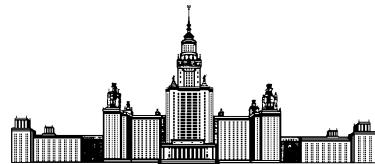


Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики
Кафедра Математических Методов Прогнозирования

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

«Генерация признаков формы объектов на изображении»

Выполнил:
студент 3 курса 317 группы
Суглобов Кирилл Алексеевич

Москва, 2022

Содержание

| | |
|---|-----------|
| 1 Постановка задачи | 1 |
| 2 Данные | 2 |
| 3 Решение задачи | 2 |
| 3.1 Сегментация руки | 2 |
| 3.2 Отделение кисти | 3 |
| 3.3 Распознавание ключевых точек кисти | 6 |
| 3.3.1 Распознавание кончиков пальцев | 7 |
| 3.3.2 Распознавание долин | 7 |
| 3.3.3 Сортировка точек | 8 |
| 3.4 Распознавание сомкнутых пальцев и корректировка долин | 9 |
| 4 Программная реализация | 11 |
| 5 Эксперименты | 13 |
| 6 Заключение | 14 |

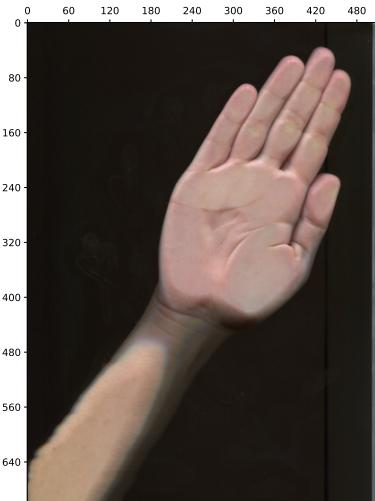
1 Постановка задачи

Цель работы – разработать и реализовать программу для классификации изображений ладоней, обеспечивающую:

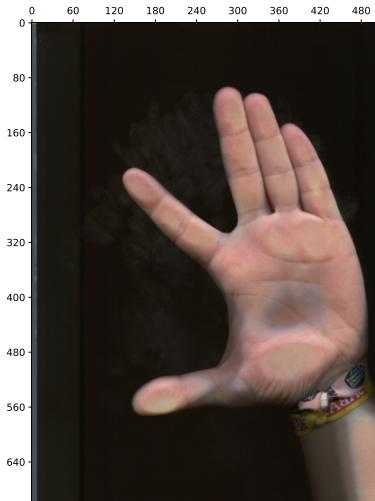
- Чтение и отображение входного изображений ладоней
- Сегментацию изображения на основе точечных, пространственных и морфологических операций
- Определение позы ладоней, жеста – расположения сомкнутых пальцев. Здесь и далее: пальцы нумеруются от 1 до 5, начиная с большого пальца против часовой стрелки. Соответствующие долины нумеруются от 1 до 4. Поза ладони описывается кодом $1^*2^*3^*4^*5$, где значок «*» обозначает «-», если пальцы разомкнуты, или «+», если они прижаты друг к другу
- Определение линии пальцев ладони – ломаной линии, соединяющей точки на кончиках пальцев (tips) с точками в основаниях пальцев (valleys, будем именовать их долинами)
- Вывод маркированного изображения с нанесённой на него линией пальцев и запись в текстовый файл результата работы программы:
 1. Первая строка – код позы ладони, описывающий сжатые и разомкнутые пальцы: $1^*2^*3^*4^*5$
 2. Вторая строка – координаты точек ломаной линии пальцев:
!,0000.tif,T Xt1 Yt1,T Xt2 Yt2,T Xt3 Yt3,T Xt4 Yt4,T Xt5 Yt5,
V Xv1 Yv1,V Xv2 Yv2,V Xv3 Yv3,V Xv4 Yv4,?
Здесь (Xti, Yti) и (Xvi, Yvi) – координаты пикселей i -й найденной точки, определяющих кончики пальцев T (tips) и основания пальцев V (valleys).

2 Данные

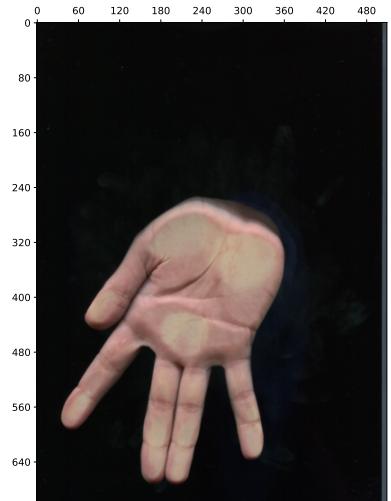
На вход программе подаётся изображение из датасета сканов левых рук человека. Несмотря на то, что все руки левые, при тестировании программы были использованы отражённые изображения. Одно из таких и будет представлено в данно отчёте как демонстрация устойчивости алгоритма и его независимости от типа руки. Входное изображение имеет формат .tif, состоит из 3-х каналов по 8 бит в каждом из них. На [рис. 1](#) показаны примеры входных данных.



(a) 017.tif



(b) 174.tif h-flipped



(c) 280.tif

Рис. 1: Примеры входных изображений

3 Решение задачи

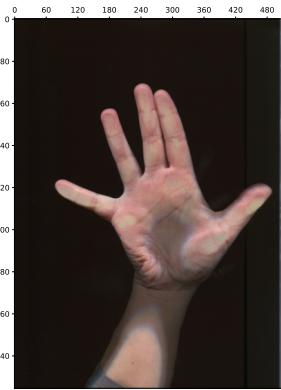
Решение задачи – распознавание жеста и построение линии пальцев – делится на несколько этапов: сегментация руки, выделение кисти, нахождение ключевых точек (вершин tips и долин valleys), распознавание сомкнутых пальцев.

3.1 Сегментация руки

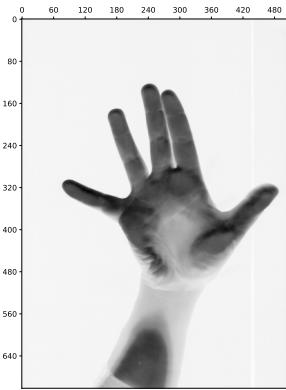
На входных изображениях на тёмном фоне изображена рука. Отделим руку от фона. Сделаем это так, чтобы полученная маска различала сомкнутые пальцы: это нужно для определения долин. Для начала переведём изображение руки в канал red. У человека красная кровь, поэтому красная компонента в руке явно выражена, [рис. 2](#).

Далее, из одноканального изображения создаются три маски:

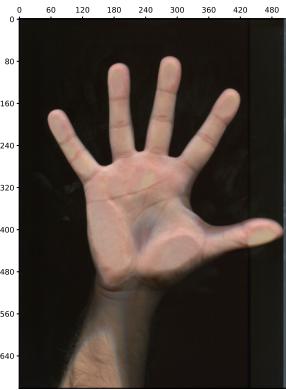
1. Основная маска mask-0: различает сомкнутые пальцы, но может содержать посторонние шумы. Алгоритм получения:
 - img = GaussianBlur(img)
 - img = img - sharpen(Sobel(img))
 - img = smart_bin(img)
 - img = morph(img)
2. Маска коррекции №1 mask-1: не различает сомкнутые пальцы, отделяет от руки шумы. Алгоритм получения:



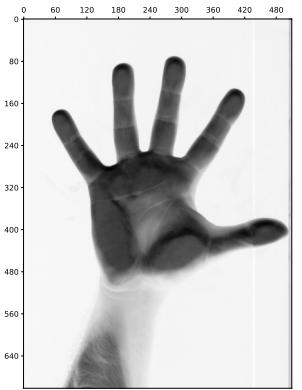
(a) 107.tif



(b) 107.tif red channel



(c) 144.tif



(d) 144.tif red channel

Рис. 2: Сегментация руки, маски

- `img = smart_bin(img)`
- `img = thin(img)`
- `img = morph(img)`

3. Маска коррекции №2 mask-2: не различает сомкнутые пальцы, не содержит некоторых шумов на фоне. Алгоритм получения:

- `img = GaussianBlur(red)`
- `img = smart_bin(img)`
- `img = thin(img)`

`smart_bin` – это собственный алгоритм бинаризации: на гистограмме находится максимальный пик и начинается сканирование от него в противоположных направлениях, пока значения не достигнут определённой отметки частоты встречаемости, в данном случае - медианы частоты. Нас интересует всё, что больше, чем значение яркости, соответствующее точке пересечения правого (большего) сканера с медианой частоты пикселей.

