Министерство образования республики Беларусь

Белорусский национальный технический университет

Факультет информационных технологий и робототехники

Кафедра «Системы автоматизированного проектирования»

***Курсовой проект***

по дисциплине

«Методы распознавания образов»

по теме:

«Разработка подсистемы распознавания цены товаров на изображении

гарантийного талона»

Исполнитель: студент гр.107522,

Балашков В.И.

Руководитель проекта: доц. Ковалева И. Л.

Минск 2015

Содержание

[Введение 4](#_Toc438134403)

[1. Обзор предметной области 5](#_Toc438134404)

[2. Постановка задачи 9](#_Toc438134405)

[3.Адаптация и описание используемых алгоритмов 12](#_Toc438134406)

[3.1 Бинаризация изображения гарантийного талона 12](#_Toc438134407)

[3.2. Сегментация 16](#_Toc438134408)

[3.3 Масштабирование цифры 21](#_Toc438134409)

[3.4 Распознавание цифры 22](#_Toc438134410)

[4.Программная реализация 26](#_Toc438134411)

[4.1 Схема программного обеспечения 26](#_Toc438134412)

[4.2 Описание используемых классов, функций и методов 27](#_Toc438134413)

[4.3 Тестирование и руководство пользователя 27](#_Toc438134420)

[Заключение 31](#_Toc438134421)

[Литература 32](#_Toc438134427)

[Приложение 1 33](#_Toc438134428)

# Введение

Распознавание образов играет достаточно значительную роль в нашей жизни, оно способствует упрощению жизни современного человека. Запрограммировав этот процесс, мы можем без особых проблем классифицировать и идентифицировать предметы, явления, различные процессы, сигналы и ситуации и т.п. объектов, которые характеризуются конечным набором заданных нами свойств и признаков и тем самым быстро решать сложные задачи, на которые у человека уйдет весомая часть времени.

В особенности в информационный век человеку может понадобиться переносить хранимую информацию с различных бумажных носителей (конспектов, записных книжек, чеков, гарантийных талонов) на электронные носители, для учета личной информации в удобном для него виде.

Для осуществления этого переноса необходимо выполнить следующие действия: бинаризация, сегментация, масштабирование, распознавание.

В данном курсовом проекте необходимо распознать цены товаров на гарантийном талоне. Возможность автоматически распознавать интересующую информацию о ценах на гарантийном талоне может быть довольно востребованной в случае использования системы учета личных финансов. Данный курсовой проект состоит из 4 разделов:

* В первом разделе описывается предметная область, существующие программные продукты и их недостатки, существующие методы для подготовки изображения к распознаванию.
* Во втором разделе описывается цель курсового проекта, какими должны быть входные и выходные параметры, ограничения.
* В третьем разделе идет описание используемых алгоритмов для подготовки к распознаванию и непосредственно самого распознавания.
* В четвертом разделе представлена программная реализация используемых алгоритмов.

# Обзор предметной области

В процессе распознавания изображений главной задачей является получение описания изображения (рассмотрение его характеристик, качеств), поданного на вход системы. Процедура распознавания применяется к некоторому изображению и обеспечивает преобразование его в некоторое абстрактное описание(будь то набор чисел, цепочку символов и т.п.). Последующая обработка такого описания позволяет отнести исходное изображение к определенному классу. Часто возникает ряд проблем, которые обычно связаны с входными изображениями, они не совпадают с эталонами за счет помех и неточностей, с тем, что входные изображения предъявляются на сложном фоне.

Для улучшения распознавания изображение должно пройти предварительную подготовку, которая заключается в его бинаризации, сегментации и масштабировании.

***Методы бинаризации***

Особое место в предварительной обработке изображения занимает выбор алгоритмов бинаризации. Бинаризацией называется процесс преобразования полутонового изображения в изображение, яркость пикселей которого может иметь только два значения – 0 или 1. Такое изображение называется бинарным. В зависимости от вида изображения применятся различные способы бинаризации:

1. Метод 120:

а) Строится гистограмма

б) Находится максимальный уровень серого в диапазоне t = [0, 120]

в) Черный цвет устанавливается у текущих пикселей, яркость которых меньше либо равна порогового значения плюс 13…15, в остальных случаях цвет белый.

1. Метод средней яркости (локальный метод):

а) Вычисляем среднюю яркость

б) Определяем граничное значение яркости по формуле:

ГЗ=255 – СЯ, где ГЗ- граничное значение, СЯ- средняя яркость.

в) Пиксель становится черным, если среди его соседей есть хотя бы один пиксель яркость которого удовлетворяет условию: яркость ≤ ГЗ, либо яркость анализируемого пикселя ≤ яркости связного с ним +Step, Step= 1…2.

При этом важно чтобы связующий пиксель был уже включен в группу пикселей, которые будут отмечены как черные.

1. Метод 40%:

а) Строится гистограмма

б) Находится количество пикселей, соответствующий заданному проценту

в) Отсчитывается количество пикселей по градациям яркости, начиная с 0 и до момента, когда это количество будет превышать количество пикселей, заданных в постановке задачи бинаризации, и градация, на которой произошла постановка становится порогом бинаризации.

4) Пороговый метод:

a) Определяется порог яркости t.

б) Все что меньше этого порога считается черным, все что больше – белым.

Проанализировав изображения, был сделан вывод о том, что оптимальными для решения поставленной задачи будет выбор метода 120 и порогового метода.

***Методы сегментации***

Это процесс разделения цифрового изображения на несколько сегментов. Цель сегментации заключается в упрощении и/или изменении представления изображения, чтобы его было проще и легче анализировать. Сегментация изображений обычно используется для того, чтобы выделить объекты и границы (линии, кривые, и т. д.) на изображениях. Более точно, сегментация изображений — это процесс присвоения таких меток каждому пикселю изображения, что пиксели с одинаковыми метками имеют общие визуальные характеристики.

Результатом сегментации изображения является множество сегментов, которые вместе покрывают всё изображение, или множество контуров, выделенных из изображения. Все пиксели в сегменте похожи по некоторой характеристике или вычисленному свойству, например, по цвету, яркости или текстуре. Соседние сегменты значительно отличаются по этой характеристике. Выделяют методы сегментации: метод гистограмм, метод преобразования Хафа, метод бинарного стека, метод жука, сегментации по связности.

Сравнив все методы было принято решение использовать следующие методы сегментации: преобразования Хафа и метод жука.

***Методы масштабирования***

Масштабирование изображения — изменение размера изображения с сохранением пропорций. Под масштабированием подразумевается, как увеличение, так и уменьшение разрешения изображения.

При этом, в зависимости от типа графики (растровая, векторная), масштабирование производится разными алгоритмами. Масштабирование векторных изображений происходит без потерь качества изображения, при увеличении растровых может происходить потеря качества изображения: возможны существенные искажения геометрии мелких деталей и появление ложных узоров на текстурах. Поэтому при масштабировании растровых изображений используются специализированные алгоритмы, сглаживающие нежелательные эффекты.

К существующим методам масштабирование можно отнести: методы ближайшего соседа, билинейная интерполяция, бикубическая интерполяция, и т.д.

