Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

РОЗРОБКА ТА ДОСЛІДЖЕННЯ ПОРІВНЯННЯ ПОЛІГОНАЛЬНИХ СУПЕРПІКСЕЛЬНИХ СЕГМЕНТАЦІЙ ЗОБРАЖЕНЬ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-21-1
Фоменко Н.О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір В.П.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Фоменку Назару Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження порівняння полігональних суперпіксельних сегментацій зображень

затверджена наказом по університету від 9 листопада 2022 року № 1469Ст

2. Термін подання студентом роботи до екзаменаційної комісії 2 грудня 2022 р.

3. Вихідні дані до роботи науково-методична література, дані інтернет-мережі, методи та засоби класифікації зображень: інтелектуальні оператори сегментації. Математичні моделі перетворювання зображень, перелік використовуваних програмних засобів: теоретичні відомості про методи сегментації текстурних зображень.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд та постановка задачі: сегментація.
2. Суперпікселі та алгоритми з вибором базового.
3. Теоретична частина суперпіксельної сегментації багатокутників, порівняння з сегментацією.
4. Реалізація та експерименти: порівняння звичайної суперпіксельної сегментації та багатокутників.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми, мета та об'єкт дослідження, постановка задачі, аналіз предметної області, методи вирішення поставлених задач, вихідні данні, дослідження запропонованих методів при вирішення поставлених задач.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Творошенко І.С.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	09.11.2022	
2	Аналіз завдання, підбір літератури	09.11.22-10.11.22	
3	Аналіз літератури з досліджуваної проблеми	11.11.22	
4	Аналіз методів розпізнавання дорожніх знаків	12.11.22	
5	Розробка методів розпізнавання	13.11.22	
6	Програмна реалізація	13.11.22-15.11.22	
7	Оформлення пояснювальної записки	15.11.22-18.11.22	
8	Перевірка на плагіат	21.11.2022	
9	Рецензування	22.11.2022	
10	Підготовка презентації та доповіді	22.11.2022	
11	Занесення роботи в електронний архів	29.11.2022	
12	Попередній захист кваліфікаційної роботи	12.12.2022	

Дата видачі завдання 9 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Машталір В.П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 48 с., 16 рис., 35 джерел.

СЕГМЕНТАЦІЯ, СУПЕРПІКСЕЛІ, ПОРІВНЯННЯ СЕГМЕНТАЦІЙ, SLIC, РЕАЛІЗАЦІЯ ТА ЕКСПЕРЕМЕНТ.

Об’єктом дослідження є процес розроблення застосунку для полігональних суперпіксельних сегментацій зображення.

Метою дослідження є порівняння полігональних суперпіксельних сегментацій зображень.

В експеременті використані алгоритми : кластерного типу SLIC, SLICO (Simple linear iterative clustering), топологічний Watershed, щільний QS, графовий FH (Felzenszwalb and Huttenlocher), заснований на однорідності сегментів і меж SEEDS (Superpixels Extracted via Energy- (Linear Spectral Clustering Superpixel).

Проведено дослідження класичних методів розпізнавання та класифікації образів.

Результати роботи здійснена програмна реалізація різних методів , як окремих компонентів.

SEGMENTATION, SUPERPIXELS, SEGMENTATION REPAIR, SLIC, IMPLEMENTATION OF THAT EXPERIMENT.

The object of research is the process of splitting the image for polygonal superpixel image segmentation.

The method of accomplishment is the alignment of polygonal superpixel segmentation images.

In the experiment, there are algorithms: cluster-type SLIC, SLICO (Simple linear iterative clustering), topological Watershed, scalable QS, graph FH (Felzenszwalb and Huttenlocher), grounding on homogeneity of segments and between SEEDS (Superpixels Extracted via Energy- (Linear Spectral Clustering Superpixel) .

A follow-up of classical methods of recognition and classification of images was carried out.

The results of the work are designed software implementation of various methods, like other components.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ASLIC (Adaptive Simple Linear Iterative Clustering)

MSLIC (Manifold Simple Linear Iterative Clustering)

SLIC (Simple Linear Iterative Clustering)

СТЗ – система технічного зору

ІП – інваріантна пряма

СС – суперпіксельна сегментація

ВСТУП

Комп’ютерна графіка розділяється за трьома основними напрямками: візуалізація, обробка зображень і розпізнавання образів [1-6].

Візуалізація – це створення зображення на основі якогось опису (моделі). Приміром, це може бути відображення графіку, схеми, імітація тривимірної віртуальної реальності в комп’ютерних іграх, у системах архітектурного проектування.

Основне завдання – порівняння полігональних суперпіксельних сегментацій зображень Мета розпізнавання може бути різна: як виділення окремих елементів на зображенні, так і класифікація зображення в цілому. У якомусь сенсі завдання розпізнавання є зворотним стосовно завдання візуалізації.

Актуальність дослідження полягає у тому що більшість стандартних алгоритмів обробки зображень використовують їх подання у вигляді регулярної піксельної сітки, що обумовлено переважно вихідним способом зберігання зображень у цифровій формі.

Однак це не завжди оптимально з багатьох точок зору і насамперед для організації подальшої обробки. У цьому плані відносно новим і активно розвивається насамперед за кордоном є похід, заснований на так званій суперпіксельній сегментації (СС).

СС реалізує розбиття зображення на безліч дрібних фрагментів (суперпікселів), що являють собою відносно однорідні групи розташованих поруч пікселів.

Кожен суперпіксель потенційно є атомарним регіоном (фрагментом) зображення, тобто всі пікселі, що входять до нього, розглядаються при подальшій обробці як єдине ціле. Тому актуальність роботи полягає в тому що суперпіксельна сегментація знижує складність обчислення.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ СУПЕРПІКСЕЛЬНИХ СЕГМЕНТАЦІЙ

1.1 Перший метод SLIC

Проста лінійна ітераційна кластеризація є адаптацією *k*-середніх для генерації суперпікселя з двома важливими відмінностями.

Кількість обчислень відстані під час оптимізації зменшено шляхом обмеження простору пошуку областю, пропорційною розміру суперпікселя. Це зменшує складність лінійної кількості пікселів і незалежності від кількості суперпікселів [7].

Зважена міра відстані поєднує колір і просторову близькість, водночас забезпечуючи контроль над розміром і компактністю суперпікселів.

SLIC простий у використанні. Єдиним параметром алгоритму є бажана k кількість суперпікселів приблизно однакового розміру.

Для кольорових зображень у колірному просторі CIELAB процедура кластеризації починається з етапу ініціалізації, де k початкові центри кластерів $C_i = [l_i, a_i, b_i, x_i, y_i]^T$ відбираються на звичайній сітці на відстані S пікселів один від одного. Для створення суперпікселів приблизно однакового розміру інтервал сітки становить $S = \sqrt{N / k}$. Центри переміщуються до початкових місць, що відповідають найнижчому положенню градієнта 3×3 в околі. Це робиться, щоб уникнути центрування суперпікселя на краю та зменшити ймовірність засівання суперпікселя пікселем із шумом.

На етапі призначення кожен піксель асоціюється з найближчим центром кластера, область пошуку якого перекриває його розташування. Це можливо лише завдяки введенню міри відстані *D*, яка визначає найближчий центр кластера для кожного пікселя. Колір пікселя представлений у просторі кольорів CIELAB $[l, a, b]^T$, діапазон можливих значень якого відомий. Позиція пікселя $[x, y]^T$, з іншого боку, може приймати діапазон значень, який змінюється відповідно до розміру зображення.

Щоб об’єднати дві відстані в одну міру, необхідно нормалізувати колірну та просторову близькість за їхніми відповідними максимальними відстанями в межах кластера N_S та N_C.

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

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

$$D = \sqrt{(d_C / N_C)^2 + (d_S / N_S)^2}.$$

Максимальна просторова відстань, очікувана в межах даного кластера, має відповідати інтервалу вибірки $N_S = S = \sqrt{N / k}$. Визначити максимальну колірну відстань N_C не так просто, оскільки колірні відстані можуть значно відрізнятися від кластера до кластера та зображення до зображення. Цієї проблеми можна уникнути N_C зафіксувавши константу m таким чином, що стає

$$D = \sqrt{(d_C / m)^2 + (d_S / S)^2},$$

який спрощує вимірювання відстані, яке використовується на практиці:

$$D = \sqrt{d_C^2 + (d_S / S)^2 \cdot m^2}.$$

Оскільки очікуваний просторовий розмір суперпікселя є областю приблизного розміру $S \times S$, пошук подібних пікселів виконується в області $2S \times 2S$ навколо центру суперпікселя.

Після того, як кожен піксель буде пов’язано з найближчим центром кластера, етап оновлення налаштовує центри кластера на середній вектор усіх пікселів $[l,a,b,x,y]^T$ що належать до кластера.

Норма використовується для обчислення залишкової помилки E між новим розташуванням центру кластера та попереднім розташуванням центру кластера. Щоб виправити пікселі, які не належать до того самого підключеного компонента, що й центр кластера, цим пікселям призначається мітка найближчого центру кластера за допомогою алгоритму підключених компонентів [8-10].

1.2 Другий метод ASLIC

Adaptive-SLIC, або ASLIC, адаптує колірну та просторову нормалізацію в кожному кластері. Як описано для SLIC, і є припущеними максимальними просторовими та колірними відстанями в межах кластера.

Ці постійні значення використовуються для нормалізації кольору та просторової близькості, тому їх можна об’єднати в одну міру відстані для кластеризації. Замість використання постійних значень ASLIC динамічно нормалізує наближення для кожного кластера, використовуючи максимальні спостережувані просторові та колірні відстані з попередньої ітерації.

Таким чином, міра відстані набуває вигляду:

$$D = \sqrt{\left(\frac{d_C}{m_C}\right)^2 + \left(\frac{d_S}{m_S}\right)^2}.$$

Як і раніше, для першої ітерації використовуються постійні коефіцієнти нормалізації, але потім алгоритм відстежує максимальні відстані для кожного кластера.

Перевага цього підходу полягає в тому, що компактність суперпікселя більш стабільна, і ніколи не потрібно встановлювати m.

1.3 Третій метод MSLIC

Manifold SLIC розширює SLIC для обчислення чутливих до вмісту суперпікселів, водночас він успадковує всі переваги SLIC, такі як простота та висока продуктивність.

Подібно до SLIC, MSLIC представляє колір у просторі кольорів CIELAB і позначається $c(u,v) = (l(u,v), a(u,v), b(u,v))$ колір пікселя (u,v) на зображенні I. Потім ми визначаємо карту розтягування $\Phi : I \to \square^5$ для надсилання пікселів у 2-розбірник M, вбудований у 5-вимірне комбіноване зображення та колірний простір $\Phi(u,v) = (\lambda_1 p, \lambda_2 c) = (\lambda_1 u, \lambda_1 v, \lambda_2 l, \lambda_2 a, \lambda_2 b)$, де λ_1 і λ_2 – глобальні коефіцієнти розтягування. Ми встановлюємо $\lambda_1 = 1/N_S$ і $\lambda_2 = 1/N_C$ та приймаємо метрику SLIC D для вимірювання відстані між точками на різноманітті M:

$$D(\Phi(p_1), \Phi(p_2)) = \sqrt{\lambda_1^2 d_S^2 + \lambda_2^2 d_C^2}.$$

За піксель $p = (u,v)$ позначимо через одиничний квадрат 1×1 з центром p. Дозволяє p_1, p_2, p_3, p_4 бути чотирма кутами, кожен з яких визначається середнім значенням чотирьох сусідніх пікселів. Квадрат складається з двох трикутників,.

Потім ми наближено оцінюємо площу вигнутого трикутника $\Phi(\square p_1 p_2 p_3)$ площею плоского трикутника $\square\Phi(p_1)\Phi(p_2)\Phi(p_3)$,

$$Area(\Phi(\square p_1 p_2 p_3)) \approx \frac{1}{2}\left\|\overrightarrow{\Phi(p_2)\Phi(p_1)}\right\|\left\|\overrightarrow{\Phi(p_2)\Phi(p_3)}\right\|\sin\theta,$$

де θ – кут між векторами $\overrightarrow{\Phi(p_2)\Phi(p_1)}$ і $\overrightarrow{\Phi(p_2)\Phi(p_3)}$.

Зверніть увагу, що довжина $\left\|\overrightarrow{\Phi(x)\Phi(y)}\right\|$ вимірюється за допомогою метрики *D*. $Area(\Phi(\square p_3 p_4 p_1))$ можна обчислити подібним чином. Площа області $\Phi(\Omega) \subset M$ це просто сума всіх пікселів $p_i \in \Omega$.

Цей метод базується на важливому спостереженні: для регіону $\Omega \subset I \subset \square^2$ площа відповідного регіону $\Phi(\Omega) \subset M$ залежить як від площі Ω так і від інтенсивності або зміни кольору в Ω. Чим більша варіація кольорів у Ω, тим більша площа $\Phi(\Omega)$, і навпаки. Якщо ми можемо обчислити рівномірну тесселяцію на M , інверсне відображення Φ^{-1}спричинить чутливу до вмісту мозаїку на зображенні *I*.

1.4 Постановка задачі дослідження

Постановка задачі дослідження за умовами поставленого завдання потрібно дослідити методи, який буде розпізнавати та сегментувати зображення. Проєктування та розробку потрібно зробити відповідно до поставлених вимог та поставлених строків.

Об’єктом дослідження є процес розроблення застосунку для полігональних суперпіксельних сегментацій зображення.

