МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ Государственное автономНОЕ образовательное

учреждение высшего образования

«Новосибирский НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ государственный университет» (нОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

Факультет **ФИЗИЧЕСКИЙ**

Кафедра Автоматизации физико-технических исследований

1. Направление подготовки **03.03.02 ФИЗИКА**
2. Образовательная программа: **БАКАЛАВРИАТ**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

Мустафин Владимир Михайлович

Тема работы:

“Разработка алгоритма поиска специфических сигнатур и их переходов в видеопотоке”

|  |  |
| --- | --- |
| **«К защите допущена»** |  |
| Заведующий кафедрой: Лысаков Константин Федорович | **Научный руководитель Таранцев Игорь Геннадьевич** |
| ученая степень, звание: | ученая степень, звание |
| должность, место работы: | должность, место работы |
| ………………/…………..  (фамилия И., О.) / (подпись, МП) | ………………/…………..  (фамилия И., О.) / (подпись, МП) |
| «……»………………20…г. | «……»………………20…г. |

Дата защиты: «……»………………20…г.

Новосибирск, 2021

Table of Contents

[Введение 3](#_Toc73010384)

[Обзор предметной области 3](#_Toc73010385)

[Постановка задачи 3](#_Toc73010386)

[Описание проблемы 3](#_Toc73010387)

[Разработка алгоритма 6](#_Toc73010388)

[Введение 6](#_Toc73010389)

[Разбиение ролика на сегменты 6](#_Toc73010390)

[Анализ входящего видеопотока 8](#_Toc73010391)

[Поиск очередного кадра по уже найденным сегментам. 8](#_Toc73010392)

[Работа с сигнатурами внутри окна 11](#_Toc73010393)

[Работа с VideoPart в рамках SearchInfo. 13](#_Toc73010394)

[Работа с найденными VideoPart 14](#_Toc73010395)

[Вывод 17](#_Toc73010396)

[Список литературы 17](#_Toc73010397)

# Введение

Текущая работа посвящена проблеме мониторинга информации, которая проходит по каналу прямого вещания. Такие каналы можно встретить как на телевидении, так и на просторах интернета. Однако вопрос контроля показываемого изображения до сих пор остается актуальным. Особенно актуальна это задача в сфере рекламного бизнеса. Контроль такого типа поможет ответить на различные вопросы рекламодателей, например: «Была ли показана моя реклама?», «Была ли показана моя реклама в полном объеме?», «В какое время была показана реклама?», и так далее.

Для решения задач такого типа, необходимо сравнивать изображения из видеопотока прямого вещания с набором заранее заданных видеороликов. При применении метода прямого сравнения изображений (попиксельное сравнение), производительности современных компьютеров достаточно примерно для . Это означает, что за один кадр входящего видеопотока, возможно произвести поиск по видеороликам суммарной длинной не более 6 - ти секунд. В рамках данной дипломной работы, необходимо обрабатывать базу размером 1000 видеороликов и  кадров в реальном времени.

Существует возможность построить некоторый слепок изображения – его сигнатуру. При этом схожесть сигнатур коррелирует со схожестью изображений. Использование сигнатур позволяет сократить размер хранимых данных, а также уменьшить время сравнения изображений. Существует исследование[], показывающее достаточную точность сравнения сигнатур для сравнения изображений. Производительности современных компьютеров достаточно примерно для выполнения сигнатур, размером 128 слов. Соответственно, для поиска по базе данных сигнатур необходимо увеличение производительности минимум на 2 порядка.

Известно, что в видеороликах соседние кадры похожи друг на друга. Используя эту избыточность данных, можно попытаться уменьшить количество сравнений сигнатур.

Целью текущей дипломной работы является уменьшение количества сравнений, за счет анализа видеороликов и выделения в них ключевых сигнатур. Количество сравнений должно быть достаточным для работы с потоком видео реального времени и обработки базы с характерным размером  сигнатур.

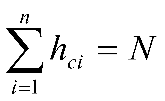
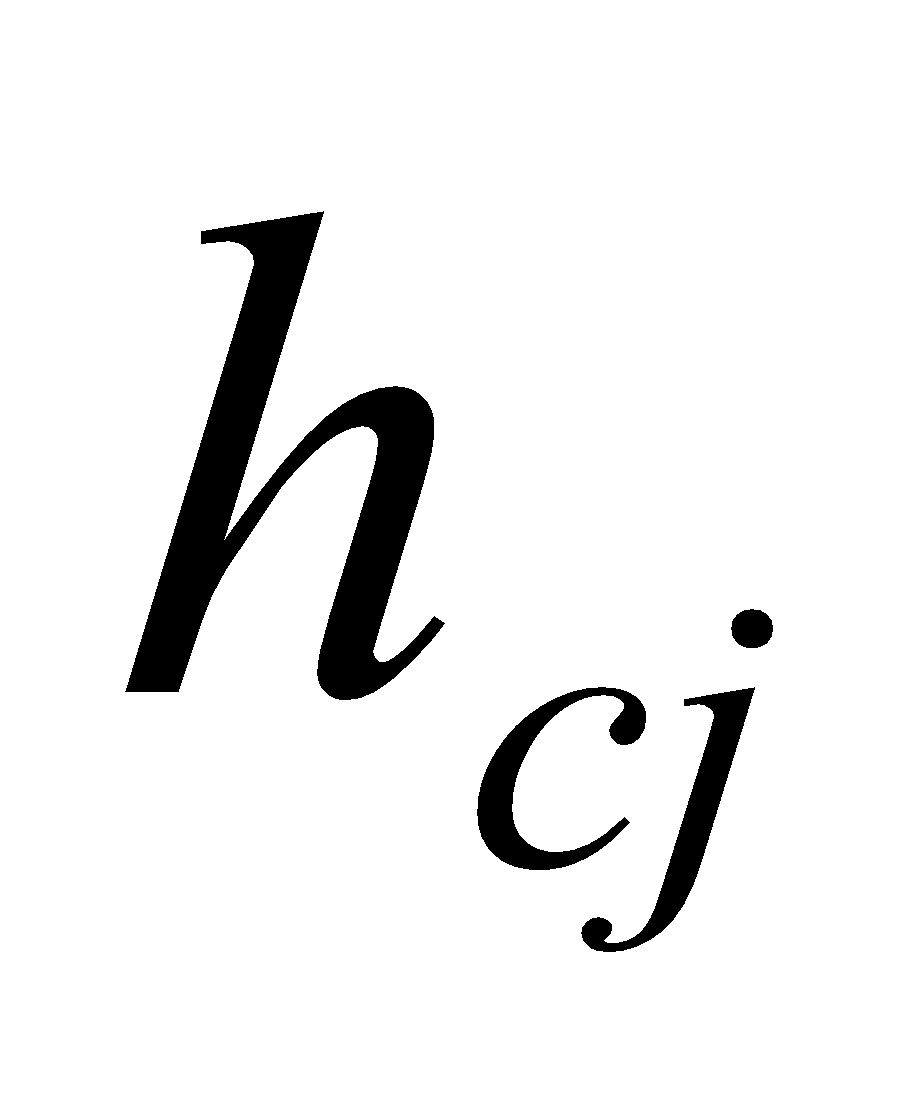
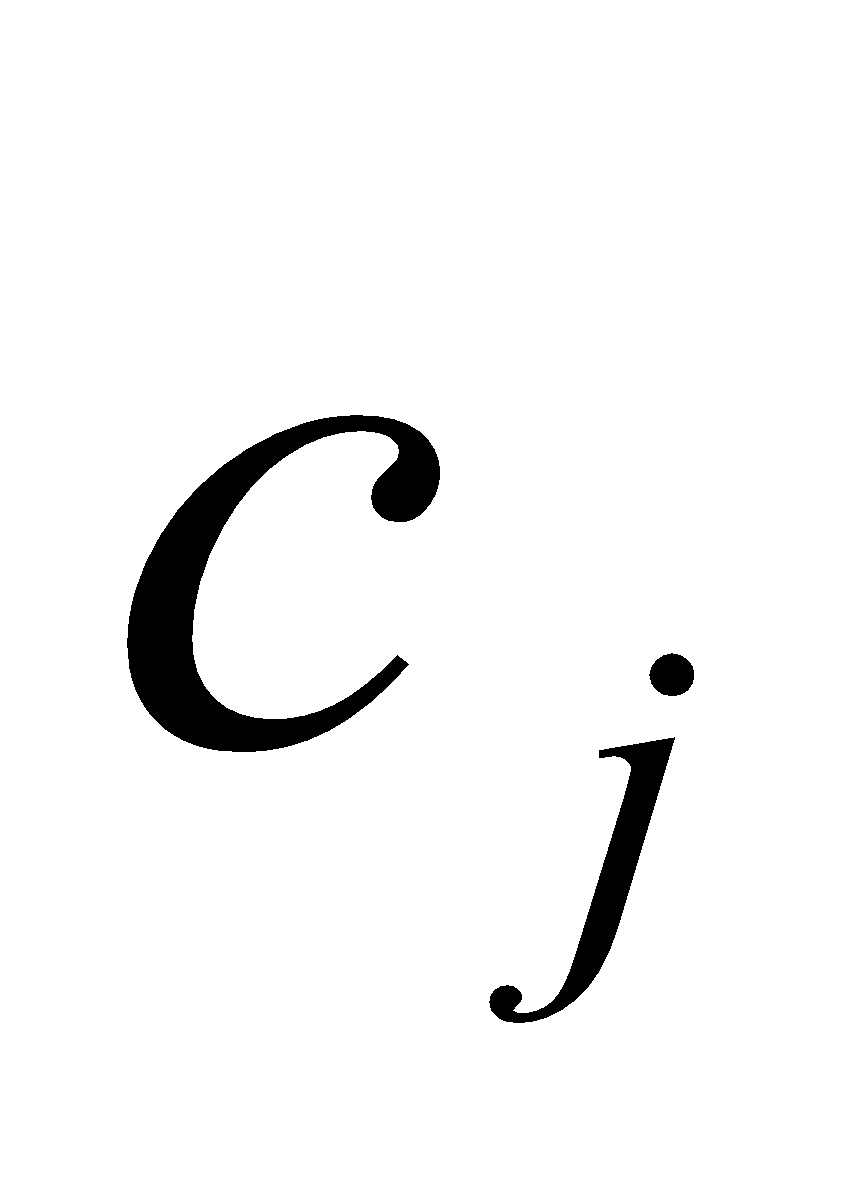
На основе поставленной цели, возникают следующие задачи:

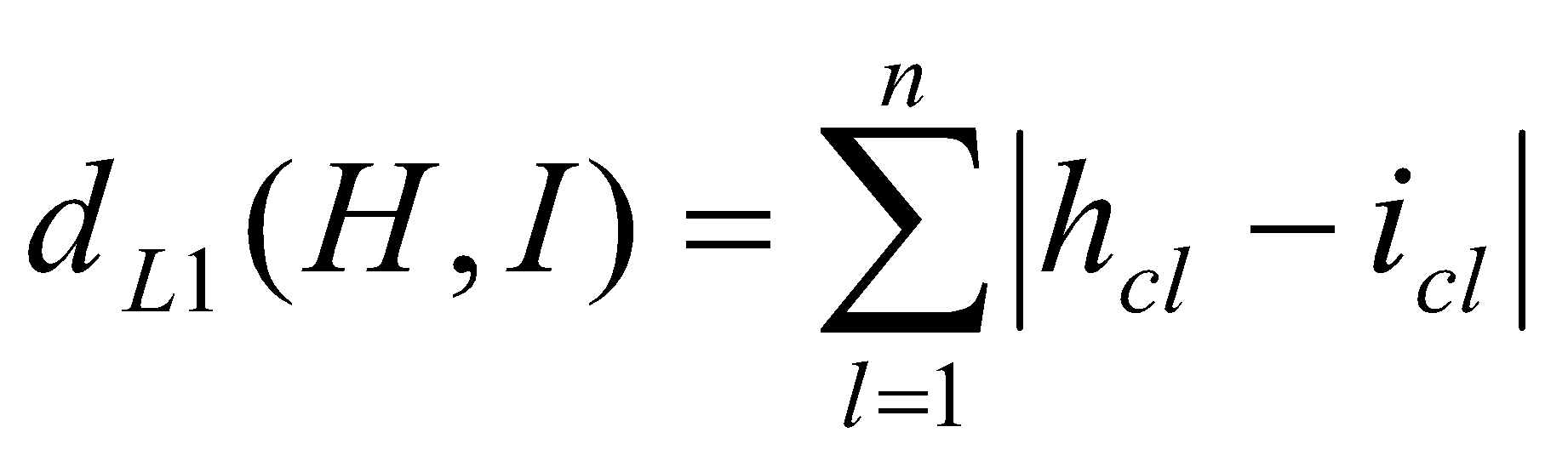
* Провести анализ предметной области
* Разработать алгоритм выделения ключевых сигнатур
* Разработать алгоритм поиска видеороликов по ключевым сигнатурам
* Реализовать и протестировать разработанные алгоритмы

