# Архитектура алгоритма

Далее будет описан алгоритм анализа бесконечного входного видеопотока (последовательность изображений) с целью поиска в нем любых частей из массива заранее подготовленных роликов. Для достижения поставленной цели, необходимо создать базу роликов, в которой будет производиться поиск.

Перед тем, как добавить новый ролик в базу данных, его необходимо подготовить. Это делается для ускорения поиска сигнатур по базе данных.

Основная идея, состоит в том, что любой ролик можно разбить на части. Части будут строиться по принципу схожести, т.е. если в ролике подряд идут похожие кадры, они будут добавляться в одну и ту же часть, если порог, при сравнении сигнатур, больше заданного, то будем считать, что началась следующая часть. Порог в текущем исследовании будет носить рекомендательный характер и может быть изменен пользователем.

Далее будут описаны структуры хранения частей ролика, а также алгоритм их наполнения.

Основные понятия и константы:

* VideoPart – структура хранения части ролика.

* – порог сравнения первого и последнего кадра из VideoPart.

* – 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. Так же данный порог подается в алгоритм поиска. Ориентировочные характеристики: , n = 2,3,4,5…
* mainSignature – сигнатура, являющаяся представителем ячейки VideoPart.

## Структуры и классы для хранения и работы с информацией

## Класс для хранения сигнатур ролика, а также его подготовленных частей (VideoPart)

За хранение набора сигнатур, вычисленных из кадров подготавливаемого ролика, отвечает класс **SignatureHandler**. А именно в нем хранится массив сигнатур. Так как поиск в массиве происходит за константное время O(1), в создаваемой структуре **VideoPart** не будет хранится сигнатура, а лишь разметка. При этом у класса **SignatureHandler** будет константный метод **getSignature(size\_t index)**, который будет возвращать необходимую сигнатуру по индексу.

## Структура хранения части ролика: “VideoPart”

Далее будут описаны поля структуры хранения части ролика:

* Номер ключевого кадра. Ключевой кадр, это кадр подготавливаемого ролика, который был выбран как основной для текущей VideoPart. (mainSignatureIndex).
* Индекс последнего кадра, входящего в промежуток текущего VideoPart. (lastFrameIndex)
* selfIndex – собственный индекс в классе **signatureHandler.**

## Структура для хранения размера поискового окна: “**SearchWindow**”

**SearchWindow** – структура используемая для поиска кадров в режиме “окно”. Для стандартных случаев, это +- 1 кадр (стандартная возможная ошибка пропуска кадров), в других случаях данный размер может быть произвольным.

Поля структуры **searchWindow**:

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

## Структура для хранения данных о поиске: “**SearchInfo**”

Так как необходимо работать с несколькими videoPart одновременно, объединяя данные структуры в массив, появляется возможность работы с несколькими **VideoPart** одновременно.

* + **searchWindow** – размеры окна с которым ведется работа.
  + **videoPart** – непосредственно **VideoPart** с которым ведется работа.
  + **sigIndex** – текущий индекс в массиве сигнатур из **SignatureHandler**, на который указывает VideoPart.
  + **isWorkWith** – булевская переменна, отвечающая за то, что мы уже работали с этим блоком.

## Структура, представляющая из себя: ключ/значение для создания структуры поиска: “SearchCell”

Ключ: **signatureIndex** – индекс сигнатуры, которая учувствует в построении структуры поиска.

Значение:

* **signatureHandlerIndex** – индекс класса **SignatureHandler**, в массиве всех роликов (**DataInformation**). В этом **SignatureHandler** находится сигнатура по индексу: signatureIndex.
* **videoPartIndex** – индекс класса **VideoPart** в массиве videoParts, который относится к **SignatureHandler** по индексу signatureHandlerIndex. Так же в этом **VideoPart** сигнатура по индексу **signatureIndex**, является **mainSignature**.

## Класс, отвечающий за хранение информации обо всех роликах: “DataInformation”

Данный класс, призван объединить все подготовленные ролики, в единую базу. Так же **DataInformation** выполняют важную функцию подготовки имеющейся информации для создания базы поиска, а также структурирования информации. Для этого создается массив структур **SearchCell**.

Данный класс отвечает за сохранения информации о роликах в памяти долгого хранения, а также чтения этой информации и восстановления информации о базе из памяти.