Метою дослідження є порівняння полігональних суперпіксельних сегментацій зображень.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих методів сегментування зображень;

– розробити алгоритм сегментації суперпікселей;

– реалізувати комп’ютерну модель.

2 ОПИС АЛГОРИТМІВ СУПЕРПІКСЕЛЬНИХ СЕГМЕНТАЦІЙ БАГАТОКУТНИКАМИ

2.1 Аналіз області

Сегментація зображень продовжує залишатися проблемою, яка приваблює як предметно-спеціальні, так і загальні рішення.

Щоб уникнути боротьби із семантикою при використанні традиційних алгоритмів сегментації, останнім часом дослідники звернули свою увагу на набагато простішу та здійсненну задачу, а саме на спрощення зображення на маленькі кластери з’єднаних пікселів, які називаються суперпікселями.

Сегментація суперпікселів швидко стала потужним інструментом попередньої обробки, який спрощує зображення з, потенційно, мільйонів пікселів до приблизно на два порядки величини менших кластерів подібних пікселів [11-16].

Після їх представлення кілька додатків, таких як локалізація об’єктів багатокласова сегментація, оптичний потік, оцінка моделі тіла, відстеження об’єктів і оцінка глибини скористався перевагами суперпікселів.

Для цих застосувань зазвичай очікується, що суперпікселі мають такі властивості:

– тверде дотримання меж області;
– містить невеликий кластер подібних пікселів;
– однорідність; кластери приблизно однакового розміру;
– компактність; обмеження ступеня примикання;
– ефективність обчислень.

Одним із найвідоміших алгоритмів суперпіксельної сегментації є простий лінійний ітеративний алгоритм кластеризації (рис. 2.1).

На зображеннях праворуч показані відповідні багатокутні перегородки SNICPOLY.

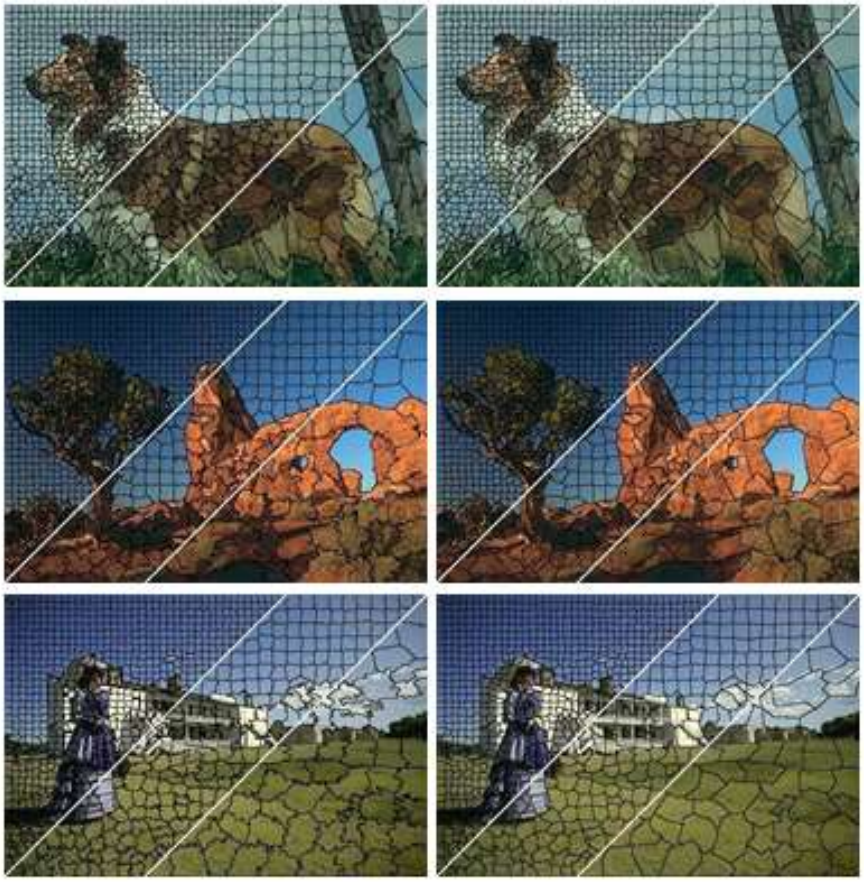


Рисунок 2.1 – Зображення ліворуч показують сегментацію SNIC для трьох різних розмірів суперпікселя

SLIC, який задовольняє цим критеріям і є дуже ефективним з точки зору вимог до обчислень і пам’яті. Незважаючи на широке використання, SLIC має кілька недоліків. Для зближення центроїдів потрібно кілька ітерацій. Він використовує карту відстані такого ж розміру, як і кількість вхідних пікселів, що призводить до значного споживання пам’яті для стеків зображень або обсягів відео. Нарешті, SLIC забезпечує підключення лише як етап постобробки.

Спочатку представляємо вдосконалену версію алгоритму SLIC, яка долає вищезазначені обмеження: наш алгоритм виконується за одну ітерацію він не використовує карту відстані і тому потребує менше пам’яті і наш алгоритм явно забезпечує підключення з самого початку. Крім того, покращується наш алгоритм обчислювальна ефективність, споживання пам’яті, і якість

сегментації. Оскільки наш алгоритм є неітераційною версією алгоритму SLIC, ми називаємо його простою неітеративною кластеризацією (SNIC).

По-друге, ми пропонуємо алгоритм полігональної сегментації під назвою SNICPOLY, який використовує суперпіксельну сегментацію SNIC як основу. Було показано , що полігональна сегментація зображень особливо підходить для програм, які мають справу із зображеннями, що містять геометричні або створені людиною структури . Приклад сегментації SNIC і полігонального розбиття SNICPOLY представлено

Ми порівнюємо SNIC із найсучаснішим у сфері суперпіксельної сегментації , а полігональне поділ – із відповідним сучасним розвитком . В обох випадках наші алгоритми працюють краще як кількісно, так і з точки зору ефективності обчислень.

Решта паперу така. Ми розглядаємо інші алгоритми сегментації , описуємо алгоритм SNIC і пояснюємо, як отримати полігональне поділ зображення за допомогою SNIC. Ми порівнюємо SNIC і нашу полігональну сегментацію з існуючими алгоритмами, і завершіть роботу.

2.2 Фон

У цьому розділі ми розглядаємо SLIC та пов’язані з ним алгоритми. Для повноти ми також розглядаємо інші сучасні методи. SLIC

Простий лінійний ітеративний алгоритм кластеризації або SLIC, є одним із найвідоміших алгоритмів сегментації суперпікселів [17].

Своїм успіхом як алгоритм попередньої обробки він завдячує своїй простоті, обчислювальній ефективності та здатності генерувати суперпікселі, які задовольняють вимогам гарного дотримання меж і обмеженої суміжності.

SLIC виконує локалізовану оптимізацію *k*-середніх у п’ятивимірному просторі кольорів і зображень CIELAB для кластеризації пікселів, починаючи з початкових значень, вибраних на регулярній сітці.

На практиці SLIC має лише два вхідні параметри – необхідну кількість суперпікселів і коефіцієнт компактності, який визначає, наскільки компактними є суперпікселі. Автори SLIC також представляють версію, яка не потребує введення параметра компактності, оскільки він автоматично встановлює його значення на максимальну колірну відстань у межах суперпікселя.

Завдяки його широкому використанню були представлені варіанти SLIC. Лі та Чен представляють лінійну спектральну кластеризацію (LSC), яка проектує п’ятивимірний простір просторових і колірних координат у десятивимірний простір перед виконання кластеризації *k*-середніх.

Лю та ін. з іншого боку, представляє Manifold-SLIC (MSLIC), який проектує той самий п’ятивимірний простір у двовимірний простір перед виконанням кластеризації. Автори демонструють кількісні покращення якості сегментації, але основні обмеження алгоритму SLIC, такі як численні ітерації та відсутність явного підключення, залишаються незмінними.

2.3 Алгоритм на основі графів

Деякі з інших відомих підходів до сегментації зображень покладаються на піксельні графіки. Алгоритм нормалізованих розрізів (NCUTS) створює суперпікселі шляхом рекурсивного обчислення нормалізованих розрізів для піксельного графіка [18-21].

Felzenszwalb і Huttenlocher пропонують підхід до сегментації на основі мінімального остовного дерева (MST). Їх алгоритм поступово об’єднує компоненти, доки не буде виконано критерій зупинки, який запобігає охопленню остовним деревом усього зображення.

Мур та ін. генерують суперпікселі (SLAT), знаходячи найкоротші шляхи, які розділяють зображення на вертикальні та горизонтальні смуги.

Подібним чином Zhang et al. створюють суперпікселі (SPBO), застосовуючи горизонтальні та вертикальні розрізи графіків до смуг зображення, що перекриваються.

Замість того, щоб знайти розрізи на зображенні, генерують суперпікселі шляхом зшивання фрагментів зображення, що перекриваються. Нещодавно представляють інший підхід на основі графів (ERS), який з'єднує підграфи шляхом максимізації швидкості ентропії випадкового блукання.

2.4 Алгоритм без графів

Є кілька інших алгоритмів, які не засновані на графах. Алгоритм вододілу (WSHED) накопичує подібні пікселі, починаючи з локальних мінімумів, щоб знайти вододіли, тобто лінії, які розділяють сегменти [22-26].

Алгоритм середнього зсуву (MSHIFT) ітераційно знаходить локальні максимуми функції щільності в просторі площини кольору та зображення. Пікселі, які ведуть до того самого локального максимуму, належать одному сегменту. Швидкий зсув (QSHIFT) створює суперпікселі шляхом пошуку локальних максимумів, як MSHIFT, але є більш ефективним з точки зору обчислень.

Алгоритм Turbopixels (TURBO) генерує суперпікселі шляхом прогресивного розширення піксельних початкових значень, розташованих у регулярних центрах сітки, використовуючи підхід набору рівнів.

SEEDS генерує суперпікселі шляхом ітераційного вдосконалення початкового прямокутного наближення суперпікселів, використовуючи грубий обмін пікселями на дрібний із сусідніми суперпікселями.

2.5 Анотації

MST, MSHIFT і WSHED є традиційними алгоритмами сегментації, вони не націлені на отримання компактних сегментів однакового розміру. Решта вважаються суперпіксельними алгоритмами – вони спрямовані на створення кластерів сегментів, які мають однакові розміри та невелику кількість суміжних

сегментів. З них NCUTS, SLAT, TURBO, SLIC і SPBO демонструють більшу компактність, тобто більше співвідношення площі суперпікселя до його периметра. MST, SLIC, SEEDS і LSC є найшвидшими в обчисленні.

TURBO, SLIC, ERS і SEEDS дозволяє користувачеві контролювати кількість вихідних сегментів. Ця остання властивість суперпікселів стала важливою, оскільки вона дозволяє користувачеві вибирати розмір суперпікселів на основі потреб програми.

2.6 Багатокутне розбиття

З невеликим відхиленням від теми суперпікселів Дуан і Лафарж представляють метод (CONPOLY), який розбиває зображення на опуклі багатокутники однакового розміру замість створення суперпікселів довільної форми.

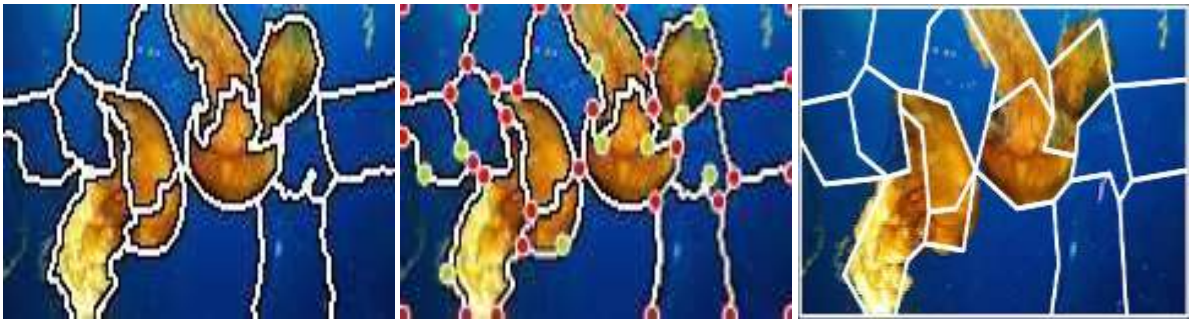
Таке розділення знаходить застосування в таких програмах, як реконструкція поверхні і локалізація об’єктів. Автори CONPOLY виявляють попередні відрізки ліній за допомогою детектора відрізків ліній.

Вони будують тесселяцію Вороного, яка відповідає цим відрізкам ліній, а потім гомогенізують згенеровані багатокутники додатковими розділами. Отриманий алгоритм є обчислювально ефективним і демонструє хороші властивості дотримання меж.

У роботі ми описуємо метод виконання полігонального поділу зображень, використовуючи сегментацію SNIC як вихідну точку. Наш метод полігонального поділу SNICPOLY перевершує CONPOLY за стандартними тестами, а також за ефективністю обчислень.

Полігональне розділення, яке ми виконуємо, спирається на межі, згенеровані суперпіксельною сегментацією SNIC. Кожен суперпіксель призводить до одного багатокутника. Багатокутники створюються з урахуванням того, щоб суміжні суперпікселі мали спільні краї багатокутника.

Щоб створити багатокутники з початкової сегментації, ми виконуємо наступні кроки (рис. 2.2).



(а) Сегментація SNIC (б) Вибір вершин (в) Розбиття багатокутників

Рисунок 2.2 – Візуальне пояснення етапів формування полігону

На рисунку 2.2 (а) Початкова сегментація за допомогою SNIC.