Проанализировав изображения и используемые средства, был сделан вывод о том, что оптимальными для решения поставленной задачи будет выбор стандартного метода масштабирования с помощью масштабного коэффициента в среде canvas html5.

***Методы распознавания***

Распознавание образов – это отнесение исходных данных к определенному классу с помощью выделения существенных признаков, характеризующих эти данные из общей массы несущественных данных. В основе работы программ распознавания текста лежит достаточно серьёзный математический аппарат. И это оправдано сложностью задачи, особенно самого процесса распознавания.

Системы распознавания бывают следующих типов:

Системы распознавания объектов с учителем;

Системы распознавания объектов без обучения;

В системах распознавания без обучения первоначально заданной информации достаточно, чтобы разделить все множество объектов на классы в соответствии с каким-либо набором признаков.

Работу системы с учителем можно разделять на две части: обучение и распознавание. В ходе обучения системы ей предъявляются отдельные объекты с указанием того, к какому классу относится данный объект. Затем происходит распознавание: предъявляется неизвестный объект, и система сама должна определить его принадлежность. Это решение принимается на основании решающих правил.

Существует огромное количество методов распознавания в системах с учителем: метод потенциальных функций, метод эталонов, метод ближайшего соседа, метод ближайших соседей, нейронные сети.

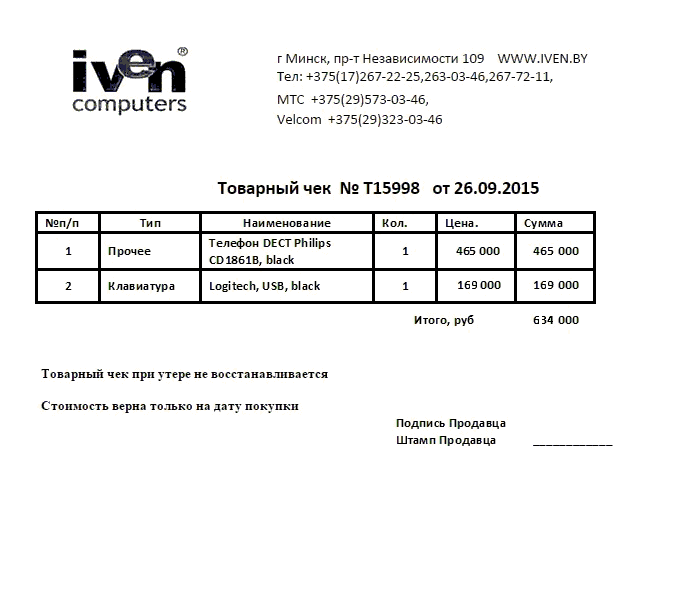
Сравнив все методы было принято решение использовать метод нейронных сете.

# Постановка задачи

Целью курсового проекта является разработка приложения, позволяющего распознать цены товаров на изображении гарантийного талона. Приложение реализовано в среде MS Visual Studio на языке Javascript для различных операционных систем.

Входные данные:

* сканированное изображение гарантийного талона (рис. 2.1):



*Рис. 2.1 Сканированное изображение гарантийного талона*

Ограничения:

* Изображение должно быть полутоновым;
* Размер сканированного изображения гарантийного талона должен быть в пределах: 600 х 600 - 1000 х 1000;
* Формат изображения bmp;
* Изображение должно быть ориентировано как на рис 2.1;
* На изображении обязательно должна располагаться одна таблица, с количеством столбцов не менее двух.
* Ширина таблицы равна не менее 80% от ширины исходного изображения. Ориентация наклона таблицы как на рисунке 2.1.

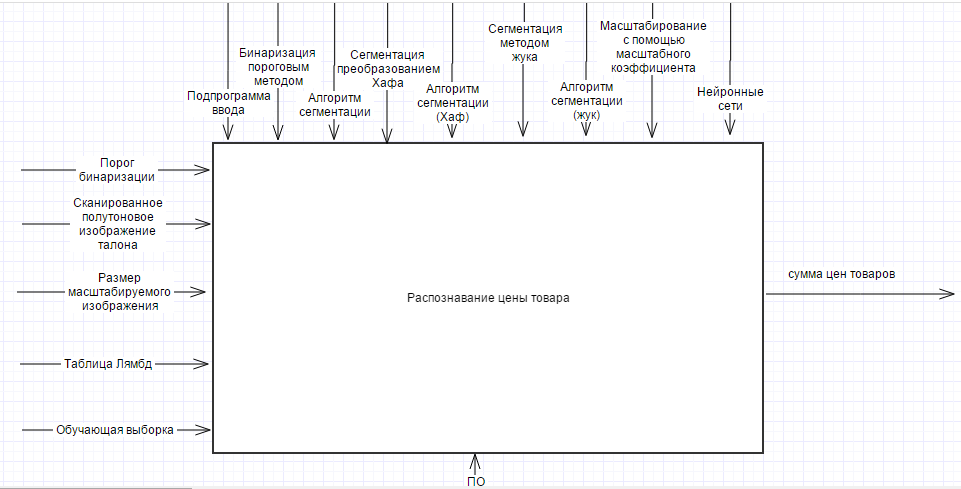
Выходные данные:

* Суммарная цена товаров

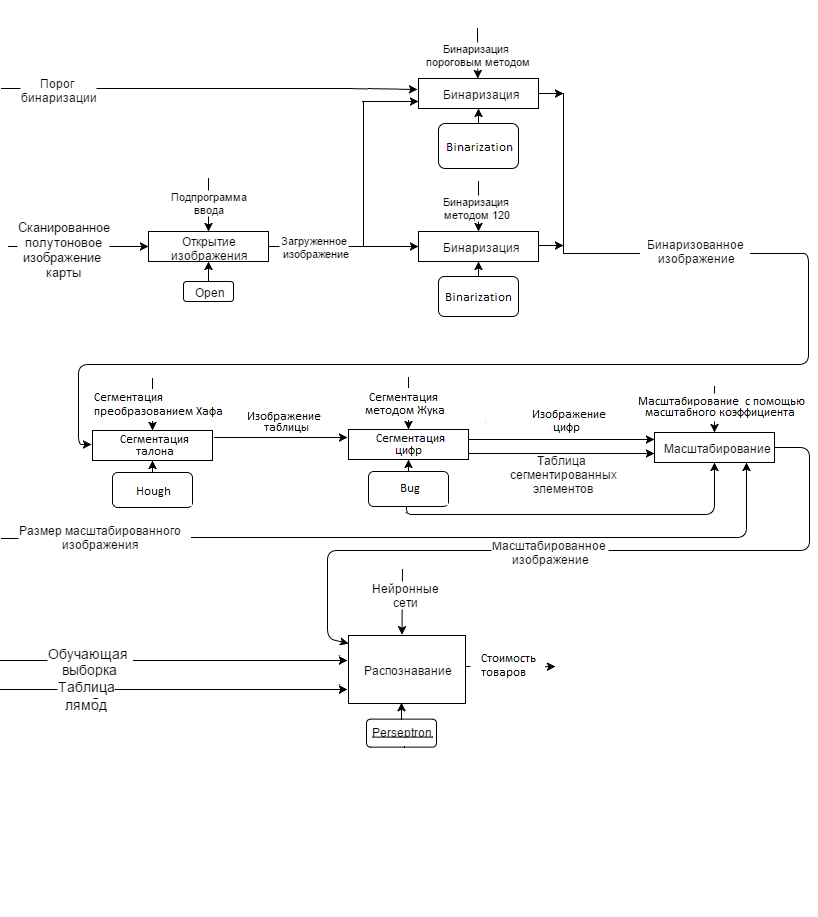
При выполнении курсовой проекта необходимо было решить следующие задачи по обработке изображения:

* Открытие изображения;
* Бинаризация пороговым методом (эмпирически был установлен рекомендуемый порог равный 135);
* Бинаризация методом «120»;
* Сегментация таблицы преобразованием Хафа;
* Сегментация цифр цен товаров методом Жука;
* Масштабирование посредством масштабируемого коэффициента;
* Распознавание с помощью нейронных сетей;

На основании этого были построены функциональные схемы приложения, в формате IDEF приведены на рис. 2.2 и рис. 2.3.



*Рис. 2.2 Схема IDEF (Первый уровень)*



*Рис. 2.3 Схема IDEF (Второй уровень)*

Среда разработки – MS Visual Studio.

Операционная система – Windows 8.1.

# 3.Адаптация и описание используемых алгоритмов

## ****3.1 Бинаризация изображения гарантийного талона****

В рамках данного проекта реализованы два алгоритма бинаризации: по порогу и метод «120».

* + 1. Пороговый метод

Если значения объектов и фона достаточно однородны, то можно использовать одно пороговое значение для всего изображения (глобальное). В этом случае задается значение яркости, которое выступает в качестве порога. Все пиксели полутонового изображения обращаются в 1, если их яркость меньше порога, остальные 0:

Пользователь вводит порог t от 0 до 255. Далее интенсивность каждого исследуемого пикселя вычисляется по формуле:

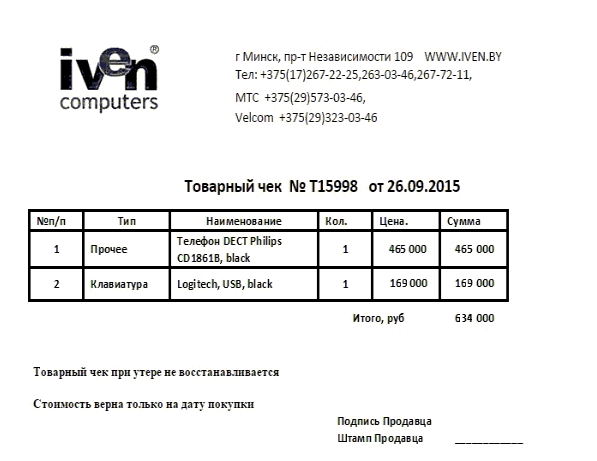
где:

яркость пикселя на исходном изображении,

– яркость пикселя на новом изображении,

– пороговое значение.

Результат пороговой бинаризации представлен на рис. 3.1:



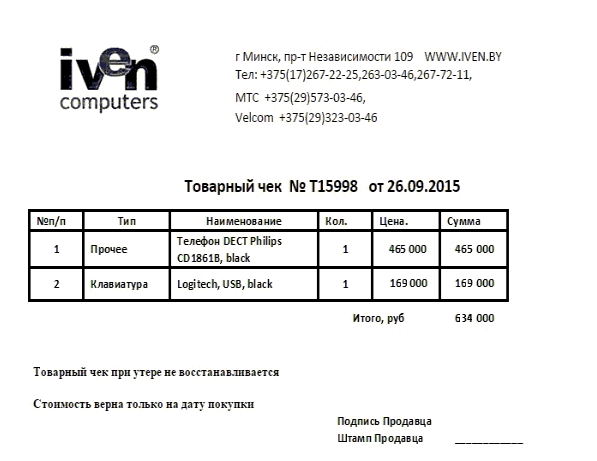


*Рис. 3.1 Бинаризация пороговым методом (порог 135)*

* + 1. Метод «120»

1. Строится гистограмма
2. Находится максимальный уровень серого в диапазоне t = [0, 120]
3. Черный цвет устанавливается у текущих пикселей, яркость которых меньше либо равна порогового значения плюс 13…15, в остальных случаях цвет белый.

Результат бинаризации методом «120» представлен на рис. 3.2:

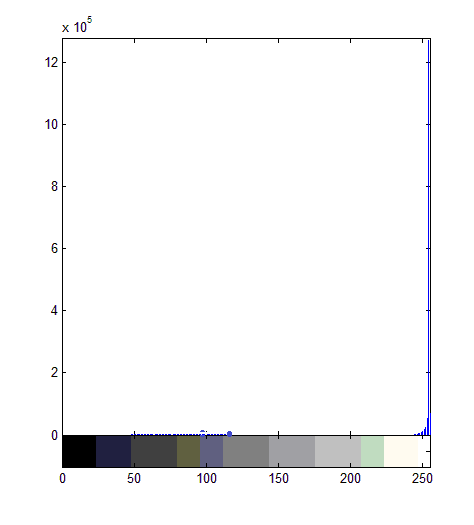




*Рис.3.2 Бинаризация методом 120 (порог 119)*

Следует отметить, что при использовании различного метода бинаризации, а также различного порога можно получить немного разные результаты.

Так же было определено что лучшим методом бинаризации в данном случае будет метод “120”, это можно увидеть исходя из гистограммы рис. 3.3.:



*Рис. 3.3 Гистограмма гарантийного талона*

## 3.2. Сегментация

3.2.1 Сегментация изображения

В данном проекте применяется два метода сегментации бинарного изображения : методом Жука и преобразованием Хафа.

Работа алгоритма жука проходит следующим образом: жук начинает движение с белой области по направлению к чёрной, как только жук встречает чёрный пиксель, он поворачивает налево и переходит к следующему пикселю, если следующий пиксель чёрный жук поворачивает налево, если белый – направо и так до тех пор, пока жук не вернётся в исходную точку. Точка перехода с чёрного на белый и сбелого на чёрный является точкой контура. Начинаем двигаться с лева на право, сверху вниз.

Работа преобразования Хафа: в простейшем случае оно является линейным преобразованием для обнаружения прямых. Прямая может быть задана уравнением y = mx + b и может быть вычислена по любой паре точек (x, y) на изображении. Главная идея преобразования Хафа — учесть характеристики прямой не как уравнение, построенное по паре точек изображения, а в терминах её параметров, то есть m — коэффициента наклона и b — точки пересечения с осью ординат. Исходя из этого прямая, заданная уравнением y = mx + b, может быть представлена в виде точки с координатами (b, m) в пространстве параметров.

Однако прямые, параллельные осям координат, имеют бесконечные значения для параметров m и b. Поэтому удобней представить прямую с помощью других параметров, известных как r и  θ .Параметр r — это длина радиус-вектора ближайшей к началу координат точки на прямой (т.е. нормали к прямой, проведенной из начала координат), а  θ— это угол между этим вектором и осью абсцисс. При таком описании прямых не возникают бесконечные параметры. Таким образом, уравнение прямой можно записать как:



Следовательно первоначально мы проходим по пикселям изображения и забираем данные о координатах черных пикселей, создавая отдельную таблицу с этими данными. Следующий шаг – это подсчет длин радиус-вектора координат точек, также создается отдельная таблица с данными.

