Санкт-Петербургский государственный университет

Кафедра технологии программирования

**Бородин Анатолий Анатольевич**

Магистерская диссертация

**Извлечение данных о слайдах презентации из учебного видео**

Направление 010300

Информационные технологии

Магистерская программа Технологии баз данных

Научный руководитель,  
старший преподаватель  
Севрюков С. Ю.

Санкт-Петербург

2016

Оглавление

[Введение 2](#_Toc451717396)

[Формулировка цели, постановка задачи 4](#_Toc451717397)

[Обзор предыдущих исследований 5](#_Toc451717398)

[Объект исследования 7](#_Toc451717399)

[Классификация видео-лекций 7](#_Toc451717400)

[Способы хранения презентаций 7](#_Toc451717401)

[Сценарий работы продукта 8](#_Toc451717402)

[Формирование требований 9](#_Toc451717403)

[Предобработка изображения и работа с видеоматериалом 11](#_Toc451717404)

[Фильтрация изображения 11](#_Toc451717405)

[Бинаризация изображений 14](#_Toc451717406)

[Детектор границ Canny 20](#_Toc451717407)

[Перспективное преобразование 23](#_Toc451717408)

[Сопоставления изображений 25](#_Toc451717409)

[Перцептивный хэш 25](#_Toc451717410)

[SIFT (Scale Invariant Feature Transform) 28](#_Toc451717411)

[SURF (Speeded up robust features) 31](#_Toc451717412)

[Оптическое распознавание символов 33](#_Toc451717413)

[Реализация программного продукта 34](#_Toc451717414)

[Тестирование программного продукта 37](#_Toc451717415)

[Выводы 41](#_Toc451717416)

[Заключение 42](#_Toc451717417)

[Список литературы 44](#_Toc451717418)

[Приложение 1. Сценарий использования программы 46](#_Toc451717419)

[Приложение 2. Образцы видео, используемых для тестирования программы 47](#_Toc451717420)

**Введение**

Информационные технологии за последние десятилетия плотно вошли в нашу жизнь. Они облегчают рабочий процесс, а также вводят новые возможности. И сейчас одним из важных направлений является развитие человека. Человек развивается всю свою жизнь, от рождения до смерти. Одним из удобных методов изучения новой информации является дистанционное обучение, так как данный формат позволяет получать знания независимо от географических или физических возможностей.

История дистанционного образования насчитывает около 200 лет. Его прототипом стали почтовые отправления Исаака Питмана своим студентам, которых он обучал стенографии. Более серьезным шагом было создание отделения дистанционного обучения в университете Чикаго, которое вскоре распространилось на весь мир. В частности, оно широко применялось для обучения детей, которые не могли ходить в обычные школы. Один из созданных центров во Франции дошёл до наших дней и сейчас является крупнейшим учебным заведением дистанционного обучения в Европе. В СССР методика дистанционного обучения получила название – “консультационной” или же “заочной”.

В общем, дистанционное обучение продолжает развиваться с каждым днём. И это легко объяснимо, так как происходит развитие общества, что влечет за собой рост интернет аудитории. И, кроме того, растёт потребность человека к знаниям. Одним из важных факторов успешности дистанционного обучения является компьютерная грамотность пользователя. Учащиеся таких образовательных программ обязаны владеть навыками работы с электронной почтой, учебными системами и другим инструментарием. Однако не всегда у пользователя хватает навыков для разрешения всех трудностей.

Например, иногда требуется получить информацию из видео в более содержательном виде. Но, к сожалению, не все лекции позволяют это сделать без проблем. Первый недостаток заключается в том, что не всегда видео-лекции записываются на хорошее оборудование и порой авторам приходится дополнительно монтировать записанное видео - править, создавая на экране отдельную область с электронными слайдами. Другой недостаток - это отсутствие навигации. Например, бывает ситуация когда нужно попасть на определенный слайд, и в таком случае отсутствие навигации будет серьезным минусом, так как заставит человека потратить часть времени на поиск этого момента в видео-лекции. Еще одним недостатком служит отсутствие индексации видео, что порой может запутать пользователя, особенно когда тема популярна и её заголовок можно трактовать по-разному, например “Cache” СУБД и “Cache” CPU.

Для уменьшения количества таких ситуаций стоит разработать инструментарий, который поможет разрешить такие проблемы с минимальными людскими и временными затратами. Это позволит сделать обучение более эффективным.

**Формулировка цели, постановка задачи**

Основная цель работы: улучшение эффективности дистанционного обучения с помощью видео-лекций путем упрощения навигации по видеоматериалу.

Основная задача работы: написание системы анализа видеоизображения, с целью извлечения из него информации о видео лекциях. Для возможной синхронизации видео-лекций с электронными слайдами, а также для создания навигации по видео-файлу.

Подзадачи, которые необходимо решить для достижения основной задачи:

1. Анализ предметной области
   1. Изучение предыдущих исследований
   2. Классификация видео-лекций
2. Формирование требований к программному продукту
3. Анализ методов обработки изображений
   1. Фильтрация
   2. Бинаризация
   3. Сопоставления
4. Разработка прототипа
5. Тестирование прототипа
   1. Проверка работы приложения на тестовых данных
   2. Проверка работы приложения при распараллеливании вычислений

**Обзор предыдущих исследований**

С каждый годом растет количество видео-лекций с презентациями. В слайдах видео-лекций содержатся полезные данные, которые можно использовать для такой задачи, как синхронизация слайдов и видео-лекций, и работы по такой синхронизации идут уже давно. Первые [1] статьи появились в середине 90-ых годов и основывались на ручном редактировании видео-файла, посредством установления временных меток – моментов переключения слайдов. На сегодняшний день существует программные продукты, основанные на данном подходе - MediaSite и Microsoft Producer. Они могут синхронизировать электронные презентации вместе с видео-лекциями и выводить слайды бок о бок с видео. Однако эти системы не способны производить синхронизацию в автоматическом режиме. В качестве решения этой проблемы в публикации [2] было предложено использовать систему, которая получала бы видеосигнал от проектора, и, основываясь на сравнении двух кадров, определяла бы момент переключения слайда. Естественным недостатком является требование дополнительных аппаратных или программных средств на момент проведения презентаций.

Для устранения такого рода недостатков было предложено несколько решений, основанных на автоматическом подходе без использования дополнительного оборудования. Такие работы [3,4], обычно акцентируют своё внимание на двух этапах:

1. Определение местоположения слайдов.

2. Идентификация слайдов.

В первую очередь, стоит выделить публикацию [5], в которой был предложен подход по классификации изображений, и на основе класса изображения к последнему применялись определенные методы, например, сравнение гистограмм, маски строки или контуров фона.

Другие решения основывались на оптическом распознавании символов (OCR) для извлечения текста из видеокадров и сравнении их с электронными слайдами[6], но данный метод обладает рядом недостатков – смена слайдов, на которых присутствует только графика, и чувствительность к качеству изображения. Однако данный метод будет полезен в этой работе, по причине извлечения текста – информации.

Публикация [7] основана на детектировании особых точек и их сравнении на нескольких изображениях. Примером такого подхода выступает метод SIFT (Scale-invariant feature transform). У него также есть значительные недостатки – большие временные затраты на обработку одного кадра, что накладывает определенные трудности и делает почти невозможным сравнение абсолютно каждого кадра со слайдами в объёмных презентациях. Но результаты метода были весьма впечатляющие за счет огромной проделанной работы. Так, был написан классификатор для определения слайдов, в котором использовались не только локальные, но и глобальные признаки изображения. Однако стоит отметить, что в работе не рассматривались ни классификация самих видео-лекций, ни дополнительные модули, такие как OCR системы.

# Объект исследования

## Классификация видео-лекций

Прежде чем приступать к исследованию, стоит рассмотреть классификацию видео-лекций, на которых могут присутствовать слайды. Такие видео-лекции могут быть трех типов:

1. Только электронные слайды, записанные вместе с голосом лектора.
2. Выступление преподавателя на лекции, записанное на камеру.
3. Комбинирование первого и второго случая.

В первом случае из видео-слайдов презентации можно получить время переключения слайдов для дальнейшей навигации, а также список слов для создания индексации, как по видео, так и в какой-нибудь внешней системе. Во втором случае дополнительно к полученным данным из первого случая, можно склеить качественные слайды из презентации и видеопотока. Третий случай будет рассматриваться как второй.

## Способы хранения презентаций

Существует несколько способов хранения презентаций в электронном виде.

PDF – Portable Document Format. Является одним из самых распространенных форматов хранения электронных документов. Для извлечения данных из PDF существует множество программ редакторов, таких как Adobe Acrobat DC/PRO, Foxit PDF Reader, и проекты с открытым исходным кодом – PDFBox, iText, PDFMiner. Последними пользоваться предпочтительнее, так как большинство крупных платных проектов не предоставляют API (Application Programming Interface – Интерфейс прикладного программирования), который бы позволил с помощью одного из языков программирования полностью автоматизировать процесс извлечения данных. Также немаловажным является тот факт, что будущая программа разрабатывается для свободного использования, в то время как коммерческие проекты разрешают пользоваться своим API (в случае если они предоставляют его), только после приобретения продукта. Поэтому выбор был сделан в пользу проектов с открытым исходным кодом.

