**Ефективна сегментація зображень на основі графів**

Pedro F. Felzenszwalb  
Artiflcial Intelligence Lab, Massachusetts Institute of Technology  
[pfi@ai.mit.edu](mailto:pfi@ai.mit.edu)

Daniel P. Huttenlocher  
Computer Science Department, Cornell University  
[dph@cs.cornell.edu](mailto:dph@cs.cornell.edu)

**Анотація**

У цій статті розглядається задача сегментування зображення. Буде визначено предикат для визначення необхідності розділення сегментів, що використовує представлення зображення у вигляді графу. Потім буде розроблено ефективний алгоритм сегментації на основі цього предиката і показано, що, незважаючи на те, що алгоритм приймає «жадібні рішення», отримана сегментація задовольняє глобальним потребам. Алгоритм буде застосовано для сегментації зображень, з використанням двох різних типів вибору сусідів при побудові графу. Тести будуть проведені з реальними та штучними зображеннями. Алгоритм виконується за час близький до лінійного по відношенню до кількості ребер у графі, дає гарну швидкість на практиці. Важливою характеристикою методу є його здатність зберігати деталі в областях зображення з низькою варіативністю, при цьому ігноруються деталі в високоваріативних регіонах.

**Ключові слова:** сегментація зображень, кластеризація, перцептивне групування, робота з графом

**ВСТУП**

Задачі сегментації і групування зображень залишаються складними для комп'ютерного зору. З часів гештальтпсихології (наприклад, [17]) було відомо, що сприйняття форми(перцептивне сприйняття) відіграє важливу роль у зоровому сприйнятті людини. Широкий спектр задач обчислювального бачення може використовувати сегментовані зображення, якщо такі надійно і ефективно обчислюються. Наприклад, задачі бачення середнього рівня складності, такі як стереобачення та оцінка руху потребують підтримки відповідних операцій. Просторово неоднорідні області можуть бути ідентифіковані за допомогою методів сегментації. Задачі бачення вищого рівня, такі як розпізнавання та індексація зображень, також можуть використовувати результати сегментації для розпізнавання по частинах.

Нашою метою є розробка обчислювальних підходів до сегментації зображень, які були б настільки ж корисними як виявлення країв в задачах комп’ютерного бачення. Для досягнення такої широкої користі ми вважаємо важливим, щоб метод сегментації мав наступні властивості:

1. Захоплював перцептуально важливі групи або регіони, які часто відображають глобальні аспекти зображення. Двома центральними питаннями є забезпечення точних характеристик того, що є перцептуально важливим, і вміння конкретизувати, що робить дана методика сегментації. Повинні бути чіткі визначення властивостей результуючої сегментації, щоб краще зрозуміти метод, а також полегшити порівняння різних підходів.
2. Був високоефективним, працюючи в часі майже лінійно залежним від кількості пікселів зображення. Для того, щоб бути практичним, метод сегментації повинен працювати зі швидкістю, подібною до визначення країв або іншої техніки візуальної обробки низького рівня, тобто майже за лінійний час і з низькою залежністю від коефіцієнтів по часу. Наприклад метод сегментації, що може обробляти декілька кадрів за секунду може бути використаний в додатках для обробки відео.

Хоча за останні кілька років спостерігається значний прогрес в методах сегментації, що засновані на власних векторах (наприклад, [14, 16]), вони дуже повільні для практичних застосувань. На відміну від них, метод, описаний у цій роботі, був використаний у великих програмах, як описано в [13]. Хоча є й інші підходи до сегментації зображень, які є високоефективними, ці методи, як правило, не здатні захопити перцептуально важливі нелокальні властивості зображення, як описано нижче. Розроблена методика сегментації не тільки захоплює певні перцептуально важливі нелокальні характеристики зображення, а й обчислювально ефективна (виконуються за O (n log n) часу для n пікселів зображення, з низькою залежністю від коефіцієнтів по часу і може виконуватися на практиці при обробці відео.

Як і в деяких класичних методах кластеризації [15, 19], наш метод базується на виборі ребер з графа, де кожен піксель відповідає вузлу в графі, і деякі сусідні пікселі з'єднані неорієнтованими ребрами. Ваги на кожному ребрі вимірюють відмінність між пікселями. Однак, на відміну від класичних методів, наша техніка адаптивно налаштовує критерій сегментації на основі ступеня варіативності сусідніх областей зображення. В результаті ми отримуємо метод, який при прийнятті «жадібних рішень» може показувати, що він підкоряється певним неочевидним глобальним властивостям. Також покажемо, що інші критерії адаптації, тісно пов'язані з розробленими тут, призводять до проблем, які є складними для обчислень (NP складність).

Тепер ми звернемося до простого штучного прикладу, який ілюструє деякі не локальні характеристики зображення, захоплені нашим методом сегментації. Розглянемо зображення, зображене у верхньому лівому куті на малюнку 1. Більшість людей скаже, що це зображення має три окремі області: прямокутну форму в лівій половині, область постійної інтенсивності з отвором у правій половині, і високоваріативну прямокутну область всередині області з постійною інтенсивністю. Цей приклад ілюструє деякі перцептуально важливі властивості, які повинні бути враховані алгоритмом сегментації. По-перше, дуже різну інтенсивність не слід розглядати лише як доказ існування декількох сегментів. Широкий розкид інтенсивностей відбувається як в зліва, так і в області з високою варіабельністю справа. Таким чином, недостатньо припустити, що області мають майже постійну або повільно змінювану інтенсивність.