## Алгоритмы

## Алгоритм сравнения кадра из потока видео с массивом VideoPart: **“PartSearcher”**.

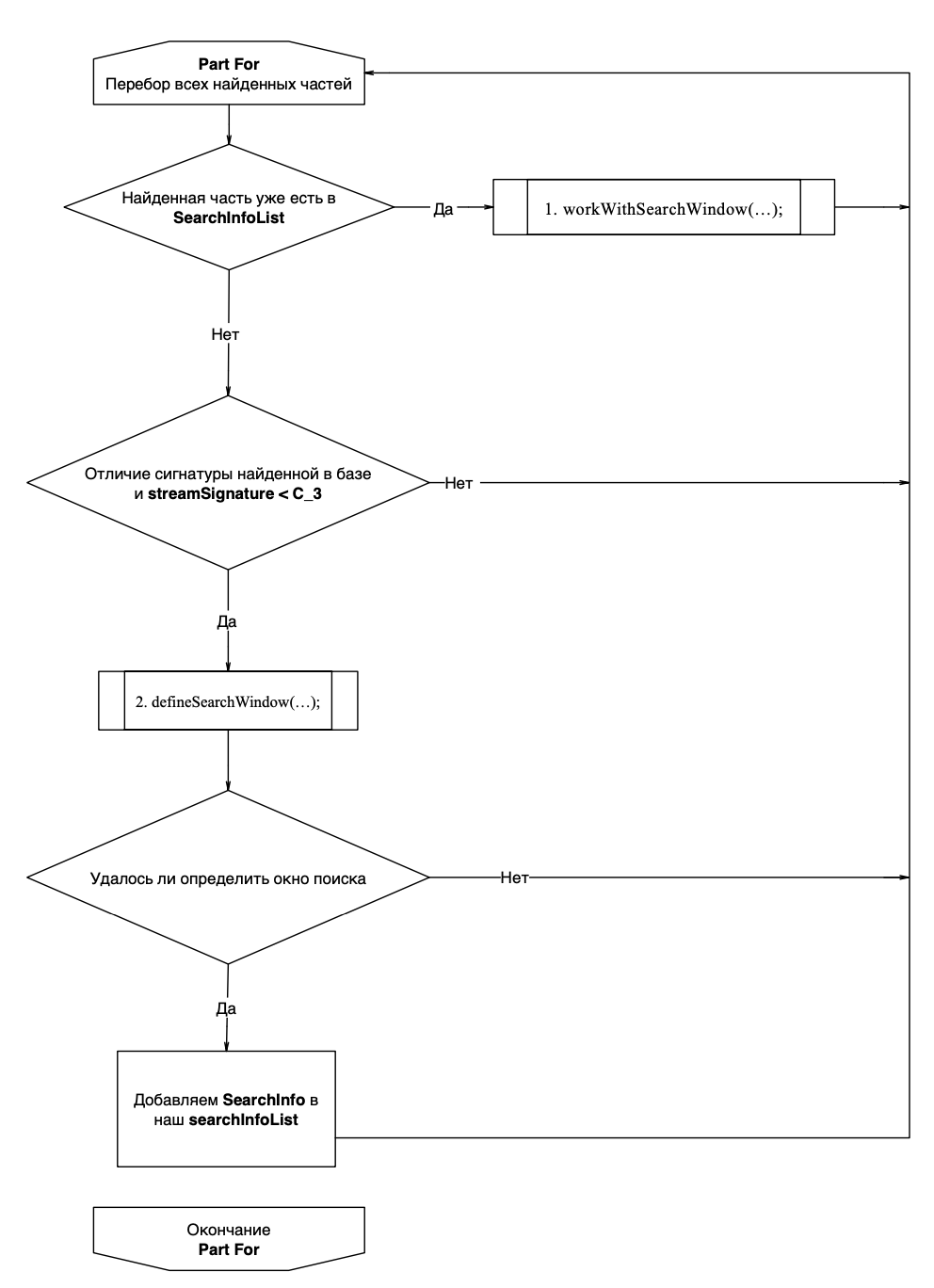
**PartSearcher** – класс отвечающий за поиск совпадающих частей из входящего видеопотока и структуры хранения данных.

## StartSearching()

Данная функция является StateMachine. Берет кадр из потока и начинает свою работу, пока в потоке есть кадры.

По полученному кадру выполняется поиск похожих частей с порогом C\_3. Если похожие части найденные, выполняется алгоритм представленный на рисунке ниже.

