УДК 004.043

О. В. Шпортько

ВИКОРИСТАННЯ РІЗНИЦЕВИХ КОЛЬОРОВИХ МОДЕЛЕЙ ДЛЯ СТИСНЕННЯ RGB-ЗОБРАЖЕНЬ БЕЗ ВТРАТ

The method of forming of the differential coloured models for the improvement of indexes of compression of RGB-images without losses by using predictors is offered. The fragments of the programs in C language for the calculation of parameters of such coloured models for large images are presented. As experiments display the uses of the worked differential coloured models, allows to decrease the aspect ratios images approximately over 4.5%.

Запропоновано спосіб формування різницевих кольорових моделей для покращання показників стиснення RGB-зображень без втрат із застосуванням предикторів. Наведено фрагменти програм мовою С для обчислення параметрів таких кольорових моделей для великих зображень. Як свідчать експерименти, використання розрахованих різницевих кольорових моделей дає змогу зменшити коефіцієнти стиснення зображень у середньому більше, ніж на 4,5%.

Зображення є невід'ємною частиною мультимедійної інформації, що найчастіше передається каналами зв'язку чи зберігається на електромагнітних носіях. Тому проблема підвищення ефективності їх стиснення є актуальною сьогодні і буде актуальною в найближчому майбутньому.

Всі алгоритми стиснення зображень поділяють на два основні класи: з втратами та без втрат. І якщо для переважної більшості алгоритмів компресії зображень з втратами можна забезпечити потрібний коефіцієнт стиснення (відношення розмірів стиснутого до нестиснутого файлів зображення, надалі – КС) внаслідок погіршення якості, то КС зображень без втрат залежить, власне, лише від перепадів кольорів їх пікселів та алгоритму стиснення, не регулюється програмно і становить у середньому тільки 30...70%. Як відомо, будь-яке стиснення даних можливе лише внаслідок зменшення надлишковостей. Саме тому підвищити показники стиснення зображень без втрат намагаються не тільки за допомогою розробок та вдосконалення алгоритмів компресії, а й за рахунок алгоритмів попередньої обробки (препресингу), які підвищують рівень надлишковості.

У цій статті ми пропонуємо спосіб формування різницевої кольорової моделі для кожного зображення з метою покращення їх КС у форматах без втрат.

Аналіз останніх досліджень і публікацій. Для збереження кольорів пікселів зображень без втрат найчастіше використовують кольорову модель RGB з точністю дискретизації 8 бітів, яка відображає роботу комп'ютерних дисплеїв [3]. Тобто колір довільного піксела формується злиттям яскравостей червоної, зеленої та синьої компоненти, кожна з яких є рівнозначною і подається цілим числом з діапазону [0; 255]. Зображення з втратами найчастіше зберігаються в кольоровій моделі YCrCb (яскравість, червоність, синява), що використовується в телебаченні. У цій моделі основна енергія зображення концентрується у першій компоненті, що дає змогу зберігати менше інформації для двох інших компонентів і завдяки цьому покращувати КС [5]. Загалом, колір кожного піксела можна подати у вигляді координат по трьох лінійно-незалежних векторах будь-яких базових кольорів [4], що й застосовувалося під час дослідження.

Пікселі зображення найчастіше обробляються під час стиснення без втрат по рядках зверху вниз, а у кожному рядку – послідовно зліва направо (як символи у текстах), формуючи тим самим вхідний потік. Як правило, під час стиснення дані пікселів зображення обробляються після препресингу контекстно-залежним

алгоритмом, що зменшує надлишковості між різними фрагментами зображення, а його результати опрацьовуються контекстно-незалежним алгоритмом, який враховує частоти окремих елементів даних. У різних форматах стиснення зображень без втрат використовуються різні контекстно-залежні алгоритми, що орієнтуються, як правило, на різні класи зображень. Але ефективність цих алгоритмів невисока, оскільки в зображеннях найчастіше повторюються подібні, але не однакові фрагменти. Тому розглянемо принцип кодування контекстно-незалежних алгоритмів та особливості використання предикторів, які найчастіше застосовують для попередньої обробки зображень.