Под язык программирования Python существует несколько таких проектов, самые популярные – PDFMiner и PyPDF, и после ознакомления с этими двумя проектами предпочтение было отдано PDFMiner.

При работе с PDFMiner структура PDF страницы может состоять из текстовой зоны, растровых и векторных изображений, линий. В данной работе используется только анализ текста, а также возможное извлечение изображений из PDF файлов и поиск их на видео-фреймах.

Также стоит обратить внимание на формат PPTX, который относится к продукции Microsoft Office PowerPoint и тоже является одним из популярных форматов хранения презентаций. Для извлечения информации из PPTX существует большое количество программных продуктов. По аналогичным причинам, как у PDF продуктов, будет использоваться проект с открытым исходным кодом – “Python-pptx”.

## Сценарий работы продукта

На вход прототипу подаётся видео-лекция и электронная презентация в формате PDF или PPTX. Пользователь выбирает тип видео-лекций, и больше от него участия не требуется. После этого из электронной презентации извлекается информация в виде текста и для каждого слайда создается словарь слов, с которым в дальнейшем будет взаимодействовать один из модулей приложения. При необходимости презентация разбивается на отдельные слайды и переводится полностью в формат PNG. Аудиопоток из видео-лекций использоваться не будет. Для видео-лекций первого типа изображение из видеопотока будет сравниваться с изображением-эталоном из электронных слайдов для получения текущего номера слайда. В свою очередь, для видео-лекций второго и третьего типов, сначала на изображении из видеопотока будет находиться область расположения экрана с презентацией, из этой области с помощью оптического распознавания символов текст будет извлекаться и сравниваться со словарем для получения текущего номера слайда. В дальнейшем программа будет автоматически сверяться с предыдущими кадрами, и если система определит значительное отклонение в изображениях, тогда снова начнется определение номера слайда на изображении. После окончания работы программа должна предоставить пользователю следующую информацию о слайдах: какой слайд был в определенный момент времени на видеоизображении, а также список слов для определенного слайда.

## Формирование требований

Перед разработкой прототипа необходимо выдвинуть ряд требований к будущему продукту:

* Поддерживаемые форматы видео-лекций: avi, mp4
* Поддерживаемые форматы презентаций: PDF, PPTX
* Наличие графического интерфейса
* Возможность выбора алгоритма для достижения оптимального соотношения время - качество
* На выходе пользователь должен получить информацию о времени переключения слайдов
* Дополнительные настройки должны включать в себя возможность получения текста со слайдов

Данные требования будут уточняться и пополняться при разработке прототипа.

# Предобработка изображения и работа с видеоматериалом

При работе с цифровыми изображениями можно столкнуться с проблемами, которые возникли на этапе съёмки: расфокусировка объектива видео- или фотокамеры, высокая настройка чувствительности к свету (ISO). Далеко не всегда есть возможность переснять материал, поэтому приходится работать с тем, что есть. Для решения данных проблем применяют фильтрацию изображений.

## Фильтрация изображения

Одной из основополагающих операций компьютерного зрения является фильтрация изображений. Под задачей "фильтрации изображений" можно понимать любые действия по обработке изображений, при условии, что на вход будет подаваться одно изображение, а на выходе будет создаваться уже другое, видоизмененное, изображение. Существует несколько типов фильтраций, их также называют фильтрами – линейные и нелинейные. Каждый из данных фильтров используется для определенных целей. Так, если необходимо уменьшить шум у изображения либо выделить особые зоны, используют линейные фильтры. Для операций же с бинарными изображениями хороши нелинейные. В данной работе рассмотрим более подробно только линейные фильтры.

Применение линейного фильтра к изображению – это по сути свертка изображения по заданной функции. В формуле (1.1) представлена свертка функции по и , результатом которой является новое изображение

с новыми значениями в каждом пикселе, вычисляемыми посредством среднего от значений соседних элементов. Значения и – коэффициенты, определяемые из значения фильтров (1.2).

К линейным фильтрам относятся сглаживающие фильтры, которые размывают изображение – происходит потеря резкости у изображения, делая его нечетким и размытым. Самым простым примером такого фильтра является прямоугольник с размером матрицы , значения которой могут быть получены как результат выражения (1.3). Сумма по всем элементам матрицы соответственно будет равняться единице.

Представим, что есть изображение, состоящее из полностью черного фона с единственной белой точкой. Если применим фильтрацию с вышеописанным ядром, то результатом будет изображение с серым квадратом вместо белой точки. Размер сторон квадрата равен , то есть размеру матрицы свертки, и чем больше размер этой матрицы, тем сильнее будет размыто изображение.

У данного подхода есть как плюсы, так и минусы. К плюсам стоит отнести полное избавление от возникших шумов – шумоподавление. Результатом будет усреднение шумов, при условии, что если математическое ожидание шума будет равно нулю то, шумы соседних пикселей скомпенсируют сами себя. А вот к минусам стоит отнести существенное размытие значимых деталей изображения. Также при использовании большой маски (например, 5x5 пикселей), соседние пиксели окажут влияние аналогично дальним, что только усилит размытие изображения.

Естественно, если влияние пикселя уменьшится в зависимости от расстояния, то можно будет добиться более эффективного шумоподавления. Таким свойством обладает Гауссовский фильтр (1.4). Где - расстояние от источника в горизонтальной оси, - расстояние от источника в вертикальной оси, и - стандартное отклонение распределения Гаусса или же степень размытия.

Гауссовская фильтрация [8] является фильтрацией сглаживающего типа, но, в отличие от прямоугольной фильтрации, размытие будет равномерно распределяться от центральной точки к краям, что больше соответствует реальному размытию от расфокусировки линз. Исходя из выражения (1.4) можно сделать вывод о том, что влияние пикселей друг на друга будет обратно пропорционально их расстоянию.

Фильтр Гаусса является высокочастотным фильтром, а значит, его выгодно применять для повышения резкости изображения. После получения из видеопотока изображения каждого 30-ого кадра система автоматически применяет к кадру фильтр Гаусса, для удаления лишних шумов, а также для увеличения резкости самого изображения. Данный шаг позволяет упростить нахождение границ экрана и в целом повысить четкость изображения для дальнейшей передачи его системе распознавания символов. На рисунке 1 приведен пример. На первый взгляд изображение было ухудшено, но при использовании бинаризации изображения результаты после применения фильтра Гаусса будут лучше, чем у обычных изображений за счет размытия, границы станут более плавными, без резких скачков. Так как результатом

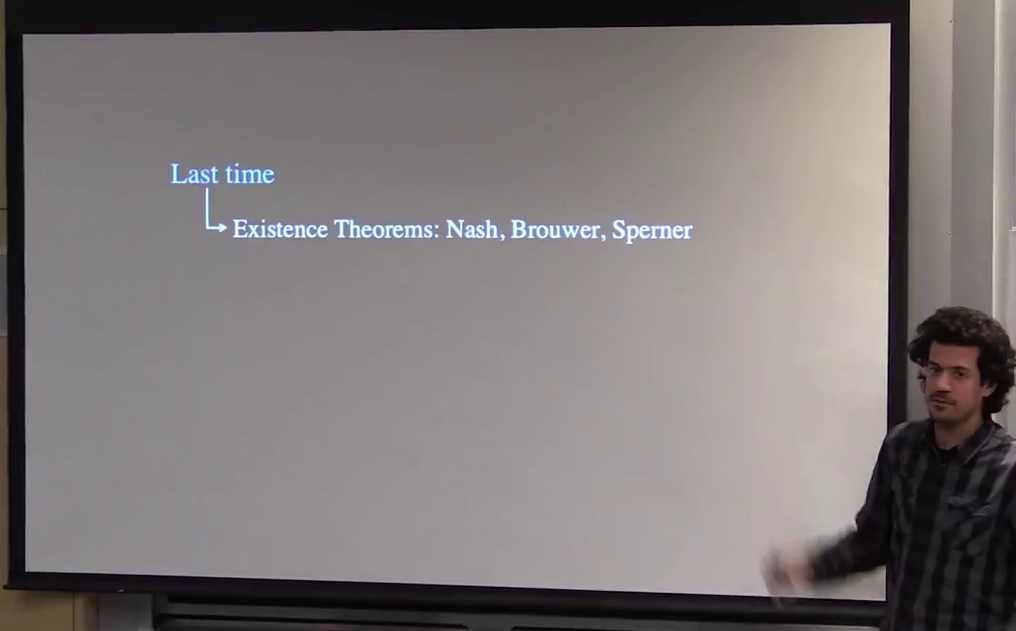
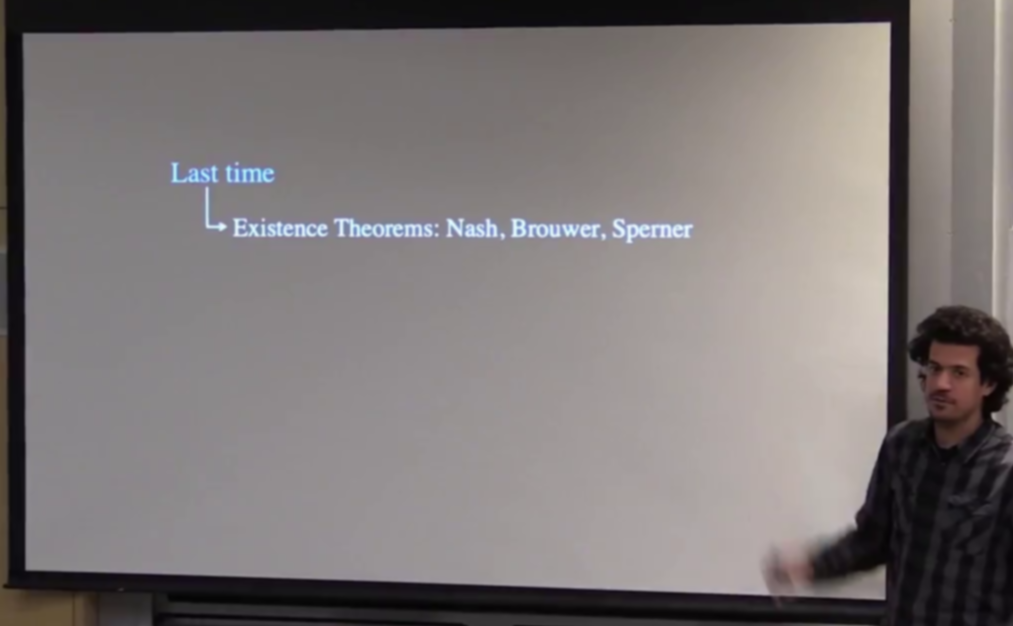


