УДК 004.4, 004.6, 004.932, 004.051, 004.021 https://doi.org/10.17721/1812-5409.2021/1.6

І. Бєда[1], аспірант

Локалізація зміни сцен у відео

Київський національний університет імені Тараса Шевченка, Україна, 03022, м.Київ, просп. Академіка Глушкова, 4Д, 03022
e-mail: igor.beda@gmail.com

I. Bieda[1], PhD student

Scene Change Localization in a Video

Taras Shevchenko National University of Kyiv,
4D, Academician Glushkov ave.,
Kyiv, Ukraine, 03022
e-mail: igor.beda@gmail.com

Мільйони відео щодня завантажуються на Youtube та подібні платформи. Однією з багатьох проблем, з якими стикаються ці служби, є вилучення корисних метаданих. Наприклад, розміщувати рекламу краще в середині відео, і рекламодавець вважає за краще показувати рекламу в перервах між сценами, де це буде менш нав'язливо. Інший приклад – подивитись лише найцікавіші чи найважливіші фрагменти відеозапису. Зрозуміло, що краще застосовувати автоматичний підхід до розпізнавання сцени замість того, щоб розмічати тисячі відеозаписів вручну. Виявлення змін сцени може допомогти автоматично проаналізувати відеопотік: відстежувати, які персонажі з'являються в яких сценах, як довго взаємодіють, їх стосунки та важливість. Потенційне рішення може враховувати різні фактори: появу об'єкта, зміну контрасту чи інтенсивності, зміну фону, зміни звуку.

У цій роботі запропоновано метод для ефективного виявлення зміни сцени, який базується на аналізі сцени з пороговими значенням, а також плавними змінами сцен. Він використовує підходи комп'ютерного зору та аналізу зображень для виявлення зміни сцени. Експериментальні результати демонструють ефективність запропонованого підходу до виявлення змін сцени.

Ключові слова: зміна сцени, переключення сцени, аналіз відео, локалізація сцени.

Millions of videos are uploaded each day to Youtube and similar platforms. One of the many issues that these services face is the extraction of useful metadata. There are a lot of tasks that arise with the processing of videos. For example, putting an ad is better in the middle of a video, and as an advertiser, one would probably prefer to show the ad in between scene cuts, where it would be less intrusive. Another example is when one would like to watch only through the most interesting or important pieces of video recording. In many cases, it is better to have an automatic scene cut detection approach instead of manually labeling thousands of videos. The scene change detection can help to analyze video-stream automatically: which characters appear in which scenes, how they interact and for how long, their relations and importance, and also to track many other issues. The potential solution can rely on different facts: objects appearance, contrast or intensity changed, other colorization, background chang, and also sound changes.

In this work, we propose the method for effective scene change detection, which is based on thresholding, and also fade-in/fade-out scene analysis. It uses computer vision and image analysis approaches to identify the scene cuts. Experiments demonstrate the effectiveness of the proposed scene change detection approach.

Keywords: scene change detection; scene cut; scene break; video analysis; scene localization.

Статтю представила к.ф.-м.н., доц. Розора І.В.

Вступ

Методи комп'ютерного зору (CV) допомагають вирішувати широкий спектр задач, використовуючи відповідні підходи: розпізнавати обличчя, виявляти різні об'єкти, застосовувати необхідні трансформації до сцен. Ще більше задач з'являється при аналізі відео. Так, можна реконструювати тривимірну сцену, відстежувати рухомі об'єкти. Одне з важливих завдань – виявлення зміни сцени. Під сценою ми маємо на увазі частину відео, в якій дія залишається на одному місці протягом деякого неперервного

періоду, а також усіх людей або речі, пов'язані з цим. Поняття сцени є корисним у багатьох випадках:

- розділити відео-потік на більш-менш незалежні та корисні частини (для різних цілей: реклама, ефективна пауза),
- проаналізувати присутність персонажів, що цікавлять, у певній зоні спостереження, та їх поведінку (діяльність акторів та аналіз частоти їх появи; з метою безпеки – знайти епізоди в записах з деякими конкретними видами діяльності; для таких ігор, як футбол – знайти моменти, що цікавлять, або кульмінації; відстеження дій гравців і персонажів),
- відстежувати людську діяльність або переміщення предметів у загальному контексті та знаходити моменти, що мають важливе значення, у відео-потоках або записах,
- для швидшого переміщення по відео,
- для кращої навігації по фільмах, телевізійних шоу або записах ігрових процесів після цього – для визначення моментів, що викликають найбільший інтерес; це однозначно покращить досвід перегляду для кінцевих користувачів.