На рисунку 2.2 (б) початкові вершини, виділені червоним, вибираються як пікселі, які торкаються принаймні трьох різних сегментів, принаймні двох сегментів і меж зображення, або є кутами зображення.

Додаткові вершини, виділені зеленим кольором, отримані за допомогою алгоритму Дугласа-Пюкера.

На рисунку 2.2 (в) після об’єднання надто близьких вершин ми отримуємо багатокутники, з’єднуючи решту вершин відрізками.

Трасування контурів: замкнутий шлях уздовж межі кожного суперпікселя трасується за допомогою стандартного алгоритму трасування контурів . Це генерує впорядковану послідовність позицій пікселів уздовж меж суперпікселя.

Початкові вершини: оскільки суміжні суперпікселі мають спільні межі, вибираються деякі спільні вершини. Усі позиції пікселів уздовж граничних шляхів, які торкаються принаймні трьох суперпікселів або принаймні двох суперпікселів і меж зображення, приймаються як початкові спільні вершини (рис. 2.2 (б)). Крім того, кути зображення вважаються вершинами.

Додаткові вершини: тепер ми спрощуємо відрізок шляху між двома вершинами.

Для кожного сегмента шляху між двома вершинами ми додаємо нові вершини за допомогою алгоритму Дугласа-Пюкера. Це спрощує сегмент шляху від кількох позицій пікселів до кількох вершин полігону.

Об’єднання вершин: залежно від розміру суперпікселя, вершинам, які вважаються дуже близькими одна до одної відповідно до порогу (одна десята очікуваного радіуса суперпікселя), призначається спільна вершина. Ця загальна вершина вибирається як вершина з найбільшою величиною градієнта зображення.

Нарешті, багатокутники отримують шляхом з’єднання вершин, отриманих до цього моменту, прямими відрізками (рис. 2.2 (в)).

Після створення багатокутників ми переназначаємо пікселі на основі полігональних меж. Весь процес створення полігонів і призначення нових міток займає лише 20% більше часу, ніж початкова сегментація SNIC. Це робить процес полігонального розбиття SNICPOLY швидшим

CONPOLY. Як зауваження, хоча ми покладаємося на суперпікселі SNIC, алгоритм полігонального розподілу, представлений у цьому розділі, також може генерувати полігони для суперпікселів, отриманих за допомогою іншого алгоритму. Однак, на відміну від CONPOLY, деякі з наших полігонів можуть бути не- опуклі, особливо для природних зображень. Якщо для програми потрібні опуклі багатокутники або трикутники, можна додати ребра всередину неопуклих багатокутників, щоб зробити їх опуклими. Ми порівнюємо SNIC1 з SLIC , а також кілька найсучасніших суперпіксельних методів: NCUTS , MST , TURBO , SEEDS , ERS і LSC .

Використовувати доступні онлайн реалізації для всіх методів. Одночасно ми порівнюємо SNICPOLY із найсучаснішим CONPOLY.

Порівняльний аналіз виконується на наборі даних Berkeley 300, беручи до уваги як кольорові, так і градаційні зображення землі для діапазону від 50 до 2000 суперпікселів. Кількісні порівняння можна переглянути. Візуальне порівняння SNIC та SNICPOLY із сучасними наведено. Помилка недостатньої сегментації (рис. 2.3).

Помилка недостатньої сегментації вимірює помилку перекриття, яка виникає, коли суперпіксель порівнюється з базовим сегментом істинності, що займає те саме місце. Нойберт і Протцель помітили, що обчислення помилки недостатньої сегментації, представлене в TURBO і SLIC, покарає перекривання з обох сторін, коли суперпіксель перетинає межу правдивості землі.

Вони пропонують виправлену помилку недостатньої сегментації (CUSE), яка виправляє цю помилку(рис. 2.4).

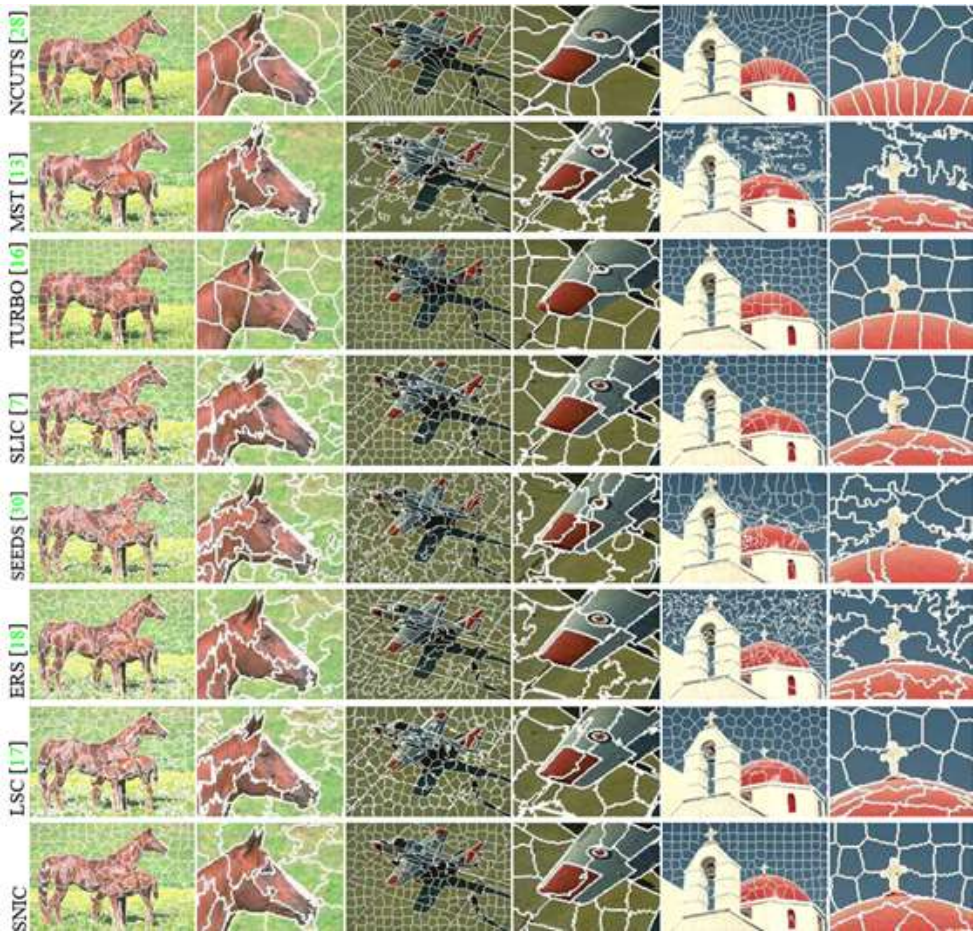


Рисунок 2.3 – Суперпіксельна сегментація

Рисунок 2.4 – Багатокутне розбиття

Візуальне порівняння суперпікселів SNIC і полігонального поділу з найсучаснішими алгоритмами. Зверніть увагу на збільшені області, як SNIC добре адаптується до кожної області зображення відповідно до її локальної структури. Результати SNIC генеруються з m = 10. Примітка: межі полігонів CONPOLY і SNICPOLY виглядають зміщеними, оскільки для їх малювання використовуються дискретні мітки пікселів.

Щоб оцінити продуктивність будь-якої техніки виявлення, два значення запам’ятовування та точності розглядаються разом. Пригадування – це відношення справжніх позитивних результатів до суми справжніх позитивних і хибно-негативних результатів, тоді як точність – це відношення справжніх позитивних результатів до суми справжніх і помилкових позитивних результатів. Пригадування або точність, розглянуті самі по собі, можуть ввести в оману, оскільки можливо мати дуже високе пригадування з надзвичайно низькою точністю, і навпаки.

Суперпіксельна та наземна граничні карти тих самих розмірів, що й вхідне зображення. У цих картах значення в позиції пікселя xi дорівнює 1, якщо це граничний піксель, і 0 в іншому випадку. Обчислюємо кількість справжніх позитивних результатів як:

$$TP = \frac{X}{i=1} 1 j \in N(i)(B^G[i] \longrightarrow B^S[j]),$$

де N – це окіл навколо позиції пікселя . Функція 1 повертає 1, якщо границя суперпікселя перекривається з основним граничним пікселем істинності в околиці N і 0 в іншому випадку. Використовуємо для наших оцінок як найсучасніших.

Помилкові спрацьовування – це кількість граничних пікселів суперпікселя в околиці, які не є справжніми спрацьовуваннями, тобто не мають пікселя основного істинного поряд. Ми обчислюємо його значення як:

$$TP = \frac{X}{i=1} 1 - 1 j \in N(i)(B^G[i] \longrightarrow B^S[j]).$$

Сума справжніх позитивних і помилкових негативних результатів є кількістю всіх граничних пікселів, тобто

$$TP + FN = \frac{X}{i=1}(B^G[i]),$$

дотримуючись визначення BG[i].

На рисунках 2.5 – 2.7 показуємо графіки залежності точності від запам’ятовування (центральний графік) і *F*-міри від кількості суперпікселів (нижній графік).

