]SOFTWARE AND UI DEVELOPING[

Порівняльний аналіз графічних ефектів.
Алгоритми розмивання за Гаусом.

І. Вступ

Мета роботи – розібрати основні способи редагування зображення за допомогою графічних фільтрів на прикладі матричного фільтра Гауса. А також, провести аналіз даних методів та порівняти отримані результати з результатом алгоритму, що реалізовано у Adobe Photoshop CC.

Розв'язуючи проблему зменшення чіткості зображення, використовують розмивання Гауса. Так, під час друку відсканованого зображення воно виявляється дуже різким, тоді зображення розмивають та отримують після друку набагато реалістичнішу картинку:

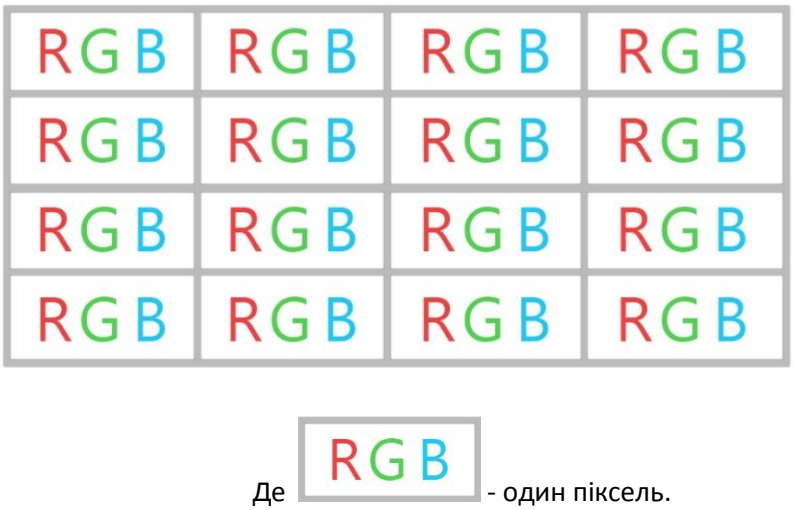
Оригінал	Розмите зображення (r = 1 pix)
	

Також, нерідко розмивання Гауса використовують у дизайні інтерфейсів користувача, дизайні веб-сторінок, або просто як елемент ОС. Найбільш ефектно це виглядає у поєднанні з мінімалізмом. Сильне розмивання певної частини зображення створює дуже м'який градієнт, що завжди «лаконічно» вписується в картину.