Ідея використання контекстно-незалежних алгоритмів полягає у заміні елементів з більшою частотою (тут і надалі – абсолютною) кодами з меншої кількості бітів, ніж для елементів з меншою частотою. Згідно з теоремою Шеннона, елемент s_i з ймовірністю появи $p(s_i)$ найвигідніше кодувати $-\log p(s_i)$ бітами (тут і надалі логарифм береться за основою 2). Тоді середня довжина коду елемента після застосування контекстно-незалежного алгоритму має наближатися до *ентропії джерела* [2, 5]

$$H = -\sum_i p(s_i) \times \log(p(s_i)) . \qquad (1)$$

Ентропія джерела зменшується у випадку збільшення нерівномірності розподілу ймовірностей між елементами [1]. У сучасних форматах стиснення зображень без втрат для контекстно-незалежного стиснення найчастіше використовуються коди Хафмана чи арифметичне кодування [2].

Зменшити ентропію в процесі обробки зображень без втрат найчастіше намагаються за допомогою *предикторів* [6, 5], які прагнуть, використовуючи значення відомих суміжних елементів, спрогнозувати значення чергового елемента. Для трикомпонентних кольорових моделей предиктор кожної компоненти прогнозує значення згідно з відповідними компонентами суміжних пікселів. У процесі використання цієї технології обчислюють і надалі кодують відхилення чергової компоненти від значення, прогнозованого предиктором. Тому процес застосування предикторів до кожної компоненти піксела у вузлі (i, j) записують формулою

$$\Delta_{ij} = C_{ij} - predict_{ij} , \qquad (2)$$

де C_{ij} – значення компоненти до застосування предиктора; Δ_{ij} – значення компоненти після застосування предиктора; $predict_{ij}$ – значення предиктора, обчисленого для обраної компоненти. Враховуючи близькість кольорів переважної більшості суміжних пікселів, унаслідок перетворення (2) більшість значень Δ_{ij} виявляються близькими до 0. Такий перерозподіл частот значень (а, отже, і ймовірностей) значно підвищує нерівномірність розподілу і тому зменшує ентропію джерела (згідно з (1)). Отже, предиктори хоча й не виконують безпосереднє стиснення даних, але зменшують, насамперед, КС контекстно-незалежного алгоритму [3].

Для забезпечення високої швидкості декодування у форматах графічних файлів здебільшого використовують статичні предиктори, що виражають значення компоненти піксела через лінійну комбінацію відповідних компонентів суміжних вже закодованих пікселів [2, 5]. Найчастіше серед таких лінійних предикторів (LPC) використовують: *LeftPredict*, який прогнозує значення компоненти піксела рівним значенню аналогічної компоненти суміжного піксела зліва; *RightPredict* – суміжного піксела зверху; *AveragePredict* – середньому арифметичному цих значень. Під час застосування таких предикторів фактично обчислюють прирости яскравостей в обраному напрямку. Функції мовою С для обрахунку цих предикторів наведені в [6], класифікація предикторів – в [1].

Проблема зменшення ентропії значень компонентів RGB-зображень шляхом переходу до альтернативних кольорових моделей досі залишається **невирішеною**, хоча фіксовані альтернативні кольорові моделі використовуються в програмному забезпеченні для стиснення не перший рік (наприклад, в архіваторі WinRAR стиснення RGB зображень відбувається після переходу до моделі R-G, G, B-G). Тому **метою статті** є обґрунтування та опис алгоритму формування та переходу до альтернативних різницевих кольорових моделей у форматах, що використовують лінійні предиктори, для покращення КС RGB-зображень завдяки зменшенню ентропії.

Результати дослідження. Очевидно, що замінювати окрему компоненту кольорової моделі комбінацією компонентів доцільно лише тоді, коли така заміна зменшить ентропію даних зображення. Нехай зображення має *height* рядків та *width* стовпців пікселів. Як зазначено в [5], "експерименти з великими числами показують, що значення Δ прагне мати розподіл, близький до розподілу Лапласа". Густина ж ймовірностей розподілу Лапласа задається відомою формулою $f(x)=\frac{\lambda}{2}e^{-\lambda|x|}$, тобто нерівномірність розподілу зростає зі збільшенням λ. Відповідно, λ виражається через дисперсію розподілу співвідношенням $\lambda = 2/\sqrt{D(\xi)}$, отже, зі зменшенням дисперсії нерівномірність розподілу частот навколо нуля зростає (дисперсію компонентів пікселів зображення ще називають його *енергією* [5]). Тобто, замінювати, наприклад, значення компоненти R_{ij} лінійною комбінацією R_{ij}–kG_{ij} для всіх пікселів зображення доцільно лише тоді, коли

