УДК 621.397

Шутко М.О., д.т.н., проф.;
Кушнір О. М., аспірант

СПЛАЙН-ФУНКЦІЇ В АЛГОРИТМАХ СТИСНЕННЯ ПОТОКОВОГО ВІДЕО

В даній статті проаналізовано сучасні методи стиснення потокового відео і обґрунтовано переваги використання сплайн-функцій в даних алгоритмах.

В данной статье проанализированы современные методы сжатия потокового видео и обоснованно преимущества использования сплайн- функций в данных алгоритмах.

In this article the modern methods of compression of streaming video are analysed and advantage of the use of spline functions in these algorithms are grounded.

Вступ. Під потоковим відео розуміють технології стиснення і буферизації даних, які дозволяють передавати відеоінформацію в реальному часі по локальним комп'ютерним мережам та через мережу Internet. Цифрова відео індустрія завдячує своєму розвитку стандарту MPEG-2, який на сьогодні є морально застарілим. В даний момент створені більш досконалі стандарти відео кодування, наприклад, H.264/AVC. Не зважаючи на, застосування нових методів кодування, постійне зростання об'єму пам'яті носіїв інформації та збільшення пропускної здатності каналів зв'язку, актуальною залишається задача підвищення ефективності алгоритмів стиснення відеоматеріалу, оскільки вимоги до якості відео ростуть разом з доступними обчислювальними ресурсами.

Аналіз досліджень і публікацій. Зберігання та передача цифрових даних пов'язані з задачами стиснення. Методи стиснення можна розділити на дві основні групи [3]:

- стиснення з точним відновленням інформації;
- стиснення з певним рівнем втрат (lossy).

Алгоритми стиснення зображень без втрат інформації ґрунтуються на принципі статичної надлишковості. В даних алгоритмах використовуються оптимальні способи кодування, наприклад, групове кодування (RLE-Run Length Encoding) [2], словниковий метод Лампела-Зіва-Велча (LZW-Lempel-Ziv-Welch)[4], кодування Хафмена (CCITT). При цьому досягається коефіцієнт стиснення в 1,5-4 рази.

Друга група методів ґрунтується на візуальній надлишковості , якої можна позбутися за рахунок часткової втрати інформації. Під час розробки цих алгоритмів використовують методи апроксимації даних, адаптивну

дискретизацію та певні фізіологічні властивості сприйняття відео зображень людиною. Стиснення сягає 10-20 разів з прийнятною якістю зображень[5]. Звернемо нашу увагу на останню групу. В ній можна виділити два основні напрямки:

Wavelets методи, що мають прекрасне теоретичне обґрунтування , масу ілюстрованих прикладів і майже не мають практичних програмних реалізацій;

Евристичні методи, що практично не мають теоретичного обґрунтування, спираються на фізіологічні особливості людського сприйняття і фактично стали алгоритмічним стандартом (JPEG алгоритм).

Постановка завдання. Стиснення даних є надзвичайно важливою задачею, в зв'язку із інтенсивним розвитком комп'ютерних засобів комунікації. Саме тому проблеми підвищення ефективності програмних засобів стиснення відео зображень є актуальним напрямком досліджень. Перспективним є використання сплайн-функцій для створеня нових методів стиснення зображень, що мають вищий коефіцієнт стиснення та кращу швидкодію декомпресії, ніж ті математичні методи, що використовуються у відомих фото- та відеокодеках JPEG, JPEG2000, MPEG4, DivX 6.x, H.263.

Сплайн-функції в алгоритмах стиснення потокового відео.

Відеоінформація являє собою тривимірний масив кольорових пікселів. При цьому два виміри – це горизонтальне та вертикальне розподілення кадрів, а третій – часовий. Кожен кадр, таким чином, являє собою масив пікселів на даний момент часу, тобто, фактично, зображення.

Типові алгоритми стиснення відео починають зі стиснення першого кадру методами стиснення зображень. Далі виявляється та кодується інформація про відмінності наступного кадру від попереднього. Кадри, що істотно відрізняються від попереднього кодуються окремо.

Більшість сучасних методів стиснення використовують дискретне косинусоїдальне перетворення (ДКП) або його модифікації для усунення просторової надмірності. Інші методи, такі як фрактальне стиснення та дискретне вейвлет-перетворення , також були об'єктами досліджень, але зараз зазвичай використовуються тільки для компресії нерухомих зображень.

