Отчет по лабораторной работе №4

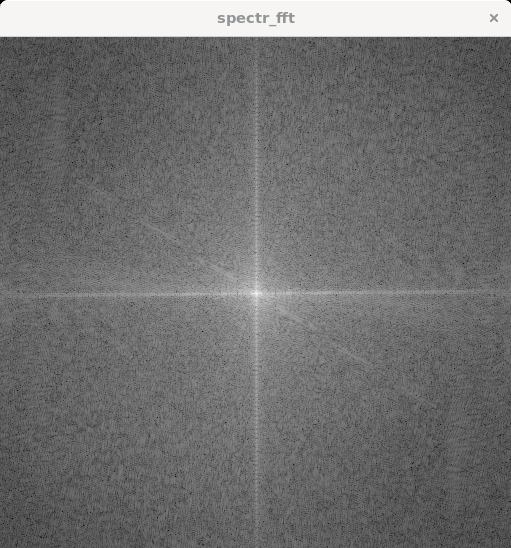
1. В первом пункте задания нужно было реализовать “лобовой подход” дискретного преобразования Фурье. Преобразование выполняется с помощью функции void ft::simpleDft. Реализация подразумевает выполнения одномерного преобразования для столбцов матрицы и дальнейшее одномерное преобразования для строк. Перед созданием данной реализации была предпринята попытка написать монолитное двумерное преобразование, но оно работало в десятки раз медленнее. Для вычисления обратного преобразования используется та же функция, а выбор режима закодирован в параметре *flag* *(0 – forward, 1 – inverse).*

 Результат работы первого блока представлен ниже. Левая картинка - оригинальное одноканальное изображение. Следующие два изображения – спектры, полученные при помощи дискретного преобразования Фурье (OpenCV и собственная реализация). Последнее изображение – результат работы обратного преобразования Фурье в собственном исполнении.

2. Быстрое преобразование Фурье (алгоритм Radix-2) выполняется с помощью функции void ft::fastDft.

Для корректной работы этого алгоритма необходимо, чтобы длины сторон соответствовали степеням 2. Таким образом, исходник пришлось «раздувать» до размера 512 х 512 точек.

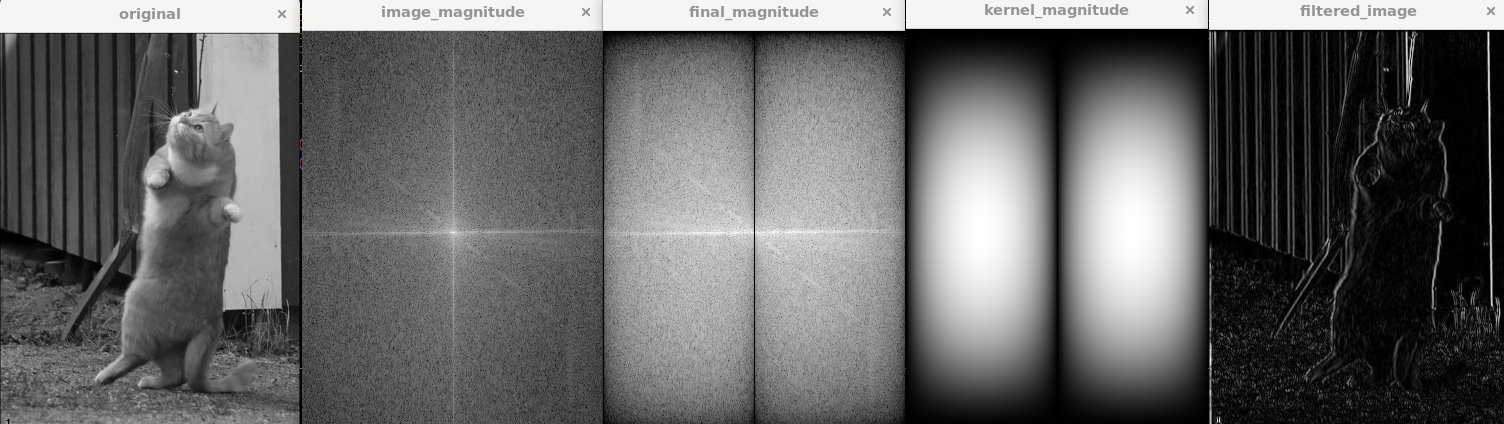
Результат работы алгоритма (спектр) представлен ниже.