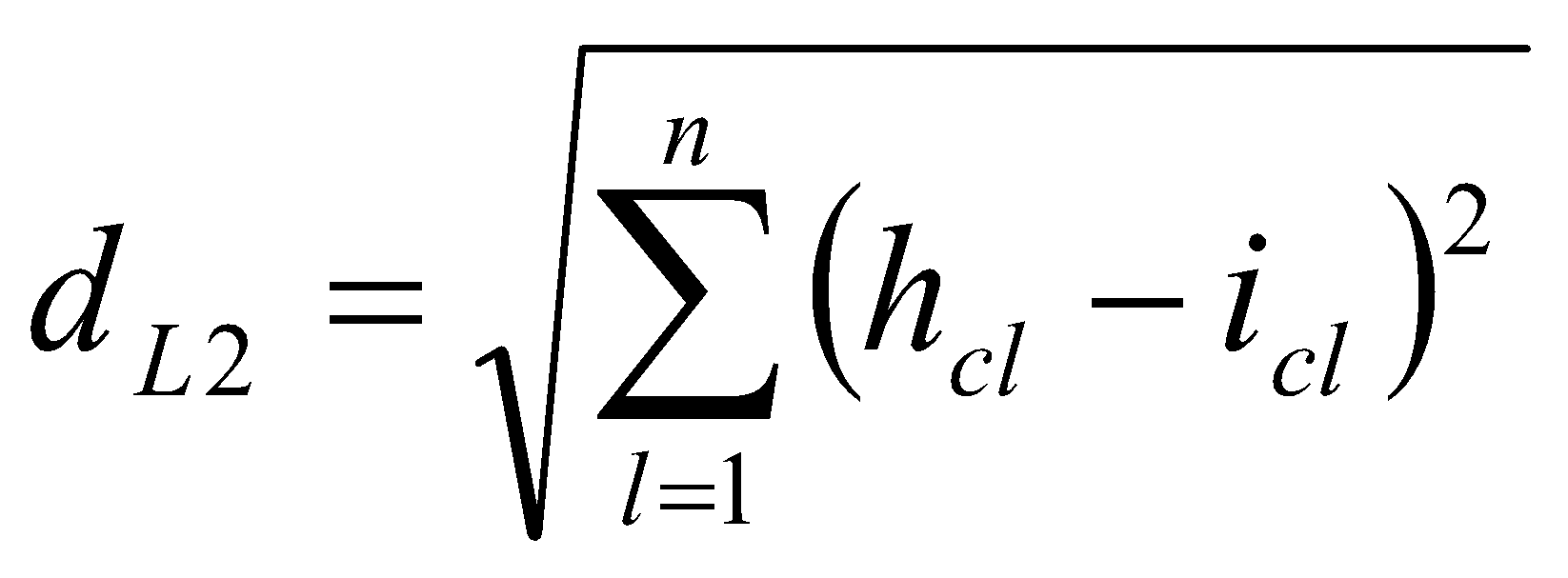
# Обзор предметной области

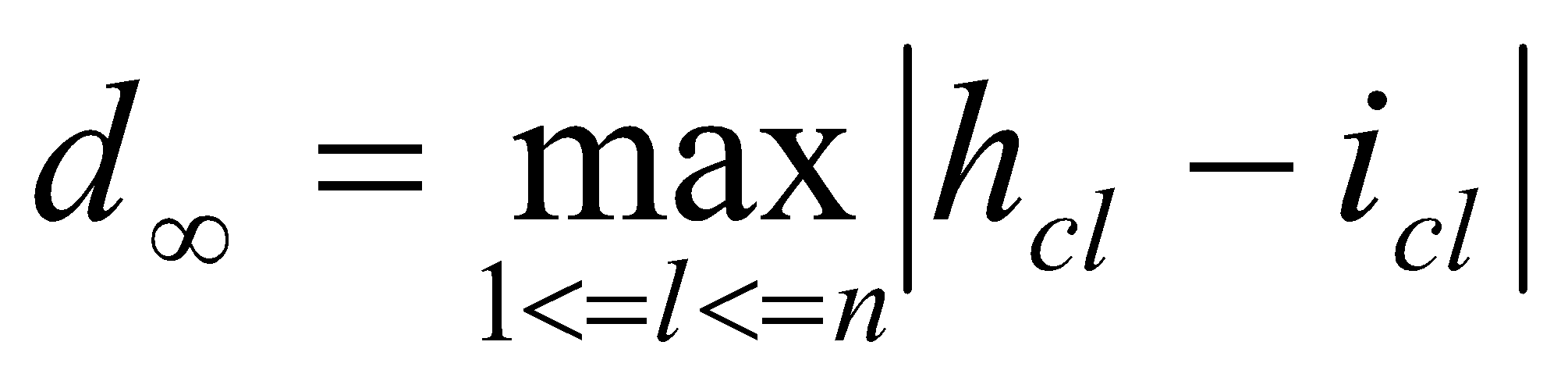
## Вычисление сигнатур

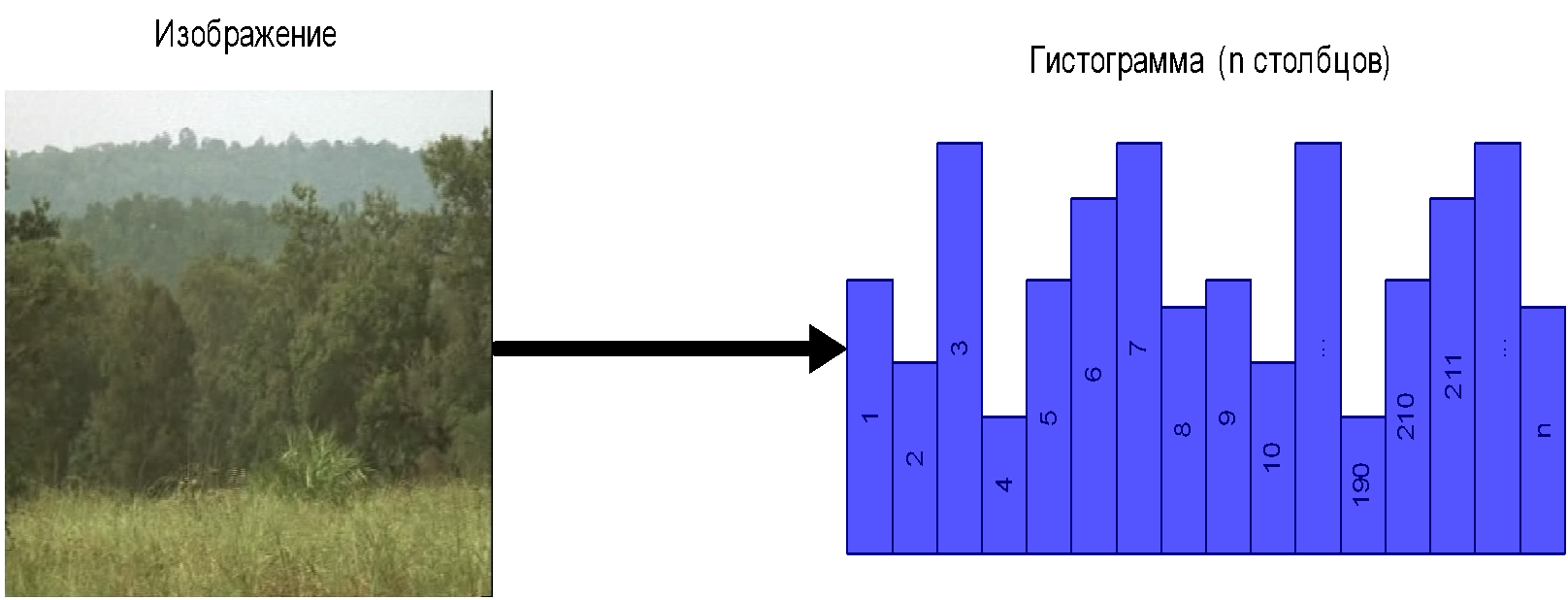
## Использование цветовой и яркостной гистограмм в качестве сигнатур

Использование яркостной и цветовой гистограмм, для обработки, а также подписи изображений является одним из самых старых и простых методов []. Так же, построенную яркостную гистограмму можно использовать в качестве сигнатуры изображения. Алгоритм состоит в том, чтобы отобразить цвета изображения *M* в дискретное цветовое пространство, содержащее *n* цветов, тогда цветовая гистограмма *H(M)* – это вектор ,где  является числом точек изображения *М* с цветом . Если *N* число точек на изображении *M*, тогда . Для определения расстояния между гистограммами *Н* и *I* используются различные функции:

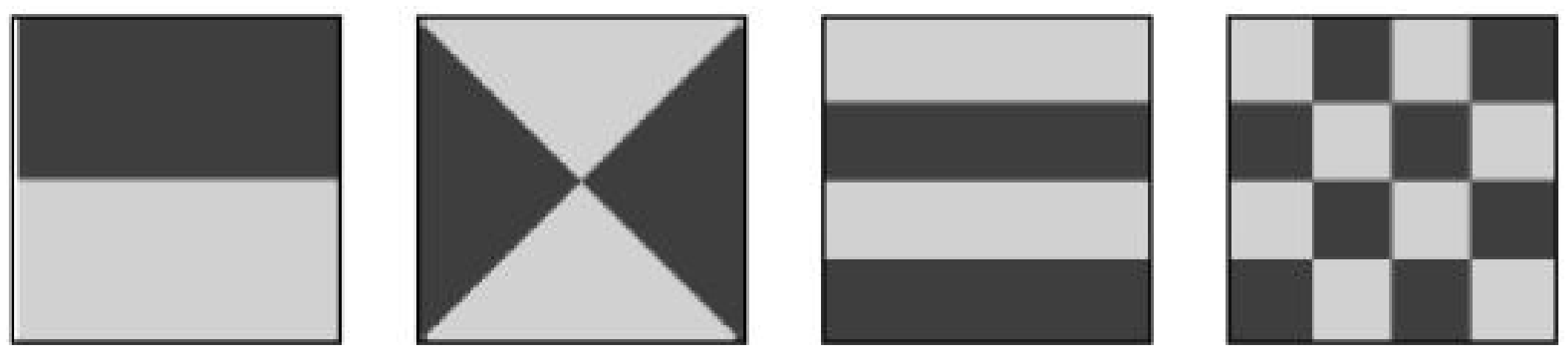






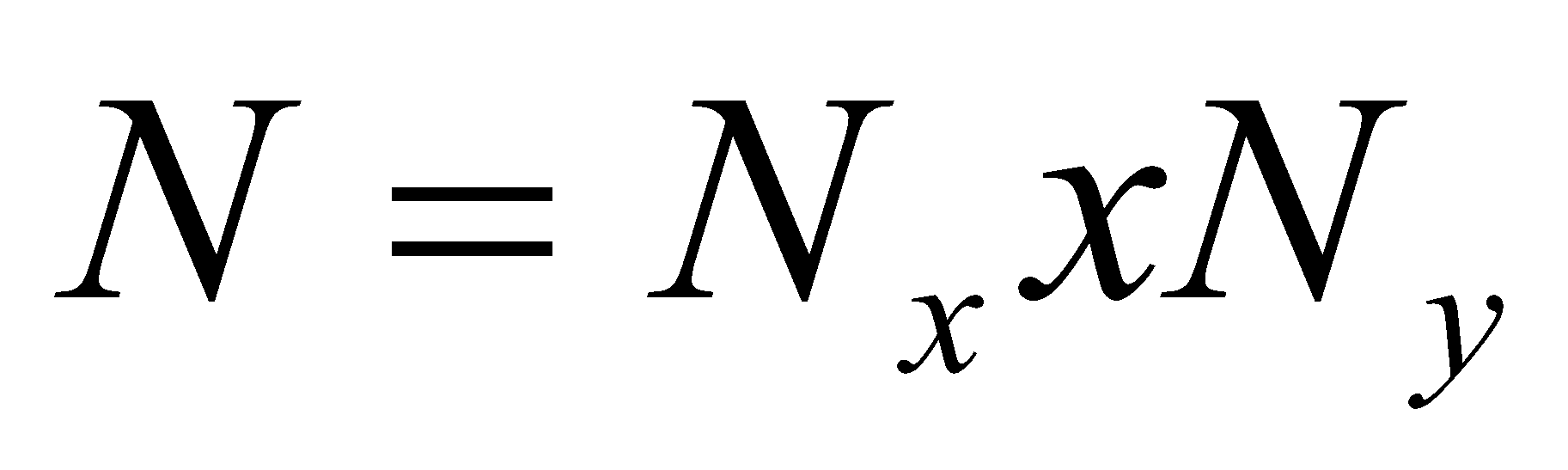


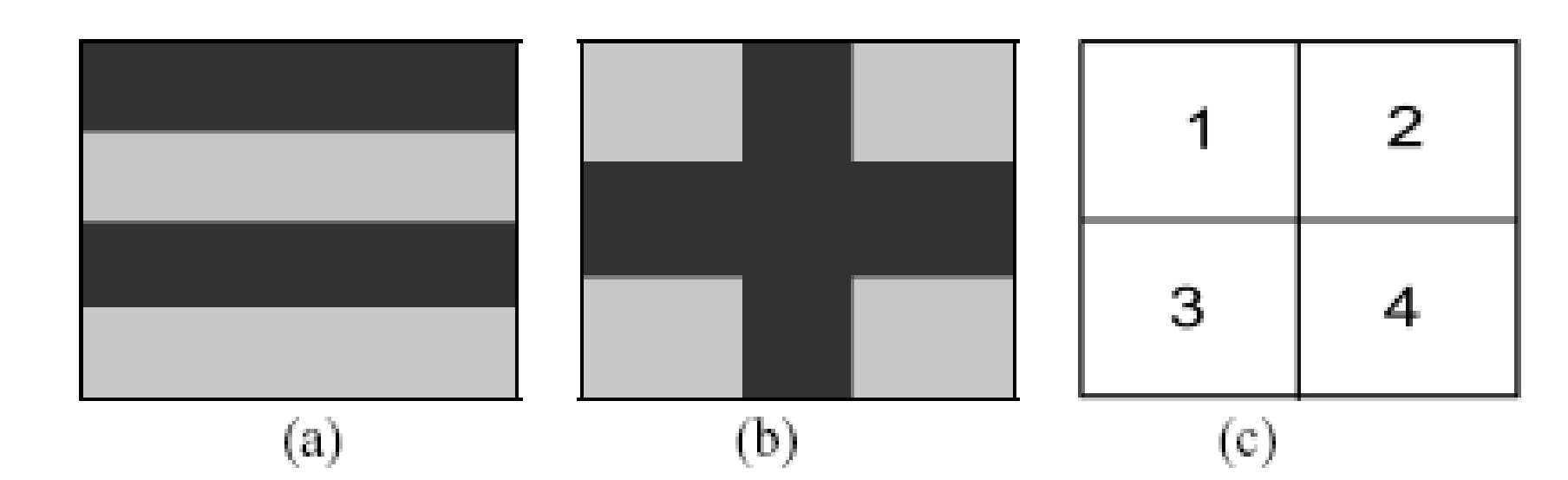
Цветовая гистограмма достаточно устойчива к шуму в изображении, а так же к поворотам и переносам. Но у неё есть несколько недостатков. Один из них, это то, что гистограмма описывает только распределение цвета или яркости на изображении, не учитывая при этом какую-либо пространственную информацию.



