*МИНОБРНАУКИ РОССИИ*

*Федеральное государственное бюджетное образовательное*

*учреждение высшего образования*

*«Ярославский государственный университет им. П.Г. Демидова»*

***Кафедра компьютерной безопасности и математических методов обработки информации***

***Дипломная работа***

*Разработка системы альтернативного управления компьютером*.

***(Специальность 10.05.01 Компьютерная безопасность)***

***Научный руководитель***

**Д-р наук, профессор**

**(степень, звание)**

**В.Г. Дурнев**

**(подпись) (ФИО)**

***«\_\_\_» \_\_\_\_\_\_\_\_\_ 20\_\_ г.***

***Студент группы КБ-61СО***

С.М. Соловьев

**(подпись) (ФИО)**

***«\_\_\_» \_\_\_\_\_\_\_\_\_ 20\_\_ г.***

***Ярославль 2017 г.***

Оглавление

[Введение 5](#_Toc503643687)

[Глава 1. Введение в биологическую терминологию 8](#_Toc503643688)

[1.1 Структура человеческого глаза 8](#_Toc503643689)

[1.2 Фиксация глаза 8](#_Toc503643690)

[1.3 Перемещения глаза 9](#_Toc503643691)

[Глава 2. Формирование требований к системе 11](#_Toc503643692)

[2.1 Обзор существующих систем альтернативного управления ПК 11](#_Toc503643693)

[2.1.1 Tobii eye tracker 4c 11](#_Toc503643694)

[2.1.2 Microsoft Kinect. 12](#_Toc503643695)

[2.2 Основные требования к системе. 12](#_Toc503643696)

[2.2.1 Основные понятия. 13](#_Toc503643697)

[2.2.2Приблизительная оценка точности алгоритма. 13](#_Toc503643698)

[2.3 Вывод. 15](#_Toc503643699)

[Глава 3. Методы поиска объектов на изображении 16](#_Toc503643700)

[3.1 Нейронные сети 16](#_Toc503643701)

[3.1.1 Введение в терминологию 16](#_Toc503643702)

[3.1.2 «Классические» нейронные сети 17](#_Toc503643703)

[3.1.3 Алгоритмы обучения 21](#_Toc503643704)

[3.1.4 Сверточные нейронные сети 25](#_Toc503643705)

[3.2 Метод Виолы-Джонса 26](#_Toc503643706)

[3.2.1 Принцип сканирующего окна 27](#_Toc503643707)

[3.2.2 Интегральное представление изображений 27](#_Toc503643708)

[3.2.3 Признаки Хаара 29](#_Toc503643709)

[3.2.4 Используемая в алгоритме модель машинного обучения 33](#_Toc503643710)

[3.2.5 Обучение классификатора в методе Виолы-Джонса 34](#_Toc503643711)

[3.3 Вывод 35](#_Toc503643712)

[Глава 3. Алгоритмы распознавания взгляда 36](#_Toc503643713)

[3.1 Инфракрасное отражение 36](#_Toc503643714)

[3.2 Алгоритм отслеживания взгляда в реальном времени с компенсацией движения головы(RTAC) 37](#_Toc503643715)

[Глава 4. Концепция приложения 39](#_Toc503643716)

[4.1 Технические проблемы 39](#_Toc503643717)

[4.2 Алгоритм 39](#_Toc503643718)

[4.3 Получение координат частей лица 42](#_Toc503643719)

[4.4 Определение угла положения головы 42](#_Toc503643720)

[4.5 Определение направления взгляда 43](#_Toc503643721)

[4.6 Объединение и аппроксимирование результата 44](#_Toc503643722)

[4.6.1 Упрощенный способ объединения сигналов 45](#_Toc503643723)

[4.6.2 Модифицированный способ объединения сигналов 45](#_Toc503643724)

[4.6.3 Итеративное сглаживание результата 47](#_Toc503643725)

[4.7 Итеративное сглаживание результата 48](#_Toc503643726)

[Глава 5. Реализация проекта 49](#_Toc503643727)

[5.1 Поиск пары глаз 49](#_Toc503643728)

[5.2 Определение зрачка 50](#_Toc503643729)

[5.2.1 Бинаризация изображения 50](#_Toc503643730)

[5.3 Поиск угла 53](#_Toc503643731)

[Глава 6. Экспериментальная оценка эффективности системы 55](#_Toc503643732)

[Заключение 55](#_Toc503643733)

[Список литературы 56](#_Toc503643734)

[ПРИЛОЖЕНИЕ А РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ ЭКСПЕРИМЕНТАЛЬНОГО ОБРАЗЦА СИСТЕМЫ 59](#_Toc503643735)

[А.1 Тестирование влияния внешних факторов на точность вычисления координат 59](#_Toc503643736)

[А.2 Тестирование влияния разрешения входного изображения 60](#_Toc503643737)

[А.3 График влияния разрешения входного изображения 61](#_Toc503643738)

# Введение

За последние десять лет персональный компьютер стал обыденностью, а управление им – привычным делом для многих. Заслугой этого можно считать простые и удобные механизмы взаимодействия пользователя с устройством. Попутно с этими механизмами развиваются и технологии альтернативного управления.

Принцип работы «бесконтактных» систем управления заключается в том, что они позволяют человеку взаимодействовать с каким-либо устройством используя нестандартный метод управления. Например, в смартфонах Apple(iPhone) есть голосовой помощник, в автомобилях Mercedes система управления жестами.

С технической точки зрения, система альтернативного управления состоит из считывающего устройства, регистрирующего события, и вычислительного устройства, интерпретирующего полученную информацию в наборы команд для системы. Преимущество бесконтактных систем заключается в альтернативе управления устройством, когда использование традиционных методов затруднено или невозможно.

На сегодняшний день системы альтернативного управления персональным компьютером востребованы на рынке технологий. Все они отличаются по следующим основным параметрам:

1. Характер исходных сигналов. В качестве примера можно привести управление жестами, голосовыми командами, направлением взгляда.
2. Считывающее устройство. Системы с одним характером исходных сигналов могут регистрировать их с помощью разных типов устройств.
3. Расстояние от пользователя до регистрирующего устройства. *Инвазивно* или *неинвазивно* устройство для считывания исходных сигналов.
4. Физическое поведение пользователя. В зависимости от системы, оператор может свободно перемещаться или должен быть зафиксирован во время управления.
5. Точность системы распознавания команд.

Актуальность дипломной работы заключается в конечной стоимости продукта, реализованного с использованием веб-камеры вместо дорогого оборудования. Намного больше, чем на текущий момент, людей с ограниченными возможностями смогут обрести возможность общаться, обучаться, развиваться и работать с другими людьми.

Цель работы.

Главная цель работы – реализация системы альтернативного (бесконтактного) управления персональным компьютером, основанная на определении направления взгляда оператора по изображению.

Задачи:

* рассмотреть алгоритмы подходящие для реализации,
* предложить свой алгоритм определения направления взгляда,
* проанализировать влияние внешних факторов на точность устройства,
* понять трудности создания подобных систем,
* создать базовое приложение для определения направления взгляда,
* **[Получить обратную связь от потенциальных пользователей???]**

Научная новизна работы.

1. Предложена архитектура системы бесконтактного управления.
2. Разработан и реализован алгоритм определения углового положения головы и направления взгляда пользователя.
3. Предложен способ экспериментальной оценки эффективности, точности, влияния внешних факторов.
4. Экспериментальным путем собраны данные по эффективности работы системы, построен отчет и сделаны выводы.

Практическая значимость исследования состоит в том, что полученный прототип базовой системы может быть использован для разработки действующего образца системы бесконтактного управления, которую можно внедрить в самых различных областях, от обучения и медицины до маркетинговых исследований.

Достоверность результатов подтверждена неоднократной серией экспериментов с фиксацией полученных данных. Для проведения испытаний был разработан специальный демонстрационный стенд. Результаты данных экспериментов приведены в приложении.

# Глава 1. Введение в биологическую терминологию

Используя ПК или мобильное устройство, человек устремляет взгляд на конкретные секторы экрана. Фиксацию взгляда на конкретном секторе будем называть фокусом взгляда. Для управления ПК требуется постоянный трекинг (слежение) фокуса взгляда. Для понимания физики процесса требуется информация о биологии и поведении глаза. Данный раздел предоставит некоторые базовые знания для создания системы глазного трекинга.

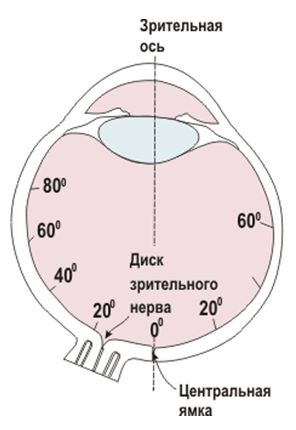
## 1.1 Структура человеческого глаза

Для того чтобы человек мог максимально четко рассмотреть объект, ему требуется сфокусировать на нем взгляд. Получение изображения происходит при попадании его через хрусталик на рецепторы глаза. Важным фактором при рассмотрении структуры глаза является то, где располагаются эти рецепторы. В [2] говорится, что если бы рецепторы были расположены равномерно внутри глаза, то была бы проблема понять, как происходит фокусирование. Вместо этого мы имеем различную плотность распределения рецепторов. Наиболее четкое изображение человек получает при попадании его в точку максимального скопления фоторецепторов-колбочек, и эта точка находится в желтой ямке (пятно фовеа).

## **1.2 Фиксация глаза**

Но, нельзя забывать, что человеческий зрачок не может замереть на месте для получения стабильного изображения. В [1] используют понятие дрейфа. Дрейф представляет собой неупорядоченное и относительно медленное движение осей глаз, при котором для каждого глаза изображение точки фиксации остается внутри фовеа. Дрейф всегда сопровождается тремором – высоким по частоте, но очень маленьким по амплитуде колебательным движением осей глаз. Проще говоря, глаз даже при фокусировке изображения немного дергается, незаметно для самого человека.

Так как желтая ямка находится внутри глаза и имеет крайне малый размер (диаметр чуть больше 1мм), то фиксировать изменение ее положения не предоставляется возможным. Вместо этого можно отслеживать зрачок и через него выводить точку фокусировки, это становится возможным потому что, объект фокуса находится в центре глаза.



*Рис.1. Локализация центральной ямки (0 градусов) и различных областей сетчатки по отношению к ней (в градусах эксцентриситета).*

## **1.3 Перемещения глаза**

Человеческий глаз может прибывать только в двух состояниях: фокусировка, сопровождающаяся дрейфом и перемещение (саккада).

