Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных технология и систем»

Лабораторная работа №2

**«Программа обработки изображений»**

Отчёт по дисциплине:

«Распознавание образов и обработка изображений»

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Бочкарев Б. А.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Принял преподаватель:

\_\_\_\_\_\_\_\_\_\_ / Назаров А. Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Великий Новгород

2022

**Цель лабораторной работы**

* Изучение основ обработки изображений и базовых алгоритмов фильтрации изображений.
* Доработка рабочего программного окружения для выполнения лабораторной работы с использованием OpenCV. Установка библиотеки может быть выполнена одним из двух методов: сборка из исходников с помощью cmake, установка prebuilt сборки.
* Доработать программу обработки изображений, создать дополнительный плагин для выполнения медианного фильтра (median filter), ядро - 3x3. (Без использования стандартной функции OpenCV).

**Программное окружение**

Для выполнения данной лабораторной работы было собрано следующее окружение:

* Язык программирования: C++
* Операционная система: MS Windows 10
* Компилятор: MSVC2015\_64bit
* Среда программирования: QtCreator с версией Qt 5.15.2

Установка библиотеки OpenCV выполнена с помощью prebuilt сборки.

* Версия библиотеки OpenCV: 4.5.5

**Медианный фильтр**

**Медианный фильтр** — один из видов цифровых фильтров, широко используемый в цифровой обработке сигналов и изображений для уменьшения уровня шума. Медианный фильтр является нелинейным КИХ-фильтром.

С помощью цикла пройдем по всем пикселям исходного изображения, игнорируя пиксели рамки. Во время выполнения цикла, мы останавливаемся на каждом пикселе и работаем с окном 3x3. Значение каждого канала в этом окне заносится в отдельный вектор, впоследствии сортируются. На выходе из этого окна мы получим пиксель, у которого для каждого канала была вычислена медиана в данном окне.

