УДК 004.932.2

Варфоломєєв А. Ю.

АЛГОРИТМ ДЕТЕКТУВАННЯ РУХУ НА ОСНОВІ КОРЕЛЯЦІЙНОГО СПІВСТАВЛЕННЯ ІЗ ЗАПОБІГАННЯМ СТРОБОСКОПІЧНОГО ЕФЕКТУ

Національний технічний університет України «КПІ»

Запропоновано детектор руху на основі алгоритмів кореляційного порівняння зображень та відновлення фону із запобіганням стробоскопічного ефекту для стаціонарних систем спостереження. Проведено ряд експериментальних досліджень, в ході яких підтверджено ефективність запропонованого підходу з огляду на швидкодію та надійність.

Вступ

Алгоритми детектування руху знаходять широке застосування у системах відеоспостереження і безпеки, відеоаналітиці, робототехніці та інших галузях технічного зору. Водночас, створення дійсно надійного та швидкодіючого способу детектування не є тривіальною задачею. Це головним чином обумовлено складністю пристосування до складних робочих умов: різких змін освітлення [1-3], коливань камери спостереження та фонових об'єктів (наприклад, дерев під час вітру) [4, 5], захоплення разом із рухомими об'єктами їх тіней та ін. Означені питання в роботах [1-5] вирішуються лише частково, тому детектування руху на відеопослідовностях досі залишається складною та актуальною для дослідження проблемою.

Таким чином, мета даної роботи – запропонувати альтернативний простий та швидкодіючий, але водночас надійний підхід детектування рухомих об'єктів на відеопослідовностях. Даний підхід оснований на відновленні фонового зображення (зображення, що містить тільки нерухомі об'єкти), яке далі порівнюється із кожним кадром відеопотоку. В роботі вирішуються наступні задачі: вибір та обґрунтування алгоритму порівняння зображень, розробка надійного алгоритму відновлення фону, а також проведення експериментального дослідження алгоритму на реальних відеопослідовностях та його порівняльний аналіз із сучасним підходом наведеним в роботі [5].

Алгоритм порівняння зображень

Під порівнянням зображень будемо розуміти процедуру, яка дає можливість встановити в яких точках два зображення відрізняються, а в яких – є подібними.

У якості метода порівняння зображень вирішено використовувати нормалізовану кореляцію. Даний метод має ряд переваг, що полягають у його малій чутливості до зміни яскравості зображення, можливості швидкого обчислення та простоті реалізації. В загальному випадку рівняння для обчислення нормалізованої кореляції має вигляд [6]:

$$C_n(u,v)=\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}[f(u+i,v+j)\cdot t(i,j)]}{\sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N}f^2(u+i,v+j)\cdot\sum_{i=1}^{M}\sum_{j=1}^{N}t^2(i,j)}}, \quad (1)$$

де f – зображення, на якому здійснюється пошук шаблону; t – зображення-шаблон, розмір якого складає $M\times N$; $C_n(u,v)\in[0;1]$.

Для порівняння двох зображень здійснюється їхнє розбиття на блоки, між відповідними парами яких обчислюється нормалізована кореляція C_n. Якщо значення C_n менше деякого порогу T, вважається, що блоки мають різні текстури, і зображення у відповідних точках відрізняються (рис 1).

Ця процедура дуже подібна до підходів оцінювання руху, яка застосовуються в багатьох алгоритмах стиснення відеоінформації, за виключенням того, що в даному випадку немає необхідності знаходити напрямок та швидкість руху в межах блоків. Крім того, з метою отримання більшої роздільної здатності під час порівняння, блоки можуть накладатись один на одного. Оскільки, розглядається випадок стаціонарної (нерухомої) камери спостереження, фонове зображення від кадру до кадру не матиме значних відхилень, а отже при порівнянні непотрібно виконувати перевірку із різними зміщеннями, як це здійснюється при реєстрації зображень. Натомість можна виконувати обчислення тільки для безпосереднього суміщення блоків. Це дає можливість значно зменшити кількість операцій при знаходженні нормалізованої кореляції і користу-

ватися так званими інтегральними зображеннями для швидкого обчислення сум у чисельнику і знаменнику виразу (1).

