УДК 004.932.72 **doi: 10.26906/SUNZ.2019.4.072**

О. М. Маковейчук[1], І. В. Рубан[1], Г. В. Худов[2]

[1] Харківський національний університет радіоелектроніки, Харків, Україна
[2] Харківський національний університет Повітряних Сил імені Івана Кожедуба, Харків, Україна

ВИКОРИСТАННЯ ГЕНЕТИЧНИХ АЛГОРИТМІВ ДЛЯ ЗНАХОДЖЕННЯ ІНВЕРСНИХ ПСЕВДОВИПАДКОВИХ БЛОЧНИХ ПЕРЕСТАНОВОК

Предметом вивчення в статті є метод знаходження інверсних псевдовипадкових блочних перестановок пікселів у зображенні. **Метою** є розробка "сліпого"методу знаходження інверсних псевдовипадкових блочних перестановок за допомогою генетичних алгоритмів. **Завдання:** провести аналіз факторів, що впливають на інверсні псевдовипадкові блочні перестановки на зображенні, розробити метод кодування перестановок в генетичних алгоритмах, обґрунтувати вибір цільової функції для оптимізації за допомогою генетичних алгоритмів. Використовуваними **методами** є: методи цифрової обробки зображень, теорії ймовірності, математичної статистики, криптографії та захисту інформації, математичний апарат теорії матриць. Отримані такі **результати.** Проведено аналіз факторів, що впливають на інверсні псевдовипадкові блочні перестановки на зображенні. Визначено фактори, що впливають на максимальний розмір блоку, при якому ще можливе знаходження інверсної перестановки. Розроблено метод знаходження інверсних псевдовипадкових блочних перестановок пікселів у пермутованому зображенні за допомогою генетичних алгоритмів. **Висновки.** Наукова новизна отриманих результатів полягає в наступному. Встановлено, що знаходження інверсних перестановок можливе лише при умові, що розмір блоку є менший за радіус кореляції зображення. Запропоновано ефективний спосіб кодування перестановок, при якому стандартні оператори генетичних алгоритмів будуть породжувати нові і тільки допустимі перестановки. Запропоновано у якості цільової функції використовувати суму квадратів градієнтів. Показано, що дана цільова функція має глобальний мінімум для коректної перестановки, що дозволяє знаходити інверсні блочні перестановки "всліпу" без додаткової апріорної інформації.

Ключові слова: генетичні алгоритми, інверсна псевдовипадкова блочна перестановка, пермутовані зображення, (псевдо)голографічне кодування, радіус кореляції.

I have never encountered any problem where genetic algorithms seemed to me the right way to attack it. Further, I have never seen any computational results reported using genetic algorithms that have favorably impressed me.

Я особисто ніколи не стикався ні з одним завданням, для вирішення якого генетичні алгоритми виявилися б найкращим засобом. Більш того, я ніколи не зустрічав ніяких результатів обчислень, отриманих за допомогою генетичних алгоритмів, які б справляли на мене позитивне враження.

Steven Skiena [1, p. 267]

Вступ

Постановка проблеми у загальному вигляді. Автори спробують довести, що скепсис твердження, що було винесено в епіграф, є дещо передчасним. У даний час у зв'язку з широким розповсюдженням цифрового медіаконтенту актуальним є питання захисту інформації від спотворення, незаконного використання тощо.

В ряді публікацій (див., наприклад, роботи [2-5] і цитовану в них літературу) пропонується з криптографічною метою використовувати блочні псевдовипадкові перестановки пікселів. При цьому вважається, що криптографічна стійкість цих методів є дуже високою, оскільки навіть для порівняно невеликого розміру блоку 8x8 загальна кількість комбінацій є надзвичайно великою 64! ≈ 1.27e+89.

У даній роботі на основі підходящої моделі формування зображення виводяться обмеження на розмір блоку і пропонується ефективний спосіб "сліпого" (без залучення додаткової апріорної інформації) відновлення перкутованого (від англ. "permutation" – перестановка) зображення за допомогою генетичного алгоритму. Вхідними параметрами алгоритму є тільки розмір блоку і саме пермутоване зображення.

Аналіз останніх досліджень і публікацій. Генетичні алгоритми (ГА) належать до класу еволюційних алгоритмів і широко використовуються для вирішення оптимізаційних задач. ГА є метаевристикою, що моделює природній відбір рішень з використанням операторів мутації, схрещування та відбору [6-8].

В задачах обробки зображень і комп'ютерного зору ГА використовуються здебільшого для стеганографії [9, 10], побудови стереопар [11], сегментації зображень [12], пошуку оптимальних шляхів на графах [13] та інше [14].

Властивості пермутованих зображень (іноді у не-англомовній літературі використовується термінологія "(псевдо)голографічне кодування") досліджуються у роботах [15-25].

Існуючі методи вирішення задач відновлення пермутованих зображень базуються на еволюційних алгоритмах оптимізації, зокрема за допомогою генетичних алгоритмів [26-30].

Основним недоліком відомих методів є необхідність додаткової апріорної інформації.

Метою статті є розробка ефективного методу "сліпого"визначення інверсних псевдовипадкових блочних перестановок на зображеннях на основі генетичних алгоритмів.

Постановка задачі та викладення матеріалів дослідження

Нехай f – вихідне зображення, з якого за допомогою блочної перестановки пікселів (псевдовипад-

кової, але однакової для кожного блоку) отримується пермутоване зображення *g*. Тобто вихідне зображення *f* розбивається на квадратні блоки розміром *b*x*b*y кожному з яких пікселі переставляються однаковим чином. Схематично це можна записати як

$$g^{(n)} = f^{(n)}(p), \tag{1}$$

де *n* нумерує блок, *p* – задана перестановка (вектор індексів довжиною b^2 елементів).

Необхідно знайти перестановку *p* і відновити вихідне зображення *f*, знаючи тільки пермутоване зображення *g* і розмір блоку *b*.

Для вирішення указаної задачі будемо використовувати ГА.

Існуючі імплементації ГА найбільш пристосовані до представлення генома як вектора дійсних чисел, тому необхідно знайти ефективний спосіб кодування перестановок (які природнім чином представляють собою вектор індексів), такий що основні оператори ГА (мутація, схрещування тощо) породжували інші допустимі перестановки, що рівномірно покривають область допустимих значень. Ця задача не є тривіальною.

