

Разработка и исследование массивно-многопоточного алгоритма нелинейной фильтрации изображений в CUDA-среде

Магистрант

Елкин Д. А.

Руководитель работы

Фурсов В.А.

Цель работы

Целью работы является разработка и исследование модели нелинейной фильтрации изображений для устранения динамических искажений. Создание программного обеспечения для исследования качества восстановления изображений с помощью разработанной модели.

Задачи:

1. Анализ задачи построения нелинейного фильтра для устранения динамических искажений.
2. Разработка модели нелинейного фильтра для искажений с радиальной симметрией.
3. Разработка программного комплекса для идентификации параметров фильтра и восстановления изображений.
4. Проведение экспериментов по восстановлению изображений с помощью разработанной модели нелинейного фильтра.

Постановка задачи

Полином Колмогорова-Габора:

$$y(k_1, k_2) = c_0 + \sum_{n_1, n_2=1}^{N_1, N_2} c_{n_1, n_2} x(n_1, n_2) + \sum_{n_1, n_2=1}^{N_1, N_2} \sum_{m_1, m_2=1}^{N_1, N_2} c_{n_1, n_2; m_1, m_2} x(n_1, n_2) x(m_1, m_2) + \dots$$

Матричное соотношение: $y = Xc + \xi$

Задача заключается в том, чтобы по одной реализации (фрагменту изображения) построить оценку \hat{c} вектора параметров c по доступным для непосредственного наблюдения $N \times M$ -матрице X и N -вектору Y ($N > M$), при неизвестном N -векторе ошибок ξ .

Модификация модели для случая радиально симметричных искажений

$$x(r) = \frac{1}{m} \sum_{i=1}^m x_i(n_1, n_2, r)$$

$$1. \quad y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_i x(r_i) + \sum_{i=1}^6 \sum_{j=1}^6 c_{i,j} x(r_i) x(r_j).$$

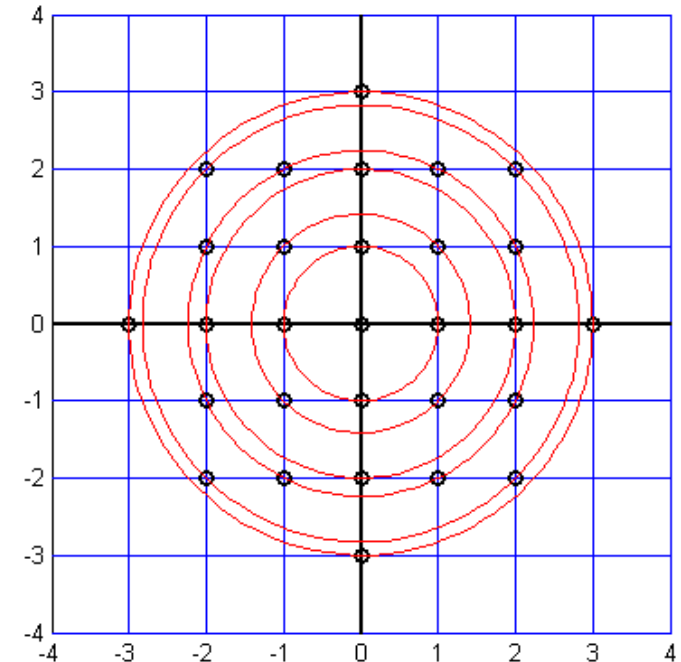
$$2. \quad y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i).$$

$$3. \quad y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i) + \sum_{i=1}^6 c_{2,i} x^3(r_i).$$

$$4. \quad y(k_1, k_2) = c_0 + \sum_{i=1}^3 c_{1,i} x(r_i) + \sum_{i=1}^3 c_{2,i} x^2(r_i) + \sum_{i=1}^3 c_{2,i} x^3(r_i).$$

$$5. \quad x'(r_3) = \frac{1}{2}(x(r_3) + x(r_4)); \quad x'(r_4) = \frac{1}{2}(x(r_5) + x(r_6));$$

$$y(k_1, k_2) = c_0 + \sum_{i=1}^4 c_i x(r_i) + \sum_{i=1}^4 \sum_{j=1}^4 c_{i,j} x(r_i) x(r_j).$$



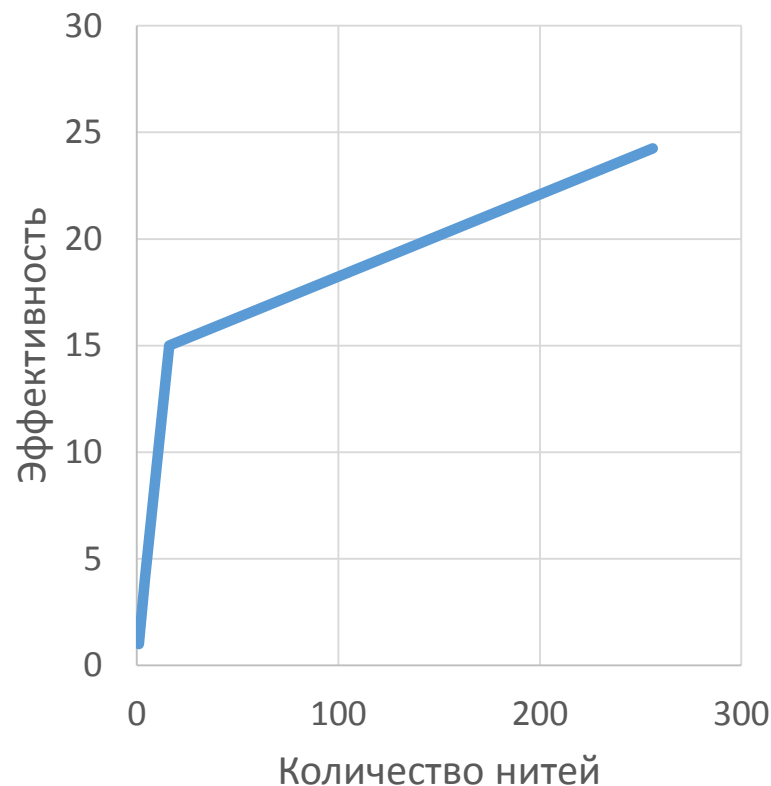
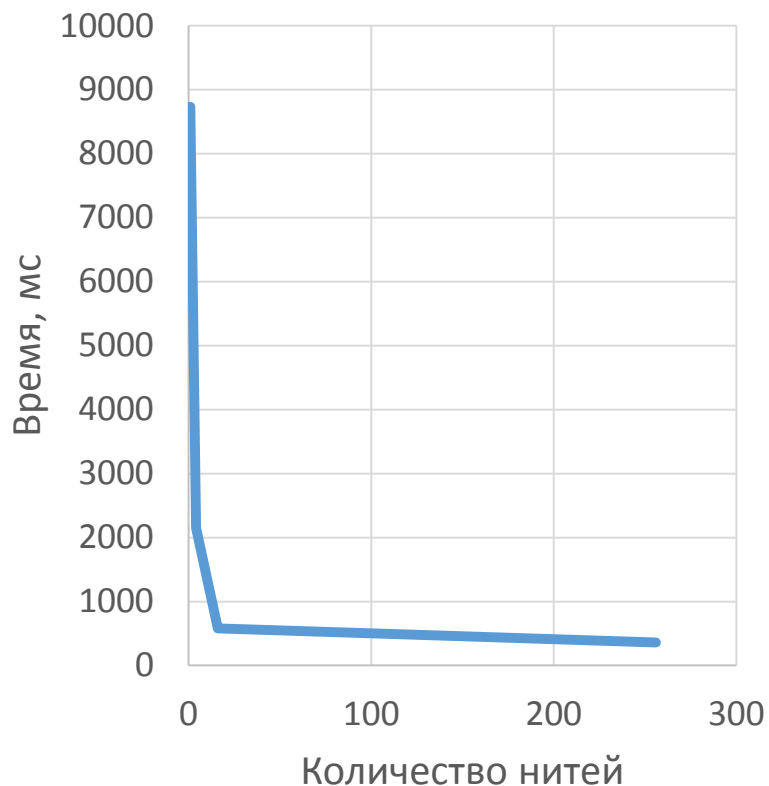
Параллельная реализация восстановления изображений

$$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i) + \sum_{i=1}^6 c_{2,i} x^3(r_i)$$

Шаги:

1. Инициализация y значением c_0 .
2. Параллельно для каждого отсчета рассчитать $\sum_{i=1}^6 c_{1,i} x(r_i)$.
3. Параллельно для каждого отсчета рассчитать $\sum_{i=1}^6 c_{2,i} x^2(r_i)$.
4. Параллельно для каждого отсчета рассчитать $\sum_{i=1}^6 c_{2,i} x^3(r_i)$.