Вычисляется бинарное произведение трёх рассмотренных масок и выбирается наибольший по площади регион. Получаем в результате маску руки, которая различает сомкнутые пальцы, не содержит шумов и не "теряет" ладонь и пальцы – они остаются целыми, неповреждёнными. "Повреждается" только ненужное для решения задачи запястье или предплечье, если оно остается. На [рис. 3](#) показаны этапы получения маски. Единственные потери на такой маске могут быть в местах сильных складок кожи, то есть около долин, [рис. 4](#). Этого, скорее всего, не удастся избежать: при таком способе решения сохранить и промежутки между плотно сжатыми пальцами и не убрать такие же по цвету и форме морщины в долинах не получится. Но этот вопрос разрешается на соответствующем этапе обработки - при поиске долин.

3.2 Отделение кисти

Все маски рук сохраняют ладонь и пальцы в целости. Но на маске присутствует избыток в виде запястья или предплечья. Для решения поставленной задачи нам необходима только кисть, отделим её. К полученной маске руки применим процедуру скелетонизации и процедуру `distanceTransform`. Пересечём две полученные маски: скелета и `distanceTransform`, и оставим в ней только те точки, которые входят в наибольшие 20% по яркости. Возьмём среднее этих точек и получим центр ладони. Даже, если на маске есть части предплечья, наиболее удалёнными от границы маски точками являются точки в центре ладони.