В роботі при розробці способу кодування перестановок в геномі буде запропоновано використовувати підхід, що наведений у MATLAB в імплементації функції randperm [31], а саме кодувати геном як вектор дійсних чисел, сортувати їх за зростанням, отримуючи вектор індексів розташування елементів у відсортованому масиві, який і задає перестановку.

Легко бачити, що при такому способі кодування перестановок стандартні оператори ГА будуть породжувати нові і тільки допустимі перестановки. Обчислювальна складність цього способу, очевидно, така ж сама як й у алгоритмів сортування $\sim O\left(N\log(N)\right)$, де $N = b^2$ – довжина генома (кількість змінних у задачі).

Для вибору цільової функції скористаємось тим, що псевдовипадкові перестановки роблять сусідні пікселі зображення некорельованими, отже середня різниця яскравості для пермутованого зображення зросте. Тому підходящою буде будь-яка неспадаюча функція від модуля градієнта зображення $|\nabla g|$, наприклад,

$$S = \frac{1}{4}\left\langle |\nabla g|^2 \right\rangle, \tag{2}$$

де трикутні дужки $\langle ... \rangle$ позначають операцію усереднення, а коефіцієнт 1/4 введено з міркувань зручності.

Представимо функцію, що задається рівнянням (1) у явному вигляді:

$$S = \frac{1}{4\cdot w\cdot h}\sum_x\sum_y\begin{pmatrix}\left(g_{x+1,y}-g_{x,y}\right)^2+\\ +\left(g_{x,y+1}-g_{x,y}\right)^2\end{pmatrix}, \tag{3}$$

де *w*, *h* – довжина і ширина зображення відповідно, а сумування проводиться по всіх координатах *x* та *y*.

Враховуючи, що

$$\begin{aligned}\left\langle |\nabla g|^2 \right\rangle &= \left\langle |\nabla_x g|^2 + |\nabla_y g|^2 \right\rangle = \\ &= \left\langle |\nabla_x g|^2 \right\rangle + \left\langle |\nabla_y g|^2 \right\rangle,\end{aligned} \tag{4}$$

і, що з міркувань відсутності виділеного напрямку у пермутованому зображені очевидно, що

$$\left\langle |\nabla g|^2 \right\rangle = \left\langle |\nabla_y g|^2 \right\rangle, \tag{5}$$

то достатньо розглянути тільки

$$\left\langle |\nabla_x g|^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y\left(g_{x+1,y}-g_{x,y}\right)^2. \tag{6}$$

Розпишемо вираз (6):

$$\left\langle |\nabla_x g|^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y\begin{pmatrix}g_{x+1,y}^2+g_{x,y}^2-\\ -2g_{x+1,y}\cdot g_{x,y}\end{pmatrix}, \tag{7}$$

і, усереднюючи вираз у дужках, врахуємо що для некорельованих пікселів середні квадратів співпадають:

$$\frac{1}{w\cdot h}\sum_x\sum_y g_{x+1,y}^2 = \frac{1}{w\cdot h}\sum_x\sum_y g_{x,y}^2 = \left\langle g^2 \right\rangle \tag{8}$$

а середні добутків – це є просто добутки середніх:

$$\frac{1}{w\cdot h}\sum_x\sum_y g_{x+1,y}g_{x,y} = \langle g\rangle^2, \tag{9}$$

Таким чином,

$$\left\langle |\nabla_x g|^2 \right\rangle = \left\langle |\nabla_y g|^2 \right\rangle = 2\left\langle g^2 \right\rangle - 2\langle g\rangle^2 = 2D(g), \tag{10}$$

де *D(g)* – дисперсія зображення *g*, що розраховується стандартним чином. Отже, маємо

$$\frac{1}{4}\left\langle |\nabla g|^2 \right\rangle = \frac{1}{4}\left\langle |\nabla_x g|^2 \right\rangle + \frac{1}{4}\left\langle |\nabla_y g|^2 \right\rangle = D(g), \tag{11}$$

оскільки дисперсія (як і середнє значення) не залежать від порядку відліків, то

$$D(g) = D(f) = \mathrm{D}_0. \tag{12}$$

Таким чином, у початковий момент часу *t*=0 значення цільової функції дорівнює дисперсії зображення

$$S(t=0) = \mathrm{D}_0. \tag{13}$$

Виникає питання – наскільки зміниться значення цільової функції у процесі оптимізації? Зрозуміло, що у випадку збіжності алгоритму, у певний момент часуt_c будемо мати:

$$S(t=t_c) = \frac{1}{4}\left\langle |\nabla f|^2 \right\rangle, \tag{14}$$

тому для відповіді на поставлене питання необхідно оцінити величини *D(g)=D(f)* і значення $\left\langle |\nabla f|^2 \right\rangle$, для оцінки якого будемо вважати, що вихідне (непермутоване) зображення *f* є реалізацією двовимірного

випадкового процесу [32] і, як буде показано нижче, функція розподілу цього процесу може бути довільною. У найпростішому випадку зображення f формується в результаті згортки зображення r, що складається з рівномірно розподілених на інтервалі [0, 1] випадкових пікселів, з лінійним усереднюючим фільтром u з квадратною апертурою розміру a x a

$$f = r * u, \tag{15}$$

або
$$f_{x,y} = \frac{1}{a^2}\sum_{i=1}^{a}\sum_{j=1}^{a} r_{x+i,y+j}, \tag{16}$$

Розрахуємо спочатку $\left\langle \left|\nabla_x f\right|^2 \right\rangle$. Аналогічно до формули (6)

$$\left\langle \left|\nabla_x f\right|^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y \left(f_{x+1,y} - f_{x,y}\right)^2, \tag{17}$$

З урахуванням (16) маємо, що

$$\left\langle \left|\nabla_x f\right|^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y \left(\begin{array}{l} \frac{1}{a^2}\sum_{i=1}^{a}\sum_{j=1}^{a} r_{x+1+i,y+j} - \\ -\frac{1}{a^2}\sum_{i=1}^{a}\sum_{j=1}^{a} r_{x+i,y+j} \end{array} \right)^2, \tag{18}$$