Для повноти викладення наведемо пояснення щодо використання інтегральних зображень. Інтегральне зображення (таблиця підсумовування площі) є масивом, кожен елемент якого містять суму значень пікселів вихідного зображення у прямокутнику, обмеженому лівим верхнім кутом та координатами (x, y) даного елементу [7] (рис. 2 *а*).

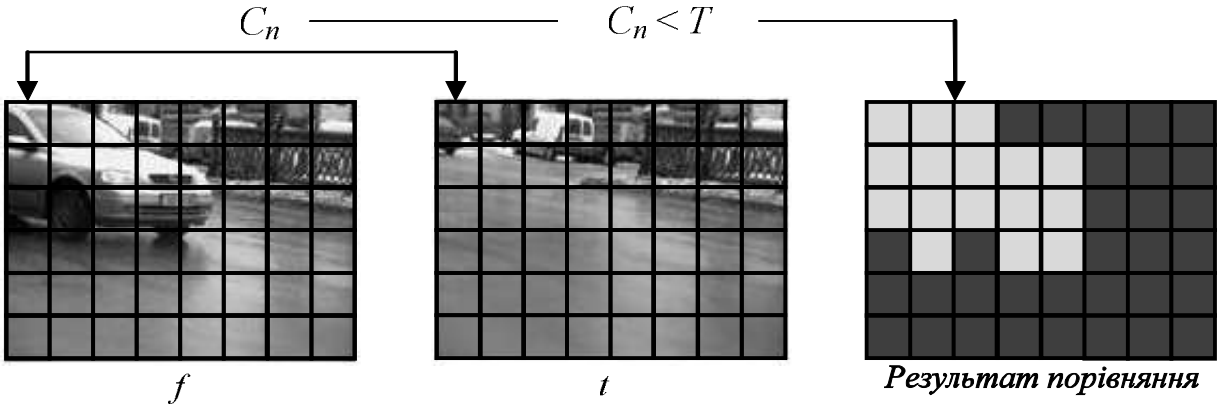


Рис. 1. Порівняння зображень: світлі точки результату порівняння – це області, де зображення відрізняються, темні – області, в яких зображення однакові

Сума пікселів у будь-якому прямокутнику вихідного зображення може бути знайдена за чотири звернення до інтегрального зображення. Наприклад, для знаходження суми пікселів у чорному прямокутнику на рис. 2 *б* спочатку обчислюється сума прямокутної області від (0; 0) до $(x_R; y_B)$. Потім від неї віднімаються суми областей від (0; 0) до $(x_R; y_T)$ та від (0; 0) до $(x_L; y_B)$. Оскільки сірий прямокутник (від (0; 0) до $(x_L; y_T)$) в ході таких дій був віднятий двічі, для отримання остаточного результату необхідно його додати.

Таким чином, при відомому інтегральному зображенні *ii* суму пікселів у чорному прямокутнику (рис. 2 *б*) можна знайти наступним чином:

$$\Sigma = ii(x_R; y_B) - ii(x_R; y_T) - \\ - ii(x_L; y_B) + ii(x_L; y_T) \quad . \qquad (2)$$

Викладений вище, алгоритм порівняння зображень узагальнено в табл. 1.

В табл. 1 знак $\otimes$ позначає поелементне множення масивів. Зображення *f*, *t* мають однакову розмірність, що складає $M \times N$.

Відновлення фонового зображення

Процес відновлення фонового (опірного) зображення є надзвичайно важливою частиною детектора руху, оскільки якість фону чинить безпосередній вплив на загальну надійність та точність алгоритму виявлення. У швидкодіючих детекторах руху для відновлення фону найчастіше використовуються альфа-змішування або медіана, обчислена по заданій кількості кадрів [8]. Зазначені підходи мають суттєвий недолік: наявність рухомих об'єктів перед камерою в момент відновлення призводить до значних спотворень опірного зображення, які проявляються як нечіткості, «розводи» та фрагменти рухомих об'єктів (рис. 3 *б*, *в*).