**Примеры выполнения работы медианного фильтра**



Рисунок 1

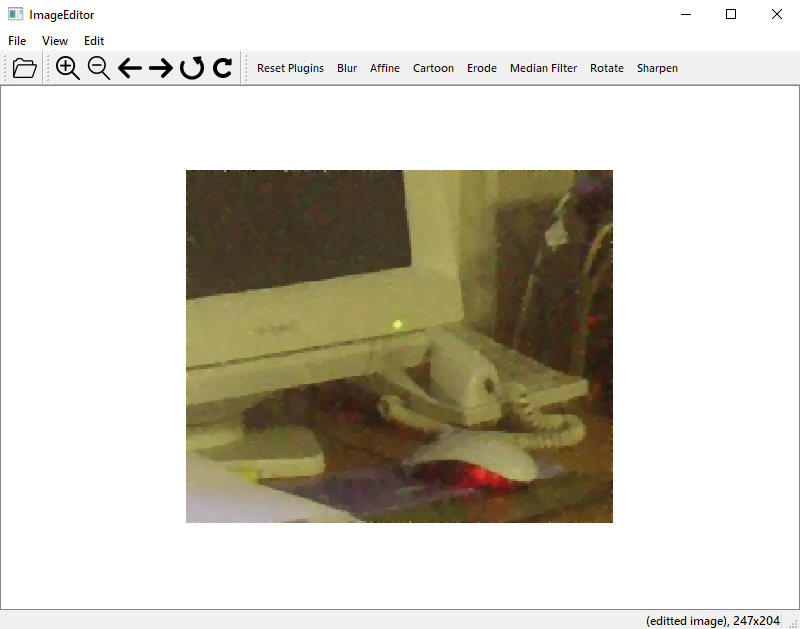


Рисунок 2

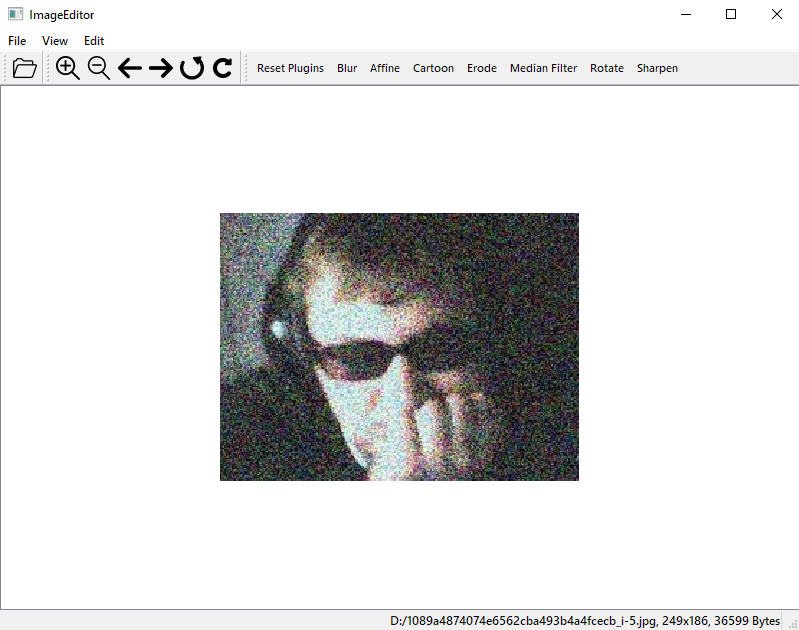


Рисунок 3

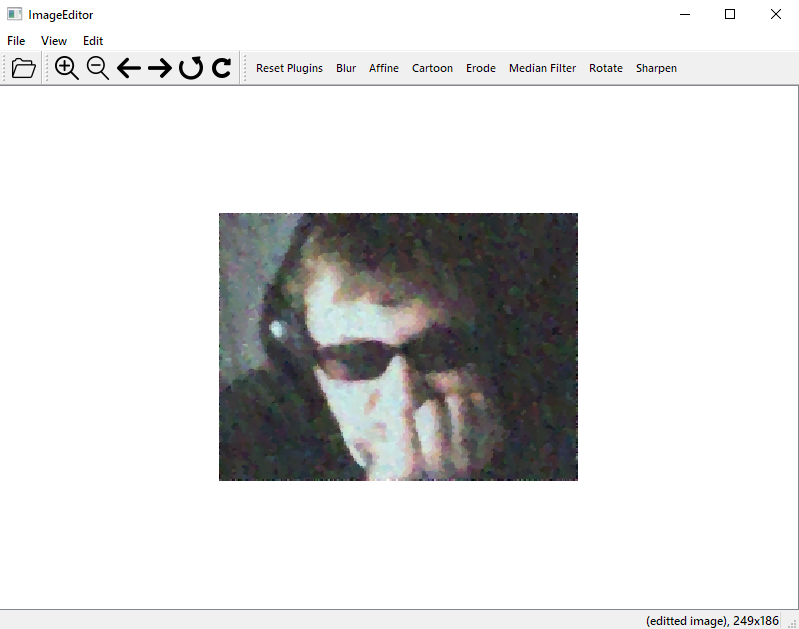


Рисунок 4

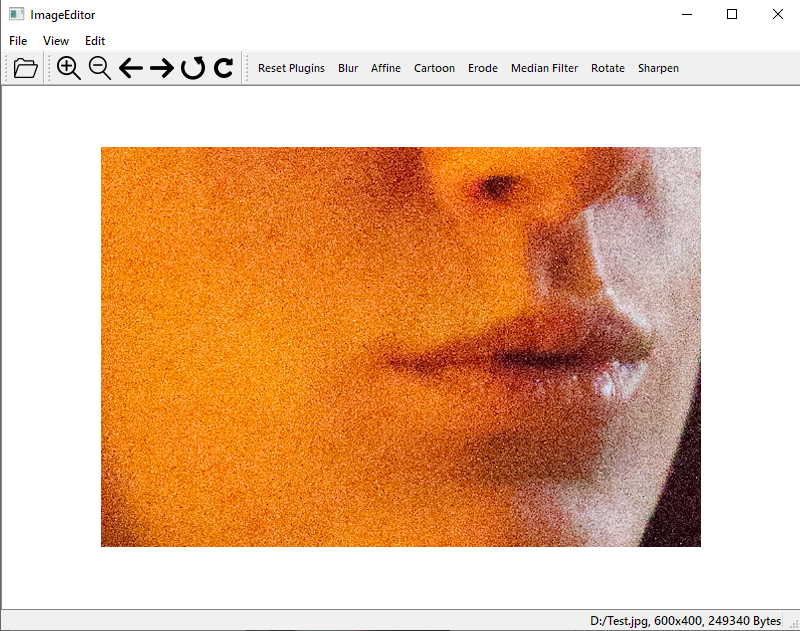


Рисунок 5

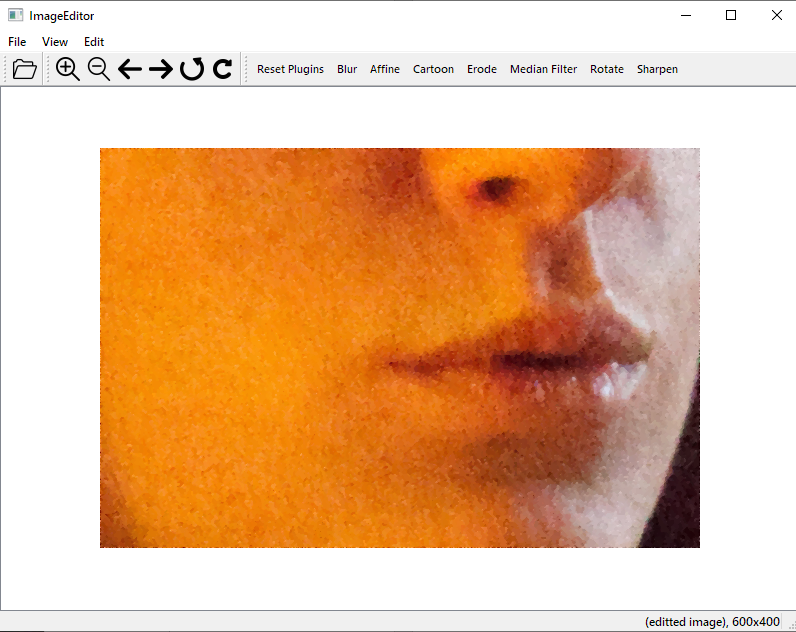


Рисунок 6

**Ответы на вопросы**

Вопрос 1: Как узнать, поддерживает ли функция OpenCV работу на месте или нет?

Ответ: Что бы узнать, поддерживает ли функция OpenCV работу на месте, можно обратится к документации OpenCV, найти нужную функцию, в описании к функции должно быть написано, поддерживается ли работа на месте. Если такой информации в описании к функции нету, можно считать, что функция не поддерживает работу на месте.

Вопрос 2: Как мы можем добавить горячую клавишу для каждого действия, которое мы добавили в качестве плагина?

Ответ: Проверяем с помощью условного оператора if, действие какого плагина было инициализировано. После мы можем назначить горячие клавиши на это действие с помощью метода setShortcut():

if (plugin\_ptr->name() == "Median Filter"){

action->setShortcut(QKeySequence("Ctrl+6"));

}

Вопрос 3: Как мы можем добавить новое действие, чтобы отменить все изменения текущего изображения в нашем приложении?

Ответ: Для функции отмены изменений у изображения, создадим новый Action, обращаться который будет к слоту cancelImg(), где мы вызываем функцию showImage(currentImagePath), где в переменной currentImagePath, храниться текущий путь к изображению.

Вопрос 4: Как мы можем изменить размер изображений с помощью OpenCV?