або

$$\left\langle \left|\nabla_x f\right|^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y \left(\frac{1}{a^2}\sum_{i=1}^{a}\sum_{j=1}^{a} \left(\begin{array}{l} r_{x+1+i,y+j} - \\ -\, r_{x+i,y+j} \end{array} \right) \right)^2, \tag{19}$$

різниця у дужках легко розписується як

$$\sum_{i=1}^{a}\sum_{j=1}^{a} \left(\begin{array}{l} r_{x+1+i,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right) = \sum_{j=1}^{a} \left(\begin{array}{l} r_{x+a+1,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right). \tag{20}$$

Квадрат цієї суми в свою чергу можна представити як

$$\begin{aligned} \left(\sum_{j=1}^{a} \left(\begin{array}{l} r_{x+a+1,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right) \right)^2 &= \sum_{j=1}^{a} \left(\begin{array}{l} r_{x+a+1,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right)^2 + \\ &+ \sum_{j\neq j'}^{a} \left(\begin{array}{l} r_{x+a+1,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right) \left(\begin{array}{l} r_{x+a+1,y+j'} - \\ -\, r_{x+1,y+j'} \end{array} \right). \end{aligned} \tag{21}$$

Усереднюючи, маємо

$$\begin{aligned} &\left\langle \sum_{j=1}^{a} \left(r_{x+a+1,y+j} - r_{x+1,y+j} \right)^2 \right\rangle = \\ &= 2\mathrm{a}\left(\left\langle r^2 \right\rangle - \left\langle r \right\rangle^2 \right) = 2\,\mathrm{aD}(r), \end{aligned} \tag{22}$$

$$\begin{aligned} &\left\langle \sum_{j\neq j'}^{a} \left(\begin{array}{l} r_{x+a+1,y+j} - \\ -\, r_{x+1,y+j} \end{array} \right) \left(\begin{array}{l} r_{x+a+1,y+j'} - \\ -\, r_{x+1,y+j'} \end{array} \right) \right\rangle = \\ &= a(a-1)\left(\left\langle r^2 \right\rangle - \left\langle r \right\rangle^2 - \left\langle r \right\rangle^2 + \left\langle r \right\rangle^2 \right) = 0. \end{aligned} \tag{23}$$

Отже

$$\left\langle \left|\nabla_x f\right|^2 \right\rangle = \left\langle \left|\nabla_y f\right|^2 \right\rangle = \frac{1}{a^4}\cdot 2aD(r) = \frac{2D(r)}{a^3}. \tag{24}$$

і, відповідно,

$$\frac{1}{4}\left\langle \left|\nabla f\right|^2 \right\rangle = \frac{1}{4}\left\langle \left|\nabla_x f\right|^2 \right\rangle + \frac{1}{4}\left\langle \left|\nabla_y f\right|^2 \right\rangle = \frac{1}{a^3}D(r). \tag{25}$$

Зауважимо, що при a = 1 вираз (25) переходить у вираз (11), як і повинно бути. З іншого боку,

$$D(f) = \left\langle f^2 \right\rangle - \left\langle f \right\rangle^2. \tag{26}$$

Оскільки (за побудовою)

$$\left\langle u \right\rangle = 1, \tag{27}$$

то
$$\left\langle f \right\rangle = \left\langle r * u \right\rangle = \left\langle r \right\rangle. \tag{28}$$

Розрахуємо тепер

$$\left\langle f^2 \right\rangle = \frac{1}{w\cdot h}\sum_x\sum_y \left(\frac{1}{a^2}\sum_{i=1}^{a}\sum_{j=1}^{a} r_{x+1+i,\mathrm{y+j}} \right)^2, \tag{29}$$

що дає

$$\left\langle f^2 \right\rangle = \left(a^2\left\langle r^2 \right\rangle + 2\binom{a^2}{2}\left\langle r \right\rangle^2 \right) \Big/ a^4, \tag{30}$$

де
$$\binom{a^2}{2} = \frac{a^2!}{\left(a^2-2\right)!\,2!}. \tag{31}$$

Отже

$$\left\langle f^2 \right\rangle = \frac{1}{a^4}\left(a^2\left\langle r^2 \right\rangle + a^2\left(a^2-1\right)\left\langle r \right\rangle^2 \right), \tag{32}$$

збираючи подібні члени маємо:

$$\begin{aligned} \left\langle f^2 \right\rangle &= \left(a^2\left\langle r^2 \right\rangle + \left(a^4 - a^2\right)\left\langle r \right\rangle^2 \right) \Big/ a^4 = \\ &= \left(\left\langle r^2 \right\rangle - \left\langle r \right\rangle^2 \right) \Big/ a^2 + \left\langle r \right\rangle^2. \end{aligned} \tag{33}$$

Отже

$$D(f) = \frac{1}{a^2}\left(\left\langle r^2 \right\rangle - \left\langle r \right\rangle^2 \right) = \frac{1}{a^2}\mathrm{D}(r). \tag{34}$$

Таким чином маємо, що цільова функція на нульовій ітерації приймає значення:

$$S(t=0) = D(f) = \frac{1}{a^2}D(r), \tag{35}$$

а на ітерації, коли алгоритм знайшов правильну перестановку:

$$S(t=t_c) = \frac{1}{4}\left\langle \left|\nabla f\right|^2 \right\rangle = \frac{1}{a^3}D(r), \tag{36}$$

і їх відношення при цьому є таким:

$$\frac{S(t=t_c)}{S(t=0)} = \frac{1}{a}. \tag{37}$$

Тобто, у даній моделі формування зображення, для будь-якого закону розподілу r, значення цільової функції у процесі оптимізації зменшується у a разів. Зауважимо, що параметр a у цьому випадку відіграє роль радіусу кореляції [33], отже будь-який алгоритм зможе відновити перемутоване зображення, тільки якщо розмір блоку перемутації менше радіуса кореляції

$$\mathrm{b} < \mathrm{a} = \frac{D(f)}{\frac{1}{4}\left\langle |\nabla f|^2 \right\rangle}. \qquad (38)$$

Зворотне твердження також справедливе – якщо умова (38) не виконується, тобто $b >> a$, то завжди можна знайти такі перестановки, які переведуть дане зображення у будь-яке інше (рис. 1) оптимальним у сенсі заданої цільової функції чином.

