МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Технология программирования
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка приложения компьютерного зрения для распознавания множества объектов на изображении»

Студент В.Я. Олексюк
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н, доцент, В.С. Климов
(ученая степень, звание, И.О. Фамилия)

Консультант М.В. Дайнеко

Тольятти 2020

АННОТАЦИЯ

Тема бакалаврской работы: «Разработка приложения компьютерного зрения для распознавания множества объектов на изображении».

В данной бакалаврской работе исследуется вопрос о применении алгоритма Виолы-Джонса при распознавании сразу несколько объектов разного типа на одном кадре изображения. В работе представлены результаты апробации алгоритма на практике.

Объект исследования – приложение компьютерного зрения в задаче распознавания множества объектов на изображении.

Предмет исследования – использование метода Виолы-Джонса в задаче распознавания множества объектов на изображении.

Цель исследования – разработать приложение компьютерного зрения для распознавания множества объектов разного типа на изображении с применением алгоритма Виолы-Джонса.

Структура бакалаврской работы представлена введением, тремя разделами, заключением и списком литературы.

Во введении описывается актуальность проводимого исследования, формулируется цель и ставятся задачи, которые необходимо решить.

В первом разделе производится анализ метода Виолы-Джонса, рассматриваются его модификации и производится постановка задачи.

Во втором разделе описывается проектирование программного обеспечения.

В третьем разделе предоставляется реализация и тестирование разработанного программного обеспечения, а также демонстрируются результаты работы.

В заключении представлены выводы по проделанной работе.

В работе использована 1 таблица, 25 рисунков, список литературы содержит 20 литературных источников. Общий объем выпускной квалификационной работы составляет 51 страницы.

ABSTRACT

The title of the graduation work is *Development of a computer vision system for recognizing multiple objects in an image*.

The graduation work consists of 1 table, 25 figures and the list of 20 references.

The given graduation work focuses on the use of the Viola-Jones algorithm for recognizing several objects of different types on a single image frame at once. In the research, the results of the algorithm practice test are presented as well.

The object of the investigation is a computer vision system in terms of recognizing multiple objects in an image.

The subject of the investigation is the use of the Viola-Jones method in terms of recognizing multiple objects in an image.

The aim of the research is to develop a computer vision system for recognizing multiple objects of different types in an image by using the Viola-Jones algorithm.

The graduation work consists of an introduction, three sections, conclusions, and a list of references.

The introduction explains the relevance of the research, sets the aim and the tasks that need to be solved.

In the first section, the Viola-Jones method is analyzed, its modifications are considered, and the problem is stated.

The second section describes software design process.

The third section features the implementation and testing of the developed software, as well as the results of the work.

In conclusion, the completed tasks and the results obtained during testing of the Viola-Jones algorithm are described.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 5

1 АНАЛИЗ СОСТОЯНИЯ ВОПРОСА ПРИМЕНЕНИЯ АЛГОРИТМА ВИОЛЫ-ДЖОНСА 7

1.1 Исследование предметной области 7

1.2 Описание работы алгоритма Виолы-Джонса 8

1.3 Рассмотрение модификаций алгоритма Виолы-Джонса 14

1.4 Постановка задачи 20

2 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ ПО РАСПОЗНАВАНИЮ МНОЖЕСТВА ОБЪЕКТОВ НА ИЗОБРАЖЕНИИ 23

2.1 Концептуальная модель 23

2.2 Логическая модель 25

2.3 Физическая модель 32

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ 34

3.1 Программная реализация приложения 34

3.2 Тестирование приложения 41

ЗАКЛЮЧЕНИЕ 47

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 49

ПРИЛОЖЕНИЕ А КОД ПРОГРАММЫ 52

ВВЕДЕНИЕ

С течением времени количество информации в мире становиться всё больше. Технологии не стоят на месте и постоянно появляются всё новые методы для организации обработки данных. Даже для простого анализа трафика или нарушений на дорогах уже задействованы своеобразные алгоритмы и такую задачу не требуется выполнять человеку. Однако, даже для больших компаний такая задача является достаточно сложной. Большое количество людей работает над обучением компьютера выполнять обычную для человека работу, в повседневной жизни, и это приносит свои плоды.

Актуальность темы исследования заключается в том, что распознавание множества объектов на изображении, является достаточно популярным направлением. Существует множество областей, где приложение компьютерного зрения может найти своё применение: от распознавания номерных знаков на автомобиле, который едет по проезжей части, до распознавания эмоций человека, чтобы узнать его душевное состояние.

В данной работе пойдёт речь о методе распознавания объектов под названием «Виолы-Джонса», о его преимуществах и недостатках, а также для решения каких задач он может подходить. Алгоритм будет задействован при обнаружении разного рода объектов на изображении. Также будет разобран вопрос, насколько он является эффективным при решении проблем в своей области. Существует множество разных алгоритмов предназначенных для локализации объектов, например SIFT, детектор Харриса, бинаризация изображений, операции математической морфологии и т.д., однако каждый хорош по-своему и имеет свои подводные камни.

Решения задачи с распознаванием объектов могут быть очень полезно, например возникающие экстренной ситуацией, в процессе работы алгоритма. Предположим, возник пожар при аварии на проезжей части. Приложение компьютерного зрения может оказать быстрый вызов служб спасения для оказания помощи пострадавшим [12].

Методы компьютерного зрения в основном разрабатываются для решения конкретных задач основываясь на выявлении значимых признаков распознаваемого объекта. Идеальных методов компьютерного зрения не существует. Если взять во внимание одну область и постараться сделать всё, чтобы порог ошибки был минимальный, метод всё равно будет обладать недостатками. Например, одним из недостатков методов является то, что их обучение на локализацию определённого объекта является достаточно трудоёмким процессом по времени, хоть и возрастает порог точности. Чем больше времени затрачено на обучения модуля компьютерного зрения, тем лучше будет проходить процесс обнаружения объектов, но ошибочные результаты алгоритма никуда не делись, лишь уменьшилось их количество.

Новизна исследовательской работы заключается в использовании алгоритма Виолы-Джонса при распознавании сразу нескольких объектов разного типа на одном кадре изображения.

Объект исследования – приложение компьютерного зрения в задаче распознавания множества объектов на изображении.

Предмет исследования – использование метода Виолы-Джонса в задаче распознавания множества объектов на изображении.

Цель исследования – разработка приложения компьютерного зрения для распознавания множества объектов разного типа на изображении с применением алгоритма Виолы-Джонса.

Для достижения поставленной цели необходимо решить следующие задачи:

- изучить особенности работы алгоритма Виолы-Джонса;
- разработать программное обеспечение для демонстрации работы алгоритма при локализации разного рода объектов с применением технологий распараллеливания;
- протестировать метод Виолы-Джонса на однопоточном варианте программного модуля и многопоточном.

1 АНАЛИЗ СОСТОЯНИЯ ВОПРОСА ПРИМЕНЕНИЯ АЛГОРИТМА ВИОЛЫ-ДЖОНСА

1.1 Исследование предметной области

В настоящее время существует множество разных подходов к упрощению жизни человечества. Но не все они достаточно эффективны на первый взгляд. Представим такую ситуацию, на дороге было совершено транспортное нарушение, камера зафиксировала всё происходящее и отправила отчёт в отдел по обработке фото-регистрационного материла для вынесения вердикта – виновен ли человек и нужно ли выписать ему штраф или нет [12]. Это делается для того, что искусственный интеллект имеет порог ошибки гораздо выше, чем человеческий взгляд на вещи. Если приложение само выносило заключение и составляла санкции на правонарушение это давало бы возможность гибко распределять человеческие ресурсы и направить их на более важные вещи.

Перед автором поставлена задача – разработать приложение компьютерного зрения для распознавания множества объектов на изображении.

Существует множество разнообразных алгоритмов по распознаванию объектов на изображении, к примеру: Eigenface основанный на представлении изображений как сумму базисных компонент, ISODATA использующий кластеризацию при заданном количестве групп, ERDAS Imagine максимизирующий спектральный отклик целевого вектора и минимизирующий влияние фона, основанного на ортогональном преобразовании корреляционной матрицы и т.д [8].

В рассмотрение будет взят алгоритм Виолы-Джонса разработанный Полом Виолой и Майклом Джонсоном в 2001 году. Изначально он был разработан для распознавания человеческих лиц, но также нашёл своё применение и для других объектов. Неотъемлемым плюсом данного алгоритма является то, что он способен достаточно быстро работать на

слабых устройствах. Из-за этого даже в те годы он показывал достаточно неплохие результаты по распознаванию объектов [9].

В результате приложение получит широкую область применения благодаря её возможностям, если взять за основу алгоритм Виолы-Джонса.

1.2 Описание работы алгоритма Виолы-Джонса

Точность анализа данных возрастает с каждым днём, как и скорость их обработки, а порог ошибки стремиться к нулевому значению. Вот и своё применение в данной сфере нашёл алгоритма Виолы-Джонса. Разрабатываемое приложение будет основано на данном алгоритме. Ранее было сказано, что алгоритм отлично показал себя на аппаратуре 2001 года, так как он включает в себя достаточно негромоздкие вычисления, благодаря чему можно быстро получить желаемый результат [8].

Алгоритм позволяет находить объекты на изображениях с высокой точностью и низким количеством ложных срабатываний.

Алгоритм Виолы-Джонса работает следующим образом: на вход поступает изображение, представленное в виде двухмерной матрицы, элементы которой – это пиксели с яркостью, расположенной в диапазоне от 0 до 255. Если изображение на входе является чёрно-белым, то достаточно одной двухмерной матрицы, если же изображение является цветным, то такое изображение должно быть преобразовано в три двухмерные матрицы, которые описывают цвета в формате RGB [13].

Одним из главных принципов метода является преобразование черно-белой изображения в интегральное представление (рисунок 1.1).

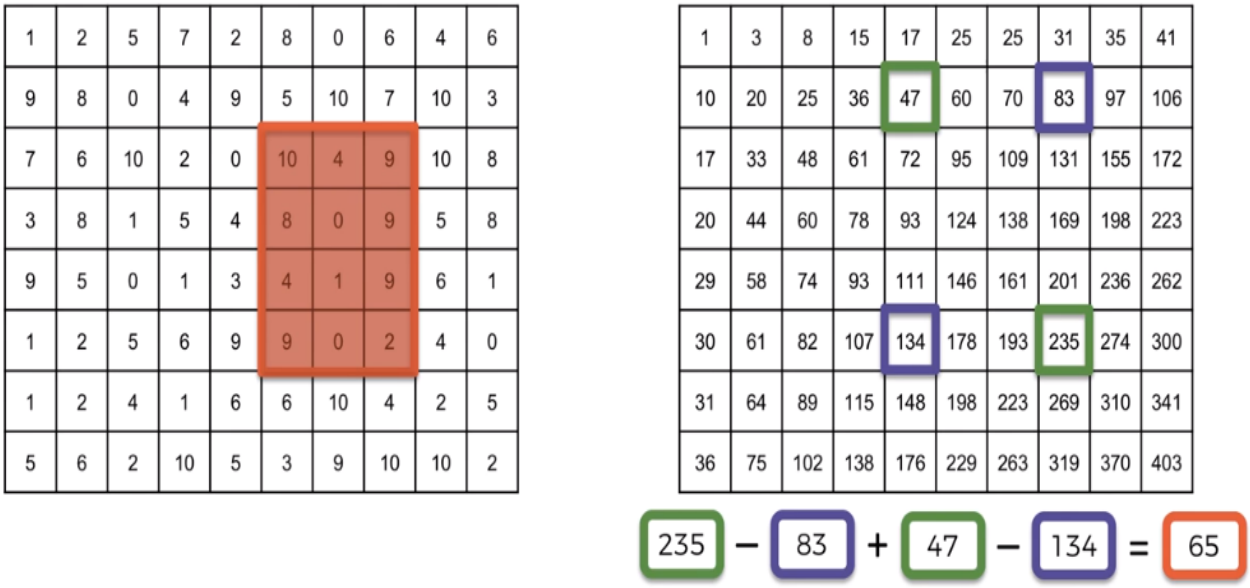


Рисунок 1.1 – Пример вычисления интегрального изображения

Данное представление используется зачастую и в других алгоритмах, к примеру, в вейвлет-преобразованиях, SURF и многих других разобранных алгоритмах [9]. В каждом элементе интегральной матрицы хранится сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента. Это действие позволяет посчитать суммарную яркость произвольного прямоугольника, не зависимо от его размеров на изображении, что отлично подходит для применения признаков Хаара. Интегральное представление по размеру совпадает с исходным. Если значений матрицы не существует, они считаются равными нулю. Подсчёты яркости в произвольном прямоугольнике производится по формуле:

$$II(x,y) \quad = \sum_{i=0,j=0}^{i\leq x,\leq y} I(i,j), \tag{1}$$

где I(i,j) – это яркость пикселя исходного изображения.

Элемент матрицы II(x,y) представляет собой сумму пикселов в прямоугольной области от элемента с индексами (0,0) до элемента (x,y). Время подсчёта такой матрицы – линейно. Для того, чтобы вычислить сумму яркостей пикселов, попадающих в область прямоугольника достаточно всего

четыре операции по обращению к массиву и три арифметические операции [6].