Также, если в кадре есть небольшой участок визуально важной информации, при использовании данного метода расчета сигнатур, данный участок может не внести существенного вклада в гистограмму и сравнение будет некорректным. Данный алгоритм в основном используется для разбиения последовательности изображения на похожие фрагменты, однако для нашей задачи он не походит.

## Сигнатура на основе порядковой меры

Алгоритм на основе порядковой меры [] является неким улучшение гистограммного метода. В данном алгоритме сначала изображение разбивается на N () областей. Для каждой из этих областей считается среднее значение яркости. Далее каждой области присваивается порядковый номер от нуля до N, в зависимости величины средней яркости. Вектор, компонентами которого являются эти числа, и есть сигнатура изображения. Основными преимуществами данной сигнатуры являются высокая скорость её вычисления, устойчивость к изменению яркости, а также некоторый учёт пространственной характеристики изображения. Однако существуют большое количество реальных изображений, которые имеют одинаковую порядковую меру, но совсем разное содержание. Это основная причина, по которой данный алгоритм не может быть использован в данной работе.



## Сигнатура на основе использования быстрого преобразования Фурье

Как дискретное преобразование Фурье (ДПФ), так и дискретно-косинусное преобразование (ДКП) широко используется при обработке видеосигнала. Известно, что большая часть информации об изображении содержится в низкочастотных коэффициентах ДПФ и ДКП. Поэтому эти преобразование позволяют уменьшить размерность при обработке изображения.

В алгоритме, рассмотренном в [1], используется трёхмерное ДКП для учёта пространственно-временных характеристик видео клипа. Сначала производится преобразование видео последовательности к единому пространственно-временному формату, в результате которого получается последовательность из 64 изображений, размерами 32х32 пикселя. Далее из них составляется матрица 32х32х64, над которой выполняется трёхмерное дискретно-косинусное преобразование. Для получения сигнатуры используются 64 низкочастотных коэффициентов ДКП (, i = 0..63) из левой верхней подматрицы размером 4х4х4. Далее производится сортировка и квантование коэффициентов. Все коэффициенты, которые больше медианы объявляются равными 1, меньшие же коэффициенты – 0. Медиана считается следующим образом , где - отсортированный список низкочастотных коэффициентов ДКП. Одно битное квантование добавляет устойчивости к небольшим различиям в сигнатуре. Для определения дистанции между сигнатурами в рассмотренной работе используется расстояние Хэмминга — мера различия объектов одинаковой размерности ( , где – объекты размерности p). Результаты экспериментов, проведённых в рассмотренной работе, показывают, что полученная таким образом сигнатура является устойчивой ко многим типам шумов (размывание, изменение яркости, небольшие геометрические изменения, артефакты сжатия), а также обладает уникальностью, т.е. она различна у двух разных изображений.

В работе [2] используется подобный метод построения сигнатуры. Существуют несколько отличий. Во-первых, не учитываются временные характеристики, т.е. ДКП выполняется над двумерной матрицей, представляющей собой обрабатываемое изображение. Во-вторых, квантование происходит относительно среднего значения подматрицы низкочастотных коэффициентов ДКП, т.е. , где nxn – размерность подматрицы (в работе n = 5..8). В качестве функции сравнения, так же, как и в [1], использовалось расстояние Хэмминга.

При использовании данного алгоритма в выпускной работе на соискание степени магистра, выпускника кафедры автоматизации физико-технических исследований [3], в ходе работы, было рекомендовано использовать быстрое преобразование Фурье. В реализации алгоритма, используемого в текущей работе, в качестве сигнатуры были использованы коэффициенты низкочастотной подматрицы без выполнения их квантования. В этом случае в качестве функции сравнения в работе [3] было выбрано Эвклидово расстояние между двумя одномерными векторами размерности .

## Итоги

Алгоритм использования цветовой и яркостной гистограмм в качестве сигнатур, не учитывает какую-либо пространственную информацию, а также небольшие участки визуально важной информации не вносят существенный вклад в сигнатуру, что может понести некорректное сравнение сигнатур, поэтому от данного алгоритма было решено отказаться.

Алгоритм расчёта сигнатур на основе пространственной меры, не обладает достаточной уникальностью, т.к. реальные изображения, могут иметь одинаковую пространственную меру, но совсем разное содержимое, поэтому от данного алгоритма было решено отказаться.

Из рассмотренных алгоритмов, наиболее устойчивым к многим типам шумов, таким как, размывание, изменение яркости, небольшие геометрические изменения, артефакты сжатия и т.п., является алгоритм вычисления сигнатур на основе быстрого преобразования Фурье. Так же, данный алгоритм обладает высоким показателем уникальности [1]. Данный алгоритм был выбран в качестве основного в текущей работе.

# Анализ видеороликов

### Разбиение на сцены

В современном мире существует бессчетное количество различных видеороликов. В каждом видеоролике есть так называемые сцены. Как правило, сцена характерна тем, что кадры внутри одной сцены обладают высокой степенью схожести. В данной главе, степень схожести будет измеряться путем сравнения сигнатур построенных на основе кадров видеоролика. С помощью измерения схожести сигнатур кадров видеоролика, можно попытаться разделить видеоролик на сцены. Такое разделение в будущем поможет сократить количество сравнений при работе с базой данных.

Для исследования были взяты несколько видеороликов, которые отличаются по своему содержанию. Первое видео было взято по типу “интервью”, для такого видео характерно большое количество кадров похожих друг на друга (рис рис рис )

А так же смена сцены характеризуется резкой сменой ракурса съемки (рис рис )



Что бы наглядно продемонстрировать большое количество похожих кадров, было произведено покадровое сравнение всего ролика. А именно, брались соседние кадры и считалась разница сигнатур. На рис (), изображен график отличия соседних сигнатур.

Как видно из графика выше, разница между соседними кадрами в основном не превышает абсолютной величины, равной 0.05. На рис() красным квадратом выделены особые случаи, с величиной больше 0.4, именно на этих кадрах и происходит смена сцены. Таким образом, появляется возможность разделить видео на сцены или видео сегменты анализируя схожесть сигнатур соседних кадров – если разница больше порога, то это новая сцена.

В качестве второго видео, было взято вступление к новостному выпуску.

(Тут будет раскадровка)

В этом ролике, помимо более частой смены ракурсов, камера движется относительно объектов, а объекты не статичны, что в итоге дает низкую степень схожести соседних кадров. На рис () изображен результат сравнений соседних сигнатур.

Как видно из представленного графика, разделение на сцены данного видео по какому-то одному, фиксированному порогу не так очевидно. Например, в промежутке кадров [1000;1600], степень схожести сигнатур в основном колеблется в диапазоне значений от 0.0009 до 0.05, а красными квадратами выбраны особые случаи отличия соседних сигнатур, по которым можно детектировать начала новых сцен. При этом выбрав порог 0.2, в этом промежутке сигнатур, происходит четкое разделение куска видео на сегменты. Однако стоит обратить внимание на промежуток сигнатур, выделенных зеленым квадратом. В этой части графика прослеживается более сложная кластеризация результатов сравнения сигнатур. Задача выбора порога для разделения на сегменты становится уже не такой тривиальной, а выбрав высокий порог, например 0.5, разделение ролика на сцены окажется не корректным.

Как видно из примеров выше, таким способ можно разбить видеоролики на сцены, в каких-то случаях разбиение окажется корректным, однако в других, какой бы мы порог не выбрали, разбиение на сцены будет неправильным или сцен окажется слишком много.

### Выделение ключевых сигнатур

Изначально, идея разбиения на сцены была взята как способ сократить количества сравнений. Например, все ролики разбиваются на сцены, где кадры внутри сцены обладают высокой степенью схожести, как на рис(первое видео). Далее, среди похожих сигнатур, выбирается ключевая сигнатура. Такая сигнатура будет являться представителем всей сцены. Это значит, что если производить поиск не по всем сигнатурам, а только среди ключевых сигнатур, можно с некоторой точностью говорить, что искомая сигнатура находится внутри сегмента. Таким образом, храня в базе поиска не все сигнатуры видеоролика, а только ключевые и выполняя поиск по ним, появляется возможность сократить количество сравнений, в зависимости от того на сколько сцен был разбит исходный ролик. Однако остается вопрос, как наиболее правильным образом разделить видеоролик на некие сегменты, чтобы кадры внутри обладали высокой степенью схожести, а выделение ключевой сигнатуры стало тривиальной задачей. Ведь как видно из видеоролика №2, предложенный способ сравнения соседних кадров и разбиение по порогу не дает желаемых результов. Правильное разделение видеоролика на сегменты и выделение ключевой сигнатуры, даст возможность с высокой достоверностью утверждать, что при выборе этого сегмента по ключевой сигнатуре, искомая сигнатура находится внутри этого сегмента.

Далее предлагается использовать другой подход при разбиении ролика на сегменты. А именно все сигнатуры в сегменте должны отличаться от ключевой не более заданного порога. Таким образом, в качестве ключевой сигнатуры, выбирается первая сигнатура сегмента, а все последующие сравниваются с выбранной ключевой сигнатурой. Если результат сравнения превышает некоторый заданный порог, то сегмент сформирован, а эта сигнатура является началом нового сегмента.

На рис () представлен график результатов работы такого подхода. На отрицательных значениях оси красными квадратами помечены начала новых сегментов. В качестве исследуемого ролика, представлен первый ролик из предыдущей главы: рис(номера)(раскадровок)(первого видео).