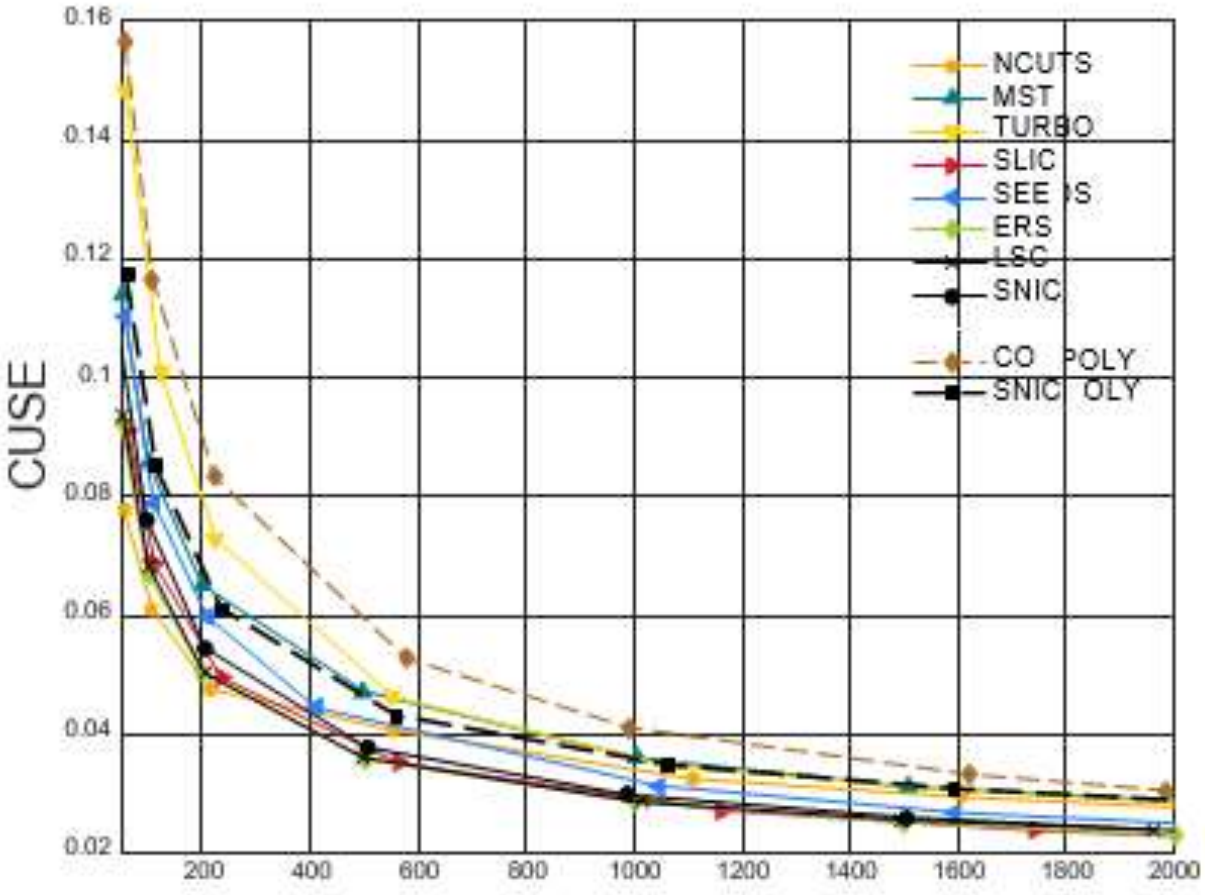


Рисунок 2.5 – Кількість суперпікселів

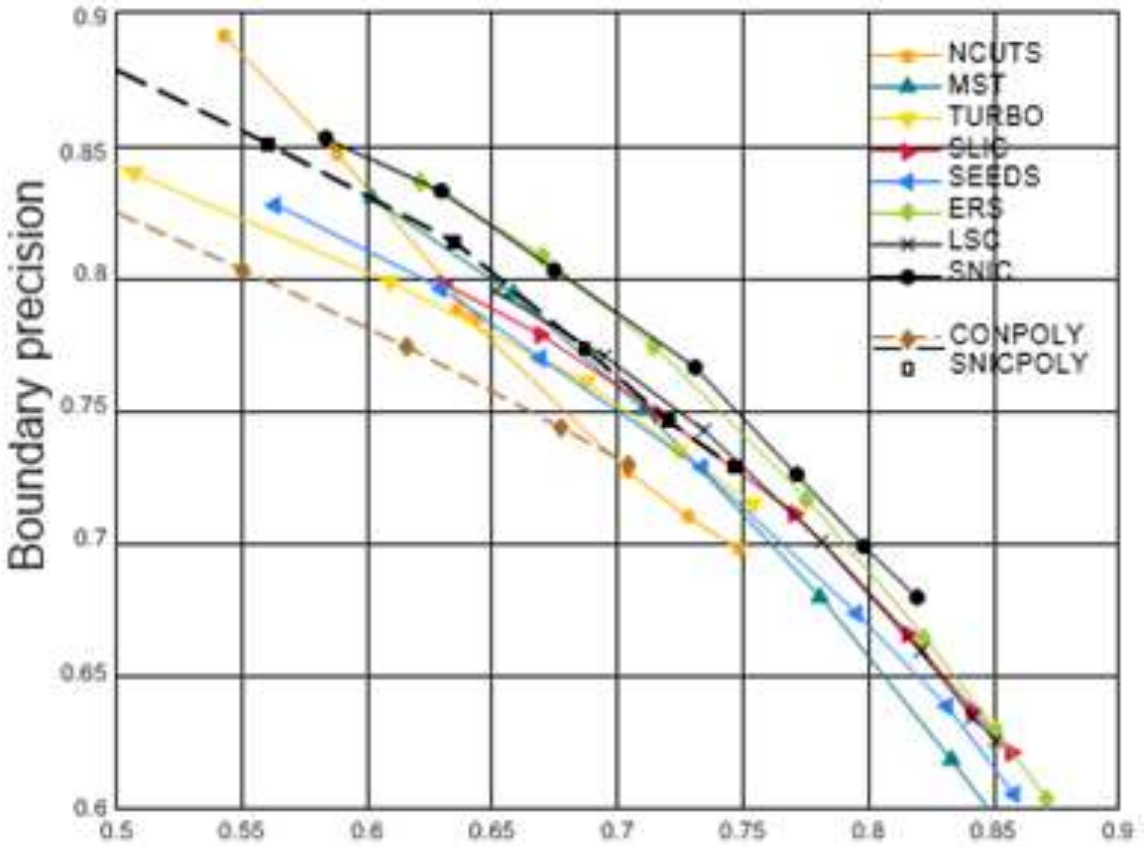


Рисунок 2.6 – Відкликання кордону

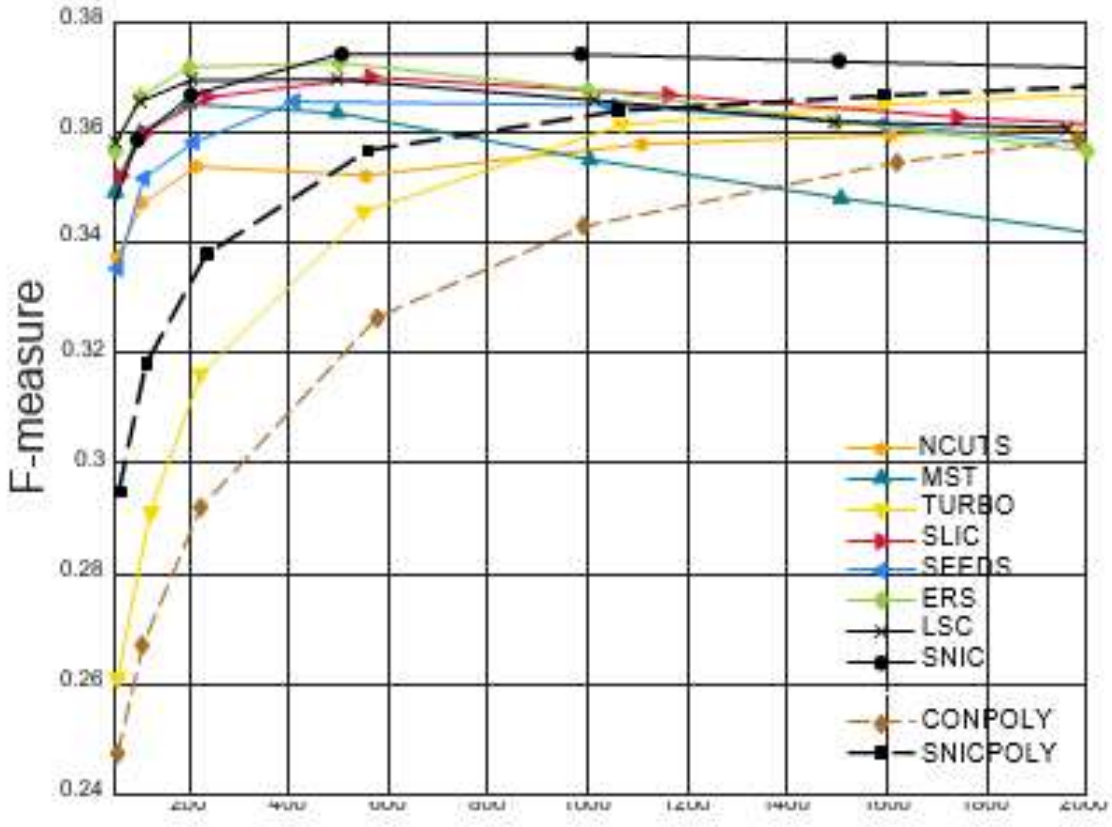


Рисунок 2.7 – Кількість суперпікселів

Порівняння SNIC і SNICPOLY з сучасними методами. Помилка недостатньої сегментації (CUSE) SNIC є однією з найнижчих серед усіх алгоритмів. SNIC показує кращу точність порівняно з відкликанням, а також кращий показник *F*, ніж найсучасніший. Так само SNICPOLY значно перевершує сучасний метод багатокутного розділення CONPOLY .

Помилка недостатньої сегментації (CUSE) SNIC є низькою та порівнюється з найкращою.

На кривій «точність-запам’ятовування» SNIC показує найкращу продуктивність, переконливо доводячи, що SNIC найкраще дотримується всіх меж об’єктів у базовій істині (високе запам’ятовування), але в той же час лише до справжніх меж об’єктів (висока точність).

Цей факт також підтверджується графіком F-міри, де SNIC явно перевершує всі інші методи порівняно з включенням SLIC. Цікаво, що незважаючи на нові алгоритми, показують, що SLIC продовжує залишатися конкурентоспроможним з точки зору якості та ефективності.

Багатокутне розділення SNICPOLY показує таку ж обнадійливу ефективність. Значення CUSE для SNICPOLY значно нижчі, ніж CONPOLY. Як на графіках точності, так і на *F*-вимірюванні, криві SNICPOLY значно кращі, ніж криві CONPOLY.

З точки зору обчислювальної ефективності, SNIC є найшвидшим алгоритмом після SEEDS. SNIC швидше, ніж SLIC і LSC, варіант SLIC, з яким ми порівнювали (рис. 2.8).

Рисунки 2.8 та 2.9. Порівняння швидкості в секундах. Верхній графік порівнює час обчислення з різною кількістю суперпікселів, усередненим для 100 зображень розміром 321×481.

Нижній графік порівнює середній час обчислення з лінійно зростаючими розмірами зображень. Обидва графіки показують, що SNIC демонструє лінійну складність на практиці, а також є найшвидшим методом після SEEDS.

Примітка: криві швидкості SNICPOLY і CONPOLY (не нанесені) лежать досить близько до кривих SNIC і SLIC відповідно.

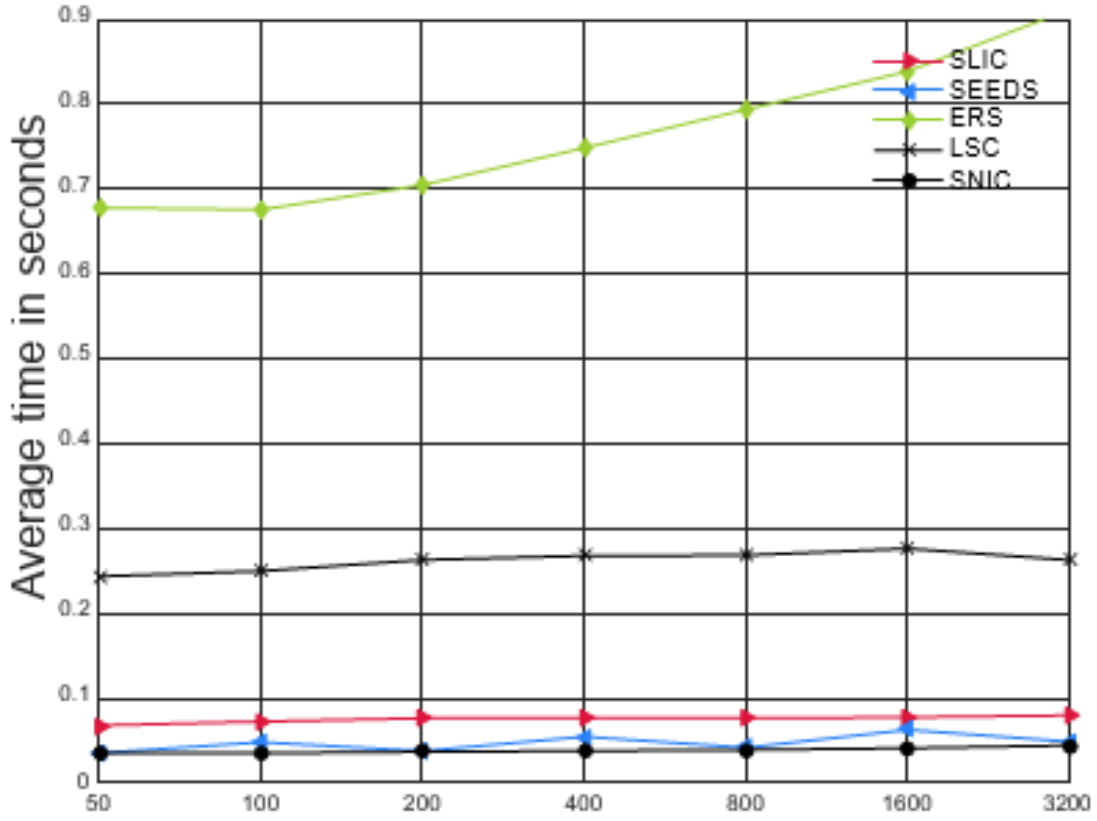


Рисунок 2.8 – Розмір зображення в пікселях

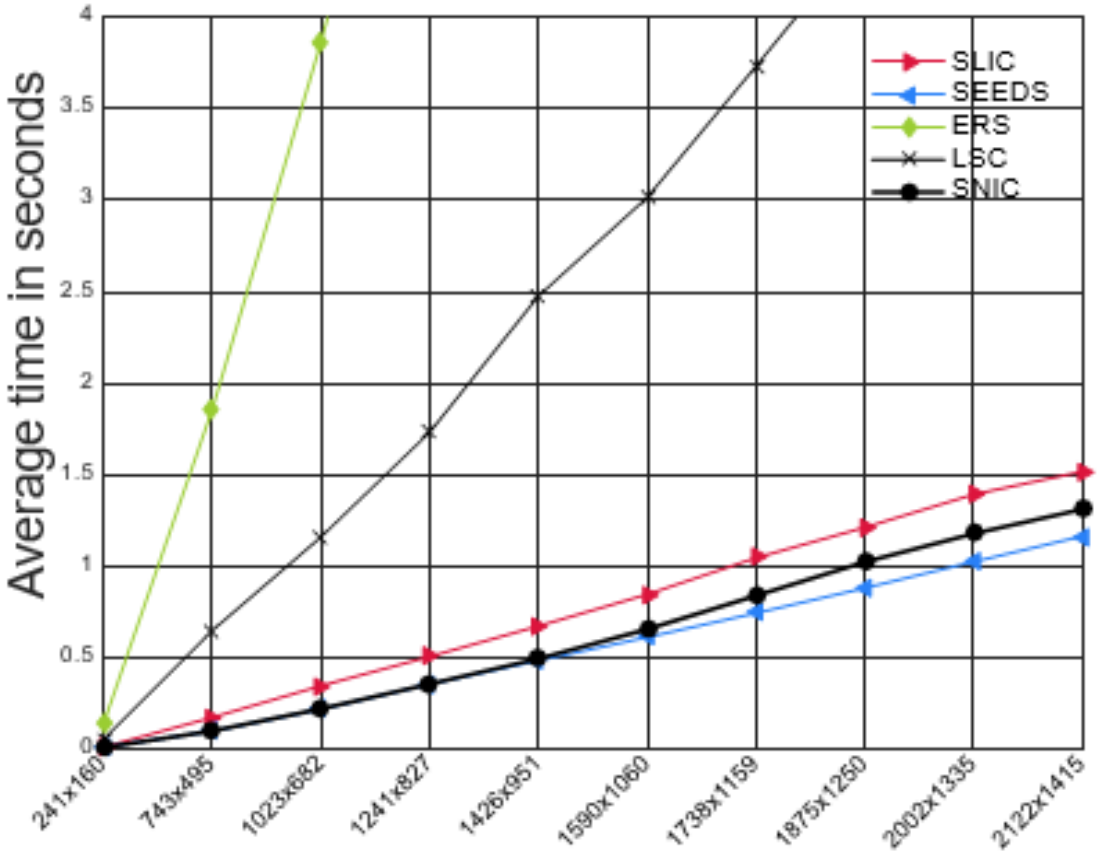


Рисунок 2.9 – Розмір зображення в пікселях

Ми представляємо SNIC, покращену версію добре відомого алгоритму сегментації суперпікселів SLIC. Наш алгоритм зберігає бажані властивості SLIC, а саме обчислювальну ефективність, простоту реалізації та використання, а також контроль над кількістю та компактністю суперпікселів.