Рис. 1 - Разница между оригинальным изображением (слева) и отфильтрованным (справа)

после применения фильтра Гаусса является низкочастотное изображение, то можно получить высокие частоты (определенные границы/линии и прочие области с резким перепадом яркости пикселей) посредством простого вычитания из оригинального изображения – размытого. Для увеличения четкости оригинального изображения достаточно добавить к нему получившиеся частоты с некоторым коэффициентом. Данная операция позволит повысить точность определения границ экрана в дальнейшем.

## Бинаризация изображений

После получения отфильтрованного изображения необходимо найти на нём месторасположение экрана проектора. Зачастую встречаются экраны для проектора белого цвета, в редких случаях серого. В некоторых случаях изображения проецируется на стену/либо белую маркерную доску, что в целом не сильно отличается от проекции на экран. Иногда экран может занимать не всю площадь кадра и если рядом с экраном есть другой источник света (например, окно), то может возникнуть проблема с точным определением границ экрана. В данном случае нельзя использовать только один метод бинаризации.

Адаптивная бинаризация – адаптивный метод пороговой обработки. Основная идея алгоритма состоит в том, что каждый пиксель сравнивается с окружающими пикселями. Если значение у текущего пикселя ниже, чем среднее значение других прилежащих пикселей, то он закрашивается в черный цвет, иначе – в белый. Рассмотрим полутоновое изображение, в котором ∈ [0,255] будет интенсивностью пикселя в (x, y). В адаптивном методе пороговой обработки цель состоит в том, чтобы вычислить порог для каждого пикселя (1.5).

В методе бинаризации Saulova [9] порог вычислен, используя среднее значение и стандартное отклонение интенсивностей пикселей в зоне , центрируемой вокруг пикселя

где R - максимальное значение стандартного отклонения (R = 128 для полутонового документа), и k - параметр, который принимает положительные значения в диапазоне [0.2 0.5]. Локальное среднее значение и стандартное отклонение адаптирует значение порога согласно контрасту в локальном окружении пикселя. Когда есть высокий контраст в некоторой области изображения,, это приводит к что даёт аналогичный результат методу Niblack [10]. Однако разница проявляется, когда контраст в локальном окружении довольно низок. В этом случае порог понижается, и, таким образом, происходит успешное отделение деталей относительно темных областей фона.

После проведения ряда испытаний было выявлено, что адаптивная бинаризация на определенных данных, показывает плохие результаты. Из-за закрашивания почти всех областей в белый цвет порой сложно определить местоположение экрана проектора, если в кадре он имеет не наибольший размер. Поэтому было решено для нахождения границ использовать пороговую бинаризацию.

Пороговая бинаризация, в отличие от адаптивной, в простейшем случае работает не с конкретными участками изображения, а с изображением в целом, и цвет пикселя высчитывается определенным пороговым значением по яркости данного пикселя.

Одним из самых эффективных методов глобальной бинаризации является метод Otsu [11]. Метод использует гистограмму распределения значений яркости пикселей изображения. Строится гистограмма по значениям , где в данном случае общее количество пикселей на изображении, – пиксели с яркостью . Диапазон яркостей делится на 2 класса с помощью порогового значения уровня яркости , причем может принимать значения от 0 до . Каждому из классов будут соответствовать относительные частоты (1.6, 1.7)

Теперь необходимо вычислить среднее арифметическое каждого класса (1.8,1.9).

Далее вычисляем максимальное значение(1.10) оценки качества разделения изображения на две части, где  *–* межклассовая дисперсия, а - общая дисперсия для всего изображения целиком.

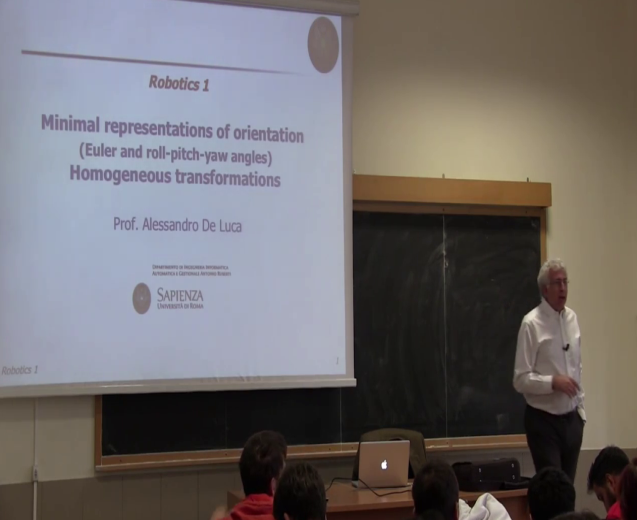
Таким образом, за счет того, что экран почти всегда является самой яркой зоной на изображении, он будет выделяться белым цветом. Но в некоторых случаях, например, если есть помеха в виде дополнительного источника света, падающего на границу экрана, граница будет размыта при бинаризации (Рис. 2).

Рис. 2 - Проблема при бинаризации методом Otsu на изображениях с несколькими источниками света

Для определения границ в таких случаях было использовано два способа.

1. Выбор обычной пороговой бинаризации.
2. Усиление линий границ экрана.

Обычная пороговая бинаризация плоха тем, что параметры пороговой обработки для разных изображений могут быть разные. Поэтому было решено усилить линии границы на изображении. Для усиления линий границ сначала необходимо их выделить. Будем использовать термин край – это резкий переход яркости на изображении. Края возникают по определенным причинам:

1. Изменяется освещенность (яркость) пикселей
2. Происходит изменение цвета у пикселей
3. Изменяется ориентация поверхности

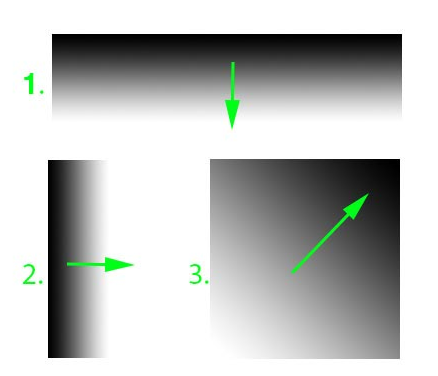
С помощью анализа первой и второй производной изображения можно выделить области с резким перепадом яркости, то есть найти края изображения. Максимальное изменение функции происходит по направлению её градиента. На рисунке 3 приведены примеры.

Рис. 3 - Изменение яркости и надлежащие им направленности градиентов

Где градиенту (1,2,3) соответствуют формулы (1.11, 1.12, 1.13). То есть по факту необходимо измерить частные производные по и .

Направления градиента вычисляются по формуле (1.14), абсолютная величина по формуле (1.15), а угол между направлениями градиента и осью X, по формуле(1.16).

Сила края задаётся абсолютной величиной градиента (1.17)

Операция получения частных производных сводится к вычислению разности пределов (1.18). Полученная функция будет обладать свойствами линейности и инвариантности к переносу. Значит, эту функцию можно будет вычислить как свертку.

Было предложено целое семейство фильтров для вычисления границ разной сложности, они были основаны на приближенном вычислении градиента, анализе его направления и абсолютной величины. Самые известные из них: фильтры (операторы) Робертса, Превитта, Собеля.

Фильтр Робертса

Фильтр Превитта

Фильтр Собеля

После серии тестов было выявлено, что фильтр Собеля лучше всего справляется с выделением краев на изображении, но на изображениях со слабыми границами он мог их не найти, поэтому было решено использовать фильтр Собеля только как один из шагов по выделению краев на изображении.

## Детектор границ Canny

Для выделения границ обычные операторы Собеля, Робертса, Превитта, недостаточно хороши. Поэтому для более точного их нахождения лучше использовать: детектор границ Canny [12]. Он состоит из 5 шагов:

1. Сглаживание и удаление шума.
2. Расчет градиентов изображения.
3. Подавление не максимумов.
4. Двойная пороговая фильтрация
5. Связывание краев в контуры.

