Санкт-Петербургский государственный университет

**Кафедра высшей математики**

**Патин Михаил Владиславович**

**Магистерская диссертация**

**Автоматизация подготовки обучающего набора для классификации похожих изображений**

Направление 01.04.02

Прикладная математика и информатика

Магистерская программа

Прикладная математика и информатика в задачах цифрового управления

Научный руководитель:  
кандидат тех. наук,

доцент   
Гришкин В.М.

Санкт-Петербург

2018

Оглавление

[Введение 3](#_Toc515411249)

[Цель 5](#_Toc515411250)

[Обзор литературы 6](#_Toc515411251)

[Глава 1. Обработка информации на изображениях 8](#_Toc515411252)

[1.1 Особые точки и дескрипторы 8](#_Toc515411253)

[1.1.1 ORB – Oriented FAST and Rotated BRIEF 9](#_Toc515411254)

[1.1.2 BRISK – Binary Robust Invariant Scalable Keypoints 11](#_Toc515411255)

[1.1.3 AKAZE – Accelerated-KAZE 13](#_Toc515411256)

[1.2 Словарь визуальный слов 15](#_Toc515411257)

[1.3 Классификация 18](#_Toc515411258)

[Глава 2. Разработка приложения 20](#_Toc515411259)

[2.1 Программная реализация 20](#_Toc515411260)

[2.2 Проектирование базы данных 26](#_Toc515411261)

[Глава 3. Результаты работы приложения 29](#_Toc515411262)

[3.1 Изображения общего содержания 30](#_Toc515411263)

[3.2 Изображения определенных объектов 33](#_Toc515411264)

[Выводы 35](#_Toc515411265)

[Заключение 37](#_Toc515411266)

[Список литературы 38](#_Toc515411267)

[Приложение 40](#_Toc515411268)

# Введение

В век стремительного развития цифровых технологий, когда объёмы информации растут громадными темпами, многие IT компании стремятся автоматизировать процессы для ее обработки, создавая сложные, высоконагруженные программные и аппаратные комплексы.

В последние время для задач анализа данных быстрыми шагами развивается направление нейронных сетей, позволяющие получить сравнимые с человеком результаты в различных сферах. Но для достижения этих показателей требуется приложить не малые усилия, так как не существует алгоритма определяющего оптимальную архитектуру нейронной сети для конкретной задачи. Проектирование сети требует работы высококвалифицированного в данной области персонала, а также огромных как аппаратных, так и материальных средств, что может позволить себе не каждая современная компания.

Но недостаточно просто смоделировать архитектуру нейронной сети, ведь перед ее использованием так же необходима стадия обучения. Данная стадия предполагает наличие заранее известных начальных значений и предполагаемых конечных результатов. Формирование обучающих выборок так же требует не малых человеческих затрат. Данную проблему можно увидеть на актуальной на сегодняшний день задаче обработке изображений. Ведь в текущее время, почти у каждого современного человека имеется мобильное устройство, с помощью которого люди создают тысячи и миллионы фотографий каждый день. И для создания необходимой обучающей выборки, по которой возможно обучить спроектированную нейронную сеть или иной инструмент, позволяющий анализировать изображения, приходиться использовать кропотливый и малооплачиваемый ручной труд простых людей.

В данной работе предлагается разработать приложение, автоматизирующее данный процесс и обладающее необходимым функционалом, позволяющим без использования огромного массива данных и при умеренных вычислительных затратах, осуществить первичную классификацию изображений с высокой точностью. Тем самым позволив существенно сократить общее время на подготовку обучающей выборки.

Существуют методы, которые позволяют, хоть и при использовании в ограниченном круге задач компьютерного зрения, получить информацию об общем характере изображений. Данные методы основаны на поиске особенностей изображений – особых точек и дескрипторов. Их идея разработана достаточно давно, но даже сейчас они применяются во многих сферах, связанных с обработкой изображений. Для их использования в задаче классификации необходимо систематизировать и выделить общие черты дескрипторов, присущих определенным типам изображений.

В рамках рассматриваемой области исследований под общими чертами можно представить некие слова, совокупность которых и будет описывать характер изображений одного типа. Некоторый набор таких слов по всем необходимым типам образует словарь визуальных слов. Предполагается, что использование алгоритмов классификации позволит выделить слова, присущих определенным типам изображений для дальнейшей классификации неизвестных изображений по определенным типам.

Данная работа является продолжением исследований в области алгоритмов поиска особых точек и дескрипторов, описанных в бакалаврской работе автора: “Сравнительный анализ дескрипторов особых точек изображений с внедрением алгоритмов под операционной системой «Android»”. В данной работе предлагается сравнить результаты их применения в задаче классификации изображений, используя словарь визуальных слов.

# Цель

Целью данной работы является разработка программного приложения, основанном на комплексе существующих методов компьютерного зрения, которое позволит упростить процесс формирования обучающей выборки, из необходимых типов изображений, для дальнейшего его использования в различных задачах. В ходе работы планируется провести дополнительные исследования результатов работы программы при использовании таких алгоритмов как ORB, BRISK, AKAZE осуществляющих поиск особенностей изображения в виде особых точек и их дескрипторов.

Решение данной задачи требует реализации следующих этапов:

* Анализ методов поиска особых точек и дескрипторов.
* Рассмотрение подхода для работы с дескрипторами особых точек.
* Рассмотрение метода классификации.
* Создать программную реализацию для проведения исследований.
* Оценку результатов на различных выборках изображений.

Разрабатываемое программное приложение должно обеспечивать следующий функционал:

* Интуитивный интерфейс
* Сопровождение пользователя на всех этапах работы
* Сохранение и загрузка необходимого набора данных
* Настройка методов
* Контроль качества результатов
* Сохранение результатов

# Обзор литературы

В [1] рассматривается алгоритм ORB (Oriented FAST and Rotated BRIEF), представляющий из себя комбинацию из модифицированных алгоритмов нахождения особых точек с помощью FAST [4] и последующим определением их особенностей в виде бинарной строки по модифицированному методу BRIEF [5]. Данный подход дает, по результатам их тестирования, значительный выигрыш в скорости при сопоставимой или лучшей точности, чем SIFT (Scale Invariant Feature Transform) и SURF (Speeded Up Robust Features) соответственно.

В работе [2] представлен алгоритм BRISK (Binary Robust Invariant Scalable Keypoints), в котором особые точки обнаруживаются с использованием FAST [4] при линейном изменении масштаба начального изображения. Дескрипторы описываются в бинарном виде по улучшенному алгоритму BRIEF [5]. В сравнительных тестах получена сравнимая с SIFT и SURF точность на различном типе изображений, но при этом скорость работы алгоритма, по их результатам, в разы выше.

В [3] представлен алгоритм A-KAZE, в котором поиск особых точек осуществляется при нелинейном масштабировании изображения с использованием схемы FED (Fast Explicit Diffusion) [6]. В качестве бинарного дескриптора применяется M-LDB (Modified-Local Difference Binary). Результаты работы алгоритма, по данным приведенных исследований, превосходят по всем параметрам ORB, BRISK, SIFT и SURF.

В работах [8] и [9] описывается принцип работы мешка слов на основе особенностей изображения. Так же в [9] рассмотрен общий подход к классификации изображений при использовании визуальных слов, похожий алгоритм используется в данной работе, но с некоторыми изменениями.

Улучшенный алгоритм k-средних описывается в [10], основным улучшением является инициализация начальных центроид, что позволяет сократить время на работу алгоритма и, в некоторых случаях, существенно повысить точность определения центров классов.

Разбор метода опорных векторов, механизм которого был разработан достаточно давно, представлен в [11]. Имеется подробное теоретическое описание возможных вариантов использования данного классификатора, рекомендации к выбору необходимых параметров и компонент алгоритма.

Использование словаря визуальных слов предполагает работу с гистограммой распределения слов из словаря на изображении, то для лучшей работы применяемого классификатора используется соответствующая функция ядра, рассмотренная в [12]. Так же там приведен сравнительный анализ с другими известными ядрами, где данное ядро показало лучшие результаты.

# Глава 1. Обработка информации на изображениях

Общая структура рассматриваемого далее подхода описана в статье [9] и представляет собой следующие этапы:

* + 1. Поиск особых точек и дескрипторов на коллекции изображений
    2. Формирование общего словаря визуальных слов
    3. Поиск ближайших слов из обучающей выборки и словаря
    4. Составление гистограммы распределения слов
    5. Формирование массива характеристик типов изображений
    6. Обучение классификатора на полученных данных

## Особые точки и дескрипторы

Введем некоторые определения.