У той же час він долає обмеження SLIC: наш алгоритм SNIC є неітераційним, явно забезпечує зв'язок, обчислювально дешевший, використовує менше пам'яті, і все ж перевершує SLIC за кількісними тестами. Отриманий алгоритм є спрощенням вихідного алгоритму SLIC.

Ми також представляємо алгоритм полігонального розбиття, який спирається на межі суперпікселів SNIC.

Подібно до SNIC, алгоритм полігонального поділу SNICPOLY також перевершує найсучасніші з точки зору якості та швидкості сегментації.

2.7 Опис Phi-функцій для опуклих багатокутників

Внутріній перетин*:* $intA \cap intB \neq \varnothing$.

Дотик*:* $intA \cap intB = \varnothing \, u \, frA \cap frB \neq \varnothing$.

Неперетин: $A \cap B = \varnothing$.

Включення: $A \subset B \Leftrightarrow intA \cap intB^* = \varnothing, \; B^* = R^2 \setminus \operatorname{int} B$.

Phi-функція Φ^{AB} визначена для пари phi-об'єктів A і B, безперервна і задовольняє наступним властивостям: $\Phi^{AB} > 0$, $\Phi^{AB} = 0$, $\Phi^{AB} < 0$.

Phi-функція для опуклих багатокутників *K1* та *K2*.

Нехай *(x1i, y1i), i = 1 ,.., n, (x2j, y2j), j = 1,.., m* – вершини багатокутників *K1(0)* та *K2(0),* задані проти годинникової стрілки $v_1 = (x_1, y_1)$ – вектор трансляції *K1* $v_2 = (x_2, y_2)$ – вектор трансляції *K2.*

Може бути визначена як

$$\Phi(v1, v2) = h\,(x2 - x1, y2 - y1),$$

де

$$h(x,y)=\max\left\{\max_{i=1,..n} h_{1i}(x,y),\max_{j=1,..m} h_{2j}(x,y)\right\}=\max_{l=1,..\sigma} h_l(x,y),\ \sigma\le m+n,$$

*h1i(x, y) = A1ix – B1iy + C*1i =0 де C*1i = min{g1i(x2j, y2j), j = 1,..,m},*

*h2j(x, y) = –A2jx + B2jy + C*2j =0 де C*2j = min{g2j(x1i, y1i), i = 1,..,n},*

g1i(x, y) = A1ix – B1iy + C1i,

g2j(x, y) = A2jx – B2jy + C2j,

A1i = y1i+1 – y1i, B1i = x1i+1 – x1i,

C1i = –A1ix1i + B1iy1i ≡y1ix1i+1 – x1i,y1i+1,

A2j= y2j+1 – y2j, B2j = x2j+1 – x2j,

C2i = –A2jx2j + B2jy2j≡y2jx2j+1 – x2j,y2j+1.

Тут *g1i(x, y) = 0, g2j(x, y) = 0* – рівняння прямих, які проходять через сторони багатокутників *K1(v1)* та *K2(v2)* відповідно. Рівняння побудовано таким чином, якщо *g(x, y) < 0* точка *(x, y)* лежить з тієї ж сторони, що і багатокутник.

Phi-функція для phi-многокутників *K1* і *K2*.

Нехай *A* і *B* – опуклі многокутники. Позначимо $(x_i^{'}, y_i^{'})$, $i=1,2,...,m_1$ вершини *A*. І визначається , $\varphi_i=\alpha_i^{'}x+\beta_i^{'}y+\gamma_i^{'}\le 0$, $i=1,2,...,m_1$. Позначимо також через, $(x_i^{''}, y_i^{''})$, $i=1,2,...,m_2$, вершини B. І визначається, $\psi_j=\alpha_j^{''}x+\beta_j^{''}y+\gamma_j^{''}\le 0$, $j=1,2,...,m_2$.

Для пари «опуклий багатокутник-опуклий багатокутник» безрадикальна phi-функція Φ^{AB} задана у формі

$$\Phi^{AB} = \max\{\max_{1\le i\le m_1}\min_{1\le j\le m_2}\varphi_{ij}, \max_{1\le j\le m_2}\min_{1\le i\le m_1}\psi_{ji}\},$$

де

$$\varphi_{ij} = \alpha_i^{'}x_j^{''} + \beta_i^{'}y_j^{''} + \gamma_i^{'},\ \psi_{ji} = \alpha_j^{''}x_i^{'} + \beta_j^{''}y_i^{'} + \gamma_j^{''},\ i = 1,2,...,m_1,\quad j = 1,2,...,m_2.$$

3 РЕАЛІЗАЦІЯ ТА ЕКСПЕРЕМЕНТИ

3.1 Результати експериментальних досліджень

Підкреслимо, що суперпіксельна сегментація за своєю природою є надмірною (oversegmentation). Враховуючи її стійкість за рахунок однорідності суперпікселів, практично всі алгоритми поряд з пошуком валідних результатів можуть використовуватися для порівняння результатів розбиття або покриття поля зору [27-33].

Однак саме з цієї причини (перевизначеності) необхідно враховувати обчислювальну складність. У цьому плані слід орієнтуватися як і стільки кількість областей, а й у їх форму.

Переважно мати опуклі сегменти, оскільки їх простіше і точніше представляти опуклими багатокутниками і далі за допомогою phi-функцій обчислювати відмінності. Всі експерименти проводилися з зображеннями з роздільною здатністю пікселів.

Специфіка використаного цветотекстурного алгоритму просторової сегментації полягає в наступному. Він є емпіричним розширенням методу Отсу і заснований на максимізації поділу областей та концентрації внутрішніх елементів, що базується на положеннях двофакторного дисперсійного аналізу.

Для побудови розбиття поля зору D на K кластерів r_k, $k = \overline{1, K}$ з кількістю елементів у кожному $\mathrm{card}\, \mathrm{r_k}$,, тобто. отримання областей таких, що

$$\mathrm{D} = \bigcup_{\mathrm{k}=1}^{K} \mathrm{r_k},$$

$$\forall \mathrm{k}', \mathrm{k}'' \in \{1,2,...,K\} : \mathrm{k}' \neq \mathrm{k}'' \Rightarrow \mathrm{r_{k'}} \bigcap \mathrm{r_{k''}} = \varnothing,$$

користуючись відомими положеннями двофакторного дисперсійного аналізу, вважаючи $\mathrm{z} \in \square^{2}$, розглянемо середнє в кожній області

$$m_k = \frac{1}{\text{card } r_k} \sum_{z \in r_k} B(z)$$

та загальне середнє

$$m = \frac{1}{\sum_k \text{card } r_k} \sum_{z \in D} B(z).$$

Тоді, визначаючи «внутрішньокластерну» і «міжкластерну» дисперсії відповідно

$$d' = \sum_{k=1}^{K} \sum_{z \in r_k} \left\| B(z) - m_k \right\|^2,$$

$$d' = \sum_{k=1}^{K} \sum_{z \in r_k} \left\| B(z) - m_k \right\|^2,$$

де $\left\| \circ \right\|^2$ – евклідова норма, отримуємо критерій

$$C = (d'' - d') / d'.$$

Неважко бачити: що більше значення С, то краще області розділені, а внутрішні елементи «концентровані» і навпаки. Таким чином, перераховуючи С по кожній сегментованій області та вводячи позначення [34].

$$C^* = \frac{1}{n} \sum_{k=1}^{K} C_k \text{ card } r_k,$$

отримуємо наступний критерій: знайти мінімум C^* по всіх можливих розбивках для фіксованого числа областей.

Дійсно, якщо сегментація виконана досить добре, то значення C для кожної області малі, а отже, і C^* прагне до нуля.

Достоїнством (і причиною вибору) цього кольорово-текстурного алгоритму є простота управління якістю просторової сегментації, що забезпечує адаптивність ознаки відеокадра. Наведено приклади сегментації, не всі з яких є прийнятними для подальшої обробки (рис. 3.1).

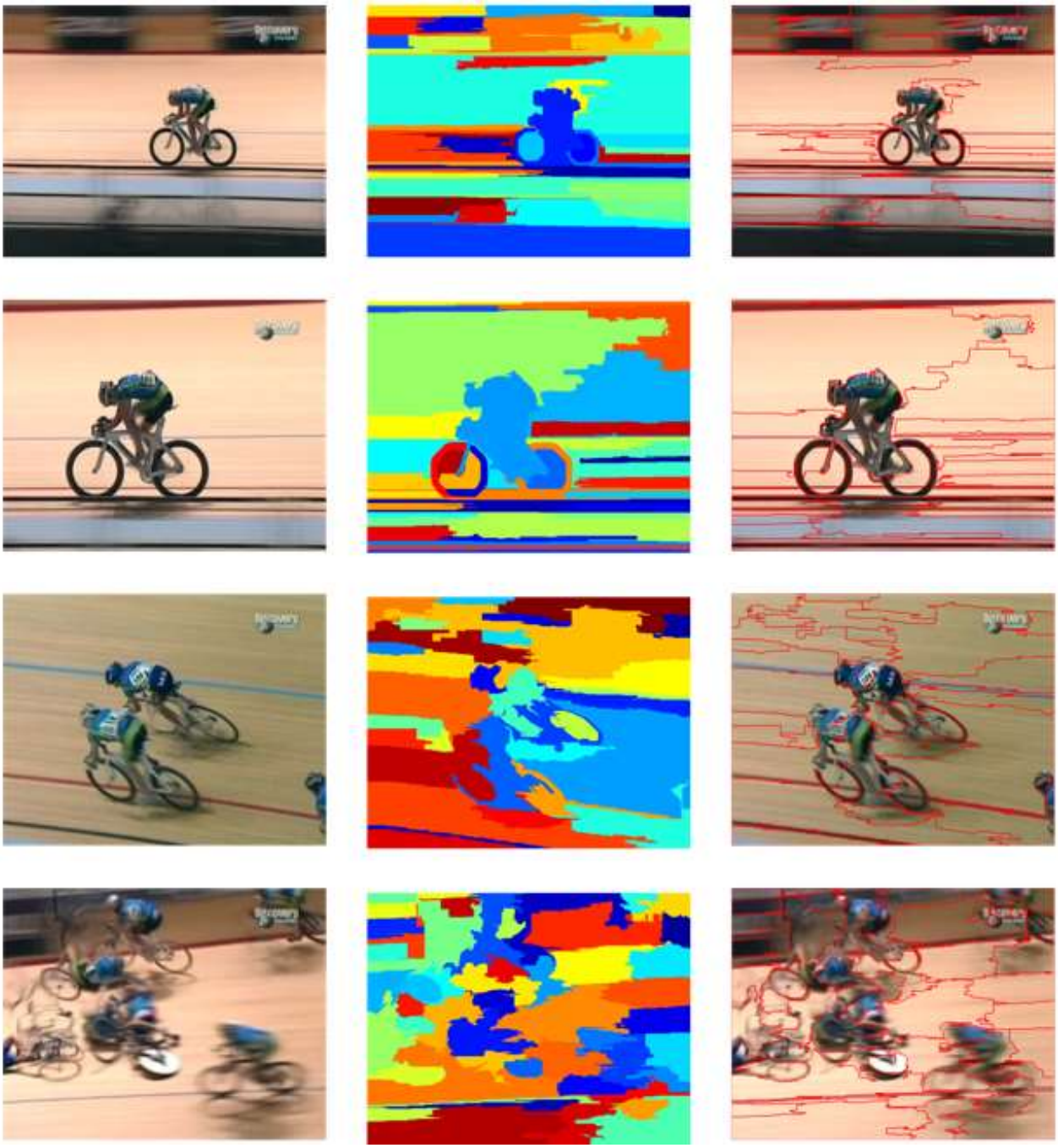


Рисунок 3.1 – Приклади сегментації зображень

Порівняємо отримані результати з результатами суперпіксельної обробки (рис. 3.2).

В експерименті використані алгоритми: кластерного типу SLIC, SLICO (Simple linear iterative clustering), топологічний Watershed, щільний QS, графовий FH (Felzenszwalb and Huttenlocher), заснований на однорідності сегментів і меж SEEDS (Superpixels Extracted via Energy (Linear Spectral Clustering Superpixel).