$$S(k) = D(\Delta(R - kG)) < D(\Delta R), \tag{3}$$

де $\Delta R = \{\Delta R_{ij}\}$, $\Delta(R-kG) = \{\Delta(R_{ij} - kG_{ij})\}$, $i = \overline{0, height-1}$, $j = \overline{0, width-1}$. З (3) випливає, що k слід обирати таким, щоб дисперсія значень предиктора лінійної комбінації компонентів пікселів зображення $S(k)$ була мінімальною. Нехай цей мінімум досягається при $k = k_0$:

$$k_0 : D(\Delta(R - k_0 G)) = \min_k S(k), k \in \Re . \tag{4}$$

Оскільки лінійна комбінація кольорів фактично задає інший колір, то (4) відображає процес пошуку напрямку в площині, стосовно якого енергія приростів є мінімальною. Визначимо k_0 методом найменших квадратів. Дисперсія за умови нульового математичного сподівання дорівнює сумі квадратів окремих елементів, тобто

$$D(\xi) = \sum_{i=0}^{height-1} \sum_{j=0}^{width-1} \xi_{ij}^2 , \tag{5}$$

тому з (4) і (5) одержимо

$$S(k) = \sum_{i=0}^{height-1} \sum_{j=0}^{width-1} (\Delta(R_{ij} - kG_{ij}))^2 \to \min . \tag{6}$$

Враховуючи, що для лінійних предикторів $\Delta(R_{ij} - kG_{ij}) = \Delta R_{ij} - k\,\Delta G_{ij}$, прирівняємо похідну функції $S(k)$ з (6) до нуля:

$$\frac{dS(k)}{dk} = -2 \sum_{i=0}^{height-1} \sum_{j=0}^{width-1} (\Delta R_{ij} - k\,\Delta G_{ij})\,\Delta G_{ij} = 0 ,$$

звідки

$$k_0 = \left(\sum_{i=0}^{height-1} \sum_{j=0}^{width-1} \Delta R_{ij}\, \Delta G_{ij} \right) \Bigg/ \left(\sum_{i=0}^{height-1} \sum_{j=0}^{width-1} (\Delta G_{ij})^2 \right). \qquad (7)$$

Формули для обчислення коефіцієнта k_0, що мінімізує дисперсію значень предиктора лінійної комбінації інших пар компонентів, отримують аналогічно. Розглянемо значення цього коефіцієнта (див. табл. 1) для предиктора *LeftPredict* та різних пар лінійних комбінацій компонентів пікселів восьми файлів 24-бітних зображень стандартного набору ACT (завантажити TIFF-версії зображень можна, наприклад, з http: //compression.ca/act/act-files.html, а віднайти характеристики – в [6]). Цей набір містить штучні (№№ 1, 2, 7) і природні (всі інші) зображення.

Таблиця 1. Значення коефіцієнта k_0 з точністю до трьох знаків для предиктора *LeftPredict* та різних пар лінійних комбінацій компонентів пікселів файлів зображень набору ACT

Лінійна комбінація	№ файла з набору ACT							
	1	2	3	4	5	6	7	8
B-kG	0,095	0,140	0,699	0,828	0,789	0,920	0,068	0,960
G-kB	0,085	0,143	0,752	0,905	0,853	0,951	0,078	0,852
B-kR	0,160	0,306	0,572	0,798	0,665	0,849	0,036	0,885
R-kB	0,130	0,279	0,374	0,846	0,623	0,923	0,034	0,617
G-kR	0,311	0,129	0,851	0,852	0,780	0,930	0,187	0,929
R-kG	0,285	0,115	0,518	0,826	0,677	0,978	0,155	0,730

Як свідчать дані табл. 1 та аналогічні розрахунки, проведені авторами для інших зображень, k_0, як правило, належить інтервалу (0; 1]. І це не дивно, адже значення ΔR_{ij}, ΔG_{ij} та ΔB_{ij} для змістовних зображень найчастіше близькі між собою і мають однаковий знак. Для форматів графічних файлів, що виконують стиснення без втрат, дії над дійсними числами неприпустимі, оскільки заокруглення можуть призвести до втрат точності, тому ***k* в (3) та подібних формулах для інших пар компонентів слід покласти рівним одиниці**, хоча для форматів стиснення з втратами *k* слід розраховувати як відношення перехресної кореляції приростів яскравостей до дисперсії приростів компоненти згідно з (7).