Особая точка (от лат. Key Point – ключевая точка) – необходима для обнаружения локальным особенностям изображений, удовлетворяющая ряду свойств.

* + 1. Определенность (distinctness) – данная точка должна выделяться среди соседних.
    2. Устойчивость (repeatability) – изменение цветовых характеристик изображения не должно менять её положения.
    3. Инвариантность (invariance) – особые точки также должны быть устойчивыми к аффинным преобразованиям изображения и объектов на нем.
    4. Стабильность (stability) – различные визуальные дефекты, не должна ухудшать работу детектора.
    5. Интерпретируемость (interpretability) – формат найденных особых точек должен быть пригодным для дальнейшей работы с ними.

Для их поиска используют различные алгоритмы так называемых детекторов.

Дескриптор (от лат. descriptor — описывающий) – описание особой точки, определяющее характер её окрестности, обычно представляет из себя числовой или бинарный вектор. Длина вектора и его значения определяются применяемым алгоритмом. Дескриптор позволяет определить особую точку из их множества.

Совокупность полученных данных можно охарактеризовать как описание изображения.

В качестве алгоритмов поиска особых точек и дескрипторов предполагается использовать алгоритмы ORB, BRISK, A-KAZE. Их выбор основан на том, что они являются одними из современных и популярных алгоритмов данного предназначения. Так же они имеют бинарный вектор значений дескриптора, что позволяет увеличить скорость при работе с ними.

## ORB – Oriented FAST and Rotated BRIEF

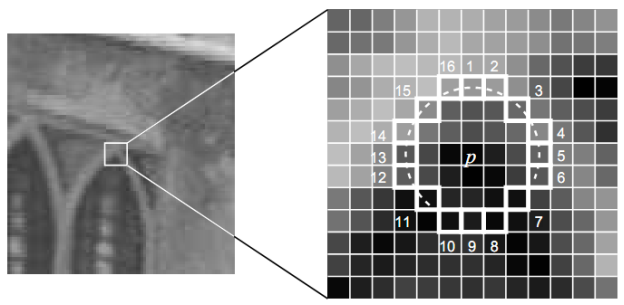
ORB представлен в 2011г [1]. В его основе лежит комбинация таких алгоритмов как детектор FAST (**F**eatures from **A**ccelerated **S**egment **T**est) [4] и дескриптор BRIEF (**B**inary **R**obust **I**ndependent **E**lementary **F**eatures) [5] с некоторыми улучшениями.

*Детектор FAST.*

Для поиска угловых точек поочерёдно рассматриваются окрестности по 16 пикселей вокруг каждого пикселя *p*.

Точка *p* считается подозрительной на особую, если существует *N* пикселей (в данной работе *N=9*) в её окружности длиной 16 пикселей, если все *N* ярче или темнее , где  – яркость точки *p*, *t* –пороговая величина. При выполнении этого условия далее исследуется значения яркости на окружности под номерами 1, 5, 9, 13 (рис. 1.1). Если для трех пикселей из четырех выполняется условие  или *,* , тогда *p* считается особой точкой.

Выбор только 4 пикселей на окружности позволяет быстро отсеять не подходящие точки, но в некоторых случаях возможно определение разных особенностей в одной окружности. В алгоритме ORB максимальное количество особых точек по умолчанию не более 500, если их больше, то к ним применяется детектор углов Харриса, для исключения наименее значимых.

  
Рис. 1.1.Рассматриваемая окрестность точки *p* FAST детектора [1].

Для инвариантности к масштабированию применяется вышеописанный алгоритм на пирамиде Гаусса. Октавами которой является изначальное изображение сжатое с линейным шагом.

Введение параметра угловой ориентации позволяет добиться устойчивости детектирования при вращении объекта. Он основан на направлениях градиента яркости относительно центра точки, направление с наибольшей интенсивностью назначается ориентацией особой точки .

*Дескриптор направленный BRIEF.*

Данный дескриптор представляется в виде вектора длиной 256, состоящего из результатов бинарных тестов вокруг особой точки. В окрестности пиксель сравниваются средние значения яркостей между и , где – области пикселей:

*средняя яркость выбранной области.*

Для достижения инвариантности к вращению область вычисления дескриптора ориентируется по ориентации особой точки .

Все наборов и формируют матрицу размерностью . Далее с помощью матрицы поворота ориентируется в соответствии с углом :

.

А сам вектор дескриптора записывается как:

*,*

*где .*

## BRISK – Binary Robust Invariant Scalable Keypoints

Данный метод представлен в 2011г. Детектирование особых точек осуществляется с помощью FAST (**F**eatures from **A**ccelerated **S**egment **T**est), а дескриптор BRIEF, но в их работу были внесены некоторые изменения.

*Поиск особых точек.*

Для достижения инвариантности к масштабу, предлагается выбирать наилучшую особую точку с максимальным значением интенсивности в пирамиде, которая состоит из октав и внутренних октав . Октавы формируются как сжатие оригинального изображения в раза. Внутренние октавы расположены между и и представлены в виде сжатой в раза. Поиск особых точек в октавах осуществляется детектором FAST.

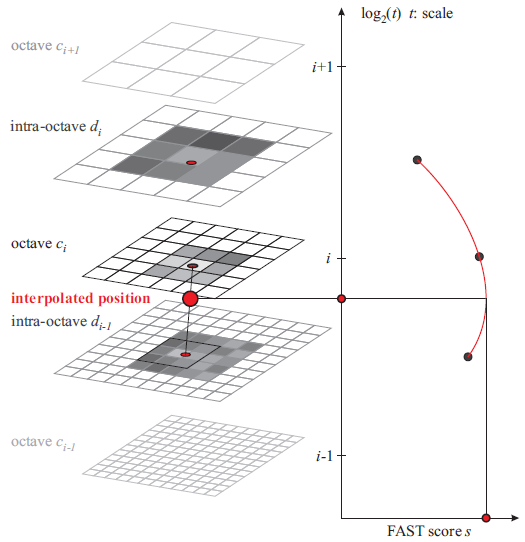
****

Рис. 1.2.Пример поиска особой точки с максимальным значением S [2].

*Дескриптор BRISK****.***

Область вокруг особой точки разбивается на 60 участков (рис. 1.3):

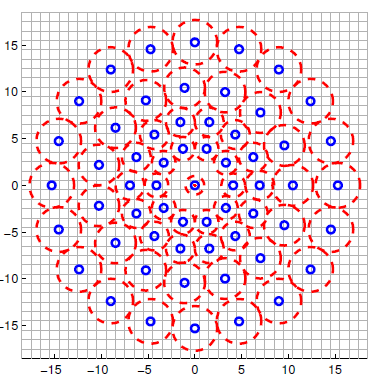


Рис 1.3. Область вычисления дескриптора [2].

Множество разбивается на 2а подмножества:

*где размер особой точки.*

Вычисляется среднее значение градиента множества :

Дескриптор состоит из бинарной строки длиной 512, заполненной результатами проведенных тестов в множестве :

*,*

*где интенсивность окрестности радиуса точки,*

*угол направления градиента .*

## AKAZE – ****Accelerated-KAZE****

При разработке данного метода, представленного в 2013 году, старались добиться высокой скорости работы как детектора, так и дескриптора. При этом найденные особые точки и их дескрипторы должны были удовлетворять высоким показателям точности при сравнении изображений.

Применение алгоритма FED **-** **Fast Explicit Diffusion [6] на пирамидальной схеме позволяет построить нелинейную многомасштабную пирамиду. Применение нелинейного коэффициента масштабирования позволяет увеличить скорость нахождения нужной особой точки по сравнению с Гауссовой пирамидой:**

Вычисление данного коэффициента основано на изменении яркости изображения при масштабировании.

*Детектор.*

Для каждого октавы в **пирамиде** вычисляется определитель Гессиана.

*где нормализированный относительно масштаба коэффициент, для вычисления с учетом размера октавы*

Производные второго порядка вычисляются с помощью фильтра Шарра[11] с шагом . Данный фильтр позволяет учитывать ориентацию особых точек. С помощью такого подхода ищем такие точки в октаве, значение фильтра которых выше заданного порога и является наибольшим из окрестности точки пикселей.

Далее, для каждой точки из потенциальных максимумов сравнивается её значение относительно результатов в соседних октавах в окне размером соответственно. В итоге расположение особой точки оценивается с субпиксельной точностью соответствуя квадратичной функции к определителю Гессиана в соседних пикселей для поиска максимума.

