DOI https://doi.org/10.15407/usim.2020.02.012
УДК 004.932.2

В.М. КИЙКО, канд. техн. наук, ст. наук. співробітник, ст. наук. співробітник,
Міжнародний науково-навчальний центр інформаційних технологій та систем
НАН та МОН України, просп. Глушкова, 40, Київ 03187, Україна,
vkiiko@gmail.com

В.В. МАЦЕЛЛО, канд. техн. наук, ст. наук. співробітник, зав. відділом,
Міжнародний науково-навчальний центр інформаційних технологій та систем
НАН та МОН України, просп. Глушкова, 40, Київ 03187, Україна,
matsello@gmail.com

ПРОСТЕЖЕННЯ ОБ'ЄКТІВ ПІД ЧАС ВІДЕОСПОСТЕРЕЖЕННЯ

Розглядається алгоритм простеження об'єктів у відео, що базується на спільному використанні відомих алгоритмів MOG (Mixture of Gaussians) та KCF (Kernelized Correlation Filters). Наведено результати пошуку та простеження рухомих об'єктів у відео. Показано, що алгоритм є ефективнішим у порівнянні з MOG та KCF в умовах, коли об'єкти, що простежуються, рухаються із порівняно різкими змінами швидкості, напряму руху та орієнтації, а також в умовах часткового укриття або зникнення з поля зору.

Ключові слова: *відеоспостереження, пошук та простеження об'єктів, різницеве зображення, обробка зображень.*

Вступ

Відеоспостереження є поширеним засобом розв'язання задач, пов'язаних із забезпеченням безпеки та моніторингу подій у довкіллі. Одна з головних задач, що виникає при відеоспостереженні, полягає у виявленні, простеженні та ідентифікації рухомих об'єктів. У подальшому ми розглядатимемо загальний випадок виявлення будь-яких рухомих об'єктів у полі зору, а не тільки з наперед заданого класу. Результат пошуку об'єкта на зображенні ми представлятимемо у вигляді найменшого обмежувального прямокутника цього об'єкта. Виявлення зазвичай виконується не на всіх, а тільки на окремих кадрах у відео за допомогою алгоритмів пошуку об'єктів або оператора. Після цього на наступних кадрах виконується простеження об'єктів (визначення траєкторії руху) до зникнення з поля зору або втрати можливості простеження внаслідок надто різких змін орієнтації, швидкості руху, укриття за загорожу або ж перекриття іншим об'єктом.

Наявна значна кількість алгоритмів для пошуку та простеження об'єктів на зображеннях [1–9]. Кожен із них не вповні відповідає практичним потребам, має певні переваги, але також і недоліки у порівнянні з іншими алгоритмами. Значна частина алгоритмів простежують об'єкти на поточному кадрі, здійснюючи пошук цих об'єктів на зображенні (*tracking-by-detection*). Зокрема, до цієї групи належать алгоритми [6,7], що використовують так званий «бустинг»-підхід. Ці алгоритми простеження, на відміну від алгоритмів детектування [8], перенавчаються в режимі «*on line*» на кожному зображенні, використовуючи дані про розташування об'єкта, отримані через обробку попереднього кадра у відео. При цьому

локалізоване зображення на попередньому кадрі використовується як позитивний приклад об'єкта, а певна кількість близько розташованих частин на зображенні цього кадра – як множина негативних прикладів, тобто таких, що мають відрізнятись від позитивного внаслідок процедури навчання та побудови відповідного класифікатора або фільтра. Під час навчання алгоритми [6,7] використовують значно меншу кількість прикладів зображень об'єкта для побудови класифікатора порівняно з [8], але сам процес залишається ітеративним і внаслідок цього доволі повільним.

Одна з відмінностей [9] полягає в тому, що рішення про положення об'єкта, який простежують, на поточному зображенні приймається на підставі результатів не одного, а двох алгоритмів – алгоритму простеження об'єкта на двох сусідніх кадрах за допомогою обчислення оптичного потоку, а також алгоритму детектування таких об'єктів, що навчається у процесі обробки відео. Як іще один крок у цьому напрямку у цій роботі досліджується можливість спільного використання двох відомих алгоритмів *KCF* (*Kernelized Correlation Filters*) [4] та *MOG* (*Mixture of Gaussians*) [5], що реалізовані в *OpenCV* [10], з метою підсилення якості їхньої роботи при простеженні.