Саккады (от старинного французского слова, переводимого как «хлопок [паруса](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D1%83%D1%81)») — быстрые, строго согласованные движения глаз, происходящие одновременно и в одном направлении. В работе [3] было обнаружено, что «трансаккадная пространственная память приносится в жертву, чтобы поддерживать перцептивную стабильность»[3]. Это означает, что во время саккадного движения визуальная информация не получается, чтобы сохранить восприятие гладкости и стабильности. Этот эффект известен как саккадическое подавление. Подобным эффектом обладает постоянное использование мыши в качестве интерфейса управления ПК, оно оттачивает навык до сглаженных, точных перемещений курсора в нужное положение.

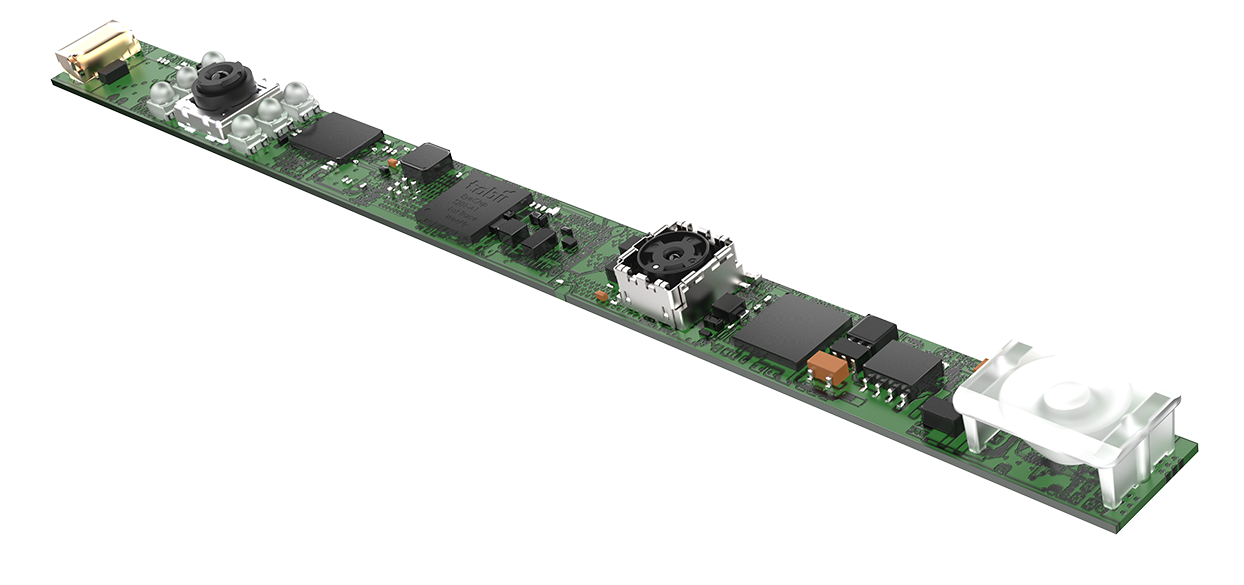
Саккады важно отличать от других движений глаз, называемых Saccadic Eye Movements (SACs) в [5] (Преследование, вергентное и вестибулярное). Преследование – движения глаз «преследуют» точку фокусировки, перемещающийся в пространстве, выполняя саккады, чтобы догнать, если точка движется слишком быстро. Движение заметно медленнее, чем у саккад. Вергентные движения (лат. *Vergo* - склоняюсь) - [макродвижения глаз](http://dic.academic.ru/dic.nsf/psihologic/995), приводящие к изменению угла между зрительными осями левого и правого глаза. Вергентные движения – выступают фактором, обеспечивающим [бинокулярное зрение](http://dic.academic.ru/dic.nsf/psihologic/253). Они разделяются на движения конвергенции – сводящие глаза, дивергенции –разводящие глаза, циклофузионные, или [торсионные движения](http://dic.academic.ru/dic.nsf/psihologic/1898), меняющие ориентации сетчаток левого и правого глаз. Вестибулярные движения - это вращения глаза, чтобы компенсировать большие движения головы или тела (стабилизация изображения). Эти три движения менее важны, чем саккады для обработки информации, и как таковые для отслеживания зрачков.

# Глава 2. Формирование требований к системе

## 2.1 Обзор существующих систем альтернативного управления ПК

Самые точные методы для обнаружения глаз включают в себя инвазивное или дорогостоящие оборудование. В [2] упоминается метод, который использует контактную линзу, с металлическим кольцом. Фиксация координат происходит с помощью измерения магнитного поля. В [6] в качестве системы фиксации изображения используют шлем с ИК камерами. Минусы данных систем заключаются в том, что они неудобны для использования, не мобильны. Ниже будут рассмотрены самые популярные неинвазивные системы на рынке технологий.

### 2.1.1 Tobii eye tracker 4c

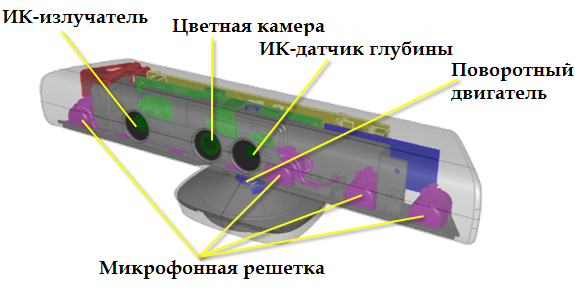


*Рис 2. Устройство трекера* Tobii eye tracker 4c

В основе технологии Tobii Eye Tracker 4C лежат инфракрасные лучи ближнего диапазона. Принцип работы оборудования заключается в следующем – зрачок респондента незаметно подсвечивается инфракрасным светом, при этом несколько высокоточных инфракрасных камер непрерывно снимает зрачок респондента. Tobii 4C с помощью ИК-излучения, может отслеживать положение головы, направление взгляда, положение глаз.

### 2.1.2 Microsoft Kinect.

В основе технологии Tobii Eye Tracker 4C лежат инфракрасные лучи ближнего диапазона. Принцип работы оборудования заключается в следующем – зрачок респондента незаметно подсвечивается инфракрасным светом, при этом несколько высокоточных инфракрасных камер непрерывно снимает зрачок респондента. Tobii 4C с помощью ИК-излучения, может отслеживать положение головы, направление взгляда, положение глаз.

Бесконтактный сенсорный игровой контроллер от компании Microsoft. Данное устройство позволяет пользователю взаимодействовать с программным обеспечением с помощью жестов и голосовых команд. Аппаратное обеспечение состоит из цветной видеокамеры, двух сенсоров глубины и микрофонной решетки, заключѐнных в едином корпусе.

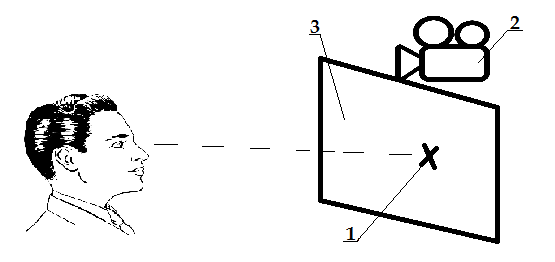
*Рис 3. Устройство трекера Microsoft Kinect*

## 2.2 Основные требования к системе.

Приведенные выше системы используют определение направления взгляда либо определение положения тела. Объектом управления для этих систем являются программные комплексы для обучения или развлечений.

### 2.2.1 Основные понятия.

Цель алгоритма определения направления взгляда – определить местоположение точки, на которую направлен взгляд оператора. В целях дальнейшей разработки требуется ввести понятие «экранная плоскость», на которой находится точка, чьи координаты требуется определить. Эту точку назовем «цель». Также, следует обозначить, что цель находится только в экранной плоскости, а эту плоскость можно разбить на пиксели с разрешением *Xs* на *Ys*. (Рис.) В этом случае, координаты цели (*Cx*, *Cy*) определяются в пикселях.



*Рис. 4. Схема расположения объектов. 1 – цель, 2 – видеокамера, 3 – экранная плоскость*.

### 2.2.2Приблизительная оценка точности алгоритма.

Минимальное перемещение изображения зрачка на изображении составляет один пиксель. Примем угол между вектором взгляда и экранной плоскостью ортогональным. Угол, на который поворачивается глазное яблоко оператора обозначим как φ.

где *d`* – размер зрачка на изображении в пикселях.

где:

*d* – диаметр зрачка.

*L* – расстояние от камеры до зрачка.

– угол обзора камеры.

*Xm* – разрешение камеры по горизонтали.

Таким образом, минимальное смещение цели на экранной плоскости можно выразить:

где *Ls* – расстояние от зрачка до экранной плоскости, в данной работе примем это расстояние равным расстоянию от зрачка до камеры (*Ls* = *L*).

Рассчитаем минимальное смещение цели при следующих параметрах:

*L* = 0,6м (рекомендуемое расстояние от глаз до экрана ПК);

*Xm* = 640px;

= 60° (угол обзора камеры, заявленный в характеристиках ПК);

*d* = 0,024 м.

Тогда *d`* ≈ 15px (пикселей), то есть φ ≈ ; 0,06м.

Таким образом, при вышеприведѐнных параметрах точность позиционирования составляет ±3см, это является достаточно неплохим результатом. На точность позиционирования влияет разрешение входного изображения, но более высокое разрешение повлияет на скорость обработки изображения. Рассчитаем для *Xm* = 1280px и *Xm* = 1920px.

|  |  |  |  |
| --- | --- | --- | --- |
| *Xm* | d’ | φ |  |
| *Xm* = 640px | 15px |  | 0,06м |
| *Xm* = 1280px | 30px |  | 0,03м |
| *Xm* = 1920px | 44px |  | 0,02м |

*Таблица 1. Теоретический расчет точности позиционирования.*

### 2.3 Вывод.

1. Существующие системы бесконтактного управления оснащены специальными ИК-камерами, сенсорами, что повышает их качество работы, а так же стоимость.
2. Произведен теоретический расчет точности определения направления взгляда. Определено, что точность определения зависит от разрешения входного изображения.

# Глава 3. Методы поиска объектов на изображении

В данной работе в качестве видеокамеры используется обычная веб-камера персонального компьютера, поэтому под *изображением* понимается один кадр из видеоряда. Все утверждения относительно *изображения* верны и для видеоряда, т.к. видеоряд – это массив изображений.

Для решения задачи нахождения лица и зрачков на изображении требуется использовать алгоритмы распознавания объектов. Рассмотрим методы, достигающие высокой точности при большой скорости вычисления.

## **3.1 Нейронные сети**

### 3.1.1 Введение в терминологию

*Иску́сственная нейро́нная се́ть (ИНС)* — [математическая модель](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C), а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования [биологических нейронных сетей](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) — сетей [нервных клеток](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD) живого организма. Это понятие возникло при изучении процессов, протекающих в [мозге](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B7%D0%B3), и при попытке [смоделировать](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) эти процессы. [У. Маккалока](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D0%BA%D0%B0%D0%BB%D0%BE%D0%BA,_%D0%A3%D0%BE%D1%80%D1%80%D0%B5%D0%BD) и [У. Питтса](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D1%82%D1%81,_%D0%A3%D0%BE%D0%BB%D1%82%D0%B5%D1%80) попытались создать первый прототип нейронной сети.

