МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

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

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**Отчет по учебной практике**

**«Сегментация листов дерева»**

**Выполнил:** студент группы 381906-1

Яшин Кирилл Евгеньевич.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Научный руководитель:** Гетманская Александра Александровна, преподаватель МОСТ ИИТММ.  
 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород

2021.

Содержание

[1. Введение 3](#_Toc86336696)

[2. Структура программы 4](#_Toc86336697)

[3. Эксперимент 5](#_Toc86336698)

[4. Выводы 11](#_Toc86336699)

# Введение

Основная цель данной работы – сегментировать изображения листов яблони.

В датасете есть 12 изображений листов с повреждениями от болезни на них. Задача состоит в том, чтобы выявить эти повреждения. Работа будет вестись с помощью библиотеки opencv на языке Python. При обработке изображений будет использоваться 5 различных видов фильтров, убирающих шум. После этого результаты работы фильтров будут сравнены, и по результатам будет сделан вывод об эффективности каждого.

# Структура программы

Программа состоит из двух скриптов. Я решил сначала выделить лист с фотографии с помощью cv2.inRange()

1. colortracker.py - простое приложение с элементарным GUI для узнавания параметров цвета в кодировке HSV, которые нужны для сегментации конкретного листа. Оказалось, что у 10 из 12 листов можно спокойно взять одинаковые параметры, а для 4 и 6 они отличаются, потому что там другие цвета у фона и листа.

Скрипт просто динамически накладывает маску на изображение (маска генерируется с помощью cv2.inRange()), наложение происходит с помощью cv2.bitwise\_and().

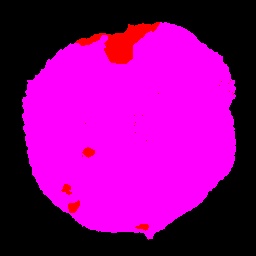
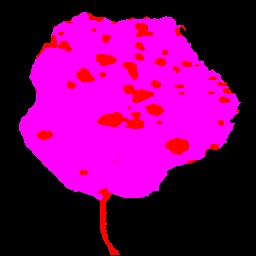
1. main.py – скрипт, который выполняет непосредственно сегментацию с помощью функции damaged\_leaves(image\_num, mode), где image\_num – номер изображения в датасете, mode – фильтр шумоподавления (0 – эрозия, 1 – Non Local Means, 2 – билатеральный, 3 – медианный, 4 – размытие по Гауссу).

# Эксперимент

Сделаем запуски на разных фильтрах и с разными параметрами этих фильтров.

1. Эрозия

Эрозия в совокупности с предварительной сегментацией листа работает, на мой взгляд, лучше всех. Можно посмотреть на самые сложные с точки зрения сегментации болезни изображения и в этом убедиться.



Остальные результаты сегментации представлены в репозитории.

1. Non Local Means

Запустим сначала с параметрами h=7, templateWindowSize=7, searchWindowSize=21

Некоторые изображения были сегментированы неправильно, к примеру, не была найдена здоровая часть листа или в качестве болезни была выделена часть фона.



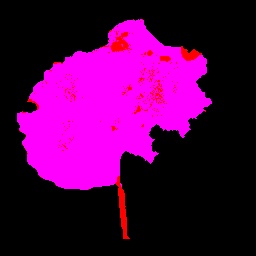
Неправильная сегментация здоровой части, ножка листа была выделена, как болезнь.



Фон был выделен, как болезнь.

Уменьшим параметры (h=3, templateWindowSize=3, searchWindowSize=11).

Результаты такие же.



Увеличим параметры (h=11, templateWindowSize=13, searchWindowSize=27).

Результат стал лучше. К примеру, сегментация здоровой части везде работает правильно.

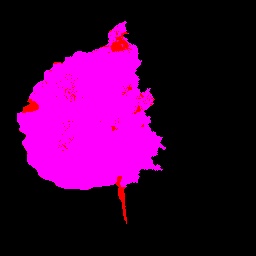


Вывод: лучше использовать фильтр с большими параметрами, хоть это и медленнее.

1. Билатеральный фильтр

Применим билатеральный фильтр с параметрами d=15, sigmaColor=75, sigmaSpace=75.

Билатеральный фильтр показывает худшие результаты, сегментация здоровых частей листа неправильная, и много где.



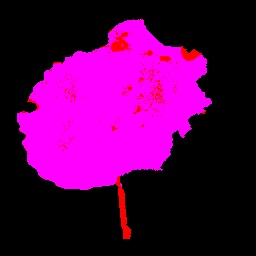
Уменьшим параметры (d=9, sigmaColor=35, sigmaSpace=35).

Результаты стали лучше.



Уменьшим параметры еще сильнее (d=5, sigmaColor=15, sigmaSpace=15).

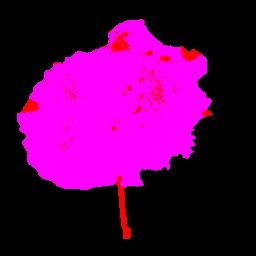
Уже лучше, но далеко не идеально.



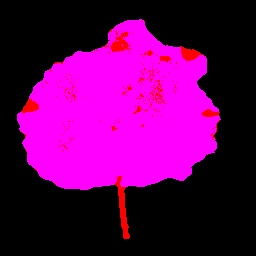
1. Медианный фильтр

Применим медианный фильтр с размером ядра=3.

Медианный фильтр также мешает сегментации здоровой части, хоть и не так сильно, как билатеральный.



Изменим размер ядра на 5. Стало значительно лучше. Но иногда части фона тоже выделяются, как болезнь.



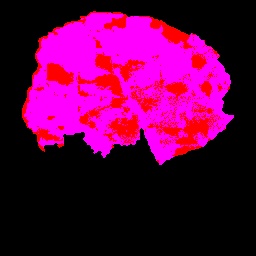
1. Размытие с помощью фильтра Гаусса.

Применим размытие с помощью Гаусса c размером ядра (3,3). И снова проблемы с сегментацией здоровой части.

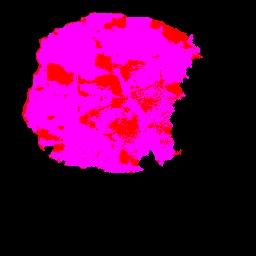
\



Увеличим размер ядра до (5,5).



Стало даже хуже. Попробуем уменьшить размер до (1,1).



Стало лучше, но до медианного фильтра и эрозии еще далеко.

# Выводы

По ходу проведения экспериментов я выяснил, что лучше всего с задачей справляется предварительное применение медианного фильтра с достаточно большим размером ядра (5-7) или эрозии. Худшие результаты показал билатеральный фильтр. Я считаю, что таких результатов я смог добиться потому, что перед обработкой изображения постарался максимально избавиться от фона. В этом мне помогло написанное мной приложение для сегментации фона. Но на достаточно больших датасетах этот процесс нужно автоматизировать. Больше всего проблем было с изображениями 4 и 6, это связано с тем, что на них много собственных теней на листах.