Угол θ принимаем в радианах, различному значению которого соответствует поиск различных типов расположения линий на изображении.

Под углом 0,180 градусов. – задаем значение радиан 1.57





Рис 3.4 Гистограмма и результат сегментации таблицы при θ = 1.57 рад

Под углом -45 градусов значение радиан 0.785

Под углом 45 градусов значение радиан 2.355

Под углом 90, -90 градусов 0, 3.14



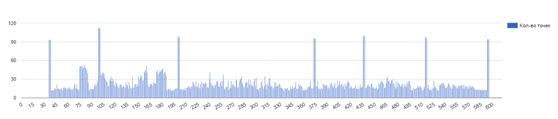


Рис 3.5 Гистограмма и результат сегментации таблицы при θ = 0 рад



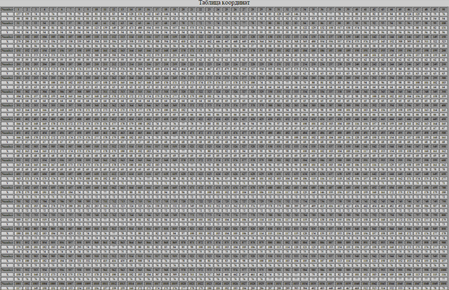


Рис 3.6.Таблица пространства Хафа и таблица координат черных пикселей.

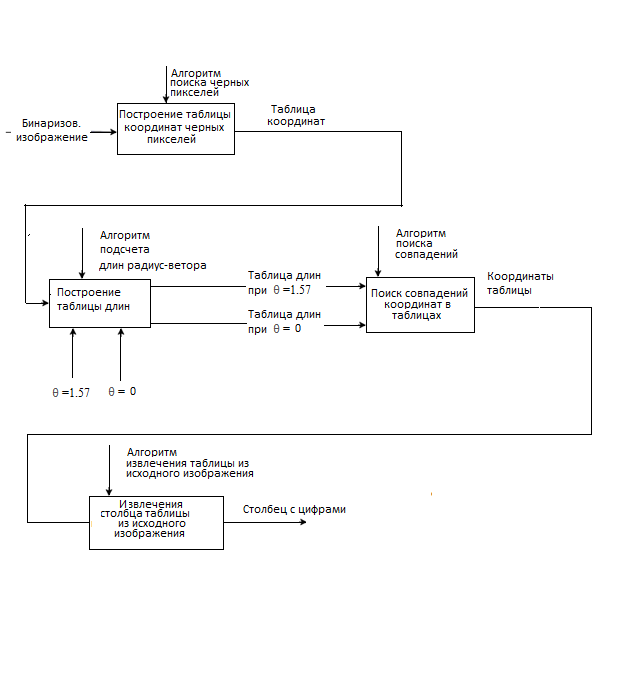
После этого приступаем к построению гистограммы(рис 3.5 и рис 3.6), на которой изображено количество пикселей в зависимости от значения длин радиус-вектора.

Самые большие значения на гистограмме и будут относиться к нашей сегментируемой таблице. Берем первое полученное значительно отличающиеся от остальных значение количества повторяющихся длин радиус-вектора и находим у этих точек минимальное и максимальное значение координаты по вертикали (при угле 0 рад), а также по горизонтали (при угле 1.57 рад) – благодаря чему и получаем координату верхнего левого пикселя таблицы и нижнего правого пикселя таблицы.

В результате реализации алгоритма жука был сделан вывод, что данный метод долго работает на большом изображении и может частично сегментировать таблицу, в связи с чем для сегментации таблицы будем использовать преобразования Хафа, а метод Жука можно использовать для сегментации цифр в стоимости товара рис. 3.7., после чего каждый выделенный объект соответствует отдельной цифре.

*Рис. 3.7 Сегментация цифр стоимости товара.*

**

*Рис. 3.8 Схема IDEF (Третий уровень)преобразование Хафа*

## 3.3 Масштабирование цифры

В рамках данного проекта реализовано масштабирование с помощью масштабируемого коэффициента.

Этот метод добавляет преобразование масштабирования на холсте единиц х по горизонтали и по вертикали у. Мы используем его, чтобы увеличить или уменьшить количество единиц в нашей сетке холста. Это может быть использовано для отрисовки уменьшения или увеличения формы изображений. По умолчанию, один блок на холсте равен ровно одному пикселю. Если применить, например, масштабный коэффициент, равный 0,5, в результате блок станет равен 0,5 px и так форма блока будет составлен на половине размера. Аналогичным образом установка коэффициента масштабирования 2.0, чтобы увеличить размер блока в 1px, мы изменяем его ширину и высоту в 2 раза и теперь блок становится равным 2 px.

Результат масштабирования с помощью масштабируемого коэффициента рис. 3.8:



*Рис. 3.8 Масштабирование с помощью масштабируемого коэффициента*

Размер масштабированного изображения устанавливается 20 х 20.

## 3.4 Распознавание цифры

В этом разделе пойдёт речь о том, как составить для каждого изображения составить набор информативных признаков и затем по этим признакам отнести изображения к тому или иному классу.

3.4.1 Распознавание с помощью нейронных сетей

Одним из методов распознавания в данном курсовом проекте является распознавание с помощью персептрона. Крупный толчок развитию нейросетей дал нейрофизиолог Френк Розенблатт, предложивший модель узнающей машины, названную им “Персептрон” (от латинского percepto - понимаю, познаю). При ее разработке он исходил из некоторых принятых в его время представлений о структуре мозга и зрительного аппарата. По мнению Ф.Розенблатта, персептрон являлся, прежде всего, и, главным образом, моделью мозга, а не устройством, служащим только для распознавания образов. Стремясь воспроизвести функции человеческого мозга, Ф.Розенблатт использовал простую модель биологического нейрона и систему связей между ними. Воспринимающим устройством персептрона служит фотоэлектрическая модель сетчатки – поле рецепторов, состоящее из нескольких сотен фотосопротивлений (S- элементов).