а

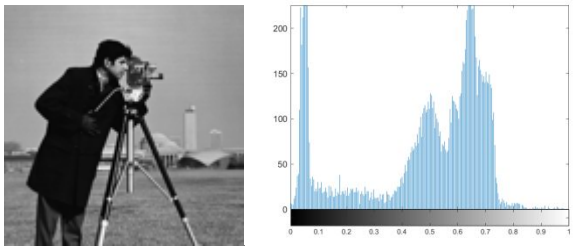
б

Рис. 1. Приклади зображень, що складаються з тих самих пікселів, розташованих у іншому порядку, зліва – псевдовипадкова перестановка, справа – відновлене зображення: а –‘Cameraman’; б – ‘Baboon’

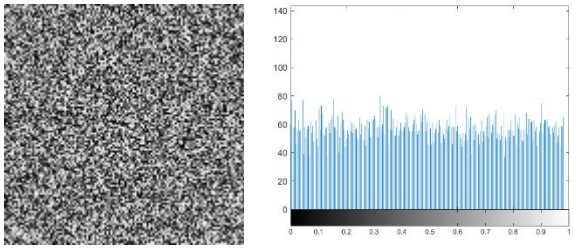
Це твердження досить очевидно, якщо розглядаються глобальні перестановки, тобто в якості блоку перестановки береться зображення в цілому, оскільки такі операції як, наприклад, віддзеркалення, транспонування чи повороти на кути, кратні $\pi/2$, вочевидь не змінюють значення цільової функції (2).

Зауважимо, що для зображення, що складається з випадкових пікселів, яскравість яких рівномірно розподілена на інтервалі [0, 1], знаходження такої (глобальної) перестановки, яка переводить його у інше (цільове) зображення, еквівалентно операції вирівнювання гістограми цільового зображення (при цьому, якщо порівнювати візуально, то пермутоване і вирівняне зображення є досить подібними, але розподіл по яскравості у пермутованого суттєво більш рівномірний, рис. 2).

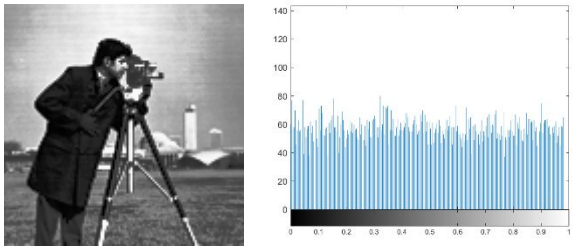
Експериментальні результати отримано за допомогою MATLAB 9.4.0.813654 (R2018a) з використанням Global Optimization Toolbox Version 3.4.4 [33].



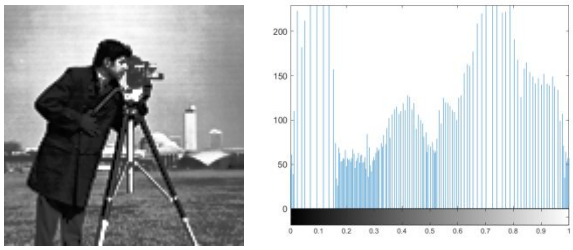
а



б



в



г

Рис. 2. Вирівнювання гістограм, зліва – зображення, справа – його гістограма: а – вихідне зображення ‘Cameraman’; б – зображення, що складається з випадкових пікселів, яскравість яких рівномірно розподілена на інтервалі [0, 1]; в – результат перестановки пікселів зображення; г – результат застосування операції вирівнювання гістограми для зображення а

Параметри генетичного алгоритму (якщо їх значення відрізняються від значень по замовчуванню), що використовувались для моделювання, наведено у табл. 1.

Всі експерименти проводилися з однаковим набором параметрів.

Для того, щоб алгоритм автоматично зупинявся, коли знайдена правильна перестановка, цільова функція для кожного виду зображень (що задається виразом (2)) була зменшена на значення цільової функції вихідного зображення.

Таблиця 1 – **Параметри ГА, що використовувалися для моделювання**

Параметр генетичного алгоритму	Значення	Примітка
'PopulationSize'	10000	Розмір популяції
'EliteCount'	5000	Додатнє ціле число, яке визначає, скільки індивідуумів у поточному поколінні гарантовано виживе до наступного покоління
'Generations'	50000	Максимальна кількість ітерацій перед тим, як алгоритм зупиниться
'StallGenLimit'	1000	Алгоритм зупиняється, якщо стільки ітерацій середня відносна зміна найкращого значення цільової функції менше або дорівнює значенню Function Tolerance
'StallTimeLimit'	50000	Алгоритм зупиняється, якщо немає покращення цільової функції протягом Max Stall Time секунд
'TolFun'	0	Відносна зміна найкращого значення цільової функції
'CreationFcn'	@gacreationuniform	Функція, що створює початкову популяцію
'CrossoverFcn'	@crossoverscattered	Функція, що алгоритм використовує для створення кросоверів нащадків
'CrossoverFraction'	0.8	Частка популяції у наступному поколінні, не рахуючи елітних нащадків, що створює функція кросовера
'MutationFcn'	{@mutationuniform, 0.8}	Функція, що використовується для мутації нащадків
'FitnessLimit'	0	Якщо цільова функція досягає цього значення, то алгоритм зупиняється

Таким чином її мінімальним значенням буде 0, що і є умовою зупинки алгоритму (задається параметром 'Fitness Limit' рівним 0).

Зауважимо, що від цієї умови можна відмовитися (в дусі чисто "сліпого" підходу), замінивши її більш жорсткою умовою на кількість ітерацій, при яких цільова функція не міняється (параметр 'Stall Gen Limit').

Експериментальні результати отримано за допомогою MATLAB 9.4.0.813654 (R2018a) з використанням Global Optimization Toolbox Version 3.4.4 [34].