Ответ: Масштабирование — это просто изменение размера изображения. OpenCV поставляется с функцией cv.resize() для этой цели. Размер изображения можно указать вручную или указать коэффициент масштабирования. Используются различные методы интерполяции. Предпочтительными методами интерполяции являются cv.INTER\_AREA для сжатия и cv.INTER\_CUBIC и cv.INTER\_LINEAR для масштабирования.

cv.resize (src, dst, dsize, fx = 0, fy = 0, interpolation = cv.INTER\_LINEAR)

Где:

* src - входное изображение;
* dst - выходное изображение;
* dsize - размер выходного изображения;
* fx - масштабный коэффициент по горизонтальной оси
* fy - масштабный коэффициент по вертикальной оси
* interpolation - интерполяционный метод

**Исходный код всех исходных файлов и файлов проекта**

Исходный код всех исходных файлов и файлов проекта доступен по ссылке: https://github.com/borya178/Pattern\_recognition\_and\_image\_processing

**Дистрибутив программы**

Дистрибутив был создан с помощью **windeployqt.exe.**

Дистрибутив был протестирован на виртуальной машине с ОС MS Windows 7, созданной с помощью программы VirtualBox.

Дистрибутив проекта в архиве доступен по ссылке: https://github.com/borya178/Pattern\_recognition\_and\_image\_processing