Сглаживание шумов производится с помощью фильтра Гаусса, о котором было ранее упомянуто. Стоит лишь заметить, что сигма выбирается из отношения размер матрицы свертки. Для расчета градиентов изображения будет использоваться также ранее упомянутый оператор Собеля.

Подавление не максимумов, происходит при условии: направление градиента должно быть кратно , а также значение градиента в пикселе должно достигать локального максимума.

Двойная пороговая фильтрация необходима для определения, является ли пиксель границей (краем) изображения. Естественно, если порог маленький, то больше пикселей будет входить в границу, что приведет к зашумлению изображения. Если же порог наоборот слишком большой, то граница может получиться не целостной (Рисунок 2). Поэтому в данном детекторе используется два порога фильтрации. При значении меньшем, чем граница нижний фильтрации, пиксель не учитывается в дальнейшем – подавляется; если пиксель находится между верхней и нижней границей, он принимает среднее фиксированное значение и будет учитываться потом; если принимает значения выше верхней границы, считается подтвержденным и принимает максимальное значение.

На последнем этапе - связывание краев в контуры - происходит коррекция по пикселям, которые на предыдущем шаге попали в промежуточный интервал: если эти пиксели соприкасаются хотя бы с одним из пикселей, находящихся выше максимальной границы, они добавляются к этим пикселям. Соприкосновение учитывается по восьми направлениям.

После получения границ от детектора Canny линии остаются слишком тонкими. Необходимо воспользоваться операциями математической морфологии. К ним относятся:

1. Сужение (1.11)
2. Расширение (1.12)
3. Закрытие (1.13)
4. Раскрытие (1.14)

Где, – множества из

Где, – дополнение

На полученном изображении необходимо провести операцию расширения для увеличения линий границ экрана. Достаточно увеличить на несколько пикселей толщину границ и после этого к бинаризированному изображению необходимо добавить эти границы, предварительно инвертировав у них цвет.

В итоге полученный результат позволит избежать проблем, показанных на рисунке 2. Получив бинаризированное изображение с сильно выделенными границами, можно приступить к поиску экрана проектора. Как правило, экран проектора обладает следующими свойствами – его соотношение сторон может быть от 4:4(т.е. квадратное) до 16:9(4:3), а также площадь на изображении будет одной из наибольших. Если отсечь все непопадающие под эти параметры выделенные зоны из бинаризированого изображения, то в большинстве случаях останется только область экрана либо область, в которую будет попадать экран с лишними элементами. Для нахождения координат линий был использован метод, описанный в работе [13]. После получения списка координат (не обязательно это будет 4 координаты углов прямоугольника, так как на изображении могут быть помехи, к тому же алгоритм может отметить на контуре промежуточные точки) необходимо выделить из этого списка координаты углов. Просуммируем точки по . Тогда минимальное значение будет соответствовать верхнему левому углу, максимальное - правому нижнему. А для подсчета оставшихся координат углов нужно подсчитать разницу между x и y по всем точкам. Значение с максимальной разностью – левая нижняя точка угла, а с минимальной разностью – правая верхняя. После этого, зная все 4 координаты углов, можно выделить зону со слайдами (Рис. 4) с сохранением перспективы.

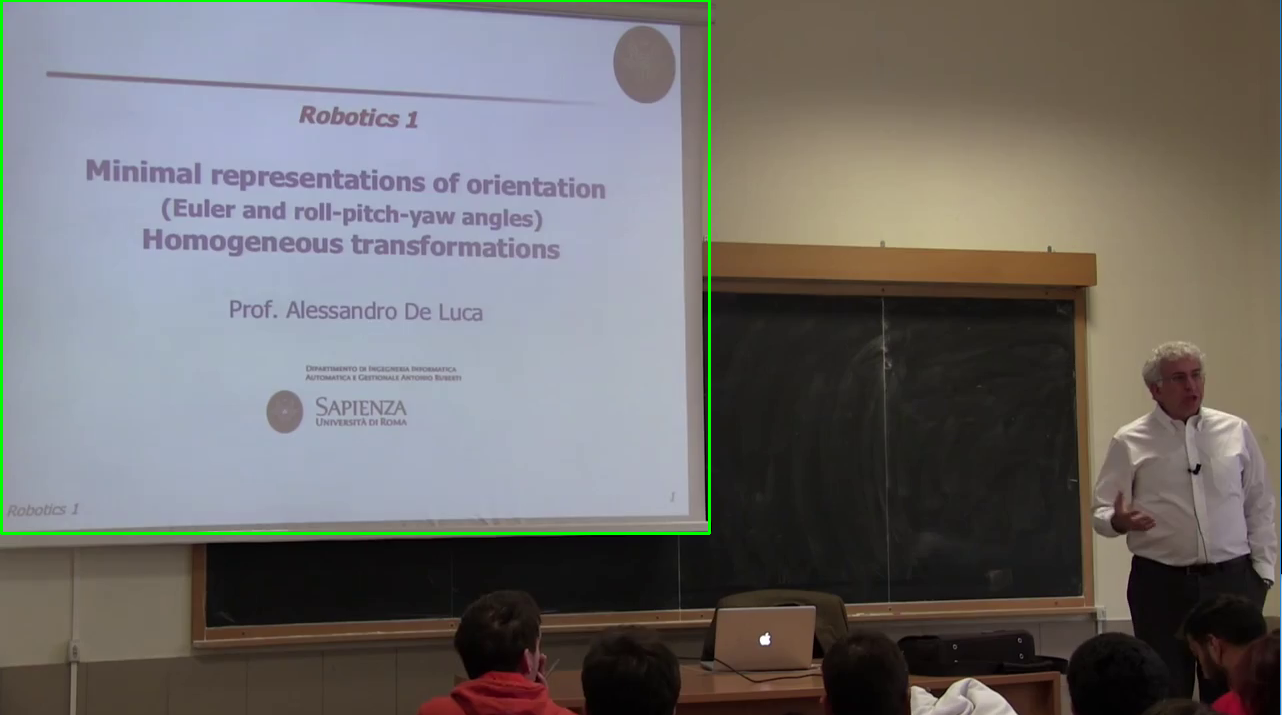


Рис. 4 - Точное нахождение экрана проектора

## Перспективное преобразование

 Перспективное преобразование - это преобразование одного трехмерного пространства в другое, при котором размер объекта уменьшается с увеличением расстояния до центра проекции, и происходит неоднородное искажение линий объекта, зависящее от ориентации и расстояния от объекта до центра проекции [14].

Применение перспективного преобразования может улучшить качество работы алгоритма перцептивный хэш, но может и ухудшить за счет появления в кадре неожиданных объектов либо из-за резкого изменения яркости. Поэтому решено было оставить перспективное преобразование в виде дополнительной настройки, которая по желанию пользователя может быть отключена.

В результате была выполнена предварительная обработка изображения, с использованием следующих операций:

1. Фильтрация изображения с помощью фильтра Гаусса.
2. Бинаризация изображения
3. Выделение и усиление линий границ на изображении
4. Объединение полученных результатов
5. Получение координат экрана на изображении
6. Перспективное преобразование

На выходе было получено новое изображение экрана проектора, с которым в дальнейшем будем работать.

# Сопоставления изображений

Что собой представляют изображения, если дать их компьютеру? Изображения для него будут являться просто набором данных: в отличие от человека он не способен сравнить их либо же найти какие-нибудь визуально похожие объекты. Поэтому одной из проблем стало обучение компьютера для решения данной задачи.

Существуют конкретные подходы для сравнения изображений, базирующиеся на сравнении данных об изображениях. Один из самых простых способов анализа изображений - сравнение гистограмм (правда, в этом случае можно только говорить о похожести двух изображений). Также можно построить хэш изображений, и при совпадении хэшей можно будет утверждать, что изображения одинаковые. Рассмотрим один из таких методов.

## Перцептивный хэш

Далее будет рассмотрен обобщенный алгоритм на основе данной статьи [15]

У изображений существует высокие и низкие частоты, и, как ранее уже обсуждалось, высокие частоты служат для детализации объектов, а низкие могут показать структуру изображения. Если изображение большое, в нем много деталей, то есть высоких частот, но, если это изображение уменьшить до маленьких размеров (например, 32x32 или 8x8 пикселей), оно полностью будет состоять из низких частот. Суть алгоритма можно описать в 6 шагов:

1. Уменьшить размер изображения до 32x32 пикселей.
2. Перевести получившееся изображение в градации серого цвета.
3. Использовать дискретное косинусное преобразование.
4. Взять из полученной матрицы (после шага 3), подматрицу (8x8) со сдвигом от левого верхнего угла изображения (1x1)
5. Вычислить среднее значение полученной подматрицы.
6. Построить Хэш по подматрице: значение 1 принимается, если значение в данном пикселе выше среднего, и 0 если ниже.

Для сравнения хэшей будет использоваться расстояние Хэмминга – количество разных бит в строках двух хэшей, если расстояние будет равняться нулю, скорее всего, изображения похожи.