Для експериментів було використано такі вихідні зображення (рис. 3):

‘Cameraman’ (Copyright Massachusetts Institute of Technology, використано з [34]);

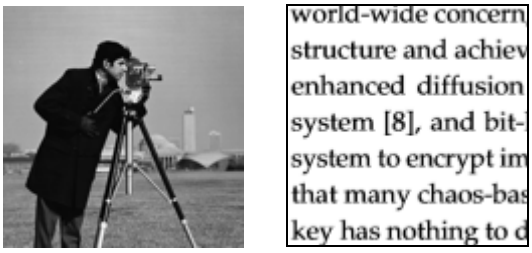
‘Текст’ (скріншот з програми Adobe Acrobat Reader DC [35] під час перегляду англомовної статті).

Всі зображення були перемасштабовані до однакового розміру (120 × 120), представлені у градаціях сірого і динамічний діапазон яскравості нормовано на 1.

Результати обробки зображення ‘Cameraman’. Експерименти проведено для різних значень розміру блоку b=2, 3, 4, 5, 6, 7, 8.

При всіх цих значеннях ГА знаходив правильну перестановку (рис. 4).

Результати обробки зображення ‘Текст’. Було проведено експерименти для різних значень розміру блоку b = 2, 3, 4, 5, 6 (рис. 5).



а б

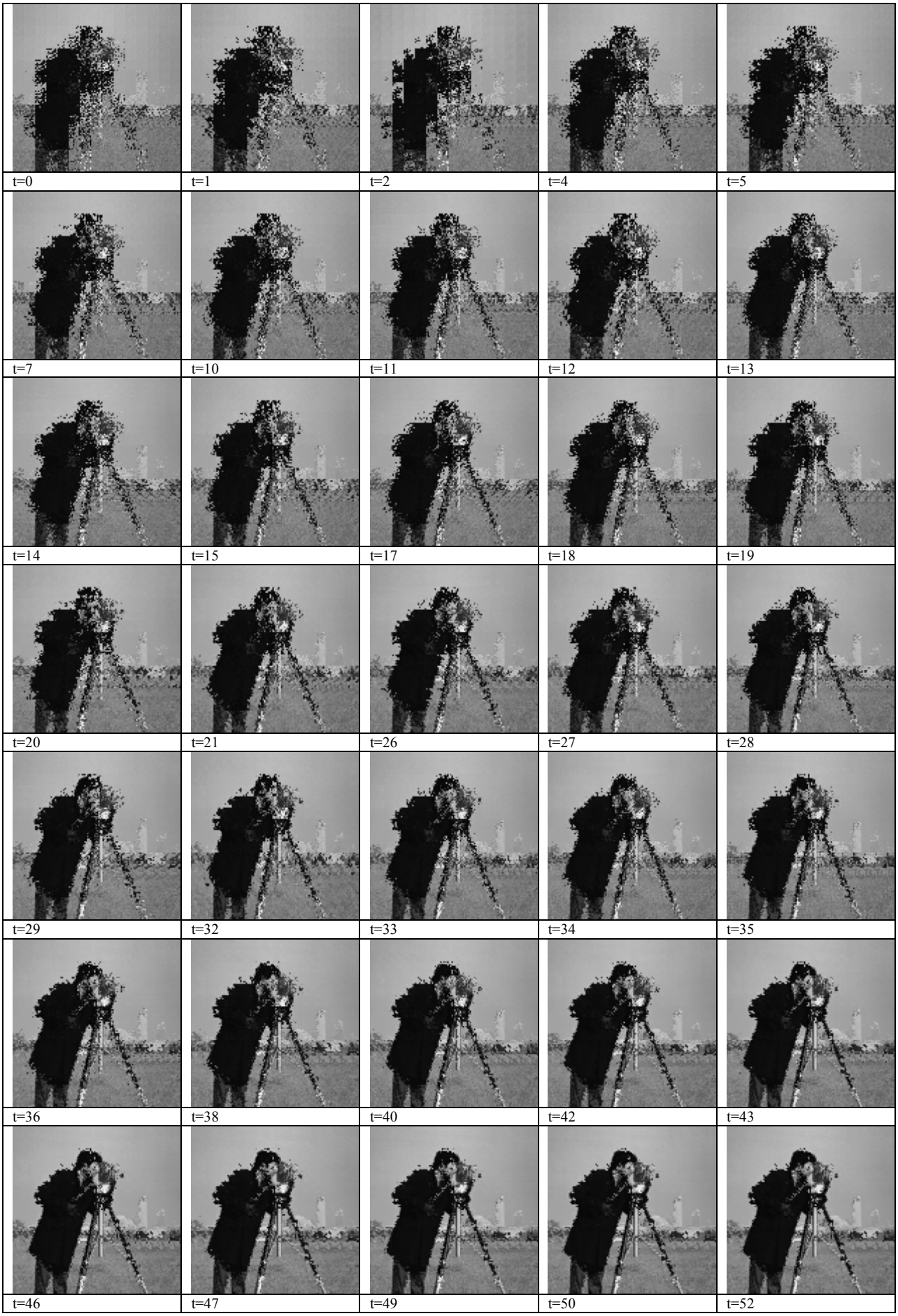
Рис. 3. Вихідні зображення:
а – ‘Cameraman’; б – ‘Текст’

Значення коефіцієнта кореляції (порахованого різними способами) для вихідних зображень наведено у табл. 2.

На рис. 7 представлені автокореляційні функції (АКФ) для вихідних і пермутованих зображень, зауважимо, що ці АКФ є суттєво анізотропними.