Запустить программу можно, нажав на файл **ImageEditor.exe.**

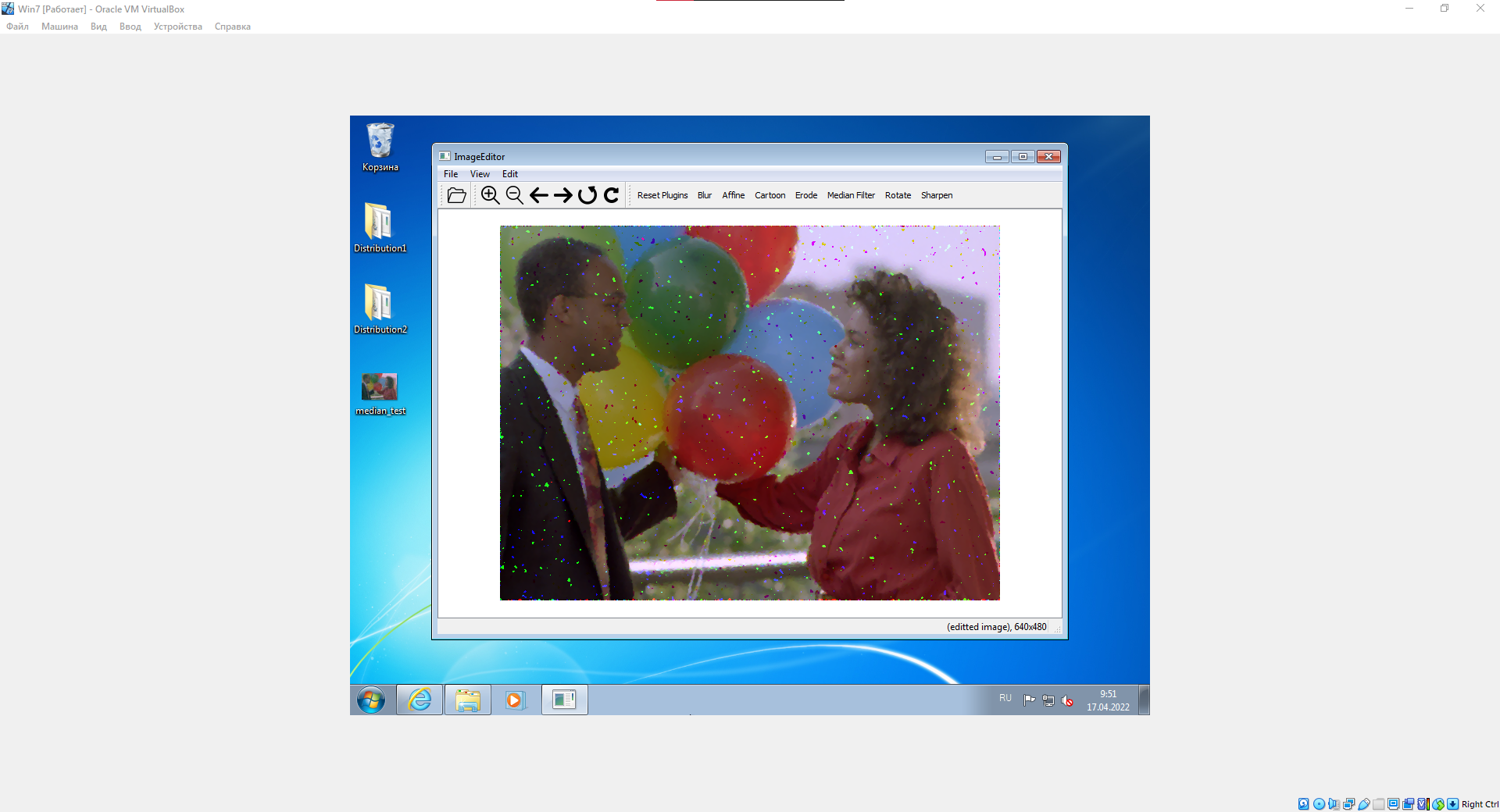


Рисунок 7 Демонстрация работы на виртуальной машине

**Скриншоты выполнения программы с демонстрацией функционального состава**

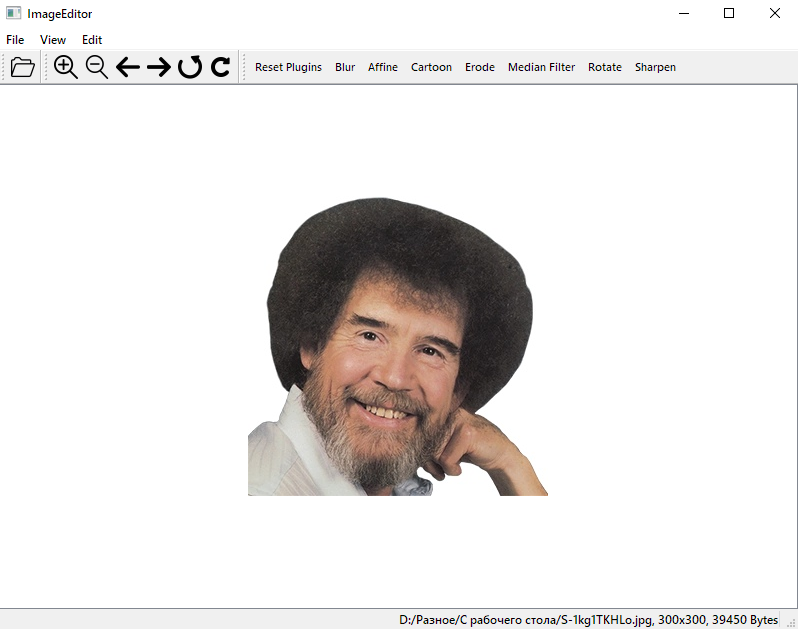