Алгоритм *KCF* згідно з результатами тестування [4] є порівняно ефективним за надійністю та швидкістю простеження. Він використовує значно більшу кількість негативних прикладів у процесі навчання порівняно з [6,7], які формуються за допомогою всіх можливих зсувів еталонного зображення (позитивний приклад) об'єкта, і при цьому є порівняно швидшим алгоритмом. Отримання ефективного класифікатора залежить від кількості використаних негативних прикладів.

Тому те, що *KCF* використовує значну кількість негативних прикладів об'єктів, є важливим фактором, який забезпечує порівняно високу надійність простеження. Алгоритм має оцінку обчислювальної складності $O(\text{Log } N)$, де N – кількість клітинок еталонного зображення. При цьому швидкодія *KCF* переважно зумовлюється тим, що побудова дискримінантного кореляційного фільтра під час навчання, а також обчислення двовимірної кореляції цього фільтра з поточним зображенням під час пошуку об'єкта здійснюються у частотній Фур'є-області за допомогою швидкого перетворення Фур'є.

Алгоритм *KCF* має також недоліки, притаманні більшості інших алгоритмів простеження – 1) перед застосуванням необхідно виконати ініціалізацію *KCF*, вказуючи прямокутник, що обмежує об'єкт на зображенні; 2) простеження може бути перервано внаслідок швидких переміщень об'єкта, змін його розмірів (масштабу), напряму руху, орієнтації та умов освітлення.

Зазначені недоліки можна усунути завдяки спільному використанню іншого алгоритму для пошуку та простеження об'єктів за допомогою побудови та оновлення моделі зображення місцевості, на фоні якої пересуваються об'єкти. Ця модель має бути динамічною, адаптивною, а також стійкою до змін зображень, пов'язаних із можливими природними змінами довкілля – гойдання дерев, змін освітлення, тощо. Алгоритми, що використовують поклітинне порівняння або віднімання двох або трьох сусідніх зображень у відео, не є достатньо ефективними для розв'язання цієї задачі.

Порівняно ефективнішим є алгоритм *MOG*, який передбачає, що розподіл імовірності яскравості в кожній клітинці фонового зображення є сумішшю кількох розподілів Гауса, параметри та вагові коефіцієнти яких залежать від координат цієї клітинки та результатів обробки певної кількості попередніх кадрів. Обробка поточного зображення цим алгоритмом полягає у визначенні для кожної клітинки, належить вона об'єкту чи фону шляхом обчислення відповідних значень ймовірності, а також у коригуванні параметрів моделі фонового зображення клітинки за умови ухвалення рішення про її належність до фону.

Обробка кожної клітинки зображення алгоритмом *MOG* виконується незалежно від інших клітинок. Це, з одного боку, забез-

печує достатньо високу швидкість обробки, а з іншого – призводить до появи завад на різницевому зображенні, яке відображає клітинки простежуваних об'єктів. Наявність завад потребує додаткової обробки (сегментації) цього зображення з використанням морфологічних операцій. У [2] розглядаються інші алгоритми простеження, що також визначають параметри фонового зображення, зокрема, повільніші алгоритми, що базуються на використанні прихованих марківських моделей зображення (*HMM*), а також засоби сегментації об'єктів на вихідних зображеннях цих алгоритмів. Нині виробляються також спеціалізовані відеокамери із вбудованою функцією вирізнення пікселів на вхідному зображенні, що можуть належати до рухомих об'єктів.

Суттєвий недолік *MOG* і подібних йому алгоритмів полягає в тому, що простеження може перерватися, коли об'єкт:

1) зупиняється,

2) рухається на фоні іншого рухомого об'єкта або перекривається ним із завадами на різницевому зображенні,

3) суттєво ховається за загорожею або межами поля зору відеоспостереження.

Головні передумови для спільного використання алгоритмів *KCF* та *MOG* полягають у:

1) *MOG* може бути використано для автоматичної ініціалізації *KCF*, а також він є порівняно стійким до змін масштабу та орієнтації об'єктів;

2) *KCF*, на відміну від *MOG*, не зупиняє простеження під час зупинки руху об'єкта, а також в умовах його руху на фоні інших рухомих об'єктів. Алгоритми *KCF* та *MOG* використовують різні ознаки рухомого об'єкта, тому спільне використання їхніх результатів може сприяти усуненню накопичення помилки простеження.

Метою роботи є розробка алгоритму простеження об'єктів у відео, що базується на спільному використанні алгоритмів *KCF* та *MOG*. У наступних розділах розглядається цей алгоритм і результати його тестування щодо пошуку та простеження об'єктів у відео з відомих баз даних, а також знятих у довкіллі.