Щоб забезпечити однозначність декодування, в зображенні можна виконати **максимум дві** заміни значень різних компонентів різницями з іншими компонентами по всіх пікселах. Очевидно, що під час попередньої обробки зображень серед можливих шести замін для всіх пікселів (значень компоненти R_{ij} різницями $R_{ij}–G_{ij}$ або $R_{ij}–B_{ij}$, значень G_{ij} різницями $G_{ij}–R_{ij}$ або $G_{ij}–B_{ij}$ чи значень B_{ij} різницями $B_{ij}–R_{ij}$ або $B_{ij}–G_{ij}$) **доцільно виконати лише ті дві, що максимально зменшать дисперсію результатів дії обраного предиктора**. З метою опису алгоритму формування різницевих кольорових моделей запишемо досліджувані дисперсії у вигляді матриці аналізу *A*:

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} D(\Delta(r)) & D(\Delta(R)-\Delta(G)) & D(\Delta(R)-\Delta(B)) \\ D(\Delta(G)-\Delta(R)) & D(\Delta(G)) & D(\Delta(G)-\Delta(B)) \\ D(\Delta(B)-\Delta(R)) & D(\Delta(B)-\Delta(G)) & D(\Delta(B)) \end{pmatrix}. \qquad (8)$$

Враховуючи (5), неважко пересвідчитись, що $a_{mn} = a_{nm}$, тобто для формування цієї матриці необхідно обчислити значення лише шести елементів. Нехай в межах кожного піксела компонента R має індекс 0, G – 1, B – 2. Бачимо, що різницева кольорова модель визначається щонайбільше двома недіагональними еле-

ментами різних рядків матриці *A*, які серед елементів, менших за діагональні елементи своїх рядків найбільше від них відхиляються (забезпечують максимальні зменшення дисперсій). У випадку наявності таких елементів індекс рядка кожного з них визначає зменшувану компоненту, а індекс стовпця – компоненту, що від неї віднімається. Наприклад, вибір елемента a_{12} вказує, що в альтернативній кольоровій моделі для кожного піксела зображення необхідно значення компоненти *G* зменшити на значення компоненти *B*.

Покроково алгоритм формування та переходу до різницевої кольорової моделі перед стисненням зображення з використанням предикторів записується так:

1. Розрахувати для кожної компоненти всіх пікселів зображення результати дії обраного лінійного предиктора;

2. Обчислити дисперсії компонентів і різниць компонентів результатів дії обраного предиктора та зберегти їх в матриці *A* згідно з (8);

3. Занести значення 0 у змінні *index11*, *index12*, *index21* та *index22*, що визначають можливі різниці альтернативної кольорової моделі;

4. Визначити в матриці *A* недіагональний елемент, який серед елементів, менших за діагональні елементи своїх рядків, найбільше відхиляється від діагонального елемента рядка. Якщо такий елемент відсутній, то перейти до кроку 9. Інакше занести в змінну *index11* номер рядка цього елемента, а в змінну *index12* – номер його стовпця;

5. Аналогічно визначити в матриці *A* недіагональний елемент, який не міститься в рядку *index11* і не симетричний попередньому знайденому елементу стосовно головної діагоналі. Якщо такий елемент відсутній, то перейти до кроку 7. Інакше занести в змінну *index21* номер рядка цього елемента, а в змінну *index22* – номер його стовпця;

6. Якщо значення змінних *index11* та *index22* однакові, то переставити значення змінних *index11* з *index21* та *index12* з *index22*, тобто змінити черговість віднімання компонентів кольорової моделі;

7. Якщо значення змінних *index11* та *index12* різні, тобто перша різниця кольорової моделі визначена, то зменшити для кожного піксела зображення значення компоненти з індексом *index11* на значення компоненти з індексом *index12*;

8. Якщо значення змінних *index21* та *index22* різні, тобто друга різниця кольорової моделі визначена, то зменшити для кожного піксела зображення значення компоненти з індексом *index21* на значення компоненти з індексом *index22*;

9. Завершити формування та перехід до різницевої кольорової моделі і виконати стиснення перетвореного зображення з використанням предикторів.