Зупинимося більш детально на алгоритмі стиснення даних JPEG. Оперує алгоритм областями 8x8, на яких яскравість і колір міняються порівняно плавно. Внаслідок цього при розкладанні матриці такої області в подвійний ряд по косинусах значущими виявляються тільки перші коефіцієнти. Таким чином, стиснення в JPEG здійснюється за рахунок плавності зміни кольорів в зображенні.

В цілому алгоритм заснований на дискретному косинусоїдальному перетворенні, вживаному до матриці зображення для отримання деякої нової матриці коефіцієнтів. Для отримання початкового зображення застосовується зворотне перетворення.

ДКП розкладає зображення по амплітудах деяких частот. Таким чином, при перетворенні ми одержуємо матрицю, в якій багато коефіцієнтів або близькі, або рівні нулю. Крім того, завдяки недосконалості людського зору можна апроксимувати коефіцієнти грубіше без помітної втрати якості зображення.

Для цього використовується квантування коефіцієнтів (quantization). У найпростішому випадку - це арифметичне побітове зсунення вправо. При цьому перетворенні втрачається частина інформації, але може досягатися великий ступінь стиснення.

Отже, розглянемо алгоритм докладніше (рис. 1). Нехай ми стискаємо 24-бітове зображення.

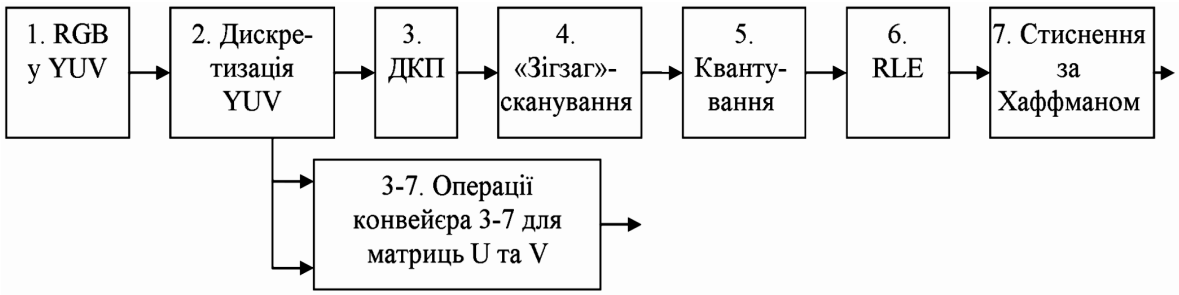

Рис. 1. Конвеєр операцій, який використовується в алгоритмі JPEG

Кроки шифратора JPEG:

1) Плавне перетворення колірного простору: [R G B] $\rightarrow$ [Y Cb Cr]

(*R,G,B* - 8-бітові величини без знака):

$$\begin{vmatrix} Y \\ Cb \\ Cr \end{vmatrix} = \begin{vmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.4187 & -0.0813 \\ 0.1687 & -0.3313 & 0.5 \end{vmatrix} * \begin{vmatrix} R \\ G \\ B \end{vmatrix} + \begin{vmatrix} 0 \\ 128 \\ 128 \end{vmatrix}$$

Колірний простір *Y Cb Cr* іноді ще називають *YUV*.

Нова величина

$Y = 0.299*R + 0.587*G + 0.114*B$ названа яскравістю.

Величини

$Cb = 0.5 * R - 0.4187 * G - 0.0813 * B + 128$ і

$Cr = - 0.1687 * R - 0.3313 * G + 0.5 * B + 128$ названі колірними величинами і представляють 2 координати в системі, що вимірює відтінок і насичення кольору.

Ці 2 координати коротко названі кольоророзністю.

2) Дискретизація

JPEG Стандарт бере до уваги те, що око більш чуттєве до яскравості кольору, чим до відтінку цього кольору.

Так, для більшості JPG, яскравість узята для кожного піксела, тоді як кольоророзність – як середня величина для блоку 2x2 пікселей.

$$Cb = 0.25 * \sum U_\tau ; \qquad Cr = 0.25 * \sum V_\tau$$

$\tau = 1...4;$ $\tau = 1...4$

Зсунення рівня. Усі 8-бітові величини без знака (Y, Cb, Cr) у зображенні - "зміщені за рівнем": вони перетворюються в 8-бітове знакове представлення вирахуванням 128 з їхньої величини.

3) 8x8 дискретне косинусоїдальне перетворення (DCT)

Зображення поділяється на блоки 8x8 пікселей. Потім для кожного блоку 8x8 застосовується DCT-трансформація. Блоки 8x8 обробляються зліва направо і зверху вниз.

Математичне визначення прямого DCT (FDCT) і зворотного DCT (IDCT):

FDCT:

$$F(u,\nu) = \frac{c(u,\nu)}{4} \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) cos\left(\frac{2x+1}{16} u\pi\right) cos\left(\frac{2y+1}{16} \nu\pi\right)$$

u,v = 0,1...7

c(u,v)=1/2, коли u = v = 0;

c(u,v)= 1 – в інших випадках.

IDCT:

$$f(u,\nu) = \frac{1}{4} \sum_{x=0}^{7} \sum_{y=0}^{7} c(u,\nu) F(u,\nu) cos\left(\frac{2x+1}{16} u\pi\right) cos\left(\frac{2y+1}{16} \nu\pi\right)$$

x, y = 0,1...7

4) Зигзагоподібна перестановка 64 DCT коефіцієнтів

Зміст складається в тому, що ми переглядаємо коефіцієнти 8x8 DCT у порядку підвищення просторових частот. (Числа в блоці 8x8 указують порядок, у якому ми переглядаємо 2-мірну матрицю 8x8.)

Результат - відсортований вектор з 64 коефіцієнтами (0...63).

5) Квантування

На цьому етапі, у нас є відсортований вектор з 64 величинами, що відповідають амплітудам 64 просторових частот у блоці 8x8.

Далі, кожна величина поділяється на число, визначене для вектора з 64 величинами - таблицю квантування, потім округляється до найближчого цілого:

для (i = 0; i<=63; i++)

вектор[i] = (округлити) (вектор[i] / таблиця_квантування[i] + 0.5)

6) RunLenght кодування нулів (RLE)

Тепер у нас є квантований вектор з довгою послідовністю нулів. Ми можемо використовувати це, кодуючи послідовні нулі.

7) Кінцевий крок - кодування Хаффмана

Це є стандартний конвеєр операцій, який використовується в алгоритмах стиснення зображень (на прикладі алгоритму JPEG). Основною операцією при стисненні з втратами є ДКП в алгоритмі JPEG (рис. 1), або вейвлет-розклад у JPEG2000.

Для покращення існуючих алгоритмів стиснення даних пропонується на третьому кроці конвеєра проводити сплайновий багатомасштабний розклад.

Так як сплайни – кусково-поліноміальні функції, то вони легко можуть бути використані при обчисленнях. Дійсно, алгоритми для графічного зображення кривих з допомогою сплайнів та для обчислення їх поліноміальних складових надзвичайно ефективні [1]. Більш того, так як сплайни мають найменший можливий носій, то можуть застосовуватись схеми локальної інтерполяції для апроксимації функцій у $C \bigcap L^2(\mathbf{R})$ з допомогою будь-якого сплайн-підпростору V_j. Усі такі алгоритми можуть використовуватись у режимі реального часу.

У вейвлет-алгоритмах розклад "шарами" ведеться зазвичай із кратністю 2, оскільки реалізація розкладу з кратністю не 2, або зі змінною кратністю потребує значних ускладнень та часових затрат на обробку.

Розробка сплайнового багатомаштабного аналізу (БА) дозволить під час розрахунку матриці планування за відомими формулами на кожному етапі отримати швидку реалізацію БА з кратністю не 2, або зі змінною кратністю. Тому таку сплайн-фільтрацію будемо називати багатомасштабною, а не кратномасштабною.

Нехай початкові дані представлені N дискретними відліками. За допомогою апроксимації сплайном згідно з формулами

$S_3(x_i) = \hat{a}_{j-2}{}^1X_{ij} + \hat{a}_{j-1}{}^2X_{ij} + \hat{a}_j{}^3X_{ij} + \hat{a}_{j+1}{}^4X_{ij}$,

$i = \overline{1+m_{j-1}, m_j}$, $j = \overline{1,r}$, $a_{-1} = a_{r+1} = 0$.

знаходимо вихідну функцію *f(t)* (рис. 2).