*Дескриптор* ***M-LDB.***

Первоначальный дескриптор LDB [9] основывался на тех же принципах что и рассмотренный выше BRIEF, но к сравнениям яркостных показателей областей добавили сравнение значений градиентов яркости по оси и , в итоге результат одного теста состоит из трех битов вместо одного. Проведение тестов проводилось в окне размером пикселей, деленном на 4, 9 и 16 областей.

Но LDB имеет недостатки такие как не инвариантность к вращению и масштабированию. И в качестве решения этих проблем в AKAZE используется его улучшенная версия – M-LDB:

Окно дескриптора ориентируется по ориентации особой точки.

Инвариантность к масштабу получена с помощью выбора размера окна дескриптора в зависимости от размера октавы в которой найдена его особая точка.

В отличии от LDB в M-LDB тесты проводятся не между средним значением всех пикселей в области, а между заданным их количеством в зависимости от размера . Что позволяет ускорить вычисление дескриптора.

Итоговый бинарный дескриптор имеет длину 486 по три составляющих.

## Словарь визуальный слов

1. Формирование общего словаря визуальных слов
2. Поиск совпадений слов из обучающих групп и словаря
3. Составление гистограммы распределения слов
4. Формирование массива описаний обучающих групп изображений

Для выделения общего описания групп изображения рассматривается подход, основанный на “визуальном” словаре слов [8].

Визуальный - означает, что слова представляют из себя не текстовую информацию в привычном нам понимании определения “слова” в тексте, а некоторое цифровое описание изображения, в данном случае для этих целей используются рассмотренные ранее дескрипторы особых точек.

Рассматриваемый далее алгоритм работы представлен на рис 1.4.

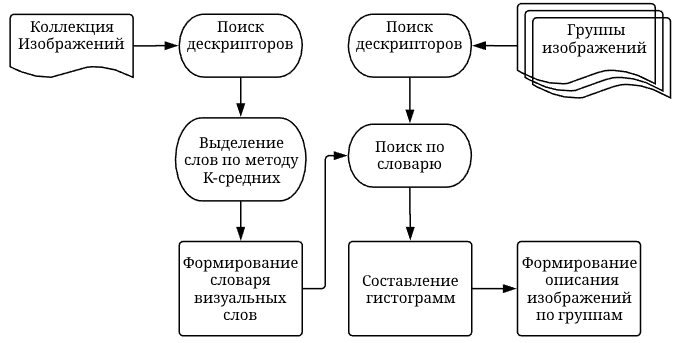


Рис 1.4. Схема работы словаря визуальных слов.

Составляется массив дескрипторов коллекции изображений, включающая в себя различные изображения из предполагаемых групп для классификации. Далее вычисляются k центров кластеров данного массива в качестве визуальных слов, по методу k-means++ [10], который отличается от обычного k-means выбором начальных центров:

Инициализируются центры кластеров (центроиды)

1. Выбирается первый центроид случайным образом среди всех векторов дескрипторов из
2. Для каждого вектора вычисляется значение квадрата расстояния до ближайшего центроида (если их несколько):

,

где – номер дескриптора , , – шаг итерации.

1. Выбирается следующий центроид так, чтобы вероятность выбора вектора была пропорциональна вычисленному для .
2. Повторяются шаги 2 и 3 до тех пор, пока не будут найдены все необходимые центроиды .

Далее суть метода сводится к минимизации до центра масс

где – наборы векторов для каждого центроида

Если выполняется условие:

То переходим на следующую итерацию , обновляем центр масс

и вычисляем новые распределения дескрипторов по центроидам.

Процесс повторяется пока или не выполнится

Для определения описания изображений так же вычисляются их дескрипторы и далее для каждого осуществляется поиск ближайшего слова из словаря.

Далее, полученные совпадения позволяют составить гистограмму описания изображения, показывающая распределение значений визуальных слов на изображении. Данные нормализуются по количеству полученных дескрипторов на изображении. Т.е. получаем гистограмму совпадений, где i значение представляет собой нормализованное количество слова i на изображении (рис 1.5).

Совокупность гистограмм одной группы позволяет определить общее описание этих изображений, для этих целей применяется алгоритм классификации.

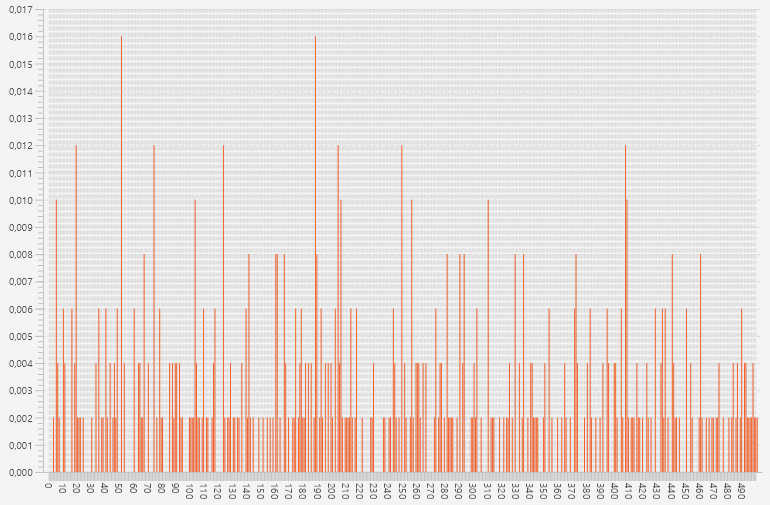


Рис 1.5 Гистограмма изображения по словарю

## Классификация

Для формирования индивидуального описания наборов изображений используется один из широко распространенных алгоритмов классификации с учителем метод опорных векторов SVM [11].

Классификацию с учителем в данном случае можно представить в виде

где множество наблюдений, гистограмм размерности , множество номеров классов, целевая зависимость, ответы которой заранее известны только на обучающей коллекции , В итоге задача сводится к постройке алгоритма , аппроксимирующего целевую зависимость на всем множестве .

Для данной задачи в методе опорных векторов требуется найти гиперплоскости максимальной разности, которые разделят множество гистограмм изображений на заранее известных классов.

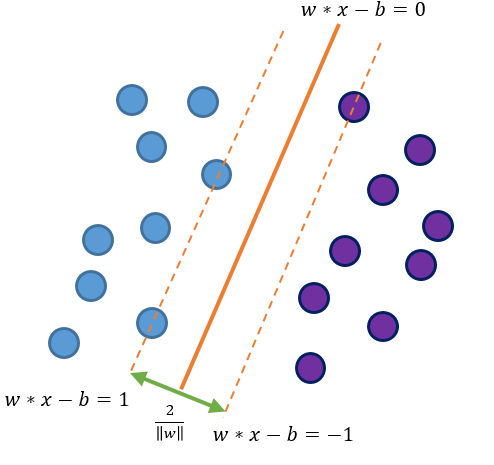


Рис 1.6 Гиперплоскость

Где гиперплоскость можно представить в виде множества точек , удовлетворяющих условию (рис 1.6), - скалярное произведение нормали к гиперплоскости на вектор , параметр, определяющий смещение гиперплоскости относительно начала координат вдоль нормали . В итоге алгоритм сводится к квадратичной задаче оптимизации

При ограничениях .

Дальнейшее решение приведено в статье [11].

В связи с большой размерностью вектора гистограммы применяется kernel trick, при котором осуществляется замена скалярных произведений, в алгоритме линейной классификации, на функцию ядра, что позволяет построить нелинейную гиперплоскость. В полученном пространстве предполагается существование оптимальной разделяющей гиперплоскости (рис 1.7).

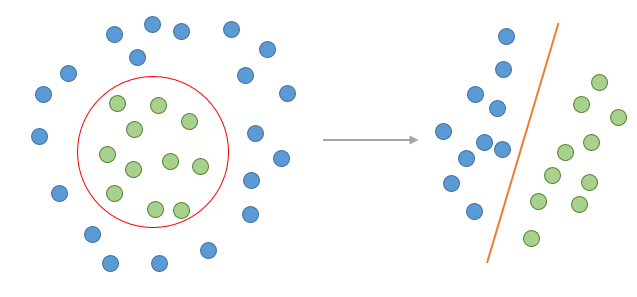


Рис 1.7 Применение функции ядра

В качестве ядра была выбрана функция, учитывающая природу гистограмм [12] – histogram intersection kernel:

Для корректной нормировки функции требуется, чтобы выполнялось , что и было проведено при нормализации гистограммы.

# Глава 2. Разработка приложения

## Программная реализация

Для реализации приложения выбран объектно-ориентированный язык программирования Java версии 8, его выбор основан на личных навыках. Пользовательский интерфейс разработан с помощью инструмента JavaFX.

Для работы с алгоритмами компьютерного зрения, обработки и анализа изображений и использования численных алгоритмов общего назначения используется библиотека OpenCV версии 3.4.1, являющейся самой распространённой библиотекой для данного рода задач, так же она свободно распространяемая и имеет открытый исходный код.

Общая схема разработанного приложения представлена на рис 2.1.

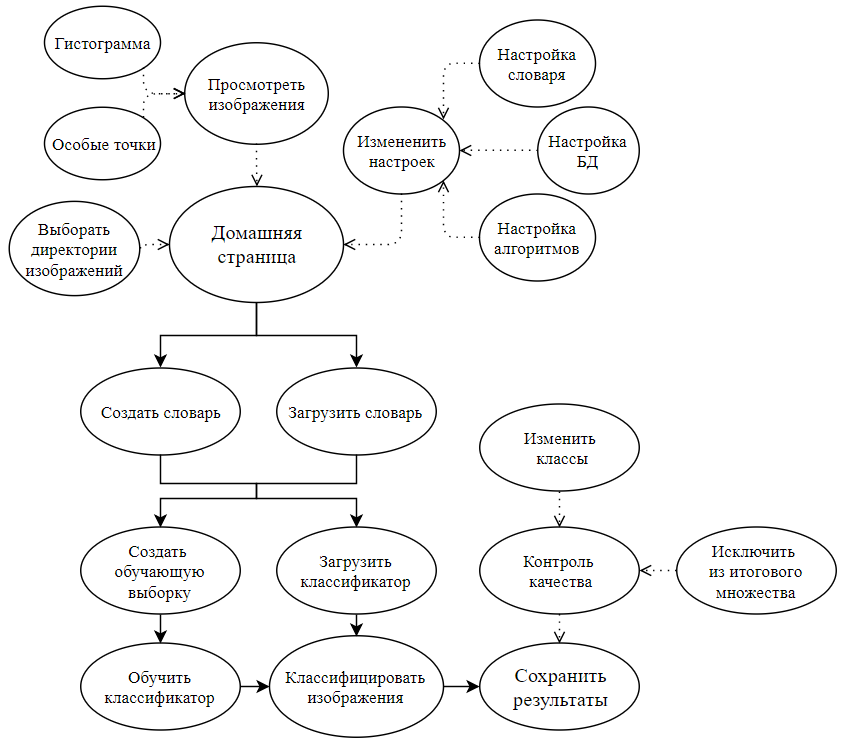


Рис 2.1 UML диаграмма

Домашняя страница, на которую попадает пользователь при запуске программы, позволяет воспользоваться всеми необходимыми функциями для достижения поставленной цели (рис 2.2).

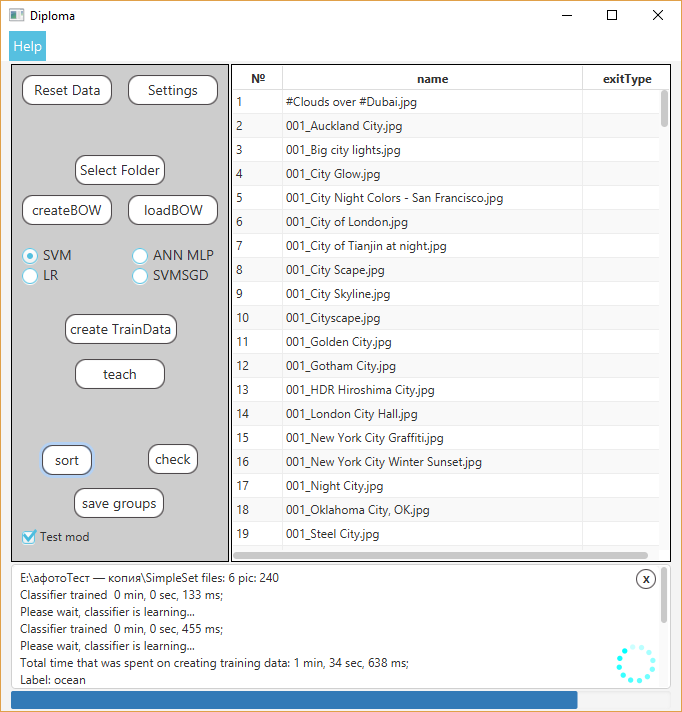


Рис 2.2 Домашняя страница

Перед началом работы доступен выбор основных настроек приложения, позволяющих провести необходимые исследования в рамках данной работы (рис 2.3):

* Выбор алгоритма поиска особых точек и дескрипторов ORB, BRISK или AKAZE
* Максимальный размер стороны изображения, до которого оно будет сжато, этот параметр влияет на количество особенностей, найденных на изображениях (для ORB их максимальное количество ограничено 500)
* Размер словаря, количество слов, которые будут сформированы на этапе создания словаря.
* Опции сохранения в базу данных

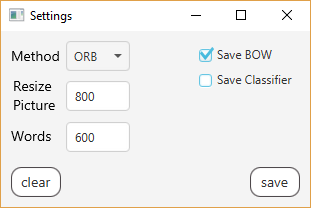


Рис 2.3 Настройки приложения

Выбор каталога с изображениями представлять собой следующий вид (рис 2.4):

* Каждая дочерняя папка, выбранного каталога, считается как отдельная группа, на этапах создания обучающей выборки и тестирования приложения на тестовых данных.
* На этапах формирования словаря и классификации изображений изображения обрабатываются не зависимо от их расположения.

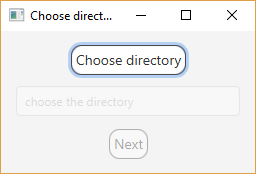


Рис 2.4 Выбор директории

После обработки выбранного каталога, загруженные изображения отображаются в таблице (рис 2.2). Выбор произвольной строки в таблице позволяет просмотреть изображение (рис 2.5), найденные особые точки, в зависимости от выбранного алгоритма, и, при наличии словаря, визуализировать гистограмму визуальных слов (рис 1.5).

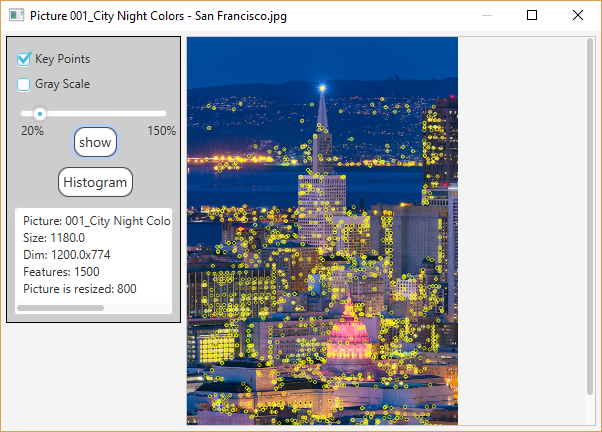


Рис 2.5 Просмотр изображений

После запуска создания словаря инициализируется соответствующий процесс, описанный в пункте 1.2. Загрузка словаря позволяет использовать ранее сформированного и сохраненного в базу данных словаря. Список содержит словари, подходящие под заданные настройки приложения (рис 2.6).

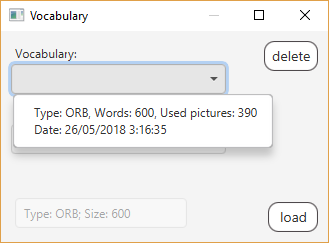


Рис 2.6 Загрузка словаря

Далее, для обучения классификатора, создается обучающая выборка по выбранному каталогу. Обучение классификатора осуществляется по процессу, описанному в пункте 1.3. Так же возможно загрузить ранее обученный классификатор, в зависимости от выбранного словаря (рис 2.7). Оптимальные параметры классификатора подбирались эмпирическим путем, на основе проведения тестов.

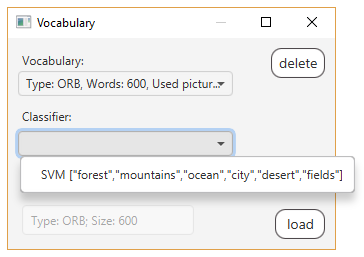


Рис 2.7 Загрузка классификатора

Если сохраненные данные не являются более актуальными, то пользователь может их удалить. При удалении словаря, так же удаляются все данные классификаторов, обученных по нему.

Результаты классификации отображаются в таблице изображений в колонке “Label”, контроль качества классификации осуществляется по сопоставлению выходного класса и названия каталога, из которого было загружено тестовое изображение (рис 2.8).

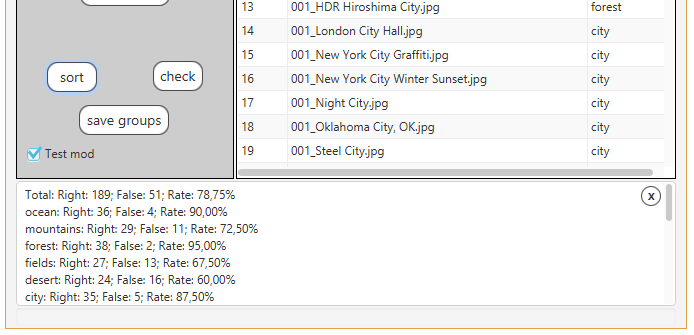


Рис 2.8 Результаты классификации тестовой выборки

Следующим шагом является проверка и при необходимости редактирование назначенных классов для изображений, так же пользователь может исключить изображение по каким-либо критериям из итогового множества, назначив соответствующий класс (рис 2.9).

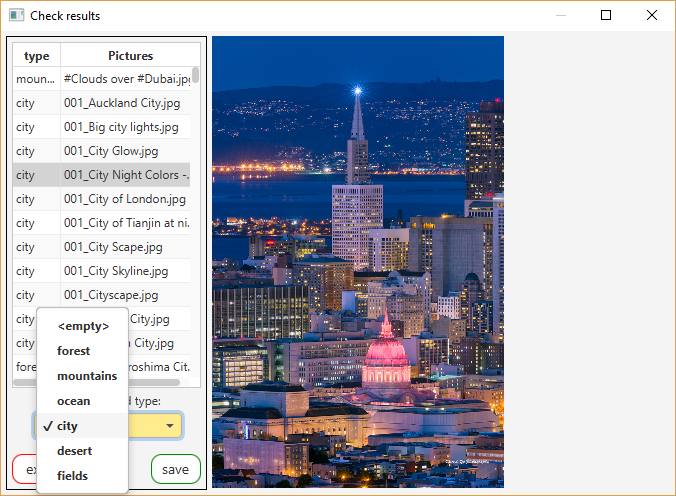


Рис 2.9 Контроль результатов классификации

Данный функционал значительно ускоряет работу по сортировке изображений, что является одной из главных целей работы.

После редактирования пользователю предлагается сохранить изображения по выбранным группам в указанный каталог для дальнейшего использования.

На всех этапах работы программы осуществляется контроль целостности, и при отсутствии необходимых данных на каком-либо этапе работы, пользователь получает соответствующее сообщение о требуемых данных или о возникновении ошибки (рис 2.10).

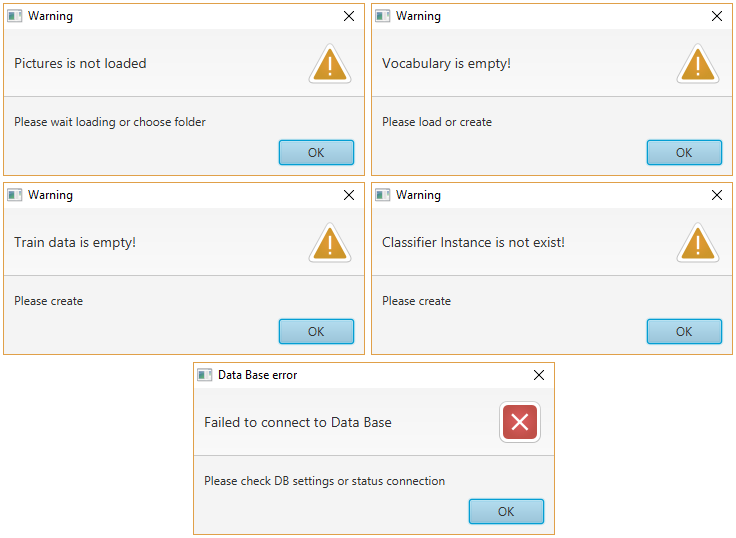


Рис 2.10 Примеры информационных сообщений

## Проектирование базы данных

В качестве базы данных была выбрана [свободная](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%9F%D0%9E) [реляционная система управления базами данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94) MySQL. Выбор данной системы обусловлен ее гибкостью и простотой, но в то же время ее функционал в полной мере позволяется реализовать поставленные задачи.

Применение базы данных позволяет ускорить получение результатов при повторном использовании полученных ранее данных словаря и классификатора, или сократить временные затраты на переобучение последнего.

Рассмотрим спроектированную базу данных (рис 2.11.

Group представляет информацию об обрабатываемых классах изображений.

Classifier и Feature являются словарями и содержат допустимые значения используемых алгоритмов.

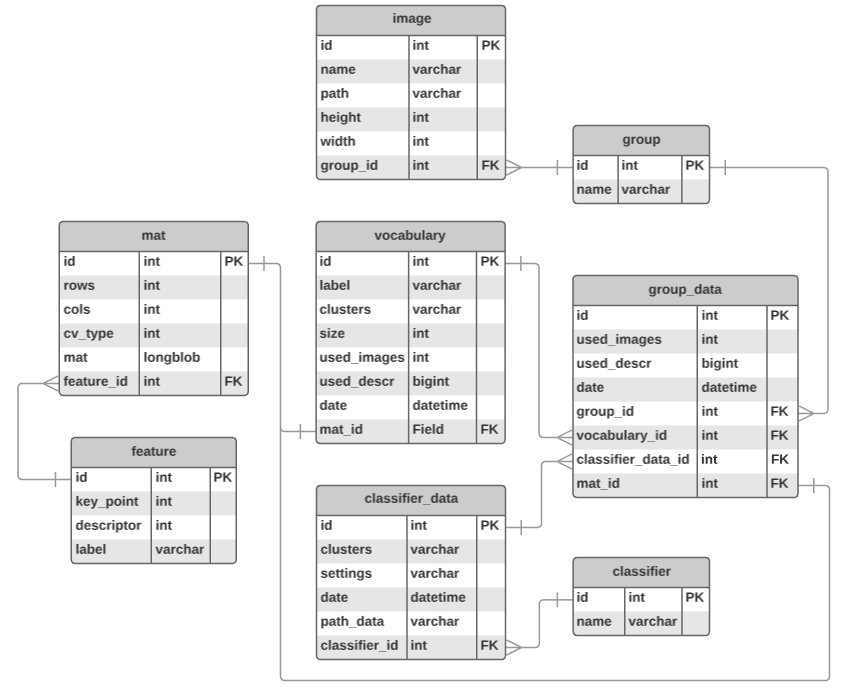


Рис 2.11 Схема базы данных

Таблицу Image предполагается использовать при необходимости поиска изображений определенного типа, она содержит в себе следующие поля:

* Имя изображения
* Путь до него
* Разрешение – высота и ширина
* Идентификатор группы

Vocabulary содержит информацию о словарях:

* Имя выбранного каталога
* Классы, на основе которых формировался словарь
* Количество визуальных слов
* Количество использованных изображений
* Изначальное количество дескрипторов
* Дата создания словаря
* Идентификатор матрицы словаря

Group\_data хранит данные об обучающих выборках, необходимых для обучения классификатора:

* Количество изображений в тестовой группе
* Количество дескрипторов в ней
* Дата создания данных
* Идентификатор группы
* Идентификатор словаря
* Идентификатор классификатора
* Идентификатор матрицы гистограмм группы

Classifier\_data требуется для хранения данных об обученном классификаторе:

* Классы, на основе которых обучался классификатор
* Настройки алгоритма
* Дата создания
* Путь до xml файла (данные классификатора сохраняются средствами OpenCV)
* Идентификатор алгоритма классификатора

Таблица mat содержит объект типа Mat, используемой библиотеки OpenCV, и информацию для корректного восстановления данного типа из массива байтов:

* Количество строк
* Количество столбцов
* Формат данных
* Массив байт
* Идентификатор особенностей, на основе которых был получен данный объект

# Глава 3. Результаты работы приложения

Для проверки стабильности и подбора оптимальной настройки алгоритмов был проведен ряд тестов, заключающихся в создании словаря различного объема и типа особенностей.

Алгоритм работы программы:

* + 1. На вход подается группа изображений.
    2. Выбираются настройки словаря: количество слов и тип особенностей изображения ORB, BRISK, AKAZE.
    3. Вычисляется словарь визуальных слов.
    4. Выбирается обучающая выборка изображений.
    5. Вычисляются характеристики групп.
    6. По полученным данным обучается алгоритм классификации.
    7. Оценивается качество классификации на обучающем множестве.
    8. Далее классификатор применяется на выбранном множестве изображений, которые требуется классифицировать.

Работа приложения оценивалась на двух группах:

* + 1. Изображения общего характера – в данной выборке предлагается изучить возможности классификации на изображениях, содержащих в себе преимущественно представление о их тематики.
    2. Изображения объектов – в данной выборке наоборот содержится только информация об определенных объектах.

Анализ полученных результатов позволит понять сильные и слабые стороны рассмотренного комплекса алгоритмов компьютерного зрения.

## Изображения общего содержания

В качестве исследуемых типов были рассмотрены следующие 6 групп изображений: города, пустыни, поля, леса, горы, океаны

Общее количество изображений для формирования словаря - 390, по 65 на каждую группу. Таблица 3.1 отображает общее количество найденных особенностей, по которым формируется словарь визуальных слов. Можно отметить существенное различие этого параметра у рассматриваемых алгоритмов, что следует из разного подхода детектора особых точек.



Таблица 3. 1 Общее количество найденных особенностей

На графике 3.2 представлена зависимость времени, затраченного на формирования словаря, от заданного количества слов в нем.

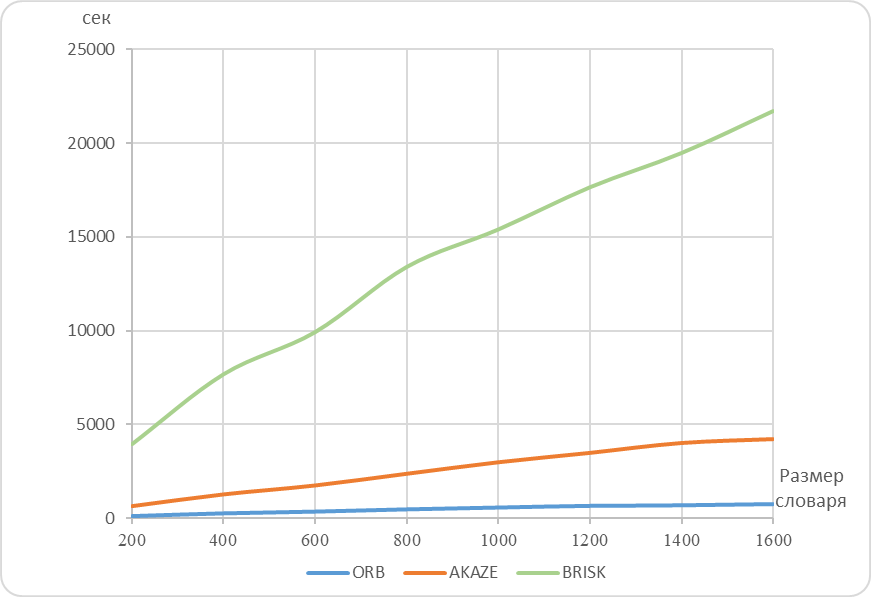


График 3.2 Зависимость затраченного времени от количества слов в словаре

Как можно заметить, вычисление визуальных слов линейно зависит от общего количества особенностей, что приводит к увеличению итогового времени на формирование словаря.

Далее рассмотрим получение результаты приложения на размерах словаря 200, 400, 600, 800, 1000, 1200, 1400 и 1600 слов, дальнейшее увеличение этого показателя не приводило к улучшению качества классификации на этапе тестирования программы.

На графиках 3.3, 3.4 и 3.5 приведены результаты тестов на основе тестовой выборки по 40 изображений в 6 группах на исследуемых алгоритмах поиска особенностей ORB, AKAZE и BRISK.