Другим важливим аспектом прикладу на рисунку 1 є те, що три значущі регіони не можуть бути отримані з використанням суто місцевих критеріїв прийняття рішень. Це пояснюється тим, що різниця інтенсивності на кордоні фактично менша, ніж багато відмінностей інтенсивностей в межах області високої варіативності. Таким чином, для сегментації такого зображення необхідно використовувати якийсь адаптивний або нелокальний критерій.

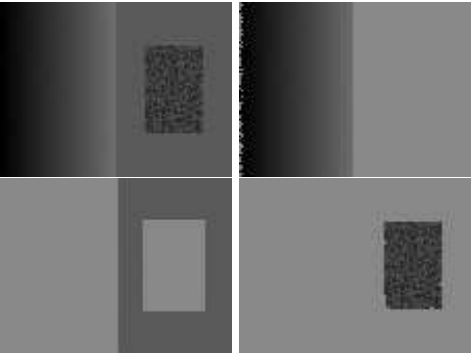


Рис 1. Синтетичне зображення для тестів

Метод Розділі 3.1, досліджує наявність кордону між двома областями шляхом порівняння двох величин: першу на основі відмінностей інтенсивності по всьому можливому кордоні, а другу — на основі відмінностей інтенсивності між сусідніми пікселями в кожній області. Інтуїтивно, відмінності інтенсивностей по межах двох областей перцептуально важливі, якщо вони відрізняються від інтенсивностей хоча б однієї з областей. Хоча цей метод приймає жадібні рішення, він дає результати, які охоплюють певні глобальні властивості. Також метод є досить швидким(менше 1 секунди на зображенні 320 240).

Організація цієї роботи полягає в наступному. У наступному розділі ми обговоримо деякі споріднені роботи, включаючи класичні формулювання сегментації та недавні методи на основі графів. У Розділі 3 ми розглянемо постановку задачі сегментації на основі графу і визначимо предикат порівняння регіонів. Потім у Розділі 4 ми опишемо алгоритм для ефективного сегментування зображення за допомогою цього предиката і опишемо деякі глобальні властивості, які він має. У Розділі 5 ми показуємо результати для ряду зображень з використанням ***grid graph*** на основі зображення. Далі в розділі 6 ми ілюструємо метод, використовуючи більш загальні графи, але де число ребер залишається лінійно залежним від кількості пікселів. У Додатку ми показуємо, що пряме узагальнення предиката порівняння областей, представлене в розділі 3, робить проблему пошуку хорошої сегментації NP-складною.

**2. Споріднені роботи**

Існує велика кількість літератури по сегментації та кластеризації, що датується більш ніж 30 роками тому, з додатками в багатьох областях, навіть за межами комп'ютерного зору(пор. [9]). У цьому розділі ми коротко розглянемо деякі з пов'язаних з цим робіт, які найбільш актуальні для нашого підходу: ранні методи на основі графів (наприклад, [15, 19]), методи об'єднання областей (наприклад, [5, 11]), методи, що ґрунтуються на відображенні пікселів зображення у деякий простір ознак (наприклад, [3, 4]) і більш пізні формулювання в термінах розрізів графів (наприклад, [14, 18]) та спектральні методи (наприклад, [16]).

Методи сегментації зображень на основі графів зазвичай представляють проблему в термінах графа G = (V; E), де кожен вузол відповідає пікселю зображення, а ребра з E з'єднують певні пари сусідніх пікселів. Вага асоціюється з кожним ребром на основі деякої властивості пікселів, що з'єднуються, наприклад їх інтенсивності. Залежно від методу, може існувати або не існувати з’єднання кожної пари вершин. Найбільш ранні методи засновані на графах, використовували фіксовані порогові значення та локальні міри в знаходженні сегментації. У роботі Zahn [19] було представлено метод сегментації, заснований на мінімальному остовному дереві (MST) графа. Цей метод застосовувався як для кластеризації точок, так і для сегментації зображень. Для сегментації зображення ваги ребер у графі засновані на відмінностях між інтенсивностями пікселів, тоді як для кластеризації точок ваги засновані на відстанях між точками.

Критерієм сегментації в методі Zahn–а є розрив ребер MST з великими вагами. Однак, неадекватність зламу великих ребер ілюструє рис 1. Як згадувалося у вступі, різниця між пікселями в області високої варіативності може бути більшою, ніж в області з постійною інтенсивністю. Таким чином, в залежності від порогу, простий розрив ребер з великими вагами призвів б до того, що область високої варіативності була б поділена на декілька частин, або до того, що область з постійною інтенсивністю об’єдналася б з областю з малим перепадом інтенсивностей. Алгоритм, запропонований Urquhart [15], намагається усунути цей недолік шляхом нормалізації ваги ребра, використовуючи найменшу вагу з ребер, до яких входять вершини ребра, що розглядається. Однак у задачах сегментації зображень цього недостатньо для забезпечення адаптивного критерію сегментації. Наприклад, багато пікселів в області високої варіативності на Рис. 1 мають дуже схожого сусіда.

Іншим раннім підходом до сегментації зображень був поділ і злиття областей відповідно до того, наскільки добре кожна область відповідає деякому критерію однорідності (наприклад, [5,11]). Як правило, критерії однорідності підпорядковуються властивості підмножини, такій, що коли предикат однорідності U(A) істинний для деякої області A, то U (B) також істинний для будь-якого B A. Зазвичай такі критерії спрямовані на знаходження або однорідної інтенсивності, або однорідних градієнтних областей. Жоден із запропонованих на сьогоднішній день критеріїв однорідності регіонів не може бути використаний для правильного сегментування прикладу, наведеного на рис.1. Або ця область буде розділена на частини, або вона буде об'єднана з навколишньою областю.