****

Рисунок 8 Оригинал изображения

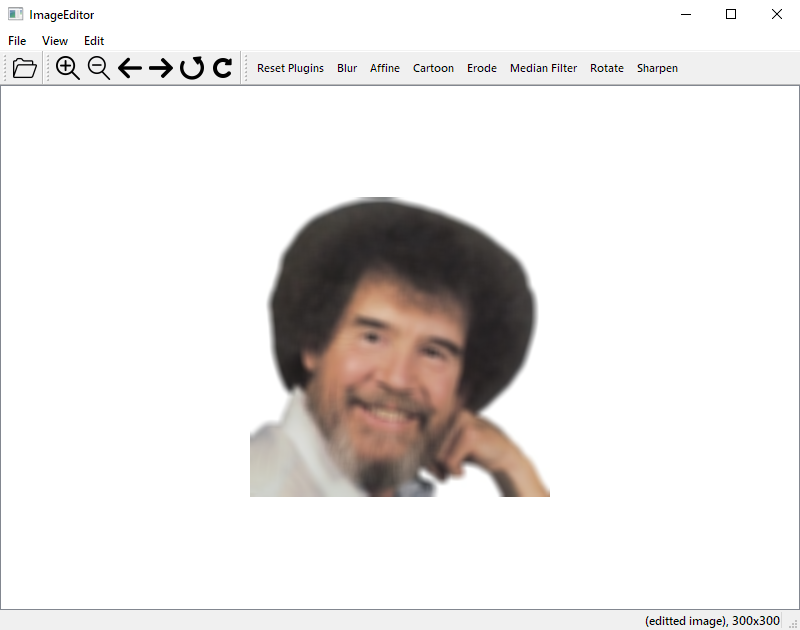


Рисунок 9 Blur

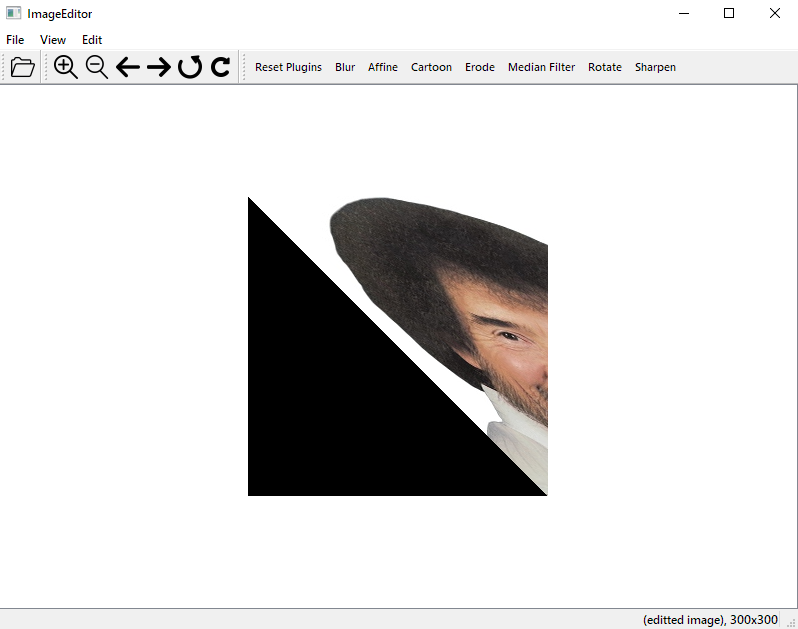


Рисунок 10 Affine

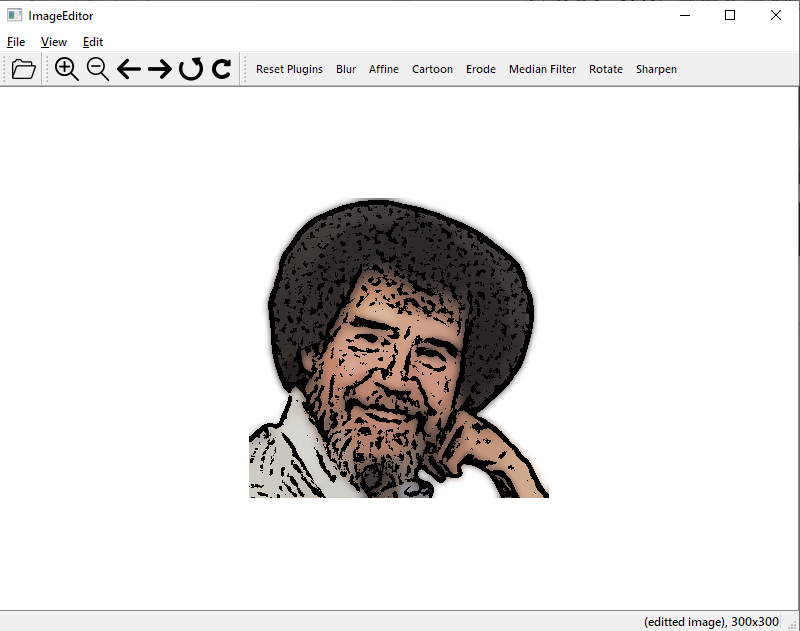


Рисунок 11 Cartoon