II. Комп’ютерне зображення

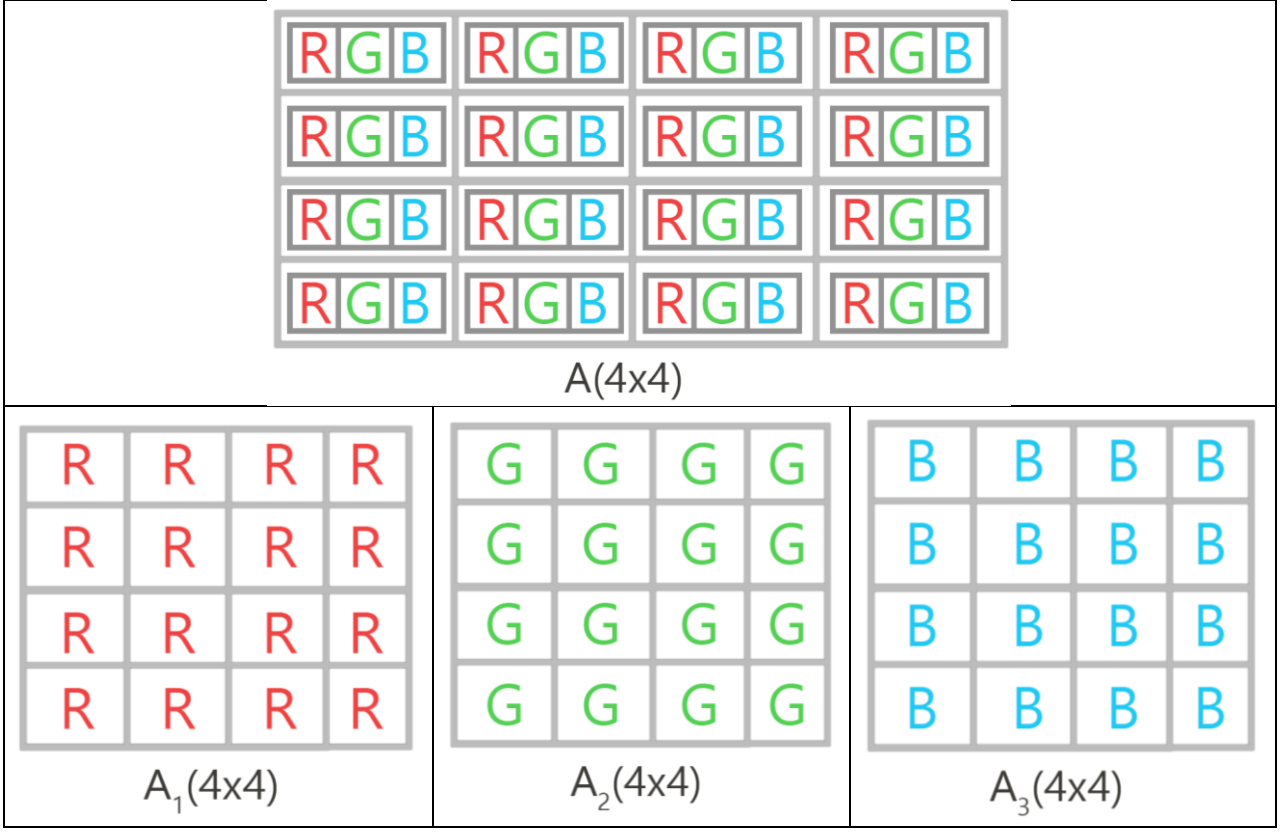
Оскільки далі мова йтиме про комп’ютерну графіку, нам необхідно визначити, що таке зображення та з чого воно складається. Кожне зображення складається з окремих самостійних частинок – пікселів. А піксель, в свою чергу, характеризується (або може характеризуватися) трьома значеннями: R(red), G(green), B(blue). Цим однозначно ставиться у відповідність один з можливих відтінків від поєднання червоного, зеленого та синього кольорів, взятих у певних пропорціях.

Приклад пікселів растрового зображення



Будемо вважати, що зображення складається з трьох матриць однакових розмірів, елементами яких є тільки одне з значень R, G або B.

Представлення растрового зображення у вигляді трьох матриць



Таким чином, піксель A(1;1) зображення A складається з: A_1(1;1), A_2(1;1) та A_3(1;1).

III. Матриці згортки. Методи модифікації зображення за допомогою матриць згортки

Оскільки ми визначили зображення як систему з трьох матриць, то найбільш ефективний та зручний спосіб накласти певний ефект – це застосувати матрицю згортки. В даній ситуації, доцільніше використати наступне визначення.

Матрицею згортки називається деяка матриця K(m x n), де m та n – непарні коефіцієнти, заповнена числами за певним законом, що визначає майбутній ефект від згортки. Ядром матриці називається елемент, розташований на перетині $r_i = \frac{m-1}{2} + 1$ рядка та $r_j = \frac{n-1}{2} + 1$ стовпця матриці К. Умовно позначимо його $К_с$. Так, кожен елемент матриці визначає «важливість» (вагу) співставленого з ним пікселя вхідного зображення. Зазвичай, ядро має найбільшу вагу, та це – не обов'язкова умова.