Таблиця 2 – **Значення коефіцієнта кореляції (порахованого різними способами) для вихідних зображень, що використовувалися для моделювання**

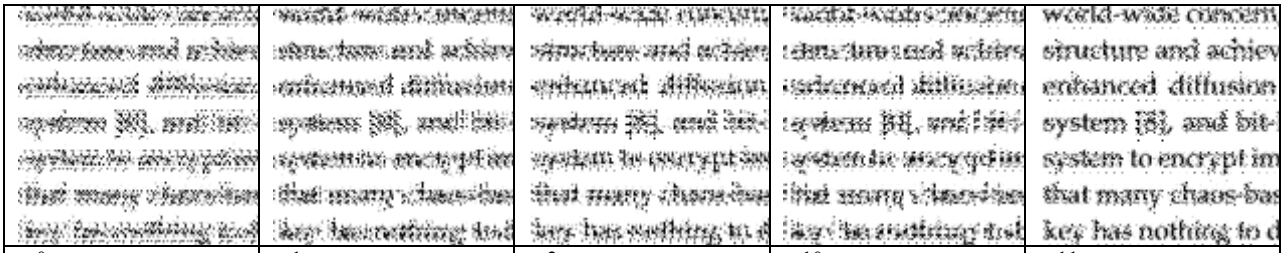
Зображення	Коефіцієнт кореляції		Максимальне значення блоку
	за формулою (38)	за формулою [30, 5.47]	
‘Cameraman’	24.0368	47.1123	b = 8
‘Текст’	04.1073	43.6819	b = 4


t=0
t=1
t=2
t=4
t=5
t=7
t=10
t=11
t=12
t=13
t=14
t=15
t=17
t=18
t=19
t=20
t=21
t=26
t=27
t=28
t=29
t=32
t=33
t=34
t=35
t=36
t=38
t=40
t=42
t=43
t=46
t=47
t=49
t=50
t=52

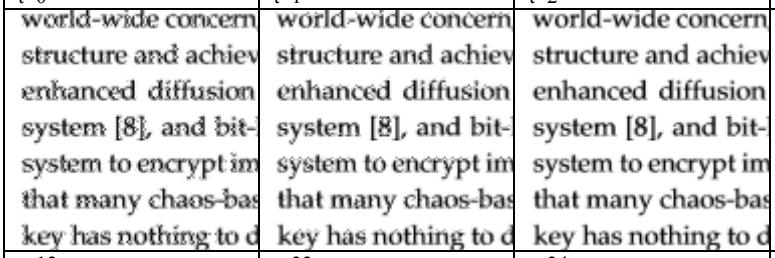
				
t=54	t=55	t=56	t=57	t=58
				
t=59	t=60	t=61	t=62	t=63
				
t=64	t=65	t=66	t=67	t=68
				
t=69	t=70	t=71	t=94	t=97
				
t=102	t=108	t=223	t=228	t=229
				
t=231	t=235	t=239	t=271	t=281
				
t=284	t=285	t=286	t=376	t=381

t=448	t=449	t=452		

Рис. 4. Результати обробки зображення ‘Cameraman’ (*b*=8), показано результати ітерацій, при яких мінялося значення цільової функції

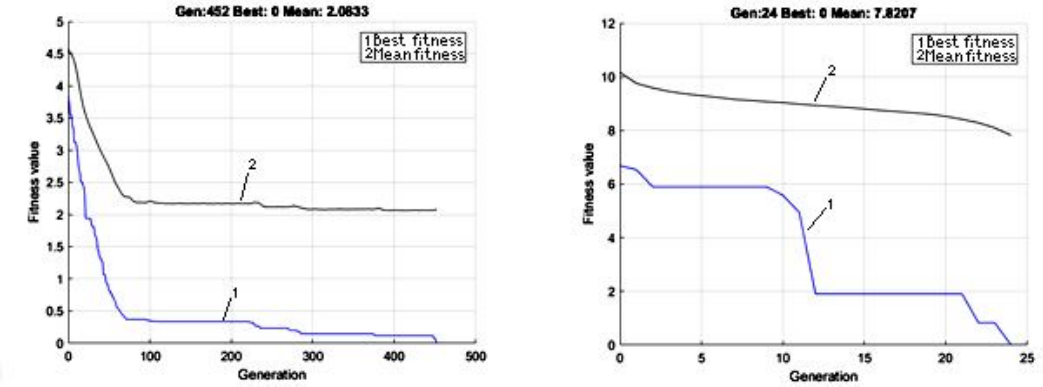


t=0	t=1	t=2	t=10	t=11



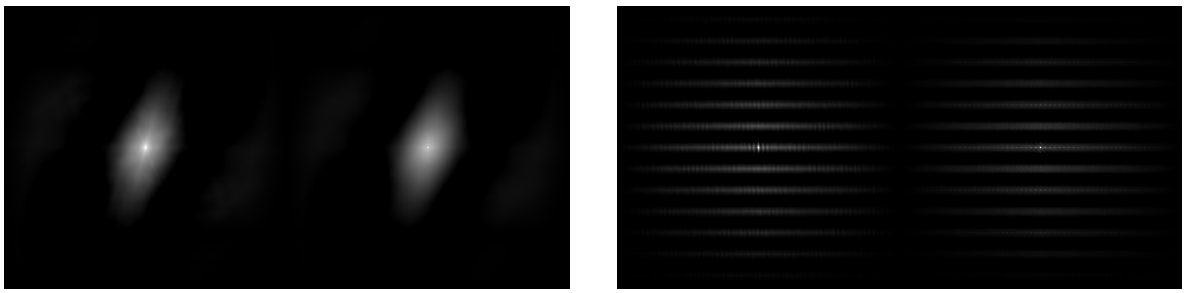
t=12	t=22	t=24		

Рис. 5. Результати обробки зображення ‘Текст’ (*b*=4), показано результати всіх ітерацій, при яких мінялося значення цільової функції



а б

Рис. 6. Графіки зміни цільової функції (найкращого і середнього значення по популяції):
а) ‘Cameraman’ (розмір блоку *b* = 8), ГА збігається за 452 ітерації;
b) ‘Текст’ (розмір блоку *b* = 4), ГА збігається за 24 ітерації



а б

Рис. 7. Зображення автокореляційних функцій для вихідних (зліва) і пермутованих (справа) зображень:
а) ‘Cameraman’ (розмір блоку *b*=8); б) ‘Текст’ (розмір блоку *b*=4)

Висновки і напрямки подальших досліджень

У роботі показано, що використання генетичних алгоритмів є надзвичайно потужним методом вирішення задачі знаходження інверсних блочних перестановок. Запропоновано ефективний спосіб кодування перестановок в геномі, такий що основні оператори ГА породжували інші допустимі перестановки, які рівномірно покривають область допустимих значень.