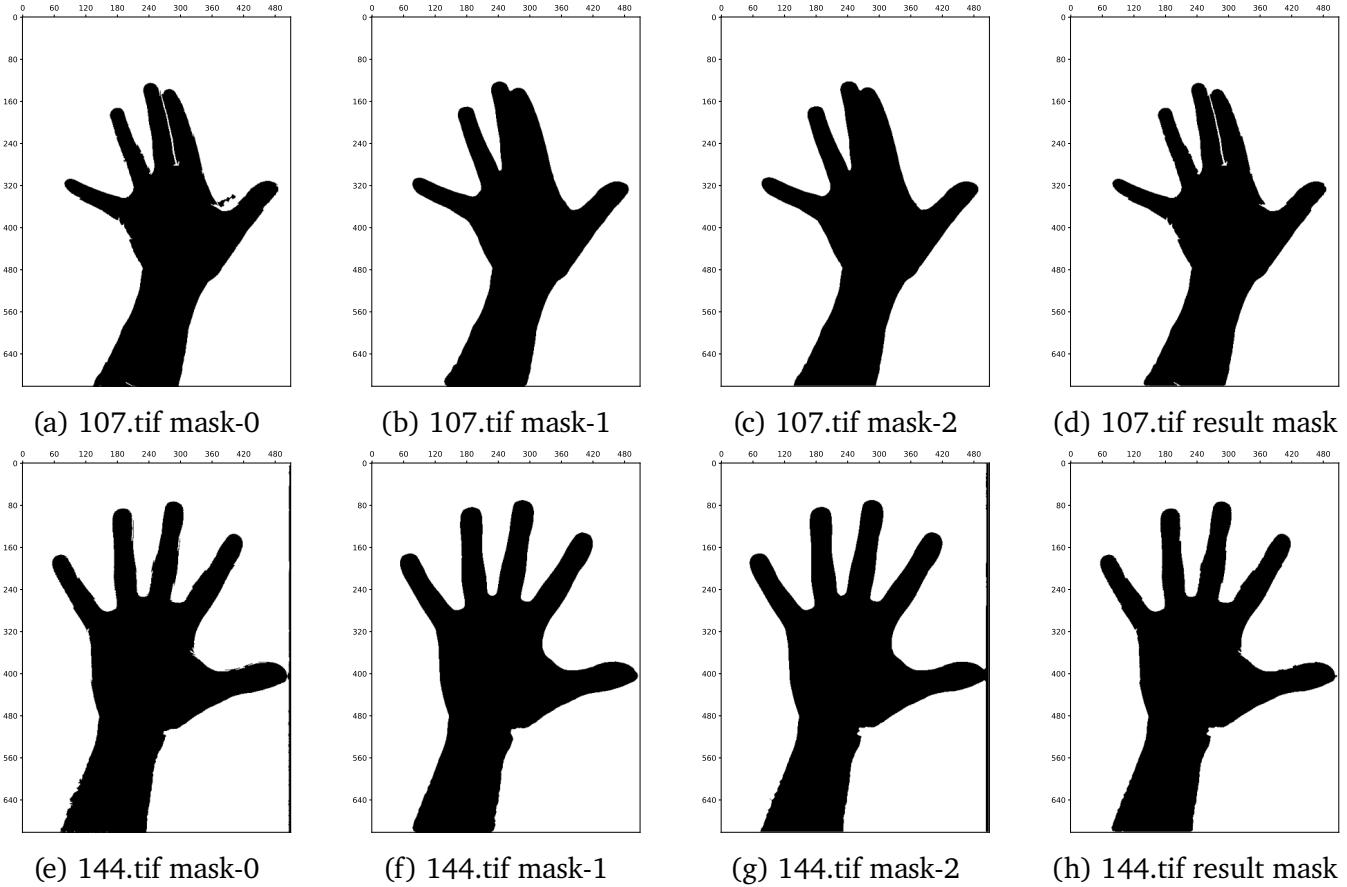


Рис. 3: Сегментация руки, маски

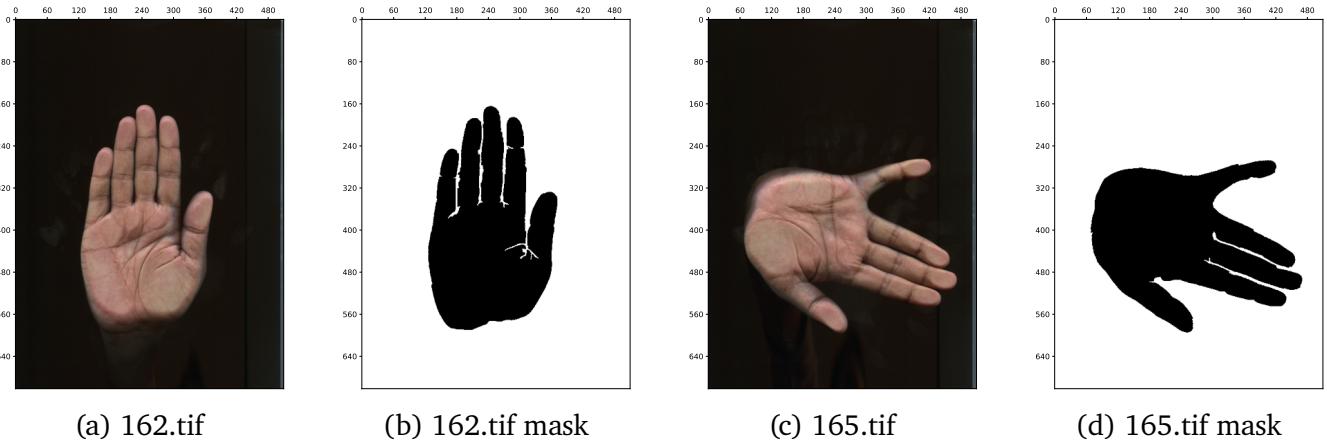
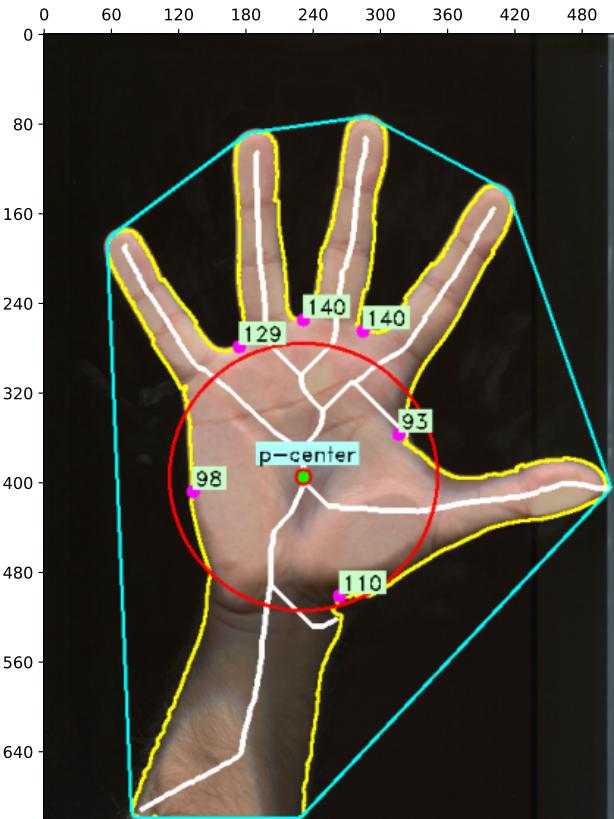


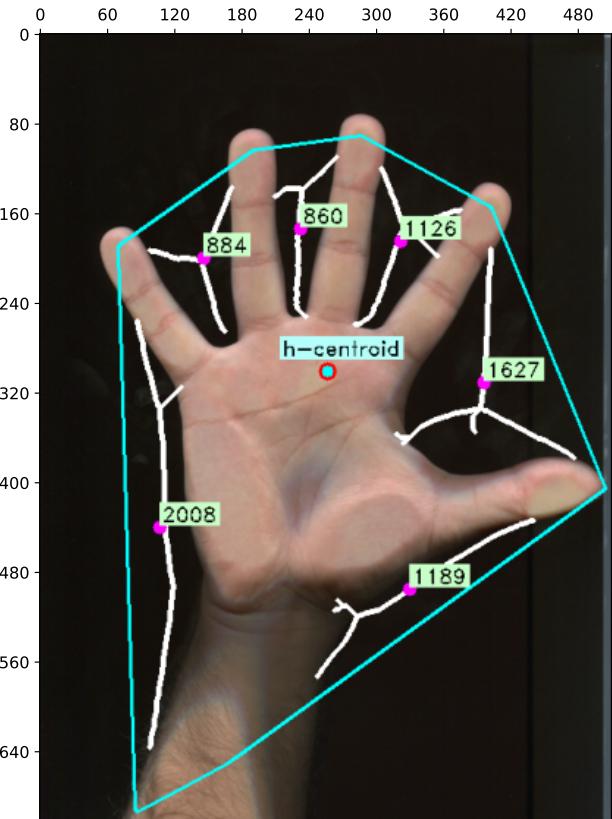
Рис. 4: Сегментация руки, некоторые дефекты

Далее по имеющейся маске найдём топ-6 относительных минимумов расстояния от центра ладони до контура маски. Относительный экстремум – это экстремум дискретного набора точек, в котором для того, чтобы считать рассматриваемую точку экстремумом нужно сравнить её с k соседними точками до и после неё. И если в среди этих $2k$ точек на большей их части подтверждается «экстремальность» точки, то она признаётся экстремумом. В данном случае значение ширины окна = $2k = 100$. Этого достаточно, чтобы отмечались не небольшие впадины, а основные. И берём топ-6 из них, потому что пространство вокруг ладони локально делится пальцами и предплечьем на 6 «пустых» частей (видно с выпуклой оболочкой ладони). Получив эти точки, возьмём медиану (робастная оценка) расстояний от центра ладони до них и построим окружность с таким радиусом. Такая окружность хорошо приближает ладонь, по крайней мере в районе запястья.

Выпуклая оболочка контура ладони, контур ладони, относительные минимумы с подписями расстояний до центра, окружность ладони, центр ладони и скелет руки изображены на рис. 5а



(a) 144.tif palm center



(b) 144.tif hollows centroid

Рис. 5: Центр ладони и центроид впадин ладони

Исходя из того, что окружность ладони проходит примерно по запястью, отделить кисть можно по касательной к этой окружности. Для этого нужно определить некое среднее «направление руки», которое перпендикулярно запястью. В силу анатомического строения руки, запястью перпендикулярно некое усреднённое направление пальцев. Построим скелеты обратной маски руки, пересечённой с выпуклой оболочкой скелета самой руки. Так мы отделим друг от друга полости вокруг руки, почти не уменьшив полости между пальцами, но можем уменьшить «лишнее» пространство слева и справа от руки. Посчитаем площади всех скелетов и их центроиды. Далее применим следующие два метода отбора скелетов:

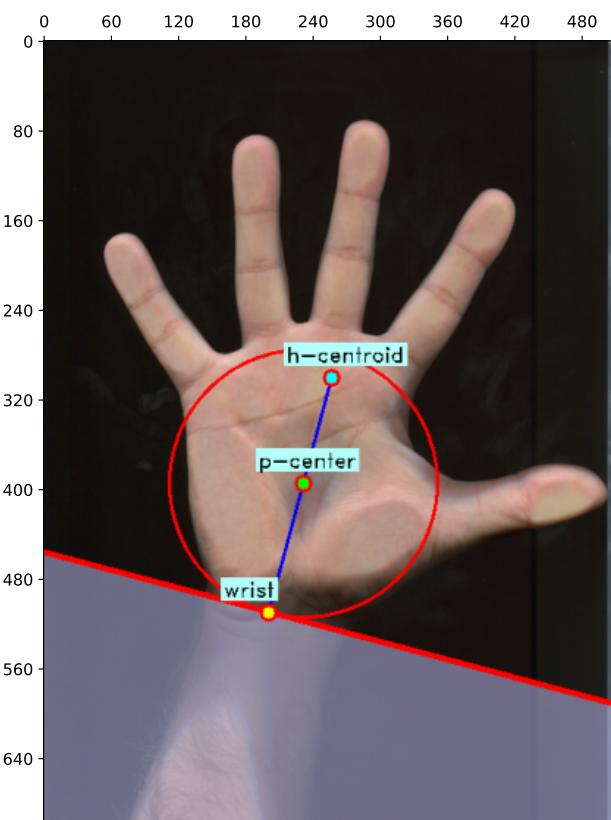
1. Метод №1: отберём из скелетов топ-4 по модулю разности площади с медианой площади и оставим их только те, которые различаются с медианой не более, чем на 300 пикселей. Такой метод связан с тем, что наиболее важные для направления руки полости между 2-3, 3-4 и 4-5 пальцами обычно медианные по площади скелета. Полости между 1-2 пальцами обычно имеет больший скелет, если палец отогнут. Поэтому мы отбираем топ-4 и исключаем слишком сильно отличающиеся полости.
2. Метод №2: отберём из скелетов только те, которые по площади с медианой площади отличаются не более, чем на 100 пикселей. Этот метод на случай, если в первом методе мы отбросили слишком много скелетов.