Простеження рухомих об'єктів на зображеннях на базі спільного використання алгоритмів MOG та KCF.

Результатом роботи алгоритму *MOG* є отримання різницевого зображення. З метою доведення *MOG* до використання для пошуку та простеження об'єктів цей алгоритм доповнено процесами усунення завад на різницевому зображенні, пошуку зв'язних областей кліток на бінарному різницевому зображенні, що можуть відповідати цим об'єктам, а також простеження цих областей у відео.

Алгоритм *KCF* простежує на поточному зображенні об'єкт завдяки обчисленню кореляцій фільтра із зображенням в околі визначених координат цього об'єкта на попередньому зображенні. Визначене нове положення об'єкта у вигляді обмежувального прямокутника відповідає максимальному значенню кореляції, що само по собі не гарантує правильності отриманого результату. При цьому алгоритм *KCF* не налаштовано на можливу зміну розмірів або масштабу об'єкта під час його руху, тому розміри прямокутника не змінюються й залишаються незмінними протягом простеження.

Значення в клітинках різницевого зображення є тим більшими, чим більше відрізняються яскравості цих клітинок на поточному зображенні порівняно зі значеннями, що відповідають моделі фонового зображення. Позначмо через B_r і B_{rf} відповідно середні яскравості клітинок прямокутника r та його зовнішньої рамки на різницевому зображенні (рис. 1). Значення $sim(r) = B_r - B_{rf}k$, де k – коефіцієнт, є певною міркою того, що цей прямокутник є найменшим за розмірами обмежувальним прямокутником об'єкта, що простежується. Найбільші значення $sim(r)$ відповідають умовам, коли r є мінімальним обмежувальним прямокутником об'єкта (збільшення B_r) і не перетинається з іншими рухомими об'єктами (зменшення B_{rf}). Кожне значення $sim(r)$ може бути обчислено за

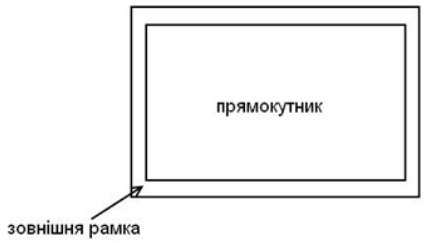

Рис. 1. Прямокутник із зовнішньою рамкою

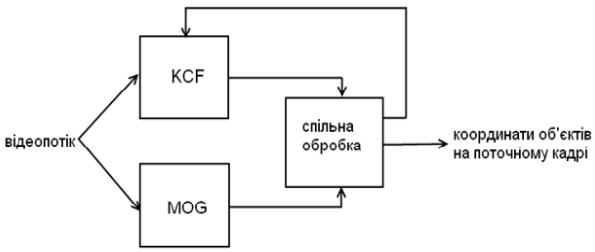

Рис. 2. Простеження об'єктів на зображеннях у відео на базі спільного використання алгоритмів *MOG* та *KCF*

допомогою лише 12 простих операцій завдяки використанню інтегрального представлення різницевого зображення [11].

Загальну схему простеження об'єктів на зображеннях у відеопотоці на базі спільного використання алгоритмів *MOG* та *KCF* представлено на рис. 2.

Позначмо через R^K знайдену сукупність обмежувальних прямокутників простежених рухомих об'єктів на K–му кадрі відео. Обробка наступного $(K+1)$-го кадру полягає у виконанні таких дій:

1. $R^{K+1} = \varnothing$.

2. Змінювання (адаптація) моделі фону, віднімання фону в кожній клітці та отримання в результаті різницевого зображення за допомогою *MOG* алгоритму. Яскравість клітинки цього зображення є тим меншою, чим більшою є імовірність її приналежності до фону, а світліші (яскравіші) клітинки є такими, що не відповідають моделі фону й можуть належати рухомим об'єктам.

3. Пошук нової позиції r_{KCF}^{k+1} об'єкта, що простежується, виконується за допомогою алгоритму *KCF* в околі кожного прямокутника $r \in R^K$. Якщо схожість $sim(r_{KCF}^{k+1})$ перевищує задане порогове значення, то прямокутник r_{KCF}^{k+1} додається до сукупності R^{K+1}. Інакше виконується пошук нової позиції r_{KCF}^{k+1} з використанням даних про різницеве зображення. За виконання умов, що розглядаються далі, здійснюється додавання прямокутника r_{KCF}^{k+1} до сукупності R^{K+1}, а також ініціалізація алгоритму *KCF* на цьому прямокутнику. Інакше здійснюється перевірка умов переривання простеження й за результатом цієї перевірки або прямокутник r_{KCF}^{k+1} додається до сукупності R^{K+1}, або ухвалюється рішення про припинення подальшого простеження об'єкта.