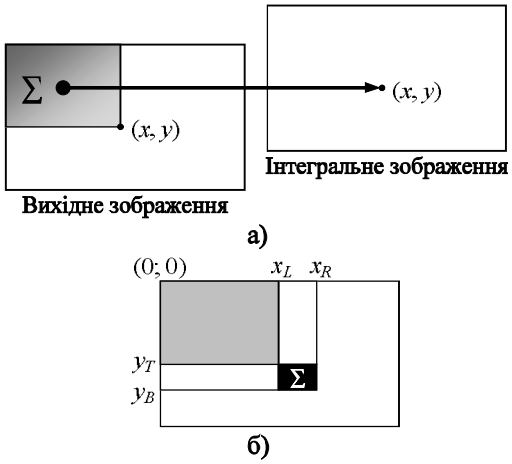


Рис. 2. Інтегральне зображення:
а) спосіб формування;
б) пошук суми в заданому прямокутнику по заданому інтегральному зображенню

Таблиця 1.

Алгоритм порівняння зображень

Вхід:
f, t – зображення, що порівнюються;
a – розмір блоку;
b – відстань між суміжними блоками;
T – поріг порівняння.

Вихід:
r – бінарне зображення, що є результатом порівняння та мітить «1» в тих точках, в яких f і t відрізняються.

$c = f \otimes t$;
$iic = \text{IntegralImage}(c)$;
$iif^2 = \text{IntegralImage}(f \otimes f)$;
$iit^2 = \text{IntegralImage}(t \otimes t)$;
while ($i < M - a$)
  while ($j < N - a$)
    $num = iic_{i+a,j+a} - iic_{i+a,j} - iic_{i,j+a} + iic_{i,j}$;
    $den = (iif^2_{i+a,j+a} - iif^2_{i+a,j} - iif^2_{i,j+a} + iif^2_{i,j}) \cdot$
      $\cdot (iit^2_{i+a,j+a} - iit^2_{i+a,j} - iit^2_{i,j+a} + iit^2_{i,j})$;
    $r(i/b, j/b) = \begin{cases} 1, & \frac{num}{\sqrt{den}} < T \\ 0, & \text{в іншому випадку} \end{cases}$;
    $j = j + b$;
  end
  $i = i + b$;
end

З метою уникнення вказаного недоліку пропонується використовувати алгоритм відновлення фону на основі порівняння кадрів. Його суть полягає у наступному: із відеопослідовності з досить тривалим періодом беруться кадри і порівнюються між собою, якщо деякі області кадрів виявляються подібними, то можна вважати, що вони належать фону. Наприклад, при порівнянні двох кадрів f і t (рис. 1), областями, що належать фону можна вважати ті, які на результуючому зображені показані темним. Описаний процес повторюється доти, доки опірне зображення не буде повністю відновлене (рис. 3 *г*).

Під час відновлення фону можлива ситуація, за якої рухаючись подібні об'єкти опинятимуться в одному й тому ж місці з інтервалом часу рівним періоду порівняння кадрів, викликаючи стробоскопічний ефект. Очевидно, це призводитиме до невірної реконструкції опірного зображення. Щоб цьому запобігти, доцільно виконувати порівняння кадрів по декільком періодам, а потім поєднувати результати (для поєднання можна, наприклад, використовувати операцією перетину множин).

Запропонований алгоритм відновлення фону наведено в табл. 2.

Таблиця 2.

Алгоритм відновлення фону

Вхід:
F – вектор (потік) кадрів;
P – вектор періодів;
K – кількість елементів у векторі **P**.

Вихід:
bg – відновлений фон.

$i = 0$;
$\mathbf{F}_0^{prev} = \mathbf{0}$;
do
  $j = 0$;
  while ($j < K$)
    if (mod(i, $\mathbf{P}_j$) == 0)
      $\mathbf{R}_j = \text{imCmp}(\mathbf{F}_i, \mathbf{F}_j^{prev})$;
      $\mathbf{F}_j^{prev} = \mathbf{F}_i$;
    end
    $j = j + 1$;
  end
  $m = \text{formMask}(\mathbf{R}_{\{0,\ldots,K-1\}})$;
  $bg = \{f \mid m == 0\}$;
  $i = i + 1$;
while (bg is not recovered)

У табл. 2 функція *mod* повертає остачу від ділення; функція *imCmp* повертає множину точок, у яких передані їй зображення відрізняються (дану функцію може реалізовувати алгоритм описаний в попередньому розділі); функція *formMask* обробляє вектор множин $\mathbf{R}_{\{0,\ldots,K-1\}}$, об'єднуючи результати порівняння по періодам і формує маску елементів фону (у якості цієї функції в найпростішому випадку може виступати оператор логічного «*І*», що застосовується до всіх відповідних точок множин $\mathbf{R}_j$, $j = 0,\ldots,K-1$).

Результати експериментальних досліджень

На основі описаних в попередньому розділі алгоритмах порівняння зображень та відновлення фону було створено та випробувано систему детектування руху. Тестування проводилось на комп'ютері з процесором *Intel Core Duo* 1,83 ГГц, кеш першого рівня *L*1 32 Кб, другого – *L*2 2 Мб, ємність ОЗУ 1024 Мб. При цьому, було використано дві відеопослідовності, одна з яких взята із бази проекту *CAVIAR* [9] (рис. 4 *а*), а друга – є записом руху автомобілів по трасі в реальних умовах (рис. 4 *б*).