Получили две независимые группы скелетов в результате работы двух методов. Далее в каждой из групп считаем центроиды и среднее арифметическое координат точек, усредняем

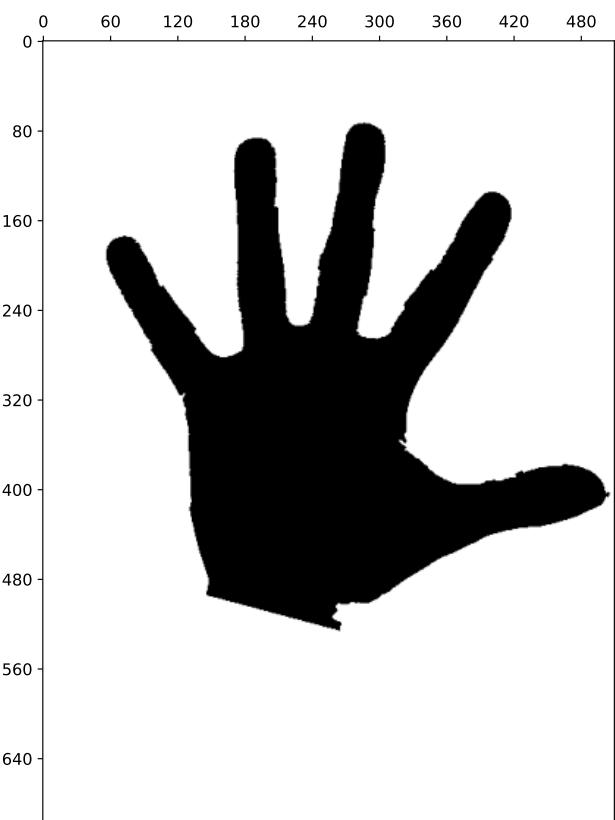
их. Наконец, финальные результаты каждой из групп усредняем с весами 0.3 и 0.7 соответственно. Получаем достаточно хорошее приближение некоего центроида полостей, который относительно центра ладони указывает «направление руки».

Выпуклая оболочка скелета руки, центроиды скелетов полостей с подписями площадей, центроид полостей и скелеты полостей вокруг ладони изображены на [рис. 5b](#)

Отрезок, соединяющий центр ладони и центроид полостей достаточно хорошо приближает усреднённое направление пальцев. Причём этот отрезок всегда достаточной длины, не короткий. То есть центроид уверенно показывает направление руки от центра. Продлим этот отрезок до пересечения с окружностью ладони. Пересечение будет в двух точках, возмём наиболее удалённую от центроида полостей. Это будет точка, лежащая на запястьи, а перпендикуляр. А касательная к окружности ладони в этой точке и будет отделять нужную нам кисть от остальной части маски. Итоговая геометрия руки вместе с точкой запястья и маска кисти изображены на [рис. 6](#).



(a) 144.tif hand geometry



(b) 144.tif only hand mask

Рис. 6: Геометрия руки и маска кисти

3.3 Распознавание ключевых точек кисти

Итак, на данном этапе решения задачи имеется маска кисти, центр ладони и точка запястья. Исходя из анатомии кисти:

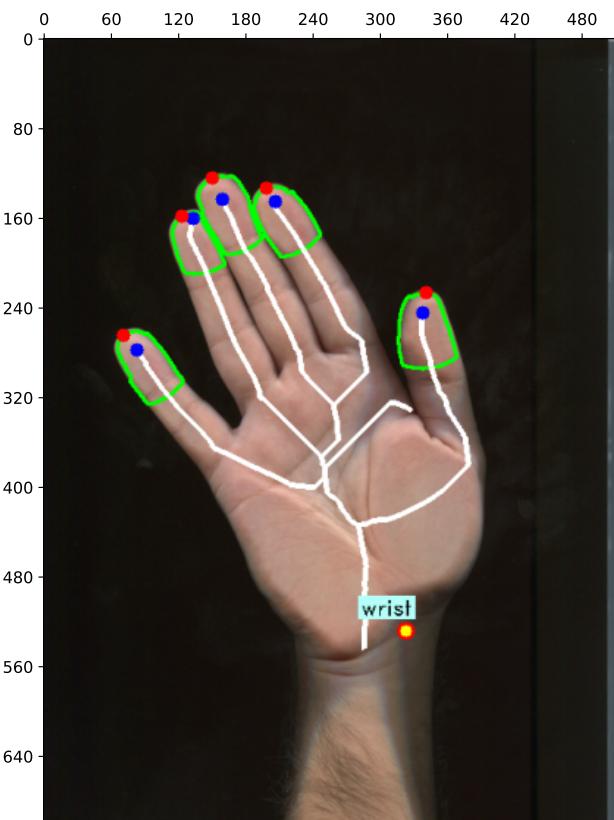
- Кончики пальцев – это наиболее удалённые от запястья точки кисти / скелета кисти, так как кости пальцев рук «сходятся» именно в костях запястья.
- Долины пальцев – это наиболее близкие к центру ладони точки кисти между парами соседних пальцев, так как ладонь, на границе которой и образуются долины, расположена, собственно, вокруг своего центра.

3.3.1 Распознавание кончиков пальцев

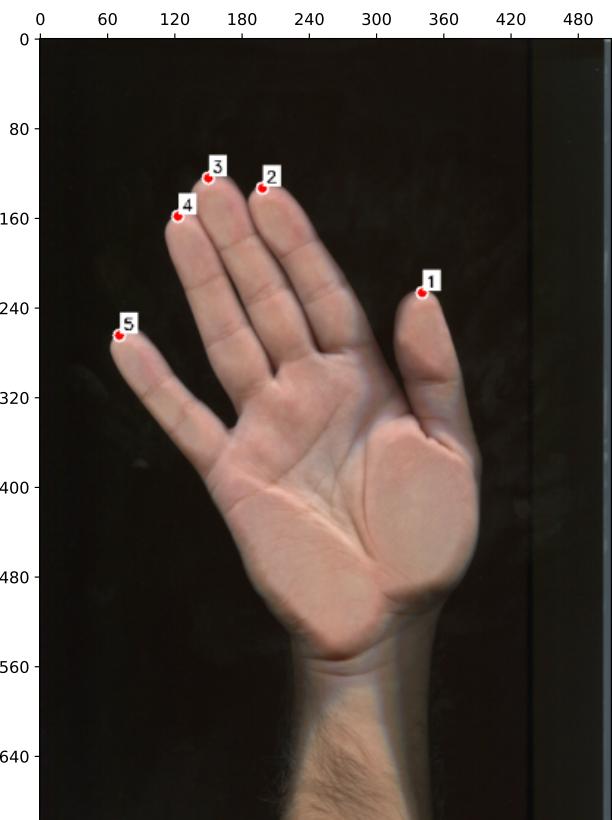
Обрежем полученный на этапе определения центра ладони скелет руки по маске кисти. На полученном скелете кисти найдём топ-5 относительных максимума (с достаточным окном в 100 пикселей) по расстоянию от запястья. Это будут кончики скелетов пальцев. Вокруг каждой такой точки вырежем окрестность пальца радиусом 50 пикселей – получим примерно подушечки пальцев. Для каждой такой подушечки найдём наиболее удалённую от запястья точку – кончики пальцев.

Это не очень эффективный метод: раз уже определены подушечки, можно было бы придумать более точный метод, который по скелет каждого пальца продолжал бы до пересечения с границей. Таким образом получили кончики пальцев. После, зная контур кисти, который настроен на проход против часовой стрелки, сортируем кончики в порядке вхождения в контур. Теперь кончики идут циклически последовательно. Найдём соседние максимально удалённые друг от друга кончики, выберем наименьший из них по индексу в контуре кисти и сдвинем весь массив кончиков циклически так, чтобы этот выбранный кончик стал первым. В итоге получили пять последовательных кончиков пальцев: от 1 до 5, либо наоборот. Сортировка будет произведена после нахождения долин.

Скелет кисти, точка запястья, **кончики скелетов пальцев**, **подушечки пальцев** и **кончики пальцев** изображены на [рис. 7а](#).



