Метод предобработки изображения кисти руки в системе распознавания жестовых символов

Танцевов Г. М.[a]

[a] *МГТУ им. Н. Э. Баумана, Москва, Россия*

Аннотация

Рассмотрены методы обработки изображения с целью выявления наиболее применимого на этапе предобработки изображения в задаче распознавания жестовых символов. Были поставлены эксперименты для сравнения работы описанных алгоритмов, на основании результатов которых был проведен сравнительный анализ. Были выделены два метода: выделение силуэта и построение скелета по ключевым точкам. Предложены возможные пути их улучшения для дальнейшего построения метода распознавания жестовых символов.

Ключевые слова: жесты, выделение границ, скелет кисти, выделение контура

1. Введение

Одной из перспективных задач машинного зрения является распознавание жестовых символов. Актуальность данной темы обусловлена множеством сфер применения: от новых методов взаимодействия с ПК до систем распознавания жестовых языков. Как правило, такие системы состоят из трех частей:

1. Получение данных о жесте
2. Предобработка данных
3. Классификация

В качестве исходных данных можно использовать снимок с камеры, например, смартфона. В этом случае, для упрощения классификации, важен этап предобработки данных. На данном этапе необходимо выделить на изображении основные признаки исходного жеста. Алгоритмы, применимые для достижения данной цели, можно разделить на следующие группы:

- Выделение контура фигуры
- Выделение силуэта кисти руки
- Построение скелета кисти руки

Электронная почта: email_tantsevov@gmail.com (Танцевов Г. М.)

2. Выделение контура фигуры

Для выделения контура кисти руки можно использовать операторы преобразования изображения. К таким методам можно отнести:

- Оператор Собеля [1]
- Оператор Прюитт [2]
- Перекрестный оператор Робертса [3]
- Оператор Кэнни [4]

Рассмотрим каждый подробнее:

2.1. Оператор Собеля

Основная идея оператора Собеля[1] заключается в вычислении градиента освещенности каждой точки изображения. Вычисление производится примерное с помощью свертки изображения двумя сепарабельными целочисленными фильтрами размера 3x3 в вертикальном и горизонтальном направлениях (рисунок 1). Благодаря этому вычисление работа данного оператора имеет низкие трудозатраты. В результате получаются два новых изображения Gx и Gy, в каждой точке которого записано приближенное значение производных по x и по y соответственно. Пусть A - исходное изображение, тогда вычисляются они следующим образом:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{1}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{2}$$

В итоге значение градиента вычисляется как $G = \sqrt{G_x^2 + G_y^2}$, а его направление как $\theta = \arctan(\frac{G_x}{G_y})$.

Результат показывает скорость изменения яркости изображения в конкретной точке, т.е. вероятность ее нахождения на границе изображения.

2.2. Оператор Прюитт

Данный метод[2], как и оператор Собеля, использует свертку обесцвеченного изображения (рисунок 1) ядром размера 3x3. Отличие заключается в способе задачи маски:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{3}$$

$$G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{4}$$

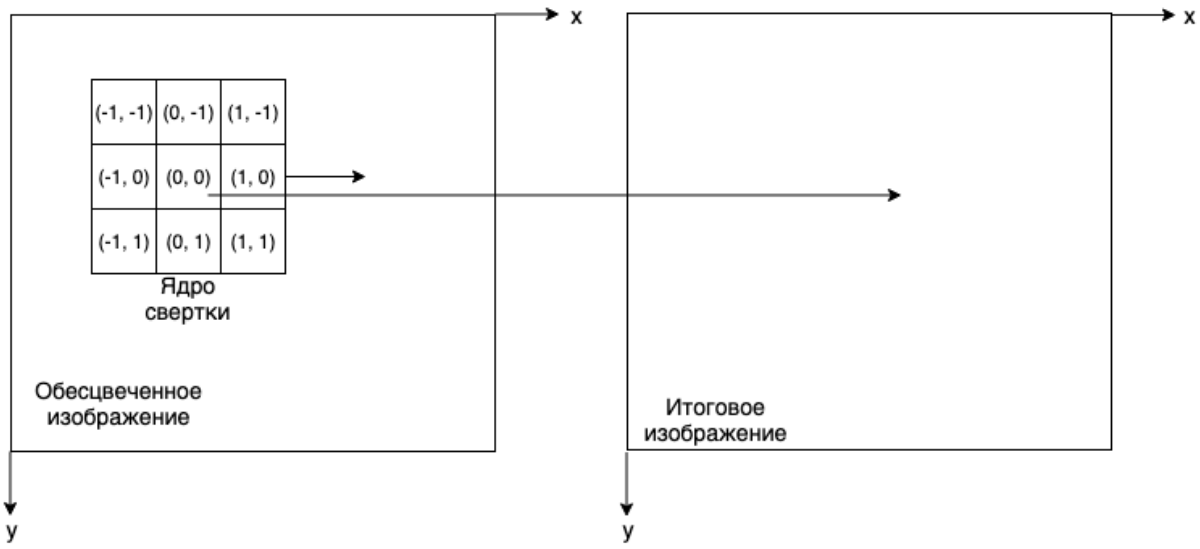

Рис. 1: Свертка изображения для получения контура

Из-за меньшего значения средних элементов итоговое изображение имеет более явный эффект сглаживания.