Одне із завдань – розділити відео на окремі сцени – для подальшої анотації, спрощеної навігації, якщо користувачеві потрібна лише якась певна частина відео, та для подальшої обробки. Це спрощує збір наборів даних для багатьох нейронних мереж, які працюють з функціями з відео, економить час для користувачів, яким потрібно шукати певний момент на відео без тривалого перегляду, і пришвидшує роботу людей, що працюють з відео в цілому.

Існують методи, засновані на комп'ютерному зорі та машинному навчанні (ML), але вони не дають чудових результатів, і ми продемонструємо це далі в статті.

Основними критеріями цього завдання є точність та швидкість алгоритму для практичної зручності використання. Складність обчислень є також звичайним недоліком і перешкодою для реального використання для багатьох подібних алгоритмів, оскільки методи обробки відео зазвичай ресурсомісткі.

Деякі важливі результати щодо цієї задачі можна знайти у [1–7]. Здебільшого, сучасні роботи концентруються на певній підзадачі:

- виявлення зміни місця події у місті або вулиці (з зовнішніх вуличних камер, для міркувань безпеки),
- виявлення змін на супутникових знімках,
- розпізнавання рухомих об'єктів у відео безпілотних літальних апаратів (БПЛА),
- аналіз 3D-сцен з кількох камер,
- аналіз зображень радіолокатора із синтетичною апертурою (SAR).

Ми дослідимо класичне завдання виявлення сцени (або зміни сцени) у відео.

Аналіз індикаторів зміни сцени

Визначимо можливі показники зміни сцени. Не існує чіткого визначення такої події, але в нашому підході ми розглянемо наступне:

- затухання / згасання, плавний перехід, коли відео плавно зникає, а потім з'являється знову, але зі зміненим фоном (сценою),
- миттєва зміна сцени, коли фон відео негайно змінюється між двома послідовними кадрами,
- аналіз зсуву колірних компонентів та зміни значень об'єктів (інтенсивності, контрастності, насиченості, яскравості тощо) – у контексті різних колірних просторів; серед них ми проаналізуємо: інтенсивність (середнє значення кольорових каналів R, G, B) та контрастність (різниця максимальної та мінімальної насиченості пікселів у ділянці, поділена на суму мінімальної та максимальної насиченості),
- аналіз контурів об'єктів зображення (його структура, рух, інші зміни),
- аналіз покадрової різниці, який допомагає відстежувати, наскільки змінився вміст кадру (а не лише контури),
- розпізнавання обличчя для відстеження їх руху – для ідентифікації певних персонажів сцени,
- крім простого аналізу зображення в русі, може бути застосований аналізу звуку.

Одне з очевидних рішень, яке виникає при знаходженні відмінностей між двома зображеннями – використання оператора Кенні [8] для пошуку контурів зображення. Оператор Кенні дозволяє отримати матрицю з межами. Але матриця – це великий набір значень, який повільно обробляється. Тому ми пропонуємо розширити межі на цих матрицях, перетинати їх з контурами іншого кадру, а з отриманих накладок знаходити контури розбіжностей вже як масиви координат, а потім з'ясовувати, наскільки вони

змінилися між двома послідовними кадрами, описуючи це з одним значенням.

Таким чином, ми будемо обчислювати різницю не між місцями розташування, а між самими ланцюжками з точок, що утворюють контури. Цей шлях набагато репрезентативніший, оскільки нам потрібно знати, наскільки змінився вміст кадру, а не лише контури. Пропонуємо такий алгоритм дій:

• перетворити два послідовні кадри в чорно-білі, перевівши подання на кольорову схему HSV та видаливши насиченість (S),

• зменшити вплив шуму під час знаходження меж за допомогою розмиття Гауса,

• застосувати оператор Кенні і отримати матрицю меж для двох кадрів,

• розширити отримані межі,

• накласти розширені межі одного кадру на інший і навпаки – отримаємо два значення, які представляють, наскільки відрізняються межі, навіть розширюючись,

• знайти контури як масиви координат точок, які утворюють межі на матриці невідповідності,

• з'ясувати, наскільки велика різниця в контурах, взявши максимум з двох.