Ряд підходів до сегментації заснованих на пошуку компактних кластерів в деякому просторі об'єктів ([3, 9]). Ці підходи зазвичай припускають, що зображення кусково-постійне, тому що пошук пікселів, які знаходяться близько один до одного в деякому просторі об'єктів, неявно вимагає, щоб пікселі були однаковими (наприклад, мали однаковий колір). Недавній метод з використанням кластеризації просторових об'єктів [4] спочатку перетворює дані, згладжуючи їх таким чином, щоб зберегти межі між регіонами. Ця операція згладжування має загальний ефект зближення точок в кластері. Потім метод знаходить кластери, розширюючи кожну точку гіперсферою деякого фіксованого радіусу і знаходячи пов'язані компоненти розширених точок. Цей метод пошуку кластерів не вимагає, щоб всі точки в кластері знаходилися на фіксованій відстані. Метод насправді тісно пов'язаний з предикатом порівняння областей, який ми вводимо в розділі 3.1, який можна розглядати як адаптивний спосіб вибору відповідного радіуса розширення. Ми повернемося до цього питання в розділі 6.

Нарешті, ми коротко розглянемо клас методів сегментації, заснованих на пошуку мінімальних зрізів на графі, де критерій зрізу розроблений для мінімізації подібності між пікселями, які розбиваються. Робота Wu та Leahy [18] ввела такий критерій скорочення, але він був зміщений у бік пошуку невеликих компонентів. Це зміщення було усунуто з допомогою нормалізованого критерію зрізу, розробленого Shi та Malik [14], який враховує самоподібність регіонів. Ці підходи сегментації на основі зрізів захоплюють нелокальні властивості зображення, на відміну від ранніх методів на основі графів. Однак вони дають тільки характеристику кожного зрізу, а не остаточної сегментації.

Нормалізований критерій зрізу забезпечує значний прогрес порівняно з попередньою роботою в [18], як з теоретичної, так і з практичної точки зору (отримані сегментації захоплюють інтуїтивно помітні частини зображення). Однак нормалізований критерій розрізу також дає NP-важку обчислювальну задачу. У той час як Shi та Malik розробляють методи апроксимації для обчислення мінімального нормалізованого розрізу, похибка в цих наближеннях недостатньо вивчена. На практиці ці апроксимації все ще досить важко вирахувати, що обмежує метод відносно невеликими зображеннями, через вимагання часу обчислення в декілька хвилин. Нещодавно Weiss [16] показав, як апроксимації на основі власних векторів, розроблені Shi і Malik, відносяться до більш стандартних методів спектрального розбиття на графах. Однак всі такі методи занадто повільні для багатьох практичних застосувань

Альтернативою підходу зрізу графу є пошук циклів у графі, вбудованому в площину зображення. Наприклад, в [10] якість кожного циклу нормується таким чином, що це стає тісно пов'язаним з підходом нормалізованих зрізів.

**3 Сегментація на основі графу**

Використаємо заснований на графі підхід до сегментації. Нехай G = (V, E) неорієнтований граф з вершинами vi ∈ V - набір елементів для сегментації, і ребра ∈ *E* відповідають парам сусідніх вершин. Кожне ребро має відповідну вагу , яка є невід'ємної мірою відмінності між сусідніми елементами vi і vj. У разі сегментації зображення елементи з V є пікселями, а вага ребра є деякою мірою відмінності між двома пікселями, пов'язаними цим ребром (наприклад, різниця в інтенсивності, кольорі, місцезнаходженні або якому-небудь іншому локальному атрибуті). У розділах 5 і 6 ми розглянемо конкретні набори ребер і вагових функції для сегментації зображень. Однак формулювання тут не залежить від цих визначень.

У графовому підході сегментація S являє собою розбиття V на компоненти таким чином, що кожен компонент (або область) С∈ S відповідає з’єднаним компонентом у графі G’ = (V, E’), де E’ Е. Іншими словами, будь-яка сегментація є підмножиною ребер в Е. Існують різні способи вимірювання якості але вцілому ми хочемо, щоб елементи в компоненті були схожі, а елементи в різних компонентах-несхожі. Це означає, що ребра між двома вершинами в одному компоненті повинні мати відносно низькі ваги, а ребра між вершинами в різних компонентах повинні мати більш високі ваги.

**3.1. Предикат для попарного порівняння областей**

У цьому розділі ми визначимо предикат D для оцінки необхідності розділення, двох компонентів в сегментації (двох області зображення). Цей предикат заснований на вимірюванні відмінностей між елементами уздовж кордону двох компонентів, щодо міри відмінностей між сусідніми елементами всередині кожного з двох компонентів. Отриманий предикат порівнює міжкомпонентні відмінності з відмінностями всередині компонента і тим самим адаптується до локальних характеристик даних.

Визначимо *внутрішню різницю* компонента C ⊆ V як найбільшу вагу в MST(C; E), тобто

 (1)

Ідея, що лежить в основі цієї міри полягає в тому, що компонент С залишається зв’язаним доти, доки ребра, що розглядаються мають вагу меншу за Int (C).