2.3. Перекрестный оператор Робертса

Рассмотрим область 3x3, представленную ну рисунке 2.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Рис. 2: Окрестность 3x3 внутри изображения

Вычисление первых частных производных, которые обозначают перепад яркости, в точке z_5 можно провести следующим образом:

$$G_x = (z_9 - z_5) \tag{5}$$
$$G_y = (z_8 - z_6) \tag{6}$$

Для вычисления данных производных в каждой точке изображения в данном методе[3] применяется свертка изображения двумя ядрами размера 2x2:

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{7}$$

В результате получается изображение пространственного градиента исходного изображения, где точки с наибольшим значением соответствуют границе.

Проблемой данного метода является отсутствие четко выраженного центрального элемента у ядра свертки. Но в следствии этого недостатка алгоритм так же имеет высокую скорость обработки изображения.

2.4. Оператор Кэнни

Данный фильтр[4] был разработан с учетом удовлетворения следующих свойств:

- хорошее обнаружение (Кэнни трактовал это свойство как повышение отношения сигнал/шум);
- хорошая локализация (правильное определение положения границы);
- единственный отклик на одну границу.

Алгоритм состоит из пяти последовательных шагов. Рассмотрим каждый из них подробнее с наглядной визуализацией обработки. Для этого применим данный оператор шаг за шагом к изображению 3.

Рис. 3: Исходное изображение для обработки оператором Кэнни

1. Размытие изображения для удаления лишнего шума. Для этого можно применить фильтр Гаусса[5]. Функция, используемая этим фильтром, для двумерного случая задается формулой 8.

$$Gaus(x, y, \sigma) = \frac{1}{2\pi\sigma^2} * e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{8}$$

Результат применения фильтра Гаусса к изображению 3 представлен на рисунке 4

Рис. 4: Результат применения фильтра Гаусса

2. Поиск градиентов, для определения границ с максимальным значением градиента. На данном этапе можно использовать оператор Собеля [1], работа которого была описана выше.

   Результат поиска градиентов в размытом изображении представлен на рисунке 5.

Рис. 5: Результат поиска градиентов

3. Подавление не-максимумов, т.е. исключение из границ не локальных максимумов.

   На данном шаге происходит проверка, является ли конкретный пиксель локальным максимумом вдоль направления градиента. Таким образом исключаются ложные границы.

Рис. 6: Изображение после подавления не-максимумов

4. Определение потенциальных границ с помощью двойной пороговой фильтрации.

   Фильтр использует два порога фильтрации:

   - Все пиксели со значением больше верхней границы принимают максимальное значение (достоверная граница).
   - Все пиксели со значением меньше нижней границы подавляются.
   - Все пиксели со значением в диапазоне границ принимают фиксированное среднее значение. Их уточнение происходит на следующем этапе.

   Пример фильтрации с порогами 0,01 и 0,07 представлен на рисунке 7.

5. Трассировка области неоднозначности

   На данном этапе происходит разделение пикселей, получивших промежуточное значение на предыдущем шаге, на границы и фон (увеличение значения и подавление). Пиксель добавляется к границе, если он соприкасается с ней по одному из 8-ми направлений.

   Результат работы оператора Кэнни представлен на рисунке 8.

Рис. 7: Результат двойной пороговой фильтрации

Рис. 8: Результат работы оператора Кэнни

Стоит обратить внимание, что описанные выше методы принимают на вход изображение в серых тонах. То есть нулевым шагом данных методов можно указать преобразование изображения из цветного в черно-белое.

3. Выделение силуэта кисти руки

Помимо классических методов определения границ можно использовать сегментацию по цвету кожи [6]. Данный метод преобразует RGB изображение в бинарное с помощью фильтрации пикселей по цвету, близкому к цвету кожи. Для улучшения работы алгоритма перед фильтрацией изображение переводят в цветовое пространство YCrCb, в котором различные цвета кожи расположены близко друг к другу [7].

Данный метод предусматривает обработку каждого пикселя независимо от других, проверяя его на принадлежность заданному диапазону. Это позволяет ускорить работу алгоритма реализацией параллельной обработки отдельных пикселей.

В большинстве случаев после бинаризации на изображении присутствуют шумы и артефакты, вызванные тем, что на фоновой части изображения находились пиксели, попадающие в ограничения фильтра. Для их устранения можно использовать морфологические операции: "наращивание"и "эрозия"[8]:

Пусть имеется бинарное изображение A и структурный элемент B с началом координат в его центре (рисунок 9.)