Как видно из графика выше, разбиение ролика на сегменты таким способом, совпадает с исходным разбиением ролика на сцены. Такое разбиение гарантирует, что внутри сегмента все сигнатуры не отличаются друг от друга больше, чем выбранный порог (на данном графике порог равен 0.2). После разбиения первого видео на сегменты, из 2600 кадров получается 2 сегмента и 2 ключевые сигнатуры, что позволяет при поиске сигнатуры сократить количество сравнений более чем в 1000 раз.

Однако на более динамичных видео роликах не удается добиться такого высокого ускорения. Например, на втором видеоролике из предыдущей главы рис(раскадровка)(второго)(видео). На рис() представлен результат работы разделения на сегменты способом описанном выше. Так же в отрицательных значениях оси помечены начала новых сегментов.

Новый сегмент

В данном видео из 2000 сигнатур, с помощью представленного алгоритма, удалось сформировать 94 сегмента, что дает уменьшение количества сравнение при поиске по ключевым сигнатурам примерно в 21 раз.

Как можно заметить из сравнения рис (первый ролик) и рис (второй ролик), чем более динамичный ролик, тем большее количество сегментов формируется в результате работы алгоритма предложенного в этой главе.

В качестве третьего ролика для исследования, был выбран музыкальный клип. Для этого видеоролика характерно сильно изменяемое освещение от кадра к кадру рис() и рис(), а так же помимо динамических сцен, они меняются каждые 5-10 кадров.

Как видно из рис (), такой видеоролик плохо разбивается на сегменты. В отрицательной части оси , точки означающие начало нового сегмента образуют сплошную линию, а именно на 2090 кадров видеоролика, было сформировано 462 сегмента. При этом, при обработки данного видеоролика, был выбран самый высокий порог смены сегмента, равный 0.4. Таким образом количество сравнений удалось сократить только в 4 раза, что является неприемлемым в рамках нашей задачи.

На примере данно видео, становится очевидным необходимость создания ограничения минимального размера сегмента. Такие видео сегменты нуждаются в дополнительной постобработке.

В текущей дипломной работе, размер минимального видео сегмента составляет 25 кадров. В действительности, такое ограничение можно менять в зависимости от типа ролика. Так же, такое ограничение позволяет фиксировать уменьшение количество сравнений в 25 раз.

### Особенности вычисления сигнатур с помощью БПФ

### Алгоритм разбиения ролика на сегменты

# Постановка задачи

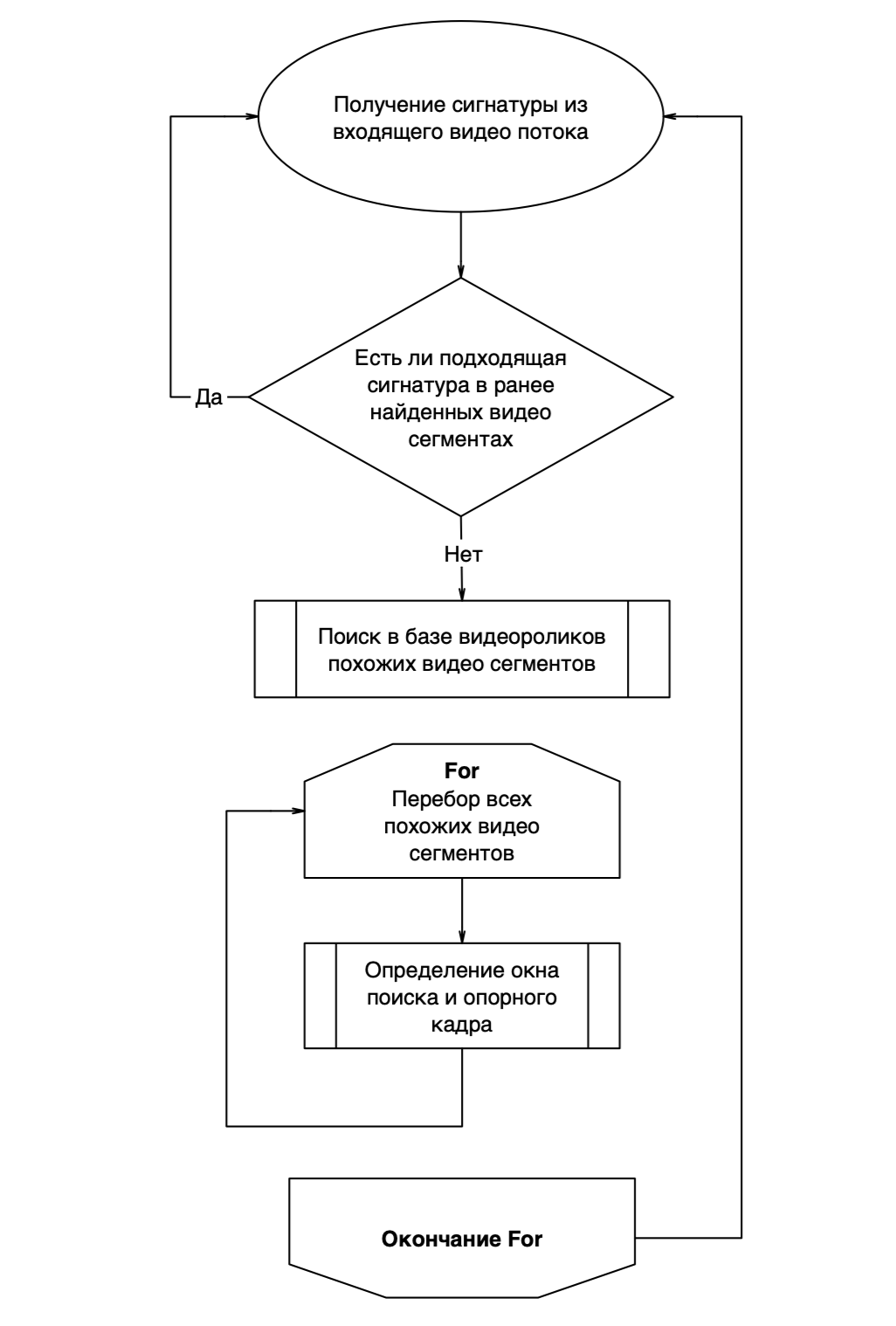
## Описание проблемы

Для того, чтобы получить список совпадающих видеороликов во входящем видеопотоке ~~реального времени~~, необходимо провести сравнение входящих сигнатур кадров с базой роликов. При работе с базой, состоящей из 1000 роликов со средней длительностью ролика в одну минуту c частотой кадров 25 необходимо сравнить сигнатуру из входящего видеопотока с более чем сигнатур из базы роликов. Характерное время сравнения двух сигнатур составляет 5 мкс. Соответственно, за время одного кадра можно сравнить только 8000 сигнатур. Необходимо в тысячу раз сократить число сравнений.

Практически в любом видео существуют последовательности похожих кадров. Они могут быть разной длины и разной степени схожести. Можно объединять такие последовательности в сегменты и для каждого сегмента задавать одну «ключевую» сигнатуру. Если сравнивать сигнатуру входящего видеопотока не со всеми сигнатурами подряд, а только с ключевыми сигнатурами видео сегмента, то можно сократить количество сравнений в десятки или даже в сотни раз. Сегменты формируется путем сравнения соседних сигнатур видео ролика. Если очередной кадр отличается от предыдущего кадра слишком сильно, то с него начинается новый сегмент. ~~В данной дипломной работе, ключевой сигнатурой является первая сигнатура в видео сегменте.~~

В процессе анализа бесконечной последовательности входных кадров выполняется хранение ранее найденных «похожих» частей видеороликов – список «активных» роликов. При получении очередной сигнатуры небходимо выполнять поиск совпадающей сигнатуры в каждом «активном» ролике. Если сигнатуры сильно отличаются, то соответствующую часть необходимо удалить из списка «активных» роликов. При таком сравнении можно требовать точного совпадения сигнатур, в то время как для поиска в базе данных роликов, на первом этапе, можно требовать грубого совпадения сигнатур. То есть, на первом этапе нужно отсеять гарантированно непохожие ролики, а на втором этапе – оставить только гарантированно похожие.

Работа с входящем видеопоток формирует конечный автомат. После получения очередной сигнатуры из бесконечной входящей видеопоследовательности эта сигнатура подается на вход алгоритму поиска «похожих» видео сегментов. Поиск выполняется с “большим” порогом, чтобы гарантированно отсеять несовпадающие видео части. Алгоритм поиска выполняет сравнение с ключевыми сигнатурами видео сегментов ~~используя порог, который задается пользователем~~. Если разница сигнатур меньше порога, этот видео сегмент помечается как похожий и является результатом поиска. Результатом работы алгоритма поиска могут быть несколько похожих сегментов, так как поиск выполняется с «большим» порогом. После получения набора похожих видео частей, необходимо, использовать более низкий порог, чтобы гарантированно оставить только похожие видео сегменты. Так же возможны случаи, когда искомая сигнатура находится внутри сегмента или сегмент, состоит из абсолютно одинаковых кадров (продолжительный черный экран). Что бы допустить минимум ошибок при работе с такими сегментами, необходимо использовать окно поиска. То есть, выполнить расширение зоны поиска кадра из входящего видеопотока. Для определения границ такого окна, выполняется поиск сигнатур внутри видео сегмента, которые совпадают с сигнатурой из входящего видеопотока. После нахождения нескольких похожих кадров, внутри видео сегмента, необходимо сформировать окно. Окно формируется по принципу: берется первый совпадающий кадр как левая граница окна, а последний, как правая граница окна. Индекс опорной сигнатуры совпадает с левой границей окна. Опорная сигнатура, это сигнатура кадра, которая является искомой на данной итерации конечного автомата. Если на следующей итерации конечного автомата, сигнатура из входящего видеопотока не совпадает с ожидаемой сигнатурой из видео сегмента, то выполняется корректировка окна, таким же способом, как и его определение. Однако, если корректировка окна не требуется, или после корректировки левая граница не изменилась, то индекс опорной сигнатуры увеличивается на единицу, а предыдущая сигнатура помечается как совпавшая. Сохраняя найденные сегменты на каждой итерации конечного автомата, появляется возможность отслеживать количество ошибок при поиске. Отслеживая количество ошибок, которое приходится на тот или иной видео сегмент, выполняется отсеивание ошибочных видео сегментов. После того как работа с видео сегментом закончена, например, закончились кадры, если видео сегмент подходит под условия поиска, т.е. количество ошибок не превышает заданного, можно с высоким уровнем достоверности утверждать, что этот видео сегмент и есть искомый. Таким образом выполняется достоверный поиск совпавших видео сегментов в входящем видеопотоке реального времени.