1	0.5	0	0.5	1
2	1	0.5	1	2
5	2.5	1 Kc	2.5	5
2	1	0.5	1	2
1	0.5	0	0.5	1

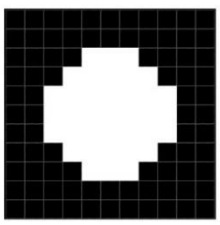
Приклад матриці згортки $К_1$(5x5)

Згортку зображення можна умовно поділити на 3 етапи (для кожного пікселя):

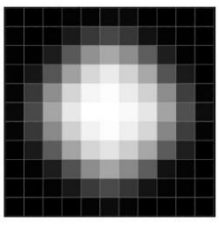
1. Накладання на піксель вхідного зображення матриці згортки так, щоб цей піксель співпав з ядром матриці; обчислення нового значення пікселя, виходячи з ваги оточуючих його пікселів та їх значень для кожного параметру R, G та B;
2. Обчислення коефіцієнта нормалізації;
3. Нормалізація пікселя.

Такий алгоритм вимагає зробити неймовірно багато обчислень, що негативно впливає на час обробки зображення. Тому, на практиці широко використовують не двовимірні матриці згортки, а одновимірні. Такий підхід на порядок зменшує кількість обчислень, та несуттєво впливає на результат. Використання одновимірних матриць при розмиванні за Гаусом дещо зменшить «чесність» результату, та це майже неможливо побачити навіть при значному збільшенні зображення:

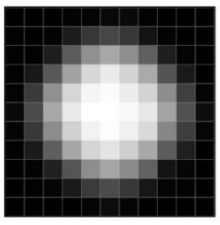
Результат розмивання зображення за Гаусом:



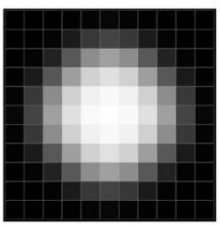
Оригінальне зображення (11x11)



Обговорюваний алгоритм (радіус 1 піксель, 1D)



Обговорюваний алгоритм (радіус 1 піксель, 2D)



Adobe Photoshop (радіус 1 піксель, 2D)

Описаний «одновимірний алгоритм» має похибку ~0.5 - 1%

**Тут і надалі вважаємо результат розмивання зображення за Гаусом засобами програми Adobe Photoshop – еталоном.*

За такого підходу, ми маємо деякі незначні зміни:

1. Матриця згортки K_2([m = 1] x n) матиме вигляд:

5	2.5	1 Kc	2.5	5

2. Ядро матриці буде розташовуватись у $r_j = \frac{n-1}{2} + 1$ стовпці
3. Необхідно буде опрацювати зображення двічі. Спочатку зліва направо, потім згори донизу, чи навпаки (це не має значення).

IV. Алгоритми формування та заповнення матриці згортки за розподільним законом Гауса

Отже, тепер, коли нам відомі основні два методи згортки зображення, ми повинні створити матрицю згортки саме для розмивання за Гаусом. Приведемо алгоритми формування одновимірних та двовимірних матриць згортки.

Для одновимірної матриці ми використаємо вектор-рядок K, що вміщує $(3 \cdot \sigma) \cdot 2 + 1$ елементи (множник 3 біля σ відображає правило трьох сигм) та наступну формулу щільності розподілу:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{x^2}{2\sigma^2}}$$

Де σ – середньоквадратичне відхилення випадкової величини, тобто, у даному випадку, бажаний радіус розмивання.

Заповнимо наш вектор-рядок за наступним правилом:

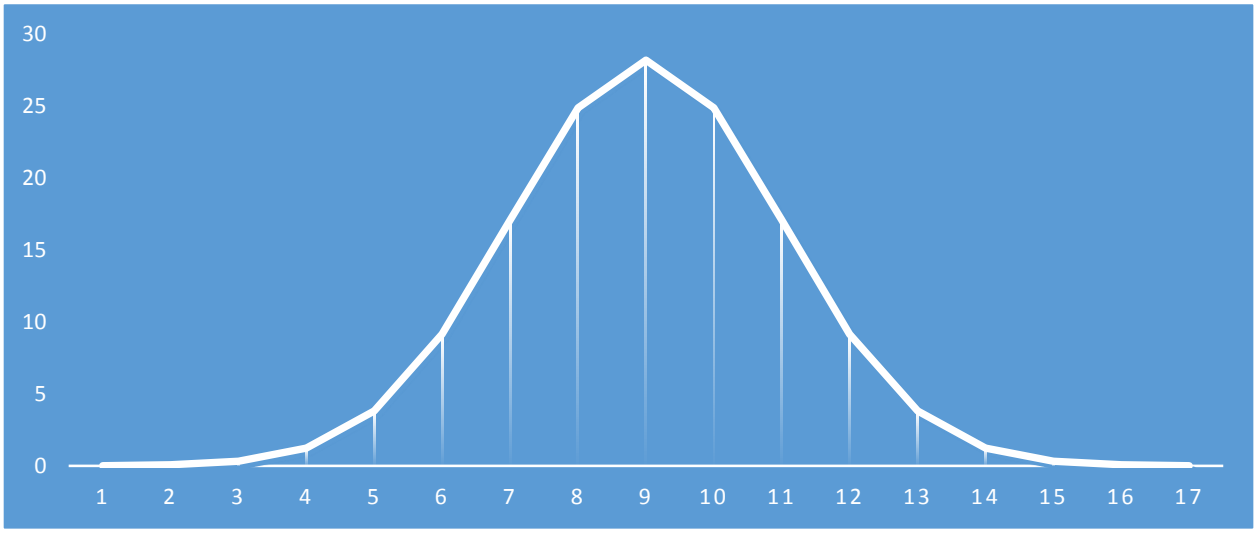
$$K(j) = \begin{cases} f(r_j - j),\ j \le r_j \\ f(j - r_j),\ j > r_j \end{cases}$$

Для *(σ = 1)* отримаємо матрицю вигляду:

0.009	0.061	0.313	1.239	3.817	9.158	17.109	24.894	28.209	24.894	17.109	9.158	3.817	1.239	0.313	0.061	0.009

Матриця згортки, заповнена за нормальним розподілом. *Кожен елемент помножено на 100.*

Якщо побудувати графік з елементів матриці, побачимо, що ядро має найбільшу вагу, а сусідні елементи - тим меншу, чим вони далі від ядра:



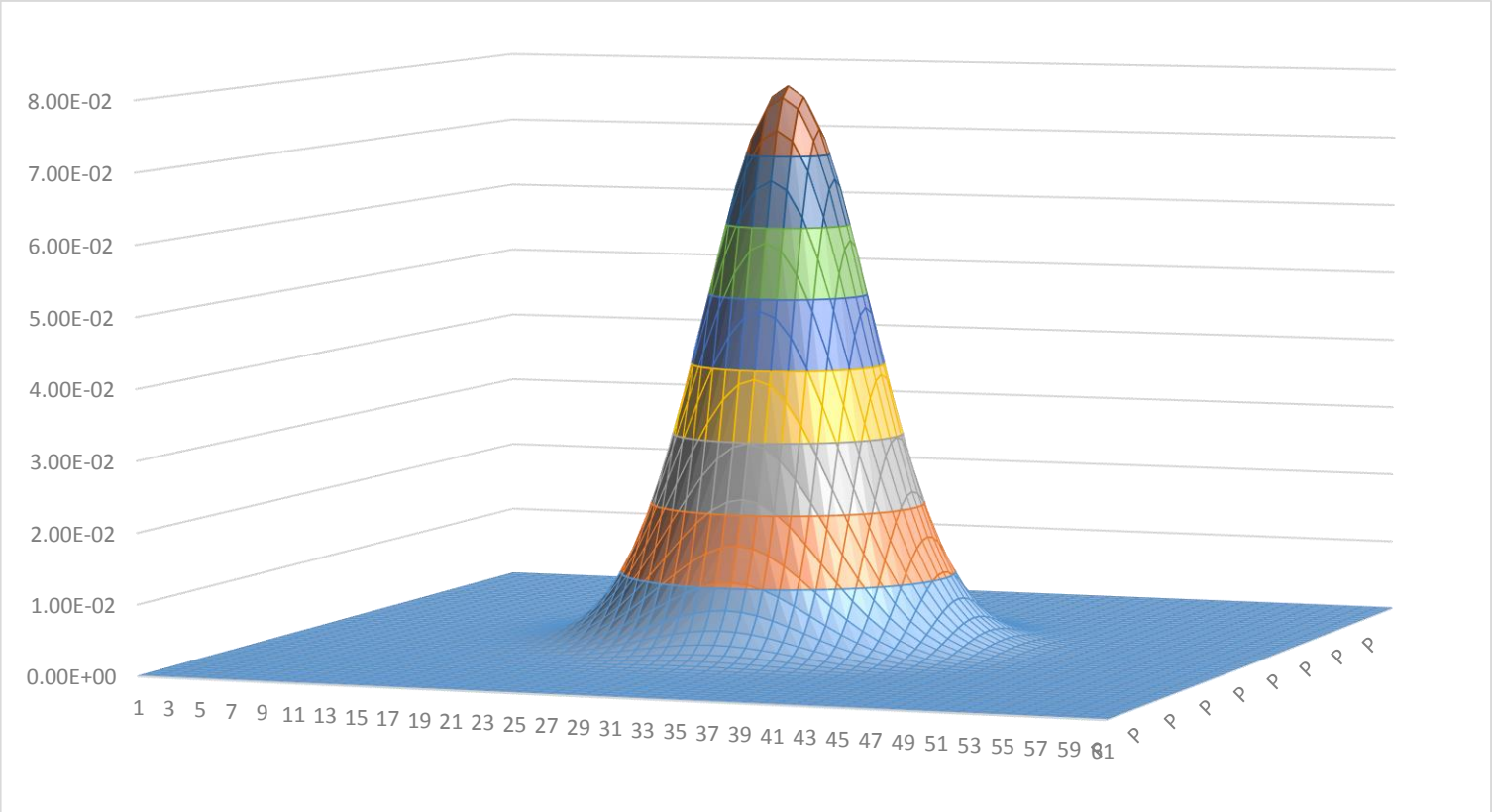
Створення двовимірної матриці згортки потребує більше часу та обчислень. Для цього необхідно створити матрицю K($(3 \cdot \sigma \cdot 2 + 1)$ x $(3 \cdot \sigma \cdot 2 + 1)$) та заповнити її за наступним законом:

$$f(x, y) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$K(i,j) = \begin{cases} f(r_i - i, & r_j - j), & i \le r_i, \ j \le r_j \\ f(i - r_i, & j - r_j), & i > r_i, \ j > r_j \\ f(r_i - i, & j - r_j), & i \le r_i, \ j > r_j \\ f(i - r_i, & r_j - j), & i > r_i, \ j \le r_j \end{cases}$$

Уточнімо, що параметри *x* та *y* формули f(x,y) визначають зміщення елемента відносно ядра матриці по осі ординат та абсцис відповідно.

Матриця згортки, створена у такий спосіб, містить занадто велику кількість елементів, щоб навести її приклад на аркуші А4, тому наведемо приклад поверхні, створеної з її елементів – адже вона також відображає властивості згортки:



Тепер, коли матрицю згортки створено, можна переходити безпосередньо до загортки зображення.

V. Алгоритми накладання матриці згортки на зображення. Отримання зображення, розмитого за Гаусом.

Наведемо алгоритм згортки зображення для двовимірних матриць згортки. Як було зазначено вище, згортка зображення зводиться до того, щоб:

1) «Накласти» матрицю згортки на піксель, який необхідно розмити;
2) Обрахувати нове значення пікселя та нормалізувати його.