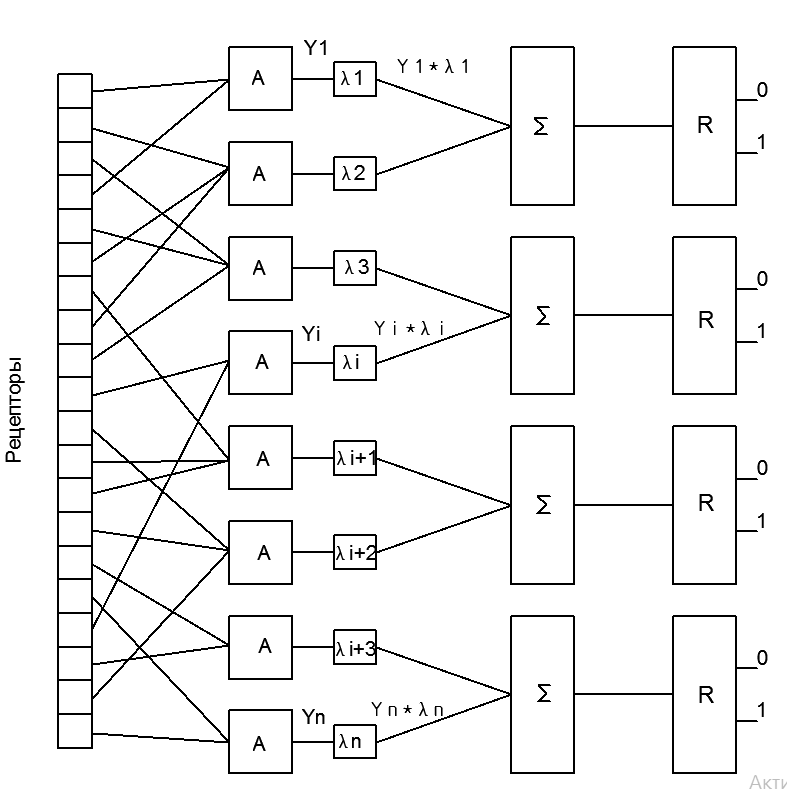


Рис. 3.9 – Структурная схема персептрона с одним выходом

Каждый элемент поля рецепторов может находиться в двух состояниях – возбужденном или невозбужденном состоянии. На выходе каждого элемента появляется сигнал xi (i = 1, 2, …, n, где n- число элементов), равный единице, если элемент возбужден, и нулю – в противном случае. Следующей ступенью персептрона служат, так называемые, ассоциативные элементы или А-элементы. Каждый А-элемент имеет несколько входов и один выход. При подготовке персептрона к эксперименту к входам А-элемента подключаются выходы рецепторов, причем подключение любого из них может быть произведено со знаком плюс или со знаком минус. Выбор рецепторов, подключаемых к данному А-элементу, также как и выбор знака подключения, производится случайно. В ходе эксперимента связь рецепторов с А-элементами остается неизменной.

А-элементы производят алгебраическое суммирование сигналов [2], поступивших на их входы, и полученную сумму сравнивают с одинаковой для всех А-элементов величиной ϑ. Если сумма больше ϑ, А-элемент возбуждается и выдает на выходе сигнал, равный единице. Если сумма меньше ϑ, А-элемент остается невозбужденным и выходной его сигнал равен нулю. Выходные сигналы Α-элементов с помощью специальных устройств (усилителей) умножаются на переменные коэффициенты λj. Каждый из этих коэффициентов может быть положительным, отрицательным или равным нулю и меняться независимо от других коэффициентов. Выходные сигналы усилителей суммируются, и суммарный сигнал поступает на вход, так называемого, реагирующего элемента или R-элемента. Если σ положительна или равна нулю, R-элемент выдает на выходе единицу, а если σ отрицательна – ноль. Таким образом, выходной сигнал каждого R элемента равен 0 или 1.

Алгоритм обучения учитывающий правильность ответа в ходе обучения: при использовании этого алгоритма коэффициенты λj остаются неизменными, если персептрон правильно распознал предъявленную ему в данном такте фигуру. Изменение коэффициентов λj производится только в тех тактах, в которых персептрон «ошибся». Изменение коэффициентов производится таким образом, чтобы повысить правильность ответов персептрона. Если, например, персептрон «ошибся» и при предъявлении объекта образа А выдал на выходе объект А, коэффициенты возбужденных А-элементов увеличиваются. Если был предъявлен объект образа В, а персептрон выдал на выходе объект А, коэффициенты возбужденных А-элементов уменьшаются. [2]

На рисунке 3.10 показано подключение Х элементов. Здесь каждая таблица описывает R элемент, а 1 в строке с номером X элемента показывает к какому А элементу подключён данный Х элемент.

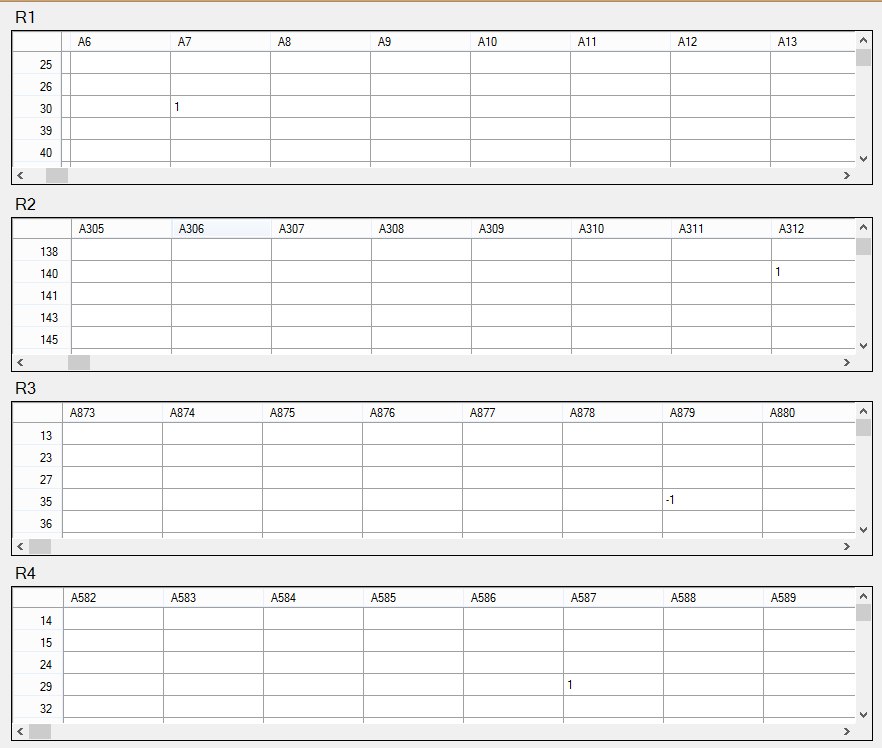


Рис. 3.10 – Подключение X элементов.

На 3.11 показаны значения лямбд для конкретного А элемента обученного персептрона.

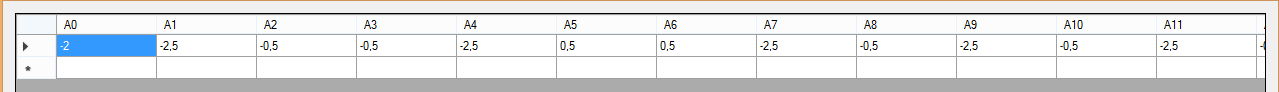


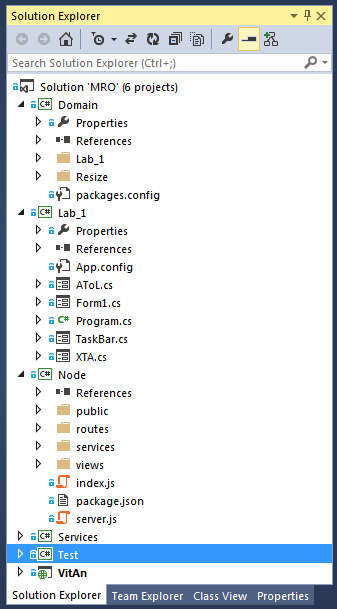
Рис. 3.11 – Значения лямбд А элементов.