ИНС – это система взаимодействующих между собой простых [искусственных нейронов](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD). Каждый нейрон сети может только принимать и отдавать сигнал. Но, будучи соединёнными в достаточно большую сеть с управляемым взаимодействием, такие по отдельности простые процессоры вместе способны выполнять довольно сложные задачи.

С точки зрения [машинного обучения](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), нейронная сеть представляет собой частный случай методов [распознавания образов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%80%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2), [дискриминантного анализа](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B8%D0%BC%D0%B8%D0%BD%D0%B0%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7), [методов кластеризации](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) и т. п.

С [математической](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) точки зрения, обучение нейронных сетей — это [многопараметрическая задача](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%B0%D1%80%D0%B0%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B0&action=edit&redlink=1) [нелинейной оптимизации](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BB%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5).

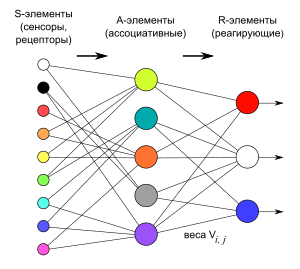
С точки зрения [кибернетики](https://ru.wikipedia.org/wiki/%D0%9A%D0%B8%D0%B1%D0%B5%D1%80%D0%BD%D0%B5%D1%82%D0%B8%D0%BA%D0%B0), нейронная сеть используется в задачах [адаптивного управления](https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D0%B0%D0%BF%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5) и как [алгоритмы](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) для [робототехники](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B1%D0%BE%D1%82%D0%BE%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0).

С точки зрения развития [вычислительной техники](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0) и [программирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), нейронная сеть — способ решения проблемы эффективного [параллелизма](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D0%B8%D0%B7%D0%BC_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)).

А с точки зрения [искусственного интеллекта](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82), ИНС является основой [философского](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%BB%D0%BE%D1%81%D0%BE%D1%84%D0%B8%D1%8F) течения [коннективизма](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%BD%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%B8%D0%B7%D0%BC) и основным направлением в [структурном подходе](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D1%8B%D0%B9_%D0%BF%D0%BE%D0%B4%D1%85%D0%BE%D0%B4&action=edit&redlink=1) по изучению возможности построения (моделирования) [естественного интеллекта](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82) с помощью [компьютерных](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) [алгоритмов](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC).

### 3.1.2 «Классические» нейронные сети

Большинство людей под «обычными» или «классическими» нейросетями понимает полносвязные нейронные сети прямого распространения с обратным распространением ошибки:



*Рис 5. Логическая схема перцептрона с тремя выходами*

*Перцептро́н, или персептрон*([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA" \o "Английский язык) perceptron от [лат.](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) perceptio — [восприятие](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%81%D0%BF%D1%80%D0%B8%D1%8F%D1%82%D0%B8%D0%B5); [нем.](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BC%D0%B5%D1%86%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Perzeptron) — [математическая](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%B8%D1%82%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) или [компьютерная модель](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [восприятия](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%81%D0%BF%D1%80%D0%B8%D1%8F%D1%82%D0%B8%D0%B5) [информации](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D1%8F) [мозгом](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B7%D0%B3) ([кибернетическая](https://ru.wikipedia.org/wiki/%D0%9A%D0%B8%D0%B1%D0%B5%D1%80%D0%BD%D0%B5%D1%82%D0%B8%D0%BA%D0%B0" \o "Кибернетика)[модель мозга](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%BC%D0%BE%D0%B7%D0%B3%D0%B0)), предложенная [Фрэнком Розенблаттом](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D0%B5%D0%BD%D0%B1%D0%BB%D0%B0%D1%82%D1%82,_%D0%A4%D1%80%D1%8D%D0%BD%D0%BA) в [1957 году](https://ru.wikipedia.org/wiki/1957_%D0%B3%D0%BE%D0%B4) и впервые реализованная в виде электронной машины [«Марк-1»](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D0%BA-1) в [1960 году](https://ru.wikipedia.org/wiki/1960_%D0%B3%D0%BE%D0%B4). Перцептрон стал одной из первых моделей [нейросетей](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C" \o "Искусственная нейронная сеть), а «Марк-1» — первым в мире [нейрокомпьютером](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80).

Перцептрон состоит из трёх типов элементов, а именно: поступающие от [датчиков](https://ru.wikipedia.org/wiki/%D0%94%D0%B0%D1%82%D1%87%D0%B8%D0%BA) [сигналы](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB) передаются [ассоциативным](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) элементам, а затем реагирующим элементам. Таким образом, перцептроны позволяют создать набор [«ассоциаций»](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%86%D0%B8%D1%8F_(%D0%BF%D1%81%D0%B8%D1%85%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F)) между входными [стимулами](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B8%D0%BC%D1%83%D0%BB) и необходимой [реакцией](https://ru.wikipedia.org/w/index.php?title=%D0%A0%D0%B5%D0%B0%D0%BA%D1%86%D0%B8%D1%8F_(%D0%B1%D0%B8%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F)&action=edit&redlink=1) на выходе. В биологическом плане это соответствует преобразованию, например, зрительной информации в [физиологический ответ](https://ru.wikipedia.org/w/index.php?title=%D0%A4%D0%B8%D0%B7%D0%B8%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B5%D0%B0%D0%BA%D1%86%D0%B8%D1%8F&action=edit&redlink=1) от двигательных [нейронов](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D1%8B). Согласно современной терминологии, перцептроны могут быть классифицированы как искусственные нейронные сети:

с одним скрытым слоем;

с [пороговой передаточной функцией](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD#Пороговая_передаточная_функция);

с [прямым распространением сигнала](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C#Сети_прямого_распространения_(Feedforward)).

Элементарный перцептрон состоит из элементов трёх типов: S-элементов, A-элементов и одного R-элемента.

*S-элементы* — это слой сенсоров или рецепторов. В физическом воплощении они соответствуют, например, светочувствительным клеткам [сетчатки глаза](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D1%87%D0%B0%D1%82%D0%BA%D0%B0) или [фоторезисторам](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%82%D0%BE%D1%80%D0%B5%D0%B7%D0%B8%D1%81%D1%82%D0%BE%D1%80) матрицы камеры. Каждый рецептор может находиться в одном из двух состояний — покоя или возбуждения, и только в последнем случае он передаёт единичный сигнал в следующий слой, ассоциативным элементам.

*A-элементы* называются ассоциативными, потому что каждому такому элементу, как правило, соответствует целый набор (ассоциация) S-элементов. A-элемент активизируется, как только количество сигналов от S-элементов на его входе превысило некоторую величину θ. Таким образом, если набор соответствующих S-элементов располагается на сенсорном поле в форме буквы «Д», A-элемент активизируется, если достаточное количество рецепторов сообщило о появлении «белого пятна света» в их окрестности, то есть A-элемент будет как бы ассоциирован с наличием/отсутствием буквы «Д» в некоторой области.

Сигналы от возбудившихся A-элементов, в свою очередь, передаются в сумматор R, причём сигнал от *i*-го ассоциативного элемента передаётся с коэффициентом *wi*. Этот коэффициент называется *весом* A—R связи.

Так же как и A-элементы, R-элемент подсчитывает сумму значений входных сигналов, помноженных на веса ([линейную форму](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0)). R-элемент, а вместе с ним и элементарный перцептрон, выдаёт «1», если линейная форма превышает порог θ, иначе на выходе будет «−1». Математически, функцию, реализуемую R-элементом, можно записать так:

[Обучение](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) элементарного перцептрона состоит в изменении весовых коэффициентов *wi* cвязей A—R. Веса связей S—A (которые могут принимать значения {−1; 0; +1}) и значения порогов A-элементов выбираются случайным образом в самом начале и затем не изменяются.

После обучения перцептрон готов работать в режиме [распознавания](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2) или [обобщения](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%BE%D0%BD%D1%8F%D1%82%D0%B8%D0%B9). В этом режиме перцептрону предъявляются ранее неизвестные ему объекты, и перцептрон должен установить, к какому классу они принадлежат. Работа перцептрона состоит в следующем: при предъявлении объекта возбудившиеся A-элементы передают сигнал R-элементу, равный сумме соответствующих коэффициентов *wi*. Если эта сумма положительна, то принимается решение, что данный объект принадлежит к первому классу, а если она отрицательна — то ко второму.

###### Описание на основе предикатов

Марвин Минский изучал свойства [параллельных вычислений](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%BB%D0%BB%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F), частным случаем которых на то время был перцептрон. Для анализа его свойств ему пришлось переизложить теорию перцептронов на язык [предикатов](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%B8%D0%BA%D0%B0%D1%82). Суть подхода заключалась в следующем[20]:

множеству сигналов от S-элементов была сопоставлена переменная X;

каждому A-элементу был сопоставлен предикат φ(X), названный частным предикатом;

каждому R-элементу был сопоставлен предикат ψ, зависящий от частных предикатов;

наконец, перцептроном было названо устройство, способное вычислять все предикаты типа ψ.

Применительно к «зрительному» перцептрону, переменная X символизировала образ какой-либо геометрической фигуры (стимул). Частный предикат позволял «распознавать» каждый свою фигуру. Предикат ψ означал ситуацию, когда линейная комбинация  ( *ai*— коэффициенты передачи) превышала некоторый порог θ.

Учёные выделили 5 семейств перцептронов, обладающих, по их мнению, интересными свойствами:[[20]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-26)

**Перцептроны, ограниченные по диаметру** — каждая фигура X, распознаваемая частными предикатами, не превосходит по диаметру некоторую фиксированную величину.

**Перцептроны ограниченного порядка** — каждый частный предикат зависит от ограниченного количества точек из X.

**Перцептроны Гамбы** — каждый частный предикат должен быть линейной пороговой функцией, то есть мини-перцептроном.

**Случайные перцептроны** — перцептроны ограниченного порядка, где частные предикаты представляют собой случайно выбранные булевы функции. В книге отмечается, что именно эта модель наиболее подробно изучалась группой Розенблатта.

**Ограниченные перцептроны** — множество частных предикатов бесконечно, а множество возможных значений коэффициентов  *ai* конечно.

Хотя такой математический аппарат позволил применить анализ только к элементарному перцептрону Розенблатта, он вскрыл много принципиальных ограничений для параллельных вычислений, от которых не свободен ни один вид современных искусственных нейронных сетей.