Розглянемо спочатку приклад сплайн–розкладу із кратністю 2. Першим кроком буде проріджування вузлів "склеювання" сплайна у два рази, тобто матимемо $N/2$ вузлів. Апроксимація значень функції у цих $N/2$ вузлів дасть схожу функцію, але, звичайно з деякими похибками. Для збереження інформації про похибки знаходимо різниці між значеннями початкової та нової функцій у N вузлах. Частина таких різниць буде достатньо мала щоб можна було ними знехтувати. Тобто потрібно встановити поріг, нижче якого значення різниць приймаються рівними нулю. Кількість вагомих (тобто ненульових) деталізуючих коефіцієнтів першого рівня позначимо $\det_1$. Тоді результатом першого кроку проріджування буде $N/2 + \det_1$ значень, які потрібно зберігати для можливого відновлення початкової функції (рис.3).

Другий крок. Знову аналогічно проріджуємо вузли склейки сплайна, отриманого після першого кроку. Тоді замість $N/2$ значень матимемо $N/4+\det_2$. І відповідно після другого кроку маємо для зберігання $N/4+\det_2+\det_1$ коефіцієнтів (рис. 4).

Відповідно третій крок дасть $N/8+\det_3+\det_2+\det_1$ значень (рис.5).

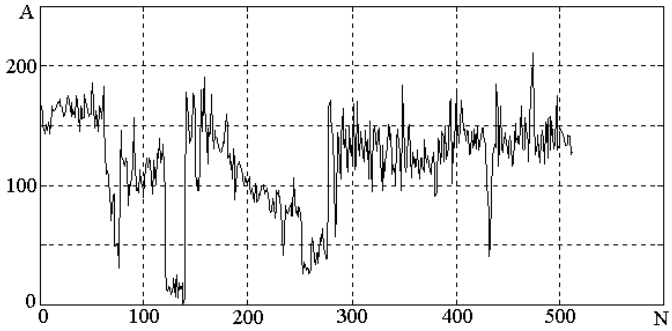

Рис.2 Оброблюваний сигнал

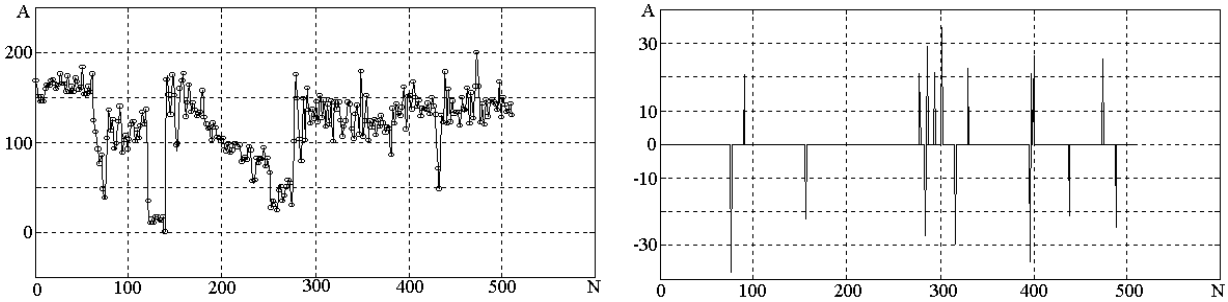

Рис.3 Сплайн-апроксимація сигналу і деталізуючі коефіцієнти після першого кроку.

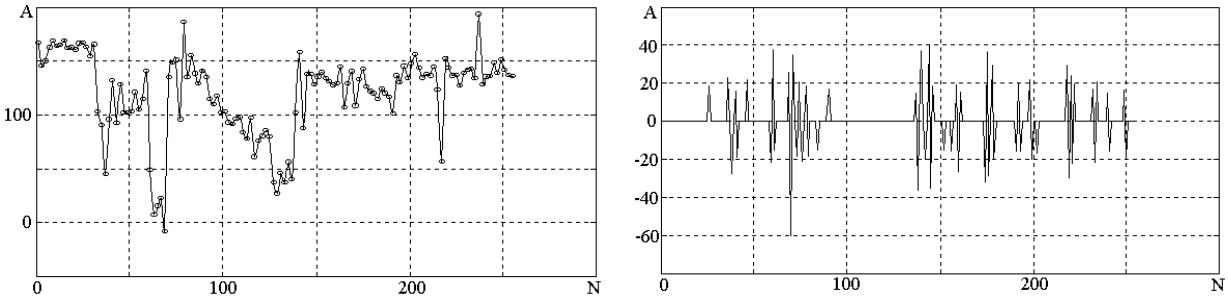

Рис.4 Сплайн-апроксимація сигналу і деталізуючі коефіцієнти після другого кроку.