Параллельная реализация восстановления изображений



Сравнение результатов обработки при $\sigma=3$

Фильтр	СКО «Лена»	СКО «Город»
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i) + \sum_{i=1}^6 c_{2,i} x^3(r_i)$	8,7016	15,1467
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i)$	8,7043	15,1555
$x'(r_3) = \frac{1}{2}(x(r_3) + x(r_4)); \quad x'(r_4) = \frac{1}{2}(x(r_5) + x(r_6))$	8,9613	15,6698
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_i x(r_i) + \sum_{i=1}^6 \sum_{j=1}^6 c_{i,j} x(r_i) x(r_j)$	8,9870	15,2948
$y(k_1, k_2) = c_0 + \sum_{i=1}^3 c_{1,i} x(r_i) + \sum_{i=1}^3 c_{2,i} x^2(r_i) + \sum_{i=1}^3 c_{2,i} x^3(r_i)$	8,8932	16,8469
OpenCV (Винеровский фильтр)	9,8238	19,3741

Сравнение результатов обработки при $\sigma=5$

Фильтр	СКО «Лена»	СКО «Город»
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i) + \sum_{i=1}^6 c_{2,i} x^3(r_i)$	11,6262	23,5168
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_{1,i} x(r_i) + \sum_{i=1}^6 c_{2,i} x^2(r_i)$	11,7534	23,5207
$x'(r_3) = \frac{1}{2}(x(r_3) + x(r_4)); \quad x'(r_4) = \frac{1}{2}(x(r_5) + x(r_6))$	11,8041	23,6472
$y(k_1, k_2) = c_0 + \sum_{i=1}^6 c_i x(r_i) + \sum_{i=1}^6 \sum_{j=1}^6 c_{i,j} x(r_i) x(r_j)$	11,9505	23,7339
$y(k_1, k_2) = c_0 + \sum_{i=1}^3 c_{1,i} x(r_i) + \sum_{i=1}^3 c_{2,i} x^2(r_i) + \sum_{i=1}^3 c_{2,i} x^3(r_i)$	12,1636	23,6343
OpenCV (Винеровский фильтр)	13,2166	23,8003