### 3.1.3 Алгоритмы обучения

Важным свойством любой нейронной сети является [способность к обучению](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5). Процесс обучения является процедурой настройки весов и порогов с целью уменьшения разности между желаемыми (целевыми) и получаемыми векторами на выходе. В своей книге Розенблатт пытался классифицировать различные алгоритмы обучения перцептрона, называя их системами подкрепления.

***Система подкрепления*** — это любой набор правил, на основании которых можно изменять с течением времени матрицу взаимодействия (или состояние памяти) перцептрона[[21]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD" \l "cite_note-roslearn-27).

Описывая эти системы подкрепления и уточняя возможные их виды, Розенблатт основывался на идеях [Д. Хебба](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D0%B1%D0%B1,_%D0%94%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%B4) об обучении, предложенных им в [1949 году](https://ru.wikipedia.org/wiki/1949_%D0%B3%D0%BE%D0%B4)[22], которые можно перефразировать в следующее правило, состоящее из двух частей:

Если два нейрона по обе стороны синапса (соединения) активизируются одновременно (то есть синхронно), то прочность этого соединения возрастает.

Если два нейрона по обе стороны синапса активизируются асинхронно, то такой синапс ослабляется или вообще отмирает[[21]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-28).

#### 3.1.3.1 Обучение с учителем

Классический метод обучения перцептрона — это метод коррекции ошибки[[23]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-teorshod-12). Он представляет собой такой вид [обучения с учителем](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81_%D1%83%D1%87%D0%B8%D1%82%D0%B5%D0%BB%D0%B5%D0%BC), при котором вес связи не изменяется до тех пор, пока текущая реакция перцептрона остается правильной. При появлении неправильной реакции вес изменяется на единицу, а знак (+/-) определяется противоположным от знака ошибки.

Допустим, мы хотим обучить перцептрон разделять два класса объектов так, чтобы при предъявлении объектов первого класса выход перцептрона был положителен (+1), а при предъявлении объектов второго класса — отрицательным (−1). Для этого выполним следующий алгоритм:

Случайным образом выбираем пороги для A-элементов и устанавливаем связи S—A (далее они изменяться не будут).

Начальные коэффициенты *wi* полагаем равными нулю.

Предъявляем обучающую выборку: объекты (например, круги либо квадраты) с указанием класса, к которым они принадлежат.

Показываем перцептрону объект первого класса. При этом некоторые A-элементы возбудятся. *wi*, соответствующие этим возбуждённым элементам, увеличиваем на 1.

Предъявляем объект второго класса и коэффициенты *wi* тех A-элементов, которые возбудятся при этом показе, уменьшаем на 1.

Обе части шага 3 выполним для всей обучающей выборки. В результате обучения сформируются значения весов связей *wi.*

[Теорема сходимости перцептрона](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B5%D0%BC%D0%B0_%D1%81%D1%85%D0%BE%D0%B4%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8_%D0%BF%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD%D0%B0)[[23]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-teorshod-12), описанная и доказанная Ф. Розенблаттом (с участием Блока, Джозефа, Кестена и других исследователей, работавших вместе с ним), показывает, что элементарный перцептрон, обучаемый по такому алгоритму, независимо от начального состояния весовых коэффициентов и последовательности появления стимулов всегда приведёт к достижению решения за конечный промежуток времени.

#### 3.1.3.2 Обучение без учителя

Кроме классического метода обучения перцептрона Розенблатт также ввёл понятие об [обучении без учителя](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B1%D0%B5%D0%B7_%D1%83%D1%87%D0%B8%D1%82%D0%B5%D0%BB%D1%8F), предложив следующий способ обучения:

***Альфа-система подкрепления*** — это система подкрепления, при которой веса всех активных связей *wi*, которые ведут к элементу *wi*, изменяются на одинаковую величину r, а веса неактивных связей за это время не изменяются[[25]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-29).

Затем, с разработкой понятия [многослойного перцептрона](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D1%81%D0%BB%D0%BE%D0%B9%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD_%D0%A0%D1%83%D0%BC%D0%B5%D0%BB%D1%8C%D1%85%D0%B0%D1%80%D1%82%D0%B0), альфа-система была модифицирована и её стали называть [*дельта-правило*](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BB%D1%8C%D1%82%D0%B0-%D0%BF%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D0%BE). Модификация была проведена с целью сделать функцию обучения [дифференцируемой](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8) (например, [сигмоидной](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BC%D0%BE%D0%B8%D0%B4)), что в свою очередь нужно для применения метода [градиентного спуска](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D0%B4%D0%B8%D0%B5%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D1%81%D0%BF%D1%83%D1%81%D0%BA), благодаря которому возможно обучение более одного слоя.

#### 3.1.3.3 Метод обратного распространения ошибки

Для обучения многослойных сетей рядом учёных, в том числе [Д. Румельхартом](https://ru.wikipedia.org/wiki/%D0%94%D1%8D%D0%B2%D0%B8%D0%B4_%D0%A0%D1%83%D0%BC%D0%B5%D0%BB%D1%8C%D1%85%D0%B0%D1%80%D1%82), был предложен [градиентный алгоритм](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D0%B4%D0%B8%D0%B5%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D1%81%D0%BF%D1%83%D1%81%D0%BA) обучения с учителем, проводящий сигнал ошибки, вычисленный выходами перцептрона, к его входам, слой за слоем. Сейчас это самый популярный метод обучения многослойных перцептронов. Его преимущество в том, что он может обучить все слои нейронной сети, и его легко просчитать локально. Однако этот метод является очень долгим, к тому же, для его применения нужно, чтобы передаточная функция нейронов была дифференцируемой. При этом в перцептронах пришлось отказаться от бинарного сигнала, и пользоваться на входе [непрерывными значениями](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE)[[26]](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD#cite_note-30).

#### 3.1.3.4 Сигмоидальные функции активации

Наиболее часто в качестве функций активации используются следующие виды [сигмоид](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BC%D0%BE%D0%B8%D0%B4" \o "Сигмоид):

*Функция Ферми* (экспоненциальная сигмоида):

*Рациональная сигмоида* (при вырождается в так называемую пороговую функцию активации):

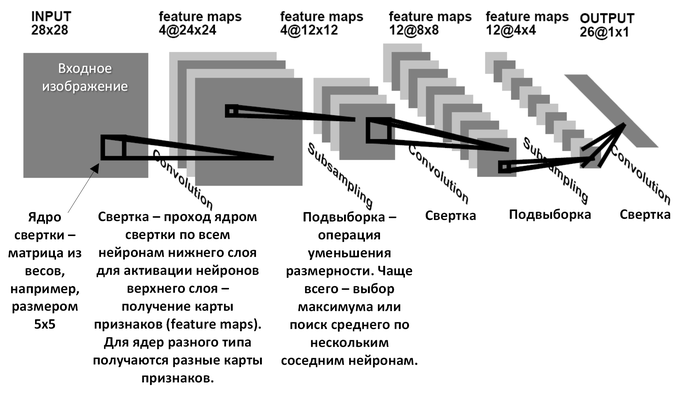
*Гиперболический тангенс*:

где s — выход сумматора нейрона, — произвольная константа.

Менее всего, сравнительно с другими сигмоидами, процессорного времени требует расчет рациональной сигмоиды. Для вычисления гиперболического тангенса требуется больше всего тактов работы процессора. Если же сравнивать с пороговыми функциями активации, то сигмоиды рассчитываются очень медленно. Если после суммирования в пороговой функции сразу можно начинать сравнение с определенной величиной (порогом), то в случае сигмоидальной функции активации нужно рассчитать сигмоид (затратить время в лучшем случае на три операции: взятие модуля, сложение и деление), и только потом сравнивать с пороговой величиной (например, нулём). Если считать, что все простейшие операции рассчитываются процессором за примерно одинаковое время, то работа сигмоидальной функции активации после произведённого суммирования (которое займёт одинаковое время) будет медленнее пороговой функции активации как 1:4.

### 3.1.4 Сверточные нейронные сети

*Сверточная нейронная сеть* ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *convolutional neural network*, CNN) — специальная архитектура [искусственных нейронных сетей](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), предложенная [Яном Лекуном](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%BA%D1%83%D0%BD,_%D0%AF%D0%BD) в 1988 году и нацеленная на эффективное [распознавание изображений](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%80%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2), входит в состав технологий [глубинного обучения](https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D1%83%D0%B1%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *deep learning*). Таким образом, идея сверточных нейронных сетей заключается в чередовании сверточных слоев ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *convolution layers*) и субдискретизирующих слоев ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *subsampling layers* или [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *pooling layers*, слоёв подвыборки). *Структура сети* — однонаправленная (без обратных связей), принципиально многослойная. Для обучения используются стандартные методы, чаще всего [метод обратного распространения ошибки](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BE%D0%B1%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B3%D0%BE_%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F_%D0%BE%D1%88%D0%B8%D0%B1%D0%BA%D0%B8). Функция активации нейронов (передаточная функция) — любая, по выбору исследователя.



*Рис 6. Схема сверточной нейронной сети.*

Название архитектура сети получила из-за наличия операции [свёртки](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%91%D1%80%D1%82%D0%BA%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7)), суть которой в том, что каждый фрагмент изображения умножается на матрицу (ядро) свёртки поэлементно, а результат суммируется и записывается в аналогичную позицию выходного изображения.

## **3.2 Метод Виолы-Джонса**

Хотя метод был разработан и представлен в 2001 году Полом Виолой и Майклом Джонсом [17, 18], он до сих пор на момент написания данной работы является основополагающим для поиска объектов на изображении в реальном времени.[18]

Основные принципы, на которых основан метод, таковы:

* используются изображения в [интегральном представлении](http://habrahabr.ru/blogs/algorithm/102919/), что позволяет вычислять быстро необходимые объекты;
* используются [признаки Хаара](http://en.wikipedia.org/wiki/Haar-like_features), с помощью которых происходит поиск нужного объекта (в данном контексте, лица и его черт);
* используется [бустинг](http://machinelearning.ru/wiki/index.php?title=%D0%91%D1%83%D1%81%D1%82%D0%B8%D0%BD%D0%B3) (от англ. *Boost* – улучшение, усиление) для выбора наиболее подходящих признаков для искомого объекта на данной части изображения;
* все признаки поступают на вход [классификатора](http://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8), который даёт результат «верно» либо «ложь»;
* используются [каскады признаков](http://courses.graphicon.ru/main/vision/2011/lectures/7) для быстрого отбрасывания окон, где не найдено лицо.

Требуется подробный разбор принципов, на которых основан алгоритм Виолы-Джонса. Данный метод в общем виде ищет лица и черты лица по общему принципу сканирующего окна.