Данный метод достаточно устойчив к изменениям гистограммы изображений, но в случае сравнения электронного слайда и слайда с изображения видеокамеры он показал относительно плохие результаты. Метод не защищен от появления в видео потоке лишних объектов, например, человека. Также у него существенные проблемы с анимированными слайдами и плохими углами обзора. Поэтому можно сделать вывод, что этот метод является достаточно быстрым и точным, если в видео потоке присутствуют только электронные (первый тип видео-лекций) и не анимированные слайды.

В изображениях, сфотографированных под разным углом, в разное время суток, или на видео- и фотокамеры с разной настройкой, как правило, простые способы не могут определить, одинаковые там объекты или нет. То есть, если работать с глобальными признаками изображения, достаточно сложно сопоставить слайды. Поэтому стоит воспользоваться локальными признаками и алгоритмами для них.

Для каждого пикселя на изображении будем вычислять значение функции, и по итогу таких значений получится общая функция – характеристика, которую можно сравнить с другими характеристиками. Но данный метод работает только при очень хороших условиях: если на изображении появится новый объект или шумы, изменится положение камеры или освещение на изображении - тогда любая из этих причин может привести к резкому изменению характеристик изображений. Поэтому для решения подобных проблем, было придумано сохранить подход высчитывания определенной характеристики, но брать не абсолютно все точки изображения, а выбирать особые точки. Стоит заметить, что особой точкой является точка, которая с высокой долей вероятности будет найдена на другом изображении такого же объекта. Введем понятие детектор – способ извлечения ключевых точек изображения. Он должен находить одни и те же особые точки на разных изображениях, независимо от преобразований над этими изображениями. Детектор способен только находить особые точки, но не сопоставлять их между двумя изображениями. И тут на помощь приходят дескрипторы. Дескриптор – идентификатор ключевой точки, выделяющий ее из остальной массы особых точек. В свою очередь, дескрипторы должны обеспечивать инвариантность нахождения соответствия между особыми точками относительно преобразований изображений.

Для решения задачи сопоставления изображений необходимо:

1. Выделить особые точки и их дескрипторы
2. На основании совпавших дескрипторов выделить ключевые точки.
3. По ключевым точкам построить модель преобразования изображений.

Рассмотрим алгоритмы, которые помогут выделить ключевые точки и их дескрипторы [16,17]. Также обратим внимание, что эти алгоритмы должны быть устойчивы к масштабируемости изображений, перепадам яркости и поворотам камеры. Так как камера, на которую могут снимать лекции, не всегда будет правильно настроенной и, возможно, ею кто-то будет управлять.

## SIFT (Scale Invariant Feature Transform)

Для обнаружения особых точек необходимо построить пирамиды гауссианов и разностей гауссианов. Гауссианом называется изображение, к которому применили фильтр Гаусса (2.1)

Где — значение гауссиана в точке с координатами , а — радиус размытия. — гауссово ядро, — значение исходного изображения, \* — операция свертки.

Пирамида гауссианов состоит из нескольких уровней. Каждый уровень (октава) содержит в себе несколько интервалов, состоящие в свою очередь из изображений, к которым применили фильтр Гаусса с разными

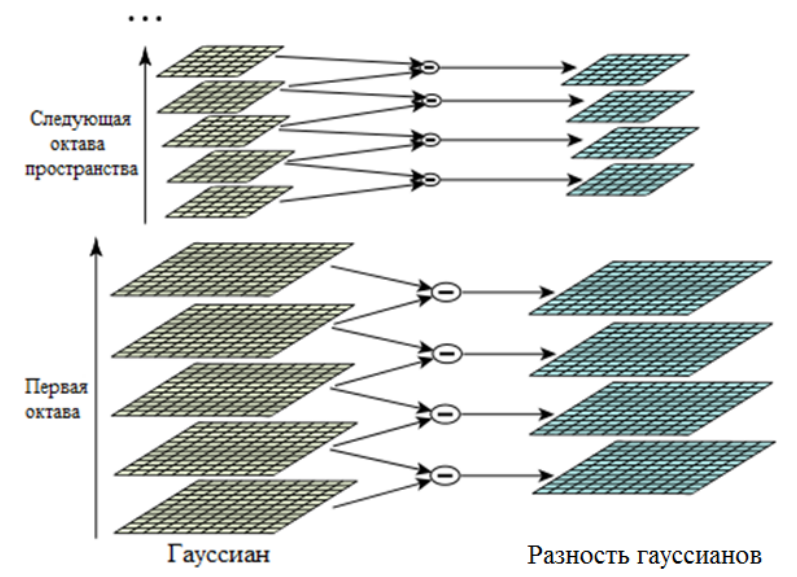


Рис. 6 - Пирамида гауссиан и их разностей.

радиусами размытия, причем каждое последующее изображение размывается с большим радиусом размытия. Если происходит переход от одной октавы к другой, то размеры первоначального изображения октавы уменьшаются вдвое, благодаря чему достигается инвариантность к масштабируемости [18].

Если попиксельно вычесть из одного гауссиана другой с различными радиусами размытия (сигма), это будет разностью гауссианов (2.2).

Здесь – шаг, который влияет на увеличение степени размытия на каждом последующем изображении.Для каждой октавы строится определенное число интервалов N, из разницы которых строится разность гауссианов. Число интервалов в разности гауссианов, естественно, будет равняться N-1 (рис. 6). Также стоит отметить, что, так как каждая октава содержит бесконечное множество гауссианов, необходимо выбрать из них N, учитывая шаг размытия. Дополнительно необходимо достроить 2 интервала к каждой октаве для обнаружения в дальнейшем экстремумов на N и N+1 в разности гаусианнов.

Для нахождения локальных максимумов и минимумов разности гауссианов каждая точка сравнивается с 8 его соседними точками и с 9 точками в соседних интервалах. Точка является локальным экстремумом, если значение разности в этой точке будет больше или же меньше, чем во всех соседних точках. После обнаружения локальных экстремумов (Рис.7) необходимо проверить, являются ли данные точки ключевыми: если точки являются ключевыми, вычисляется их ориентация, в противном случае они отбрасываются.

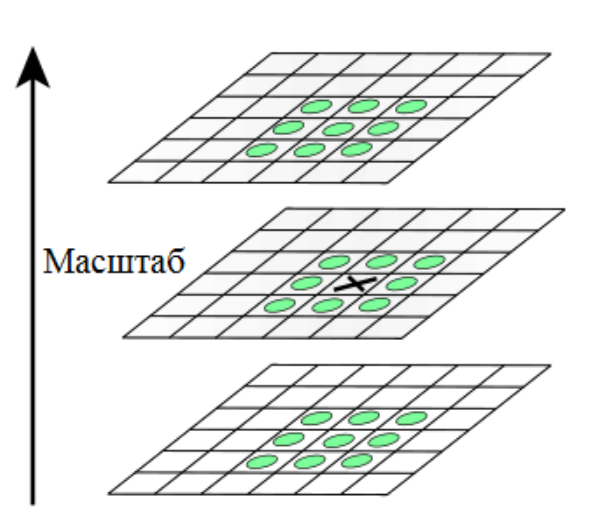


Рис. 7 -Точка будет являться локальным экстремумом, если она принимает значения больше или меньше чем все соседние (отмеченные зеленым) точки.

Направление ключевой точки вычисляется с масштабом, близким к масштабу ключевой точки, иными словами, само направление зависит от направлений градиентов соседних точек. Стоит заметить, что точка может иметь несколько направлений.

Заключительным шагом является построение дескрипторов для окрестностей точек. Благодаря этому действию добавится инвариантность к освещению и точке обзора. Дескриптор вычисляется на гауссиане, ближайшей по масштабу к ключевой точке и исходя из градиентов в некотором окне ключевой точки. Для начала окно поворачивают на угол направления ключевой точки, и после этого происходит вычисление дескриптора. Эти действия нужны для достижения инвариантности относительно поворота.

На рисунке 8 слева под маленькими белыми квадратами изображаются пиксели, эти пиксели берутся из окна дескриптора – большой квадрат, который в свою очередь поделен на 4 меньших квадрата или же региона. Для отображения градиента используются маленькие стрелки.

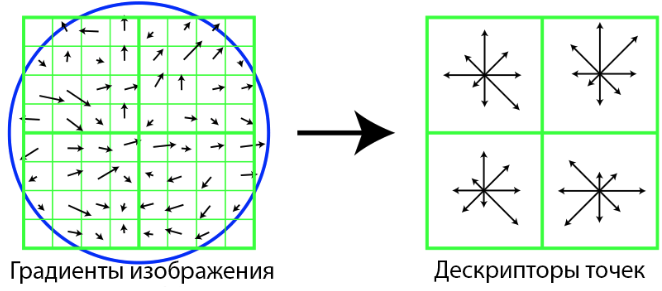


Рис. 8 - Слева располагается часть изображения, справа дескрипторы, построенные на его основе.

Центр окна нужно выбирать как можно ближе к координатам ключевой точки. Синий круг служит для обозначения окна свертки с гауссовым ядром. для данного ядра будет равняться половине ширины окна дескриптора. Каждое значение точки умножается на коэффициент равный значению ядра в этой точке. Справа отображаются дескрипторы особой точки и имеют размерность равную 32(2x2x8). 2x2 – это значение количества регионов по осям то есть по горизонтали и вертикали. Количество компонент гистограммы регионов определяет последнее значение, и, как правило, оно равняется всегда 8. На практике используются дескрипторы с вектором из другого количества компонент – 128. В итоге получается, что дескриптором является вектор, состоящий из 128(4x4x8) компонент и содержащий значения всех элементов от гистограммы направлений.