Введемо числову міру від 0 до 1, яка описує різницю контурів об'єктів між двома кадрами. Цей параметр є хорошим сигналом зміни сцени, оскільки він характеризує, наскільки сильно сцена могла змінитися з точки зору контуру об'єкта або зміни меж.

Цей підхід споживає багато обчислювальної потужності, тому його слід оптимізувати. Хороший момент полягає в тому, що його можна розпаралелювати природним шляхом – аналізом окремих кадрів. Зараз він працює в режимі реального часу в паралельному режимі на процесорі Intel Xeon E5 (16 ядер).

Обчислювальний експеримент та валідація моделі

Продемонструємо аналіз та етапи алгоритму на деяких експериментальних даних (аналіз прикладу відеозапису).

На рис. 1 показано зміну інтенсивності в часі, де кожен значний стрибок є потенційно можливою зміною сцени.

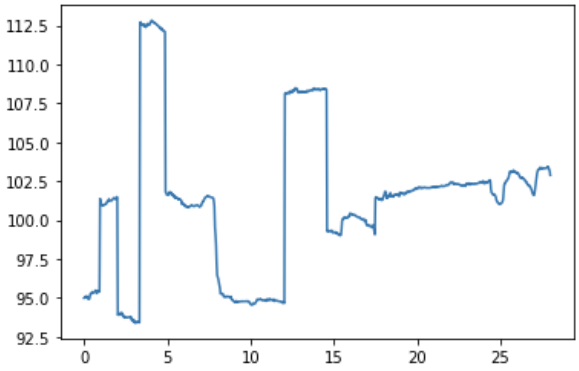

Рис. 1. Зміна інтенсивності освітлення у часі

Різниця міжкадрової інтенсивності в кожен момент часу показана на рис. 2.

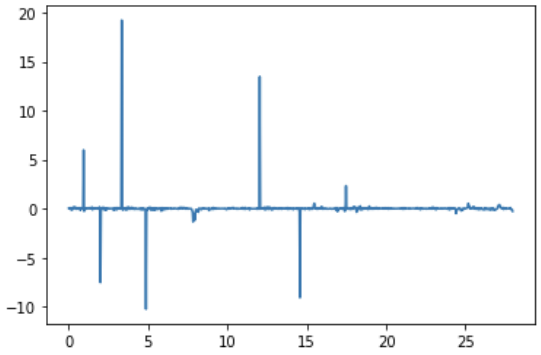

Рис. 2. Зміна відносної інтенсивності освітлення між кадрами

Справді, у нас спостерігаються зміни сцени в ці часові кліки, але є також кілька невеликих піків, і їх слід дослідити на предмет зміни сцени. У даному відео – 9 змін сцени. Отже, ми продовжуємо аналіз інших кольорових показників.

Графік зміни насиченості в залежності від часу представлений на рис. 3, де ми можемо побачити нові можливі точки зміни сцени. На рис. 4 видно різницю в насиченості кадрів.

Основні піки знаходяться в однакові моменти часу з попередніми графіками. У той же час менші піки втрачаються порівняно з рис. 2, показуючи, що на інтенсивність впливають також відтінок та значення компонентів кольорової схеми HSV. Отже, ми робимо висновок, що насиченість тут мало впливає на виявлення зміни сцени.