### 3.2.1 Принцип сканирующего окна

В общем виде, задача обнаружения лица и черт лица человека на цифровом изображении выглядит именно так:

* + Имеется изображение, на котором есть искомые объекты. Оно представлено двумерной матрицей пикселей размером w\*h, в которой каждый пиксель имеет значение:  
    — от 0 до 255, если это черно-белое изображение;  
    — от 0 до 2553, если это цветное изображение (компоненты R, G, B).
  + в результате своей работы, алгоритм должен определить лица и их черты и пометить их – поиск осуществляется в активной области изображения прямоугольными признаками, с помощью которых и описывается найденное лицо и его черты:  
    где x, y – координаты центра i-го прямоугольника, w – ширина, h – высота, a – угол наклона прямоугольника к вертикальной оси изображения.

### 3.2.2 Интегральное представление изображений

Для того, чтобы производить какие-либо действия с данными, используется интегральное представление изображений [19] в методе Виолы-Джонса. Такое представление используется часто и в других методах, к примеру, в вейвлет-преобразованиях, SURF и многих других разобранных алгоритмах. Интегральное представление позволяет быстро рассчитывать суммарную яркостьпроизвольного прямоугольника на данном изображении, причем какой бы прямоугольник не был, время расчета неизменно.

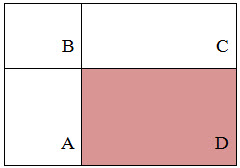
***Интегральное представление изображения*** – это матрица, совпадающая по размерам с исходным изображением. В каждом элементе ее хранится сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента. Элементы матрицы рассчитываются по следующей формуле:

где I(i,j) — яркость пикселя исходного изображения.  
Каждый элемент матрицы L[x,y] представляет собой сумму пикселей в прямоугольнике от (0,0) до (x,y), т.е. значение каждого пикселя (x,y) равно сумме значений всех пикселов левее и выше данного пикселя (x,y). Расчет матрицы занимает линейное время, пропорциональное числу пикселей в изображении, поэтому интегральное изображение просчитывается за один проход.

Расчет матрицы возможен по формуле 1.3:

По такой интегральной матрице можно очень быстро вычислить сумму пикселей произвольного прямоугольника, произвольной площади.

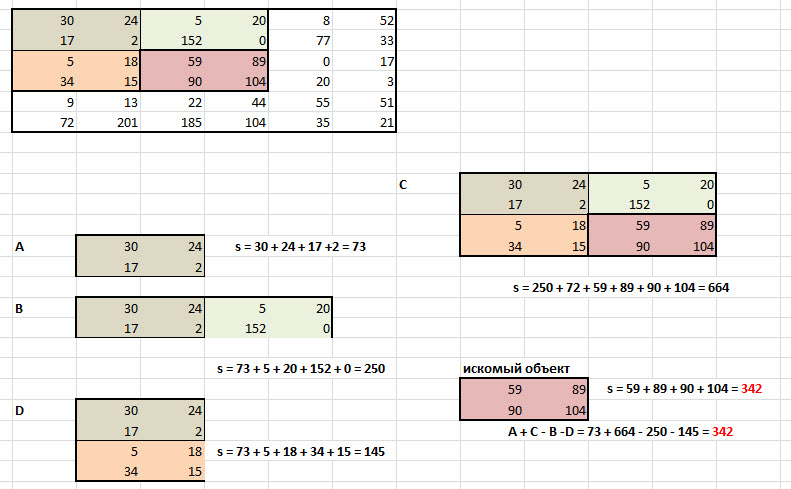
Пусть в прямоугольнике ABCD есть интересующий нас объект D:



*Рис 7.Прямоугольник ABCD пример перевода в интегральную матрицу.*

Из рисунка понятно, что сумму внутри прямоугольника можно выразить через суммы и разности смежных прямоугольников по следующей формуле:

Пример вычисления показан на рисунке ниже:

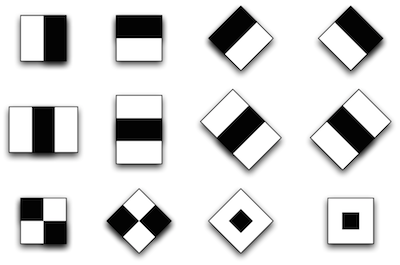


*Рис 8. Пример вычисления интегральной матрицы по изображению.*

### 3.2.3 Признаки Хаара

*Признаки Хаара* — признаки цифрового изображения, используемые в [распознавании образов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D1%80%D0%B0%D1%81%D0%BF%D0%BE%D0%B7%D0%BD%D0%B0%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2). Своим названием они обязаны интуитивным сходством с [вейвлетами Хаара](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B9%D0%B2%D0%BB%D0%B5%D1%82_%D0%A5%D0%B0%D0%B0%D1%80%D0%B0). Признаки Хаара использовались в первом детекторе лиц, работающем в реальном времени.

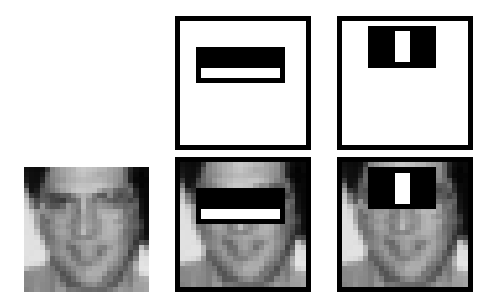
Обычно у каждого метода есть основа, то, без чего этот метод не мог бы существовать в принципе, а уже над этой основой строится вся остальная часть. В методе Виолы-Джонса эту основу составляют примитивы Хаара, представляющие собой разбивку заданной прямоугольной области на наборы разнотипных прямоугольных подобластей:



*Рис 9. Наборы прямоугольных областей для вычисления признаков Хаара.*

В оригинальной версии алгоритма Виолы-Джонса использовались только примитивы без поворотов, а для вычисления значения признака сумма яркостей пикселей одной подобласти вычиталась из суммы яркостей другой подобласти [8]. В развитии метода были предложены примитивы с наклоном на 45 градусов и несимметричных конфигураций. Также вместо вычисления обычной разности, было предложено приписывать каждой подобласти определенный вес и значения признака вычислять как взвешенную сумму пикселей разнотипных областей [9]:

Основной причиной, почему в основу метода легли примитивы Хаара, являлась попытка уйти от пиксельного представления с сохранением скорости вычисления признака. Из значений пары пикселей сложно вынести какую-либо осмысленную информацию для классификации, в то время как из двух признаков Хаара строится, например, первый каскад системы по распознаванию лиц, который имеет вполне осмысленную интерпретацию [8]:



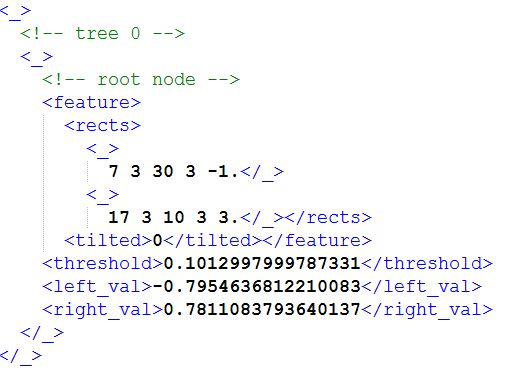
*Рис 10.Пример работы каскада системы по распознаванию лиц из двух признаков.*

Сложность вычисления признака, так же как и получения значения пикселя, остается O(1): значение каждой подобласти можно вычислить скомбинировав 4 значения интегрального представления (Summed Area Table — SAT), которое в свою очередь можно построить заранее один раз для всего изображения за O(n), где n — число пикселей в изображении, используя формулу [9]:

Таким образом, создали быстрый алгоритм поиска объектов, который пользуется успехом уже больше десятилетия. Но вернемся к нашим признакам. Для определения принадлежности к классу в каждом каскаде, находится сумма значений слабых классификаторов этого каскада. Каждый слабый классификатор выдает два значения в зависимости от того больше или меньше заданного порога значение признака, принадлежащего этому классификатору. В конце сумма значений слабых классификаторов сравнивается с порогом каскада, и выносится решение, найден объект данным каскадом или нет.

Для разных видов объектов создаются разные XML классификаторы. Существуют базы и приложения, с помощью которых можно получить XML классификатор объекта. Мы воспользуемся каскадом, полученным студентами из Modesto Castrillon-Santana (IUSIANI, University of Las Palmas de Gran Canaria, Spain) в 2006 году. Лицензия открыта для некоммерческого использования.

Пример структуры описания:



*Рис 11.Пример структуры описания каскада в XML-файле.*

Создается впечатление, что здесь куча непонятных цифр и странной информации, но на самом деле все просто:

* Дерево объектов — набор слабых классификаторов, на основе которых выносится решение о том, находится объект на изображении или нет,
* Два объекта в узле (без имени) — это параметры конкретного слабого классификатора.
  + Расшифровка первого объекта слева направо: первые два значения в нашем случае не используются, третье — номер признака в общей таблице признаков (она располагается дальше в XML), четвертое — пороговое значение слабого классификатора. Так как у нас используется классификатор, основанный на одноуровневых решающих деревьях (Decision Stump), то если значение признака Хаара меньше порога слабого классификатора (четвертое значение), выбирается значение left\_val, если больше —right\_val.

Все эти признаки в какой-то степени являются самыми обыкновенными детекторами границ. На основе этого базиса вычисляется решение о том, распознал ли каскад объект на изображении или нет. Второй по важности момент в методе Виола-Джонса — это использование каскадной модели или вырожденного дерева принятия решений: в каждом узле дерева с помощью каскада принимается решение, содержится объект на изображении или нет. Если объект не содержится, то алгоритм заканчивает свою работу, если он может содержаться, то мы переходим к следующему узлу. Обучение построено таким образом, чтобы на начальных уровнях с наименьшими затратами отбрасывать большую часть окон, в которых не может содержаться объект. В случае распознавания лиц — первый уровень содержит всего 2 слабых классификатора, в случае распознавания глаз — 6.

### 3.2.4 Используемая в алгоритме модель машинного обучения

*Обучение машины* — это процесс получения модулем новых знаний. Есть признанное определение данному процессу:

*«Машинное обучение – это наука, изучающая компьютерные алгоритмы, автоматически улучшающиеся во время работы» (Michel, 1996)*

Данный процесс входит в концепцию и технологию под названием [*Data mining*](http://ru.wikipedia.org/wiki/Data_mining) (извлечение информации и интеллектуальный анализ данных), куда входят помимо Машинного обучения такие дисциплины, как Теория баз данных, Искусственный интеллект, Алгоритмизация, Распознавание образов и прочие. Машинное обучение в методе Виолы-Джонса решает такую задачу как классификация.

### 3.2.5 Обучение классификатора в методе Виолы-Джонса

В контексте алгоритма, имеется множество объектов (изображений), разделённых некоторым образом на классы. Задано конечное множество изображений, для которых известно, к какому классу они относятся (к примеру, это может быть класс «фронтальное положение носа»). Это множество называется обучающей выборкой. Классовая принадлежность остальных объектов не известна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества [4].