## SURF (Speeded up robust features)

Вторым по популярности после SIFT алгоритма является SURF. Его преимущество заключается в скорости работы, но, если учитывать все дескрипторы, он проигрывается по качеству.

Отличия SURF от SIFT заключается в нескольких моментах:

1. Особые точки ищутся с помощью аппроксимации матрицы Гессе (2.3/2.4).

Где – матрица Гессе, а – функция изменения градиента яркости.

1. Дескрипторы описываются с помощью вейвлетов Хаара, и высчитываются отклики по (горизонтально и вертикально направленные) в пределах окрестности особой точки. Полученные результаты взвешиваются Гауссином в особой точке и суммируются по каждому региону.

По проведенным экспериментам метод SURF выполняется в среднем в 2 раза быстрее, чем SIFT, по качеству же результаты отличаются в особо редких случаях, поэтому было решено оставить оба алгоритма и предоставить выбор алгоритма пользователю. В данной работе метод SIFT и SURF будут использоваться для нескольких целей:

1. Сравнение соседних кадров
2. Сравнение кадров со слайдами–эталонами (изображениями из электронной презентаций)

Если сравнивать соседние кадры из видео потока, то можно будет определить, на каких кадрах предположительно произошло изменение слайдов либо случилось событие, при котором экран пропал с изображения.

Во втором же случае, сравнение будет происходить для определения, какой слайд в данный момент транслируется на экран. Сразу стоит заметить, если изображение слайда будет слишком плохого качества – низкого разрешения либо же сильно засвеченным, это приведет если не к ошибочному определению слайда, то к низкому количеству нахождения похожих особых точек между электронным слайдом и изображением, что в свою очередь сделает невозможным точное определение текущего слайда.

## Оптическое распознавание символов

Дополнительным модулем для распознавания текущего слайда будет использоваться оптическая система распознавания с открытым исходным кодом Tesseract. На текущий момент Tesseract является одним из самых лучших бесплатных решений с хорошим качеством распознавания. Естественно, качество распознавания будет полностью зависеть от качества слайдов изображения: при плохом качестве неудивительно получить на выходе из OCR ошибки в словах. Для коррекции таких ошибок будет использоваться расстояние Левенштейна. После получения из электронных слайдов текстовой информации можно будет составить словарь, который будет служить для проверки слов расстоянием Левенштейна [19]. Математически расстояние Левенштейна между двумя строками и - длиной и соответственно - можно вычислить по формуле (2.5).

# Реализация программного продукта

Для написания программного продукта было решено использовать язык программирования Python 3.5 x64, а также внешние библиотеки:

* OpenCV 3.1 – реализация большого количества алгоритмов в области компьютерного зрения.
* Numpy 1.9.2 – ускорение математических вычислений
* Matplotlib 1.5.0rc2 – рисование графиков и изображений
* Pyteseseract 0.1.6 – обертка OCR Tesseract
* PythonMagick 0.9.12 – разбиение PDF и PPTX файлов на отдельные изображения
* PyQt5 5.6 – графический интерфейс
* PDFMiner 20160202 – извлечение текста из PDF файлов
* Python-pptx 0.5.8 – извлечение текста из PPTX файлов

Выбор данного языка программирования и библиотеки OpenCV был обусловлен следующими факторами:

1. Python удобен для академических исследований
2. Python имеет большое количество вспомогательных библиотек
3. OpenCV имеет свободный доступ для использования, а также имеет реализацию большого количества алгоритмов из области компьютерного зрения. В данной работе использовались алгоритмы SIFT, SURF, бинаризация методом Otsu, фильтр Гаусса.

Была написана программа – прототип (Рис. 9) с минимальным интерфейсом, в котором можно выбрать тип видео-лекции, метод бинаризации, а также метод, по которому будет происходить сравнение слайдов. Стоит отметить, что сравнение происходит не для каждого кадра, а

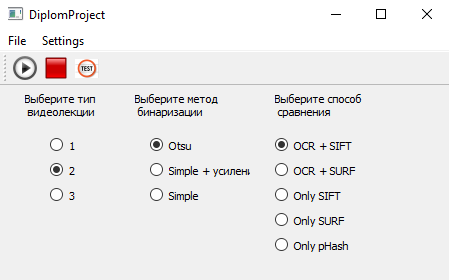


Рис. 9 - Интерфейс программы-прототипа

для каждого 30 по умолчанию (в настройках можно увеличить или уменьшить это число). Сравнивать абсолютно каждый кадр достаточно затратно по ресурсам: если при сравнении каждого 30 кадра для 90-минутного видео методом SIFT потребуется около 90 минут, то при сравнении каждого 5 кадра время увеличится пропорционально и будет равняться 9 часам. В программу можно загрузить на выбор пользователя pdf или pptx презентацию, в любом случае из презентации будут постранично извлечены все слова для возможной индексации в будущем, а также создадутся отдельные изображения каждого слайда для возможного сравнения со слайдами из видео. Из возможных форматов загрузки видео файлов – avi, mp4.

Из доступных метод бинаризации присутствует:

1. Метод Otsu
2. Обычная бинаризация + усиление границ
3. Обычная бинаризация с возможностью самому задать порог обработки.

Из способов сравнения слайдов изображений с электронными слайдами используются:

1. SIFT
2. SURF
3. SIFT + OCR(Tesseract)
4. SURF + OCR(Tesseract)
5. Перцептивный Хэш

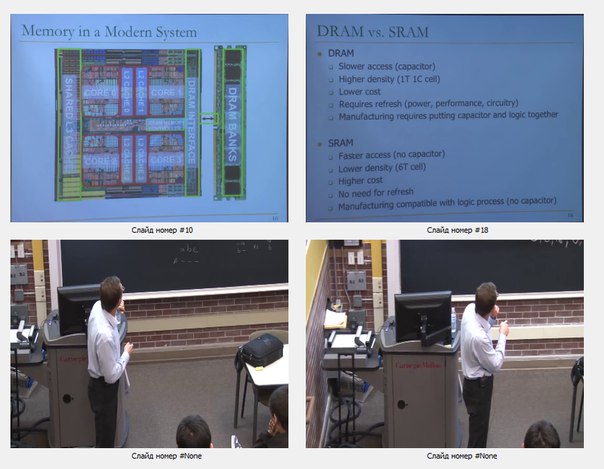
Так как это прототип программы, было решено в независимости от выбранного типа видео-лекции дать возможность пользователю выбрать любой метод сравнения изображений. Конечно же, при выборе определенного типа видео-лекций всплывет подсказка, предлагающая наилучшие настройки.

Рис. 10 - Результат выполнения тестового запуска

Также имеется возможность, позволяющая при текущих выбранных настройках запустить тестовую проверку (Рис. 10): из видео будут выбраны 4 случайных кадра и проанализированы. Пользователю будет представлена информация, изображены ли на кадрах слайды презентации и если изображены то их номера. Дополнительно в настройках можно указать, использовать или нет перспективное преобразование. При запуске программы не в тестовом режиме в любой момент можно остановить выполнение программы и получить промежуточные результаты. Конечными результатами являются данные: когда и какой слайд присутствовал на видео и (при включенной настройке) список слов для каждого слайда.

Программный код занимает достаточно места и не представляется возможным полностью включить его в приложения. Поэтому предоставляется ссылка на репозиторий с исходным кодом: https://github.com/ymmi/Diplom

## Тестирование программного продукта

Конфигурация системы, на которой проводились тесты, выглядит следующим образом:

* Процессор - Intel Core i7 5820k
* Оперативная память (RAM) - 32 gb DDR4
* Твердотельный накопитель (SSD) - 550/510 mb/sec I/O
* Система – Windows 10 x64
* Расчеты на видеокарте (GPU) не производились.

Для тестирования продукта было выбрано 5 видео-лекций с доступными электронными презентациями с видео-хостинга YouTube. Этот способ был выбран по причине отсутствия наборов данных для данной задачи. Сравнение производилось с помощью экспертной оценки, чем объясняется такое малое количество данных. Также от сотрудников УСИТ по направлению юриспруденция Санкт-Петербургского университета был предоставлен доступ к публичным видео-лекциям и проведены эксперименты на их коллекции. Стоит отметить, что эти видео-лекции имели относительно низкое качество и по классификации были второго типа, что привело к достаточно плохим результатам, программа порой даже не могла найти экран изображения, не говоря уже о правильном сравнении.