Для практичного формування альтернативних кольорових моделей ми обрали предиктор *LeftPredict*, оскільки він використовує лише значення компоненти попереднього піксела активного рядка і тому розраховується найшвидше, хоча з цією метою можна використовувати і інші лінійні предиктори. Крім цього, для уникнення піднесень до квадрату значень відхилень всіх компонентів зображення в процесі розрахунку дисперсій ми використовували масив значень квадратів для різних відхилень з урахуванням знаків байтових чисел. З метою уникнення переповнень під час операцій з цілими числами значення елементів цього масиву обмежувалися числом 100.

Якщо в обраній оболонці мови програмування реалізовані операції з дійсними числами, то для формування альтернативних кольорових моделей у (8) можна оцінювати не дисперсії, а прогнозовані довжини кодів компонентів пікселів після застосування лінійного предиктора за допомогою ентропії. Нехай кожен з елементів s_i трапляється N_i разів у послідовності довжиною $N = \sum_i N_i$. Тоді $p(s_i) = N_i / N$ і загальна довжина закодованих даних, враховуючи (1), наближається до значення

$$N \times H = N \log(N) - \sum_i N_i \log(N_i) . \quad (9)$$

Функція мовою С для обчислення прогнозованої довжини за частотами окремих елементів згідно з (9) із врахуванням формули переходу від двійкового до натурального логарифму має такий вигляд:

```
int sizeEntropiCode(int *masFreq)
{int count=0; double size=0;
 for (int i=0; i<256; i++) // цикл по елементах
  {count+=masFreq[i];
   if (masFreq[i]>1) // для існування log
    size+=masFreq[i]*log(masFreq[i]); }
 if (count)
  size=(count*log(count)-size)/log(2);
 return size; }
```

Для наближеного розрахунку довжин кодів компонентів та різниць компонентів після дії предиктора необхідно спочатку підрахувати частоти окремих елементів, а потім скористатися (9). Модифікований фрагмент програми мовою *C* для формування елементів матриці *A* під час переходу до різницевої кольорової моделі у цьому випадку матиме такий вигляд:

```
for (m=0; m<=2; m++) // цикл по рядках масиву A прогнозованих довжин кодів
 for (n=0; n<=2; n++) // цикл по стовпцях масиву A
 {memset(freq, 0, sizeof(freq)); // обнулення масиву частот елементів
  if (m==n) // елемент діагональний – оцінюємо довжину коду компоненти предиктора
   {for (i=0; i<height; i++) // цикл по рядках масиву значень предиктора
     for (j=0; j<row_width; j+=3) // цикл по пікселах чергового рядка
      freq[corect[i][j+m]]++;
    a[m][n]=sizeEntropiCode(freq); }
  else // якщо елемент недіагональний
   {if (n>m) // вище діагоналі – оцінюємо довжину коду різниці компонент предикторів
     {for (i=0; i<height; i++)
       for (j=0; j<row_width; j+=3)
        freq[(byte)(corect[i][j+m]-corect[i][j+n])]++;
      a[m][n]=sizeEntropiCode(freq); }
    else // нижче діагоналі – присвоюємо значення симетричного елемента
     a[m][n]=a[n][m]; }}
```

Зауважимо, що, по-перше, під час декодування значення компонентів слід додавати у зворотній послідовності: спочатку до значень компоненти *index21* значення компоненти *index22*, а вже потім до значень компоненти *index11* значення компоненти *index12*. По-друге, якщо під час кодування доводиться відмовлятися від застосування предикторів, то слід відмовитися також і від використання альтернативної кольорової моделі, оскільки вона орієнтується саме на дію предикторів.