*Классифицировать объект* — значит, указать номер (или наименование класса), к которому относится данный объект.

*Классификация объекта* — номер или наименование класса, выдаваемые алгоритмом классификации в результате его применения к данному конкретному объекту.

*Классификатор(classifier)* — в задачах классификации это аппроксимирующая функция, выносящая решение, к какому именно классу данный объект принадлежит.

*Обучающая выборка* – конечное число данных.

В машинном обучении задача классификации относится к разделу обучения с учителем когда классы поделены. Распознавание образов по сути своей и есть классификация изображений и сигналов. В случае алгоритма Виолы-Джонса для идентификации и распознавания лица классификация является двухклассовой.

Постановка классификации выглядит следующим образом:

Есть X – множество, в котором хранится описание объектов, Y – конечное множество номеров, принадлежащих классам. Между ними есть зависимость – отображение Y\*: X => Y. Обучающая выборка представлена Xm = {(x1,y1), …, (xm,ym)}. Конструируется функция f от вектора признаков X, которая выдает ответ для любого возможного наблюдения X и способна классифицировать объект x∈X. Данное простое правило должно хорошо работать и на новых данных.

## **3.3 Вывод**

Чтобы отслеживать взгляд пользователя, нам нужно знать, где находится зрачок. Системе не требуется чрезмерно мощная камера, поскольку фиксация глаза – единственные значимые данные, которые требуются, чтобы найти взгляд, и они соответствуют самому длинному периоду времени отсутствия движения глаза. Существует большое количество доступных для использования алгоритмов, в которых используется большее различных методов. Основными требованиями являются:

* обнаружение глаз,
* обнаружение зрачка,
* обнаружение дополнительной функции,
* преобразование в координаты фокуса.

# **Глава 4. Алгоритмы распознавания взгляда**

Система отслеживания взглядов состоит из сложной комбинации алгоритмов для получения точных координат взгляда. Поиск глаз на изображении – это только часть. Существует много способов комбинирования методов для получения требуемого результата. Рассмотрим некоторые примеры алгоритмов отслеживания взгляда.

## **4.1 Инфракрасное отражение**

Принцип работы метода ИК отражения заключается в том, что повышается контраст между зрачком и радужной оболочкой. Ниже приведен пример использования инфракрасного отражения для обнаружения глаз.

В [7] авторы сначала определяют область 60х60 пикселей (исходное разрешение монитора 1280 × 1024 пикселей) с ориентировочной оценкой центра зрачка. Поскольку глаза имеют разное расположение от источника ИК света, они имеют разные показатели преломления, зрачок кажется черным, а остальное нет. Использование простого порогового алгоритма для обработки, отфильтрованной серой шкалы изображения, полученного с камеры, проверяются все точки ниже для определения ROI. Пороговое уравнение, использованное в [7], приведено ниже.

Трудности, возникающие в процессе разработки подобных систем неизбежны. В настоящее время исследования в этой области направлены на создание чувствительного к пользователю и легкодоступного устройства. Идеальным решением для этого является бесконтактное и неинвазивное устройство. Но с такими устройствами возникают новые трудности, такие как алгоритмы поиска головы, глаз, зрачка.

Общая идея использования инфракрасного излучения заключается в том, что отражение инфракрасного излучения упрощает процесс определения центра зрачка. Таким образом, уменьшается сложность расчета и повышается точность обнаружения зрачков, повышается эффективность системы отслеживания глаз. Отрицательная сторона этого заключается в потребности инфракрасного света. Также могут быть другие источники света, или тени, которые нарушают это изображение и создают «шум», что усложняет вычисление, но это проблема актуальна для большинства ненавязчивых систем.

## **4.2 Алгоритм отслеживания взгляда в реальном времени с компенсацией движения головы(RTAC)**

Объяснение RTAC в [10] начинается со ссылкой на «Pupil Center Corneal Reflection (PCCR) »[10, стр. 395]. Согласно PCCR, вычисления направления взгляда сначала приобретают вектор блика зрачка, а затем используется функция отображения взгляда. Блик является отражением инфракрасного источника света в глазу. Вектор блика зрачка является двумерным вектором между бликом и зрачком. Затем алгоритм использует функцию для отображения этого вектора к направлению 3D-взгляда. Функция отображения определяется следующими функциями:

Где θh – угол между направлением взгляда и горизонтальным направлением, а также θv – угол между направлением взгляда и вертикальным направлением. Коэффициенты, a и b, оцениваются с использованием наборов пар векторов бликов. Следующим шагом в этом алгоритме является вычисление движения головы. Предыдущий расчет не включает адаптацию к движениям головы. Таким образом, система должна компенсировать эти движения головы. Этот расчет довольно сложен. Основа для этого два уравнения:

Эти два уравнения описывают изменения в положении головы в горизонтальном и вертикальном положении соответственно. Вывод этих уравнений можно найти в [10, стр. 397].

Метод компенсации, используемый в этом алгоритме, использует два набора уравнений:

1. Вычисление пропорционального изменения увеличения и последующее использование этих значений для вычисления значения компенсации в горизонтальном и вертикальном направлениях. [10, стр. 397].
2. Затем следующим шагом является фиксация головы пользователя во время калибровки, чтобы получить начальные требуемые параметры. После этой калибровки пользователь может свободно перемещать свою голову и параметры сравниваются, чтобы выполнить вычисления.

В [10] было обнаружено, что этот алгоритм имел точность порядка 1 градуса, И что между разными предметами мало что изменилось. Существовала также небольшая разница между различными положениями головы, но ошибка действительно возрастала, когда голова перемещалась около границ видения камеры.

# **Глава 5. Концепция приложения**

Для реализации проекта потребуется веб-камера, открытые библиотеки Computer Vision OpenCV, tensorflow. В качестве приложения создана форма с девятью блоками, каждые 1,5 секунды загорается один случайный блок. Оператору дается 1 секунда для фокусировки взгляда на этом блоке и, затем, сохраняется снимок с веб-камеры с пометкой какой блок загорелся. Далее эти снимки используются для сбора данных по точности алгоритмов.

## **5.1 Технические проблемы**

Традиционные бесконтактные системы отслеживания глаз имеют ряд проблем, которые необходимо учесть при расчете фокуса или при нахождении глаза.

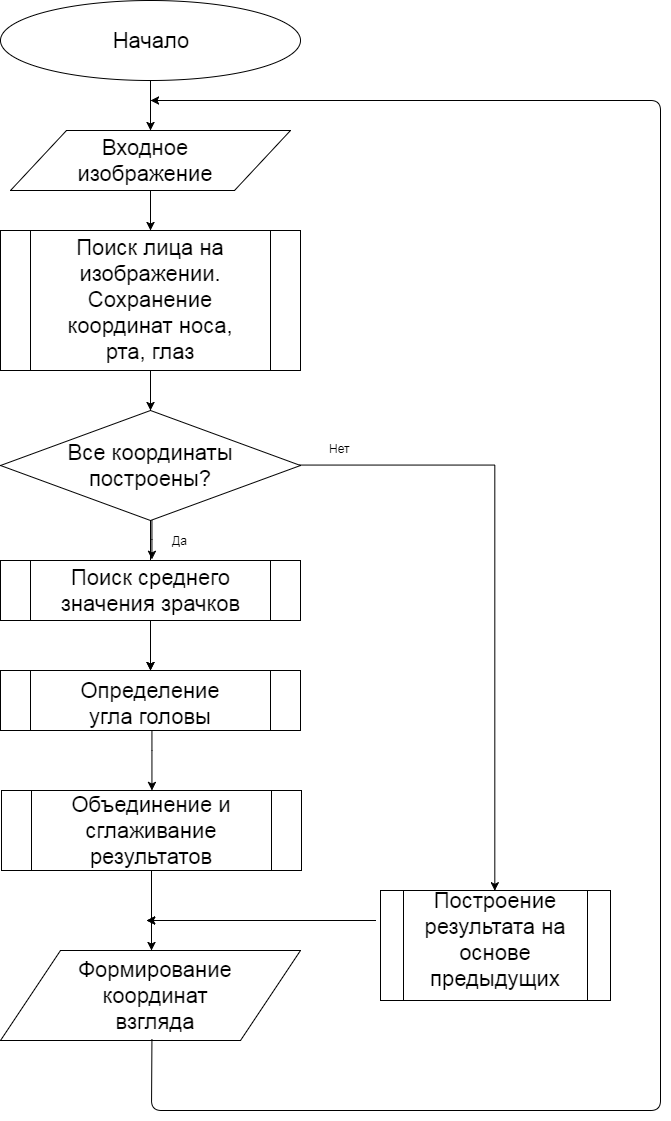
Первая и самая главная проблема, это движение головы, о котором упоминалось в [11]. Старые системы отслеживания глаз требуют, чтобы пользователь продолжал удерживать голову неподвижно, чтобы точно зафиксировать данные. Алгоритмы, рассмотренные ранее и включающие в себя инфракрасное освещение, не поддерживают изменение положения головы в кадре.

Вторая проблема, с которой сталкивается область глазного трекинга, известна как проблема «Midas Touch». Эта проблема является прямым следствием движения глаз. Глаза используются как устройства ввода информации и поэтому нужно точно определять перемещение ли это или мелкие колебания (дрейфы).

## **5.2 Алгоритм**

Общий алгоритм построен на основе непрерывной обработке потока изображения.

Шаги цикла – это шаги, необходимые для получения всех необходимых данных из одного кадра. Алгоритм определения направления взгляда:



*Рис 12. Блок-схема алгоритма определения направления взгляда.*

## **5.3 Получение координат частей лица**

Первый шаг алгоритма бесконтактного управления – это поиск отдельных частей лица на его изображении, а именно:

1. Лицо
2. Рот
3. Нос
4. Глаза

Основываясь на взаимном относительном расположении этих частей изображения можно определить угловое положение головы пользователя. Местоположение глаз необходимо для определения направления взгляда. Выше были рассмотрены некоторые подходящие алгоритмы.