Визначимо різницю між двома компонентами C1, C2 ⊆ V як мінімальну вагу з ребер, що з’єднує ці компоненти. Тобто:



Якщо немає ребра, що з'єднує C1 і C2, то нехай Dif(C1; C2) = . Ця міра різниці в принципі може бути проблематичною, оскільки вона відображає тільки найменшу вагу ребра між двома компонентами. На практиці ми виявили, що ця міра працює досить добре, незважаючи на це очевидне обмеження. Більше того, зміна визначення для використання медіанної ваги або якого-небудь іншого квантиля, щоб зробити його більш стійким до аномалій, робить проблему пошуку хорошої сегментації NP-складною, про це йде мова у додатку.

Предикат порівняння областей обчислює чи необхідно розділяти пару компонентів, перевіряючи, чи є різниця між компонентами Dif(C1; C2) великою у порівнянні до внутрішньої різниці принаймні в одному з компонентів, Int(C1) і Int(C2). Порогова функція використовується для керування ступенем, в якій різниця між компонентами повинна бути більше мінімальної внутрішньої різниці. Визначимо предикат попарного порівняння як,



де мінімальна внутрішня різниця, MInt, визначається як,



Порогова функція керує ступенем, в якому відмінність між двома компонентами повинна бути більше за їх внутрішні відмінності, щоб між ними була межа (D повертав істинне значення). Для невеликих компонентів Int (C) не є хорошою оцінкою локальних характеристик даних. В крайньому випадку, коли = 1, Int (C) = 0. Тому ми використовуємо порогову функцію, засновану на розмірі компонента



Де позначає розмір C, а k-деякий постійний параметр. Тобто для невеликих компонентів нам потрібні більш сильні докази кордону. На практиці k встановлює шкалу спостереження, в якій більше k викликає перевагу більших компонентів.

Будь-яка невід'ємна функція одного компонента може бути використана для без зміни алгоритму в розділі 4. Наприклад, метод сегментації може віддати перевагу компонентам певних геометричних форм, визначивши таку функцію , яка повертає великі значення для компонентів, які не відповідають деякій бажаній формі, і маленькі значення для тих, які відповідають. Це змусить алгоритм сегментації агресивно об'єднувати компоненти, які не мають бажаної форми. Такі переваги форми були б настільки слабкими, як переважні компоненти, які не є довгими і тонкими (наприклад, використовуючи відношення периметра до площі) або  сильні, як переважні компоненти, які відповідають конкретній моделі форми. Зверніть увагу, що результатом цього будуть не тільки компоненти бажаної форми, проте серед будь-яких двох сусідніх компонентів один завжди буде бажаної форми

**4. Алгоритм і його властивості**

У цьому розділі ми опишемо і проаналізуємо алгоритм отримання сегментації з використанням критерію прийняття рішення D, введеного вище. Ми покажемо, що сегментація, отримана за цим алгоритмом, не надто груба і не надто добра, згідно з такими визначеннями:

**Визначення 1** *Сегментація S називається «надто доброю», якшо існує пара областей C1; C2 ∈ S, для яких немає доказів існування границі між ними.*

Щоб визначити, що значить для сегментації бути занадто грубою (мати занадто мало компонентів), ми спочатку введемо поняття *уточнення сегментації*. Для двох заданих сегментацій S і T, утворених з одних початкових даних T називають уточненням S, коли кожен компонент з T міститься (або еквівалентний) в деякому компоненті з S. На додачу, ми кажемо, що T є *правильним уточненням* S коли T S. Зауважимо, що якщо T-правильне уточнення S, то T може бути отримано шляхом розщеплення однієї або декількох областей S. Коли T є правильним уточненням S ми кажемо, що T більш добра за S і S більш груба за T.

**Властивість 1.** Для будь-якого (скінченного) графу G = (V; E) існують деякі сегментації, які не є ні надто грубими, ні надто добрими.

Легко зрозуміти чому ця властивість виконується. Розглянемо сегментацію, де всі елементи знаходяться в одному компоненті. Очевидно, що ця сегментація не надто добра, тому що є тільки один компонент. Якщо сегментація також не надто груба, ми закінчили. В іншому випадку, за визначенням надто грубої сегментації, отримане правильне уточнення не є надто добрим. Обравши одне з цих удосконалень можна повторювати цю процедуру до тих пір, поки ми не отримаємо сегментацію, яка не буде надто грубою. Процедуру можна повторювати N-1 разів, тому що всякий раз, коли ми вибираємо правильне уточнення, ми збільшуємо кількість компонентів у сегментації принаймні на один. Тоді *найкраща* сегментація, яку ми можемо отримати та, кожен елемент якої міститься у своєму компоненті.

Тепер звернемося до алгоритму сегментації, який тісно пов'язаний з алгоритмом Краскала для побудови мінімального остовного дерева графа [6]. Його реалізація може працювати за час O(m log m), де m – кількість ребер у графі.

**Алгоритм 1** *Алгоритм сегментації.*

На вхід подається граф G = (V; E), з n вершинами і m ребрами. Результатом є сегментація V на компоненти S = (C1;…; Cr).

1. Сортуємо E π = (o1, . . . , om) у порядку зростання за вагами.
2. Починаємо з сегментації S0 , де кожна вершина vi знаходиться у своєму компоненті.
3. Повторюємо крок 3 для q = 1; … ; m.
4. Будуємо Sq за заданою Sq-1. Нехай vi і vj позначають вершини, з’єднані ребром q-й послідовності, тобто oq = (vi; vj). Якщо vi і vj знаходяться в системі неперетинних множин Sq-1 і w(oq) мала, в порівнянні з внутрішньою різницею обох цих компонентів, то необхідно об’єднати ці два компоненти. В іншому випадку – нічого не робити. Більш формально, Нехай є компонентом з Sq−1, що містить vi і  - компонент, що містить vj. Якщо  і w(oq) ≤ MInt(, ), тоді при утворенні сегментації Sq компоненти об’єднуються. В іншому випадку компоненти  і переходять до сегментації Sq без змін.
5. Повернути S = Sm.