4. Визначення нових об'єктів для простеження шляхом виділення найменших обмежувальних прямокутників цих об'єктів на поточному зображенні за допомогою оператора або автоматичної обробки різницевого зображення. У першому разі оператор зупиняє обробку відео й виділяє обмежувальний прямокутник об'єкта на поточному кадрі для наступного простеження. У другому здійснюються такі операції: 1) бінаризація різницевого зображення й усунення шумів на бінарному зображенні, 2) знаходження зв'язних сукупностей білих клітинок на бінарному різницевому зображенні та визначення кожної із них як нового об'єкта для простежування, якщо найменший обмежувальний прямокутник цієї сукупності має допустимі розміри й не перетинається із жодним прямокутником $r \in R^K$; ініціалізація алгоритму *KCF* на кожному з цих прямокутників та додавання їх до множини R^{K+1}.

Розгляньмо детальніш визначення координат простежуваного об'єкта з використанням алгоритмів *KCF* та *MOG*. В околі $O(r_{KCF}^{k+1})$ прямокутника r_{KCF}^{k+1} розглядаються прямокутники, що відрізняються координатами або масштабом (1,0, 1,1 та 0,9) відносно r_{KCF}^{k+1}. Для кожного із цих прямокутників r обчислюється значення $sim(r) = sim(r)k_1(r, r^k)$, де $k_1(r, r^k)$ – коефіцієнт, який залежить від відстані й орієнтації прямокутника r стосовно обмежувального прямокутника r^k відпо-

відного об'єкта в попередньому кадрі відео. При цьому $k_1(r, r^k)$ зменшується за порівняно значних змін координат і напряму руху r стосовно r^k у попередньому кадрі. Внаслідок цих дій визначається прямокутник $r_{\max}$, що має найбільше значення $sim_1(r_{\max})$ серед інших прямокутників в околі $O(r_{KCF}^{k+1})$.

Зазначені перетворення прямокутника r_{KCF}^{k+1} (зміщення та два значення масштабу) є лише частиною всіх можливих змін зображення простежуваного об'єкта у сусідніх кадрах. Тому на наступному кроці здійснюється додаткове перетворення прямокутника $r_{\max}$ з метою отримання прямокутника $r_{\max} = \arg\max sim(r)$, де $S = \{rect(x, y, w, h)\}$, $|x - x_m| < rx$, $|y - y_m| < ry$, $|w - w_m| < rx$, $|h - h_m| < ry$ є сукупністю допустимих прямокутників; (x_m, y_m), w_m, h_m – відповідно координати лівого верхнього кута, ширина та висота прямокутника $r_{\max}$; rx, ry – задані порогові значення.

Пошук здійснюється за допомогою послідовних зміщень кожної із 4-х сторін прямокутника $r_{\max}$, що збільшують значення схожості $sim(r_{\max})$, тобто дій, що належать до класу так званих «жадібних» алгоритмів. Знайдений прямокутник $r^*_{\max}$ вважається таким, що відповідає позиції простежуваного об'єкта на зображенні ν^{K+1}, якщо виконується одна з двох умов:

1. $sim_1(r^*_{\max}) > \max(sim(r_{KCF}^{k+1})k_s)$, $B_f(r^*_{\max}) < thr_B$, де $k_s > 1$, thr_S, thr_B – задані значення.

2. $sim_1(r^*_{\max}) > \max(thr_S, sim(r_{KCF}^{k+1})k_{S1})$, $B_f(r^*_{\max}) < thr_B$, $d_{KLT}(r^K, r_{KCF}^{k+1}) > d_{KLT}(r^K, r^*_{\max})$, де $k_S > k_{S1} > 1$, $d_{KLT}(r^K, r_{KCF}^{k+1})$; $d_{KLT}(r^K, r^*_{\max})$ – відстані (помилки пошуку відповідності) оптичного потоку між зображенням $\nu^K(r^K)$ та відповідно зображеннями $\nu^{K+1}(r_{KCF}^{k+1})$ і $\nu^{K+1}(r^*_{\max})$, обчисленими за допомогою реалізації методу Лукаса-Канаде [10]. Наведена друга умова передбачає обчислення значень оптичного потоку між зображеннями прямокутників для прийняття надійнішого рішення в умовах, коли відношення схожості $sim_1(r^*_{\max})/sim(r_{KCF}^{k+1})$ є більшим, ніж 1, але меншим від порогового значення k_s.