Как наиболее оптимальным способом хранить подготовленные видео сегменты, для ускорения поиска, выходит за рамки данной дипломной работы. Время, затраченное на эту процедуру, не будет учитываться.

# Разработка алгоритма

# Введение

### Разбиение ролика на сегменты

Для подготовки ролика и разделения на сегменты, необходимо определять: резкую смену сцены и плавную смену сцены. Резкая смена сцены характеризуется большим значением отличия соседних сигнатур. Для определения резкой смены сцены вводится порог . Но кадры в роликах не всегда имеют характерные границы разделения по порогу . Возможны случаи, когда сцены сменяются плавно, т.е. отличие сигнатур соседних кадров ⋘ . Такой случай характеризует плавную смену сцены. При этом, если разделять видео по порогу , будут образовываться сегменты больших размеров, что негативно влияет на скорость поиска необходимого сегмента. Что бы корректно детектировать и избегать такие случаи, вводится порог  – порог сцены. Данный порог используется как ограничитель между первым и последним кадром в сегменте, т.е. если сравнение первой и последней сигнатуры больше порога , считается что данный сегмент сформирован. Порог настраивается пользователем, однако в данной дипломной работе будут даны рекомендации по настройке порога.

Для работы с сигнатурами, были введены 3 основных порога сравнения:

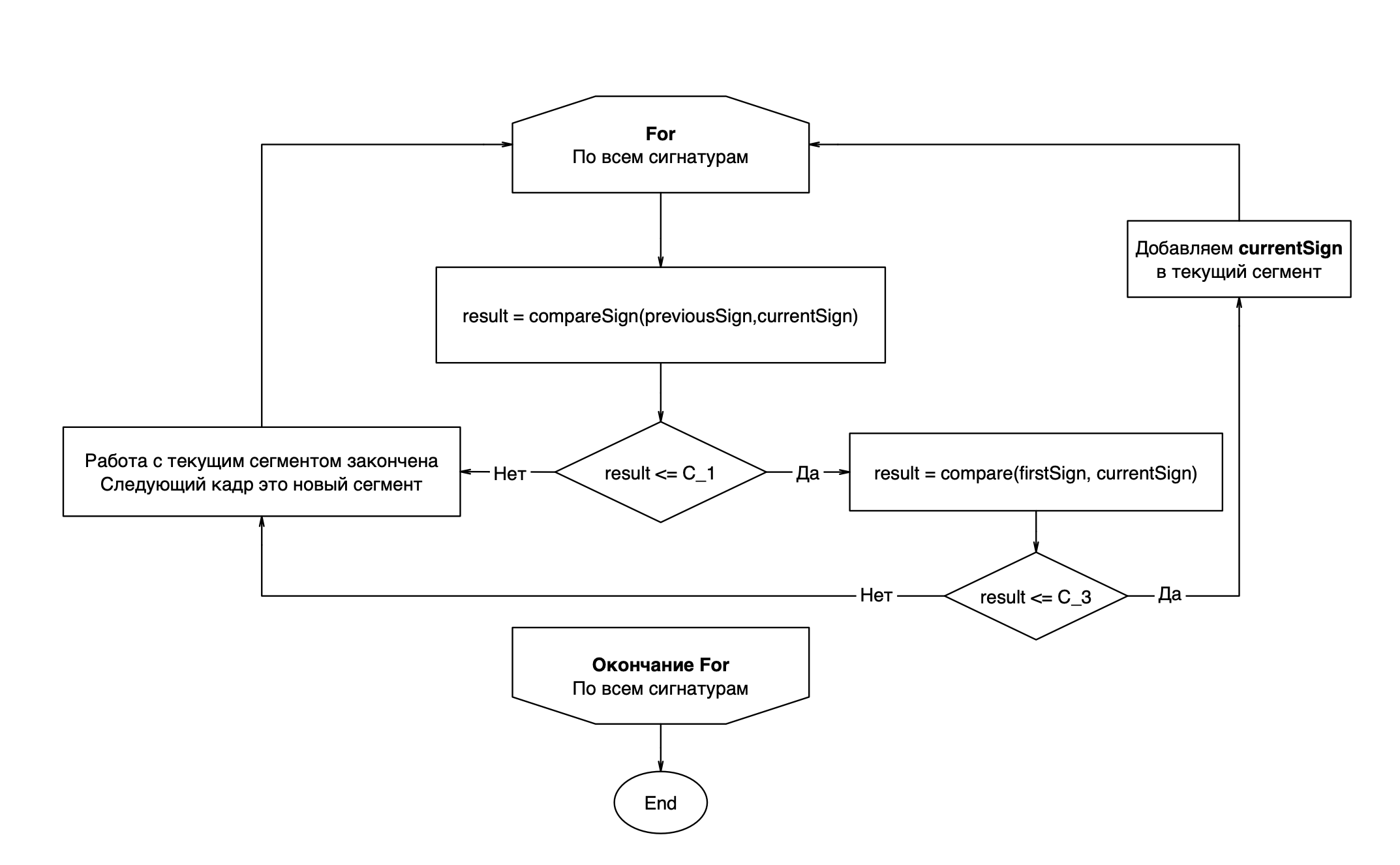
* – порог резкой смены сцены.

* – порог сжатия. Данный порог необходимо вычислить эмпирически, путем сравнения исходного видео и после различных популярных алгоритмов сжатия.

* – порог сцены. Данный порог настраивается в зависимости от категории видео. Ориентировочные характеристики: , n = 2,3,4,5…

Далее представлена принципиальная схема работы алгоритма разбиения видеоролика на сегменты. На вход данный алгоритм принимает массив сигнатур ролика. firstSign – первая сигнатура в сегменте.

* lastSign – последняя сигнатура в сегменте.
* previousSign и currentSign – предыдущая и соответственно текущая сигнатура в массиве сигнатур.



Для хранения информации о сегменте, используется структура VideoPart.

Поля структуры хранения части ролика:

* mainSignatureIndex – индекс ключевой сигнатуры. Ключевая сигнатура – как правило первая сигнатура в VideoPart, которая используется при поиске по сегментам.
* lastSignatureIndex - индекс последнего кадра, входящего в промежуток текущего VideoPart.
* selfIndex – собственный индекс в массиве состоящем из VideoPart.

# Анализ входящего видеопотока

### Поиск очередного кадра по уже найденным сегментам.

После получения очередного кадра из входящего видеопотока, выполняется поиск по базе данных подготовленных видеороликов. Данный алгоритм производит поиск с порогом . Результатом поиска является массив VideoPart, которые подходят под условия поиска. Далее необходимо обработать как новые VideoPart (с которыми не производилась работа), так и VideoPart, с которыми производилась работа на предыдущем шаге.

Порог выбран в алгоритме поиска для того, чтобы корректно распознать ситуацию, когда искомый кадр находится внутри VideoPart. Ведь сравнение в алгоритме поиска производится только с первым кадром VideoPart. VideoPart устроен так, что в нем находятся похожие кадры, в связи с этим не всегда можно утверждать что найденный кадр и есть искомый, возможны ситуации когда таких кадров несколько, в связи с этим возникает необходимость создания “окна” поиска – SearchWindow. Данная структура, хранит данные о размере окна. Так же к отдельной категории относятся ситуации пропуска кадра в входящем видеопотоке, для этого, стандартным размером окна является +- 1 кадр, от кадра с которым ведется работа. В других случаях размер может быть произвольным.