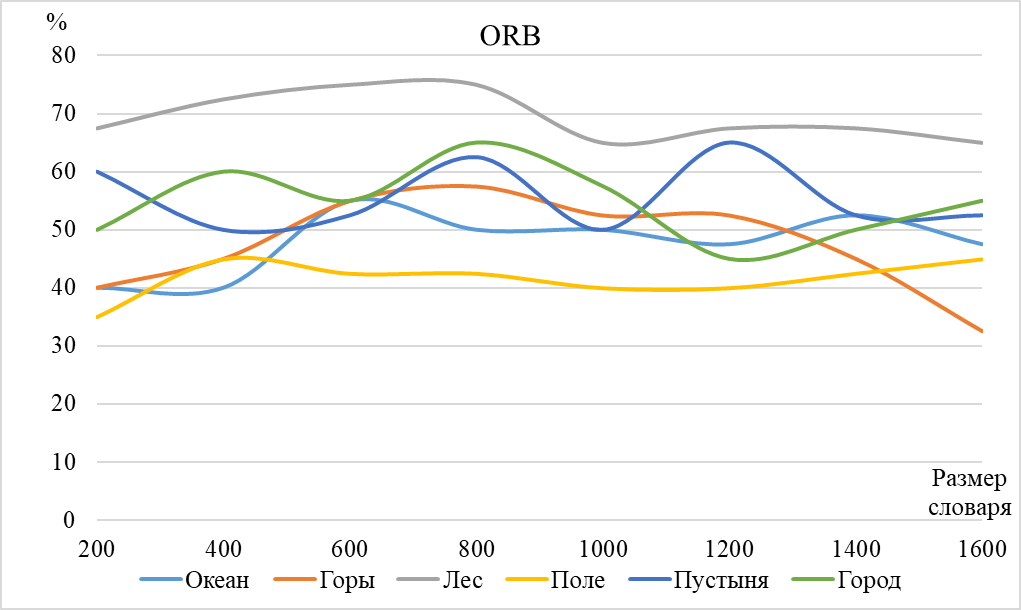


График 3.3 Результат работы с алгоритмом ORB

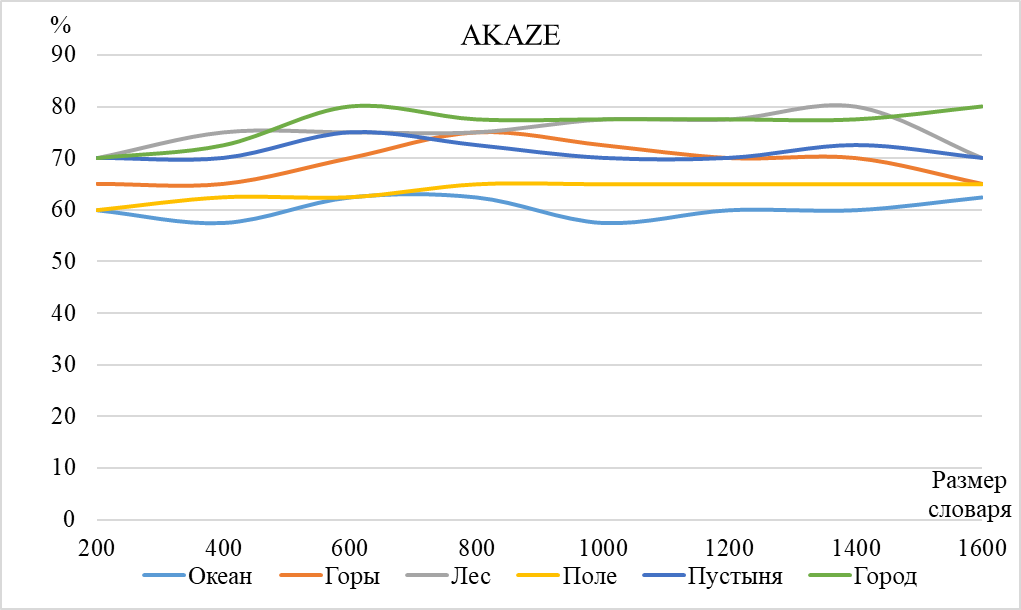


График 3.4 Результат работы с алгоритмом AKAZE

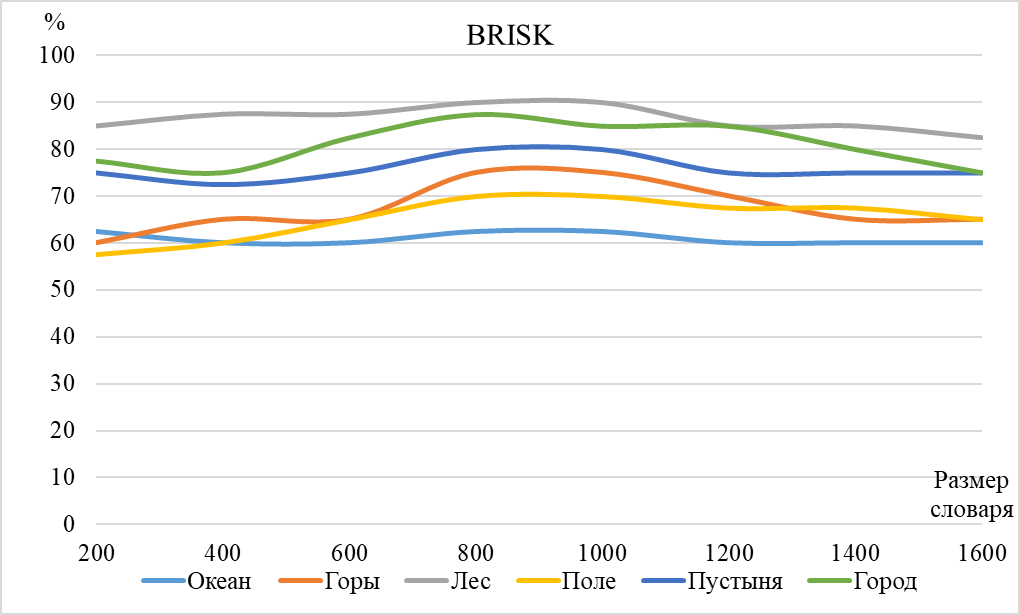


График 3.5 Результат работы с алгоритмом BRISK

На графике 3.6 отображается средняя итоговая точность по данным алгоритмам.

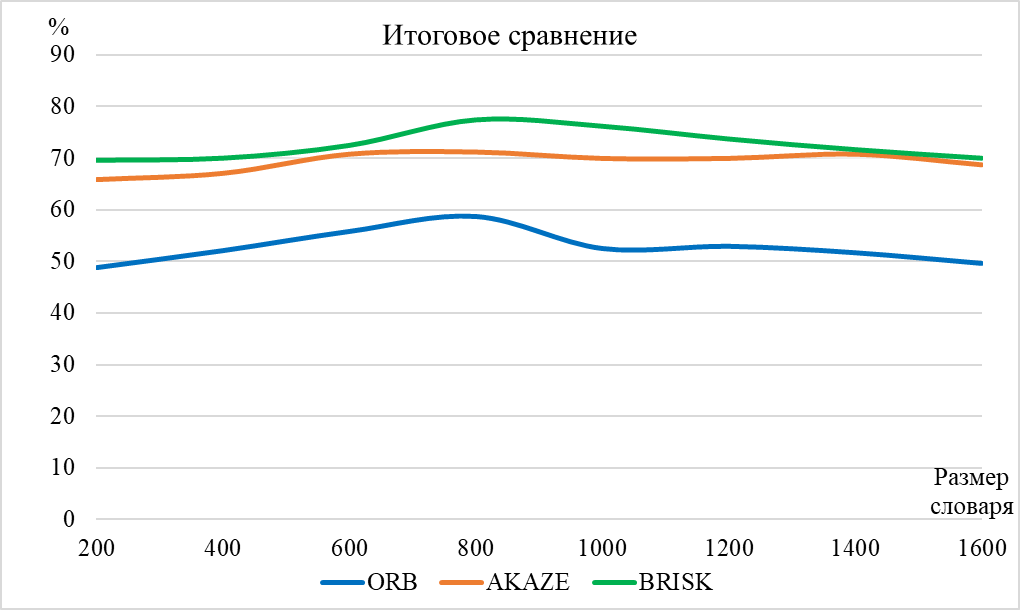


График 3.6 Средняя точности алгоритмов по всем группам

Лучший результат удалось достичь для тестовой выборки на словаре из 800 слов для всех алгоритмов, а наилучший 77,5% - с применением BRISK.

Далее приведено (таблица 3.7) поведение приложения на более объемной коллекции изображений, в сумме 2900 из 6 групп.



Таблица 3.7 Тестирование на 2900 общих изображениях

Итоговый результат 72,76% при применении BRISK.

## Изображения определенных объектов

Рассмотрим работу приложения на задаче поиска объекта на сцене изображения, для этих целей были составлены 8 групп различных шаблонов, в сумме 838 фотографий. Тестирование проводилось на размере словаря 600, 800 и 1000 (таблица 3.8).



Таблица 3.8 Тестирование на 838 шаблонных изображениях

Классификация статических объектов, представляет собой сильную сторону алгоритмов поиска особенностей, так как поиск похожих дескрипторов позволяет высокой точностью определить его пару на другом изображении этого же объекта. Рассмотрим это на примере классификации объектов, сфотографированных с разных ракурсов и расстояний (таблица 3.9)



Таблица 3.9 Тестирование на 240 статических изображениях

В данном случае все алгоритмы показывают отличные результаты.

Как и в предыдущем параграфе наилучшие результаты получены при применении алгоритма BRISK.

# Выводы

По приведенным исследованиям можно сказать, что так как у алгоритма ORB фиксированное максимальное количество дескрипторов на изображении – 500, то этого не хватает для полноценного формирования словаря и дальнейшей классификации, но данное ограничение алгоритма существенно облегчает трудозатраты вычислений. AKAZE определяет меньшее количество дескрипторов чем BRISK, и за счет этого работает быстрее, но итоговое качество значительно лучше чем у ORB. Благодаря большему количеству особенностей, алгоритм BRISK позволяет получить хорошие итоговые результаты на тестируемых выборках изображений. Можно сделать вывод, что на конечное качество кластеризации влияет количество дескрипторов и их структура, что позволяет определить наиболее часто встречаемые визуальные слова на изображениях одного типа. То есть применяемые алгоритмы хорошо работают только на изображениях с большим количеством выделяющихся различных деталей и плохо подходят для распознавания текстурных изображений. Но в задачах поиска известного шаблона исследуемые алгоритмы показывают себя с лучшей стороны, позволяя с высокой точностью классифицировать фотографии, содержащие искомый шаблон. Схожие результаты были получены в бакалаврской работе автора данной работы.

Плюсами приведенного подхода являются:

* + 1. Простота использования, все действия автоматизированы и не требуют вмешательства пользователя в работу алгоритмов.
    2. Быстродействие, для работы с программой не требуется применение высокопроизводительных машин.
    3. Хорошие результаты классификации на общих тематиках изображений, но существенно меньше чем при применении готовых нейронных сетей от крупных IT компаний, в частности от Google и Microsoft.
    4. Отличные результаты классификации определенных объектов на изображениях.

Так же стоит отметить и явные минусы приведенного подхода:

* + 1. Обработка изображения осуществляется без использования пространственной информации, то есть наличие посторонних объектов на сцене негативно сказывается на итоговую работу классификатора.
    2. Требуется составление эталонной выборки, из пункта 1.
    3. Не стабильность, возможны резкие скачки точности классификации при незначительных вариациях количества слов в словаре.
    4. Результат классификации представляет всего 1 класс для изображения, но в тоже время на нем могут присутствовать участки, принадлежащие нескольким типам. Современные нейронные сети могут классифицировать большую часть объектов на одной фотографии одновременно и с точность более 90 %.

Проведение дополнительных тестов на более объемном множестве групп изображений по техническим ограничениям провести не удалось. Так же необходимо изучить оптимальные размеры словаря в зависимости от количества групп, но данные исследования выходят за рамки данной работы.

Полученные результаты вполне могут говорить о достижении поставленной цели, так как разработанное приложение позволяет сократить время на сортировку изображений с приемлемой точностью или другими словами - подготовить обучающую выборку.

# Заключение

В данной работе был проведен сравнительный анализ методов поиска особых точек и их дескрипторов на основе результатов кластеризации. Для достижения поставленной задачи была разработана программа для стационарных устройств на языке Java, обладающая всем требуемым функционалом:

1. Реализацию методов ORB, BRISK, AKAZE.
2. Формирование словаря.
3. Обучение алгоритма кластеризации.
4. Сортировка изображений по заданным группам.
5. Настройка параметров программы и алгоритмов.
6. Корректировка групп изображений
7. Сохранение результатов

Были проведены сравнительные тесты работы алгоритмов.

В результате выделен наилучший метод поиска особенностей, при кластеризации изображений.