Результати експериментів. Розглянемо результати застосування описаних алгоритмів формування та переходу до різницевої кольорової моделі для стиснення зображень вищезазначеного набору ACT. Тестування проводили за допомогою модифікованої програми з CD [3] для запису зображень у форматі PNG, котра послідовно виконує формування та застосування різницевої кольорової моделі; розбиття зображення на блоки однорідних рядків; застосування до пікселів кожного рядка обраного предиктора; стиснення даних зображення контекстно-залежним словниковим алгоритмом LZ77 [7], що кодує повтори; стиснення отриманих елементів даних контекстно-незалежними префіксними кодами Хафмана. Стиснення кожного зображення виконували у трьох кольорових моделях: стандартній RGB, різницевій, що розраховувалася за допомогою дисперсій та різницевій, що формувалася з використанням прогнозованих ентропійних довжин ко-

дів. Різницеві кольорові моделі, розраховані цими двома способами, наведені в табл. 2. Коефіцієнти стиснення файлів зображень у трьох описаних кольорових моделях подано в табл. 3, час кодування – в табл. 4, а час декодування – в табл. 5.

Таблиця 2. Різницеві кольорові моделі для стиснення зображень набору ACT із застосуванням предикторів

Спосіб формування	№ файла з набору ACT							
	1	2	3	4	5	6	7	8
Дисперсійний	R,G-R,B	RGB	R,G-R,B-G	R,G-B,B-R	R,G-R,B-G	R-G,G,B-G	RGB	R,G-R,B-G
Ентропійний	R,G-R,B	RGB	R,G-R,B-G	R,G-B,B-R	R,G-B,B-R	R-G,G,B-G	RGB	R,G-R,B-G

Таблиця 3. Коефіцієнти стиснення файлів зображень набору ACT у форматі PNG для різних кольорових моделей, %

Кольорова модель	№ файла з набору ACT								Середній
	1	2	3	4	5	6	7	8	
RGB	20,89	6,74	60,21	52,04	5,.84	65,65	7,10	57,50	40,50
Різницева дисперсійна	20,85	6,74	58,78	44,41	47,72	52,73	7,10	48,57	35,86
Різницева ентропійна	20,85	6,74	58,78	44,41	48,24	52,73	7,10	48,57	35,93

Таблиця 4. Час кодування файлів зображень набору ACT у формат PNG для різних кольорових моделей на комп'ютері з частотою процесора 300 МГц, с

Кольорова модель	№ файла з набору ACT								Разом
	1	2	3	4	5	6	7	8	
RGB	49,48	68,27	20,26	42,40	26.14	29,60	28,45	37,46	302,06
Різницева дисперсійна	51,25	72,06	20,76	49,77	29.55	39,99	30,04	42,01	335,43
в т.ч. перехід до моделі	1,38	2,58	0,66	0,83	0.54	0,88	1,16	0,93	8,96
Різницева ентропійна	51,02	71,52	20,71	49,77	29.06	39,88	29,66	41,42	333,04
в т.ч. перехід до моделі	1,26	2,47	0,61	0,82	0.55	0,83	0,99	0,88	8,41

Таблиця 5. Час декодування файлів зображень набору ACT з формату PNG для різних кольорових моделей на комп'ютері з частотою процесора 300 МГц, с

Кольорова модель	№ файла з набору ACT								Разом
	1	2	3	4	5	6	7	8	
RGB	3,84	4,01	2,31	3,13	2,25	3,57	1,59	3,46	24,16
Дисперсійна, ентропійна	4,01	4,01	2,36	3,29	2,31	3,57	1,59	3,46	24,60
в т.ч. перехід від моделі	0,33	0,00	0,17	0,16	016	0,17	0,00	0,17	1,16

Як видно з даних табл. 2, для стиснення різних зображень доцільно використовувати різні різницеві кольорові моделі, що розраховують згідно з розглянутими алгоритмами. Під час стиснення зображень № 2, 7 застосування предикторів виявилося недоцільним, що й зумовило необхідність повернення до кольорової моделі RGB. Найчастіше в різницевих кольорових моделях зменшується значення компонентів B та G, оскільки вони, як правило, мають найбільшу енергію. Тестування на цьому та інших наборах зображень свідчать, що дисперсійні та ентропійні кольорові моделі можна вважати еквівалентними, оскільки для 88 % файлів вони збігаються, а для решти хоча й відрізняються, але відхилення КС між ними не перевищує 1 % (причому ефективнішими навіть для подібних зображень можуть виявитися різні кольорові моделі). Дисперсійні кольорові моделі формуються в середньому на 6,5 % повільніше від ентропійних, оскільки під час розрахунку перші опрацьовують кожне відхилення окремо, а другі аналізують