4.Программная реализация

Программа разработана на языке программирования Javascript в среде разработки Visual Studio 2015.

## 4.1 Схема программного обеспечения

Схема приложения представлена на рис. 4.1:



*Рис. 4.1 Схема приложения*

## 4.2 Описание используемых классов, функций и методов

## Основные классы С#:

## Perceptron – инкапсулирует функциональность персептрона

## Основные файлы js:

## bug\_v.js – содержит реализацию алгоритма сегментации методом жука;

## hough.js – содержит реализацию алгоритма преобразования Хафа;

## binarization.js – содержит реализацию алгоритма пороговой бинаризации и бинаризации методом 120;

main.js – основной файл отвечающий за логику меню, переключений кнопок,

chartJs.js - содержит реализацию построения гистограмм.

Table.js – отвечает за формирование таблиц пространства Хафа и таблицы координат.

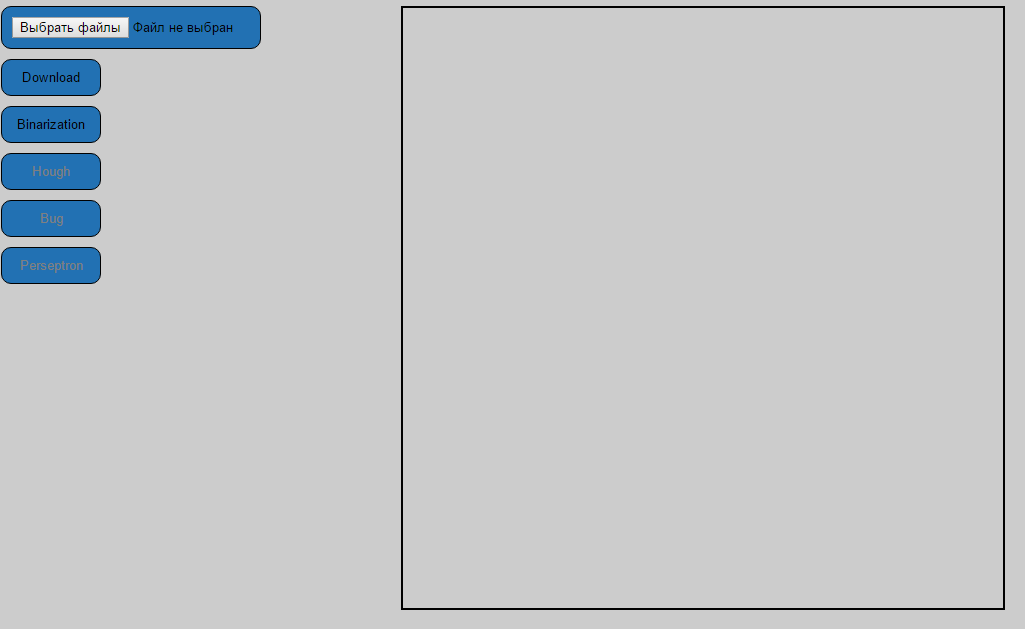
Основные функции:

getArrayPixel() – возвращает массив пикселей исходного изображения в двумерном массиве;

getXandY(arr) – задает координаты текущего черного пикселя в поиске.

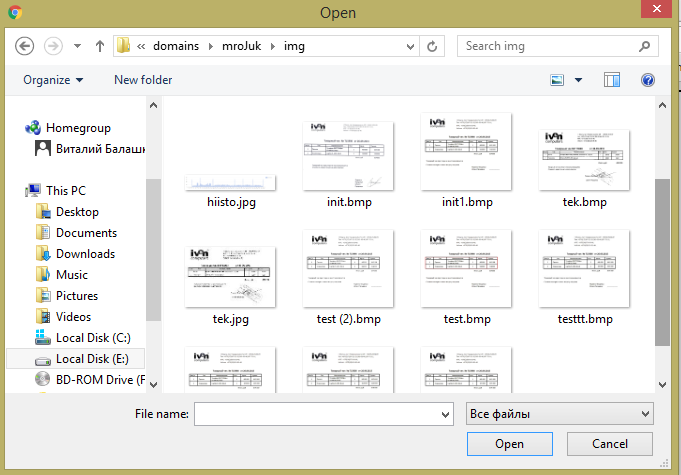
## 4.3 Тестирование и руководство пользователя

Запускаем приложение. Вид окна при запуске показан на рис. 4.2.



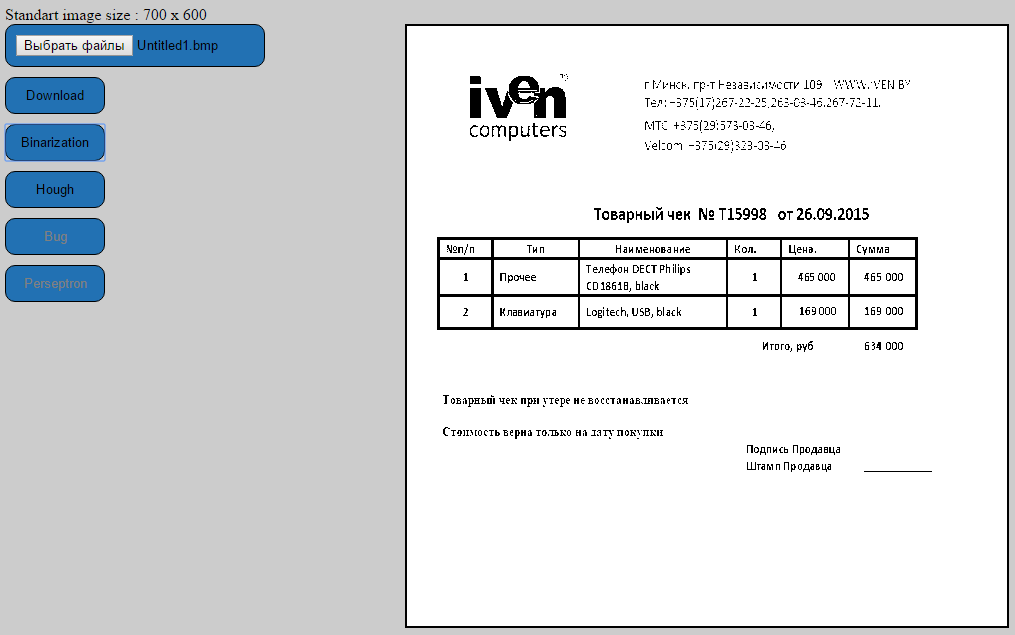
*Рис. 4.2 Вид окна при запуске*

Для начала необходимо загрузить изображение. Нажимаем на кнопку **«Выбрать файл»,** откроется окно выбора изображения (рис. 4.3), выбираем изображение и нажимаем **«Открыть».**



