завантажити зображення».

Лістинг 2.5

Функція, що виконується при натисканні на кнопку «Завантажити зображення»

```
private void ButtonOpenImagePart_Click(object sender, RoutedEventArgs e)
```

```
    {
        OpenFileDialog OpenDialogImagePart = new OpenFileDialog
        {
            DefaultExt = ".bmp",
            Filter = "Файли зображень|*.bmp|*.jpg|*.png" };
        if (OpenDialogImagePart.ShowDialog() == true)
        {
          BitmapImage BMImagePart = new BitmapImage();
          BMImagePart.BeginInit();
          BMImagePart.UriSource=new Uri(OpenDialogImagePart.FileName);
          BMImagePart.EndInit();
                ImagePart.Source = BMImagePart;
                ImagePart.Height = BMImagePart.PixelHeight;
                ImagePart.Width = BMImagePart.PixelWidth;
                ImagePartCanvas.Width = BMImagePart.PixelWidth;
                ImagePartCanvas.Height = BMImagePart.PixelHeight;
        ListStarsImagePart.CreateListOfStars(ref BMImagePart);
        PrintInformation(ref ListStarsImagePart, ref ImagePartDataGrid);
        DrawSystem(ref ListStarsImagePart);
      }
       else
MessageBox.Show( "Перевірте формат файлу","Помилка при відкритті файлу");
}
```

Як видно з лістингу 2.5, основний формат графічних растрових файлів, які можуть оброблятися у програмі, "*.bmp", але окрім нього допускається обрання "*.jpg" та "*.png".

Для доступу до вмісту обраного графічного файлу використаємо спеціалізований клас BitmapImage простору імен System.Windows.Media.Imaging, який оптимізований для завантаження растрових зображень за допомогою XAML [20]. Через створений екземпляр BMImagePart визначаємо реальні розміри завантаженого зображення та задаємо його як джерело для виводу зображення образу на головне вікно застосунку.

Безпосередній аналіз зображення для виокремлення об’єктів пошуку проводиться у функції CreateListOfStars. Згідно з припущеннями (п.2.1), оголошеними перед описом алгоритму ідентифікації, програмою розглядається чорно-біле зображення із числовим значенням кольорів в межах

від 0 до 255. Тому спочатку здійснюється конвертація завантаженого графічного файлу з довільною RGB чи CMYK кольоровою моделлю у формат Gray8, де на задання сірого кольору кожного пікселя виділяється 8 біт, з використанням класу FormatConvertedBitmap того ж простору імен як і BitmapImage (лістинг 2.6).

Лістинг 2.6

Перетворення зображення з BMImagePart у чорно-біле зображення

```
FormatConvertedBitmap GrayScaleBitmap = new FormatConvertedBitmap();
GrayScaleBitmap.BeginInit();
GrayScaleBitmap.Source = BMImagePart;
GrayScaleBitmap.DestinationFormat = PixelFormats.Gray8;
GrayScaleBitmap.EndInit();
```

Для подальшої роботи із вхідними даними, перетворимо отримане чорно-біле зображення у динамічний масив байтів GrayColorArr розмірності Width×Height, де Width – ширина зображення у пікселях, а Height – висота (лістинг 2.7). Для цього скористаємося методами класу BitmapEncoder (простір імен System.Windows.Media.Imaging), який кодує об'єкт BitmapFrame до потоку MemoryStream, звідки беруться дані для заповнення двовимірного масиву.

Лістинг 2.7

Перетворення чорно-білого зображення у двовимірний масив байтів

```
MemoryStream ImagetoStream = new MemoryStream();
BitmapEncoder encoder = new BmpBitmapEncoder();
encoder.Frames.Add(BitmapFrame.Create(GrayScaleBitmap));
encoder.Save(ImagetoStream);
Bitmap GrayBitMap = new Bitmap(ImagetoStream);
byte[,] GrayColorArr = new byte[GrayBitMap.Width, GrayBitMap.Height];
   for (int y = 0; y < GrayBitMap.Height; y++)
       for (int x = 0; x < GrayBitMap.Width; x++)
              GrayColorArr[x, y] = GrayBitMap.GetPixel(x, y).B;
```

Кожен піксель чорно-білого зображення має три характеристики: колір і координати, аналізуючи які потрібно знайти сукупність сусідніх точок, колір

яких відмінний від нуля. Таким чином визначається положення та розмір зірки на зображенні. Для збереження корисної інформації після обробки двовимірного масиву оголошено структуру LigthPoint та клас Star в програмному модулі LibStars.cs, поля яких наведено в лістингу 2.8. Як видно із оголошення класу Star в ньому міститься список ListLightPoints світлових точок, які формують зображення зірки. Крім того полями класу є основні атрибути, які повність характеризують об’єкти пошуку, а саме:

- Mass – маса (розмір) зірки;
- Brightness – її яскравість;
- CenterX, CenterY – координати центра яскравості;
- Length – відстань до головної зірки;
- Angle – кут між сусідніми елементами списку.

Лістинг 2.8

Опис структури LigthPoint і полей класу Star

```
public struct LigthPoint
   {
       public int  CoordX     { get; set; }
       public int  CoordY     { get; set; }
       public byte ColorGray  { get; set; }
   }

public class Star
    {
        public int  Mass { get; set; }
        public int  Brightness { get; set; }
        public int  CenterX { get; set; }
        public int  CenterY { get; set; }
        public double Length { get; set; }
        public double Angle { get; set;
        public List<LigthPoint> ListLightPoints;
    }
```

Окрім розглянутих полів клас Star містить метод CalculateStarOwnCharacteristics( ) (лістинг 2.9) для обчислення характеристик Mass, Brightness, CenterX та CenterY, визначення яких базується на даних

списку світлових точок ListLightPoints.

Лістинг 2.9

Метод CalculateStarOwnCharacteristics( ) класу Star

```
public void CalculateStarOwnCharacteristics()
{
    foreach (LigthPoint LP in ListLightPoints)
    {
        CenterX += LP.CoordX * LP.ColorGray;
        CenterY += LP.CoordY * LP.ColorGray;
        Brightness += LP.ColorGray;
    }
    CenterX = CenterX / Brightness;
    CenterY = CenterY / Brightness;
    Mass = ListLightPoints.Count;
}
```

Знайдені об’єкти класу Star зберігаються у класі ListOfStars, основним полем якого є список зірок:

```
public List <Star> ListStars.
```

Окрім того клас ListOfStars має два конструктори. Один з них без параметрів, який динамічно створює екземпляр списку, а другий – конструктор копіювання.

Основні функції для розв’язання задачі ідентифікації складного графічного образу CreateListOfStars, LocalizationStar, CalculateListStarsCharacteristics теж належать класу ListOfStars.

Розглянемо частину функції CreateListOfStars (лістинг 2.10), яка виконується після отримання двовимірного масиву GrayColorArr. Для знайденого ненульового елементу масиву створюється новий екземпляр класу Star NewStar і виклинається функція LocalizationStar. Ця рекурсивна функція починаючи з координат (x, y) "білого" пікселя перевіряє сусідні елементи масиву, виділяючи множину світлових точок та заповнюючи ними список ListLightPoints. Сформувавши сукупність світлових точок, які відносяться до об’єкту NewStar, обчислюємо власні характеристики зірки, використовуючи функцію CalculateStarOwnCharacteristics( ) та добавляємо новий елемент у

список зірок ListStars.

Після проходження усього масиву, отримаємо заповнений список об’єктів пошуку образу. Згідно з алгоритмом даний список має бути посортований по спаданню мас зірок. Для сортування використовуємо стандартний метод Sort класу List (лістинг 2.10). Для класу Star можна реалізувати інтерфейс IComparable, а отже його екземпляри, можна сортувати за допомогою алгоритмів сортування C#. Інтерфейс IComparable визначає метод CompareTo, який використовується для порівняння об’єктів [21].

Лістинг 2.10

Частина методу CreateListOfStars ( ) класу ListOfStars

```
public void CreateListOfStars(ref BitmapImage BMImage)
{
//частина функції, яка виконується після отримання двовимірного масиву
  for (int y = 0; y < GrayBitMap.Height; y++)
  {
   for (int x = 0; x < GrayBitMap.Width; x++)
   {
     if (GrayColorArr[x, y] != 0)
     {
        Star NewStar = new Star();
        LocalizationStar(x, y, ref GrayColorArr,
        ref NewStar.ListLightPoints);
        NewStar.CalculateStarOwnCharacteristics();
        ListStars.Add(NewStar);
     }
   }
  }
        ListStars.Sort((Star1, Star2) =>
                       Star2.Mass.CompareTo(Star1.Mass));
        CalculateListStarsCharacteristics();
}
```

Впорядкований по спаданню список зірок дозволяє нам визначити головну зірку образу S_0' та знайти відносно неї характеристики зірок, які залежать від положення інших об’єктів, таких як відстань Length до головної

зірки та кут Angle між двома сусідніми елементами списку. Дані величини обчислюються за формулами (2.3), (2.4) у методі CalculateListStarsCharacteristics класу ListOfStars (лістинг 2.11). Дана функція завершує роботу алгоритму ідентифікації складного графічного образу.

Лістинг 2.11

Метод CalculateListStarsCharacteristics ( ) класу ListOfStars

```
public void CalculateListStarsCharacteristics()
  {
      int ListCount = ListStars.Count();
      int AX, AY, BX, BY, CX, CY;
           AX = ListStars[0].CenterX;
           AY = ListStars[0].CenterY;
           ListStars[0].Angle = 0;
           ListStars[0].Length = 0;
      for (int i=1; i<ListCount; i++)
      {
        BX = ListStars[i].CenterX;
        BY = ListStars[i].CenterY;
        if (i == ListCount - 1)
           {
               CX = ListStars[1].CenterX;
               CY = ListStars[1].CenterY;
           }
        else
           {
               CX = ListStars[i + 1].CenterX;
               CY = ListStars[i + 1].CenterY;
           }
  double AB = Math.Sqrt(Math.Pow(AX - BX, 2) + Math.Pow(AY - BY, 2));
  double AC = Math.Sqrt(Math.Pow(AX - CX, 2) + Math.Pow(AY - CY, 2));
  double cosAngle = ((BX-AX)*(CX-AX)+(CY-AY)*(BY-AY))/AB/AC;
  double sinAngle = ((BX-AX)*(CY-AY)-(CX-AX)*(BY-AY))/AB/AC;
        ListStars[i].Angle = Math.Acos(cosAngle);
        ListStars[i].Angle = ListStars[i].Angle * 180 / Math.PI;
        if (sinAngle < 0) ListStars[i].Angle = - ListStars[i].Angle;
        ListStars[i].Length = AB;
      }
  }
```

Сформувавши список об’єктів образу, викликаємо у методі CreateListOfStars ( ) функцію DrawSystem для візуалізації ідентифікованих зірок. Її текст наведено в лістингу 2.12.

Для зображення системи зірок використаємо класи Ellipse, Line, TextBlock простору імен System.Windows.Shapes [22].

Екземпляри Ellipse позначають положення зірок, для цього їх центр визначається координатами центру зірки, а розміри Width та Height обрані однаковими, щоб отримати круглу точку.

Екземпляри Line будують відрізок між центрами двох зірок та демонструють відстань від головної зірки до решти.

Використовуючи екземпляри TextBlock виводимо порядковий номер зірки у впорядкованому списку, що дозволяє нам оцінити кути між сусідніми об’єктами.

Вивід створених графічних елементів із заданими характеристиками здійснюється з допомогою класа Canvas, екземпляр якого має прозоре тло і такі ж параметри як Image панелі образу. Таким чином Canvas розміщується над рисунком образу і містить діаграму кореневого дерева, утворену з вершин (Ellipse) і ребер (Line), де коренем являється головна вершина.

Лістинг 2.12

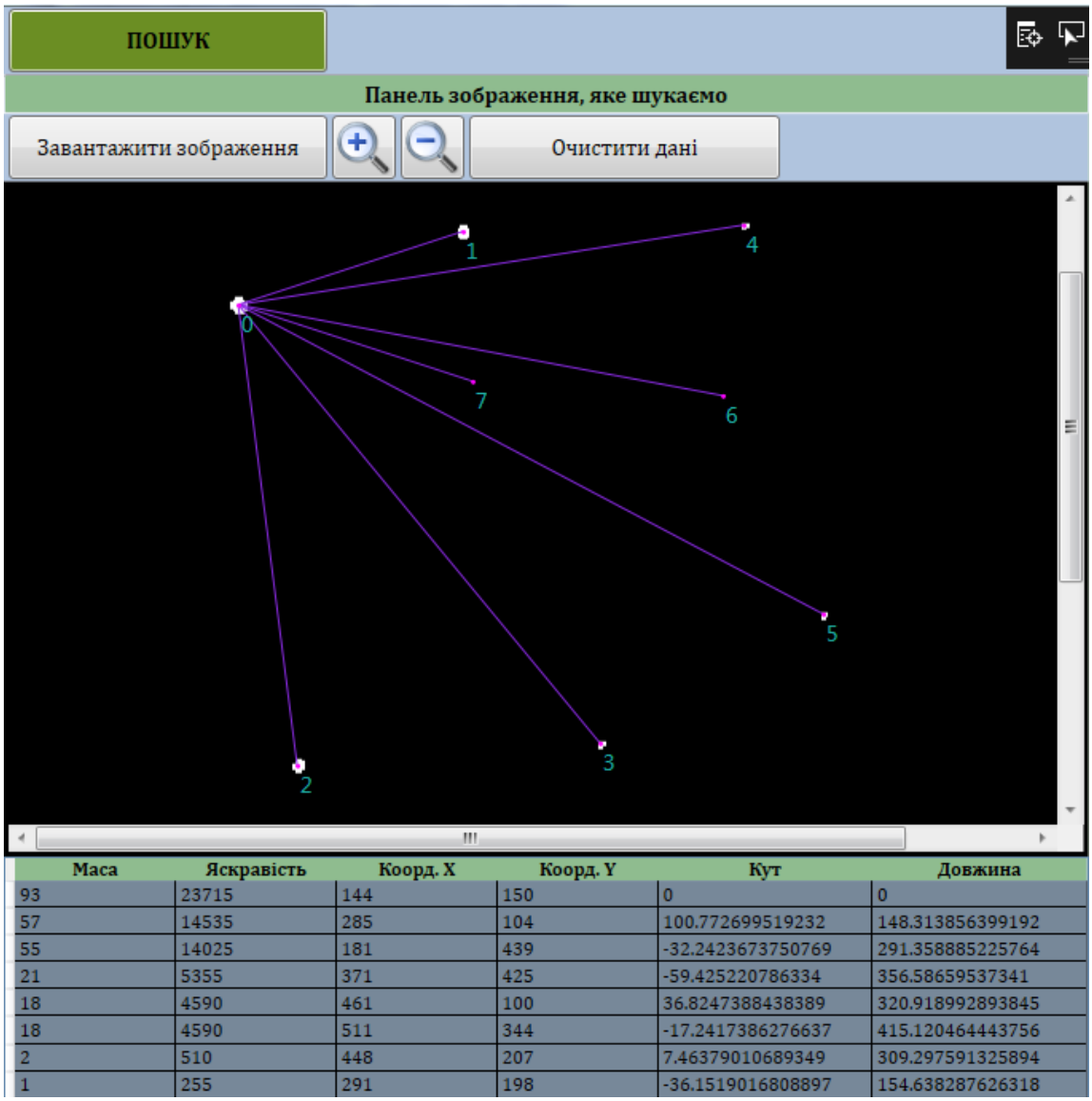
Функція DrawSystem файлу MainWindow.xaml.cs

```
public void DrawSystem(ref ListOfStars LS)
    {
       ImagePartCanvas.Height = ImagePart.Height;
       ImagePartCanvas.Width = ImagePart.Width;
       System.Windows.Media.Color TextColor =
       System.Windows.Media.Colors.Blue;
       Star MainStar = LS.ListStars[0];
       //Круг для зображення головної зірки
       Ellipse MainStarPoint = new Ellipse
         {
            Stroke = System.Windows.Media.Brushes.Magenta,
            Fill = System.Windows.Media.Brushes.Magenta,
            Stretch = Stretch.Fill,
            Width = 3,  Height = 3,
```

```
        Margin = new Thickness
        (MainStar.CenterX - 1, MainStar.CenterY - 1, 0, 0)
    };
  //Підпис для головної зірки
  TextBlock NumberOfMainStar = new TextBlock
    {
        Text = "0",   FontSize = 14,
        Foreground = new SolidColorBrush(TextColor),
        Margin=new Thickness(MainStar.CenterX+2,MainStar.CenterY+2,0,0)
    };
   //Побудова ліній від головної зірки до решти
   for (int i = 0; i < LS.ListStars.Count; i++)
     {
        Star S = LS.ListStars[i];
   //створюємо екземпляр класу Line і задаємо його атрибути
        Line LineBetweenStars = new Line
          {
             Stroke = System.Windows.Media.Brushes.BlueViolet,
             StrokeThickness = 1,
             X1 = MainStar.CenterX,
             Y1 = MainStar.CenterY,
             X2 = S.CenterX,
             Y2 = S.CenterY
          };
   //створюємо екземпляр класу Ellipse і задаємо його атрибути
        Ellipse StarPoint = new Ellipse
          {
             Stroke = System.Windows.Media.Brushes.Magenta,
             Fill = System.Windows.Media.Brushes.Magenta,
             Stretch = Stretch.Fill,
             Width = 3,
             Height = 3,
             Margin = new Thickness(S.CenterX-1, S.CenterY-1,0,0)
          };
   //створюємо TextBlock для виводу номера зірки у списку
        TextBlock NumberOfStar = new TextBlock
           {
             Text = i.ToString(),
             FontSize = 14,
             Foreground = new SolidColorBrush(TextColor),
             Margin = new Thickness(S.CenterX+2, S.CenterY+2, 0, 0)
```

```
                };
                //Додаємо створені графічні об’єкти на Canvas
                ImagePartCanvas.Children.Add(NumberOfStar);
                ImagePartCanvas.Children.Add(LineBetweenStars);
                ImagePartCanvas.Children.Add(StarPoint);
            }
        ImagePartCanvas.Children.Add(MainStarPoint);
        ImagePartCanvas.Children.Add(NumberOfMainStar);
    }
```

Результат виконання функції DrawSystem продемонстровано на рисунку 2.3.



Маса	Яскравість	Коорд. X	Коорд. Y	Кут	Довжина
93	23715	144	150	0	0
57	14535	285	104	100.772699519232	148.313856399192
55	14025	181	439	-32.2423673750769	291.358885225764
21	5355	371	425	-59.425220786334	356.58659537341
18	4590	461	100	36.8247388438389	320.918992893845
18	4590	511	344	-17.2417386276637	415.120464443756
2	510	448	207	7.46379010689349	309.297591325894
1	255	291	198	-36.1519016808897	154.638287626318

Рис. 2.3. Вигляд панелі образу пошуку з візуалізацією ідентифікованих об’єктів

Засовувавши перелічені вище функції до зображення, яке є базою пошуку, отримаємо ще один список зірок з відповідними атрибутами впорядкований по спаданню мас. Враховуючи, що розміри бази значно більші за розміри образу, доцільно застосовувати процедури ідентифікації об'єктів бази та пошуку до окремих частин зображення, поділивши його на квадрати із стороною $2d$, де d – діагональ прямокутника образу, із зсувом розглядуваної області на крок d.

Після завершення процесу ідентифікації об'єктів обох зображень, розпочинає роботу алгоритм пошуку складного графічного образу, який запускається натисканням на кнопку «ПОШУК» (рис. 2.2, 2.3). Відповідна функція наведена в лістингу 2.13. В ній перевіряються вхідні дані: якщо відсутня інформація про елементи образу або бази, а також якщо в образі зірок більше ніж в базі, то виводиться повідомлення про помилку і процедура пошуку не розпочинається.

В випадку коректно сформованих списків зірок викликається функція Search() класу SearchStars для бібліотеки LibStars.cs. Оскільки для пошуку аналізуються зображення, які фіксуються за невизначених умов, а отже можуть мати різний масштаб, роздільну здатність, кут та час зйомки, це має вплив на визначення числових характеристик зірок. Звідси виникає потреба здійснювати пошук декількома спробами із різною точністю ε. Спочатку приймаємо $\varepsilon = 0.0001$. Якщо були знайдені всі об'єкти образу, то пошук зупиняється, інакше збільшуємо ε і робимо ще одну спробу. Граничне значення відхилення кутів образу від сум кутів бази визначено в $\varepsilon = 1$ градус.

Лістинг 2.13

Обробник натискання на кнопку «Пошук»

```
private void ButtonSearch_Click(object sender, RoutedEventArgs e)
    {
        int N = ListStarsImageAll.ListStars.Count;
        int M = ListStarsImagePart.ListStars.Count;
//перевірка вхідних даних на коректність
```

```
if (N == 0)
MessageBox.Show("Відсутні дані для пошуку \n
             (зображення, в якому здійснюється пошук)."," Помилка");
else
if (M == 0)
MessageBox.Show("Відсутні дані для пошуку \n
             (зображення, яке шукаємо)."," Помилка");
else
if (M > N)
MessageBox.Show("В образі більше елементів ніж у базі."," Помилка");
  else { //початок серії процедур пошуку
      List<SearchStars> ListOfSearches = new List<SearchStars>();
      double eps = 0.0001;
      do // пошук елементів образу в базі з різною точністю
      {
         SearchStars NewSearch =
         new SearchStars(ListStarsImagePart, ListStarsImageAll);
            NewSearch.Epsilon = eps;
            NewSearch.Search();
            ListOfSearches.Add(NewSearch);
            eps *= 10;
         //Зупинка пошуку якщо знайдено всі елементи образу
         if (Search.ResultImagePart.ListStars.Count ==
             ListStarsImagePart.ListStars.Count) break;
      } while (eps <= 1);
    string TextResults = "";
    //вивід результатів пошуку для різної точності
    foreach (SearchStars S in ListOfSearches)
    {
      TextResults+="Знайдено зірок:"+S.ResultImagePart.ListStars.Count+
                   "\t з точністю: " + S.Epsilon + ".\n";
    }
     MessageBox.Show(TextResults, "Результати пошуку");
    //співставлення знайдених елементів бази та образу
     SelectInformation(ref ListOfSearches.Last().ResultImagePart,
                       ref ImagePartDataGrid);
  }
}
```

Клас SearchStars містить два поля для збереження вхідних даних – списків характеристик зірок образу і бази, а також відповідні поля для

фіксування знайдених об’єктів (лістинг 2.14).

Лістинг 2.14

Оголошення полів класу SearchStars

```
public class SearchStars
  {
      public  double Epsilon { get; set; }
      public ListOfStars  StarsAll;
      public ListOfStars  ResultImageAll;
      public ListOfStars  StarsPart;
      public ListOfStars  ResultImagePart;
  }
```

Окрім полів, визначених у лістингу 2.14, клас SearchStars містить метод Search( ) (лістинг 2.15), який безпосередньо реалізовує процедуру пошуку.

Розглянувши лістинг 2.15, бачимо що зовнішній цикл while, який залежить від булевої змінної SearchComplete, виконується доки не знайдено жодного елемента або поки не встановлена відсутність об’єктів образу в базі. В найкращому випадку, коли співпадіння зірок встановлено на першій ітерації, він виконується один раз, в найгіршому випадку, коли пошук завершується невдачею, – $N - M + 1$ раз.

Внутрішній цикл for по i перебирає всі зірки списку StarsAll, що містить елементи бази по спаданню їх мас. Для поточної зірки S_i визначаються циклічні накопичувальні суми кутів Suma_Angles, які порівнюються в циклі for по j з кутами зірок S'_j образу. Якщо різниця між цими величинами не перевищує ε, то здійснюється перевірка додаткової умови (2.11) викликом функції CheckStars. При виконанні обох умов, фіксуємо факт співпадіння двох об’єктів, додавши знайдені елементи у результуючі списки.

Лістинг 2.15

Метод Search ( ) класу SearchStars

```
public void Search()
  {
    bool SearchComplete = false;
    int iteration = 0;
```

```
    int k;
    while (!SearchComplete)
      {
        int M = StarsPart.ListStars.Count;
        int N = StarsAll.ListStars.Count;
        for (int i = 1; i < N; i++)
          {
            double Suma_Angles = StarsAll.ListStars[i].Angle;
            k = i;
            for (int q = 1; q < N; q++)
            {
               for (int j = 1; j < M; j++)
               if (Math.Abs(StarsPart.ListStars[j].Angle –
                            Suma_Angles) < Epsilon)
               if(CheckStars(StarsPart.ListStars[j],
                             StarsAll.ListStars[i]))
               {
               ResultImageAll.ListStars.Add(StarsAll.ListStars[i]);
               ResultImagePart.ListStars.Add(StarsPart.ListStars[j]);
               }
               k++;
               if (k == N) k = 1;
               Suma_Angles += StarsAll.ListStars[k].Angle;
            }
          }
//Перевірка чи є знайдені елементи в базі пошуку
//Якщо немає, то видаляємо головну зірку та проводимо перерахунок
//кутів і довжин, використовуючи в якості головної наступну по величині
       if (ResultImageAll.ListStars.Count==0 &&
                   StarsAll.ListStars.Count >= M)
           {
                StarsAll.ListStars.RemoveAt(0);
                StarsAll.CalculateListStarsCharacteristics();
            }
//Перевірка чи є знайдені елементи в базі пошуку
//В цьому випадку пошук завершується успіхом
       if (ResultImageAll.ListStars.Count>0)
          {
             SearchComplete = true;
             ResultImageAll.ListStars.Insert(0,StarsAll.ListStars[0]);
             ResultImagePart.ListStars.Insert(0,StarsPart.ListStars[0]);
```

```
            }
//Перевірка чи образ містить більше елементів ніж база
//В цьому випадку пошук завершується невдачею
        if(StarsAll.ListStars.Count <M)
                    SearchComplete = true;
            }
  }
```

Після виконання внутрішніх циклів аналізуємо отримані результуючі списки знайдених елементів. Якщо вони порожні, то замінюємо головну зірку бази, викликавши вбудовану функцію RemoveAt(0) для роботи зі списками типу List, яка видаляє елемент з індексом 0, та функцію CalculateListStarsCharacteristics( ), яка обчислить значення кутів та довжин зірок бази відносно нової головної зірки. Після чого повторюємо процедуру пошуку, використовуючи оновлені дані бази.

Якщо списки знайдених елементів не порожні, то вважаємо що пошук завершився успіхом. Маючи частковий успіх, тобто знайдено не всі елементи образу, збільшимо ε і зробимо ще одну спробу (лістинг 2.13).

У випадку коли $N - M + 1$ ітерацій алгоритму були неуспішними, тоді база містить менше об’єктів пошуку ніж образ, припиняємо пошук і вважаємо, що встановлено відсутність образу в базі.

Класу SearchStars містить окрім головної функції Search( ) допоміжний метод CheckStars (лістинг 2.16) для перевірки умови (2.11), яка визначає пропорційність мас та довжин зірок образу та бази.

У функції CheckStars обчислюються співвідношення коренів мас CorrelationMass та співвідношення довжин CorrelationLength двох екземплярів S1 та S2 класу Star (при виклику функції вказується один об’єкт з образу, а другий – із бази). При співпадінні зірок, ці числа мають незначно відрізнятися, а їх дріб повинен бути близький до 1. При перевірці умови у функції CheckStars допускається відхилення відношення CorrelationMass/CorrelationLength від одиниці на величину не більшу 0.1.

Лістинг 2.16

Метод CheckStars класу SearchStars

```
public bool CheckStars(Star S1, Star S2)
    {
      double CorrelationMass =
      Math.Sqrt(Convert.ToDouble(S1.Mass) / Convert.ToDouble(S2.Mass));
      double CorrelationLength = S1.Length / S2. Length
      if (Math.Abs(1-CorrelationMass/CorrelationLength)<0.1)
                return true;
            else
                return false;
    }
```

Аналізуючи кількість операцій для визначення складності алгоритму, зауважимо, що для кожної з $N-1$ зірки образу, за виключенням головної, обчислюється $N-1$ сума кутів, що порівнюються з кутом кожної з $M-1$ зірки образу, без врахування головної. Таким чином отримаємо кількість перевірок, виконаних на одній ітерації зовнішнього циклу, порядку $O(N^2 \cdot M)$.

Враховуючи, що зовнішній цикл while може виконуватися від 1 до $N-M+1$ разів, загальна кількість операцій порівняння не буде перевищувати $O(N^2 \cdot M \cdot (N-M+1))$.

РОЗДІЛ 3
МОДЕЛЬНІ ЕКСПЕРИМЕНТИ ДЛЯ АНАЛІЗУ ЕФЕКТИВНОСТІ АЛГОРИТМУ ПОШУКУ СКЛАДНОГО ГРАФІЧНОГО ОБРАЗУ

Для встановлення ефективності алгоритму пошуку складного графічного образу в кваліфікаційній роботі розглянемо низку прикладів з числовими розрахунками, на яких продемонструємо покрокову роботу алгоритму, встановимо його точність та складність.

В якості першого експерименту покажемо виконання алгоритму для тестового зображення на рисунку 3.1.

В даному розділі розміщуватимемо інвертовані графічні растрові файли, де зірки зображені чорним (сірим) кольором на білому фоні, оскільки в такому вигляді зручніше вводити позначення для основних елементів.

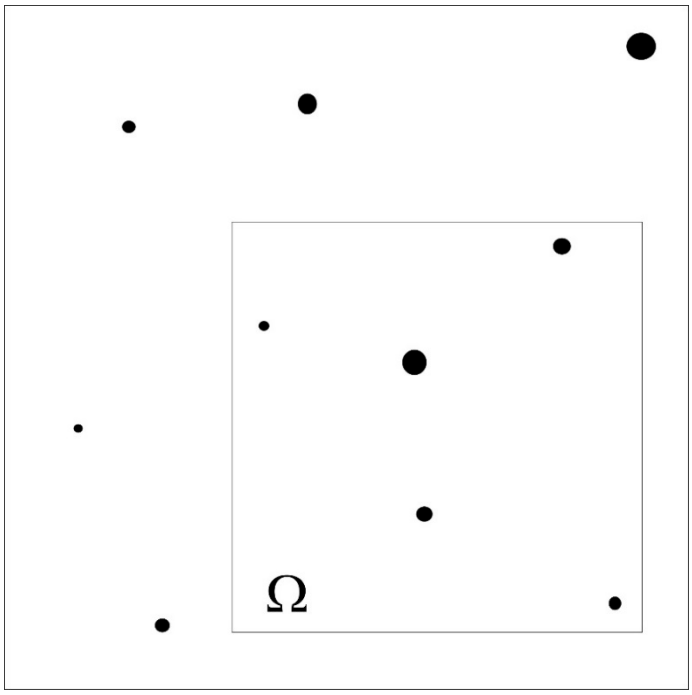


Рис. 3.1. Інвертоване зображення бази та образу пошуку

Як видно на рисунку 3.1. база містить $N = 10$ зірок. В якості образу використаємо область Ω без додаткової деформації зображення, яка містить $M = 5$ зірок.

Після процедури ідентифікації, описаної в пункті 2.1 та застосованої до

бази отримаємо список елементів $S_i^{(0)}$ $(i = 0, 1, ... N-1)$ впорядкованих по зростанню маси для аналізу на нульовій ітерації алгоритму пошуку. Кожен об'єкт бази має набір числових характеристик, які наведені в таблиці 3.1. Кути $\varphi_i^{(0)} = \angle S_i^{(0)} S_0^{(0)} S_{i+1}^{(0)}$ $(\varphi_{N-1}^{(0)} = \angle S_{N-1}^{(0)} S_0^{(0)} S_1^{(0)})$ та відстані $l(S_0^{(0)}; S_i^{(0)})$ від головної зірки $S_0^{(0)}$ до решти $S_i^{(0)}$ $(i = 0, 1, ... N-1)$ вказані на рисунку 3.2. Кути $\varphi_i^{(0)}$, обчислені при переміщенні зірки від $S_i^{(0)}$ до $S_{i+1}^{(0)}$ за годинниковою стрілкою (напрям вказаний стрілкою на рисунку), мають додатне значення, а проти годинникової стрілки – від'ємне. Сума кутів всіх елементів бази $\sum_{i=1}^{N-1} \varphi_i^{(0)} = 0^0$ або $\sum_{i=1}^{N-1} \varphi_i^{(0)} = 360^0$.

Таблиця 3.1

Характеристики зірок бази на нульовій ітерації алгоритму

Зірка $S_i^{(0)}$	Маса	Яскравість	Координата центра X	Координата центра Y	Кут $\varphi_i^{(0)}$ в градусах	Відстань $l(S_0^{(0)}; S_i^{(0)})$
$S_0^{(0)}$	843	194922	732	46	0	0
$S_1^{(0)}$	649	149379	471	410	44,46094	447,90289
$S_2^{(0)}$	426	93908	348	113	-58,60127	389,80123
$S_3^{(0)}$	313	68170	641	277	3,29392	248,27807
$S_4^{(0)}$	268	56814	483	585	14,764307	593,73563
$S_5^{(0)}$	226	46733	181	713	41,46771	865,15316
$S_6^{(0)}$	180	37298	143	139	-78,34778	596,29691
$S_7^{(0)}$	179	37247	702	687	50,74737	641,70164
$S_8^{(0)}$	114	21930	298	368	2,39604	540,40725
$S_9^{(0)}$	87	15895	84	486	-20,18124	783,26496

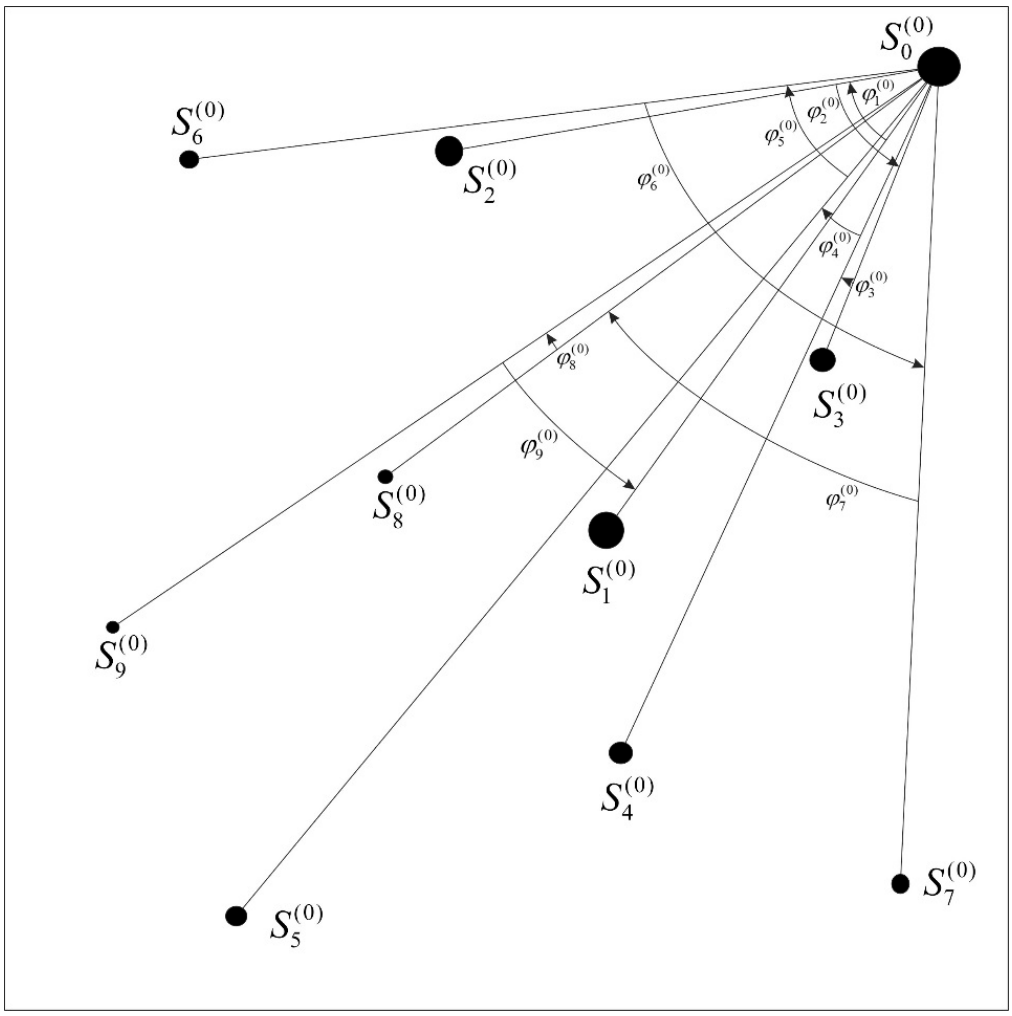


Рис. 3.2. Інвертоване зображення зірок бази на нульовій ітерації із зазначеними кутами та відстанями

Аналогічні обчислення проведемо для образу й отримаємо список елементів S'_j ($j = 0, 1, ...M - 1$), числові характеристики яких вказано у таблиці 3.2 і позначено на рисунку 3.3.

Таблиця 3.2

Характеристики зірок образу без геометричних перетворень

Зірка S'_j	Маса	Яскравість	Координата центра X	Координата центра Y	Кут φ'_j в градусах	Відстань $l(S'_0;S'_j)$
S'_0	649	149379	471	410	0	0
S'_1	313	68170	641	277	124,11524	215,84485
S'_2	268	56814	483	585	-35,90319	175,41094
S'_3	179	24973	702	687	143,47186	360,67991
S'_4	114	21930	298	368	128,31609	178,02527

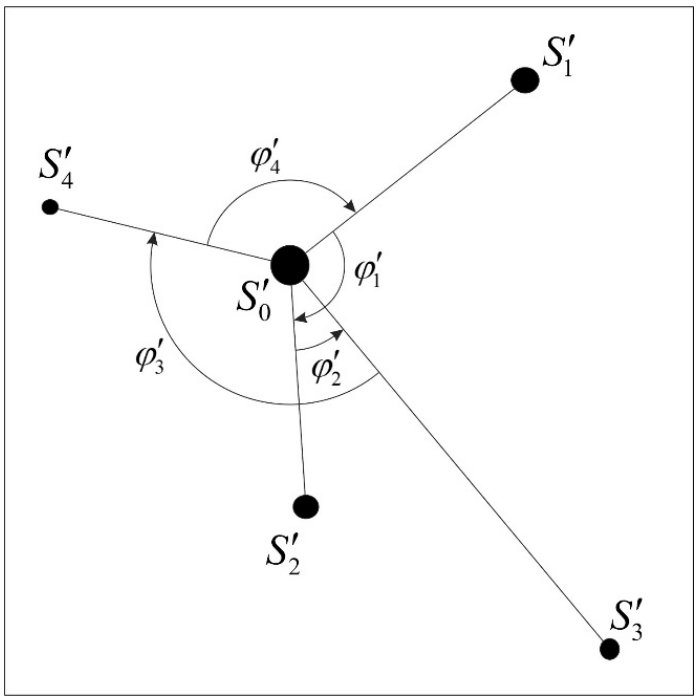


Рис. 3.3. Інвертоване зображення зірок образу
із зазначеними кутами та відстанями

Згідно з алгоритмом, запропонованим в кваліфікаційній роботі для інтелектуального пошуку елементів складного графічного образу, потрібно знайти співпадіння кутів образу з кутами (сумами кутів) бази.