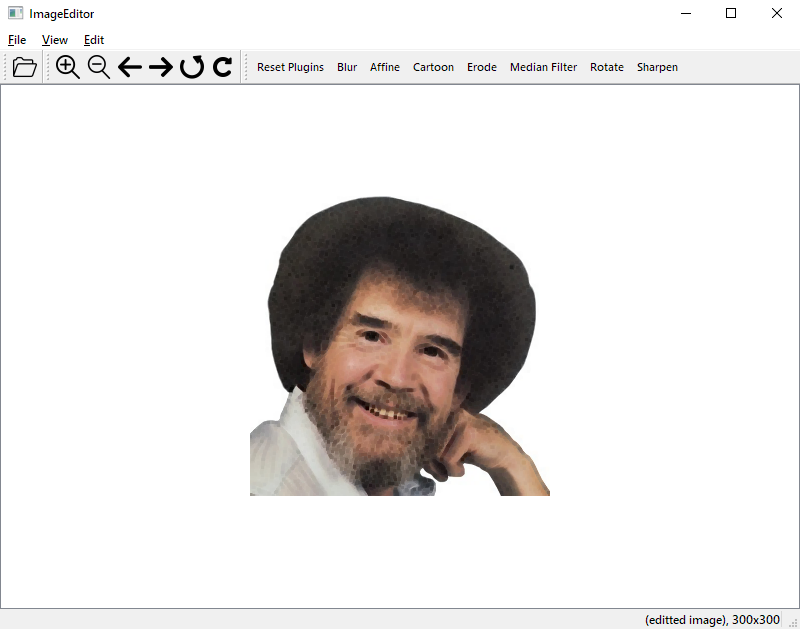


Рисунок 12 Erode

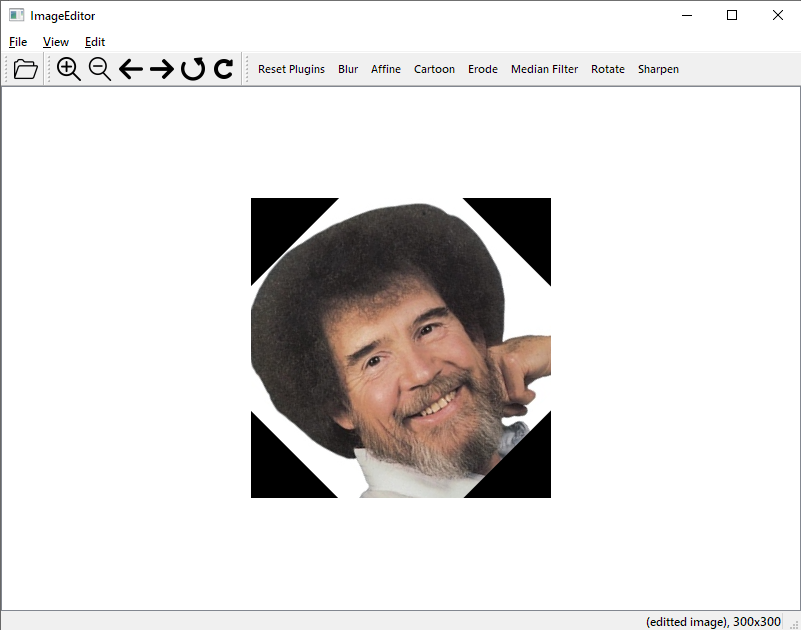


Рисунок 13 Rotate

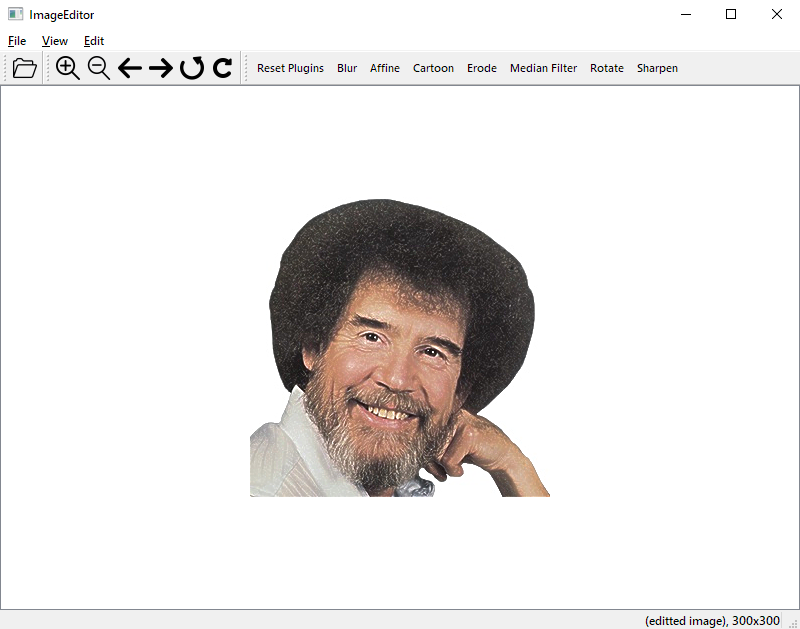


Рисунок 14 Sharpen

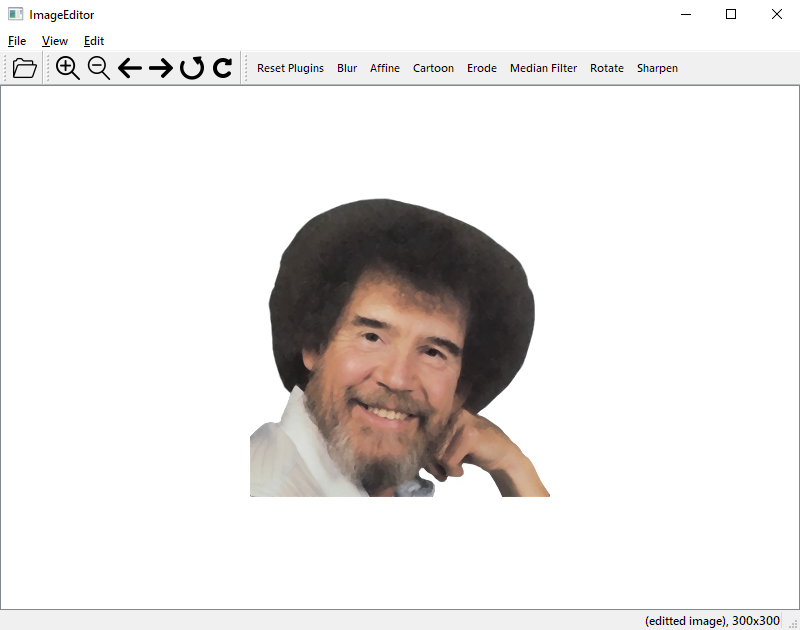


Рисунок 15 Median Filter

**Вывод**

В ходе выполнения лабораторной работы я познакомился с базовыми алгоритмами обработки изображений, доработал программное окружение установив OpenCV и доработал программу обработки изображений, создав дополнительный плагин для выполнения медианного фильтра