ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Обробка зображень та мультимедіа

Індивідуальне завдання 1

Виконав

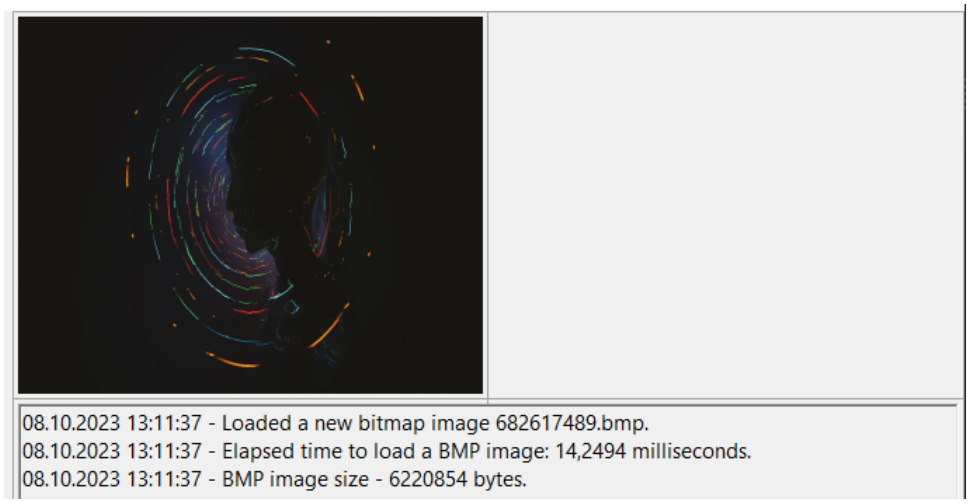
Студент групи ПМІ-42

Юрас Назар

2023

Обрав ось такий BMP файл з глибиною кольору 24 біти:

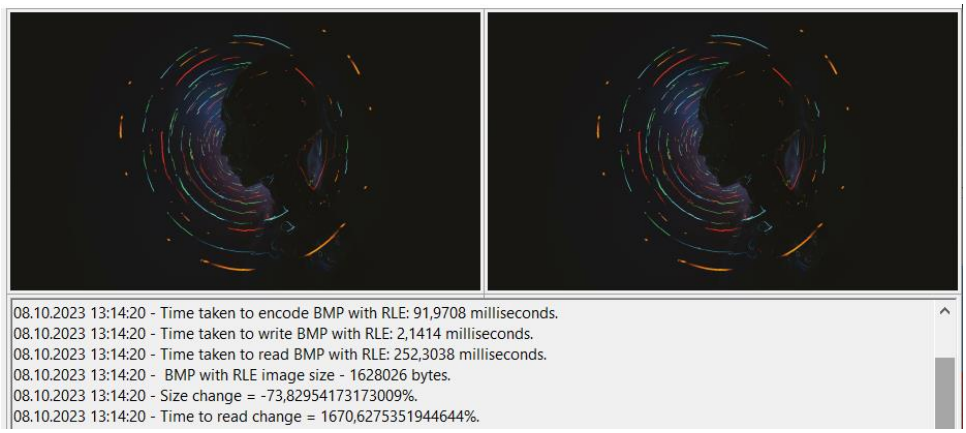
Відкриваю його у програмі:



Для його відкриття знадобилось 14,2 мілісекунди з розміром приблизно 6.22 МБ і розширенням 1920x1080 пікселів.