Суми кутів для кожного елемента бази за виключенням головної зірки обчислюються починаючи з кута розглядуваного об'єкту $\varphi_i^{(0)}$ і проводяться циклічно по списку зі збільшенням кількості доданків q $(q = 1, 2, \ldots N-1)$ з наступної суми $\sum \varphi_i^{(0)} = \varphi_i^{(0)} + \sum_{k=i+1}^{N-1} \varphi_k^{(0)} + \sum_{k=1}^{i-1} \varphi_k^{(0)}$ $(i = 1, 2, \ldots N-1)$. Тобто, наприклад, для зірки $S_1^{(0)}$ знаходиться $N-1$ величин: $\varphi_1^{(0)}$, $\varphi_1^{(0)} + \varphi_2^{(0)}$, …, $\varphi_1^{(0)} + \varphi_2^{(0)} + \varphi_2^{(0)} + \ldots + \varphi_9^{(0)}$.

Числові результати отримані на нульовій ітерації, наведено в таблиці 3.3. Звернемо увагу, що сума кутів всіх елементів бази у випадку головної зірки $S_0^{(0)}$ дорівнює нулю. Циклічно обчислені суми з $N-1$ кутів, числові значення яких наведено в градусах, в кожній колонці також закінчуються нулем.

Таблиця 3.3

Суми кутів елементів бази, отримані на нульовій ітерації алгоритму

$\varphi_i^{(0)}$	Значення кутів	$\sum\varphi_1^{(0)}$	$\sum\varphi_2^{(0)}$	$\sum\varphi_3^{(0)}$	$\sum\varphi_4^{(0)}$	$\sum\varphi_5^{(0)}$	$\sum\varphi_6^{(0)}$	$\sum\varphi_7^{(0)}$	$\sum\varphi_8^{(0)}$	$\sum\varphi_9^{(0)}$
$\varphi_1^{(0)}$	44,46	44,46								
$\varphi_2^{(0)}$	-58,60	-14,14	-58,60							
$\varphi_3^{(0)}$	3,29	-10,85	-55,31	3,29						
$\varphi_4^{(0)}$	14,76	3,92	-40,54	18,06	14,76					
$\varphi_5^{(0)}$	41,47	45,39	0,92	59,53	56,23	41,47				
$\varphi_6^{(0)}$	-78,35	-32,96	-77,42	-18,82	-22,12	-36,88	-78,35			
$\varphi_7^{(0)}$	50,75	17,79	-26,68	31,93	28,63	13,87	-27,60	50,75		
$\varphi_8^{(0)}$	2,40	20,18	-24,28	34,32	31,03	16,26	-25,20	53,14	2,40	
$\varphi_9^{(0)}$	-20,18	0,00	-44,46	14,14	10,85	-3,92	-45,39	32,96	-17,79	-20,18
$\varphi_1^{(0)}$	44,46		0,00	58,60	55,31	40,54	-0,92	77,42	26,68	24,28
$\varphi_2^{(0)}$	-58,60			0,00	-3,29	-18,06	-59,53	18,82	-31,93	-34,32
$\varphi_3^{(0)}$	3,29				0,00	-14,76	-56,23	22,12	-28,63	-31,03
$\varphi_4^{(0)}$	14,76					0,00	-41,47	36,88	-13,87	-16,26
$\varphi_5^{(0)}$	41,47						0,00	78,35	27,60	25,20
$\varphi_6^{(0)}$	-78,35							0,00	-50,75	-53,14
$\varphi_7^{(0)}$	50,75								0,00	-2,40
$\varphi_8^{(0)}$	2,40									0,00

Проаналізувавши числові дані таблиці 3.3, приходимо до висновку, що співпадінь сум $\sum\varphi_i^{(0)}$ $(i=1, 2, ...N-1)$ з кутами образу φ'_j $(j=1, ...M-1)$ при деякій точності ε не виявлено, а це означає, що головна зірка бази $S_0^{(0)}$ не належить шуканому образу. Цей факт добре видно на рисунках 3.1 та 3.2.

Виключимо зірку $S_0^{(0)}$ зі списку елементів бази, зменшимо їх кількість N на 1 та перейдемо на першу ітерацію роботи алгоритму. Використовуючи наступну за величиною зірку $S_1^{(0)}$ в якості головної $S_0^{(1)}$, проведемо обчислення кутів $\varphi_i^{(1)}$ та відстаней $l(S_0^{(1)};S_i^{(1)})$. Отримані результати наведено в таблиці 3.4 та на рисунку 3.4. Зауважимо, що в даному випадку $\sum_{k=1}^{N-1}\varphi_k^{(1)}=360^0$.

Таблиця 3.4

Характеристики зірок бази на першій ітерації алгоритму

Зірка $S_i^{(1)}$	Маса	Яскравість	Координата центра X	Координата центра Y	Кут $\varphi_i^{(1)}$ в градусах	Відстань $l(S_0^{(1)};S_i^{(1)})$
$S_0^{(1)}$	649	149379	471	410	0,00000	0,00000
$S_1^{(1)}$	426	93908	348	113	74,45852	321,46228
$S_2^{(1)}$	313	68170	641	277	124,11524	215,84485
$S_3^{(1)}$	268	56814	483	585	47,66685	175,41095
$S_4^{(1)}$	226	46733	181	713	85,82004	419,41507
$S_5^{(1)}$	180	37298	143	139	-169,39009	425,47033
$S_6^{(1)}$	179	37247	702	687	143,47186	360,67991
$S_7^{(1)}$	114	21930	298	368	-24,75645	178,02528
$S_8^{(1)}$	87	15895	84	486	78,61402	394,39194

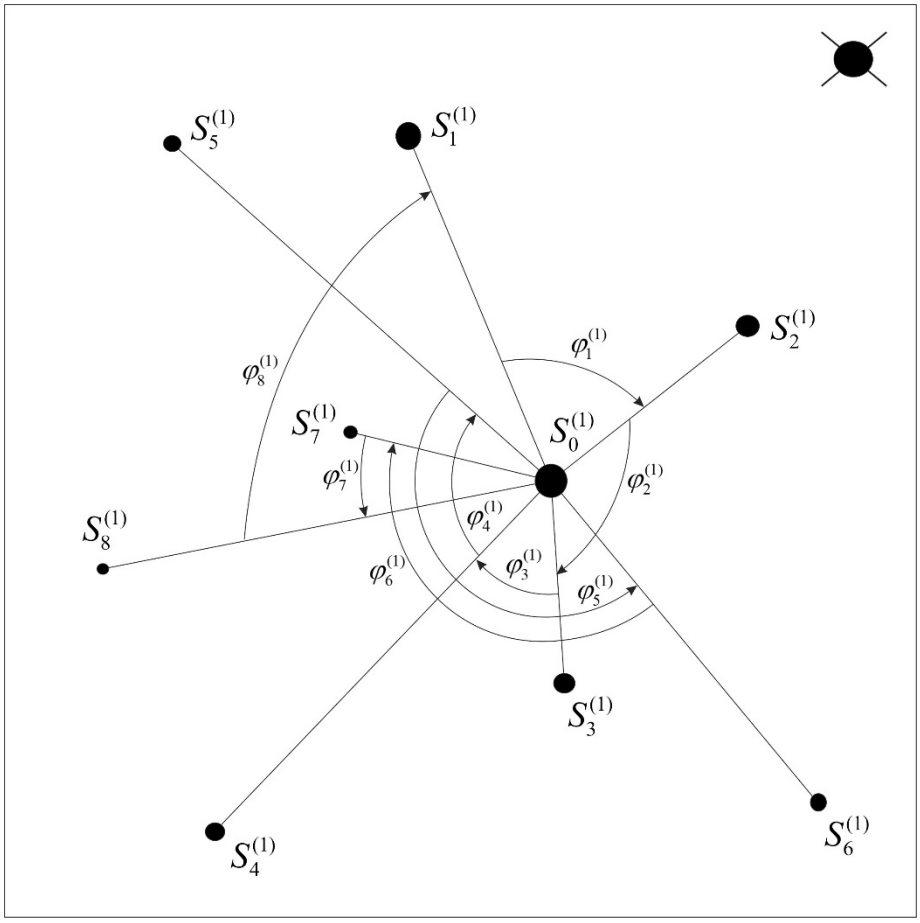


Рис. 3.4. Інвертоване зображення зірок бази на першій ітерації із зазначеними кутами та відстанями

Таблиця 3.5

Суми кутів елементів бази, отримані на першій ітерації алгоритму

$\varphi_i^{(1)}$	Значення кутів	$\sum\varphi_1^{(1)}$	$\sum\varphi_2^{(1)}$	$\sum\varphi_3^{(1)}$	$\sum\varphi_4^{(1)}$	$\sum\varphi_5^{(1)}$	$\sum\varphi_6^{(1)}$	$\sum\varphi_7^{(1)}$	$\sum\varphi_8^{(1)}$
$\varphi_1^{(1)}$	74,459	74,459							
$\varphi_2^{(1)}$	124,115	198,574	**124,115**						
$\varphi_3^{(1)}$	47,667	246,241	171,782	47,667					
$\varphi_4^{(1)}$	85,820	332,061	257,602	133,487	85,820				
$\varphi_5^{(1)}$	-169,390	162,671	88,212	**-35,903**	-83,570	-169,390			
$\varphi_6^{(1)}$	143,472	306,142	231,684	107,569	59,902	-25,918	**143,472**		
$\varphi_7^{(1)}$	-24,756	281,386	206,927	82,812	35,145	-50,675	118,715	-24,756	
$\varphi_8^{(1)}$	78,614	360,000	285,541	161,426	113,759	27,939	197,329	53,858	78,614
$\varphi_1^{(1)}$	74,459		360,000	235,885	188,218	102,398	271,788	**128,316**	153,073
$\varphi_2^{(1)}$	124,115			360,000	312,333	226,513	395,903	252,431	277,188
$\varphi_3^{(1)}$	47,667				360,000	274,180	443,570	300,098	324,855
$\varphi_4^{(1)}$	85,820					360,000	529,390	385,918	410,675
$\varphi_5^{(1)}$	-169,390						360,000	216,528	241,285
$\varphi_6^{(1)}$	143,472							360,000	384,756
$\varphi_7^{(1)}$	-24,756								360,000

Проаналізувавши числові дані таблиць 3.2 та 3.5, бачимо наближену рівність (з точністю ε) наступних величин:

$$\varphi_1' \approx \varphi_2^{(1)};$$

$$\varphi_2' \approx \varphi_3^{(1)} + \varphi_4^{(1)} + \varphi_5^{(1)};$$

$$\varphi_3' \approx \varphi_6^{(1)};$$

$$\varphi_4' \approx \varphi_7^{(1)} + \varphi_8^{(1)} + \varphi_1^{(1)}. \qquad (3.1)$$

Відповідні суми виділені в таблиці 3.5. напівжирним шрифтом.