Рассмотрим прямоугольник ABCD (рисунок 1.2).

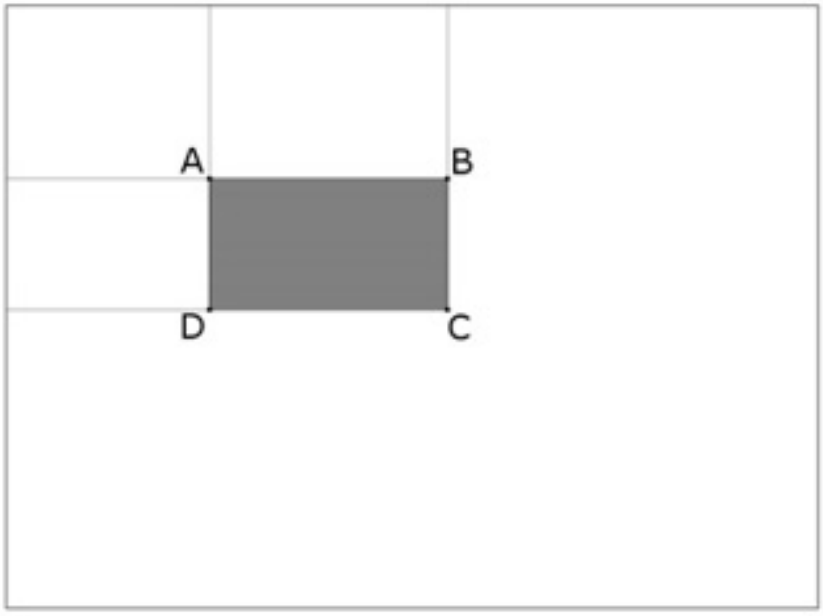


Рисунок 1.2 – Подсчёт суммарной яркости пикселов попадающий в прямоугольную область

Просуммировать значение яркости пикселов, попавших в прямоугольную область можно по формуле [7]:

$$\sum ABCD = II(A) + II(C) - II(B) - II(D), \quad (2)$$

где II(A), II(B), II(C), II(D) – смежные прямоугольники относительно выделенной области.

Работа алгоритма заключается в поиске некоторых признаков на изображении. Признаки, используемые алгоритмом Виолы-Джонса, похожи на признаки Хаара, однако они могут обладать сложной формой и содержат гораздо больше смежных прямоугольных областей [9]. Интегральным представление активно пользуется признаками Хаара. Они представляют собой двоичную аппроксимацию вейвлета Хаара. В стандартном методе Виолы–Джонса используются прямоугольные примитивы, то есть каждый

признак представляет собой двоичную маску – черно-белое изображение (рисунок 1.3).

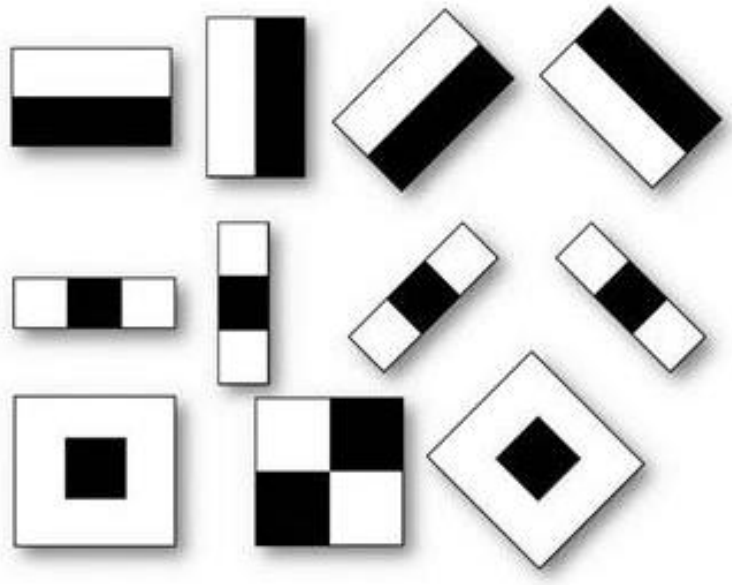


Рисунок 1.3 – Примитивы Хаара

С помощью данных примитивов выноситься заключение имеется ли в данной области распознаваемый объект или эту область следует убрать из рассмотрения [18]. Вердикт выноситься благодаря вычисляемому признаку по формуле:

$$F = X - Y, \tag{3}$$

где X – сумма яркостей пикселов закрываемой светлой частью признака;

Y – сумма яркостей пикселов под чёрной областью признака.

Данное значение признака сравнивается со значением в каскаде-классификаторе, обученном на распознавание определённого объекта, если же это значение больше определённого порога, следовательно, в данной области имеется распознаваемый объект, в противном случае область отбрасывается из рассмотрения.

Обучение каскадов-классификаторов осуществляется с помощью алгоритма AdaBoost [18]. Это алгоритм машинного обучения, разработанный

такими людьми, как Йоава Фройнд и Роберт Шапир. Отличительной чертой является то, что во время обучения строит целую конструкцию из базовых алгоритмов обучения с целью улучшить эффективности. AdaBoost является достаточно адаптивным алгоритмом так как каждый следующий классификатор строится по объектам, которые предыдущие классификаторы не смогли классифицировать верно. Каждому выбранному для обучения объекту задаются некоторые числовые характеристики, называемые весами. Эти весы характеризуют важность для алгоритма. Каждую итерацию, когда классификатор не смог локализировать объект, данные веса увеличиваются, что заставляют классификатор заострить на этом внимание.

Рассмотрим достоинства алгоритма AdaBoost:

- способность к обобщению. Алгоритм строит композиции, которые дают прирост к точности обучения, превосходя по качеству базовые алгоритмы, которые заложены в его основе. Обобщение можно увеличивать, дополняя алгоритм AdaBoost новыми базирующимися алгоритмами.
- вычислительные ресурсы, потребляемые алгоритмом AdaBoost не значительные. Время построения композиции зависит от времени обучения основополагающих алгоритмов;
- простота реализации;
- возможность обработать объекты, которые являются «шумом».

Так же, как и все придуманные алгоритмы человеком, в AdaBoost есть и свои недостатки:

- алгоритм переобучает каскады-классификаторов при значительных уровнях шума в исходных данных. Имеется экспоненциальная функция потерь, которая назначает высокие веса трудным объектам, на которых дают осечку большинство базовых алгоритмов. Нередко бывает ситуации, когда эти трудоёмкие объекты является шумом и AdaBoost начинает обучаться на него, что ведёт к переобучению. Это решается применения более мягких функций потерь;

- алгоритм должен получать на вход большое количество выборок;
- метод последовательного добавления может приводить к созданию неоптимального набора базовых алгоритмов. Решить данную проблему можно с мощью отката к ранее построенным алгоритмам;
- AdaBoost иногда строит громоздкие композиции из базовый алгоритмов обучения. Из-за этого повышается время обучения каскада-классификатора, также требуется больше памяти для хранения построенных композиций.

На рисунке 1.4 представлен пример работы метода Виолы-Джонса с обученным каскадом-классификатором для распознавания лиц.

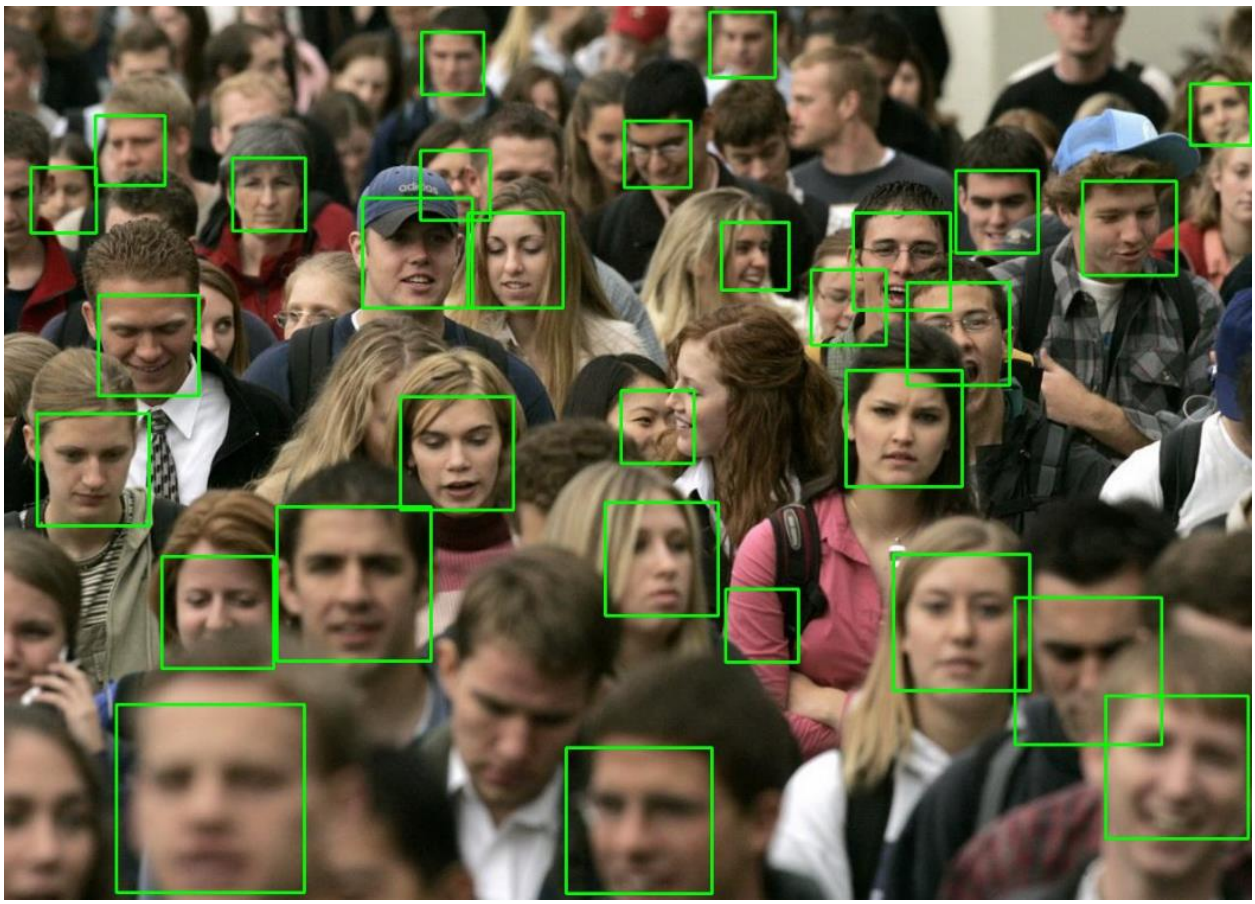


Рисунок 1.4 – Пример работы метода Виолы-Джонса

Как видно ни рисунке, алгоритм не может распознать некоторых людей в толпе, так как они загорожены другими людьми. Но он хорошо себя показывает при размытии изображения. Это связано с тем, что каскад был хорошо обучен на объёмной выборке.

Исходя из всего составляющего можно выявить некоторые достоинства и недостатки алгоритма Виолы-Джонса.

Достоинства:

- достаточно быстрое работа алгоритма;
- способность алгоритма распознавать любые объекты, следует только обучить каскад-классификатора;
- малое потребление вычислительных мощностей.
- высокая точность распознавания. Чем больше выборка, тем лучше точность.

Недостатки:

- алгоритм плохо реагирует на посторонние вещи, на объекте распознавания. К примеру, на человеке надеты солнце защитные очки;
- плохо сказывается распознавание при смене ракурса на объект.
- чувствительность к освещению;
- длительное время обучения каскадов-классификаторов.

Некоторые недостатки алгоритма решаются с помощью его модификаций, а также они увеличивают производительность.

1.3 Рассмотрение модификаций алгоритма Виолы-Джонса

Алгоритмы созданы для решения прикладных задач. Каждый из них отличается своими особенностями и у каждого есть свои недостатки, вот и метод Виолы-Джонса не стал исключением. Но человечество не стоит на месте, они придумали, как можно модернизировать столь привлекательное решение.

Существуют несколько способов модернизации алгоритма Виолы-Джонса:

- предобработка входного изображения;
- изменение алгоритма работы Виолы-Джонса.

К предобработке входного изображения относятся такие методы, как сглаживающий фильтр Гаусса, поиск границ на изображении с использованием оператора Преввита, Собеля или Лапласа и метод Канни [8].

Изменение алгоритма работы Виолы-Джонса включает в себя метод RASW.

Познакомимся с некоторыми способами модификаций метода Виолы-Джонса, представленными выше.

Объём работы алгоритма Виолы-Джонса тесно связано с разрешением входного изображения. Если размер изображения увеличится, то также увеличится количество вычислений, а это не очень хорошо при работе метода в реальном времени. Для этого был придуман метод RASW - Runtime Adaptive Sliding Window. Его ещё просто называют сканирующее окно. Где размер сканирования определяется некоторым числом. Шаг сканирования сильно влияет на точность распознаваемого объекта, а также характеризует пропускную способность. Если задать слишком большой шаг, то пропускная способность увеличится, но в свою очередь, точность распознавания уменьшится, так как не все объекты могут попасть в распознаваемое окно и будут пропущены [9].