(a) 150.tif tips geometry



(b) 150.tif tips

Рис. 7: Кончики пальцев и их геометрия

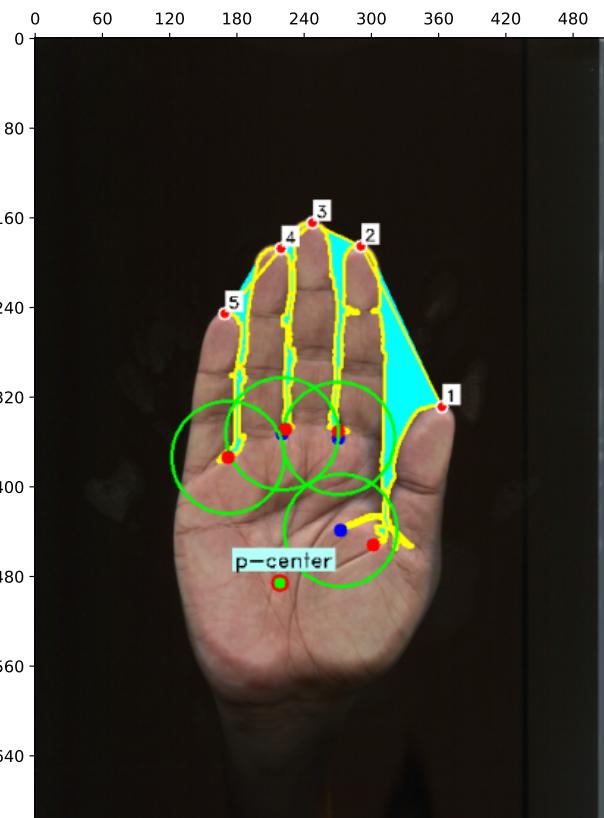
3.3.2 Распознавание долин

Для определения долин используем две маски: та маска кисти, которая имеется и маска, в которой исправляются дефекты, вызванные большими складками кожи. «Исправляются»

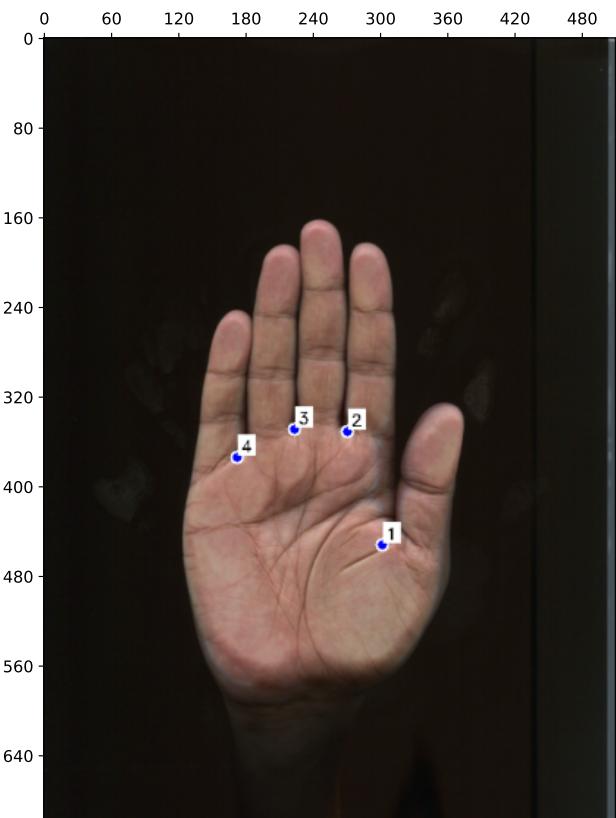
вырезанные складки кожи на маске с помощью морфологической операции закрытия, медианного фильтра и повторного закрытия. Для каждой из масок найдём контур. Теперь для каждой из двух соседних кончиков пальцев (кроме первого с последним: они не соседи) на каждой из масок ищем ближайшие к кончикам точки, принадлежащие контуру. Для каждой из таких двух точек вырезаем участок конутра между ними и заполняем его, то есть делаем маску полости между пальцами. Таким образом, для каждой из двух масок найдена полость между каждыми соседними пальцами. Для каждой такой полости находим ближайшую в центре ладони точку, это потенциальная долина. То есть для каждой полости между двумя соседними пальцами имеем две потенциальные долины – по двум маскам.

По предположению, что дефекты, связанные с вырезанными складками кожи «уходят» в ладонь не более, чем на 50 пикселей, выбираем в качестве долины долину с потенциальной исправленной маски, если она находится в 50-ти пикселях от потенциальной долины оригинальной маски. Иначе оставляем долину с оригинальной маски. Таким образом, если в результате «исправления» дефектов маски случится закрытие в месте плотного смыкания пальцев, то потенциальная долина с исправленной маски будет, скорее всего, за окрестностью оригинальной долины.

Оригинальный контур кисти, кончики пальцев, **полости между пальцами**, **оригинальные долины кисти**, **долины «исправленной» кисти**, **окрестности оригинальных долин** и **центр ладони** изображены на **рис. 7**.



(a) 162.tif valleys geometry



(b) 162.tif valleys

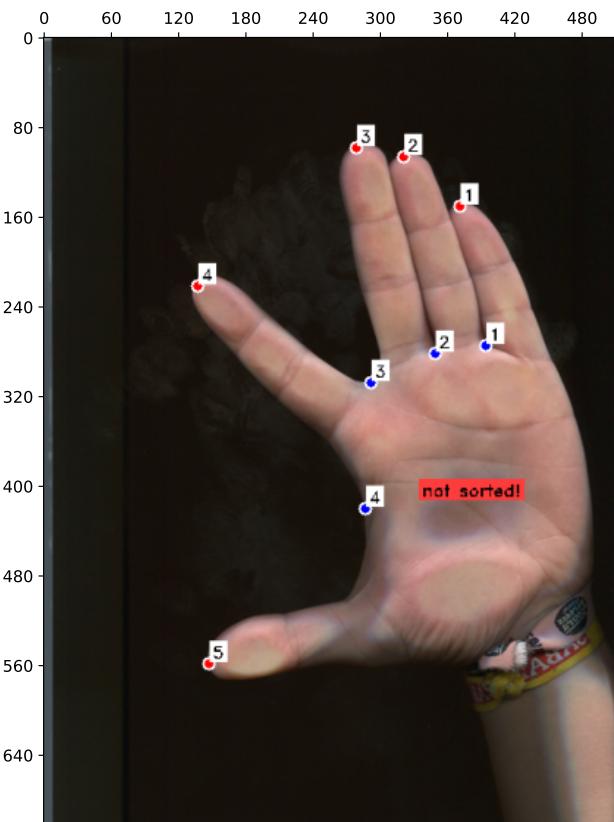
Рис. 8: Долины ладони и их геометрия

3.3.3 Сортировка точек

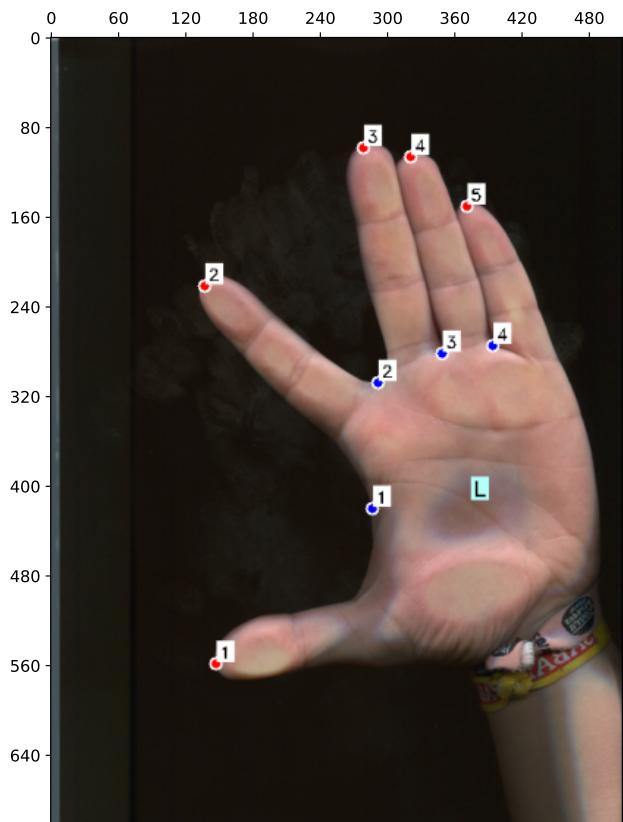
В задании написано, что в датасете представлены изображения левых рук. Изображения выглядят так, как будто на стекло положили правую руку, что, видимо, связано с особенностями сканирования. Будем считать, что не знаем этого и что это изображения правых