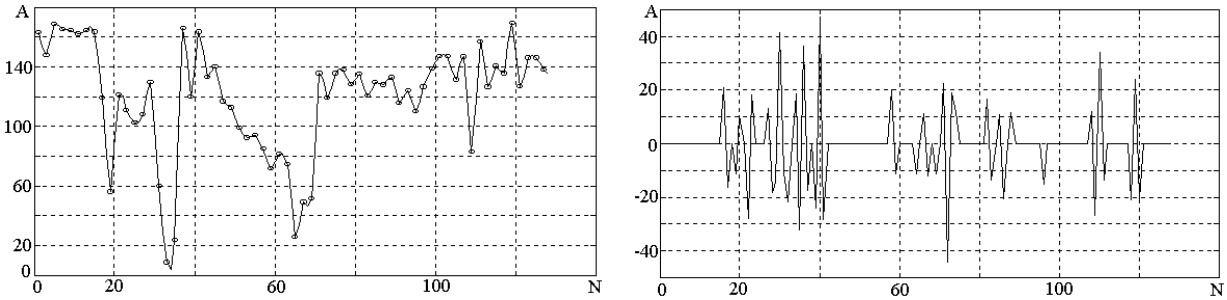

Рис.5 Сплайн-апроксимація сигналу і деталізуючі коефіцієнти після третього кроку.

Відновлення початкового сигналу відбувається в зворотному порядку:

1. До значень коефіцієнтів у вузлах склейки сплайну додаються деталізуючі коефіцієнти $\det_3$ та виконується інтерполяція. Маємо сигнал другого кроку стиснення.

2. Аналогічно додаються $\det_2$ та інтерполюються одержані коефіцієнти. Встановлено сигнал першого кроку стиснення.

3. Додаванням $\det_1$ та інтерполяцією відновлюється початковий сигнал.

Після будь-якого кроку ми можемо відразу ж відновити вихідні дані. Звичайно із збільшенням кількості кроків похибка буде накопичуватись.

Неважко побачити, що аналогічно до наведеного алгоритму можна побудувати алгоритм з іншою кратністю. Наприклад наведемо алгоритм з дробним кроком, тоді кількість коефіцієнтів буде:

I. $\frac{N}{2{,}83} + \det_1$.

II. $\frac{N}{(2{,}83)^2} + \det_2 + \det_1$.

Великою перевагою є те, що така процедура дає змогу змінювати кратність крок від кроку (наприклад, можна з кожним наступним кроком кратність зменшувати).

Вказаний підхід дає можливість варіювати алгоритм в широких межах підбираючи його параметри під кожний тип сигналів так, щоб результат був найкращим.

Крім того, при застосуванні сплайнів істотно зменшується обсяг обчислень. Адже це – прості функції з малим носієм, які найбільш ефективні як при їх програмній, так і технічній реалізації.

Висновки.

Таким чином, застосування сучасних методів кодування не дозволяє в повній мірі уникнути появи характерних спотворень. Тому використання і розробка алгоритмів стиснення потокового відео з використанням сплайн-функцій є актуальною задачею. Оскільки, застосування даних алгоритмів може покращити якість декомпресованих зображень або збільшити коефіцієнт стиснення при тій самій якості.

Використані джерела інформації:

1. *Алберг Дж., Нильсон Э., Уолш Дж.* Теория сплайнов и ее приложения. - М.: Мир, 1972. - 316 с.
2. *Ватолин Д., Ратушняк А., Смирнов М.* Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео//-М - Диалог-МИФИ.-2003.-384с.
3. *Мюррей Д., У.ван Райнер*. Энциклопедия форматов графических файлов. - К.:BHV,1997. -678 с.
4. *Сэломон Д.* Сжатие даных, зображений и звука//-М.-Техносфера.-2004.-368с.
5. *Шелевицький І. В.* Методи та засоби сплайн-технологій обробки сигналів складної форми .-Кривий Ріг:Європейський університет,2002.-304с.

Рецензент: д.т.н. Лисенко О.І.