Було обрано цільову функцію, мінімізація якої проводиться за допомогою генетичних алгоритмів. Аналізуючи її, було отримано умову принципової розв'язуваності даної задачі – розмір блоку перестановки повинен бути менше радіусу кореляції вихідного зображення, що добре узгоджується з експериментом. При цьому отримано альтернативне представлення радіусу кореляції зображення, що узгоджується, але не співпадає з традиційним визначенням (через переріз головного піку автокореляційної функції).

Представляються цікавими подальші теоретичні та експериментальні (наприклад, за допомогою спеціально згенерованих синтетичних зображень) дослідження у цьому напрямку, окремого розгляду потребує питання анізотропії і мультимодальності автокореляційної функції.

СПИСОК ЛІТЕРАТУРИ

1. S. Skiena, S. The Algorithm Design Manual / Steven S. Skiena // Second Edition. Springer. – 2008.
2. Salleh, M. Image Encryption Algorithm Based on Chaotic Mapping / M. Salleh, S. Ibrahim, I. FauziIsninl //JurnalTeknologi. – 2003.
3. Mitra, A. A new image encryption approach using combinational permutation techniques / A. Mitra, V. Y. SubbaRao, R. S. MahadevaPrasanna // International Journal of Electrical and Computer Engineering. – 2006.
4. Rhouma, R. Cryptoanalysis of a new image encryption algorithm based on hyper-chaos /R. Rhouma, B. Safya// Phyiscal Letters A. – 2008. – Vol. 372. – P. 5973-5978.
5. Al-Najjar,H . Digital Image Encryption Algorithm Based on a Linear Independence Scheme and the Logistic Map / H. al-Najjar// International Arab Conference on Information Technology (ACIT 2011). – 2011.
6. Holland, J. Adaptation in Natural and Artificial Systems / J. Holland // Cambridge, MA: MIT Press. – 1992. – ISBN 978-0262581110.
7. Mitchell, M. An Introduction to Genetic Algorithms / M. Mitchell // Cambridge, MA: MIT Press. – 1996. – ISBN 9780585030944.
8. Banzhaf, W. Genetic Programming – An Introduction / W. Banzhaf, P. Nordin, R. Keller, F.Francone // San Francisco, CA: Morgan Kaufmann. –1998. –ISBN 978-1558605107.
9. G. Jacob. Genetic Algorithm based Mosaic Image Steganography for Enhanced Security /George, Jacob and Janahanlal Stephen. – 2014.
10. Evolutionary Algorithms for Feature Selection, [Електронний ресурс]. – Режим доступу до ресурсу: https://www.kdnuggets.com/2017/11/rapidminer-evolutionary-algorithms-feature-selection.html. звернення 2018-02-19).
11. Dos Santos-Paulino, A. Evolutionary algorithm for dense pixel matching in presence of distortions / A. dos Santos-Paulino, J.-C. Nebel, F.Florez-Revuelta //EvoStar Conference, Granada, Spain. – 23–25 April 2014.
12. Paulinas, Mantas. A survey of genetic algorithms applications for image enhancement and segmentation / Paulinas, Mantas and Usinskas, Andrius // Information Technology and Control. – 2007. – Vol. 36. – P. 278-284.
13. Behzadi, S. Developing a Genetic Algorithm to Solve Shortest Path Problem on a Raster Data Model / S. Behzadi, Ali A. Alesheikh, E. Poorazizi //Journal of Applied Sciences. – 2008. – N. 8. –P. 3289-3293.
14. Cagnoni, Stefano. Genetic and evolutionary computation for image processing and analysis / Cagnoni, Stefano &Lutton, Evelyne&Olague, Gustavo. – 2007.
15. Bruckstein, A.M. Holographic representation of images / A.M. Bruckstein, R.J. Holt, A.N. Netravali // IEEE Transactions on Image Processing. – 1998. – N 7. – P. 1583-1587.
16. Bruckstein, A.M. Holographic image representations: the subsampling method / A.M. Bruckstein, R.J. Holt, A.N. Net-ravali // IEEE Int. Conference on Image Processing – Santa Barbara, California, USA, October. – 1997. – Vol. 1. – P. 177-180.
17. Bruckstein, A.M. Technique for Holographic Representation of Images / A.M. Bruckstein, R.J. Holt, A.N. Netravali // US 6,091,394. – July 18, 2000. – 6 p.
18. Колесов, В.В. Псевдоголографическое кодирование цифровой информации / В.В. Колесов, Н.Н. Залогин, Г.М. Воронцов // Радиотехника и электроника. – 2002. – Т. 2, № 5. – С. 583-588.
19. Марковский, А.В. О квазиголографическом кодировании цифровых изображений / А.В. Марковский // Автоматика и телемеханика. – 2001. – № 9. – С. 163-173.
20. Кузнецов, О.П. Квазиголографический подход к кодированию графической информации / О.П. Кузнецов, А.В. Марковский // Искусственный интеллект. – 2002. – № 2. – С. 474-482.
21. Баринова, Д.А. Разработка и исследование алгоритмов обработки цифровых изображений, представленных в псевдоголографических кодах / Д.А. Баринова // Компьютерная оптика. – 2005. – № 27. – С. 149-154.
22. Воронин, В.В. Голографическое представление в задачах обработки изображений / В.В. Воронин // Тезисы конференции РОАИ – 5. – 2000. – С. 237-241.
23. Yaloveha V., Hlavcheva D., Podorozhniak A. Usage of convolutional neural network for multispectral image processing applied to the problem of detecting fire hazardous forest areas. *Сучасні інформаційні системи*. 2019. Т. 3, № 1. С. 116–120. DOI: https://doi.org/10.20998/2522-9052.2019.1.19.
24. Kuchuk G., Kovalenko A., Komari I.E., Svyrydov A., Kharchenko V. Improving big data centers energy efficiency: Traffic based model and method. Studies in Systems, Decision and Control, vol 171. Kharchenko, V., Kondratenko, Y., Kacprzyk, J. (Eds.). Springer Nature Switzerland AG, 2019. Pp. 161-183. DOI: http://doi.org/10.1007/978-3-030-00253-4_8
25. Amin Salih Mohammed, Saravana Balaji B., Hiwa Abdulkarim Mawlood. Conceptual analysis of Iris Recognition Systems. Advanced Information Systems. 2019. Vol. 3, No. 2. P. 86-90. DOI : https://doi.org/10.20998/2522-9052.2019.2.15