# Список литературы

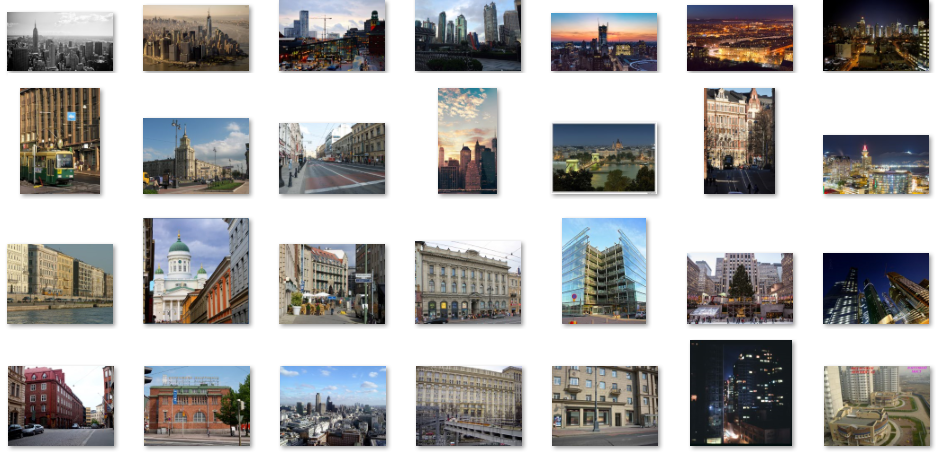
1. Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski: "ORB: an efﬁcient alternative to SIFT or SURF", *Computer Vision (ICCV), IEEE International Conference on. IEEE*, pp. 2564 – 2571, 2011.
2. Stefan Leutenegger, Margarita Chli, Roland Siegwart: “BRISK: Binary Robust Invariant Scalable Keypoints”.  *Computer Vision (ICCV),* pp. 2548 – 2555, 2011.
3. Pablo F. Alcantarilla, Jesús Nuevo, Adrien Bartoli: “Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces”. *In British Machine Vision Conference (BMVC*), 2013.
4. Rosten, Edward, Tom Drummond: "Machine learning for high-speed corner detection”, *9th European Conference on Computer Vision (ECCV)*, pp. 430 – 443, 2006.
5. Michael Calonder, Vincent Lepetit, Christoph Strecha, Pascal Fua, “BRIEF: Binary Robust Independent Elementary Features”, *11th European Conference on Computer Vision (ECCV)*, pp. 778 – 792, 2010.
6. S. Grewenig, J. Weickert, C. Schroers, A. Bruhn: “Cyclic Schemes for PDE-Based Image Analysis”, *In International Journal of Computer Vision, 2013.*
7. X. Yang, K. T. Cheng: “LDB: An ultra-fast feature for scalable augmented reality”. *In IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*, pp. 49 – 57, 2012.
8. [Josef Sivic](https://dl.acm.org/author_page.cfm?id=81300230901&coll=DL&dl=ACM&trk=0), [Andrew Zisserman](https://dl.acm.org/author_page.cfm?id=81100465760&coll=DL&dl=ACM&trk=0): “A Text Retrieval Approach to Object Matching in Videos”
9. Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cedric Bray: “Visual Categorization with Bags of Keypoints”, 2004.
10. David Arthur, Sergei Vassilvitskii: "*k*-means++: the advantages of careful seeding.” *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics Philadelphia, PA, USA. pp. 1027–1035, 2007.
11. C. Burges: “A Tutorial on Support Vector Machines for Pattern Recognition
12. Annalisa Barla, Francesca Odone, Alessandro Verri: “Histogram intersection kernel for image classification.” *Proceedings of International Conference on Image Processing 2003,* Vol. 2(1) , pp. 513, 2003
13. OpenCV: http://opencv.org

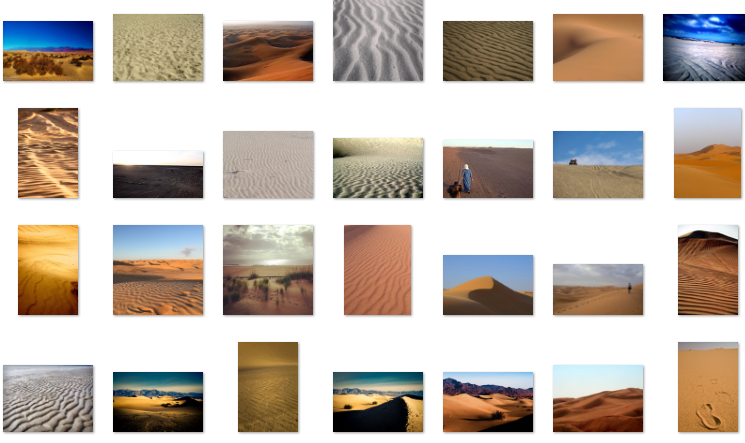
# Приложение

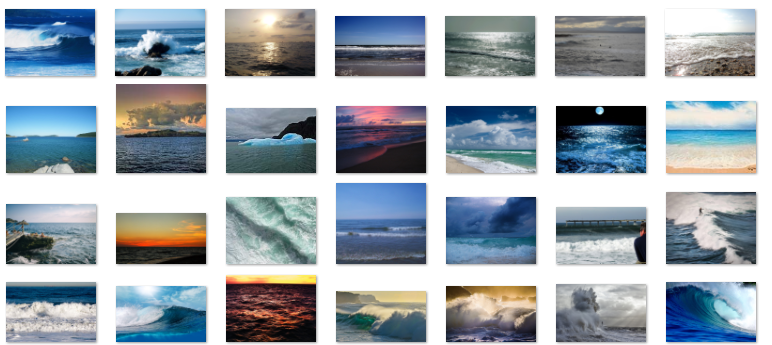
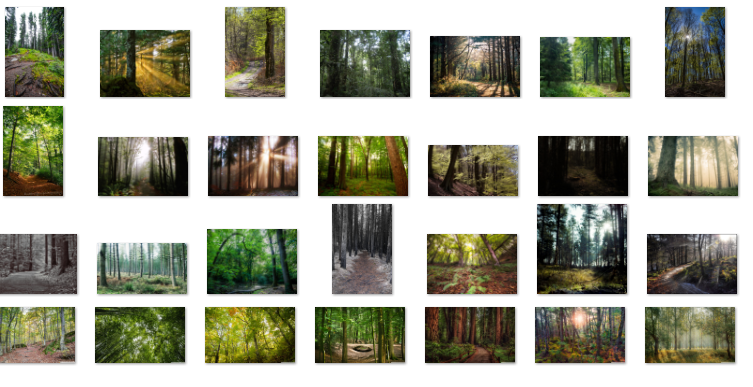
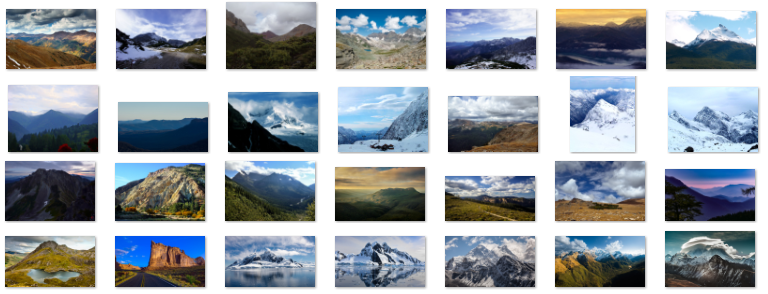
**Приложение 1. Затраченное время на формирование словаря**



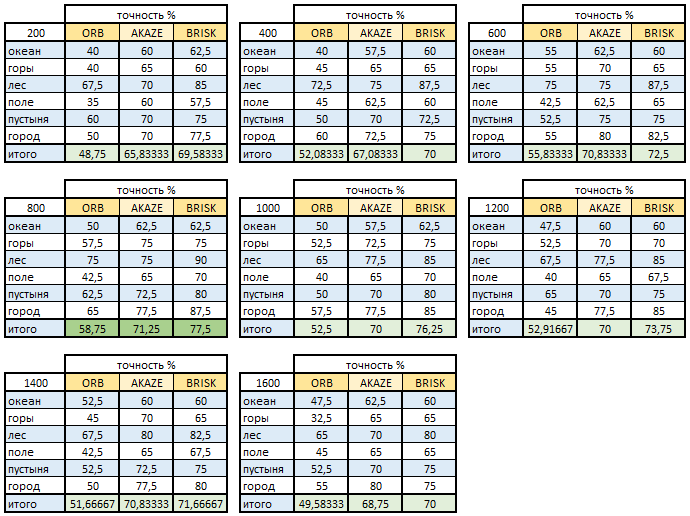
**Приложение 2. Примеры изображений первой группы**



****

****

**Приложение 3. Подробные результаты работы приложения на первой группе изображений**



**Приложение 4. Примеры изображений второй группы**