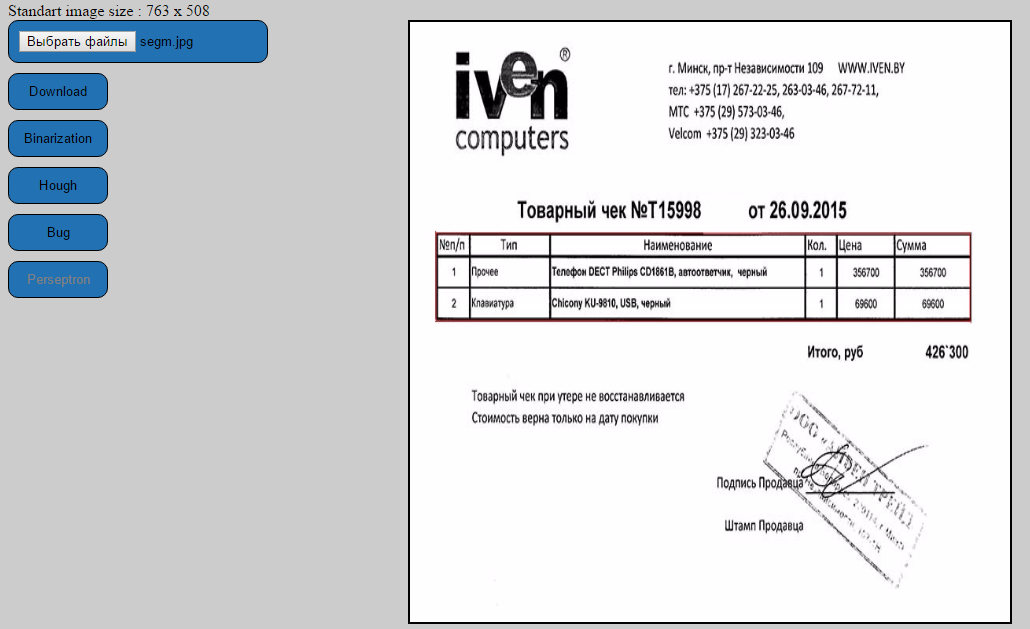
*Рис. 4.3 Выбор изображения*

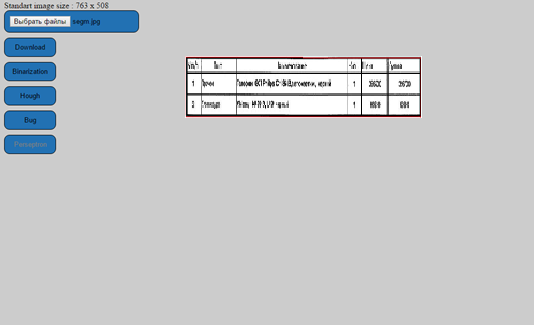
Далее необходимо провести бинаризацию изображения (рис. 4.3). Нажимаем на кнопоку: **«Binarization»** и происходит бинаризация методом 120**.**

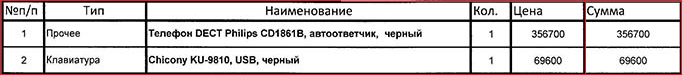


*Рис. 4.4 Бинаризированное изображение методом 120.*

Далее необходимо выполнить сегментацию изображения (рис. 4.4), это можно сделать, нажав кнопку **«Segmentation»**.







*Рис. 4. 5 Сегментированное изображение*

Далее необходимо выполнить сегментацию цифр стоимости товара (рис. 4.6), это можно сделать, нажав кнопку **«Bug»**.



*Рис. 4.6 Сегментированная цифра*

Теперь у нас последовательно сохраняются сегментированные изображения цифр в папку для распознаваемых изображений.

На данном этапе заканчиваются все подготовительные работы. Далее переходим непосредственно к распознаванию.

Далее действия определяются в соответствии с описанным алгоритмом работы персептрона с распознавания без учителя в разделе распознавание.. Цифры можно отправить на распознавание в персептрон нажав на кнопку “**Perseptron**” в результате получив (рис. 4.7):



*Рис. 4.8 Распознавание с помощью персептрона*

# Заключение

# В ходе работы над курсовым проектом было разработано веб приложение на языке Javascript, с помощью которого можно произвести бинаризацию, сегментацию, масштабирование и распознание с помощью нейронной сети.

# Цель курсового проекта была достигнута.

# В результате тестирования были сделаны следующие выводы:

Используется метод бинаризации 120, не происходит потеря пикселей в необходимом изображении.

# Для ускорения работы приложения рекомендуется уменьшить размер входного изображения. Метод преобразования Хафа помогает определить линии под различным углом, что помогает найти нашу таблицу(состоящую из горизонтальных и вертикальных линий).

# При распознавании методом жука столкнулись с проблемой: очень долго происходит процесс сегментации элементов на большом изображении поиска. Для корректной работы алгоритма жука необходимо применять его только для сегментации цифр.

Распознавание осуществляется с помощью нейронных сетей: персептрона с несколькими R- элементами, что не гарантирует 100% распознания цифр товара, т.к. процент распознавания данной нейронной сети не высок, около 60%.

# Литература

1. Конспект лекций по дисциплине «Методы распознавания образов», Ковалёва И.Л., 2013 г.
2. Статья «Бинаризация черно-белых изображений. Состояние и перспективы развития», Федоров А.

# Приложение 1

var count, img, width, height, thisX, thisY, titles, all, images, objPixels, countObj, pixel, arrayPixel, canvas, ctx, buffer, img,

count = 0; thisY = 0, thisX = 0, countObj = 0, objPixels = [], distanceR = [], sortCoordsValue = [] , numberOfValuesInArray = [],

histogram = [], colors = [];

var maxCoordValue;

var a = 0, x = 1, r = 1;

var titles = []; all = []; images = [];

Array.prototype.max = function() {

return Math.max.apply(null, this);

};

Array.prototype.min = function() {

return Math.min.apply(null, this);

};

canvas = document.getElementById('drawingCanvas');

ctx = canvas.getContext('2d');

width = 700;

height = 600;

img = new Image();

img.src = '004.bmp';

function addNewRow(array){

var tabl,str,length,count;

count=0;

length = array.length/2;

tabl = $('.tab tbody');

str ='<tr class="first"><th>'+'Number'+'</th>';

strX = '<tr><th>'+'X'+'</th>';

strY = '<tr><th>'+'Y'+'</th>';

for(var i=1;i<=length;i++){

str+='<th>'+i+'</th>';

strX+='<th>'+array[count]+'</th>';

strY+='<th>'+array[count+1]+'</th>';

count+=2;

if(i%50==0){

str+='</tr>';

strX+='</tr>';

strY+='</tr>';

tabl.append(str);

tabl.append(strX);

tabl.append(strY);

str ='<tr class="first"><th>'+'Number'+'</th>';

strX = '<tr><th>'+'X'+'</th>';

strY = '<tr><th>'+'Y'+'</th>';

}

}

str+='</tr>';

strX+='</tr>';

strY+='</tr>';

tabl.append(str);

tabl.append(strX);

tabl.append(strY);