Тепер покажемо, що сегментація S, отримана алгоритмом 1, підпорядковується глобальним властивостями не бути ні надто доброю, ні надто грубою при використанні предиката порівняння областей D, визначеного в (3). Тобто, хоча алгоритм і приймає тільки жадібні рішення, він породжує сегментацію, що задовольняє цим глобальним властивостям. Крім того, ми покажемо, що будь-які можливі сортування у порядку зростання, на 0 кроці алгоритму, виробляють ту ж сегментацію.

**Лемма 1.** *Якщо на 3 кроці алгоритму, при розгляді ребра oq, компоненти, що розглядаються, не об’єднуються, то один з них обов’язково буде в кінцевій сегментації. Нехай  і  позначають два компоненти, що з’єднані ребром oq = (vi; vj), коли це ребро розглядається алгоритмом. Тоді, або , де - компонентом, що містить vi і  - компонент, що містить vj. укінцевій сегментації.*

**Доведення.** Є два випадки, які призведуть до того, що злиття не відбудеться. Розглянемо випадок . Оскільки ребра розглядаються в неспадаючому порядку за вагою , для всіх . Таким чином, ніяких додаткових злиттів з цим компонентом не відбудеться, тобто, . Аналогічно для .

Зверніть увагу, що у Леммі 1 мається на увазі, що ребро, яке є причиною злиття двох компонентів є саме ребром мінімальної ваги між компонентами. Таким чином, ребра, що викликають злиття, - це саме ті ребра, які будуть обрані алгоритмом Краскала для побудови мінімального остовного дерева (MST) кожного компонента.

**Теорема 1.** Сегментація, отримана за алгоритмом 1, яка використовує предикат порівняння областей D, визначений в (3), не є надто доброю грубою, відповідно до визначення 1.

**Доведення.** За визначенням, для того, щоб S була надто доброю, повинна існувати деяка пара компонентів, для яких D не виконується. Між такою парою компонентів повинно бути хоча б одне ребро, яке розглядалося на кроці 3 і не викликало злиття. Нехай  перше таке ребро в послідовності. У цьому випадку алгоритм вирішить не об’єднувати з , що . За леммою 1 відомо, * або  У будь-якому випадку ми бачимо, що* . Мається на увазі, що D має місце для і , що є протиріччям.

**Теорема 2.** Сегментація, отримана за алгоритмом 1, яка використовує предикат порівняння областей D, визначений в (3), не є надто грубою, відповідно до визначення 2.

**Доведення.** Для того, щоб S була надто грубою, має бути якесь правильне уточнення, T, яке не надто добре. Розглянемо мінімальну вагу ребра e, яке є внутрішнім для компонента C S, але з'єднує різні компоненти A, B T. Відзначимо, що за визначенням уточнення A C і B C.

Оскільки T не надто добра  або . Не втрачаючи загальності, скажемо, що модель вірна. За конструкцією, будь яке ребро, що з’єднує А з іншим підкомпонентом С має масу щонайменше таку ж велику, як w (e), яка, у свою чергу, перевищує максимальну вагу ребра в MST(A; E) тому що . Таким чином, алгоритм, який розглядає ребра в неспадаючому порядку за вагами, повинен врахувати всі ребра з MST(A; E) перед іншими частинами з C. Таким чином, алгоритм повинен сформувати A перед формуванням C, і при формуванні C він повинен об'єднати A з деяким іншим підкомпонентом C. Вага ребра, що викликала це злиття, повинна бути не менше w (e). Однак алгоритм не об'єднав би A в цьому випадку, оскільки . Протиріччя.

**Теорема 3.** *Сегментація, отримана за алгоритмом 1, не залежить від того, який неспадний порядок ребер за вагами використовується.*

***Доведення.*** Будь-який порядок може бути змінений на інший шляхом заміни сусідніх елементів з однаковою вагою. Таким чином, достатньо показати, що заміна двох сусідніх ребер однакової ваги в неспадаючому ваговому порядку не змінює результат, отриманий алгоритмом 1.

Нехай та – сусідні ребра в неспадній послідовності з однаковою вагою. Ясно, що якщо вони з'єднують непересічні пари компонентів, то й порядок, в якому вони розглядаються, не має значення. Єдиний випадок, який нам треба перевірити, - це коли знаходиться між двома компонентами A і B, а - між одним з цих компонентів, скажімо B, та деяким іншим компонентом C.

Тепер ми покажемо, що викличе злиття, коли розглядається після так само як і при розгляді перед . Спершу, припустимо, що викликає злиття при розгляді до . Це означає, що . Якби розглядався перед , або не викликав би злиття, і тривіально все одно викликав би злиття, або викликав би злиття, і в цьому випадку новий компонент B С мав би  Таким чином, ми знаємо , що означає, що все одно викличе злиття. З іншого боку, припустимо, що не викликає злиття, якщо розглядати його до . Це означає, що. Тоді або , і в цьому випадку це все одно було б вірно, якби спочатку розглядався (бо не стосується A), або . У другому випадку, якщо розглядався першим, він не міг викликати злиття, оскільки  і тому . При розгляді після у нас все ще є  і не викликає злиття.