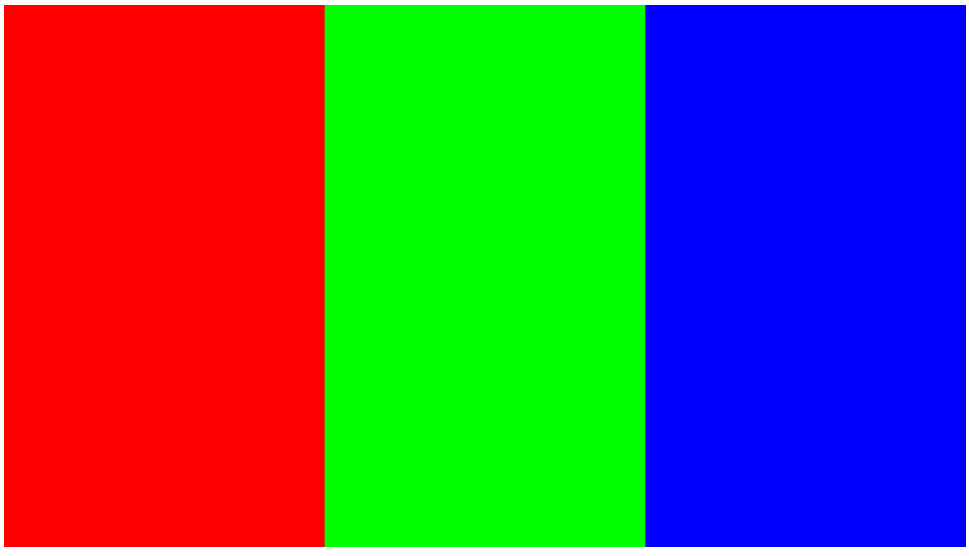
Стиснемо його за допомогою методу RLE і збережемо у форматі RLE.

RLE - алгоритм без втрат, тому ми можемо отримати початкове зображення без втрат якості

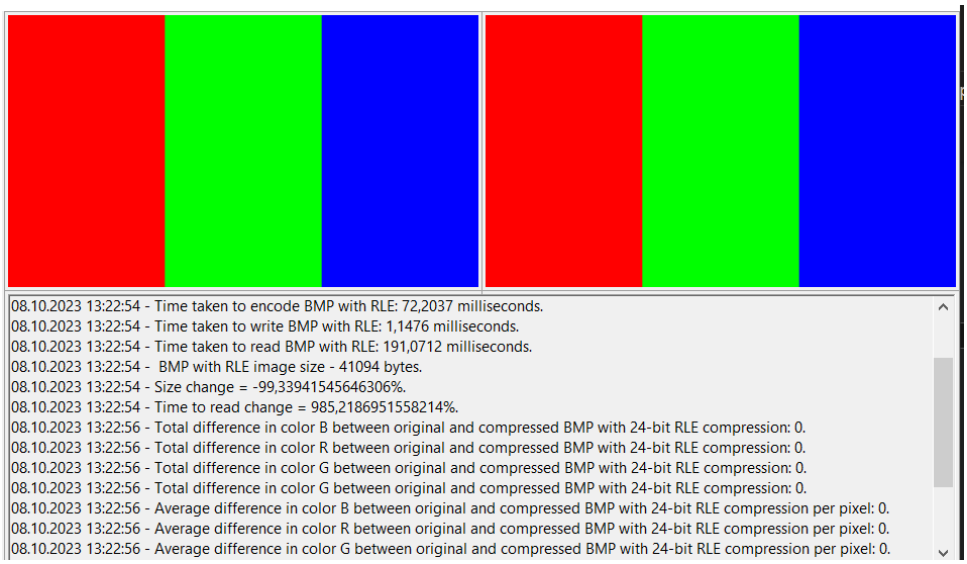


Нам вдалось зменшити розмір зображення до 1.63 МБ (на цілих 74%!!), проте час на читання збільшився у майже 17 разів, тому що нам необхідний час на перетворення зі зменшеного формату у стандартний BMP.

Найкраще ж цей алгоритм працює з зображеннями, на яких є великі послідовності однакових кольорів підряд, наприклад:

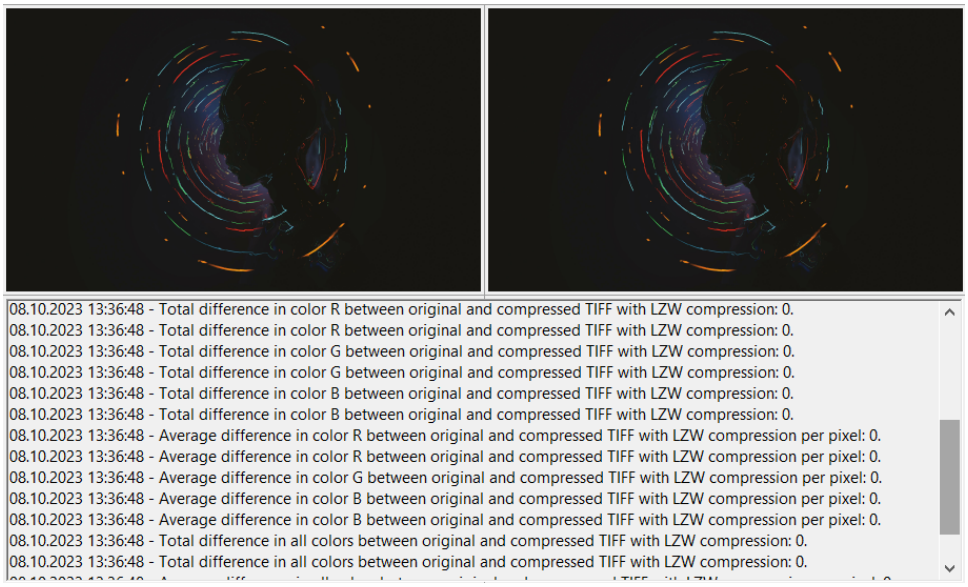


Розмір цієї картинки 5.92 МБ і тут якраз є великі послідовності однакових кольорів



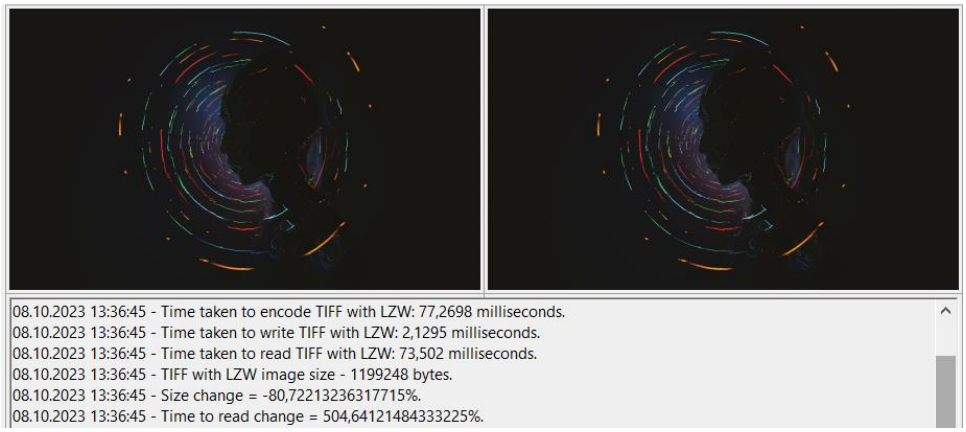
Нам вдалось зменшити розмір файлу з 5.92 МБ до 41 000 байт (на 99.34%) і через зменшений розмір навіть швидше читається (на 700% в порівнянні з попереднім зображенням) і швидше перетворюється на оригінал.

Те ж саме робимо і для формату TIFF для стиснення LZW:



Як видно на скріні, цей алгоритм так само як і RLE дозволяє зберігати зображення без втрат якості.

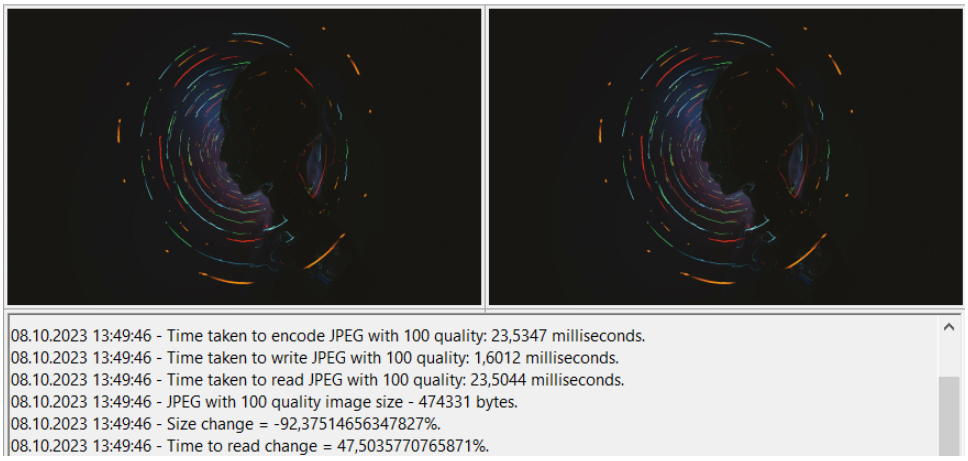
Тут також вийшло значно зменшити розмір зображення. Через необхідність розтискання файлу час на читання так само збільшився, тільки зараз вже не на такий значний відсоток.