З (3.1) випливає ототожнення наступних зірок образу і бази:

$$S_0' \sim S_0^{(1)}, \quad S_1' \sim S_2^{(1)}; \quad S_2' \sim S_3^{(1)}; \quad S_3' \sim S_6^{(1)}; \quad S_4' \sim S_7^{(1)}. \qquad (3.2)$$

Порівняємо характеристики відповідних зірок з (3.2) образу (таблиця 3.2) та бази на першій ітерації алгоритму (таблиця 3.4). Зауважимо, що виконуються рівності мас, яскравостей, координат центра та відстаней. Це можливо лише у випадку недеформованих зображень при співпадінні головних зірок $S_0' \sim S_0^{(1)}$.

Застосунок, який реалізовує алгоритм пошуку складного графічного образу в кваліфікаційній роботі, в даному випадку знайшов усі $M = 5$ зірок образу з точністю $\varepsilon = 0.0001$ (рис.3.5).

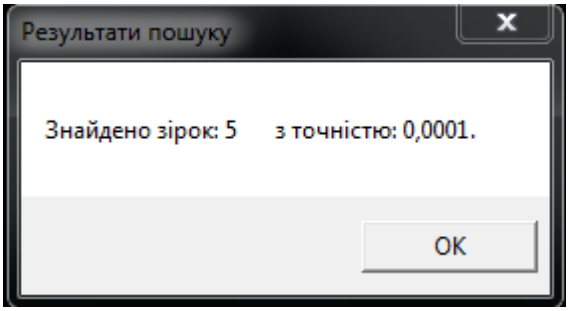


Рис. 3.5. Вікно застосунку з результатами пошуку

Розглянемо випадок, коли до області Ω (рис. 3.1) застосовано поворот навколо її центра на кут 45^0. Отримаємо нове зображення для пошуку (рис. 3.6). База пошуку залишається та ж сама (рис. 3.1), а це означає, що всі її характеристики (таблиці 3.1 та 3.4) та обчислення алгоритму (таблиці 3.3 та 3.4) будуть без змін.

Обчислимо характеристики образу після повороту (таблиця 3.6) та порівняємо їх з характеристиками зображення без геометричних перетворень (таблиця 3.2). Бачимо відмінність усіх параметрів: маси, яскравості, координат центра, кутів між елементами та відстаней до головної зірки.

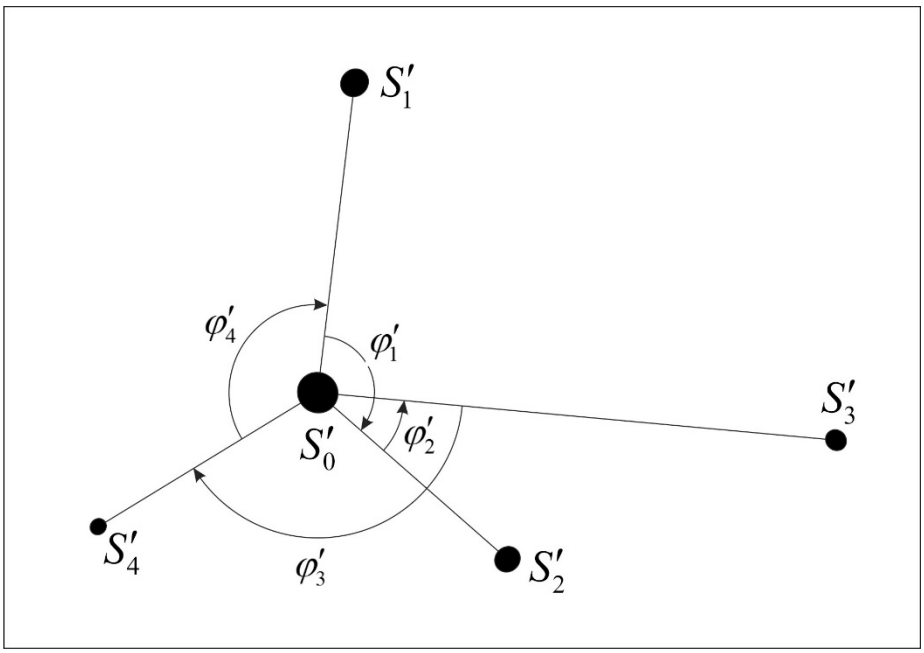


Рис. 3.6. Інвертоване зображення зірок образу

при повороті на 45^0 із зазначеними кутами та відстанями

Таблиця 3.6

Характеристики зірок образу при повороті зображення на 45^0

Зірка S'_j	Маса	Яскравість	Координата центра X	Координата центра Y	Кут φ'_j в градусах	Відстань $l(S'_0;S'_j)$
S'_0	656	149362	321	455	0	0
S'_1	317	68238	347	240	124,16746	216,56639
S'_2	263	56712	453	570	-35,98316	175,06856
S'_3	183	37162	681	487	143,73542	361,41942
S'_4	113	21998	169	547	128,08028	177,67386

Похибки, що виникають при перетворення зображення, впливають на точність роботи алгоритму. Як видно на рисунку 3.7, де показано результати пошуку, програмний застосунок здійснив п’ять спроб знайти елементи образу в базі. Частково, три зірки з п’яти, знайдено з точністю $\varepsilon = 0.1$ та повністю – з точністю $\varepsilon = 1$.

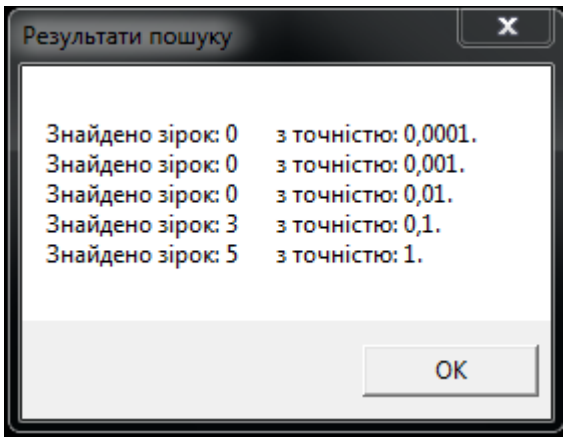


Рис.3.7. Вікно застосунку з результатами пошуку

Розглянемо виконання алгоритму при $\varepsilon = 1$. На нульовій ітерації (таблиця 3.3) виявлено близькі значення кутів:

$$\varphi_2' = -35.98316 \text{ та } \varphi_5^{(0)} + \varphi_6^{(0)} = -36.88.$$

Їх різниця по модулю $\left|\varphi_2' - (\varphi_5^{(0)} + \varphi_6^{(0)})\right| = 0.89684 < 1$, але на рисунку 3.2 видно, що зірки S_2' та $S_5^{(0)}$ не тотожні. Ця проблема усувається додатковою перевіркою (2.11). Визначимо співвідношення коренів мас та відстаней даних об'єктів:

$$\sqrt{\frac{m(S_5^{(0)})}{m(S_2')}} = \sqrt{\frac{226}{263}} = 0.92698;$$

$$\frac{l(S_0^{(0)}, S_5^{(0)})}{l(S_0', S_2')} = \frac{865.15216}{175.06856} = 4.94179.$$

Бачимо, що отримані коефіцієнти значно відрізняються, а отже співпадіння зірок S_2' та $S_5^{(0)}$ виключається.

На першій ітерації алгоритму виконуються наближені рівності (3.1) кутів об'єктів образу та сум кутів об'єктів бази, а отже і ототожнення зірок (3.2) Здійснимо перевірку додаткової умови та знайдемо похибки для кожного елемента образу, окрім головної зірки, для випадку $\varepsilon = 1$. Результати проведених обчислень представлено в таблицях 3.7 – 3.9.

Таблиця 3.7

Співвідношення між масами тотожних зірок образу та бази

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Маса зірки образу $m(S'_j)$	Маса зірки бази $m(S_i^{(1)})$	$K_1 = \sqrt{\frac{m(S_i^{(1)})}{m(S'_j)}}$
S'_1	$S_2^{(1)}$	317	313	0,98738
S'_2	$S_3^{(1)}$	263	268	1,01901
S'_3	$S_6^{(1)}$	183	179	0,97814
S'_4	$S_7^{(1)}$	113	114	1,00885

Таблиця 3.8

Співвідношення між відстанями до головної зірки тотожних зірок образу та бази

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Відстань $l(S'_0, S'_j)$	Відстань $l(S_0^{(1)}, S_i^{(1)})$	$K_2 = \frac{l(S_0^{(1)}, S_i^{(1)})}{l(S'_0, S'_j)}$
S'_1	$S_2^{(1)}$	216,56639	215,84485	0,99667
S'_2	$S_3^{(1)}$	175,06856	175,41095	1,00196
S'_3	$S_6^{(1)}$	361,41942	360,67991	0,99795
S'_4	$S_7^{(1)}$	177,67386	178,02528	1,00198

Таблиця 3.9

Визначення похибок для кутів об’єктів образу

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Кут φ'_j в градусах	Сума кутів $\sum \varphi_i^{(1)}$ в градусах	Абсолютна похибка $\Delta_j = \left\lvert \varphi'_j - \sum \varphi_i^{(1)} \right\rvert$	Відносна похибка $\delta_j = \frac{\Delta_j}{\left\lvert \varphi'_j \right\rvert}$
S'_1	$S_2^{(1)}$	124,16746	124,115	0,05246	0,00042
S'_2	$S_3^{(1)}$	-35,98316	-35,903	0,08016	0,00222
S'_3	$S_6^{(1)}$	143,73542	143,472	0,26342	0,00183
S'_4	$S_7^{(1)}$	128,08028	128,316	0,23572	0,00184

Проаналізувавши дані таблиць 3.7 та 3.8, можемо зробити висновок про наближену рівність коефіцієнтів K_1 та K_2 з різницею до 0.02. Це означає виконання додаткової умови перевірки (2.11), що підтверджує виконання співвідношень (3.2) про тотожність відповідних елементів бази та образу.