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

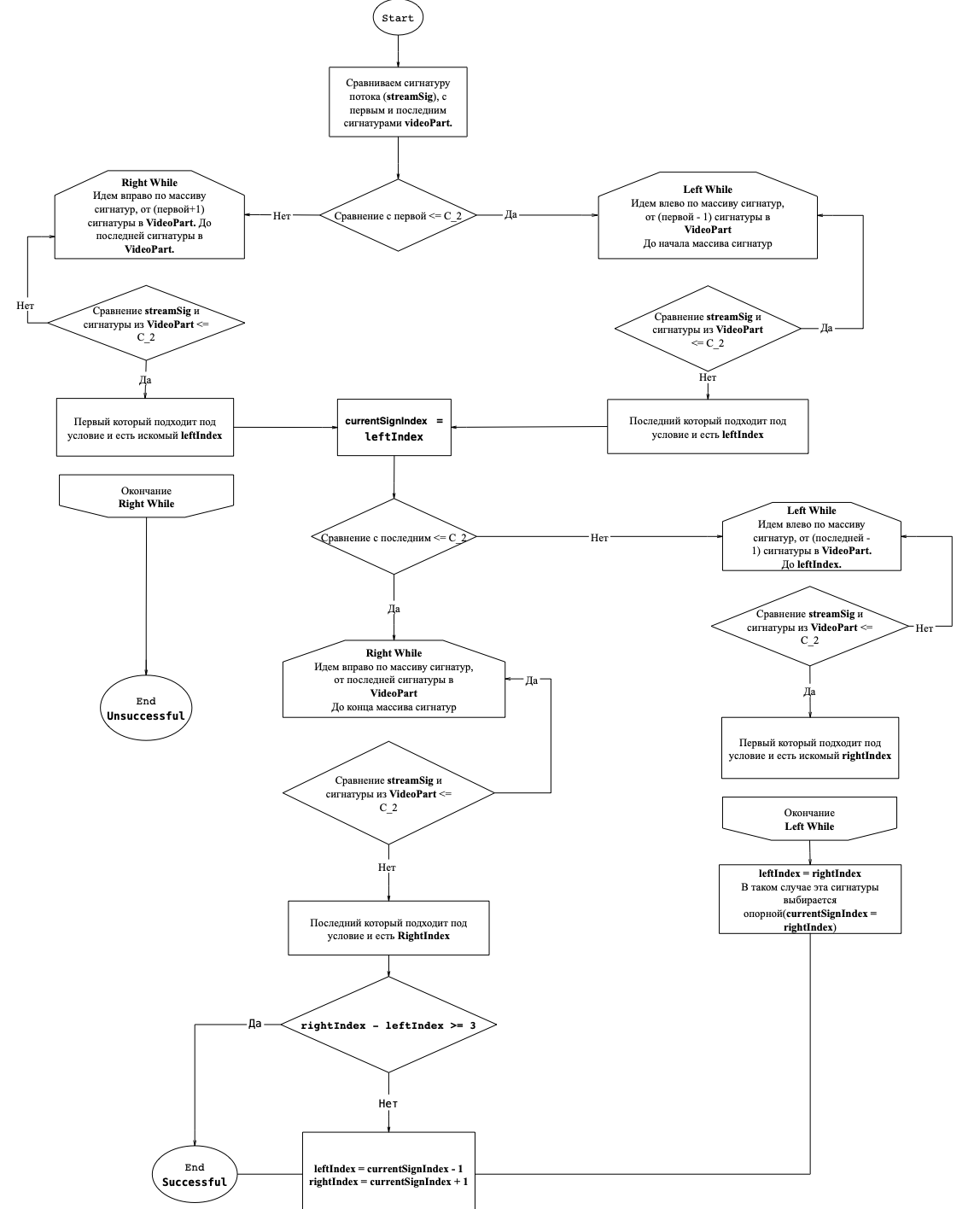


После окончания работы с найденными частями, выполняется повторный   
проход по **SearchInfoList** и обработка тех **SearchInfo**, с которыми работа   
не производилась, функцией **workWithSearchWindow**.