* 1. **Реалізація та час виконання**

Наша реалізація отримує сегментацію S, використовуючи систему неперетиних множин з об'єднанням по рангу і стисненням шляхів [6]. Час роботи алгоритму можна розділити на дві частини. Спочатку на кроці 0 необхідно відсортувати ваги у порядку зростання. Для цілочисельних ваг це може бути зроблено за лінійний час за допомогою сортування підрахунком, і в цілому це може бути зроблено за час O(M log m) за допомогою декількох методів сортування.

Кроки 1-3 алгоритму займають час O(m (m)), де -дуже повільно зростаюча зворотня функція Аккермана. Щоб перевірити, чи є дві вершини в одному компоненті, ми використовуємо set-flnd на кожній вершині, а для об'єднання двох компонентів ми використовуємо set-union. Таким чином, на ребро припадає не більше трьох операцій неперетинрих множин. Обчислення MInt може бути виконано за константний час для одного ребра, якщо ми знаємо Int і розмір кожного компонента. Збереження Int для компонента може виконуватися за константний час для кожного злиття, оскільки максимальна вага ребра в MST компонента - це просто ребро, яке викликає злиття. Це пов'язано з тим, що Лемма 1 передбачає, що ребро, яке викликає злиття, є ребром мінімальної ваги між двома об'єднуваними компонентами. Розмір компонента після злиття - це просто сума розмірів двох компонентів, які об'єднуються.

1. **Результати для (**Grid Graphs**)**

Спочатку розглянемо випадок монохромних (інтенсивних) зображень. Кольорові зображення обробляються як три окремих монохромних зображення, як описано нижче. Як і в інших графових підходах до сегментації зображень [14, 18, 19], ми визначаємо неорієнтований граф G = (V; E), де кожен піксель зображення pi має відповідну вершину vi V. Набір ребер E будується шляхом з'єднання пар пікселів, які є сусідами в 8-зв'язному шаблоні(можна використовувати інші шаблони з’єднань). Це породжує граф, де m = O(n), тому час роботи алгоритму сегментації становить O (N log n) для зображення з n пікселів. Ми використовуємо вагову функцію для ребер, засновану на абсолютній різниці інтенсивності між пікселями:



де I (pi) - інтенсивність пікселя pi. Загалом, ми використовуємо гаусів фільтр, щоб трохи згладити зображення перед обчисленням ваг ребер, щоб компенсувати артефакти оцифровки. Ми завжди використовуємо гауссіан з σ = 0.8, який не викликає видимих змін зображення, але допомагає видалити артефакти.

Для кольорових зображень ми запускаємо алгоритм тричі: для червоного, зеленого і синього каналів, а потім перетинаємо три отримані набори компонентів. Зокрема, ми поміщаємо два сусідніх пікселя в один компонент, коли вони з'являються в одному компоненті у всіх трьох сегментаціях. В якості альтернативи можна запустити алгоритм тільки один раз на графі, де ваги ребер вимірюють відстань між пікселями в деякому просторі кольорів, проте експериментально ми отримали кращі результати, перетинаючи одноканальні сегментації описаним вище способом.

Існує один параметр часу виконання алгоритму, який є значенням k, яке використовується для обчислення порогової функції . Нагадаємо, ми використовуємо функцію , де  -кількість елементів у С. Таким чином, k ефективно встановлює шкалу спостереження, в якій більше k викликає перевагу більших компонентів. Ми використовуємо два різних налаштування параметрів для прикладів у цьому розділі (і по всій статті), у залежності від дозволу зображення і ступеня, в якому дрібні деталі важливі в сцені. Наприклад, в зображеннях 128\*128 з достатньюю кількістю деталей ми використовуємо k = 150. У 320 \* 240 або більших зображеннях, з великими сценами ми використовуємо k = 300.

На рисунку 2 показана вулична сцена. Зверніть увагу, що існує значна варіація трав'янистого схилу, що веде до паркану. Наш алгоритм розроблено для обробки саме такого виду варіабельності (згадаємо область високої мінливості в штучному прикладі на Рис. 1). На другому зображенні показана сегментація, де кожній області присвоюється випадковий колір. Шість найбільших компонентів, знайдених алгоритмом: три трав'янистих ділянки за парканом, трав'янистий схил, фургон і проїжджа частина. Відсутня частина проїжджої частини в лівому нижньому кутку-це явно помітна область на кольоровому зображенні, з якого була отримана ця сегментація (пляма через артефакт зображення). Зверніть увагу, що фургон також неоднорідний за кольором через відбиття, але вони досить розсіяні, тож розглядаються як внутрішня варіативність і включені в одну область.

На рисунку 3 зображені два бейсболісти (з [14]). Як і в попередньому прикладі, існує трав'яниста область зі значною варіативністю. Уніформа гравців також має суттєві відмінності через складки на тканині. Справа від оригіналу – отримана сегментація. Шість найбільших компонентів, знайдених алгоритмом: задня стіна, емблема Mets, велика трав'яниста область (включаючи частину стіни під верхнім гравцем), форма кожного з двох гравців і невелика трав'яниста латка під другим гравцем. Велика трав'яниста область включає частину стіни через відносно високу варіацію в регіоні. Також важлива тривала повільна зміна інтенсивності між травою і стіною(немає чітко визначеного кордону). Ця "межа" аналогічна за величиною тим, які знаходяться всередині уніформи гравця через складки на тканини. На рис. 4 показані результати алгоритму для сцени в приміщенні, де є як дрібні деталі, так і великі структури, які є перцептивно важливими. Зверніть увагу, що сегментація зберігає невеликі області, такі як бейджі з іменами, які носять люди, і речі за вікнами, створюючи окремі більші області для областей з високою варіативністю, таких як повітропровід кондиціонера у верхній частині зображення, одяг і меблі. Це зображення показує, що іноді зустрічаються невеликі "граничні області", наприклад, на краю куртки або сорочки. Такі вузькі області виникають тому, що існують межі(шириною 1-2 пікселі), які знаходяться посередині між двома сусідніми областями за кольором та інтенсивністю. Це властиве для всіх сегментацій, заснованих на графах. Такі області при бажанні можна усунути, видаливши довгі тонкі області, колір або інтенсивність яких близькі до середнього значення сусідніх областей.

На рис. 5 показані три нескладних об'єкти з бази даних зображення Columbia COIL. Для кожного з них показана найбільша область, знайдена нашим алгоритмом, яка не є частиною чорного фону. Зверніть увагу, що кожен з цих об'єктів має істотний градієнт інтенсивності по поверхні об'єкта, але області правильно сегментовані. Це ілюструє іншу ситуацію, для обробки якої було розроблено наш алгоритм, - повільних змін інтенсивності через освітлення.

1. Результати для Nearest Neighbor Graphs

Загальний підхід до сегментації зображення заснований на зіставленні кожного пікселя з точкою в деякому просторі, а потім на об'єднанні кластерів схожих точок ( [3, 4, 9]). В цьому розділі ми досліджуємо використання алгоритму сегментації на основі графу з розділу 4 для знаходження кластерів схожих точок. У цьому випадку граф G = (V; E) має вершину, відповідну кожній точці об'єкту (кожному пікселю), і існує ребро (vi; vj), що з'єднує пари точок об'єкта vi і vj, які існують поряд у просторі об’єктів, замість використання сусідніх пікселів у сітці зображення. Існує кілька можливих способів визначення того, які точки об'єкта з'єднуються ребрами. Ми з'єднуємо кожну точку з фіксованим числом найближчих сусідів. Іншим варіантом є використання всіх сусідів на деякій відстані d. У будь-якому випадку бажано уникати розгляду всіх o(n2) пар характерних точок.

Вага w((vi; vj)) ребра - це відстань між двома відповідними точками в просторі об'єктів. Для експериментів, показаних тут, ми зіставляємо кожен піксель з точкою об'єкта (x; y; r; g; b), де (x; y) - розташування пікселів на зображенні і (r; g; b) - значення кольору пікселя. Ми використовуємо L2 (Евклідову) відстань між точками в якості ваг ребер, хоча можливі й інші функції відстані.

Внутрішня міра різниці, Int (C), має відносно просте пояснення в просторі ознак. Вона визначає мінімальний радіус розширення необхідний для з'єднання набору точок, що містяться в C, в один том в просторі ознак.

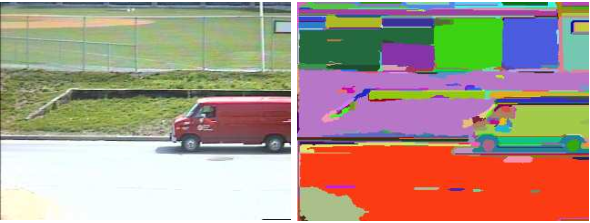


Рис 2. Вулична сцена(кольорове зображення 320\*240) і отримана сегментація (параметри k = 300).

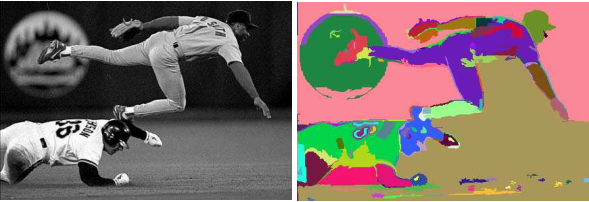


Рис 3. Бейсбольна сцена(кольорове зображення 432\*294) і отримана сегментація (параметри k = 300).



Рис 4. Вулична сцена(кольорове зображення 3210\*240) і отримана сегментація (параметри k = 300).



Рис 5. Зображення з бази даних COIL і найбільші знайдені не фонові (зображення 128\*128) (параметри k = 150).

Розглянемо можливість заміни кожної точки об'єкта кулею з радіусом r. З визначення MST видно, що об'єднання цих куль утворює один єдиний об’єм тільки тоді, коли r Int(C)/2. Різниця між компонентами, Dif (C1; C2), також має просте пояснення. Вона визначає мінімальний радіус розширення, необхідний для підключення принаймні однієї точки C1 до C2. Таким чином, наша методика сегментації тісно пов'язана з роботою [4], яка аналогічним чином використовує підхід до кластеризації на основі розширення точок у просторі параметрів (проте спочатку вони використовують перетворення даних, яке ми не виконуємо, а потім використовують фіксований радіус розширення, а не змінний, як ми).

Замість побудови повного графу, де всі точки є сусідами один одного, ми знаходимо невелике фіксоване число сусідів для кожної точки. Це призводить до графу з O(n) ребрами для n пікселів зображення і загального часу виконання методу сегментації O(n log n). Існує безліч можливих способів вибору невеликої фіксованої кількості сусідів для кожної точки. Ми використовуємо алгоритм ANN [1] для пошуку найближчих сусідів для кожної точки. Цей алгоритм досить швидкий на практиці, враховуючи 5-мірний простір ознак на декілька сотень тисяч точок. Алгоритм ANN може використовуватися для апроксимування найближчіх сусідів, що виконується швидше ніж пошук фактичних сусідів. Для наведених тут прикладів ми використовуємо десять найближчих сусідів кожного пікселя для генерації ребер графу.