лише абсолютні частоти відхилень, хоча й використовують для цього максимум 257 разів функцію логарифмування. Як свідчать експерименти, для зображень, що містять до 10000 пікселів, доцільно розраховувати різницеву кольорову модель за допомогою дисперсій, а для більших – з використанням прогнозованих ентропійних довжин кодів. Унаслідок застосування різницевих кольорових моделей у середньому по набору АСТ КС зменшився більше, ніж на 4,55 % в основному завдяки природним зображенням. Аналогічне зменшення КС отримано як для інших зображень, так і для інших форматів (JPEG-LS, BMF), що не виконують міжкомпонентну декореляцію. Застосування різницевих кольорових моделей може суттєво збільшити час кодування (до 35 %) не лише внаслідок розрахунків параметрів кольорової моделі, а й через орієнтацію на контекстно-незалежний алгоритм, що кодує окремі елементи. Зате час декодування зображень внаслідок використання цих кольорових моделей збільшується лише на десяті долі секунди, що в сукупності з суттєвим зменшення КС дає змогу ефективно використовувати їх на практиці.

Для досягнення кращих КС **у подальшому** доцільно розробити алгоритм фрагментування зображень на області однакових різницевих кольорових моделей та алгоритм компактного зберігання таких областей.

З наведених результатів дослідження можна зробити такі **висновки**:

1. Підвищити КС зображень у трикомпонентних кольорових моделях можна не лише внаслідок декореляції даних окремих компонентів, а й за допомогою міжкомпонентної декореляції.

2. Міжкомпонентну декореляцію слід виконувати так, щоб покращити властивості зображення, що використовуються алгоритмами препресингу та безпосереднього стиснення обраного графічного формату.

3. У випадку використання лінійних предикторів для стиснення зображень без втрат застосування різницевих кольорових моделей дає змогу покращити КС в середньому більше, ніж на 4,5%. Для природних зображень КС може зменшитися більше, ніж на 12% (див., наприклад, дані зображення № 6 в табл. 3). Формування різницевих кольорових моделей відображає процес пошуку площин, стосовно яких енергія приростів яскравостей є мінімальною.

4. Різницеві кольорові моделі дають змогу суттєво підвищити ефективність стиснення без втрат трикомпонентних природних зображень у форматах, що використовують лінійні предиктори, і тому можуть бути впроваджені в наступні версії цих форматів на рівні стандартів.

1. *Бредихин Д. Ю.* Сжатие графики без потерь качества [Електронний ресурс]. – 2004. – http: //www.compression.ru/ download/ articles/ i_lless/ bredikhin_2004_lossless _image_-compression_ doc.rar
2. *Методы* сжатия данных. Устройство архиваторов, сжатие изображений и видео / Д. Ватолин, А. Ратушняк, М. Смирнов, В. Юкин. – М.: ДИАЛОГ-МИФИ, 2003. – 384 с.
3. *Миано Дж.* Форматы и алгоритмы сжатия изображений в действии: Учеб. пос. – М.: Триумф, 2003. – 336 с.
4. *Седов С. А.* Индивидуальные видеосредства: телеантенны, телевизоры, видеомагнитофоны, видеокамеры, видеопроигрыватели, видеодиски: Справоч. пос. – К.: Наук. думка, 1990. – 752 с.
5. *Сэломон Д.* Сжатие данных, изображений и звука. – М.: Техносфера, 2006. – 336 с.
6. *Шпортько О. В.* Оптимізація використання статичних предикторів у процесі стиснення зображень без втрат // Відбір і обробка інформації. – 2008. – № 28(104). – С. 82–89.
7. *Ziv J., Lempel A.* A universal algorithm for sequential data compression // IEEE Transactions on Information Theory. – May, 1977. – Vol. 23(3). – P. 337–343.

Рівненський державний гуманітарний університет

Одержано 11.03.2009