Поля структуры SearchWindow:

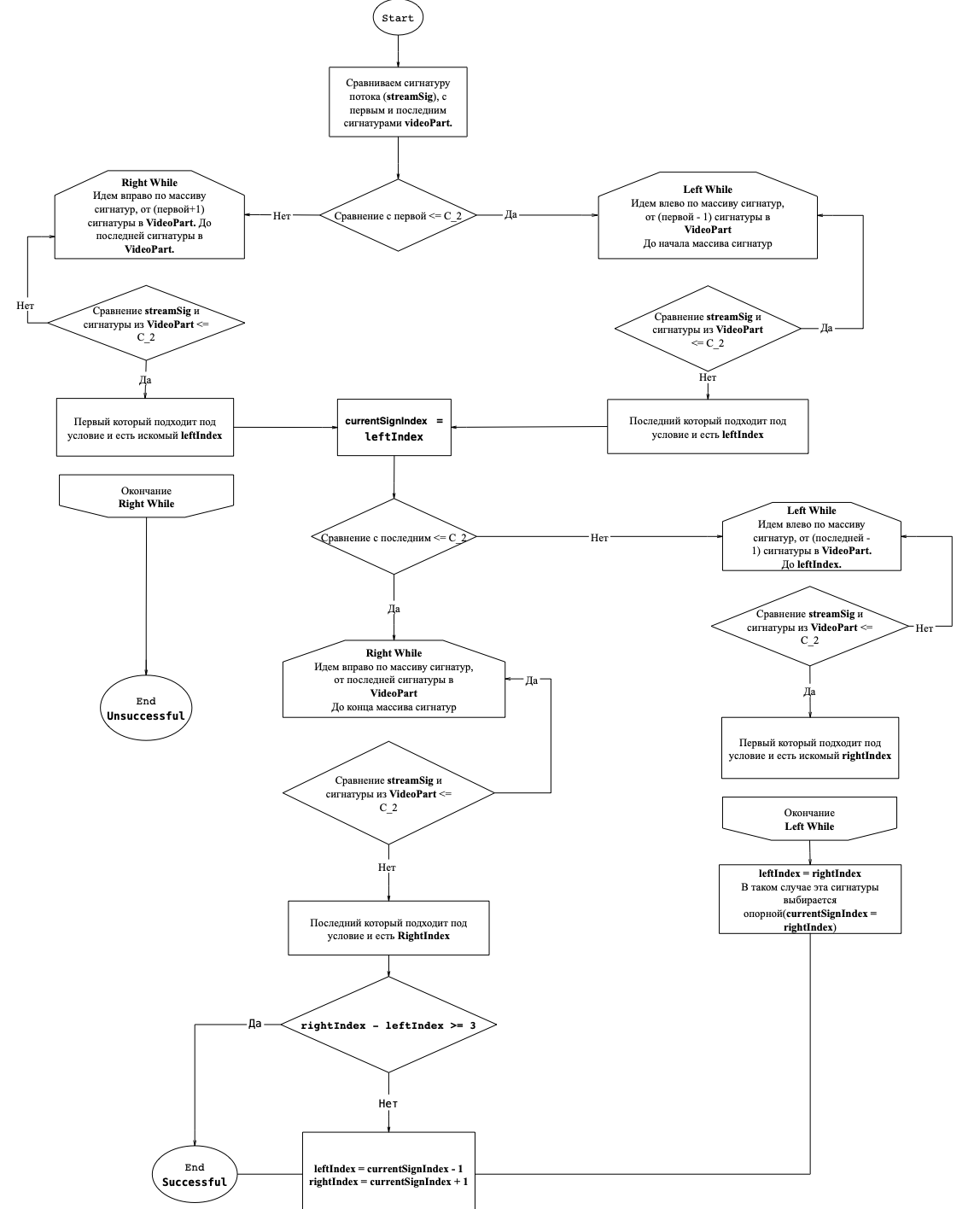
* leftIndex – абсолютный индекс левого края окна.
* rightIndex – абсолютный индекс правого края окна.

Алгоритм определения окна поиска выходит за рамки VideoPart. Такой прием используется для исключения ситуации, когда искомый кадр находится в соседнем VideoPart. Производится поиск сигнатур, результат сравнения с которыми будет ниже порога C\_2. Первый такой кадр выбирается опорным, а размеры окна поиска ограничиваются последней сигнатурой, сравнение с которой ниже порога С\_2.

Далее представленная схема алгоритма определения окна поиска SearchWindow.

Данная функция позволяет выбрать кадр, с которого начать работать с текущим VideoPart, а также определить окно поиска в рамках текущего VideoPart.

* leftIndex – абсолютный индекс сигнатуры в массиве сигнатур ролика. Обозначает левый край окна поиска.
* rightIndex - абсолютный индекс сигнатуры в массиве сигнатур ролика. Обозначает правый край окна поиска.
* currentSignIndex – индекс опорного кадра. Предполагается что именно этот кадр и есть искомый (или с него нужно начать поиск). Указывает на сигнатуру в массиве сигнатур.

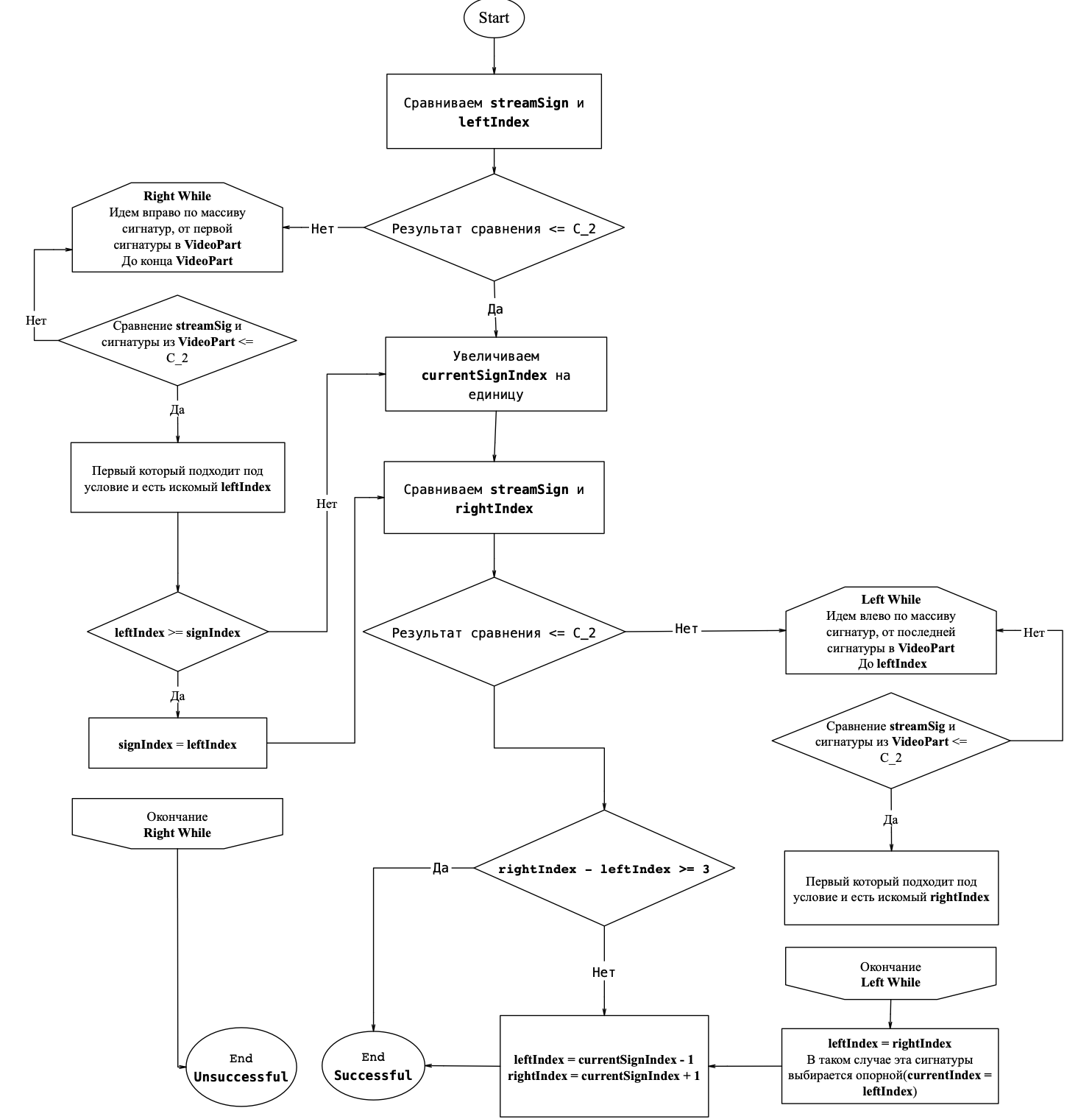


## Работа с сигнатурами внутри окна

На следующих шагах алгоритма поиска, необходимо обработать кадры, которые находятся в окне поиска, а также произвести корректировку окна поиска. Данный шаг выполняется в целях сужения окна поиска, а также определения корректности ранее найденной информации. Алгоритм выполняет поиск сигнатуры с левого края SearchWindow до правого, выполняя поиск кадра, результат сравнение сигнатуры которого с сигнатурой из входящего видеопотока будет < C\_2. Такая сигнатура становится опорной, а размеры окна корректируются. Если размер окна не изменились, индекс ожидаемой сигнатуры увеличивается на единицу. Для определения второй границы окна поиска выполняется поиск справа на лево по SearchWindow. Первый кадр, который подходит по условию и является второй границей SearchWindow.

Далее представлена схема алгоритма работы с окном поиска.

Данная функция выполняет корректировку окна (если это необходимо), а также меняет индекс опорного кадра.

* streamSig – сигнатура из видеопотока,
* leftIndex - левый край окна в массиве сигнатур ролика,
* rightIndex - правый край окна в массиве сигнатур ролика,
* currentSignIndex - индекс ожидаемого кадра в массиве сигнатур ролика.

Алгоритм поиска похожих частей по сигнатуре входящего видео с порогом C\_3, может возвращать несколько таких частей, так как каждую из них необходимо обработать, так же продолжать работать на нескольких итерациях сравнения, возникает потребность работы с несколькими VideoPart одновременно. Структура SearchInfo созданная решить эту проблему, объединяя в себе всю необходимую информацию для работы.