Знову ж таки, так само як і RLE алгоритм LZW дуже добре працює для зображень з великою кількістю однакових пікселів підряд, тому візьму те ж зображення що і вище

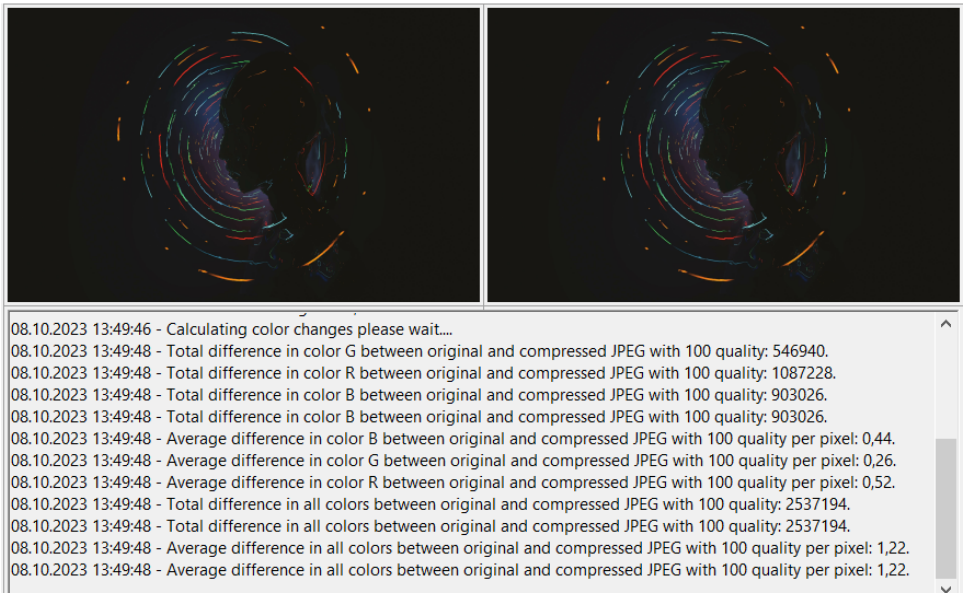
Тут розмір вдалось зменшити на 99,7%, також час на читання зріс у 5 разів

JPEG: робимо те ж що і для попередніх алгоритмів.



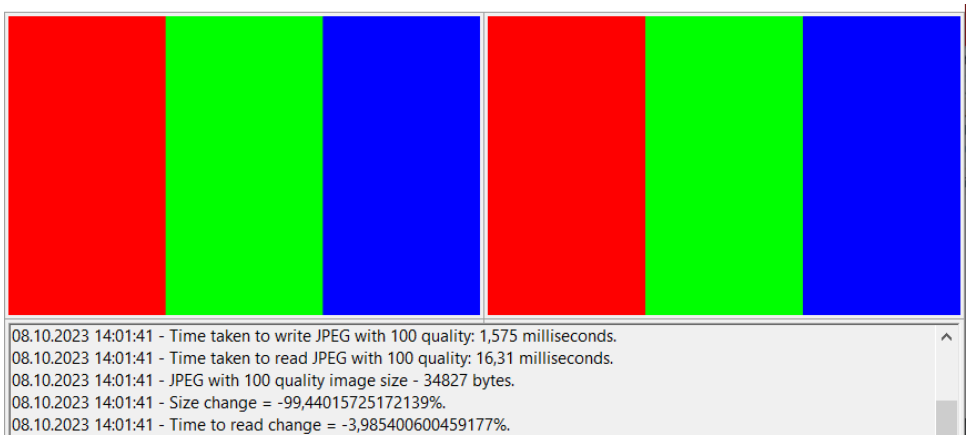
Вдалось зменшити розмір зображення на 92%, час на читання збільшився тільки на 47% (найменше зі всіх розглянутих варіантів).

Для початку виставимо параметр якості на 100.