Для перемещений сканирующего окна следовало найти особую закономерность, чтобы окно двигалось быстрее в областях, которые не содержат распознаваемого объекта, и медленнее в непосредственной близости от объекта. Исходя из этого можно было повысить скорость сканирования без ухудшения качества. Также особым плюсом было бы то, что шанс ошибки классификатора снизилась обнаружить не искомый объект в другом месте, где его вообще нет.

В стандартной реализации метода Виолы-Джонса шаг задаётся маленьким значением, к примеру один или два. Так как высокое значение приводит к ухудшению качества распознавания.

Создатели алгоритма RASW провели анализ и обнаружили взаимосвязь между наличием искомого объекта в области изображения и номером

классификатора каскада, на котором данное окно считается отброшенным. В областях, которые содержат только фон – окно отвергается раньше, чем находясь в непосредственной близости к распознаваемому объекту. В результате пришли к такому выводу, что чем ближе искомый объект, тем выше ступень выхода и наоборот.

Алгоритм работы метода RASW будет представлен на рисунке 1.5.

Require: scaled input image X, classifier cascade S
Ensure: vector V of detected faces (bounding boxes)

```
1:  for y ← 0 to X.height − subwindow.height do
2:      Δ_x ← 1
3:      for x ← 0 to X.width − subwindow.width do
4:          Δ_x ← Δ_x − 1
5:          Δ_y[x] ← Δ_y[x] − 1
6:          if Δ_x = 0 AND Δ_y[x] = 0 then
7:              exit-stage ← S(x, y)
8:              if exit-stage = n then
9:                  R ← (x, y, width, height)
10:                 push R into V
11:             end if
12:             if exit-stage < Δ_{x,t1} then
13:                 Δ_x = Δ_{x,max}
14:             else if Δ_{x,t1} ≤ exit-stage < Δ_{x,t2} then
15:                 Δ_x = Δ_{x,nom}
16:             else
17:                 Δ_x = Δ_{x,min}
18:             end if
19:         else if Δ_x = 0 then
20:             Δ_x ← Δ_{x,min}
21:         else if Δ_y[x] = 0 then
22:             Δ_y[x] ← Δ_{y,min}
23:         end if
24:     end for
25: end for
```

Рисунок 1.5 – Алгоритм метода RASW

Каждый сдвиг сканирующего окна запускает каскад-классификатор, который возвращает ступень выхода (строка 7). После идёт проверка условия, что, если ступень выхода равна n – числу классификаторов в каскаде, следовательно распознаваемый объект найден (строка 8) и

координаты текущего окна помещаются в вектор V (строка 10). Шаг окна может принимать разные значения по оси абсцисс и ординат, в зависимости от ступени выхода предыдущего положения окна. Для того, чтобы сдвиг окна был правильным, следует хранить информацию о ступени выхода предыдущего положения окна. На каждой итерации алгоритма шаг по оси абсцисс и оси ординат уменьшаются на единицу, когда обе переменные будут равны нулевому значению, то окно будет обработано каскадом-классификаторов [19]. Если только одно из этих значений станет равно нулю, то это значение принимается за минимальную величину шага (строки 20 и 22) [8].

Достоинством данного метода является ускорение алгоритма Виолы-Джонса без потерь качества распознавания, а именно быстрое перемещение распознаваемого окна по областям, где не содержится распознаваемый объект и медленное перемещение в его близости.

Недостатки метода:

- сложность реализации;
- влияние шума на область улучшения данного метода. При большом количестве шума, скорость работы будет уменьшаться.

Следующей модификацией будет особо известный алгоритм, разработанный Джоном Канни на заре компьютерного зрения в 1986 году. Он называется «Метод выделения границ Канни».

Алгоритм состоит из пяти шагов:

- размытие изображения для удаления шума (сглаживание);
- поиск градиентов. Выявление максимальных значений градиентов на изображении и отрисовка границ;
- подавление не максимальных градиентов. Максимумы являющиеся локальными отмечаются как градиенты;
- фильтрация границ. Выявление потенциальных границ называющимися – порогами;

- исправление неоднозначностей. Результирующие границы выявляются подавлением всех краёв, не относящихся к определенным границам.

Создавая этот алгоритм за основу, были взяты следующие критерии:

- устойчивость к шуму;
- качественная локализация границ объектов;
- одна граница объекта должна вызывать лишь один отклик алгоритма.

Благодаря данным критериям была определена функция стоимости ошибок, в результате минимизации значений которой, был найден линейный оператор, производящий свертку изображения.

Главной особенностью алгоритма Канни, по сравнению с другими алгоритмами, было вычисление градиента сглаженного изображения, а также удаление точек, лежащих вблизи границы, что позволило определить грань целиком, без разрывов и заломов, вблизи локальных максимумов градиента. Избавление от слабых границ обеспечило использование этих двух порогов. Если кусок границы обрабатывающейся, как неделимое целое, на протяжении всего этого фрагмента ни разу не будет удовлетворять условию преодоления этих двух порогов, то этот фрагмент исключается из рассмотрения. Такое условие позволяет в разы уменьшить количество разрывов в конечных границах. Данное использование шумоподавления создаёт чёткие и устойчивые результаты, также увеличивает вычислительную нагрузку, создаваемую алгоритмом, и это приведёт к потере деталей границ. В результате углы будут значительно сглажены у распознаваемых объектов на изображении, а также исчезнут границы в различных точках соединений [8].

Исходя из сказанного выше алгоритм границ Канни работает следующим образом: первым делом происходит удаление шума, а именно сглаживание, которое можно представить в виде формулы Гаусса для двухмерного случая:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \quad (4)$$

где σ – это коэффициент сглаживания, чем он больше, тем сильнее сглаживается входное изображение.

После процесса удаление шумов (сглаживания) производится вычисление градиентов сглаженного изображения. Оно вычисляются в двух направлениях, в вертикальном и горизонтальном представлении. Данная операция рассчитывается на основе первой производной. Значения, полученные в результате подобной обработки, объединяются по формуле:

$$E_{ij} = \sqrt{g_{v\,ij}{}^2 + g_{h\,ij}{}^2} \quad (5)$$

где g_v – вертикальное представление;

g_h – горизонтальное представление.

Направление градиента вычисляется по формулам:

$$\theta_{ij} = tan^{-1} \frac{g_{v\,ij}}{g_{h\,ij}} \quad (6)$$

$$E_{T\,ij} = \begin{Bmatrix} E_{ij}\text{, если } E_{ij} > T, \\ 0 \text{ иначе} \end{Bmatrix}$$

где θ_{ij} – угол направления вектора;

T – порог удаления шумов.

Коэффициент T подставляется таким образом, что границы распознаваемого объекта будут выделены и останутся в результате без изменений, после удаления большей части шумов.

Следующая операция алгоритма Канни подавления не максимальных градиентов. Чтобы это сделать следует осуществлять обнуление градиентов по двум порогам T1 и T2, причем с выполнением условия. Рассматриваемый градиент нельзя обнулять, пока его значение в выбранной точке меньше, чем порог T1 и больше, чем порог T2 [9]. Данные действия производятся в расчет обнаружения наиболее значимых контуров на изображении.

Далее будет представлен пример работы алгоритма Канни (рисунок 6).

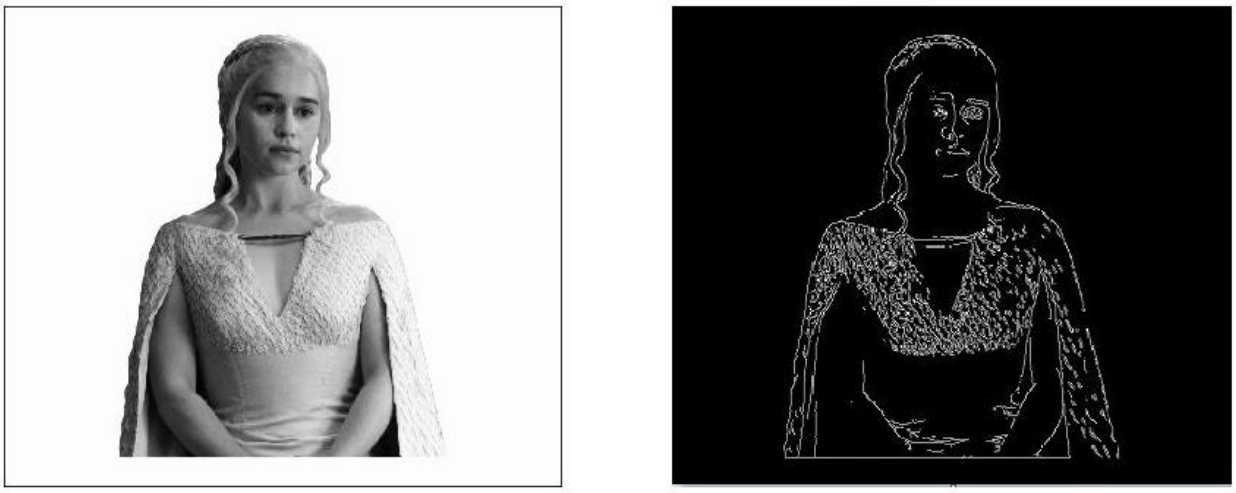


Рисунок 1.6 – Пример работы метода выделения границ Канни

Достоинством данного улучшения алгоритма Виолы-Джонса является низкий коэффициент ошибки при поиске объекта на изображении [9].

Недостатками являются:

- высокой вычислительной сложность;
- некоторые реализации Канни подвержены высокой чувствительности к шуму.

В результате, приведённые модификации выше показывают то, что существу методы улучшения алгоритма Виолы-Джонса и у каждой модификации есть свои минусы и плюсы, но в рассмотрение будет взята классическая реализация метода Виолы-Джона.

1.4 Постановка задачи

Отталкивая от того, для чего был создан алгоритм Виолы-Джонса, он так же нашёл своё применение в распознавание объектов разного рода, вне зависимости от формы, цвета, расположения в пространстве. И по сей день он достаточно хорошо показывает себя в задачах по распознаванию искомых объектов. Исходя их этого, напрашивается задача, как покажет себя алгоритм Виолы-Джонса при нагрузке. А также появляются вопросы, какое количество каскадов-классификаторов заставит изменить производительность метода в

худшую сторону и сколько объектов за раз можно будет распознавать алгоритмом Виолы-Джонса при комфортной работе? Ответы на данные вопросы даст разрабатываемое приложение по распознаванию множества объектов на изображении с использованием данного метода.

В качестве примера, что должно быть результатом работы разрабатываемого приложения приведём рисунок 1.7.

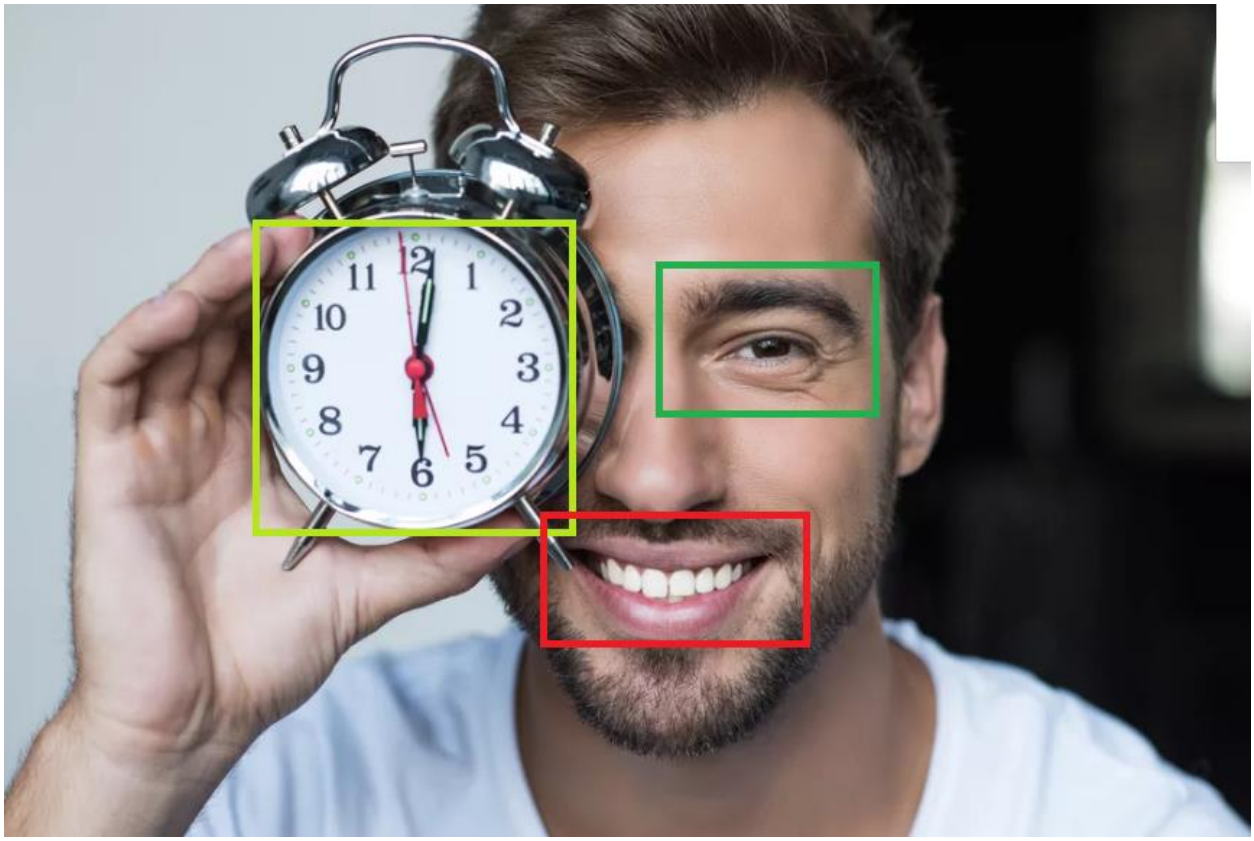


Рисунок 1.7 – Пример результата работы разрабатываемого приложения

Данное изображение является примерным результатом работы приложения по распознаванию множества объектов на изображении. Как видно, на изображении было распознано несколько объектов, таких как глаз человека, его улыбка и будильник в руке. Чтобы добиться данного результата следует предусмотреть следующие пункты:

- приложение по распознаванию множества объектов на изображении должна поддерживать, как многопоточную реализацию распознавания разного рода объектов, так и однопоточный подход;

- поддержка модулей приложения по распознавания объектов, как на изображении, так и на входном видео потоке;
- приложение обязано фиксировать результаты свой работы (вести лог журнал, сохранять фотографию после обработки);
- поддержка добавления и удаления каскадов-классификаторов в приложении;
- внедрение тест кейсов на базе Unit-тестов или Cucumber тестирования для поддержки работоспособности программы во всевозможных способах взаимодействия;
- приложение должно обладать дружественным интерфейсом пользователя для удобного и интуитивного взаимодействия;
- в результате работы приложение обязано демонстрировать полученные результаты в реальном времени.