В таблице 1 представлены полученные результаты, и стоит отметить, что метод SIFT + OCR показал достаточно высокие результаты для видео-лекций типа 2. Благодаря работе двух алгоритмов можно легко проводить корректировку в правильности выбора слайдов, к недостаткам же стоит отнести проблемы с распознаванием при отсутствии текста на слайдах. С видео-лекциями типа 1, как и предполагалось, перцептивный хэш справляется хорошо, ошибки по большой части были за счет сильно схожих слайдов по структуре или анимированных.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип видео-лекции | Способ сравнения | Кол-во слайдов | Правильно распознанные слайды | Неправильно распознанные слайды | Точность |
| 2 | SIFT + OCR | 56 | 51 | 5 | 91,07% |
| 1 | Перцептивный хэш | 37 | 32 | 5 | 86,49% |
| 3 | SIFT/SURF | 13 | 4 | 9 | 30,77% |
| 1 | SIFT/SURF | 37 | 33 | 4 | 89,19% |
| 2 | Перцептивный хэш | 42 | 11 | 31 | 26,19% |

Табл. 1 - Результаты тестирования разных алгоритмов

Для видео-лекций типа 2 перцептивный хэш показал плохие результаты, для улучшения качества требуются дополнительные операции – гомографическое преобразование с сохранением абсолютно точно всех пропорций изображения, а также очень точным определением границ экрана презентации на видео-лекции.

Алгоритмы SIFT/SURF без применения OCR выдали весьма плохие результаты на видео-лекциях типа 2 и 3, и это можно объяснить тем, что алгоритмы не инвариантны к цветовым изменениями изображений совместно с шумовыми. Одним из вариантов возможного решения данной проблемы – цветокоррекция на основе электронных слайдов.

С целью определения зависимости производительности от количества потоков, была произведена модификация программы (вычисления производились параллельно в несколько потоков). Прироста в производительности при обработке видео кадров получено не было, зачастую даже время выполнения операций увеличивалось, также росли и затраты на обработку: требовалось хранить видео в RAM памяти компьютера, что в среднем для 90-минутного видео требовало около 20gb RAM памяти компьютера. При использовании распараллеливания на алгоритмах сравнения изображений, конкретно на алгоритме SIFT, удалось получить прирост по скорости сравнения в 2-3 раза. При использовании же с алгоритмом SURF прироста получено не было. Результаты представлены в таблице 2. Алгоритмы SIFT и SURF сравнивали одно изображение с 56 слайдами. Для фильтрации и бинаризации было использовано 5 минутное видео. Все результаты значения среднего арифметического по 10 тестам.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | SIFT(1x56) | SURF(1x56) | Бинаризация(5 мин) | Фильтрация(5 мин) |
| 1 поток | 21 | 10,4 | 22,3 | 26,1 |
| 8 поток | 10,4 | 11,3 | 22,4 | 27,6 |
| 6 мульт. проц. | 22,7 | 18 | 31,9 | 32,3 |

Табл. 2 - Результаты распараллеливания, значения в секундах

# Выводы

Полученные результаты тестирования продемонстрировали правильность подхода к решению поставленной задачи. Однако  
стоит отметить, что прототип не является полноценной системой, он справляется далеко не со всеми видео-лекциями и имеет ряд недостатков. К одному из таких недостатков стоит отнести нахождение экрана проектора. Не всегда экран проектора будет белого цвета, или занимать наибольшую площадь на изображении. Для решения такой задачи наилучшим решением будет реализация классификатора, который сможет определить, присутствует на изображении экран проектора или нет.

При распараллеливании вычислений значительный результат был получен только при работе с SIFT алгоритмом. Производительность по времени удалось улучшить в 2 раза. Во всех остальных случаях, особого прироста по времени не было получено.

# Заключение

Во время исследования были решены следующие подзадачи:

Проведен анализ предметной области, были рассмотрены ранние исследования, на основе которых было решено обратить внимание на метод сопоставления изображений SIFT и OCR системы. А также были классифицированы видео-лекции на три типа.

На основе анализа предметной области были сформулированы требования к программному продукту, рассмотрены варианты поведения программы при определенных типах видео-лекций, а также приведена общая концепция работы прототипа.

Были рассмотрены различные методы обработки изображений: фильтрация, бинаризация, сопоставление. Из методов фильтрации подробно описаны фильтры – усреднение и Гаусса. Рассмотрены методы адаптивной и пороговой бинаризации и на основании экспериментов предпочтение было отдано пороговой бинаризации. Описаны методы поиска краев на изображении на основе фильтров Робертса, Превитта и Собеля. На основе последнего был выбран детектор границ Canny. Из методов сопоставления изображений представлены SIFT, SURF, перцептивный хэш, на основе которых были выбраны способы сопоставлений изображений для определенных типов видео-лекций.

Разработан прототип на языке программирования Python совместно с библиотекой OpenCV, был реализован минимальный графический интерфейс для удобства работы.

Проведено тестирование, на основании которого можно сделать вывод о правильности подхода к решению поставленной задачи. Также произведено тестирование при распараллеливании вычислений, значительные результаты были получены только для SIFT алгоритма.

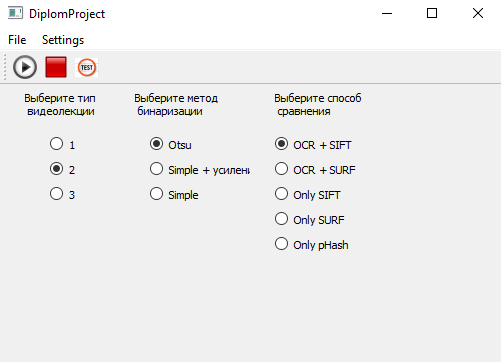
Таким образом, основная задача, поставленная в диссертации, а именно написание системы анализа видеоизображения с целью извлечения из него информации о видео-лекциях для возможной их синхронизации с электронными слайдами, а также для создания навигации по видео файлу, была полностью решена. Разработанный прототип и его исходный код могут свободно использоваться для обработки видео и для разработки более качественного программного обеспечения.

**Список литературы**

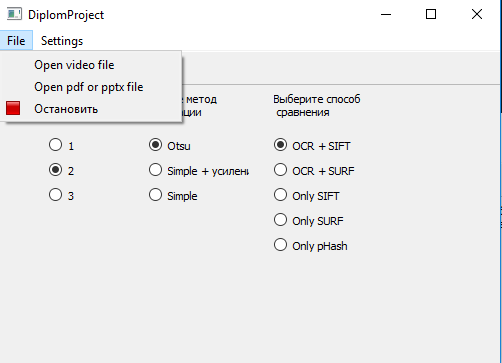
1. G. D. Abowd, C. G. Atkeson, A. Feinstein, C. E. Hmelo, R. Kooper, S. Long, N. N. Sawhney, and M. Tani. Teaching and learning as multimedia authoring: The classroom 2000 project. In ACM Multimedia, pages 187–198, 1996.
2. C. Zhang, J. Crawford, Y. Rui, and L. He, “An automated end-to-end lecture capturing and broadcasting system,” in ACM Multimedia, 2005, pp. 808–809.
3. B. Erol, J. J. Hull, and D. Lee. Linking multimedia presentations with their symbolic source documents: algorithm and applications. In ACM Multimedia, pages 498–507, 2003.
4. F. Wang, C.-W. Ngo, and T.-C. Pong. Synchronization of lecture videos and electronic slides by video text analysis. In ACM Multimedia, pages 315–318, 2003.
5. B. Erol, J. J. Hull, and D. Lee, “Linking multimedia presentations with their symbolic source documents: algorithm and applications.” in ACM Multimedia, 2003, pp. 498–507.
6. F. Wang, C.-W. Ngo, and T.-C. Pong, “Synchronization of lecture videos and electronic slides by video text analysis.” in ACM Multimedia, 2003, pp. 315–318.
7. Q. Fan, K. Barnard, A. Amir, and A. Efrat. Robust spatiotemporal matching of electronic slides to presentation videos. IEEE Transactions on Image Processing, 20:2315–2328, 2011.
8. Frisch M. J., Frisch A. E., Foresman J. B. Gaussian 94 User's Reference. – Gaussian, 1996.
9. J. Sauvola and M. Pietikäinen, “Adaptive document image binarization,” Pattern Recognit., vol. 33, no. 2, pp. 225–236, Feb. 2000.
10. Niblack, "An Introduction to Digital Image Processing", pages 115-1 16. Englewood Cliffs, N.J.: Prenlice Hall, 1986.
11. N. Otsu (1979). «A threshold selection method from gray-level histograms». *IEEE Trans. Sys., Man., Cyber.* **9**: 62-66.
12. Canny J. A computational approach to edge detection //Pattern Analysis and Machine Intelligence, IEEE Transactions on. – 1986. – №. 6. – С. 679-698.
13. Suzuki, S. and Abe, K., Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, pp 32-46 (1985)
14. Роджерс Д., Адамс Дж. Математические основы машинной графики. М.: Мир, 2001. 604 с.
15. https://habrahabr.ru/post/237307/
16. Surf: Speeded up robust features [H Bay](https://scholar.google.ru/citations?user=qjrp4S0AAAAJ&hl=ru&oi=sra), [T Tuytelaars](https://scholar.google.ru/citations?user=EuFF9kUAAAAJ&hl=ru&oi=sra), [L Van Gool](https://scholar.google.ru/citations?user=TwMib_QAAAAJ&hl=ru&oi=sra) - Computer vision–ECCV 2006, 2006 - Springer
17. David G. Lowe «Distinctive image features from scale-invariant keypoints»
18. Brown, M. and Lowe, D.G. 2002. Invariant features from interest point groups. In British Machine Vision Conference, Cardiff, Wales, pp. 656-665.
19. В. И. Левенштейн. Двоичные коды с исправлением выпадений, вставок и замещений символов. Доклады Академий Наук СССР, 1965. 163.4:845-848.