Пример 1 – сравнение результатов при $\sigma=3$



Исходное изображение



Размытое изображение

Пример 1 – сравнение результатов при $\sigma=3$



Восстановленное изображение
СКО=15,1467

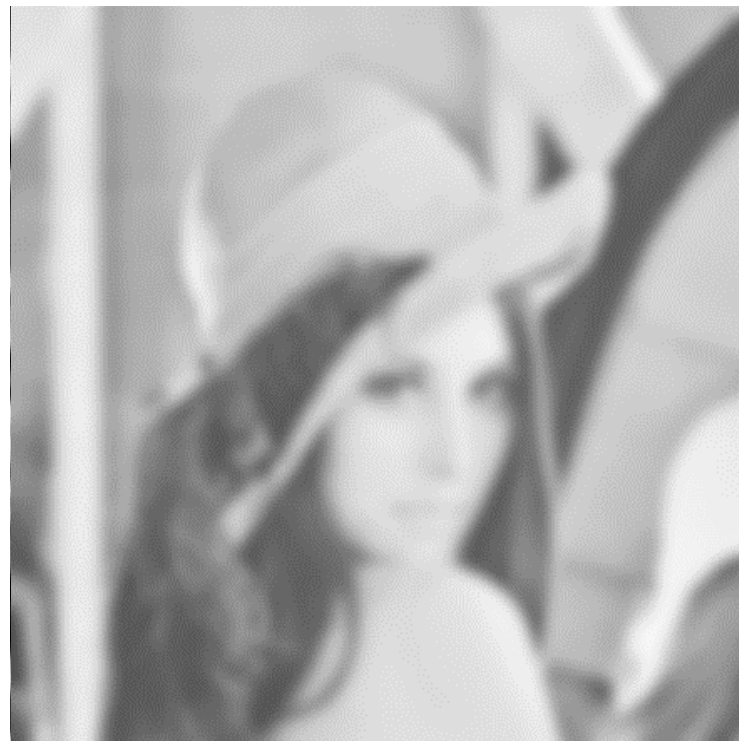


OpenCV
СКО=19,3741

Пример 2 – сравнение результатов при $\sigma=5$

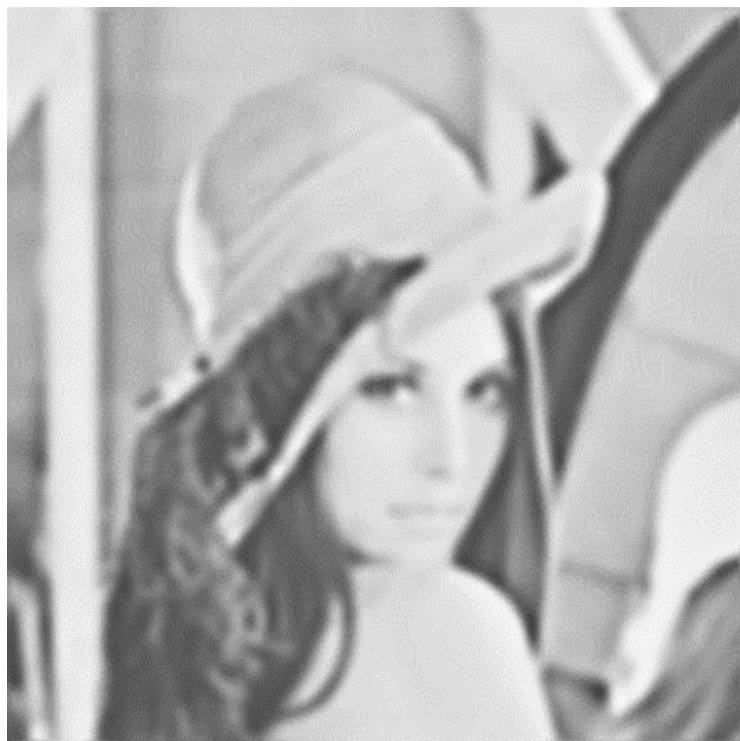


Исходное изображение

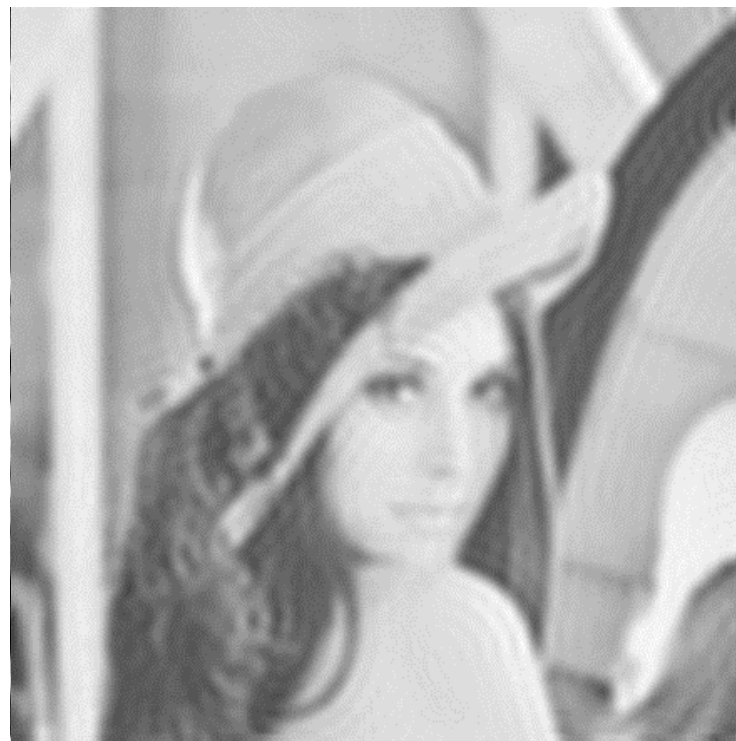


Размытое изображение

Пример 2 – сравнение результатов при $\sigma=5$



Восстановленное изображение
СКО=11,6262



OpenCV
СКО=13,2166

Заключение

По представленным результатам можно судить о преимуществе использования нелинейного фильтра: границы объектов, при восстановлении с помощью нелинейного фильтра более четкие.

Также следует отметить удобство и простоту использования алгоритма идентификации параметров модели для задачи восстановления изображений.