- Наращивание. Задается как $A \oplus B = \cup_{b \in B} A_b$. Каждый раз, когда начало координат структурного элемента совмещается с единичным бинарным пикселем, ко всему структурному элементу применяется перенос и последующее логическое сложение с соответствующими пикселями бинарного изображения (рисунок 10).

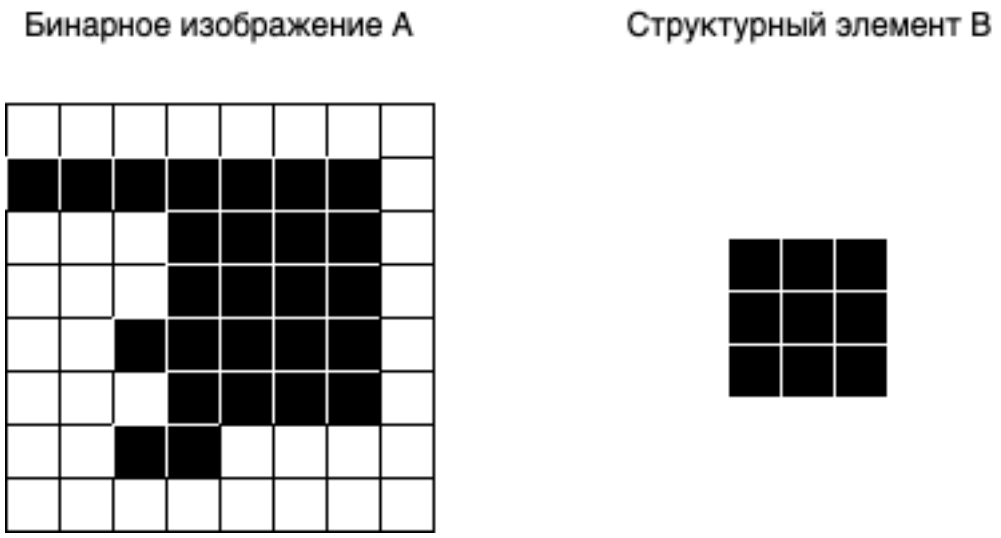

Рис. 9: Бинарное изображение и структурный элемент

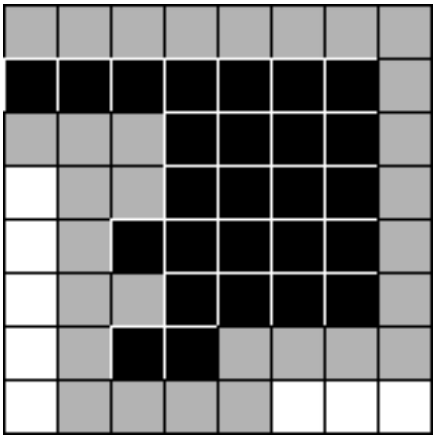

Рис. 10: Наращивание бинарного изображения A структурным элементом B

- Эрозия. Задается как $A \ominus B = \{z \in A | B_z \subseteq A\}$. При выполнении операции эрозии структурный элемент тоже проходит по всем пикселям изображения. Если в некоторой позиции каждый единичный пиксель структурного элемента совпадет с единичным пикселем бинарного изображения, то выполняется логическое сложение центрального пикселя структурного элемента с соответствующим пикселем выходного изображения (рисунок 11).

4. Построение скелета кисти руки

Для ускорения процесса классификации жеста руки можно использовать скелетную модель. Данный тип входных данных в силу своей специфики может упростить вычисление признаков, необходимых классификатору.

Для построения скелета кисти можно использовать метод построения скелета выпуклой

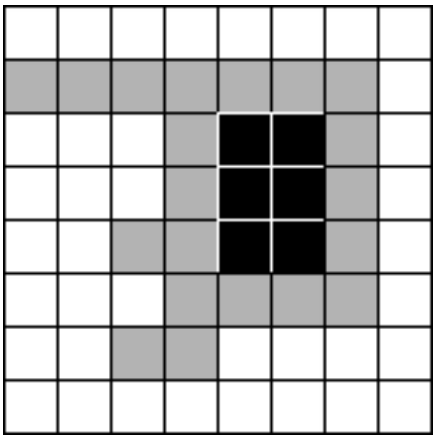

Рис. 11: Эрозия бинарного изображения A структурным элементом B

фигуры [8].

В качестве выпуклой фигуры можно использовать результат работы метода, описанного в разделе 3.

В данном методе предлагается поиск скелета с помощью морфологической операции "сужение". Данная операция применяется до тех пор, пока последующие применение не приведет к очищению изображения.

Проблемой данного метода являются побочные ветви скелета, образованные из-за возможной зашумленности или неточности фигуры. Другим недостатком можно считать отсутствие гарантии обеспечения связного набора пикселей для всего скелета, или обеспечения одинаковой ширины ветвей во всем скелете.

Для решения данных проблем можно обратиться к технологиям машинного обучения. Скелет кисти можно построить на основании ключевых точек, получаемых с помощью нейронной сети [9]. Данная нейронная сеть определяет на изображении 22 ключевых точки, 21 из которых относятся к кисти руки, а 22 отмечает фон. Пример расположения точек представлен на рисунке 12.

Далее для построения скелета необходимо соединить полученные точки в последовательностях, описанной в таблице 1.

Таблица 1: Сравнение алгоритмов выделения источников

Ветвь скелета	Последовательность точек
Большой палец	$0 \to 1 \to 2 \to 3 \to 4$
Указательный палец	$0 \to 5 \to 6 \to 7 \to 8$
Средний палец	$0 \to 9 \to 10 \to 11 \to 12$
Безымянный палец	$0 \to 13 \to 14 \to 15 \to 16$
Мизинец	$0 \to 17 \to 18 \to 19 \to 20$

В ходе экспериментов было выявлена проблема с нахождением ключевых точек. На некоторых изображениях алгоритм либо не находил точки вообще, либо находил не полное их количество.

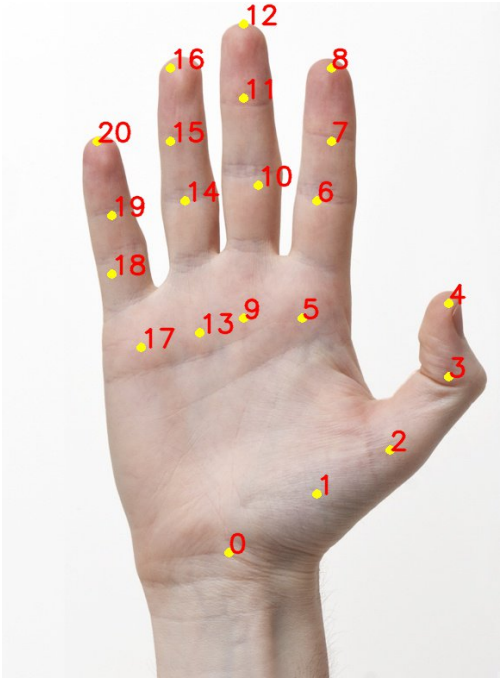

Рис. 12: Ключевые точки кисти руки

5. Методология

Для сравнительного анализа описанных выше методов они были реализованы на языке Python 3 с использованием следующих библиотек:

- Python Imaging Library
- scipy
- numpy
- OpenCV

Каждый алгоритмом был обработан одинаковый набор данных, состоящий из растровых изображений кистей рук тестовая выборка составлялась из нескольких наборов данных, различающиеся разными форматами изображений, их размерами и людьми, чьи руки использовались для получения изображений кистей. Все данные находятся в открытом доступе:

- ASL Alphabet. Image data set for alphabets in the American Sign Language[10]. Данный датасет состоит из 87000 изображений американского дактиля в обучающей выборке и 29 проверочных изображений. Каждое изображение цветное, сохранено в формате JPG, имеет размерность 200x200 пикселей. Для проверки работы алгоритмов были использованы проверочные изображения.
- Hand Gesture of the Colombian sign language. Hand gestures, recognizing the numbers from 0 to 5 and the vowels[11]. Данный датасет состоит из фотографий жестов, изображающий гласные буквы колумбийского языка и цифры от 0 до 5. Приведены снимки как

мужских, так и женских рук. Каждый жест имеет 3 разных фото с разных ракурсов. Каждое изображение цветное, сохранено в формате JPG, имеет размерность 4608 x 2592 пикселей. Для проверки работы алгоритмов были отобраны случайные изображения, по одному на каждый жест.

- ASL Fingerspelling Images (RGB & Depth) [12]. Данная выборка состоит из изображений американского дактиля. В выборке участвуют 5 разных людей, у каждого человека на каждый символ приходится более 1000 изображений. Все изображения цветные, формата PNG, имеют различную размерность. Для проверки работы алгоритмов были отобраны случайным образом по одному изображению для каждой буквы, то есть в итоговой выборке участвовали снимки разных людей под разными ракурсами.

- sign language between 0 9[13]. Данная выборка состоит из изображений жестов, обозначающих цифры от 0 до 10. Все изображения цветные, формата JPG, размерности 300x300. Данные разделены на обучающие, где на каждую цифру приходится более 100 различных изображений, и проверочные, где на каждую цифру представлено одно изображение. Для проверки работы алгоритмов были выбраны все изображения из проверочной выборки.

Были проведены замеры времени обработки каждого изображения для получения статистики по минимальному, максимальному и среднему времени работы алгоритма.

Эксперимент проводился на ноутбуке Lenovo ThinkPad E580 со следующими конфигурациями:

- Операционная система Linux Mint 19 Cinnamon
- Четырех ядерный процессор Intel© Core™ i5 с тактовой частотой 1.60 ГГц
- 8 Гб оперативной памяти

6. Результаты

Целью экспериментов было определение наиболее оптимального метода преобразования исходного изображения с целью упрощения процесса классификации. Для этого необходимо сравнить визуально результаты работы перечисленных выше алгоритмов и время их работы. Результаты экспериментов представлены ниже, отдельно для каждого набора данных:

- Результаты для ASL Alphabet представлены на рисунке A.13 и в таблице A.2
- Результаты для Hand Gesture of the Colombian sign language представлены на рисунке A.14 и в таблице A.3
- Результаты для ASL Fingerspelling Images представлены на рисунке A.15 и в таблице A.4
- Результаты для sign language between 0 9 представлены на рисунке A.16 и в таблице A.5

В результате экспериментов установлено, что среди методов выделения контура по качеству работы лидирует оператор Кэнни. Учитывая небольшую разницу во времени их работы, можно отбросить из рассмотрения все остальные операторы.

Простое выделение силуэта показало наилучшие временные результаты. Так же при правильной предварительной настройке метода можно добиться удовлетворительной четкости выделения. Тем не менее, предварительная настройка является главной проблемой этого алгоритма.

Морфологическое построение скелета показало плохой результат. Как говорилось выше, в результате получаются побочные ветви, а так же скелет получается неполносвязным. Данные недостатки не позволят сильно упросить работу классификатора в силу зашумленности итоговых данных.

Алгоритм построения скелета по ключевым точкам не справился со своей задачей на большинстве результатов. Так же для любого типа данных он работает за одно и тоже время. Это одновременно и хорошо (результаты Hand Gesture of the Colombian sign language) и плохо (остальные результаты).

7. Заключение

В результате данной работы были исследованы возможные методы предобработки изображения в задачах классификации жестовых символов. Были проведены эксперименты с целью определения наиболее оптимального по скорости работы и качеству выделения основных признаков метода.

В результате сравнительного анализа можно выделить два метода:

- Выделение силуэта
- Построение скелета по ключевым точкам

Первый метод показал наилучшие результаты по скорости работы алгоритма, кроме тестов на широкоформатных изображениях. В дальнейшем можно предложить улучшение путем упрощения первичной конфигурации цвета кожи и оптимизации работы на больших изображениях.

Второй метод, несмотря на неудачные результаты тестов, расходящимся с результатами авторов[9], можно попробовать оптимизировать по качеству через переобучение модели. По скорости данный алгоритм можно оптимизировать путем реализации его на другом языке программирования.

Список литературы

[1] Irwin Sobel. An isotropic 3x3 image gradient operator. *Presentation at Stanford A.I. Project 1968*, 02 2014.

[2] Judith M. S. Prewitt. Object enhancement and extraction picture processing and psychopictorics. *Academic*, pages 75–149, 1970.

[3] Lawrence Roberts. *Machine Perception of Three-Dimensional Solids*. 01 1963.

[4] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8:679 – 698, 12 1986.

[5] L.G. Shapiro and G.C. Stockman. *Computer Vision*. Prentice-Hall, Upper Saddle River, NJ, 2001.

[6] S.L. Phung, Abdesselam Bouzerdoum, and Douglas Chai. Skin segmentation using color and edge information. pages 525 – 528 vol.1, 08 2003.

[7] Joshi Siddharth and Srivastava Gaurav. Face detection. *E368: Digital Image Processing*, pages 101 – 112, 2003.

[8] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Pearson Prentice Hall, third edition, 2008.

[9] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. 04 2017.

[10] ASL Alphabet. Image data set for alphabets in the American Sign Language [Электронный ресурс]. https://www.kaggle.com/grassknoted/asl-alphabet.
[11] Hand Gesture of the Colombian sign language. Hand gestures, recognizing the numbers from 0 to 5 and the vowels. [Электронный ресурс]. https://www.kaggle.com/evernext10/hand-gesture-of-the-colombian-sign-language.
[12] ASL Fingerspelling Images (RGB & Depth). [Электронный ресурс]. https://www.kaggle.com/mrgeislinger/asl-rgb-depth-fingerspelling-spelling-it-out.
[13] sign language between 0 9. [Электронный ресурс]. https://www.kaggle.com/beyzance96/sign-language-between-0-9.

ПриложениеA. Результаты эксперимента

Таблица A.2: Время работы алгоритмов (в секундах) на наборе данных ASL Alphabet

Название алгоритма	Минимальное время	Максимальное время	Среднее время
Оператор Кэнни	0.1783	0.4642	0.2553
Оператор Робертса	0.0687	0.1252	0.0852
Оператор Прюитт	0.1175	0.2211	0.1424
Оператор Собеля	0.1177	0.3112	0.1527
Выделение силуэта	0.0668	0.2101	0.0901
Морфологическое построение скелета	0.2084	1.2112	0.3068
Построение скелета по ключевым точкам	1.408	3.4571	2.042

Таблица A.3: Время работы алгоритмов (в секундах) на наборе данных Hand Gesture of the Colombian sign language

Название алгоритма	Минимальное время	Максимальное время	Среднее время
Оператор Кэнни	53.5126	72.6076	62.971
Оператор Робертса	22.2837	54.2706	25.8842
Оператор Прюитт	38.7491	99.2961	46.5944
Оператор Собеля	38.8739	104.1867	46.9016
Выделение силуэта	22.3001	32.8930	23.5992
Морфологическое построение скелета	65.3335	88.3565	69.0014
Построение скелета по ключевым точкам	3.0844	4.2156	3.2775

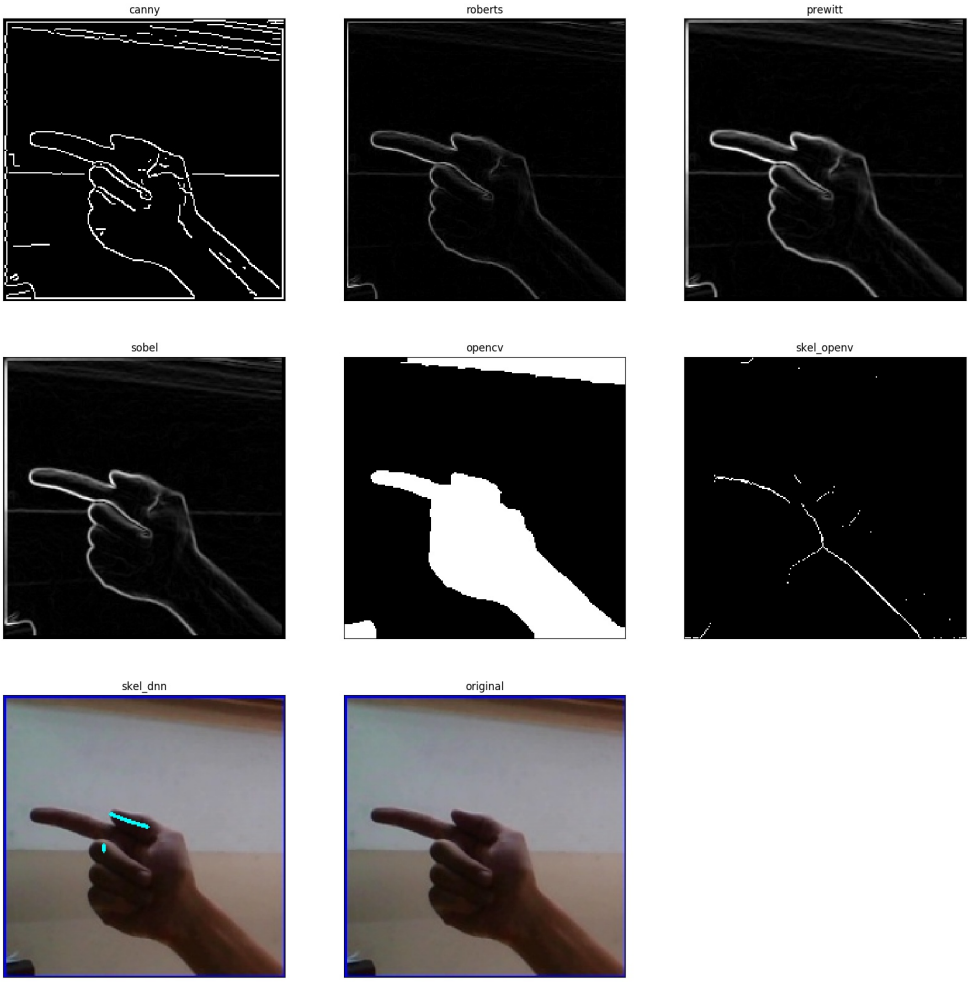

Рис. A.13: Результат работы алгоритмов (в секундах) на наборе данных ASL Alphabet

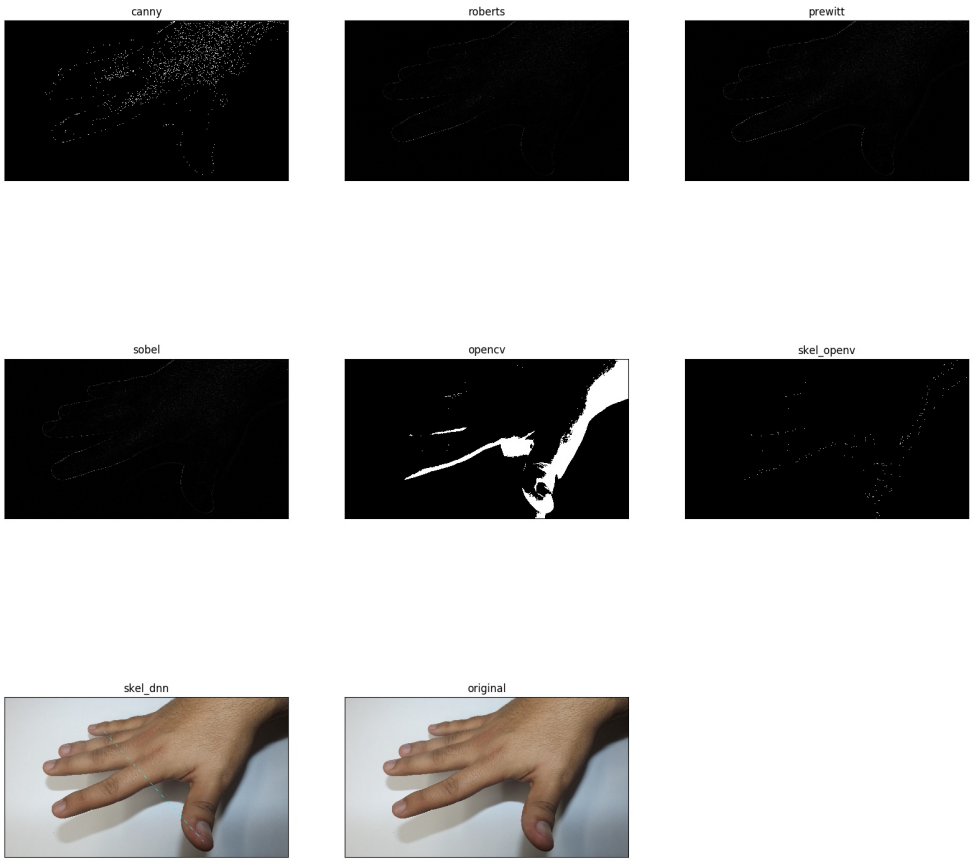

Рис. A.14: Результат работы алгоритмов (в секундах) на наборе данных Hand Gesture of the Colombian sign language

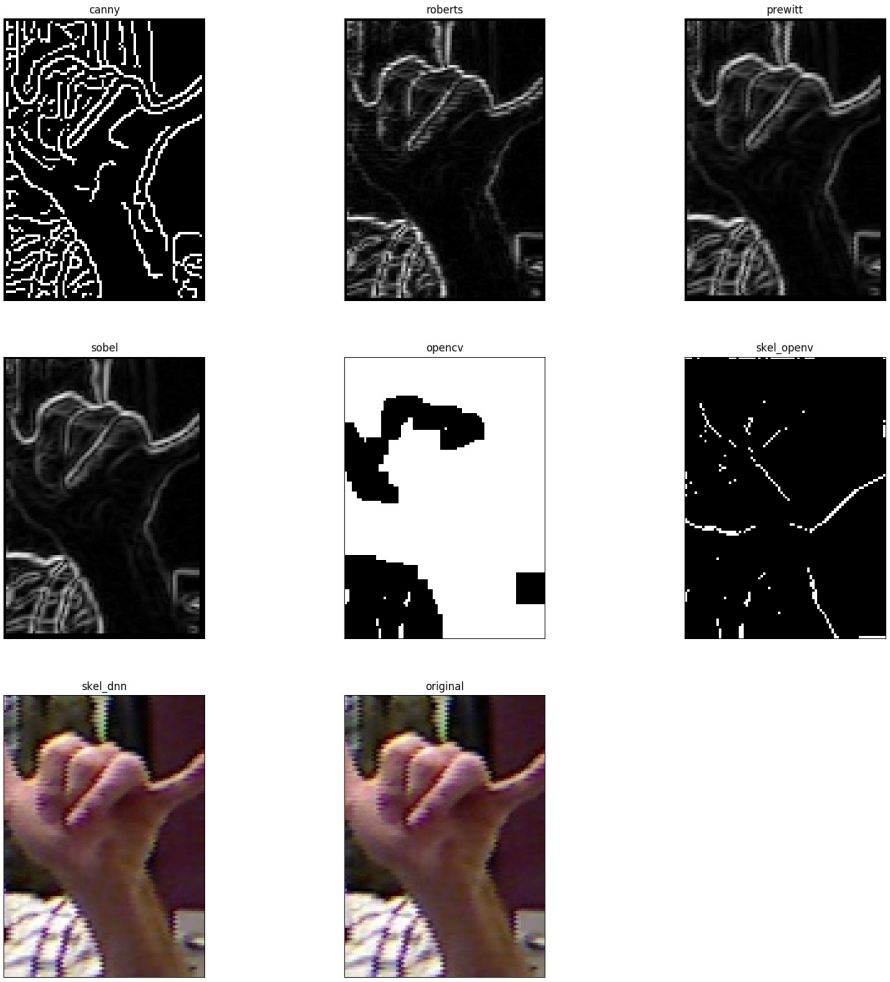

Рис. A.15: Результат работы алгоритмов (в секундах) на наборе данных ASL Fingerspelling Images

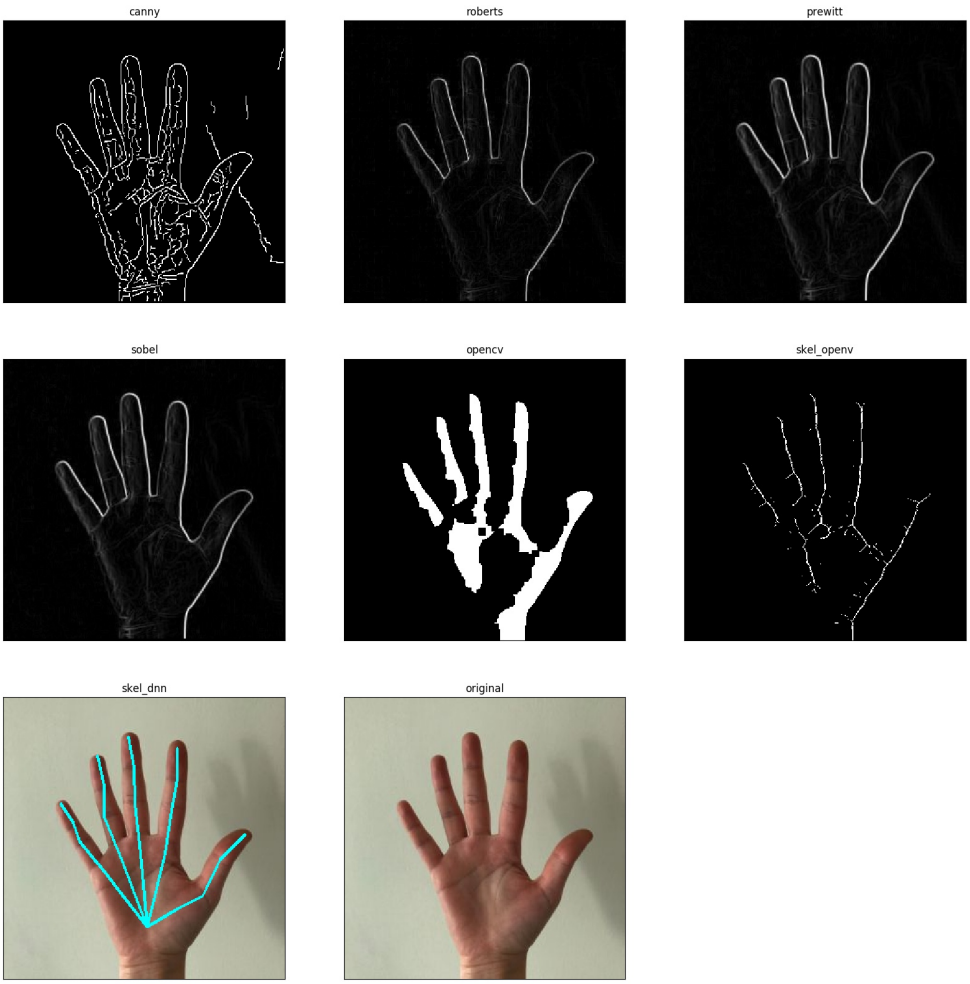

Рис. A.16: Результат работы алгоритмов (в секундах) на наборе данных sign language between 0 9

Таблица А.4: Время работы алгоритмов (в секундах) на наборе данных ASL Fingerspelling Images

Название алгоритма	Минимальное время	Максимальное время	Среднее время
Оператор Кэнни	0.0193	0.0816	0.0545
Оператор Робертса	0.0111	0.0476	0.0286
Оператор Прюитт	0.0179	0.0864	0.0495
Оператор Собеля	0.0217	0.0847	0.0469
Выделение силуэта	0.0114	0.0506	0.0277
Морфологическое построение скелета	0.0343	0.1852	0.0884
Построение скелета по ключевым точкам	0.6251	3.3092	1.5665

Таблица А.5: Время работы алгоритмов (в секундах) на наборе данных sign language between 0 9

Название алгоритма	Минимальное время	Максимальное время	Среднее время
Оператор Кэнни	0.333	0.7686	0.4708
Оператор Робертса	0.1566	0.246	0.1778
Оператор Прюитт	0.271	0.3751	0.3027
Оператор Собеля	0.2718	0.655	0.3295
Выделение силуэта	0.1486	0.4709	0.2085
Морфологическое построение скелета	0.4471	1.637	0.6518
Построение скелета по ключевым точкам	1.4346	3.2976	2.0723