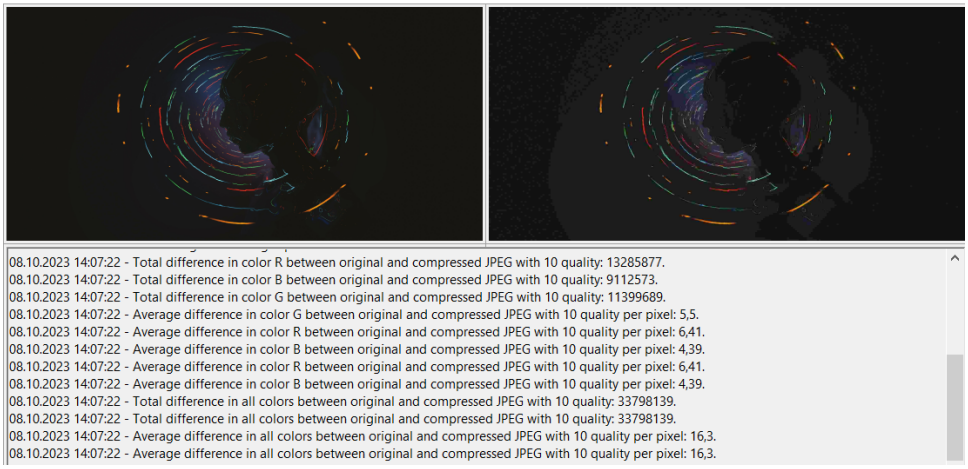


Як видно, з стандартним енкодером і параметром якості 100 відбулась втрата якості зображення, хоч і незначна (1,22 біти на піксель).

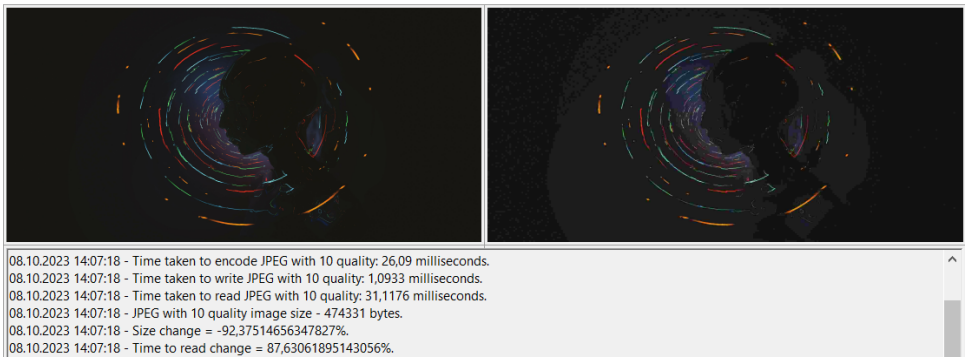
На відміну від двох попередніх алгоритмів, він не показує кращої ефективності для картинок з великою кількістю однакових пікселів підряд (через втрати), хоча час на читання трішечки зменшився.



Розглянемо наше зображення, тільки параметр якості вже виставимо на 10.



Тут вже очевидна різниця у якості зображень, зміщення по кольорах більше 16 бітів на піксель.



Також тут очевидне і дуже значне зменшення розміру картинки.

Висновок: отже, якщо нам важливо взагалі не втратити якості зображення то можна використовувати алгоритми стиснення RLE та LZW (на основі проведених тестів LZW трішки ефективніший).

Якщо ж ми знаємо, що наше початкове зображення містить багато

пікселів однакового кольору підряд, тоді також доцільно використовувати RLE та LZW, оскільки для таких зображень вони дуже ефективно працюють і що не менш важливо без втрат якості зображення.

Зручність ж JPEG полягає у тому, що ми самі можемо вибирати, наскільки якісний результат нам потрібен. Наприклад, з параметром якості 100 ми можемо отримати значне зменшення розміру зображення з мінімальними втратами. З меншим параметром якості ми зможемо в рази зменшити розмір зображення, проте якість зображення значно погіршиться і зміни буде видно неозброєним оком. Ще однією перевагою є ефективність роботи з абсолютно всіма зображеннями, а не тільки з тими, на яких є велика послідовність кольорів однакового кольору підряд.