Порівнюючи значення абсолютних похибок $\Delta_j = \left|\varphi'_j - \sum \varphi_i^{(1)}\right|$, бачимо що $\max \Delta_j = 0,26342 < \varepsilon$. Для відносних похибок $\delta_j = \dfrac{\Delta_j}{\left|\varphi'_j\right|}$ найбільше значення складає $\max \delta_j = 0,00222$, що є гарним результатом для пошуку в умовах нечітких даних.

Застосуємо до зображення області Ω (рис. 3.1) комбіноване перетворення: поворот на 90^0 та збереження зі збільшенням dpi вдвічі. Отримаємо новий образ для пошуку (рис.3.8), база залишається тією ж.

Dpi (англійською *dots per inch* – українською *кількість точок на дюйм*) – показник роздільної здатності растрового графічного файлу. Цей показник впливає на розмір зображення у пікселях, а отже і на масштаб.

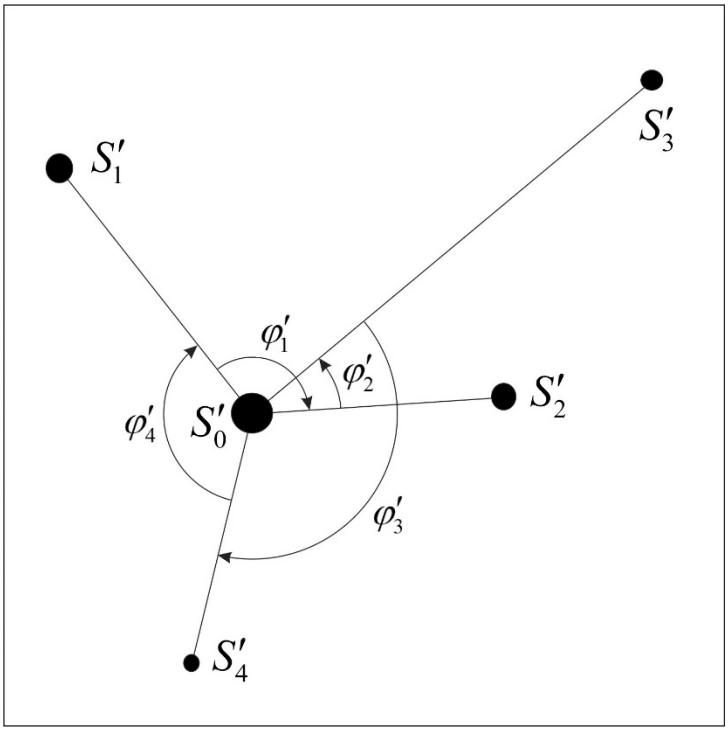


Рис.3.8. Інвертоване зображення зірок образу при повороті на 90^0 та збільшенні масштабу вдвічі із зазначеними кутами та відстанями

Обчислимо характеристики для об’єктів образу при повороті на 90^0 та збільшенні масштабу вдвічі (таблиця 3.10). Порівнюючи ці параметри із відповідними значеннями недеформованого зображення (таблиця 3.2), спостерігаємо значні відмінності в масі, яскравості, координатах центра та відстанях. Кути не значно змінилися під час перетворення графічного файлу.

Таблиця 3.10

Характеристики зірок образу при повороті на 90^0
та збільшенні масштабу вдвічі

Зірка	Маса	Яскравість	Координата центра X	Координата центра Y	Кут φ'_j в градусах	Відстань $l(S'_0;S'_j)$
S'_0	2558	596377	551	934	0	0
S'_1	1213	272374	283	594	124,47594	432,92493
S'_2	1029	227375	900	911	-36,05542	349,75706
S'_3	697	149498	1105	472	143,47186	721,35983
S'_4	430	87805	467	1280	128,10762	356,05056

Оскільки значення кутів близькі до попереднього розглянутого варіанту пошуку, то і хід алгоритму співпадатиме. На нульовій ітерації відкинемо зі списку бази головну зірку $S_0^{(0)}$, а на першій ітерації побачимо виконання тотожностей (3.1) та (3.2). В цьому прикладі цікаво дослідити вплив зміни зображення. Для цього обчислимо співвідношення між масами тотожних зірок образу та бази (таблиця 3.11) та їх відстанями до головної зірки (таблиця 3.12).

Аналізуючи числові результати в таблицях 3.11 та 3.12, бачимо, що найбільша різниця між коефіцієнтами $|K_1 - K_2| < 0.15$. Це задовольняє задану точність, але в декілька разів перевищує максимальну різницю $K_1 - K_2 = 0,01981$ при повороті зображення образу на 45^0.

Таблиця 3.11

Співвідношення між масами тотожних зірок образу та бази

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Маса зірки образу $m(S'_j)$	Маса зірки бази $m(S_i^{(1)})$	$K_1 = \sqrt{\frac{m(S_i^{(1)})}{m(S'_j)}}$
S'_1	$S_2^{(1)}$	2558	313	0,34980
S'_2	$S_3^{(1)}$	1213	268	0,47004
S'_3	$S_6^{(1)}$	1029	179	0,41708
S'_4	$S_7^{(1)}$	697	114	0,40442

Таблиця 3.12

Співвідношення між відстанями до головної зірки тотожних зірок образу та бази

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Відстань $l(S'_0, S'_j)$	Відстань $l(S_0^{(1)}, S_i^{(1)})$	$K_2 = \frac{l(S_0^{(1)}, S_i^{(1)})}{l(S'_0, S'_j)}$
S'_1	$S_2^{(1)}$	432,92493	215,84485	0,49857
S'_2	$S_3^{(1)}$	349,75706	175,41095	0,50152
S'_3	$S_6^{(1)}$	721,35983	360,67991	0,50000
S'_4	$S_7^{(1)}$	356,05056	178,02528	0,50000

Обчислимо абсолютні та відносні похибки для кутів тотожних об’єктів образу (таблиця 3.13). Порівнюючи значення абсолютних похибок Δ_j, бачимо що $\max \Delta_j = 0,36094 < \varepsilon$. Для відносних похибок δ_j найбільше значення складає $\max \delta_j = 0,00423$. Значення $\max \Delta_j$ на 70% більше за найбільшу абсолютну похибку при повороті образу на 45^0, а $\max \delta_j$ – більше на 50% за найбільшу відносну похибку. Це свідчить про вплив деформацій зображення на точність обчислень. Чим більше геометричних перетворень було застосовано до графічного файлу, тим більші похибки отримаємо в процесі пошуку.

Таблиця 3.13

Визначення похибок для кутів об’єктів образу

Зірка образу S'_j	Відповідна зірка бази $S_i^{(1)}$	Кут φ'_j в градусах	Сума кутів $\sum \varphi_i^{(1)}$ в градусах	Абсолютна похибка $\Delta_j = \left\|\varphi'_j - \sum \varphi_i^{(1)}\right\|$	Відносна похибка $\delta_j = \frac{\Delta_j}{\left\|\varphi'_j\right\|}$
S'_1	$S_2^{(1)}$	124,47594	124,115	0,36094	0,00290
S'_2	$S_3^{(1)}$	-36,05542	-35,903	0,15242	0,00423
S'_3	$S_6^{(1)}$	143,47186	143,472	0,00014	0,00000
S'_4	$S_7^{(1)}$	128,10762	128,316	0,20838	0,00163

Розглянемо ситуацію, коли шукане сузір’я зафіксовано іншим фотоапаратом з кращою якістю зображення. Тоді на ньому можливі додаткові малі об’єкти, які не зміг вловити перший прилад. Для моделювання цього випадку добавимо в область Ω дві додаткові зірки меншого розміру і отримаємо новий образ для пошуку (рис.3.9).

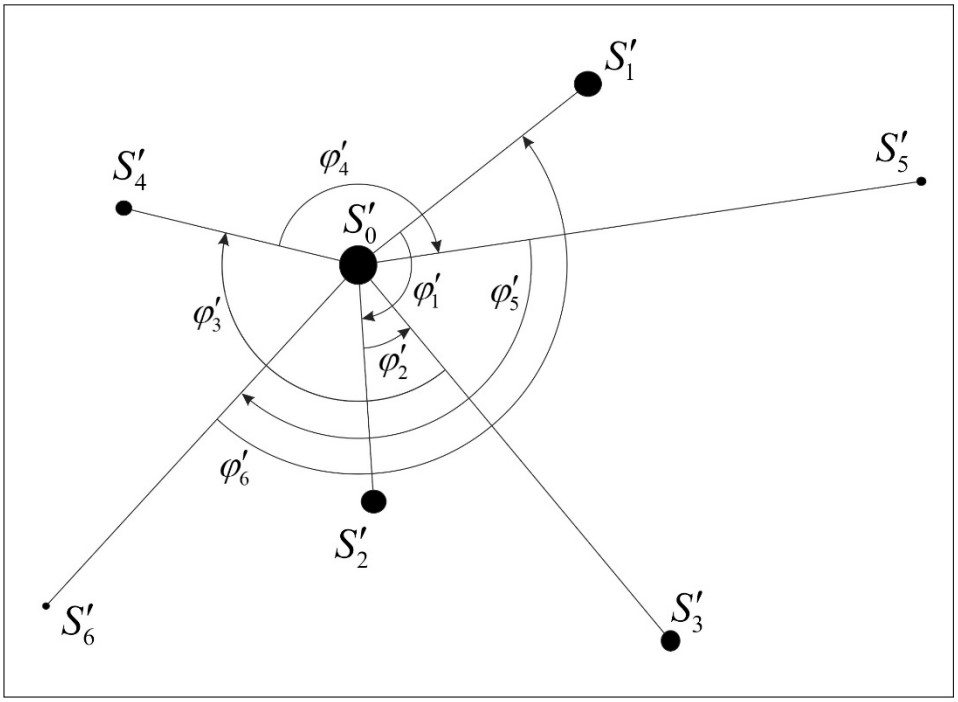


Рис.3.9. Інвертоване зображення зірок образу при переміщенні та наявності додаткових елементів із зазначеними кутами та відстанями

Крім цього до зображення образу застосовано паралельне перенесення. База пошуку лишається без змін (рис.3.1). Порівнюючи рисунки, бачимо, що S'_5 та S'_6 – зірки, які добавлені в образ, але відсутні у базі.