Таким образом, было проанализирована предметная область разрабатываемого программного решения. Проанализирован алгоритм Виолы-Джонса. Приведены способы модификации данного алгоритма. Было выявлено, для чего этот алгоритм может быть полезен в текущее время.

Исходя из полученных данных были поставлены требования к программному обеспечению и выявлены ожидаемые результаты.

Проделанные в данной главе действия необходимы для успешного проектирования приложения по распознаванию множества объектов на изображении с использованием алгоритма Виолы-Джонса.

2 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ ПО РАСПОЗНАВАНИЮ МНОЖЕСТВА ОБЪЕКТОВ НА ИЗОБРАЖЕНИИ

2.1 Концептуальная модель

Человек привык дробить огромные задачи на более мелкие с целью углубленного изучения решаемой задачи, экономии времени и разбор непонятных ракурсов проблемы. Для этого придуманы довольно много удобных методологий и парадигм, что гораздо упрощают разработку программных решений.

Для понимания работы разрабатываемого приложения по распознаванию множества объектов на изображении была разработана диаграмма бизнес-процесса на базе методологии IDEF0.

IDEF0 — это методология функционального моделирования и графическая нотация, предназначенная для описания и формализации бизнес-процессов. IDEF0 выделяется одной очень яркой особенностью – это ее акцент на подчиненности объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность. Также она демонстрирует связи между функциями разрабатываемого программного обеспечения (рисунок 2.1) [10].

Рисунок 2.1 – IDEF0-диаграмма распознавания объекта

Как можно увидеть на приведённом выше рисунке в приложении компьютерного зрения (ПКЗ), которая является механизмом, поступает на вход кадр изображения, на котором находится искомый объект. Для этого изображения в качестве объектов управления используется уже подгруженный каскад-классификаторов и сам алгоритм Виолы-Джонса, с помощью которого производится распознавание объекта и выделение области на результирующем выходе. Эта диаграмма подходит для понимания работы к двум требуемым к разработке модулям:

- распознавание объекта на изображении;
- распознавание объекта на видео потоке.

Ещё одним не мало важной процедурой является загрузка уже обученного каскада-классификаторов в приложение по распознаванию объекта (рисунок 2.2).

Рисунок 2.2 – IDEF0-диаграмма загрузки обученного каскада-классификаторов в приложение

На данном рисунке можно увидеть, как на вход в приложение поступает новый каскад, к нему в качестве элементов управления

применяются правила валидации, к примеру: в приложении не должно иметься двух одинаковых каскадов-классификаторов, у них должно быть расширение «xml» и так далее. После успешной валидации в приложении компьютерного зрения подгруженный каскад готов к работе и его можно использовать.

2.2 Логическая модель

Методика разработки логической модели подразумевает под собой отражение функциональные аспекты логической составляющей приложения с помощью диаграммы вариантов использования, рассмотрение элементных аспектов логической составляющей приложения на основе диаграммы классов предметной области, отражение динамический аспект логической составляющей приложения по средству построения диаграммы последовательности. Логическая модель строиться полностью на базе UML-представления.

UML – язык моделирования, предназначенный для объектно-ориентированного анализа и проектирования. Он содержит в себе три основных идеи: диаграммы, сущности и связи между ними [10].

- сущности – объекты абстракции, которые являются основными составляющими модели;
- связи – элементы представления свойством который является соединение сущностей между собой;
- диаграммы – блоки, представляющие группировки сущностей и связей между собой.

Язык UML необходим при визуализации, конструирования и документирования разрабатываемого программного решения.

Одним из важных показателей при бизнес-анализе является применение диаграммы вариантов использования, которая даст

представление о том, кто именно сможет использовать приложение и какие действия ему дозволено в нём делать (рисунок 2.3) [10].

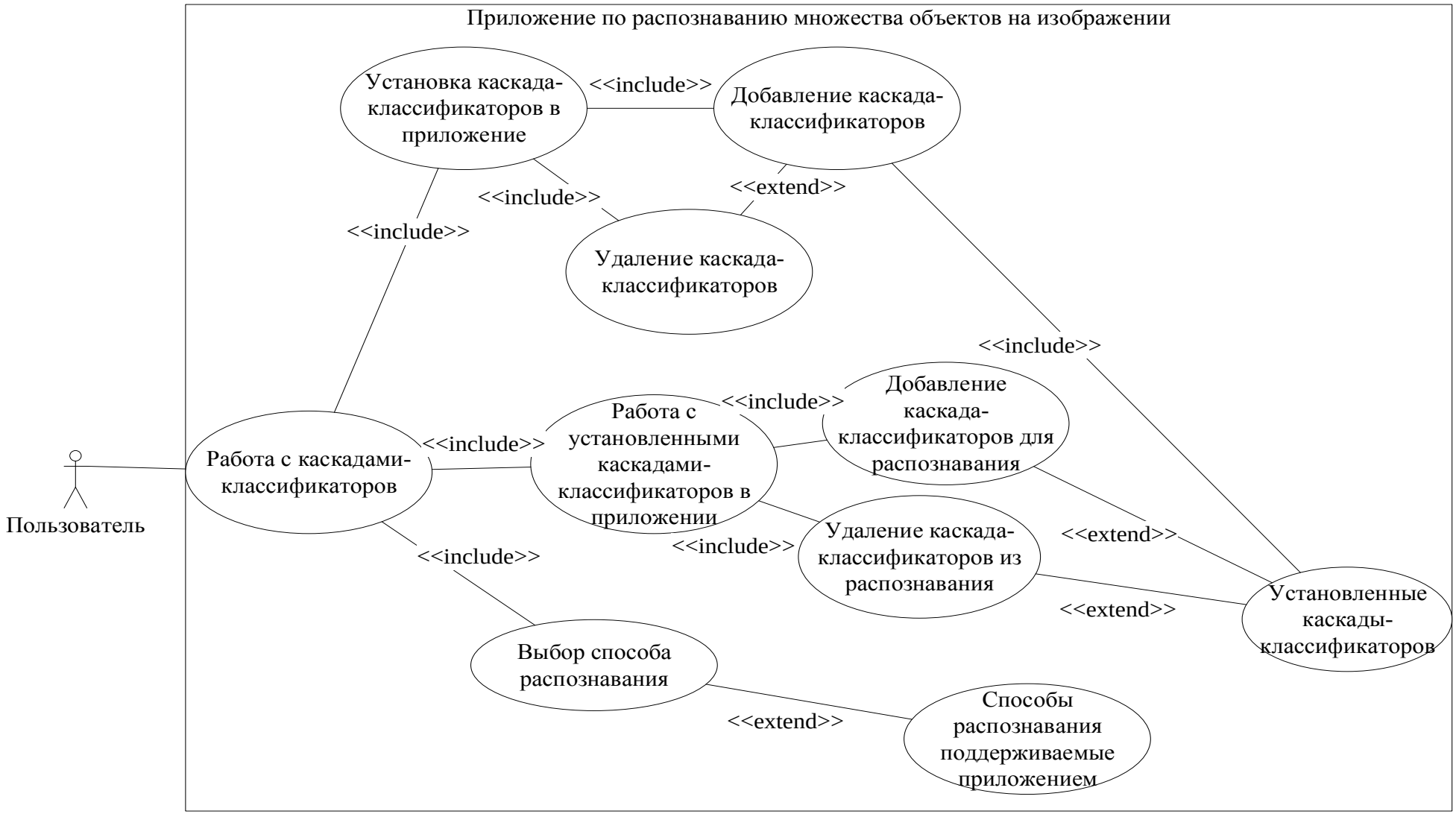

Приложение по распознаванию множества объектов на изображении
Установка каскада-классификаторов в приложение
<<include>>
Добавление каскада-классификаторов
<<include>>
<<extend>>
<<include>>
Удаление каскада-классификаторов
<<include>>
Работа с каскадами-классификаторов
<<include>>
Работа с установленными каскадами-классификаторов в приложении
<<include>>
Добавление каскада-классификаторов для распознавания
Пользователь
<<include>>
Удаление каскада-классификаторов из распознавания
<<extend>>
<<extend>>
Установленные каскады-классификаторов
<<include>>
Выбор способа распознавания
<<extend>>
Способы распознавания поддерживаемые приложением

Рисунок 2.3 – Диаграмма вариантов использования разрабатываемого приложения

Основными составляющими элементами диаграммы являются актеры, сущности, которые выполняют действия и варианты использования, то есть возможные действия, выполняемые актерами.

На данной диаграмме видно, что имеется один актёр, то есть пользователь, который может взаимодействовать с программой. Пользователю разрешено удалять и добавлять существующие каскады-классификаторов для распознавания в программе. Также разрешено добавлять новые каскады-классификаторов в приложение и удалять их. Возможен выбор пользователем поддерживаемый приложением способ распознавания объекта.

Неотъемлемой частью логического моделирования являются диаграммы классов, используемые при моделировании и проектировании приложений. Данные диаграммы описывают приложение в статическом представлении, демонстрируя ее структуру.

Диаграмма классов включает в себя некоторое множество составных элементов, которые в свою очередь отражают декларативные знания о предметной области. Декларативность интерпретируются в классы, интерфейсы и отношения между ними, которые являются основными понятиями языка UML [10].

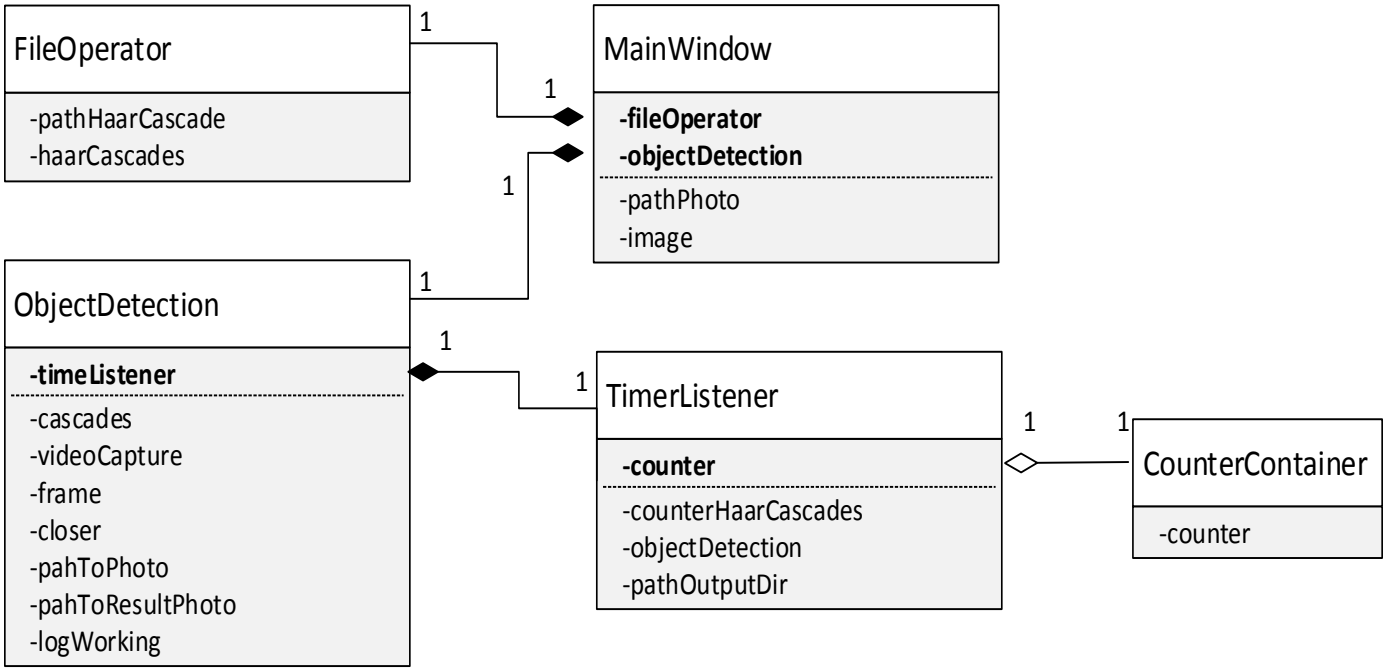


Рисунок 2.4 – Диаграмма классов разрабатываемого приложения

На данной диаграмме изображено пять классов, имеющих между собой отношения композиции и агрегации. Рассмотрим семантическую составляющую каждого класса:

- MainWindow – это класс отвечающий за главное окно графического интерфейса, включающее в себя все составные части программы. В нём будут находится взаимодействия с каскадами-классификаторов, логирование результатов работы программы и выбор способа распознавания;
- FileOperator – функциональный помощник в работе с файлами программы, а именно сбор информации о существующий каскадах-классификаторов в приложении, взаимодействия с этими файлами (добавление и удаление) и нужный для работы приложения дополнительный вспомогательный функционал при обработке файлов;
- ObjectDetection – класс включающий в себя главные модули разрабатываемого приложения по распознаванию объектов, а именно реализацию метода Виолы-Джонса с применением его на входное изображение и видео поток данных, также поддержку встроенного логирования результатов;

- TimerListener – класс отвечающий за отдельный поток программы для получения кадров в секунду и вывод результатов в файл;
- CounterContainer – класс отвечающий за инкрементирования счётчика кадров для класса TimerListener.

Также следует напомнить, что наша программа будет иметь сразу два поддерживаемых модуля для распознавания объектов на изображении (распознавание объекта на изображении, загруженном пользователем и распознавание объекта на видео потоке) и каждый модуль будет иметь аналоги, но с подходом распараллеливания предлагаемым языком программирования. Распараллеливаться будет часть, отвечающая за распознавание объекта, то есть, каждый поток будет брать себе свой собственный каскад-классификаторов и применять его на входной кадр изображения с использованием алгоритма Виолы-Джонса.

Первым делом для распараллеливания модуля должна быть спроектирована однопоточная вариация алгоритма. Это сделано для того, чтобы выявить особо критичные части по вычислительной нагрузке. Чтобы это сделать, воспользуемся блок-схемой – графическим представлением, описывающим алгоритмы и процессы, в которых отдельные шаги представляется в виде блоков различной формы, соединенных между собой линиями, указывающими направление последовательности (рисунок 2.5) [10].

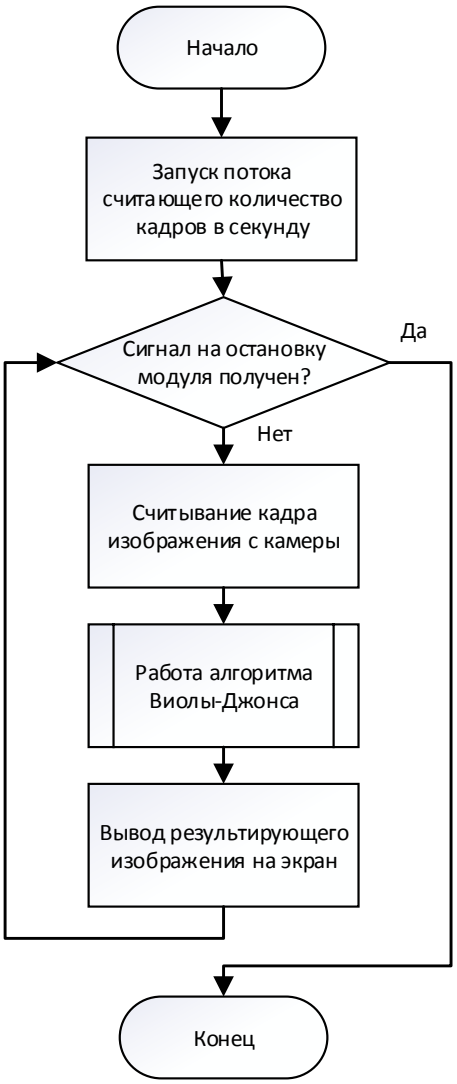


Рисунок 2.5 – Блок-схема обобщённого варианта однопоточного модуля по распознаванию объекта с видео потока

На данной рисунке блок процесса, отвечающий за запуск потока, считывающий количество кадров в секунду, осуществляется до цикла, следовательно только один поток будет следить за тем, как в совокупности наш алгоритм работает при однопоточной реализации во время нагрузки. Это даст возможность определить на сколько алгоритм эффективно работает при увеличении количества каскадов-классификаторов. Также имеется цикл, отвечающий за окончание работы модуля ждущий сигнала от пользователя программы. Входной кадр для распознавания поступает с помощью камеры. Далее в дело вступает сам алгоритм Виолы-Джонса и выдаёт результат, который в последствии выводится на экран.

Смотря на блок-схему, было выдвинуто решение распараллелить процесс распознавания объекта между потоками, назначив каждому из них свой собственный каскад-классификаторов (рисунок 2.6).

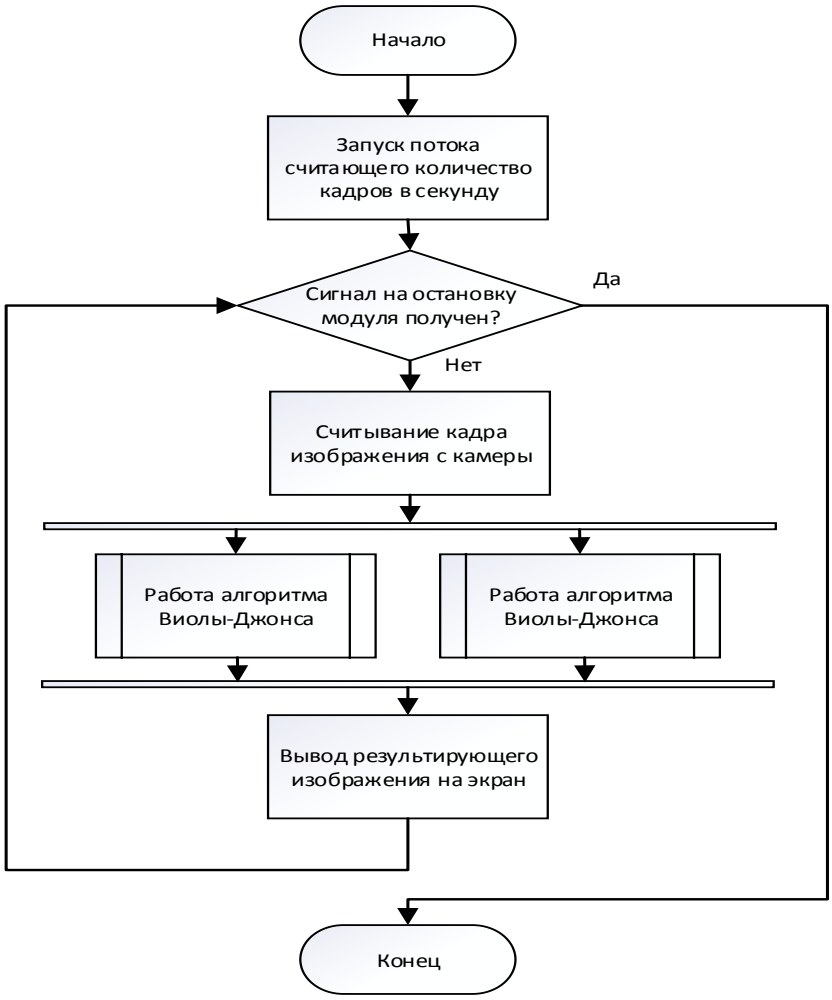


Рисунок 2.6 – Блок-схема обобщённого варианта многопоточного модуля

Распараллеливание части по распознаванию объекта на изображении даст хороший результат в производительности, так как это действие будут выполнять несколько потоков, в зависимости от того, сколько каскадов-классификаторов выберет пользователь.

Выше были показаны однопоточная и многопоточная вариации модуля по распознаванию объектов на видео потоке. Модуль, отвечающий за распознавание на изображении, будет иметь схожую структуру, только он будет отличаться лишь в источнике изображения, оно будет поступать из операционной системы, так как пользователь имеет возможность загружать свои собственные фотографии.

2.3 Физическая модель

Разнообразные составляющие элементы логической модели существуют материально или физически. Они показывают понимание строения разрабатываемого приложения и точку зрения его поведения. Чтобы сконструировать конкретную физическую модель приложения необходимо реализовать все элементы логического представления в материальные сущности. Для рассмотрения данных сущностей предназначен аспект модельного представления, а именно физическое представление модели.

Главной составляющей физической модели является диаграмма развертывания. Она используется, как приложение к техническому заданию. Такую диаграмму составляет менеджер проекта перед тем, как начать обсуждение архитектуры приложения с разработчиками. Она может стать хорошим стартом проекта, потому что, именно в основополагающий понятиях аппаратуры для многих систем формируется существенная часть их фундаментальных требований.

Диаграмма развертывания может давать хорошее понимание общего представления о приложении людям, которые не являются разработчиками программного решения, так как содержит минимум программистских деталей, которые нужно выделить [10].

Далее на рисунке 2.7 будет представлена диаграмма развёртывания для приложения по распознаванию множества объектов на изображении.

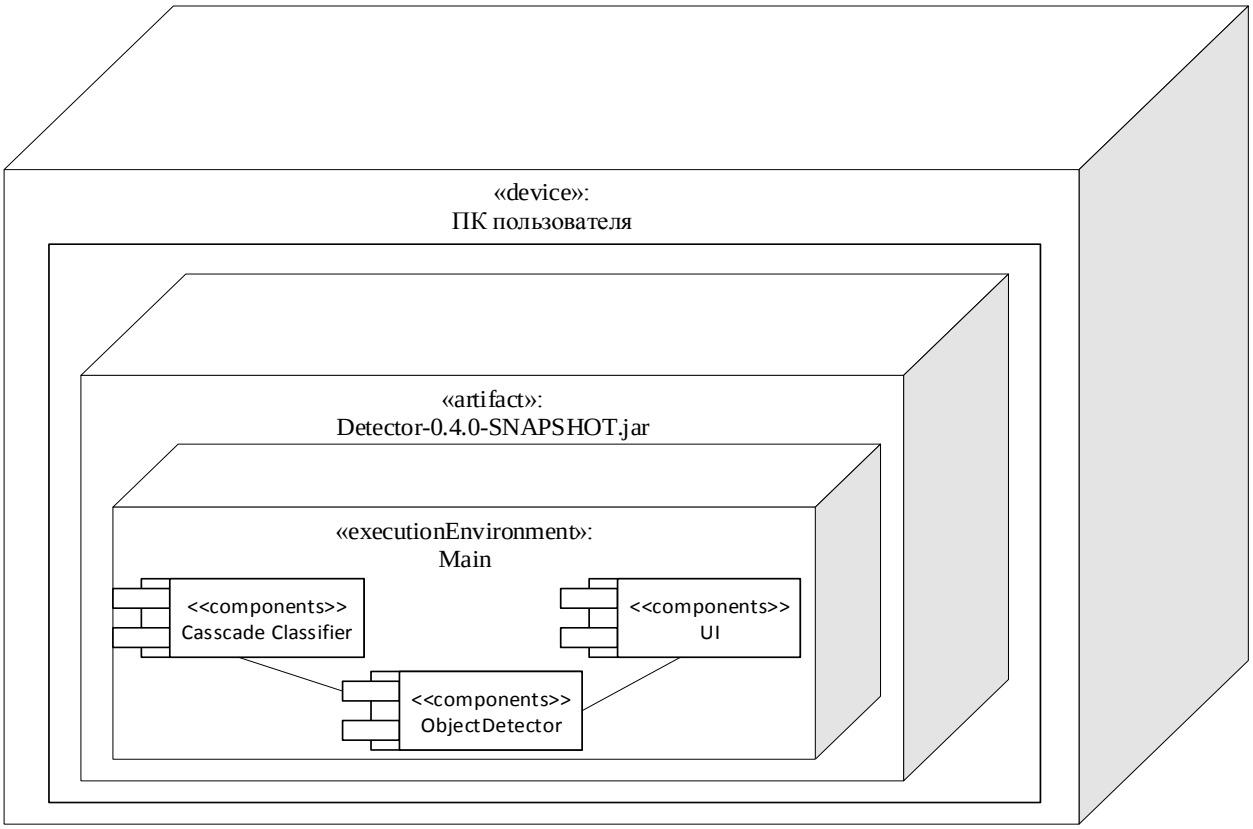


Рисунок 2.7 – Диаграмма развёртывания приложения по распознаванию объектов

Персональный компьютер пользователя содержит в себе установленное приложение по распознаванию объектов. Само приложение представляет из себя запакованный Java-архив с расширением «jar», включающий в себя набор ключевых компонентов «Casscade Classifier», «Object Detector» и «UI», которые выполняют главные функции приложения.