Однією з ключових відмінностей від попереднього розділу, де сітка зображення використовувалася для визначення графу, є те, що найближчі сусіди в просторі об'єктів захоплюють більш просторово нелокальні властивості зображення. У разі grid-graph всі сусіди на графіку є сусідами на зображенні. Тут точки можуть бути далеко один від одного на зображенні і все ще бути серед кількох найближчих сусідів (якщо їх колір дуже схожий, а проміжні пікселі зображення мають різний колір). Наприклад, це може призвести до сегментів з областями, які роз’єднані в зображенні, чого не відбулося в разі grid-graph.

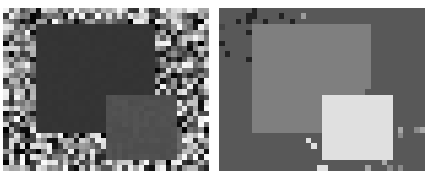


Рис 6. Штучне зображення (40\*32. Відтінки сірого) і отримана сегментація (параметри k = 150).

На рис. 6 показано штучне зображення з [12] і [8] результати сегментації з параметром k = 150 і без згладжування (σ = 0). У цьому прикладі просторово роз'єднані області не відображають цікаві структури сцени, але ми побачимо нижче приклади, які це роблять.

Для інших прикладів в цьому розділі ми використовуємо k = 300 і σ = 0: 8, як і в попередньому розділі. По-перше, ми відзначимо, що ***nearest neighbor graph*** дає аналогічні результати з ***grid graph***, в яких перцептивно помітні області пов'язані просторово. Наприклад, вулична сцена і сцена бейсболіста розглянуті в попередньому розділі дають дуже схожі сегментації, використовуючи або ***nearest neighbor graph****, або* ***grid graph***, як видно з порівняння результатів в Рис. 7 з Рис. 2 і рис. 3.

На рис. 8 Показані два додаткових приклади з використанням графу ***nearest neighbor graph***. Ці результати неможливо досягти з допомогою підходу grid graph, оскільки деякі цікаві області не пов'язані просторово. У першому прикладі показаний квітковий сад, де червоні квіти просторово роз'єднані на передньому плані зображення, а потім зливаються разом на задньому плані. Більшість з цих кольорів об'єднані в одну область, Що було б неможливо за допомогою методу grid-graph. Другий приклад на рис. 8 показує Ейфелеву вежу вночі. Яскраве жовте світло утворює просторово роз'єднані області.

Ці приклади показують, що метод сегментації в поєднанні з використанням ***nearest neighbor graph*** може захоплювати дуже високі властивості зображень, зберігаючи при цьому перцептивно важливі границі області.

1. **Висновки**

У даній роботі представлений новий метод сегментації зображень, заснований на попарному порівнянні областей. Ми показали, що поняття надто грубої або надто доброї сегментації можуть бути визначені в термінах функції, яка вимірює наявність кордону між парою областей. Наш алгоритм сегментації приймає прості жадібні рішення, і все ж породжує сегментації, які підкоряються глобальним властивостям бути не надто грубими і не надто добрими, відповідно до конкретної функції порівняння регіонів. Метод виконується за час O (m log m) для m ребер графа, що на практиці дає добрі результати (частка секунди для невеликих зображень).

Предикат попарного порівняння областей, який ми використовуємо, враховує мінімальну вагу ребра між двома областями при вимірі різниці між ними. Таким чином, наш алгоритм об'єднає дві області, навіть якщо між ними є одне ребро з низькою вагою. Це не така велика проблема, як може здатися, тому, що ця вага ребра порівнюється тільки з мінімальними ребрами остовного дерева кожного компонента. Наприклад, приклади, розглянуті в розділах 5 та 6, ілюструють, що метод знаходить сегментації, які захоплюють багато перцептивно важливих аспектів складних образів. Тим не менш, можна уявити міри, які вимагають більш ніж одного зв’язку, перш ніж вирішувати про відсутність границі між двома областями. Один з природних способів вирішення цієї проблеми-використання квантилю, а не мінімальної ваги ребра. Однак у цьому випадку сегментація, яка не є ні надто грубою, ні надто доброю, є NP-складною задачею (як показано в додатку). Наш алгоритм унікальний тим, що він одночасно і високоефективний, і фіксує нелокальні властивості зображень.

Ми проілюстрували наш алгоритм сегментації зображень двома різними видами графів. Перший з них використовує сітку зображення для визначення локальної околиці між пікселями зображення і вимірює різницю в інтенсивності (або кольорі) між кожною парою сусідів. Другий з них зіставляє пікселі зображення точкам в просторі об'єктів, яке поєднує в собі (x; y) місце розташування і (r; g; b) значення кольору. Ребра на графі з'єднують точки, які знаходяться близько один до одного в цьому просторі. Алгоритм дає хороші результати, використовуючи обидва види графів, але останній тип графа захоплює більш перцептивно глобальні аспекти зображення

Сегментація зображень залишається складною проблемою, проте ми починаємо отримувати істотний прогрес за рахунок впровадження графових алгоритмів, які допомагають відобразити наше розуміння проблеми і надають корисні обчислювальні інструменти. Представлена тут робота і підхід нормалізованих зрізів [14] є лише кількома ілюстраціями цих останніх досягнень.