## **5.4 Определение угла положения головы**

Алгоритмы определения частей лица на изображении вычисляют минимальный прямоугольник, внутри которого полностью находится искомый объект. Для упрощения вычислений координаты центра прямоугольника будем считать координатами искомого объекта. Полученные данные о расположении частей лица на изображении могут быть недостоверными из-за ошибок алгоритма распознавания. Поэтому, для контроля результатов алгоритмов распознавания объекта на изображении проводится ряд проверок, позволяющий отсеять недостоверные результаты. В качестве такой проверки можно привести пример линии, проходящей через центры областей, отмечающих губы и нос, пересекающей отрезок, соединяющий центры зрачков посередине и под прямым углом (с допустимой погрешностью). В результате, координаты цели на двухмерной плоскости, полученные на основе положения головы оператора, можно определить по следующим формулам:

где

*B1x, B1y* – координаты цели по осям *x* и *y* соответственно,

*Nx, Ny* – координаты центра области носа,

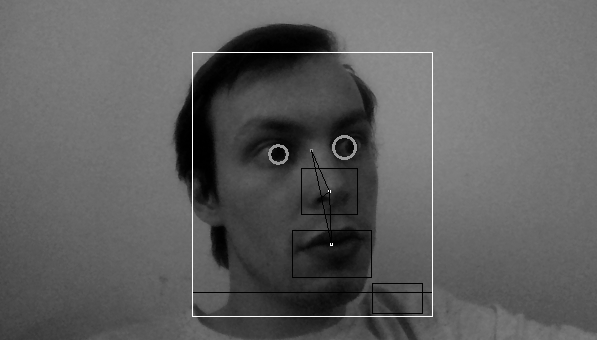
*Mx, My* – координаты центра области губ,

*Rx, Ry* – координаты центра области правого зрачка,

*Lx, Ly* – координаты центра области левого зрачка,

*K1x, K1y* – коэффициенты корректировки, зависят от характеристик камеры. Данные коэффициенты подбираются опытным путем.

*X, Y* – это размер в пикселях экранной плоскости.



*Рис 13. Пример построения проверочных линий для вычисления направления взгляда.*

## **5.5 Определение направления взгляда**

Центры найденных эллипсов, внутри области пары глаз, указывают на расположение зрачков. Далее смещение зрачков относительно центров областей глаз умножается на коэффициенты, зависящие линейно от положения головы пользователя. В результате получаются координаты B2x и B2y, *x* и *y* координаты цели на экранной плоскости соответственно, определяемые направлением взгляда:

где

*ΔRx, ΔRy* – Смещение зрачка правого глаза

*ΔLx, ΔLy* – Смещение зрачка левого глаза

*K2x, K2y* – Коэффициенты корректировки, характеризуют влияние углового положения головы на изображение зрачков.

*K3x, K3y* – Коэффициенты корректировки, зависят от характеристик камеры, различий в характере вертикальных и горизонтальных движений глаз человека.

## **5.6 Объединение и сглаживание результата**

Предлагаемый алгоритм для системы бесконтактного управления использует данные от разных подсистем, а именно:

* Подсистема определения угла поворота головы.
* Подсистема определения направления взгляда.

Данные, полученные от подсистем, в силу неоднородности источников различаются по параметрам. Можно выделить четыре параметра подсистемы:

* Точность,
* Переодичность обновления,
* Достоверность,
* Скорость обработки информации.

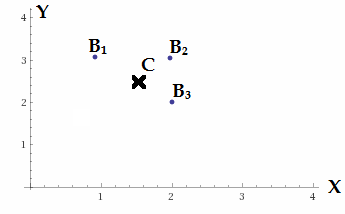
В качестве примера можно привести подсистему определения направления взгляда. Подсистема дает самый точный результат из всех каналов получения данных, но вероятность возникновения ошибки у данной подсистемы велика.

Такой недостаток подсистем стал причиной для доработки алгоритма определения направления взгляда функцией взаимной коррекции и объединения данных.

В качестве точки на экранной плоскости в дискретный момент времени *t* возьмем точку *Ct*, заданную дискретными координатами (x, y). Воздействие, полученное по каналу i, как точку *Bi* на этой же плоскости. Рассмотрим два алгоритма объединения сигналов.

5.6.1 Упрощенный способ объединения сигналов

Первый способ, использованный в процессе разработки, заключается в получении результирующей точки *Ct* через усреднение координат точек *i* *Bt* :

**

*Рис 14. Пример работы упрощенного способа объединения сигналов.*

Данный алгоритм объединения сигналов довольно простой и легко реализуемый, но в нем не учитывается неоднородность данных, получаемых от разных систем.

5.6.2 Модифицированный способ объединения сигналов

Введем понятия характеристических коэффициентов:

*Ri* – достоверность данных канала *i*.

*Si* – точность данных канала *i*.

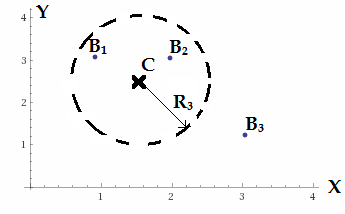
*τi* – постоянная времени, характеризует убывание актуальности данных канала *i* с течением времени *t*.

Сначала, требуется определить актуальность данных канала, если они были получены не в текущий момент времени *t*. Для более быстрой обработки изображения, подсистемы реализованы таким образом, что различные каналы работают асинхронно, сообщая результат по мере получения. То есть, для канала *i* в момент времени *t* есть интервал времени . Это интервал прошедшего с последнего опроса этого канала.Тогда закон убывания актуальности данных с канала *i* можно представить в виде:

Далее, результирующую точку выразим как средневзвешенное значение координат от разных *n* каналов:

В связи с помехами на изображении, каналы могут предоставлять недостоверные данные, исключим из результата те данные, для которых расстояние от результирующей точки без учета рассматриваемого канала *j* больше, чем порог доверия *Ri* :

На изображении ниже сигнал от подсистемы 3 не учитывается при получении результирующего сигнала, так как расстояние от общего центра *С* точек *B1* и *B2* до точки *B3* больше, чем порог доверия *R3.*



*Рис 15. Пример работы модифицированного способа объединения сигналов. Сигнал от B3 не учитывается при вычислении результирующего сигнала, т.к. расстояние от общего центра каналов В1 и В2 до точки В3 больше порога доверия R3.*

По причине количества коэффициентов и отсутствия четкого критерия оптимальности, решение задачи с помощью представленного алгоритма, а именно нахождение значений коэффициентов *Ri*, *Si*, *τi* является проблемой. С помощью экспериментального подбора удалось вычислить коэффициенты, удовлетворяющие скорости и достаточной точности алгоритма.

5.6.3 Итеративное сглаживание результата

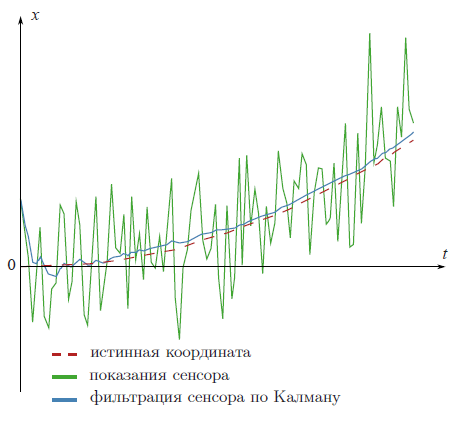
Для плавности управления системой требуется провести сглаживание результатов. В данной работе применен фильтр Калмана для достижения это цели.

***Фи́льтр Ка́лмана*** — [эффективный](https://ru.wikipedia.org/wiki/%D0%AD%D1%84%D1%84%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81) [рекурсивный фильтр](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%BB%D1%8C%D1%82%D1%80_%D1%81_%D0%B1%D0%B5%D1%81%D0%BA%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B9_%D0%B8%D0%BC%D0%BF%D1%83%D0%BB%D1%8C%D1%81%D0%BD%D0%BE%D0%B9_%D1%85%D0%B0%D1%80%D0%B0%D0%BA%D1%82%D0%B5%D1%80%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%BE%D0%B9), оценивающий вектор состояния [динамической системы](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), используя ряд неполных и зашумленных измерений. Назван в честь [Рудольфа Калмана](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BB%D0%BC%D0%B0%D0%BD,_%D0%A0%D1%83%D0%B4%D0%BE%D0%BB%D1%8C%D1%84).

Формула описывает уравнение изменения координаты.

где *zk –* полученное значение с сенсора, *uk* – известная управляющая функция, – оптимальное отфильтрованное значение на *k-ой* итерации.

Основная идея фильтра Калмана заключается в том, что требуется найти коэффициент *K* такой, чтобы отфильтрованное значение по формуле в среднем меньше всего отличалось бы от реального значения координаты *xk+1*. Таким образом очевидно, что отфильтрованное значение зависит от всех предыдущих показаний датчика линейно:



*Рис 16. Пример работы фильтра Калмана.*

### 5.7 Выводы

1. Предложен алгоритм, работающий на основе данных, получаемых от двух подсистем.
2. Приведены теоретические расчеты, используемые в алгоритме для определения координат цели на экранной плоскости.
3. Приведен способ построения проверочных линий.
4. Рассмотрен и выбран наиболее оптимальный способ объединения сигналов.
5. Рассмотрен и применен алгоритм фильтра Калмана для итеративного сглаживания результатов объединения сигналов.

## Глава 6. Реализация проекта

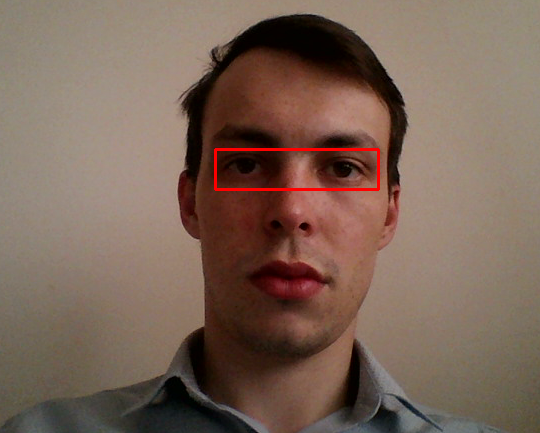
[Блок находится в стадии переработки]

## **6.1 Поиск пары глаз**

Как уже было сказано ранее, поиск пары глаз осуществляется с помощью каскадов Хаара. Библиотека OpenCV поддерживает работу с классификаторами Хаара, данный функционал встроен в библиотеку. В ходе разработки было выяснено, что лучше использовать классификаторы Хаара следующим образом:

1. Классификаторы нужно загрузить на этапе инициализации программы
2. Изображение требуется подготовить перед запуском поиска. Изображение приводится к черно-белому виду, так как в данном случае классификаторы отрабатывают более устойчиво.

Метод обнаружения Хаара принимает на вход изображение, каскад классификаторов Хаара, переменную для хранения результата, а затем несколько других необязательных параметров. Затем эта функция возвращает последовательность обнаруженных граней, состоящая из координат x и y, а также ширина и высота найденной пары глаз. Для более наглядного тестирования, вокруг пары глаз рисуется прямоугольник. Для проверки работы алгоритма, будем считать, что находится всегда только одна пара глаз, именно та, которая и будет управлять системой.



Пример обнаружения пары глаз

## **6.2 Определение зрачка**

Изображение пары глаз делится ровно пополам. Далее будет разъяснена реализация получения координат для одного глаза, но эта реализация справедлива и для другого.

Изначально изображение подвергается инверсии (изображение изначально цветное). Инверсия производится с помощью библиотеки AForge уже знакомой из [13]. Результат инверсии на изображении ниже.