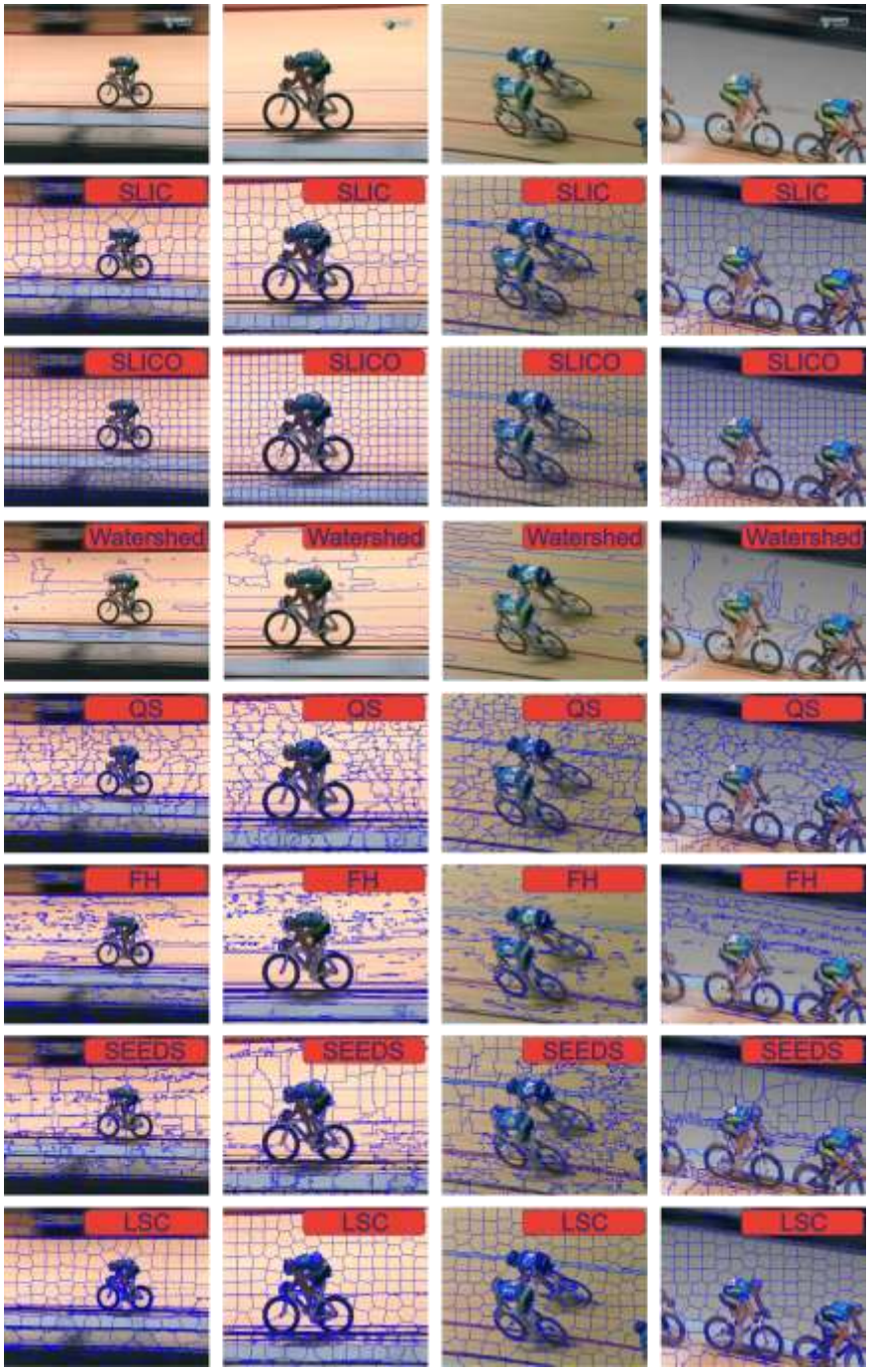


Рисунок 3.2 – Приклади суперпіксельної сегментації

На рисунку 3.2 наведено результати сегментування.

По-перше, слід підкреслити різноманітність форм суперпікселів, тобто. далеко не всі алгоритми придатні для застосування при порівнянні сегментацій з використанням опуклих полігональних апроксимацій. Більше того, всі алгоритми – параметричні, а раціональний вибір параметрів істотно пов'язаний з апріорною інформацією про «зміст» зображення.

На рисунку 3.3 наведено результати сегментації найбільш прийнятного алгоритму SLIC с різним числом передбачуваних областей при постійному значенні компактності. Простий аналіз показує, що для кожного зображення існує деяка раціональна кількість сегментів. Однак, стартуючи з явно надмірної сегментації, досить нескладно проводити адекватне метричне порівняння розбиття на основі тільки вершин апроксимуючих багатокутників.

По-друге, метричне ієрархічне об'єднання сегментів з урахуванням реальних меж об'єктів може порушувати умова опуклості. В цьому випадку (як і для інших суперпікселізації) доцільно передбачити інші варіанти полігональної апроксимації.

У цьому плані можуть представляти розбиття Вороного з опорними точками – центрами суперпікселів (наприклад, медоїдами).

Розглянемо алгоритм порівняння розбиття Вороного, що полігонально апроксимує суперпісельну сегментацію.

Нагадаємо, що якщо $\{p_1, p_2, ..., p_n\}$ – безліч опорних точок, то діаграма Вороного є розбиттям поля зору D на опуклі багатокутники $V = \{v(p_1) \bigcap D, v(p_2) \bigcap D, \ldots, v(p_n) \bigcap D\}$ таким чином, що для кожної області Вороного виконується наступна нерівність:

$$v(p_i) = \{z \in R^2 : d(z, p_i) \leq d(z, p_j)\ \forall i \neq j\},$$

де $d(\circ, \circ)$ – Евклідова метрика.

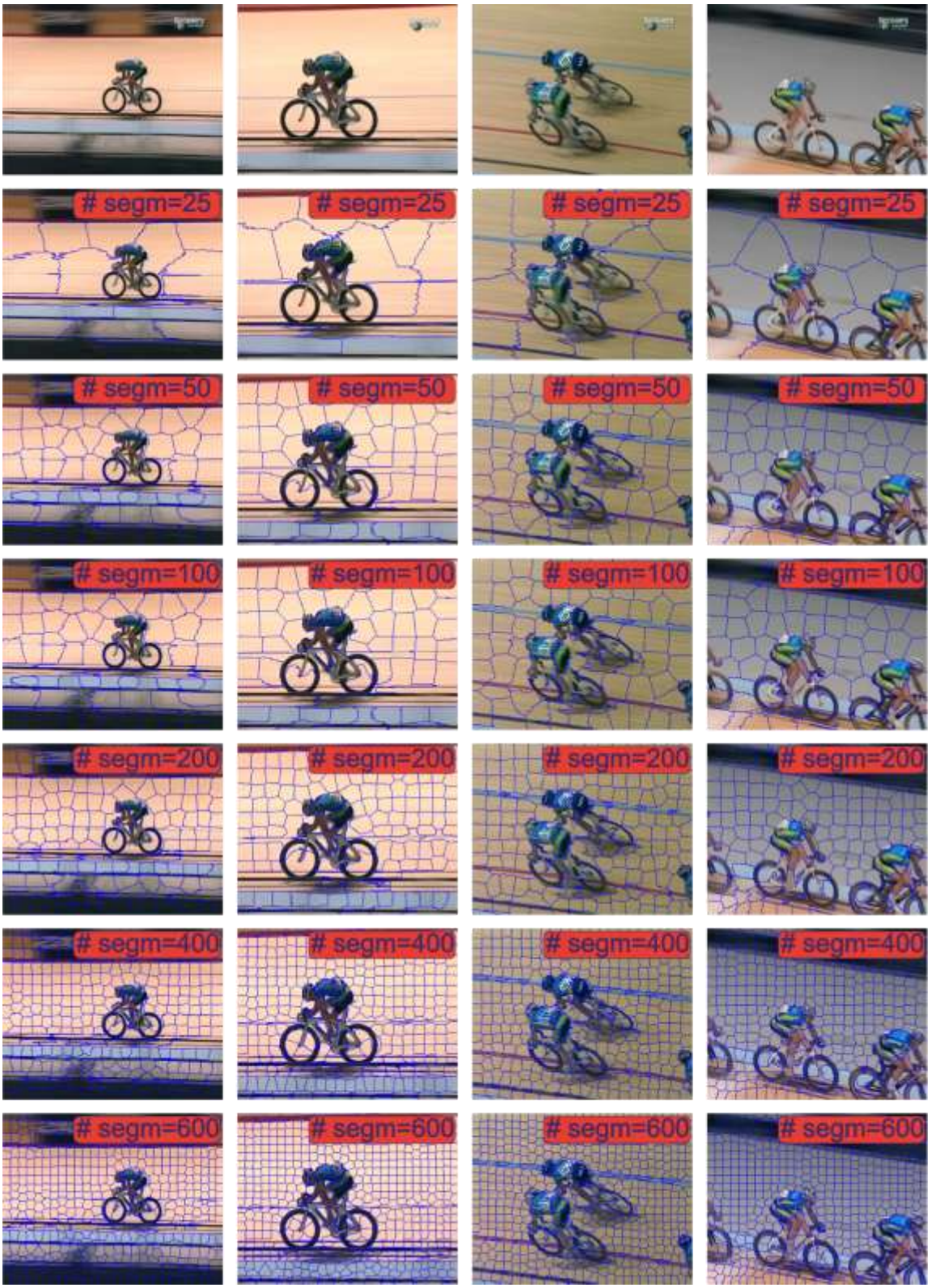


Рисунок 3.3 – Приклади SLIC- сегментації з варіаціями кількості областей

Інакше кажучи, область Вороного $v(p_i)$, пов'язана з опорною точкою p_i, є набір точок *Z*, відстань кожної з яких до неї менше або дорівнює відстані до будь-якої іншої опорної точки.

Виходячи з того, що області Вороного будуються по серединних перпендикулярах між суміжними опорними точками p_i і p_λ, запишемо область Вороного $v(p_i)$ як перетин напівплощин, що проходять через суміжні опорні точки:

$$v(p_i) = \bigcap_{\lambda \in [1;\psi]} H(p_i, p_\lambda),$$

де ψ – кількість суміжних з p_i опорних точок p_λ.

Для суміжних опорних точок p_i і p_λ виконується таке:

– $\exists \gamma_\psi, d(p_i, \gamma_\psi) = d(p_\lambda, \gamma_\psi)$, де γ_ψ – вершина багатокутника Вороного;

– $(\frac{x_\lambda - x_i}{2}; \frac{y_\lambda - y_i}{2}) \in v(p_i), v(p_\lambda)$, де $p_i(x_i; y_i)$, $p_\lambda(x_\lambda; y_\lambda)$;

– $v(p_i) \bigcap v(p_\lambda) \neq \varnothing$, де $v(p_i) \bigcap v(p_\lambda)$ – ребро багатокутника Вороного чи його вершина.

Опорні точки є суміжними тільки в тому випадку, коли відповідні їм області мають невироджений кордон (ребро області Вороного).

Наочний приклад, що ілюструє, коли в результаті перетину прямих, що містить серединний перпендикуляр, виходить точка і ребро можна побачити на рисунку 3.4.

Щоб уявити подібність кадрів через опорні точки, а чи не через області Вороного, досить записати рівняння для серединних перпендикулярів між суміжними опорними точками p'_i та p'_λ однієї кадру $B'(z)$ і суміжними опорними точками p''_j та p''_λ іншого кадру $B''(z)$.

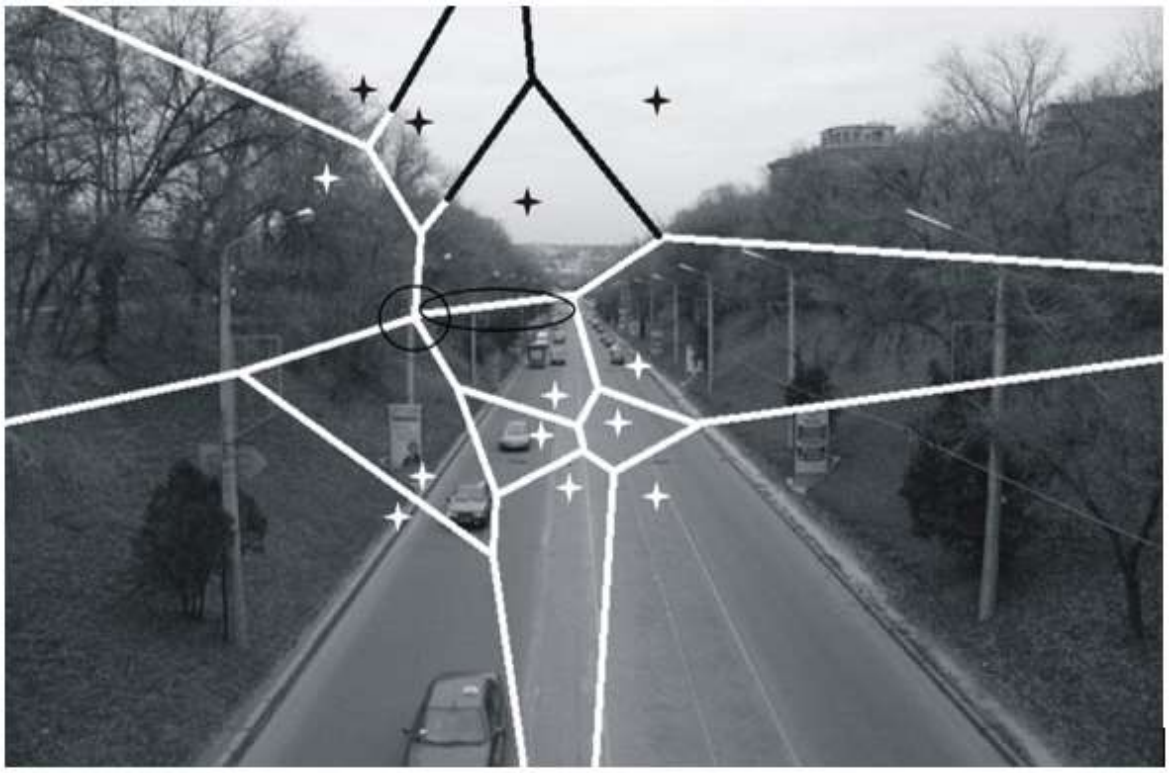


Рисунок 3.4 – Точка и ребро , що утворюються в результаті перетину прямих, що містять серединний перпендикуляр між суміжники опорними точками

Рівняння прямої, що проходить через дві точки з координатами $p_i'(x_i; y_i)$ і $p_\lambda'(x_\lambda, y_\lambda)$, виглядає наступним чином:

$$\frac{y - y_i}{y_\lambda - y_i} = \frac{x - x_i}{x_\lambda - x_i}$$

або

$$(y_i - y_\lambda)x + (x_\lambda - x_i)y + (x_i y_\lambda - x_\lambda y_i) = 0\,.$$

Звідси можна визначити кут нахилу θ для прямої, що проходить через суміжні опорні точки:

$$\text{tg}\theta = -\frac{y_i - y_\lambda}{x_\lambda - x_i}.$$

З рисунку 3.5, тому $\text{tg}\varphi = \text{tg}(\theta + \frac{\pi}{2})$ кут нахилу для серединного перпендикуляра можна виразити як $\text{tg}\varphi = -\text{ctg}\theta$.