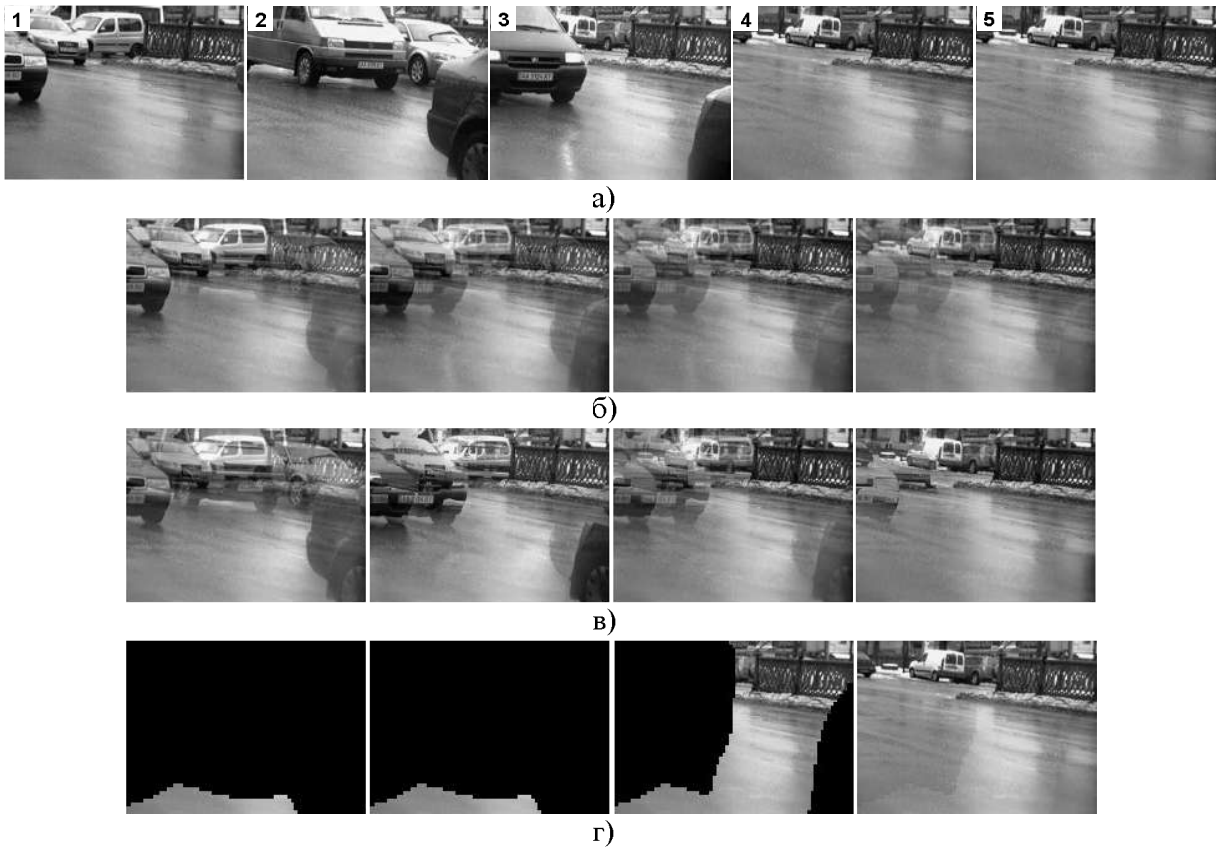


Рис. 3. Процес відновлення фону: а) кадри, по яким здійснюється відновлення; б) відновлення за допомогою альфа-змішування ($\alpha = 0{,}3$); в) відновлення на основі медіани; г) відновлення за допомогою запропонованого алгоритму (для альфа-змішування та медіани результати показані після обробки 2, 3, 4 та 5 кадрів відповідно; для запропонованого алгоритму результат отримано після порівнянь кадрів 1 і 2, 2 і 3, 3 і 4 та 4 і 5)

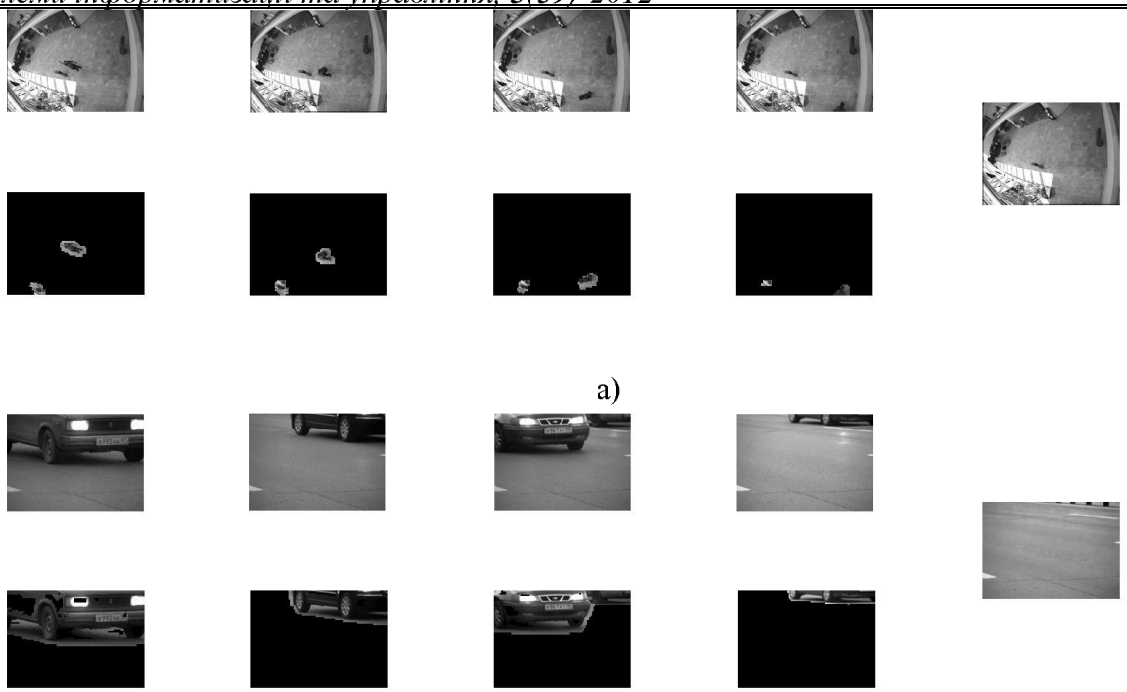
Зазначимо, що послідовність, що містить рух автомобілів є більш складною, оскільки через зміну освітлення та автоматичне налаштування камерою балансу білого виявляти нерухомі об'єкти стає досить проблематично. Більш того, постійний потік машин та хитання камери суттєво ускладнюють процес відновлення фону. Проте як видно з рис. 4, запропонована система справляється із даною відеопослідовністю доволі успішно. Можна бачити також, що спосіб порівняння зображень, розглянутий в попередньому розділі, дозволяє подавлювати незначні тіні та відблиски.