Исходное изображение Инвертированное изображение

### **6.2.1 Бинаризация изображения**

Следующий шаг – это бинаризация изображения, использован для этого алгоритм Брэдли. Процесс бинаризации – это перевод цветного (или в градациях серого) изображения в двухцветное черно-белое. Главным параметром такого преобразования является порог t – значение, с которым сравнивается яркость каждого пикселя. По результатам сравнения, пикселю присваивается вес 0 или 1. Существуют различные методы бинаризации, которые можно условно разделить на две группы – глобальные и локальные. В первом случае величина порога остается неизменной в течение всего процесса бинаризации. Во втором изображение разбивается на области, в каждой из которых вычисляется локальный порог.

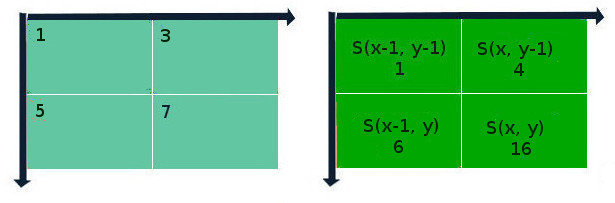
Существуют разные виды алгоритмов бинаризации изображения (метод Ниблэка, метод Кристиана). Наиболее быстрым из классических локальных алгоритмов считается метод Ниблэка, но он плохо работает с низкоконтрастными неровностями фона, которые в случае изображения глаза и близлежащего фона заполняют изображение чуть меньше, чем полностью. Чтобы исправить этот недостаток, было разработано несколько алгоритмов на основе метода Ниблэка, называемых „ниблэковскими“. В качестве примера, метод Кристиана, показывал хорошие результаты, и рассматривался как подходящий вариант. Однако на изображении с нестабильным расстоянием от камеры, после применения этого алгоритма появлялись искажения.

Трудность алгоритма бинаризации заключается в нахождении порогового значения, которое должно максимально надежно отделять символы не только от фона, но и от шума, теней, бликов и подобного. Часто для этого прибегают к методам математической статистики. В методе Брэдли подход с другой стороны – со стороны интегральных изображений.

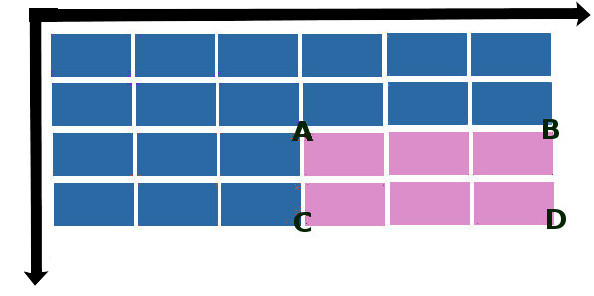
Интегральные изображения – это не только эффективный и быстрый (всего за один проход по изображению) способ найти сумму значений пикселей, но и простой способ найти среднее значение яркости в пределах заданного участка изображения.

Допустим, у нас есть 8-битное изображение в оттенках серого (цветное изображение можно перевести в оттенки серого, пользуясь формулой ). В таком случае, значение элемента интегрального изображения рассчитывается по формуле:

где S – результат предыдущих итераций для данной позиции пикселя, I – яркость пикселя исходного изображения. Если координаты выходят за пределы изображения, они считаются нулевыми. Для понимания принцип работы представлен на схеме ниже:

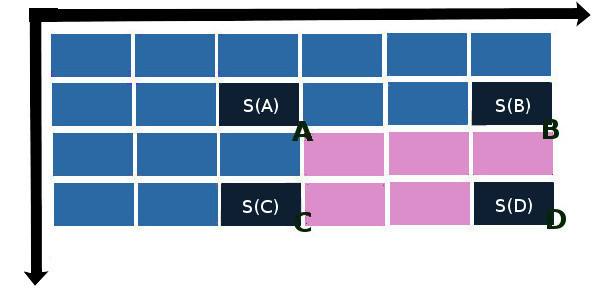


Хитрость алгоритма состоит в том, что, один раз составив интегральную матрицу изображения, можно быстро вычислять сумму значений пикселей любой прямоугольной области в пределах этого изображения. Пусть ABCD – интересующая нас прямоугольная область.



Тогда суммарная яркость S в этой области вычисляется по формуле:

где S(A), S(B), S(C) и S(D) – значения элементов интегральной матрицы в направлении на „северо-запад“ от пересечений сторон прямоугольника:



Реализацию метода можно посмотреть в листинге программы. Результат бинаризации:

Инвертированное (ЧБ) изображение Бинаризированное изображение

Применение бинаризации объясняется тем, что в таком случае на определения зрачка не влияет контраст и яркость изображения. Для того что б проверить данное утверждение был реализован модуль изменяющий контраст и яркость изображения. Результатом такой проверки стало подтверждение высказывания. Для дальнейшего снижения шума применяется сглаживающий фильтр Гаусса.

Следующий этап – поиск окружностей на изображении. Используемая для этого функция называется cvHoughCircles. Она ищет в изображении форму эллипса, которая упомянутая в нескольких работах. Чтобы функция обнаруживала только лучший круг на изображении, требуется правильно установить самый важный параметр – минимальное расстояние до размера изображения. Этот параметр представляет собой минимальное расстояние между двумя обнаруженными кругами. Для наглядности на изображении зрачка отображаются круги (те, что находит метод cvHoughCircles).

## 6.3 Поиск угла

Поскольку оба глаза обнаружены, и обнаружение углов использует ROI (Region Of Interest) от обнаружения глаз, то их достаточно просто вычислить. Угловое детектирование работает, уравнивая гистограмму изображения, а после вызывается метод OpenCV cvGoodFeaturesToTrack. Эта функция находит наиболее заметные углы изображения. Функция имеет параметр с именем maxCorners, который позволяет пользователю указать количество возвращаемых углов. Функция также оценивает углы по силе, поэтому, выбираем maxCorners = 1, ведь нужно получить только самый сильный угол. Проблема была быстро обнаружена, самый сильный угол в полном изображении глаза может быть либо глазным углом, либо ошибочным углом, возникшим из-за рельефа вокруг глаз. Таким образом, лучше организовывать поиск угла только в изображении зрачка, найденном шагом ранее. Далее данные о координатах зрачка и угла передаются в модуль математики MathModule, реализованный специально для более точного подсчета координат и дальнейшего применения аппроксимации координат курсора.

# Глава 7. Экспериментальная оценка эффективности системы

[

1. Реализация на основе алгоритма Виола-Джонса
2. Реализация приложения для создания обущающей (тестирующей выборки).
3. Сбор и сравнение данных

]

# Заключение

# Список литературы

1. А.Л. Ярбус Академия наук СССР «Роль движений глаз в процессе зрения». Москва 1996г.
2. Jacob, R. J. K. Eye tracking in advanced interface design, 1995.
3. Klingenhoefer, S., and Bremmer, F. Saccadic suppression of displacement in face of saccade adaptation. Vision Research 51, 8 (2011), 881 – 889. Perception and Action: Part II.
4. Гиппенрейтер Ю. Б. Движения человеческого глаза / Ю. Б. Гиппенрейтер.
5. Grossberg, S., Srihasam, K., and Bullock, D. Neural dynamics of saccadic and smooth pursuit eye movement coordination during visual tracking of unpredictably moving targets. Neural Networks 27, 0 (2012), 1 – 20
6. Mouse cursor control with head and eye movements: A low-cost approach by Yat-Sing Yeung, BSc. 1220 Vienna, 2012
7. Gao, D., Yin, G., Cheng, W., and Feng, X. Non-invasive eye tracking technology based on corneal reflex. Procedia Engineering 29, 0 (2012), 3608 – 3612. 2012 International Workshop on Information and Electronics Engineering.
8. P. Viola and M. Jones. Robust real-time face detection. IJCV 57(2), 2004
9. Lienhart R., Kuranov E., Pisarevsky V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: PRS 2003, pp. 297-304 (2003)
10. Huang, Y., Wang, Z., and Ping, A. Non-contact gaze tracking with head movement adaptation based on single camera, 2009.
11. Sewell, W., and Komogortsev, O. Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. In Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems. New York, NY, USA, 2010.
12. Torricelli, D., Conforto, S., Schmid, M., and DAlessio, T. A neuralbased remote eye gaze tracker under natural head motion. Computer Methods and Programs in Biomedicine 92, 1 (2008), 66 – 78.
13. Соловьев С.М., Разработка системы альтернативного управления компьютером. Управление цветным объектом. г. Ярославль, 2016г.
14. @Llammt, «Бинаризация изображений: алгоритм Брэдли», 2016. <https://habrahabr.ru/post/278435/>
15. Derek Bradley , Gerhard Roth Adaptive thresholding using the integral image, 2007.
16. Gaze Tracking Using A Regular Web Camera, Grahamstown, South Africa, 2012.
17. P. Viola and M.J. Jones, «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2001
18. P. Viola and M.J. Jones, «Robust real-time face detection», International Journal of Computer Vision, vol. 57, no. 2, 2004., pp.137–154
19. Р.Гонсалес, Р.Вудс, «Цифровая обработка изображений», ISBN 5-94836-028-8, изд-во: Техносфера, Москва, 2005. – 1072 с.
20. Минский, Пейперт, с. 11—18.
21. Розенблатт, Ф., с. 85—88.
22. Donald Olding Hebb. The Organization of Behavior: A Neuropsychological Theory. — Wiley, 1949. — 335 p. Современное издание: Donald Olding Hebb. [The Organization of Behavior: A Neuropsychological Theory](https://books.google.com/books?id=VNetYrB8EBoC). — Lawrence Erlbaum Associates, 2002. — 335 p.
23. Розенблатт Ф., с. 102.
24. Розенблатт Ф., с. 86
25. Хайкин С., 2006, с. 225—243, 304—316.

# ПРИЛОЖЕНИЕ А РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ ЭКСПЕРИМЕНТАЛЬНОГО ОБРАЗЦА СИСТЕМЫ

## А.1 Тестирование влияния внешних факторов на точность вычисления координат

Таблица 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № испытания | Время вычисления координат по изображению, мс | | | Расстояние до предыдущих координат, px | | |
|  | Отсутствие внешнего фактора | Использование очков | Использование контактных линз | Отсутствие внешнего фактора | Использование очков | Использование контактных линз |
| 1 | 30 | 35 | 31 | 5 | 10 | 7 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## А.2 Тестирование влияния разрешения входного изображения

Таблица 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № испытания | Время вычисления координат по изображению, мс | | | Расстояние до предыдущих координат, px | | |
|  | 640х480 | 1280х720 | 1920х1080 | 640х480 | 1280х720 | 1920х1080 |
| 1 | 30 | 35 | 31 | - | - | - |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## А.3 График влияния разрешения входного изображения