26. Freeman, H. .Apictorial Jigsaw Puzzles: The Computer Solution of a Problem in Pattern Recognition / H. Freeman, L. Garder // IEEE Transactions on Electronic Computers, EC-13(2): – 1964. – P. 118-127.
27. Cho, T. A Probabilistic Image Jigsaw Puzzle Solver / T. Cho, S. Avidan, W. Freeman // Int. IEEE Conference on Computer Vision and Pattern Recognition. – 2010. – P. 183-190.
28. Yang, X. Particle Filter with State Permutations for Solving Image Jigsaw Puzzles / X. Yang, N. Adluru, L. J. Latecki // Int. IEEE Conference on Computer Vision and Pattern Recognition. – 2011. –P. 2873–2880.
29. Pomeranz, D. A Fully Automated Greedy Square Jigsaw Puzzle Solver / D. Pomeranz, M. Shemesh, and O. Ben-Shahar // In IEEE Conference on Computer Vision and Pattern Recognition. – 2011. – P. 9-16.
30. Sholomon, D. A Genetic AlgorithmBased Solver for Very Large Jigsaw Puzzles / D. Sholomon, O. David, N. Netanyahu // Int. 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2013. – P. 1767-1774.
31. MATLAB в імплементації функції randperm, [Електронний ресурс]. – Режим доступу до ресурсу: https://www.mathworks.com/help/matlab/ref/randstream.randperm.html. звернення 2018-04-03).
32. Li, X. An efficient two dimensional moving average model for texture analysis and synthesis / X. Li, J.A. Cadzow, D.M. Wilkes, R.A. Peters, M. Bodruzzaman //Publication Year. – 1992. – Vol. 1. –P. 392-395.
33. Красильщиков М.Н., Себряков Г.Г. (ред.). Современные информационные технологии в задачах навигации и наведения беспилотных маневренных летательных аппаратов. М.: Физматлит, 2009. – 556 с. – ISBN: 978-5-9221-1168-3.
34. MATLAB 9.4.0.813654 (R2018a) з використанням Global Optimization Toolbox Version 3.4.4, [Електронний ресурс]. – Режим доступу до ресурсу: https://www.mathworks.com/matlabcentral/answers/404947-using-syms-matlab-stops-working. звернення 2018-04-19).
35. Adobe Acrobat Reader DC [Електронний ресурс]. – Режим доступу до ресурсу: https://www.adobe.com/. звернення 2018-05-10).

Рецензент: д-р техн. наук, проф. К. С. Васюта,
Харківський національний університет Повітряних Сил імені Івана Кожедуба, Харків.
Received (Надійшла) 21.05.2019
Accepted for publication (Прийнята до друку) 31.07.2019

Использование генетических алгоритмов для нахождения инверсных псевдослучайных блочных перестановок

А. Н. Маковейчук, И. В. Рубан, Г. В. Худов

Предметом изучения в статье является метод нахождения инверсных псевдослучайных блочных перестановок пикселей в изображении. **Целью** является разработка "слепого" метода нахождения инверсных псевдослучайных блочных перестановок с помощью генетических алгоритмов. **Задачи**: провести анализ факторов, влияющих на инверсные псевдослучайные блочные перестановки на изображении, разработать метод кодирования перестановок в генетических алгоритмах, обосновать выбор целевой функции для оптимизации с помощью генетических алгоритмов. Используемыми **методами** являются: методы цифровой обработки изображений, теории вероятности, математической статистики, криптографии и защиты информации, математический аппарат теории матриц. Получены следующие **результаты**. Проведен анализ факторов, влияющих на инверсные псевдослучайные блочные перестановки на изображении. Определены факторы, влияющие на максимальный размер блока, при котором еще возможно нахождение инверсной перестановки. Разработан метод нахождения инверсных псевдослучайных блочных перестановок пикселей в пермутированном изображении с помощью генетических алгоритмов. **Выводы.** Научная новизна полученных результатов заключается в следующем. Установлено, что нахождение инверсных перестановок возможно лишь при условии, что размер блока является меньше радиус корреляции изображения. Предложен эффективный способ кодирования перестановок, при котором стандартные операторы генетических алгоритмов будут порождать новые и только допустимые перестановки. Предложено в качестве целевой функции использовать сумму квадратов градиентов. Показано, что данная целевая функция имеет глобальный минимум для корректной перестановки, которая позволяет находить инверсные блочные перестановки "вслепую" без дополнительной априорной информации.

Ключевые слова: генетические алгоритмы, инверсная псевдослучайная блочная перестановка, пермутированное изображение, (псевдо)голографическое кодирования, радиус корреляции.

Using genetic algorithms to find inverse pseudo-random block permutations

O. Makoviechuk, I. Ruban, G. Hudov

The **subject matter** of the article is a method for finding inverse pseudorandom block permutations of pixels in an image. The **goal** is to develop a "blind" method for finding inverse pseudo-random block permutations using genetic algorithms. **Tasks**: to analyze the factors affecting inverse pseudorandom block permutations in the image, to develop a method for coding permutations in genetic algorithms, to justify the choice of the objective function for optimization using genetic algorithms. The **methods** used are: methods of digital image processing, probability theory, mathematical statistics, cryptography and information protection, the mathematical apparatus of matrix theory.The following **results** are obtained. The analysis of factors affecting inverse pseudorandom block permutations in the image is carried out. The factors affecting the maximum block size at which the inverse permutation is still possible are determined. A method has been developed for finding inverse pseudorandom block permutations of pixels in a permuted image using genetic algorithms. **Conclusions.** The scientific novelty of the results is as follows.It was established that finding inverse permutations is possible only on condition that the block size is smaller than the image correlation radius. An effective method for coding permutations is proposed, in which the standard operators of genetic algorithms will generate new and only permissible permutations. It is proposed to use the sum of the squared gradients as the objective function.It is shown that this objective function has a global minimum for correct permutation, which allows one to find inverse block permutations "blindly" without additional a priori information.

Keywords: genetic algorithms, inverse pseudo-random block permutation, permuted image, (pseudo) holographic coding, correlation radius.