Таким образом были разработаны концептуальная, логическая и физическая модели приложения. Это поможет грамотно реализовать приложение по распознаванию множества объектов с использованием алгоритма Виолы-Джонса и все его важные составляющие части, с точки зрения объектно-ориентированного языка программирования.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ

3.1 Программная реализация приложения

В ходе выполнения исследовательской работы была задействована среда разработки IntelliJ IDEA.

IntelliJ IDEA — это интегрированная среда разработки программного обеспечения, разработанная компанией JetBrains для многих языков программирования, в особенности Java, JavaScript и Python.

Для реализации приложения по распознаванию множества объектов на изображении был выбран язык объектно-ориентированного программирования Java. Главным плюсом языка является то, что достаточно написать один раз программу и она будет работать на разных операционных системах, где будет установлена Java-машина. Неотъемлемой частью является наличие уже готовых библиотек, которые могут пригодиться при разработке, чтобы «не создавать велосипед зонного» [20]-[17].

Для использования алгоритма Виолы-Джонса было решено воспользоваться библиотекой OpenCV с открытым исходным кодом, которая включает обширный выбор алгоритмов компьютерного зрения [11].

Для того, чтобы реализовать многопоточную идентификацию разного рода объектов было принято решение воспользоваться встроенным классом Thread в язык программирования Java. Данный класс инкапсулирует стандартные механизмы работы с потоками, которые достаточно удобны при использовании [15].

Функциональные возможности программы:

- поддержка множественного распознавания объектов разного типа;
- реализован выбор способа распознавания из двух возможных (распознавание объектов на изображении и распознавание на видеопотоке);
- поиск и выделение объектов разного типа и формы на изображении с использованием метода Виолы-Джонса;

- реализована возможность одновременного использования сразу нескольких каскадов-классификаторов для распознавания множества объектов на изображении;
- поддержка подключения своих собственных каскадов-классификаторов в приложение;
- поддерживается валидация подключаемых каскадов-классификаторов для избегания неправильной работы программы;
- поддержка логирования эффективности работы программы в реальном времени при распознавании множества объектов на видеопотоке, в качестве кадров в секунду;
- реализовано сохранение результирующего кадра с выделенными объектами при распознавании на изображении.

Проект разработанного приложения в среде IntelliJ IDEA представлен на рисунке 3.1.

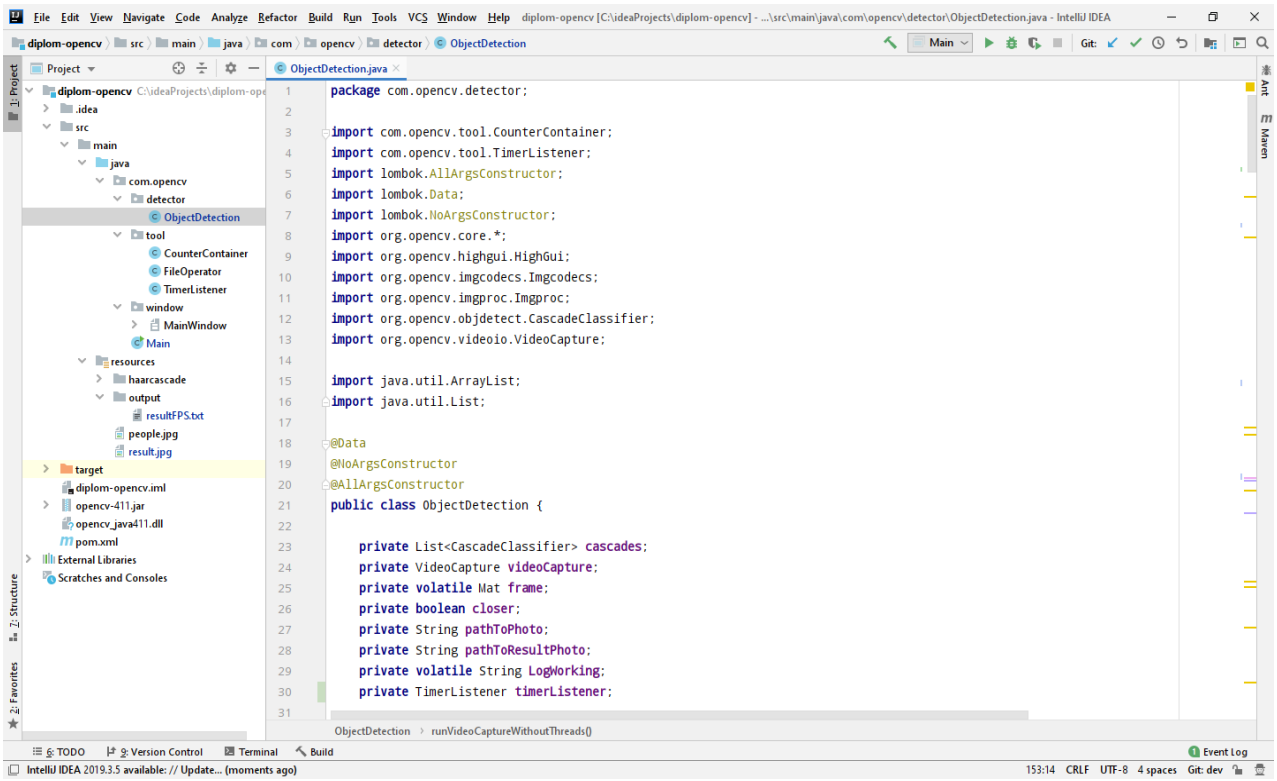


Рисунок 3.1 – Проект приложения в среде IntelliJ IDEA

В качестве системы контроля версий была выбрана платформа — GitLab. Данная платформа управляет Git-репозиториями, помогает разработчикам вести непрерывный процесс развертывания для тестирования, создания и деплоя кода, следить за ходом тестов, повышать контроль над качеством, фокусирования на построении продукта вместо настройки инструментов.

На рисунке 3.2 можно увидеть созданный репозиторий на платформе GitLab для приложения по распознаванию множества объектов на изображении.

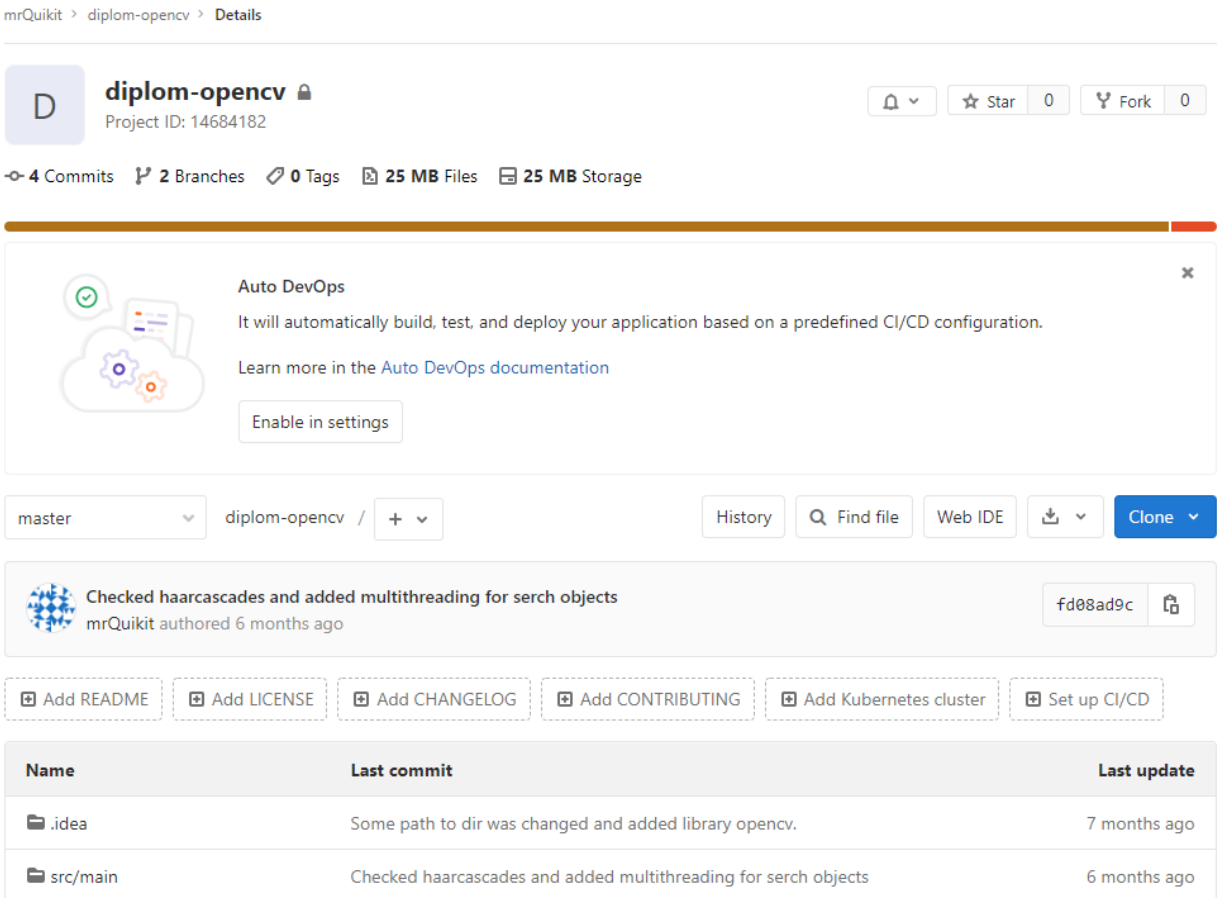


Рисунок 3.2 – Репозиторий для хранения версий разрабатываемого приложения

В данном репозитории находятся две ветки master и dev. Ветки — это разные версии разрабатываемого продукта. В master-ветке находится готовое и стабильное решение, которое было протестировано на наличие

критических ошибок, а dev-ветка содержит продукт в стадии обновления или разработки, там исправляются баги разрабатываемого продукта. После чего dev ветка с помощью функции merge перетекает в master, в этот момент решаются конфликты разных версий и как они будут решены в master будет содержаться актуальная версия продукта.

Приступим к рассмотрению функционала программы по распознаванию множества объектов на изображении с использованием однопоточной реализации и многопоточной. На рисунке 3.3 будет представлена реализация фикции в однопоточном варианте, по локализации объектов на изображении.

```
public void runPictureDetectionWithoutThreads(){
    frame = Imgcodecs.imread(pathToPhoto);
    for (CascadeClassifier cascadeClassifier: cascades) {
        detectOnFrame(frame, cascadeClassifier, new Scalar(0, 255, 0));
    }
    Imgcodecs.imwrite(pathToResultPhoto, frame);
    Thread thread = new Thread(() -> {
        HighGui.imshow( winname: "Photo Detector (Threads)", frame);
        HighGui.waitKey( delay: 10);
        while (true) {
            HighGui.imshow( winname: "Photo Detector (Threads)", frame);
            HighGui.waitKey( delay: 10);
            if (closer) {
                HighGui.waitKey(HighGui.n_closed_windows);
                Thread.currentThread().stop();
                break;
            }
        }
    });
    thread.start();
}
```

Рисунок 3.3 – Фрагмент кода однопоточного варианта функции по распознаванию множества объектов на изображении

На представленном фрагменте кода переменная frame представляет из себя загруженное изображение в приложение. С помощью цикла for производится перебор выбранных каскадов-классификаторов пользователем, и передача их в функцию detectOnFrame в которой и содержится реализация алгоритма Виолы-Джонса. После чего идёт создание отдельного потока на вывод результата распознавания в окно графического интерфейса пользователя, который будет работать до тех пор, пока пользователь не решит его закрыть.

На основе однопоточной реализации была создана функция с применением класса Thread в решении вопроса с распараллеливанием

```
public void runPictureDetection(){
    frame = Imgcodecs.imread(pathToPhoto);
    List<Thread> threads = new ArrayList<>();
    for (int i = 0; i < cascades.size() ; i++) {
        int finalI = i;
        Thread thread = new Thread(() -> {
            int r = (int) (Math.random() * (256));
            int g = (int) (Math.random() * (256));
            int b = (int) (Math.random() * (256));
            System.out.println(Thread.currentThread().getName());
            detectOnFrame(frame, cascades.get(finalI), new Scalar(r, g, b));
        });
        threads.add(thread);
    }
    for (Thread t : threads) {
        t.start();
    }
    boolean flag = true;
    while (flag){
        int counter = 0;
        for (int i = 0; i < threads.size() ; i++) {
            if(threads.get(i).getState() == Thread.State.TERMINATED){
                ++counter;
            }
        }
        if (counter == threads.size()){
            flag = false;
        }
    }
```

множества объектов разного типа на изображении (рисунок 3.4).

Рисунок 3.4 – Фрагмент кода многопоточного варианта функции по распознаванию множества объектов на изображении

В представленном фрагменте кода используется коллекция List под названием «threads» для хранения созданных потоков, которые берут себе по одному каскаду-классификаторов [17]. В самом потоке используется функция random из библиотеки Match для генерирования цвета выделенной области идентифицированного объекта, в формате RGB [20]. Значения генерации каждого из аргументов находятся в диапазоне от 0 до 255 включительно. Далее происходит вызов функции detectOnFrame и передача созданных аргументов цвета в объекте класса Scalar [1]. Когда описание работы потоков закончена начинает перебор функции threads и запуск каждого отдельного потока с помощью функции start. Далее в цикле while производится проверка, что все потоки встают в состояние Terminated. Это означает, что потоки закончили свою работу. Также имеется вывод результата в графический интерфейс пользователя, что и при однопоточном варианте, код является аналогичным.