Таким чином,

$$\text{tg}\varphi = \frac{x_\lambda - x_i}{y_i - y_\lambda}.$$

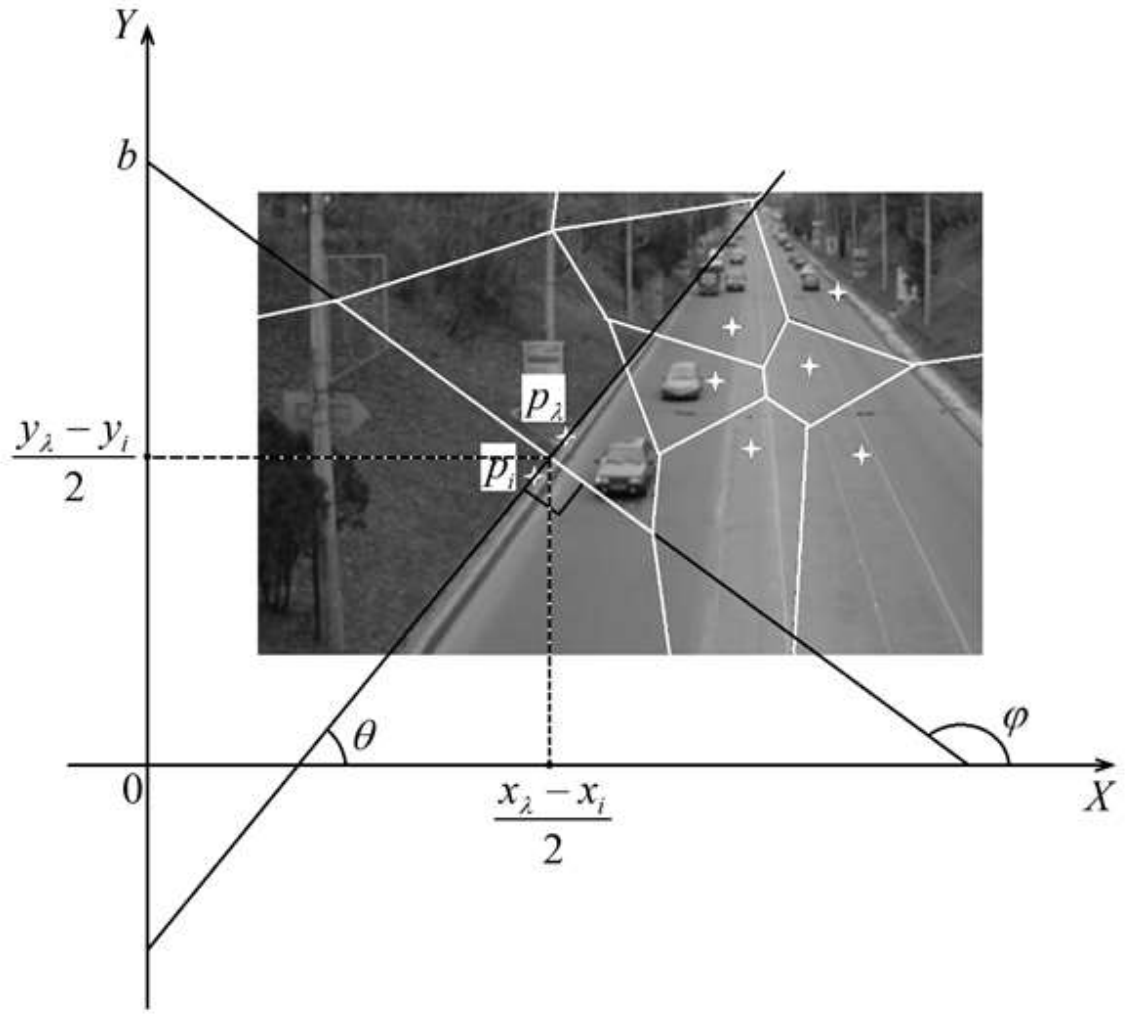


Рисунок 3.5 – Кут нахилу прямий, що містить серединний перпендикуляр між суміжними опорними точками

Знаючи кут нахилу φ і координати точки $(\frac{x_\lambda - x_i}{2}; \frac{y_\lambda - y_i}{2})$, що лежить на прямій, що утворює напівплощину, нескладно визначити рівняння прямої, що містить серединний перпендикуляр між суміжними опорними точками.

З урахуванням того, що координати точки повинні задовольняти рівняння прямої з кутовим коефіцієнтом ($y = kx + b, k = \mathrm{tg}\varphi$).

Визначимо точку перетину серединного перпендикуляра з віссю ординат:

$$b = \frac{y_\lambda - y_i}{2} - \mathrm{tg}\varphi \times \frac{x_\lambda - x_i}{2}$$

тобто

$$b = \frac{y_\lambda - y_i}{2} - \frac{x_\lambda - x_i}{y_i - y_\lambda} \times \frac{x_\lambda - x_i}{2}.$$

Запишемо рівняння прямої, що містить серединний перпендикуляр між суміжними опорними точками:

$$y = \frac{x_\lambda - x_i}{y_i - y_\lambda} x + \frac{y_\lambda - y_i}{2} - \frac{x_\lambda - x_i}{y_i - y_\lambda} \times \frac{x_\lambda - x_i}{2},$$

$$y = \frac{x_\lambda - x_i}{y_i - y_\lambda} (x - \frac{x_\lambda - x_i}{2}) + \frac{y_\lambda - y_i}{2}.$$

Перетин прямих, що містять середні перпендикуляри між суміжними опорними точками, утворюють області Вороного у вигляді багатокутників. Щоб знайти координати вершин багатокутників, достатньо визначити координати точок перетину таких прямих:

$$\bigcap_{\lambda\in[1;\psi]} \frac{x'_\lambda - x'_i}{y'_i - y'_\lambda}\left(x - \frac{x'_\lambda - x'_i}{2}\right) + \frac{y'_\lambda - y'_i}{2}.$$

Знаючи координати опорних точок $(x_i; y_i)$ і координати вершин багатокутників $(x_{\gamma_\psi}; y_{\gamma_\psi})$ можна за допомогою формули Герона визначити площу кожного багатокутника, розбивши їх на трикутники з вершиною в опорній точці і основою у вигляді ребер багатокутників (сторона між двома вершинами з відомими координатами) або відрізка поля зору, що обмежує кадр, координати якого також відомі.

При цьому якщо опорна точка знаходиться всередині трикутника, основою якого є відрізок поля зору, то площа такої області розраховується з урахуванням вершини і відрізка в основі, без урахування самої опорної точки (рис. 3.6).

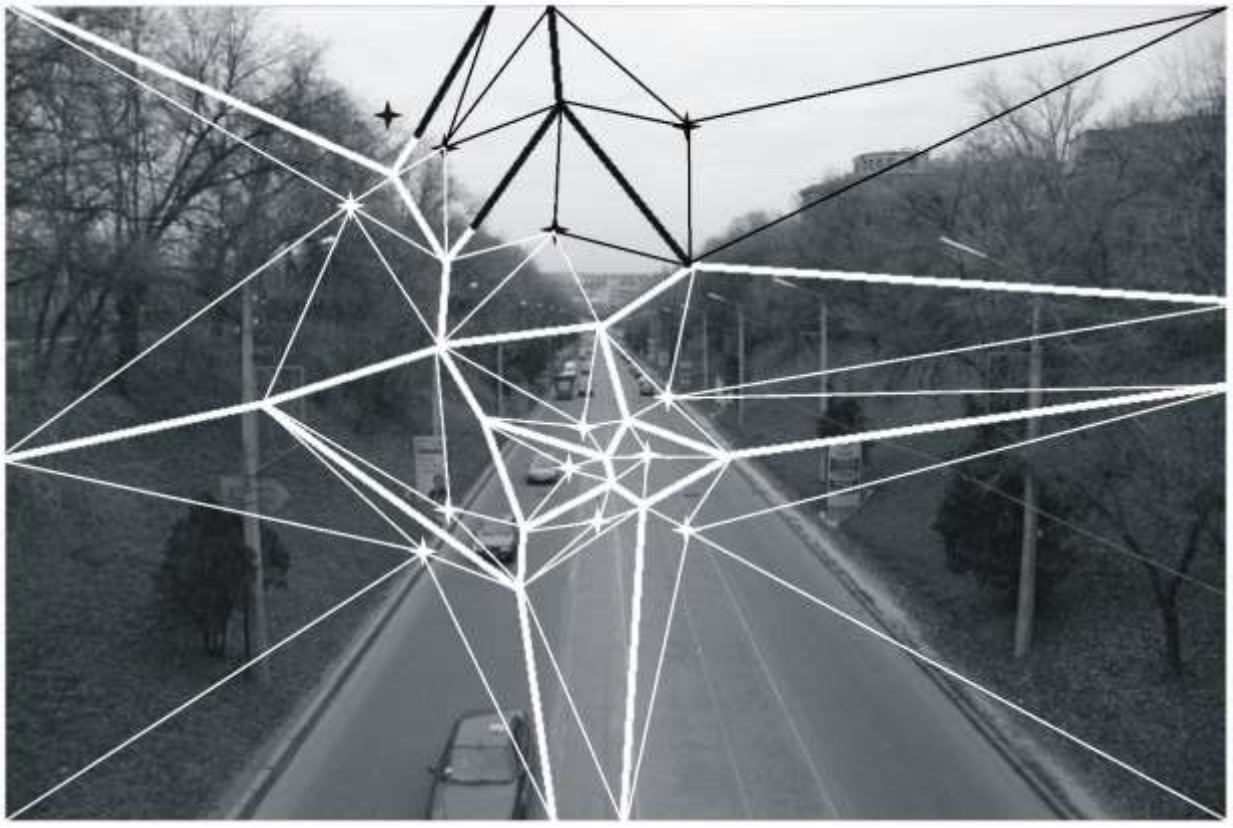


Рисунок 3.6 – Розбиття багатокутників Вороного на трикутники для визначення їх площі

Таким чином, можна розрахувати площу кожної області одного кадру $A(v(p'_i))$ та іншого $A(v(p''_i))$, щоб порівнювати розбиття.

Загальна площа всіх відеокадрів однакова $A(B'(z)) = A(B''(z))$ і визначається полем зору. Діапазон значень гранулярності розбиття (або густини областей Вороного) варіює в межах $0 < G(B'(z)) \le 1$.

Даний показник площі областей тим більше, чим менше кількість областей і чим вони більші за площею.

Площу близькість розбиття двох кадрів можна обчислити за формулою:

$$\rho_G(B'(z), B''(z)) = \left| G(v'(p_i)) - G(v''(p_j)) \right|,$$

де

$$G(v(p'_i)) = \frac{\sum_i \{A(v'(p_i))\}^2}{\{A(B'(z))\}^2}.$$

Площа близькість кадрів не враховує просторове розміщення розбиття, тому по-різному впорядковані, але в той же час однакові за площею, області матимуть загальне значення гранулярності.

Розглянемо можливі варіанти розміщення об’єкта на діаграмах Вороного у двох послідовних відеокадрах (рис. 3.7).

Об’єкт (у даному випадку це автомобілі) може бути:

– в одній області обох кадрів $B'(z)$ та $B''(z)$;

– В одній області одного кадру $B'(z)$ і в різних областях іншого кадру $B''(z)$;

– У кількох областях одного кадру $B'(z)$ і в одній області іншого кадру $B''(z)$;

– у різних областях обох кадрів $B'(z)$ та $B''(z)$.

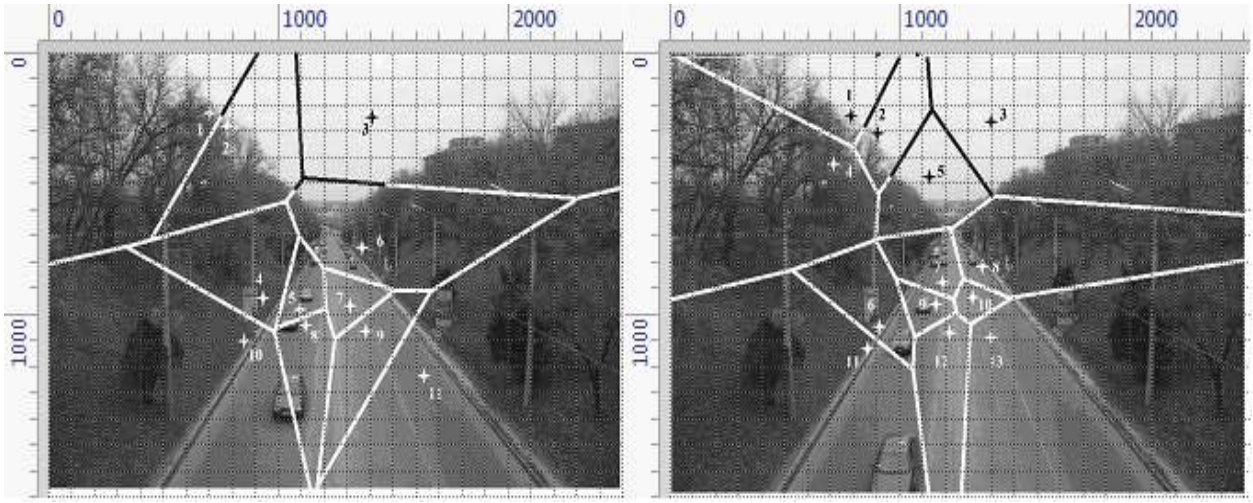


Рисунок 3.7 – Варіанти розміщення об’єктів на діаграмах Вороного

Таким чином, метрику, що дозволяє порівняти форму областей Вороного) можна остаточно уявити з використанням координат опорних точок:

$$\rho_1(V', V'') = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathrm{card}\left(H\left(\bigcap_{\lambda\in[1;\psi]} \frac{x'_\lambda - x'_i}{y'_i - y'_\lambda}\left(x - \frac{x'_\lambda - x'_i}{2}\right) + \frac{y'_\lambda - y'_i}{2} \right)\Delta \right.$$
$$\left. \Delta H\left(\bigcap_{\lambda\in[1;\psi]} \frac{x''_\lambda - x''_j}{y''_j - y''_\lambda}\left(x - \frac{x''_\lambda - x''_j}{2}\right) + \frac{y''_\lambda - y''_j}{2} \right) \right) \times$$
$$\times \mathrm{card}\left(H\left(\bigcap_{\lambda\in[1;\psi]} \frac{x'_\lambda - x'_i}{y'_i - y'_\lambda}\left(x - \frac{x'_\lambda - x'_i}{2}\right) + \frac{y'_\lambda - y'_i}{2} \right) \bigcap \right.$$
$$\left. \bigcap H\left(\bigcap_{\lambda\in[1;\psi]} \frac{x''_\lambda - x''_j}{y''_j - y''_\lambda}\left(x - \frac{x''_\lambda - x''_j}{2}\right) + \frac{y''_\lambda - y''_j}{2} \right) \right).$$

На основі викладеного можна зробити висновок, що при використанні суперпіксельної сегментації особливе місце займає полігональна апроксимація областей, що істотно знижує обчислювальну складність.

Використання багатокутників може здійснюватися у двох напрямках: безпосереднім описом областей і подання їх діаграмами Вороного.

3.2 Опис програмної реалізації

Для програмної реалізації вибрано дослідження SLIC, мову Python з додатком OpenCV. Цей алгоритм може розбивати зображення на різну кількість сегментацій. Приклад програми показано на нижче:

```
Mat image = imread("coins.jpg", CV_LOAD_IMAGE_COLOR);
// виділимо контури
Mat imageGray, imageBin;
cvtColor(image, imageGray, CV_BGR2GRAY);
threshold(imageGray, imageBin, 100, 255, THRESH_BINARY);
std::vector<std::vector<Point> > contours;
std::vector<Vec4i> hierarchy;
findContours(imageBin, contours, hierarchy, CV_RETR_TREE, CV_CHAIN_APPROX_SIMPLE);
Mat markers(image.size(), CV_32SC1);
markers = Scalar::all(0);
int compCount = 0;
for(int idx = 0; idx >= 0; idx = hierarchy[idx][0], compCount++)
{
   drawContours(markers, contours, idx, Scalar::all(compCount+1), -1, 8, hierarchy,
INT_MAX);
}
std::vector<Vec3b> colorTab(compCount);
for(int i = 0; i < compCount; i++)
{
   colorTab[i] = Vec3b(rand()&255, rand()&255, rand()&255);
}
watershed(image, markers);
Mat wshed(markers.size(), CV_8UC3);
for(int i = 0; i < markers.rows; i++)
{
   for(int j = 0; j < markers.cols; j++)
   {
       int index = markers.at<int>(i, j);
       if(index == -1)  wshed.at<Vec3b>(i, j) = Vec3b(0, 0, 0);
       else if (index == 0) wshed.at<Vec3b>(i, j) = Vec3b(255, 255, 255);
            else  wshed.at<Vec3b>(i, j) = colorTab[index - 1];
   }
}
imshow("watershed transform", wshed);
waitKey(0);
```

ВИСНОВКИ

Результатом сегментації зображення є множина сегментів, які разом покривають все зображення, або множина контурів, виділених з зображення. Всі пікселі в сегменті схожі за деякою характеристикою або за визначеною властивістю, наприклад колір, яскравість або текстура.

Сусідні сегменти істотно відрізняються за цими характеристиками.

У ході кваліфікаційної роботи було проведено аналіз переваг суперпіксельного представлення зображень, які зумовлюють використання суперпіксельної сегментації на попередніх етапах обробки зображень.

Розглянуто різні підходи до суперпіксельної сегментації зображень, що є завданням оптимізації. У ході порівняння сучасних алгоритмів суперпіксельної сегментації встановлено, що жоден з них не має явної переваги над іншими, маючи як сильні, так і слабкі сторони.

Запропоновано модифікований алгоритм суперпіксельної сегментації, що забезпечує, на відміну від відомих, можливість виділення фрагментів із різнорідною текстурою. Було проведено порівняння сегментацій, опис метрик для експериментального порівняння отриманих суперпіксельних сегментацій (разом з базовою формулою) з результатами, коли області суперпіксельних сегментацій є багатокутниками), опис алгоритмів суперпіксельних сегментацій (родина SLIC), опис алгоритмів суперпіксельних сегментацій багатокутниками, опис Phi-функцій для опуклих багатокутників та здійснено приклад схожого дослідження. На основі викладеного можна зробити висновок, що при використанні суперпіксельної сегментації особливе місце займає полігональна апроксимація областей, що істотно знижує обчислювальну складністю.

Використання багатокутників може здійснюватися у двох напрямках: безпосереднім описом областей і подання їх діаграмами Вороного.

Результати дослідження апробовано у вигляді тез доповіді під час 7th International scientific and practical conference «Innovative areas of solving problems of science and practice» [35].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Творошенко, І. С., & Табашник, В. А. (2018). Розробка просторової моделі геоінформаційної підтримки людей з обмеженими можливостями, що пересуваються на інвалідних колясках, у місті Харків.

2. Творошенко, И. С. (2004). Структура и функции интеллектуальных средств принятия решений в сложных системах. Искусственный интеллект, (4), 462−470.

3. Творошенко, И. С. (2010). Анализ процессов принятия решений в интеллектуальных системах. Системи обробки інформації, (2), 248−253.

4. Кучеренко, Е. И., & Творошенко, И. С. (2010). Прикладные аспекты моделирования нечетких процессов в сложных системах. Збірник наукових праць Харківського університету Повітряних сил, (1), 127−131.

5. Кучеренко, Є. І., & Творошенко, І. С. (2011). Оперативне оцінювання простору станів складних розподілених об’єктів з використанням нечіткої інтервальної логіки. Штучний інтелект.

6. Творошенко, І. С. (2016). Конспект лекцій з дисципліни «Технології підтримки прийняття рішень в геоінформаційних системах» для студентів 1 курсу денної форми навчання спеціальності 193–Геодезія та землеустрій спеціалізації (освітньої програми)–«Геоінформаційні системи і технології».

7. Кучеренко, Є. І., Кучеренко, В. Є., Глушенкова, І. С., Творошенко, І. С. (2012). Методи, моделі та інформаційні технології оцінювання станів складних об’єктів: монографія.

8. Кучеренко, Е. И., & Творошенко, И. С. (2003). Процессы принятия решений в сложных системах на основе нечетких интервальных представлений. Вісник Національного технічного університету” ХПІ”. Тематичний випуск: Системний аналіз, управління та інформаційні технології.–Х.: НТУ” ХПІ, 1(7), 79−86.

9. Кучеренко, Є. І., Творошенко, І. С., & Анопрієнко, Т. В. (2016). Моделювання та оцінювання станів складних об’єктів із застосуванням формальної логіки. Системи обробки інформації, (2), 76−82.

10. Кучеренко, Е. И. (2005). Интеллектуальные технологии в задачах принятия решений технологических комплексов на основе нечеткой интервальной логики. Восточно−Европейский журнал передовых технологий, (2), 92−96.

11. Мамонов, К. А., & Творошенко, І. С. (2014). Математичні методи і моделі в оцінці нерухомості: навч. посіб.

12. Кучеренко, Е. И., Корниловский, А. В., & Творошенко, И. С. (2010). О методах настройки функций принадлежности в нечетких системах. Системы управления, навигации и связи, (1), 13.

13. Бодянский, Е. В., Кучеренко, Е. И., & Творошенко, И. С. (2004). О синтезе нечетких методов на основе композиции фрагментов правил и моделей. АСУ и приборы автоматики, (128), 19−28.

14. Творошенко, І. С. (2015). Конспект лекцій з дисципліни «Цифрова обробка зображень»(для студентів 5 курсу денної та заочної форм навчання спеціальності 7.08010105–Геоінформаційні системи та технології).

15. Творошенко, І. С. Конспект лекцій з дисципліни «Цифрова обробка зображень» для студентів 4 курсу денної форми навчання напряму 6.080101–Геодезія, картографія та землеустрій.

16. Творошенко, І. С. (2018). Основи цифрової обробки зображень: конспект лекцій для студентів 4 курсу денної форми навчання напряму 6.080101–Геодезія, картографія та землеустрій.

17. Творошенко, И. С., & Дехтярь, А. П. (2005, June). Информационные технологии в задачах компьютерной диагностики с использованием интеллектуальных систем. In Научно−методический журнал Клиническая информатика и телемедицина: Материалы научно−практической конференции с международным участием «Компьютерная медицина 2005» (Vol. 2, No. 1, p. 138).

18. Творошенко, І. С. (2018). Спеціалізоване програмне забезпечення: конспект лекцій для магістрів денної та заочної форм навчання спеціальності 193–Геодезія та землеустрій освітньої програми «Геодезія та землеустрій».

19. Творошенко, І. С. (2016). Конспект лекцій з дисципліни «Інтелектуальні геоінформаційні системи» для студентів 1 курсу денної форми навчання спеціальності 193–Геодезія та землеустрій спеціалізації (освітньої програми)–«Геоінформаційні системи і технології».

20. Творошенко, І. С., & Шевченко, А. Р. Практичні аспекти застосування геоінформаційного аналізу для забезпечення раціонального благоустрою міста Сєвєродонецька.

21. Творошенко, І. С. (2017). Конспект лекцій з дисципліни «Геоінформаційні системи в управлінні територіями»(для студентів 5 курсу денної форми навчання спеціальностей 7.08010105–Геоінформаційні системи та технології, 8.08010105–Геоінформаційні системи та технології та студентів 6 курсу заочної форми навчання спеціальності 7.08010105–Геоінформаційні системи та технології).

22. Творошенко, І. С. Конспект лекцій з дисципліни «Геоінформаційні системи в задачах моніторингу» для студентів 1 курсу денної форми навчання спеціальності 193–Геодезія та землеустрій спеціалізації (освітньої програми)«Геоінформаційні системи і технології».

23. Творошенко, І. С. (2015). Конспект лекцій з дисципліни «Основи моделювання складних систем»(для студентів 2 курсу заочної форми навчання напряму підготовки 6.080101–Геодезія, картографія та землеустрій).

24. Творошенко, И. С., Мгеброва, В. Р., & Белый, В. В. (2015). Практические аспекты применения современных геоинформационных систем для создания муниципальной геоинформационной системы города Харькова. Системи обробки інформації, (7), 65−70.

25. Кучеренко, Є. І., Глушенкова, І. С., & Творошенко, І. С. Методичні вказівки до виконання лабораторних, розрахунково-графічних та самостійних робіт з дисципліни «Планування та управління ГІС-проектами.

26. Гончаренко, М. О. (2015). Сравнительный анализ методов формирования дескрипторов изображений в контексте задачи сегментации видеопотока.

27. Гороховатский, В. А. (2014). Структурный анализ и интеллектуальная обработка данных в компьютерном зрении.

28. Putyatin, Y., Gorohovatsky, V., Gorohovatsky, A., & Peredriy, E. (2008). Projective methods of image recognition.

29. Власенко, Н. В. (2013). Построение информативных компактных описаний и классификация объектов путем представления в ортогональном базисе.

30. Путятин, Е. П., Гороховатский, В. А., & Кузьмин, С. В. (2006). Распознавание изображений в пространстве инвариантных локальных признаков. Радиоэлектроника и информатика, (1).

31. Куликов, Ю. А., & Лисовин, Д. В. (2010). Исследование структурно-иерархических методов распознавания изображений.

32. Ляшенко, В. В., Кобылин, О. А., & Томич, С. Н. (2016). Основные этапы обработки изображений цитологических препаратов с использованием идеологии вейвлетов.

33. Кочина, М. Л., & Калиманов, В. Г. (2008). Методы обработки изображений для автоматизации диагностики патологии экстраокулярных мышц.

34. Гороховатский, В. А. (2008). Применение отношений на множестве характерных признаков изображений при распознавании на основе голосования.

35. Фоменко Н. (2022) Розробка та дослідження порівняння полігональних суперпіксельних сегментацій зображень, Abstracts of VII International Scientific and Practical Conference «Innovative areas of solving problems of science and practice» (November 8-11, 2022). Oslo, Norway, pp. 678-680.