## defineSearchWindow(SearchInfo &currentSearchInfo, CSignature \*streamSig)

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

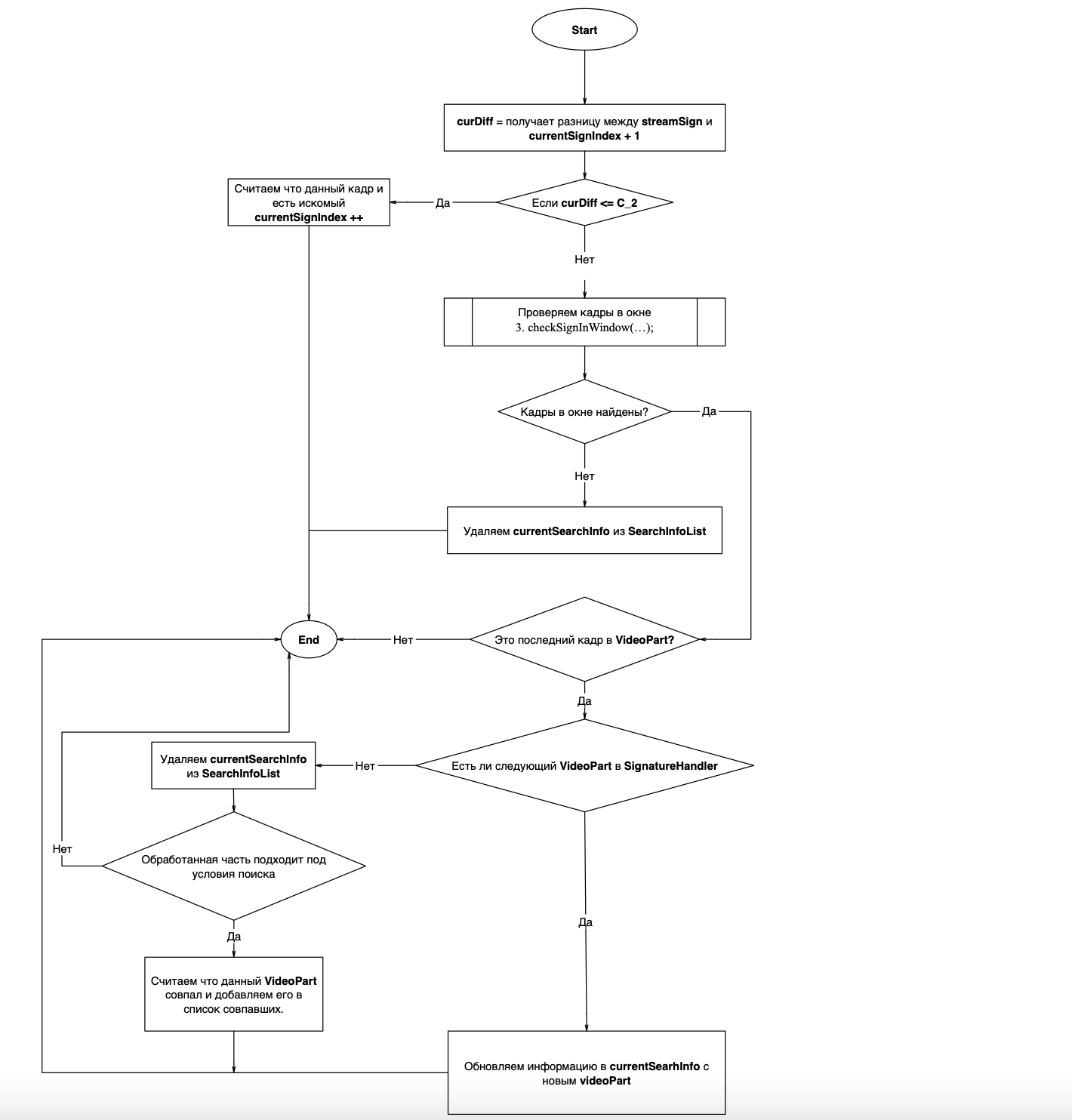
* **leftIndex** – абсолютный индекс сигнатуры в массиве сигнатур ролика. Обозначает левый край окна поиска.
* **rightIndex** - абсолютный индекс сигнатуры в массиве сигнатур ролика. Обозначает правый край окна поиска.
* **currentSignIndex** – индекс опорного кадра. Предполагается что именно этот кадр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



## workWithSearchInfo(SearchInfo &currentSearchInfo, CSignature \*streamSign)

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.



## checkSignInWindow(SearchInfo &currentSearchInfo, CSignature \*streamSig)

Данная функция выполняет корректировку окна(если это необходимо), а так же меняет опорный кадр.

* **currentSearchInfo** - класс **SearchInfo** с которым ведется работа,
* **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.