Рассмотрим подробнее функцию detectOnFrame в которой содержится реализация алгоритма Виолы-Джонса (рисунке 3.5).

```
public void detectOnFrame(Mat frame, CascadeClassifier cascade, Scalar scalar){
    Mat frameGray = new Mat();
    Imgproc.cvtColor(frame, frameGray, Imgproc.COLOR_BGR2GRAY);
    Imgproc.equalizeHist(frameGray, frameGray);
    MatOfRect findedObject = new MatOfRect();
    try {
        cascade.detectMultiScale(frameGray, findedObject, scaleFactor: 1.3, minNeighbors: 1, flags: 0,
                new Size( width: 50, height: 50),new Size( width: 300, height: 300));
    }catch (CvException e){
        e.printStackTrace();
    }
    findedObject.toList().forEach(object->{
        Imgproc.rectangle(frame, new Point(object.x, object.y),
                new Point( x: object.x + object.width, y: object.y + object.height),
                scalar, thickness: 2);
            });
}
```

Рисунок 3.5 – Фрагмент кода функции «detectOnFrame» с использованием алгоритма Виолы-Джонса

На представленном фрагменте кода обратим внимание на аргументы функции. Объект frame класса Mat — это входной кадр изображения, на котором находится искомый объект, объект cascade представляет из себя переданный каскад-классификаторов для локализации объекта, а scalar, как и говорилось выше, содержит в себе готовые аргументы формата RGB. Первым делом формируется объект frameGray, который будет содержать в себе искомое изображение в градиенте серого. Это преобразование получается путём прогона через функцию cvtColor класса Imgproc. Далее идёт выравнивание гистограммы изображения с целью увеличения контрастности через функцию equalizeHist, для более точного распознавания алгоритмом Виолы-Джонса [1]. Когда предобработка изображения завершена, в дело вступает алгоритм Виолы-Джонса под функцией detectMultiScale из библиотеки OpenCV. В качестве аргументов данной функции выступает frameGray – изображение в градиенте серого, результирующий массив яркостей пикселов с выделенной областью расположения объекта findedObject, значение scaleFactor – компромисс между точностью распознавания и скоростью (чем выше, тем быстрее и соответственно хуже распознавание), minNeighbors является значением уровня отбрасывания ложных распознаваний. Далее в функции detectMultiScale идёт размер минимального объекта для распознавания и максимального. После того, как область объекта найдена и перенесена в findedObject производится отрисовка области расположения объекта на искомом кадре изображения – функция rectangle [11].

Модули по распознаванию на видео потоке имеют схожую реализацию, отличие состоит в том, что источником изображения является камера. Далее

```
public ObjectDetection(List<String> pathCascades){
    cascades = getValidationObjectCascade(pathCascades);
    frame = new Mat();
    videoCapture = new VideoCapture( index: 0);
    closer = false;
    pathToPhoto = "src/main/resources/people.jpg";
    pathToResultPhoto = "src/main/resources/result.jpg";
}
```

на рисунке 3.6 будет представлен конструктор класса ObjectDetection, где производится инициализация вспомогательных объектов и переменных.

Рисунок 3.6 – Фрагмент кода конструктора класса ObjectDetection

Конструктор – это инструкция по инициализации объекта класса, которому он принадлежит. Аргументом конструктора является коллекция List в которой хранится пути до каскадов-классификаторов, выбранных пользователем. Также в самом конструкторе производится инициализация объекта frame, в котором будет находится входное изображение. Производится инициализация объекта videoCapure, который позволяет взаимодействовать с камерой, установленной в компьютере [19]. И инициализируются переменная closer дающая сигнал на убийство потоков при подаче сигнала пользователем, а также объекты, которые содержат путь до стандартного изображения, установленного в приложении и место сохранения результата распознавания.

В результате реализации приложения получается на выходе два рабочих модуля по распознаванию объектов на изображении, а именно распознавание на входном кадре изображения и распознавание на видео потоке. Эти модули имеют как однопоточную реализацию, так и подход с применением технологии распараллеливания, встроенным в язык Java. Результаты работы программы рассмотрены в следующем подразделе.

3.2 Тестирование приложения

Тестирование будет проводиться с использованием загруженного изображения в приложение, а также с помощью распознавания на видео потоке в реальном времени с использованием камеры, разрешение которой 0,3 МП (640×480).

На рисунке 3.7 будет представлен результат распознавания множества объектов разного типа с использованием модуля по распознаванию на загружаемом изображении в приложение в однопоточной реализации.

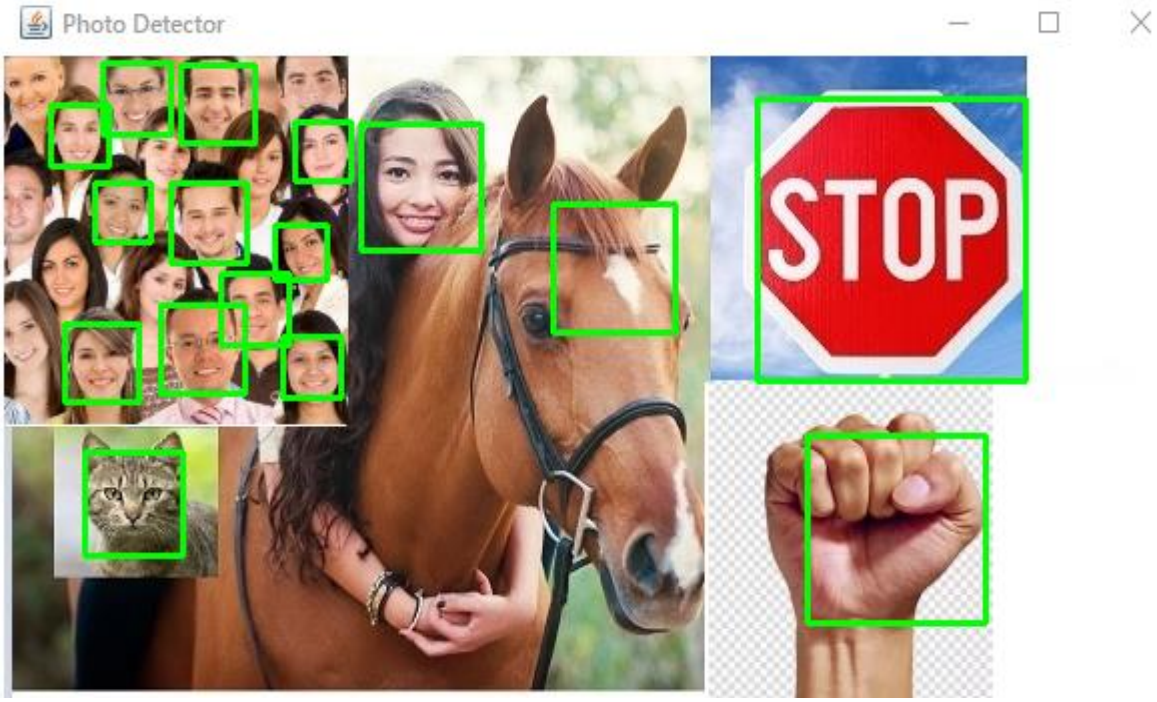


Рисунок 3.7 – Результат работы модуля по распознаванию на изображении в однопоточной реализации

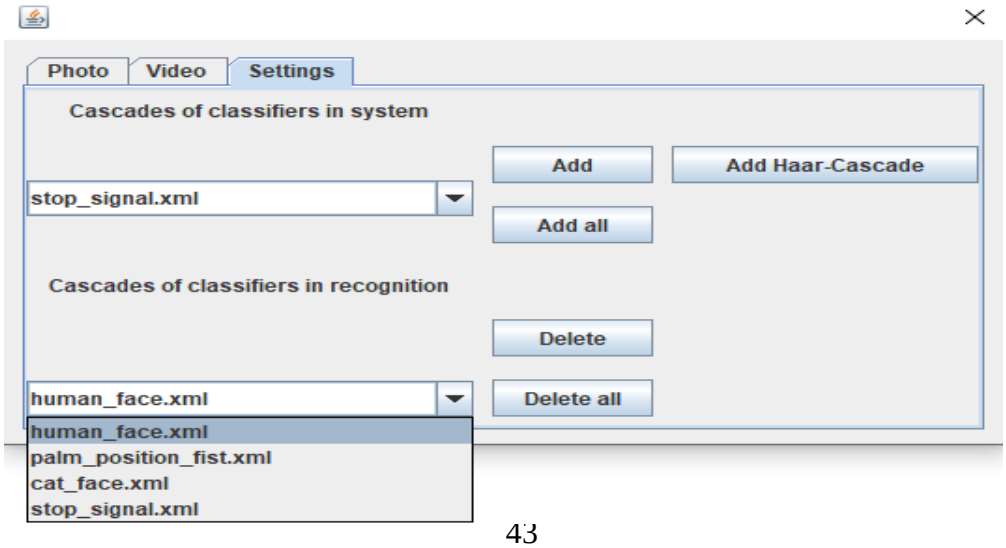
На данном рисунке видно, что алгоритм Виолы-Джонса смог распознать сразу несколько объектов разного типа: лица людей, голову кошки, стоп-сигнал и кулак человека. Но также произошло появления артефакта, а именно выделение области, где находится лошадь. Каскад-классификатора для распознавания лошади не был задействован для тестирования на данном изображении. Артефакты такого рода решаются с помощью увеличения значения в аргументах функции, а именно коэффициент отбрасывания ложный распознаваний.

Для того, чтобы увидеть разницу между однопоточной реализацией тестируемого модуля и многопоточной возьмём за основу текущее изображение и задействуем его в многопоточной реализации текущего модуля (рисунок 3.8).

Рисунок 3.8 – Результат работы модуля по распознаванию на изображении в многопоточной реализации

Основное отличие двух реализаций модуля по распознаванию объектов на загруженном изображении в приложение является то, что каждый распознаваемый объект выделяется своим собственным цветом.

Для данного тестируемого модуля было выбрано всего четыре загруженных каскада-классификаторов из тридцати девяти возможных в



графическом интерфейсе пользователя (рисунок 3.9).

Рисунок 3.9 – Графический интерфейс пользователя, раздел по взаимодействию с каскадами-классификаторов

На данном рисунке продемонстрирован графический интерфейс пользователя раздела настроек и показано, что было задействовано при распознавании всего четыре каскада из выпадающего списка.

Теперь приступим к тестированию модуля, отвечающего за распознавание на видео потоке в реальном времени посредством видео камеры (рисунок 3.10).