Обчислимо характеристики усіх $M = 7$ елементів образу, результати розрахунків наведені в таблиці 3.14.

Таблиця 3.14

Характеристики зірок образу при переміщенні
та наявності додаткових елементів

Зірка	Маса	Яскравість	Координата центра X	Координата центра Y	Кут φ'_j в градусах	Відстань $l(S'_0;S'_j)$
S'_0	654	149124	338	291	0	0
S'_1	315	68425	508	157	124,62909	216,46247
S'_2	265	56525	349	465	-36,20858	174,34735
S'_3	183	25211	569	568	143,47186	360,67991
S'_4	113	21998	165	249	157,87717	178,02528
S'_5	40	6443	754	229	141,10082	420,59482
S'_6	24	3145	107	542	-170,87037	341,11875

Порівнюючи кути φ'_j таблиці 3.14 та числові дані таблиці 3.5, де наведено суми кутів на першій ітерації алгоритму для об’єктів бази, знаходимо наближені рівності:

$$\varphi'_1 \approx \varphi_2^{(1)}; \quad \varphi'_2 \approx \varphi_3^{(1)} + \varphi_4^{(1)} + \varphi_5^{(1)}; \quad \varphi'_3 \approx \varphi_6^{(1)}. \tag{3.3}$$

З (3.3) випливає ототожнення наступних зірок образу і бази:

$$S'_0 \sim S_0^{(1)}, \quad S'_1 \sim S_2^{(1)}; \quad S'_2 \sim S_3^{(1)}; \quad S'_3 \sim S_6^{(1)}. \tag{3.4}$$

Еквівалентність зірок S'_4 та $S_7^{(1)}$ по рівності кутів не виявлено, тому що на визначення кутів впливають об’єкти S'_5 та S'_6. Кількість знайдених елементів при заданій точності ε видно на рисунку 3.10. Тобто найбільша кількість знайдених елементів дорівнює чотирьом із п’яти можливих при $\varepsilon = 1$.

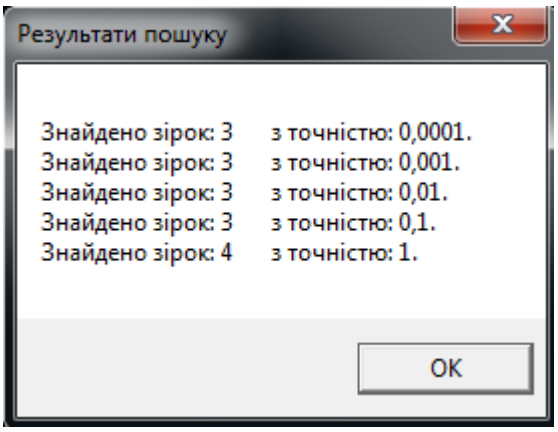


Рис.3.10. Вікно застосунку з результатами пошуку

Проаналізувавши дану проблему, можна запропонувати два шляхи її розв’язання. В першому варіанті доцільно вилучати зі списку елементів образу зірки з найменшою масою, проводити перерахунок кутів та повторювати процес пошуку спочатку.

Для другого підходу розглянемо суми кутів:

$$\varphi_7^{(1)} + \varphi_8^{(1)} + \varphi_1^{(1)} = 128{,}316\,;$$

$$\varphi_4' + \varphi_5' + \varphi_6' = 128{,}10762\,. \qquad (3.5)$$

Бачимо наближену рівність, з якої слідує, що можна виконувати порівняння між сумами кутів елементів бази та сумами кутів елементів образу.

ВИСНОВКИ

У кваліфікаційній роботі запропоновано оригінальний підхід до вирішення інтелектуальної задачі розпізнавання складного графічного образу за умов нечітко визначених даних. В якості модельного прикладу розглянуто задачу ідентифікації та пошуку сузір'я на карті зоряного неба, що було сфотографоване при невизначених умовах фіксації зображення.

Формалізація подання складного образу здійснено шляхом переходу від піксельного представлення у графічному файлі до наборів числових параметрів, що визначають розміри, взаємне розміщення та положення складових частин образу. На основі отриманих списків атрибутів розроблено алгоритм пошуку позиції входження складного графічного образу у велике зображення.

Тестування алгоритму на наборі модельних прикладів дозволило засвідчити його ефективність в наступних випадках ускладнення вхідних даних:

- коректно опрацьовуються випадки пошуку образу, що входить у базове зображення із масштабуванням: як із збільшенням, так і зменшенням розміру;
- наявність переміщень та поворотів образу незначно впливає на точність результату пошуку. Похибка результату з'являється за рахунок незначних спотворень числових параметрів образу, які виконуються графічними редакторами при перетвореннях.
- комбінація декількох видів перетворень зображень застосованих до образу, наприклад поворот навколо точки та зміна розміру, збільшує похибку порівняння кутів відповідних зірок образу і бази.

Дослідження в даному напрямку та вдосконалення алгоритму можуть бути продовжені з метою зменшення впливів перетворень зображень, покращення результатів пошуку при наявності сторонніх елементів, розширення процесу ідентифікації на кольорові зображення.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Дэвид А. Форсайт, Джин Понс Компьютерное зрение. Современный подход. М.: «Вильямс», 2004. 928 с.
2. Копча-Горячкіна Г. Е. Теорія розпізнавання образів: навчально-методичний посібник: у 2 ч. Ужгород: Видавництво ДВНЗ «Ужгородського національного університету», 2016. Ч. І: 59 с.
3. Довбиш А. С., Шелехов І. В. Основи теорії розпізнавання образів : навчальний посібник: у 2 ч. Суми : Сумський державний університет, 2015. Ч. 1: 109 с.
4. Кутковецький В. Я. Розпізнавання образів : навчальний посібник. Миколаїв: Вид-во ЧНУ ім. Петра Могили, 2017. 420 с.
5. Потапов А. С. Распознавание образов и машинное восприятие: Общий подход на основе принципа минимальной длины описания. СПб.: Политехника, 2007. 548 с.
6. Гонсалес Р., Вудс Р. Цифровая обработка информации. М.: Техносфера, 2005. 1072 с.
7. Фурман Я. А., Кревецкий А. В., Рожнецов А. К. Введение в контурный анализ и его приложения к обработке изображений и сигналов. М.: ФИЗМАТЛИТ, 2003. 592 с.
8. Шапиро Л., Стокман Дж. Компьютерное зрение. М.: Бином. Лаборатория знаний, 2006. 752 с.
9. Обработка и анализ изображений в задачах машинного зрения / Желтов С.Ю. и др. М.: Физматкнига, 2010. 672 с..
10. Добровольський Ю.Г., Прохоров Г.В. Інженерна та комп’ютерна графіка. Чернівці: БДФЕУ, 2012. 142 с.
11. Васильев В.И. Распознающие системы. К.: «Наукова думка», 1983. 422 с.
12. Чабан Л.Н. Теория и алгоритмы распознавания образов. Учебное пособие. М.: МИИГАиК. 2004. 70 с.
13. Ту Дж., Гонсалес Р. Принципы распознавания образов. М.: Мир, 1978. 405 с.

14. Фукунага К. Введение в статистическую теорию распознавания образов. М.: Наука, 1979. 368 с.
15. Теорія розпізнавання образів. *Українська Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Теорія_розпізнавання_образів (дата звернення: 17.01.2022).
16. Комп'ютерний зір. *Українська Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Комп'ютерний_зір (дата звернення: 21.01.2022).
17. Руководство. Создание приложения WPF с помощью C #. *Microsoft Learn*. URL: https://learn.microsoft.com/ru-ru/visualstudio/get-started/csharp/tutorial-wpf?view=vs-2022 (дата звернення: 12.07.2022).
18. Руководство по классическим приложениям (WPF .NET). *Microsoft Learn*. URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/overview/?view=netdesktop-6.0 (дата звернення: 12.07.2022).
19. Обзор XAML (WPF .NET). *Microsoft Learn*. URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/xaml/?view=netdesktop-6.0 (дата звернення: 12.07.2022).
20. System.Windows.Media.Imaging Namespace. *Microsoft Learn*. URL: https://learn.microsoft.com/en-us/dotnet/api/system.windows.media.imaging?view=windowsdesktop-7.0 (дата звернення: 08.09.2022).
21. Comparisons and sorts within collections. *Microsoft Learn*. URL: https://learn.microsoft.com/en-us/dotnet/standard/collections/comparisons-and-sorts-within-collections (дата звернення: 22.09.2022).
22. Новые графические и мультимедийные возможности в WPF 4. *Microsoft Learn*. URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/graphics-multimedia/?view=netframeworkdesktop-4.85 (дата звернення: 10.10.2022).
23. Шевцова Н.В., Сяський В.А. Вирішення інтелектуальної задачі розпізнавання та пошуку складного графічного образу. *Інформаційні технології в професійній діяльності :* матеріали XV Всеукраїнської науково-практичної конференції. Рівне : РВВ РДГУ, 2022. С. 220-223.