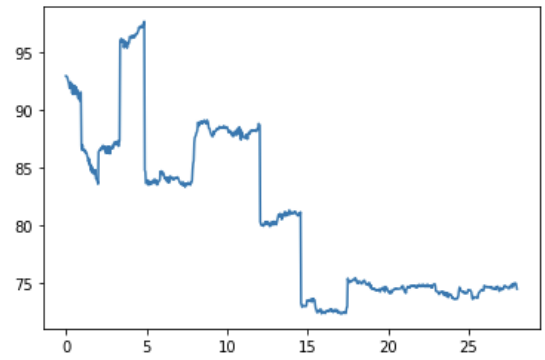

Рис. 3. Зміна насиченості освітлення у часі

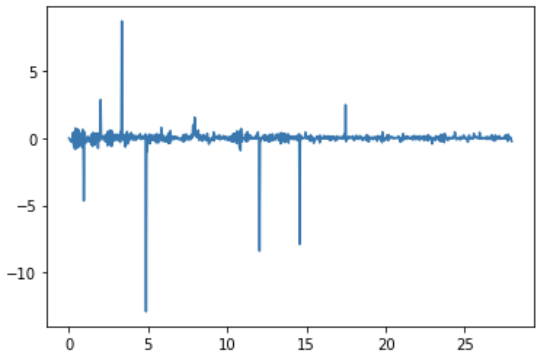

Рис. 4. Зміна відносної насиченості освітлення між кадрами

При такому аналізі пропускається щонайменше одна зміна сцени через її поступове зменшення / зменшення (плавну зміну). Отже, нам також потрібно знайти та дослідити локальні піки в коефіцієнті зміни кордонів (від 0 до 1), який був описаний вище.

З попереднього аналізу випливає, що найвищі піки зміни інтенсивності відображають негайну зміну сцени, тоді як серія суміжних, але нижчих піків показує часовий інтервал повільної (або зникаючої / згасаючої) зміни сцени. Також рекомендується поєднувати аналіз інтенсивності та насиченості одночасно, щоб не пропустити точки зміни сцени. Далі ми покажемо, як локалізувати ці піки. Повинен бути певний поріг, щоб їх виявити. Цей поріг не може бути статичним, оскільки він, очевидно, залежить від якості відео, і може виявити неважливі піки або пропустити зміну сцени на відео. Отже, нам потрібен певний адаптивний поріг.

Хорошими кандидатами на адаптивний поріг є ентропійне порогове значення та динамічне порогове значення на основі вікна, де однією з проблем є підрахування належного розміру вікна (кількість кадрів, які потрібно проаналізувати одночасно для отримання найбільш точних результатів роботи – власне, виявлення зміни сцени).

Використаємо рекурсивний адаптивний поріг пошуку та коригування. Це означає, що відео розділимо на частини за першим базовим порогом (обговорено пізніше) на основі зміни інтенсивності (рис. 5 – це рис. 2, взятий за модулем). Потім процес поділу відео триває рекурсивно, поки це залишається можливим – обчислюється новий пороговий рівень і знаходяться піки, які його перевищують. Наприклад, для розглянутого відео цей процес дав 3 ітерації (рис. 5).

Ми пропонуємо розраховувати поріг як різницю початкової та кінцевої інтенсивності (за модулем), помножену на деякий пороговий коефіцієнт. Його оптимальне значення було знайдено експериментально – див. Таблицю 1. Значення 5 дає найменше значення «Хибнопозитивні + хибні негативні» та більшість справжніх позитивних результатів у досліджуваному наборі відео.

Таблиця 1. Граничні коефіцієнти

Граничний коефіцієнт	True Positives	False Positives	False Negatives
2	17.8% (74)	82.2% (342)	0% (0)
5	97.4% (74)	2.6% (2)	0% (0)
8	100% (68)	0% (0)	8% (6)
15	100% (57)	0% (0)	23% (17)

Алгоритм зупиняється, коли на відео більше немає піків, які перевищують розрахований поріг для цієї конкретної частини відео.

Для змінення сцени з поступовим зменшенням нам все одно потрібно згрупувати локальні піки в кластери та проаналізувати їх. Нам потрібно встановити максимальний розмір кластера. Гіпотеза полягає в тому, що 0,5 секунди – це хороша оцінка. Цей розмір передбачувано залежить від загальної швидкості відео, а отже і від швидкості зміни сцени. Тут ми визначаємо найбільш підходяще середнє значення, яке зручно для більшості відео (принаймні, для проаналізованих зразків) і представляє найкраще

обране експериментальне значення на даний момент - див. Таблицю 2 для детальних розрахунків.