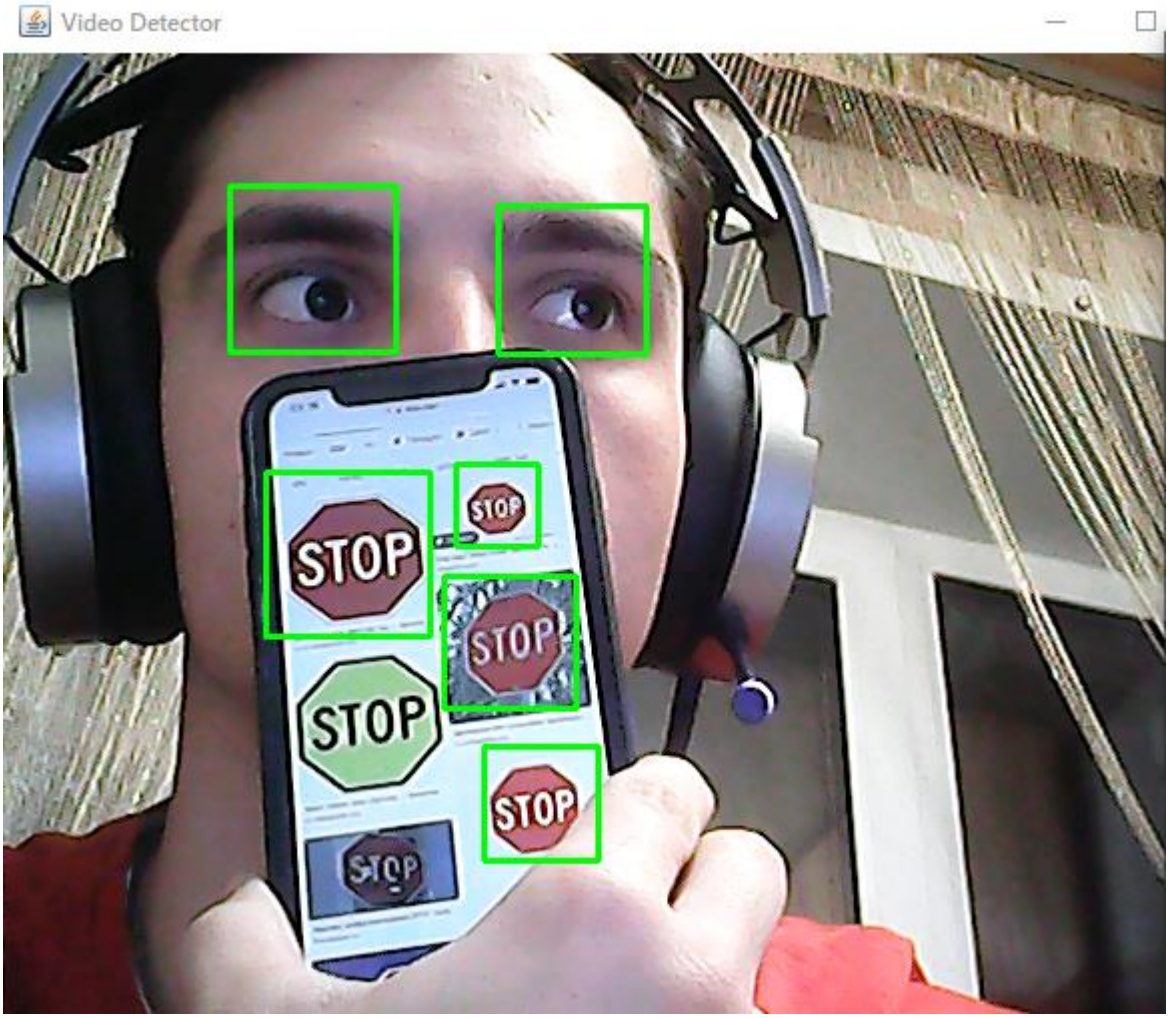
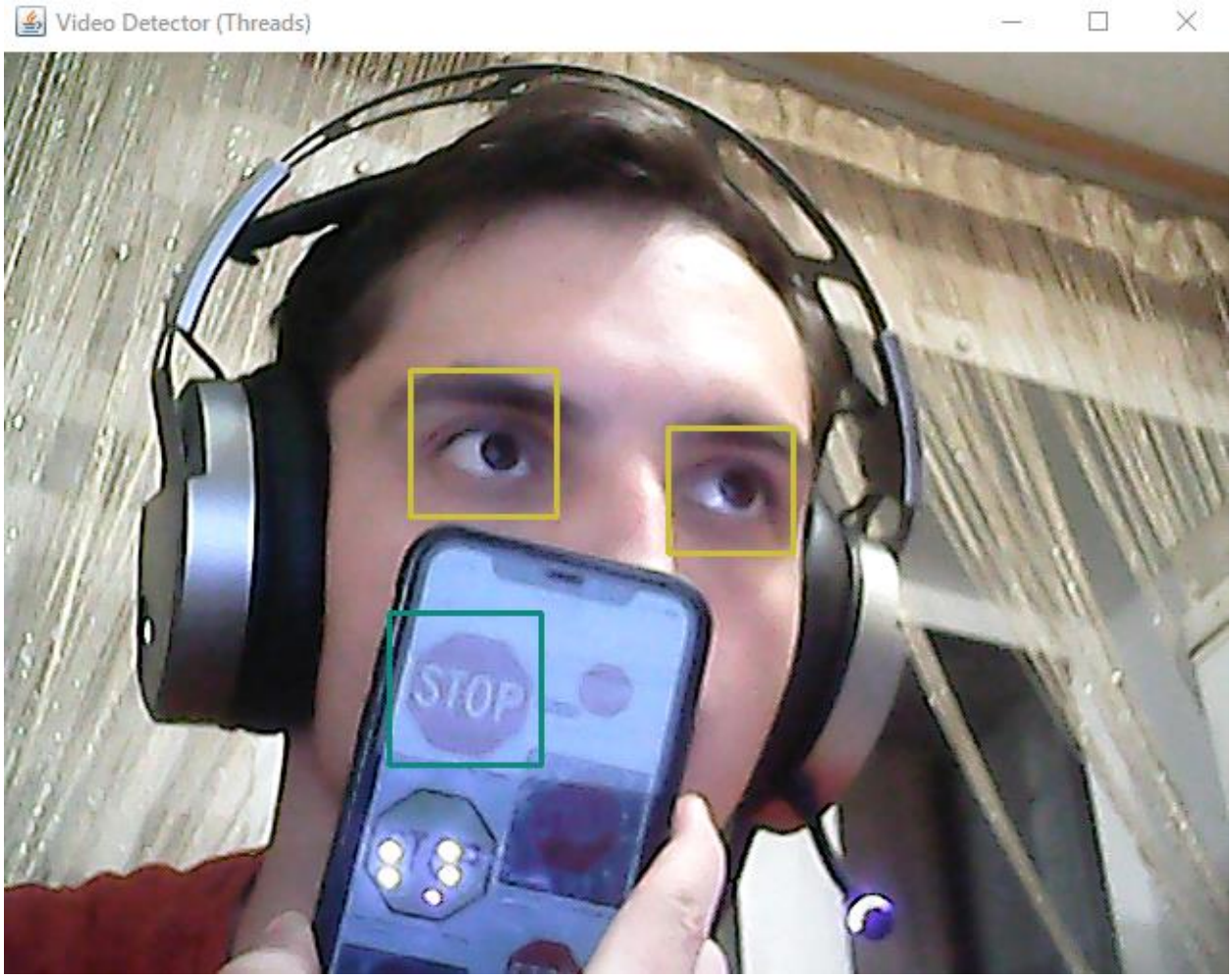


Рисунок 3.10 – Результат работы модуля по распознаванию на видео потоке в реальном времени, в однопоточной реализации

На данном рисунке видно, что приложению удалось распознать два глаза человека и стоп-сигналы, представленные на телефоне посредством

изображения. Повторим проделанный тест с темы же двумя каскадами-



классификаторов на многопоточной реализации представленного модуля (рисунок 3.11).

Рисунок 3.11 – Результат работы модуля по распознаванию на видео потоке в реальном времени, в многопоточной реализации

Основное отличие реализаций данного модуля заключается в том же моменте, что и при распознавании объектов на изображении, загруженном в приложение, то есть при распознавании однопоточной реализацией разные объекты выделяются одним цветом, а при многопоточной – разными цветами.

Теперь ответим на главный вопрос: «Как ведёт себя алгоритм Виолы-Джонса при нагрузке и сколько кадров в секунду он обрабатывает?».

Результаты тестирования алгоритма Виолы-Джонса при нагрузке несколькими каскадами-классификаторов будут представлены в таблице 3.1.

Таблица 3.1 – Результаты работы алгоритма Виолы-Джонса при нагрузке

Количество каскадов-классификаторов	Однопоточная реализация	Многопоточная реализация
	Кадры в секунду (640×480)	
1	26	25
3	15	16
6	9	15
9	5	14
12	4	10

На данной таблице видно, как ведёт себя алгоритм Виолы-Джонса при нагрузке каскадами-классификаторов разного рода для распознавания множества объектов на изображении в реальном времени. При однопоточной реализации метода по распознаванию объектов на видео потоке работать становиться проблематично при подключении шести каскадов-классификаторов, так как количество обрабатываемых изображений сходится в среднем к девяти. Но это можно исправить путём применения многопоточности. При таком результате можно применить для более-менее комфортной работы двенадцать каскадов-классификаторов.

Таким образом, пределы нагрузки стандартного алгоритма Виолы-Джонса были выявлены, можно считать, что поставленная цель работы достигнута.

ЗАКЛЮЧЕНИЕ

Выполненная работа посвящена исследованию эффективности работы алгоритма Виолы-Джонса при нагрузке. Цель работы определила её основное направление – разработка приложения по распознаванию множества объектов на изображении.

В результате исследования были выполнены следующие задачи:

- изучить особенности работы алгоритма Виолы-Джонса;
- разработать программное обеспечение для демонстрации работы алгоритма при локализации разного рода объектов с применением технологий распараллеливания;
- протестировать метод Виолы-Джонса на однопоточном варианте программного модуля и многопоточном.

К тому же были выявлены недостатки алгоритма Виолы-Джонса и приведены некоторые модификации, которые устраняют эти проблемы с описанием возможных последствий.

Разработанное приложение на языке Java (в среде IntelliJ IDEA) с использованием свободной библиотеки OpenCV позволяет распознавать объекты на входном кадре изображения путём использования спроектированных и внедрённых модулей:

- распознавание множества объектов на изображении, загруженном в приложение;
- распознавание множества объектов на входном потоке, в реальном времени с использованием веб-камеры.

Приложение увеличивает требования к вычислительным мощностям компьютера с увеличением количества каскадов-классификаторов. Так как увеличивается выполнение математических операции, в основном идёт нагрузка на процессор компьютера.

Алгоритм Виолы-Джонса был протестирован на разном количестве каскадов-классификаторов, были продемонстрированы результаты

тестирования и выявлены максимально возможное количество каскадов-классификаторов для более-менее комфортной работы программы при распознавании в реальном времени.

Данная работа подтвердила, что алгоритм Виолы-Джонса подходит для распознавания объектов разного типа. Но алгоритм уже не является перспективным направлением в исследовании, в настоящий момент уже имеются более производительные алгоритмы по локализации объектов.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Визильтер Ю. В. Обработка и анализ изображений в задачах машинного зрения: Курс лекций и практических занятий. М.: Физматкнига, 2010. 672 с.

2. Гарсия Г. Б., Исмаэль С.Г., Ноэлия В. Э. Обработка изображений с помощью OpenCV; М.: ДМК Пресс, 2016. 210 с.

3. Гонсалес Р. С., Вудс Р.Е. Цифровая обработка изображений; М.: Техносфера, 2012. – 1104 с.

4. ГОСТ Р 2.105-2019. Национальный стандарт Российской Федерации. Единая система конструкторской документации. Общие требования к текстовым документам. Введ. 2020-02-01.- М.: Изд-во стандартов, 2020. 36 с.

5. ГОСТ 2.316-2008. Межгосударственный стандарт. Единая система конструкторской документации. Правила нанесения надписей, технических требований и таблиц на графических документах. Общие положения. Взамен ГОСТ 2.316-68; Введ. 2009-07-01.- М.: Изд-во стандартов, 2009. 12 с.

6. ГОСТ 7.9-95. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Реферат и аннотация. Общие требования (ИСО 214-76). Взамен ГОСТ 7.9-77; Введ. 30.06.1997.- М.: Изд-во стандартов, 2001. 28 с.

7. ГОСТ Р 54521-2011. Национальный стандарт Российской Федерации. Статистические методы. Математические символы и знаки для применения в стандартах (ИСО 80000-2:2009). Введ. 2012-12-01.- М.: Изд-во стандартов, 2011. 32 с.

8. Клейнберг Д., Тардос Е. Алгоритмы: разработка и применение. Классика Computers Science / Пер. с англ. Е. Матвеева. СПб.: Питер, 2016. 800 с.

9. Клетте Р. Компьютерное зрение. Теория и алгоритмы; М.: ДМК Пресс, 2019. 508 с.

10. Машинное зрение: понятия, задачи и области применения // Казахский национальный технический университет К.И.Сатпаева, Алматы, Казахстан [Электронный ресурс]: Электронные данные. URL: http://rusnauka.com/25_SSN_2009/Informatica/51050.doc.htm (дата обращения: 14.01.2020).

11. Носова, Л.С. Case-технологии и язык UML: учебно-методическое пособие/ Носова Л.С. Челябинск, Саратов: Южно-Уральский институт управления и экономики; Ай Пи Эр Медиа, 2019. 67 с.

12. Прохоренок Н. А. OpenCV и Java. Обработка изображений и компьютерное зрение; СПб.: БХВ-Петербург, 2018. 320 с.

13. Системы компьютерного зрения: современные задачи и методы [Электронный ресурс]: Электронные данные. URL: http://controleng.ru/innovatsii/sistemy-komp-yuternogo-zreniya-sovremenny-e-zadachi-metody/ (дата обращения: 18.01.2020).

14. Что такое машинное зрение и чем оно отличается от человеческого? [Электронный ресурс]: Электронные данные. URL: https://meduza.io/feature/2019/03/30/chto-takoe-mashinnoe-zrenie-i-chem-ono-otlichaetsya-ot-chelovecheskogo-seychas-ob-yasnim-ponyatno (дата обращения: 23.01.2020).

15. Andrew F.: Oracle Certified Professional Java SE Programmer study guide: contains around 300 practice questions with solutions; 2020, 650 pages.

16. Beyerer J., Fernando P., Christian F.: Automated Visual Inspection: Theory, Practice and Applications; Springer Berlin Heidelberg, 2016. 798 pages.

17. Cay S. Horstmann: Core Java Volume I Fundamentals, 11th Edition; Prentice Hall, 2018. 928 pages.

18. Efficient face detection algorithm: using Viola Jones method [Электронный ресурс]: Электронные данные. URL: https://www.codeproject.com/Articles/85113/Efficient-Face-Detection-Algorithm-using-Viola-Jon (дата обращения: 28.01.2020).

19. OpenCV: Cascade Classifier [Электрон. ресурс]: Документация URL: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html (дата обращения: 10.02.2020).

20. Yang H.: Java Swing Tutorials - Herong's Tutorial Examples; HerongYang.com, 2018. 201 pages.

ПРИЛОЖЕНИЕ А КОД ПРОГРАММЫ

Исходный код программы доступен для скачивания в электронном виде.

Режим доступа:

https://gitlab.com/Quikit/diplom-opencv.git