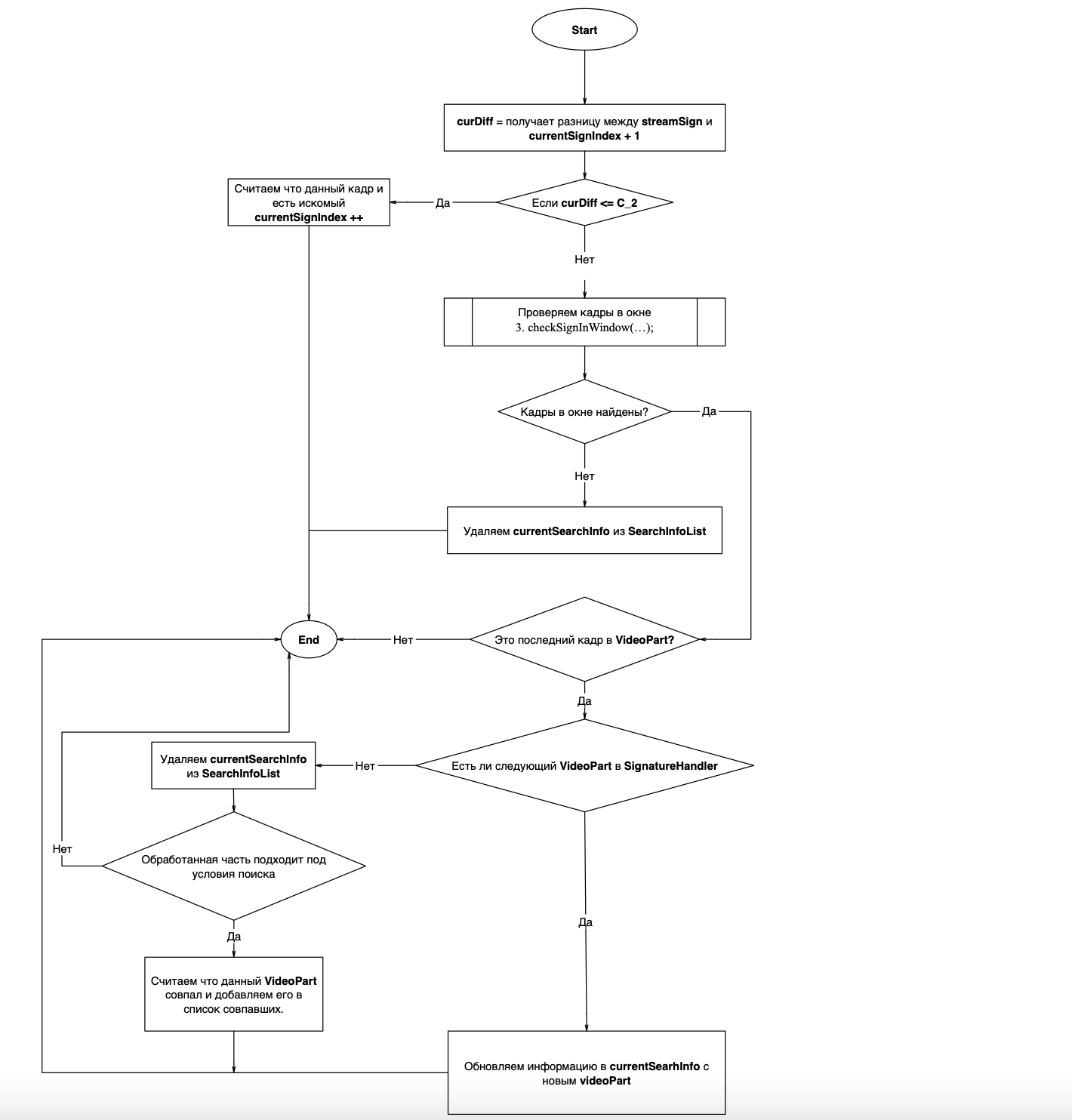
## Работа с VideoPart в рамках SearchInfo.

В алгоритме, представленном в данной дипломной работе, производится работа с каждым VideoPart отдельно. Для начала выполняется сравнение сигнатуры кадра входящего видеопотока с сигнатурой кадра, которого мы ожидаем следующим в рамках VideoPart. Если результат сравнения < C\_2, считается что данный кадр и есть искомый, иначе производится работа с сигнатурами внутри окна. Если следующий кадр выходит за рамки текущего VideoPart, работа с данным VideoPart закончена. Если сегмент подходит под условия поиска, он добавляется в список совпавших частей.

Далее представлена схема работы алгоритма с очередным VideoPart

Данная функция выполняет работу с SearchInfo. Берет сигнатуру из VideoPart и считает разницу сигнатур. В зависимости, от которой выполняет те или иные действия.

* currentSignIndex - индекс текущего кадра в VideoPart.
* currentSearchInfo – класс SearchInfo с которым ведется работа.

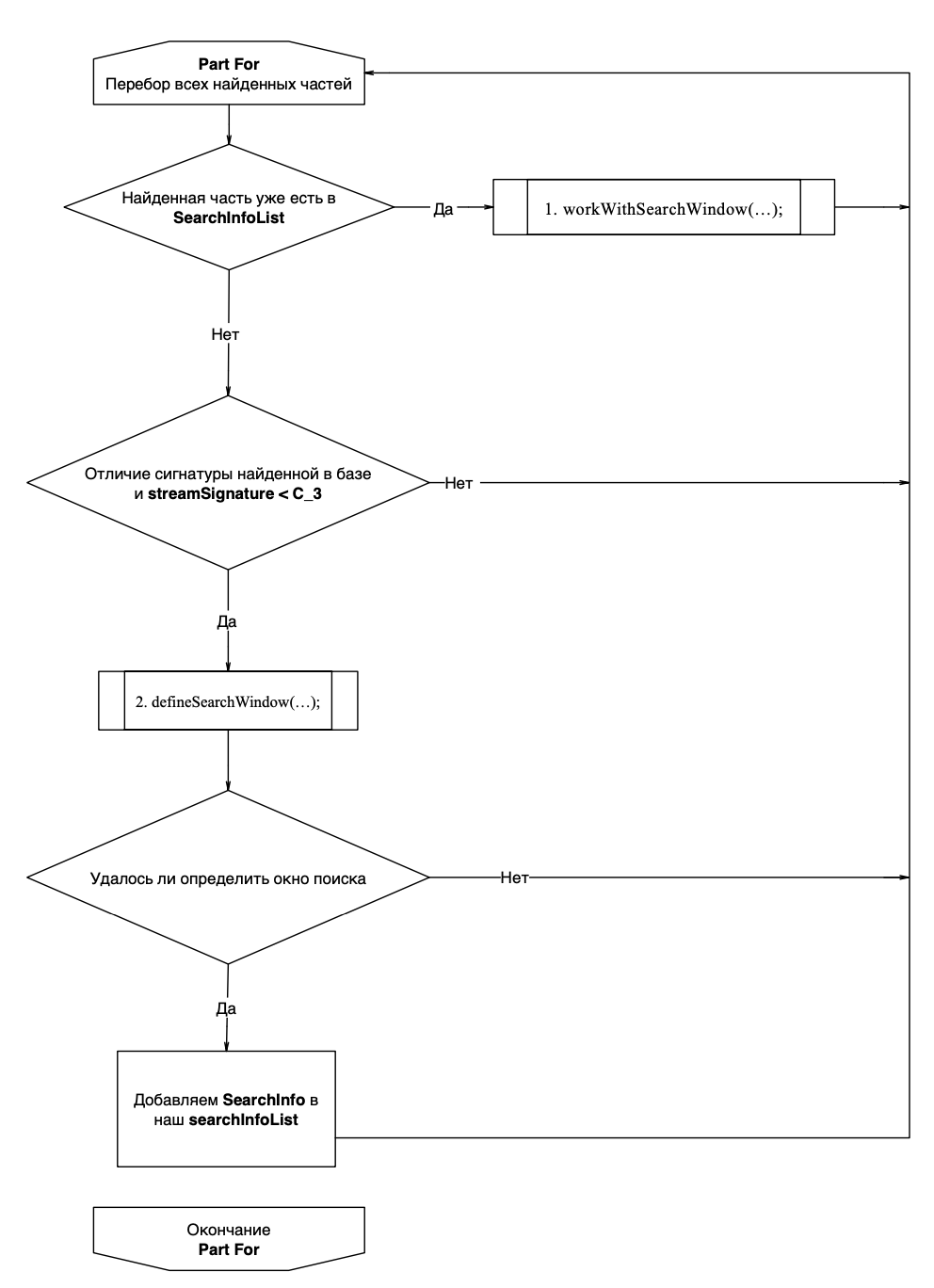


## Работа с найденными VideoPart

Как было описано выше, алгоритм поиска по текущему кадру из видеопотока внутри структуры хранения сегментов ролика может возвращать как несколько VideoPart, так и не одного. Все они сохраняются в рамках SearchInfo и производится обработка. Выше были описаны вспомогательные алгоритмы, которые находятся внутри основного. Основной алгоритм работает как конечный автомат. Т.е. берет кадр из входящего потока и выполняет обработку. В ходе работы этого алгоритма выполняется обработка как вновь найденного набора VideoPart, так и набора VideoPart найденных на предыдущем шаге конечного автомата. Для хранения VideoPart выбрана структура хранения – лист. Хоть она и не отличается высокой скоростью поиска, но в этом и нет необходимости, так как работа с VideoPart будет выполняться последовательно, а операции добавления и удаление производятся часто.

Далее представленная общая схема работы алгоритма при том условии, что в результате работы алгоритма поиска похожих сегментов с порогом C\_3, сегменты были найдены.

* streamSignature - сигнатура текущего кадра из видеопотока,
* searchInfoList – лист из экземпляров класса SearchInfo.



После окончания работы с найденными частями, выполняется повторный проход по SearchInfoList и обработка тех SearchInfo, с которыми работа не производилась, функцией workWithSearchWindow.

# Вывод

# Список литературы