рук, которые клали на стекло. Для любой, неважно, как расположенной, и неважно, какой, руки мы на текущем шаге имеем кончики пальцев и соответствующие им долины. И они расположены либо в прямом, либо в обратном порядке (либо от 1 до 5, либо наоборот). То есть начиная с большого пальца или с мизинца, но всегда против часовой стрелки, по обходу контура кисти. Значит, на представленных изображениях, на правых, как было оговорено выше, руках, получаем сразу верный порядок (потому что пальцы расположены против часовой стрелки на таких изображениях).

Но для решения задачи для разных рук заметим следующее: наибольшее расстояние от долины до кончика пальца (у долины два пальца) – от долины большого указательного пальцев до указательного пальца. Тогда, учитывая, что мы имеем на данном этапе, наибольшее расстояние между долиной и кончиком пальца может быть либо от долины 1-2 до 2-го пальца, либо от долины 4-5 до пальца 4. В первом случае оставляем нумерацию кончиков и долин как есть и объявляем руку правой, иначе делаем обратный порядок для кончиков и долин и называем руку левой. На [рис. 9](#) показан результат сортировки для левой (по договорённости) руки на горизонтально отражённом изображении из датасета (таких рук в датасете не представлено).



(a) 174.tif (h-flipped) unsorted tips and valleys



(b) 174.tif (h-flipped) sorted tips and valleys

Рис. 9: Сортировка кончиков и долин

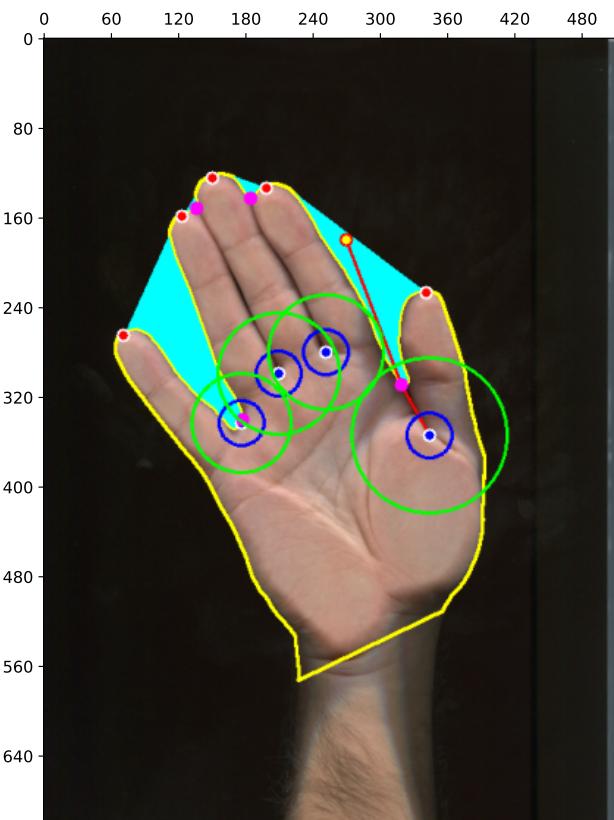
3.4 Распознавание сомкнутых пальцев и корректировка долин

Для распознавания сомкнутых пальцев нужно найти долины на маске кисти, которая не разделяет сомкнутые пальцы. Для этого отлично подходит `mask-1` из [разд. 3.1](#). Обрежем эту маску по линии запястья, как в [разд. 3.2](#). Как и в случае поиска долин в [разд. 3.3.2](#), строится контур маски кисти, на которой не различаются сомкнутые пальцы, и для каждой пары кончиков соседних пальцев ищутся ближайшие соответствующие точки контура. А далее

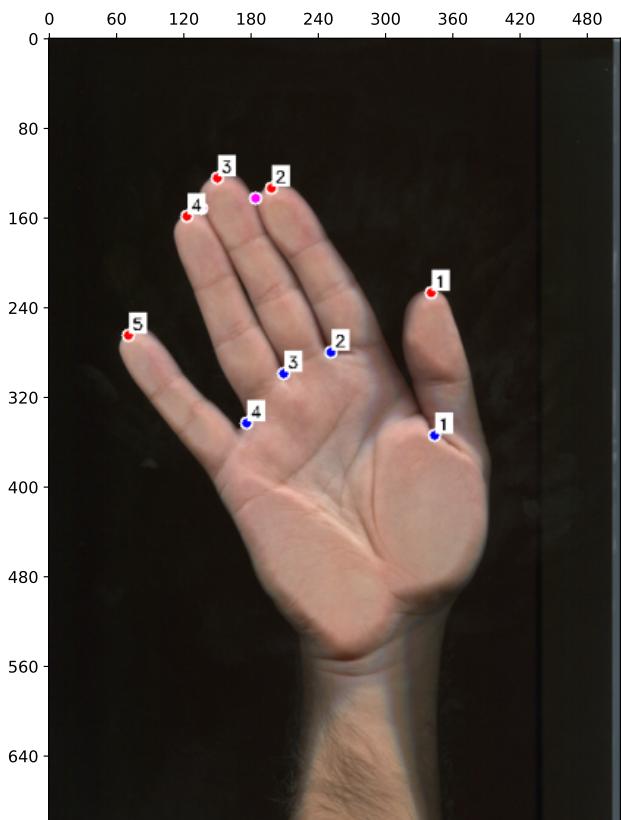
для каждого участка контура между соседними кончиками строится выпуклая оболочка, у которой ищется наибольший дефект выпуклости. Назовём каждую такую фальшивой долиной.

Таким образом, имеем набор долин и фальшивых долин для каждой пары соседних пальцев. Для каждой долины строится две окрестности и выполняются соответствующие действия:

1. Первая окрестность для всех долин одинаковая: $r_1 = 20$ пикселей. При по падании фальшивой долины в эту окрестность берётся её среднее с настоящей долиной и эти новые координаты объявляются новой долиной, а соответствующие пальцы – разомкнутыми. Такое усреднение корректирует последствия от некоторых дефектов точной маски. Но эта коррекция долин не может быть проведена, если пальцы сомкнуты, так как фальшивая долина в этом случае находится недалеко от кончиков пальцев, в месте смыкания пальцев.
2. Вторая окрестность: пусть l – это минимум из расстояний от долины до соответствующих ей пальцев. Тогда радиус этой окрестности $r_2 = \frac{l}{2} + 5$ для первой долины и $r_2 = \frac{l}{3}$ для остальных. Буквально это означает, что вторая долина покрывает область примерно первой фаланги, это область допустимой погрешности. Если фальшивая долина не попала во вторую окрестность, то пальцы считаются сомкнутыми. А если фальшивая долина попадает во вторую окрестность, но не попадает в первую, то:
 - Для первой долины: сравнивается расстояние от фальшивой долины до долины и от фальшивой долины до полусуммы соответствующих кончиков пальцев. Если фальшивая долина ближе к долине, то пальцы считаются разомкнутыми, иначе – сомкнутыми.
 - Для остальных долин: пальцы считаются разомкнутыми.



