МИНЕСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Программная инженерия»

**Отчет по лабораторной работе №2**

на тему:

**«Выделение на изображении листа больной и здоровой части при помощи watershed алгоритма»**

**Выполнил**:

Студент группы 381908-2

Григорьев Матвей Сергеевич

**Проверила**:

Преподаватель, программист 1 категории

Гетманская Александра Александровна

Нижний Новгород

2021

**Содержание**

[Постановка задачи 3](#_Toc86414475)

[Руководство программиста 3](#_Toc86414476)

[Подобранные параметры 5](#_Toc86414477)

[Выводы 6](#_Toc86414478)

# Постановка задачи

На наборе изображений листов выделить больные участки. Использовать алгоритмы фильтрации и сегментации. Определить оптимальные параметры алгоритма фильтрации.

# Руководство программиста

Программа разрабатывалась в среде разработки Visual Studio Code с

использованием интерпретатора Python версии 3.8.10.

В программе использовались следующие библиотеки:

1. OpenCV – библиотека алгоритмов обработки изображений, компьютерного зрения и численных алгоритмов общего назначения с открытым исходным кодом
2. Numpy – библиотека с открытым исходным кодом, предоставляющая возможности по обработке многомерных массивов

Для загрузки и подключения библиотек использовался PIP версии 21.3.1.

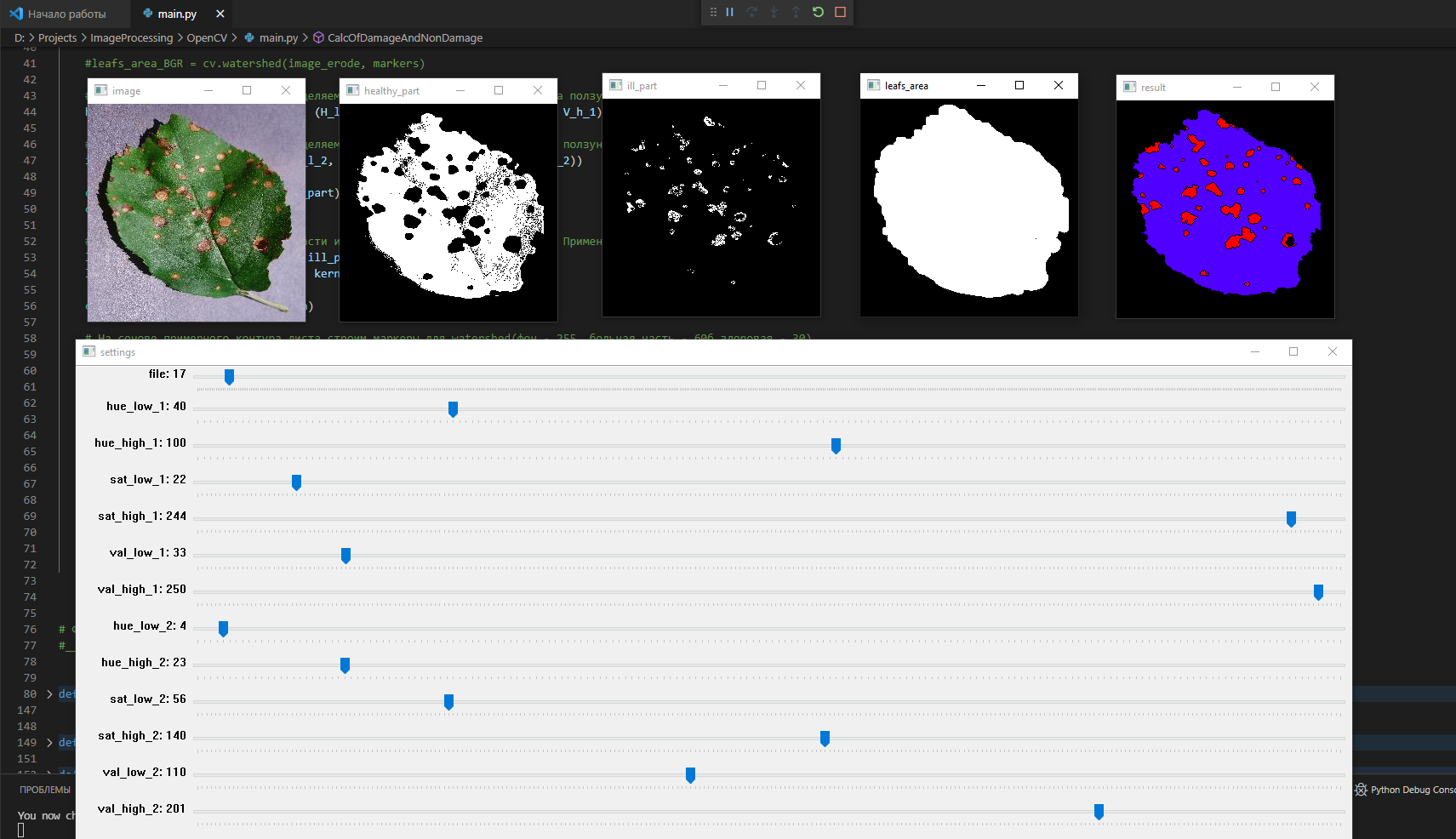
**Описание структуры программы**

Программу можно условно разделить на 3 части:

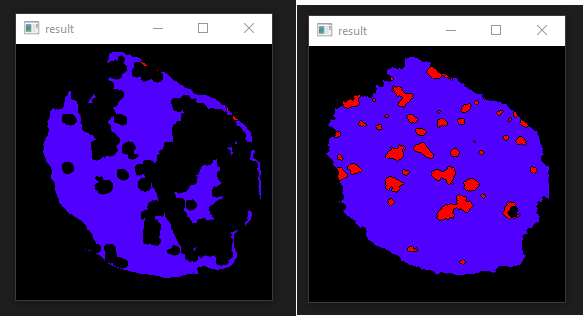
1. Функция обнаружения поражений на листах
2. Блок инициализации окон, ползунков и функций для их работы.
3. Бесконечный цикл, который обеспечивает работу интерфейса и позволяет подбирать параметры фильтров на выбранном массиве изображений

**Описание работы алгоритма:**

Основной алгоритм обработки содержится в функции **CalcOfDamageAndNonDamage.** Эта функция считывает изображение и переводит его в цветовое пространство HSV с помощью функции **cvtColor**. От применения алгоритмов сглаживания было решено отказаться, так как в ходе проведения экспериментов выяснилось, что применение сглаживания негативно сказывается на качестве работы алгоритма. Дальше в работу вступают два пороговых фильтра. С их помощью мы выделяем «здоровую» и «больную» части листа. В качестве порогового фильтра используется функция **inRange**. Параметры этих пороговых фильтров задаются ползунками. Затем с помощью функции **add** мы «складываем» изображения, полученные на предыдущем шаге. На основе этого изображения мы стоим маркеры для алгоритма **watershed**. После работы функции **watershed** мы строим маску итогового изображения и раскрашиваем его в нужные цвета. На выходе имеем изображение с отмеченными здоровой и больной частями.

****

1. Общий вид программы



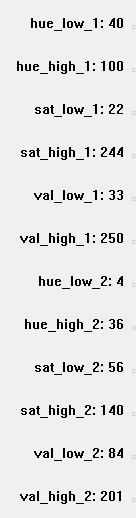
. Сравнение результатов с использованием сглаживания и без

# Подобранные параметры

Для коричневых точек оптимальными оказались следующие параметры

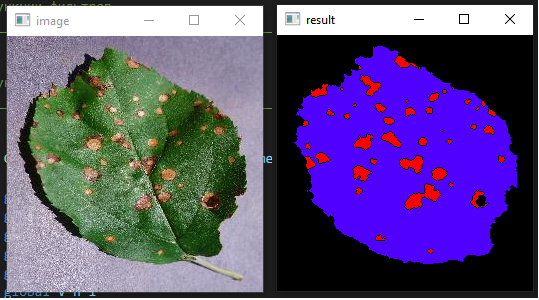


Для некоторых изображений с черными точками удачными оказались следующие параметры, однако подобрать их оптимальными для всего массива этих изображений не удалось.

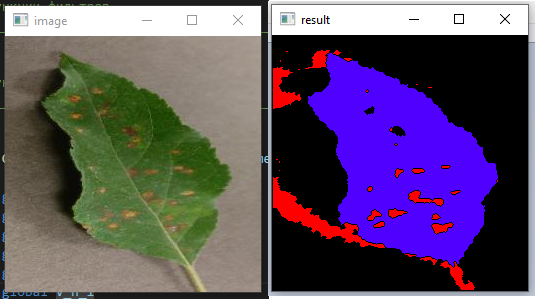


# Выводы

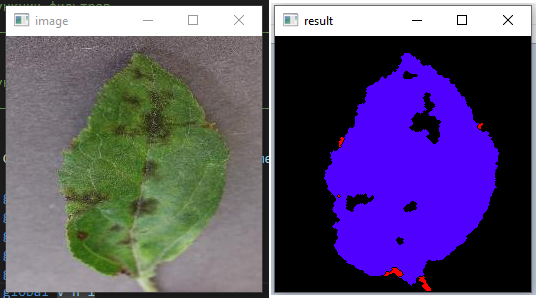
Для разных типов поражений, отличающихся по цвету, нужны разные настройки фильтров. Так же желательно делать изображения без теней и световых бликов на листьях, которые искажают цвета



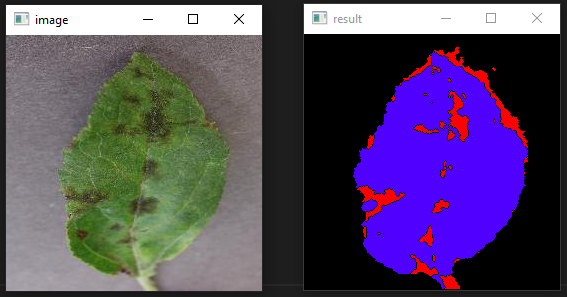
2.Для коричневых точек



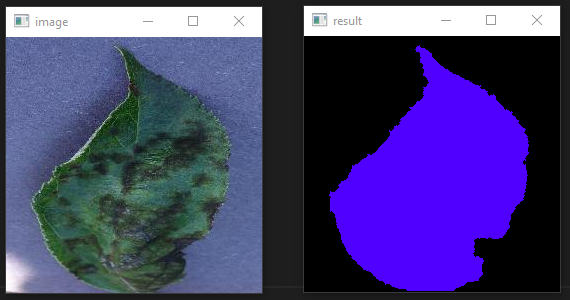
3. Точки коричневые, но цвет листа отличается



4. Точки другого цвета, требуется полная перенастройка



. Корректировка параметров дает некоторые результаты



6. Но не дает положительных результатов для разнообразных типов повреждений