Таблиця 2. Максимальний розмір кластера

Макс. розмір кластера, сек.	True Positives	False Positives	False Negatives
0.2	4	0	7
0.5	11	0	0
0.7	11	2	0
1.2	11	7	0
2	11	22	0

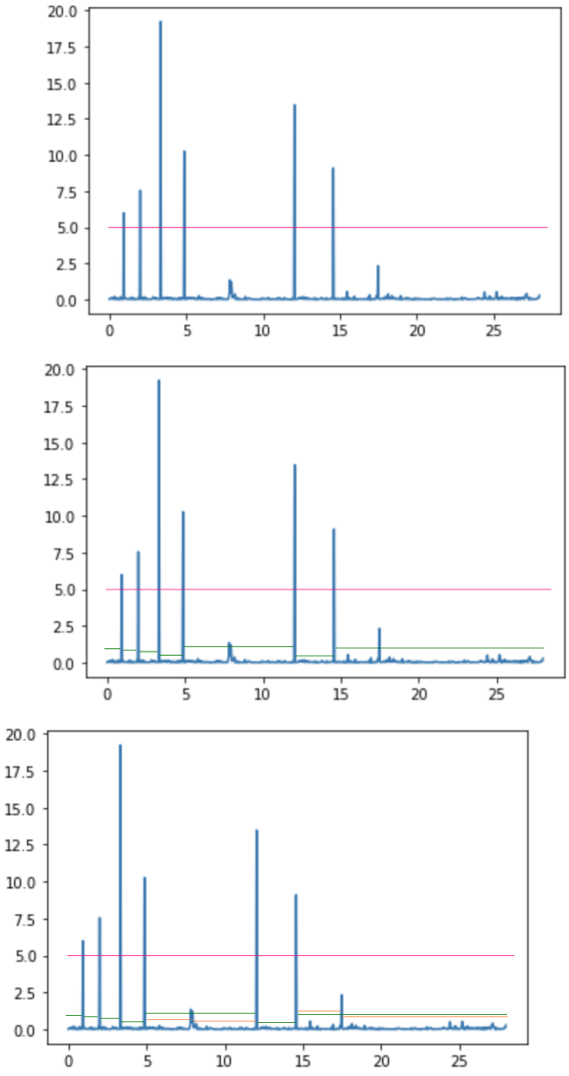

Рис. 5. Зміна відносної насиченості освітлення між кадрами

Дуже короткі періоди часу не фіксують всіх змін сцени (див. Стовпець "True positives"), тоді як тривалі періоди (більше приблизно 1 сек.) починають враховувати незначні локальні піки (за їх кількістю) і генерувати False Positives. Далі ми вибираємо часову позначку в кожному кластері з максимумом оператора Кенні, згаданого раніше, що є мірою різниці між кадрами.

Порівняння методу

Як ми помітили, запропонований метод може працювати в режимі реального часу. Щодо точності, порівняємо результати з PySceneDetect [9], як найбільш доступною бібліотекою для цієї задачі, за набором відео з різними типами змін сцени, з різним освітленням та динамікою. У 11 відео з набору даних є 68 змін сцени. Результати замірів наведено у таблиці 3.

Видно, що запропонований метод пропускає лише 4 з 64 змін сцени, і виявляє 94%, тоді як провідна бібліотека PySceneDetect виявила лише 77%. Ми також перевірили це на різних відібраних вручну відео з подібними результатами. Більше False Positives було виявлено через артефакти, які з'являються на місці несподівано. Таким чином, ми вважаємо, що це також можна зарахувати до зміни сцени.

Таблиця 3. Порівняння методу

Метод / Програмне забезпечення	True Positives	False Positives	False Negatives
PySceneDetect	49	2	19
Запропонований метод	64	6	4

Висновки

У даній роботі запропоновано ефективний підхід до виявлення зміни сцени. Актуальність та випадки використання було розглянуто у вступі. Метод є більш точним, ніж відомий PySceneDetect на підготовленому тестовому наборі відеозаписів.

Для подальшого підвищення точності методу ми додамо аудіоаналіз для виявлення змін або сигналів негайних змін сцени. Метод також можна покращити за допомогою більш точної локалізації фантомних об'єктів, які можуть з'являтися у відеопотоці, відокремлюючи їх від загальної ситуації зміни сцени.

Для оптимізації розрахунків ми спробуємо зменшити масштаб (зменшити роздільну здатність зображення – кадрів відео). Крім того, ми збираємось використовувати запропонований спосіб для обробки відео у промислових масштабах.