}

function addNewRows(array,grad){

distanceR = [];

var tabl,str,length,count;

count=0;

length = array.length/2;

tabl = $('.tab2 tbody');

str ='<tr class="first"><th>'+'Point'+'</th>';

strX = '<tr><th>'+grad+'</th>';

for(var i=1;i<=length;i++){

distanceR[i-1] = Math.floor(array[count] \* Math.cos(grad) + array[count+1] \* Math.sin(grad));

str+='<th>'+i+'</th>';

strX+='<th>'+distanceR[i-1]+'</th>';

count+=2;

if(i%50==0){

str += '</tr>';

strX += '</tr>';

tabl.append(str);

tabl.append(strX);

str = '<tr class="first"><th>' + 'Point' + '</th>';

strX = '<tr><th>' + grad + '</th>';

}

}

str+='</tr>';

strX+='</tr>';

tabl.append(str);

tabl.append(strX);

}

function getMaxValue(value) {

for(var i = 0; i < distanceR.length; i++)

if (distanceR[i]==value) {

y = coordArray[(i-1)\*2+1];

x = coordArray[(i-1)\*2];

coord = y \* width \* 4 + x\*4;

pix[coord]=color[0];

pix[coord+1]=color[1];

pix[coord+2]=color[2];

}

}

var pixelCount = 1;

var coordArray = [];

var setCoord = (function() {

var count = 0;

return function(x,y){

coordArray[count] = x;

coordArray[count + 1] = y;

count += 2;

};

})();

function getNumberOfValuesInArray(array) {

var buff = 0 , count = 0 , length = array.length;

for(var i = 0 ; i < length ; i++) {

numberOfValuesInArray[array[i]]++;

}

return numberOfValuesInArray;

}

function init() {

for(var i = 0 ; i <= maxCoordValue ; i++) {

numberOfValuesInArray[i] = 0;

// colors[i] = '#001';

}

options.colors = '#001';

}

function setHistogram() {

for(var i = 0 ; i <= maxCoordValue ; i++)

histogram[histogram.length]=[i+'',numberOfValuesInArray[i]];

}

function search() {

getXandY(arrayPixel);

if(thisY<height && thisX<width){

setCoord(thisX,thisY);

pixelCount++;

}

}

img.onload = function () {

ctx.drawImage(img, 0, 0);

arrayPixel = getArrayPixel();

while(thisY<height && thisX<width) {

search();

}

addNewRow(coordArray);

//0 , 1.57 , 3.14

addNewRows(coordArray,0);

maxCoordValue = distanceR.max();

init();

getNumberOfValuesInArray(distanceR);

getArrayValueOfHistogram();

setHistogram();

google.setOnLoadCallback(drawChart);

};

function getArrayValueOfHistogram() {

var arr, length, count, value, count;

arr = [],arrayX = [], arrayY = [];

count=0;

length = numberOfValuesInArray.length;

for(var i = 0; i < length; ++i) {

if(numberOfValuesInArray[i]>300){

arr[count] = i;

++count;

}

}

var x,y,coord,array , arrayCount,l;

array = []; arrayCount = 0;

value = arr[0];

l = distanceR.length; console.log(value);

for(var i = 0; i < l; i++)

if (distanceR[i]==value) {

y = coordArray[(i-1)\*2+1];

x = coordArray[(i-1)\*2];

arrayX[arrayCount] = x;

arrayY[arrayCount] = y;

arrayCount++;

}

sortArrayStep(arrayX,arrayY);

console.log(arrayX);

console.log(arrayY);

return arrayX;

}

function sortArrayStep(array,array2) {

l = array.length;

for(var i = 2;i < l; ++i){

if(i==2 && (array[i-2]!=(array[i-1])-1)){

array = array.slice(i-2,l);

i--;

}

if((array[i-2]!=(array[i-1])-1) && (array[i-1]!=(array[i])-1)) {

array = array.slice(i,l);

i--;

// array2.slice(i-1,1);

}

}

}

function getArrayPixel(){

var data, light, arr, pix, time1;

picLength = width \* height;

arr=[];

console.log("getArrayPixel - Start");

time1 = new Date();

for( i = 0;i<height;i++)

arr[i]=[];

for(var i = 0;i<height;i++){

for(var j = 0;j<width;j++){

pix = ctx.getImageData(j,i, 1, 1);

data = pix.data;

light = (data[0] + data[1] + data[2] ) /3;

arr[i][j]=Math.floor(light);

}

}

timeLog(time1);

console.log("getArrayPixel - End");

return arr;

}

function timeLog(time1){

var time2, hour, minute;

time2 = new Date();

hour = time2.getHours()-time1.getHours();

minute = time2.getMinutes()-time1.getMinutes();

console.log(hour+" : "+minute);

}

function setArrayPixel(array){

var data, light, arr, pix;

picLength=width\*height;

arr=[];

data = ctx.getImageData(0,0,width,height);

pix = data.data;

var r,g,b,o;

r=0,g=1,b=2,o=3;

for(var i = 0;i<height;i++){

for(var j = 0;j<width;j++,r+= 4,g+= 4,b+= 4,o+= 4){

if(objPixels[countObj].rgb[0]==array[i][j]){

arr[r]=objPixels[countObj].rgb[0];

arr[g]=objPixels[countObj].rgb[1];

arr[b]=objPixels[countObj].rgb[2];

arr[o]=255;

}

else{

arr[r]=array[i][j];

arr[g]=array[i][j];

arr[b]=array[i][j];

arr[o]=255;

}

}

}

for(var i = 0,j = 1,n = 2;i<picLength\*4;i += 4,j += 4,n += 4){

pix[i]=arr[i];

pix[j]=arr[j];

pix[n]=arr[n];

}

data.data=pix;

putColorData(data);

}

function getXandY(arr){

for(thisY; thisY < height; thisY++){

for(thisX; thisX < width;){

thisX++;

if(arr[thisY][thisX] == 0){

return ;

}

}

thisX = -1;

}

thisY = height;

thisX = width;

};

function setpixel(value){

color = [255,0,0];

var x,y,coord;

var data = ctx.getImageData(0,0,width,height);

pix = data.data;

for(var i = 0; i < distanceR.length; i++)

if (distanceR[i]==value) {

y = coordArray[(i-1)\*2+1];

x = coordArray[(i-1)\*2];

coord = y \* width \* 4 + x\*4;

pix[coord]=color[0];

pix[coord+1]=color[1];

pix[coord+2]=color[2];

}

data.data = pix;

putColorData(data);

}

function putColorData(img){

ctx.putImageData(img, 0, 0);

}

function downloadCanvas(link, filename) {

link.href = canvas.toDataURL();

link.download = filename;

}

document.getElementById('download').addEventListener('click', function() {

downloadCanvas(this, 'test.bmp');

}, false);

histogram=[

['Значение R', 'Кол-во точек'],

];

google.load("visualization", "1", {packages:["corechart"]});

var options = {

title: 'Гистограмма',

hAxis: {title: 'Значение R'},

vAxis: {title: 'Кол-во точек'},

colors: [],

};

function drawChart() {

var data = google.visualization.arrayToDataTable(

histogram);

var options = options;

var chart = new google.visualization.ColumnChart(document.getElementById('oil'));

chart.draw(data, options);

}