Це досягається головним чином за рахунок інваріантності формули (1) до змін яскравості. В той же час нечутливість до змін яскравості в деяких випадках може призводити до небажаних ефектів – різні за яскравістю, але однорідні області розцінюватимуться алгоритмом як схожі між собою. Це добре видно на першому кадрі рис. 4 *б*, де бічна частина автомобіля та частина фари розпізнана як фон. Зазначений ефект, однак, не є критичним, оскільки з легкістю усувається за допомогою морфологічного заповнення.

З метою оцінювання швидкодії, запропонований детектор руху був реалізований із використанням високоефективних функцій бібліотеки *OpenCV v.* 2.1. На тестовому комп'ютері йому вдалося досягти затримки на обробку кадру, що не перевищує 11 мс у випадку роботи із відеопотоком роздільною здатністю 384×272 пікселів. При цьому, використовувались наступні налаштування:

- для алгоритму порівняння зображень $a = 20$, $b = 4$;
- для алгоритму відновлення фону $\mathbf{P} = \{17, 29, 37\}$ і відповідно $K = 4$.

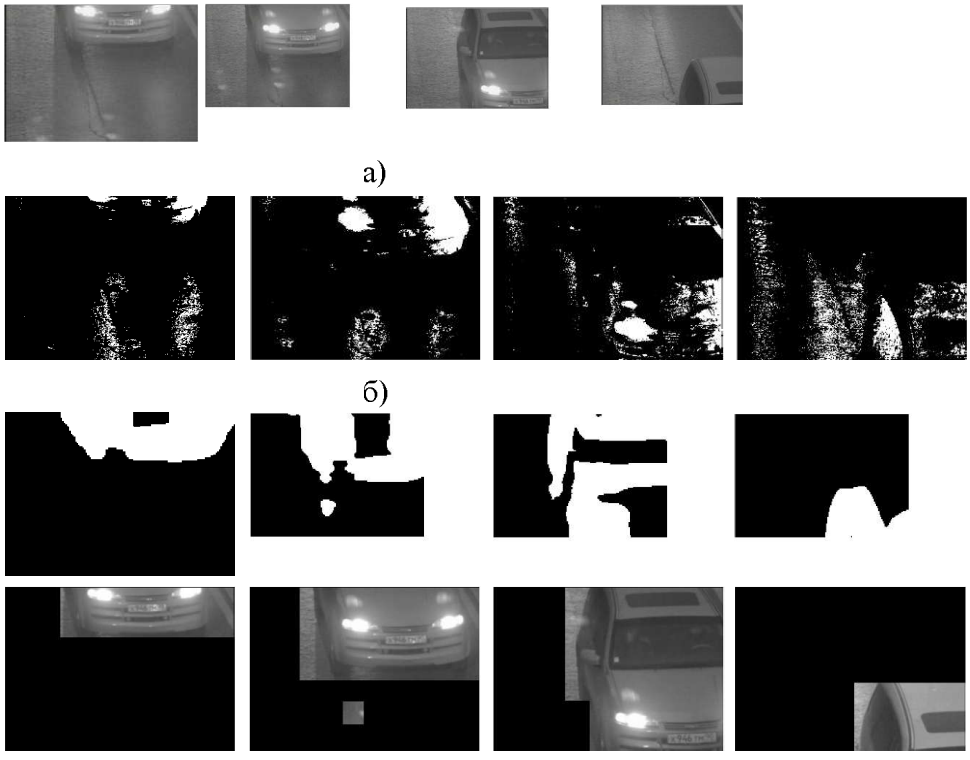
Для порівняння: алгоритм виявлення руху, оснований на моделюванні фону за допомогою розподілів Гауса [5], реалізований в бібліотеці *OpenCV*, на тих самих тестових комп'ютері та відеопослідовності здійснює обробку кадру із затримкою щонайменше у 22 мс. Крім того, детектор на снові моделювання фону є менш надійним, особливо у випадку різких змін освітлення (рис. 5).