**Приложение 1.   
Сценарий использования программы**

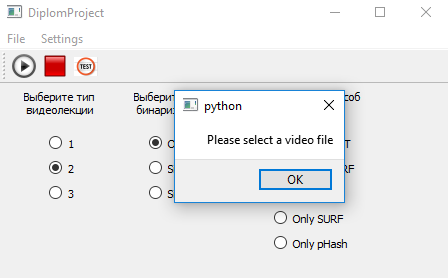
После запуска приложения открывается окно программы.



В меню File необходимо выбрать видео файл (avi или mp4 ) и презентацию (pdf или pptx). Операция остановить прекращает все действия программы, возвращая промежуточные значения.

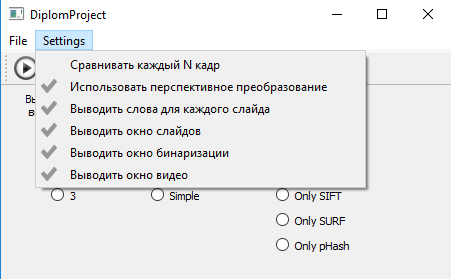


Если видео файл или файл презентации не будут выбраны, то появится окно с предупреждением.

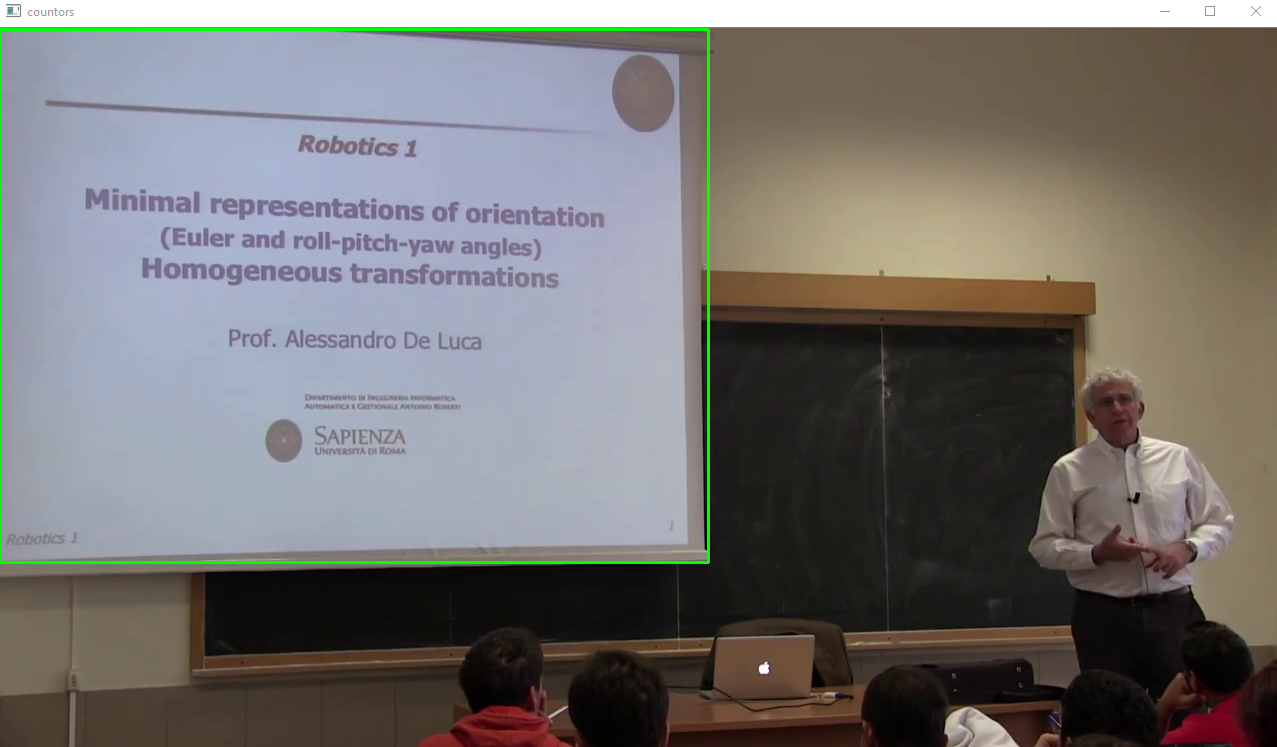
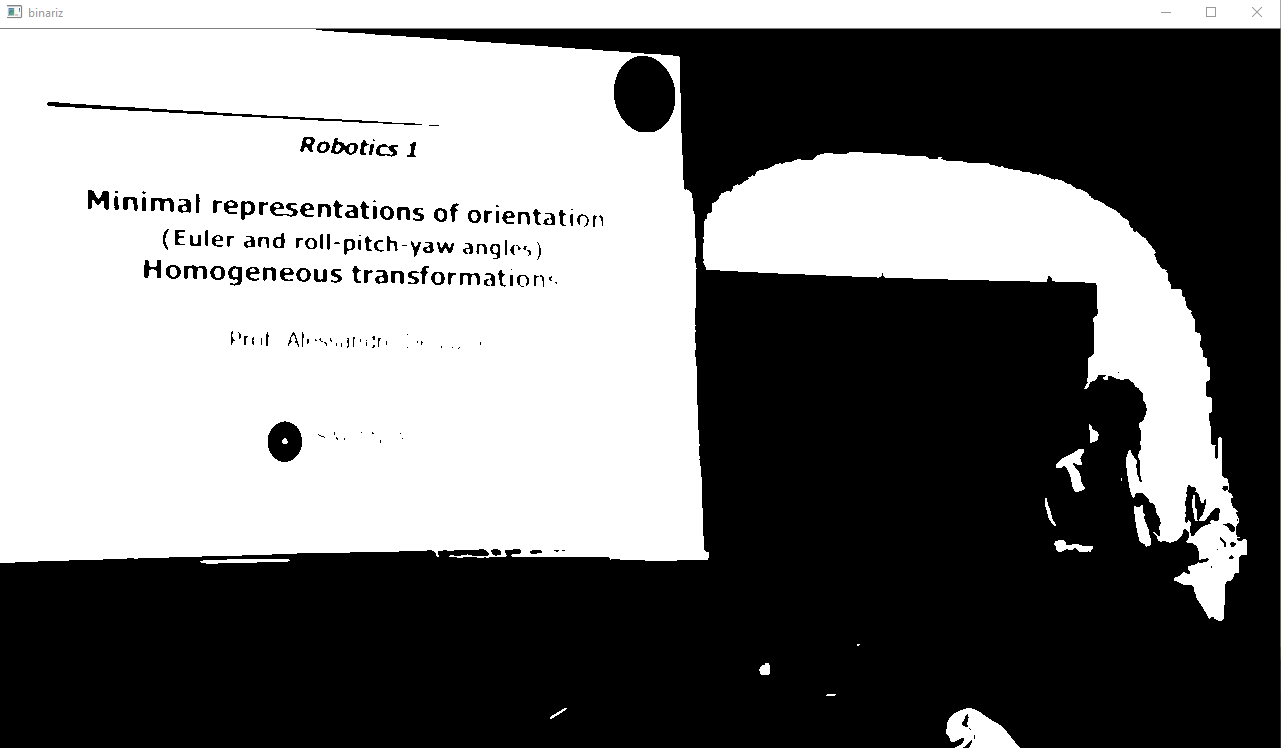


Пользователь может выбрать тип видео-лекции, метод бинаризации или способ сравнения.

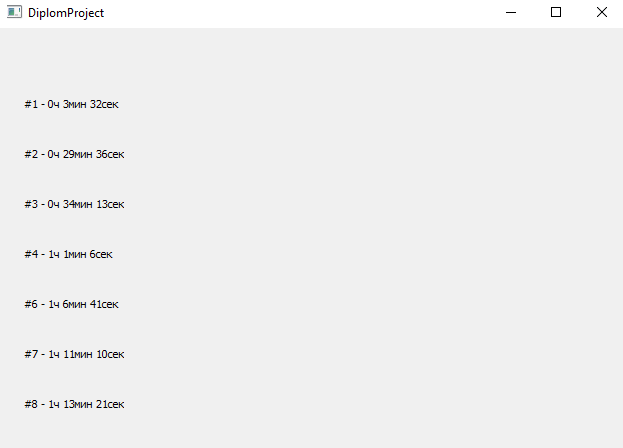
В меню Settings также можно выбрать дополнительные настройки.



При нажатии кнопки старт начнется обработка видео и презентации. При включенных дополнительных настройках появятся окна видео, бинаризации, слайда.

По окончанию работы программа предоставит пользователю данные о переключении слайдов и, если была выбрана настройка вывода текста, появится список слов для каждого слайда. Также результат выведется в файл output.txt.



# Приложение 2. Образцы видео, используемых для тестирования программы

1. https://www.youtube.com/watch?v=JBdfZ5i21cs
2. https://www.youtube.com/watch?v=L0tNWY15Y6w
3. https://www.youtube.com/watch?v=Ih0cPR745fM
4. https://www.youtube.com/watch?v=fVJeJMWZcq8
5. https://www.youtube.com/watch?v=U8K13P6loyk