(a) 150.tif



(b) 150.tif

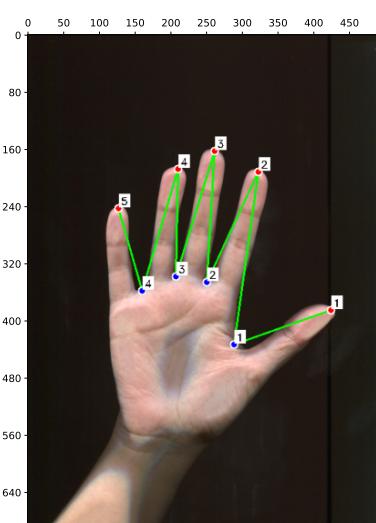
Рис. 10: Распознавание сомкнутых пальцев

- Если фальшивая долина не попала во вторую окрестность (вторая окрестность содержит в себе первую), то пальцы считаются разомкнутыми.

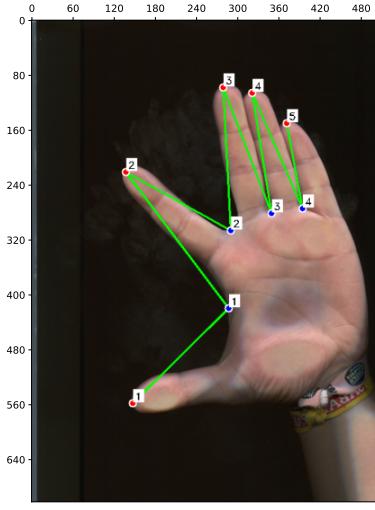
Некоторые сложности были обусловлены особой анатомией большого пальца: у многих людей, когда он прижат, точка смыкания находится примерно на уровне конца одной и начала другой фаланги, а сам кончик не прижат.

Контур фальшивой маски кисти, полости между соседними пальцами, фальшивые долины, первые окрестности, вторые окрестности долины и кончики пальцев изображены на рис. 10. Фальшивые долины оставляем только у сомкнутых пальцев и получаем, что теперь известны кончики пальцев, долины и какие пальцы сомкнуты, а какие разомкнуты.

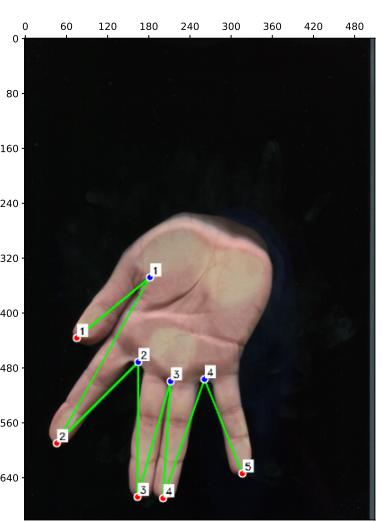
Таким образом, имеем алгоритм, который распознаёт кончики пальцев, долины ладони и жесты, основанные на сомкнутых пальцах. Результаты работы алгоритма, жесты и построенная линия пальцев показаны на рис. 11.



(a) 001.tif result, 1-2-3-4-5



(b) 174.tif (h-flipped) result,
1-2-3+4+5



(c) 280.tif result, 1+2-3+4-5

Рис. 11: Результат работы алгоритма

4 Программная реализация

Программа написана на языке программирования Python в среде Jupyter Notebook с использованием библиотек:

- numpy для векторных и матричных вычислений
- [3] SciPy для поиска относительных экстремумов дискретного множества, для матрицы расстояний
- matplotlib для отрисовки графиков
- pillow для ввода и вывода изображений
- os для работы с системными директориями (чтение, запись)
- [2] scikit-image для операций над изображениями
- [1] OpenCV для поиска контуров, для пространственных операций, для вывода изображений в окнах

При разработке программы использовались собственные идеи и знания, полученные на [4].

Программа находится в Jupyter Notebook, который состоит из импортов, вспомогательных функций, алгоритма решения задачи, и, собственно, программы (ячейки после «Program»).

```
Program
Ввод [*]: # parameters
input_folder = 'Training'
vis_folder = 'vis'
output_folder = 'out'

chk_vis = False

# input
name = input()
img = cv2.imread(f'{input_folder}/{name}')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

# show input file
show_images_cv([img], [name])

# processing
img_res, gesture, coords, vis_img_arr = hand_points_detection(img, name, vis=chk_vis)
out_info = f'{gesture}\n{coords}'

# output
if chk_vis:
    for i, vis_img in enumerate(vis_img_arr):
        show_images([vis_img], save_path=f'{vis_folder}/vis_{i}_{name}')
print(out_info)

# save
try:
    Image.fromarray(img_res).save(f'{output_folder}/res_{name}')
    with open(f'{output_folder}/Results_{name}.txt', 'w') as fout:
        fout.writelines(out_info)
except:
    print('Can not save result!')

# show result file
show_images_cv([img_res], [f'{name} result'])

300.tif
```

Рис. 12: Запуск программы и ввод пути к изображению

Для взаимодействия с программой нужно запустить последнюю ячейку, вписать путь к входному файлу, то есть обрабатываемому изображению.

Далее появится окно с входным файлом, нужно нажать любую клавишу. После программа начнёт работать и через некоторое время появится окно с результатом работы, размеченным изображением (с построенной линией пальцев и отмеченными ключевыми точками на нём).

Перед выводом размеченного изображения на консоль выводятся две строчки: в первой распознанный жест, во второй – координаты ключевых точек кисти, кончиков пальцев и долин ладони. Эта же информация записывается в текстовый файл.

Также перед работой программы предлагается настроить её - настроить перед запуском значения следующих переменных в ячейке:

- `input_folder` - директория, в которой находится изображение, имя которого введено
- `vis_folder` - директория, куда сохранять промежуточные результаты работы программы. Надо самому создать её
- `output_folder` - директория, куда сохранять результаты работы программы (размеченное изображение и текстовое описание фигур на изображении)
- `chk_vis` - флаг, отвечающий за вывод и сохранение промежуточных результатов работы программы

Program

```

Ввод [22]: # parameters
    input_folder = 'Training'
    vis_folder = 'vis'
    output_folder = 'out'

    chk_vis = False

    # input
    name = input()
    img = cv2.imread(f'{input_folder}/{name}')
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    # show input file
    show_images_cv([img], [name])

    # processing
    img_res, gesture, coords, vis_img_arr = hand_points_detection(img, name, vis=chk_vis)
    out_info = f'{gesture}\n{coords}'

    # output
    if chk_vis:
        for i, vis_img in enumerate(vis_img_arr):
            show_images([vis_img], save_path=f'{vis_folder}/vis_{i}_{name}')
        print(out_info)

    # save
    try:
        Image.fromarray(img_res).save(f'{output_folder}/res_{name}')
        with open(f'{output_folder}/Results_{name}.txt', 'w') as fout:
            fout.writelines(out_info)
    except:
        print('Can not save result!')

    # show result file
    show_images_cv([img_res], [f'{name} result'])

300.tif
1-2-3+4-5
!300.tif,T 418 309,T 399 141,T 339 79,T 294 84,T 184 117,V 297 378,V 285 272,V 249 248,V 191 253,?