а)

б)

Рис. 4. Приклад роботи пропонованої системи виявлення руху:
а) детектування руху людей у кадрі б) детектування руху машин
(у верхніх рядках показані вихідні кадри, а у нижніх – виділені рухомі об'єкти; справа кожної із послідовностей показано відновлений системою фон)



а)

б)

в)

Рис. 5. Порівняння якості детекторів руху за умови зміни освітлення: а) вхідні кадри;
б) результат отриманий детектором на основі моделювання фону розподілами Гауса;
в) результат отриманий запропонованим детектором (нижній рядок – фрагменти вхідних кадрів,
які потрапляють до прямокутників описаних навколо рухомих областей).

Висновки

Запропоновано алгоритм детектування руху на основі кореляційного порівняння зображень та відновлення фону із запобіганням стробоскопічного ефекту. На ряді відеопослідовностей експериментально підтверджено ефективність запропонованого підходу. При цьому, створений детектор руху виявився одночасно надійним та швидкодіючим, може працювати в складних умовах – за наявності змін освітлення, безперервного руху об'єктів перед камерою та незначних коливань камери спостереження.

Таким чином, зазначені особливості запропонованого детектора руху роблять його доцільним для використання в системах відеоспостереження загального призначення, а особливо, з огляду на швидкодію, в системах, що створені на базі вбудовуваних обчислювальних засобів, таких, зокрема, як *DSP* процесори та системи на кристалі (*SoC*).

Список літератури

1. KaewTraKulPong P. An Improved Adaptive Background Mixture Model for Real-time Tracking with Shadow Detection / KaewTraKulPong P., Bowden R. // European Workshop on Advanced Video Based Surveillance Systems, – 2001. – P. 149-158.

2. Porikli F. Bayesian background modelling for foreground detection / Porikli F., Tuzel O. // Proc. of ACM Visual Surveillance and Sensor Network. 2005. – P. 55-58.

3. Porikli F. Multiplicative Background-Foreground Estimation under Uncontrolled Illumination using Intrinsic Images // IEEE Workshop on Motion and Video Computing. – 2005. – Vol. 2. – P. 20- 27.

4. Porikli F. Change Detection by Frequency Decomposition: Wave-Back / Porikli F., Wren C. R. // Workshop on Image Analysis for Multimedia Interactive Services. – April, 2005. – P. 245-248.

5. Toyama K. Wallflower: Principles and Practice of Background Maintenance. / Toyama K., Krumm J., Brumitt B., Meyers B. // International Conference on Computer Vision, 1999. – Vol. 1. – P. 255-261.

6. Претт У. Цифровая обработка изображений / Претт У. [Пер. с англ]. – М.: Мир, 1982, Кн. 2. – 480 с.

7. Viola P. Robust real-time face detection / Viola P., Jones M. // Intl. J. of Computer Vision. – 2004. – № 52 (2). – P. 137-154.

8. Porikli F. Achieving Real-Time Object Detection and Tracking Under Extreme Conditions // Journal of Real-Time Image Processing, Springer – 2006. – Vol. 1, № 1. – P. 33-40.

9. CAVIAR Test Case Scenarios [Електронний ресурс]. – Режим доступу: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/.