Для того, щоб об’єднати ці два пункти та не робити зайвих обчислень, ми будемо використовувати відносні значення для отримання елементу матриці згортки. Та перш ніж це зробити, обрахуємо коефіцієнт нормалізації (він – однаковий для кожного пікселя) як суму всіх елементів матриці згортки K(m x n):

$$\varphi = \sum_{i=1}^{m} \sum_{j=1}^{n} K(i,j)$$

Тепер згадаймо, що вхідне зображення A представлено у вигляді трьох матриць A_R, A_G, A_B, що містять відповідні параметри R, G та B кожного пікселя зображення A.

$$A(x;y) = \begin{cases} A_R(x;y) \\ A_G(x;y) \\ A_B(x;y) \end{cases}$$

Розглянемо згортку для каналу R (матриця A_R). Відтак, нове, ненормалізоване значення пікселя буде обраховуватись як сума добутків елементів матриці A_R, обмежених матрицею згортки, на відповідні значення елементів матриці згортки:

$$A_R(x,y) = \sum_{i=-(r_i-1)}^{r_i-1} \sum_{j=-(r_j-1)}^{r_j-1} A_R(x+i,\ y+j) \cdot K(r_i+i,\ r_j+j)$$

Для нормалізації цього значення, поділімо його на коефіцієнт нормалізації. Тепер, повна формула для обчислення нового значення елемента матриці A_R буде мати вигляд (справедливо і для матриць A_G, A_B):

$$A_R(x,y) = \frac{\sum_{i=-(r_i-1)}^{r_i-1} \sum_{j=-(r_j-1)}^{r_j-1} A_R(x+i,\ y+j) \cdot K(r_i+i,\ r_j+j)}{\varphi}$$

Приклад:

Нехай, маємо таку довільну матрицю згортки:

0.5	1.5	0.5
1.5	2	1.5
0.5	1.5	0.5

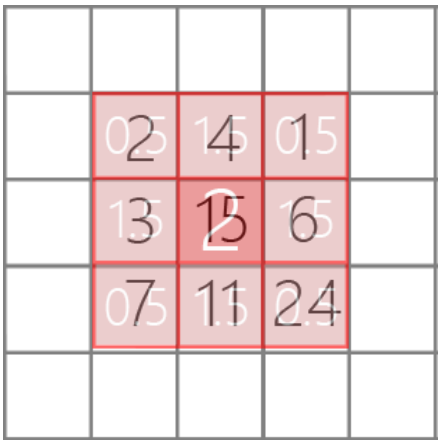
**Матрицю заповнено не за нормальним розподілом*

Її коефіцієнтом нормалізації буде: $\varphi = 0.5 + 1.5 + 0.5 + 1.5 + 2 + 1.5 + 0.5 + 1.5 + 0.5 = 10$.

Та таку матрицю A_R, де, наприклад, на поточному етапі необхідно обчислити нове значення елемента на перетині 3 рядка та 4 стовпця (порожні клітинки – довільні значення від 0 до 255):

		2	4	1	
		3	15	6	
		7	11	24	

Тоді це нове значення буде дорівнювати:



$$A_R(3,4) = \frac{(2 \cdot 0.5 + 4 \cdot 1.5 + 1 \cdot 0.5) + (3 \cdot 1.5 + 15 \cdot 2 + 6 \cdot 1.5) + (7 \cdot 0.5 + 11 \cdot 1.5 + 24 \cdot 0.5)}{10} \approx 8$$

І, як результат, матриця A_R прийняла такий вигляд:

		2	4	1	
		3	8	6	
		7	11	24	

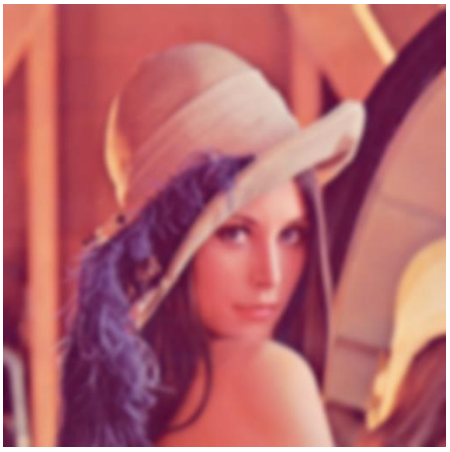
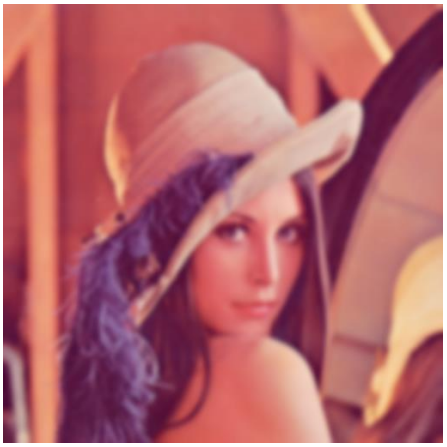
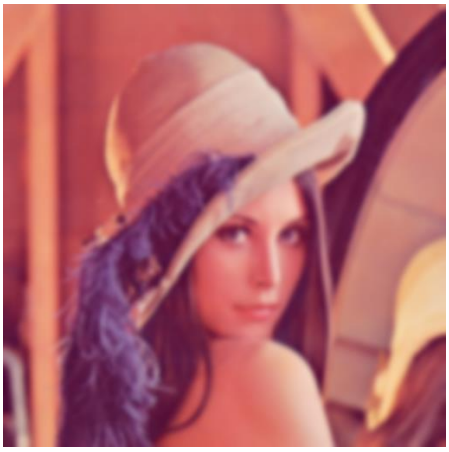
Отже, для повної обробки всього зображення необхідно накласти матрицю згортки на кожен елемент матриць A_R, A_G, A_B. Але тоді, для розрахунку одного нового значення пікселя необхідно задіяти в обчисленнях $3 \cdot ((3 \cdot \sigma) \cdot 2 + 1)^2$ елементів, а це – **дуже** багато. Щоб уникнути такої кількості обчислень, використаємо одновимірну матрицю згортки.

Для використання одновимірної матриці згортки необхідно обробити матриці A_R, A_G, A_B двічі. Спочатку зліва направо, потім згори донизу. За таких умов, необхідно дещо змінити алгоритм обробки:

1. Коефіцієнт нормалізації збереже свою сутність, проте буде обраховуватись за дещо модифікованою формулою:
$$\varphi = \sum_{j=1}^{n} K(j)$$
2. Нове значення елементу буде обчислюватись почергово двома формулами:
$$A_R(x,y) = \begin{cases} \dfrac{\sum_{j=-(r_j-1)}^{r_j-1} A_R(x,\ y+j) \cdot K(r_j+j)}{\varphi} \\ \dfrac{\sum_{i=-(r_j-1)}^{r_j-1} A_R(x+i,\ y) \cdot K(r_j+i)}{\varphi} \end{cases}$$

Тому, тепер для обчислення нового значення пікселя потрібно зробити лише $3 \cdot ((3 \cdot \sigma \cdot 2 + 1) \cdot 2)$ обчислень.

Відтак, алгоритми створення матриць згортки та алгоритми згортки зображення розібрано, тож порівняємо результати розмивання нашими алгоритмами з результатом від розмивання в Adobe Photoshop:

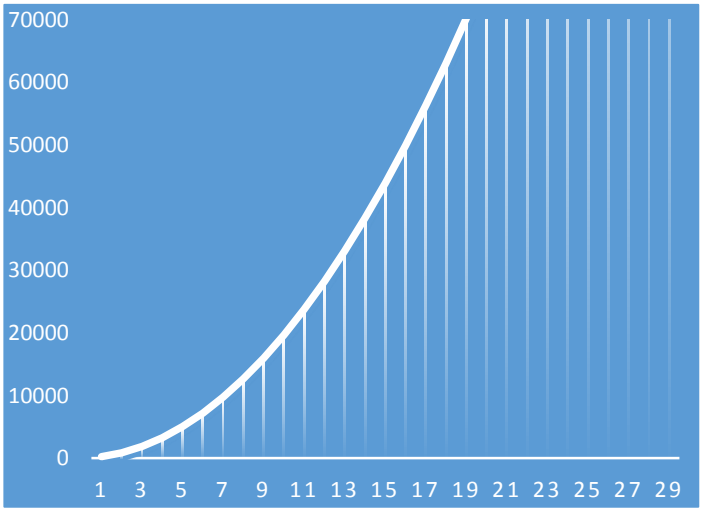
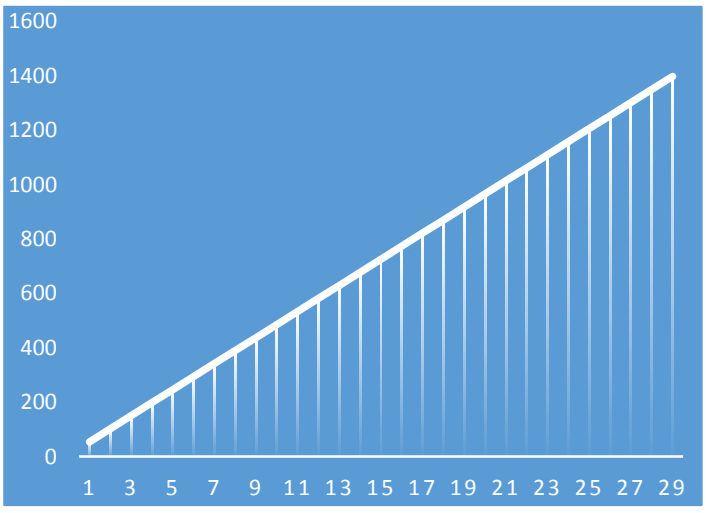
Оригінал (512x512)		
		
Adobe Photoshop $\sigma = 3$	Обговорений алгоритм ($\sigma = 3$, 1D)	Обговорений алгоритм ($\sigma = 3$, 2D)
		
Час обробки: ~00:00.0041245 мс	Елементів обраховано: 35 770 368 Час обробки: 00:01.5401536 мс	Елементів обраховано: 406 748 208 Час обробки: 00:20.4673921 мс

**Така різниця у часі обробки між обговорюваними алгоритмами та алгоритмом Adobe Photoshop зумовлена використанням CPU для обчислень, а не GPU, як у Adobe Photoshop.*

***Різниця у часі обробки обговорюваними 2D та 1D алгоритмами відносно справедлива.*

VI. Висновок.

Результат обробки алгоритмами з одновимірною та двовимірною матрицями наближається до еталонного. Проте, кількість обчислень суттєво відрізняється:

Графік залежності кількості елементів, необхідних для обчислення одного нового пікселя, від радіусу розмивання	
Двовимірна матриця згортки	Одновимірна матриця згортки
	

Таким чином, якщо важливіша швидкість обробки, аніж еталонна якість, слід використовувати одновимірну матрицю для згортки зображення. Та навпаки, якщо кількістю використовуваних ресурсів можна знехтувати, а приділяти більше уваги якості обробки – використовують двовимірні матриці для згортки зображення.

Автор: Калініченко М. В.
Редагувала: Вяла Ю. Є.
м. Київ, 2016

Реалізація алгоритмів

Програмна реалізація описаних алгоритмів на мові C# доступна за посиланням: https://github.com/Nk185/GaussianBlur

Використана література

1. «Теория вероятностей и математическая статистика» В.Е. Гмурман, г. Москва, 1977.
2. «Введение в теорию вероятностей», Колмогоров А. Н., Журбенко И. Г., Прохоров А. В., 1982.
3. «Розмивання Гауса» Вікіпедія.