```

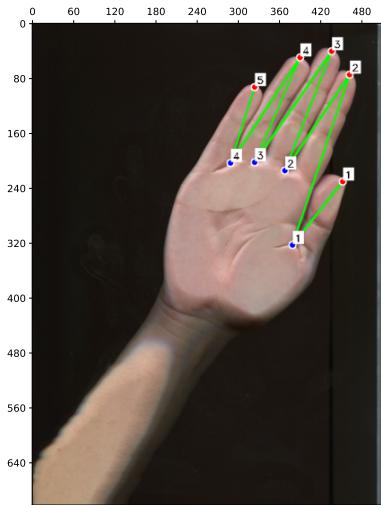
Рис. 13: Вывод программы

5 Эксперименты

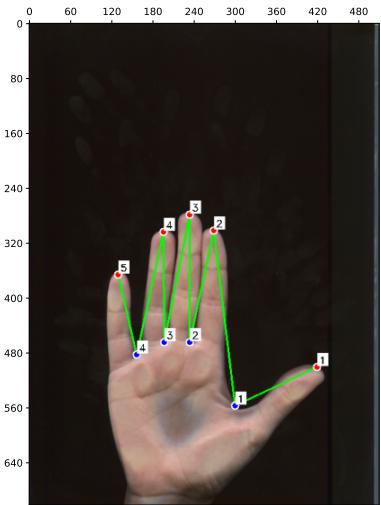
Предложенный алгоритм распознавания ключевых точек кисти и жестов смыканием пальцев был протестируирован на всех предлагаемых образцах. Некоторые результаты представлены на [рис. 14](#).

Среди экспериментов были следующие:

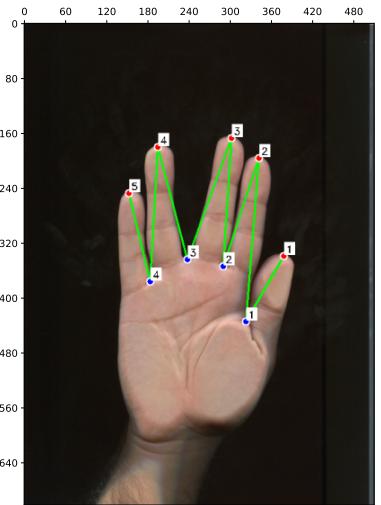
- Первая версия алгоритма была полностью основана на дефектах выпуклости. По контуру маски руки (а не кисти) строилась выпуклая оболочка, топ-4 максимальных дефекта объявлялись долинами, а соответствующие им опоры выпуклости - кончиками пальцев. Далее точки сортировались и выводились. Алгоритм работал плохо: на выданных образцах при разметке линии пальцев ошибка была на 9/67 изображениях без учёта того, что некоторые точки были неточными, смещёнными. Ошибки были на изображениях, где мизинец «не дотягивал» до выпуклой оболочки или где один из 4-х наибольших дефектов был не между пальцев, а в районе запястья или предплечья. Решение было простым и некачественным, оно находится в `old/conv_solution.ipynb`.
- На этапе сегментирования руки нужна была точная маска, которая разделяла бы даже сомкнутые пальцы. Для этого подбирались параметры оператора Собеля, с помощью которого находятся границы, которые вычитаются из изображения. И всё равно качество маски было плохим, потому что складки кожи, в частности повторяющие вид межпальцевого пространства (для сомкнутых пальцев) так же вычитались из руки, образовывая рваную ладонь. Тогда было решено задействовать гауссовское размытие и наведение резкости уже на изображение с вычтеными границами. Так не очень ясные границы складок размывались до того, как оператор Собеля их сильно выделит. Это сильно улучшило качество. А дефекты маски, которые всё же оставались, на этапе определения впадин слаживались медианным фильтром после морфологического закрытия.



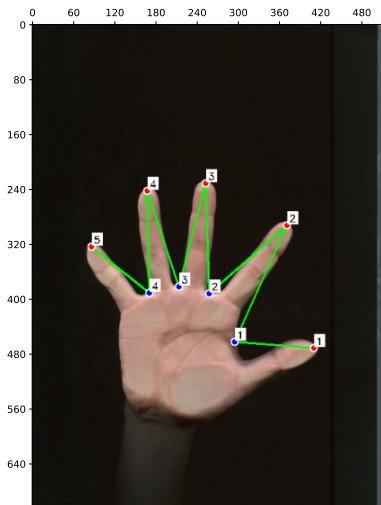
(a) 017.tif result, 1+2+3+4+5



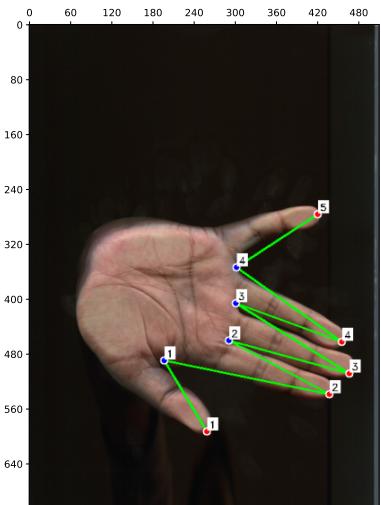
(b) 090.tif result, 1-2+3+4-5



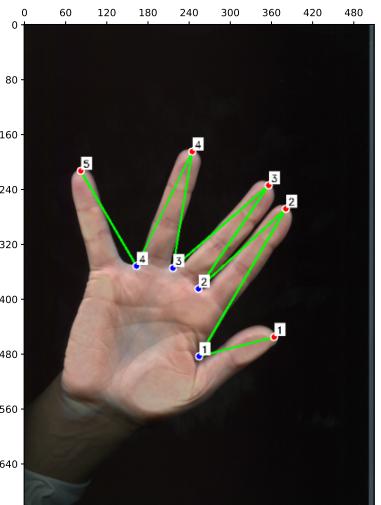
(c) 146.tif result, 1+2+3-4+5



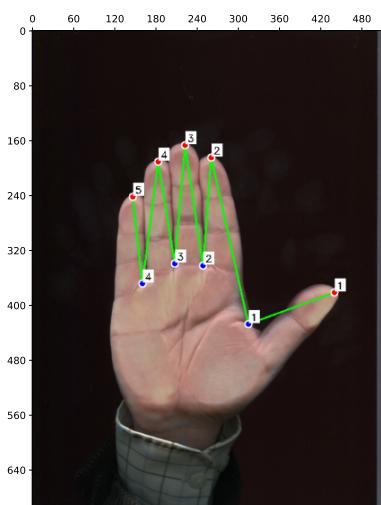
(d) 156.tif result, 1-2-3-4-5



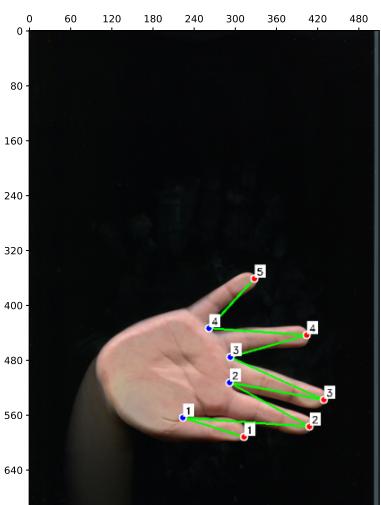
(e) 165.tif result, 1-2+3+4-5



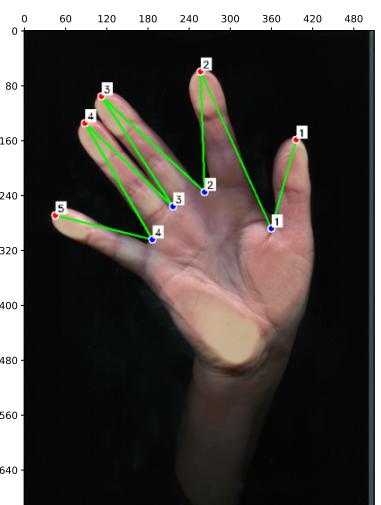
(f) 248.tif result, 1-2+3-4-5



(g) 267.tif result, 1-2+3+4+5



(h) 286.tif result, 1+2-3-4-5



(i) 295.tif result, 1-2-3+4-5

Рис. 14: Некоторые результаты работы алгоритма