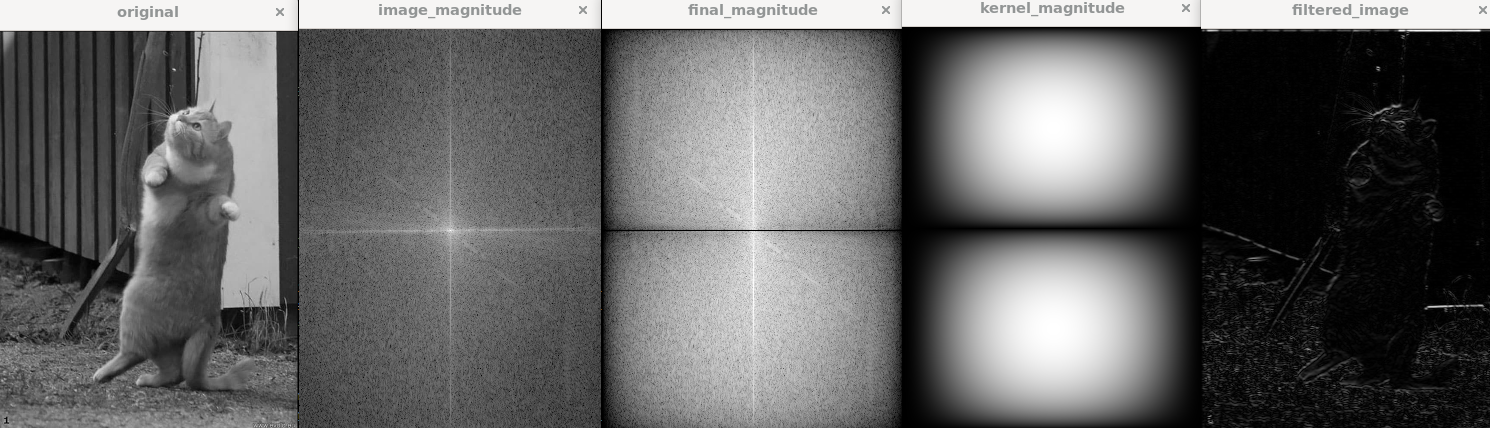


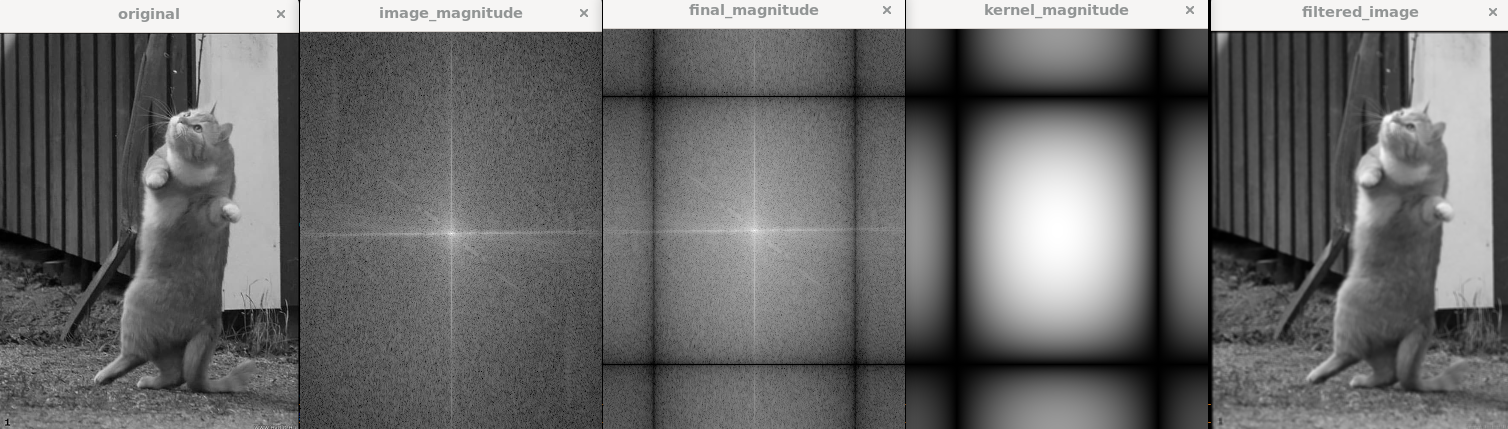
Изображение выглядит как текст, в помещении, черный, темный

Автоматически созданное описание3. На картинке ниже представлено время выполнения трех реализаций преобразования Фурье. Встроенная функция работает быстрее всего, собственная реализация быстрого преобразования Фурье оказалась гораздо быстрее реализации простого метода. Заметим, что учитывается время исключительно на вызов функций преобразования, OpenCV собран в релизе, как и сама программа.

4. В данном разделе представлена фильтрация исходного изображения различными фильтрами. Операция свертки производилась в частотной области при помощи перемножения спектров (void cv::mulSpectrums), при этом холсты ядер фильтров предварительно необходимо расширять. В программе вызывается высокоуровневая функция void ft::convolution, производящая свертку и выводящая изображения.

