УДК 004.932.2

Алгоритм поиска событий в видеопоследовательностях

Лукин В.Н.*, Никитин И.К.**

Московский авиационный институт (национальный исследовательский университет), МАИ, Волоколамское шоссе, 4, Москва, А-80, ГСП-3, 125993, Россия

**e-mail: lukinvn@list.ru*

***e-mail: w-495@yandex.ru*

Аннотация

В работе рассмотрены алгоритмы поиска событий в видео. Задача поиска событий сведена к задаче сегментации видео. Введено понятие аномалии в видео. Предложен метод поиска событий, основанный на сравнении скользящих средних с окнами разного размера. Проведено сравнение этого метода с другими методами поиска аномалий. Предложен язык потоковой обработки видео, описана его грамматика. Приведены примеры использования этого языка. Сформулирована теорема о нечетких дубликатах видео.

Ключевые слова: поиск разладок, сегментация видео, дубликаты видео, скользящая средняя оценка, потоковая обработка видео.

Введение

Актуальность проблемы поиска событий в видео объясняется интенсивным использованием видеоданных в современных вычислительных системах, что закономерно порождает проблему эффективного их поиска и быстрого доступа к ним. За 20-30 лет человечество накопило большое количество видеоинформации. Речь идет не только о видео в интернете. Речь идет, в первую очередь, о камерах наблюдения. Этот массив информации невозможно просмотреть целиком. Эффективных методов анализа видео, подобных анализу текста, в настоящее время нет. Необходимо иметь возможность индексировать видео и осуществлять поиск по нему. Для решения задачи поиска применяют ассоциативный метод. В англоязычной литературе ассоциативный поиск видео называют «content based video retrieval» — поиск по содержимому. Поиск событий в видео — его частный случай.

Подходы

Ассоциативный поиск видео состоит из следующих шагов:

- анализ временной структуры видео: деление видео на фрагменты, которое включают обнаружение границ съёмок;
- определение характеристик фрагментов: исследование параметров ключевых кадров, характеристик движения и объектов;
- извлечение информации из полученных характеристик.

Несмотря на то, что видео может быть представлено последовательностью неподвижных изображений, оно содержит больше информации, чем просто серия кадров. Временная структура видео может быть представлена рядом ключевых кадров, на которых показаны события. Под событиями понимается не смысловая составляющая сюжета видео, а только изменение содержимого кадров. В такой постановке задачу поиска событий иногда называют сегментацией видео. Существует два типа методов сегментации видео: пороговые и статистические.

Пороговые методы

Пороговые методы попарно сравнивают подобия кадров с заданным порогом [2, 15]. Если сходство между кадрами или «окнами» оказывается ниже порога, граница съёмок считается найденной. Порог может быть глобальным, адаптивным или комбинированным. Глобальный порог подбирается экспериментально для всего видео целиком [2]. Локальные особенности при этом не учитываются, что отрицательно влияет на точность обнаружения. Алгоритмы адаптивного порога, основанные на скользящем окне, вычисляют порог локально [7, 13]. Скользящее окно запоминает предыдущие точки временного ряда, возвращает их вместе с текущей точкой в виде последовательности заданного размера. Для применения адаптивного порога необходимы априорные знания о самом видео, например, для выбора размера окна. Комбинация адаптивных и глобальных порогов позволяет подстраивать локальные пороги при наличии значения общих параметров. Например, в работе [11] введена пороговая функция. Значение функции изменяется

локально в пределах двух глобальных порогов. Глобальные пороги настраиваются в зависимости от требований точности и полноты. Однако отношения локальных адаптивных порогов и двух глобальных порогов не могут быть легко определены.

Статистические методы

Статистические методы находят события на основе характеристик кадров. Определение границ сцен может рассматриваться как задача классификации на основе статистического обучения. Кадры классифицируются как принадлежащие сцене и как ей не принадлежащие. Используются классификаторы как с учителем — контролируемые, так и без учителя – неконтролируемые. Преимущество методов классификации с учителем в том, что не нужно искусственно устанавливать пороги определения границ. Вместе с тем, для повышения точности могут быть использованы самые разные виды характеристик. Недостаток такой классификации — необходимость аккуратного выбора примеров для обучения, как положительных, так и отрицательных. К контролируемым классификаторам относят AdaBoost (Adaptive Boosting — адаптивное улучшение) и метод опорных векторов (MOB, SVM —support vector machine). Применяют и некоторые другие алгоритмы. В работе [3] применяют классификатор kNN на основе поиска ближайших соседей, в котором сходства кадров на определенном временном интервале используются как входные данные. В работе [1] различные виды перехода съёмок находят с помощью скрытых моделей Маркова (HMM). В работах [10, 12, 14] используется метод опорных векторов для того, чтобы обнаружить монтажные склейки. В работе Чжао

[9] для скользящего окна применяются два MOB-классификатора, чтобы отличить плавную смену съёмок от линейных склеек. В работе [5] MOB использовали для классификации кадров в трех категориях: постепенный переход, склейка и прочие кадры. В работах [4] и [8] MOB комбинируют с пороговыми методами. Предварительно, границы-кандидаты ищут с помощью порога. После этого применяют MOB-классификатор для более строгого отбора.

Поиск событий в видео

На низком уровне представления видеоряд содержит последовательность изображений и последовательность аудио-отчетов. Мы будем обсуждать только визуальный ряд. Видео может не содержать звука, но при этом, звука без изображения в видео не бывает.

События в видео

Из каждого кадра можно выделить большое количество признаков: цветовые моменты, цветовые коррелограммы, цветовые гистограммы, гистограммы направлений границ, вейвлет-текстуры [19]. Для поиска событий не важно, какие признаки использовать. Важен факт изменения этих признаков во времени. Более того, для анализа изменений совокупности признаков удобно свести всю совокупность к одной единственной характеристике. В исходном виде кадр представляет собой набор точек-пикселей. Каждый пиксель описывается цветом в заданном цветовом пространстве (RGB, YCrCb, Lab). Без ограничения общности, в качестве цветового пространства исходного видео будем рассматривать RGB.

Пусть F_t — кадр исходного видео в момент времени t.

$$F_t = \begin{pmatrix} x_{1,1}, & \cdots & x_{1,m} \\ \cdots & \cdots & \cdots \\ x_{n,1}, & \cdots & x_{n,m} \end{pmatrix}_t; \quad (1)$$

$$x_{i,j} = \{r_{i,j}; g_{i,j}; b_{i,j}\} \in \mathbb{R}^3_{[0,1]};$$

Где $r_{i,j}$, $g_{i,j}$ и $b_{i,j}$ — интенсивности цветов красного, зелёного и синего. Для каждого кадра вычислим значение *по выбранной* норме:

$$\|F_t\|_{L_1} = \frac{1}{n \cdot m} \sum_{i=0}^{n} \sum_{j=0}^{m} f(x_{i,j}); \quad (2)$$

$$f : \mathbb{R}^3_{[0,1]} \longmapsto \mathbb{R}_{[0,1]}$$

Каждая точка кадра может быть представлена яркостью — величиной Y в цветовом пространстве YUV [19]. С помощью формулы перехода из цветового пространства RGB в цветовое пространство YUV получим:

$$y_{i,j} = Y(x_{i,j}) = 0,299 \cdot r_{i,j} + 0,587 \cdot g_{i,j} + 0,114 \cdot b_{i,j} \quad (3)$$

Коэффициенты при цветовых составляющих задаются формулой перехода из RGB в YUV. Зададим функцию f как $f(x_{i,j}) = Y(x_{i,j})$. В общем случае, величина $||F_t||_{L1}$ характеризует состояние видео в момент времени t. Далее это состояние мы будем называть точкой видео, и обозначать P_t:

$$\mathrm{P}_t = \|F_t\|_{L1} \quad (4)$$

Временной ряд таких точек будем обозначать $P(t)$:

$$\mathrm{P(t)} = \{\|F_t\|_{L1}\}_0^n \quad (5)$$

Сами величины $P(t)$ не представляют большого интереса. Но изменение этих величин характеризует изменение состояния потока видео. А изменение видео потока указывает на события, происходящие в сюжете видео.

Поиск аномалий в видео

Изменения свойств точек видео указывают на события, происходящие в видео. Но так как мы имеем нестационарный нерегулярный временной ряд, изменения свойств точек видео происходят в каждый момент времени. Таким образом, необходимо отличать значимые изменения и изменения, которыми можно пренебречь. Значимые изменения свойств будем называть «аномалией» временного ряда.

Определение 4.1. Событие *(или аномалия)* — резкое изменение свойств наблюдаемого ряда в заранее неизвестный момент времени. Иногда такие аномалии называют «разладками».

Существует много методов для поиска аномалий во временных рядах. Большая часть методов применима для стационарных рядов случайных величин [18]. В методах, которые применимы для видеоряда, понятие аномалии обычно переопределяется согласно логике самого метода. Для оценки правильности методов поиска аномалий требуется ручная проверка на основе исходного видеоряда или временного ряда точек видео. Ниже описаны эксперименты по выделению событий из видео. В качестве исследуемых видео использовались два

класса видео: советские мультипликационные фильмы и видео, снятые камерами БПЛА. Все материалы взяты из открытых источников.

Пороговые методы со статическим порогом

Самый простой метод, который может быть применен для выделения аномалий — это сравнения разностей соседних величин с заранее заданным порогом.

$$\begin{cases} D_t > T_{const}; \\ D_t = P_t - P_{t-1}. \end{cases} \tag{6}$$

Превышение порога T_{const} будет считаться аномалией. Такие методы просты в реализации, в том числе, аппаратной, и не требовательны к ресурсам. Однако требуется заранее знать порог. Кроме того, эти методы не применимы для разных типов видео; чувствительны к случайным всплескам; ловят только краткосрочные события.

Рассмотрим наивный пороговый метод, в котором вычисляется значение нормированной попиксельной абсолютной разности яркостей двух соседних кадров. В качестве нормы используется векторная норма $L1$.

$$\begin{gathered} D_t = \|F_t\|_{L1} - \|F_{t-1}\|_{L1}; \\ D_t = \|F_t - F_{t-1}\|_{L1}; \end{gathered} \tag{7}$$

$$D_t = \frac{1}{n \cdot m} \sum_{i=0}^{n} \sum_{j=0}^{m} |y_{i,j,t} - y_{i,j,t-1}|, \text{ где} \tag{8}$$

- $y_{i,j,t} = Y(x_{i,j,t})$ — яркость пикселя кадра F_t в момент времени t;
- $y_{i,j,t-1} = Y(x_{i,j,t-1})$ — яркость пикселя кадра в м F_{t-1} омент времени $t-1$;

- $|y_{i,j,t-1} - y_{i,j,t}|$ — разность яркостей двух пикселей.

Если средняя разность пикселей превысит заранее заданное число — порог, мы считаем, что «разладка» найдена.

Язык потоковой обработки видео

Для решения задачи нами был разработан метаязык на базе языка Python. Основная сущность языка потоковой обработки видео — фильтр. Фильтр — это абстрактный тип на языке Python с перегруженными операторами. Фильтры описывают набор действий над временным рядом $P(t)$. Наивный пороговый метод может быть задан следующим образом:

```
delay = DelayFilter() # Фильтр линейной задержки.
orig = delay(0) # Входящий сигнал без изменения.
shift = ShiftSWFilter() # Сдвиг сигнала на один кадр.
diff = orig - shift # Разность соседних кадров.
norm = NormFilter() # Норма сигнала.
T_CONST = 0.08 # Значение порога. Константа.
threshold = orig > T_CONST # Пороговый фильтр.
d_filter = diff | abs | norm(l=1) # Фильтр нормированной абсолютной разности.
result_filter = d_filter | threshold # Результирующий фильтр: точки разладок.
```

Оператор «|» означает «конвейер». При любых операциях над фильтрами вычисления с кадрами видео не производится. В этот момент происходит только создание дерева операций, с помощью которого будет обработано видео. Полное

дерево операций выражено итоговым фильтром («*result_filter*»). Непосредственная обработка видео происходит в момент вызова метода *«filter_objects()»* итогового фильтра «*result_filter*».

Грамматика языка потоковой обработки видео

За основу языка потоковой обработки взят язык программирования Python. Любой фильтр может быть реализован на языке Python. Однако для простоты реализации использовано ограниченное подмножество языка. Ниже описана полная грамматика применяемого языка потоковой обработки видео. Полную грамматику языка Python можно найти в работе [16]. Используются следующие обозначения для лексем.

- *FILTER_NAME* — имена атомарных фильтров, которые нельзя реализовать в рамках предложенного языка.
- *POINT_FILTER_NAME* — имена атомарных фильтров, которые работают только над текущей точкой временного ряда.
- *WINDOW_FILTER_NAME* — имена атомарных фильтров, которые формируют скользящее окно.
- *arglist* — допустимый список аргументов функции в языке Python.
- *op* — допустимый бинарный оператор в языке Python.
- *expr* — допустимое выражение в языке Python.

Описание грамматики имеет вид:

FILTER ::= *FILTER_NAME '(' [arglist] ')' | FILTER '|' FILTER | FILTER '|' expr | FILTER op expr*

FILTER_NAME ::= *POINT_FILTER_NAME | WINDOW_FILTER_NAME*

POINT_FILTER_NAME ::= *'Filter' | 'NormFilter' | 'DelayFilter' | 'SignAngleDiff1DFilter' | 'SignAngleDiff2DFilter' | 'SignChangeFilter' | 'ColourFilter'*

WINDOW_FILTER_NAME ::= *'BaseSWFilter' | 'ShiftSWFilter'*

op ::= *'<' | '>' | '==' | '>=' | '<=' | '<>' | '!=' | '<<' | '>>' | '^' | '+' | '-' | '*' | '**' | '/' | '%' | 'in' | 'not' 'in'| 'is' |'is' 'not'*

expr ::= *python_expression*

Поиск аномалий с помощью FFmpeg

На практике для выделения аномалий в видео используют стандартный механизм, встроенный в FFmpeg. FFmpeg — набор программ и библиотек, которые позволяют записывать, обрабатывать и передавать цифровое видео в различных форматах. В том числе, FFmpeg содержит в себе возможность выделения «сцен». С порогом сравнивают величины D_t^{ffmpeg} — наименьшую величину из двух: нормированной разности яркостей двух соседних кадров и абсолютной разности двух последовательных разностей.

$$\begin{cases} D_t^{ffmpeg} > T_{const}; \\ D_t^{ffmpeg} = \min\left(D_t, |D_t - D_{t-1}|\right); \\ D_t = \frac{1}{n \cdot m} \sum_{i=0}^{n} \sum_{j=0}^{m} |y_{i,j,t} - y_{i,j,t-1}| \end{cases}, \text{ где} \quad (9)$$

- $y_{i,j,t} = Y(x_{i,j,t})$ — яркость пикселя кадра F_t в момент времени t;
- $y_{i,j,t-1} = Y(x_{i,j,t-1})$ — яркость пикселя кадра в м F_{t-1} омент времени $t - 1$;
- T_{const} — порог, превышение которого будет считаться аномалией.

Минимум $\min(D_t, |D_t - D_{t-1}|)$ используется для того, чтобы избежать всплесков, если график сигнала монотонно убывает или возрастает. Разность двух последовательных разностей имеет смысл скорости изменения яркости. Малая скорость изменения при значительном изменении говорит о том, что яркость равномерно меняется в некоторой окрестности данного кадра. При быстром, но плавном изменении яркостей кадров с точки зрения авторов порога из FFmpeg — разладки нет.

Недостатки статического порога

Несмотря на свою простоту и удобство, статический порог обладает рядом существенных недостатков: величину порога надо знать заранее; нельзя использовать везде одну и ту же величину порога. Проведенные эксперименты показывают, что величина порога, которая хорошо определяет аномалии на видео с резкой сменой событий, не подходит для плавного видео. В качестве плавного видео мы рассматривали видео бортовых камер БПЛА. В качестве динамичных

видео — художественные фильмы. Таким образом, при поиске событий в видео нужно учитывать «динамичность» самого видео. Для этого потребуется или масштабировать разности кадров по окрестности, или учитывать среднюю величину и дисперсию.

Нормировка по размаху разности кадров

Нормировка по размаху разности кадров подразумевает применение следующего алгоритма:

1. Выбрать размер скользящего окна — *s;*
2. Для каждого кадра из последовательности:
    a. вычислить максимальное $D_{\min,\, s,\, t} = \min(D_{t-s}, \ldots, D_t)$ и минимальное $D_{\max,\, s,\, t} = \max(D_{t-s}, \ldots, D_t)$ значение разности кадров на этом окне;
    b. вычислить размах разностей $D_{\max,\, s,\, t} - D_{\min,\, s,\, t}$ для данного скользящего окна;
    c. от текущего значения разностей соседних кадров вычесть минимальное значение разностей на скользящем окне, и поделить на размах

$$D_{s,t} = \frac{D_t - D_{\min,s,t}}{D_{\max,s,t} - D_{\min,s,t}}.$$

Пороговый метод поиска событий в видео с нормировкой по размаху разностей кадров выражается системой:

$$\begin{cases} D_{s,t} > T_{const}; \\ D_{s,t} = \dfrac{D_t - D_{\min,s,t}}{D_{\max,s,t} - D_{\min,s,t}}; \\ D_{\min,s,t} = \min(D_{t-s}, \ldots, D_t); \\ D_{\max,s,t} = \max(D_{t-s}, \ldots, D_t); \\ D_t = \dfrac{1}{n \cdot m} \sum_{i=0}^{n} \sum_{j=0}^{m} |y_{i,j,t} - y_{i,j,t-1}| \end{cases}$$, где

- $y_{i,j,t} = Y(x_{i,j,t})$ — яркость пикселя кадра F_t в момент времени t;
- $y_{i,j,t-1} = Y(x_{i,j,t-1})$ — яркость пикселя кадра в м F_{t-1} омент времени $t - 1$;
- T_{const} — порог, превышение которого будет считаться аномалией;
- s — выбранный размер скользящего окна.

Описание алгоритма на метаязыке пороговой обработки будет иметь вид:

S=400 # Параметр алгоритма. Размер скользящего окна.

delay = ***DelayFilter****(); orig = delay(0); shift =* ***ShiftSWFilter****()*

diff = orig - shift # Разность соседних кадров.

norm = ***NormFilter****() # Норма сигнала.*

sw = ***BaseSWFilter****(s=S) # Скользящее окно размером S для каждого кадра.*

sw_max = sw | max # Максимум по скользящему окну.

sw_min = sw | min # Минимум по скользящему окну.

sw_norm = (orig - sw_min) / (sw_max - sw_min) # Масштабирование.

d_sw_filter = diff | abs | norm(l=1) | sw_norm # Масштабированная разность.

***result_filter** = d_sw_filter | threshold # Результирующий фильтр..*

Достоинство нормировки по размаху — относительная величина порога: не нужно подбирать для каждого случая; можно использовать везде одну и ту же величину. Однако размер скользящего окна приходится подбирать: при малом размере окна — происходит нормировка для паразитных шумов, при большом размере окна — пропускаются некоторые аномалии.

Пороговые методы с адаптивным порогом

Динамичность видео можно учесть с использованием статистических величин выборочного среднего и выборочного среднего квадратичного отклонения. Адаптивные пороговые методы подробно рассмотрены в работах [7, 11, 13].

Опишем метод, который использует критерий Смирного-Граббса [18]. На основе выборочного среднего и выборочного среднего квадратичного отклонения строится модель сигнала, после чего сигнал сравнивается со своей моделью.

$$\begin{cases} D_t & > & E_{s,t}; \\ E_{s,t} & = & \overline{D}_{s,t} + \sigma \cdot S_{s,t}; \\ S_{s,t} & = & \sqrt{\frac{1}{s-1} \sum\limits_{k=t-s}^{t} \left(D_k - \overline{D}_{s,t}\right)}; \\ \overline{D}_{s,t} & = & \frac{1}{s} \sum\limits_{k=t-s}^{t} D_k; \\ D_t & = & \frac{1}{n \cdot m} \sum\limits_{i=0}^{n} \sum\limits_{j=0}^{m} |y_{i,j,t} - y_{i,j,t-1}| \end{cases}$$, где (12)

- $y_{i,j,t} = Y(x_{i,j,t})$ — яркость пикселя кадра F_t в момент времени t;
- $y_{i,j,t-1} = Y(x_{i,j,t-1})$ — яркость пикселя кадра в м F_{t-1} омент времени $t - 1$;
- s — выбранный размер скользящего окна;
- σ — коэффициент возможного отклонения модели видеосигнала.

Описание алгоритма на метаязыке пороговой обработки будет иметь вид:

SIGMA = 3.0; # Параметр модели сигнала.

S = 100 # Размер скользящего окна.

delay = ***DelayFilter****(); orig = delay(0); shift =* ***ShiftSWFilter****()*

diff = orig - shift # Разность соседних кадров.

norm = ***NormFilter****() # Норма сигнала.*

d_filter = diff | abs | norm(l=1)

sw = ***BaseSWFilter****(s=S, min_size=2)*

sw_mean = sw | numeric.mean # Скользящее среднее

sw_std = sw | numeric.std # Скользящее среднеквадратическое отклонение.

*estimation = sw_mean + SIGMA * sw_std*

sigma_threshold = (orig > sigma_estimation) | int

***result_filter** = d_filter | sigma_threshold # Результирующий фильтр.*

К недостаткам подхода стоит отнести необходимость подбора размера скользящего окна. Метод весьма чувствителен к случайным всплескам и не применим для разных типов видео.

Сравнение скользящих средних

Применим методы сравнения скользящих средних, известные из экономического анализа. В экономическом анализе скользящие средние оценки временных рядов используют для определения момента начала или окончания биржевых торгов. При этом в экономическом анализе рекомендуют использовать экспоненциальное скользящее среднее, т.к. оно позволяет учитывать каждую точку ровно один раз. Это позволяет избежать ситуации, когда одна и та же точка влияет на входе в скользящее окно и на выходе из него. Но в задаче поиска событий видео оказалось, что удобнее использовать обычное скользящее среднее, т.к. важны как факт входа в скользящее окно, так и выхода из него.

Сигнал оценивается через свои скользящие средние. Считаем событием точку, в которой скользящие средние стали равны.

Метод может быть описан формулой:

$$\begin{cases} 0 > |\overline{Y}_{A,t} - \overline{Y}_{B,t}|; \\ \overline{Y}_{s,t} = \frac{1}{s}\sum_{k=t-s}^{t} Y_k; \\ Y_t = \frac{1}{n \cdot m}\sum_{i=0}^{n}\sum_{j=0}^{m} |y_{i,j,t}| \end{cases} \text{, где} \quad (13)$$

- $y_{i,j,t} = Y(x_{i,j,t})$ — яркость пикселя кадра F_t в момент времени t;
- $y_{i,j,t-1} = Y(x_{i,j,t-1})$ — яркость пикселя кадра в м F_{t-1} омент времени $t - 1$;
- A, B — выбранные размеры скользящих окон.

В ходе исследования авторы столкнулись с тем, что на реалистичных тестовых данных скользящие средние не будут равны, но при этом в точках разладок будут максимально близки друг к другу. При этом разность скользящих средних поменяет свой знак. Сформулируем алгоритм поиска аномалий с помощью сравнения скользящих средних:

1. Выбрать размеры скользящих окон — *A и B* соответственно*;*
2. Для каждого кадра из последовательности:
    a. вычислить среднюю яркость кадра по формулам (1-5);
    b. вычислить скользящее среднее для окна размером *A:*

$$\overline{Y}_{A,t} = \frac{1}{A}\sum_{k=t-A}^{t} Y_k; \quad (14)$$

    c. вычислить скользящее среднее для окна размером *B:*

$$\overline{Y}_{B,t} = \frac{1}{B} \sum_{k=t-B}^{t} Y_k; \quad (15)$$

a. вычислить разность $\overline{Y}_{A,t} - \overline{Y}_{B,t}$ и запомнить ее знак;

a. если знак разности отличается от знака на предыдущем кадре — сигнализировать об аномалии; иначе перейти к следующему кадру.

Описание алгоритма на метаязыке пороговой обработки будет иметь вид:

A = 100; B = 200 # Параметры алгоритма: размеры скользяших окон.

delay = ***DelayFilter****(); orig = delay(0); norm =* ***NormFilter****()*

sign_change = ***SignChangeFilter****() # Фильтр смены знака.*

sw = ***BaseSWFilter****(min_size=2) # Скользящее окно для каждого кадра.*

sw_mean = sw | numeric.mean # Скользящее среднее.

 # Разность скользящих средних.

sw_mean_diff = (sw_mean(s=A) – sw_mean(s=B))

result_filter *= norm(l=1) | sw_mean_diff | sign_change | abs*

В отличие от пороговых методов, предложенный алгоритм обладает лучшей устойчивостью и к резким и к плавным переходам между элементами сюжета.

По сравнению с ранее рассмотренными пороговым методы с адаптивным порогом и нормировкой по размаху метод очень нетребователен к ресурсам и в отличие от статистических методов из работ [1, 3, 4, 8, 9, 10, 12, 14] не требует процесса обучения. Предложенный метод обладает недостатками, которые присущи всем методам со скользящими окнами: при малом размере ловят шум; при большом —

могут пропустить событие. Однако, при размере окон 50 и 100 элементов количество ложных срабатываний меньше, чем у адаптивных пороговых методов.

Теорема о дубликатах

Видео — последовательность фактов или событий, развивающихся во времени. Свойства событий — пространственная характеристика видео [17], продолжительность и порядок фактов — временная.

Пусть даны два видео — V и W. Рассмотрим эти видео как последовательности кадров и по формулам (1-5) построим для них временные ряды V и W. Продолжительность каждого видео равна T_v и T_w . В качестве отсчетов для этих рядов выступают точки P_t в момент времени t. Единицы измерения времени для каждого видео могут не совпадать. В каждый момент времени каждый видеоряд обладает набором характеристик $v_t = V(t_v)$ и $w_t = W(t_w)$.

Из точек временного ряда $V(t)$ в которых наблюдались аномалии, постоим временной ряд $(ev_i)^k_{i=1}$

$$\begin{cases} (ev_i)_{i=1}^{k} & = & ev_1, ev_2, \ldots, ev_k; \\ \{ev_1, ev_2, \ldots, ev_k\} & \subset & V \end{cases} \tag{16}$$

Для $W(t)$ аналогичным образом построим $(ew_j)^n_{j=1}$.

Определение 1. Будем считать, что V и W выражают одинаковую последовательность событий, если можно выделить последовательность $(x_i)^m_{i=1}$, такую что $\{(x_i)^m_{i=1}\} \subset \{(ev_i)^k_{i=1}\}$ и $\{(x_i)^m_{i=1}\} \subset \{(ew_j)^n_{j=1}\}$.

$$V \overset{e}{\sim} W \Leftrightarrow \exists (x_i)_{i=1}^{m};\ \begin{cases} \{(x_i)_{i=1}^{m}\} & \subset & \{(ev_i)_{i=1}^{k}\} & \subset & V; \\ \{(x_i)_{i=1}^{m}\} & \subset & \{(ew_i)_{i=1}^{n}\} & \subset & W. \end{cases} \tag{17}$$

Определение 2. Считаем, что V и W — нечеткие дубликаты друг друга, если можно выделить последовательность $(x_i)^m_{i=1}$, такую что $\{(x_i)^m_{i=1}\} \subset \{(v_i)^k_{i=1}\}$ и $\{(x_i)^m_{i=1}\} \subset \{(w_j)^n_{j=1}\}$.

$$V \overset{nd}{\approx} W \Leftrightarrow \exists (x_i)_{i=1}^m; \left\{ \begin{array}{ccccc} \{(x_i)_{i=1}^m\} & \subset & \{(v_i)_{i=1}^k\} & \subset & V; \\ \{(x_i)_{i=1}^m\} & \subset & \{(w_i)_{i=1}^n\} & \subset & W \end{array} \right. \quad (18)$$

Теорема (О дубликатах). Нечеткие дубликаты видео выражают одинаковую последовательность событий.

$$V \overset{e}{\sim} W \Longrightarrow V \overset{nd}{\approx} W \quad (19)$$

Выводы

В статье рассмотрены различные алгоритмы поиска событий в видеопоследовательностях и предложен способ поиска на основе сравнения скользящих средних. По классификации, приведенной в начале статьи, метод относится к статистическим методам, но отличается от своих аналогов вычислительной простой скоростью работы. При использовании предложенного в статье метаязыка, метод сегментации видео на основе сравнения скользящих средних может быть легко сформулирован в виде декларативного описания на интерпретируемом языке программирования. К сожалению, большинство существующих методов компьютерной обработки видео используют исключительно императивный подход к описанию алгоритмов, что часто затрудняет их восприятие человеком. Как показали приведенные эксперименты, реализация предложенного метода поиска событий в видео, в отличие от своих аналогов, способна работать и для потокового видео, в условиях реального времени и потенциально бесконечной

последовательности кадров. Такие преимущества в совокупности с низкими вычислительными требованиями к ресурсам делают реализацию метода востребованной в авиационной и космической технике. К недостаткам предложенного алгоритма стоит отнести необходимость подбора параметров алгоритма для конкретного класса задач.

Библиографический список

1. Boreczky J.S., Wilcox L.D. A hidden Markov model framework for video segmentation using audio and image features // Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (Seattle, WA), 12 May 1998, vol. 6, pp. 3741–3744.

2. Cernekova Z., Pitas I., Nikou C. Information theory-based shot cut/fade detection and video summarization // IEEE Transactions on Circuits and Systems for Video Technology, 2006. Vol. 16, no. 1, pp. 82–91.

3. Cooper Matthew, Liu Ting, Rieffel Eleanor G. Video Segmentation via Temporal Pattern Classification // IEEE Transactions on Multimedia. 2007. Vol. 9, no. 3, pp. 610–618.

4. Jinhui Yuan, Huiyi Wang, Lan Xiao et al. A Formal Study of Shot Boundary Detection // IEEE Transactions on Circuits and Systems for Video Technology, 2007. Vol. 17, no. 2, pp. 168–186.

5. Xue Ling, Li Chao, Xiong Zhang, Li Huan. A General Method for Shot Boundary Detection // Multimedia and Ubiquitous Engineering, International Conference on, 2008,

pp. 394-397, doi:10.1109/MUE.2008.102, URL: http://dblp.uni-trier.de/db/conf/mue/mue2008.html.

6. Hanjalic Alan. Shot-boundary detection: unraveled and resolved? // IEEE Transactions on Circuits and Systems for Video Technology. 2002. Vol. 12, no. 2, pp. 90–105.

7. Hoi Steven C.H., Wong Lawson, Lyu Albert. Chinese University of Hong Kong at TRECVID 2006: Shot Boundary Detection and Video Search // Int. TREC Video Retrieval workshop (TRECVID'06). 2006. URL: http://people.csail.mit.edu/lsw/papers/trecvid06-shotbdry.pdf

8. Liu Chunxi, Liu Huiying, Jiang Shuqiang et al // JDL at Trecvid 2006 Shot Boundary Detection. 2006. URL: https://scholar.google.com.sg/citations?user=ivHE9dkAAAAJ&hl=en

9. Wan-Lei Zhao, Chong-Wah Ngo, Hung-Khoon Tan, Xiao Wu. Near-Duplicate Keyframe Identification With Interest Point Matching and Pattern Learning // IEEE Transactions on Multimedia. 2007. Vol. 9, no. 5, pp. 1037–1048.

10. Ngo Chong-Wah. A Robust Dissolve Detector by Support Vector Machine // Proceedings of the Eleventh ACM International Conference on Multimedia. MULTIMEDIA '03. New York, USA, 2003, pp. 283–286.

11. Quénot Georges, Moraru Daniel, Besacier Laurent. CLIPS at TRECvid: Shot Boundary Detection and Feature Detection // TRECVID 2003 Workshop Notebook Papers. Gaithersburg, MD, USA, 2003, pp. 18–21

12. Kazunori Matsumoto, Masaki Naito, Keiichiro Hoashi, Fumiaki Sugaya. SVM-Based

Shot Boundary Detection with a Novel Feature // IEEE International Conference on Multimedia and Expo. 2006, Toronto, ON, Canada, pp. 1837 - 1840.

13. Xiaomeng Wu, Masao Takimoto, Shin'ichi Satoh, Jun Adachi. Scene Duplicate Detection Based on the Pattern of Discontinuities in Feature Point Trajectories // Proceedings of the 16th ACM International Conference on Multimedia (MM), New York, NY, USA, ACM, 2008, pp. 51–60.

14. G Camara-Chavez, F Precioso, M Cord, S Phillip-Foliguet, A de A Araujo. Shot boundary detection by a hierarchical supervised approach // Systems, Signals and Image Processing, 2007 and 6th EURASIP Conference focused on Speech and Image Processing, Multimedia Communications and Services. 14th International Workshop on, June 2007, pp. 197–200.

15. Prem Kumar Kalra, Shmuel Peleg. Computer Vision, Graphics and Image Processing. // 5th Indian Conference, ICVGIP 2006, Madurai, India, December 13-16, 2006, Proceedings. Lecture Notes in Computer Science 4338, Springer 2006, URL: http://dblp.uni-trier.de/db/conf/icvgip/ icvgip2006.html. 3

16. van Rossum Guido, Drake Fred L. The Python Language Reference Manual. Bristol, United Kingdom, Network Theory Ltd., 2003. 144 p.

17. Гусев В.Ю. Крапивенко А.В. Методика фильтрации периодических помех цифровых изображений // Труды МАИ. 2012. № 50. URL: http://trudymai.ru/published.php?ID=28805

18. Марчук В.И. Токарева С.В. Способ обнаружения аномальных значений при

анализе нестационарных случайных сигналов. – Шахты, Ростовская обл.: ЮРГУЭС, 2009. – 209 с.

19. Прутов И.С. Синча Д.П. Исследование методов и разработка алгоритмов обработки видеоинформации в задачах локализации положения беспилотного летательного аппарата на основе распознавания изображений при помехах и искажениях // Труды МАИ. 2012. № 52. URL: http://trudymai.ru/published.php?ID=29441