Список використаних джерел

1. *Reddy B.* Comparison of Scene Change Detection Algorithms for Videos / B. Reddy, A. Jadhav // 2015 Fifth International Conference on Advanced Computing & Communication Technologies (ACCT). –Haryana, India. – 2015. – pp. 84-89.
2. *Rotman D.* Robust and Efficient Video Scene Detection Using Optimal Sequential Grouping / D. Rotman, D. Porat, G. Ashour // 2016 IEEE International Symposium on Multimedia (ISM). – 2016. – pp. 275-280.
3. *Rotman D.* Robust video scene detection using multimodal fusion of optimally grouped features / D. Rotman, D. Porat, G. Ashour // 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP). – 2017. – pp. 1-6.
4. *Li Li.* Video shot segmentation using graph-based dominant-set clustering / Li Li, Xianglin Zeng, Xi Li, Weiming Hu, Pengfei Zhu // Proceedings of the First International Conference on Internet Multimedia Computing and Service (ICIMCS '09). – Association for Computing Machinery, New York, NY, USA. – 2009. – pp. 166–169.
5. *Zabih R.* A feature-based algorithm for detecting and classifying scene breaks / Ramin Zabih, Justin Miller, Kevin Mai // Proceedings of the third ACM international conference on Multimedia (MULTIMEDIA '95). – Association for Computing Machinery, New York, NY, USA. – 1995. – pp. 189–200.
6. *Sze K.-W.* Scene cut detection using the colored pattern appearance model / Kin-Wai Sze, Kin-Man Lam, G. Qiu // Proceedings 2003 International Conference on Image Processing. – 2003. – pp. II-1017.
7. *Krulikovská L.* Fast Algorithm of Shot Cut Detection / Krulikovská L., Polec J., Hirner, T. // World Academy of Science, Engineering and Technology, Open Science Index 67, International Journal of Electronics and Communication Engineering. – 2012. – № 6 (7), pp. 633-636.
8. *Canny J.* A Computational Approach To Edge Detection / Canny J. // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1986. – № 8 (6). – pp. 679–698.
9. PySceneDetect documentation [Online]. – Resource Access mode: https://pyscenedetect.readthedocs.io/

References

1. REDDY, B., JADHAV, A. (2015): *Comparison of Scene Change Detection Algorithms for Videos*, 2015 Fifth International Conference on Advanced Computing & Communication Technologies (ACCT), Haryana, India, pp. 84-89.
2. ROTMAN, D., PORAT, D., ASHOUR, G. (2016): *Robust and Efficient Video Scene Detection Using Optimal Sequential Grouping*, 2016 IEEE International Symposium on Multimedia (ISM), pp. 275-280.
3. ROTMAN, D., PORAT, D., ASHOUR, G. (2017): *Robust video scene detection using multimodal fusion of optimally grouped features*, 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP), pp. 1-6.
4. LI, LI, ZENG, X., LI, XI, HU, W., ZHU, P. (2009): *Video shot segmentation using graph-based dominant-set clustering*, Proceedings of the First International Conference on Internet Multimedia Computing and Service (ICIMCS '09), Association for Computing Machinery, New York, NY, USA, pp. 166–169.
5. ZABIH, R., MILLER, J., MAI, K. (1995): *A feature-based algorithm for detecting and classifying scene breaks*, Proceedings of the third ACM international conference on Multimedia (MULTIMEDIA '95), Association for Computing Machinery, New York, NY, USA, pp. 189–200.
6. SZE, K.-W., LAN, K.-M., QIU, G. (2003): *Scene cut detection using the colored pattern appearance model*, Proceed. 2003 International Conference on Image Processing, pp. II-1017.
7. KRULIKOVSKÁ, L., POLEC, J., HIRNER, T. (2012): *Fast Algorithm of Shot Cut Detection*, World Academy of Science, Engineering and Technology, Open Science Index 67, Internatinal Journal of Electronics and Communication Engineering, 6 (7), pp. 633-636.
8. CANNY, J.,(1986):*A Computational Approach to Edge Detection*, IEEE Trans. on Pattern Analysis and Machine Intelligence, 8 (6), pp. 679-698.
9. PySceneDetect documentation [Online] – Resource Access mode: https://pyscenedetect.readthedocs.io/

Надійшла до редколегії 23.10.2020