Результат свёртки с горизонтальным ядром Собеля:

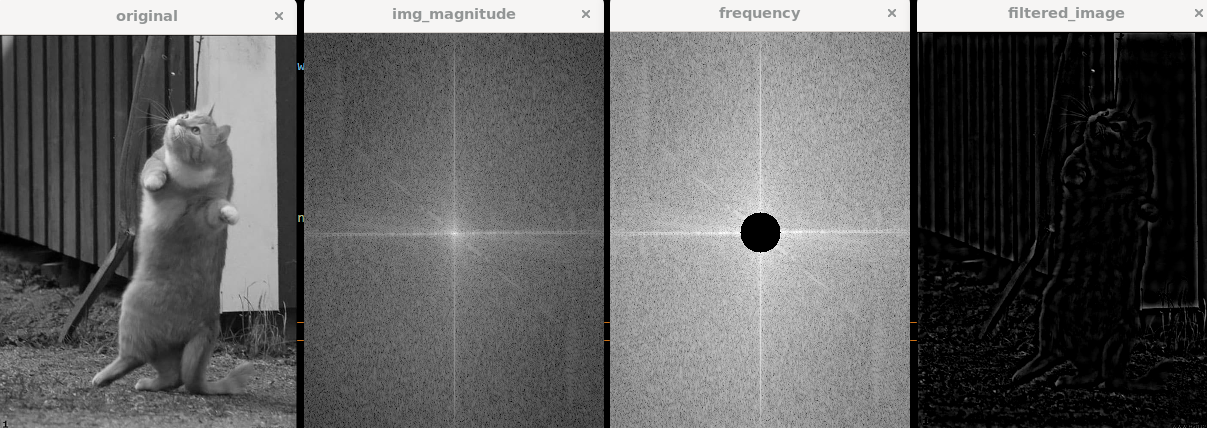
Результат свёртки с вертикальным ядром Собеля:

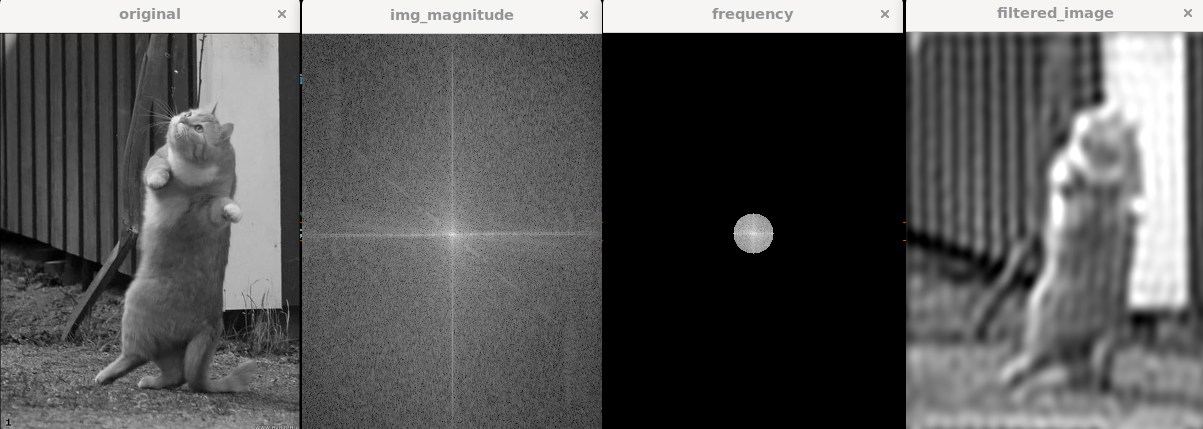
Результат свёртки с бокс-фильтром:

Изображение выглядит как диаграмма

Автоматически созданное описаниеРезультат свёртки с ядром Лапласа:

5. В данном разделе представлены результаты работы фильтров низких и высоких частот. Фильтрация реализована в функции ft::frequencyFilter, а диапазон частот кодируется параметром *flag* *(0 – высокие частоты, 1 – низкие частоты).*

Результат работы фильтра высоких частот на изображении снизу. Стоит сразу пояснить, что обрезанный спектр существенно ярче из-за того, что высокие частоты не учувствуют в нормализации перед выводом.

**Результат работы фильтра низких частот на изображении снизу.

6. В данном разделе представлены результаты поиска по шаблону при помощи корреляции спектров. Операция свертки производилась в частотной области при помощи перемножения спектров (void cv::mulSpectrums). Функция, производящая поиск по шаблону – void ft::detectNumber. Для проведения корреляции было выбрано следующее изображение:



Символы для поиска:



Изображение выглядит как Веб-сайт

Автоматически созданное описаниеРезультат детекции первого символа:

Изображение выглядит как Веб-сайт

Автоматически созданное описаниеРезультат детекции второго символа:

Изображение выглядит как Веб-сайт

Автоматически созданное описаниеРезультат детекции третьего символа: