Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Национальный исследовательский университет «МЭИ»

|  |  |  |  |
| --- | --- | --- | --- |
| Институт: | ИВТИ | Кафедра: | ВМСС |
| Направление подготовки: | | 09.03.01 Информатика и вычислительная техника | |

**ОТЧЕТ по практике**

|  |  |
| --- | --- |
| **Наименование практики:** | Производственная практика: *научно-исследовательская работа* |

**СТУДЕНТ**

|  |  |
| --- | --- |
|  | / Сорокин А.Ю. / |
| *(подпись )* | (*Фамилия и инициалы*) |

|  |  |
| --- | --- |
| Группа | А-08м-20 |
|  | *(номер учебной группы)* |

**ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ ПО ПРАКТИКЕ**

|  |
| --- |
| отлично |
| *(отлично, хорошо, удовлетворительно, неудовлетворительно)* |

|  |  |
| --- | --- |
|  | / /Абросимов Д.И. |
| *(подпись )* | (*Фамилия и инициалы члена комиссии*) |

|  |  |
| --- | --- |
|  | / / |
| *(подпись )* | (*Фамилия и инициалы члена комиссии*) |

**Москва**

**2021**

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Национальный исследовательский университет «МЭИ»

|  |  |  |  |
| --- | --- | --- | --- |
| Институт: | *ИВТИ* | Кафедра: | *ВМСС* |
| Направление подготовки/специальность: | | *09.03.01 Информатика и вычислительная техника* | |

**ЗАДАНИЕ НА ПРАКТИКУ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование практики:** | Производственная практика:  научно-исследовательская работа | |  |
| **Студент:** | *Сорокин Александр Юрьевич* | |  |
|  | *(Фамилия, имя, отчество (при наличии) полностью)* | |  |
| **Группа:** |  | |  |
|  | *(номер учебной группы)* | |  |
| **Место прохождения практики:** | *Институт информационных и вычислительных технологий* | |  |
|  | *(наименование предприятия, организации, учреждения, подразделения МЭИ в соответствии с приказом о направлении на практику)* | |  |
| **Сроки практики:** | *10.02.21 по 14.06.21 г.* | |  |
|  | *(в соответствии с приказом о направлении на практику)* | |  |
| **Содержание задания:** | |  | |
| 1. Выбрать научного руководителя для выполнения НИР и других работ включая выполнение научных исследований и выпускной квалификационной работы. | | | |
| 1. Выполнить обзорную работу по выбранному направления научно-исследовательской работы. В данном обзоре рассмотреть современное состояние данного направления, перспективы развития, а также ознакомиться с научными публикациями в выбранной области. Обосновать актуальность выбранной тематики. | | | |
| 1. Отчитаться руководителю НИР о проделанной работе. | | | |
|  | | | |
| *(вопросы, подлежащие изучению в соответствии с планируемыми результатами обучения,*  *заполняются руководителем практики от МЭИ)* | | | |

**По результатам прохождения практики студент оформляет отчет по установленной форме.**

|  |  |  |
| --- | --- | --- |
| Руководитель практики |  | / Абросимов Л.И / |
|  | *(подпись)* | (*Фамилия и инициалы*) |
| Студент |  | / / |
|  | *(подпись)* | (*Фамилия и инициалы*) |

ГРАФИК прохождения НИР

|  |  |  |
| --- | --- | --- |
| Номер  п/п | Перечень работ в соответствии с заданием | Отметка о выполнении работы  (выполнено /не выполнено) |
| 1 | Рассмотреть и проанализировать основные типы сканеров отпечатков. | выполнено |
| 2 | Для исследования и практического применения отобрать сканеры для ПК | выполнено |
| 3 | Реализовать стенд на базе выбранного сканера (ZK6500 фирмы ZKTeco) | выполнено |
| 4 | Рассмотреть и проанализировать методы обработки отпечатков пальцев | выполнено |
| 5 | Проанализировать этапы выполнения выбранного метода обработки отпечатка пальца. | выполнено |
| 6 | Реализация метода обработки отпечатков пальцев на одном из языков программирования | выполнено |
| 7 | Провести анализ полученных результатов | выполнено |
| 8 | Сделать выводы | выполнено |

|  |  |  |
| --- | --- | --- |
| Руководитель практики (от МЭИ) |  | / Абросимов Л.И. / |
|  | *(подпись )* | (*Фамилия и инициалы*) |

Оглавление

[Введение 7](#_Toc74475921)

[1. Сканеры отпечатков пальцев 7](#_Toc74475922)

[2. Принцип работы сканеров 8](#_Toc74475923)

[3. Основные типы сканеров отпечатков пальцев 8](#_Toc74475924)

[3.1 Оптический сканер 8](#_Toc74475925)

[3.2 Полупроводниковый сканер 9](#_Toc74475926)

[4. Характеристики сканеров 10](#_Toc74475928)

[5. Принципы распознавания отпечатков пальцев 12](#_Toc74475929)

[5.1 Корреляционное сравнение 12](#_Toc74475930)

[5.2 Сравнение по узору 13](#_Toc74475931)

[5.3 Сравнение по особым точкам 14](#_Toc74475932)

[6. Метод сравнения отпечатков по особым точкам 15](#_Toc74475933)

[6.1 Бинаризация полученного изображения 15](#_Toc74475934)

[6.1.1 Глобальный метод 15](#_Toc74475935)

[6.1.2 Метод Беркли 16](#_Toc74475936)

[6.1.3 Метод Бернсена 16](#_Toc74475937)

[6.1.4 Метод Эйквеля 17](#_Toc74475938)

[6.1.5 Метод Яновица и Брукштейна 17](#_Toc74475939)

[6.2 Скелетизация изображения 18](#_Toc74475940)

[6.2.1 Шаблонная скелетизация 18](#_Toc74475941)

[6.2.2 Волновой метод 19](#_Toc74475942)

[6.2.3 Алгоритм Зонга-Суня 20](#_Toc74475943)

[6.2.4 Алгоритм Щепина 21](#_Toc74475944)

[6.2.5 Алгоритм утончения областей 22](#_Toc74475945)

[6.3 Выделение особых точек (разветвления и окончания папилярных линий) 23](#_Toc74475946)

[7. Сравнения отпечатков по особым точкам 24](#_Toc74475947)

[8. Методика применения контроля отпечатков пальцев на ПК 25](#_Toc74475948)

[9. Минимальная конфигурация ПК для установки системы контроля доступа к помощью сканера отпечатков 26](#_Toc74475949)

[10. Реализация метода сравнения отпечатков по особым точкам 29](#_Toc74475950)

[10.1 Бинаризация изображения глобальным методом 29](#_Toc74475951)

[10.2 Алгоритм скелетизации (Метод Зонго-Суня) 32](#_Toc74475952)

[10.3 Алгоритм обнаружения особых точек 35](#_Toc74475953)

[11. Альтернативные методы обработки изображения 38](#_Toc74475954)

[Заключение 40](#_Toc74475955)

[Литература 41](#_Toc74475956)

[Приложение А Сканеры для исследования и применения 43](#_Toc74475959)

[Приложение Б Сканер для исследования в данной работе 53](#_Toc74475979)

[Приложение В Программная реализация метода сканирования отпечатка по особым точкам 56](#_Toc74475982)

**ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

КМОП - комплементарная структура металл-оксид-полупроводник

ПК - персональный компьютер

ПО – программное обеспечение

ПЗС- прибор с зарядовой связью

# Введение

Постановка задачи: В данной работе необходимо разработать аппаратно-программную систему по защите персональных компьютеров предприятия от несанкционированного доступа. Система основывается на персональных данных пользователей имеющим к ним доступ.

Вместе с внедрением персональных компьютеров на предприятиях с начала 80-х годов прошлого столетия возникла необходимость защищать хранящуюся на них информацию. Одним из способов является аутентификация личности с помощью логинов и паролей. Этого недостаточно, т.к. возможна передача логина и пароля посторонним лицам. По мере развития аппаратных средств, таких как видеокамеры и сканеры отпечатков пальцев, появилась возможность защищать данные на ПК с помощью персональных данных пользователя, а именно, отпечатков пальцев и видеоизображения. На современном этапе, системы защиты информации находящейся на ПК способны распознать личность. Если злоумышленник применит специальные маски для лица и накладные отпечатки пальцев, распознавание не всегда возможно. Противостоять этому возможно с помощью усовершенствования программного обеспечения и аппаратных средств. Разработки в этих направлениях ведутся постоянно.

Аппаратная часть основывается на применении двух типов устройств для идентификации:

* сканеры отпечатков пальцев;
* видеокамеры.

Рассмотрим принципы действия, классификацию и основные характеристики этих устройств.

# Сканеры отпечатков пальцев

Дактилоскопия - это способ идентификации [человека](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D0%BA) по [следам пальцев](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BB%D0%B5%D1%86) рук (в том числе ладоней рук), основанный на неповторимости рисунка [кожи](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B6%D0%B0). Широко применяется в [криминалистике](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BC%D0%B8%D0%BD%D0%B0%D0%BB%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0). В среде ученых существует мнение, что  полное совпадение папиллярных рисунков может случиться не чаще, чем в одном случае из 64 миллиардов. Однако, подтвердить эту теорию на практике, получив отпечатки пальцы у всех проживающих на планете людей, не представляется возможным. Тем не менее, данный способ широко используется, как один из способов для идентификации личности, в том числе, в судебной практике.

1. **Принцип работы сканеров**

а) сканируются отпечатки пальцев пользователей (возможно с последующим шифрованием);

б) заносятся в базу данных пользователей для данного ПК, с присвоением уровня доступа;

в) при попытке пользователя получить доступ к ПК, программное обеспечение осуществляет проверку соответствия отпечатка пользователя отпечаткам, которые хранятся в его базе данных;

г) если соответсвующий отпечаток в базе данных сканера найден, пользователь получает доступ к ПК, если нет, осуществляется блокировка доступа (блокируется экран, отдельные папки и т.д.).

**3. Основные типы сканеров отпечатков пальцев**

Существует три основных группы сканеров отпечатков пальцев.

### 3.1 Оптический сканер

Для получения изображения отпечатка пальца используется световое излучение. Существует несколько способов реализации оптического метода.

* + 1. Отражение светового излучения

Используется явление полного внутреннего отражения. Излученный свет проходит сквозь стекло сканера и отражается от приложенного пальца пользователя. Попавший на границу сред световой поток раздваивается – одна часть излучения отражается, другая – переходит в иную среду. От угла отражения зависит процент отраженного света, при определенных углах отражается все излучение – полное внутреннее отражение. Отраженное излучение попадает на ПЗС или КМОП-матрицу, где фиксируется изображение.

Недостатки метода:  
- возможно создание муляжей;  
- чувствительность к загрязнениям.

* + 1. Фиксация остаточного излучения

Для получения скана отпечатка применяется оптоволоконная матрица с волноводами, к каждому из которых подключен фотодатчик. Последние улавливают остаточное световое излучение, проходящее через палец в точке его соприкосновения с матрицей.

Достоинства:  
- высокая надежность считывания,  
- устойчивость к обману.  
Недостатки:

- сложность реализации.

* + 1. Бесконтактный

Палец пользователя не контактирует со сканером. Его облучают со всех сторон, а линза, через которую проходит отраженный свет, проецирует изображение на КМОП-матрицу.

Общий недостаток реализации оптических методов - большие габариты.

### Полупроводниковый сканер

Для получения отпечатка пальца используются свойства полупроводниковых материалов, которые изменяют свои параметры в местах контакта гребней узора с поверхностью матрицы. Различают следующие полупроводниковые сканеры отпечатков пальцев:

3.2.1 Сканеры Давления

Сканеры реагируют на повышение давления в месте касания гребней и его отсутствие напротив впадин.

Недостатки:

- низкая чувствительность;  
- возможно создание муляжей;

- сканеры хрупкие и относительно быстро выходят из строя при избыточном давлении.

3.2.2 Сканеры термические

Основаны на изменении сопротивления в зависимости от температуры. В таких устройствах используются датчики, которые состоят из пироэлектрических элементов, позволяющих фиксировать разницу температуры и преобразовывать ее в напряжение.

Достоинства:

- высокая защита от муляжей.

Недостатки:

- быстрое выравнивание температур (по истечении короткого времени (менее одной десятой доли секунды) изображение исчезает, поскольку палец и датчик имеют одинаковую температуру).

3.2.2 Сканеры емкостные

КМОП датчики пропускают через поверхность кожи пальца миниатюрные токи. Поверхность пальца – одна пластина конденсатора, матрица датчика – вторая. КМОП датчики реагируют на разницу в проводимости кожи и воздуха, создается “емкостная ” картинка отпечатка. Именно такой тип сканеров используется в смартфонах и во многих моделях сотовых телефонов.

3.3 Ультразвуковой сканер

### Поверхность пальца сканируется звуковыми волнами частотой свыше 20 кГц. Расстояния между гребнями и впадинами определяются благодаря параметрам отраженного от них ультразвука – эха. От гребня звук идет быстрее, чем от впадины – так определяется рельефность поверхности кожи.

Достоинства:

- очень высокое качество снимков.

Недостатки:

- высокая стоимость;

- относительно большое время сканирования.

**4. Характеристики сканеров**

На сегодняшний день многие фирмы производят сканеры отпечатков, как устройства готовые к применению. Их характеристики зависят от типа сканера, устройства для которого применяется сканер, прилагаемого программного обеспечения.

Основные характеристики сканеров:

* тип сенсора;
* разрешение отпечатка;
* плотность сканирования;
* количество сохранённых отпечатков;
* количество уровней доступа пользователей;
* ложные положительные срабатывания;
* ложные отрицательные срабатывания;
* время сравнения отпечатка;
* устойчивость к электростатическому разряду;
* интерфейсы;
* напряжение питания;
* ток потребления;
* габариты считывателя.

Основное внимание нужно уделять двум характеристикам:

* ложные положительные срабатывания ( оценивает вероятность ложной идентификации клиента, отсутствующего в базе данных);
* ложные отрицательные срабатывания (вероятность отказа в распознавании зарегистрированного пользователя).

По этим характеристикам оценивают качество работы сканера, как аппаратной его части, так и программной. Чем меньше их абсолютное значение, тем лучше.

К каждому сканеру прилагается программное обеспечение. Помимо основных, его функции могут существенно расширены, оказывать дополнительные услуги по подключению к различным сервисам, предоставлять услуги дополнительной защиты исходных данных пользователей.

В дальнейшем, в рамках данной работы, будут рассматриваться сканеры для персональных компьютеров.

В приложении А представлены сканеры для ПК с высоким рейтингом, бытовые и для профессионального применения, с характеристиками и кратким описанием. Их характеристики и функциональные возможности существенно отличаются в зависимости от области применения сканера.

**5. Принципы распознавания отпечатков пальцев**

Дактелоскопия, как способ идентификации [человека](https://ru.wikipedia.org/wiki/%D0%A7%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D0%BA) по [следам пальцев](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BB%D0%B5%D1%86) рук, предполагает пусть и очень маленькую, но вероятность совпадения отпечатков пальцев у двух разных людей. Однако, в ограниченном сообществе людей, таком как сотрудники предприятия, или группа лиц имеющая доступ к ПК, такую вероятность легко исключить с помощью соответствующих программ и простым визуальным сравнением. Таким образом, все дальнейшие рассуждения будут основываться на положении, что отпечатки пальцев поступающие для идентификации у разных людей будут отличаться.

Кожа человека на ладонях и пальцах имеет объемную структуру, которая образована как из отдельных бугорков, так и из сросшихся между собой бугорков. Между бугорками остаются пустые бороздки. После сканирования, на снимке, бугорки видны как темные пятна или линии разной конфигурации, а пустые бороздки между ними как светлые (Рисунок 5.1).



Рисунок 5.1 Отпечаток пальца человека

Рисунок, который они собой представляют уникален. На сегодняшний день существуют несколько методов распознавания отпечатков [1,2,3,4,5].

## 5.1 Корреляционное сравнение

Полученное изображение отпечатка накладывают на эталонное изображение и подсчитывается корреляция (по уровню интенсивности) между соответствующими пикселями. По значению коэффициента корреляции принимается решение об идентичности отпечатков.

Недостатки:

- необходимо осуществлять поэтапные сдвиги изображения отпечатка относительно эталонного в различных направлениях, в том числе вращать вокруг центра, чтобы максимально совместить изображения;

- при поиске совпадений со многими эталонными изображениями количество операций возрастает многократно.

Из-за сложности и длительности работы данного алгоритма он практически не используется.

**5.2 Сравнение по узору**

Полученное со сканера изображение отпечатка пальца, разбивается на ячейки (размер ячеек зависит от требуемой точности) Рисунок 5.2.

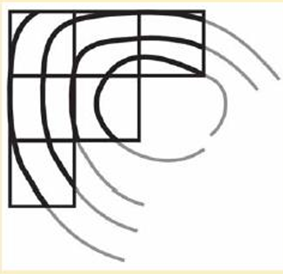


Рисунок 5.2 Изображение отпечатка разбитого на ячейки

Расположение линий в каждой ячейке описывается параметрами некоторой синусоидальной волны. Данные волн (длина волны, направление волны) используются для идентификации. Сравнение отпечатков осуществляется по содержимому ячеек.

Достоинства:

- для данного метода подходят отпечатки с относительно низким качеством сканирования.

Недостатки:

- сложная реализация и сложный, объемный математический аппарат.

Вследствие указанных недостатков данный метод применяется редко.

**5.3 Сравнение по особым точкам**

Отпечаток каждого человека уникален. Однако, при более близком рассмотрении, становится видно, что рисунок состоит из повторяющихся по структуре фрагментов (Рисунок 5.3), которые могут присутствовать на отпечатках других людей. Это окончания папилярных линий и места, где эти линии разветвляются.

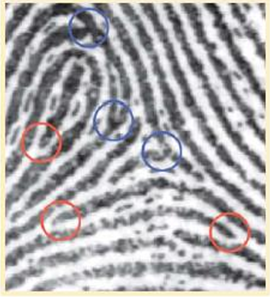


Рисунок 5.3 Разветвления и окончания папилярных линий

По изображению отпечатка пальца формируется шаблон (Рисунок 5.4), на котором выделены конечные точки и точки ветвления. При сравнении, карта этих точек сравнивается с шаблоном эталона и по количеству совпавших точек принимается решение по идентичности отпечатков.



Рисунок 5.4 Шаблон отпечатка пальца

Сравнение точек осуществляется корреляционными методами, из предположения, что положение точек соответствуют друг другу.

Достоинсва:

- простая (относительно) реализации и высокая скорость работы.

Недостатки:

- для реализации необходимо, чтобы снимок отпечатка содержал не менее 500 пикселей. Большинство современных сканеров обеспечивают такое разрешение и больше.

Благодаря своим достоинствам и возможностям технической реализации, данный метод чаще всего применяют на практике. Рассмотрим возможности применения этого метода для создания системы по защите персональных компьютеров предприятия.

**6. Метод сравнения отпечатков по особым точкам**

Для эталонного и поступающего на контроль отпечатка пользователя ПК создаются шаблоны. После сравнения шаблонов принимается решение о совпадении отпечатков. Для создания шаблона изображение отпечатка проходит следующие этапы обработки:

- бинаризация полученного изображения;

- скелетизация изображения;

- выделение особых точек.

Рассмотрим каждый из этапов подробнее.

* 1. Бинаризация полученного изображения

Бинаризация - это преобразование, которое заключается в превращении изображения в двухцветное черно-белое. Главным параметром такого преобразования является порог – значение, которое будет критерием проверки интенсивности точки изображения. Главная цель процесса бинаризации - это уменьшение количества информации, с которой приходится работать до двух состояний.

Существуют различные методы бинаризации и их много, которые можно условно разделить на две группы – глобальные и локальные. Рассмотрим некоторые из них.

* + 1. Глобальный метод

Для глобального метода выбирается порог преобразования, который остается неизменным в процессе обработки изображения.

Достоинства:

- простая реализация и высокая скорость выполнения.

Недостатки:

- могут быть утрачены существенные отдельные детали изображения, которые оказались ниже порога.

* + 1. Метод Беркли

Метод Беркли - локальный метод бинаризации. Суть метода заключается в том, что последовательно определяется яркость пикселей изображения, которая вычисляется по некоторому закону используя яркость квадратной ячейки из четырех пикселей.

Достоинства:

- простота реализации и высокая скорость выполнения. Метод хорошо работает с неоднородным фоном;

- обработка изображения осуществляется один раз.

Недостатки:

- плохая чувствительность к низкоконтрастным деталям изображения (Например, на темно-сером фоне черные пятна или на белом фоне бледно серые пятна).

### 6.1.3 Метод Бернсена

Используется для схематических и картографических изображений. Для каждого пикселя (x;y) выбирается порог яркости

B(x;y)=(Bмин-Bмакс)/2,

где Bмин,Bмакс – соответственно, самый низкий и самый высокий уровень яркости пикселей из квадратной окрестности пикселя (x;y).

Если уровень контраста (разность самого высокого уровня и самого низкого уровней) превышает некоторый порог, то пиксель считается либо белым, либо черным. Для всего изображения этот порог контраста является константой и должен подбираться интерактивно.

Достоинства:

- высокая скорость обработки, даже с учетом постпроцессинга.

Недостатки:

- после обработки монотонных областей яркости формируются сильные паразитные помехи, в некоторых случаях приводит к появлению ложных черных пятен.Можно скомпенсировать с помощью дополнительной обработки – постпроцессинга.

### 6.1.4 Метод Эйквеля

Применяется для обработки четких и контрастных изображений. Изображение обрабатывается с помощью двух окон: маленького и большого. Обычно форма окон принимается квадратной. Оба окна последовательно слева направо сверху вниз накладываются на изображение с шагом равным стороне маленького окна S. Для большогоокна рассчитывается порог B так, чтобы поделить пиксели на два кластера. Если математические ожидания уровня яркости в двух кластерах имеют разницу, превышающую некоторый заданный пользователем уровень /μ1-μ2/≥l, то все пиксели внутри маленького окна бинаризуются в соответствии с порогом T. В противном случае, яркость пикселей из маленького окна S заменяется некоторым близким значением.

Достоинства:

- метод один из самых производительных.

### 6.1.5 Метод Яновица и Брукштейна

Применяется для обработки сканированных бумажных картографических изображений. В качестве пороговой поверхности бинаризации используется поверхность потенциалов, строящаяся на основе локальной максимизации градиента яркости. Значение градиента яркости часто рассчитывается с помощью контурного оператора Собеля или Кэнни. Изображение фильтруется с целью получения контурных линий толщины в 1 пиксель, а затем усредняющим фильтром 3×3 и потенциальная поверхность, теперь, строится по итерационной интерполирующей схеме. Расчет поверхности идет в порядке, начиная от контурных пикселей.

Достоинства:

- метод не дает паразитного шума на бинарном растре. Низкая ошибка разрыва линейных объектов.

Недостатки:

- низкая производительность.

6.2 Скелетизация изображения

Скелетизация изображения производится с целью утончения линий изображения отпечатка. Утончение производится до тех пор, пока линии не будут шириной 1 пиксель.

На текущий момент времени существует и описаны алгоритмы скелетизации изображения [6]. Ниже перечислены некоторые из них:

- шаблонная скелетизация;

- волновой метод;

- алгоритм Зонга-Суня;

- алгоритм [Щепина](http://www.mathnet.ru/php/person.phtml?personid=8985&option_lang=);

- алгоритм утончения областей.

6.2.1 Шаблонная скелетизация

Метод поочерёдно просматривает все точки бинарного изображения f(x,y) E {0,1} , где значение 1 соответствует бинарной линии узора, и анализирует окрестности только закрашенных (информативных) пикселей. Анализ окрестности представлен следующей формулой:

K=,

где K – величина оценки окрестности, используемая как индекс; (x, y) – координаты исследуемого пикселя; f(x, y) – значение пикселя на бинарном изображении; h(i, j) – маска, элементы которой задают вес точек, представленная в виде:

h=

Индекс K – число в интервале от 0 до 255 включительно. Оно рассматривается алгоритмом как индекс ячейки в таблице правил, заранее заданной в процессе проектирования.

Таблица представляет собой закодированный набор правил и операций по обработке окрестности. Фактически она организовывает связь между конкретным шаблоном и действием, которое необходимо сделать с окрестностью исследуемой точки при вычислении cоответствующего индекса K, который определяет номер элемента в таблице. Каждый элемент таблицы содержит код операции, нулевой код соответствует отсутствию операции (см. таблицу Рисунок 6.1).



Рисунок 6.1 Закодированный набор правил

После обработки всех пикселей, метод итерационно повторяется. Если после очередной итерации ни один из пикселей не был закрашен или перемещён, алгоритм завершает свою работу.

* + 1. Волновой метод

Метод заключается в анализе пути прохождения сферической волны по изображению. На каждом этапе анализируется смещение центра масс точек, образующих новый шаг волны, относительно его предыдущих положений.

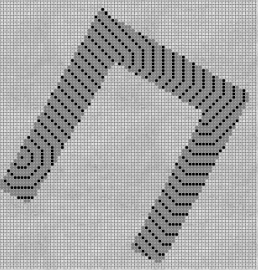


Рисунок 6.2 Распространение волны

Его задачей является векторное представление изображения в виде нагруженного графа – то есть определение концевых точек, точек пересечения (вершины графа), а также линий и дуг, составляющих фигуры (рёбра графа) Рисунок 6.3 .

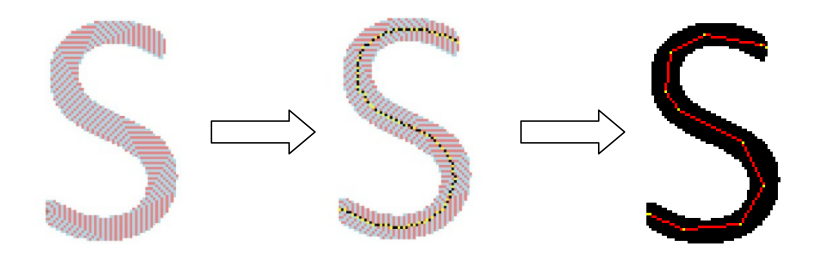


Рисунок 6.3 Основные этапы скелетизации: построение первичного графа и оптимизация графа.

Первичный граф обычно содержит большое количество избыточной информации, поэтому на этапе оптимизации графа лишние точки из него удаляются.

Недостатки:

* метод хорошо подходит для распознавания букв, но не для создания скелета отпечатка пальцев, структура папиллярных линий пальца намного сложнее, распространение волны и её дальнейший анализ проходит с большими погрешностями.
  + 1. Алгоритм Зонга-Суня

Введем следующую матрицу:

|  |  |  |
| --- | --- | --- |
| P9 | P2 | P3 |
| P8 | P1 | P4 |
| P7 | P6 | P5 |

Данная матрица попиксельно накладывается на изображение таким образом, чтобы P1 поочередно совмещался со всеми пикселями изображения (там где это возможно). Стоит уточнить, что белый пиксель на изображении имеет значение 0, а черный 1. Удаление пикселей происходит по описанным в алгоритме правилам:

На первой субитерации пиксель удаляется, если соответствует следующим условиям:

2<=B(P1)<=6

B(Pi)=

A(P1)=1

P2\*P4\*P6=0

P8\*P4\*P6=0

где А(Р1) – количество конфигураций 0/1 в последовательности Р1-Р8 (заканчивая пикселем Р1 ).

Это значит, что вокруг пикселя Р0 существует всего один переход из 0 в 1.

Вторая субитерация аналогична первой, но с отличием:

P2\*P4\*P8=0

P2\*P6\*P8=0

Алгоритм останавливается, когда после очередного прохода не был удален ни один пиксель.

Его отличие от обычного утоньшения в том, что прежде, чем удалить пиксель в процессе эрозии, исследуется, насколько повлияет удаление этого пикселя на соединенность скелета. В случае, если удаление пикселя не создаст разъединенных фрагментов скелета, пиксель удаляется.

Достоинства:

* простота реализации;
* алгоритм производит более точные скелеты, чем обычное утоньшение, хотя и они содержат разъединенные фрагменты.

Недостатки:

- возможно появление ложных ветвей скелетов.

* + 1. Алгоритм [Щепина](http://www.mathnet.ru/php/person.phtml?personid=8985&option_lang=)

Для каждого внешнего и внутреннего контура изображения находятся исходные верхние левые точки. Для очередной точки контура рассматривается конфигурация восьми ее соседей. Точка удаляется, если она не является концевой, и если после ее удаления ее соседи по-прежнему будут образовывать связное множество. После анализа точки и ее соседей и возможного удаления точки осуществляется переход к следующей точке контура таким образом, чтобы остаться на границе изображения. Далее шаг за шагом удаляется один слой точек. Слои удаляются до тех пор, пока не останутся только неудаляемые точки (Рисунок 6.4).

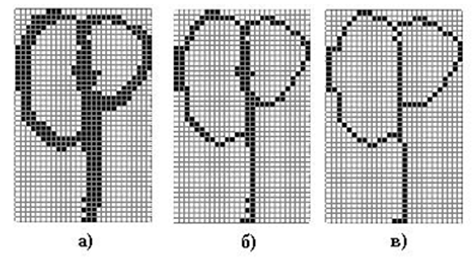


Рисунок 6.4 Скелетизация изображения по методу Щепина

. а) исходное изображение; б) удаление одного слоя; в) удаление второго слоя

Достоинства:

* простота реализации.

Недостатки:

* алгоритм использует много проходов по изображению и может быть относительно медленным.

6.2.5 Алгоритм утончения областей

Данный алгоритм основан на простом анализе окрестности

каждой из его информативных точек. Соседи в окрестности информативного пикселя нумеруются по цепочке от P1 до P9.

|  |  |  |
| --- | --- | --- |
| P9 | P2 | P3 |
| P8 | P1 | P4 |
| P7 | P6 | P5 |

В зависимости от цветов закраски окрестных пикселей, вычисляются два параметра: A(P1) – число переходов от белого пикселя к чёрному в цепочке, а так же B(P1) – общее количество всех чёрных пикселей в окрестности. В зависимости от значений этих параметров, а так же закраски конкретных пикселей в четырёхсвязной области (пиксели P2, P4, P6, P8), принимается одно из двух решений – либо закрасить центральный пиксель в белый цвет, либо оставить его в том же виде.

После того, как все информативные пиксели изображения прошли данную процедуру обработки, алгоритм обхода запускается вновь, но уже на обработанном изображении. Алгоритм считается завершённым тогда, когда после очередной итерации не был закрашен ни один пиксель изображения.

Существуют несколько вариантов модификации данного метода, улучшающие работу основного алгоритма, для полутоновых и цветных изображений.

Результатом работы метода является скелет отпечатка пальца (Рисунок 6.5).



Рисунок 6.5 Отпечаток и его скелет

Достоинства:

* простота реализации и высокая скорость обработки.

Недостатки:

* не устраняются возможные шумы, находящиеся на оригинальном изображении. Эти шумы могут оказать своё влияние при

определении особых точек, а, вследствие чего, и на результат верификации отпечатка.

6.3 Выделение особых точек (разветвления и окончания папилярных линий)

Для выделения особых точек изображение разбивается на блоки 9х9 пикселей и исследуется с последовательным сдвигом на 1 пиксель. Если центральный пиксель белый (нулевой) осуществляют сдвиг на пиксель. Если центральный пиксель черный (ненулевой), ищут чёрные (ненулевых) пиксели, находящихся вокруг центра. Если соседних ненулевых пикселей один - это «окончание», если три - это «раздвоение».

Координаты обнаруженных деталей и их углы ориентации записываются в вектор:

W(p)=[(x1, y1, 1), (x2, y2, 2)...(xp, yp, p)],

где p - число деталей. При регистрации пользователей этот вектор считается эталоном и записывается в базу данных.

**7. Сравнения отпечатков по особым точкам**

Отсканированный отпечаток пользователя проходит все те же этапы обработки, что и эталонный отпечаток:

- бинаризация полученного изображения;

- скелетизация изображения;

- выделение особых точек.

После чего можно приступать к сравнению отпечатков. Отсканированный отпечаток будет отличаться от эталонного поворотом, смещением, изменением масштаба и площадью соприкосновения в зависимости от того, как пользователь прикладывает палец к сканеру.

Определяются параметры аффинных преобразований (угол поворота, масштаб и сдвиг), при которых некоторая деталь из одного вектора соответствует некоторой детали из другого. Эти параметры пытаются применить к другим деталям вектора Рисунок 7.1.

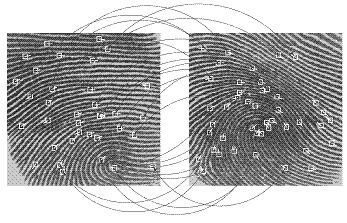


Рисунок 7.1 Сопоставление деталей отпечатков

Оценка соответствия отпечатков выполняется по формуле:

K=(D\*D\*100%)/(p\*q),

где D - количество совпавших деталей, p - количество деталей эталона, q - количество деталей идентифицируемого отпечатка).

В случае если результат превышает 65%, отпечатки считаются идентичными (порог может быть понижен выставлением другого уровня бдительности).

Таким образом, идентификация отпечатков пальцев методом сравнения по особым точкам может быть реализована, при этом на каждом этапе реализации не одним способом.

**8. Методика применения контроля отпечатков пальцев на ПК**

После включения компьютера обязательно должны запускаться программы аутентификации личности. Список пользователей ПК определяется администратором системы. После входа в систему, под своим логином и паролем, пользователь должен проходить идентификацию по отпечатку пальца.

После завершения работы пользователь выходит из системы, предоставив возможность работать другим пользователям. В случае, если пользователь оставил ПК, не завершив сеанс работы, через некоторое время (например 10 минут), с помощью настроек в операционной системе, ПК должен уходить в «спящий» режим. Выход из «спящего» режима сопровождается аутентификацией пользователя.

Периодический контроль во время работы нецелесообразен по нескольким причинам:

* мешает непосредственно работе;
* не увеличивает степень защищенности данных хранящихся на ПК (возможна скрытая видеосъемка во время работы).

**9. Минимальная конфигурация ПК для установки системы контроля доступа к помощью сканера отпечатков**

Конфигурация ПК определяется, в первую очередь, типом операционной системы. Наиболее распространены в настоящее время ОС Windows 7,8,10 фирмы Microsoft. Ниже представлены минимальные требования к установке операционных систем.

Для установки Windows 7 :

- Процессор: 32-разрядный (x86) или 64-разрядный (x64) процессор\* с тактовой частотой 1 ГГц или выше;

- Оперативная память: 1 ГБ (для 32-разрядного процессора) или 2 ГБ (для 64-разрядного процессора) ОЗУ;

- Свободное дисковое пространство: 16 ГБ (для 32-разрядной системы) или 20 ГБ (для 64-разрядной системы) свободного места на жестком диске;

- Видеокарта: Графическое устройство DirectX 9 с драйвером WDDM 1.0 или более поздней версии.

Для установки WINDOWS 8:

- Процессор: 1 гигагерц (ГГц) или выше с поддержкой PAE, NX и SSE2;

- Оперативная память: 1 гигабайт (ГБ) для 32-разрядной Windows или 2 ГБ для 64-разрядной версии;

- Свободное дисковое пространство: 16 ГБ для 32-разрядной Windows или 20 ГБ для 64-разрядной версии;

- Видеокарта: Microsoft DirectX 9 графический адаптер с WDDM драйвером.

Для установки WINDOWS 10:

- Процессор: Процессор с частотой работы не менее 1ГГц или SoC;

- Оперативная память: – 1Гб для версии х32 и 2Гб для х64;

- Свободное дисковое пространство: 16 Гб для версии ОС х32 и 20 Гб для х64;

- Монитор с разрешением не менее 800х600;

- Видеокарта: с поддержкой не ниже DirectX 9 с драйвером WDDM 1.0.

В комплекте со сканерами производитель поставляет программное обеспечение (ПО). ПО представляет собой, как правило, набор драйверов и интерфейс. С помощью интерфейса формируется база данных отпечатков пользователей ПК с соответствующей информацией (Фамилия, Имя, Отчество; подразделение; должность; фотография и т.д.). Возможны другие дополнительные сведения и функциональность.

Объем требуемого дискового пространства для установки ПО индивидуален для каждого производителя. Он может достигать существенных величин. Например, он может достигать 1-1.5 ГГб для сканеров фирмы Smartec.

Таким образом, минимальные требования к объему требуемого дискового пространства ПК складываются из объема необходимого для операционной системы и сканера.

Производительность ПК зависит от тактовой частоты центрального процессора и числа его ядер. Чем выше тактовая частота процессора, тем быстрее работает ПО. Дополнительные ядра процессора позволяют ускорять работу ПО, если разработчик ПО использует механизмы параллельных вычислений. Минимальное значение тактовой частоты процессора указано в требованиях к установке операционной системы.

В рамках данной работы для исследования отпечатков пальцев применяется сканер ZK6500 с возможностью экспорта отпечатка пальцев. Описание на сканер находится в Приложении Б. Для инсталяции программного обеспечения и драйверов необходимо 10641496 байт дискового пространства. Прилагаемая программа позволяет сканировать и экспортировать отпечатки пальцев Рисунок 9.1.

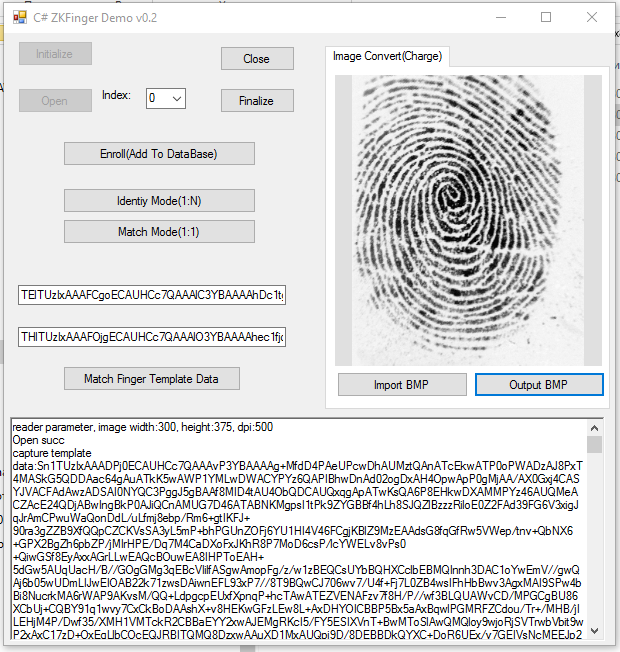


Рисунок 9.1 Экспорт отпечатка пальца

Полученные изображения отпечатков предоставляют возможность исследовать их с помощью самостоятельно разработанного программного обеспечения (Рисунок 9.2).



Рисунок 9.2 Отпечаток пальца в формате bmp

**10. Реализация метода сравнения отпечатков по особым точкам**

## 10.1 Бинаризация изображения глобальным методом

Для реализации бинаризации изображения глобальным методом необходимо задать порог преобразования. Поскольку фон изображения может варьироваться в широких пределах, то выбор порога становится непростой задачей. Неправильно выбранный порог может приводить к существенным потерям изображений папилярных линий или полной потере изображения. Внесем некоторые изменения в метод. Для того, чтобы не потерять изображение совсем, нужно найти значение яркости самого светлого пикселя и самого темного. Порог нужно выбирать в диапазоне между ними. На Рисунке 10.1 отображена бинаризация изображения глобальным методом, где порог задается в диапазоне между самым светлым и самым темным пикселями (в процентах).

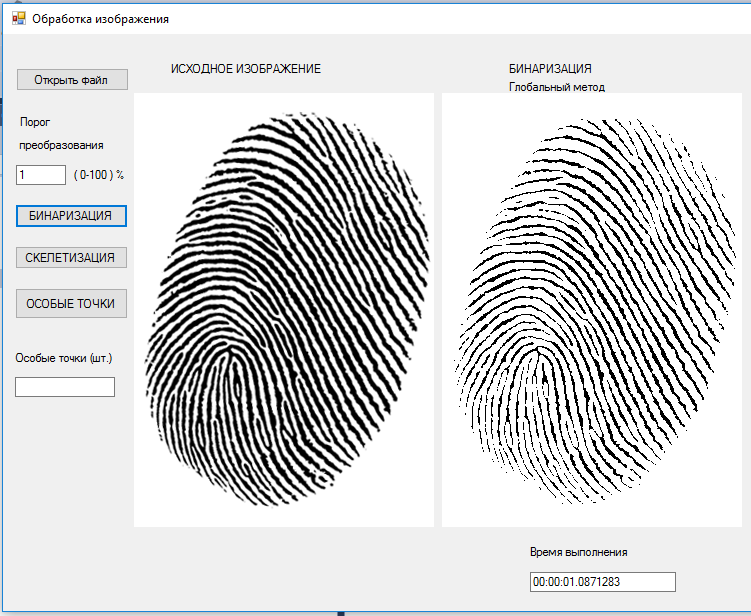


Рисунок 10.1 Бинаризация изображения с порогом 1 %

При сравнении верхней части обоих изображений видно, что часть папилярных линий утрачена.

На Рисунке 10.2 отображена бинаризация изображения с порогом 99 %.

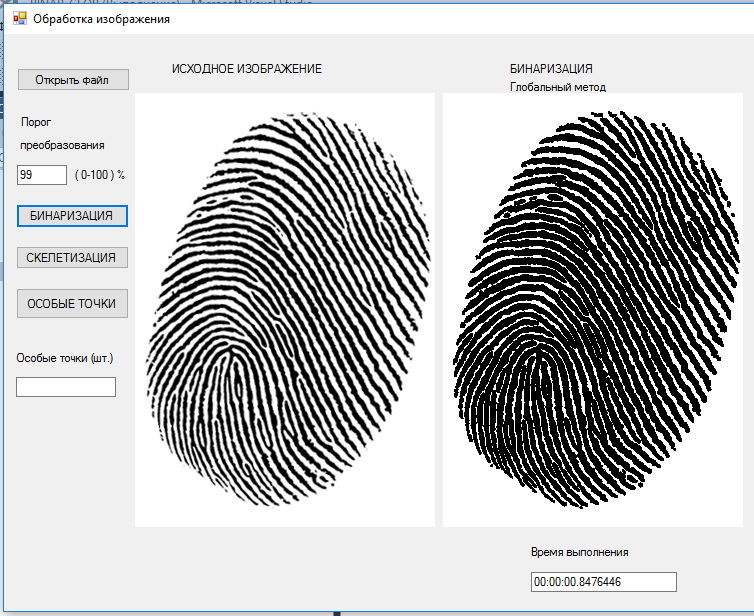


Рисунок 10.2 Бинаризация изображения с порогом 99 %

При сравнении нижней части обоих изображений видно, что часть папилярных линий соединились в общую линию.

Поскольку порог который выбирается находится в диапазоне уже существующих на изображении пикселей, то утратить изображение совсем невозможно. Что касается величины порога, то оптимальным будет значение в диапазоне 35-65%. При этом возможные искажения линий будут минимальны.

На Рисунке 10.3 изображена бинаризация изображения с порогом 50 %.

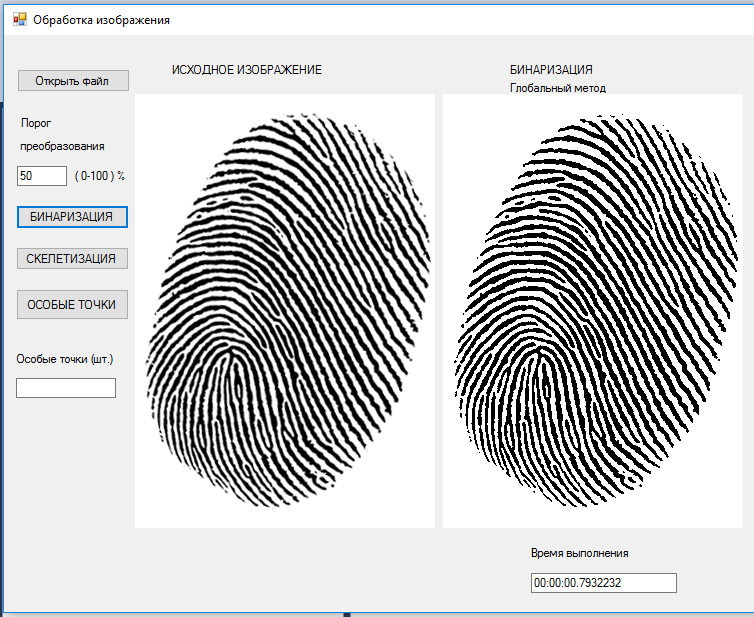


Рисунок 10.3 Бинаризация изображения с порогом 50 %

Программный код программы находится в Приложении Б.

10.1.1 Описание алгоритма бинаризации (Глобальный метод)

а) В изображении отпечатка находим максимальное (max) и минимальное (min) значения яркости пикселей;

б) Выбираем значение порога яркости (porog) в диапазоне между max и min (оптимально посередине);

в) Каждое значение яркости пикселя сравниваем с порогом, и если оно ниже порога устанавливаем яркость равную 0,если выше устанавливаем яркость равную 255.

Яркость пикселя = 0

Да

Яркость пикселя = 255

Нет

Яркость пикселя больше порога

Нет

Пока не закончатся пиксели

Установка порога яркости (porog) в диапазоне между max и min

Поиск максимального (max) и минимального (min) значения яркости пикселей в изображении

Начало

Конец

Да

Рисунок 10.4 Алгоритм бинаризации (глобальный метод)

## 10.2 Алгоритм скелетизации (Метод Зонго-Суня)

Метод Зонго-Суня не требует входных параметров и является одним из самых простых. Алгоритм является многопроходовым (обрабатывает изображение пока не исключит пиксели подлежащие изменению). Вследствие этого, время обработки достаточно велико 23 сек (Рисунок 10.5).

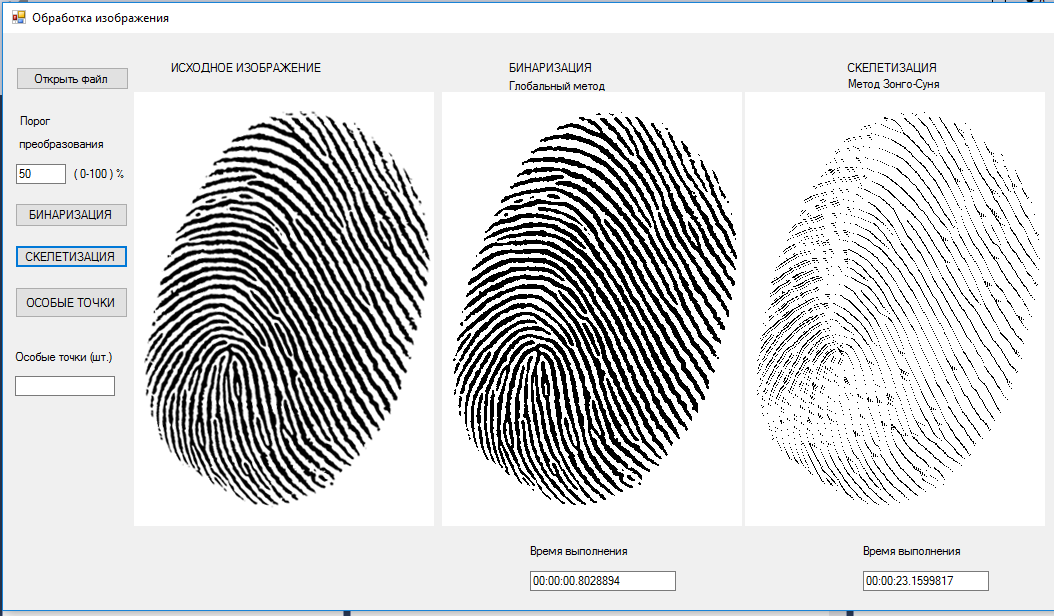


Рисунок 10.5 Скелетизация изображения (Метод Зонго-Суня)

10.2.1 Описание алгоритма скелетизации (Метод Зонго-Суня)

а) Создадим матрицу байтов такого же размера как изображение отпечатка. Проинициализируем нулями. В ней будем фиксировать пиксели, которые уже изменяли, устанавливая 1;

б) Выставим флаг окончания алгоритма finish=true;

в) Введем следующую матрицу:

|  |  |  |
| --- | --- | --- |
| P9 | P2 | P3 |
| P8 | P1 | P4 |
| P7 | P6 | P5 |

Данная матрица попиксельно накладывается на изображение таким образом, чтобы P1 поочередно совмещался со всеми пикселями изображения.

Белый пиксель на изображения при этом имеет значение 0, а черный 1;

г) Пиксель меняет свой цвет с белого на черный, если выполняется ряд условий:

1) 2 <= P2+P3+...+P8+P9 <=6

2) S(P1) = 1,

где S(P1) - количество найденных последовательностей 01 в последовательности P2, P3, P4, P5, P6, P7, P8, P9, P2. Т.е. для удаления пикселя, вокруг него должен существовать только один переход от нуля к единице.

3) P2\*P4\*P6 = 0

4) P4\*P6\*P8 = 0

Для работы с крайними левыми пикселями 3) и 4) пункты необходимо заменить на:

3) P2\*P4\*P8 = 0

4) P2\*P6\*P8 = 0

Конец

Да

Начало

Инициализация матрицы rempic

Флаг окончания алгоритма finish=true

Пока finish=true

2 <= P2+P3+...+P8+P9 <=6

S(P1) = 1

X=0

P2\*P4\*P6 = 0 P4\*P6\*P8 = 0

P2\*P4\*P8 = 0 P2\*P6\*P8 = 0

Да

Да

Да

Нет

Нет

Нет

Нет

Пиксель P1 окрасим в белый цвет

Пиксе

Пока X<imageI.Width

Пока Y<imageI.Height

Да

Нет

Да

Finish=false

Нет

rempic(x,y)=1

Нет

Да

imageI(x,y)=0

P1…P9=1

P1…P9=0

Да

Нет

Рисунок 10.6 Алгоритм скелетизации (Метод Зонго-Суня)

## 10.3 Алгоритм обнаружения особых точек

Алгоритм обнаружения особых точек ищет на изображении отпечатка после скелетизации особые точки разветвления и окончания папилярных линий. Для сравнения отпечатков наибольший интерес представляют собой точки разветвления, их меньше, но достаточно для идентификации (Рисунок 10.7).

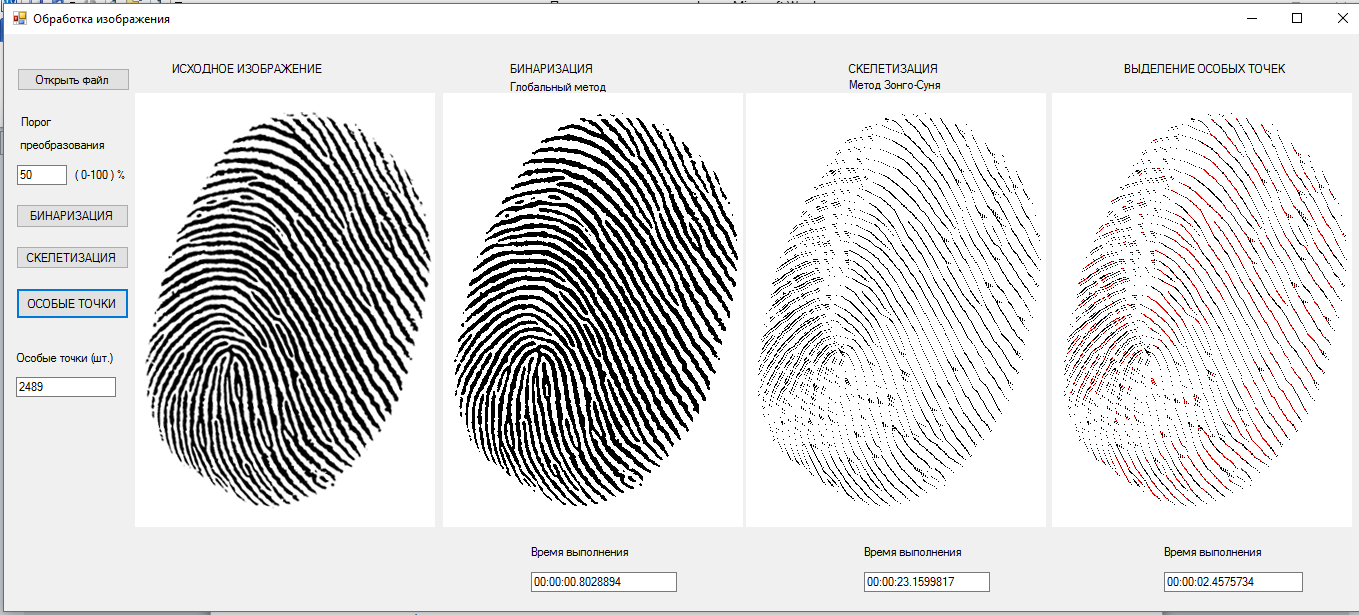


Рисунок 10.7 Выделение особых точек

10.3.1 Описание алгоритма обнаружения особых точек

а) Устанавливаем счетчик особых точек count=0, счетчик суммы sum=0;

б) Введем следующую матрицу:

|  |  |  |
| --- | --- | --- |
| P9 | P2 | P3 |
| P8 | P1 | P4 |
| P7 | P6 | P5 |

Данная матрица попиксельно накладывается на изображение таким образом, чтобы P1 поочередно совмещался со всеми пикселями изображения. Белый пиксель на изображения при этом имеет значение 0, а черный 1;

в) Особая точка считается найденой если:

1) Центральный пиксель P1=1 (т.е. он черный);

2) sum = (P9 + P2 + P3 + P8 + P4 + P7 + P6 + P5)=2 (разветвление), если sum=1 (окончание);

г) Если особая точка найдена, увеличим счетчик count на 1.

Конец

Да

Начало

P1=1

sum = (P9 + P2 + P3 + P8 + P4 + P7 + P6 + P5)

count=count+1

Да

Да

Sum=2

Нет

Sum=0, count=0

Пиксель P1 окрасим в красный цвет

Пиксе

Пока X<imageI.Width

Пока Y<imageI.Height

Да

Да

Нет

imageI(x,y)=0

P1…P9=1

P1…P9=0

Да

Нет

Пиксе

Нет

Рисунок 10.8 Алгоритм выделения особых точек

Выводы: Проанализируем полученные результаты. Алгоритм обработки отпечатка пальца реализован, но общее время обработки около 26 сек. Такое время обработки не приемлемо для практического применения. Необходимо искать альтернативные методы на каждом этапе обработки.

Программный код находится в Приложении B.

# 11. Альтернативные методы обработки изображения

Из разработанных методов, самое большое время реализации имеет метод скелетизации Зонго-Суня (около 23 сек). Анализ программы показал, что такое время было получено за 18 проходов по изображению. Одним из способов уменьшить времени обработки является применение других методов скелетизации. Результаты подобных исследований можно посмотреть в статье “ Сравнительный анализ алгоритмов скелетизации бинарных изображений” Одесского национального политехнического университета [7].

Исследованию подлежали:

1. Алгоритм морфологической скелетизации Серра, основанный на последовательном применении морфологических операций эрозии и дилатации.

2. Алгоритм Розенфельда или волновой алгоритм, основанный на имитации «пожара травы в прерии» путем последовательного перекрашивания граничных точек объекта.

3. Алгоритм Жанга-Суена, который является параллельным алгоритмом с применением скользящей утончающей маски размером 3х3 пиксела и осуществляется в две последовательные итерации.

4. Параллельный алгоритм Гуо–Холла, также состоящий из двух итераций .

5. Алгоритм Стентифорда, относящийся к итерационным алгоритмам с последовательным применением четырех утончающих шаблонов размером 3x3 пиксела.

Язык программирования C++.

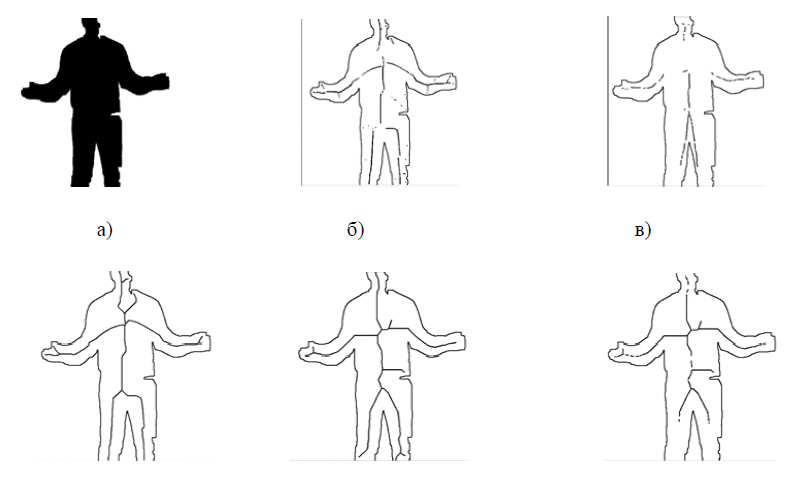


Рисунок 11.1 Обрабатываемое изображение (а) и его скелетизация по различным алгоритмам: морфологическому (б), по алгоритму Розенфельда (в), Жанга-Суена (г), Гуо-Холла (д) и Стентифорда (е)

Временные затраты составили соответственно: б) 124 мс, в) 232 мс, г)1080 мс, д) 1493 мс, е) 526 мс. В других экспериментах временные затраты были пропорциональны приведенным.

Были сделаны следующие выводы:

“Быстродействующие алгоритмы (б, в) имеют определенные дефекты скелетизации, выражающиеся в разрывах связности скелета и появлении ложных его ветвей.

Алгоритмы высокого качества (г, д) требуют существенно больших затрат счетного времени. Для применения в системах обработки реального времени в качестве компромиссного, т. е. сочетающего достаточное качество с быстродействием, может быть рекомендован алгоритм Стентфорда”.

Среди исследованных алгоритмов есть алгоритм Зонго-Суня п.в) (Разница в имени возникла из-за разричных переводов с китайского).

С научной точки зрения, время выполнения программ, расчеты которых велись на компьютерах с разными техническими характеристиками, сравнивать некорректно. Тем не менее, очевидно, что результаты существенно отличаются.

Уменьшить время реализации в нашем исследовании возможно. Для этого необходимо:

а) Для разработки использовать язык программирования C++ или Си. В тестах производительности языков программирования, эти два языка занимают 1-2 строки и считаются самыми “быстрыми”.

б) Кроме того, применение приемов параллельного программирования, может уменьшить время обработки в несколько раз.

Выводы:

Для раелизации систем обработки отпечатков пальцев в режиме реального времени необходимо:

1. Использовать языки программирования с высокой производительностью (например С и С++).

2. Применять оборудование с многоядерными процессорами, что позволит применять алгоритмы параллельного программирования.

3. Выбор алгоритма зависит от двух параметров: качества и быстродействия. Чем выше качество, тем медленней работает алгоритм..Если не слишком важно время обработки, то лучше выбрать более качественный алгоритм.

Поскольку, в данной работе исследовалась возможность по защите персональных компьютеров предприятия от несанкционированного доступа, то для сохранения важной или секретной информации, необходимо выбирать алгоритмы с высоким качеством.

**Заключение**

В данной работе были исследованы возможности по реализации защиты персональных компьютеров от несанкционированного доступа в несколько этапов:

1. Рассмотрены и проанализированы основные типы сканеров отпечатков.
2. Для исследования и практического применения были отобраны сканеры для ПК с высоким рейтингом, бытовые и для профессионального применения, с характеристиками и кратким описанием (Приложение А).
3. Для исследования отпечатков пальцев реализован стенд на базе сканера ZK6500 фирмы ZKTeco (Приложение Б).
4. Рассмотрены и проанализированы методы обработки отпечатков пальцев. Был выбран метод сравнения по особым точкам.
5. Метод сравнения по особым точкам реализуется в несколько этапов. Для каждого этапа произведен отбор и анализ алгоритмов реализации (4-5 шт. ).
6. Реализована обработка отпечатка пальца на языке C#. (Программный код в Приложении В).
7. Произведен анализ полученных результатов. Анализ показал, что обработка осуществляется успешно, но время обработки не приемлемо для практического применения.
8. Рассмотрены и выбраны методы для улучшения технической и программной реализации системы защиты компьютеров предприятия от несанкционированного доступа.

В продолжение исследований в 3-м семестре планируется исследовать возможности видеоконтроля для идентификации пользователя для чего необходимо выполнить следующие работы:

1. Рассмотреть и проанализировать основные типы видеокамер для ПК .
2. Для исследования и практического применения выбрать видеокамеру.
3. Для исследования установить видеокамеру на ПК.
4. Рассмотреть и проанализировать методы обработки видеоизображения для идентификации личности.
5. Выбрать метод обработки видеоизображения для идентификации личности (по критериям качество и время выполнения).
6. Осуществить поиск алгоритмов и программных решений для реализации метода обработки.
7. В случае необходимости, осуществить реализацию метода обработки видеоизображения для идентификации личности на одном из языков программирования.
8. Оценить полученные результаты и сделать выводы.

Литература

1. Б. Яне “Цифровая обработка изображений”, Техносфера, Москва 2007г.;
2. “Некоторые вопросы применения методов сравнения отпечатков пальцев для биометрических систем идентификации личности” Фам Зуй Тхай Аспирант, кафедра МОСИТ, Московский государственный университет информационных технологий, радиотехники и электроники (МИРЭА);
3. [Задорожный В., "Идентификация по отпечаткам пальцев", Часть 1, 2004;](http://www.cherry.ru/interes/97.html)
4. [Задорожный В., "Идентификация по отпечаткам пальцев", Часть 2, 2007;](http://www.4delo.ru/inform/articles/419370.html)
5. [Давлетханов М., "Способы идентификации по отпечаткам пальцев", 2004;](http://infobez.com/article.asp?ob_no=2568)
6. В.Ю. Гудков, Д.А. Клюев “Скелетизация бинарных изображений и выделение особых точек для распознавания отпечатков пальцев” Южно-Уральский государственный университет, г. Челябинск 2015г.;
7. “ Сравнительный анализ алгоритмов скелетизации бинарных изображений” К. т. н. В. А. Болтенков, Д. В. Малявин, Нгуен Гуи Кионг Одесский национальный политехнический университет Украина, г. Одесса 2014г.;
8. [Вакуленко А., Юхин А., "Биометрические методы идентификации личности: обоснованный выбор и внедрение", 2007;](http://www.npo-inform.com/press/biovybor/)
9. [Зиятдинов А. И., "Принципы построения систем биометрической аутентификации"/ МФТИ, 2005, 8 с.](http://www.re.mipt.ru/infsec/2005/essay/2005_Construction_Principles_of_Biometric_Authentication_Systems__Ziyatdinov.htm)

Приложение А

Сканеры для исследования и применения

1. **PQI Mini USB Fingerprint Reader**



**Технические характеристики**

* Интерфейс: USB A (USB 2.0, 3.0, 3.1)
* Количество регистрируемых отпечатков: до 10
* Угол сканирования: 360 градусов
* Максимальное время распознавания: до 0,15 секунд
* Поддерживаемые ОС (32/64 бит): Windows 7, Windows 8, Windows 10
* Функция Windows Hello: есть (для Windows 10)
* Поддержка браузеров: Google Chrome, Internet Explorer
* Габаритные размеры: 2.0х1.7х1.2 см
* Вес: 5 г
* Цена 2450 р

**Краткое описание**

У сканера хорошее время распознавания, работает с операционными системами Windows 7,8,10. Для работы с Windows 10 предусмотрена функция Windows Hello. Она позволяет проходить идентификацию гораздо быстрее и удобней, чем при использовании пароля. Напомним, что помимо отпечатков пальцев, эта функция опирается также на распознавание лица, для которого понадобилась бы совместимая с Windows Hello веб-камера. Предназначен для небольшого количества пользователей.

[**2. Verifi P2000 Fingerprint Reader**](https://setphone.ru/stati/luchshie-usb-skanery-otpechatkov-paltsev-dlya-kompyutera/#Verifi_P2000_Fingerprint_Reader)



**Технические характеристики**

* Интерфейс: USB 2.0
* Поддерживаемые ОС (32/64 бит): Windows 7, Windows 8, Windows 10
* Биометрическая структура Windows: Да
* Сенсорная технология: технология Trueprint® Subsurface RF
* Разрешение изображения: высокое разрешение 500 dpi
* Область изображения: 9,75 мм x 0,41 мм; 192 x 8 пикселей
* Покрытие поверхности: устойчивое к царапинам
* Сопротивление электростатическому разряду: +/- 15 кВ
* Влажность: от 5% до 90% при 30 ° C Диапазон температур: от 0 ° C до 70 ° C Требования к питанию Напряжение: 4,5-5,25 В
* Потребляемая мощность: активный 50 мА
* Цена 5110р

**Краткое описание**

P2000 дает возможность биометрически идентифицировать пользователей в сети или на одном ПК. При создании модели P2000, компания Verifit планировала сделать прочное и долговечное устройство так, чтобы это не оказывало никакого влияния на процесс сканирования и точность распознавания результатов. Verifi P2000 Fingerprint Reader это достаточно портативное устройство, которое можно легко носить с собой. Кроме того, оно совместимо с куда большим количеством компьютеров, чем другие подобные модели. Устойчив к электростатическому разряду, способен работать при высокой влажности.

**3. Kensington VeriMark**



**Технические характеристики**

* Интерфейс: USB 2.0
* Шифрование данных USB • AES-256 / SHA-256
* Угол сканирования: 360 градусов
* Поддерживаемые : 7, 8.1, 10 Для Win 7, 8 требуется установка программного обеспечения Plug & Play
* Функция Windows Hello: есть (для Windows 10)
* Уровень ложного отказа (FRR) 2%
* Коэффициент ложного принятия (FAR) 0,001%
* Цена 6716р

**Краткое описание**

Все полученные данные устройство хранит в зашифрованном виде. Когда датчик сканирует отпечаток пальца, он тут же его шифрует. Прервать данный процесс невозможно, поэтому взломать программное или аппаратное обеспечение и извлечь биометрические данные не получится. Кроме того, устройство использует особый шаблон для расшифровки полученных сканов, который есть только у производителя. Это делает защищает его от хакерских атак и других попыток получения несанкционированного доступа.

Сканер поддерживает работу с относительно старыми [USB-A](https://setphone.ru/stati/usb-a-i-usb-c-v-chem-raznitsa/) портами независимо от версии выпуска.

Идеально подходит для развертывания на предприятии и легко интегрируется в существующую ИТ-инфраструктуру. Пользователи сохраняют свой ключ для безопасного входа в популярные службы, а ИТ-отдел может легко управлять доступом, привилегиями и паролями сотрудников. • Обеспечивает новейшие веб-стандарты, одобренные Консорциумом World Wide Web (W3C). Выполняет аутентификацию без сохранения паролей на серверах, обеспечивая профессионалам большую безопасность, удобство, конфиденциальность и масштабируемость.

**Сертификат FIDO U2F**

Универсальная двухфакторная аутентификация (U2F) Fast IDentity Online (FIDO), улучшающая ваш опыт работы в мире двухфакторной аутентификации, обеспечивает беспроблемное взаимодействие и отвечает требованиям для входа в систему с помощью ключа безопасности для поставщиков облачных услуг и программного обеспечения, включая Google, Dropbox, GitHub. и Facebook.

**Совместимость с менеджером паролей**

Поддерживает популярные инструменты, такие как Dashlane, LastPass (Premium), Keeper (Premium) и Roboform, чтобы позволить отпечатку пальца аутентифицироваться и автоматически вводить имена пользователей и пароли для веб-сайтов

**4.**  **Benss Fingerprint Reader Analyzer**



**Технические характеристики**

* Интерфейс: USB 2.0
* Количество регистрируемых отпечатков: до 10
* Угол сканирования: 360 градусов
* Поддерживаемые : Windows7 / 8 / 8.1 / 10 32/64-бит
* Функция Windows Hello: есть (для Windows 10)
* Уровень ложного отказа (FRR) <3%
* Коэффициент ложного принятия (FAR) 0,002%
* Размер: 0,83x0,55x 0,31 дюйма
* Цена 1825р

**Краткое описание**

Он поддерживает шифрование входа в Windows, блокировки экранной заставки, файла / папки.

Benss Fingerprint Reader Analyzer работает за счет самообучающегося алгоритма, который автоматически создает и обновляет ваш профиль при каждом касании. Чем дольше вы пользуетесь устройством, тем точнее будут результаты. Благодаря этому сканер труднее обмануть при помощи поддельных отпечатков пальцев. Так как датчик постоянно сканирует ваш палец и обновляет базу данных, то он сразу же обнаруживает попытки получения несанкционированного доступа. Это одно из самых эффективных устройств для противостояния хакерским атакам.

**5. IDEM BioScan Compact**



**Технические характеристики**

* Интерфейс: USB 2.0
* Количество регистрируемых отпечатков: до 10
* Угол сканирования: 360 градусов
* Поддерживаемые : только Windows 10 - без поддержки Linux, Mac или Windows 10 Enterprise Version.
* Быстрый вход в Windows 10, время входа <0.25 c
* Функция Windows Hello: есть (для Windows 10)
* Сертификат FCC, простота установки и использования, интуитивно понятное и простое руководство по быстрому запуску и поддержка заказчиков в США сроком на 1 год.
* Функция защиты для распознавания фальшивых биометрических данных

• Поддержка шифрования файлов / папок и входа в Интернет одним касанием (с дополнительным программным обеспечением)

* Цена 1427 р

**Краткое описание**

Данный сканер имеет встроенную защиту для распознавания фальшивых биометрических данных. При установке дополнительного программного обеспечения сможет шифровать на компьютере отдельные файлы и папки, а также хранить в специальной программе данные логинов и паролей для доступа к веб-сайтам. Так пользователь сможет авторизоваться на любых интернет порталах при помощи обычного прикосновения пальца к датчику.

**6. Blucoil SecuGen Hamster Pro 20**



**Технические характеристики**

* Разрешение изображения: 500 точек на дюйм
* Размер изображения: 300x400p
* Размер стола: 18,2 x 22,9 мм
* Зона чувствительности: 15,2 x 20,3 мм
* Шкала серого изображения: 256 уровней (8 бит)
* Скорость интеллектуального захвата: 0,2 ~ 0,5 секунды
* Биометрические стандарты: INCITS 378, ISO / IEC 19794-2, ISO / IEC 19794-4, FIPS 201, FBI (PIV, Mobile ID FAP 20)
* Поддерживаемые операционные системы: Windows 10 / 8.1 / 8/7, Windows Server 2016/2012/2008 R2, Android 3.1 и выше, Linux
* Функция Windows Hello: есть (для Windows 10) без дополнительной установки
* Интерфейс: USB 1.1 Full-Speed, USB 2.0 Hi-Speed
* Напряжение питания: 5 В через USB
* Потребление тока: макс. 150 мА
* Рабочая температура: от -20 до 65 градусов Цельсия
* Влажность при эксплуатации: относительная влажность не более 90%, без конденсации
* Сертификаты: FCC, CE, KC, RoHS, STQC, WHQL
* Пыле / водонепроницаемость: сертифицировано IP65
* Длина кабеля: 1500 мм
* Размеры: 53,9 х 58,5 х 31,6 мм
* Вес: 98 г
* Порты: 4 порта - USB-A 3.0 с выходом 5 В, 1 А
* Интерфейс: кабель Type-C
* Длина кабеля: 15 см
* Восходящее питание: через интерфейс type-C + порт micro USB
* Пропускная способность: до 5 Гбит / с
* Сертификация: CE, FCC
* Цена 8170 р

**Краткое описание**

Соответствует стандартам качества STQC И ФБР - компактный USB-считыватель отпечатков пальцев Hamster Pro 20 сертифицирован ФБР и соответствует требованиям FIPS 201 (PIV) и Mobile ID FAP 20. Низкопрофильный датчик также сертифицирован по стандарту STQC и соответствует стандартам UIDAI, Aadhaar. Он также имеет степень защиты от пыли и воды IP65.

Сканер аутентифицирует, идентифицирует и проверяет, но и отклоняет поддельные отпечатки пальцев. Кроме того, ридер не принимает оставленные отпечатки. Сканер автоматически обнаруживает палец и сканирует высококачественные отпечатки пальцев, даже если у вас есть шрамы на пальцах.

**7. Suprema BioMini Plus 2**



**Технические характеристики**

* Интерфейс: USB 2.0
* Разрешение изображения: 500 точек на дюйм
* Сенсор: оптический
* Размер изображения: 315x354p
* Размер стола: 16 x 18 мм
* Поддерживаемые операционные системы: Windows 7,8,8.1,10 32/64 бит, Linux, Android
* Диапазон рабочих температур от -10°C до 50°C
* Сертификация: FBI PIV/FBI Mobile ID FAP 10
* Цена 10018 р

**Краткое описание**

Многофункциональная настольная модель предназначена для работы с биометрическими данными в централизованных системах доступа или для использования при учете рабочего времени. Устройство оснащено оптическим сенсором по фирменной запатентованной технологии MDR.

Настольный сканер отпечатков пальцев BioMini Plus 2 отличается исключительно высокой производительностью и точностью и может успешно захватывать даже пересушенные и влажные пальцы, а также стабильно работать в условиях прямой засветки. Сканер оснащён современным оптическим сенсором, способным захватывать изображение отпечатка пальца с разрешением 500 ppi, что позволяет строить высокоточные биометрические шаблоны, обеспечивающие надёжную идентификацию с крайне низкой вероятностью ошибки. Корпус  класса IP65 защищает устройство от пыли и влаги: сканер будет продолжать работать даже при попадании на него мелкой пыли или водяных струй (для корректной работы потребуется лишь протереть сенсор, удалив с него загрязнения и излишнюю влагу).

**8. Smartec ST-FE800**



**Технические характеристики**

* Интерфейс: USB 2.0/USB 1.1
* Разрешение изображения: 500 точек на дюйм
* Сенсор: оптический
* Разрешение: 500-1000 dpi
* Глубина цвета: 8 бит (256 уровней серого)
* Размер стола: 15 x 20 мм
* Поддерживаемые операционные системы: Windows 7,8,10 и Windows Server 2012,2016
* Питание 5 В через USB порт
* Потребляемый ток при сканировании: 200 мА
* Диапазон рабочих температур: от -20 до + 50 °C без конденсации
* Диапазон рабочей влажности: 20%-90%
* Габариты: 49x44x20 мм
* Цена 4648 р

**Краткое описание**

Модель отличается хорошей совместимостью и расширенными возможностями по интеграции с ПО от разных производителей. Устройство оснащено датчиком с высоким разрешением до 1000 dpi, может стабильно работать в условиях сильной освещенности, а также без проблем считывает отпечатки вне зависимости от состояния кожного покрова. Повышает надежность устройства широкий диапазон рабочих температур. Сканер предусматривает защиту от фальсификации отпечатков, может использоваться совместно с различными программными приложениями для ввода в базу данных шаблонов отпечатков пальцев, при этом для интеграции с ПО стронних производителей можно использовать SDK.

**9. Anviz U Bio**



**Характеристики**

* Интерфейс: USB 2.0
* Область сканирования: 22мм\*18мм
* Разрешение: 500 DPI
* Питание: 5V  (по USB)
* Материал: Износостойкий промышленный пластик
* Цена 6500 р

**Краткое описание**

Предназначен для занесения отпечатков пальцев пользователей в единую базу данных, с последующей их отправкой на терминалы Anviz по TCP/IP, GPRS, RS485 и т.д. Так же, может быть использован для авторизации пользователей в различных средах и программах. Устройство Plug and Play (не требует установки драйверов). Рабочая поверхность сканера не бьется, устойчива к царапинам, защищена от попадания влаги и пыли.

Устройство подключается к компьютеру при помощи USB-порта, через него же осуществляется и питание модели.

Приложение Б

Сканер для исследования в данной работе

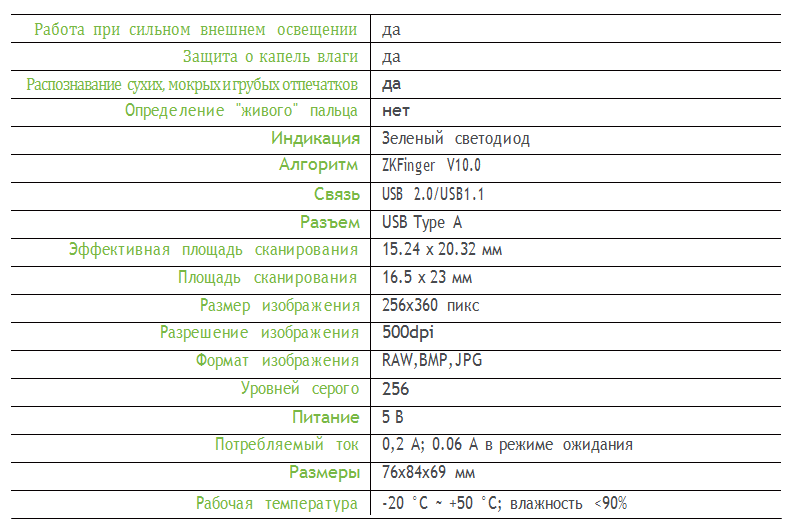
Описание

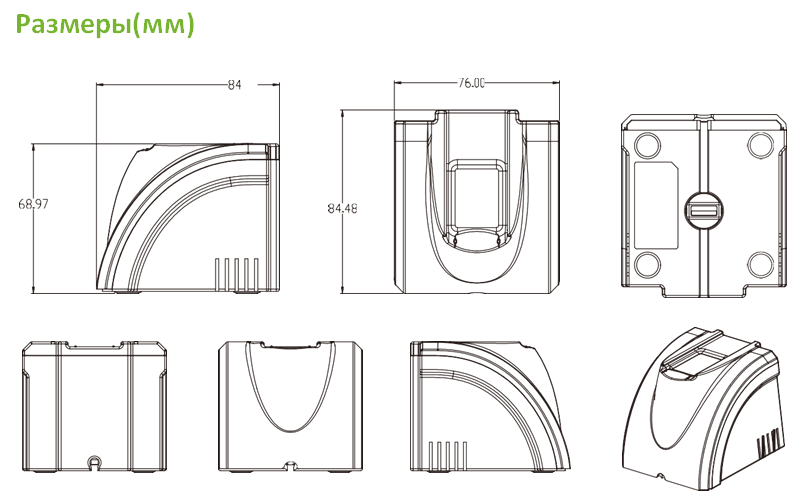
**ZK6500** является новейшим оптическим сканером отпечатков пальцев, разработанным ZKTeco. Он прост в использовании и обладает высоким разрешением, большой скоростью сканирования и низким энергопотреблением. Может быть подключен к мобильным устройствам на ОС Android. Мы предоставляем SDK (Windows, Android, Linux) для разработчиков.

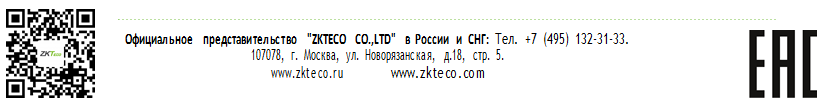
# Функции

* Функция обнаружения отпечатков пальцев.
* Быстрое распознавание сухих, мокрых и грубых отпечатков пальцев.
* SDK для Windows, Android, Linux.
* Высокоскоростной USB 2.0.
* Современный дизайн.





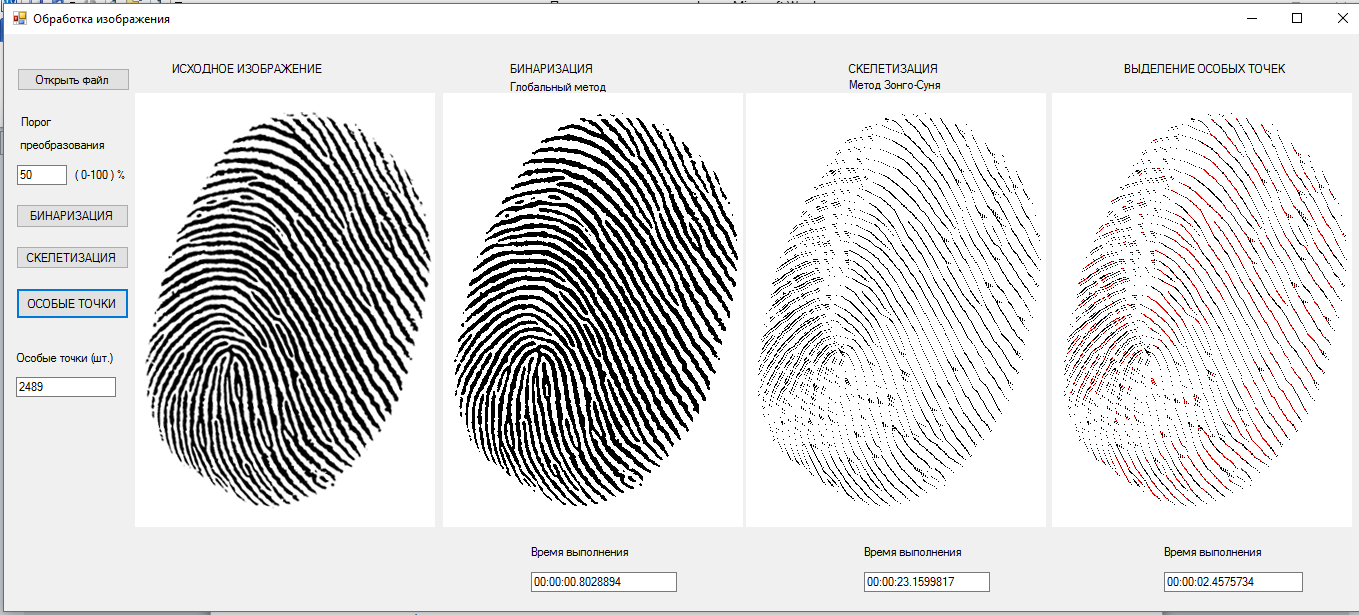




Приложение В

Программная реализация метода сканирования отпечатка по особым точкам

Внешний вид формы



Код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

namespace BINAR\_GLOB

{

public partial class Form1 : Form

{

// Изображение Исходного кадра

Bitmap imageIsx;

// Изображение Кадра после БИНАРИЗАЦИИ

Bitmap imageBin;

// Изображение Кадра после СКЕЛЕТИЗАЦИИ

Bitmap imageSkel;

//Изображение Кадра после СКЕЛЕТИЗАЦИИ

// c выделением точек разветвления

Bitmap imagePoi;

//-------- Структура для особых точек

struct MyPoint

{

public int xt; // Координаты особой точки в изображ.

public int yt;

// Тип точки - 0 - окончание или 1-разветвление

public byte typepoint;

// Номера в ячейке 3x3

// Единственный черный пиксель у окончания

public byte ok;

// Два черных пикселя у разветвления

public byte v1;

public byte v2;

public MyPoint(int xt, int yt, byte typepoint, byte ok, byte v1, byte v2)

{

this.xt = xt;

this.yt = yt;

this.typepoint = typepoint;

this.ok = ok;

this.v1 = v1;

this.v2 = v2;

}

}

//------------------------------

// Флаг открытия файла

Boolean flag ;

// Значение процента

byte persent;

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

// ============ КНОПКА " ОТКРЫТЬ....ФАЙЛ" =======

//Флаг - файл не открывали

flag = false;

// Создаем экземпляр класса окна диалога

// и настраиваем его

OpenFileDialog openFileDialog1 = new OpenFileDialog

{

//-- Настройки диалогового окна -------------

// Название папки

Title = "Поиск файла с отпечатком ",

// Проверка наличия файла

CheckFileExists = true,

// Проверка наличия пути до файла

CheckPathExists = true,

// Тип файлов bmp

DefaultExt = "bmp",

Filter = "(\*.bmp)|\*.bmp",

// Разрешен выход в вышестоящие папки

RestoreDirectory = true,

// Читаем только выбранный тип файлов bmp

ReadOnlyChecked = true,

// Отображаем только выбранный тип файлов bmp

ShowReadOnly = true

//---------------------------------------------

};

// Если окно диалога открылось....

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

try

{ // Если файл открылся

// Исходное изображение

imageIsx = new Bitmap(openFileDialog1.FileName);

// если в pictureBox1 уже заносили изображение

if (pictureBox1.Image != null)

{

// Сотрем изображение

pictureBox1.Image = null;

// Перерисовка изображения

pictureBox1.Invalidate();

}

// Размер кадра

this.pictureBox1.Size = imageIsx.Size;

// Занесли изображение исходного кадра

pictureBox1.Image = imageIsx;

// Перерисовка изображения

pictureBox1.Invalidate();

// Флаг что изображение загружено

flag = true;

}

catch

{

// Не смогли открыть файл

DialogResult rezult = MessageBox.Show("Невозможно открыть выбранный файл",

"Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

// Флаг что изображение НЕ загружено

flag = false;

}

}

//=======================================================

}

private void button2\_Click(object sender, EventArgs e)

{

//========== КНОПКА БИНАРИЗАЦИЯ ===================

// ----- Открыт ли файл ------------

if (flag==false)

{

MessageBox.Show(" Откройте файл с отпечатком! ");

return;

}

//------- Проверка правильно ли введено ли значение процента ------

// Окно пустое ?

if ( textBox1.Text=="")

{

MessageBox.Show(" Введите значение процента! ");

return;

}

try

{

// Конвертация строки с процентами в byte

persent = Convert.ToByte(textBox1.Text);

}

catch

{

// если конвертация не состоялась,

// сработает catch и выдаст сообщение об ошибке

MessageBox.Show(" Введите значение процента в диапазоне 0-100 ! ");

return;

}

// если конвертация успешна, но значение процента больше 100

// выдаем сообщение об ошибке

if ( persent>100)

{

MessageBox.Show(" Введите значение процента в диапазоне 0-100 ! ");

return;

}

//---------------------------------------------------

//=== 1 === БИНАРИЗАЦИИ ГЛОБАЛЬНЫМ МЕТОДОМ===============

// Очистили время выполнения

textBox2.Text = "";

// Очистили изображение

pictureBox2.Image = null;

// Перерисовка изображения

pictureBox2.Invalidate();

// Создали экземпляр класса секундомера

Stopwatch stopwatch = new Stopwatch();//Секундомер

// Запустили секундомер

stopwatch.Start();

// ------ПОДПРОГРАММА БИНАРИЗАЦИИ ГЛОБАЛЬНЫМ МЕТОДОМ

imageBin = GlobalMetod(imageIsx,out imageBin, persent);

// Остановили таймер

stopwatch.Stop();

// Время выполнения

TimeSpan ts = stopwatch.Elapsed;

// Время выполнения преобразовали в строку

// и вывели на экран в textBox2

textBox2.Text = Convert.ToString(ts);

// ПРОРИСОВКА КАДРОВ НА ЭКРАНЕ

//----- Отобразим новый кадр на экране ------------------

pictureBox2.Image = imageBin;

// Размер изображения

pictureBox2.Size = imageBin.Size;

// Перерисовка изображения

pictureBox2.Invalidate();

}

private void button3\_Click(object sender, EventArgs e)

{

//==== 2 === СКЕЛЕТИЗАЦИЯ МЕТОДОМ ЗОГО-СУНЯ============

// Очистили время выполнения

textBox3.Text = "";

// Очистили изображение

pictureBox3.Image = null;

// Перерисовка изображения

pictureBox3.Invalidate();

// Создали экземпляр класса секундомера

Stopwatch stopwatch = new Stopwatch();//Секундомер

// Запустили секундомер

stopwatch.Start();

//==== ПОДПРОГРАММА СКЕЛЕТИЗАЦИЯ МЕТОДОМ ЗОГО-СУНЯ============

imageSkel = ZongoSunMetod(imageIsx, out imageSkel);

// Остановили таймер

stopwatch.Stop();

// Время выполнения

TimeSpan ts = stopwatch.Elapsed;

// Время выполнения преобразовали в строку

// и вывели на экран в textBox3

textBox3.Text = Convert.ToString(ts);

//-----------------------------------------------------------

//----- Отобразим новый кадр на экране ------------------

pictureBox3.Image = imageSkel;

// Размер изображения

pictureBox3.Size = imageSkel.Size;

// Перерисовка изображения

pictureBox3.Invalidate();

}

private void button4\_Click(object sender, EventArgs e)

{

//==== 3 === ВЫДЕЛЕНИЕ ОСОБЫХ ТОЧЕК============

// Очистили число особых точек

textBox5.Text = "";

// Очистили время выполнения

textBox4.Text = "";

// Очистили изображение

pictureBox4.Image = null;

// Перерисовка изображения

pictureBox4.Invalidate();

// Создали экземпляр класса секундомера

Stopwatch stopwatch = new Stopwatch();//Секундомер

// Запустили секундомер

stopwatch.Start();

//==== ПОДПРОГРАММА СКЕЛЕТИЗАЦИЯ МЕТОДОМ ЗОГО-СУНЯ============

imagePoi = PoiskPoint(imageSkel, out imagePoi);

// Остановили таймер

stopwatch.Stop();

// Время выполнения

TimeSpan ts = stopwatch.Elapsed;

// Время выполнения преобразовали в строку

// и вывели на экран в textBox3

textBox4.Text = Convert.ToString(ts);

//-----------------------------------------------------------

//----- Отобразим новый кадр на экране ------------------

pictureBox4.Image = imagePoi;

// Размер изображения

pictureBox4.Size = imageSkel.Size;

// Перерисовка изображения

pictureBox4.Invalidate();

}

//========================================================

//=========== Подпрограммы ===================

//========================================================

public Bitmap GlobalMetod(Bitmap imageI, out Bitmap imageO, int persent)

{

// ------АЛГОРИТМ БИНАРИЗАЦИИ (ГЛОБАЛЬНЫЙ МЕТОД) -----------

//ВХОДНЫЕ ПАРАМЕТРЫ

// imageI - исходное изображение для преобразования

// persent - порог срабатывания в процентах

//ВЫХОДНЫЕ ПАРАМЕТРЫ

// imageO - преобразованное изображение

int x, y; // Координаты пикселя

// Нач. установки цикла

byte mincol = 255;// Значение яркости самого темного пикселя

byte maxcol = 0; // Значение яркости самого светлого пикселя

//----- Просмотрим кадр попиксельно и

// найдем самый светлый и самый темный пиксели

// Ширина кадра

for (x = 0; x < imageI.Width; x++)

{

// Высота кадра

for (y = 0; y < imageI.Height; y++)

{

// Параметры пикселя в экземпляр класса Color

Color pixelColor = imageI.GetPixel(x, y);

// Найдем минимальное значение яркости в кадре

// т.е. самый темный пиксель

if (pixelColor.R < mincol)

{

// Значение яркости самого темного пикселя

mincol = pixelColor.R;

}

// Найдем максимальное значение яркости в кадре

// т.е. самый светлый пиксель

if (pixelColor.R > maxcol)

{

// Значение яркости самого светлого пикселя

maxcol = pixelColor.R;

}

}

}

// --- Значение порога расчитываем в % между

// самым ярким и темным пикселями ---

// Диапазон между самым ярким и темным пикселями

byte diapazon = (byte)(maxcol - mincol);

// Рассчитаем порог

int porog = (byte)(mincol + diapazon \* persent / 100);

// Скопируем исходное изображение в преобразованное

// imageBin = imageIsx;

// ---- Преобразуем кадр по порогу ----

// Ширина кадра

for (x = 0; x < imageI.Width; x++)

{

// Высота кадра

for (y = 0; y < imageI.Height; y++)

{

// Создали экземпляр класса Color

// и занесли в него параметры пикселя

Color pixelColor = imageI.GetPixel(x, y);

// Цвет пикселя (оттенки серого)

int cv = pixelColor.R;

// Если цвет пикселя больше порога

if (cv > porog)

{

// Сделаем его белым

Color newColor = Color.FromArgb(255, 255, 255);

// Занесли белый цвет в пиксель

imageI.SetPixel(x, y, newColor);

}

else

{

// Сделаем его черным

Color newColor = Color.FromArgb(0, 0, 0);

// Занесли черный цвет в пиксель

imageI.SetPixel(x, y, newColor);

}

}

}

imageO = imageI;

return imageO;

}

public Bitmap ZongoSunMetod(Bitmap imageI, out Bitmap imageO)

{

// ------АЛГОРИТМ СКЕЛЕТИЗАЦИИ ЗОНГО-СУНЯ -----------

//ВХОДНЫЕ ПАРАМЕТРЫ

// imageBin - исходное изображение для преобразования

//ВЫХОДНЫЕ ПАРАМЕТРЫ

// imageSkel - преобразованное изображение

int x, y; // Координаты пикселя

//--- Создадим матрицу размером с кадр

// для хранения данных об удаленных пикселях

byte[,] rempic = new byte[imageI.Width, imageI.Height];

//-------- Инициализируем ее нулями

// Если в матрице 0 - пиксель НЕ УДАЛЯЛИ

// 1 - пиксель УДАЛИЛИ

// Ширина кадра

for (x = 0; x < imageI.Width; x++)

{

// Высота кадра

for (y = 0; y < imageI.Height; y++)

{

rempic[x, y] = 0;

}

}

//---------------------------------------------

// флаг окончания работы алгоритма

Boolean finish = true;

do

{

// Если флаг в конце поменяется на false

// алгоритм продолжится

finish = true;

//----- Просмотрим кадр попиксельно

int sum = 0; //Счетчик суммы

// Ширина кадра

for (x = 0; x < imageI.Width - 2; x++)

{

// Высота кадра

for (y = 0; y < imageI.Height - 2; y++)

{

// Элемент матрицы равен 1, пиксель уже меняли

if (rempic[x, y] == 1)

{

// Пиксель меняли, берем следующий

continue;

}

// Создали экземпляр класса Color

// и занесли в него параметры пикселя

Color pixelColor = imageI.GetPixel(x + 1, y + 1);

// Цвет пикселя (либо 255 -белый, 0 - черный)

byte P1 = pixelColor.R; // Пиксель P1

if (P1 == 255)

{ P1 = 0; }

else

{ P1 = 1; }

// ---1 первое условие алгоритма

//---- Соберем значения пикселей

//в окошке 3х3 P1-P9

pixelColor = imageI.GetPixel(x, y);

byte P9 = pixelColor.R; // Пиксель P9

if (P9 == 255)

{ P9 = 0; }

else

{ P9 = 1; }

pixelColor = imageI.GetPixel(x + 1, y);

byte P2 = pixelColor.R; // Пиксель P2

if (P2 == 255)

{ P2 = 0; }

else

{ P2 = 1; }

pixelColor = imageI.GetPixel(x + 2, y);

byte P3 = pixelColor.R; // Пиксель P3

if (P3 == 255)

{ P3 = 0; }

else

{ P3 = 1; }

pixelColor = imageI.GetPixel(x, y + 1);

byte P8 = pixelColor.R; // Пиксель P8

if (P8 == 255)

{ P8 = 0; }

else

{ P8 = 1; }

pixelColor = imageI.GetPixel(x + 2, y + 1);

byte P4 = pixelColor.R; // Пиксель P4

if (P4 == 255)

{ P4 = 0; }

else

{ P4 = 1; }

pixelColor = imageI.GetPixel(x, y + 2);

byte P7 = pixelColor.R; // Пиксель P7

if (P7 == 255)

{ P7 = 0; }

else

{ P7 = 1; }

pixelColor = imageI.GetPixel(x + 1, y + 2);

byte P6 = pixelColor.R; // Пиксель P6

if (P6 == 255)

{ P6 = 0; }

else

{ P6 = 1; }

pixelColor = imageI.GetPixel(x + 2, y + 2);

byte P5 = pixelColor.R; // Пиксель P6

if (P5 == 255)

{ P5 = 0; }

else

{ P5 = 1; }

sum = (P9 + P2 + P3 + P8 + P4 + P7 + P6 + P5);

//------- 1 ое условие алгоритма

if (sum < 2 || sum > 6)

{

// Если сумма не лежит в диапазоне 2-6

// берем следующий пиксель

continue;

}

//---- 2 ое условие алгоритма

int count = 0;

if (P2 == 0 && P3==1)

{ count = count + 1; }

if (P3 == 0 && P4 == 1)

{ count = count + 1; }

if (P4 == 0 && P5 == 1)

{ count = count + 1; }

if (P5 == 0 && P6 == 1)

{ count = count + 1; }

if (P6 == 0 && P7 == 1)

{ count = count + 1; }

if (P7 == 0 && P8 == 1)

{ count = count + 1; }

if (P8 == 0 && P9 == 1)

{ count = count + 1; }

if (P9 == 0 && P2 == 1)

{ count = count + 1; }

if (count!=1)

{

// переходов из 0 в 1 больше 1

continue;

}

//Если пиксель крайний левый

if (x==0)

{

//------- 3 ое условие алгоритма

if (P2 \* P4 \* P8 != 0)

{

// Произведение не 0

// берем следующий пиксель

continue;

}

//------- 4 ое условие алгоритма

if (P2 \* P6 \* P8 != 0)

{

// Произведение не 0

// берем следующий пиксель

continue;

}

}

else

{

//------- 3 ое условие алгоритма

if (P2 \* P4 \* P6 != 0)

{

// Произведение не 0

// берем следующий пиксель

continue;

}

//------- 4 ое условие алгоритма

if (P4 \* P6 \* P8 != 0)

{

// Произведение не 0

// берем следующий пиксель

continue;

}

}

// Если пиксель НЕ меняли ?

if (rempic[x, y] == 0)

{

// Меняем пиксель на белый

// Сделаем его белым

Color newColor = Color.FromArgb(255, 255, 255);

// Занесли белый цвет в пиксель

imageI.SetPixel(x, y, newColor);

//Флаг ставим в false

// значит алгоритм продолжится

finish = false;

}

// Запомнили в матрице, что его меняли

rempic[x, y] = 1;

}

}

//--------------------------------

} while (finish == false);

imageO = imageI;

// Верниули изображение скелетизации

return imageO;

}

public Bitmap PoiskPoint(Bitmap imageI, out Bitmap imageO)

{

// ------АЛГОРИТМ ПОИСКА ОСОБЫХ ТОЧЕК -----------

//ВХОДНЫЕ ПАРАМЕТРЫ

// imageI - исходное изображение для преобразования

//ВЫХОДНЫЕ ПАРАМЕТРЫ

// imageO - преобразованное изображение

int x, y; // Координаты пикселя

//----- Просмотрим кадр попиксельно

int sum = 0; //Счетчик суммы

int count = 0; // Счетчик особых точек разветвления

// Ширина кадра

for (x = 0; x < imageI.Width - 2; x++)

{

// Высота кадра

for (y = 0; y < imageI.Height - 2; y++)

{

// Создали экземпляр класса Color

// и занесли в него параметры пикселя

Color pixelColor = imageI.GetPixel(x + 1, y + 1);

// Цвет пикселя (либо 255 -белый, 0 - черный)

byte P1 = pixelColor.R; // Пиксель P1

if (P1 == 255)

{ P1 = 0; }

else

{ P1 = 1; }

// ---1 первое условие алгоритма

//---- Соберем значения пикселей

//в окошке 3х3 P1-P9

pixelColor = imageI.GetPixel(x, y);

byte P9 = pixelColor.R; // Пиксель P9

if (P9 == 255)

{ P9 = 0; }

else

{ P9 = 1; }

pixelColor = imageI.GetPixel(x + 1, y);

byte P2 = pixelColor.R; // Пиксель P2

if (P2 == 255)

{ P2 = 0; }

else

{ P2 = 1; }

pixelColor = imageI.GetPixel(x + 2, y);

byte P3 = pixelColor.R; // Пиксель P3

if (P3 == 255)

{ P3 = 0; }

else

{ P3 = 1; }

pixelColor = imageI.GetPixel(x, y + 1);

byte P8 = pixelColor.R; // Пиксель P8

if (P8 == 255)

{ P8 = 0; }

else

{ P8 = 1; }

pixelColor = imageI.GetPixel(x + 2, y + 1);

byte P4 = pixelColor.R; // Пиксель P4

if (P4 == 255)

{ P4 = 0; }

else

{ P4 = 1; }

pixelColor = imageI.GetPixel(x, y + 2);

byte P7 = pixelColor.R; // Пиксель P7

if (P7 == 255)

{ P7 = 0; }

else

{ P7 = 1; }

pixelColor = imageI.GetPixel(x + 1, y + 2);

byte P6 = pixelColor.R; // Пиксель P6

if (P6 == 255)

{ P6 = 0; }

else

{ P6 = 1; }

pixelColor = imageI.GetPixel(x + 2, y + 2);

byte P5 = pixelColor.R; // Пиксель P6

if (P5 == 255)

{ P5 = 0; }

else

{ P5 = 1; }

sum = (P9 + P2 + P3 + P8 + P4 + P7 + P6 + P5);

//Центральный пиксель черный

if (P1==1)

{

if (sum==1 )

{

// Точка окончание

}

else if (sum == 2)

{

// Точка разветвление

// Сделаем ее КРАСНОЙ

Color newColor = Color.FromArgb(255, 0, 0);

// Занесли красный цвет в пиксель

imageI.SetPixel(x+1, y+1, newColor);

count = count + 1; // Увеличили счетчик точек

textBox5.Text = Convert.ToString(count);

}

}

}

}

imageO = imageI;

return imageO;

}

//====================================================

}

}