За виконання однієї з двох зазначених умов ухвалюється рішення про позицію простежуваного об'єкта на підставі різницевого зображення з наступною ініціалізацією алгоритму *KCF* на прямокутній ділянці $\nu^{K+1}(r^*_{\max})$ зображення. Інакше ухвалюється рішення або на користь r_{KCF}^{k+1}, або про зупинку подальшого простеження, коли параметри прямокутника r_{KCF}^{k+1} не змінюються на певній кількості кадрів.

Здійснення розглянутих дій уможливлює корекцію помилкових результатів *KCF* унаслідок різких змін руху, розмірів або орієнтації простежуваного об'єкта та повторну ініціалізацію цього алгоритму на зміненому зображенні об'єкта.

Результати експериментальної перевірки алгоритмів

Відеоспостереження здебільшого здійснюється за допомогою стаціонарної відеокамери, що є переважним для застосування алгоритмів простеження, особливо тих, що базуються на відніманні фону. У процесі тестування було використано 30 відео із сайтів *http://www.viratdata.org/, http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html*, знятих із використанням стаціонарної або рухомої відеокамер, а також відео руху транспортних засобів, знятих за допомогою мобільного телефону. Розглядалися та порівнювалися між собою три алгоритми простеження – *MOG*, *KCF* та *MOG_KCF* зі спільним використанням *MOG* і *KCF*. Кожен із цих алгоритмів має свою область для переважного використання. Алгоритм *MOG* є ефективним в умовах відеоспостереження за допомогою стаціонарної відеокамери при порівняно малих змінах довкілля, пов'язаних із рухом навколишніх об'єктів (наприклад, пішоходів або різних транспортних засобів), а також таких, що зумовлені змінами освітлення й атмосферними умовами (коливання дерев, опади, тощо). Цей алгоритм забезпечує по-

a *b* *c*

Рис. 3. Приклади: *a* — кадрів відео; *b* — відповідних різницевих зображень; *c* — результатів простеження об'єктів алгоритмом *MOG*

рівняно високу швидкість обробки, зазвичай не потребує ініціалізації за допомогою оператора, а також є порівняно стійким до змін швидкості руху, масштабу й орієнтації об'єктів.

На рис. 3 наведено приклади зображень кадрів відео (стаціонарна відеокамера), відповідних їм різницевих зображень, а також об'єктів, що простежувалися за допомогою *MOG*. Розпізнавання пішоходів на цих зображеннях здійснювалося за допомогою *SVM* алгоритму з використанням *HOG* ознак (гістограми напрямів градієнтів на зображенні) [10]. У разі перетинання двох пішоходів на зображенні розмежування їх здійснювалося за допомогою додаткової попередньої обробки різницевого зображення. Надалі результати сегментації простежених об'єктів, що перетинаються, можна буде покращити завдяки використанню додаткових даних про напрями руху цих об'єктів у попередніх кадрах.

Алгоритм *KCF* може бути використаний для простеження об'єктів у відео, якщо зміна швидкості їхнього руху, масштабу (розмірів) та орієнтації відбувається достатньо повільно, а також якщо ці об'єкти не надто сильно перекриваються іншими об'єктами та не зникають за межі поля зору відеоспостереження.

Типовий приклад, що задовольняє ці вимоги, є простеження транспортних засобів на дорогах завдяки відеоспостереженню на значній відстані. Прикладами, що не задовольняють ці вимоги, можуть бути відеоспостере-

a *b* *c*

Рис. 4. Приклади кадрів відео: *a* — відповідних різницевих зображень; *b* — результатів простеження об'єктів; *c* — за спільного використання алгоритмів *MOG* та *KCF*

ження об'єктів в умовах відносно малої відстані (різкі зміни масштабу), а також порівняно швидких пересувань або різких змін напряму цього пересування.

Алгоритм *MOG_KCF* зі спільним використанням *MOG* і *KCF* має ширші можливості для свого використання порівняно з алгоритмом *KCF*. На рис. 4 наведено приклади зображень кадрів відео, відповідних їм різницевих зображень, а також об'єктів, що простежувалися за допомогою цього алгоритму.

На рис. 4 тільки одне відео (нижній рядок) знято за допомогою відеокамери з фіксованою (стаціонарною) позицією, а два інші — за допомогою нестаціонарної відеокамери. Результати простеження об'єктів на перших трьох відео на цьому рисунку за допомогою алгоритму *MOG_KCF* є суттєво кращими порівняно з алгоритмами *MOG* та *KCF*.

Алгоритм *MOG* не може бути використаним для простеження об'єктів у відео на рис. 4 внаслідок значного рівня шуму на різницевих зображеннях (використання відеокамери, що рухається, зміни освітлення в процесі зйомки), а також пересування одних об'єктів на фоні інших рухомих об'єктів. Виявлені недоліки алгоритму *KCF* під час обробки цих відео полягають у такому.

На перших двох відео (верхній і середній рядки на рис. 4) розміри автомобілів, що пересуваються, значно змінюються у процесі руху, що ускладнює використання *KCF* для

їхнього простеження. У багатьох роботах якість простеження оцінюється за *PASCAL* критерієм [12] у вигляді:

$$\frac{|r^i \cap gr^i|}{|r^i \cup gr^i|} \geq 0{,}5, \qquad (1)$$

де r^i – визначений обмежувальний прямокутник простежуваного об'єкта у i-му кадрі, gr^i – еталонний обмежувальний прямокутник цього об'єкта, визначений за допомогою оператора. За виконання умови (1) вважається, що результат простеження є правильним і таким, що відповідає еталонним даним. Перше відео (*readteam.avi*) складається з 1918 кадрів (352x240) і супроводжується еталонними даними про обмежувальні прямокутники автомобіля, що рухається.

Головні дані, отримані на цьому відео за допомогою алгоритмів *KCF* та *MOG_KCF*, представлено далі у таблиці, де *PASCAL*(%) – частина кадрів відео у відсотках, на яких виконувалася умова (1); *Avercross* – середнє значення перекриття, що відповідає лівій частині умови (1); N_{true_pos} (N_{true_neg}) – кількість визначених прямокутників, що містять (не містять) автомобіль, що рухається; *Time* – середній час обробки одного кадру *(AMD Athlon Dual Core Processor* 2.11 ГГц).

Дані в таблиці показують, що алгоритм *MOG_KCF* за *PASCAL* критерієм більш ніж удвічі перевищує алгоритм *KCF*, мало поступаючись йому у швидкодії на цих відеоданих.

Таблиця. **Головні дані, отримані за допомогою алгоритмів *KCF* та *MOG_KCF***

Алгоритм	*KCF*	*MOG_KCF*
PASCAL(%)	43,8%	94,4%
Avercross	0,51	0,7
N_{true_pos}	1918	1918
N_{true_neg}	0	0
Time (msec)	20,1	22,4

На третьому відео залізничник рухається порівняно швидко й різко змінює напрями свого руху. Це призводить до того, що простеження з використанням алгоритму *KCF* після кожної ініціалізації здійснюється тільки протягом малого проміжку часу й після цього переривається, на відміну від алгоритму *MOG_KCF*.

Результати, отримані на відеопослідовностях з відомої бази даних *VIRAT Video Dataset*, також свідчать, що розроблений алгоритм *MOG_KCF* має переваги порівняно з *MOG* та *KCF* при простеженні різних об'єктів, що рухаються зі змінною швидкістю, а також у процесі руху можуть зупинятися, змінювати напрям руху та частково зникати з поля зору.

ВИСНОВКИ

Розроблено алгоритм та програмне забезпечення для простеження рухомих об'єктів на базі спільного використання відомих алгоритмів *MOG* та *KCF*. Результати простеження показують переваги розробленого алгоритму порівняно з *MOG* та *KCF* в умовах, коли різницеві зображення мають помірний рівень завад, а простежувані об'єкти можуть рухатися з непередбачувано різкими змінами швидкості, розміру, напряму руху й орієнтації, а також можуть зупинятися та частково ховатися за загорожею.

З'ясовано, що найскладніші умови для простеження відповідають руху об'єкта на фоні інших рухомих об'єктів із наявністю порівняно різких змін розміру та напряму руху, а також суттєвих змін умов освітлення у довкіллі, що призводять до збільшення рівня завад на різницевих зображеннях. При цьому алгоритм *MOG* не може бути ефективно використаним через значні завади на різницевих зображеннях, а алгоритм *KCF* – через різкі зміни параметрів руху об'єктів. Алгоритм *MOG_KCF* є порівняно придатнішим для простеження об'єктів, але потребує подальших покращень для ефективнішої роботи у зазначених умовах.

ЛІТЕРАТУР

1. *Rout R.*, 2013. A survey on object detection and tracking algorithms. 75 p.
2. *Yilmaz, A., Javed, O., Shah, M.* Object tracking: A survey. ACM Computing Surveys (CSUR), 38 (4), pp. 1–45.
3. *Smeulders A., Chu D., Cucchiara R., Calderara S., Dehghan A., Shah M.* Visual tracking: an experimental survey. Vol. 36, No. 7, pp. 1442–1468, 2014.
4. *Henriques J. F., Caseiro R., Martins P., Batista J.* High-speed tracking with kernelized correlation filters. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 2015. Vol. 37, № 3. P. 583–596.
5. *Zivkovic, Z., & van der Heijden, F.* Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern recognition letters. 2006. Vol. 27, № 7. P. 773–780.
6. *Grabner H., Grabner M., Bischof H.* Real-time tracking via on-line boosting. Proc. BMVC. 2006. Vol. 1. P. 1–10.
7. *Babenko B., Yang M., Belongie S.* Robust object tracking with online multiple instance learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI). 2011. Vol. 33, № 8. P. 1619–1632.
8. *Viola P., Jones M.* Rapid object detection using a boosted cascade of simple features. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). Kauai, USA, 2001. Vol. 1. P. 511–518.
9. *Kalal Z., Mikolajczyk K., Matas J.* Tracking-Learning-Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI). 2012. Vol. 34, № 7. P. 1409–1422. DOI: 10.1109/TPAMI.2011.239.
10. *Bradski G.* The OpenCV Library. Dr. Dobb's Journal of Software Tools. 2000. Vol. 25, № 11. P. 122–125.
11. *Шлезингер М.И.* Быстрая реализация одного класса линейных сверток. Теоретические и прикладные вопросы распознавания изображений. Киев: ИК АН УССР, 1991. С. 61–69.
12. *Everingham M., Gool L. J. V., Williams C. K. I., Winn J. M., Zisserman A.* The pascal visual object classes (VOC) challenge. International Journal of Computer Vision (IJCV). 2010. Vol. 88, № 2. P. 303–338. DOI: 10.1007/s11263-009-0275-4.

Надійшла 09.12.2019

REFERENCES

1. *Rout, R.*, 2013. A survey on object detection and tracking algorithms. 75 p.
2. *Yilmaz, A., Javed, O., Shah, M.* Object tracking: A survey. ACM Computing Surveys (CSUR), 38 (4), pp. 1–45. DOI: 10.1145/1177352.1177355.
3. *Smeulders, A., Chu, D., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.*, 2013. Visual tracking: an experimental survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 36 (7), pp. 1442–1468. DOI: 10.1109/TPAMI.2013.230.
4. *Henriques, J. F., Caseiro, R., Martins, P., Batista, J.*, 2015. "High-speed tracking with kernelized correlation filters". IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI). 37 (3), pp. 583–596.
5. *Zivkovic, Z., van der Heijden, F.*, 2005. "Efficient adaptive density estimation per image pixel for the task of background subtraction". Pattern recognition letters, 27 (7), pp. 773–780. DOI: 10.1016/j.patrec.2005.11.005.
6. *Grabner, H., Grabner, M., Bischof, H.*, 2006. "Real-time tracking via on-line boosting". BMVC, 1, pp. 1–10.
7. *Babenko, B., Yang, M., Belongie, S.*, 2010. "Robust object tracking with online multiple instance learning". IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 33 (8), pp. 1619–1632. DOI: 10.1109/TPAMI.2010.226.
8. *Viola, P., Jones, M.,* 2001. "Rapid object detection using a boosted cascade of simple features". IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). Kauai, USA, 1, pp. 511–518.
9. *Kalal, Z., Mikolajczyk, K., Matas, J.*, 2011. "Tracking-Learning-Detection". IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 34 (7), pp. 1409–1422. DOI: 10.1109/TPAMI.2011.239.
10. *Bradski, G.,* 2000. "The OpenCV Library". Dobb's Journal of Software Tools, 25(11), pp. 122–125.
11. *Shlezinger, M.I.*, 1991. "Bystraya realizatsiya odnogo klassa lineynykh svertok", Teoreticheskiye i prikladnyye voprosy raspoznavaniya izobrazheniy, Kyiv: IK AN USSR, pp. 61–69. (In Russian).
12. *Everingham, M., Gool, L. J. V., Williams, C. K. I., Winn, J. M., Zisserman, A.*, 2010. "The pascal visual object classes (VOC) challenge", International Journal of Computer Vision (IJCV), 88 (2), pp. 303–338. DOI: 10.1007/s11263-009-0275-4.

Received 09.12.2019

V.M. Kyyko, PhD of Eng. sc., Senior researcher,
International Research and Training Center
of Information Technologies and Systemsof the NAS and MES of Ukraine,
Glushkovave., 40, Kyiv, 03187, Ukraine,
vkiiko@gmail.com

V.V. Matsello, PhD of Eng. sc., Senior researcher,
International Research and Training Center
of Information Technologies and Systems of the NAS and MES of Ukraine,
Glushkovave., 40, Kyiv, 03187, Ukraine,
matsello@gmail.com

OBJECT TRACKING AT VIDEO MONITORING

Introduction.Algorithms for visual object tracking are observed. In realistic scenarios each tracker is not superior in handling all interfering factors such as illumination and appearance variations, occlusions, changes of motion and scale of tracked object and so on. Therewith background subtraction algorithms are effective in handling motion and scale object changes, whereas appearance based ones alternatively in appearance,illumination and camera motion changes.Given the wide variety of aspects in tracking circumstances, development of trackers with co-operative actions of background subtraction and appearance based algorithms, specifically MOG (mixture of gaussians) and KCF (kernelized correlation filters) algorithms, is desirable.

The purpose of the article is to develop online tracking algorithm on the base of co-operative applying of MOG and KCF algorithms.

Method.Proposed tracker makes use of KCF to find new position of tracked object in current frame and use of MOG to get subtractive image with subsequent correction of error object position on the base of integral representation of this image.

Results.Visual online tracking algorithm based on co-operative use of MOG and KCF algorithms is developed. Testing results prove that the algorithm is more stable in comparison with KCF in the case of abrupt changes of tracked object motion speed or direction. The algorithm is also more resistant to noise and illumination changes in comparison with MOG algorithm.

Conclusions.The tracker handles appearance and scale changes of object if it does not move with other moving objects in the background and illumination variations are not large. When this occurs tracking is based on KCF algorithm.Further

В.М. Кийко, канд. техн. наук, ст. научн. сотр., ст. научн. сотр.,
Международный научно-учебный центр
информационных технологий и систем НАН и МОН Украины,
просп. Академика Глушкова, 40, Киев 03187, Украина,
vkiiko@gmail.com

В.В. Мацелло, канд. техн. наук, ст. научн., сотр., зав. отделом,
Международный научно-учебный центр
информационных технологий и систем НАН и МОН Украины,
просп. Академика Глушкова, 40, Киев 03187, Украина,
matsello@gmail.com

ПРОСЛЕЖИВАНИЕ ОБЪЕКТОВ ПРИ ВИДЕОНАБЛЮДЕНИИ

Введение. Рассматривается новый алгоритм прослеживания объектов в видео, основанный на совместном использовании алгоритмов *MOG* (*mixture of gaussians*) и *KCF* (*kernelized correlation filters*). Алгоритмы прослеживания на основе вычитания фона *MOG* и представление объектов *KCF* по своим возможностям противостоять помехам таковы, что дополняют друг друга. Вот почему задача разработки алгоритма, который сочетает преимущества этих алгоритмов, является актуальной.

Целью работы является исследование алгоритма прослеживания объектов в видео, основанного на совместном использовании алгоритмов *MOG* и *KCF*.

Метод. Прослеживание объектов в видео использует *KCF* для нахождения нового положения объекта на текущем изображении и *MOG* для получения разностного изображения с целью последующей коррекции координат и размера объекта с помощью интегрального представления этого изображения.

Результаты. Предложен алгоритм прослеживания объектов в видео, основанный на совместном использовании *MOG* и KCF. Приведены результаты поиска и прослеживания движущихся объектов в видео. Показано, что алгоритм является более эффективным по сравнению с *MOG* и *KCF* в условиях, когда объекты, которые прослеживаются, движутся со сравнительно резкими изменениями скорости, направления движения и ориентации, а также в условиях частичного укрытия или исчезновения из поля зрения. Алгоритм является также более устойчивым к помехам и изменениям освещения по сравнению с *MOG* алгоритмом.

Выводы. Прослеживание с совместным использованием *MOG* и *KCF* является устойчивым к изменениям представления и размерам объекта, если он движется на фоне другого подвижного объекта, а также изменения освещения сравнительно умеренны. В противном случае более подходящим может быть прослеживание на основе *KCF* алгоритма. Для дальнейшего улучшения результатов в таких условиях необходимо исследовать использование дополнительных признаков прослеживаемого объекта.

Ключевые слова: *видеонаблюдение, поиск и прослеживание объектов, разностное изображение, обработка изображений*.